**密级： 保密期限：**

xm 拷贝

**硕士学位论文**



**题目： 基于网络访问数据的移动**

**APP使用模式研究**

**学 号：**

**姓 名：**

**专 业：**

**导 师：**

**学 院：**

**2016年 11月 30 日**

独创性（或创新性）声明

本人声明所呈交的论文是本人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢中所罗列的内容以外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得北京邮电大学或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

申请学位论文与资料若有不实之处，本人承担一切相关责任。

本人签名： 日期：

关于论文使用授权的说明

本人完全了解并同意北京邮电大学有关保留、使用学位论文的规定，即：北京邮电大学拥有以下关于学位论文的无偿使用权，具体包括：学校有权保留并向国家有关部门或机构送交学位论文，有权允许学位论文被查阅和借阅；学校可以公布学位论文的全部或部分内容，有权允许采用影印、缩印或其它复制手段保存、汇编学位论文，将学位论文的全部或部分内容编入有关数据库进行检索。（保密的学位论文在解密后遵守此规定）

本人签名： 日期：

导师签名： 日期：

基于网络访问数据的移动APP使用模式研究

摘 要

随着以智能手机和平板电脑为代表的移动终端设备的不断普及和更新，以及移动网络相关软件、硬件的飞速发展，互联网已经从传统的桌面互联网时代进入到了移动互联网时代。伴随着移动互联网时代的来临，人们许多相关行为或者使用的工具与桌面互联网时代相比都发生了巨大的变化。近几年，异常火爆的移动应用程序APP就是移动互联网时代的产物。

在信息化爆炸的今天，每天都有大量的 APP 上线，同时有大量的APP被淘汰。一次错误的投资或决策都将会造成巨大的损失，所以在这个信息化时代需要善于从大数据中提取到想要的信息来方便做出正确的决策。但是，要想从海量的网络访问数据中提取出APP的使用模式，首先需要获得这些APP在下载或使用等情况下所具有的特征，即客户端与服务器端进行通信时产生的流量规则。

本论文旨在研究如何自动化抓取APP下载的网络流量并自动化提取其流量规则。首先对移动APP的相关课题背景进行研究，提出了论文主要内容和研究目标。其次介绍一些Wireshark抓包的相关理论知识，并在理论知识的基础上介绍手动抓取APP下载和使用情况下的网络流量，以及流量规则文件的生成和后续使用。当APP数量不多时，可以人工抓取APP的网络流量。但是，当APP数量非常多时，就需要考虑自动化的方法，节约人力和时间成本。

再次，介绍如何实现移动APP下载操作的自动化，即编写自动化测试框架Robotium程序模拟人的手势操作实现APP在安卓应用市场中的自动化下载，包括应用商店安装包重签名，Android控件定位方式等。接着利用Tcpdump命令行工具实现APP下载流量的抓包，然后编写Python多线程程序将两者结合实现移动APP下载流量抓取自动化。最后，通过编写Python程序自动化提取流量规则，缩小人为找寻流量规则的范围，减小工作量。

通过提取APP在下载或使用等情况下产生的流量规则，进而生成规则文件，随后在海量的网络访问数据中利用这些规则文件，编写MapReduce程序挖掘出哪些用户曾经下载或者使用过该APP，从而分析得到该APP的欢迎度，最终实现APP的使用模式研究。

关键词：移动APP Robotium Tcpdump 多线程 抓包

MOBILE APP USAGE PATTERN ANALYSIS BASED ON NETWORK ACCESS DATA

ABSTRACT

With the smart phones and tablet PCs as the representative of the growing popularity of mobile devices, and the development of mobile network related software and hardware, the Internet has been changed from the traditional desktop Internet era into the mobile Internet era. With the advent of the mobile Internet era, many related behaviors or tools have undergone tremendous changes. In recent years, the unusually hot mobile apps are one of the results of the mobile Internet era.

In the information age, a large number of APPs were released, while a large number of APPs were eliminated. A wrong investment or decision-making will result in huge losses. Therefore, in this information age, we should be good at extracting the desired information from the big data, and make the right decisions based on the information. But, in order to extract the desired information from the massive network access data and obtain the APP mode of using, first of all, we need to get these characteristics of the APPs when they are downloaded or used, by generating traffic rules when the client and the server communicate.

The purpose of this paper is to study how to automatically fetch the network traffic of APP’s downloading operation and extract the traffic rules automatically. Firstly, the background of mobile APP is studied, and the main contents and research objectives are proposed. Secondly, this paper introduces some theoretical knowledge about Wireshark capturing packet and introduce the network traffic under the situation of APP’s downloading and using, as well as the generation and follow-up of the traffic rules file based on the theoretical knowledge. Since there is only a small number of APPs, we can manually capture the APP network traffic. However, when the number of APPs is increasing, we need to consider the method of automation for saving manpower and time costs.

Once again, the paper introduces how to achieve the automation of mobile APPs downloading operation, that is, the preparation of automated testing framework Robotium to simulate human gestures to achieve automation-download of APPs in the Android application market, including application store installation package re-signature, Android control positioning methods. And then use the Tcpdump command-line tool to achieve the traffic of APP downloading, and then write Python multi-threaded program with combining the above two to achieve the traffic capture automation of mobile APP downloading. Finally, we automate the extraction of traffic rules by writing Python programs to reduce the scope of human search for traffic rules and reduce the workload.

At the end, the rules file is generated by extracting the APP downloading or using in the case of the traffic rules generated, and then get the welcome of the APP, and ultimately the use of APP mode of study, by using the rules file in the massive network access data.

KEY WORDS：mobile APP Robotium Tcpdump multithreading packetcapture

目录

[第一章 绪论 1](#_Toc468130947)

[1.1 研究背景 1](#_Toc468130948)

[1.2 课题来源及创新点 3](#_Toc468130949)

[1.3 本文主要内容与章节结构 5](#_Toc468130950)

[1.4 本章小结 6](#_Toc468130951)

[第二章 Wireshark提取移动流量 7](#_Toc468130952)

[2.1 Wireshark简介 7](#_Toc468130953)

[2.2 Wireshark使用 7](#_Toc468130954)

[2.2.1 网络适配器选择 7](#_Toc468130955)

[2.2.2 捕获过滤器 8](#_Toc468130956)

[2.2.3 显示过滤器 9](#_Toc468130957)

[2.2.4 封包列表 12](#_Toc468130958)

[2.3 移动流量获取 12](#_Toc468130959)

[2.3.1 获取途径介绍 12](#_Toc468130960)

[2.3.2 获取途径选择 14](#_Toc468130961)

[2.4 数据包有效性的分析 14](#_Toc468130962)

[2.5 本章小结 14](#_Toc468130963)

[第三章 移动APP使用模式分析 15](#_Toc468130964)

[3.1 不同类别移动APP的选择 15](#_Toc468130965)

[3.2 移动APP的流量特征选择 16](#_Toc468130966)

[3.2.1 流量特征的选择思路 16](#_Toc468130967)

[3.2.2 移动APP流量中的HTTP协议研究 18](#_Toc468130968)

[3.2.3 移动APP下载动作特征提取分析 21](#_Toc468130969)

[3.2.4 移动APP使用动作特征提取分析 24](#_Toc468130970)

[3.3 流量规则文件的生成与后续使用 29](#_Toc468130971)

[3.4 本章小结 30](#_Toc468130972)

[第四章 移动APP下载操作自动化 31](#_Toc468130973)

[4.1 自动化下载移动APP的实现思路 31](#_Toc468130974)

[4.2 自动化下载移动APP的实现方案 31](#_Toc468130975)

[4.2.1 下载动作模拟思路 31](#_Toc468130976)

[4.2.2 自动化测试框架选择 31](#_Toc468130977)

[4.2.3 自动化测试框架Robotium简介 33](#_Toc468130978)

[4.2.4 adb shell简介 34](#_Toc468130979)

[4.3 基于Robotium实现移动APP自动化下载 36](#_Toc468130980)

[4.3.1 自动化下载流程设计 36](#_Toc468130981)

[4.3.2 应用商店安装包重签名 38](#_Toc468130982)

[4.3.3 控件定位方式 39](#_Toc468130983)

[4.3.4 待下载移动APP的存在判断 41](#_Toc468130984)

[4.3.5 自动化下载动作实现 42](#_Toc468130985)

[4.4 运行结果及分析 44](#_Toc468130986)

[4.4.1 控制台输出信息分析 44](#_Toc468130987)

[4.4.2 Robotium程序生成的APK安装 45](#_Toc468130988)

[4.4.3 下载的移动APP安装 45](#_Toc468130989)

[4.4.4 Android模拟器运行效果 46](#_Toc468130990)

[4.5 本章小结 46](#_Toc468130991)

[第五章 移动APP下载流量抓取及流量规则提取自动化 47](#_Toc468130992)

[5.1 自动化抓取移动APP下载流量的实现思路 47](#_Toc468130993)

[5.2 自动化抓取移动APP下载流量的实现方案 48](#_Toc468130994)

[5.2.1 下载流量抓取思路 48](#_Toc468130995)

[5.2.2 抓包工具选择 48](#_Toc468130996)

[5.2.3 Tcpdump简介 49](#_Toc468130997)

[5.3 软件实现移动APP下载流量自动化抓取 50](#_Toc468130998)

[5.3.1 自动化抓取下载流量流程设计 50](#_Toc468130999)

[5.3.2 命令行运行Robotium程序 51](#_Toc468131000)

[5.3.3 待下载移动APP的安装及卸载 52](#_Toc468131001)

[5.3.4 移动APP下载流量自动化抓取实现 52](#_Toc468131002)

[5.4 流量规则特征的自动化提取 52](#_Toc468131003)

[5.4.1 自动化提取流量规则的实现思路 52](#_Toc468131004)

[5.4.2 软件实现自动化提取流量规则 53](#_Toc468131005)

[5.5 运行结果及分析 54](#_Toc468131006)

[5.5.1 控制台输出信息分析 54](#_Toc468131007)

[5.5.2 Android模拟器运行效果 55](#_Toc468131008)

[5.5.3 问题及调试方法 55](#_Toc468131009)

[5.6 本章小结 56](#_Toc468131010)

[第六章 总结及展望 57](#_Toc468131011)

[6.1 本文总结 57](#_Toc468131012)

[6.2 展望未来 58](#_Toc468131013)

[参考文献 59](#_Toc468131014)

# 第一章 绪论

## 1.1 研究背景

随着移动互联网的普及应用，电信运营商正在面临着前所未有的机遇与挑战。作为移动数据服务提供商，电信运营商不但通过建设高质量的通信网络来确保用户体验，而且还通过业务开发完成商业模式的转型。然而，因为受到以微信为代表的OTT (Over The Top)业务的影响，运营商在加大网络建设投入的同时，正面临数据流量“增量不增收”的窘境，业务经营入不敷出。目前，行业机构一般认为，流量管理是帮助电信运营商打破两难境地的有效途径。

随着电信用户规模的不断扩大，电信DPI (Deep Packet Inspection, 深度包检测)数据规模也日益扩大。 随着互联网的迅速发展，大数据时代的到来，海量的网络流量数据的处理需求也应运而生。为了满足大规模网络流量数的据处理需求，需要针对骨干网进行有效的网络流量日志采集，然后进行有效的网络流量日志分析。为了洞察网络操作和使用，可以对网络流量日志进行多维统计分析，或者改变方式以提高网络质量[11]。深入挖掘网络流量日志能够探索用户的互联网功能和偏好，并深入了解用户需求，提高客户服务效率。

当前，信息技术飞快发展，互联网应用的全面普及，各类终端设备记录了各式各样的行为，当然，这也产生了大量的数据。根据一份由国际数据公司的调查研究的报告，预计到了2020年，全球的数据量将达到 35ZB（1ZB=10 亿 TB）[12]。大数据逐渐渗透到各个行业和领域，已成为生产中的一个重要因素，被认为是“未来新石油”。近年来，大数据技术的研究和应用的快速发展，使许多国家已经意识到大数据的重要性，并作为大力推进其发展的战略技术，迎来了大数据时代。

大数据的特征主要体现在如下几个方面[13]：

（1）数据的总量呈现出巨幅增长。同一个类型的数据正在飞快的增长，目前在网络日志、数字图书馆、电子商务、即时通讯、社会网络、地理信息导航系统、传感器网络等各类领域都已经产生了海量的数据，并且规模在不断的扩大。现如今，淘宝日常的活跃数据量已经超过了50TB，共有2亿多个注册用户和4亿多个产品信息在上面不停的活动；沃尔玛每小时就有100多万次客户交易，数据库存储量已经超过了2500万亿字节。

（2）大数据增长率呈指数增长。目前在很多领域的数据增长速度惊人， 据WinLerCorp调查显示，最大的数据仓库中的数据量每两年增加3倍，其增长速度远超摩尔定律增长速度[12]。2015年最大的数据仓库数据量已经接近100PB。大数据的强大动态特性，需要对分析和处理做出快速反应，在动态变化的环境中执行快速分析的过程，有些甚至是实时的，否则这些结果可能就是无效的、过时的。例如股票分析系统、导航的定位系统等对数据的实时处理有着较高的要求，大数据分析和处理的方法必须能快速地适应相关业务更新频率加快的需求[14]。

（3）新的数据类型和数据源正在上升。随着大数据的不断发展，数据类型也已经不再是之前的那种单一音频、视频或者文本等信息，可能还包含了例如位置信息等其他类型的新型数据。而伴随着新兴技术的不断应用，在大数据中还出现了林林总总的非结构化或半结构化的数据，大大颠覆了传统数据结构。这对传统的数据处理框架和算法是很大的挑战，必须做出改变以适应新的机遇和挑战。

（4）数据的价值日益突现。大数据好比一座富矿，需要不断地挖掘，才能获取其中丰富宝藏。通过海量的数据处理、整合分析，可以在巨大的数据中获取需要的信息、发现新知识，在原有信息的基础上提炼出更有价值、更具针对性的信息。通过这种方式，大数据带来了新的机遇与挑战。充分利用大数据的价值，可以在很大程度上提高企业的竞争力和生存力。例如用户的购买记录数据通常十分庞大，虽然绝大多数都没有实际利用价值，但是通过大量的分析整合比较，商家就可以获得用户的购买喜好，以及各类商品在不同人群中的欢迎程度，从而制定相应的商业战略。大数据分析，是披沙拣金，挖掘那些有价值的信息。

所以从上面的分析可以看出，大数据之所以大，一个是因为大量的数据以及它们增长的速率之大；另一个就是在这些大量数据中所蕴含的信息价值之大。所以，大数据之所以被称为“大数据”，是因为它的数量和价值。

在各类终端每天源源不断产生大量的数据的过程中，以手机为代表的移动终端所产生的数据增长速度越来越快，越来越多。在移动互联网没有普及的过去， 传统PC在数据产生过程中占据着主导地位，人们的关注点也都是聚焦在这些数据上面。而现在，灵活、高效、及时的移动终端正在取代PC，占据了数据产生的高峰[15]。从2000年的第一款拍照手机问世以来，手机用户的数量提高了6倍之多。虽然移动设备继续被接受和使用，但移动服务范围的扩大和对移动内容的不断增长的需求将使移动用户的增长在未来五年内成为全球人口增长的两倍。而这种4G网络发展、移动设备、智能产品、移动用户的浪潮会不断推动全球移动数据流量在接下来的五年增长近八倍。

到2020年，移动智能设备及其连接占全球移动设备和连接总数的73％，高于2015年的37％。与此同时，各类智能设备将会贡献总移动数据流量的98%。而从单个设备的角度去看的话，移动智能手机占据主导地位。这些手机产生的移动数据流量在总移动流量中的比例将从目前的76％增加到2020年的81％[16]。在2020年，将会有越来越多的人拥有智能手机，这些人数将会超越拥有汽车、电力以及自来水的人的数量。

全球移动和数据流量的增长速度一直在加快。到2020年，全球移动数据流量预计将达到每月30.6EB，而就在2015年，仅仅只有3.7EB，全球年度移动数据流量会达到惊人的366.8EB，2015年仅仅只有44.2EB。2020年预计的366.8EB的移动数据相当于2010年全球移动业务总量的120倍；其中包含81万亿张照片，这相当于每年每人每天28张照片发送出去; 7万亿个视频剪辑，这又相当于每人每天2.5个视频剪辑发送出去。

从2015年到2020年，全球移动数据流量增长率将会是全球固定IP流量的两倍。到2015年，51％的整体移动流量将被转移，到2020年，55％的移动数据将被转移，75％的全球移动数据流量将是视频[17]。到那时，互联网和移动设备将变得更加智能。将有116亿移动设备使用，其中85亿个人设备和31亿M2M网络设备，移动设备在2015年只有79亿的使用。

如果说19世纪蒸汽机的发明推动了第二次产业革命，那么现在信息技术的飞速发展将会带来21世纪信息化革命。传统产业继续与互联网结合，信息技术，自动化，智能化越来越普遍。越来越多的人通过手机上的各种类型的移动APP来方便生活、工作和娱乐。在以前，人们的交流停留在书信、电话、短信，而现在即时通信工具比如 QQ、微信的出现，让人们的交流和沟通变得更加简单和便捷；在以前，人们的饮食、购物都需要自己去实体店去进行消费，而现在，外卖、淘宝等在线消费模式的兴起，让人们足不出户也能享受到优质服务；在以前，玩游戏只能呆在电视或者电脑旁，现在拿起手机随时随地就能玩手游，放松大脑。

人们生活娱乐方式的改变，让传统的产业必须遵循时代的潮流，不断地对产业进行信息化升级，来满足在新时代中人们的需要，才能在这次产业革命中生存下来不被淘汰。人们每天操作手机所产生的众多数据，在这些大量的数据中含有用户喜好，某类人群特定需求等非常珍贵的信息。谁能提炼并且利用这些信息，在残酷的信息化产业革命中就能生存下去，成为赢家。

## 1.2 课题来源及创新点

随着以平板电脑和智能手机为代表的移动终端设备不断更新和流行，以及移动网络相关的硬件和软件的迅速发展，互联网从传统的桌面互联网时代进入移动互联时代。伴随着移动互联网时代的来临，人们的许多相关的行为和使用的工具和桌面互联网时代相比都发生了很大的变化。近年来，异常火热的移动应用APP是移动互联网时代的产物。

