目录

一、谈谈你对安卓签名的理解。

Android 的签名工作机制：

1) 每个应用都必须签名

2) 应用可以被不同的签名文件签名（如果有源代码或者反编译后重新编译）

3) 同一个应用如果签名不同则不能覆盖安装

二、请解释安卓为啥要加签名机制?

https://blog.csdn.net/fyh2003/article/details/6911967

为什么要签名

1) **发送者的身份认证：**由于开发商可能通过使用相同的 Package Name 来混淆替换已经安装的程序，以此保证签名不同的包不被替换

2) **保证信息传输的完整性：**签名对于包中的每个文件进行处理，以此确保包中内容不被替换

3) **防止交易中的抵赖发生：**Market（应用市场）对软件的要求

给 apk 签名可以带来以下好处：

1) **应用程序升级：**能无缝升级到新的版本，必须要同一个证书进行

签名并且包名称要相同。（如果证书不同，可能会被系统认为是不同的应用）

2) **应用程序模块化：**Android 系统可以允许同一个证书签名的多个应用程序在一个进程里运行（系统实际把他们作为一个单个的应用程序），此时就可以把我们的应用程序以模块的方式进行部署，而用户可以独立的升级其中的一个模块

3) **代码或者数据共享：**Android 提供了基于签名的权限机制，那么一个应用程序就可以为另一个以相同证书签名的应用程序公开自己的功能。

签名的说明

1) **所有的应用程序都必须有数字证书：**Android 系统不会安装一个没有数字证书的应用程序

2) **Android 程序包使用的数字证书可以是自签名的：**不需要一个权威的数字证书机构签名认证

3) **使用一个合适的私钥生成的数字证书来给程序签名：**如果要正式发布一个 Android 应用，必须使用一个合适的私钥生成的数字证书来给程序签名，而不能使用 adt 插件或者 ant 工具生成的调试证书来发布

4) **数字证书都是有有效期的：**Android 只是在应用程序安装的时候才会检查证书的有效期。如果程序已经安装在系统中，即使证书过期也不会影响程序的正常功能

三、视频加密传输

https://blog.csdn.net/qq_24636637/article/details/50524243

1. DES 加密。用 java 中提供的加密包。

2. 将视频文件的数据流前 100 个字节中的每个字节与其下标进行异或运算。解密时只需将加密过的文件再进行一次异或运算即可。

四、App 是如何沙箱化，为什么要这么做？

https://blog.csdn.net/ljheee/article/details/53191397

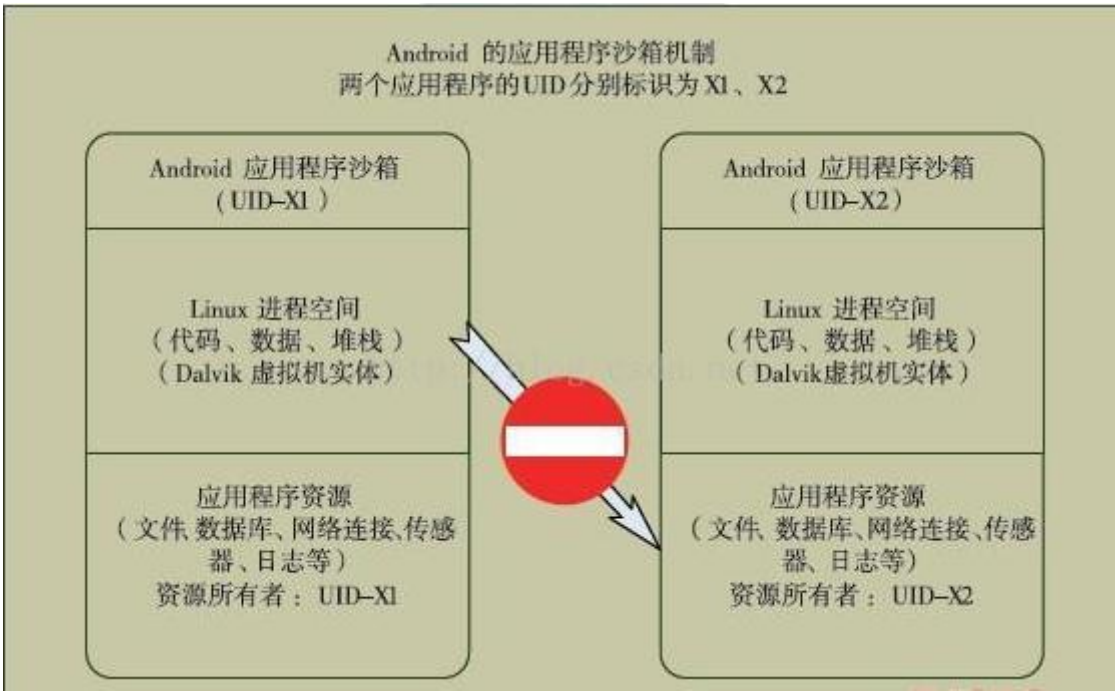
在 Android 系统中，应用（通常）都在一个独立的沙箱中运行，即每一个 Android 应用程序都在它自己的进程中运行，都拥有一个独立的 Dalvik 虚拟机实例。Dalvik 经过优化，允许在有限的内存中同时高效地运行多个虚拟机的实例，并且每一个 Dalvik 应用作为一个独立的 Linux 进程执行。Android 这种基于 Linux 的进程“沙箱”机制，是整个安全设计的基础之一。

