目录

前言 1.1
概述 1.2
安卓背景知识 1.3
安卓基本框架 1.3.1
apk编译打包流程 1.3.2
破解apk的流程 1.3.3
相关知识 1.3.4
apk文件 1.3.4.1
dex文件 1.3.4.2
smali文件 1.3.4.3
安卓加密技术 1.4
为何要加密加固 1.4.1
加密技术历史 1.4.2
加密方案概述 1.4.3
代码混淆 1.4.3.1
加固方案 1.4.3.2
常见加固技术对比 1.4.3.2.1
常见加固方案 1.4.3.2.2
安卓破解技术 1.5
如何反混淆 1.5.1
如何去壳脱壳 1.5.2
如何从apk破解出java源码 1.5.3
破解apk概述 1.5.3.1
一步: apk->java 1.5.3.2
两或三步: app->dex->jar->java 1.5.3.3
1. app导出dex 1.5.3.3.1
2.1 dex转java 1.5.3.3.2
2.2.1 dex转换出jar 1.5.3.3.3
2.2.2 jar转换出java 1.5.3.3.4
常见安卓破解工具 1.6
从app导出dex 1.6.1
FDex2 1.6.1.1
DumpDex 1.6.1.2
drizzleDumper 1.6.1.3
DexExtractor 1.6.1.4
InDroid 1.6.1.5
dex转jar 1.6.2

dex2jar 1.6.2.1
Enjarify 1.6.2.2
Dedexer 1.6.2.3
反编译器 1.6.3
常见反编译器对比 1.6.3.1
常见反编译器 1.6.3.2
jadx 1.6.3.2.1
CFR 1.6.3.2.2
JD-GUI 1.6.3.2.3
Procyon 1.6.3.2.4
Luyten 1.6.3.2.4.1
Krakatau 1.6.3.2.5
Fernflower 1.6.3.2.6
GDA 1.6.3.2.7
Dare 1.6.3.2.8
JAD 1.6.3.2.9
其他破解类工具 1.6.4
apktool 1.6.4.1
smali/baksmali 1.6.4.2
IDA 1.6.4.3
radare2 1.6.4.4
安卓XML破解 1.6.4.5
AXMLPrinter2 1.6.4.5.1
其他辅助类工具 1.6.5
安卓调试器 1.6.5.1
AndBug 1.6.5.1.1
Android-OpenDebug 1.6.5.1.2
二进制编辑器 1.6.5.2
010editor 1.6.5.2.1
EverEdit 1.6.5.2.2
ClassyShark 1.6.5.3
APK Analyzer 1.6.5.4
Fino 1.6.5.5
Introspy-Android 1.6.5.6
redexer 1.6.5.7
SmaliViewer 1.6.5.8
其他综合类工具 1.6.6
ByteCode Viewer 1.6.6.1
Android Decompiler 1.6.6.2
decompile-apk 1.6.6.3

Android-Crack-Tool For Mac 1.6.6.4
Android逆向助手 1.6.6.5
AndroidKiller 1.6.6.6
Androguard 1.6.6.7
AFE 1.6.6.8
其他相关工具 1.6.7
安卓Hook框架 1.6.7.1
XPosed框架 1.6.7.1.1
Cydia Substrate 1.6.7.1.2
frida 1.6.7.1.3
安卓模拟器 1.6.7.2
夜神安卓模拟器 1.6.7.2.1
文件浏览器 1.6.7.3
附录 1.7
参考资料 1.7.1

安卓应用的安全和破解

- 最新版本：`v2.2`
- 更新时间：`20200905`

简介

总结安卓应用的安全措施和如何出于研究目的去破解安卓应用，其中介绍好多代的加密技术发展历史，包括常见的代码混淆，自我校验，dex文件变形，dex文件隐藏、so保护等等。总结了安卓的编译和反编译的基本流程和逻辑。整理了加密和解密，反编译，加固，脱壳等相关的工具和技术。典型的反编译流程包括，如何从apk反编译得到java源代码，如何从apk转换出dex文件，如何从dex文件转换出jar文件，如何从jar文件转换出java源代码等等原理和详细步骤。且总结了和安卓反编译、逆向工程、分析等相关的各种工具和软件。

源码+浏览+下载

本书的各种源码、在线浏览地址、多种格式文件下载如下：

Gitbook源码

- crifan/android_app_security_crack: 安卓应用的安全和破解

如何使用此Gitbook源码去生成发布为电子书

详见：crifan/gitbook_template: demo how to use crifan gitbook template and demo

在线浏览

- 安卓应用的安全和破解 book.crifan.com
- 安卓应用的安全和破解 crifan.github.io

离线下载阅读

- 安卓应用的安全和破解 PDF
- 安卓应用的安全和破解 ePub
- 安卓应用的安全和破解 Mobi

版权说明

此电子书教程的全部内容，如无特别说明，均为本人原创和整理。其中部分内容参考自网络，均已备注了出处。如有发现侵犯您版权，请通过邮箱联系我 `admin 艾特 crifan.com` ，我会尽快删除。谢谢合作。

鸣谢

感谢我的老婆陈雪的包容理解和悉心照料，才使得我 `crifan` 有更多精力去专注技术专研和整理归纳出这些电子书和技术教程，特此鸣谢。

更多其他电子书

本人 `crifan` 还写了其他 `100+` 本电子书教程，感兴趣可移步至：

crifan/crifan_ebook_readme: Crifan的电子书的使用说明

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2021-01-16 23:49:07

概述

之前有接触过安卓的apk应用程序的破解，现在去整理一下其中涉及到的各种安卓的安全技术和相应的破解技术。

不准把技术用于非法用途

警告⚠：相关破解技术仅限于技术研究使用，不准用于非法目的，否则后果自负。

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-24 20:49:31

安卓背景知识

在介绍安卓的安全技术和破解技术之前，需要先去了解相关的背景知识。

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 14:54:26

安卓基本框架

为了更好的理解，安卓破解相关知识和工具，先要了解安卓的基本框架，以及每一层中都是什么东西：

安卓的基本架构

- 内核层：
  - 支持多进程和多线程的Linux内核
  - 每个应用程序都有自己的Linux ID，并在单独的进程中运行
  - 具有相同ID的两个应用程序可以彼此交换数据。
- 系统运行层
  - 主要包括一些 `开源类库` 以及 `Android运行时环境`
    - 其中 `Dalvik` 虚拟机中运行的应用程序格式为 `dex` 的 `二进制文件`
- 应用框架层
  - 具有Java接口的应用程序框架
    - 主要组成
      - Android NDK
      - Android SDK
- 应用层
  - 预安装一些核心应用程序