目前，移动APP正在以迅雷不及掩耳之势席卷全球，人们日常生活中使用互联网的方式，已经不再是桌面化的互联网，逐渐向移动化转移。与此同时，许多移动APP的设计开发技术应运而生；在应用的时候，人们重点考虑“移动优先”的策略[18]。然而，即使现在移动设备似乎已经无处不在，未来的时代，却也只是刚开始揭开帷幕。我们正面临着全新的移动终端设备，诸如可穿戴设备或者其他的众多移动配件——正是因为它们，才构成了现在的这个“万物互联”的世界。在未来几年中，这一切的一切都将影响人们的设计、开发和测试软件的方法。

移动终端设备正变得越来越重要——不只是专注于技术领域的企业重视这一点，另外一些主营传统业务的企业亦是如此。所以，在移动领域中，人们引入了网站优化与用户行为分析等新概念，其目标是更深入的挖掘用户的行为，并让移动APP更加符合用户的日常生活。在过去几年中，移动领域中不断出现了很多创业公司，他们提出了更加深入的应用分析服务[19]。然而，这一切的目的原因，是为了更进一步的掌握用户行为，并且推动移动APP 更加符合这些行为的需求，从而促使用户使用时间和移动APP留存率最大化。

在信息化爆炸的今天，每天都有大量的 APP 上线，同时有大量的APP被淘汰。每个 APP 代表的是一个想法、一个项目、一种服务，它承载着开发团队的期望也融入了他们的汗水。但是，并不是每一个 APP 都能被大家所认可，也不是每一个 APP 开发出来就注定成功。据统计，在应用市场可以占据推荐榜首 APP最长的一个月左右，大部分仅仅能出现几分钟甚至几秒钟，可见互联网竞争的激烈和残酷。一次错误的投资和决策都将会造成巨大的损失，所以在这个信息化时代需要善于从大数据中提取到想要的信息来方便做出正确的决策。

如果一个新生项目需要投资，对于投资者来说，获取的信息大部分是由项目内部的负责人提供。比如活跃用户的人数、每日APP下载量、交易额等信息。这些信息的真实性仍有待观察。而投资者又很难通过其它的途径了解到这个信息，这就需要一个第三方对于这些信息进行统计和验证，确保信息的准确性、客观性，从而保证投资者不会做出错误的决定。又比如，公司想要了解同行业其它公司的业务以及总体趋势如何，就需要在互联网上海量的资源中，找到自己需要的信息，从而了解竞争者目前的情况，为自身的发展做出正确有利的决策。

一般某个APP的日常活跃用户数，每天新用户的增长量，或者各种用户的操作细节这些信息是掌握在公司内部的，属于商业机密，是不会或者很少公开的，然而我们可以通过合法的手段在海量的网络访问数据中作为第三方获取这些有用的信息，进而可以对APP的使用模式进行研究。

当APP数量不多时，可以人工抓取APP使用模式下的网络流量。但是，当APP数量非常多时，就需要考虑自动化的方法，实现APP使用模式下的网络流量抓取，从而节约人力和时间成本。

## 1.3 本文主要内容与章节结构

用户通过移动APP在手机进行操作的同时，客户端APP和对应的服务器端通过网络来进行数据交换。在这个过程中，用户的特定行为也会产生特定的网络流量。然而不同的用户对于同一个APP的相同操作所产生的流量是具有固定相似的特征的，而这些特征是特有的，也就是其它流量中不会出现该特征。通过找到这些特征，再利用这些特征从海量的数据中筛选出符合此特征的流量，就能获取所需要的信息[20]。要想从海量的网络访问数据中提取出APP的使用模式，首先需要获得这些APP在下载或使用等情况下具备的特征，也就是当客户端和服务器通信时生成的流量规则。本课题旨在研究如何更高效的实现流量特征提取。

本文共分为六章：

第一章为绪论，主要介绍了本课题的研究背景、课题来源以及创新点。

第二章理论知识，介绍如何通过 Wireshark 来获取含有操作所产生的网络流量的集合的数据包文件。介绍Wireshark抓包的理论基础，包括Wireshark的简介和使用，移动流量的获取以及抓包后移动流量数据包的有效性分析。

第三章介绍如何基于网络访问数据实现移动APP的使用模式研究的方式，起到总括全文的作用，包括不同类别APP的选择，手动实现移动APP下载和使用情况下的流量特征的提取以及流量特征提取后生成的流量规则文件的后续使用。

第四章介绍如何实现移动APP下载操作的自动化，即在安卓应用市场中自动化下载APP，包括自动化下载移动APP的实现思路、实现方案、具体实现过程以及运行结果和分析。

第五章介绍如何实现移动APP下载流量抓取及流量规则提取自动化，即与第四章介绍的移动APP自动化下载操作结合，利用Tcpdump命令行工具实现移动APP下载流量自动化抓取，然后再实现流量规则的自动化提取。包括自动化抓取移动APP下载流量的实现思路、实现方案、移动APP下载流量自动化抓取的具体实现过程、流量规则特征的自动化提取具体实现过程以及运行结果和分析。

第六章总结全文并展望未来。

## 1.4 本章小结

本章为绪论，主要介绍了基于网络访问数据的移动APP使用模式研究的研究背景、课题来源及创新点并且简单介绍了本文的主要内容与章节结构。

# 第二章 Wireshark提取移动流量

## 2.1 Wireshark简介

Wireshark（前称Ethereal）是一款网络封包分析软件。网络分组软件的功能是拦截网络分组，并且将尽可能多地显示网络分组中的所有信息[15]。Wireshark所使用的接口是WinPCAP，它可以直接跟网卡进行数据消息交换。

这种网络分析软件的功能可以想象为“电气技术人员测量电流、电压和电阻”的工作，但是现场被移植到网络，且线路换成网络线路。在过去，网络封包分析软件是很昂贵的，但是Ethereal的出现改变了这一切，可以免费使用并且能够获取它的源代码。它是目前世界上最广泛的网络分组分析软件之一。

Wireshark的使用目的[16]：（1）网络管理员利用Wireshark检测网络问题，网络安全工程师利用Wireshark检查与信息安全相关的问题，开发人员利用Wireshark调试新协议，普通用户利用Wireshark学习网络协议。当然，有些人会“别有用心”，用它来找到一些敏感信息。（2）Wireshark不是入侵侦测系统（Intrusion Detection System，[IDS](http://baike.baidu.com/subview/34066/6727817.htm)）。Wireshark不会生成警报或提示网络上的异常流量行为。但是，仔细分析Wireshark截获的数据包可以帮助用户更清楚地了解网络的行为。Wireshark不会改变网络数据包的内容，它只是反映出当前的信息包流。Wireshark本身不会向网络发送数据包。

## 2.2 Wireshark使用

### 2.2.1 网络适配器选择

在官方网站下载并安装Wireshark后点击打开软件，如图 2-1所示。Wireshark是捕获机器上的某一个网络适配器的网络流量，当机器上有多个网络适配器的时候，需要选择其中一个。通过菜单栏中的Caputre->Interfaces选项，即可进入 Wireshark 的网络适配器的选择，如图2-2所示：

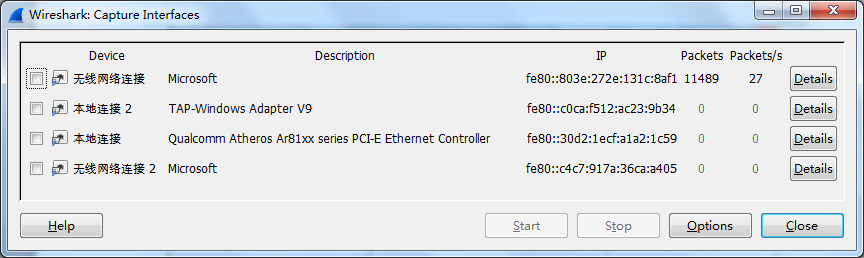
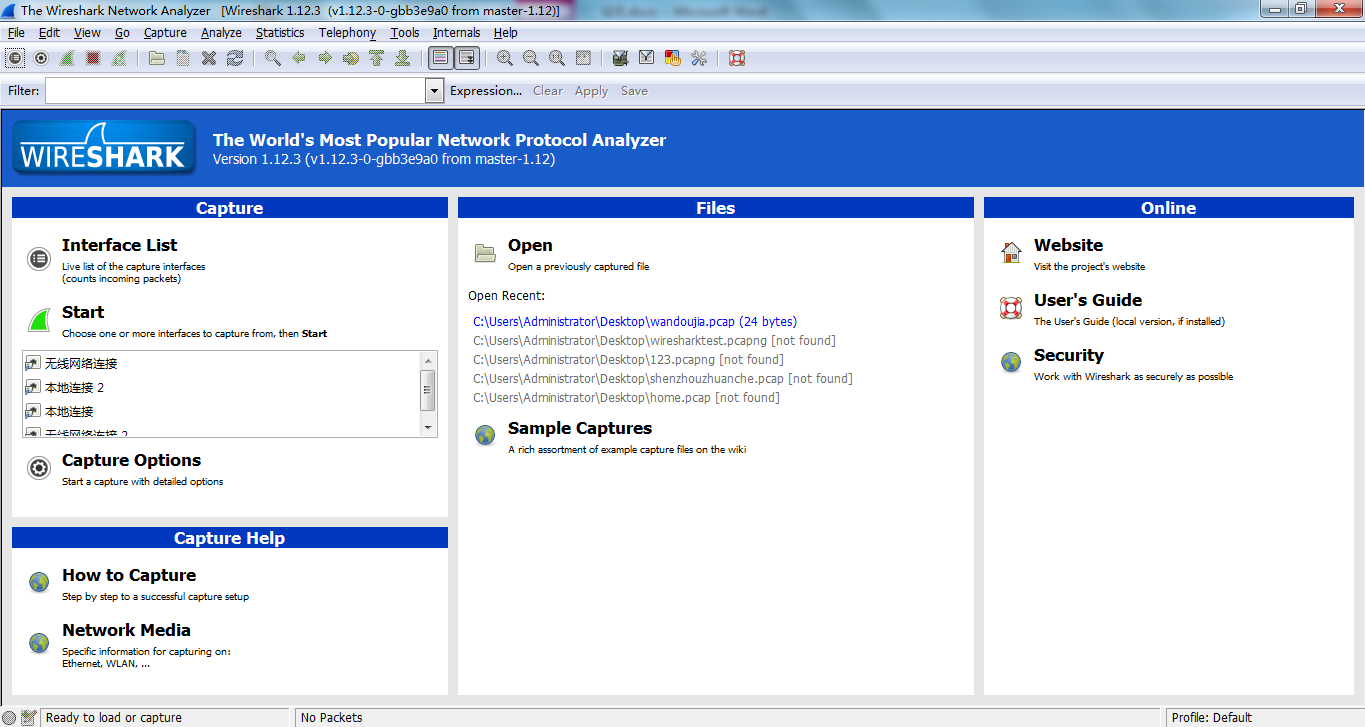
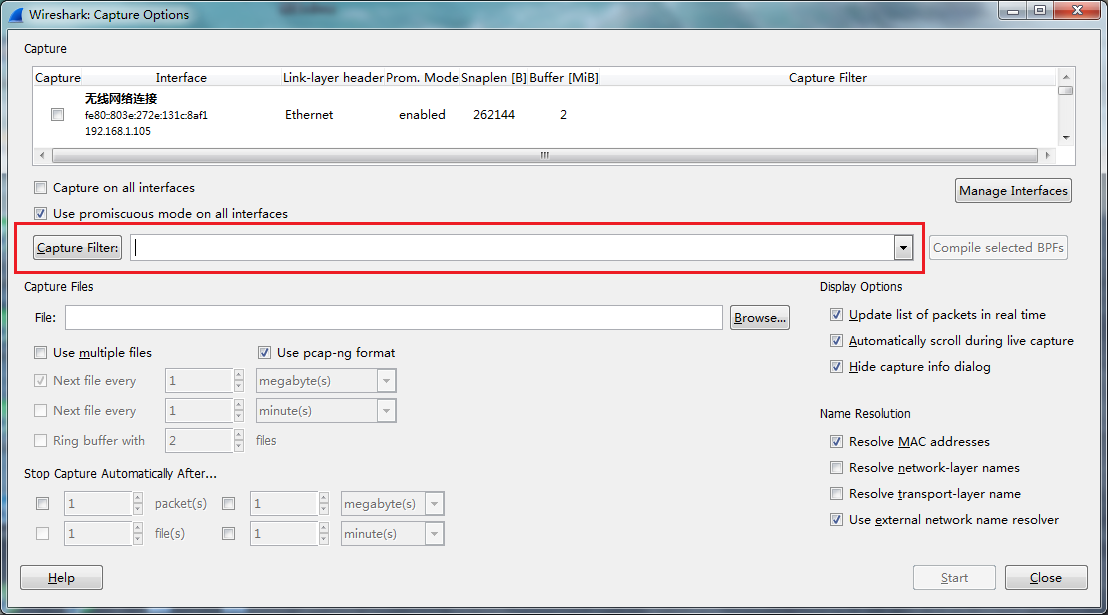
图2-1 Wireshark开始界面

图2-2 Wireshark的网络适配器选择

在这里共有四个选项，一般情况下观察Packets和Packets/s这两列是否有数字变动，如果有数字变动表示有网络流量产生，反之，则没有。

### 2.2.2 捕获过滤器

很多时候，如果捕获一段时间内的所有流量来进行分析，会造成流量数据包过大，不便于保存和处理，这个时候可以通过第一层过滤——捕获过滤器，仅仅捕获具有某种特定特征的流量。在图2-2网络适配器选择的右下方有一个Options的选项，在这里，可以输入一些限定条件，如图2-3所示。那么，当开始进行数据包抓取的时候，只有满足这些限定条件的网络流量，才能被捕获保存。

图2-3 捕获过滤器

语法[14]：Protocol Direction Host(s) Value Logical Operations Other expression。例如：tcp dst 10.1.1.1 80 and tcp dst 10.2.2.2 81。

Protocol（协议）：可能的值有: fddi 、ether、arp、 ip、decnet、rarp、lat、 moprc、sca、 tcp、mopdl、udp等。如果未指定协议，那么就默认使用所有Wireshark支持的协议。

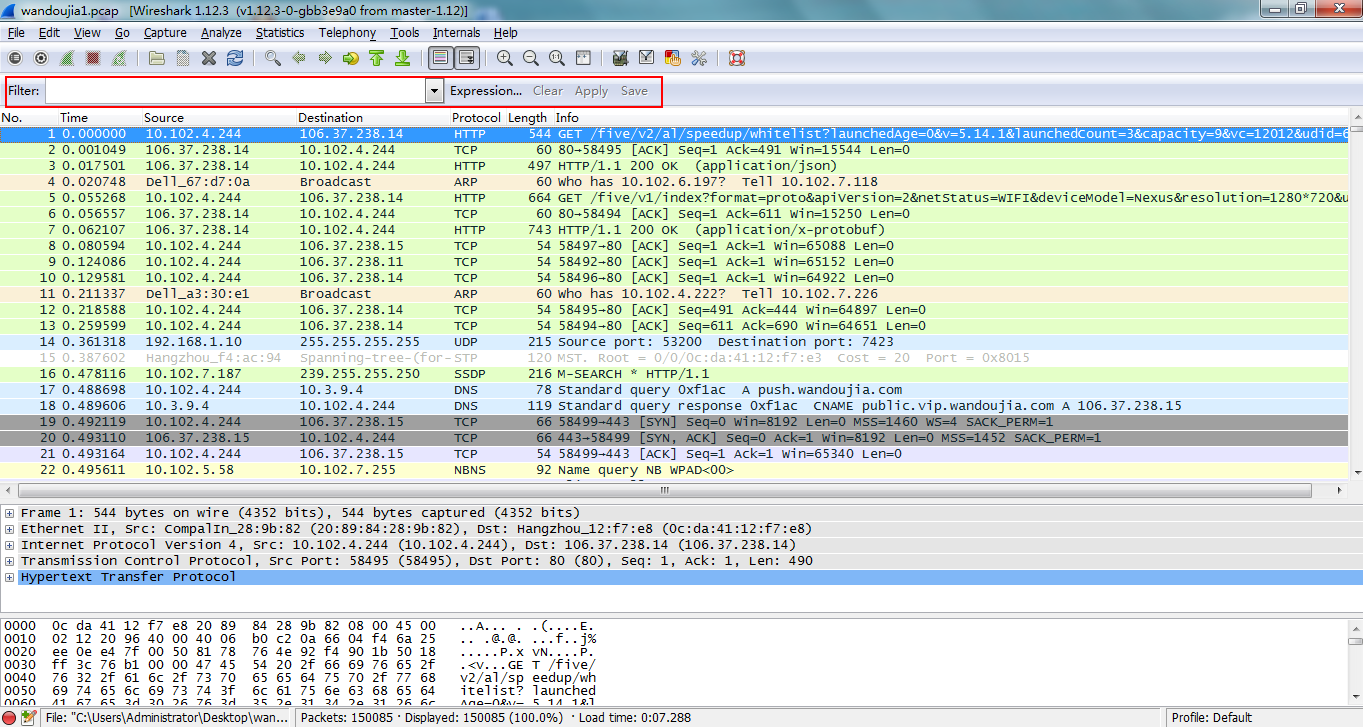
Direction（方向）：可能的值有: dst、src、src or dst、src and dst等。如果未特别的指定封包的来源或者目的地，那么就使用 "src or dst" 作为方向的默认关键字。例如，“host 11.5.7.9”与“src or dst host 11.5.7.9”是一样的。

Host(s)：可能的值有： port、net、portrange、host等。如果未特别的指定该值，那么就使用“host”作为host的默认关键字。例如，“src 10.6.8.10”与“src host 10.6.8.10”相同。

Logical Operations（逻辑运算）：可能的值有：not、and、or等。其中第一个运算符——否(“not”)具有最高的优先级。而后面两个运算符——或(“or”)和与(“and”)具有相同的优先级。运算从左至右。例如，"not tcp port 2128 and tcp port 25"与"(not tcp port 2128) and tcp port 25"相同。"not tcp port 2128 and tcp port 25"与"not (tcp port 2128 and tcp port 25)"不相同。

