**Andriod平台安全研究与防御**

**摘要：**Andriod操作系统是当前全球市场中占据份额最大、使用人数最多的智能终端操作系统。Andriod在系统不断升级，功能日趋完善，带给用户越来越好体验的同时，面对越来越多的安全威胁，严重威胁用户的财产等安全。威胁源自于Andriod系统漏洞，Andriod平台固有的特点，其平台开放性与相对较大的自由性都促使威胁漏洞的多元化。对高效可靠的Andriod系统安全加固措施，漏洞、恶意软件检测措施，安全防御机制成为系统开发商，智能终端开发商，安全公司以及开发人员投入大量精力和物力努力实现的目标。研究Andriod平台安全具有重要意义。

**关键字：**Andriod安全威胁；Andriod安全加固；漏洞检测；安全防御

**1 Andriod终端使用现状**

智能设备的发展，带来的不仅是更加便利的生活，更通过其自身的先进、丰富等带来了全民生活的革新。每一天，用户使用搭载各种操作系统的智能设备接入互联网查询资料、娱乐享受、支付所需、协助工作……智能设备同样可以根据用户需求定制各种专属功能。智能设备功能的日渐升级、稳定与多样化影响的是每一个用户再也不能脱离手机等智能设备。以手机为主的智能设备市场规模以及用户规模一直在快速增长。

Andriod智能手机在全球市场中对比其他众多手机操作系统中如IOS、BlackBerry、Windows中占据主体地位。IDC移动手机季度报告《Smartphone OS Market Share》[[[1]](#footnote-0)]，如图1.1显示，从图表中显示的2014年至2017年，Andriod智能手机市场份额一直绝对领先于IOS智能手机市场份额。

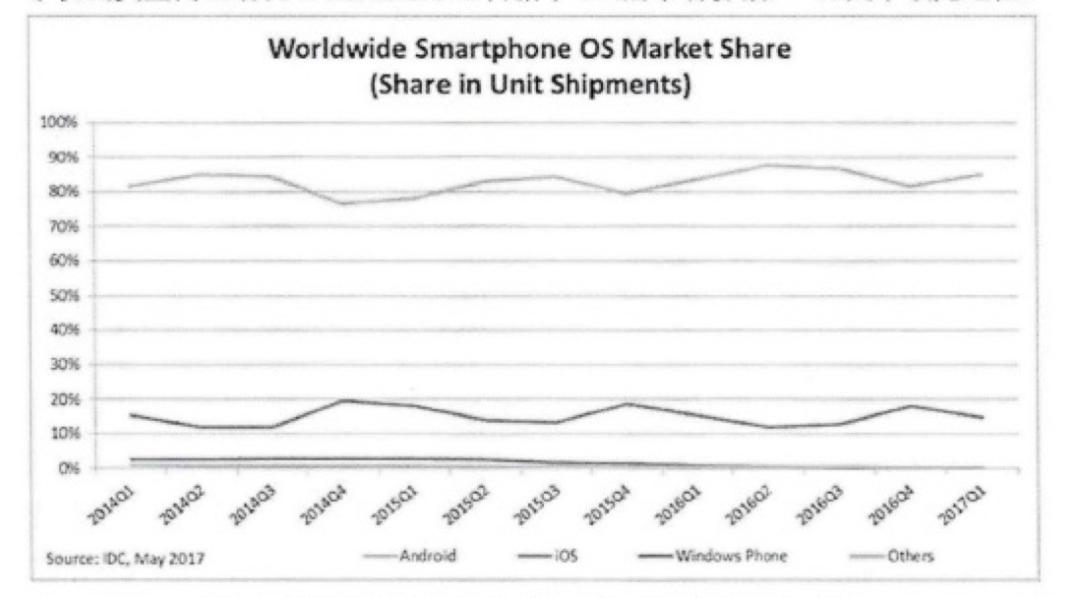


图1.1 全球智能手机操作系统市场占有率增长图（截止到2017年5月）

Gartner发布了2017年智能手机操作系统的市场份额数据报告[[[2]](#footnote-1)]，如图1.2。报告显示，99.9%的智能手机系统都是基于Android和iOS，且Android和iOS系统的份额占比分别为86%和14%。Gartner的数据再次显示了Andriod作为智能手机操作系统占据众多智能手机操作系统的绝对市场比重。



图1.2 2017年全球智能手机用户销售份额

RESEARCH AND MARKETS 2018-2022全球智能手机市场预测《Worldwide Smartphone Market Forecast, 2018-2022》[[[3]](#footnote-2)]对2018-2022年智能手机的市场情况进行了预测，该报告发现2017年全球手机市场规模达到19.6亿部，与2016年类似。尽管2017年中国智能手机出货量首次出现下滑，但全球智能手机市场仍然在2017年增长3.4％，占手机市场总量的78.4％。最重要的是，2018年智能手机的全球份额预计将达到80％。并且，研究人员预测，自2020年以来，5G也会使全球智能手机市场受益，智能手机全球销量将在2022年达到约19亿台，2017至2022年期间CAGR （复合年增长率）为4.1％。

