文章编号:1671-9352(2016)09-0059-09 DOI:10.6040/j.issn.1671-9352.1.2015.293

FlowMonitor:Android 隐私数据流向监控防护系统

查明明,王伟*
(北京交通大学计算机与信息技术学院信息安全系,北京 100044)

摘要:为了保护存储在 Android 平台中的隐私数据,提出了一个基于 Android 的隐私数据流向监控防护系统 FlowMonitor。FlowMonitor 系统为数据添加污点标记和身份标签,进行实时、动态监控;并利用基于 RBAC 的污点身份认证机制,防止共谋攻击造成的隐私泄漏,找出参与共谋攻击的恶意应用;引入基于 Open and Closed Policies 的细粒度控制隐私数据的使用方法,控制粒度细化到数据本身,用户通过自主选择来控制应用程序如何使用隐私数据。测试结果表明,相比于 TaintDroid 系统,FlowMonitor 能够细粒度地控制应用程序使用隐私数据,找到参与共谋攻击的恶意应用,从而有效地保护用户的隐私数据安全。

关键词:Android;隐私保护;身份认证;共谋攻击
中图分类号:TP309 **文献标志码**:A
引用格式:查明明,王伟. FlowMonitor: Android 隐私数据流向监控防护系统[J]. 山东大学学报(理学版),2016,51(9):59-67.

A system of monitoring and protecting Android privacy leakage

ZHA Ming-ming, WANG Wei*
(College of Computer and Information Technology, Beijing Jiaotong University, Beijing 100044, China)

Abstract: In order to protect the data of Android operating system, we present a system called FlowMonitor to monitor and protect privacy flow based on Android. FlowMonitor adds taint marks and identity tags onto data. Moreover, it is a real-time, dynamic monitoring system. In order to avoid leaking private data by collusion attack, it uses the mechanism of taint's identity authentication based on the idea of RBAC, and finds all probable malicious applications. Based on open and closed policies, labels were designed for each data to control the use of private data. The user can control applications using private data based on their choices. Evaluation results show that Flowmonitor out performs taint Droid. It can not only find out all probable malicious applications which steal private data by collusion attack, but also control applications using each private data. Extensive experimental results demonstrate that FlowMonitor effectively protects the user privacy.

Key words: Android; privacy protection; identity authentication; collusion attack

0 引言

随着移动互联网的迅速发展,智能终端在人们生活中已经成为了不可或缺的角色。美国咨询公司高德纳(Gartner)的数据显示:Android 操作系统预计将在 2015 年全面地压倒所有操作系统,成为业界的龙头老大[1],智能手机的用户数远远超过传统 PC 的用户数。存储在 Android 手机中的大量商业机密、个人隐私等隐私信息的保护变得至关重要。鉴于Android 操作系统的平台开放性和第三方应用市场管理混乱等问题,Android 智能手机已经成为众多病毒、木马和恶意软件攻击的重点目标。据网秦公司发布的《2014 年第三季度全球手机安全报

收稿日期:2015-06-07; **网络出版时间**:2016-09-07 16:22:50
网络出版地址:http://www.cnki.net/kcms/detail/37.1389.N.20160907.1622.012.html
基金项目:中央高校基本科研业务费专项资金资助项目(2015JBM025);教育部高等学校博士学科点专项科研项目(20120009120010);教育部留学回国人员科研启动基金资助项目(K14C300020)
*通讯作者:王伟(1976—),男,博士,副教授,研究方向为计算机系统安全与网络安全、大数据隐私保护等. E-mail:wangwei1@bjtu.edu.cn

告》[2]显示,2014 年第三季度查杀到手机恶意软件共计 50591 款,同比增长 18.2%;感染智能手机共计 2 188 万部,同比增长 21.4%。其中,隐私窃取类的恶意软件以 15.09% 的比例位居前三,且隐私窃取类病毒呈上升趋势,移动社交、移动支付等成为手机重点安全问题。Android 手机的隐私数据泄漏问题日益突出,因此,如何保护 Android 手机的隐私数据、防止隐私泄漏以及保护用户财产信息的安全已经成为了一个亟待解决的问题。

Android 体系结构分为 4 层:应用程序层、应用程序框架层、系统运行时库层和 Linux 内核层。Android 系统为保护数据安全提供了 3 种安全机制[3]:进程沙箱隔离机制、应用程序签名机制、权限声明机制。进程沙箱隔离机制与应用程序签名机制是指不同签名的应用程序在安装时,被授予独特的用户标识(UID),并运行在不同进程中,同一签名的应用程序共享 UID,被放入同一沙箱内。权限声明机制是指在应用程序框架层,系统实现了一个基于能力的访问控制框架。应用程序根据自我功能需求访问系统资源时,需要申请对应的权限,但是,由于权限机制的静态性[4],无法防止共谋攻击[5]。共谋攻击如图 1 所示,A、B 两个应用以相应权限各自运行在沙箱内,应用 A 的组件 A_1 虽然不具备 P1 权限,但可利用应用 B 的组件 B_1 漏洞间接获取 P1 权限,从而获取相应的敏感数据。