### 2.2.3 显示过滤器

在捕获了流量后，虽然经过捕获过滤器得到了数据，但是这些数据还是非常复杂。如果要找到一些或者一类具体的流量，仍然需要再进行一次过滤。这时候利用显示过滤器，可以更轻松、更方便的找到所需的数据包。它比捕获过滤器的功能更强大，并且如果需要修改其中的一些过滤条件，这并不需要重新捕获数据包。如图2-4所示红色标记框内，就是Wireshark中的显示过滤器的位置。

图2-4 显示过滤器

语法[14]：Protocol.String1.String2 Comparison operator Value Logical operations Other expression。例如：ftp passive ip == 11.2.3.4 xor icmp.type。

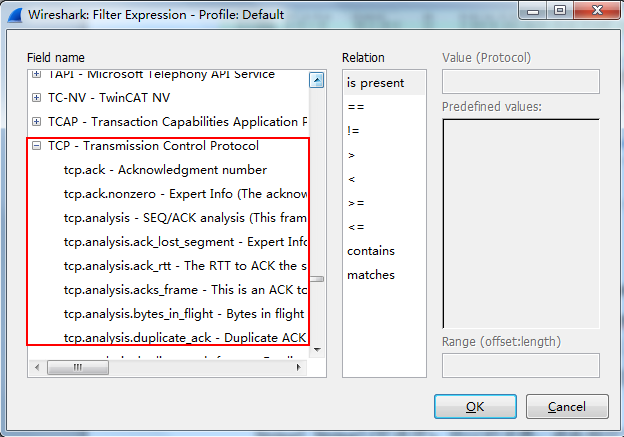
Protocol（协议）：可以使用位于OSI七层模型中第2层至第7层的任何一个协议。点击图2-4红色标记框内的"Expression..."按钮后，就可以看到这些协议名。比如：IP、TCP、DNS、SSH等。如图2-5所示。

图2-5 Wireshark的显示过滤器Expression

String1, String2 (可选项)：协议的子类。点击相应父类旁边的“+”号，然后选择它的子类。如图2-5中红框所示。

Comparison operators（比较运算符）：可以使用6种比较运算符，如表2-1所示：

表2-1 比较运算符

|  |  |  |
| --- | --- | --- |
| 英文语法 | 表示方法 | 含义 |
| eq | == | 等于 |
| ne | != | 不等于 |
| gt | > | 大于 |
| lt | < | 小于 |
| ge | >= | 大于等于 |
| le | <= | 小于等于 |

Logical expressions（逻辑运算符）：如表2-2所示，其中逻辑异或与被人们熟知的逻辑异或不相同，是一种排除性的“或”。也就是说，当它在过滤器的两个过滤条件之间使用时，这样的结果仅当满足条件之一时才显示在屏幕上。例如：“tcp.dstport 81 xor tcp.srcport 80”，只有当目的TCP端口为81或者来源于端口80（但又不能同时满足这两点）时，这样的流量才会被过滤并显示出来。

表2-2 逻辑运算符

|  |  |  |
| --- | --- | --- |
| 英文写法 | 表示方法 | 含义 |
| and | && | 逻辑与 |
| or | || | 逻辑或 |
| xor | ^^ | 逻辑异或 |
| not | ! | 逻辑非 |

显示过滤器和捕获过滤器具有不一样的语法规则，由于所有的APP使用模式抓取的网络流量数据包都是基于http协议研究的，所以这里介绍的规则都用http做举例，当然其它协议也是类似的。

1.协议的直接过滤

通过在显示过滤器中输入http协议名称，可以从数据包中过滤相应协议的流量。

2.协议属性的比较过滤

在过滤窗口中输入“http.”，Wireshark便自动提示http的一系列属性，例如 http.accept，这表示过滤出http协议中，含有accept头的网络流量。例如：http.accept ==“XXX”，这就意味着XXX是accept头中可能出现的字符串。

3.组合表达式的过滤

这里需要用到逻辑运算符：&&（逻辑与）、||（逻辑或）、^^（逻辑与或）、！（逻辑非）。例如：http.request.uri contains “Google.com” ，它的功能是过滤出http协议下的请求包中的URL中含有 Google.com的流量。

### 2.2.4 封包列表

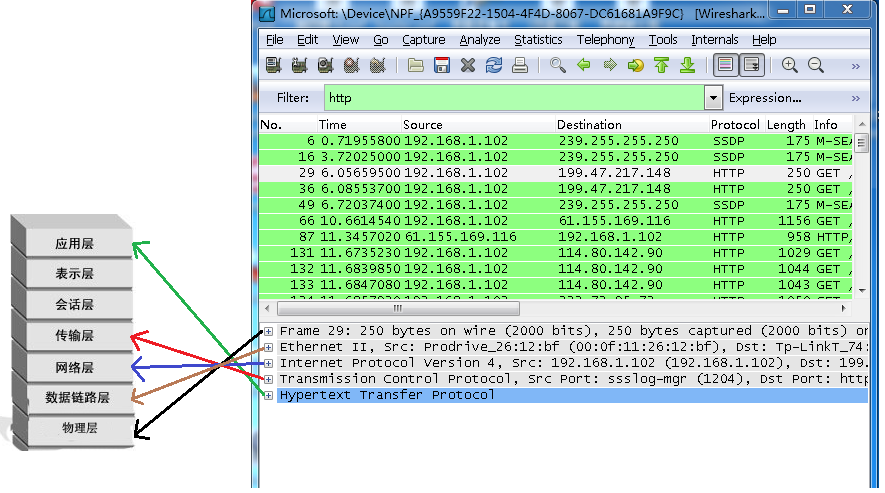
封包列表的面板中显示，编号、时间戳、源地址、目标地址、协议、长度、以及封包详细信息。可以看到不同的协议使用不同的颜色来标示。封包详细信息是最重要的，用来查看协议中的每一个字段。各行信息分别是：Frame：物理层的数据帧概况；Ethernet II：数据链路层以太网帧头部信息；Internet Protocol Version 4：互联网层IP包头部信息；Transmission Control Protocol：传输层T的数据段头部信息，此处是TCP；Hypertext Transfer Protocol：应用层的信息，此处是HTTP协议，如图2-6所示。

图2-6 封包详细信息

## 2.3 移动流量获取

### 2.3.1 获取途径介绍

Wireshark截获的是PC端的网络流量，那么如何截获移动终端生成的网络流量呢？下面提供了两种途径来进行移动网络流量的截获。

途径一：一台可以上网的PC（含有无线网卡），通过辅助软件开通wifi，把这个PC作为一个热点，手机再连入PC所开的wifi，那么该手机就通过PC端的网络上网，在期间所产生的流量也就能在PC端通过Wireshark捕获。这个辅助软件就是为了实现把一台能够联网的PC利用无线网卡，转换成wifi无线信号发送出去，确保周边的移动终端能够直接连接使用。它仅仅是一款无线路由器共享软件，并不是真正的无线路由器硬件，它的使用需要完全依赖PC网卡。该途径的原理图如图2-5所示：



图2-5 途径一原理图

途径二：在电脑内安装模拟器，通过Wireshark抓取模拟器中产生的网络流量， 从而达到移动数据抓取的目的。Android提供了一个模拟器用来模拟ARM核的移动终端设备。Android仿真器是基于QEMU开发的，而QEMU是一个著名的开源虚拟机项目，它可以提供一个虚拟的ARM移动设备。在手机上进行的各种操作，在模拟器上也基本上可以完成，并且所产生的流量信息和手机端产生的流量信息是一模一样的。但是对于苹果手机的ios系统，并没有相应的模拟器，所有ios系统终端的用户所产生的流量，不能通过这种途径进行获取。该途径的原理图如图2-6所示：



图2-6 途径二原理图

### 2.3.2 获取途径选择

一般情况下，综合使用途径一和途径二，通过Wireshark抓包工具，可以成功地抓取不同系统、不同移动APP、同一APP的不同操作等场景下的各种移动数据流量，为进一步分析研究提供帮助。

## 2.4 数据包有效性的分析

通过途径一获取的数据包，实质上就是手机通过PC上网，从而和服务器端交互所产生的网络流量数据包。PC 在这个过程中并不会让其所产生的数据包失真，只是起到了一个媒介的作用。所以通过途径一所截获的数据包是有效的。

途径二是通过安卓模拟器来进行手机模拟操作，从而在PC端直接获取手机移动流量。从外界来看，安卓模拟器就是一台普通的安卓手机，在PC上模拟出安卓手机运行环境，也能下载应用，并且对应用进行相应的操作，产生网络流量。所以通过途径二所截获的流量包也是有效的。

## 2.5 本章小结

本章介绍了如何通过Wireshark来获取含有操作所产生的网络流量的集合的数据包文件。介绍了Wireshark抓包的理论知识，包括Wireshark的简介和使用，移动流量的获取途径以及抓包后移动流量数据包的有效性分析，是全文的一个理论基础。

# 第三章 移动APP使用模式分析

## 3.1 不同类别移动APP的选择

移动应用服务，是为手机等移动终端设备连接到互联网业务或无线网卡服务而开发的应用服务。随着移动智能终端的广泛普及应用，移动终端正向功能定制化、多模化、平台开放化、增强化的方向发展，然而移动终端营销——作为新的发展渠道，正在逐渐崭露头角。但APP的同质化情况继续加剧，无论是在处于过渡期的传统企业，还是移动互联网初创企业，为了避免开发出移动APP淹没在APP数量是亿万级的各种各样的应用市场中，对于他们来说最明智的选择就是设计开发出一款具有很大的投资价值潜力的APP。

随着移动互联网的飞速发展，独立APP的门槛变得更高了，对开发团队的各方面要求也都相应地提高了。除了为企业打造企业专属的APP，还可以为企业或者个人打造行业APP，并且每个行业APP都可以作为一个该行业专属的行业信息汇聚平台。在积累了一定量的用户数并且推广普及到一定的高度后，凭借着每个APP所具有的特定的内容资源优势，自然就会吸引很多投资者们的目光，所以就需要研究APP的使用模式，并以此来判断该APP或者该类APP是否值得投资方进行投资。但是对于要选择分析研究的APP类别，以及不同类别的APP的选择，仍然需要考虑其他很多方面的东西。

对APP的外部行为可以从以下维度进行研究：（1）下载安装量统计，指标包括：下载量、下载渠道、更新量等。（2）用户活跃度统计，指标包括：用户数、用户人口属性、终端特征、使用次数、使用时长、特定页面访问次数等。（3）统计视角，指标包括：时间粒度，例如：日、周、月等；变化趋势；按公司资产（同属一家公司的APP组合在一起）、行业领域等。

而对APP的内部行为可以从以下维度进行研究：（1）搜索行为，指标包括：关键词、标签等。（2）浏览行为，指标包括：特定的网页访问，例如：产品页、推广页等；网页跳转等；关联APP跳转，例如：支付类APP等；产品、订单、成交等。（3）人口属性、用户标签、终端特征等。（4）统计视角，指标包括：时间、分布、趋势等。

同时，还需要具体问题具体分析，比如：（1）对于那些用户量很小的APP，分析的目的是从大量的此类APP里面，发现下载量、用户量、活跃度增长比较迅猛的有潜力的APP作为投资方的参考，此时对时间粒度除了日、周、月外，要求更细。（2）对于电商购物类的APP，就需要研究用户、商品、订单等基本信息；APP中的某一款或者某一类商品的站内搜索；站内页面跳转、重点页面重点商品的浏览、下单支付等情况；网站上面的数据，例如：商品价格、销量等也很有价值。（3）低于视频类的APP，需要分析用户人口属性，特征设备属性，查询方式和查询量：移动端和PC端需要单独分析研究等。

通过调研三大运营商移动和宽带用户数，参阅一些咨询公司有关APP的使用调查报告、某应用市场上APP的下载排行榜以及项目的需求，选择以下四类：（1）海淘类：小红书、洋码头、达令全球好货、考拉海购、识货、美丽说HIGO、蜜淘全球购；（2）母婴类：贝贝母婴、蜜芽、母婴之家、红孩子母婴、母爱街；（3）出行类：滴滴出行、易到用车、Uber、神州专车、一号专车、嘀嗒拼车、快的打车；（4）女性健康类：大姨吗、美柚、月来越好、姨妈来了经期助手，共24个不同类别的APP。

## 3.2 移动APP的流量特征选择

### 3.2.1 流量特征的选择思路

通过2.3节两种移动流量的获取途径可以抓包得到移动APP的网络流量生成数据包文件，那么看着繁杂的数据包如何选择出所需要的流量特征呢，这又是一个至关重要的问题。

对于获取的数据包，需要从中提取出一些特征来唯一标识某个 APP 的某个动 作。数据包中的其它数据对于这个对APP的唯一标识是无用且冗余的，所以可以把它们忽略，从而把焦点放在少数可以精确标识某APP 的特征上。

选择过程可以分为两个部分：第一个部分就是定位到特定的APP，抓取到的数据包可能不仅仅包含该APP的网络流量，可能还包含其他的网络流量，例如：PC上面其他软件产生的流量等；第二部分就是定位到APP下载或者使用等具体操作。对于特定APP的定位，可以使用host来作为一个标识，因为对于一个固定的 APP，虽然它的host可能有多个，但是它的host在比较长的一段时间是不会发生变化的，host也就是服务器的名字，通过服务器名字来定位某个特定的APP是合理的且有效的。那么，对于APP下载或者使用等操作就可以利用抓包后的数据包中的URL中某部分特征来进行唯一标识。利用URL是因为这是不同的操作对应着不同的URL请求，而这些URL中会有一部分特征能够体现出具体请求的意义。所以，通过URL的部分特征来标识APP下载或者使用等具体操作，这也是合理且有效的。那么下面分下载和使用这两种情况介绍移动APP流量特征的选择思路。

3.2.1.1 移动APP下载动作流量特征的选择思路

下载动作不是在相应的APP内部进行的，而是在某个应用商店内完成的。一般操作过程是打开某个应用商店，点击搜索框输入要下载的APP名称再搜索，在搜索出的结果中选择对应的APP进行下载安装，具体操作流程图如图3-1所示：



图3-1 APP下载动作流量特征的选择思路

3.2.1.2 移动APP使用动作流量特征的选择思路

使用动作就是在相应的APP内部进行的，例如：打开APP首页、注册、搜索等一般使用情况。为了避免使用APP时间过长，抓取过多无用的非此APP的网络流量造成数据包的冗余，所以将使用动作分解化，一个使用动作抓取一次网络流量，具体操作流程图如图3-2所示：



图3-2 APP使用动作流量特征的选择思路

### 3.2.2 移动APP流量中的HTTP协议研究

3.2.2.1 HTTP简介

HTTP（HyperText Transport Protocol）是超文本传输协议的缩写，它用于传送WWW方式的数据，用于将超文本从WWW服务器传送到本地浏览器的传输协议，是客户端浏览器或者其他程序同Web服务器之间进行通信的应用层协议[20]。超文本信息存储在因特网上的Web服务器上，而在客户端，用户需要通过HTTP协议与服务器进行通信，传输用户想要访问的超文本信息。HTTP是由命令和传输信息这两部分组成的，不仅仅用于Web访问，还可以用于其他Internet / Intranet应用程序之间的通信，以实现各种应用程序资源，超媒体访问集成。它可以让浏览器更加高效的运行，从而让网络之间的传输流量减少。它不但能够保障计算机以正确快速的传送方式传送超文本文件，还能够确定传送文件的哪个部分并且首先显示出哪个部分等。HTTP协议使用请求/响应模型[18]。 客户端向服务器发送包含所请求的方法，URL，协议版本以及包含请求修改符，客户端信息和内容的类似MIME的消息结构的请求。服务器用包括消息协议的版本，成功或错误代码加上服务器信息，实体元信息以及可能的实体内容的状态行进行响应。

通常，HTTP消息由两部分组成，第一部分是客户端到服务器端的请求消息，第二部分是服务器端到客户端的响应消息。这两种类型的消息一般包括四部分：起始行，一个或者多个头字段，指示头字段结束的空行和可选消息体。并且HTTP头字段由通用头，请求头，响应头和实体头四部分组成。每个头由域名，冒号和字段值组成。域名不区分大小写。在字段值之前可以添加任意数量的空格。标题字段可以扩展为多行。但是，在每行的开始，至少需要使用一个空格或制表符。

HTTP协议的一些主要特性总结如下：1.支持客户端/服务器模式。2.简单快捷：由于HTTP协议相对简单，使服务器的规模小，从而通信速度很快。3.灵活性：HTTP允许传输任何类型的数据对象。4.无连接：限制每个连接，只处理一个请求。5.无状态：HTTP协议对事务处理无记忆能力。

3.2.2.2 HTTP请求流程简介

首先，客户端和服务器需要建立连接。只需点击一个超链接，HTTP工作开始。在建立连接之后，客户端以URL，协议版本号，随后是MIME消息的形式向服务器端发送请求。其中MIME是由请求修改符、客户端信息以及可能的内容组成的。在接收到请求之后，服务器以状态行的形式给出相应的响应信息，响应信息是由消息的协议版本号，成功或不正确的代码，随后是包括服务器信息，实体信息和可能内容的MIME信息组成的[19]。客户端从服务器接收的信息通过浏览器显示在用户的显示器上，并且客户端与服务器断开连接。 如果在上述步骤之一中发生错误，则错误消息返回到客户端并从显示器输出。 对于用户，这个客户端与服务器端进行通信的过程是由HTTP本身完成，用户不需要理解其中的细节，只需要用鼠标点击某一个超链接，然后等待通信完成后显示在浏览器上面的信息即可。

许多HTTP通信流量是由用户代理初始化的，目的是对源服务器上的某个资源的请求。最简单的情况下，可以通过用户代理和服务器之间的单独连接实现。在互联网上，HTTP通讯是建立在TCP/IP连接之上。默认缺省端口号是TCP 80，但是其它的端口号也是可以用的。但这并不表示HTTP协议在互联网上或其他网络中完成其他协议。 HTTP只是代表一次可靠的传输。

本课题研究的是APP网络流量特征分析，所以主要关注移动端上的APP和服务器间是如何通过HTTP协议来进行数据的传输的。HTTP的请求流程包含如下几个流程：

步骤一：APP 向服务器发送了 HTTP 请求，这个请求包括了方法、路径、域名、以及其它相关的 Header。其中，方法包含了GET和POST，GET仅请求资源，而POST会附带一些用户的数据。域名由host的头决定，例如： www.baidu.com。