Andriod智能终端依托其开放性，高硬件兼容性，强跨平台性，高性价比，应用丰富性等特点促进了用户的选择。Google与开发人员的不干涉方式为应用程序可能做的事情提供了极大的自由，允许在Google的主要应用程序市场Google Play中增加多种类别的应用程序。Andriod成为全球智能终端操作系统市场占有率最高、应用最广的操作系统。

**2 Andriod安全威胁**

**2.1 Andriod威胁现状**

随着以Andriod操作系统为主的智能手机的普及，以Andriod操作系统为主的智能手机越来越广泛地渗入人们的生活、带给人们生活极大便利，甚至改变人们的生活工作学习方式，给人类社会带来质变……正是由于其影响之广之细微，除了人本身这个因素影响人们依赖于智能手机的生活安全质量，Andriod等智能手机带给用户智能功能的软件也决定着用户的生活安全质量。可靠性和安全性成为用户关注的要素，也成为三星、华为、苹果、OPPO等智能手机终端厂商和安全公司关注的重点。

Andriod智能手机面临的极大威胁，根据第三方统计机构对CVE漏洞数据库中的历年Android系统的漏洞数据统计如图2.1所示[[[4]](#footnote-3)] 。Andriod操作系统其固有的特点，Android操作系统目前仍未非常完善的修补系统漏洞的补丁机制，Android系统存在严重的碎片化问题。Andriod操作系统相对于其他移动终端操作系统较大的自由性也促使恶意开发人员有更多的漏洞可以利用。2018年中国手机安全生态报告[[[5]](#footnote-4)]统计显示99.9%的手机设备存在远程攻击漏洞，半数以上的手机设备存在20个以上的安全漏洞。99.3%的APP存在已知安全漏洞，89.6%的APP存在高危安全漏洞。其中，游戏娱乐类APP含有高风险漏洞的比例最高，为95.9%。安全漏洞涉及恶意程序，骚扰诈骗，手机盗版软件、羊毛党、代刷软件与免流软件等黑灰产业，用户隐私。

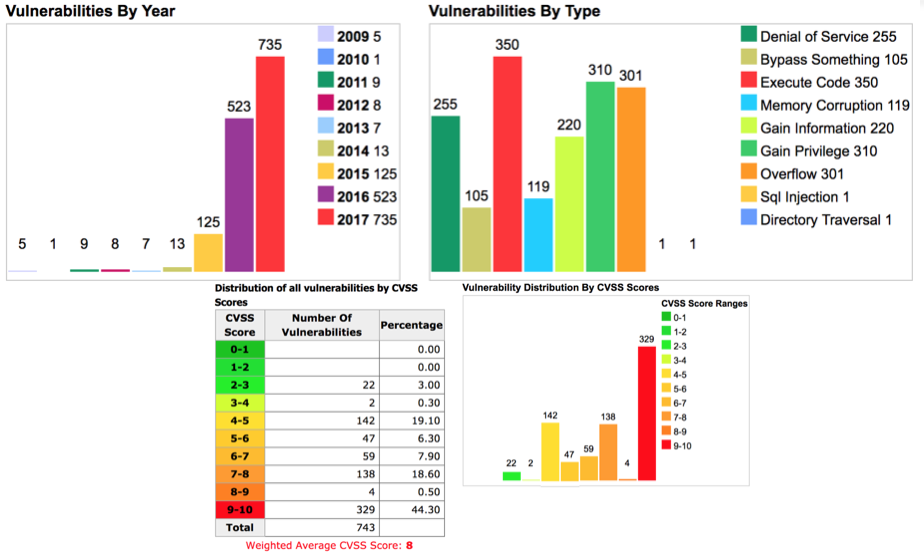


图2.1 历年Android系统的漏洞数据统计

恶意程序急剧增多。2018年，Check Point安全研究员贺飞翔跟安全团队的其它两位同事一起，在3月14号发布了一个重磅消息，就是找到了RottenSys（堕落的系统）移动端的病毒。截止今年 3 月 12 日累计受感染安卓手机总量高达 496 万 4 千余部；受感染的手机中，每天约有 35 万部轮番受到恶意广告推送的侵害。荣耀、华为、小米、OPPO等均受感染。仅 3 月 3 日到 12 日 10 天期间，RottenSys 团伙向受害手机用户强行推送了 1325 万余次广告展示，诱导获得了 54 万余次广告点击。保守估计不正当广告收入约为 72 万人民币。2017年，安全公司Palo Alto Networks报告了一款在中国流行且针对中国用户的新型Andriod恶意应用SpyDealer[[[6]](#footnote-5)],该恶意应用通过被黑客控制的无线网进行传播，一旦用户连接被控制流行应用软件的无线网络便被感染，这个恶意应用能够从微信、QQ、微博等40余款流向应用软件中窃取用户敏感信息，给用户造成严重的隐私损失。防病毒软件公司G Data对近六年Andriod平台恶意应用软件增长情况进行了统计[[[7]](#footnote-6)]，如图2.2，从2012年到2017年，Andriod平台恶意应用软件从21万新兴恶意应用软件增长至接近350万个，尤其从2012年-2013年，2015年-2016年，增长的恶意应用数量超过50万，增长速度惊人。 以Andriod为主的移动终端恶意应用软件有资源消耗、流氓行为、隐私窃取、恶意扣费、远程控制、系统破坏、诱骗欺诈和容易传播共八类。就《2017年中国手机安全生态报告》显示，Andriod目前恶意应用状况其中占比最高的恶意应用是资源消耗类，其次为流氓行为。数量众多，种类繁多的恶意软件时刻在威胁着Andriod智能终端的安全性。