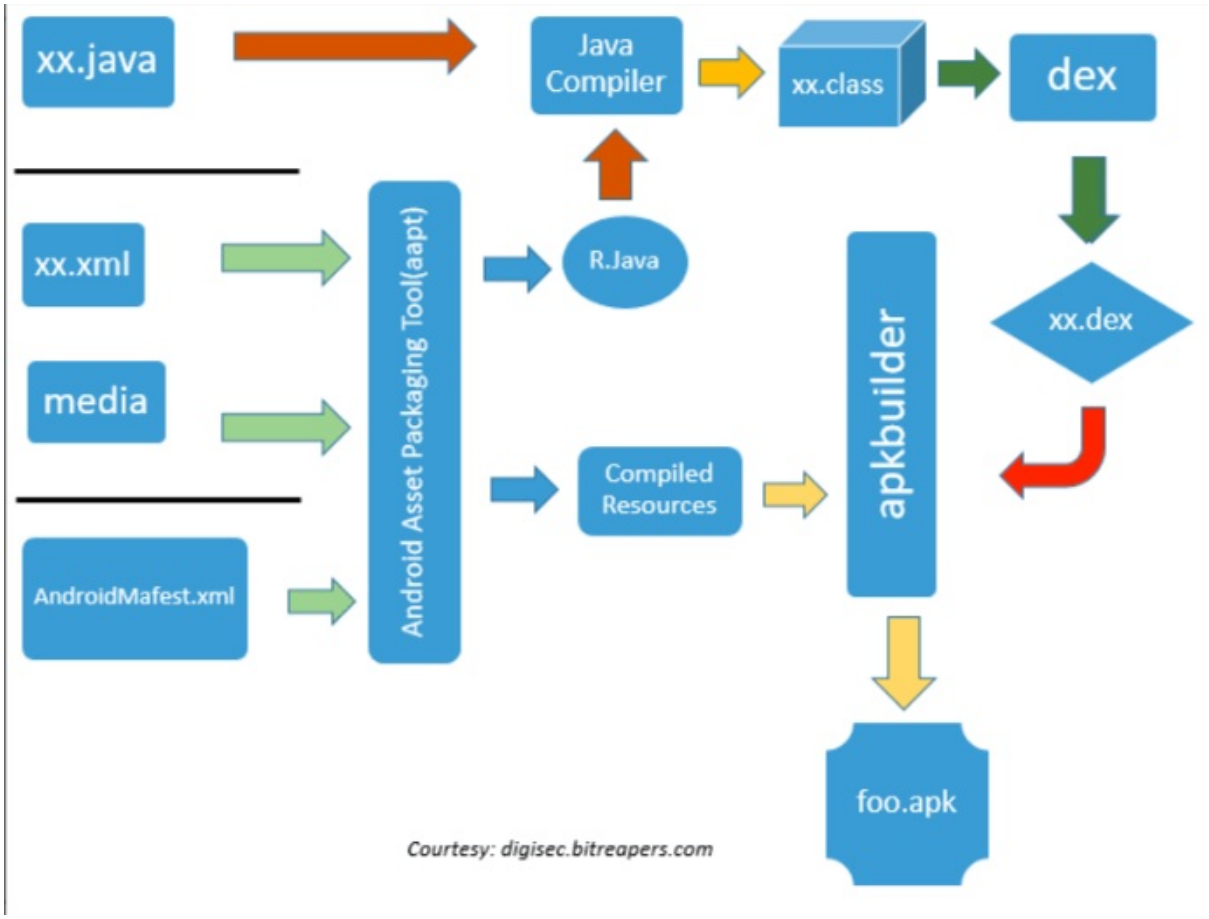
后续的很多破解工具，则是针对 `Dalvik` 虚拟机和dex文件去破解的。

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 14:53:45

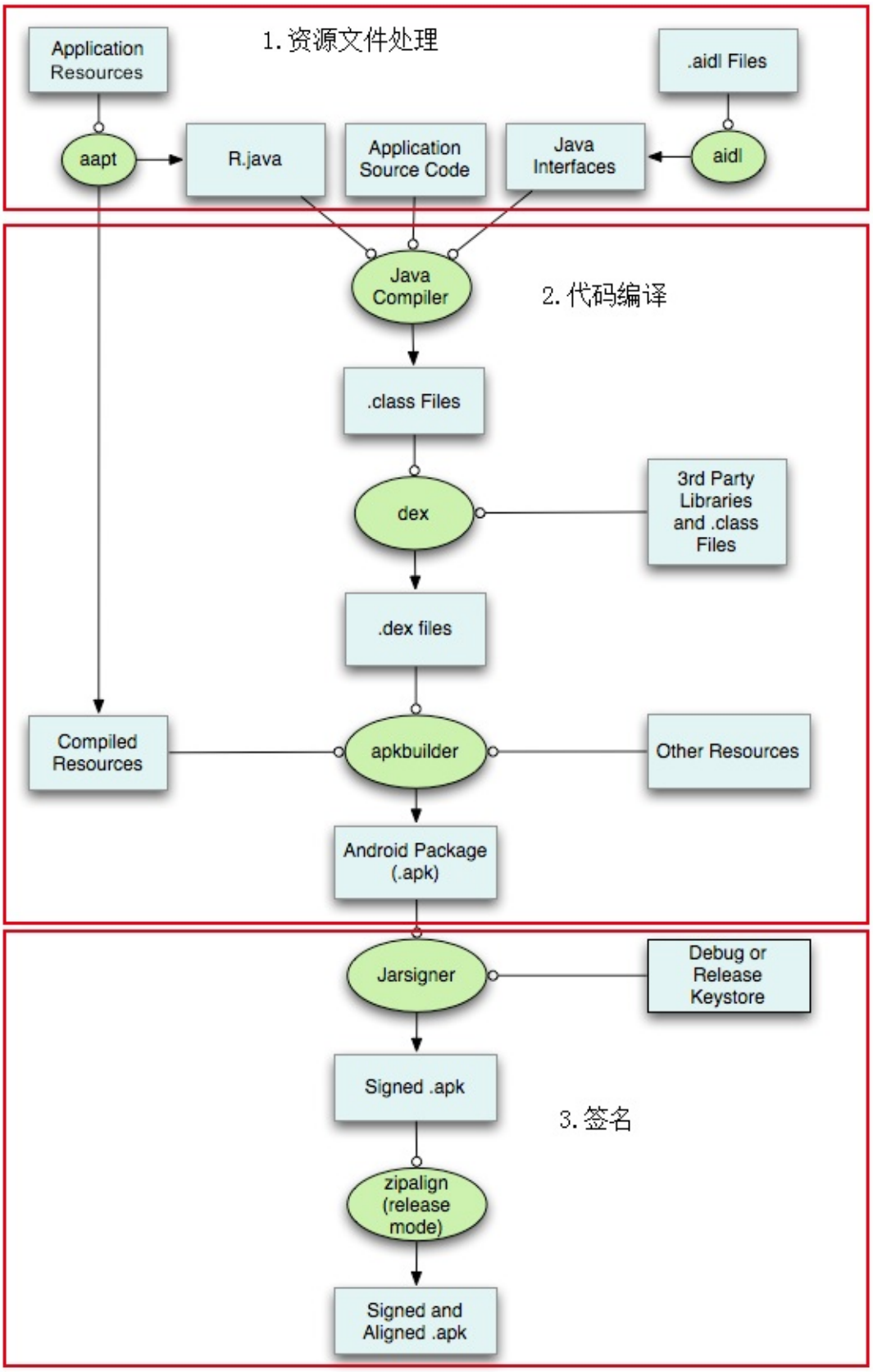
apk编译打包流程

APK打包流程

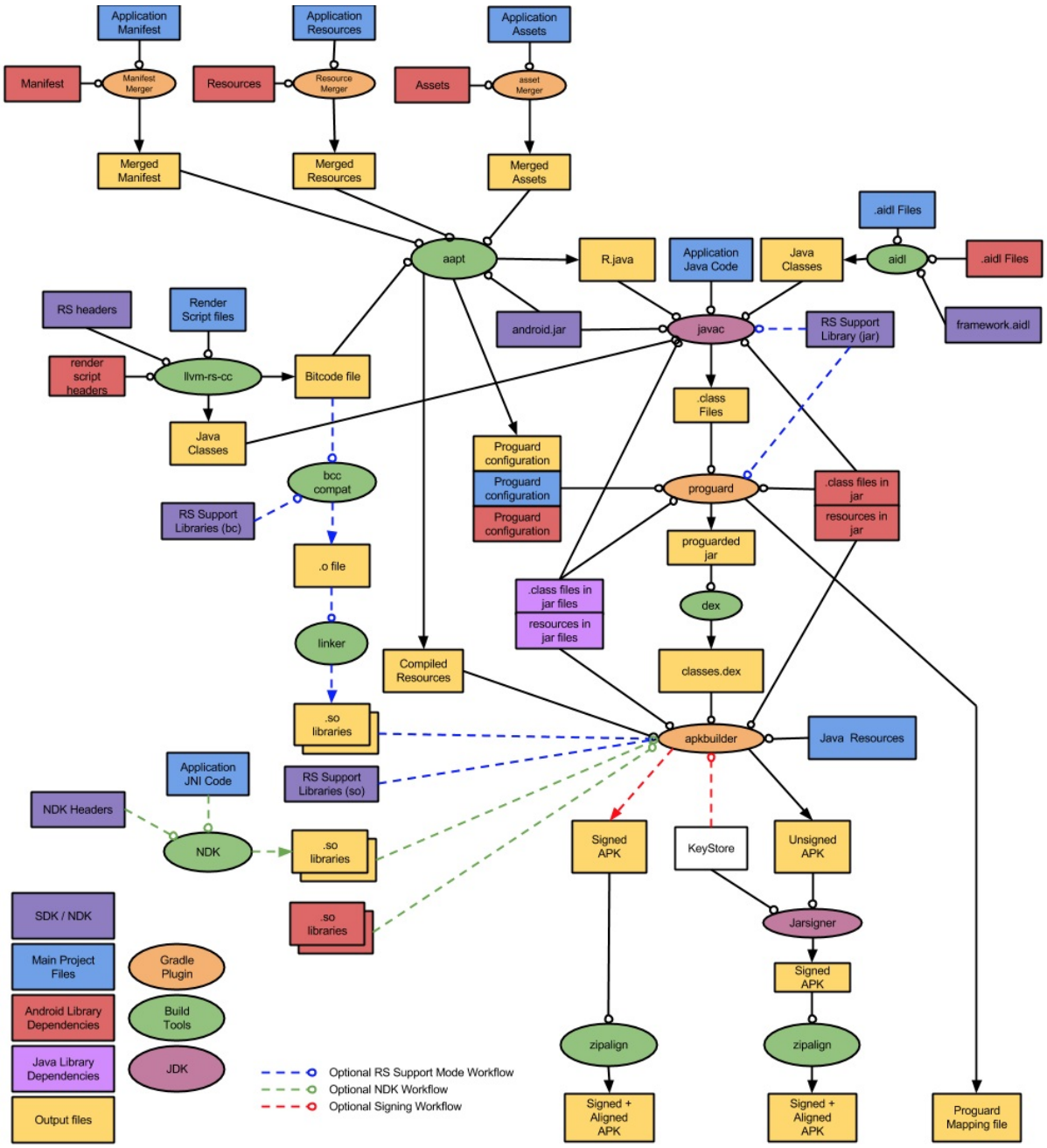
`apk` 打包流程图= `apk` 产生过程：



和


1. 资源文件处理
Application Resources
.aidl Files
aapt
R.java
Application Source Code
Java Interfaces
aidl
2. 代码编译
Java Compiler
.class Files
dex
3rd Party Libraries and .class Files
.dex files
Compiled Resources
apkbuilder
Other Resources
Android Package (.apk)
3. 签名
Jarsigner
Debug or Release Keystore
Signed .apk
zipalign (release mode)
Signed and Aligned .apk

和 安卓编译流程:



具体的解释是：

- 资源处理
  - 这一过程中主要
    - 使用appt工具进行资源文件的处理
      - 分析AndroidManifest.xml中的资源文件
      - 生成R.java和resources.arsc文件
    - aidl工具负责处理aidl文件
      - 生成对应的java接口文件
- 代码编译
  - 将上一过程中产生的R.java、java接口文件以及工程源代码一起通过Java Compiler编译成.class文件，打成Jar包
    - 这部分可以加入代码混淆）
      - 比如用 `ProGuard`
  - 然后与第三方库的Jar包一起通过dx工具转换成.dex文件

        - 注：如果apk的方法数超过了65535，会生成多个dex文件
            - 反编译的话需要对这多个dex文件均进行转换Jar包处理
    - 通过apkbuilder工具将aapt生成的resources.arsc、classes.dex（可能多个）、其他的资源一块打包生成未经签名的apk文件。
- 添加签名
    - 通过Jarsigner对生成的未签名的apk进行签名。
    - 再通过zipalign对签名后的apk进行对其处理，使apk中所有资源文件距离文件起始偏移为4字节的整数倍，从而在通过内存映射访问apk文件时会更快。

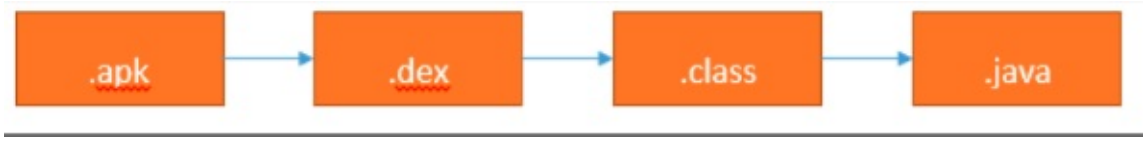
crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2020-01-16 22:59:46

破解apk的流程

简述安卓apk的产生过程：

- java源代码
  - `java编辑器` 编译
- class文件
  - `dx` 工具转换和打包压缩
    - 加上 第三方的，其他的库文件
- dex文件
  - apkbuilder打包
    - 加上 其他资源文件resources.arsc，其他库等
- （未签名的）apk文件
  - jarsigner去签名 + zipalign去处理
- （已签名的）apk文件
  - 可以用于发布和上架各种安卓应用市场
    - 供普通用户下载安装试用

-》想要破解安卓apk，也就是反向操作了：



- 从apk（安装后的运行期间的app）反向（用hook机制去dump）导出dex文件
  - 如果是普通加固
    - 用FDex2等工具是可以成功导出dex的
  - 如果是各家收费的高级的加固方案
    - 估计就比较困难了
- 从dex文件反编译出jar包（内部就是各种class了）
  - 有的dex反编译会出现各种错误
    - 估计是加固的方案比较高级导致的
  - 有的dex反编译没有出错
    - 如果又是我们希望的包含了app业务逻辑的代码
      - 那后续就可以完美的破解得到程序的java源代码了
- 从jar包反编译出java源代码
  - 即可查看和导出全部的java源代码了
  - 注意：
    - 当前如果之前混淆了代码
      - 最后此处得到的也是混淆后的代码
        - 不容易看出原始代码的业务逻辑
  - 说明
    - 此过程和之前的编译对应，所以严格的叫法就是所谓的：
      - 反编译=decompile
        - 对应工具才叫做：decompiler
          - 反编译器=解码器=逆编译程序

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 14:54:23

相关知识

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 11:00:12

apk文件

- `apk` = `android Application PacKage` = `APK`
- apk文件是啥
  - 是安卓app的安装文件
  - apk文件其实就是个zip压缩包
    - 把apk后缀改为zip，即可解压得到一堆安卓相关文件

apk内容结构

内容入口	含义解释
AndroidManifest.xml	二进制xml文件，提供设备运行应用程序所需的各种信息
classes.dex	以dex格式编译的应用程序代码
resources.arsc	包含预编译应用程序资源的二进制XML文件
res/	此文件夹中包含未编译到resources.arsc文件中的资源
assets/	此文件夹包含应用程序的原始资产，由AssetManager提供对这些资产文件的访问
META-INF/	它包含MANIFEST.MF文件，该文件存储有关JAR内容的元数据。APK的签名也存储在此文件夹中
lib/	该文件夹包含已编译的代码，例如本地代码库

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-25 14:19:49

dex文件

什么是dex文件

简答：

- `dex` = `Dalvik EXecutable format` = `dex` 文件
  - dex文件，逻辑是类似于 `java` 的 `class` 文件
    - 注：java的class文件内部是Java的字节码(Java bytecode)

详解：

安卓系统中，用 `Dalvik虚拟机` (`DVM` = `Dalvik Virtual Machine`)去把 `java` 源码编译为 `dex` 可执行文件(Dalvik Executable)。

而dex文件中保存的就是：编译后了的安卓程序代码文件

Dex文件内部格式

1. File Header
2. String Table
3. Class List
4. Field Table
5. Method Table
6. Class Definition Table
7. Field List
8. Method List
9. Code Header10. Local Variable List

相关工具

Android自带 `dexdump` ：用来反编译 `dex` 文件

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-25 14:20:24

smali文件

- Baksmali分别是指安卓系统里的Java虚拟机（Dalvik）所使用的一种.dex格式文件的汇编器，反汇编器。
- 其语法是一种宽松式的Jasmin/dedexer语法，而且它实现了.dex格式所有功能（注解，调试信息，线路信息等）

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 11:00:18

安卓加密技术

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:25:03

为何要加密加固

- 安卓应用主要基于Java开发
  - 极易被破解
    - 造成影响
      - 代码或关键接口暴露
      - 甚至被别人加入广告，病毒等二次打包发布
    - 给公司和用户均带来巨大的风险
  - 应对破解的最便捷有效的方式
    - 加固
      - 通过加固可以在一定程度上达到反编译和防止被二次打包的效果
- 其他一些原因
  - 处于学习目的，想要了解、分析、学习某个安卓app的内部设计和代码逻辑
    - 所以需要反编译和破解
      - 所以防止别人破解要加密和加固

但是加固也有些缺点：

- 加固后对应用的影响
  - 体积
  - 启动速度
  - 兼容性
    - 各个平台能正常运行，不会崩溃
  - 使用成本
    - 有些加固方案需要收费
  - 客服响应速度
    - 有些平台的加密出问题后的响应速度不同

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:25:17

加密技术历史

下面整理安卓的加密技术的历史发展。

用人类历史发展的阶段去类比解释如下：

原始社会时期

主要方式： `代码混淆`

奴隶社会时期

主要方式： `自我校验`

封建社会时期

主要方式： `dex文件变形`

资本主义社会时期

1. Dex保护
   i. 隐藏dex文件
      - 既然dex文件中包含了核心逻辑，那么把dex隐藏，再通过另外的方式加载起来，是不是就能达到保护dex的目的了呢?于是这成为一些第三方加固产品保护应用的方式。
      - 他们通过加密甚至压缩(早期是不存在压缩的，只是单纯的加密)方式把dex转换为另外一个文件。而被加固后的apk里面的dex则是那些第三方加固产品用来启动和加载隐藏dex的入口，也就是壳。
      - 感觉小花生v3.6.9 和 康美通 v.4.4.0就是这类?
      - 总之是看不到dex文件
   ii. 对dex文件进行变形
      - 这里所说的变形，不同于封建社会时期提到的变形。这种办法不隐藏dex，而是让dex保留在外面，但是当破解者去分析这个dex的时候，会发现dex里面的内容是不完整的。
   iii. 对dex结构进行变形
      - 此类方法是比较复杂的，了解dex结构的人应该很清楚，dex结构中包含DexClassDef、ClassDataItem、DexCode，这些是dalvik虚拟机运行一个dex必不可少的部分，特别是DexCode，DexCode包含了虚拟机运行的字节码指令。
      - 部分第三方加固产品开始尝试这种方式，他们的保护方案中可能抽取了DexCode中的部分，然后对字节码指令添加nop，或者连ClassDataItem和DexCode一同抽取，或者对上面提到的三个部分都做处理。抽取完之后，还要做修正、修复等工作，总之很烦锁。因为dex运行时有很多关于dex的校验，即使校验通过还有一些偏移问题。
      - Dex都被抽取修改后为什么还能运行呢?那是因为在运行之前或者运行之中对这个内存中的dex做修正。修正工作也很复杂，一般选择在运行之前做修正，这样可以减少很大的工作量，甚至可能还需要借助hook来帮忙。
2. So保护
   i. 修改Elf头、节表
      - 相关工具：
        - 010 Editor

        - IDA
    ii. 选择开源加壳工具
        - 最常用的：
            - UPX壳
                - 支持arm架构的ELF加固
    iii. 进程防调试、或增加调试难度
        - 调试一个进程首先要ptrace这个进程
            - 防止进程被ptrace