步骤二：服务器返回一个HTTP的响应，该响应包括：响应代码、响应类型以及其它相关的Header。通常，服务器的HTTP响应会携带一些内容，也就是有一个Body包含响应的内容。

步骤三：如果APP还需要继续向服务器请求其它的资源，例如视频等，那就再次发出HTTP请求，重复步骤 1、2。

正如之前在HTTP协议中所描述的，HTTP请求仅处理一个资源。这样大大简化了开发，当在编写一个页面的时候，只需要在HTTP请求中把 HTML 发出去，并不需要考虑是如何附带图片、视频等。倘若 APP 还需要请求图片和视频，它会发送另一个 HTTP 请求。

3.2.2.3 移动APP流量中的URL格式分析

URL（统一资源定位符），是网络上资源的访问地址。网络上的每一个资源，都有一个唯一的URL与之对应，其中包含了相关文件位置的基本信息以及浏览器应该以什么方式对其进行处理。

基本URL包含协议（或称模式）、服务器名称（或IP地址）、路径和文件名，一般语法格式为：(带方括号[]的为可选项)：protocol :// hostname[:port] / path / [;parameters][?query]#fragment。完整的、带有授权部分的普通统一资源标志符语法看上去如下：协议://用户名:密码@子域名.域名.顶级域名:端口号/目录/文件名.文件后缀?参数=值#标志。

protocol（协议）：指定其使用的传输协议名称。最常用的协议是超文本传输协议，即HTTP协议。其他协议介绍如下：https——用安全套接字层传送的超文本传输协议，ftp——文件传输协议，mailto——电子邮件地址，通过 SMTP 访问，ldap——轻型目录访问协议搜索，file——当地电脑或网上分享的文件，news——Usenet新闻组，gopher——Gopher协议，telnet——Telnet协议，mms——通过支持[流媒体](http://baike.baidu.com/view/794.htm)协议的播放该资源，ed2k ——通过支持ed2k（专用下载链接）协议的P2P软件访问该资源，Flashget——通过支持Flashget:（专用下载链接）协议的P2P软件访问该资源，thunder——通过支持thunder（专用下载链接）协议的P2P软件访问该资源。

hostname（主机名）：是指托管资源的服务器的域名系统（DNS）主机名或者IP地址。有时，在主机名前也可以包含连接到服务器所需的用户名和密码，格式为username:password@hostname。

端口（端口号）：整数和可选，当使用程序的默认端口时省略，各种传输协议都有默认端口号，例如默认端口号为http 80。如果省略，则使用默认端口号。有时为了安全或其他考虑，您可以重新定义服务器上的端口，即使用非标准端口号，这时，URL就不能省略端口号。

path（路径）：由零个或多个“/”符号分隔的字符串，通常用于表示主机上的目录或文件地址。

parameters（参数）：这是一个用来指定特殊参数的可选择项。

Query（查询）：可选，对于动态网页（如使用CGI，ISAPI，PHP / JSP / ASP 、 NET等）传递参数，可以有多个参数，每个参数用“&”符号分隔开，并且每个参数的名称和值由“=”号分隔。

Fragment：指定网络资源中的片段的字符串。 例如，一个具有多个名词的页面，可以直接使用片段对某个术语进行解释。

总体来说，移动APP流量中的URL格式分为两部分，第一部分是协议名，它告诉浏览器应该以什么方式处理即将打开的文件；第二部分是文件所在的服务器的名称或IP地址，后跟资源存储的路径以及资源的名称。服务器的名称或IP地址有时候后面跟着冒号和端口号。它还可以其他的与服务器进行通信所需的各方面信息，例如用户登录名、密码、验证码等。路径部分不同层级的路径可以用斜杠（/）进行分隔。查询部分通常用于传达动态查询服务器上的数据库所需的参数。有时，URL以斜杠“/”结尾，没有文件名，在这种情况下，URL指的是路径中最后一个目录中的默认文件（通常是主页），通常称为index.html或default.htm。

### 3.2.3 移动APP下载动作特征提取分析

3.2.3.1 移动APP手动下载

下载动作不是在相应的APP内部进行的，而是在各个应用商店内完成的。下载动作的特征需要抓取主流应用商店的所有下载动作，因为在不同的应用商店中进行相同的操作所产生的流量也都是不一样的，所以如果想抓取某个APP的下载动作特征，就必须将目前主流应用市场中对应的特征都抓取出来，才能保证最后在海量网络访问数据里面匹配的结果和实际情况尽可能接近而不至于出现较大的误差。在2.3.1小节中介绍了两种移动流量的获取途径，现在利用第二种途径抓取移动APP下载的动作特征，下面应用商店以豌豆荚为例，移动APP以小红书为例，进行下载动作特征提取分析。

首先打开安卓模拟器和Wireshark抓包工具，在模拟器上安装豌豆荚，再打开Wireshark进行抓包，最后手动下载小红书并且下载动作完成后结束抓包，保存生成的数据包。

3.2.3.2 移动APP下载URL的提取

数据包生成后，如图3-3所示，观察各个封包信息，发现每个HTTP协议前面都有三个TCP协议，这就是TCP的三次握手的三个数据包，如图3-4所示为TCP建立连接的三次握手的过程，第四个包才是HTTP协议的封包， 这说明HTTP协议确实是使用TCP协议来建立连接的。

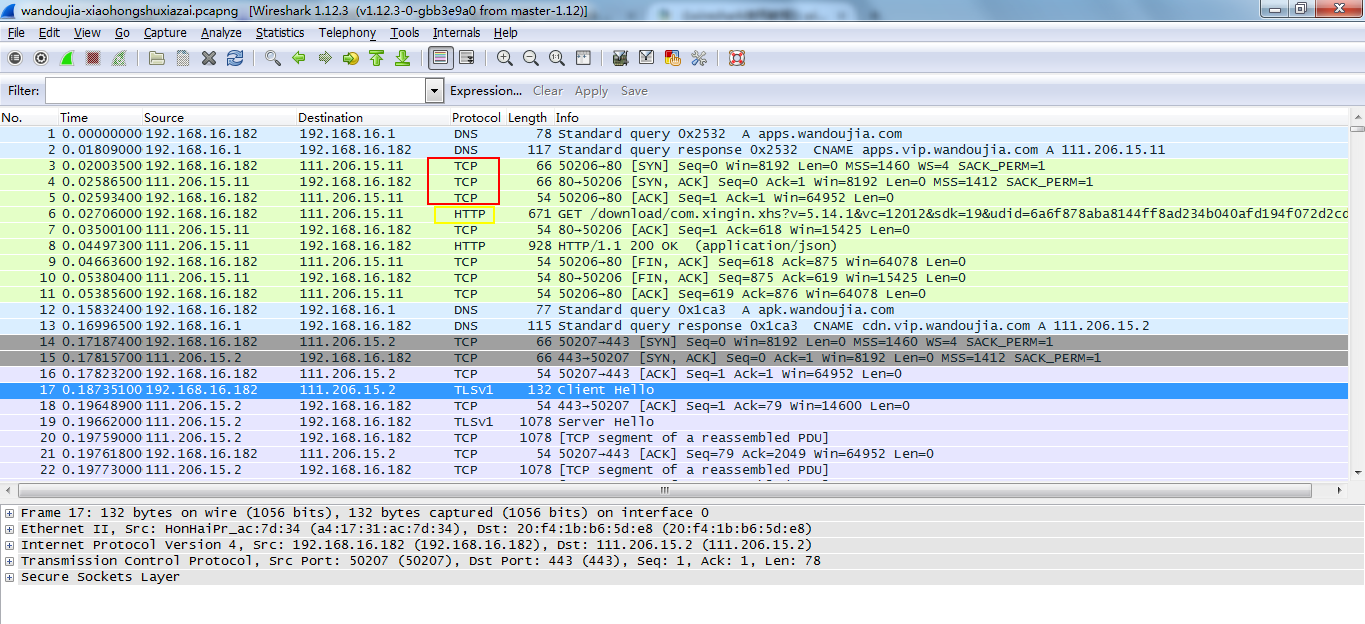
图3-3 每个HTTP协议前面都有三个TCP协议



图3-4 TCP三次握手

TCP三次握手通信过程如下：（1）建立连接，客户端向服务器发送SYN数据包并进入SYN\_SENT状态，等待服务器确认；SYN：同步序列编号（Synchronize Sequence Numbers）；（2）服务器收到SYN包，必须确认客户的SYN，同时自己也发送一个SYN包（syn=k），即SYN+ACK包，此时服务器进入SYN\_RECV状态；（3）客户端收到服务器的SYN+ACK包，向服务器发送确认包ACK，此包发送完毕，客户端和服务器进入ESTABLISHED（TCP连接成功）状态，完成三次握手。

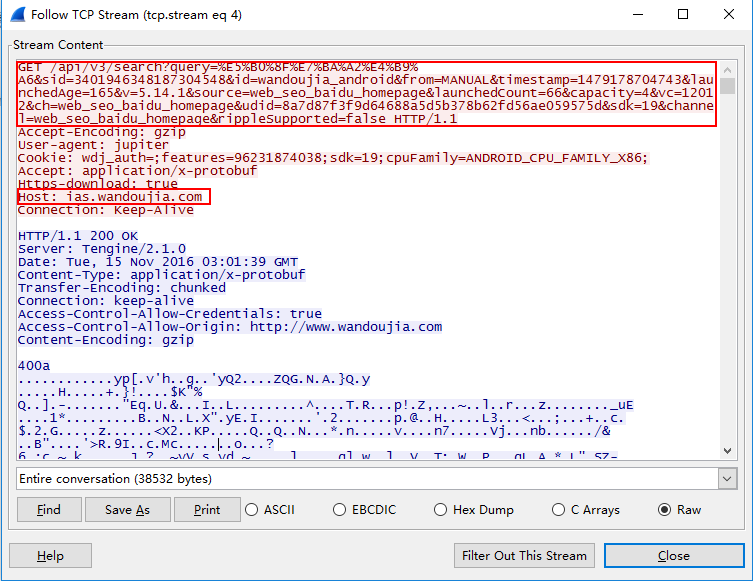
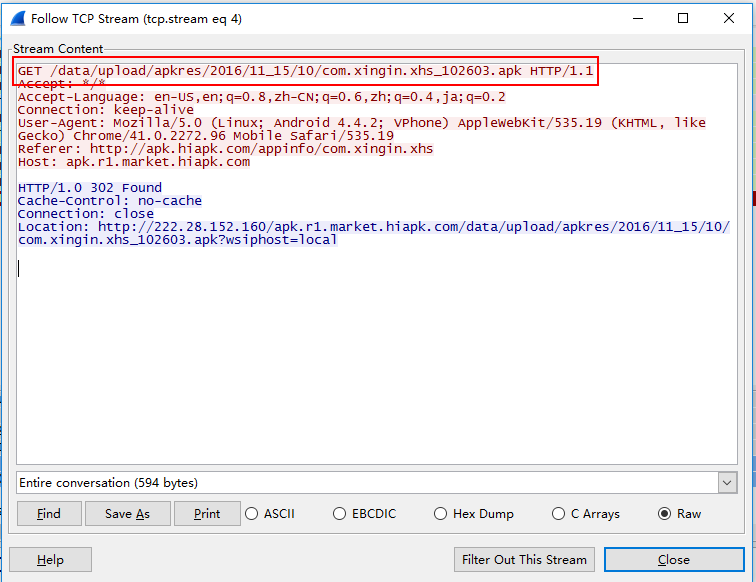
然后，利用Wireshark的过滤器过滤出http协议的报文，并依次进行相关分析，找到小红书下载动作的URL并提取出下载动作的特征。如图3-5所示为豌豆荚内小红书的搜索结果，第二个红框内发现host是“ias.wandoujia.com”，确定应该是豌豆荚的服务器，并且由第一个红框内query后面的UTF-8编码“%E5%B0%8F%E7%BA%A2%E4%B9%A6”，经过URL解码后的内容是“小红书”，这正是豌豆荚内搜索的特征；如图3-6所示为小红书下载URI特征，第一个红框内“com.xingin.xhs\_102603.apk”为下载下来的APP文件名。

图3-5 豌豆荚内小红书的搜索结果

图3-6 小红书下载的特征

3.2.3.3 问题分析及解决思路

市场上应用商店很多，在本课题研究项目上决定研究以下14个应用商店。按照类别分为，安卓：91手机助手、安卓市场、百度手机助手、豌豆荚、应用宝、机锋市场、应用汇、360手机助手、搜狗手机助手、小米应用商店、沃商店、MM商场、天翼空间；IOS：App Store。其中，91手机助手和安卓市场被百度收购，与百度手机助手的特征相同。

本课题需要研究不同应用商店下的不同APP的下载动作特征，应用商店已经有很多个，并且APP数量也很多，那么就需要考虑人力、物力成本，所以就需要思考如何通过自动化的方式节约成本。

手动下载APP，一般思路是点击搜索框搜索相应APP，然后在搜索结果中查找对应的APP，最后点击下载安装。那么，既然是人的动作，就可以想办法自动化的模拟手工操作，这样实现自动化。

综上，在提取特征时，当应用商店或者APP数量很少的情况下，可以人工进行数据包的抓取、分析以及特征提取。但是当工作量很大时，再利用人工就不现实，必须寻找自动化的途径来模拟手动操作解决问题，实现操作的自动化。

### 3.2.4 移动APP使用动作特征提取分析

3.2.4.1 使用动作的具体操作模拟

APP使用动作特征的提取，需要对APP进行像平常应用一样的操作，不同的APP情况不同，需要具体问题具体分析来完成对APP内部使用动作的流量特征提取分析。一般情况下，APP的内部使用动作的具体操作包括：注册、搜索、打开APP首页、打开APP内部其他页面、交易等常用动作。

对于用户注册操作，其实是由几个连贯的动作组成的，一般思路是输入手机号或者邮箱获得验证码，再使用验证码点击“注册”按钮完成注册。所以，就应该抓取最后一个动作——点击“注册”按钮的网络流量。

对于搜索操作，也是由几个连贯的动作组合而成的，一般思路是点击搜索框，输入要搜索的内容，最后点击搜索按钮。所以，就应该抓取最后一个动作——点击搜索按钮的网络流量。

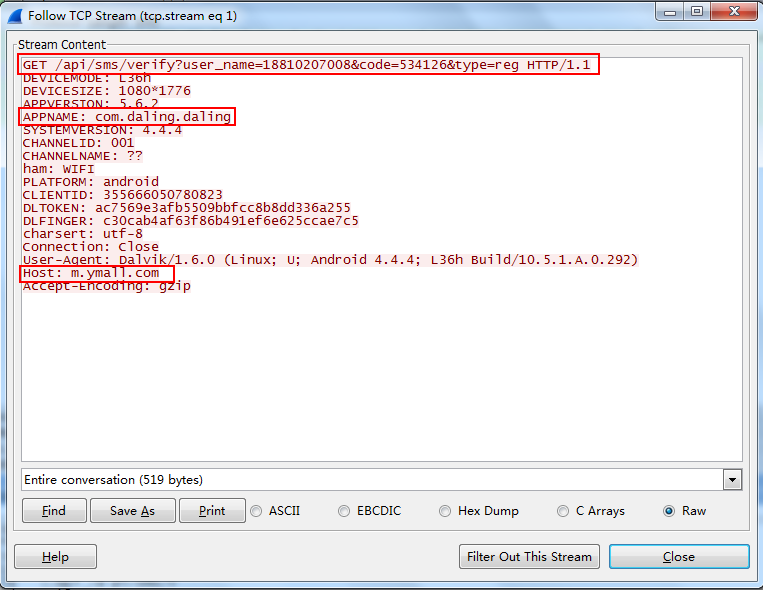
对于打开APP首页操作，就是在某款APP未处于打开使用状态下，直接打开该APP所产生的网络流量。

对于打开APP内部其他页面操作，即在使用该APP过程中，不同页面之间的跳转，例如：由首页点击登录跳转到登录界面。所以，需要抓取的是点击其他页面这个动作的网络流量。

对于交易操作，就是一些购物类APP需要提交订单然后付款完成交易的一系列操作过程的组合，所以，需要抓取的是最后一个交易完成动作的网络流量，并不是提交订单动作的网络流量。

3.2.4.2 移动APP使用中URL的提取

在本课题中选择“达令全球好货”这款APP进行使用动作的特征提取分析。首先是注册操作，抓取最后一个动作——点击“注册”按钮的网络流量后生成数据包，利用Wireshark的过滤器过滤出http协议的报文，并进行相关分析，找到达令全球好货注册动作的URL并提取出注册动作的特征。如图3-7所示为达令全球好货注册动作的特征：

图3-7 达令全球好货注册动作的特征

从图中第二个红框APPNAME可以得到操作的APP就是达令全球好货；由第三个红框可以得到达令全球好货的host是“m.ymall.com”；第一个红框内user\_name后面的“18810207008”为注册用的手机号，code后面的“534126”为注册用的验证码，所以可以断定此报文就是达令全球好货注册动作的流量特征报文。

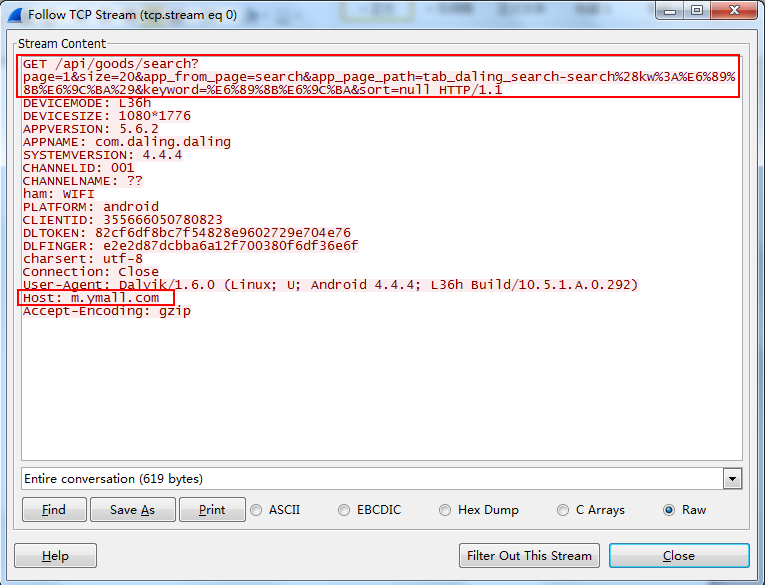
其次是搜索操作，在这里搜索的内容是“手机”，抓取最后一个动作——点击搜索按钮的网络流量后生成数据包，利用Wireshark的过滤器过滤出http协议的报文，并进行相关分析，找到达令全球好货搜索动作的URL并提取出搜索动作的特征。如图3-8所示为达令全球好货搜索动作的特征：

