

Android智能手机系统中文件实时监控的研究与实现

温敏 艾丽蓉* 王志国

(西北工业大学计算机学院,西安 710129)

摘要 保证文件的安全是保护系统安全的一个重点,通过文件监控来保证数据的完整性是保护系统安全和灾情评估的基础。通过对 Android智能手机操作系统的研究和分析,提出了在 Android系统下采用基于文件信息节点的监控机制实现对文件的实时监控,并详细的描述了在 Android平台下实现的关键数据结构、实现细节以及实验测试结果。

关键词 文件监控 Android 智能手机

中图分类号 TN929.5; 文献标志码 B

随着手机技术的不断发展,一方面,手机功能不断增强,向智能方向发展,手机与个人电脑之间的界限越来越模糊,许多用户将自己的私密信息存放在手机中,这些信息一旦泄露或损坏,对手机用户造成的损失将无法估量;另一方面,随着智能手机的普及,针对手机的恶意软件也越来越多,这些恶意软件包括蠕虫、木马、间谍软件及其他病毒等,随着智能手机在全球范围内销量的急剧增加,恶意软件大规模扩散的危险也越来越严重,给手机上用户的信息安全带来了极大的威胁。

对于一个安全的计算系统而言,保证其文件的安全是保护系统安全的一个重点,对文件实施监控,对于从文件管理器到安全工具的各种应用程序都是必要的,也是保护系统安全乃至以后的灾情评估的基础,因此需要对文件的实时监控技术进行分析和研究,也就是通过文件监控来保证数据的完整性^[1]。

针对当前智能手机操作系统多种多样,本文采

用了 Android手机系统,通过对 Android智能手机操作系统的研究,在此基础上描述了一套对手机中文件进行实时监控的方法和实现细节。在当前手机数据通信交换通道多样化的环境下能对智能手机中的文件进行有效的监控。

1 Android智能手机系统简介

Android系统是 Google公司开发的基于 Linux平台开源手机操作系统,该平台由操作系统、中间件、用户界面和应用软件组成,是首个为移动终端打造的开放和完整的移动软件。

Android系统基于 Linux 2.6内核来提供系统的核心服务,例如安全机制,内存管理,进程管理,网络堆栈和驱动模块。其包含一组核心库,提供了 Java语言核心库内的大部分功能。Android应用程序运行于 Dalvik虚拟机之上,该虚拟机是基于存储器的,运行经过 Java语言编译器的类,这些类通过"dx"工具被转换成.dex(Dalvik Executable)格式。Android中应用程序一种是 APK(Android Package)格式文件,是类似于 ZIP文件的压缩文件,可以直接运行于 Dalvik虚拟机上。

Android系统中文件系统采用的是一种针对 NAND设备的新型文件系统 Yaffs^[2],读写速度更快,并支持大容量的 NAND-Flash芯片。

2008年11月24日收到

陕西省自然科学基金研究计划项目
(2007F45)、西安市科技创新支撑
计划项目(YF07019)资助

第一作者简介:温敏(1985—),男,硕士研究生,研究方向:智能手机的安全防护。

*通信作者简介:艾丽蓉(1970—),女,副教授,研究方向:软件安全,信息安全与智能处理。

2 设计原理

Linux内核文件系统管理中,每一个目录或文件在内核中都对应一个唯一的文件信息节点(inode);inode节点是内核管理文件系统的最基本单位^[3]。如果对文件或目录对应的inode节点实行监控,将是一种细粒度的、非常有效的文件监控措施。通过对Android系统Linux内核的分析,尤其是对文件系统源代码的深入分析和理解,本文提出了在Android系统下采用基于inode节点的文件监控机制,实现对文件的有效监控,其实现可分以下几个步骤(见图1)。

(1)Android应用程序所使用的文件类型,包括txt、jpg、mp3、avi等常见类型普通文件和二进制文件,以及直接运行于Android Dalvik虚拟机上的APK(Android Package)格式文件;

(2)创建监控文件列表,要实施监控的文件将保存在此表中;

(3)通过接口函数 `inotify_add_watch()` 为监控文件列表中的每个文件或目录inode节点创建监控实例,这一过程在内核中通过系统调用 `sys_inotify_add_watch()` 实现;若要停止对某文件或目录的监控通过接口函数 `inotify_rm_watch()` 取消其所对应的监控实例,在内核中通过系统调用 `sys_inotify_rm_watch()` 实现。基于inode节点的文件监控机制使用文件描述符fd作为基本接口,通过使用文件I/O操作 `select` 和 `poll`^[4] 来监控文件系统的变化;

(4)对Android系统Linux内核修改后,使用交叉编译工具重新编译内核,用生成的新内核镜像文件替换Android系统原来的镜像;