社会主义时期

- 之前遗留问题
    1. 隐藏dex遗留的问题
        - 破解办法：
            - 实现自定义rom
            - 利用Inject原理将目标进程注入,代码进行hook系统函数来达到脱壳的目的
                - FDex2，DumpDex感觉就是用的这个机制?
    2. Dex结构变形带来的弊端
        - 安卓5.0新增了ART
            - ART可以直接将dex编译为本地指令运行
        - Dex结构变形遗留的问题很明显
            - 兼容性
            - ART模式下的编译问题
    3. ELF简单修改遗留问题
    4. UPX方面的劣势
        - 虽然upx是最为so加壳的首选，但是upx代码逻辑复杂，很难达到定制，特别是让它同时支持多种架构
            - -》基于上述原因一些第三方加固产品只是简单的利用upx加壳，并修改一些数据。不过很容易被有upx经验的人识破并脱壳

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:24:58

加密方案概述

下面整理一下常用的安卓的加密技术和方案：

- 代码混淆
  - 常用（Android自带的） `ProGuard`
- 加固
  - 发展历史
    - 目前加固技术基本都发展到第三代
      - 前2代的加固技术破解难度不大，基本被淘汰
    - 第三代加固技术，由于各加固服务商加固原理大致相同
  - 第三代加固技术主要有2种方式：
    - 对源apk整体做一个加固，放到指定位置，运行的时候再解密动态加载
      - 对apk加固的破解，叫做：脱壳=去壳
        - == Dex Method代码动态解密
    - 对so进行加固，在so加载内存的时候进行解密释放
      - 对so的加固的破解，叫做：so库反编译
        - == So代码膨胀混淆

另外参考这里的总结：

- 鉴于现在很多破解脱壳方案，都是基于 `Hook` 框架，去从安卓app中导出dex文件
  - 典型的hook框架
    - `XPosed` ：从根上Hook了Android Java虚拟机
    - `Cydia` ：支持jni和java层的HOOK功能
  - 再去从dex中转换出java代码
- 而各大安卓厂商意识到了，如果只是代码用Java写，代码运行在Java层，则被破解和被篡改的概率很大
  - 所以很多都把重要代码放到 `JIN` 层了
    - 比如：
      - 微信 的 数据库的连接操作
  - 不过从破解的技术角度说
    - 你 `加壳` -》 我 `脱壳`
    - 你把操作放在 `so` -> 我就用 `IDA` 调试
    - 你有反调试(检测？) -> 我就绕过去
  - 当然这样会增加破解难度
    - 对于普通技术不行的，还是可以防得住的
    - 但是对于高手，有一定毅力和能力，还是可以破解的

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:24:43

代码混淆

下面详细介绍安卓代码混淆的技术方案。

最常见的做法是，用 `ProGuard` 去实现代码混淆。

ProGuard

ProGuard的作用

- 压缩=Shrinking
  - 移除未被使用的类、属性、方法等，并且会在优化动作执行之后再次执行（因为优化后可能会再次暴露一些未被使用的类和成员
- 优化=Optimization
  - 优化字节码，并删除未使用的结构
- 混淆=Obfuscation
  - 将类名、属性名、方法名混淆为难以读懂的字母，比如a,b,c等，增大反编译难度

ProGuard的输出文件说明

- `dump.txt` ： 说明 APK 中所有类文件的内部结构
- `mapping.txt` ： 提供原始与混淆过的类、方法和字段名称之间的转换和对应关系
- `seeds.txt` ： 列出未进行混淆的类和成员
- `usage.txt` ： 列出从 APK 移除的代码

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:24:37

加固方案

安卓加密安全技术中，除了基本的代码混淆外，更多的是采用加固技术。

下面详细介绍加固的常见技术和方案。

`加固` 的英文说法

对于安卓apk的加固的英文说法：

- 自己暂时用 `harden`
  - 被加固了的（apk）就叫： `hardened`
- 其他（好像）也有说成 `pack` ， `packing` 的?

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:24:22

常见加固技术对比

常见Android APK的五代加固技术比较：

市面上最为常见的 Android (APK) 五代加固技术比较

阶段	名称	开发过程	核心逻辑（执行过程）	不足	优势	破解状况	实例
第一代	动态加载	程序切分成加载（Loader）与关键逻辑（Payload）两部分，并分别打包。	运行时加载部分（Loader）会先运行，释放出关键逻辑（Payload），然后java的动态加载技术进行加载，并转交控制权。	1、Payload部分必须解压及释放在文件系统（直接获取）。 2、通过自定义虚拟机，截获关键函数，在加载Payload时把解密后的内容复制。	加密Payload保存	目前基本上已经被破解，部分反编译工具已经集成修复功能。	早期版本的爱加密
第二代	内存不落地加载	加载Loader，初始化StubApplication，解密和加载Payload，初始化原始Application，用原始Application替代StubApplication，最后正常加载其他组件。	1、拦截系统IO相关的函数（如read、write），在这些函数中提供透明加解密； 2、直接调用虚拟机提供的函数进行不落地的加载。	1、在启动过程中需要处理大量解密操作，容易造成黑屏或假死。 2、Payload被加载后，在内存中是连接，内存dump即可直接获取。 针对关键函数进行拦截即可把内存dump	当前市面上最为常见，通常作为一项基础性的免费服务向用户提供。	已经出现专业人士自行研究的手工脱壳方法，但尚未出现自动脱壳工具，破解难度仍然比较大。	市面上流行的大多数在线加固服务，如腾讯乐固，360加固，百度加固，阿里聚安全，网易易盾等等。
第三代	指令抽取	首先，将保护级别降到了函数级别，然后将原始DEX内的函数内容（Code Item）清除，单独移除到一个文件中，运行阶段将函数内容重新恢复到对应的函数体。	1、加载之后恢复函数内容到DEX壳所在的内存区域。 2、加载之后将函数内容恢复到虚拟机内部的结构体上：虚拟机读取DEX文件后内部对每一个函数有一个结构体，这个结构体上有一个指针指向函数内容（CodeItem），可以通过修改这个指针修改对应的函数内容。 3、拦截虚拟机内与查找执行代码相关的函数，返回函数内容。	1、指令抽离方案跟虚拟机的JIT性能优化冲突，达不到性能最佳的性能。 2、依然使用Java虚以机进行函数内容执行，无法对抗自定义虚拟机。 3、指令抽离技术使用了大量的虚拟内部结构与未被文档的特性，再加上Android复杂的厂商定制，带来大量的兼容性问题。	在对自定义虚拟机记录函数执行时函数的内容（CodeItem），遍历触发所有的函数，从而获取全部抽离的内容，最终组装成完整的DEX文件。可通过自动化完成整个过程。	部分被破解，已经出现专业人士自行研究的手工破解方法，但是迄今为止尚未出现自动脱壳工具。	1. 现在的免费版“爱加密”。 2. 梆梆安全免费版。
第四代	指令转换	1、DEX文件内的函数被标记为native，内容被抽离并转换成一个符合JNI要求的动态库。动态库内通过JNI和Android系统进行交互。 2、DEX文件内的函数被标记为native，内容被抽离并转换成自定义的指令格式，该格式使用自定义接收器执行，和A一样需要使用JNI和Android系统进行调用。	1、不论使用指令转换/VMP加固的A方案或者B方案，其必须通过虚拟机提供的JNI接口与虚拟机进行交互，攻击者可以直接将指令转换/VMP加固方案当作黑盒，通过自定义的JNI接口对象，对黑盒内部进行探测、记录和分析，进而得到完整DEX程序。 2、第四代VMP加固技术一般配合第三代加固技术使用，所以第三代的所有兼容性问题，指令转换/VMP加固也存在。	1、不无摆脱对JNI的依赖，因此依然存在第四代加固技术的缺陷，存在被记录修复的可能性。 2、由Java转换成等价的C/C++，会导致体积呈线性增大，性能有所下降。	1、由Java转C/C++后的代码，由于虚机的保护，逆向难度会上升一个数量级。 2、对于C/C++部分逻辑，只能大量投入时间去破译虚机的指令集含义。	部分被破解，已经出现专业人士自行研究的手工破解方法，但是迄今为止尚未出现自动脱壳工具。	大部分需要定制收费的加密服务（如爱加密，梆梆安全，中国移动加固，以及手机银行自研加固等）
第五代	虚拟机源码保护	基于第四代方案的A方式（Java或Kotlin -> C/C++），基于LLVM编译工具链（同时支持C/C++, Swift, Object-C）；通过对IR进行指令转换，生成自定义指令集（IR->VM），App内部隔离出独立的执行环境，该核心代码的运行程序在此独立的执行环境里运行。	1、无法摆脱对JNI的依赖，因此依然存在第四代加固技术的缺陷，存在被记录修复的可能性。 2、由Java转换成等价的C/C++，会导致体积呈线性增大，性能有所下降。	1、由Java转C/C++后的代码，由于虚机的保护，逆向难度会上升一个数量级。 2、对于C/C++部分逻辑，只能大量投入时间去破译虚机的指令集含义。	大多数未被破解	极为少数，需要特殊定制的加固服务，通常用于银行金融机构等关乎国家安全的重点领域。

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:24:05

常见加固方案

常见加固服务提供商

市面上有很多家公司、厂家提供了免费或收费的加固方案：

360加固宝

- 主页：http://jiagu.360.cn/
  - 还有其他产品：
    - 针对手游的：
      - 360手游保
      - http://shouyoubao.360.cn
    - 针对网页的：
      - H5加固

腾讯乐固legu

- 主页：http://legu.qcloud.com/

的 `应用安全` = `Mobile Security` = `MS`

- 功能：为用户提供移动应用（APP）全生命周期的一站式安全解决方案
- 涵盖服务
  - 应用加固
  - 安全测评
  - 兼容性测试
  - 盗版监控
  - 崩溃监测
  - 安全组件
- 特点
  - 坚固
    - 应用加固在不改 Android 应用源代码的情况下，将针对应用各种安全缺陷的加固保护技术集成到应用APK，从而提升应用的整体安全水平，力保应用不被盗版侵权
  - 稳定
    - 应用安全提供的安全能力可在复杂环境下稳定运行，兼容性高、崩溃率低；不仅支持 arm、aarch64、x86、x64，还支持 android 2.0 到 android N 等几乎全系统全机型

机制和原理分析：

乐固做了一些反调试的东西，很多情况并不是反调试越厉害加固就越好 乐固仍然是常规规的函数调用和返回方式，流程清晰很多

爱加密

- http://ijiami.cn
- 移动应用安全加固
  - 安卓：http://ijiami.cn/android
    - 核心技术
      - 防逆向

            - 通过DEX加花和加壳、SO文件高级混淆和加壳等技术对DEX和SO文件进行保护，防止被IDA等逆向工具分析
        - 防调试
            - 多重加密技术防止代码注入，防JAVA层/C层动态调试、防代码注入和防HOOK攻击
        - 页面数据防护
            - 应用防劫持、应用防截屏、虚拟键盘SDK产品和技术，防界面劫持插件对组件进行全方位监听
        - 防篡改
            - 在加固时提取APP内各文件的文件特征值，当文件运行时，系统解密加密文件提取特征值进行文件校验
        - 数据防泄漏
            - 使用多种加密算法，包括国际通用算法及自主研发的加密算法等，保护本地数据
        - 传输数据防护
            - 在客户端和服务器分别嵌入数据加密SDK，保证通道中传输的数据为高强度加密后的数据
    - 主要功能
        - 提升APP安全性
            - 源代码保护、SO库保护、DEX 文件保护、数据加密保护
        - 确保APP业务安全
            - 防盗版保护、防篡改、页面防劫持技术、防截屏技术、 环境清场、短信防劫持
        - 保障APP数据安全
            - 密钥白盒技术、APP通讯链路加密、数据本地加密技术、安全键盘
        - 确保APP的整体优化
            - APP包体大小不超过原包“±5%”、全面的兼容性测试、全面的性能测试
  - iOS：http://ijiami.cn/iosProtect
  - SDK：http://ijiami.cn/sdkProtection
  - SO库：http://ijiami.cn/soProtect
    - SO文件加壳保护
        - 对SO文件进行加壳保护，加壳后使用ida工具无法看出SO库文件的导入导出函数以及定位源码，有效防止黑客反编译，解包后看到真正源码
    - SO文件混淆保护
        - 爱加密基于移动安全领域的先进的技术和经验，针对黑客在分析阶段的攻击手段和行为进行分析，利用SO混淆编译器，可以有效的增加黑客信息搜集的难度和复杂度，防止应用被破解，降低APP安全风险

梆梆安全

- 主页
  - 梆梆安全 - 移动安全领导品牌，保护智能生活，共建智慧城市！
- 产品
  - 泰固
    - 移动应用安全加固
        - 针对目前移动应用普遍存在的破解、篡改、盗版、钓鱼欺诈、内存调试、数据窃取等各类安全风险，梆梆安全为开发者提供全面的移动应用加固加密技术和攻击防范服务
        - 核心加固技术
            - 防逆向（Anti-RE）：抽取classes.dex中的所有代码，剥离敏感函数功能，混淆关键逻辑代码，整体文件深度加密加壳，防止通过apktool，dex2jar，JEB等静态工具来查看应用的Java层代码，防止通过IDA，readelf等工具对so里面的逻辑进行分析，保护native代码。
            - 防篡改（Anti-tamper）：每个代码文件、资源文件、配置文件都会分配唯一识别指纹，替换任何一个文件，将会导致应用无法运行，存档替换、病毒广告植入、内购破解、功能屏蔽等恶意行为将无法实施。
            - 防调试（Anti-debug）：多重加密技术防止代码注入，彻底屏蔽游戏外挂、应用辅助工具，避免钓鱼攻击、交易劫持、数据修改等调试行为。