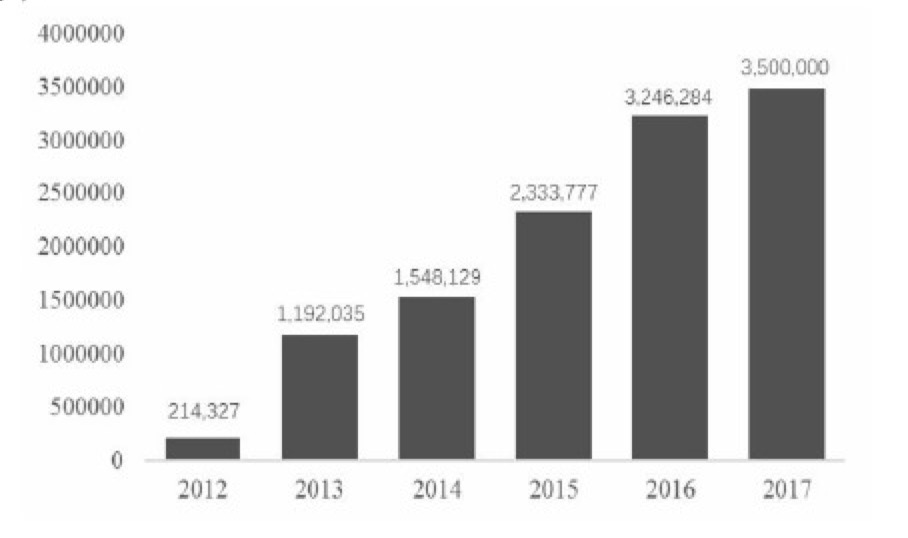


图2.2 2012年-2017年Andriod平台恶意应用增长情况

骚扰诈骗时刻发生。2018年上半年，用户通过360手机卫士标记各类骚扰电话号码数量约2943.7万个，平均每天被用户标记的各类骚扰电话号码约16.3万个。从拦截量上看，360手机卫士共为全国用户识别和拦截各类骚扰电话194.6亿次，平均每天识别和拦截骚扰电话1.1亿次。360手机先赔共接到手机诈骗举报4173起，涉案总金额高达1221.1万元。

黑灰产业猖獗发展。利用手机从事违法犯罪活动的黑灰产业多种多样，如：手机盗版软件、羊毛党、代刷软件与免流软件等。2018年上半年，360互联网安全中心共监测到活跃的盗版移动APP样本238.3万个。其中，盗版系统工具类软件最多，占21.5%。其次为盗版游戏娱乐类软件，占21.0%，影音图像类排名第三占12.0%。

用户隐私安全难保。对2018年上半年市场上流行的2160个移动应用软件的检测分析显示，当前APP权限越界行为整体呈增长趋势，尤其是在一些隐私相关的权限使用上。其中，写入通话记录权限被滥用的情况最为严重。

随着越来越多系统漏洞不断被披露，目前Android智能手机漏洞百出。手机安全加固，防御措施以及安全软件虽然可以在一定程度上解决安全问题，但不能根本且有效地修补系统中的漏洞，恶意攻击依旧在不断发生，恶意应用数量依旧快速增长。Andriod安全保护的机制之一就是基于权限，方便大多数用户且可以与用户直接交互的大多数安全软件却不能获得Andriod最高的系统权限，且安全软件即使获得系统权限又会引来众多相应的安全问题，因而Android系统安全问题一直非常棘手。

**2.2 Andriod安全攻击**

Andriod漏洞是Andriod安全威胁的根本来源，Andriod操作系统其固有的特点，Android操作系统目前仍未非常完善的修补系统漏洞的补丁机制，Android系统严重的碎片化，Andriod操作系统相对于其他移动终端操作系统较大的自由性，Andriod平台的开放性也是Andriod安全威胁的来源。Andriod智能终端安全威胁与日俱增，Andriod智能终端用户隐忧愈发明显，基于Andriod漏洞实现各种恶意攻击，如通过对应用程序进行静态分析或者代码级别动态分析实现静态攻击和动态攻击。

**2.2.1 静态分析**

静态分析的整个分析过程无需运行程序代码，只通过静态对Andriod应用程序分析就可以获得程序源代码，它无需改写Android系统源码，速度快、轻量级。

常用的静态分析工具包括apktool、ApkIDE、dex2jar等。apktool是GOOGLE提供的APK编译工具，能够反编译APK并安装反编译系统APK所需要的framework-res框架。常用的静态分析方法包括基于权限的检测工具Kirin、CoChecker、Chex、SCanDroid、DroidChecker、AndroidLeaks、Epicc等。其中Kirin是早期具有代表性的静态分析方法之一，它通过定义的一组规则识别危险的权限组合；CoChecker 通过识别两类基于Android API调用泄漏途径即权能泄露路径和敏感数据泄露路径进而进行ICC分析；Chex是基于组件的进行静态分析；ScanDal以Dalvik虚拟机字节码为输入，通过检测Android应用程序的私密信息泄露进行静态分析；SCanDroid从应用程序的AndroidManifest.xml清单文件中提取应用程序权限，通过分析数据流，自动检测应用的数据流是否与这些权限一致法进行静态分析；DroidChecker通过使用控制流图搜索和脏数据分析方法自动分析可能的敏感数据泄露路径进行静态分析。

