(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 104517054 A
(43)申请公布日 2015.04.15

(21)申请号 201410822965.9

(22)申请日 2014.12.25

(71)申请人 北京奇虎科技有限公司
地址 100088 北京市西城区新街口外大街28号D座112室(德胜园区)
申请人 奇智软件(北京)有限公司

(72)发明人 李伟

(74)专利代理机构 北京市隆安律师事务所 11323
代理人 权鲜枝 何立春

(51)Int.Cl.
G06F 21/56(2013.01)

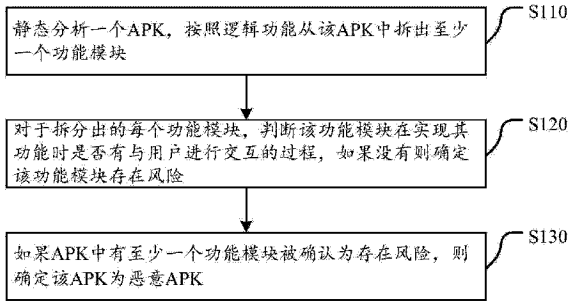
权利要求书1页 说明书12页 附图2页

(54)发明名称

一种检测恶意APK的方法、装置、客户端和服务器

(57)摘要

本发明公开了一种检测恶意APK的方法、装置、客户端和服务器，该方法包括：静态分析一个APK，按照逻辑功能从所述APK中拆出至少一个功能模块；对于拆分出的每个功能模块，判断该功能模块在实现其功能时是否有与用户进行交互的过程，如果没有则确定该功能模块存在风险；如果所述APK中有至少一个功能模块被确认为存在风险，则确定该APK为恶意APK。本发明提供的技术方案基于恶意APK通常在用户不知情的情况下进行操作的规律，通过判断APK的各功能模块是否与用户进行交互，确定该APK的安全级别。提供了一种具有启发式的、直观的检测恶意APK的方法，可以实现对未知APK的检测。



CN 104517054 A

1. 一种检测恶意 APK 的方法，其中，该方法包括：

静态分析一个 APK，按照逻辑功能从所述 APK 中拆出至少一个功能模块；

对于拆分出的每个功能模块，判断该功能模块在实现其功能时是否有与用户进行交互的过程，如果没有则确定该功能模块存在风险；

如果所述 APK 中有至少一个功能模块被确认为存在风险，则确定该 APK 为恶意 APK。

2. 如权利要求 1 所述的方法，其中，该方法进一步包括：

对于拆分出的每个功能模块，进一步结合该功能模块实现其功能时的具体操作来确定是否存在风险。

3. 如权利要求 2 所述的方法，其中，所述对于拆分出的每个功能模块，进一步结合该功能模块实现其功能时的具体操作来确定是否存在风险包括：

分解出该功能模块实现其具体功能时所使用和调用的方法；

将分解出的各方法通过与预设的规则库进行匹配，判断各方法的权限和行为是否存在风险。

4. 如权利要求 1 所述的方法，其中，所述静态分析一个 APK，按照逻辑功能从所述 APK 中拆出至少一个功能模块包括：

通过反编译所述 APK，将其中的所有 Activity、Service、Broadcast Receiver 和 Content Provider 拆分出来，再根据调用逻辑关系划分成至少一个功能模块。

5. 一种检测恶意 APK 的装置，其中，该装置包括：

拆分单元，适于静态分析一个 APK，按照逻辑功能从所述 APK 中拆出至少一个功能模块；

判定单元，适于对于拆分出的每个功能模块，判断该功能模块在实现其功能时是否有与用户进行交互的过程，如果没有则确定该功能模块存在风险；如果所述 APK 中有至少一个功能模块被确认为存在风险，则确定该 APK 为恶意 APK。

6. 如权利要求 5 所述的装置，其中，

所述判定单元，适于对于拆分出的每个功能模块，进一步结合该功能模块实现其功能时的具体操作来确定是否存在风险。

7. 如权利要求 6 所述的装置，其中，

所述判定单元，适于分解出该功能模块实现其具体功能时所使用和调用的方法；将分解出的各方法通过与预设的规则库进行匹配，判断各方法的权限和行为是否存在风险。

8. 如权利要求 5 所述的装置，其中，

所述拆分单元，适于通过反编译所述 APK，将其中的所有 Activity、Service、Broadcast Receiver 和 Content Provider 拆分出来，再根据调用逻辑关系划分成至少一个功能模块。

9. 一种检测恶意 APK 的客户端，其中，该客户端包括如权利要求 5-8 中任一项所述的检测恶意 APK 的装置。

10. 一种检测恶意 APK 的服务器，其中，该服务器包括如权利要求 5-8 中任一项所述的检测恶意 APK 的装置。

一种检测恶意 APK 的方法、装置、客户端和服务器

技术领域

[0001] 本发明涉及网络安全技术领域，具体涉及一种检测恶意 APK 的方法、装置、客户端和服务器。

背景技术

[0002] 近年来，随着 Android 系统的发展和 Android 设备的流行，Android 已成为恶意软件开发者最关注的操作系统。恶意软件开发者通过修改流行应用程序的方式，将恶意软件代码嵌入到正常的应用程序中，然后将含有恶意代码的 APK 通过第三方软件商店或者论坛进行发布。目前，恶意软件主要利用操作系统存在的系统安全漏洞，通过私自注册付费服务，删除提示或确认短信的方式，窃取用户手机话费，窃取用户隐私数据，如短信、电子邮件、通讯录等。严重危害了用户的信息和财产安全。

[0003] 现有技术中，基于特征代码 (signature-based) 的检测方案是当前恶意 APK 检测的核心技术之一，该方案是指：从恶意软件中提取出若干段具有唯一性、固定性的字节码，如一段特殊代码或字符串，作为该恶意软件的特征，并依靠由上述方法构建的特征库来对 APK 进行检测，当 APK 中的代码与以构建的特征库相匹配，则判断该 APK 为恶意软件。

[0004] 基于特征代码 (signature-based) 的检测方案存在以下缺点：1、具有滞后性，无法识别未知的恶意软件；2、字节码特征容易通过加密和混淆的方式被改变，导致特征库需要频繁更新。