            - 防窃取（Storage Encryption）：支持存储数据加密，提供输入键盘保护、通讯协议加密、内存数据变换、异常进程动态跟踪等安防技术，有效防止针对应用动、静态数据的捕获、劫持和篡改。
        - 加固服务策略
            - 提供灵活多样的App加固方案
                - 防逆向保护
                - DEX文件加壳保护
                - DEX函数抽取加密
                - HTML开发框架保护
                - SO文件加壳保护
                - SO代码压缩及加密
                - SO库设备绑定
                - 源代码深度混淆
                - 防篡改保护
                - 开发者签名校验
                - 代码、资源文件、配置文件完整性校验
                - 数据透明加密及设备绑定
                - 配置文件、数据库文件加密
                - 视频、音频、图像等文件加密
                - 防调试保护
                - 防止进程/线程附加
                - 防止进程注入
                - 防HOOK攻击
                - 防内存Dump
                - App完整性保护
                - SO数据动态清除
                - 防窃取保护
                - 本地数据文件加密
                - 键盘数据加密
                - 通讯协议加密
                - 密钥白盒加密
    - 移动应用源代码加固
        - 未经混淆的源代码受到攻击后，易暴露程序中关键算法、核心业务逻辑、数据结构和模块的控制流布局等敏感内容

顶象安全

- 主页
    - 顶象技术 - 金融业务安全的实践者，专注智能风控与全链路反欺诈
- 功能
    - Android加固保护 - 顶象技术
        - 一套纵深防御体系
            - 分别从防逆向、防调试和防篡改等几个维度提供安全保护
            - 同时针对每个维度提供又进行了不同层次的划分，加固策可依据实际场景进行动态调配，安全和性能达到完美平衡。
        - 对APK提供DEX保护、SO保护、数据加密、资源防篡改、运行时保护等多角度全方位的保护
    - 核心功能
        - Dex加壳保护
            - 对dex文件整体进行加密隐藏，并进行反编译保护处理
        - Dex VMP保护
            - 将函数方法指令翻译为自定义指令集并加密存储，在运行过程中交由顶象dex虚拟机解释执行

        - SO加壳保护
            - 加密隐藏SO文件中的导入导出符号表以及常量字符串，同时对SO进行反编译保护处理
        - SO VMP保护
            - 对二进制函数指令翻译为自定义指令集并加密存储，在运行过程中交由顶象dex虚拟机解释执行
    - 特点与优势
        - 平台兼容性高
            - 支持ARM, ARM64, x86, x86_64, mips等多种cpu框架
        - 语言支持丰富
            - 支持包括Java，Kotlin，C/C++，Objective-C，Swift等在内的多种源码类型或混合型项目
        - 操作流程便捷
            - 把编译好的App上传并选择好加固方案即可完成整个加固流程，操作简单易懂，非技术人员也能轻松掌握
        - 体积增量小
            - 整体加固增量不会超过100kb，一般情况下是80kb
    - 应对风险
        - 程序逻辑被破解
        - 核心代码被窃取
        - API接口暴露
        - 恶意代码注入

几维安全

- 主页
    - 几维安全 - 让万物互联更安全
- 功能
    - APP应用加固
        - 重点
            - 高效、专业、兼容好
            - 5分钟极速加密，轻松集成DEX加密、反调试、防盗版等多重安全防护
        - 产品简介
            - 移动应用安全加固是一项面向互联网企业和个人开发者的在线加密服务， 现支持安卓应用加密，用户只需提供APK包即可快速集成防静态工具分析、Dex文件保护、So文件加壳、内存保护、反调试、防二次打包等多项安全功能。 支持对金融、手游、电商、社交等多个行业的应用做加固保护，避免核心代码被反编译， 请求协议被伪造，APK包被植入恶意代码等诸多安全问题。
        - 功能特点
            - Dex文件保护
                - 对DEX文件进行加密保护，防止被Dex2Jar等工具逆向破解
            - Dex-Java2C
                - 将Java代码翻译为C代码，并实施Native层的代码混淆保护
            - SO文件加壳
                - 对SO文件进行整体加壳保护，防止IDA Pro等工具逆向分析
            - 防二次打包
                - 集成正版签名校验功能，运行时动态校验，防止被植入恶意代码
            - 内存加密
                - 防止内存数据被篡改或Dump，比如Dump解密后的Java代码
            - 反调试
                - 拒绝调试器对当前应用的附加操作，防止程序被恶意调试分析
            - 自身虚拟化保护
                - 专业版采用代码虚拟化技术对自身代码进行保护，防止逆向分析
            - 兼容性良好
                - 测试覆盖200+机型，兼容性达到99%，支持ART模式

        - 应对风险
            - Dex文件反编译
            - So文件反编译
            - 核心技术窃取
            - 通信模块破解
            - API接口暴露
            - 密钥窃取
            - 注入恶意代码
            - 伪造盗版应用
    - SO源码混淆保护
        - 重点
            - 基于NDK项目源码，通过安装NDK插件即可集成代码混淆、轻量虚拟化、字符串加密等多项高强度的安全保护
        - 产品简介
            - SO库源代码保护是一款离线的安全编译器工具，主要用于保护Android NDK项目中的核心代码，避免因逆向工程或破解，造成核心技术被泄漏、代码执行流程被分析等安全问题。该安全编译器和普通编译器相似，基于项目源代码可将C、C++代码编译成二进制代码，不同之处在于，安全编译器在编译的时，能够对代码进行混淆、轻量虚拟化、字符串加密等安全保护。从而避免攻击者通过IDA Pro等逆向工具反编译二进制代码，分析业务代码执行流程，进一步篡改或窃取核心技术。

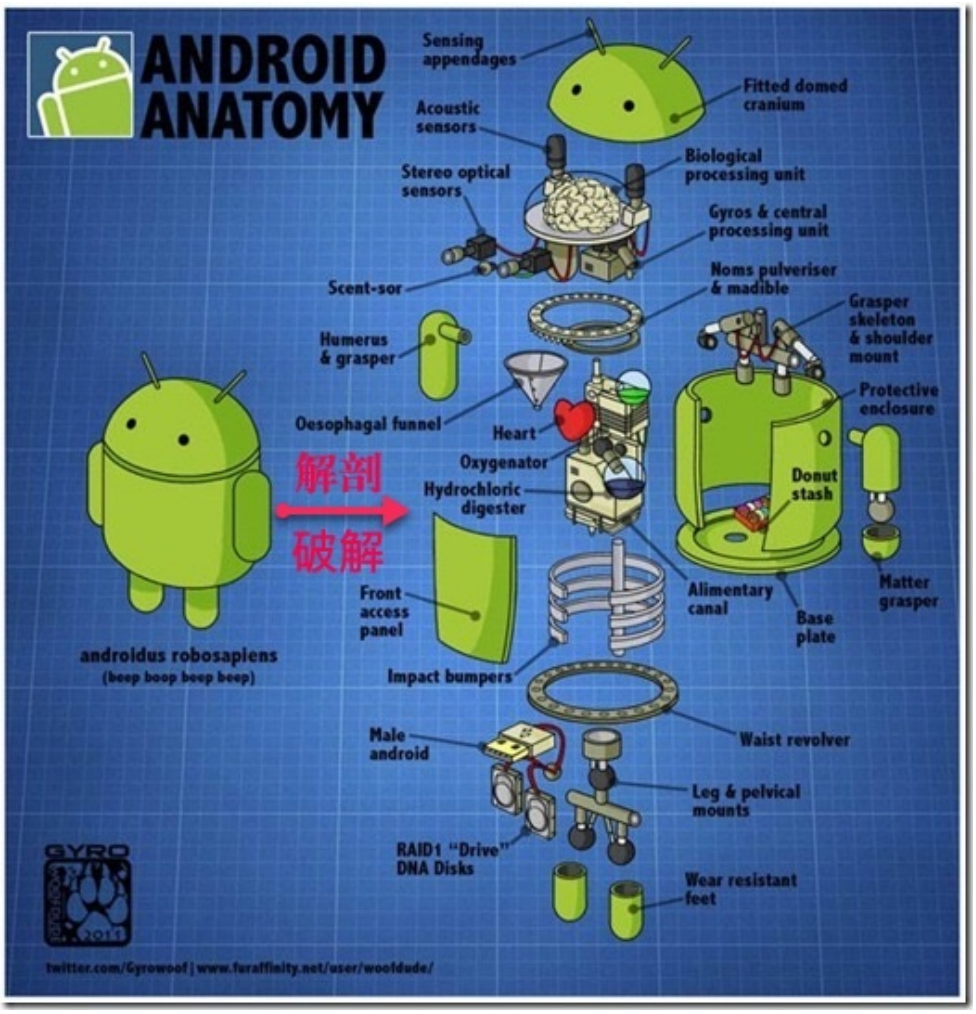
阿里聚安全

- http://jaq.alibaba.com/
    - 20180801已下线
- 原理分析：
    - Ali 利用 SP 来储存返回地址和参数所以整个流程很混乱，基本看上去就是在各种 JUMP

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:24:18

安卓破解技术

安卓破解相关的叫法有很多，整理如下：

- 安卓应用的 `存在形式`
  - 静态的
    - 安装文件： `apk`
  - 动态的
    - 运行中的程序叫： `app`
- `安卓破解` = `android crack`
  - 破解
    - = `crack` = `cracking`
    - 也叫 `安卓逆向工程` = `安卓反向工程`
      - 而逆向工程，就像对一个人去解剖
        - 

  - 针对于
    - 不同的加密和安全技术
      - 之前： `加固` = `安卓加固` = `安卓应用加固` = `Android app hardening`
        - = `加壳`
          - 注：壳是一段保护软件不被非法修改或反编译的程序
        - 破解加壳所以叫： `去壳` = `脱壳` = `反加固`
    - 之前正常编译(`compile` = `compiling`)出apk的过程

- 破解也常被叫：`反编译` = `decompile` = `decompiling`
  - 所以相关工具往往也叫做：`反编译器` = `decompiler`

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2020-01-16 22:32:51

如何反混淆

背景：使用反编译工具只能看到混淆之后的代码结构，看不到混淆之前的原始的代码

而如何反混淆：

- 如果有：`mapping.txt` 文件
  - 有机会反混淆，恢复和还原出原始代码
  - 背景：
    - 安卓项目的开发者，在折腾ProGuard时，才有（生成）的mapping.txt文件
  - 实际上：作为要破解的人，往往没有
- 如果有：源文件和行号文件
  - 有机会反混淆，恢复和还原出原始代码
  - 背景：
    - 许多APK开发者为了在崩溃时保存源文件类名、行号等信息会在APK混淆时添加以下规则保留源文件信息
    - -keepattributes SourceFile,LineNumberTable
  - 实际上：作为要破解的人，往往没有

一些反混淆工具

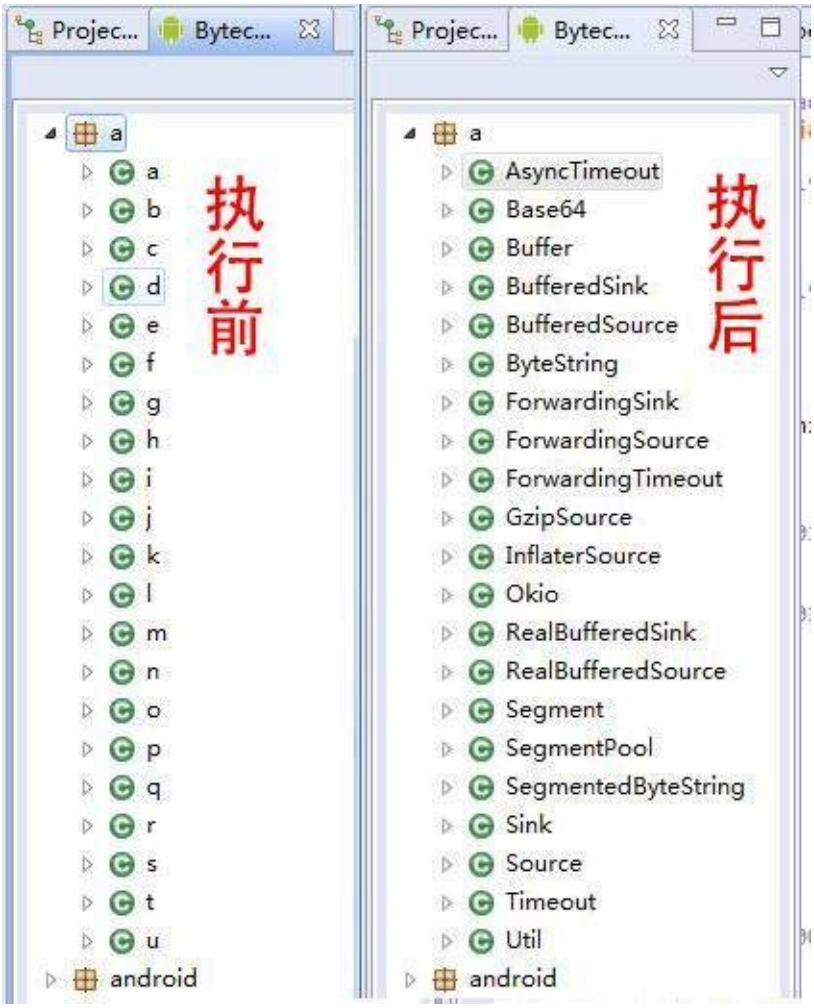
- JEB=JEB Decompiler
  - JEB2 被称为反混淆神器
  - 官网：
    - https://www.pnfsoftware.com/
  - 一些用于反混淆的插件
    - S3cuRiTy-Er1C/JebScripts: Jeb public scripts
    - flankerhqd/jebPlugins: Various Jeb plugins, including obfuscation restore
    - enovella/jebscripts: A set of JEB Python/Java scripts for reverse engineering Android obfuscated code
- Simplify
  - CalebFenton/simplify: Generic Android Deobfuscator