静态分析方法众多，对Andriod的应用程序进行分析主要通过反编译工具如apktool将APK反编译为Smali字节码进而分析和把Andriod应用程序核心文件DEX转换为Java源代码进而分析两种方法。信息反馈法、特征函数法、顺序查看法是代码定位方法，进一步辅助静态分析。

静态分析避免了分析程序运行时代码，效率高、速度快，但是静态分析无法真实模拟程序的动态运行无法，存在误报率高的缺陷，Kirin、CoChecker等代表性静态分析方法就有误报率高的问题。

**2.2.2 动态分析**

动态分析是一种程序运行时的分析方法，可以在源码级定位应用程序错误位置。在一般情形下，Andriod动态分析具有较高的精确度，能够通过动态方法的拓展较准确定位并分析，但需要改写和重新定制Android系统，同时需要通过更新ROM安装这种定制系统，因此是重量级的。基于对Andriod系统的重新定制与改写，又在在原生Android系统中加入可以实时监视数据流向的监视器，检测调用危险函数时所需权限等实现动态分析。

常用的动态分析调试工具包括AndBug、IDApro、Inspeckage和DroidBox等。AndBugAndBug是针对Android平台的Dalvik虚拟机的调试器，面向逆向工程师和开发人员，使用与Android的Eclipse调试插件，Java调试线协议（JDWP）和Dalvik调试监视器（DDM）相同的接口，以允许用户挂钩Dalvik方法，检查进程状态，甚至执行更改；IDApro是交互式通过递归下降实现的反汇编工具；Inspeckage是一种基于Xposed 开发的应用动态分析工具，可以监控Shared Preferences数据，绕过SSL使用HOOK过滤出很多有价值的数据；Droidbox是一种Android病毒的动态分析工具，通过监控网络数据，文件读写操作，网络、文件、短信泄露的信息，发送短信和打电话，启动服务和列出通过DexClassLoader加载的类等直观表现病毒的行为……日志输出法，栈跟踪法，剖析法等用于在动态分析中实现代码定位。

基于静态分析和动态分析实现静态攻击与动态攻击。

**2.2.3 攻击方式**

利用不同的Andriod系统漏洞实现不同方式的攻击。常见主要的攻击方式包括提权攻击、中间人攻击、恶意软件、恶意软件变形技术等。

**2.2.3.1 提权攻击**

提权攻击即提升应用程序的权限，从而访问原本不能访问的资源文件。Davi等人给出了下述定义：“一个拥有较低(较少)权限的应用程序访问拥有较高(较多)权限的组件而不受任何限制，称为Android系统的提权攻击”。Android提权攻击使具有不同权限集的应用可以彼此通信，那么恶意应用就有可能具有非法使用某些正常应用程序的权限。

**2.2.3.2中间人攻击**

中间人(Man-In-The-Middle,MITM)攻击通过劫持局域网中被攻击机器和服务器间的对话，并交换其所收到的数据，使通讯的两端误认为他们正在与另一方直接秘密通信，但事实上整个会话过程都被攻击者这个中间人完全控制。常见的中间人攻击包括基于Https漏洞的HTTPS中间人攻击。

**2.2.3.3 恶意软件**

恶意软件是各种漏洞利用、威胁组成软件的统称，其成为影响Andriod安全的重要因素。Zimperlich、Exploid和RageAgainstTheCage是最为著名的三大恶意软件，其他许多恶意软件也都是在这三大恶意软件的基础上开发的。资费消耗类恶意软件是恶意软件的主要类型，该类恶意软件以推销广告、消耗流量等手段，增加手机用户的流量资费，谋取不法商家的经济利益。随着Andriod安全技术的不断提升，恶意软件技术也在不断升级，对用户造成的威胁越来越大，损失不尽相同。针对系统运行库的攻击、利用Telegram软件协议木马的攻击、挖矿木马、僵尸网络发起DDOS攻击、针对企业内网安全的攻击、多种分发渠道Chamois恶意家族、应用多开技术的滥用、APT攻击（Advanced Persistent Threat，高级持续性威胁）都成为恶意软件攻击的新技术。

**2.2.3.4恶意软件变形技术**

恶意软件变形技术是为规避恶意软件检测器和分类器对恶意软件的自动检测，使恶意软件发挥作用实现攻击。恶意软件采用了一系列包括混淆、代码加密、密钥代换、置换、动态加载、代码反射和本地代码执行等自我保护的变形隐身(Stealth)技术。恶意软件变形包括可通过静态分析发现的重打包、重命名标识符、数据加密、代码重排序、无用码插入、组合变形技术等，通过动态分析可以发现代码反射与字节码加密等变形技术。恶意软件配合恶意软件变形技术使得Andriod面临越来越多的威胁，其防御威胁攻击的技术也需要不断更新。