发明内容

[0005] 鉴于上述问题，提出了本发明以便提供一种克服上述问题或者至少部分地解决上述问题的一种检测恶意 APK 的方法、装置、客户端和服务器。

[0006] 依据本发明的一个方面，提供了一种检测恶意 APK 的方法，该方法包括：

[0007] 静态分析一个 APK，按照逻辑功能从所述 APK 中拆出至少一个功能模块；

[0008] 对于拆分出的每个功能模块，判断该功能模块在实现其功能时是否有与用户进行交互的过程，如果没有则确定该功能模块存在风险；

[0009] 如果所述 APK 中有至少一个功能模块被确认为存在风险，则确定该 APK 为恶意 APK。

[0010] 可选地，该方法进一步包括：

[0011] 对于拆分出的每个功能模块，进一步结合该功能模块实现其功能时的具体操作来确定是否存在风险。

[0012] 可选地，所述对于拆分出的每个功能模块，进一步结合该功能模块实现其功能时的具体操作来确定是否存在风险包括：

[0013] 分解出该功能模块实现其具体功能时所使用和调用的方法；

[0014] 将分解出的各方法通过与预设的规则库进行匹配，判断各方法的权限和行为是否存在风险。

[0015] 可选地，所述静态分析一个 APK，按照逻辑功能从所述 APK 中拆出至少一个功能模块包括：

[0016] 通过反编译所述 APK，将其中的所有 Activity、Service、Broadcast Receiver 和 Content Provider 拆分出来，再根据调用逻辑关系划分成至少一个功能模块。

[0017] 可选地，所述对于拆分出的每个功能模块，判断该功能模块在实现其功能时是否有与用户进行交互的过程包括：

[0018] 判断该功能模块的实现其功能时是否有与 Activity 组件交互的过程，如果有则有与用户交互的过程，如果没有则没有与用户交互的过程。

[0019] 可选地，所述对于拆分出的每个功能模块，判断该功能模块在实现其功能时是否有与用户进行交互的过程包括：

[0020] 判断该功能模块中是否包含与用户交互特征库中的特征匹配的内容，如果包含则确定该功能模块在实现其功能时有与用户进行交互的过程。

[0021] 可选地，该方法进一步包括：

[0022] 根据白签名库来辅助确定所述 APK 是否为恶意 APK。

[0023] 可选地，在客户端完成上述流程；

[0024] 或者由客户端将所述 APK 上传到服务器端，由服务器端完成上述流程。

[0025] 依据本发明的另一个方面，提供了一种检测恶意 APK 的装置，该装置包括：

[0026] 拆分单元，适于静态分析一个 APK，按照逻辑功能从所述 APK 中拆出至少一个功能模块；

[0027] 判定单元，适于对于拆分出的每个功能模块，判断该功能模块在实现其功能时是否有与用户进行交互的过程，如果没有则确定该功能模块存在风险；如果所述 APK 中有至少一个功能模块被确认为存在风险，则确定该 APK 为恶意 APK。

[0028] 可选地，所述判定单元，适于对于拆分出的每个功能模块，进一步结合该功能模块实现其功能时的具体操作来确定是否存在风险。

[0029] 可选地，所述判定单元，适于分解出该功能模块实现其具体功能时所使用和调用的方法；将分解出的各方法通过与预设的规则库进行匹配，判断各方法的权限和行为是否存在风险。

[0030] 可选地，所述拆分单元，适于通过反编译所述 APK，将其中的所有 Activity、Service、Broadcast Receiver 和 Content Provider 拆分出来，再根据调用逻辑关系划分成至少一个功能模块。

[0031] 可选地，所述判定单元，适于判断该功能模块的实现其功能时是否有与 Activity 组件交互的过程，如果有则有与用户交互的过程，如果没有则没有与用户交互的过程。

[0032] 可选地，所述判定单元，适于判断该功能模块中是否包含与用户交互特征库中的特征匹配的内容，如果包含则确定该功能模块在实现其功能时有与用户进行交互的过程。

[0033] 可选地，该装置进一步包括：

[0034] 辅助单元，适于根据白签名库来辅助确定所述 APK 是否为恶意 APK。

[0035] 依据本发明的又一个方面，提供了一种检测恶意 APK 的客户端，该客户端包括如上任一项所述的检测恶意 APK 的装置。

[0036] 依据本发明的再一个方面，提供了一种检测恶意 APK 的服务器，该服务器包括如

上任一项所述的检测恶意 APK 的装置。

[0037] 由上述可知，本发明提供的技术方案，基于恶意 APK 通常在用户不知情的情况下进行操作的规律，通过判断 APK 的各功能模块是否与用户进行交互，确定该 APK 的安全级别。一方面，本方案通过静态分析可以将 APK 中的所有功能模块都拆分出来，避免了因遗漏功能模块而导致的检测失误，提高了检测的准确性；另一方面，本方案提供了一种具有启发式的、直观的检测恶意 APK 的方法，可以实现对未知 APK 的检测。本发明提供的这种恶意 APK 检测方法可以防止恶意应用程序偷窥电子设备用户的隐私信息（包括联系人信息、通话记录、短信、彩信、各种账户及密码等）的行为，防止恶意应用程序拨打扣费电话、发送扣费短信、访问耗费网络流量的网站，防止恶意应用程序安装木马和病毒程序，防止恶意应用程序记录用户的 GPS 或网络定位，拦截恶意应用程序弹出骚扰广告信息等等，可以对于任何恶意应用程序对于服务的调用进行拦截，从而提高了系统的安全性，提高了用户的体验。

[0038] 上述说明仅是本发明技术方案的概述，为了能够更清楚了解本发明的技术手段，而可依照说明书的内容予以实施，并且为了让本发明的上述和其它目的、特征和优点能够更明显易懂，以下特举本发明的具体实施方式。

附图说明

[0039] 通过阅读下文优选实施方式的详细描述，各种其他的优点和益处对于本领域普通技术人员将变得清楚明了。附图仅用于示出优选实施方式的目的，而并不认为是对本发明的限制。而且在整个附图中，用相同的参考符号表示相同的部件。在附图中：

[0040] 图 1 示出了根据本发明一个实施例的一种检测恶意 APK 的方法的流程图；

[0041] 图 2 示出了根据本发明一个实施例的一种检测恶意 APK 的装置的示意图；

[0042] 图 3 示出了根据本发明另一个实施例的一种检测恶意 APK 的装置的示意图；

[0043] 图 4 示出了根据本发明一个实施例的一种检测恶意 APK 的客户端的示意图；

[0044] 图 5 示出了根据本发明一个实施例的一种检测恶意 APK 的服务器的示意图。

具体实施方式

[0045] 下面将参照附图更详细地描述本公开的示例性实施例。虽然附图中显示了本公开的示例性实施例，然而应当理解，可以以各种形式实现本公开而不应被这里阐述的实施例所限制。相反，提供这些实施例是为了能够更透彻地理解本公开，并且能够将本公开的范围完整的传达给本领域的技术人员。