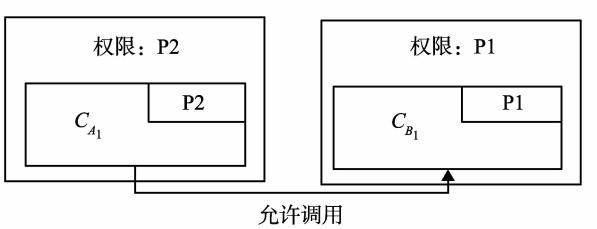


图 1 共谋攻击示例

Fig. 1 Example of collusion attack

目前,针对 Android 平台的隐私保护工作主要有 XManDroid[6]、SEAndroid[7]、TaintDroid[8] 等。XManDroid 将整体系统用图来表示,其中,图的节点表示应用程序,无向边表示组件之间通信。由图中的路径算法来决定是否允许新组件通信。并引入顶点的颜色属性,实时地记录与更新各顶点之间的通信历史记录,从而有效地防止多应用的共谋攻击。然而,由于权限机制的控制粒度仍过于粗糙,XManDroid 不能细到数据本身进行保护,无法避免恶意应用通过隐蔽通道获取隐私数据。其次,存在很多的敏感数据如敏感文件不能被权限机制所保护。SEAndroid 将原本运用在 Linux 操作系统上的 MAC 强制存取控制套件 SELinux,移植到 Android 平台上,强化了 Android 操作系统对 App 的文件访问控制。但 SELinux 策略针对性不强,控制粒度仅在进程层面上,而无法细化到进程所使用的数据上。TaintDroid 在 Android 平台上实现了一个实时动态的隐私数据污点跟踪系统,但它没有提供必要的隐私数据使用控制机制,无法防止隐私泄漏,同时无法找出参与共谋攻击的应用程序。

为解决 Android 平台的隐私泄漏问题,在借鉴 XManDroid、TaintDroid 等系统的思想基础上,本文提出了一个基于 Android 的隐私数据流向监控防护系统解决方案,并将其命名为 FlowMonitor。FlowMonitor 借助“Open and Closed Policies[9]”和“RBAC”的思想,结合数据流监控、Hook API、国密 SM4 算法[10]加解密、用户自主选择控制、污点身份认证等技术,实现隐私数据流向监控、捕获敏感的数据流、保证 Log 和 OCP 文件的机密性、细粒度地控制隐私数据的使用、防止共谋攻击等功能。经过测试,本系统能够实现实时、动态地监控应用程序对隐私数据的使用,可以了解应用程序将隐私数据发送到何处,并且能够对隐私泄漏进行拦截分析,细粒度地控制隐私数据的使用,还可以避免恶意应用通过隐蔽通道获取隐私数据,能够有效地防护恶意应用的共谋攻击,找到参与共谋攻击的恶意应用,保护隐私数据的安全。

1 系统设计

1.1 设计原理

FlowMonitor 系统的核心原理是将所有隐私数据视为污点,与 TaintDroid 不同的是,污点标签是由污点标记和污点身份信息组成,系统在为污点添加污点标记的同时,还为污点添加了污点首获者的身份信息,并且,依据策略来决定是否使用伪造的隐私数据,解决了 TaintDroid 无法防止隐私泄露的问题。在程序运行的过程中,当有应用程序对污点进行截取、拼装、加密、赋值、传递等操作时,污点标签随之传播,能够有效地解决 XManDroid 等系统无法避免恶意应用通过隐蔽通道获取隐私数据的弊端。如果带有污点标签的数据到达系统发送处,那么该数据会在系统发送处被 hook 函数所捕获,借助 XManDroid 颜色标签的思想,首先,FlowMonitor 系统将会核实污点身份,进行分析采取相应措施,并及时告知上层 FlowMonitor 应用程序发生了隐私泄露,能够解决 TaintDroid 无法找到参与共谋攻击的恶意应用

的问题。然后,上层 FlowMonitor 应用程序将泄露的信息汇总,并存入数据库中。通过该应用程序,用户查看第三方应用对隐私数据的使用情况。最后,根据各类隐私泄漏的统计信息、隐私泄漏详情和隐私数据的解释,用户可以合理地控制隐私数据的使用,控制粒度细化到数据本身,改进了权限机制控制粒度过粗的不足,防止隐私泄漏,同时,解决了目前主流的研究成果存在的界面不友好、对用户专业素养要求高、实用性差等问题,让普通用户也能加入保护隐私的大队伍中。

1.2　系统架构

基于 Android 操作系统的基本架构,分别在应用程序层、应用程序框架层和系统运行时库层修改或添加相应代码,从而实现了 FlowMonitor 系统。整个 FlowMonitor 体系结构模型如图 2 所示。

