Atititit 安卓android app apk反编译

Android APK中的Java代码可以被反编译到什么程度主要看APK的加密程度。

第一种情况：无混淆无加密无加壳。  
直接利用Dex2jar和JD-GUI可把源码从APK里抠出来，代码逻辑清晰，基本上做到可复用，只是资源文件的引用需要计算一下。

第二种情况：混淆。  
通常是利用Proguard做的防护。因为是对jar做的不可逆混淆（除非有mapping），因此不能还原成原来的代码。但是代码结构，代码逻辑一致，只要花长时间对代码进行梳理一样可找准核心代码，解密方法跟第一种一致。

第三种情况：加密。  
这里以DexGuard为例。对于这种代码加密的方法，在程序运行中必定会进行解密，只要抽出它解密的逻辑便可。PS：我自己做过DexGuard的解密，如果用Dex2jar反编译看的话逻辑是不对的，一定要从Smali代码看。后来发现网上已经有人做了。  
解密的脚本：[A look inside Dexguard](https://link.zhihu.com/?target=https://www.pnfsoftware.com/blog/2013/04/02/a-look-inside-dexguard/" \t "_blank)

第四种情况：加壳。  
这种情况跟第三种类似。无论你怎么加壳，运行的时候必定是Dalvik可识别的Odex代码，建议直接在内存里dump出来。这里推荐Xpose的ZjDroid。  
脱壳教程：[ZjDroid工具介绍及脱壳详细示例](https://link.zhihu.com/?target=http://www.cnblogs.com/goodhacker/p/3961045.html" \t "_blank)

通常都是这几种情况的混合，至于native层的只能说半桶水不敢献丑。

我开发过有别于以上的基于Java class文件和dex文件的混淆。对于Java class文件，我们可采用打乱package结构和类里方法合并的方式进行混淆，但这种混淆方式会破坏原代码里的字段和方法的Signatrue，同时也会造成一些很小的效率和内存占用影响，因此尚未商用。有兴趣的人可去我的github里了解或改进相关代码：[VinceXie/Java-ApkObfuscate · GitHub](https://link.zhihu.com/?target=https://github.com/VinceXie/Java-ApkObfuscate" \t "_blank)

参考资料

手把手教你搞懂Android反编译.html