(5)启动Android系统,通过设置的监控列表对其中的文件进行监控,获得文件被访问、修改、打开或关闭、创建、删除等信息,同时将监控文件信息及相应事件记录到日志文件,以便查看和对文件完整性的检查;

(6)监控文件信息及相应事件记录由内核发送到用户应用程序,使用Android SDK提供的日志类

建立日志文件,用来详细记录文件保护系统每天发生的各种各样的事件,达到审核和监测的目的。

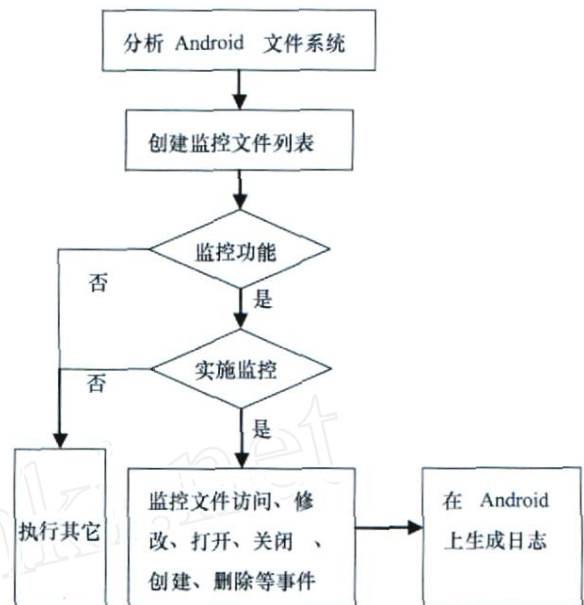


图1 文件实时监控流程图

3 具体实现

3.1 创建监控文件列表

通过对内核中文件系统及有关数据结构的分析,结合文件监控的功能需要,确定了监控文件列表所采用的数据结构 `struct filelist`,并建立链表 `watch_filelist`,所有的受监控文件的文件名和inode信息都写入到该链表中,该数据结构如下所示。

```

typedef struct filelist{
    char filename [ FLENAME_SIZE];
    struct inode * inode;
    struct filelist * next;
} * watch_filelist;
  
```

另外通过使用 `proc_dir_entry` 数据结构(`include/linux/proc_fs.h` 中),在内存 `/proc` 文件系统^[5] 中建立列表文件 `filelist_proc_file` 和控制文件 `fp_control_proc_file`,这样可以动态的添加新文件到 `/proc` 的监控文件列表中。

在 `/proc` 文件中创建的文件,可根据函数 `read_proc` 和 `write_proc` 对这些文件实现数据的读写

操作^[3]。通过对列表文件 `filelist_proc_file` 的读写,可以动态添加新的文件到 `watch_filelist` 中;通过将控制命令写入到控制文件 `fp_control_proc_file` 可实现对 `watch_filelist` 更新、删除等操作。下面是 `filelist_proc_file` 和 `fp_control_proc_file` 的实现代码:

```
filelist_proc_file = create_proc_entry("watch_filelist", 0644, fp_proc_dir); //创建 filelist_proc_file
filelist_proc_file->data = data_buf;
//将信息写到 filelist_proc_file
filelist_proc_file->read_proc = proc_read_filelist; //从 proc 读取文件信息
filelist_proc_file->write_proc = proc_write_filelist; //向 proc 写入文件信息
最后由函数 proc_write_filelist() 调用 create_list(( &watch_filelist, f_data, sizeof f_data) 函数, 将要监控的文件添加到 watch_filelist 链表中。
```

3.2 获得文件监控信息

在对 `watch_filelist` 中的文件进行监控后,需要获得在监控的文件上所发生的事件;通过分析,确定的数据结构为 `struct watch_event_queue`,即通过建立事件队列来保存该文件上所发生的事件,该数据结构如下所示:

```
struct watch_event_queue {
    int capacity; //队列容量
    int front; //上一节点位置
    int rear; //下一节点位置
    int size; //当前队列大小
    watch_event_queue ** event; //指向事件节点
} * events; 当 event->size = 0 时,表示事件队列为空,当 event->size = event->capacity 时表示事件队列已满。初始化事件队列后,对发生的事件记录在事件队列结构中,完成操作为 events->size++; events->rear = next_position(events->rear, events), 所有发生的事件将按照发生的先后次序链接在队列中,最后按照先进先出的顺序输出,并保存在生成的日志文件 watch_log 中。
```