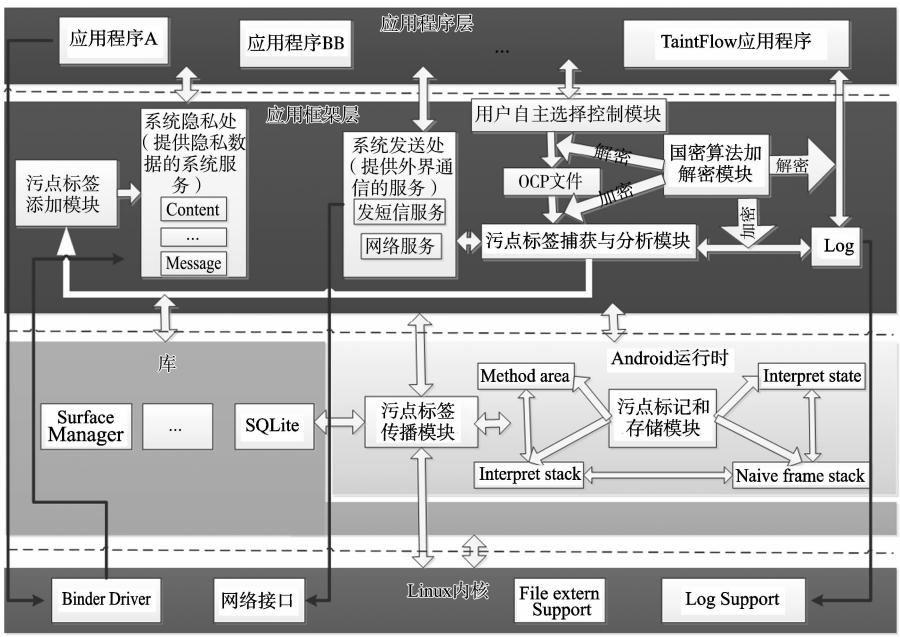


图 2　FlowMonitor 系统的体系结构模型
Fig. 2　System architecture of FlowMonitor

1.3　工作流程

整个 FlowMonitor 系统的总体工作流程可分为两部分:获取隐私数据阶段和泄漏隐私数据阶段。下面将分别具体说明。

(1) 获取隐私数据阶段(图 3)

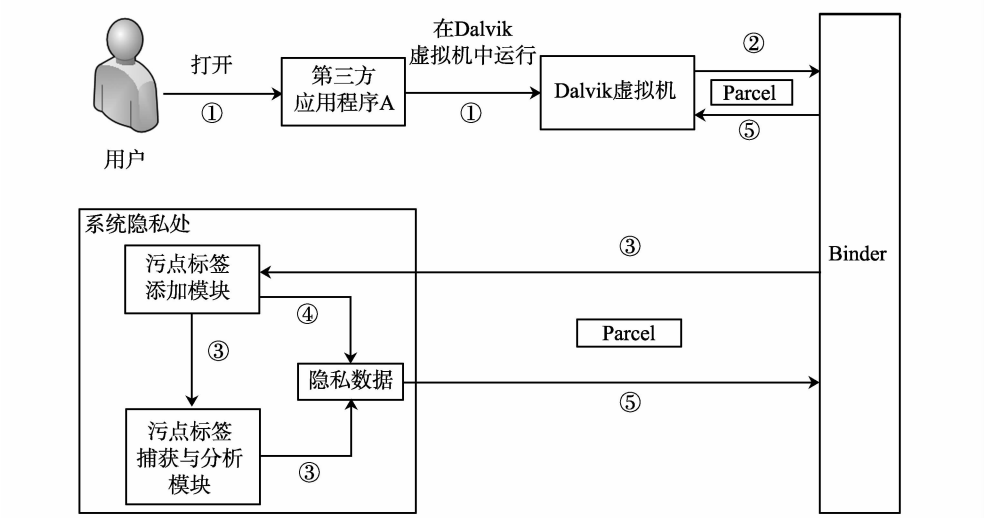


图 3　第三方应用程序 A 获取隐私数据的工作流程图
Fig. 3　Working flow for the third party application A to obtain privacy data

1) 用户打开第三方应用程序 A,第三方应用程序 A 在 Dalvik 虚拟机中运行。运行时,应用程序 A 中的所有变量的污点标签为 TAINT _ CLEAR,表示没有污点;

2) 应用程序A通过Binder来调用敏感API,如读取短信;

3) 在系统隐私处,污点标签添加模块获知应用程序A的请求,调用污点标签捕获与分析模块,依据策略判定是否使用伪造的隐私信息;

4) 最后,污点标签添加模块对A获得短信内容添加污点标记和污点身份标签,分别为短信类型和应用程序A;

5) 通过IPC进程通信,污点标签添加到通信数据结构,将带有污点的短信内容由系统隐私处传播到进程A中。

(2) 泄漏隐私数据阶段(图4)

1) 当带有污点短信内容在Dalvik虚拟机中进行截取、拼装、加密、传递等操作时,污点标签随之传播;

2) 当调用本地方法时,借助JNI方法,将污点传入并返回。当进行读写文件时,污点标签添加到文件的附加属性中;

3) 通过IPC进程通信,污点标签添加到通信数据结构,污点标签由进程A传播到进程B中;

4) 当带有污点的数据到达系统发送处,即网络接口或短信服务。系统调用污点标签捕获与分析模块。

5) 污点标签捕获与分析模块验证污点身份是否与发送者的身份相同。如果不同,则将污点身份和发送者的身份记录为共谋攻击的嫌疑对象,同时将此次隐私泄漏事件记录在系统Log文件中;

6) 国密SM4加解密模块对系统Log文件和OCP文件进行加解密保护;

7) 用户通过FlowMonitor应用程序,查看应用程序泄漏隐私的情况,然后通过自主选择,生成OCP策略文件,从而能够细粒度地控制应用程序使用隐私数据,防止隐私泄漏。