[0046] 本发明的实施例可以应用于计算机系统 / 服务器，其可与众多其它通用或专用计算系统环境或配置一起操作。适于与计算机系统 / 服务器一起使用的众所周知的计算系统、环境和 / 或配置的例子包括但不限于：个人计算机系统、服务器计算机系统、瘦客户机、厚客户机、手持或膝上设备、基于微处理器的系统、机顶盒、可编程消费电子产品、网络个人电脑、小型计算机系统、大型计算机系统和包括上述任何系统的分布式云计算技术环境，等等。

[0047] 计算机系统 / 服务器可以在由计算机系统执行的计算机系统可执行指令（诸如程序模块）的一般语境下描述。通常，程序模块可以包括例程、程序、目标程序、组件、逻辑、数据结构等等，它们执行特定的任务或者实现特定的抽象数据类型。计算机系统 / 服务器可

以在分布式云计算环境中实施，分布式云计算环境中，任务是由通过通信网络链接的远程处理设备执行的。在分布式云计算环境中，程序模块可以位于包括存储设备的本地或远程计算系统存储介质上。

[0048] 为了更加清晰的阐述本发明实施例，首选介绍下安卓恶意软件检测的基本原理，Android 安装包（APK 文件）一般通过 Android 应用市场下载、安装到手机上，也可以通过 USB 数据线等数据线接口或无线数据传输的方式从 PC 安装。Android 上的病毒、木马和其他恶意软件想要进入用户的手机，也必须打包成 APK 的形式。基于这一点，杀毒引擎就可以把查杀的目标集中到对 APK 文件的扫描上，从而大大提高扫描的效率。Android 安装包（APK 文件）中的哪些信息可以作为扫描的重点，针对此问题本申请进行了分析，具体如下：

[0049] 1) 包名

[0050] Android 操作系统通过 APK 的包名 (package name) 对各个安装的 APK 进行管理。“包名”源自于 Java 的 package 的概念，按照 Java 的 package 的命名风格，例如某个 Android 安装包的包名是 com.qihoo360.mobilesafe。Android 系统要求每个应用都声明一个唯一的包名。Android 平台下的恶意软件也需要声明一个包名，因此，包名就可以作为识别恶意软件的一个重要特征。

[0051] 2) 数字签名

[0052] 出于安全性的目的，Android 系统要求每个 APK 都要包含数字签名 (digital signature)。Android 系统在安装 APK 文件的时候会检查 APK 内部各文件的数字签名是否与其预先设定的数字签名一致，如果不一致，或者没有数字签名，则认为文件已被篡改，拒绝该 APK 的安装和运行。Android 平台下的恶意软件也不例外，所以 APK 文件的数字签名也可以作为识别恶意软件的一个重要特征。

[0053] 3)AndroidManifest.xml 中列出的各模块的入口信息

[0054] AndroidManifest.xml 是每个 APK 文件所必需的全局描述文件，里面列出了 Android 安装包中应用的每个模块的入口信息。在 Android 系统中，只有在 AndroidManifest.xml 中列出了的模块，才能够被系统调用。Android 平台下的木马，往往会伪装成正常的应用或游戏来诱骗用户安装，其中有很多木马就是寄生在一个正常的应用或游戏中，用户运行它的时候，看上去是原来的软件或游戏，但寄生在其中的木马模块在合适的时机就被激活，从而感染用户的手机。而因为 Android 系统要求所有的模块都要在 AndroidManifest.xml 中列出，这就为寻找寄生的木马提高了重要线索。因此，AndroidManifest.xml 中列出的各模块的信息，也是识别恶意软件的重要特征。

[0055] 4)Dex 文件和 ELF 文件

[0056] Android 应用通常是用 Java 语言开发的，它用 Android 开发工具编译之后变成了二进制的字节码 (byte code)，这些字节码被打包成 classes.dex 文件，由 Android 平台的 Dalvik 虚拟机来解释执行。为了能够调用 Android 系统功能，Android 系统提供了一套运行环境 (Android Framework)，Android 应用调用系统各功能都是通过调用 Android Framework 的库来实现的。

[0057] 另一方面，Android 系统也支持应用程序通过 JNI 或者 native executable 直接运行。此时应用执行的是直接在 CPU 上运行的二进制机器码，不需要经过虚拟机解释，可以直接调用 Android 库如 libc、WebKit、SQLite、OpenGL/ES 等来调用系统各功能。如果 Android

应用要通过 JNI 或者 native executable 运行，就需要将要执行的代码编译成 ELF 文件格式。ELF 是 Executable and Linkable Format 的缩写，是 Android/Linux 操作系统中可执行程序、共享库的文件格式。

[0058] Android 上的恶意软件要想在 Android 系统中运行起来，也要遵循上述架构规范。因此，在识别恶意软件的过程中，可以分别从 Dex 文件（即字节码文件）和 ELF 文件提取相应的特征。

[0059] 此外，Android 安装包的版本号、Android 安装包目录下各文件的 MD5 值等信息，也可以作为识别恶意软件的重要特征。其中，上述的恶意软件包括病毒、木马和其他恶意软件。

[0060] Android 系统有四大组件：Activity、Service、Broadcast Receiver 和 Content Provider，这四大组件均可以被 ActivityManagerService 所管理。在应用程序被自启动时会通过 ActivityManagerService 执行。

[0061] Androidmanifest.xml 文件，是安装包中较为重要的全局配置文件，其负责向系统注册 Android 系统的四大组件，以及向系统申请权限等。在加壳安装包中，将其作为需要加入加壳安装包的重要内部文件进行考虑，以与原安装包完全一致的副本被包含到加壳安装包中。由于加壳安装包中的 Androidmanifest.xml 文件即为原安装包的同名文件，其包名相同，故加壳安装包在系统中安装运行宿主应用程序之后，以 Androidmanifest.xml 向系统注册各个组件和申请系统权限，以此便建立了各个组件的入口，使经反射调用的目标应用程序的各个组件均可以被 ActivityManagerService 调用，而不必为所述各个组件构造 ActivityThread 和提供相应的 LoadedApk 对象，省去运行上下文环境的程序实现环节。