图3-8 达令全球好货搜索动作的特征

从图中第二个红框的HOST可以看出该APP就是达令全球好货；再由第一个红框keyword后面的内容“%E6%89%8B%E6%9C%BA”解码后就是“手机”，所以可以断定此报文就是达令全球好货搜索动作的流量特征报文。

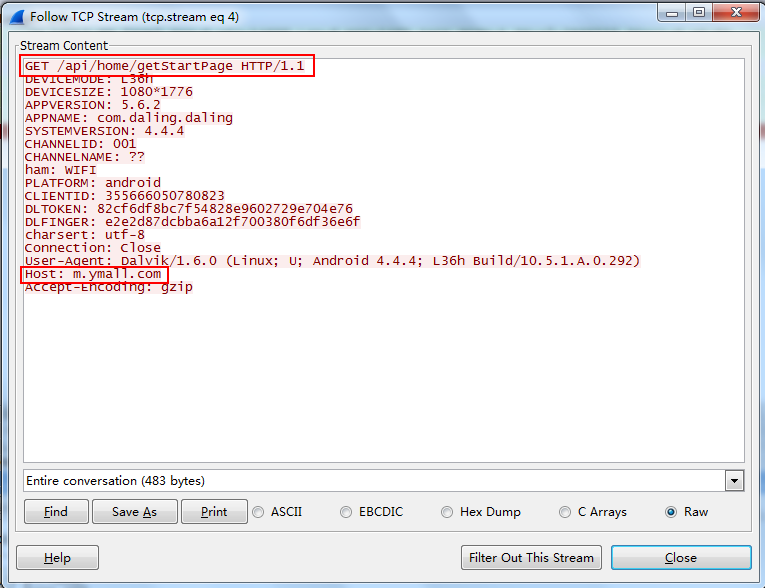
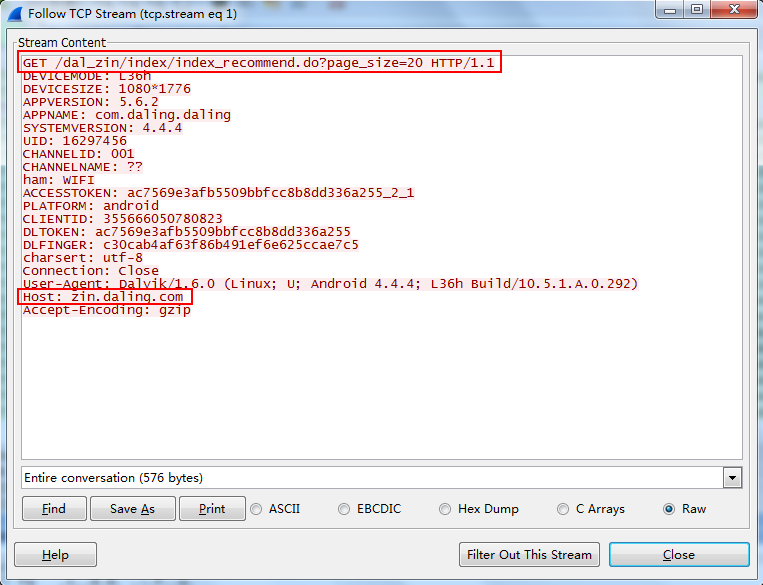
再次是打开APP首页操作，抓取直接打开达令全球好货所产生的网络流量后生成数据包，利用Wireshark的过滤器过滤出http协议并且host是“m.ymall.com”的报文，并进行相关分析，找到达令全球好货打开APP首页动作的URL并提取出动作的特征。如图3-9所示为打开达令全球好货首页的特征：

图3-9 打开达令全球好货首页的特征

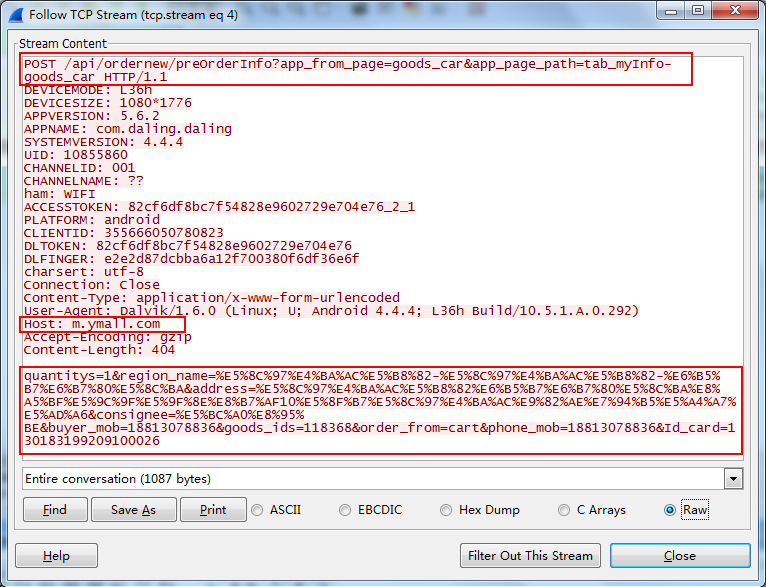
从图中第二个红框HOST可以看出该APP就是达令全球好货，并且由第一个红框可以的得到该APP打开首页的特征就是“/api/home/getStartPage”。

再就是打开APP内部其他页面操作，就是在具体使用达令全球好货过程中，不同页面之间的跳转。那么，抓取点击其他页面跳转的网络流量后生成数据包。一款APP可能有不同的host，所以该APP对应的host可能会发生变化，利用Wireshark的过滤器过滤出http协议的报文，并进行相关分析，找到达令全球好货打开APP内部其他页面动作的URL并提取出该动作的特征。如图3-10所示为打开达令全球好货内部达令帮页面的特征：

图3-10 打开达令全球好货内部达令帮页面的特征

从图中第二个红框可以看出对应的host就是达令的中文拼音，虽然host发生变化，不再是“m.ymall.com”，但是这个host仍是达令全球好货的host，该APP仍是达令全球好货；再由第一个红框可以得到打开达令帮这个页面的特征是“/dal\_zin/index/index\_recommend”，后面的page\_size并没有影响，只是限定一页显示多少个商品，在这里是显示的20个。

最后是交易操作，抓取最后一个交易完成动作的网络流量，而不是提交订单动作的网络流量后生成数据包，利用Wireshark的过滤器过滤出http协议并且host是“m.ymall.com”的报文，并进行相关分析，找到达令全球好货交易动作的URL并提取出该动作的特征。如图3-11所示为达令全球好货交易动作的特征：

图3-11 达令全球好货交易动作的特征

从图中第二个红框的host可以看出该APP就是达令全球好货；由第一个红框的“/api/ordernew/preOrderInfo”可以看出是与交易有关的网络流量，并且http请求方式为post；最后由第三个红框内可以得到订单的相关信息，包括：商品数量、用户名、商品ID等等。

当然，一个APP的使用还有很多其他的动作可以进行分析，但是限于篇幅，在此不再赘述。

3.2.4.3 问题分析及解决思路

与APP下载动作面临的问题相同，当APP数量不多的情况下，可以通过人工操作来提取APP使用动作的特征，但是，如果APP数量很多，再利用人工的方式就不现实，需要考虑人力、物力成本，那么就需要思考如何通过自动化的方式节约成本。与APP下载动作一样，也可以利用自动化的测试框架实现APP使用操作的自动化。

## 3.3 流量规则文件的生成与后续使用

在这个大数据时代，我们需要善于从大数据中提取到想要的信息，那么对于移动APP的使用模式研究同样如此。如果一个APP需要投资，对于投资者来说，获取的有效信息大部分是由内部的负责人提供，对于这些信息的真实性有待考察。而投资者又很难通过其它的途径了解到这个信息，这就需要一个第三方对于这些信息进行统计和验证，确保信息的准确性、客观性，从而保证投资者不会做出错误的决定。

一般某个APP的日常活跃用户数，每天新用户的增长量，或者各种用户的操作细节这些信息是掌握在公司内部的，属于商业机密，是不会或者很少公开的，然而我们可以通过合法的手段在海量的网络访问数据中作为第三方获取这些有用的信息。

以上介绍的都是对移动APP进行抓包，提取下载或者使用动作的特征，可以改写成正则表达式或者其他形式，然后生成流量规则文件，在本课题中，流量规则文件就是一个csv文件。因为这些特征对于一个APP来说是独一无二的，那么就可以利用这个规则文件在海量的网络访问数据中编写MapReduce程序进行某款APP的用户行为分析，得到该APP的欢迎度。最后可以通过一份报告或者一个前端网站等形式呈现给投资方，再由投资方决定这款APP是否值得投资。

比如本课题中的小红书下载动作，抓取流量后提取流量特征是com.xingin.xhs\_102603.apk，那么可以改写成“com.xingin.xhs.\*apk”的正则表达式形式作为小红书下载动作的流量规则。之后就利用这个正则表达式，编写MapReduce程序在海量的DPI数据中提取出哪些用户曾经下载过小红书这款APP，作为其中的一个维度，从而可以进行用户分析得到小红书的欢迎度。

## 3.4 本章小结

首先介绍了如何选择不同类别的移动APP，以及最后选定的APP类别和名称；再简单介绍了移动APP流量中的HTTP协议的研究；其次是移动APP手动下载和使用动作特征提取分析，以及面临的问题和解决思路；最后介绍了提取出移动APP的动作特征后生成的规则文件的后续使用。本章起到总括全文的作用，本课题的最终目的是通过海量的网络访问数据进行某款APP的用户行为分析，但是前提是这些APP独一无二的下载和使用特征需要得到，这就涉及到抓包得到数据包，再进行相关分析。

# 第四章 移动APP下载操作自动化

## 4.1 自动化下载移动APP的实现思路

在3.2.3.3小节中已经介绍过，当应用商店或者APP数量很少的情况下，可以人工进行数据包的抓取、分析以及特征提取。但是当工作量很大时，再利用人工就不现实，必须寻找自动化的途径来模拟手动操作解决问题，实现操作的自动化。那么，既然是人的手动点击动作，就可以利用安卓的自动化测试框架来模拟人的动作。

自动化测试框架，即被用来进行自动化测试的框架，也就是一个或多个基本模块，管理模块，统计模块等组成的一套工具，可以提供可重用的基本自动化测试模块，如打开某个应用程序，模拟鼠标或者键盘的点击操作动作，测试被测对象，最后验证测量对象的属性；或提供架构模块的自动化测试执行和管理功能，仅用于组织，管理和实施那些独立的自动化测试用例，在完成统计测试后测试结果，一般情况下这样的框架通常被集成到基本自动化测试模块[22]。

综上，自动化下载移动APP的实现思路就是利用安卓自动化测试框架模拟人的手势操作，实现APP的自动化下载。

## 4.2 自动化下载移动APP的实现方案

### 4.2.1 下载动作模拟思路

移动APP的下载指的是在一个应用商店中进行下载安装，一般操作过程是打开某个应用商店，点击搜索框输入要下载的APP名称再搜索，在搜索出的结果中判断是否是要下载的APP，最后进行下载安装。那么对于具体下载动作的模拟思路，也就是按照上述的操作过程，模拟手动操作实现APP的下载。

### 4.2.2 自动化测试框架选择

目前，Android基于用户层面的自动化测试框架，都可以看作是基于Android控件的，涉及Widgets和WebView两个主要方面。主要测试方法有以下几种：一个是通过Android所提供的各类服务，获取当前窗口信息的视图。然后，在当前视图中找到目标控件，并根据控件的属性信息来计算控件中心点的坐标，最后构造一个Android Input事件，从而实现自动化测试。其主要特点是：测试代码和被测应用各自运行在各自的进程内，相互独立。其代表有UiAutomator[23]。另一个是基于Instrumentation，通过测试代码和应用程序代码，更准确地说是测试APK和被测APK，在同一个进程中运行，通过Java反射机制获取窗口的所有当前视图，并根据视图查找属性信息的目标控件，并计算目标控制中心点的坐标[23]。然后，利用Instrument内部接口，实现点击操作。其代表有Robotium。

现在先来看下第一种。这类方法通常需要满足两个功能，一是能获取屏幕视图；二是能产生输入事件。这样，用户可以通过屏幕视图找到目标控件，然后计算中心点坐标，从而产生一个输入事件来模拟用户操作。通常，这样的框架还将提供屏幕截图功能，方便用户对比分析。例如，定位问题的分析等。接下来是第二种框架，即Instrumentation，它是Android对外提供的一系列的测试方法的核心。Instrumentation可看成一系列控制函数的集合，作用于系统和应用之间，类似于Windows中的Hook。在这类测试框架下，主程序和测试程序是运行在同一个进程中的，如图4-1所示：



图4-1 Android的测试框架

下面介绍一些目前常用的自动化测试框架：

Monkey是Android SDK自带的一个测试工具，具体测试方法是发送伪随机用户事件流到被测对象，如按键输入，触摸屏输入，手势输入等，实现被测对象的压力测试，并且输出日志文件。其实这个工具的局限性是只能做一些压力性的测试，并且测试事件和数据是随机的，不能根据实际情况进行定制。

MonkeyRunner也是由Android SDK自带的测试工具。严格来说，MonkeyRunner是一个API工具包，比Monkey功能强大，测试人员可以编写测试用例程序实现自定义输入数据。缺点是脚本需要用Python编写，对编程语言有太高的限制要求。

Instrumentation是Google早期提出的一款Android自动化测试工具。虽然在那时候JUnit也可以对Android进行测试，但是Instrumentation允许对应用程序做更为复杂的测试，甚至是框架层面的。通过Instrumentation可以模拟按键操作、屏幕点击或者滚动等动作。缺点是对测试人员来说对编写代码能力要求较高，并且还要对Android的相关知识有一定了解，还需要配置AndroidManifest.xml文件，不能跨多个App。

UiAutomator也是Android提供的自动化测试框架，基本上支持所有的Android事件操作，但是它不需要测试人员了解代码实现细节，也就是说可以用UiAutomatorviewer抓取APP页面上的控件属性而不需要知道源代码。缺点是只支持SDK 16（Android 4.1）及以上，不支持Hybird App、WebApp。

Calabash是一款跨平台的自动化测试框架，支持Android和iOS原生和混合的应用程序。Calabash易于理解的语法使得甚至非技术人员可以在两个移动平台上创建和执行APP的自动化验收测试。Calabash测试在Cucumber中描述，然后在运行时转换成Robotium或Frank。

Selendroid也是基于Instrumentation的测试框架，可以测试Native App、Hybird App、Web App，但是如果具体应用的话参考资料不多，不容易实现。

Robotium也是基于Instrumentation的测试框架，目前国内外用的比较多，资料比较多，社区也比较活跃。缺点是测试人员必须有一定的Java基础，并且需要了解Android的基本架构，而且不能跨多个APP进行测试。

目前存在的自动化测试框架有很多，限于篇幅不再赘述，综上各种框架的介绍和优缺点比较，在本课题中选择Robotium来实现移动APP下载动作的模拟。

### 4.2.3 自动化测试框架Robotium简介

Robotium是一个国外的Android自动化测试框架，主要作用是用于Android平台的APP自动化测试。它提供了各种API实现人的各类手势操作的模拟以及其他搜索或者断言[20]。另外，Robotium跟Android的官方框架结合实现APP的自动测试。

借助于Robotium支持，测试人员能够编写测试用例，且可以跨越多个Android activities。Robotium的自动化测试就是模拟用户的各种手势行为操作，并且能够把一些最初由测试人员完成的测试用例转变成Robotium的自动化测试用例。

Robotium的优点包括[21]：（1）以最小的应用程序知识，开发功能强大的测试案例。（2）支持多个activities的自动活动。（3）可以用最短时间写出自动化测试用例。（4）测试用例的可读性大大提高。（5）运行时绑定GUI组件，使测试用例更加强大。（6）执行测试用例速度快。（7）成功整合了Ant或Maven来执行测试。

Robotium的缺点包括[21]：（1）不能对WebView进行操作。（2）不支持跨平台。（3）API 中有很多是基于Android框架控件的位置ID来查找的，对以后测试用例的维护造成不便。（4）API是面向过程的，导致测试代码的可扩展性降低。（5）Instrumentation框架下APP和testapp是运行在一个进程中的，那么，app crash可能会导致testapp一并crash。

Robotium的局限性包括：（1）对于复杂布局，当前方法可能无法获取所需的数据。（2）对于自定义布局，必须导入源代码，调用自定义方法进行验证，增加编码难度。（3）大部分验证基于UI数据的方式，无法检查UI布局的视觉问题。

### 4.2.4 adb shell简介

adb（Android Debug Bridge），是安卓调试工具的英文缩写，起到Android应用调试桥的作用，实际上就是debug工具。adb以较特殊的方式工作，使用监视套接字Socket 5555等端口方式进行IDE和Qemu通信。

adb是Android SDK中的一个自带工具，使用此工具可以直接操作或者管理Android模拟器或真实的Android终端设备。其主要功能是：运行设备shell（命令行）; 管理模拟器或设备端口映射; 在计算机和设备之间上传/下载文件; 本地apk软件安装到模拟器或者Android设备。adb是一个客户端-服务器程序，其中客户端用于操作的计算机，而服务器是Android设备。

adb各个模块的定义如下，（1）adb server：计算机上的一个服务进程，进程名为adb；（2）adbd (adb daemon) ：Android手机上的一个服务进程，进程名为adbd；（3）adb client：可以认为是计算机上的一个终端窗口，进程名也为adb。原理逻辑图如图4-2所示：