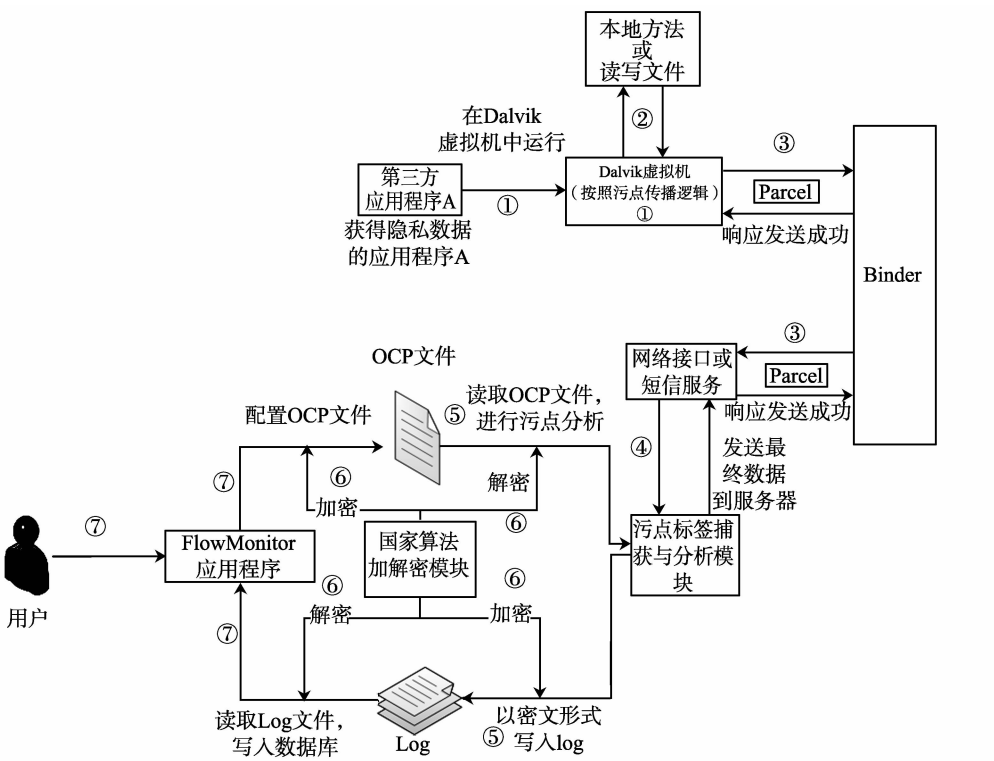


图4 第三方应用程序A泄漏隐私数据被拦截的工作流程图

Fig. 4 Working flow for blocking the privacy data leaked to the third party application A

2 系统实现

2.1 污点标记和存储模块

通过分析Android系统源码,FlowMonitor系统从中抽象出17种污点类型,如表1所示。

污点会存储在变量、文件、解释器状态和IPC通信数据中。对于变量,FlowMonitor系统使用32位存储空间来存储污点,如图5所示;其次,对于文件,本系统修改了文件的数据结构,将污点标记存储在文件的附加属性中;最后,对于解释器状态和IPC通信数据,本系统将污点标记存储在相应的数据结构中。此外,本系统将污点存储空间中剩余的16(32-16)位用于污点身份的存储。因此,一个32位的污点标签是由污点标记和污点身份组成的。

表 1　FlowMonitor 系统污点的详细编码
Table 1　Coding of taint for FlowMonitor

序号	污点类型	污点编码
1	TAINT_CLEAR(无污点)	0x00000000
2	TAINT_LOCATION(地理位置信息)	0x00000001
3	TAINT_CONTACTS(通讯录)	0x00000002
4	TAINT_MIC(麦克风输入信息)	0x00000004
5	TAINT_PHONE_NUMBER(手机号码)	0x00000008
6	TAINT_LOCATION_GPS(GPS 地理位置信息)	0x00000010
7	TAINT_LOCATION_NET(网络地址信息)	0x00000020
8	TAINT_LOCATION_LAST(最近一次地理位置信息)	0x00000040
9	TAINT_CAMERA(照相机输入信息)	0x00000080
10	TAINT_ACCELEROMETER(传感器信息)	0x00000100
11	TAINT_SMS(短信)	0x00000200
12	TAINT_IMEI(IMEI)	0x00000400
13	TAINT_IMSI(IMSI)	0x00000800
14	TAINT_ICCID(ICCID:SIM 唯一标识)	0x00001000
15	TAINT_DEVICE_SN(设备序列号)	0x00002000
16	TAINT_ACCOUNT(用户账户信息)	0x00004000
17	TAINT_HISTORY(浏览器历史记录信息)	0x00008000

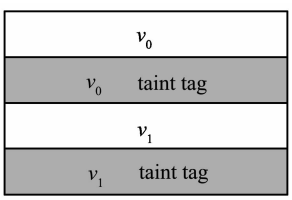


图 5　污点存储空间结构
Fig. 5　Storage structure of taint

2.2　污点标签捕获与分析模块

FlowMonitor 系统的污点标签捕获与分析模块是由污点标签捕获模块与污点标签分析模块组成。

(1)污点捕获模块。当污点传播到系统发送处(Android 系统的网络接口与短信发送服务)时,污点捕获模块将污点拦截下来,然后调用污点分析模块。

(2)污点分析模块。首先,污点分析模块核对污点的身份,进行身份认证,检测是否参与了共谋攻击,判断发送者的身份是否与污点身份相同。如果不是,那么系统将使用伪造的隐私信息来替换真实的隐私信息,然后,将污点身份和发送者的身份记录为参与共谋攻击的嫌疑对象,将此次隐私泄漏事件记录在系统 Log 文件中,同时,将发送到远端服务器的信息替换为虚假信息,防止共谋攻击;反之,则以泄漏隐私嫌疑记录到系统 Log 文件中,这样就完成了对污点标签的捕获与分析。

