doi :10.3969/j.issn.1671-1122.2016.08.004

一种基于 Android 内核的 APP 敏感行为检测方法及实现

文伟平，汤炀，谌力
（北京大学软件与微电子学院，北京 102600）

摘　要：进入移动智能终端时代后，Android 手机操作系统由于其开放、免费的特点，成为市场上的主流操作系统之一。然而，大量针对 Android 系统编写的病毒、木马和恶意软件已经严重威胁到智能手机用户的个人隐私和财产安全。虽然在 Android 系统中进行敏感操作必须向系统申请相应的权限，Android 系统中也存在权限控制相关的系统模块，但是恶意软件可以借助系统漏洞或第三方程序漏洞进行攻击。鉴于 Android 应用敏感行为检测的需要，文章研究分析了现有 Android 系统中应用程序行为检测方法，设计并实现了针对 Android 系统应用程序的动态行为监测系统，该系统能够实时监控应用程序的敏感行为，为恶意程序检测提供帮助。

关键词：Android 应用；敏感行为；动态监测
中图分类号：TP309　　**文献标识码：**A　　**文章编号：**1671-1122（2016）08-0018-06

中文引用格式：文伟平，汤炀，谌力 . 一种基于 Android 内核的 APP 敏感行为检测方法及实现 [J]. 信息网络安全，2016（8）：18-23.

英文引用格式：WEN Weiping，TANG Yang，SHEN Li. An APP Sensitive Behaviors Detection Method Based on Android Kernel and Its Implementation[J].Netinfo Security, 2016 (8) : 18-23.

An APP Sensitive Behaviors Detection Method Based on Android Kernel and Its Implementation

WEN Weiping, TANG Yang, SHEN Li
(*School of Software & Microelectronics*, *Peking University*, *Beijing* 102600, *China*)

Abstract: In the era of intelligent mobile terminal, because of the characteristics of openness and free of charge, Android has become one of the major operation systems on the market. However, a large number of Android viruses, Trojans and malicious software have been serious threats to the privacy and property securities of smart phone users. In the Android operation system, although it is necessary to apply the authorities to the system for sensitive operations, and there are some system modules related to authority control, malicious software can use the system vulnerabilities or third party program vulnerabilities to carry out the attack. To meet the need of Android applications sensitive behaviors detection, this paper analyzes the popular applications behaviors detection tools in the Android system, designs and implements an Android applications dynamic detection system. The system can monitor the Android applications sensitive behaviors in real time, and provides help for the detection of malicious programs.

Key words: Android application; sensitive behavior; dynamic monitor

收稿日期：2016-04-18
基金项目：国家自然科学基金 [61170282]
作者简介：文伟平（1976—），男，湖南，副教授，博士，主要研究方向为网络攻击与防范、恶意代码研究、信息系统逆向工程等；汤炀（1992—），男，广西，硕士研究生，主要研究方向为 Android 安全；谌力（1991—），男，湖北，硕士研究生，主要研究方向为 Android 安全。
通信作者：文伟平 weipingwen@ss.pku.edu.cn

0 引言

Android 系统是一款基于 Linux 内核且开放源代码的移动操作系统。基于 Linux 内核使得 Android 系统能够长期稳定运行；而开放源代码使得任何公司和个人都可以通过修改其源代码进行定制，使其适应不同的硬件平台。自从 Android 初创团队于 2005 年被 Google 收购以来，Android 系统借助 Google 强大的科研、资本和运营能力，逐步侵占其他移动操作系统的市场占有率，成为近几年市场占有率第一的移动操作系统。大量开发者选择在该平台上进行移动软件开发，使得 Android 系统成为全球应用最多的移动平台。

但是，由于 Android 系统的开放性以及开发者水平良莠不齐，使得 Android 系统存在许多安全问题和漏洞。Android 系统的安全问题和漏洞主要涉及通信系统、应用软件、隐私信息与设备安全四个方面。通信系统方面的安全风险包括攻击者在用户不知情的情况下拨打电话或挂断电话、发送垃圾短信等安全问题 [1]。应用软件方面则主要涉及恶意软件发起的攻击。隐私信息方面则多指用户隐私信息及敏感信息的泄露。设备安全则主要是因为手机被盗而遭遇的信息泄露以及因此带来的经济损失。