图4-2 adb原理逻辑图

adbd（ADB daemon）：在设备或模拟器守护程序上运行。主要作用是将设备或模拟器与adb server连接，设备通过USB连接，模拟器通过TCP连接; 为设备或模拟器提供adb服务。adb服务是指adbd提供的功能。在设备或模拟器端，adbd还创建了本地套接字和远程套接字，前者是通过jdwp与dalvik VM进行通信，后者通过与adb server的TCP / USB通信。

adb server是运行在后台进程中，用于管理客户端并在模拟器或真机上运行守护进程通信。在host端，adb会fork出一个守护进程，即server，而父进程继续处理client请求，而server创建本地套接字和远程套接字，前者用于与client的通信，后者用于与远端（adbd）的通信。该守护进程会长期运行在后台，并且无控制台界面，称之为adb server（adb服务端）。主要工作有如下两部分，（1）管理Android模拟器或者用USB线连接的Android设备，负责维持运行在其中的adbd进程和自身的数据通道；（2）实现设备或者模拟器与PC之间的数据拷贝。

adb client运行在开发用的计算机上，可以运行adb命令来调用该客户端，并且Android工具也能够调用adb客户端。提供给用户操作的命令行工具，对用户暴露了install、push、shell等接口，实现用户交互，称为adb client（adb客户端）。它的主要工作是解析命令的参数，做一些必要预处理，之后转换成指令或者数据，并且发送给adb server。adb server再将指令或者数据转发到模拟器或者其他设备中，再由adbd来处理，随后产生结果，最后通过adb server接收回来。总而言之就是指令通过adb client交给adb server，然后server和daemon再进行通信。所以，这里建立的连接实际上是server和daemon之间的连接。当启动adb client时，客户端首先检测adb server进程是否正在运行，若没有运行，则启动服务端，反之，则不需启动。当服务端启动的时候，它会绑定到本地的TCP5037端口，并且监听从adb客户端发来的命令——所有的adb client都使用5037端口与adb server通信。

adb整体架构和数据传输图如图4-3所示：



图4-3 adb整体架构和数据传输图

## 4.3 基于Robotium实现移动APP自动化下载

### 4.3.1 自动化下载流程设计

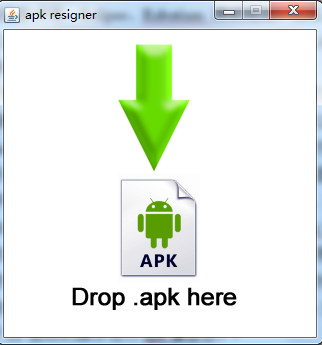
自动化下载就是编写Robotium测试程序按照4.2.1小节介绍的下载动作模拟思路模拟人的手势操作。但是考虑到应用商店中待下载的某些APP可能不存在，那么就需要对APP进行存在性判断。本课题中判断的方式是依次打开搜索出的前三个结果，判断打开后APP的详情页是否包含该APP的名称，若包含，则下载；反之，则不下载。如果前三个结果都不是要下载的APP，那么保存该APP名字到一个文件中，以备后续统计。 APP的自动化下载流程图如图4-4所示：



图 4-4 APP的自动化下载流程图

### 4.3.2 应用商店安装包重签名

所有的Android应用程序，开发者都要用证书对其进行数字签名，是因为Android系统不会安装无签名的应用程序。程序开发阶段，之所以能够在模拟器上安装并成功运行，是因为在开发期间，由于是以 Debug 模式进行编译的，ADT会自动用默认的密钥和证书来对其进行签名。但是，在以发布模式编译时，apk文件并不会得到自动签名，所以就需要手工签名。所以，在进行自动化测试时，需要对相应的应用商店安装包apk文件进行重签名，然后再进行安装才能对APP进行自动化操作。

本课题中应用商店选择“豌豆荚”，重签名是利用re-sign.jar重签名工具，如图4-5所示，基于UI层级的操作进行apk重签名。将需要重签名的豌豆荚安装包apk文件拖至重签名工具的界面上，当出现如图4-6所示的界面时，重签名成功，并且生成一个新的.apk 后缀的安装包。

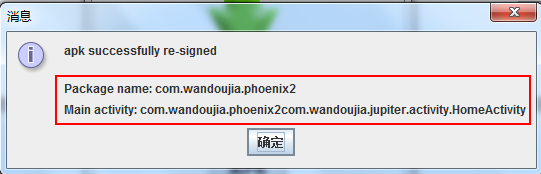
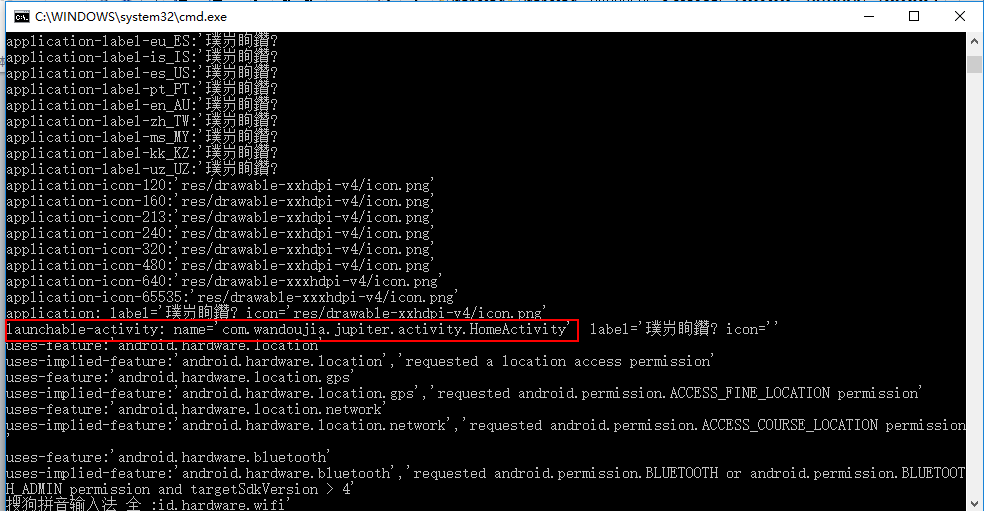
图4-5 重签名工具re-sign.jar

图4-6 豌豆荚安装包重签名成功

在此需要记录一些相关信息，如图4-6红框内所示，第一行是apk的包名，第二行是主Activity名，也就是启动应用商店后第一个画面相应的类名。

当然，这两个信息还可以利用命令行方式获取，aapt dump badging xxxx.apk。如图4-7所示红框内为豌豆荚apk的包名，如图4-8所示红框内为豌豆荚主Activity名。（因为限于cmd窗口大小，所以执行以上命令后的截图是两个。）豌豆荚安装包重签名后安装到模拟器上

图4-7 豌豆荚apk的包名

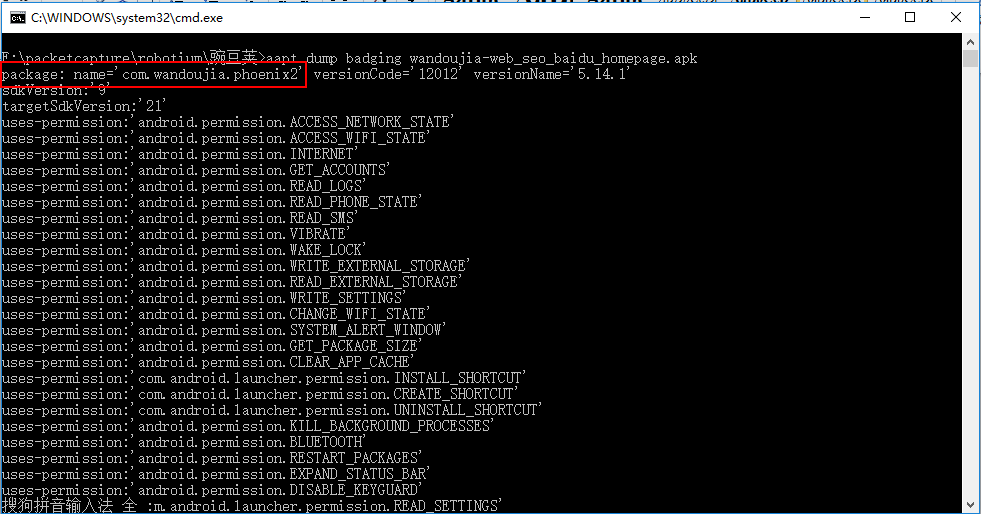
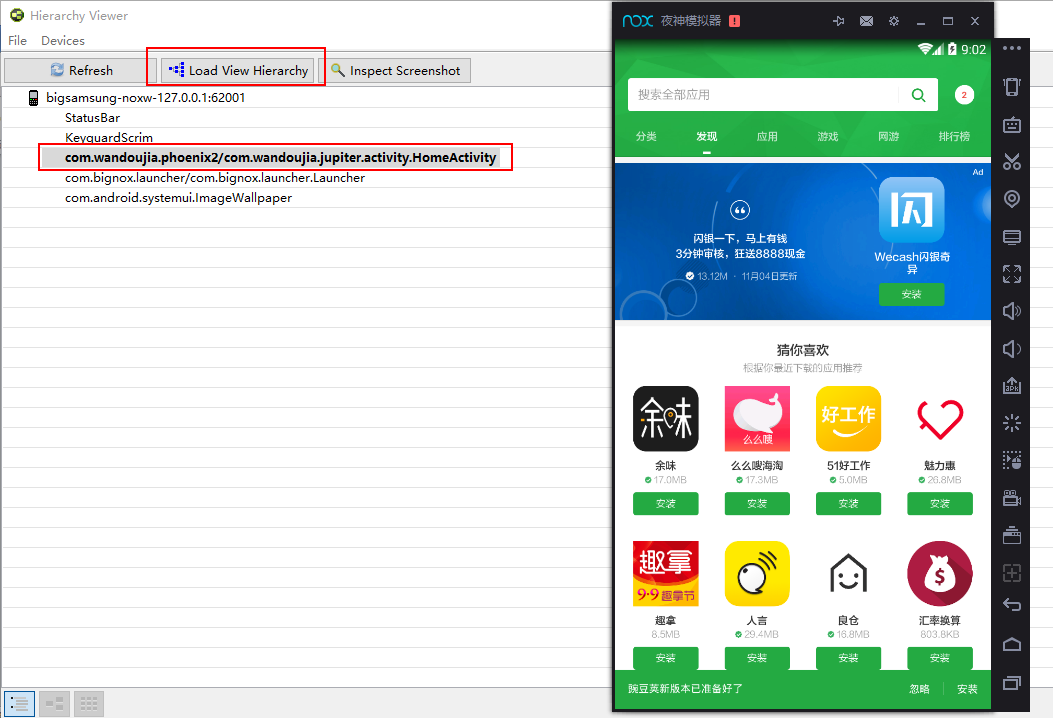


图4-8 豌豆荚主Activity名

### 4.3.3 控件定位方式

由于Robotium 模拟各种手势操作的定位是基于控件实现的，如果要对控件进行操作就必须先找到元素的定位方式。元素获取可以应用Android SDK中的Hierarchy Viewer.bat工具，在模拟器中启动应用商店来定位每个操作所对应的控件，如图4-9所示。

图4-9 利用Hierarchy Viewer.bat进行控件定位

选中高亮的部分，再点击Load View Hierarchy载入当前应用商店页面的控件。在Hierarchy Viewer中，应用商店被解构成许多控件，控件之间的结构关系形成一张复杂的网络。需要做的就是在这张“网”中找到每一步操作所涉及到的控件以及对应的控件ID，最终实现在众多控件中唯一确定它。如图4-10所示，豌豆荚的首页框架直接在Hierarchy Viewer 显示出来。

图4-10 豌豆荚的控件框架

随后，就是按照Hierarchy Viewer中显示出来的控件框架结构，找寻搜索框和搜索按钮的控件，如图4-11和图4-12所示；

图4-11 豌豆荚搜索框的控件ID

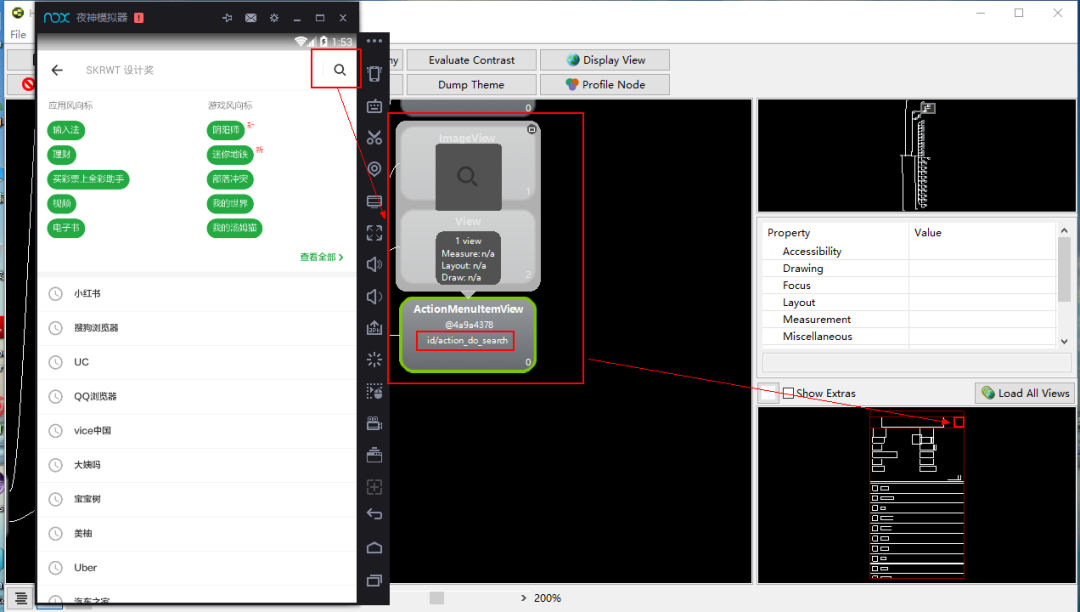


图4-12 豌豆荚的搜索按钮的控件ID

### 4.3.4 待下载移动APP的存在判断

对于待下载的APP需要判断应用市场中是否存在，经过分析研究，可以依据APP简介的控件依次打开搜索的前三个结果，进入APP详情页来判断此APP是否是待下载的APP，根据4.3.3小节介绍的控件定位方式，找到APP简介的控件ID如图4-13所示：

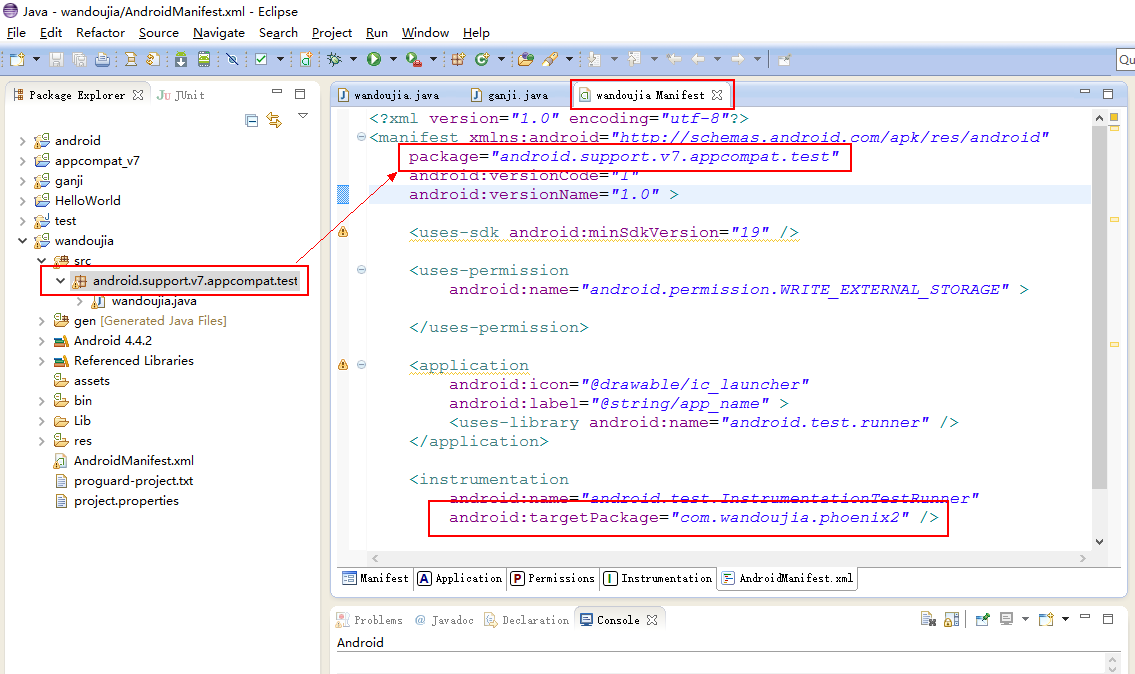
图4-13 豌豆荚中对APP简介的控件ID

### 4.3.5 自动化下载动作实现

接下来就通过编写Robotium程序进行安卓自动化测试实现APP下载动作自动化。在进行测试前需要准备的软件有：Eclipse、Robotium测试框架、SDK、ADT。将相应的软件都配置好以后开始正式进行APP自动化下载。Robotium 自动化有两种方式，第一种是白盒测试，另一种是黑盒测试。由于应用商店的APP源代码很难获取，所以这里只能进行黑盒自动化测试。

首先是把重签名后的应用商店安装包安装在模拟器上，其次是编写程序来实现APP下载操作模拟。

在Eclipse新建一个安卓测试工程，因为没有应用商店的源代码，属于黑盒测试，故新建安卓测试工程时需要选择This project作为测试目标。再将测试工程中的AndroidMainfest.xml进行相应的配置，如图4-14所示，其中package 一项必须是测试工程的文件所在的包，这里是“android.support.v7.appcompat.test”。android：targetPackage 一项必须是应用商店重签名后的包名，也就是上面提到的豌豆荚apk包名，这里是“com.wandoujia.phoenix2”。

图4-14 安卓测试工程AndroidMainfest.xml文件配置

在android.support.v7.appcompat.test包名下新建一个名字是wandoujia的测试类，最基本的一个测试类应满足以下四个条件：（1）必须继承ActivityInstrumentationTestCase2；（2）Activity必须与被测应用Activity相同；（3）必须有一个 setUp、tearDown 方法；（4）所有正式的测试方法必须以test开头。最后在测试工程新建Lib 目录，用来存放Robotium 的jar包。

然后就是模拟人的下载动作手势操作进行APP的下载，Robotium的核心类是Solo，通过Solo类可以对Android控件进行各种操作。核心操作代码介绍如下：

solo.clickOnView(solo.getView("search\_container"))，search\_container是搜索框的控件ID，实现点击搜索框。