2.3　国密算法加解密模块

国密算法加解密模块采用了国密 SM4 算法。该算法使用相同的密钥进行加解密,国密 SM4 算法的加解密速度较快,比一般的流密码的安全性高,且实现较为方便。利用国密 SM4 算法对系统 Log 文件和 OCP 文件进行加解密保护,可以保证加密的高效和安全性。对于系统 Log 文件,FlowMonitor 系统使用国密 SM4 加密算法对污点的拦截信息进行加密,然后,将密文写入系统 Log 文件中;对于 OCP 文件,本系统将用户修改的配置信息也用密文形式写入 OCP 文件。当上层 FlowMonitor 应用程序读取系统 Log 文件和 OCP 文件时,本系统再利用相同密钥进行解密。这样就能够有效地防止恶意应用对系统 Log 文件和 OCP 文件的窃取与篡改,保证了数据的机密性。

2.4　用户自主选择控制隐私数据使用模块

当有加密后的污点拦截信息被写入 Log 中时,FlowMonitor 系统就会通知上层 FlowMonitor 应用程序从 Log 中读取污点拦截信息的密文。然后,该应用程序调用节 2.3 介绍的国密算法加解密模块的解密函数,对密文进行解密,获得污点拦截信息的明文,然后将数据存在数据库中。为了让用户的选择更加合理,FlowMonitor 系统为用户提高两种不同角度来查看隐私数据泄露的情况,它们分别是:从应用程序角度和从隐私类型角度,并以列表形式呈现给用户。同时,基于 Open and Close Policies 思想,本系统提出用户细粒度地控制隐私数据的使用方法,利用用户设置的 OCP 配置文件来控制应用程序使用隐私数据。在 OCP 配置文件中,包名表示应用程序的身份,序号表示应用程序的身份标签。每行的序号、包名和配置信息三者是一一对应的。配置信息是由 16 位的 0、1 组成,每一位分别对应表 1 中后

16 种污点标签的编码，“1”表示系统提供真实的隐私数据供该应用程序使用，“0”表示系统提供对应的伪造的隐私数据供该应用程序使用，具体如表 2 所示。

表 2 OCP 配置文件的格式
Table 2 Format of configuration of OCP

序号	包名	配置信息
1	XXX. XXX. XXX	1010101010101010
2	…	…

3 系统测试

3.1 测试环境

FlowMonitor 系统是基于 Ubuntu12. 04 _ LTS 操作系统、Android4. 3 官方源代码、gcc、g ++ 和eclipse 开发环境进行代码开发实现的。该系统分别在模拟器和真机上运行测试的，具体配置如表 3、表 4 所示。

表 3 虚拟机测试环境具体配置
Table 3 Testing environment of VM

项目	配置
AVD Device	3. 7’ WVGA（Nexus One）
CPU	ARM（armeabi-v7a）
RAM	512 MB
操作系统	Android4. 3. 1（api-18）
SDcard	200 MB
网络连接	172. 31. 21. 27

表 4 真机测试环境具体配置
Table 4 Testing environment

项目	配置
AVD Device	Galaxy Nexus（4. 65）
CPU	德州仪器 OMAP4460 1 228 MHz 双核
RAM	1 GB
操作系统	Android4. 3. 1（api-18）
SDcard	200 MB
网络连接	172. 31. 21. 27

3.2 示例分析

模拟 Android 系统中存在短信助手和 UB 浏览器两个应用程序参与共谋攻击，窃取系统短信。如图 6 所示。短信助手只申请了读取系统短信的权限，而无访问网络的权限，模拟参与共谋攻击中获取隐私方；UB 浏览器只申请了完全的网络访问权限，而无读取短信的权限，模拟参与共谋攻击中发送隐私方。当短信助手读取系统短信时，该短信内容污点标记 = TAINT _ SMS | 短信助手的身份标签，当短信助手通过通信将获取的短信内容发送给了 UB 浏览器，那么 UB 浏览器所获取的短信内容也带有污点标记 = TAINT _ SMS | 短信助手的身份标签。当 UB 浏览器将带有污点标记的短信内容发送到远端服务器时，污点标签捕获与分析模块就会捕获到该带有污点标签的短信信息，通过分析污点标签的身份信息，发现污点标签的身份标签为短信助手，与发送者 UB 浏览器的身份标签不相符，违反了 Android 系统安全需求，系统将此次共谋攻击双方身份信息和隐私泄漏信息记录在系统 Log 文件中以供用户查看，同时，将使用伪造的隐私信息发送给远端服务器，防止隐私泄漏。