**3 Andriod安全机制**

使用最为广泛的Andriod智能终端已出现大量安全漏洞，智能终端安全漏洞是手机安全的根本威胁，安全漏洞也自然成为了用户、智能终端生产厂商和开发人员等关注的重点。Andriod现有的安全机制源于其Linux底层内核。Andriod不同层次中具有不同的安全机制，另外还有内存管理机制，进程间具有通信机制如图3.1，其Linux内核层，Andriod Runtime，应用程序框架层，应用层分别具有用户权限机制，沙箱机制，签名机制和权限控制机制。Andriod应用于不同部分、层次的安全机制在一定程度上解决某些固有漏洞，遏制某些特定攻击。

|  |  |
| --- | --- |
| 应用层：权限控制机制  （Application） | |
| 应用程序框架层：签名机制  （Application Framework） | |
| 系统类库  （Libraries） | Andriod Runtime：沙箱机制  （核心库+Dalvik/ALT虚拟机） |
| 硬件抽象层  （HAL） | |
| Linux内核层：用户权限机制  （Linux Kernel） | |

图3.1 应用于Andriod系统体系结构安全机制

**3.1用户权限机制**

Andriod具有Linux内核层，固然匹配有与Linux相同的安全机制。Linux的用户权限机制应用于Andriod，将Andriod系统中的每一个应用程序看成一个独立的用户，安装程序时为其分配唯一的ID号，即UID。

Linux内核以用户和用户组的概念为基础，对不同的用户资源进行隔离，通常运用UID即用户名和用户、GID即组标识分别区别表示不同的用户、用户组，需要知道，同一个用户可以被包含在不同用户组之中。每一个用户对于文件具有不同的权限，具体表现为对不同文件的可进行的不同操作，包括可读即read访问查看权限、可写即write修改权限、可执行即execute执行脚本文件权限，用三元组rwx中的单个字母标示出不同用户对文件的不同操作权限。不同的用户组对于文件受同于用户对于文件操作的权限设置，除了用户、用户组，Linux内核中的其他用户也需要按照权限三元组rwx规定的权限规则操作文件。Andriod系统内核应用Linux用户权限机制，其系统文件的所用者是root用户，具有最高系统权限，只有具有相应权限的应用程序（用户）才可以对系统文件进行访问。Andriod在Linux内核提供的基于UID和GID的安全机制的基础上，又实现了一套称为Permission的安全机制，如图3.2。不同用户拥有不同权限在一定程度上避免了安全漏洞，使得应用Linux内核的Andriod安全性和稳定性提高。

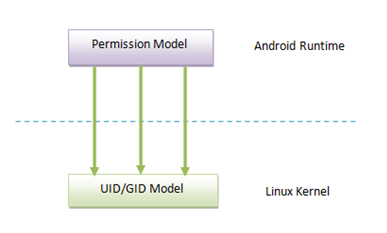


图3.2 Linux的UID/GID安全机制与Android的Permission安全机制

**3.2 沙箱机制**

沙箱隔离机制是Andriod应用于Andriod Runtime充分利用其Linux内核层用户权限机制的延伸，在确保用户权限机制的条件下，进一步提高Andriod的安全性、可靠性和稳定性。如图3.3，沙箱机制对具有不同ID的用户即应用进程进行隔离，并且通过禁止或允许应用进程对系统资源文件的访问实现权限的控制。通过不同的ID区别不同的应用程序，并限制不同的应用程序不能访问权限不匹配的系统资源或者其他应用程序，通过AndriodManifest.xml清单配置权限和共享代码、数据实现Andriod对其他资源的访问。



图3.3 Andriod应用程序沙箱机制

沙箱机制在一定程度上加强了用户权限机制又通过隔离提高不同进程运行的独立性，从而提高安全性与稳定性。

**3.3 签名机制**

签名机制应用于应用程序框架层，通过开发应用程序的唯一标识数字证书即签名提高程序的安全性。Andriod具有两套签名机制，Andriod APK签名使用用户的私钥对应用程序进行签名，ROM签名对Andriod ROM包进行签名。在安装应用程序的过程中，第一步需要检查是否存在有效证书，只有具有有效证书的应用程序才会被安装，用户有时也会根据对“该应用证书非法，是否继续安装？”提示信息做出“继续安装”的选择安装应用程序；在升级应用程序的过程中，系统首先会检查证书的签名是否相同，只有相同才会安装应用程序实现应用程序的升级。

签名机制验证相同签名、合法有效的数字证书，大量减少了恶意程序造成的严重后果，提高了Andriod的安全性。

**3.4 权限控制机制**

权限控制机制应用于应用层，不同于Linux内核的用户权限机制，却因为都是对于权限的控制具有相似之处。Andriod通过权限控制机制控制应用操作，Andriod应用程序开发时对于AndriodManifest.xml的配置说明了不同应用程序所需要的不同权限。当应用程序安装在智能终端时，会询问系统以获得自己具有的权限，只有经过系统用户允许后获得相应权限才可以访问相应的资源。例如安装微信会询问系统要求获得访问内存、访问通讯录等权限，只有系统用户允许应用程序才能够获得权限保证其各项功能的正常使用。Andriod应用权限具有signatureorsystem、signatures、dangerous、normal四个保护级别，依据不同级别限制应用程序不同权限申请与操作。