EditText AppName = (EditText) solo.getView("search\_sug\_input");

solo.enterText(AppName, name)，这两行是实现在搜索框中输入待下载的APP名称，其中search\_sug\_input是输入文本的控件ID，name是APP的名称。

solo.clickOnView(solo.getView("action\_do\_search"))，action\_do\_search是搜索按钮的控件ID，实现APP的搜索。

TextView textview = (TextView) solo.getView("description",j);

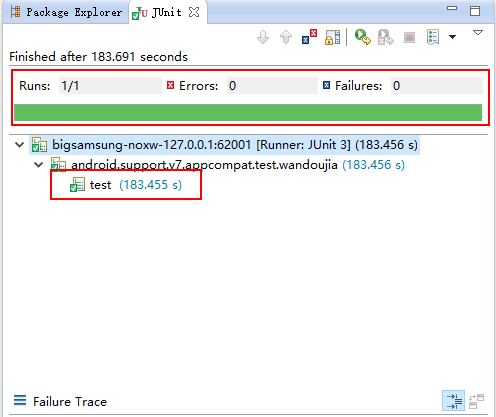
solo.clickOnView(textview)，这两行是实现依次点击进入前三个搜索结果判断是不是待下载的APP，其中description是APP简介的控件ID，j是一个计数器，用来计算APP个数。

solo.clickOnText("^安装")，实现进入APP详情页后如果判断该APP就是待下载的APP那么就点击下载，其中“安装”是豌豆荚进行APP下载时指定的按钮名称，而“安装”前面的“^”是不管该页面有多少个“安装”字眼，都是点击第一个“安装”按钮。

程序编写完成后，连接好模拟器，将工程以Android JUnit Test运行，对程序进行运行测试。本课题中待下载的APP分别是Uber、美柚、宝宝树、大姨吗、小红书、vice中国，可以在模拟器上依次看到这几个APP搜索后打开APP详情页，判断后进行下载安装的全部自动化过程。

## 4.4 运行结果及分析

### 4.4.1 控制台输出信息分析

因为工程是以Android JUnit Test运行，所以在Eclipse的Junit进度显示可以看出整个测试程序运行成功，并且运行时间是183.455s，如图4-15所示；在Eclipse的控制台输出信息可以看出测试程序是以一个测试apk的形式安装到模拟器，再执行，并成功运行，完成测试，如图4-16所示。

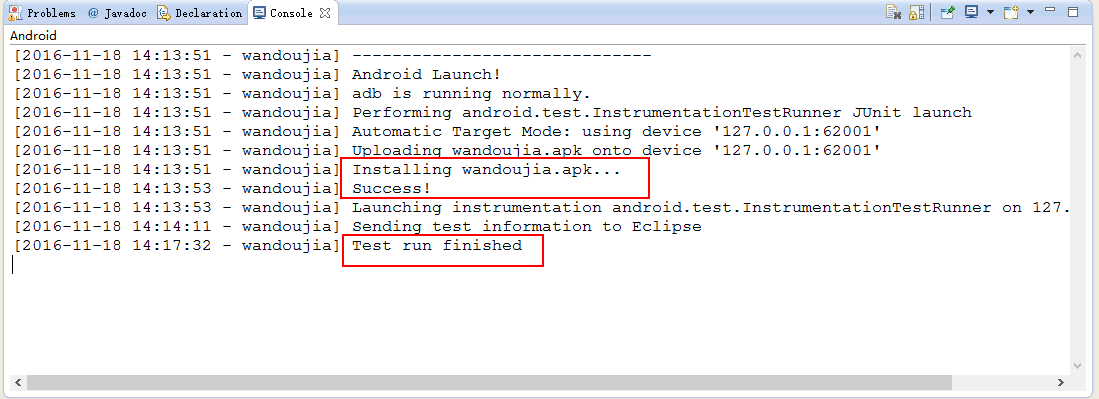
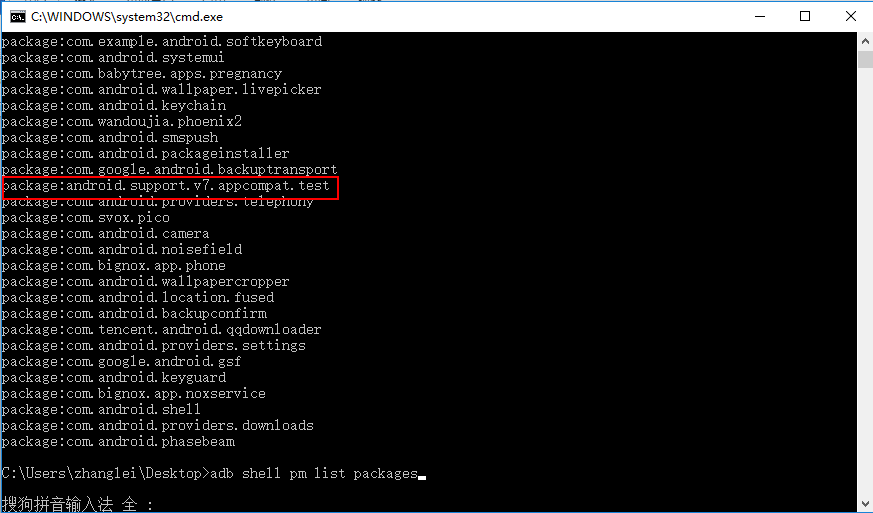
图4-15 Eclipse的Junit进度显示

图4-16 Eclipse的控制台输出信息

### 4.4.2 Robotium程序生成的APK安装

在命令行工具中执行adb shell命令——adb shell pm list packages，可以获得模拟器中所有的APP包名，其中android.support.v7.appcompat.test就是豌豆荚测试程序的包名，与编写程序时的相同，如图4-17所示：

图4-17 模拟器上Robotium程序生成的APK安装

### 4.4.3 下载的移动APP安装

安装包下载后进行安装的过程是Android系统的操作过程，与应用商店没有任何关系，所以相应的包名也不相同。因为Robotium不支持跨平台，所以需要设置应用商店安装包下载完成之后自动安装，如图4-18所示。其中，vice中国在豌豆荚中不存在，将其名称保存在“/sdcard/sms.log”中。

图4-18 模拟器中下载的移动APP安装

### 4.4.4 Android模拟器运行效果

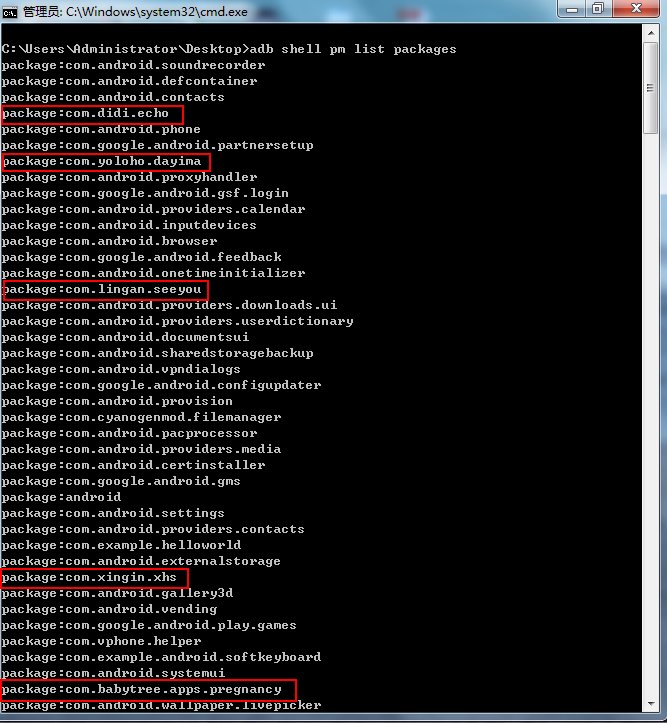
在命令行工具中执行adb命令——adb shell pm list packages，获得模拟器中所有的APP包名，其中com.didi.echo、com.yoloho.dayima、com.lingan.seeyou、com.xingin.xhs、com.babytree.apps.pregnancy分别是Uber、大姨吗、美柚、小红书、宝宝树的包名，证明APP自动化下载成功，如图4-19所示。

图4-19 模拟器中所有APP包名

## 4.5 本章小结

本章介绍了移动APP下载操作自动化。首先介绍了自动化下载移动APP的实现思路，就是编写自动化测试框架的测试程序模拟人的手势操作来实现APP自动化下载；其次简单介绍了几种常见的自动化测试框架，以及本课题用到的Robotium框架的简介；最后是介绍如何编写Robotium程序实现移动APP自动化下载，以及相应的运行结果分析。

# 第五章 移动APP下载流量抓取及流量规则提取自动化

## 5.1 自动化抓取移动APP下载流量的实现思路

第四章已经介绍了如何利用Robotium编写自动化测试程序实现移动APP的自动化下载，所以具体如何自动化下载APP已经实现。那么，再就是需要思考如何抓取APP下载的网络流量，也就是抓包，将网络传输发送与接收的数据包进行截获、重发、编辑、转存等操作，也用来检查网络安全。抓包也经常被用来进行数据截取等。

因为APP下载是自动化的，全程没有人工干预，所以抓取流量也需要自动化，不能通过人来开启Wireshark或者其他抓包工具来抓取流量。

当前Android系统越来越流行，无论是对于安卓应用的开发人员，还是对于网络安全的研究人员，都有可能需要掌握捕获Android应用通信数据包的方法。根据不同的技术手段，常用的方法分为两类，一种是直接在Android移动终端上的监控数据流，另一种是通过Android智能移动终端接入上层网络设备或线路截获数据流。

下面先介绍第一类移动APP抓包方法。

首先是需要为Android设备的Root权限解锁。Root权限是Android系统的超级管理员权限，Root权限和Windows下的系统管理员权限类似。一般情况下，为了避免不可更改的系统文件被用户无意中修改，Android系统不会把Root账户给普通用户开放，因此，必须先解锁Root权限才能获得Root权限。一般情况下，把Android手机的Root权限是否解锁，简称为Android手机是否root过。

其次就是本课题需要用到的相关工具，主要有两个：Tcpdump(Android版) 和 adb。将 Tcpdump (Android版)上传至Android设备上，给 Tcpdump 增加可执行权限，在Android设备上进行相关操作，产生通信数据包，同时使用 Tcpdump 抓包，并将结果写入一个文件保存，完成移动APP网络流量的抓取。

第二类就是2.3节提到的通过移动设备或者模拟器，在PC端抓取流量，但是本课题研究的是自动化，那么这就涉及到两个方面，一个是自动化下载APP，另一个是抓取APP下载的流量。所以这就还需要一个程序把两者结合，实现下载动作自动化，抓取流量自动化。

## 5.2 自动化抓取移动APP下载流量的实现方案

### 5.2.1 下载流量抓取思路

经过5.1节如何实现自动化抓取移动APP网络流量的分析研究，权衡两类方法的优缺点，本课题采用第二类方法。与第三章手动抓取流量不同的是实现自动化。

目前，模拟下载动作，APP的自动化下载已经实现，再就是需要考虑使用什么工具进行抓包。现在有很多抓包工具，也就是一些拦截查看网络数据包内容的软件。本课题就是运用现有的抓包工具实现自动化的抓取APP的下载流量。

### 5.2.2 抓包工具选择

针对物理网络的异构性及应用需求的多样性，大型信息系统大多采用专用网络传输协议实现数据传输，为上层各种应用提供数据通信基础。网络抓包工具用于确定网络传输内容及状态，是应用开发人员及系统维护人员在日常工作中的必备工具[29]。

下面介绍一些目前常用的抓包工具：

Wireshark，简介和简单使用已在第二章介绍过，在此不再赘述。

Fiddler，HTTP协议调试代理，记录并检查所有计算机和Internet之间的HTTP流量，设置断点，并检查所有传入和传出的Fiddler数据。Fiddler比其他网络调试器简单得多，因为它不仅公开了http通信，而且提供了用户友好的格式。它的灵活性是巨大的，可以支持大量的http调试任务，并且可以使用.NET框架语言进行扩展。

Tcpdump（dump the traffic on a network），即在网络上转储流量， 根据用户的定义截获网络包并进行包分析。它提供源代码，公开接口，并且是高度可扩展的，是网络维护和入侵者的有用工具。

嗅探器，中文可以翻译成嗅探器，也称为抓包软件，是基于被动监听网络分析的原理。使用此技术，可以监视网络的状态，数据流和通过网络传输的信息。它是一个利用以太网的特点将网络适配器卡置于凌乱模式的工具。一旦网卡设置为此模式，它可以接收通过网络传输的每个数据包。Sniffer分为软件和硬件两种，软件的有Sniffer Pro、Network Monitor、PacketBone等，其优点是易于安装和部署，易于学习使用，而且易于交换; 但是其缺点是不能真正了解网络故障和操作。硬件的Sniffer通常称为协议分析器，一般是商业化的和昂贵的，但是具有支持各种类型的扩展链路捕获和高性能实时数据捕获分析的能力。

WinNetCap，是目前唯一一款能够即时显示通讯进程名的嗅探工具，在抓包的同时，WinNetCap会告诉当前包是谁发送的或谁接收的。WinNetCap的核心功能是高速，稳定的网络数据采集，支持插件免费开发。WinNetCap有一套非常丰富的包过滤器设置，具有网络端口监控，网络诊断，流量统计，网络过程显示等功能。可用于网络应用协议分析，网络抓包，简单网络诊断，网络流量测量，手工查杀可疑网络进程等方面。

目前存在的抓包工具还有很多，限于篇幅不再赘述，综上各种工具的介绍和优缺点比较，在本课题中选择Tcpdump来抓取移动APP自动化下载流量。

### 5.2.3 Tcpdump简介

TcpDump能够把在网络中传输的数据包完全捕获下来并提供分析。它支持针对网络层、协议、主机、网络或端口的过滤，并提供逻辑语句来帮助去掉无用的信息[22]。Tcpdump命令可以在数据链路层抓取数据帧，并将帧内容以原始报文的格式输出。

Tcpdump是Internet上系统管理员的必备工具，是每个高级系统管理员分析网络问题和解决问题的强大的最重要的工具之一。

Tcpdump的输出格式为[24]：系统时间 来源主机.端口 > 目标主机.端口 数据包参数。Tcpdump支持相当多的不同参数，例如：（1）使用-i参数指定Tcpdump监听的网络接口，这在计算机具有多个网络接口时非常有用；（2）使用-c参数指定要监听的数据包数量；（3）使用-w参数指定将监听到的数据包写入文件中保存，等等。

但是，更复杂的Tcpdump参数是用于过滤的，这是因为网络流量非常大，如果不过滤，那么整个捕获过程中产生的所有封包都会被保留下来，容易造成数据冗余，且不容易找到需要的网络封包。由不同的参数定义的不同的过滤规则可以用于缩小目标，以通过捕获特定分组来更好地分析网络中存在的问题。

Tcpdump对截取下来的数据并没有完全解码，大多数数据包的内容是使用十六进制形式直接打印输出的。显然这不利于分析，通常的解决方案是使用-w参数的Tcpdump命令截取数据并保存到文件，然后使用其他程序进行解码分析。当然也可以定义过滤规则，以避免捕获填充整个硬盘的数据包。

报文输出包含原始消息输出及格式化消息输出两种方式[25]。如果原始消息输出时，输出数据链路层的数据，定位和查询工作的消息内容由用户完成。如果采用格式化消息形式输出，则将IP首部、TCP/UDP首部、专用网络传输协议首部中关键部分进行解析并输出。但是它的实现效率不高，仅适用于网络消息少的情况。

## 5.3 软件实现移动APP下载流量自动化抓取

### 5.3.1 自动化抓取下载流量流程设计

自动化抓取下载流量，其中包含两个线程，一个是自动化下载APP，另一个是抓取APP下载的流量，这就需要一个多线程程序将两者结合，实现全部自动化。

Tcpdump是命令行工具，需要用Ctrl+C停止抓包。但是，如果要实现全部自动化，就不能人为检查Robotium程序是否完成再停止抓包，所以，就需要用命令行来结束Tcpdump，也就是需要实时监控进程数，如果进程数减少一个，这就代表Robotium程序结束，再用命令行结束Tcpdump抓包。

在4.4.3小节提到过安装包下载后进行安装的过程是Android系统的操作过程，与应用商店没有任何关系，所以相应的包名也不相同。因为Robotium不支持跨平台，所以需要设置应用商店安装包下载完成之后自动安装。但是，不管模拟器的存储空间多大，若APP数量很多，终有存储空间不足的情况出现，为了避免抓取一些APP的下载网络流量后手动删除已安装的APP，所以，这就需要APP下载这个动作完成抓取下载动作的流量后自动化删除这些安装的APP。

经过上面的分析可以得出，APP下载流量自动化抓取程序需要三个线程，第一个是Robotium程序模拟手势操作的APP下载线程，第二个是Tcpdump命令行抓包线程，第三个是监控进程数来判断Robotium程序是否完成，如果进程数减少一个，那么就停止Tcpdump。最后，删除已安装的APP，具体实现框图如图5-1所示。



图5-1 自动化抓取下载流量框图

### 5.3.2 命令行运行Robotium程序

Robotium程序是以一个测试apk的形式安装到模拟器并执行，那么就可以用adb命令来执行该程序，大致分为以下四种情况，举例说明。

（1）运行测试工程下的所有用例：adb shell am instrument -w com.taobao.taobao.test/android.test.InstrumentationTestRunner。

（2）运行单个测试类或某个TestSuite，运行测试类com.taobao.taobao.test.TestRegister：adb shell am instrument -e class com.taobao.taobao.test.TestRegister –w com.taobao.taobao.test/android.test.InstrumentationTestRunner。

（3）运行某个测试类里面的某个测试方法，运行com.taobao.taobao.test.TestRegister中的测试方法testRegister：adb shell am instrument -e class com.taobao.taobao.test.TestRegister#testRegister -w com.taobao.taobao.test/android.test.InstrumentationTestRunner。

（4）运行两个不同的测试类或类中的方法，运行com.taobao.taobao.test.TestLogin和com.taobao.taobao.test.TestRegister类中的方法testRegister：adb shell am instrument -e class com.taobao.taobao.test.TestLogin, com.taobao.taobao.test.TestRegister#testRegister  -w com.taobao.taobao.test/android.test.InstrumentationTestRunner。