3.3 交叉编译内核

Android 手机系统所使用的镜像文件位于 Android SDK 中 `images` 文件中, 包含有 `ramdisk.img`, `system.img`, `userdata.img`, `kernel-qemu` 四个镜像文件。其中 `ramdisk.img` 是启动使用的根文件系统映像, `system.img` 是 Android 的系统镜像, `userdata.img` 为保存用户数据的镜像文件, `kernel-qemu` 则是需要

替换的 Linux2.6 内核镜像文件。

因为 Android 系统使用的是嵌入式 Linux 系统, 采用较新的 `arm926-ej-s` 内核, 所以要使用交叉编译工具编译内核。这里使用了 `arm-none-eabi` 交叉编译工具, 在 `linux2.6` 环境下, 执行 `make menuconfig` 命令配置 Android 的内核编译参数, 然后修改 `kernel` 目录中的 `Makefile` 文件, 其中 `ARCH = arm`, `CROSS_COMPILE = arm-none-eabi`。最后用 `make` 命令生成镜像文件, `kernel/arch/arm/boot/zImage` 文件就是我们需要的内核镜像。

3.4 查看日志信息

在对监控列表中的文件进行监控后, 发生在文件上的所有事件都将被记录在系统中的 `watch_log` 文件中; 为方便查看发生在监控文件上的事件信息, 我们在 Android 平台下设计了一个小型的浏览器 `log_browser.apk`, 应用 API 函数为 `FileReader()`, 关键实现代码如下:

```
StringBuffer newTxtFileBuf = new StringBuffer(4000); //接收信息
fileRead = new FileReader(file); //读取文件信息
BufferedReader bufRead = new BufferedReader(fileRead); //缓存信息
newTxtFileBuf.append(bufRead.readLine()); //读取信息
```

通过 `adb install` 命令将 `log_browser.apk` 安装在 Android 系统中, `log_browser` 如图 2 所示。

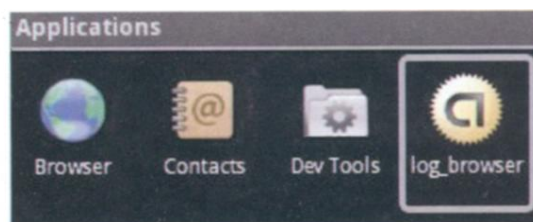


图 2 日志查看器 `log_browser`

4 测试

Android 提供了 `Emulator` 模拟器, 不需使用物理设备即可预览、开发和测试 Android 应用程序的功能。同时还提供了 `Android Debug Bridge (adb)` 调试桥, 它是通用的 `debug` 工具, 通过它可以管理设备或者模拟器的状态, 使用 `adb shell` 命令进入模拟器,

在 shell状态可以查看系统、执行命令。

测试环境使用 Android SDK为 android-sdk-windows-0.9_beta版本,其采用了 Linux-2.6.25-Android内核。将测试文件 test.txt /sys /etc加入到监控列表进行监控,对其测试的结果如图 3所示。

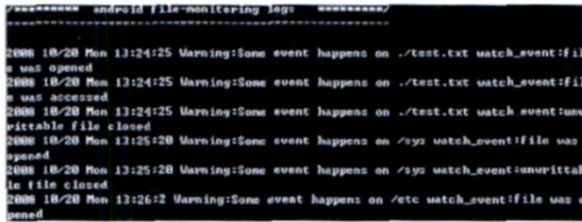


图 3 打印监控文件信息

同时,在 Android系统的根目录下生成日志文件 watch_log (如图 4所示),用 log_browser打开 watch_log可以看到发生在测试文件上的事件信息(如图 5所示)。



图 4 生成 watch_log 文件

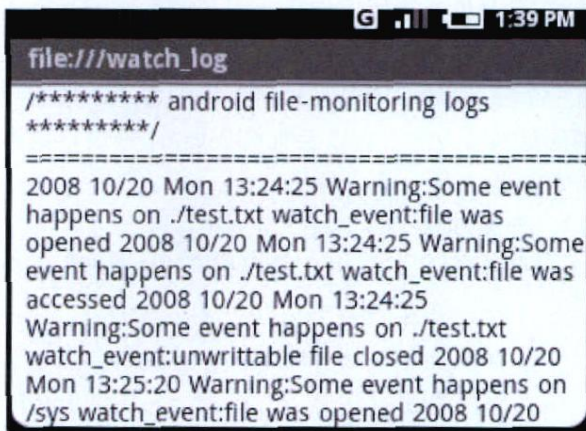


图 5 生成日志文件

从上述测试结果可以看出,对发生在监控文件上的事件能够进行有效的监控,可以达到实时监控文件的目的。

5 结论

本文实现的在 Android系统中采用文件的实时监控技术,已经在西安市科技创新支撑计划项目——“智能手机的安全防护技术”中使用,能有效、实时、准确的记录监控文件上发生的操作,并以日志形式记录到手机当中去,能较好的满足当前的一些应用。当然,该技术还需要进一步的完善,需要在日后的工作中不断地探索和研究,以形成一套实用的文件监控系统。随着 Google公司大力推广 Android平台和 Android智能手机的推出,相信会有很好的应用前景。

