#Android反编译#零基础脱掉360加固包的“外衣”

原创

superyu1992 于 2020-04-29 17:41:20 发布 5638 收藏 18
分类专栏： Android笔记 思路整理 文章标签： 反编译 逆向 Android

本文链接：https://blog.csdn.net/superyu1992/article/details/105841465

我们在开发App的过程中为了保护自己的劳动成果不被剽窃，也为了保证接口不被暴露，通常在上架之前做一些混淆或者加固的处理，市面上也有不少加固工具，更有甚者在某些平台上线App必须使用该平台的加固工具进行加固后才可上线！这些加固真的靠谱吗？这两天我就对通过某60加固的App下手了，结果作为逆向零基础的小白，用了大约两天的时间，就成功脱壳了。。。下面就来分享一下我这几天的学习与破解过程。（下面的分享都是基于Mac系统开发的，Windows会略有不同）

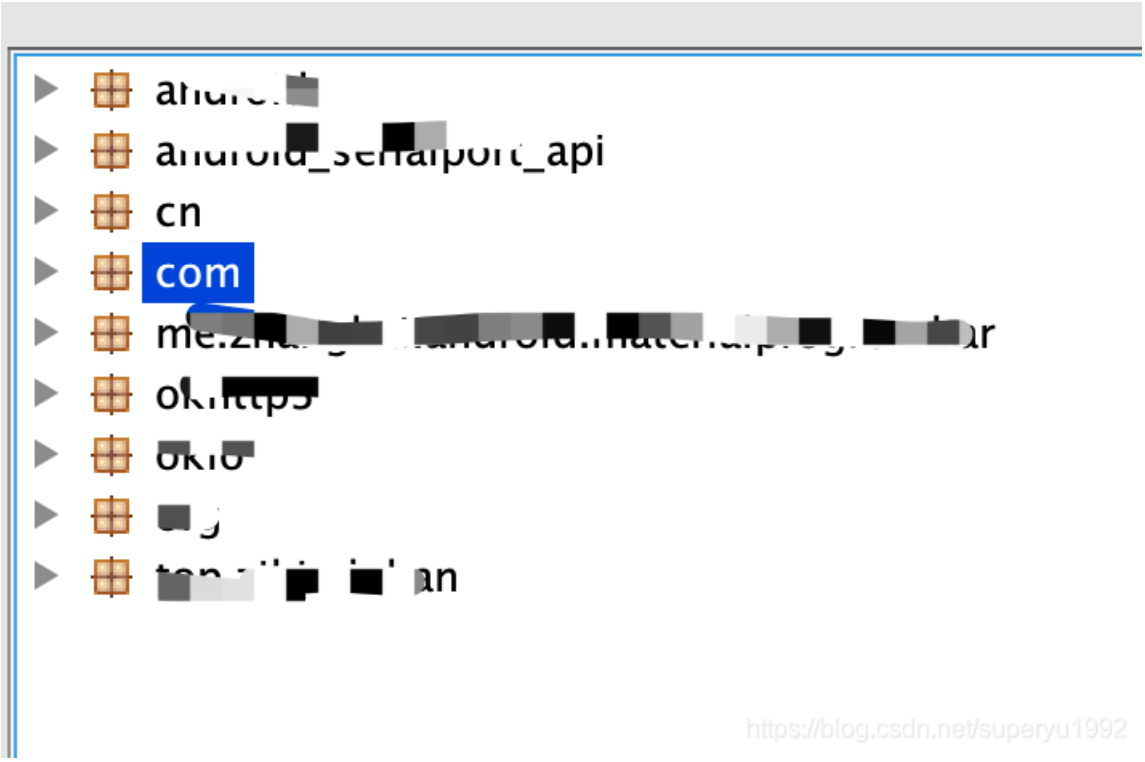
一、反编译基础三件套

首先要介绍一下反编译的基础三件套：apktool、dex2jar-2.0、jd-gui-osx；

1、apktool：apk在某种意义上来说也是一种压缩包，Android开发者应该都知道可以通过更改后缀名的方式得到App的资源文件，但是在这种方式下的manifest与XML文件都是乱码，无法查看，那么我们就需要使用apktool来获取可读取的资源文件了，命令也很简单：apktool d xxx.apk;