Andriod应用层权限控制机制最直接与用户交互，让用户切实感受到权限控制的存在与意义。Andriod智能终端生产厂商、应用商店也会基于应用层的权限控制机制来保护限制和允许访问设备功能[[[8]](#footnote-7)]，有效抵御直接破坏Android系统规则的病毒。

**3.5 内存管理机制**

Android系统是基于Linux内核开发的操作系统。Android操作系统在Linux系统其独到的内存管理机制，即会在进程活动停止后结束该进程的基础上优化了内存管理机制。包括分配机制和回收机制的Andriod内存管理机制会把进程都保存在内存中，并且为每个一个独立的应用建立一个对应独立的虚拟机。进程关闭并不立即释放进程空间，当系统的内存达到阈值即上限，需要更多内存时，才释放部分进程。这些被保存在内存中的进程，并不会影响系统的运行速度，相反，在再次打开这些进程时，进程启动的速度速度会大大提升。

Andriod内存管理机制具有对比于Linux系统内存管理机制的优势，可靠性大大提升，同时也带给用户使用Andriod智能终端的良好体验。

**3.6 通信机制**

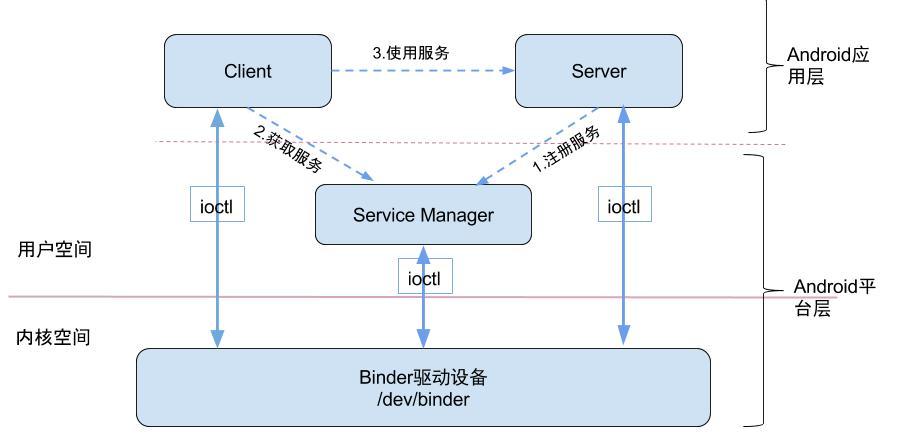
Andriod采用Binder通信机制控制进程间通讯。如图3.4，Binder通信采用C/S架构，从组件视角来说，包含使用服务的Client，提供服务的Server，将Binder名字从字符形式转化成Client中对该Binder的引用，实现对Server中Binder实体的引用，管理系统中的各种服务的ServiceManager以及建立进程间通信、在进程间传递和交互binder的binder驱动。Client、Service、ServiceManager通过与Binder驱动进行交互间接实现IPC通讯。

图3.4 Binder通讯架构

Binder中包含的注册服务（addService）、获取服务（getService）、使用服务（useService)都符合C/S架构，即都包括Client端和Server端。注册服务即Service进程注册Service到ServiceManager，此过程种Server是客户端，ServiceManager是服务端。获取服即Client进程使用某个Service前向ServiceManager中获取相应的Service，此过程中Client是客户端，ServiceManager是服务端。使用服务即Client根据得到的Service信息建立与Service所在的Server进程通信的通路实现直接与Service交互，此过程中client是客户端，server是服务端。

Andriod使用Binder机制实现进程间通讯提高了安全性。传统的进程通信方式并没有严格的验证通信双方的身份，比如Socket通信中通信双方的ip地址是客户端手动填入，这就很容易被恶意利用进行伪造从而攻击，产生安全问题，但是Binder通讯机制却通过从协议本身支持对通信双方做身份校检极大程度解决了通信双方伪造身份问题，基于身份校验使得安全性大大提高。

**4 Andriod漏洞、恶意应用检测**

**4.1 漏洞检测技术**

影响Andriod安全的根本是系统漏洞，因此对于漏洞的检测技术也就成为智能终端、软件开发商，安全公司等投入大量物力与精力攻克解决的重点。现有的漏洞检测技术种类繁多，常见漏洞依据文献[[[9]](#footnote-8)]分类为组件权限暴露漏洞、WebView远程代码执行漏洞、弱随机数漏洞、Intent注入漏洞、SQL注入漏洞、敏感数据泄露漏洞、运行时动态代码加载漏洞。对于种类繁多的漏洞，常用的检测方法从应用安全、源码安全及数据安全方面对用户管理、版本升级、界面劫持、动态调试、进程保护、程序完整性、数据储存、数据传输、业务逻辑安全、系统环境安全等内容进行静态、动态及人工分析等角度对移动应用做更全面的检测。通常情况只能检测已有特定漏洞，其结合动静态分析方法，通过被提取漏洞的特征，寻找匹配项从而定义漏洞。爱内测app的功能是检测Andriod平台应用软件APK漏洞，其“1、程序机密性检测：有代码混淆检测、DEX保护监测、so保护监测、程序签名检测、完整性校验、权限管理检测；2、组件安全检测：有Activity安全、Broadcast Receiver安全、Service安全、Content Provider安全、Intent安全、WebView安全；3、数据安全检测：有调试信息、输入检查、数据传输完整性、远程数据通讯协议、证书验证、数据访问控制、重放攻击、会话安全；4、业务安全检测：包括用户登录、密码管理、支付安全、身份认证、超时设置；5、其他检测：模拟器检测、键盘记录、界面劫持、进程保护、动态调试。”五大漏洞检测方法在一定程度上总结描述了Andrio漏洞检测的一般方法。