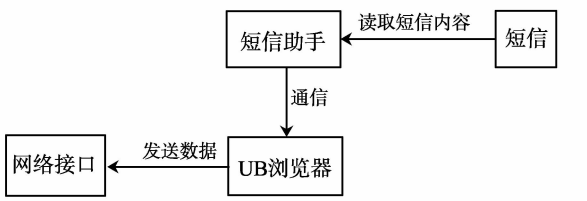


图 6 共谋攻击示例
Fig. 6 Example of collusion attack

针对共谋攻击方式造成隐私泄漏，我们设计了测试程序（短信助手、UB 浏览器和远端服务器）进行测试，测试程序的功能如表 5 所示。首先，验证当短信助手与 UB 浏览器只存在一方时，不会造成隐私泄漏；当短信助手与 UB 浏览器同时存在且运行时，系统短信被泄漏到远端服务器。测试结果如表 5 所示。然后，将 FlowMonitor 系统与 TaintDroid 系统、360 手机卫士进行对比，结果如表 6 所示。

表 5 验证共谋攻击
Table 5 Validation of collusion attack

应用程序	远端服务器接收结果
单独存在短信助手	没有任何反应
单独存在 UB 浏览器	null
同时存在短信助手与 UB 浏览器	接收到窃取的短信内容

表 6 共谋攻击的测试结果对比
Table 6 Comparison of collusion attack

隐私监控系统	能否监控短信助手泄漏隐私	能否监控 UB 浏览器泄漏隐私
FlowMonitor 系统	能，提示参与共谋攻击	能，提示参与共谋攻击
TaintDroid 系统	否	能，提示泄漏了短信类型隐私
360 手机卫士	能，提示有读取短信的行为	否

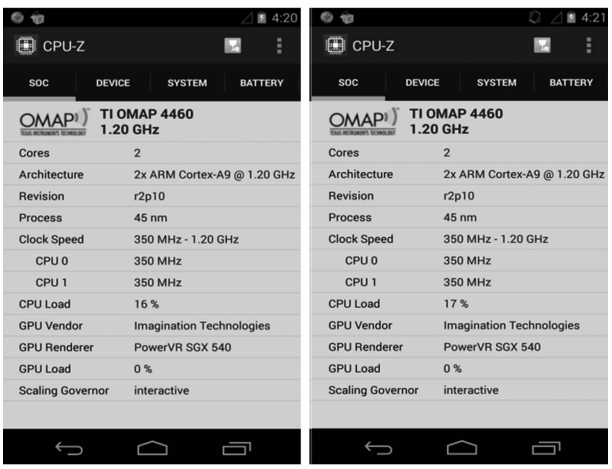
3.3 性能测试

FlowMonitor 在模拟器上运行测试，模拟器无法完全模拟真实机器，效率较低，但模拟器可以通过 DDMS 来查看系统性能信息，即当用户开启或关闭系统服务时，系统实际内存消耗占总内存大小的百分比。测试结果如表 7 所示。

表 7　性能测试结果
Table 7　Testing performance

FlowMonitor 系统监控服务状态	系统实际内存消耗占总内存的百分比/%
开启	44.55
关闭	42.39

此外,我们利用 CPU-Z 在真机上测试 FlowMonitor 系统的 CPU 使用情况。CPU-Z 是一款常见的 CPU 检测软件,目前官网已提供 Android 平台上的安装包。在 FlowMonitor 同时运行多个应用程序时,CPU 的使用率基本稳定在 20% 左右,与原生 Android 系统的 CPU 使用率几乎差不多,对用户体验的影响微乎其微。图 7 是在运行同样的应用程序下,原生 Android 系统和 FlowMonitor 系统的 CPU 使用率。



(a) 原生Android系统　　(b) Flow Monitor系统

图 7　CPU-Z 的测试结果的对比
Fig. 7　Comparison of testing results in terms of CPU-Z

3.4　功能测试

基于 Open and Close Policies 思想,FlowMonitor 系统提出细粒度用户控制隐私数据的使用方法,为用户提供应用程序和隐私类型两种不同查看角度。其中,系统初始为每个应用程序使用伪造隐私数据,即为 off,如图 8 所示。

图 8　两种查看隐私泄漏的角度
Fig. 8　Two ways of checking privacy leakage

在图 8 中,左图是从应用程序角度查看隐私泄漏情况,此时,图中显示应用程序“儿歌多多”泄漏了 2 条隐私数据,“CrackTest”泄漏了 1 条隐私数据,而其它应用程序目前未泄露隐私数据;右图是从隐私类型角度查看隐私泄漏情况,此时,图中显示短信类型的隐私数据泄漏了 1 条,IMEI 类型的隐私数据泄漏了 7 条,而其它隐私类型未泄漏。此外,两张都有“off”和“on”,用户通过点击按钮来控制应用程序对隐私数据的使用,其中,“off”表示使用伪造隐私数据,而“on”表示使用真实的隐私数据。初始情况都是“off”。