[0062] 本发明的方法所应用的环境包括可与远程服务器或云端通信的移动终端，该移动终端可以安装有 Android 操作系统，该系统处于未经 ROOT 授权或者已经获取 ROOT 权限的状态。接下来具体介绍本发明提供的技术方案。

[0063] 图 1 示出了根据本发明一个实施例的一种检测恶意 APK 的方法的流程图。如图 1 所示，该方法包括：

[0064] 步骤 S110，静态分析一个 APK，按照逻辑功能从该 APK 中拆出至少一个功能模块。

[0065] 步骤 S120，对于拆分出的每个功能模块，判断该功能模块在实现其功能时是否有与用户进行交互的过程，如果没有则确定该功能模块存在风险。

[0066] 步骤 S130，如果 APK 中有至少一个功能模块被确认为存在风险，则确定该 APK 为恶意 APK。

[0067] 可见，图 1 所示的方法基于恶意 APK 通常在用户不知情的情况下进行操作的规律，通过判断 APK 的各功能模块是否与用户进行交互，确定该 APK 的安全级别。一方面，本方案通过静态分析可以将 APK 中的所有功能模块都拆分出来，避免了因遗漏功能模块而导致的检测失误，提高了检测的准确性；另一方面，本方案提供了一种具有启发式的、直观的检测恶意 APK 的方法，可以实现对未知 APK 的检测。

[0068] 在本发明的一个实施例中，图 1 所示的方法进一步包括：步骤 S140，对于拆分出的每个功能模块，进一步结合该功能模块实现其功能时的具体操作来确定是否存在风险。本实施例将判断功能模块是否与用户进行交互与判断功能模块的操作是否存在风险这两部分结合起来，进一步提高了检测恶意 APK 的方法的准确性。例如，一个功能模块中的所有代

码的执行与用户没有一次交互，而且该功能模块中还执行了以下的行为中的一种或多种：自动添加书签、强制联网、发送恶意扣费短信、诱导扣费操作、默认联网无提示、响安全软件使用、有 Push 广告行为、强制开机自启动、卸载不干净、无法正常卸载、恶意群发短信、是否收费、使用公用证书、有积分墙广告、安装恶意插件、私自发短信、连接恶意扣费网站、卸载时有恶意行为、木马软件、使用有风险权限、普通广告、容易引起死机、扣费提示不明显、盗取用户信息、默认开机自启动、修改快捷方式和主页、使用不符的风险权限，等等，则可以确定该功能模块存在风险，需要在用户界面进行提示风险。如果执行了屏蔽短信、监控接收短信、读联系人权限、分割短信、发短信权限、监控信号变化、发多条短信代码、静默安装 apk、读短信权限、监控网络变化、获取短信内容代码、对短信有操作、发短信代码、危险 &root& 高危特征、静默安装 apk1a-r、写入 APN 设置等行为，则认为是高危特征，需要进行拦截，或者提示手机杀毒引擎进行查杀等。若存在有上述病毒文件对应的进程，则将所述病毒文件对应的进程删除。

[0069] 在本发明的一个实施例中，由服务器侧将 APK 的安全级别设定黑、灰、白三种级别，分别代表不同危险程度，并设定对应的处理规则。例如，黑 APK 一般会命中高危特征，禁止安装，灰 APK 由用户自行选择，白 APK 则可径行安装。当然，可以进一步简化为灰、白两种，灰及被认为是有风险，或者简化为黑、白两种。本领域技术人员熟悉服务器的这种云端控制技术，将在后续进一步概要揭示。无论如何，本发明将从本机远程规则库接口中获得云端服务器有关这些应用的处理规则的反馈，利用反馈结果做出相应的后续处理。具体而言，当针对当前目标应用返回黑应用标识时，可以随即停止该目标应用的安装；当标识为白应用或灰应用时，则可放行安装。出于交互性的考虑，当完成远程判断后，本发明将向用户界面弹窗提醒用户有关判断结果，并显示相应的处理建议，

[0070] 在本发明的一个实施例中，可以设置关于行为特征的规则库，将上述行为的特征收录到该行为库中，以进行判断。则在本发明的在一个具体的实施例中，上述步骤 S140 具体包括以下操作：

[0071] 步骤 S141，分解出该功能模块实现其具体功能时所使用和调用的方法。

[0072] 步骤 S142，将分解出的各方法通过与预设的规则库进行匹配，判断各方法的权限和行为是否存在风险。

[0073] 本步骤中，预设的规则库中保存了一些被认定为是风险行为的特征，如发短信、下载、安装包等等。因此与该预设的规则库相匹配的方法，其行为往往涉及到用户的隐私权限，存在风险，包括：发短信、安装软件包、下载、回传用户短信等方法。

[0074] 例如，对于一个 APK 进行检测，静态分析该 APK，按照逻辑功能从该 APK 中拆出多个功能模块，其中，一个功能模块所实现的功能为：发短信，该功能模块在实现发短信功能时没有与用户进行交互，并且，该功能模块所使用和调用的发短信的方法与预设的规则库相匹配，确定该功能模块存在风险，进而确定该 APK 为恶意 APK。

[0075] 在本发明的一个实施例中，图 1 所示方法的步骤 S110 静态分析一个 APK，按照逻辑功能从该 APK 中拆出至少一个功能模块包括：通过反编译所述 APK，将其中的所有 Activity、Service、Broadcast Receiver 和 Content Provider 等 Android 组件拆分出来，再根据调用逻辑关系划分成至少一个功能模块。

[0076] 例如，app 层的应用程序通过 NotificationManager.notify 方法向 framework 层

的 NotificationManagerService 发送状态栏通知；然后，在 NotificationManagerService 中，将应用程序发送的状态栏通知条目保存在列表中；最后，NotificationManagerService 通过 StatusBar.addNotification() 方法向系统状态栏所在的 SystemUi（系统用户界面）进程发送显示状态栏通知，这样应用程序发送的状态栏通知就会在系统状态栏中显示出来了。这是一个实现发送通知功能的模块。

[0077] Android 中的 Activity 组件表现为与用户交互的可视化界面，与 Activity 组件交互的过程即为与用户进行交互的过程。在本发明的一个实施例中，图 1 所示方法的步骤 S120，对于从被检测的 APK 中拆分出的每个功能模块，判断该功能模块在实现其功能时是否有与用户进行交互的过程包括以下方案：

[0078] 方案一，判断从被检测的 APK 中拆分出的功能模块实现其功能时是否有与 Activity 组件交互的过程，如果有则有与用户交互的过程，如果没有则没有与用户交互的过程。