apktool安装与配置可参考：https://www.jianshu.com/p/c90024f61653

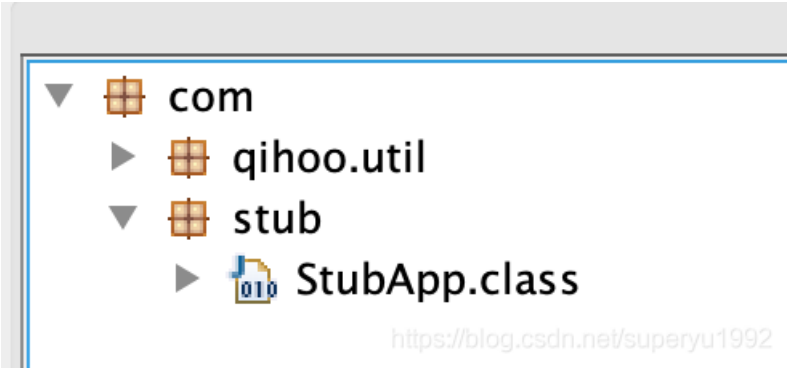
2、dex2jar-2.0+jd-gui-osx：这两个工具的联合使用，是为了查看App的源码的。首先我们apk的后缀名修改为zip，然后将其解压缩，将其中的dex文件拷贝到dex2jar-2.0，执行sh d2j-dex2jar.sh xxx.dex，便可将dex转换成jar；然后在jd-gui-osx下即可查看源码：



在Mac下使用dex2jar-2.0会出现权限问题，解决方法可参考：https://www.jianshu.com/p/f53b718d282b

二、利用Frida给加固过的App脱壳

通过基础三件套可以对没有加固的App实现反编译，但如果这个App它加固过了，那通过jd-gui-osx看到的就会是这样：



等等，qihoo.util?看来他用的是某60加固的，好的，要“对付”就是你们这些加固过的App！通过在百度上搜索qihoo.util，果真发现了有不少关于如何给360脱壳的文章，又通过一些搜索了解到一个叫frida的工具，可以实现脱壳，而且某60、某加密、某固等主流的加密工具，都在被脱之列！

1、frida简介

frida的原理在我理解就是，通过在PC上安装Frida，手机上运行frida-server，实现PC对手机的控制，同时通过js注入的方式，将dex从“壳”里“钩”出来。（如果只是想实现结果，可以不在意这些原理，直接用大神们提供的工具就好~）

它是一款基于Python的hook（钩子）工具，因此在安装它之前我们需要先配置Py环境，现在的frida仅支持3.7以下的环境，3.8以上的暂不支持，这一点需要注意，我就因为PC上的py版本太高，不得不重新安装py。

2、pc上安装frida和frida-server

我们通过pip3 install 安装frida、frida-server，这里也有一个注意点：如果你的测试设备是5.1.1的话，需要指定frida为12.1.0（**pip3 install == 12.1.0**）frida-tools为**1.2.0**，至于原因，后面再说。

在安装之前，还需要**手动下载与你py版本对应的egg文件**，否则在安装frida的过程中，会报一个找不到对应egg文件的错误。

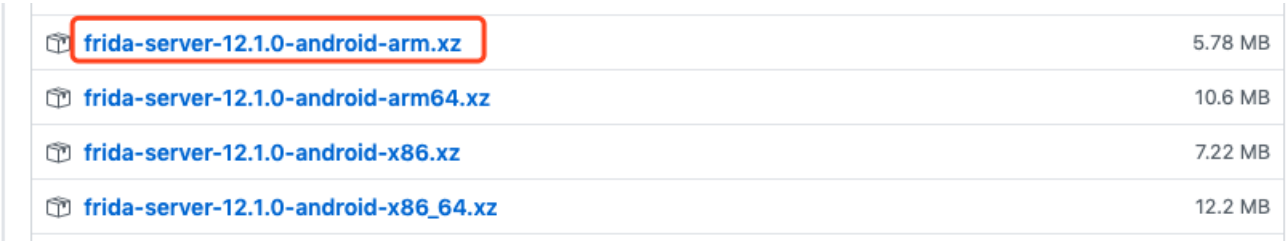
下载链接：https://github.com/frida/frida/releases