参 考 文 献

- 1 王启智,申功迈,单和平. 实用 UNIX和 Internet安全技术. 北京:电子工业出版社,1999:139—237
- 2 毛勇强,黄光明. 武汉:华中师范大学物理学院. YAFFS文件系统 在嵌入式 Linux的实现 <http://www.eaw.com.cn/news/display/article/4170>
- 3 BarM. Linux文件系统. 北京:清华大学出版社,2003:31—53
- 4 毛德操,胡希明. Linux内核源代码情景分析. 杭州:浙江大学出版社,2001:419—687
- 5 JonesM T. Access the Linux kernel using theprocfilesystem. (2006—3—14) <http://www.ibm.com/developerworks/linux/library/l/proc.html>
- 6 陈莉君,冯 锐,牛欣源. 深入理解 Linux内核 M. 北京:中国电力出版社,2001:364—412
- 7 Matthew N, Stones R. Linux程序设计. 北京:人民邮电出版社,2007:99—140
- 8 李善平,等. Linux内核指导 M. 杭州:浙江大学出版社,2002:236—277
- 9 JonesM T. Anatomy of the Linux file system. <http://www.ibm.com/developerworks/linux/library/l/linux-filesystem>. (2007—10—30)

(下转第 1724页)

神经网络控制的问题。主要从仿真试验的角度来对体操机器人进行分析和研究,仿真的结果也表明用联立约束法设计的仿真模型能够满足仿真要求,从另一个途径实现了对体操机器人数学建模,并且设计的神经网络控制器能够使机器人在不平衡位置到达很好的收敛域和较强的平衡范围,为以后研究提供了新的建模方法和研究途径。

参 考 文 献

- 1 Song M W. The Swing up control of robot manipulations IEEE Trans on the Automatic Control, 1992; 37 (11): 1782—1786
- 2 Spong M W. The Swing up control problem for the Acrobot IEEE Control System Magazine, 1995; 15 (1): 49—55
- 3 Lai X Z, She J H, Cai Z X. Fuzzy control strategy for Acrobots combining model-based control IEEE Proceedings Control Theory and Applications, 1999, 146 (6): 505—510
- 4 郑 艳,朱 媛,井元伟.一类欠驱动机械系统基于滑模变结构控制.东北大学学报,2005;26(6):511—514
- 5 郑 艳,井元伟. Acrobot系统基于滑模的离散时间变结构控制.东北大学学报,2006;27(6):591—594
- 6 赖旭芝,蔡自兴,吴 敏.一类欠驱动系统的模糊与变结构控制.自动化学报,2001;27(6):850—854
- 7 郑 艳,郑秀萍,褚俊霞,等.基于 T-S模型的体操机器人系统模糊变结构控制.控制与决策,2006;21(1):34—37
- 8 (美)约翰·F·加德纳. Simulations of Machines Using MATLAB and SMULNK 周进雄,张 陵,译.西安:西安交通大学出版社,2002:75—77

Variable Structure Control Based on Neural Network for a Class of Underactuated Mechanical System

XIE Heng, NIU Qin-zhou

(Guilin University of Technology, Guilin 541004, P. R. China)

[Abstract] Neural network control is applied to the design of balance control for Acrobot system. First, using simultaneous binding law and vector central for Acrobot system builds the mathematical model. Then developed the linear feedback theory makes its system linearized. Finally, the design of neural network balance controller for Acrobot is applied. Simulation results showed that the linear feedback theory of the neural network would keep the Acrobot system asymptotically stable in the neighborhood of up-vertical, thus enabling the variable structure control based on neural network for Acrobot system to come true.

[Key words] Acrobot simultaneous binding law neural network linear feedback theory

(上接第 1719 页)

Analysis and Implementation of File Real-time Monitoring Based on the Android Smart Phone System

WEN Min, AILi-rong*, WANG Zhi-guo

(School of Computer Science and Engineering, Northwestem Polytechnical University, Xi'an 710072, P. R. China)

[Abstract] Ensure the safety of the file-system is a key point to protect the security of system. To protect the integrity of data through the file monitoring is the base of disaster evaluation and protection of the security system. According to the research of the Android smart phone system, a file monitoring mechanism is discussed based on inode-monitor under android platform, including its key data structure, key implementations and test in detail.

[Key words] file monitor android smart phone