1 Xposed 框架简介及原理

要应对 Android 系统的安全问题，其中重要一点就是要对 Android 应用程序进行监测，发现并找出威胁 Android 系统安全的敏感行为。而要对敏感行为进行检测，不仅需要对应用程序在内核层上的操作进行监测，同时也要对 Dalvik 虚拟机中的敏感操作进行监测。Xposed 是一款可用于监测 Dalvik 虚拟机中应用程序函数调用及 hook 函数的开源框架。该框架能够在不修改 APK 的情况下影响程序的运行，也能够基于该框架制作出许多功能强大的模块。

Zygote 是 Android 系统中所有应用程序的父进程。Xposed 框架通过替换 /system/bin/app_process 程序可以修改 Zygote 进程，从而使得所有 Android 应用启动时均会带有 Zygote 中已经加载的 XposedBridge.jar 文件，而该文件中的函数可以通过 hook 函数的方式实现对 Dalvik 虚拟机的劫持。

当需要对应用程序中的函数进行 hook 时，可以通过 Xposed 框架下的 findAndHookMethod（）对指定的函数进行 hook。要在 findAndHookMethod（）中 hook 指定的函数，首先需要被 hook 的函数的相关信息，如函数所在的具体类名、函数名、函数参数的具体类型名；然后通过重写 beforeHookedMethod（）和 afterHookedMethod（）自定义被 hook 的函数执行前后所要进行的操作。被 XposedBridge.jar 文件所 hook 的函数的执行流程变为 beforeHookedMethod（）→ originMethod（）→ afterHookedMethod（）。

本文将采用本地程序与 Xposed 框架结合的方式，对应用程序在 Linux 层面上的敏感行为以及在 Dalvik 层面上的敏感行为进行监测，全面了解应用程序在运行过程中的行为。

2 检测系统设计

2.1 系统组成

Android 应用程序敏感行为动态监测系统一共有 4 个模块：隐私文件读取监测模块、联网动作监测模块、内核层与应用层的通信模块和基于 Xposed 的应用层敏感行为监测模块。

1）隐私文件读取监测模块。该模块主要利用 ARM Linux 系统调用拦截机制将读取文件的系统调用替换成自定义的函数。自定义函数对短信文件数据库文件、系统设置文件数据库文件、联系人数据库文件、日历数据库文件进行监控 [2]，如果应用程序出现读取这些文件的动作，则隐私文件读取监测模块就会将这些行为记录下来，并且通过内核层与应用层的通信模块传递给应用层。

2）联网动作监测模块。该模块主要通过 hook Netfilter 网络数据包过滤机制实现对网络数据包的拦截 [3]，将得到的网络数据包根据网络数据包格式进行解析，获取相应的数据，如数据包的 IP 地址、端口号等 [4]。相关信息通过内核层与应用层的通信模块传递给应用层。

3）内核层与应用层的通信模块。内核层与应用层的通信模块主要通过 Netlink Socket 机制实现。该机制一方面向内核层提供一些 API，另一方面向应用层提供 Socket 通信接口，通过 API 和 Socket 通信接口实现内核层和应用层的全双工通信，将在内核层检测到的信息传递给应用层。

4）基于 Xposed 的应用层敏感行为监测模块。由于使用了 Xposed 框架，应用层的敏感行为检测可以通过编写

Xposed插件的形式完成。在插件中使用Xposed提供的函数，可以很方便地完成 hook 操作而无需复杂的编程。

该系统框架的入口是自动化测试工具，自动化测试工具启动要检测的应用程序以后，系统才能对应用程序的行为进行检测[5]。

2.2 系统执行流程

Android 应用程序敏感行为动态监测系统流程图如图 1 所示。首先将隐私文件读取监测模块和联网动作监测模块加载到系统内核；然后在应用层启用基于 Xposed 的应用层敏感行为监测模块；再通过自动化测试工具启动要检测的应用程序。一旦内核层的监测模块检测到相应的恶意行为，就会通过内核层与应用层的通信模块将信息传递到应用层。应用层的监测模块一旦检测到敏感行为，也会将相应信息传递给监听程序并显示出来。