整个安装过程会比较慢，会卡在Running setup.py install for frida ... – 这里很久，一定要有耐心。如果在下载过程中出现超时，可以重新下载或者安装时添加参数 pip3 **--default-timeout=100** install -U xxx 。

安装完成后可以通过在python中，**import frida**来检测，如果没有报错，则代表安装成功。

3、在手机上运行frida-server

frida-server版本的选择由设备的内核版本与frida的版本决定，frida的版本已经确定，我们还需要确定设备的内核版本。在shell中，执行命令:**cat /proc/cpuinfo**查看系统内核。我的frida版本为12.1.0，设备内核版本为arm32位，因此选择:



下载地址：https://github.com/frida/frida/releases

下载完成后将文件解压，通过adb push导入到/data/local文件夹中,然后：

adb shell进入手机系统；

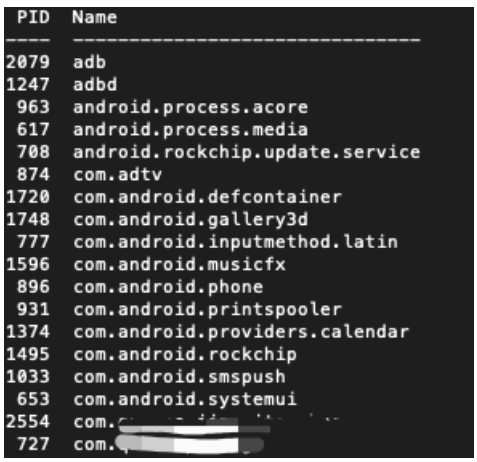
通过**su**切换刀root；

cd到**/data/local**；

赋予froda-server 777 权限：**chmod 777 frida-server**；

运行frida-server:**./frida-server**；还记得之前强调的frida版本问题吗？如果你在5.1.1的设备上，安装了高于12.1.0的frida-server上的话，这里执行就会报错：unused DT entry: type 0x6ffffef5 arg 0x1ddc，那么你又得重新安装frida、frida-tools、frida-server，那又将是一个漫长的过程了...

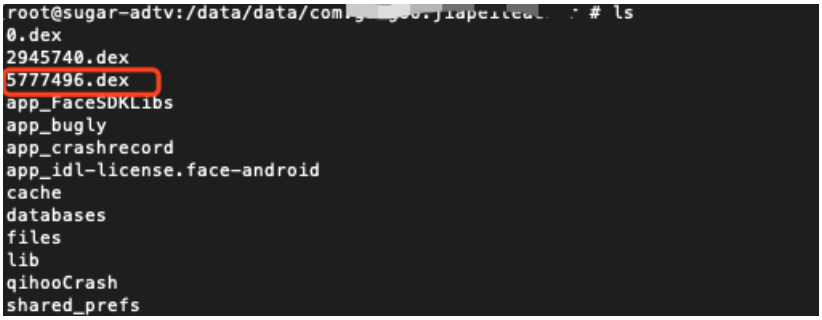
启动成功以后，新建一个终端窗口，输入：**frida-ps -U**，如果可以看到当前设备的进程和名称，则证明pc和手机通过frida联通了：



4、关键一步，利用“钩子”将dex脱出来：

以上所做的都是准备工作，下面就是真正的脱壳操作了，这里感谢“看雪”论坛里的大神，提供了可以直接脱壳的js，使得我们直接调用就可以了，献上原文链接：https://bbs.pediy.com/thread-251924.htm

按照大神的做法：**frida -U -f {包名} -l dexDump.js --no-pause**，就可以dump出dex了！包名可以从前面所说的apktool中反编译出的manifest中得到。**而dex文件会生成在/data/data/应用包名/目录下：**



接下来我们退出shell，通过adb pull将dex文件拉出，再通过jd-gui-osx查看：

当当，源码全都脱出来了！

三、总结

由此可以看出，其实现在这些所谓加固平台的加固，都只是防君子不防“小人”，甚至他们以加固作为App上架的条件，其真实目的不禁让人浮想联翩。而作为开发者，学习反编译的目的，不在于破解别人的App，剽窃别人的成果，而是通过了解其中的原理，更好的保护自己产品。当然啦，我这个也只是浅尝辄止，如果你对反编译或者逆向感兴趣，推荐一个叫做“看雪”论坛的网站，这也是我这两天发现的一个“宝藏”网站。以上。