[0079] 例如，对于一个 APK 进行检测，静态分析该 APK，按照逻辑功能从该 APK 中拆出多个功能模块，其中，一个功能模块在实现其功能时操作了一个 Activity 组件，或者启动了一个新的 Activity 组件，给用户弹出了一个窗口，供用户进行选择或确认等操作，即有与用户交互的过程，则该功能模块无风险。

[0080] 方案二，判断该功能模块中是否包含与用户交互特征库中的特征匹配的内容，如果包含则确定该功能模块在实现其功能时有与用户进行交互的过程。

[0081] 本方案中，用户交互特征库中的特征包括 Android 中需要与界面交互的多个类和方法。

[0082] 在本发明的一个实施例中，为了降低上述方案检测失误的几率，提供了进一步的辅助方案。因此，图 1 所示的方法进一步包括：步骤 S150，根据白签名库来辅助确定 APK 是否为恶意 APK。其中，白签名库中记录了已确认正常的 APK 的签名。例如，在对一个 APK 进行检测之前，用户已对该 APK 进行授权，使得该 APK 的各功能模块在执行操作时无需与用户进行交互。在此情境下，步骤 S110 至步骤 S130 的检测将会导致误判；此时，白签名库中记录了该 APK 的签名，确定该 APK 已经被确认非恶意 APK，检测结果被修正，降低了误判几率。

[0083] 在本发明的一些实施例中，可以在在客户端完成上述检测流程。或者，在本发明的另一些实施例中，可以在由终端设备侧的客户端将待检测 APK 上传到服务器端，由服务器端完成上述流程。

[0084] 终端设备侧将 APK 对应的应用程序列表发送至云端服务器，并接收由云端服务器判断应用程序列表中的应用程序是否为白或是否为黑的结果。实际应用中，终端设备中可能安装上百个应用程序，但是由于终端设备本地容量有限，一般只能识别出 20 个左右的应用程序，对于剩余的将近 80 款软件应用程序未能识别，此时的配置可以是在本地查找完后，全部上传云端服务器再次复查，或者直接将应用程序列表上传云端服务器识别。

[0085] 终端设备提取出应用程序的特征信息后，在终端设备的用户界面显示提示信息，提示用户选择在终端设备本地识别还是上传到云端服务器，由云端服务器识别。通常，如果用户终端设备有包月的上网流量，则可以选择上传云端服务器，由云端服务器识别，以提高白名单识别的准确率；如果上网流量用完，又不想额外耗费更多流量，则可以选择仅在终端设备本地识别，或者优先在终端设备本地识别，如果终端设备本地识别的结果不全，还可以

将剩余未识别的特征信息上传云端服务器，由云端服务器识别。

[0086] 如果该应用程序在该预设的本地应用程序白名单中，则说明该应用程序的名称是属于受信任的应用程序的名称。如果该应用程序不在该预设的应用程序白名单中，则说明该应用程序不受信任，可以进一步由客户端或云端服务器根据安全识别库以及本发明提供的上述检测方法进行识别。

[0087] 云端服务器存储的特征信息包括以下中的一种或几种组合：Android 安装包的包名，版本号，开发者签名，Android 组件 receiver 的特征，Android 组件 service 的特征，Android 组件 activity 的特征，可执行文件中的指令或字符串，Android 安装包目录下各文件的 MD5 值；其中，所述可执行文件包括 Dex 文件，和 / 或，ELF 文件；所述 Dex 文件包括 classes.dex 文件，扩展名为 .jar 的文件，以及，Dex 格式的文件。

[0088] 图 2 示出了根据本发明一个实施例的一种检测恶意 APK 的装置的示意图。如图 2 所示，该检测恶意 APK 的装置 200 包括：

[0089] 拆分单元 210，适于静态分析一个 APK，按照逻辑功能从该 APK 中拆出至少一个功能模块；

[0090] 判定单元 220，适于对于拆分出的每个功能模块，判断该功能模块在实现其功能时是否有与用户进行交互的过程，如果没有则确定该功能模块存在风险；如果 APK 中有至少一个功能模块被确认为存在风险，则确定该 APK 为恶意 APK。

[0091] 可见，图 2 所示的装置基于恶意 APK 通常在用户不知情的情况下进行操作的规律，通过各单元的相互配合，判断 APK 的各功能模块是否与用户进行交互，确定该 APK 的安全级别。一方面，本方案通过静态分析可以将 APK 中的所有功能模块都拆分出来，避免了因遗漏功能模块而导致的检测失误，提高了检测的准确性；另一方面，本方案提供了一种具有启发式的、直观的检测恶意 APK 的方法，可以实现对未知 APK 的检测。

[0092] 在本发明的一个实施例中，图 2 所示装置的判定单元 220，适于对于拆分出的每个功能模块，进一步结合该功能模块实现其功能时的具体操作来确定是否存在风险。本实施例将判断功能模块是否与用户进行交互与判断功能模块的操作是否存在风险这两部分结合起来，进一步提高了检测恶意 APK 的方法的准确性。在一个具体的实施例中，判定单元 220，适于分解出该功能模块实现其具体功能时所使用和调用的方法；将分解出的各方法通过与预设的规则库进行匹配，判断各方法的权限和行为是否存在风险。其中，与预设的规则库相匹配的方法，其行为往往涉及到用户的隐私权限，存在风险，包括：发短信、安装软件包、下载、回传用户短信等方法。

[0093] 例如，图 2 所示的装置对于一个 APK 进行检测，拆分单元 210 静态分析该 APK，按照逻辑功能从该 APK 中拆出多个功能模块，判定单元 220 判断出其中一个功能模块块在实现发短信功能时没有与用户进行交互；并且，判定单元 220 判断出该功能模块所使用和调用的发短信的方法与预设的规则库相匹配，确定该功能模块存在风险，进而确定该 APK 为恶意 APK。

[0094] 在本发明的一个实施例中，图 2 所示装置的拆分单元 210，适于通过反编译该 APK，将其中的所有 Activity、Service、Broadcast Receiver 和 Content Provider 拆分出来，再根据调用逻辑关系划分成至少一个功能模块。

[0095] Android 中的 Activity 组件表现为与用户交互的可视化界面，与 Activity 组件交

互的过程即为与用户进行交互的过程。在本发明的一个实施例中，判定单元 220 判断该功能模块在实现其功能时是否有与用户进行交互的过程包括以下方案：

[0096] 方案一，判定单元 220，适于判断该功能模块的实现其功能时是否有与 Activity 组件交互的过程，如果有则有与用户交互的过程，如果没有则没有与用户交互的过程。