举例

JEB反混淆效果

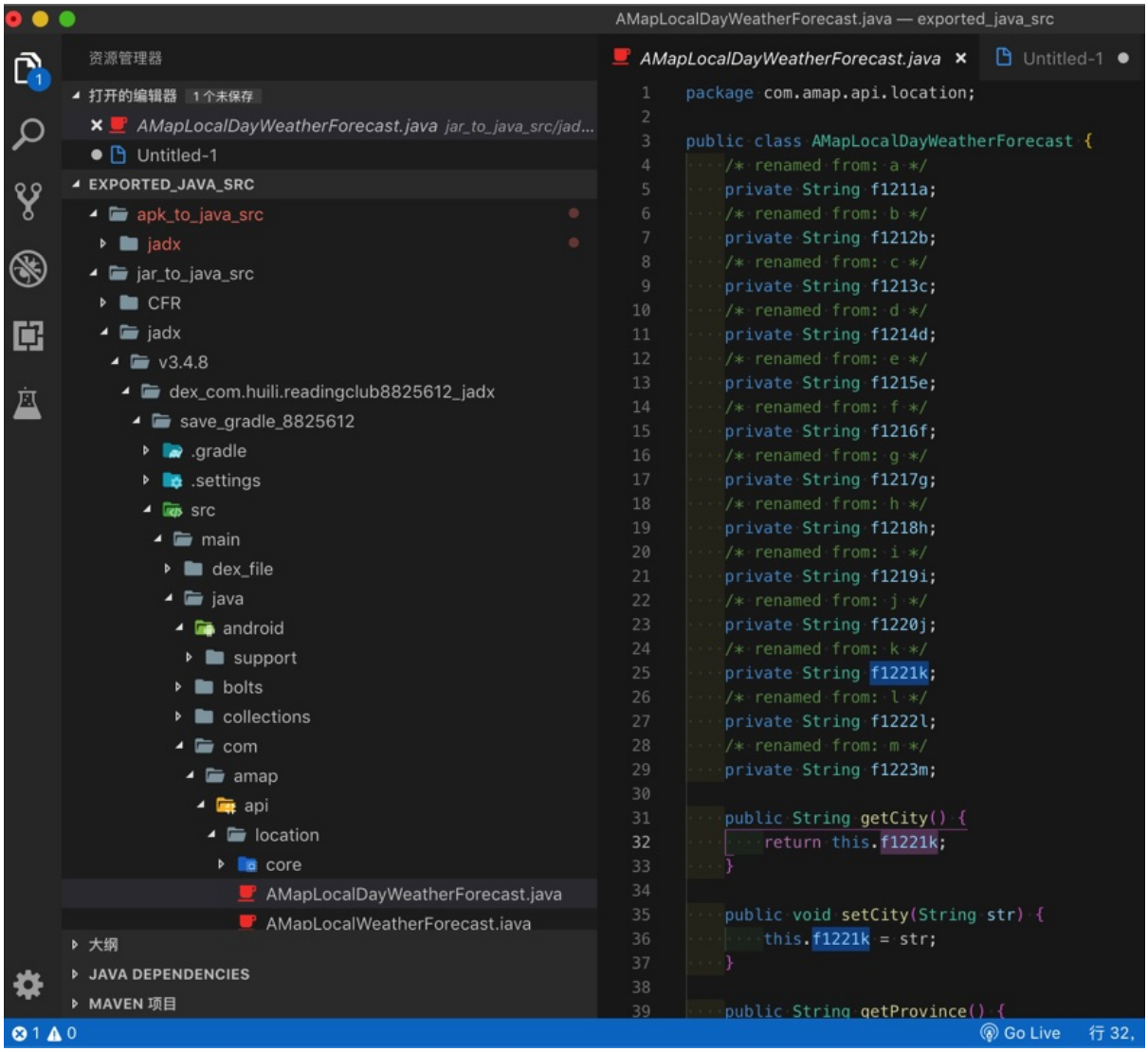
如果有了源文件和行号，则可以反混淆：

```
.class public BaseActivity
.super AppCompatActivity
.source "BaseActivity.java"
```



Jadx反混淆效果

比如：



详见

Jadx 如何反混淆deobfuscation

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 14:59:20

如何去壳脱壳

`去壳` = `去掉加固的壳` = `脱壳`

脱壳机制原理

- smali层：只做了一些简单的混淆
- Native 层：
  - 和如下内容相关
    - 各种so库
      - 比如libdvm.so
    - 对应着内部函数调用：
      - .init
      - .init_array
      - JNI_Onload
    - 分析修改ELF头信息
    - sub_xxx函数
      - 比如：sub_78614CD0
    - R2寄存器
    - 最后分析出：
      - ClassLoader
      - loadDex
      - multidex
  - （作者）QEver
    - 写的一个IDA的脚本=一个dex脱壳脚本
    - 配合kill方法，可以实现脱绝大部分运行于dalvik上的dex壳
      - 可以dump导出正确的dex文件

脱壳注意事项和说明

不是所有的加固的安卓apk都能成功脱壳的。

比如，康美通的安卓apk：

- 老版本 v2.0.7 没有加固，可以直接用 `Jadx` 反编译得到源码的
- 新版本 v4.4.0 是 `360加固保` 加固的，用FDex2也无法导出 `dex` ，无法破解

总结出来就是：

- `没有加固` 的：直接用Jadx即可导出源码
  - 比如老旧的Android的apk，很多都没有加固
    - 不管你怎么 `混淆` 都很容易被人分析得干干净净
- 部分加密不强的：可以脱壳
  - 包括
    - 老一代或免费的 `360加固保`
    - 爱加密（收费）
    - 娜迦加固（收费）
  - 用 `FDex2` 可以脱壳
    - 可以hook导出 `dex` ，再dex转 `jar` ，jar转 `java` 源码

- `腾讯乐固` ，新一代的 `360加固保` ： 没法脱壳
  - 即使用FDex2也无法脱壳无法破解，无法得到dex文件
    - 其中新一代的360加固保：用 `art` 模式+ `dex2oat` 相关机制，或许可以破解
      - 后续研究
        - 【未解决】用ART，oat，dex2oat相关机制去破解新一代360、腾讯等安卓apk的加固

对于使用加固方案的建议和结论

- 免费版的加固可以防止大多数只会反编译的小白
  - 对于普通攻击者还是很有效果的
  - 对于会用工具脱壳的，还是没太大用途的
- 除非用更加高级的，收费版的加固服务
  - 估计就很难破解，很难脱壳了
- 如果真的想要彻底防止别人破解
  - 除了考虑（用更高级的）加固方式
  - 还要花精力在app的业务逻辑层面，权限校验等方面，防止被破解

如何判断是哪家加固方案

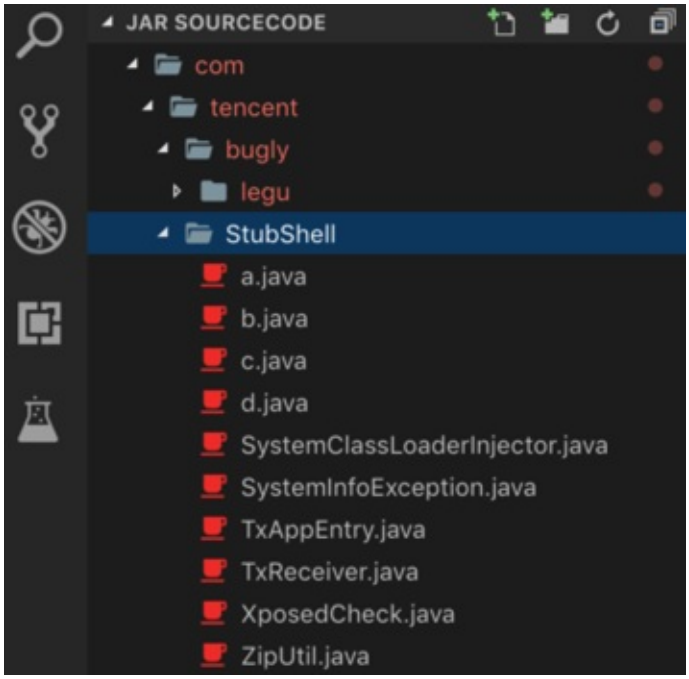
通过反编译工具后，从 `dex` 或 `jar` 包的目录结构，以及相关的文件（比如 `AndroidManifest.xml` ）的内容，往往可以看出是哪家的加密方案：