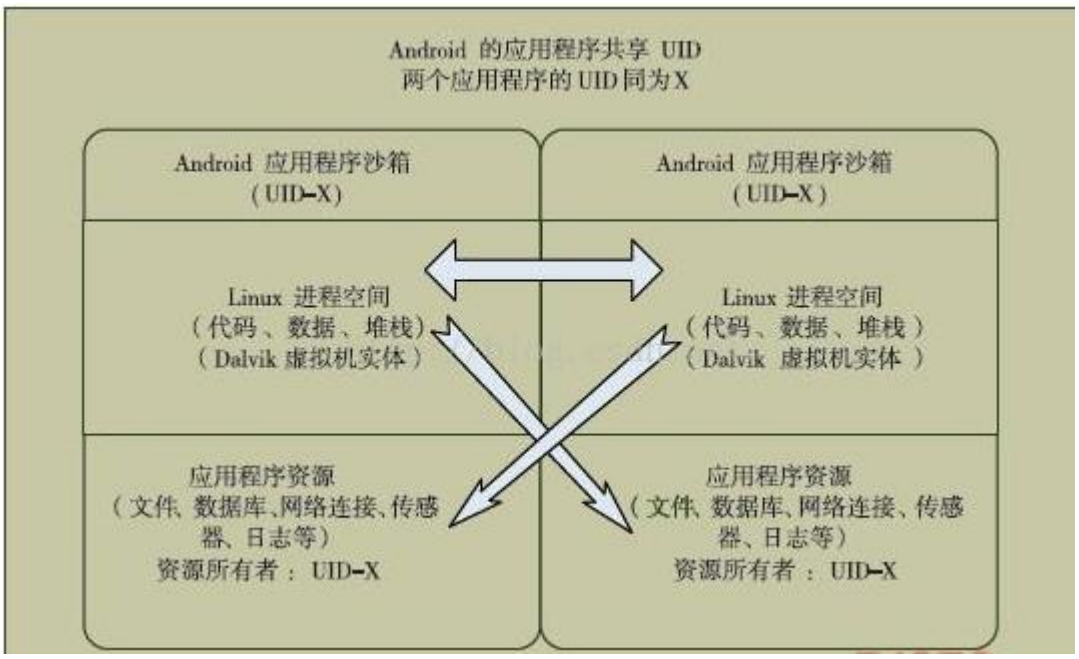
Android 扩展了 Linux 内核安全模型的用户与权限机制，将多用户操作系统的用户隔离机制巧妙地移植为应用程序隔离。将 UID（一个用户标识）不同的应用程序自然形成资源隔离，如此便形成了一个操作系统级别的应用程序“沙箱”。

Android 应用程序的“沙箱”机制：

1) 互相不具备信任关系的应用程序相互隔离，独自运行，访问是禁止的：

2) 通过共享 UID（SharedUserID）,应用程序在同一个进程运行，共享数据





Android 的权限分离的基础是建立在 Linux 已有的 uid 、gid 、 gids 基础上的 。

1) UID： Android 在 安装一个应用程序，就会为 它 分配一个 uid 。（其中普通 A ndroid 应用程序的 uid 是从 10000 开始分配， 10000 以下是系统进程的 uid ）
2) GID：对 于普通应用程序来说， gid 等于 uid 。由于每个应用程序的 uid 和 gid 都不相同， 因此不管是 native 层还是 java 层都能够达到保护私有数据的作用 。
3) GIDS： gids 是由框架在 Application 安装过程中生成，与 Application 申请的具体权限相关。

Act ivityManagerService 中的 startProcessLocked 。在通过

zygote 来启动一个 process 时，直接将 uid 传给 给了 gid 。再通过 zygote 来 fork 出新的进程（ zygote.java 中的 forkAndSpecialize ），最终在 native 层（ dalvik_system_zygote.c ）中的 forkAndSpecializeCommon 中通过 linux 系统调用来进行 gid 和 uid 和 gids 的设置。

五、权限管理系统（底层的权限是如何进行 grant（发放） 的）？

https://blog.csdn.net/wxlinwzl/article/details/38395589

1. **permission 的初始化：**是指 permission 的向系统申请，系统进行检测并授权，并建立相应的数据结构。
2. **权限检查：**Android framework 中提供了一些接口用来对外来的访问（包括自己）进行权限检查 。 这些接口 主要通过 ContextWrapper 提供， 具体实现在 ContextImpl 中 。 ContextImpl.java 中提供的 API ， 其实都是由 ActivityManagerService 中的如下几个接口进行的封装。

其中有一个 checkPermission() // 主要用于一般的 permission 检查

checkPermission 的实现分析

1) 如果传入的 permission 名称为 null ， 那么返回 PackageManager.PERMISSION_DENIED 。
2) 判断调用者 uid 是否符合要求 。
3) 如果通过 2 的检查后， 再 调用 PackageManagerService.checkUidPermission ， 判断 这个 uid 是否拥有相应的权限

3. **权限校验之后，应给分发了**