**4.1.1 漏洞检测一般流程**

Andriod虽然针对不同的漏洞具有不同的检测方法，但是可以从大量检测方法中总结出一般方法步骤，如图4.1[[[10]](#footnote-9)]Andriod的漏洞检测的一般流程为：

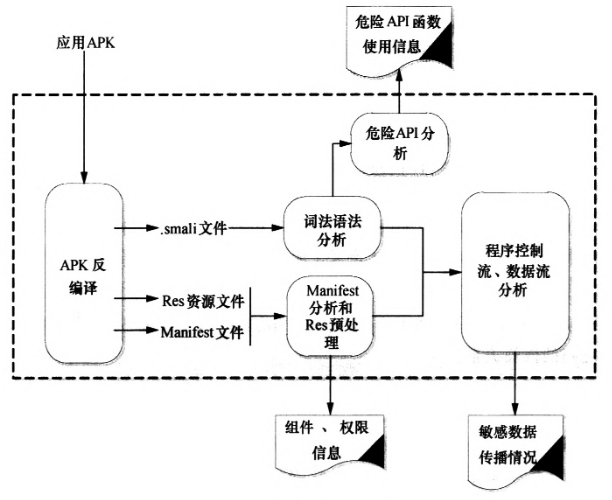


图4.1 Andriod APK静态分析流程

第一步，反编译APK文件,得到应用的smali代码文件、resources资源文件和AndroidManifest．xml文件；

第二步，对得到的Manifest文件和资源文件进行解析,获取应用的权限申请、组件声明和回调函数定义等相关信息；

第三步，对得到的smali文件进行语法和词法分析,构建对应的smali类和方 法；

第四步，对smali类和方法进行危险函数的扫描和分析,获取应用关于危险函数调用的信息；

第五步，采用程序控制流和数据流分析的方式对程序代码进行分析,获取应用关于敏感数据的使用和传播情况。

**4.2 恶意应用检测技术**

基于用户权保护限制和允许设备功能只能在一定程度上针对解决安全漏洞，但却无法阻止利用权限的Android恶意软件。实现对Andriod恶意软件的防御首先要实现对恶意应用软件的检测技术。Andriod恶意应用检测技术主要包括基于权限的检测方法、基于签名的检测方法以及基于机器学习的检测方法这三种。基于权限的检测方法主要用于风险评估，要求分析人员对于权限系统具有充分完备的知识；基于签名的检测方法通过使用字节码和应用程序接口API调用中的特定模式识别恶意应用；基于机器学习的检测方法可以通过综合使用动态和静态分析方法总结恶意与良性应用程序各自的特征，从而实现识别种类繁多的恶意应用软件。

**5 Andriod安全防御**

Andriod安全防御同样是Andriod系统开发生产的关键。Andriod开发会出现不同的漏洞，即使针对特定漏洞、特定攻击研发出解决系统，但又会出现新的漏洞、新的攻击，Andriod安全防御本就是持续对抗的过程，需要跟进新漏洞，不段针对解决，再不断升级，发布补丁提高用户端Andriod系统下应用程序的安全性，尽量减少Andriod的固有特性、较大自由性、开放性引起的众多安全问题。Andriod的安全防御也可以分为静态防御与动态防御两部分。

**5.1 静态防御技术**

静态防御技术应对Andriod静态攻击，使得应用程序在被反编译后难以进行静态攻击。静态防御技术主要包括代码混淆、数字签名、软件加密、SO文件加密、应用加壳、SO文件混淆、加壳技术等。

Andriod加壳技术，如图5.1，加壳过程需要待加密的APK即源APK、核心APK，壳程序APK即负责解密APK工作，加密工具即将源APK进行加密并和解压提取后的壳dex合并形成新的dex。第一步，拿到需要加密的核心APK和壳程序APK；第二步，用加密算法对核心APK进行加密得到加密后的APK并与将壳APK进行解压提取后的dex合并得到新的dex文件；第三步，对壳程序中的dex文件进行替换，并重新打包、签名得到新的APK。这个新的APK即为脱壳程序APK。当然，经过一系列处理的新的APK不同于原始APK的工作，它主要负责解密源APK，加载原始APK并让其正常运行起来。