[0097] 例如，图 2 所示的装置对于一个 APK 进行检测，拆分单元 210 静态分析该 APK，按照逻辑功能从该 APK 中拆出多个功能模块，其中，判定单元 220 判断出一个功能模块在实现其功能时操作了一个 Activity 组件，或者启动了一个新的 Activity 组件，给用户弹出了一个窗口，供用户进行选择或确认等操作，即有与用户交互的过程，则确定该功能模块无风险。

[0098] 方案二，判定单元 220，适于判断该功能模块中是否包含与用户交互特征库中的特征匹配的内容，如果包含则确定该功能模块在实现其功能时有与用户进行交互的过程。本方案中，用户交互特征库中的特征包括 Android 中需要与界面交互的多个类和方法。

[0099] 图 3 示出了根据本发明另一个实施例的一种检测恶意 APK 的装置的示意图。本实施例为了降低上述方案检测失误的几率，增添了进一步的辅助方案。如图 3 所示，该检测恶意 APK 的装置 300 包括：拆分单元 310、判定单元 320 和辅助单元 330。

[0100] 其中，拆分单元 310、判定单元 320 分别与图 2 所示装置的拆分单元 210、判定单元 220 对应相同，在此不再赘述。

[0101] 辅助单元 330，适于根据白签名库来辅助确定所述 APK 是否为恶意 APK。

[0102] 其中，白签名库中记录了已确认正常的 APK 的签名。例如，在对一个 APK 进行检测之前，用户已对该 APK 进行授权，使得该 APK 的各功能模块在执行操作时无需与用户进行交互。在此情境下，拆分单元 310 和判定单元 320 所执行的检测过程将会导致误判；此时，白签名库中记录了该 APK 的签名，辅助单元 330 根据白签名库确定该 APK 已经被确认非恶意 APK，检测结果被修正，降低了检测失误的几率。

[0103] 图 4 示出了根据本发明一个实施例的一种检测恶意 APK 的客户端的示意图。如图 4 所示，该检测恶意 APK 的客户端 400 包括如上文中任一实施例所述的检测恶意 APK 的装置 410。

[0104] 图 5 示出了根据本发明一个实施例的检测恶意 APK 的服务器的示意图。如图 5 所示，该检测恶意 APK 的服务器 500 包括如上文中任一实施例所述的检测恶意 APK 的装置 510。

[0105] 综上所述，本发明提供的技术方案基于恶意 APK 通常在用户不知情的情况下进行操作的规律，通过判断 APK 的各功能模块是否与用户进行交互和判断各功能模块的操作是否存在风险两方面，确定该 APK 的安全级别，能够产生以下有益效果：1、本方案提供了一种具有启发式的、直观的检测恶意 APK 的方法，可以实现对未知 APK 的检测。2、本方案通过静态分析可以将 APK 中的所有功能模块都拆分出来，避免了因遗漏功能模块而导致的检测失误，提高了检测的准确性。3、交互过程的判断和可疑行为的判断相结合，提高了检测的准确性。4、根据白签名库来辅助检测，降低了检测失误的几率。5、在客户端和服务器均可实现，使检测恶意 APK 的方案更具灵活性。

[0106] 进一步地，本发明的上述实施例中的方法可以结合以下方案：

[0107] 可以通过多个杀毒引擎进行病毒检测，驱动多个病毒引擎联合进行病毒检测；，多个病毒引擎包括：AVE 引擎、AVM 引擎、云查杀引擎、机器学习引擎或者脚本病毒检测引擎

等等。

[0108] 病毒引擎可以设置于：服务端和客户端，例如，云查杀引擎等等包括服务器端查杀工具和客户端查杀工具。服务端通过将病毒样本序列与病毒库文件中的记录匹配进行病毒检测，将病毒检测结果下发到客户端，并提供修复方案，修复方案包括：文件类型、与文件类型对应的查杀方法等等，客户端可以根据修复方案进行查杀病毒。客户端可以安装在手机、PC、PAD 等上，通过手机端等上的客户端查杀引擎，或者是手机端等上的应用分发平台等工具，为用户提供可靠的移动互联网安全服务。

[0109] 本发明实施例也可以结合无线终端安全防护产品上的手机杀毒产品，当用户点击快速扫描，则可以执行安全扫描，另外，也可以联合主动防御，沙箱等功能，运行监测未知的应用程序的特征和应用权限等。

[0110] 需要说明的是：

[0111] 在此提供的算法和显示不与任何特定计算机、虚拟装置或者其它设备固有相关。各种通用装置也可以与基于在此的示教一起使用。根据上面的描述，构造这类装置所要求的结构是显而易见的。此外，本发明也不针对任何特定编程语言。应当明白，可以利用各种编程语言实现在此描述的本发明的内容，并且上面对特定语言所做的描述是为了披露本发明的最佳实施方式。

[0112] 在此处所提供的说明书中，说明了大量具体细节。然而，能够理解，本发明的实施例可以在没有这些具体细节的情况下实践。在一些实例中，并未详细示出公知的方法、结构和技术，以便不模糊对本说明书的理解。

[0113] 类似地，应当理解，为了精简本公开并帮助理解各个发明方面中的一个或多个，在上面对本发明的示例性实施例的描述中，本发明的各个特征有时被一起分组到单个实施例、图、或者对其的描述中。然而，并不应将该公开的方法解释成反映如下意图：即所要求保护的本发明要求比在每个权利要求中所明确记载的特征更多的特征。更确切地说，如下面的权利要求书所反映的那样，发明方面在于少于前面公开的单个实施例的所有特征。因此，遵循具体实施方式的权利要求书由此明确地并入该具体实施方式，其中每个权利要求本身都作为本发明的单独实施例。

[0114] 本领域那些技术人员可以理解，可以对实施例中的设备中的模块进行自适应性地改变并且把它们设置在与该实施例不同的一个或多个设备中。可以把实施例中的模块或单元或组件组合成一个模块或单元或组件，以及此外可以把它们分成多个子模块或子单元或子组件。除了这样的特征和／或过程或者单元中的至少一些是相互排斥之外，可以采用任何组合对本说明书（包括伴随的权利要求、摘要和附图）中公开的所有特征以及如此公开的任何方法或者设备的所有过程或单元进行组合。除非另外明确陈述，本说明书（包括伴随的权利要求、摘要和附图）中公开的每个特征可以由提供相同、等同或相似目的的替代特征来代替。