用户通过隐私泄漏统计图、一周趋势图、隐私泄漏频率统计图和隐私泄漏详细信息查看隐私泄漏情况,可以更好地了解隐私泄漏,合理地控制应用程序如何使用隐私数据,如图 9 ~ 12 所示。在图 9 中,纵轴表示应用程序类别,可以上下滑动;横轴表示隐私泄漏条数。不同颜色表示不同的隐私数据类型。例如:应用程序 CrackTest 共泄漏了 3 条隐私数据。其中 2 条 IMEI 隐私数据,为深色栏;1 条短信隐私数据,为浅色栏。图 10 表示在一周之内,系统每天共有多少条隐私被泄漏了。纵轴表示条数,横轴表示时间,例如在 5 月 23 日,系统共泄漏了近 24 条隐私数据。图 11 表示某一应用程序对于 16 种类别的隐私数据分别泄漏了多少条。纵轴表示条数,横轴表示 16 种隐私类型,可以滑动。例如,在图 9 中,针对 CrackTest 应用程序,隐私类型为短信的隐私数据泄漏 1 条。在图 12 中显示,应用程序 CrackTest 在 5 月 23 日 9 点 47 分 34 秒左右,将隐私数据 000000-000000000(在模拟器中 IMEI = 0)泄漏到 IP 地址 = 172.31.23.50 的服务器中。

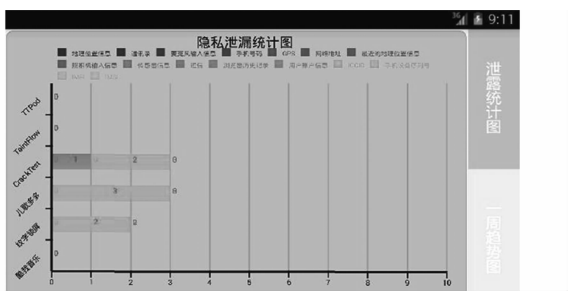


图 9　隐私泄漏统计图
Fig. 9　Statistical chart of privacy leakage

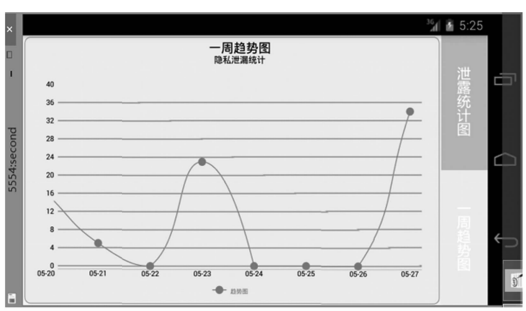


图 10　隐私泄漏一周趋势图
Fig. 10　Weekly trend of privacy leakage

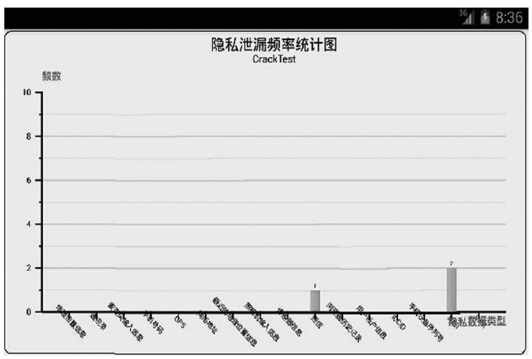


图11 隐私泄漏频率统计图
Fig. 11 The frequency of privacy leakage

图12 CrackTest 应用程序 IMEI 隐私泄漏详细信息
Fig. 12 IMEI leakage information of App CrackTest

3.5 样本分析

我们从国内豌豆荚应用市场(http://www.wandoujia.com/)上下载了13类共79个应用程序。发现其中58个应用程序是安全的,21个应用程序涉及隐私泄漏。其中,全部所测应用泄露的隐私数据均为IMEI,很多应用将隐私信息发送到2个以上的IP地址,例如:纹字锁屏、儿歌多多、口袋故事、QQ空间、淘宝、起点读书、书旗小说等。同时,在我们的检测范围内还有许多类别的应用没有检测到隐私泄露,例如:系统、输入类应用,壁纸、主题类应用,影音、图像类应用等。

4 结语

针对Android用户隐私数据安全防护方面的问题,本文提出并设计了FlowMonitor系统。FlowMonitor通过实时监控应用程序如何使用隐私数据,引入隐私数据流向实时监控跟踪机制,能够让用户清楚地了解隐私数据的具体流向,解决了XManDroid等系统无法发现恶意应用通过隐蔽通道获取隐私数据的弊端;同时,本系统利用基于RBAC思想的污点身份认证机制,在与TaintDroid相同的内存消耗前提下,有效地弥补了TaintDroid无法找到参与共谋攻击的恶意应用的不足。基于Open and Closed Policies的思想,FlowMonitor系统采用用户自主选择控制隐私数据的使用技术,控制粒度细化到数据本身,解决了XManDroid等系统自身权限机制控制粒度过粗的问题,并且,为用户提供一个界面友好、操作简单、呈现隐私数据使用状况的FlowMonitor应用程序,改善了目前主流的研究成果存在的用户交互性差、对用户专业素养要求高、使用困难等问题。用户利用该应用程序实现自主选择控制第三方应用对隐私数据的使用,同时查看第三方应用泄露隐私数据的情况,实时把握第三方应用是如何使用隐私数据、泄露了哪些隐私数据。用户通过设置OCP配置文件,实现细粒度地自主选择控制防护系统,提高手机的安全性。本系统也存在一些不足。由于部分第三方应用使用第三方库函数,而我们无法对第三方库函数进行修改,所以,当污点标签经过第三方库函数时,系统会出现污点跟踪丢失。此外,OCP文件中配置信息的初始值都为"0"。在未来工作中,我们将引入大数据技术分析第三方应用,获知第三方应用合理的隐私数据使用需求,优化配置信息的初始化工作,使系统更加智能化,防止隐私泄漏。

参考文献:

[1] 199IT. Statista:预计2015年Android将超Windows、iOS/Mac OS等设备之和[EB/OL]. [2015-03-14]. http://www.199it.com/archives/205652.html.