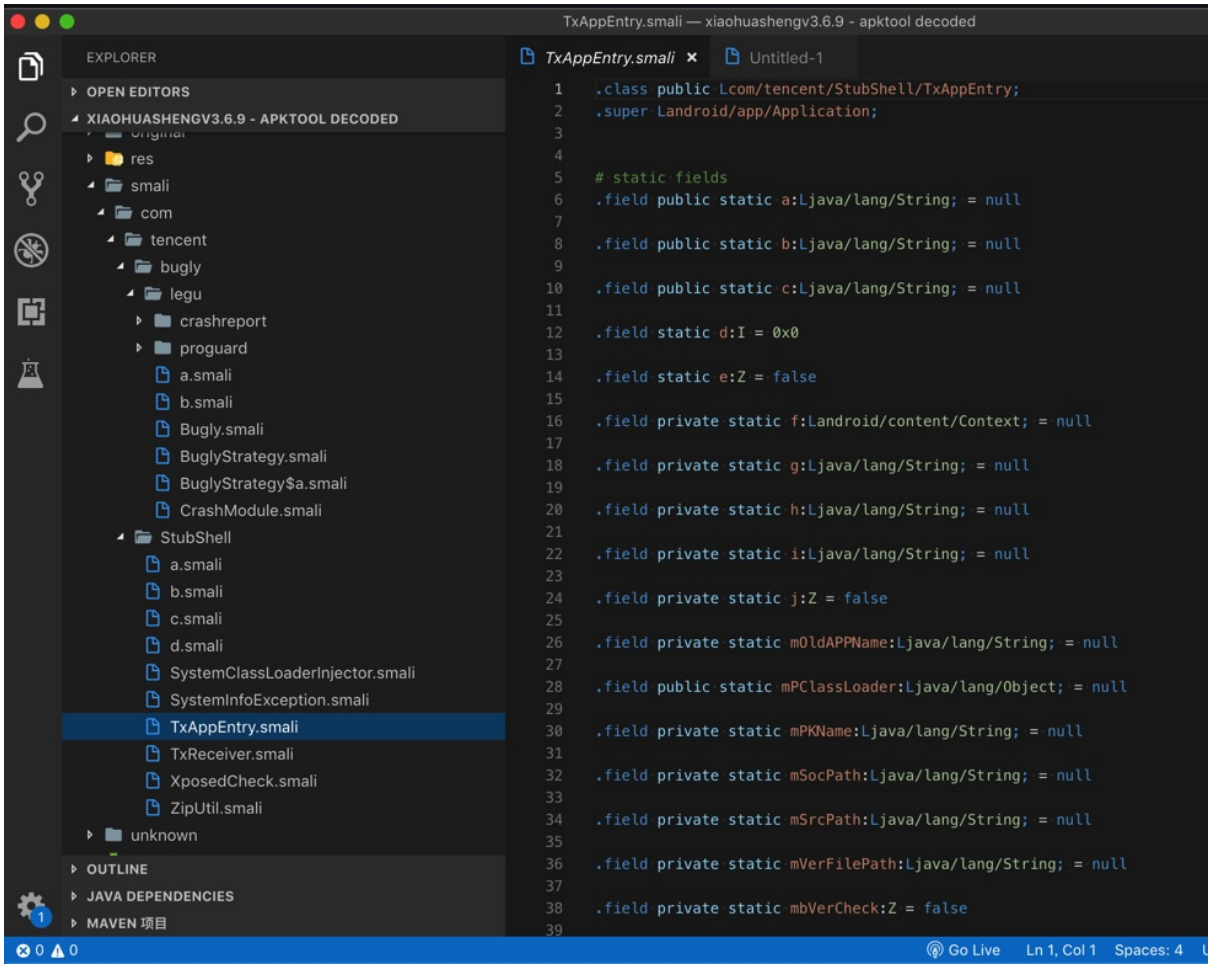
腾讯乐固加密后的目录结构和典型内容

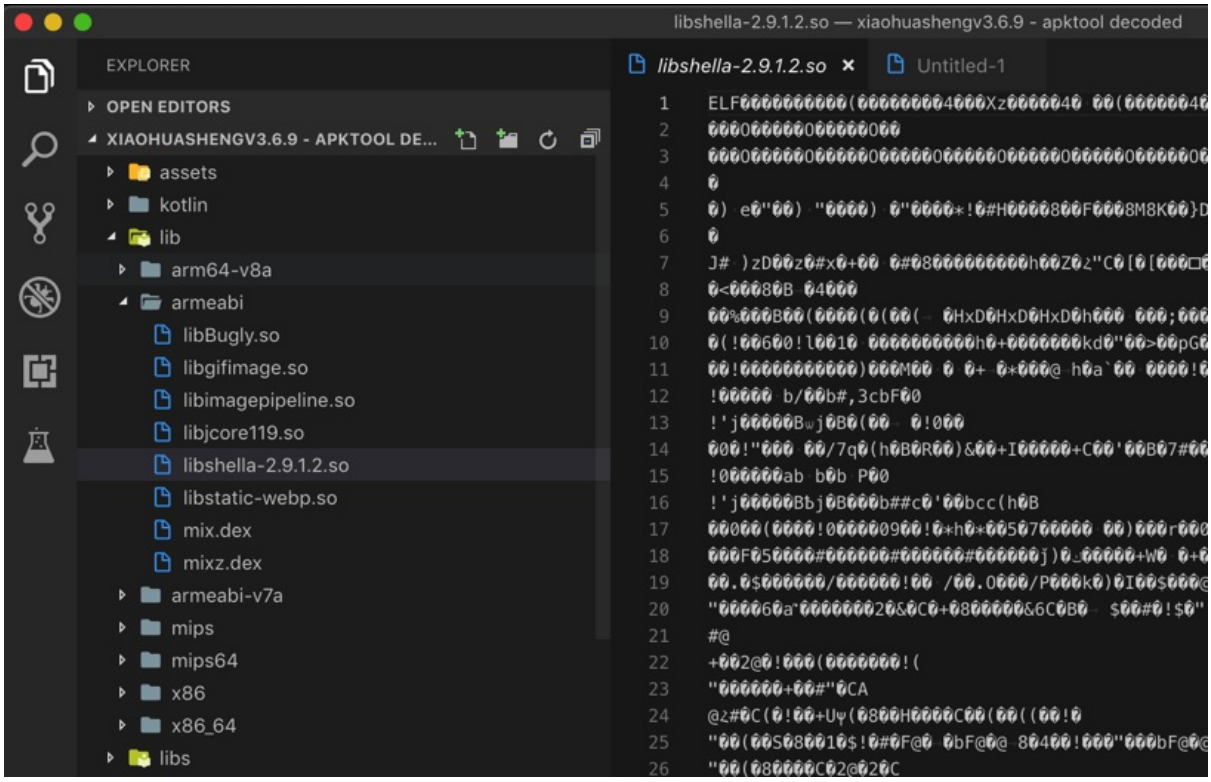
腾讯的 `乐固legu` 加密加壳后的apk，去用 `apktool` 反编译后，得到的 `jar` 包的目录结构是：

- `com.tencent.bugly`
- `com.tencent.bugly.legu`
  - `crashreport`
  - `proguard`
- `com.tencent.StubShell`
  - `TxAppEntry`

截图举例：


JAR SOURCECODE
com
tencent
bugly
legu
StubShell
a.java
b.java
c.java
d.java
SystemClassLoaderInjector.java
SystemInfoException.java
TxAppEntry.java
TxReceiver.java
XposedCheck.java
ZipUtil.java


TxAppEntry.smali — xiaohuashengv3.6.9 - apktool decoded
EXPLORER
OPEN EDITORS
XIAOHUASHENGV3.6.9 - APKTOOL DECODED
res
smali
com
tencent
bugly
legu
crashreport
proguard
a.smali
b.smali
Bugly.smali
BuglyStrategy.smali
BuglyStrategy$a.smali
CrashModule.smali
StubShell
a.smali
b.smali
c.smali
d.smali
SystemClassLoaderInjector.smali
SystemInfoException.smali
TxAppEntry.smali
TxReceiver.smali
XposedCheck.smali
ZipUtil.smali
unknown
OUTLINE
JAVA DEPENDENCIES
MAVEN 项目
TxAppEntry.smali
Untitled-1
.class public Lcom/tencent/StubShell/TxAppEntry;
.super Landroid/app/Application;
static fields
.field public static a:Ljava/lang/String; = null
.field public static b:Ljava/lang/String; = null
.field public static c:Ljava/lang/String; = null
.field static d:I = 0x0
.field static e:Z = false
.field private static f:Landroid/content/Context; = null
.field private static g:Ljava/lang/String; = null
.field private static h:Ljava/lang/String; = null
.field private static i:Ljava/lang/String; = null
.field private static j:Z = false
.field private static mOldAPPName:Ljava/lang/String; = null
.field public static mPClassLoader:Ljava/lang/Object; = null
.field private static mPKName:Ljava/lang/String; = null
.field private static mSocPath:Ljava/lang/String; = null
.field private static mSrcPath:Ljava/lang/String; = null
.field private static mVerFilePath:Ljava/lang/String; = null
.field private static mbVerCheck:Z = false
Go Live
Ln 1, Col 1
Spaces: 4



详细的目录结构和文件是：

```
➜  tencent ll
total 0
drwxr-xr-x  12 crifan  staff   384B  3 14 13:39 StubShell
drwxr-xr-x   3 crifan  staff    96B  3 14 13:39 bugly
➜  tencent tree .
.
├── StubShell
│   ├── SystemClassLoaderInjector.smali
│   ├── SystemInfoException.smali
│   ├── TxAppEntry.smali
│   ├── TxReceiver.smali
│   ├── XposedCheck.smali
│   ├── ZipUtil.smali
│   ├── a.smali
│   ├── b.smali
│   ├── c.smali
│   └── d.smali
└── bugly
    └── legu
        ├── Bugly.smali
        ├── BuglyStrategy$a.smali
        ├── BuglyStrategy.smali
        ├── CrashModule.smali
        ├── a.smali
        ├── b.smali
        ├── crashreport
        │   ├── BuglyHintException.smali
        │   ├── BuglyLog.smali
        │   ├── CrashReport$CrashHandleCallback.smali
        │   ├── CrashReport$UserStrategy.smali
        │   ├── CrashReport.smali
        │   . . .
        │   └── inner
        │       └── InnerAPI.smali
        └── proguard
            ├── a.smali
            ├── . . .
            └── z.smali
14 directories, 123 files
➜  lib tree .
```

```
.
├── arm64-v8a
│   ├── libBugly.so
│   ├── libgifimage.so
│   ├── libimagepipeline.so
│   ├── libjcore119.so
│   ├── libshella-2.9.1.2.so
│   └── libstatic-webp.so
├── armeabi
│   ├── libBugly.so
│   ├── libgifimage.so
│   ├── libimagepipeline.so
│   ├── libjcore119.so
│   ├── libshella-2.9.1.2.so
│   ├── libstatic-webp.so
│   ├── mix.dex
│   └── mixz.dex
├── armeabi-v7a
│   ├── libBugly.so
│   ├── libgifimage.so
│   ├── libimagepipeline.so
│   ├── libjcore119.so
│   ├── libshella-2.9.1.2.so
│   └── libstatic-webp.so
├── . . .
7 directories, 36 files
```

另外反编译出的 `AndroidManifest.xml` 内容：

```
<application allowBackup="true" icon="@drawable/app_logo" label="@string/app_name" name="c
om.tencent.StubShell.TxAppEntry" supportsRtl="true" theme="@style/AppTheme">
<meta-data name="TxAppEntry" value="com.huili.readingclub.MyApplication"/>
```

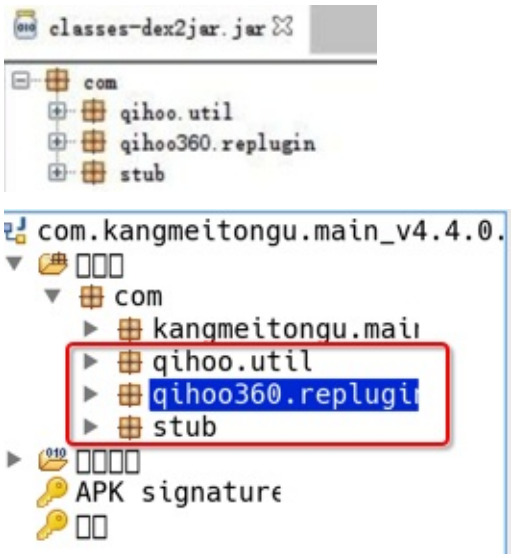
中有：

- `android:name="com.tencent.StubShell.TxAppEntry"`
  - 其中有：
    - `com.tencent.StubShell.TxAppEntry`
- `<meta-data android:name="TxAppEntry"`

也是典型的腾讯乐固的内容。

360加固保的加固的目录结构

360加固后的 `apk` 经过 `dex2jar` 反编译后的目录结构是：



- `com.qihoo.util`
- `com.qihoo360.replugin`
- `com.stub`

这种结构就说明是360加固保加固的。

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 14:59:41

如何从apk破解出java源码

下面整理，从安卓的 `apk` 文件，如何破解，反编译，逆向工程，得到 `java` 源代码。

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 14:58:59

破解apk概述

对于从apk破解得到java源代码，目前常见的几种思路：

- 没有加固的
  - 直接一步搞定
    - 用 `jadx` 从apk导出 `java` 源码
- 加固了的
  - 前提
    - 后续的hook能够导出dex文件
      - 有些新版本的加固，比如腾讯乐固legu，新版的360加固保，无法导出dex文件
        - 也就无法转换出源码
  - 分3步
    1. `apk` -> `dex`
    2. `dex` -> `jar`
    3. `jar` -> `java`

下面详细介绍如何操作。

AndroidManifest.xml

安卓中有个 `AndroidManifest.xml` 文件，是保存了相关项目的类，资源等配置信息。

而对于安卓的 `apk` 文件：

- 直接改名为 `zip` 后再解压：得到的二进制的 `AndroidManifest.xml` 文件
- 用 `apktool` 等工具去反编译：得到的是文本格式的 `AndroidManifest.xml` 文件
  - 就可以看到xml的原始内容了
  - 注：即使 `apk` 加固了，也可以用 `apktool` 反编译

AndroidManifest.xml的作用

得到了xml源码后，可以从其中看到很多有用的信息。

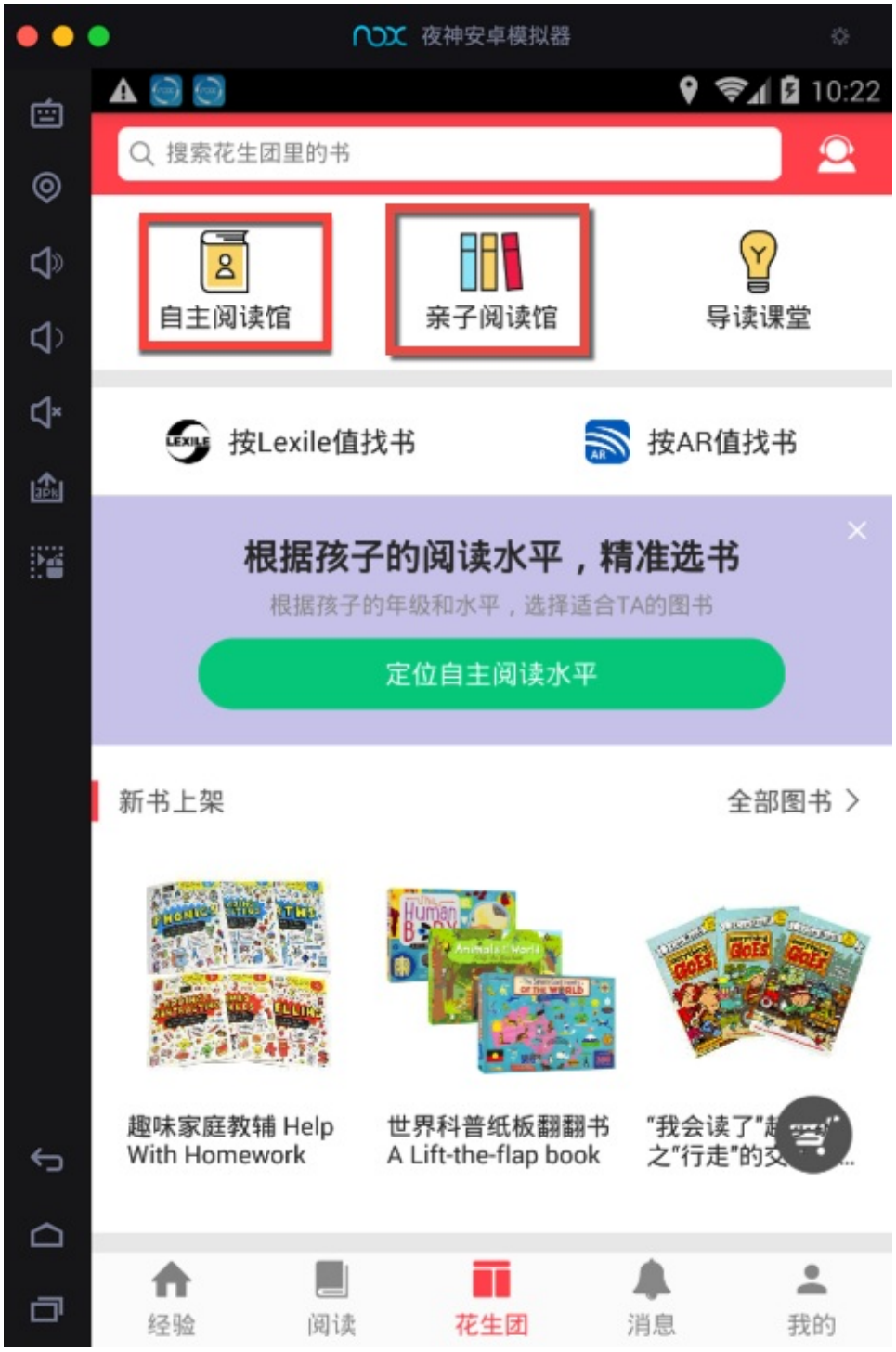
比如， 小花生安卓版 v3.6.9的 `apk` ，apktool反编译出的 `AndroidManifest.xml` 中包含：

```
<activity        name="com.huili.readingclub.activity.classroom.SelfReadingActivity"        screenOrientation="portrait"
/>
...
<activity        name="com.huili.readingclub.activity.classroom.ParentChildReadingActivity"        screenOrientation="po
rtrait"/>
```