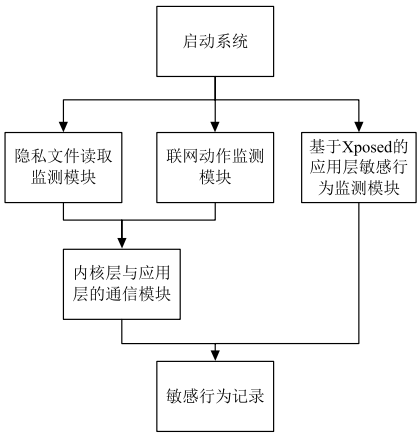

图1 Android应用程序敏感行为动态监测系统流程图

2.3 系统模块设计

2.3.1 内核层与应用层的通信模块的设计与实现

内核层与应用层的通信模块之所以采用 Netlink Socket 机制而不采用系统调用方式进行通信，主要原因是 Netlink Socket 通信和其他 Socket 通信一样，实现了消息缓冲机制。在一些特殊情况下，大量消息同时进行传递时，采用 Netlink Socket 机制可以选择性地对排队的消息进行处理，而不是每个都要处理。系统调用不存在缓冲机制，一个系统调用发生，内核就必须进行相应处理，一旦有大量消息传入内核，内核在很长时间内都在处理消息，必然会影响内核的其他正常功能。

内核层进程与应用层进程使用 Netlink Socket 机制进行通信如图 2 所示。

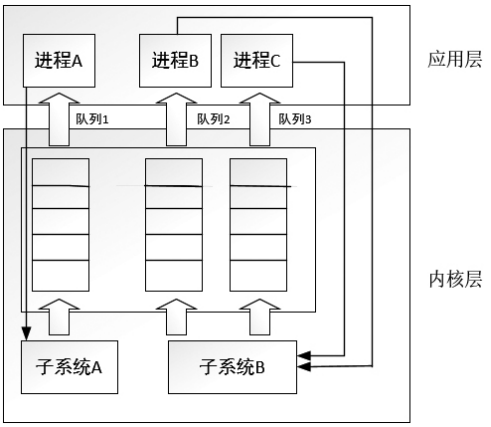

图2 内核层进程与应用层进程通信图

由图 2 可以知道，内核层进程与应用层进程使用 Netlink Socket 机制进行通信有两种方式，一种是一对一的点到点通信，另一种是一对多的广播通信。另外可以看到，内核层进程给应用层进程发送的消息会进入一个消息队列，而应用层进程给内核层进程发送的消息是实时通信的。本文使用的是点到点的通信方式。

2.3.2 隐私文件读取监测模块的设计与实现

隐私文件读取监测模块利用 LKM 机制加载到 Android 系统的内核里，实现对隐私文件读取动作的监测。无论应用程序是在应用层还是内核层对隐私文件如 Android 系统上的短信、联系人、日历、系统设置文件等进行任何方式的读取，最终都会转为内核层的系统调用[6]。打开文件的系统调用为 sys_open，读取文件的系统调用为 sys_read。通过 Android 系统的 ARM Linux 系统调用拦截机制可以获取内核层的系统调用表，每个系统调用对应着系统调用表里的一个偏移地址，找到相应的系统调用并且将其替换成自定义函数[7]。在自定义函数里对短信、联系人、日历、系统设置等相关文件进行检测，一旦发现应用程序要读取这些文件，就将相应的信息通过内核层与应用层的通信模块传递到应用层。隐私文件读取监测模块执行流程如图 3 所示。

2.3.3 联网动作监测模块的设计与实现

联网动作监测模块通过 hook Netfilter 机制实现。Netfilter 是 Linux 系统下的第三代防火墙，其主要通过将一系列系统调用函数内嵌到基于 IP 协议栈的网络数据包处理路径上实现对网络数据包的获取、处理以及转发[8]。网络数据包在内核层经过协议栈时的流向有 3 种：流入、流