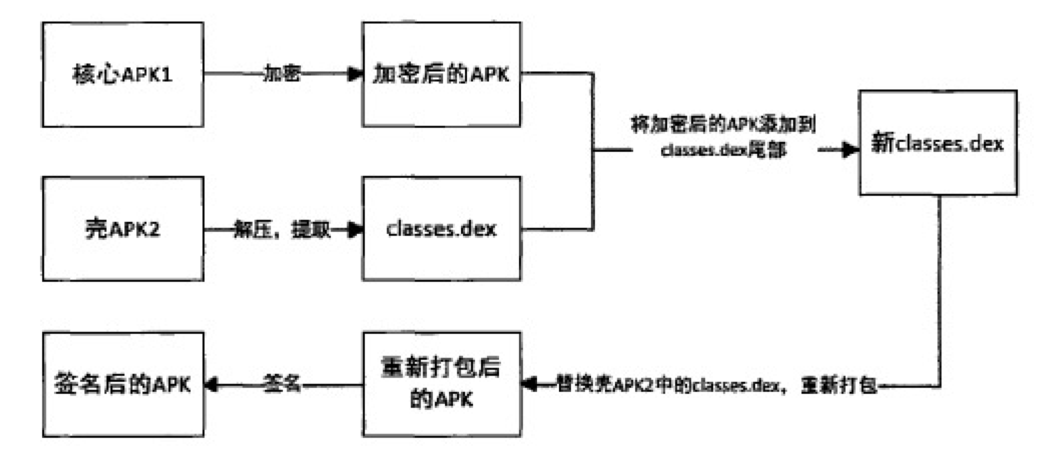


图5.1 Andriod APK加壳机制

Andriod APK加壳技术能够很好避免对于应用程序的反编译，加强Andriod应用程序的安全性。

**5.2 动态防御技术**

动态防御技术应对Andriod动态攻击，使得应用程序难以被动态跟踪调试。动态防御技术依据主要的可以进行的动态调试主要包括防PTRACE动态调试技术、防JDWP动态调试技术等[[[11]](#footnote-10)]。

**5.2.1 防JDWP动态调试技术**

JavaTM调试线协议（JDWP）是用于调试器与其调试的Java虚拟机（VM）之间的通信的协议。防JDWP动态调试技术即防御的对象为JDWP动态调试。JDWP动态调试需要运用JDWP调试工具进行调试，防止JDWP动态调试主要通过代码级别进行调试状态检查和端口监听，步骤为：第一步，在应用程序的代码中设置`andriod:debuggable=false`，禁止Andriod动态调试；第二步，不断在应用程序运行的过程中检查andriod:debuggable的值是否为false，在被修改可以进行动态调试的时候立即退出该应用程序。

**6 总结**

Andriod操作系统是当前全球市场中占据份额最大、使用人数最多的智能终端操作系统。Andriod在系统不断升级，功能日趋完善，带给用户越来越好体验的同时，面对越来越多的安全威胁，严重威胁用户的财产等安全。威胁源自于Andriod系统漏洞，Andriod平台固有的特点，其平台开放性与相对较大的自由性都促使威胁漏洞的多元化。高效可靠的Andriod系统安全加固措施，漏洞、恶意软件检测措施，安全防御机制成为系统开发商，智能终端开发商，安全公司以及开发人员投入大量精力和物力努力实现的目标。Andriod系统不断发展，现有漏洞被解决，新的漏洞出现又要求升级对应的防御技术，Andriod安全对抗一直在进行。

1. [] IDC. Smartphone OS Market Share, 2017 Q1[OL]

   <https://wwwidc.com/promoy/smartphone-market-share>/os ,2017-12-09. [↑](#footnote-ref-0)
2. [] Gartner (2017) Worldwide Smartphone Sales to End Users by Operating System.

   http://www.gartner.com/newsroom/id/3725117 [↑](#footnote-ref-1)
3. [] RESEARCH AND MARKET. Worldwide Smartphone Market Forecast, 2018-2022 https://www.researchandmarkets.com/reports/4471265/worldwide-smartphone-market-forecast-2018-2022 [↑](#footnote-ref-2)
4. [] CVE Details 历年Android系统的漏洞数据统计 https://www.cvedetails.com/product/19997/Google-Android.html?vendor\_id=1224 [↑](#footnote-ref-3)
5. [] 360互联网安全中心 2018中国手机安全生态报告 http://zt.360.cn/1101061855.php?dtid=1101061451&did=491398428 [↑](#footnote-ref-4)
6. [] Spydealer Andriod Trojan Spying on More Than 40 Apps[EB/OL] <https://researchcenter.paloaltonet> wokers.com/2017/07/unit42-spydealer-andriod-trojan-spying-40-apps/2017-07-06 [↑](#footnote-ref-5)
7. [] G DATA Mobile Malware Report[EB/OL] <https://www.gdatasoftware.com/mobile-device-management>. 2017-04-12. [↑](#footnote-ref-6)
8. [] Oberheide, J. and Miller, C. (2012) Dissecting the Android Bouncer. [↑](#footnote-ref-7)
9. [] 董国伟, 王眉林, 邵帅,等. 基于特征匹配的Android应用漏洞分析框架[J]. 清华大学学报(自然科学版), 2016(5):461-467. [↑](#footnote-ref-8)
10. [] 董国伟, 王眉林, 邵帅,等. 基于特征匹配的Android应用漏洞分析框架[J]. 清华大学学报(自然科学版), 2016(5):461-467. [↑](#footnote-ref-9)
11. [] 牛豪飞. Andriod应用保护方案的设计和实现 北京邮电大学. 2018(1):35-38 [↑](#footnote-ref-10)