其中的类名：

- activity.classroom.SelfReadingActivity
- activity.classroom.ParentChildReadingActivity

=类的文件名，对应了app界面：



中的：

- 自主阅读馆
- 亲子阅读馆

-》从而有利于后续分析内部的业务逻辑，了解到内部有哪些类和功能。

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 14:58:56

一步: apk->java

前提

- apk没有被加固
  - 怎么看出是没有加固?
    - 如果你用jadx打开看到的和导出的代码和目录结构都是 qihoo奇虎360 tencent腾讯乐固legu 字样的
      - 那么说明就是被加固了的
        - 用的是 360加固保 腾讯乐固legu 的加固方案
        - 这样导出的代码也只是加固后的代码
          - 不是app的自己的业务逻辑的java代码

思路

用 `jadx` 等工具，直接从 `apk` 转换出 `java` 源代码

准备

下载 `jadx`：

从skylot/jadx: Dex to Java decompiler的releases页面下载最新版本，比如jadx-0.9.0.zip

解压得到：

- `bin/jadx`：命令行版本
  - `bin/jadx.bat`：Windows版
- `bin/jadx-gui`：带图形界面的版本
  - `bin/jadx-gui.bat`：Windows版

详见：反编译器 jadx

jadx命令行版

直接通过命令行去反编译，效率最高。

语法：

```
jadx -d output_folder your_apk_file.apk
```

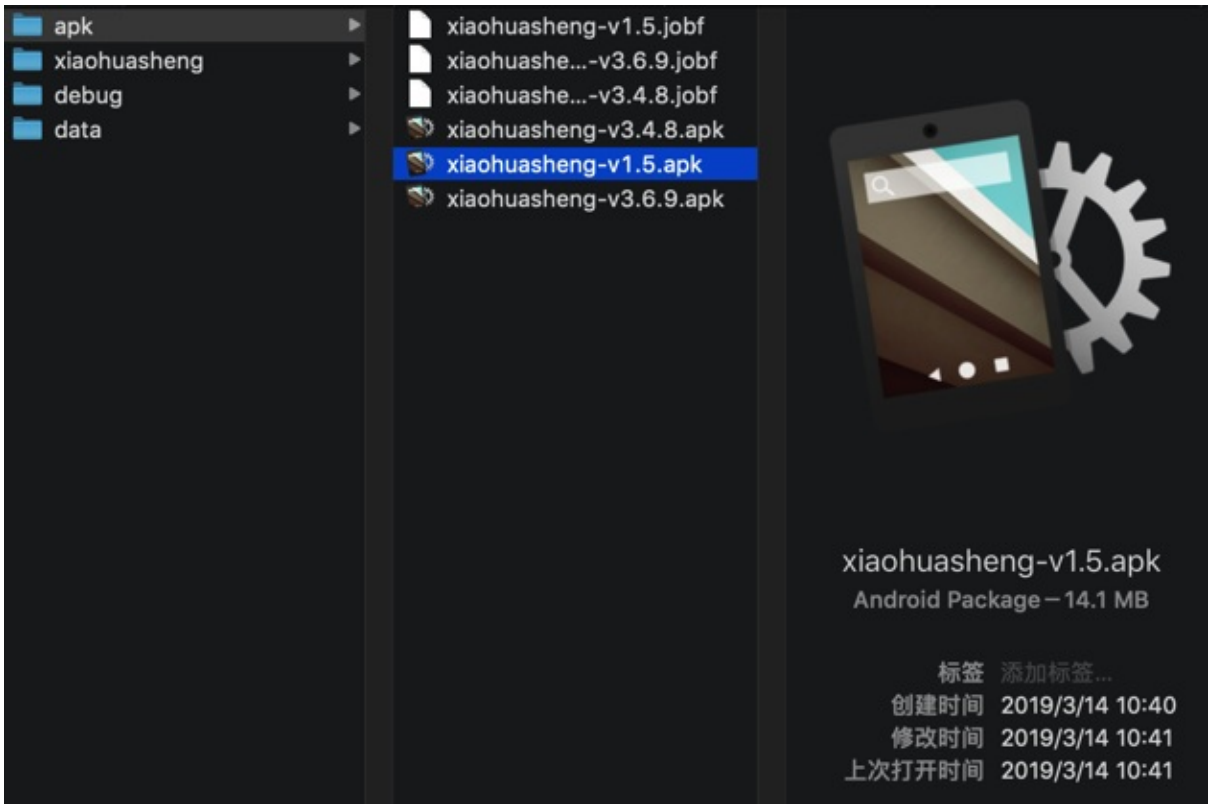
举例：

```
jadx/jadx-0.9.0/bin/jadx -d from_jadx_command xiaohuasheng-v1.5.apk
jadx/jadx-0.9.0/bin/jadx -d exported_java_src mafengwo_ziyouxing.apk
```