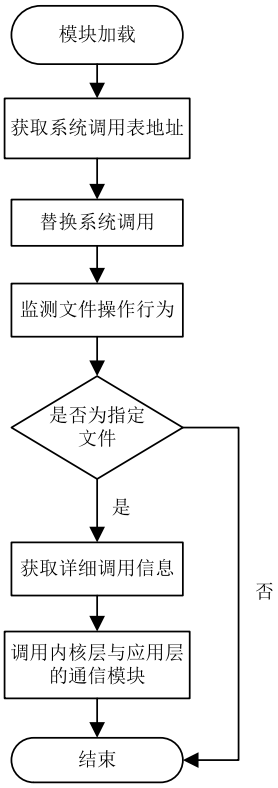

图3 隐私文件读取监测模块执行流程

出、流经。Netfilter在这3种流向上设置了5个hook点，这5个hook点的位置如图4所示。

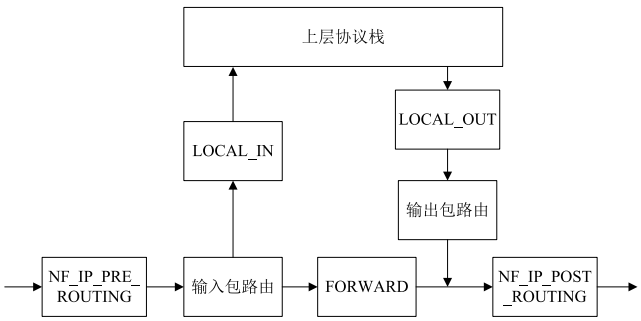

图4 Netfilter hook点设置图

利用Netfilter提供的hook机制，在NF_IP_PRE_ROUTING以及NF_IP_POST_ROUTING这两个hook点注册自定义函数对网络数据包进行拦截，并对其中的数据进行解析，提取网络数据包里相应的IP号以及端口号[9]。Android内核默认是没有Netfilter功能的，要想把相应的Netfilter功能加入到内核，需要重新对内核进行编译。

2.3.4 基于Xposed的应用层敏感行为监测模块的设计与实现

为了对APP里面的类中的函数进行hook，针对非构造函数，需要使用findAndHookMethod( )，该函数的原型声明如图5所示。

```
public static XC_MethodHook.Unhook findAndHookMethod(
  String className,
  ClassLoader classLoader,
  String methodName,
  Object... 
  parameterTypesAndCallback)
```

图5 findAndHookMethod()原型

调用上述静态函数，其参数className指要被hook的函数所在类的路径，classLoader指类加载器，methodName指被hook的函数的函数名，Object...指被hook的函数的参数类型所在的类。parameterTypesAndCallback指回调函数，即监测系统的自定义函数，本文使用Xposed提供的XC_MethodHook()作为参数，在XC_MethodHook()的函数体内实现自定义功能。以hook public void startActivityForResult(Intent intent, int request Code）为例，该函数所在的类为android.app.Activity，参数类型所在的类分别是android.content.Intent.class和int.class，因此可以把findAndHookMethod()的实现写成图6所示的形式。

```
findAndHookMethod(
  "android.app.Activity",
  lpparam.classLoader,
  "startActivityForResult",
  android.content.Intent.class,  int.class,
  new XC_MethodHook() {}
);
```

图6 findAndHookMethod（ ）的实现例子

为了将hook后得到的结果传回测试程序，在自定义函数中使用了Socket机制作为通信方式。通过建立Socket连接，将数据传回测试程序的网络连接线程，再通过线程间通信，将数据交由主线程进行展示。