[2] 网秦. 2014年Q3全球手机安全报告[R]. (2014-11-14)[2015-03-12]. http://cn.nq.com/news/592.

[3] 吴倩, 赵晨啸, 郭莹. Android安全机制解析与应用实践[M]. 北京: 机械工业出版社, 2013.
WU Qian, ZHAO Chenxiao, GUO Ying. Analysis and application of Android security mechanism[M]. Beijing: Machinery Industry Press, 2013.

[4] 吴泽智,陈性元,杨智,等. 安卓隐私安全研究进展[J]. 计算机应用研究,2014,31(8):2241-2247.
WU Zezhi, CHEN Xingyuan, YANG Zhi, et al. Research progress on privacy security of Android[J]. Computer Application Research, 2014, 31(8): 2241-2247.

[5] ENCK W, OCTEAU D, MCDANIEL P, et al. A study of Android application security[C]//Proceedings of the 20th USENIX Security Symposium. Berkeley: USENIX Association, 2011: 101-113.

[6] BUGIEL S, DAVI L, DMITRIENKO A, et al. Xmandroid: a new android evolution to mitigate privilege escalation attacks[R]. Darmstadt: Technische Universitä,

2011.
[7] SMALLEY S, CRAIG R. Security enhanced (SE) Android: bringing flexible MAC to Android[C]//Network and Distributed System Security Symposium (NDSS'13). California: Internet Society, 2013:75-84.
[8] ENCK W, GILBERT P, HAN S, et al. TaintDroid: an information-flow tracking system for realtime privacy monitoring on smartphones[C]//ACM Transactions on Computer Systems, 2014, 32(2):393-407.
[9] 斯托林斯,布朗. 计算机安全: 原理与实践[M]. 2 版. 北京:电子工业出版社,2013.
STALLINGS, BROWN. Computer security: principles and practice[M]. 2nd. Beijing: Electronic Industry Publishing House, 2013.
[10] 国家密码管理局. GM/T 0002-2012 SM4 分组密码算法[S]. 北京:中国标准出版社,2012.

(编辑:许力琴)

(上接第 58 页) 属性证明方案,对组成属性的组件之间结构进行了分析,并以构成属性的组件为有序序列为例讨论了证明协议的建立过程。多组件属性证明方案具有良好的隐私保护和灵活可配置的证明机制,可以采用一个功能模块对不同组件个数的属性进行证明,属性证书易于撤销和验证。属性与组件承诺分别进行验证,提高属性证明的证明效率的同时也使平台的隐私得以更好保护。下一步工作主要对多个组件之间的结构以及多组件属性的证明方式进行研究,找出更加符合组件与属性之间关系的映射方式来构造证明方案。

参考文献:

[1] BRICKELL E, CAMENISCH J, CHEN Liqun. Direct anonymous attestation[C]//Proceedings of the 11th ACM Conference on Computer and Communications Security. New York: ACM, 2004:132-145.
[2] 张倩颖, 冯登国, 赵世军. 基于可信芯片的平台身份证明方案研究[J]. 软件学报, 2014, 35(8):95-106.
ZHANG Qianying, FENG Dengguo, ZHAO Shijun. Research on platform identity authentication scheme based on trusted chip [J]. Journal of Software, 2014, 35(8):95-106.
[3] CHEN Liqun, LANDFERMANN R, LÖHR H, et al. A protocol for property-based attestation [C]//Proceedings of the 1st ACM Workshop on Scalable Trusted Computing. NewYork: ACM, 2006:88-102.
[4] 李尚杰,贺也平,刘冬梅,等. 基于属性的远程证明的隐私性分析[J]. 通信学报,2009,11A:146-152.
LI Shangjie, HE Yeping, LIU Dongmei, et al. Privacy analysis of remote attestation based on attribute[J]. Journal of Communication, 2009, 11A:146-152.
[5] 徐晓燕,赵荣彩,闫丽景. 软件度量的研究与进展[J]. 信息工程大学学报,2014,15(5):622-627.
XU Xiaoyan, ZHAO Rongcai, YAN Lijing. Research and development of software metrics[J]. Journal of Information Engineering University, 2014, 15(5):622-627.
[6] 秦宇,冯登国. 基于组件属性的远程证明[J]. 软件学报, 2009, 20(6): 1625-1641.
QIN Yu, FENG Dengguo. Remote attestation based on component attributes[J]. Journal of Software, 2009, 20(6):1625-1641.
[7] PORITZ J, SCHUNTER M, HERREWEGHEN E V, et al. Property attestation-scalable and privacy-friendly security assessment of peer computers[J]. Biotechniques, 2004, 27(3):223-238.
[8] CAMENISCH J, LYSYANSKAYA A. A signature scheme with efficient protocols[C]//Proceedings of 3rd Conference on Security in Communication Networks. Berlin: Springer-Verlag, 2002, 2576:268-289.
[9] CAMENISCH J, GROTH J. Group signatures: better efficiency and new theoretical aspects[J]. Lecture Notes in Computer Science, 2010, 3352:120-133.

(编辑:许力琴)