[0115] 此外，本领域的技术人员能够理解，尽管在此所述的一些实施例包括其它实施例中所包括的某些特征而不是其它特征，但是不同实施例的特征的组合意味着处于本发明的范围之内并且形成不同的实施例。例如，在下面的权利要求书中，所要求保护的实施例的任意之一都可以以任意的组合方式来使用。

[0116] 本发明的各个部件实施例可以以硬件实现，或者以在一个或者多个处理器上运行

的软件模块实现，或者以它们的组合实现。本领域的技术人员应当理解，可以在实践中使用微处理器或者数字信号处理器（DSP）来实现根据本发明实施例的一种检测恶意 APK 的装置、客户端和服务器中的一些或者全部部件的一些或者全部功能。本发明还可以实现为用于执行这里所描述的方法的一部分或者全部的设备或者装置程序（例如，计算机程序和计算机程序产品）。这样的实现本发明的程序可以存储在计算机可读介质上，或者可以具有一个或者多个信号的形式。这样的信号可以从因特网网站上下载得到，或者在载体信号上提供，或者以任何其他形式提供。

[0117] 应该注意的是上述实施例对本发明进行说明而不是对本发明进行限制，并且本领域技术人员在不脱离所附权利要求的范围的情况下可设计出替换实施例。在权利要求中，不应将位于括号之间的任何参考符号构造成对权利要求的限制。单词“包含”不排除存在未列在权利要求中的元件或步骤。位于元件之前的单词“一”或“一个”不排除存在多个这样的元件。本发明可以借助于包括有若干不同元件的硬件以及借助于适当编程的计算机来实现。在列举了若干装置的单元权利要求中，这些装置中的若干个可以是通过同一个硬件项来具体体现。单词第一、第二、以及第三等的使用不表示任何顺序。可将这些单词解释为名称。

[0118] 本发明公开了 A1、一种检测恶意 APK 的方法，其中，该方法包括：

[0119] 静态分析一个 APK，按照逻辑功能从所述 APK 中拆出至少一个功能模块；

[0120] 对于拆分出的每个功能模块，判断该功能模块在实现其功能时是否有与用户进行交互的过程，如果没有则确定该功能模块存在风险；

[0121] 如果所述 APK 中有至少一个功能模块被确认为存在风险，则确定该 APK 为恶意 APK。

[0122] A2、如 A1 所述的方法，其中，该方法进一步包括：

[0123] 对于拆分出的每个功能模块，进一步结合该功能模块实现其功能时的具体操作来确定是否存在风险。

[0124] A3、如 A2 所述的方法，其中，所述对于拆分出的每个功能模块，进一步结合该功能模块实现其功能时的具体操作来确定是否存在风险包括：

[0125] 分解出该功能模块实现其具体功能时所使用和调用的方法；

[0126] 将分解出的各方法通过与预设的规则库进行匹配，判断各方法的权限和行为是否存在风险。

[0127] A4、如 A1 所述的方法，其中，所述静态分析一个 APK，按照逻辑功能从所述 APK 中拆出至少一个功能模块包括：

[0128] 通过反编译所述 APK，将其中的所有 Activity、Service、Broadcast Receiver 和 Content Provider 拆分出来，再根据调用逻辑关系划分成至少一个功能模块。

[0129] A5、如 A1 所述的方法，其中，所述对于拆分出的每个功能模块，判断该功能模块在实现其功能时是否有与用户进行交互的过程包括：

[0130] 判断该功能模块的实现其功能时是否有与 Activity 组件交互的过程，如果有则有与用户交互的过程，如果没有则没有与用户交互的过程。

[0131] A6、如 A1 所述的方法，其中，所述对于拆分出的每个功能模块，判断该功能模块在实现其功能时是否有与用户进行交互的过程包括：

[0132] 判断该功能模块中是否包含与用户交互特征库中的特征匹配的内容，如果包含则确定该功能模块在实现其功能时有与用户进行交互的过程。

[0133] A7、如 A1 所述的方法，其中，该方法进一步包括：

[0134] 根据白签名库来辅助确定所述 APK 是否为恶意 APK。

[0135] A8、如 A1-A7 中任一项所述的方法，其中，

[0136] 在客户端完成上述流程；

[0137] 或者由客户端将所述 APK 上传到服务器端，由服务器端完成上述流程。

[0138] 本发明还公开了 B9、一种检测恶意 APK 的装置，其中，该装置包括：

[0139] 拆分单元，适于静态分析一个 APK，按照逻辑功能从所述 APK 中拆出至少一个功能模块；

[0140] 判定单元，适于对于拆分出的每个功能模块，判断该功能模块在实现其功能时是否有与用户进行交互的过程，如果没有则确定该功能模块存在风险；如果所述 APK 中有至少一个功能模块被确认为存在风险，则确定该 APK 为恶意 APK。

[0141] B10、如 B9 所述的装置，其中，

[0142] 所述判定单元，适于对于拆分出的每个功能模块，进一步结合该功能模块实现其功能时的具体操作来确定是否存在风险。

[0143] B11、如 B0 所述的装置，其中，

[0144] 所述判定单元，适于分解出该功能模块实现其具体功能时所使用和调用的方法；将分解出的各方法通过与预设的规则库进行匹配，判断各方法的权限和行为是否存在风险。

[0145] B12、如 B9 所述的装置，其中，

[0146] 所述拆分单元，适于通过反编译所述 APK，将其中的所有 Activity、Service、Broadcast Receiver 和 Content Provider 拆分出来，再根据调用逻辑关系划分成至少一个功能模块。

[0147] B13、如 B9 所述的装置，其中，

[0148] 所述判定单元，适于判断该功能模块的实现其功能时是否有与 Activity 组件交互的过程，如果有则有与用户交互的过程，如果没有则没有与用户交互的过程。