3 系统实现

首先创建Android模拟器，对Android内核进行编译，且把支持动态加载模块以及Netfilter功能编译进内核。其次对隐私文件读取监测模块和联网动作监测模块进行交叉编译,最终生成.ko文件。此时所有的测试环境都已搭建好，之后需要将Android应用程序敏感行为动态检测系统的各功能模块加载到搭建好的系统里。通过如下命令 /emulator -verbose -show-kernel-kernel/root/kernel/goldfish/arch/arm/boot/zImage -avd test 启动模拟器，让它运行在最新编译好的内核上。启动情况如图7所示。

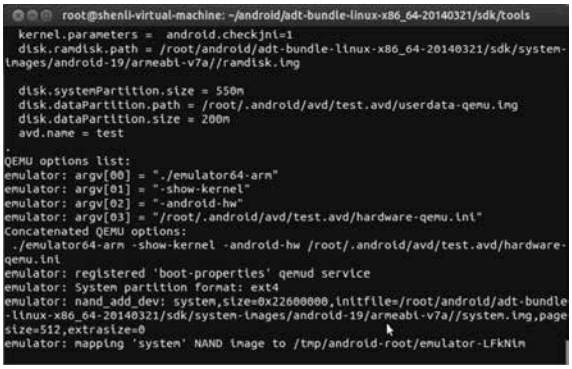

图7 模拟器启动

1）内核层模块监测效果

模拟器启动后，首先通过终端命令 adb push 将隐私文件读取监测模块和联网动作监测模块上传至模拟器 data 目录下的 local 文件夹，同时也将通信程序的可执行文件通过命令 adb push 上传到和模块一样的目录下；然后执行命令 insmod 将模块分别加载到模拟器运行的内核上[10]；再运行内核层与应用层的通信模块，将内核层获取到的信息传递给应用层。通过启动模拟器里的短信功能、拨打电话功能、系统设置功能来模拟恶意程序对隐私文件的读取，通过启动浏览器来模拟恶意程序进行联网通信。通过查看模拟器控制台以及通信程序可以观察到哪些隐私文件被读取以及联网网络数据包的相关信息。

图 8 所示为当模拟启动短信功能、拨打电话功能、手机设置功能时，相应的文件被读取时的信息。

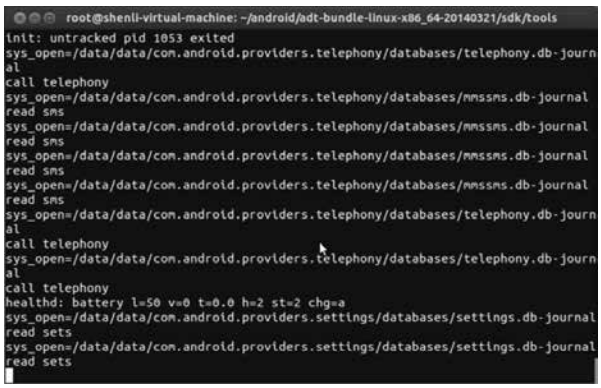

图8 内核层隐私文件读取监测

图 9 是当启动浏览器模拟恶意程序联网行为时监测到的网络数据包信息。可以检测到网络数据包的源 IP 地址、目的 IP 地址以及源主机上的端口号、目的主机上的端口号。

2）应用层 Xposed 监测效果

首先在 findAndHookMethod() 里分别对与发送短信、拨打电话、联网行为相关的方法进行注册，然后在模块中把

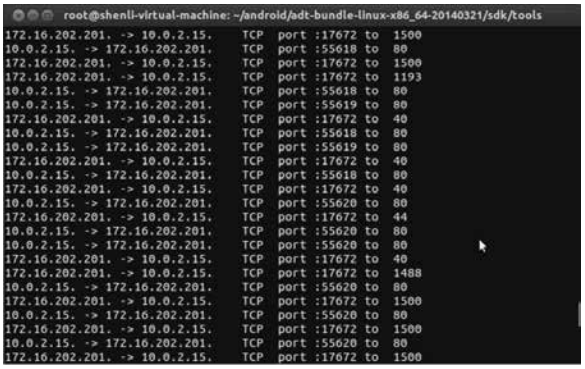

图9 内核层联网动作监测

与短信信息、拨打电话信息、联网信息相关的信息传递给通信程序 hookphone 并显示出来。

分别启动短信功能、拨打电话功能、浏览器来模拟恶意程序行为，同时运行 hookphone 通信程序，可以看到其记录的相关信息如图 10 所示。

图10 应用层信息监控

通过对隐私文件读取监测模块、联网动作监测模块和基于 Xposed 的应用层敏感行为监测模块的实验发现，我们可以监测到应用程序在运行过程中读取隐私文件、发送网络数据等敏感行为，从而判断出应用是否可能存在恶意行为。