本课题用到第二种方法，测试类就是android.support.v7.appcompat.test.wandoujia，实现命令行执行Robotium自动化下载APP程序。

### 5.3.3 待下载移动APP的安装及卸载

在4.4.2小节简要介绍了adb工具，那么可以执行adb命令来删除已安装的APP。因为APP都是下载好后自动安装的，首先，需要判断哪些是新安装的APP。这就需要在未开始下载前用“adb shell pm list packages”命令将模拟器上所有的APP包名保存下来，然后等抓包结束后，再执行一次此命令获得安装后模拟器上所有APP包名，之后进行比较，得到哪些是新安装的APP，最后运行“adb uninstall”命令把新安装的APP删除。

### 5.3.4 移动APP下载流量自动化抓取实现

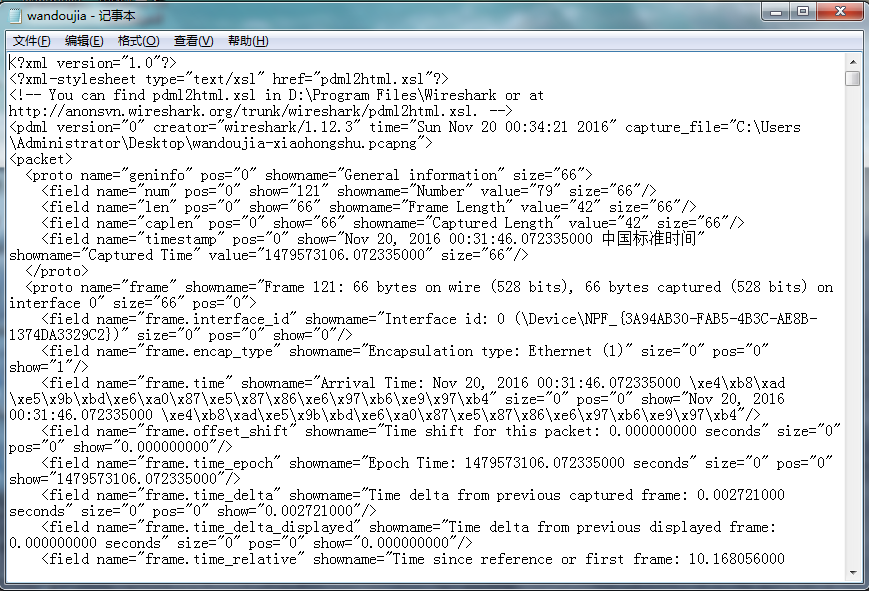
经过以上各方面的具体分析研究，接下来就是具体编程实现移动APP下载流量自动化抓取，本课题运用Python语言来实现。主程序是三个线程，第一个是Robotium程序模拟手势操作的APP下载线程，第二个是Tcpdump命令行抓包线程，第三个是监控进程数来判断Robotium程序是否完成。另外还有一个停止Tcpdump的子程序和删除APP的子程序，来实现自动化抓取移动APP下载流量。

## 5.4 流量规则特征的自动化提取

### 5.4.1 自动化提取流量规则的实现思路

Tcpdump抓包生成的数据包可以用Wireshark打开进行分析研究，Wireshark有一个follow tcp stream功能，这个功能很方便。美中不足的是提取出的stream 数据没有时间戳等其他信息，在分析数据的延时和丢包问题时就有些力不从心了。本课题用Python实现了一个简单follow tcp stream功能，同时保留了tcp信息。

原理很简单，仍然是基于Wireshark，其中有一个Export packet dissection as XML ‘pdml’ file。PDML 是一个用PHP写成的非正式的标记语言，类似于HTML。导出来之后的pdml文件内容如图5-2所示，对于pdml格式的文件用Python简单的做个xml文件解析，配合正则表达式，将数据提取出来就可以了。

图5-2 pdml文件格式

### 5.4.2 软件实现自动化提取流量规则

实际上数据包中这么多的数据不全是关心的对象，所关心的只有host和URI，其余的信息是多余的。所以利用xml文件解析，在pdml 文件中找到http协议下所有的host和URI，可以把规则的提取缩小范围。

发现在 pdml 文件中，每个标签<packet>...</packet>之间，就是一个数据包的 信息。在<field>...</field>中如果含有name=“http.host”，意味着这是一个 http协议的包，并且该标签中含有host，“show=”参数后的内容就是host。在<field>...</field>中含有 name="http.request.uri"，后面的“show=”参数后就是APP下载的URI。如图5-3所示：

图5-3 pdml文件分析host和URI

经过上面对抓取到的数据包转换成pdml文件格式后的分析研究，找到了host和URL的规律，那么，可以通过xml文件解析提取对应特征后面的需要的host和对应的URI信息，并且最后生成一个只含有host和URI的文件。

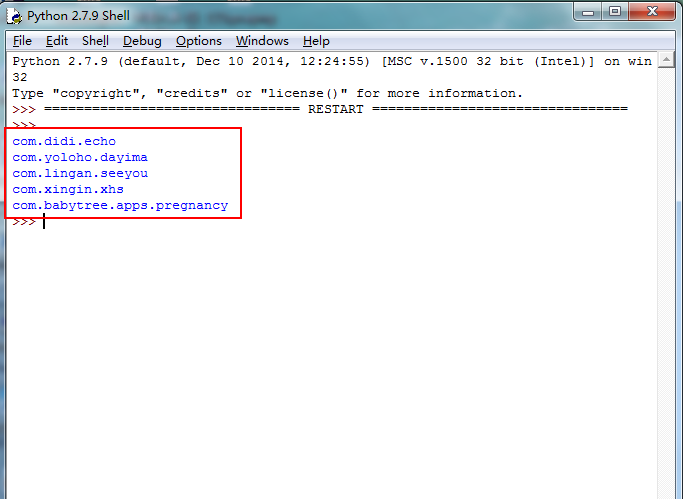
对于每个.pdml 格式文件的处理，都生成一个文件。在表格仅有两列，一列是 host，另一列是相对应的uri。相比之前巨大的信息量，把有用信息缩小到仅有十几行表格数据，对于host是能直接看出来的，但是uri还是需要人工进行筛选。因为不同的APP在http协议层上的流量特征是各不相同的，所以很难找到特定的规则可以帮助寻找所需要的特征。即使是这样，通过简要数据包筛选后，把原本 19.1M的PDML文件处理成3.19KB的表格文件，滤掉了99%的无用信息，大大减少了规则寻找时间。

最后，应用这个只有host和URI的文件，在DPI数据中编写MapReduce程序进行匹配，得到哪些用户曾经下载过这款APP，作为该APP用户行为的一个维度，进行用户行为分析，得到该APP的欢迎度，从而生成一份报告或者前端网站供投资方参考，再由投资方决定这款APP是否值得投资。

## 5.5 运行结果及分析

### 5.5.1 控制台输出信息分析

在5.3.3小节提到了，最后需要删除安装的APP，那么就需要判断哪些是新安装的APP。这就需要在未开始下载前用“adb shell pm list packages”命令将模拟器上所有的APP包名保存下来，然后等抓包结束后，再执行一次此命令获得安装后模拟器上所有APP包名，之后进行比较，得到哪些是新安装的APP，最后运行“adb uninstall”命令把新安装的APP删除。本课题中，在删除新安装的APP的时候把删除的APP包名输出在Python的控制台，这样根据包名就可以判断删除的是不是新安装的APP，起到验证的作用。控制台输出信息如图5-4所示：

图5-4 Python控制台输出删除的APP包名

### 5.5.2 Android模拟器运行效果

与4.4.3和4.4.4小节相同，自动化抓取移动APP下载流量也就是APP下载安装抓取下载流量的过程，模拟器上面依然会出现安装好的APP，并且通过adb命令可以获得新安装的APP包名。但是在本课题中，为了避免模拟器的存储空间不足，增加了一个新安装的APP删除过程，所以自动化抓取移动APP下载流量后模拟器恢复到没有进行此次操作之前的状态。

### 5.5.3 问题及调试方法

在本课题进行研究过程中遇到很多问题，以下是主要关键点和困难问题，以及其解决调试方法。

首先就是如何模拟人的手势操作。通过调研想到用自动化的测试框架实现模拟点击，先后尝试了几种框架，综合其优缺点最终选择Robotium来实现移动APP下载动作的模拟。

其次是网络流量的抓取，不能通过人来开启Wireshark或者其他抓包工具来抓取流量。通过调研，分析比较各种抓包工具的优缺点，选择Tcpdump命令行抓包工具实现抓取网络流量。

再次是如何将上述两者结合。一方面是自动化下载APP，另一方面是下载的同时抓取流量，这就是两个线程同时进行。另外，Tcpdump需要手动Ctrl+C停止，为了实现全部的自动化，需要用到杀死进程的命令来结束Tcpdump抓包。另外，还需要程序自己判断Robotium自动化下载是否结束，所以这就又引入一个检查进程数的线程。

最后，考虑到不管模拟器的存储空间多大，若APP数量很多，终有存储空间不足的情况出现，为了避免抓取一些APP的下载网络流量后手动删除已安装的APP，所以，这就需要APP下载这个动作完成抓取下载动作的流量后自动化删除这些安装的APP。但是，删除新安装的APP必须要判断哪些是新安装的APP，所以就需要在自动化下载前和下载后分别获得模拟器中的APP包名，再分析比较哪些是新安装的，之后进行删除。

## 5.6 本章小结

本章介绍了移动APP下载流量抓取及流量规则提取自动化。首先介绍了自动化抓取移动APP下载流量的实现思路，因为APP下载是自动化的，全程没有人工干预，所以抓取流量也需要自动化，不能通过人来开启Wireshark或者其他抓包工具来抓取流量。综合比较各种抓包工具，选择Tcpdump命令行工具抓取流量。之后介绍如何编写多线程程序，将自动化APP下载和抓取下载流量结合，实现移动APP下载流量抓取自动化。然后，介绍Tcpdump抓包后的文件导出pdml格式文件，编写Python程序实现流量规则提取自动化。最后是一些相应的运行结果分析。

# 第六章 总结及展望

## 6.1 本文总结

在信息化爆炸的今天，每天都有大量的 APP 上线，同时有大量的APP被淘汰。一次错误的投资和决策都将会造成巨大的损失，所以在这个信息化时代需要善于从大数据中提取到想要的信息来方便做出正确的决策。如果一个新生项目需要投资，对于投资者来说，获取的信息大部分是由项目内部的负责人提供。这些信息的真实性仍有待观察。而投资者又很难通过其它的途径了解到这个信息，这就需要一个第三方对于这些信息进行统计和验证，确保信息的准确性、客观性，从而保证投资者不会做出错误的决定。

一般某个APP的日常活跃用户数，每天新用户的增长量，或者各种用户的操作细节这些信息是掌握在公司内部的，属于商业机密，是不会或者很少公开的，然而可以通过合法的手段在海量的网络访问数据中作为第三方获取这些有用的信息，进而可以对APP的使用模式进行研究。

随着网络的快速发展和大数据时代的到来，对大规模网络流量数据的处理的需求也随之增长。一般海量的网络访问数据里面包含用户的各种基本信息、浏览过的网页或者曾经访问过哪些URL等信息。那么就可以利用海量的数据分析APP的使用模式，即获取哪些用户下载或者使用过某款APP，以此得到该APP的欢迎程度。

在用户通过APP在手机上同时操作时，客户端APP和对应的服务器端通过网络进行数据交换。在这个过程中，用户的特定行为也会产生特定的网络流量。然而不同的用户对于同一个APP的相同操作所产生的流量是具有固定相似的特征的，而这些特征是特有的，也就是其它流量中不会出现该特征。通过找到这些特征，再利用这些特征从海量的数据中筛选出符合此特征的流量，就能获取所需要的信息。要想从海量的网络访问数据中提取出APP的使用模式，首先需要得到这些APP下载或使用的情况下，客户端和服务器端的通信流量生成的规则。

当APP数量不多时，可以人工抓取APP使用模式下的网络流量。当APP数量非常多时，就需要考虑自动化的方法，实现APP使用模式下的网络流量抓取，从而节约人力和时间成本。

本文就在此研究背景下研究如何自动化抓取APP下载的网络流量。编写安卓自动化测试框架Robotium程序模拟人的手势操作实现APP的自动化下载，Tcpdump命令行工具抓包，然后编写Python多线程程序将两者结合实现移动APP下载流量抓取自动化。最后，编写Python程序自动化提取流量规则，缩小人为找寻流量规则的范围。

## 6.2 展望未来

基于网络访问数据的移动APP使用模式研究，不仅仅包含移动APP下载时所产生的网络流量，还应该包含APP在具体使用操作情况下所产生的网络流量，例如：注册、搜索、打开APP首页、打开APP内部其他页面、交易等。但是考虑到不同的APP有不同的控件名和控件ID，而Robotium是依据控件的ID进行定位模拟点击的，所以如果要实现自动化抓取APP使用操作产生的流量就需要找到一个可以代表所有或者大多数的APP的控件ID。虽然可以利用机器学习等算法训练出代表大多数APP的控件ID，但是，这些还需要更进一步的研究。

大数据之所以大，一个是因为大量的数据以及它们增长的速率之大；另一个就是在这些大量数据中所蕴含的信息价值之大。所以，大数据之所以被称为“大数据”，是因为它的数量和价值。而且，互联网不再是传统的桌面互联网，现在已经进入到移动互联网的时代，并且人们的许多相关的行为也与之前发生了很大的变化。谁能更好的从大数据中提取出有用的信息，谁就会具有更强大的竞争力。

# 参考文献

[1] Ghemawat S, Gobioff H, Leung S T. The Google file system[C]//ACM SIGOPS operating systems review. ACM, 2003, 37(5): 29-43.

[2] Dean J, Ghemawat S. MapReduce: simplified data processing on large clusters[J]. Communications of the ACM, 2004, 51(1): 107-113.

[3] Chang F, Dean J, Ghemawat S, et al. Bigtable: A distributed storage system for structured data[J]. ACM Transactions on Computer Systems (TOCS), 2006, 26(2): 4.

[4] Ng Y Y, Zhou H, Ji Z, et al. Which Android App Store Can be Trusted in China?[C]//Computer Software and Applications Conference (COMPSAC), 2014 IEEE 38th Annual. IEEE, 2014: 509-518.

[5] Harman M, Jia Y, Zhang Y. App store mining and analysis: MSR for app stores[C]//Proceedings of the 9th IEEE Working Conference on Mining Software Repositories. IEEE Press, 2012: 108-111.

[6] Gao Y, Zhang S, Wang S. A data collecting and processing method in high-speed network based on cluster[C]//Mechatronic Science, Electric Engineering and Computer (MEC), 2011 International Conference on. IEEE, 2011: 334-337.

[7] Wiratmadja I I, Govindaraju R, Athari N. The development of mobile internet technology acceptance model[C]//Management of Innovation and Technology (ICMIT), 2012 IEEE International Conference on. IEEE, 2012: 384-388.

[8] Huang W, Chen Z, Dong W, et al. Mobile Internet big data platform in China Unicom[J]. Tsinghua Science and Technology, 2014, 19(1): 95-101.

[9] Juan S, Shoulian T. Operator's Mobile Internet Strategy in the Process of Converged Network[C]//Management and Service Science (MASS), 2010 International Conference on. IEEE, 2010: 1-4.

[10] Lan Y, Tang Y, Zou S. Research and implementation of mobile internet services identification system[C]//Computer Science & Education (ICCSE), 2015 10th International Conference on. IEEE, 2015: 401-405.

[11] 朱建平, 章贵军, 刘晓葳. 大数据时代下数据分析理念的辨析[J]. 2014.

[12] 黄晓斌, 钟辉新. 大数据时代企业竞争情报研究的创新与发展[J]. 图书与情报, 2013 (6): 9-14.

[13] 陈光. 移动互联网及其应用[J]. 软件, 2012, 33(6): 133-134.

[14] 罗青林, 徐克付, 臧文羽, 等. Wireshark 环境下的网络协议解析与验证方法[J]. 计算机工程与设计, 2011, 32(3): 770-773.

[15] 狄佳玺. 移动互联网大规模流量数据分布式处理与流量分析[D].北京邮电大学,2015.

[16] 邢云鹏, 颜培. 移动数据流量的分析方法研究[J]. 中国信息界, 2012 (4): 38-42.

[17] 史辛宁, 汪颖, 张新程. 移动数据流量经营的现状分析与策略建议[J]. 2012 全国无线及移动通信学术大会论文集 (下), 2012.

[18] 徐健, 王涛. HTTP/1.1 的分析[J]. 西南师范大学学报: 自然科学版, 2004, 29(2): 315-319.

[19] 姜毅, 王兆青, 曹丽. 基于 HTTP 的实时信息传输方法[J]. 计算机工程与设计, 2008, 29(10): 2451-2453.

[20] 刘桂平. Robotium 对于安卓多内核浏览器自动化测试支持的研究[J]. 网络安全技术与应用, 2015 (5): 68-69.

[21] 祝阳阳, 侯永宏, 王宝亮. Android 自动化测试工具 Robotium 的应用与研究[J]. 信息技术, 2015, 39(10): 198-200.

[22] 刘创. 网络流量监控系统的设计与实现[D]. 北京邮电大学, 2013.

[23] 危蓉. 一种基于安卓系统的手机侧抓包分析方法[J]. 信息安全研究, 2016, 2(4): 339-342.

[24] 许坚. 计算机网络流量监控的设计与实现[D]. 电子科技大学, 2012.

[25] 朱莉, 曾慧, 代杰. 基于 Tcpdump 命令网络抓包脚本工具的设计[J]. 指挥信息系统与与技术, 2011, 2(3): 53-57.

[26] 段青玲, 魏芳芳, 张磊, 等. 基于 Web 数据的农业网络信息自动采集与分类系统[J]. Transactions of the Chinese Society of Agricultural Engineering, 2016, 32(12).

[27] 杨清龙. 基于网络日志的互联网用户行为分析[D]. 华中科技大学, 硕士学位论文, 2013.

[28] 刘森. 云计算技术的价值创造及作用机理研究[D]. 浙江大学, 2014.

[29] 王涛. HTTP 协议技术浅析[J]. 中国新技术新产品, 2013 (22): 14-14.

[30] 王珊, 陈松, 周明天. 网络流量分析系统的设计与实现[J]. 计算机工程与应用, 2009, 10.