[0149] B14、如 B9 所述的装置，其中，

[0150] 所述判定单元，适于判断该功能模块中是否包含与用户交互特征库中的特征匹配的内容，如果包含则确定该功能模块在实现其功能时有与用户进行交互的过程。

[0151] B15、如 B9 所述的装置，其中，该装置进一步包括：

[0152] 辅助单元，适于根据白签名库来辅助确定所述 APK 是否为恶意 APK。

[0153] 本发明还公开了 C16、一种检测恶意 APK 的客户端，其中，该客户端包括如 B9-B15 中任一项所述的检测恶意 APK 的装置。

[0154] 本发明还公开了 D17、一种检测恶意 APK 的服务器，其中，该服务器包括如 B9-B15 中任一项所述的检测恶意 APK 的装置。

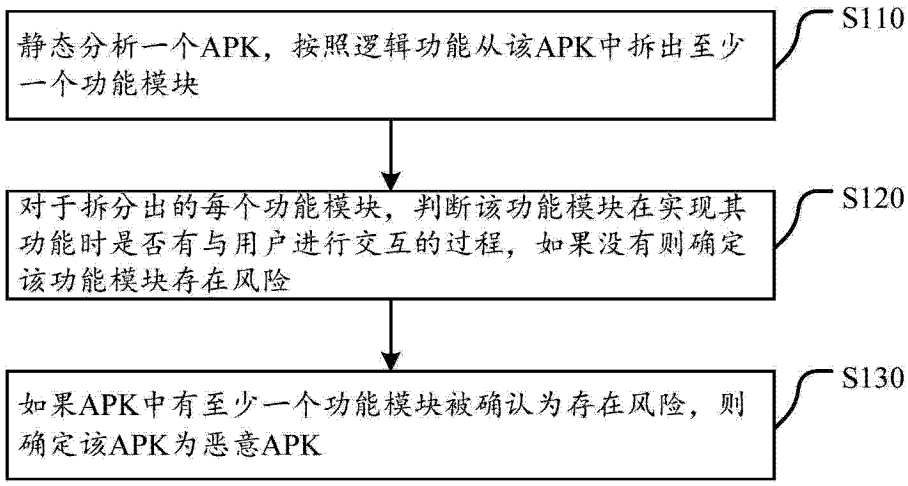

静态分析一个APK，按照逻辑功能从该APK中拆出至少一个功能模块
S110
对于拆分出的每个功能模块，判断该功能模块在实现其功能时是否有与用户进行交互的过程，如果没有则确定该功能模块存在风险
S120
如果APK中有至少一个功能模块被确认为存在风险，则确定该APK为恶意APK
S130

图 1

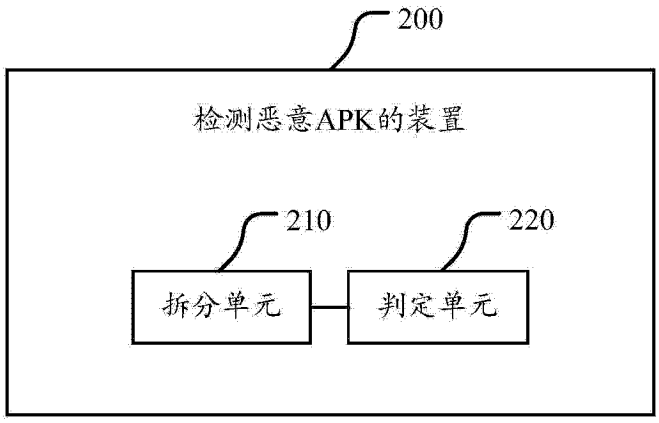

200
检测恶意APK的装置
210
拆分单元
220
判定单元

图 2

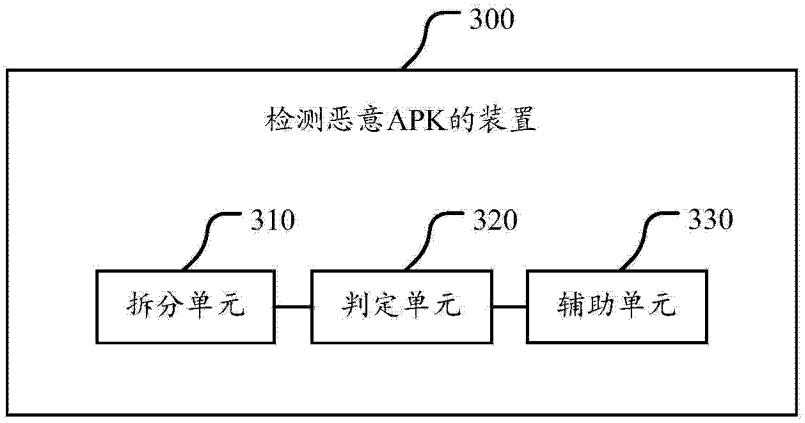

300
检测恶意APK的装置
310
拆分单元
320
判定单元
330
辅助单元

图 3

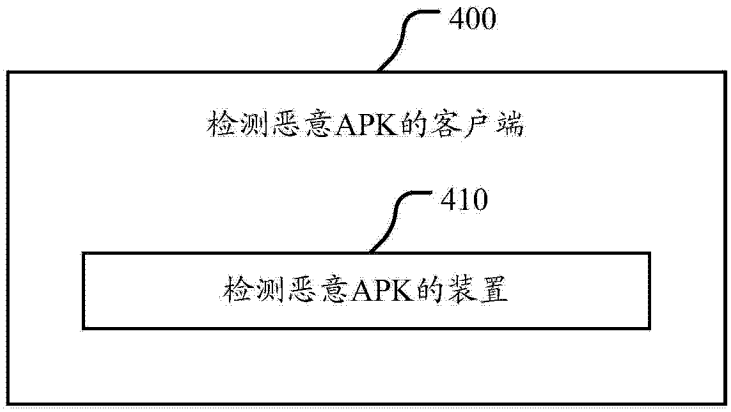

400
检测恶意APK的客户端
410
检测恶意APK的装置

图 4

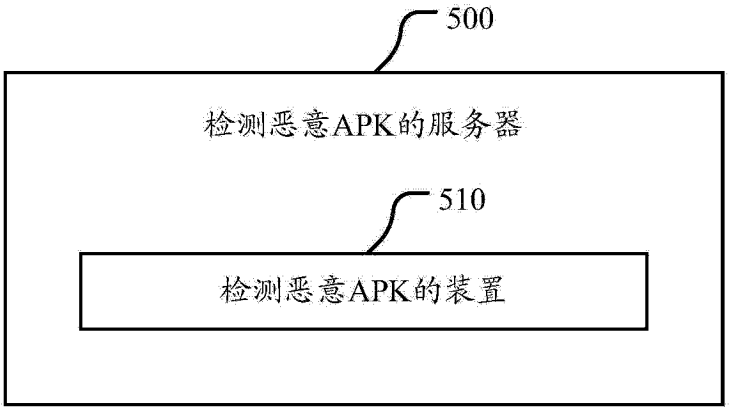

500
检测恶意APK的服务器
510
检测恶意APK的装置

图 5