4 结束语

自 2011 年以来，Android 手机的使用越来越广泛，且 Android 系统的应用程序也越来越多。然而 Android 应用程序下载平台的出现，给恶意应用程序提供了广泛的传播途径。由于很多 Android 应用程序上架前不进行检测或者检测不到位，导致很多恶意应用程序在 Android 应用程序下载平台中出现，一旦这些恶意程序下载到用户的手机里，不但给用户信息安全造成很大威胁，还可能给用户带来财

产上的损失。所以对 Android 应用程序敏感行为进行监测具有一定的现实意义。

本文提出的 Android 应用程序敏感行为动态检测系统从内核层和应用层两方面对 Android 应用程序敏感行为实施监测。在内核层基于系统调用拦截机制对应用程序隐私文件读取动作进行监测，基于 Netfilter 机制对应用程序的联网动作进行监测。利用 Netlink Socket 机制将监测到的行为传递给应用层。在应用层使用基于 Xposed 的模块对应用程序的发送短信、连接网络、拨打电话行为进行监测。通过将内核层检测与应用层检测结合起来，本文系统能对 Android 应用程序的敏感行为进行一个全面的监控，保证应用程序的敏感操作能够被完全记录下来，从而保证用户的隐私安全。

本文虽然介绍了一种监测应用程序敏感行为的方法，但并不能够真正地从众多的应用程序中检测出恶意程序。在下一步的工作中，可以将监测数据用于机器学习，探寻合法应用与恶意应用在敏感行为种类、频率等方面的区别，形成一套以敏感行为为基础的恶意应用检测算法，用于筛选出真正的恶意应用。（责编 马珂）

参考文献:

[1]ZHAO Kao, ZOU Deqing, JIN Hai, et al. Privacy Protection for Perceptual Applications on Smartphones[C]//IEEE.2015 IEEE International Conference on Mobile Services (MS),June 27-July 2, 2015. New York City, NY, USA.NJ:IEEE,2015:174-181.

[2] 张航 . 基于系统调用的文件系统入侵检测的设计与实现 [D]. 武汉 : 华中科技大学，2009.

[3]SETH S, VENKATESULU M A. sk_buff and Protocol Headers[M]// TCP/IP Architecture, Design, and Implementation in Linux. NJ:Wiley-IEEE Press, 2008: 181-203.

[4]SURIARACHCHI A. Achieving High-throughput Distributed, Graph-based Multi-stage Stream Processing[D]. Fort Collins: Colorado State University, 2016.

[5] 落红卫 . 移动恶意代码分析及检测技术研究 [J]. 中国多媒体通信，2015（7）: 36-37.

[6]BERMAN A, BOURASSA V, SELBERG E. TRON: Process-Specific File Protection for the UNIX Operating System[EB/OL].https://www.researchgate.net/publication/2656011_TRON_Process-Specific_File_Protection_for_the_UNIX_Operating_System, 2016-03-21.

[7] 时金桥，方滨兴，胡铭曾，等 . Linux 系统调用劫持 : 技术原理、应用及检测 [J]. 计算机工程与应用，2003，39（32）: 167-170.

[8] 郝传国 . 基于 Android 平台的内核 Rootkit 检测及防护研究 [D]. 成都 : 电子科技大学，2013.

[9]HE K K. Why and How to Use Netlink Socket[J]. Linux Journal, 2005(130):14-19.

[10] 周应华 . 对 Linux 可加载内核模块应用框架的研究 [J]. 计算机系统应用，2007（4）: 69-72.