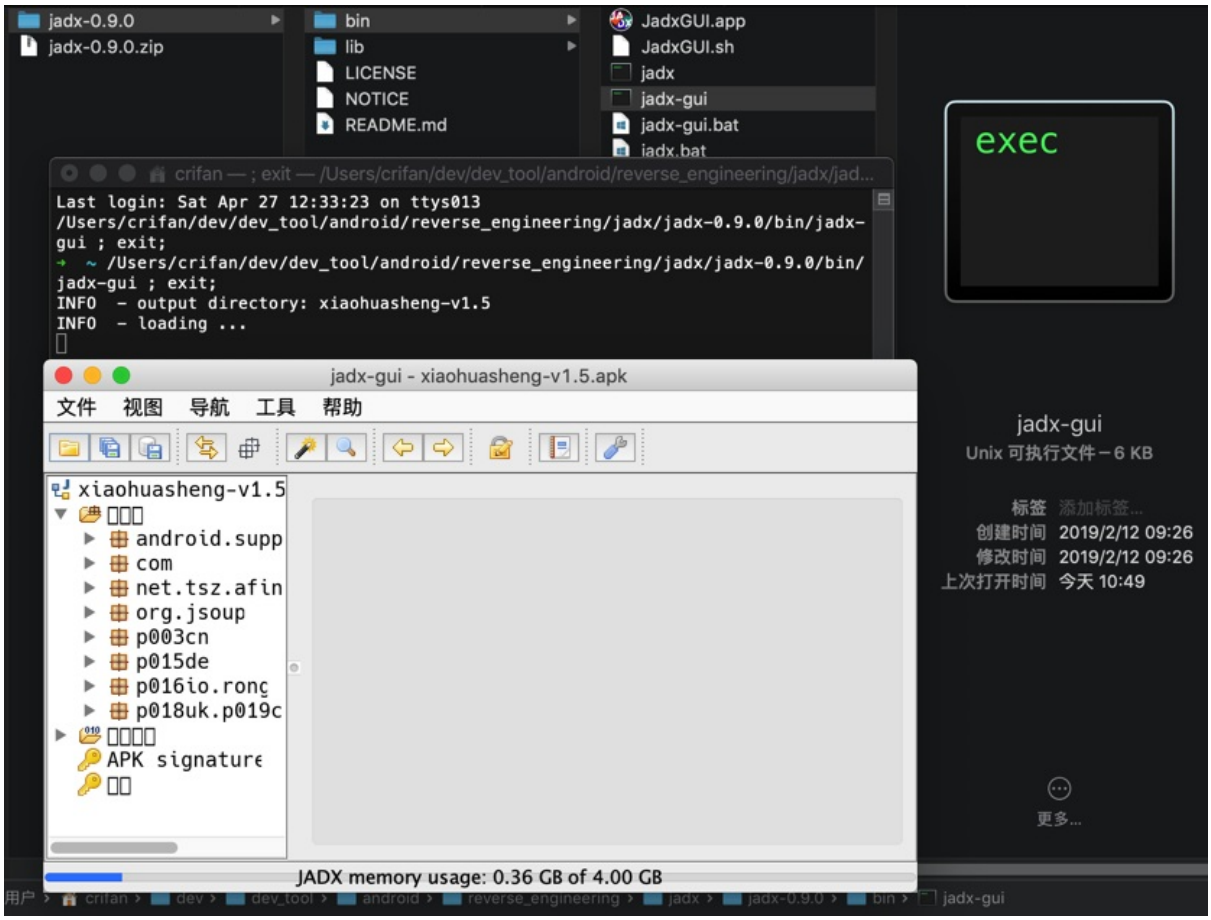
jadx gui图形界面版

下面以 `jadx-gui` (已被我改名为 `JadxGUI`)为例去解释。

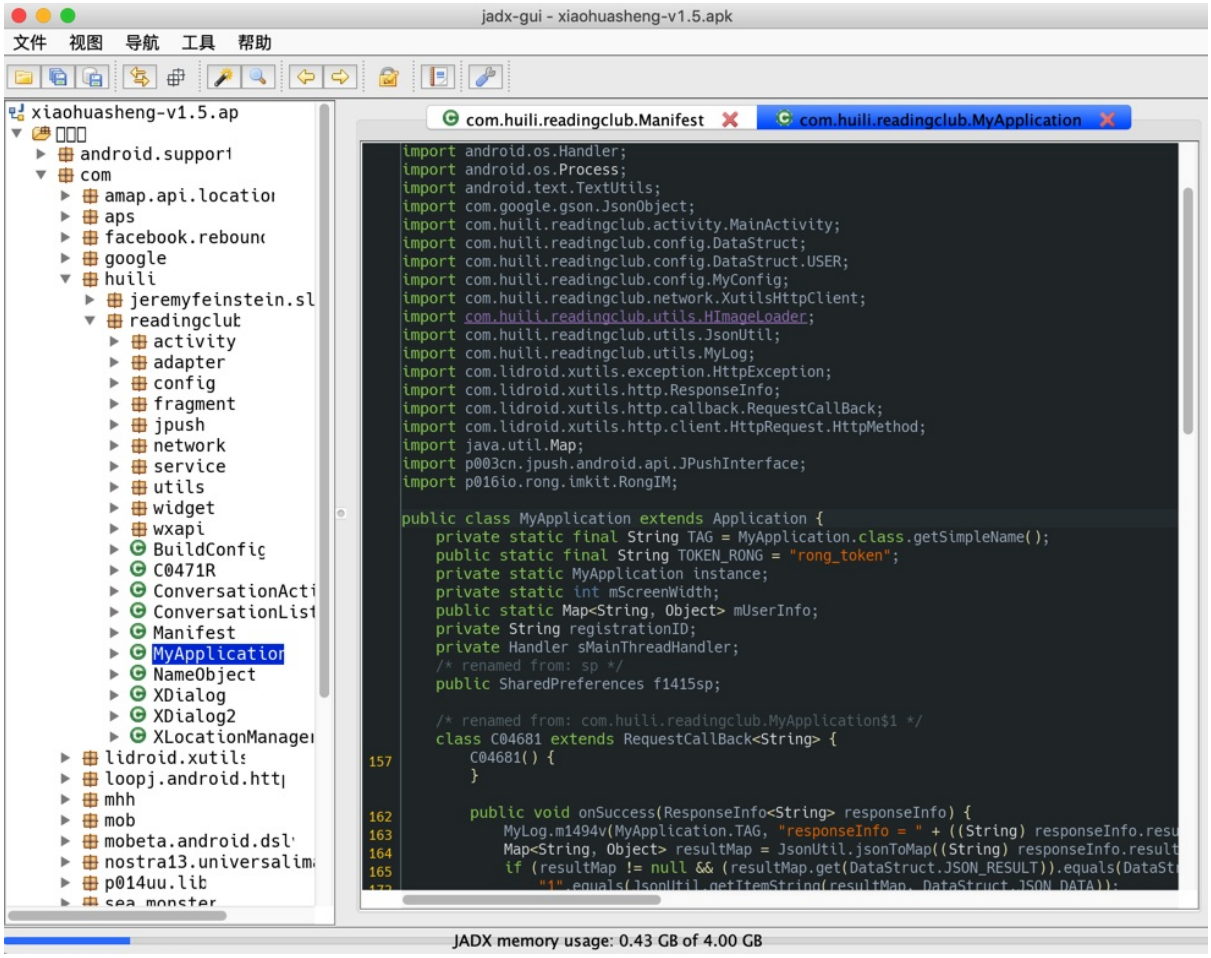
对于此处v1.5这种没有加固的apk：



`jadx` （此处指的是 `jadx-gui` ）打开后：



可以看到源码：

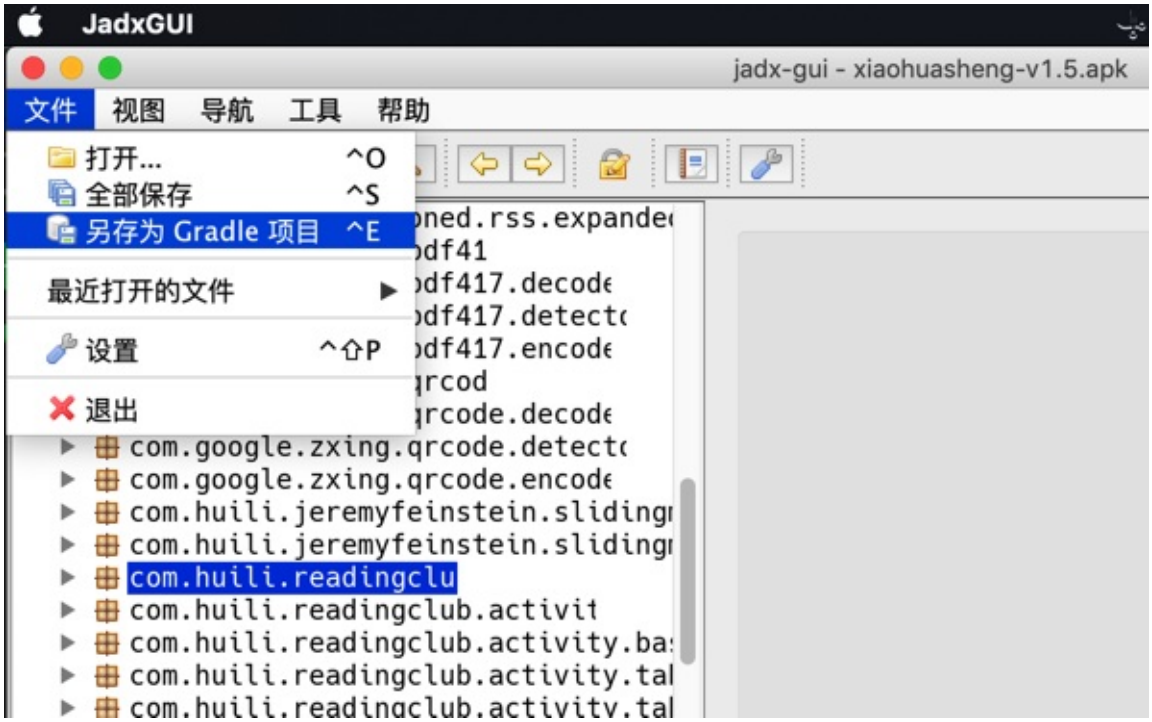


想要导出全部源码，则可以去

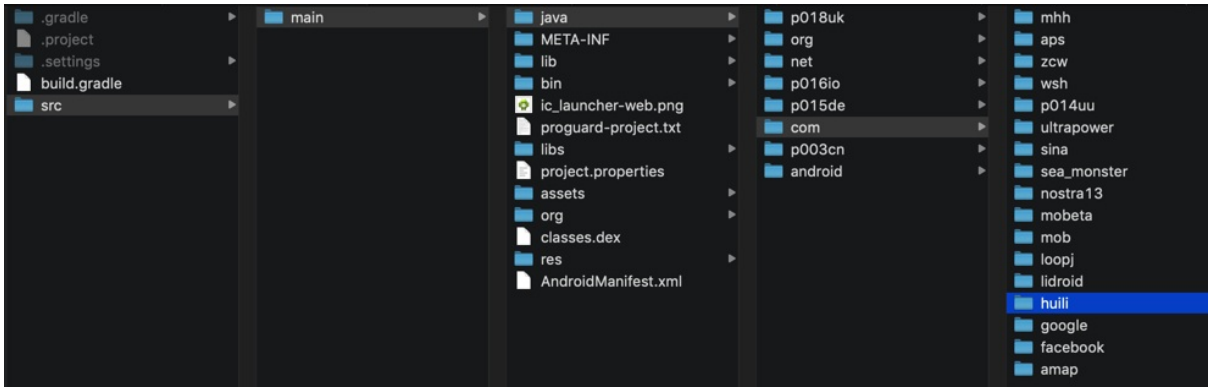
`文件 -> 全部保存`

或

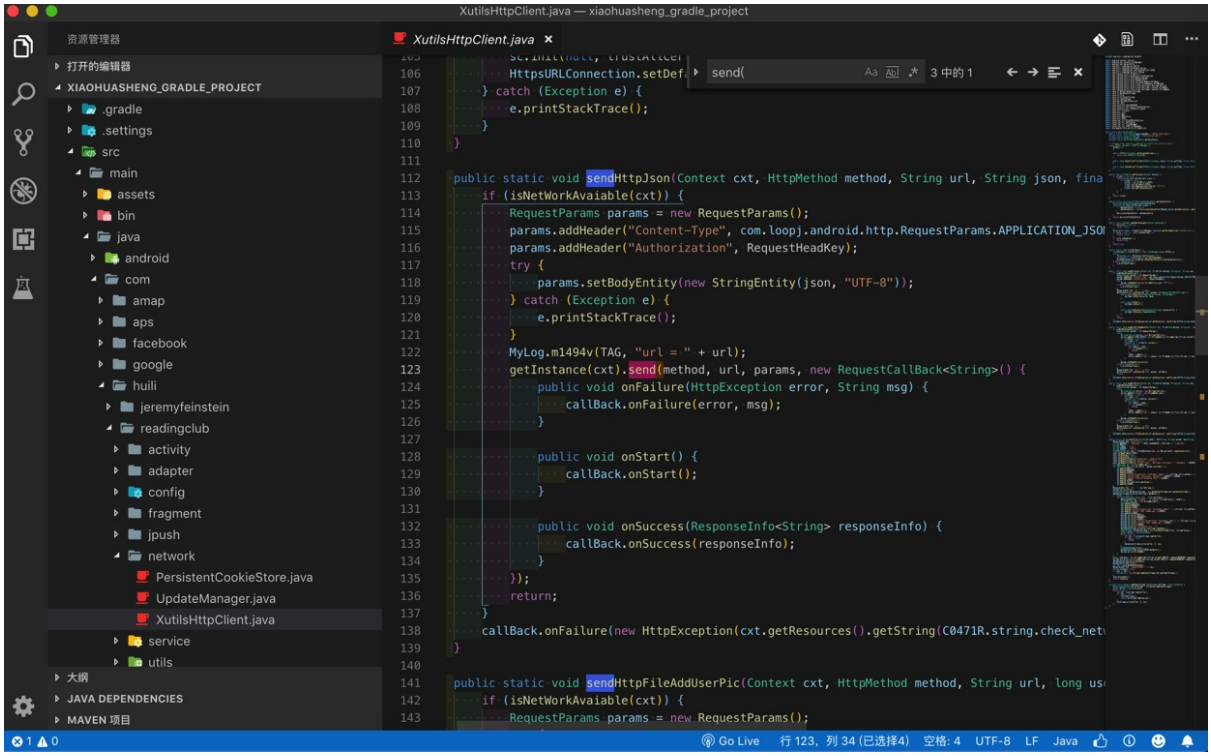
`文件 -> 另存为Gradle项目`



即可导出全部的代码：



用VSCode打开后即可找到（希望研究的app对应的业务逻辑的）代码：



crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2020-01-16 22:37:39

两步或三步: app->dex->jar->java

前提

- apk虽然被加固，但（后续）用 `FDex2` 等工具能够导出我们要的 `dex` 文件
  - 这种加固方式，往往是：旧版本的360加固保等加固方案
  - 如果导不出dex文件
    - 则无法继续导出jar再导出java代码
    - 说明加固方案往往是：
      - 新版本的360加固保
      - 腾讯乐固legu

思路

1. 从 `apk` 到 `dex` ：从运行中的安卓app导出dex文件
   - 事先安装安卓的apk到 `已root` 的安卓 `真机` /安卓 `模拟器`
   - 在安卓中已经安装了 `Xposed` 等 `Hook框架` 中
   - 再去 `Xposed` 中安装 `FDex2` / `DumpDex` 等插件
     - 用于后续导出 `dex` 文件
   - 运行安卓app
     - 自动导出我们要的 `dex` 文件
   - 从安卓（真机/模拟器）中拷贝出刚才导出的（往往是多个） `dex` 文件
2. 从 `dex` 到 `java` 源码
   - 一步： 比如用 `jadx` 支持直接从 `dex` 转换出 `java` 源代码
   - 两步：
     i. `dex` 转 `jar`
        - 用 `dex2jar` 等工具从（包含了我们要的、和app业务逻辑相关的那个） `dex` 文件中导出 `jar` 文件
     ii. `jar` 转 `java` 源码
        - 用反编译器（ `CFR` / `Procyon` / `JD-GUI` 等）把 `jar` 转换出 `java` 源代码

详细步骤

下面就来详细解释具体操作过程。

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2020-01-16 22:18:27

1. app导出dex

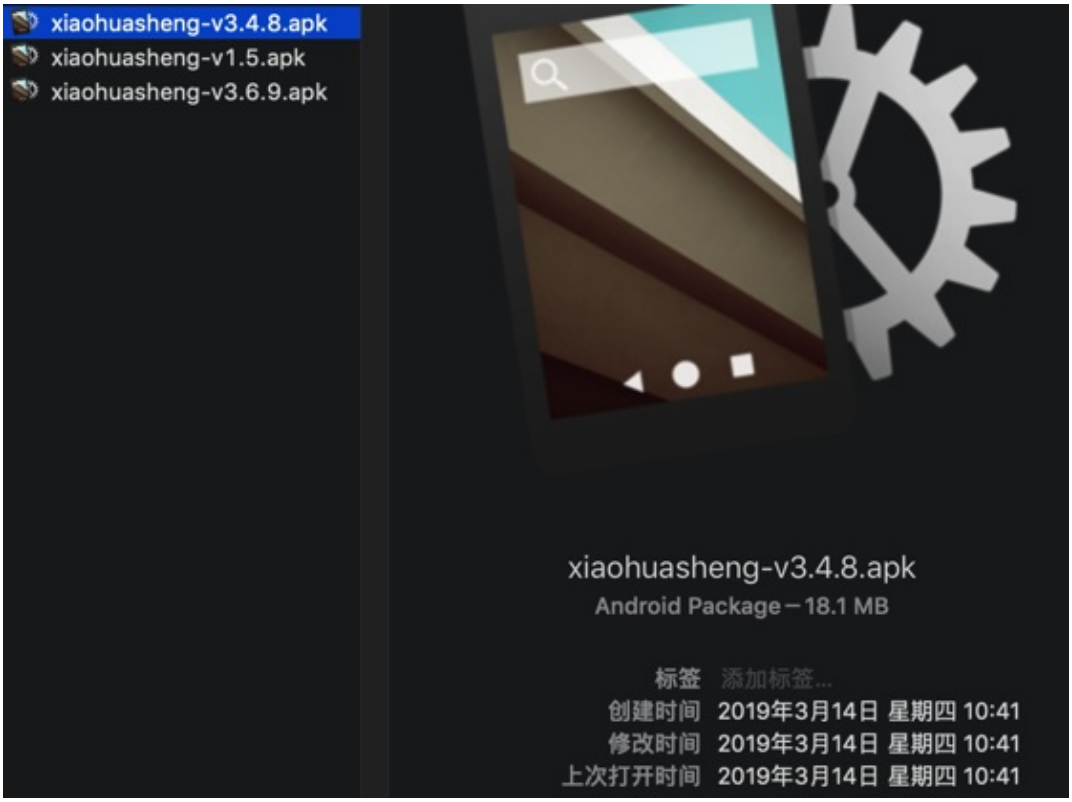
准备

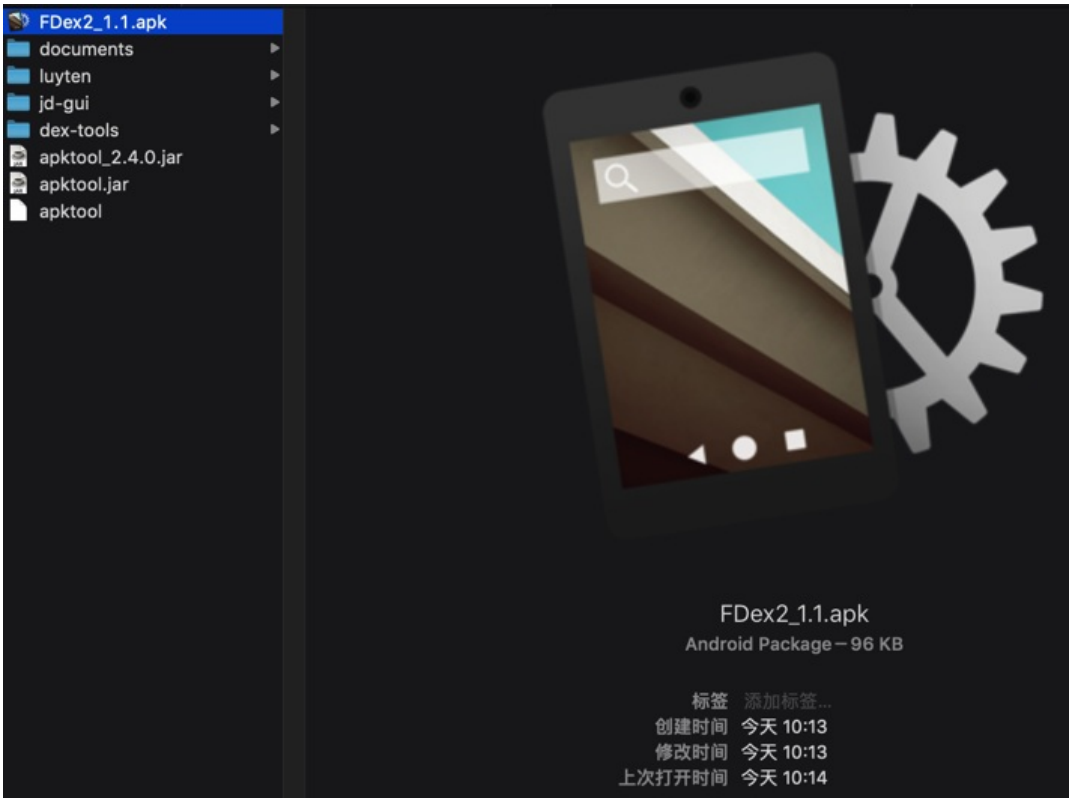
- 已 `root` 了的安卓 `4.4+` 的 `真机` 或 `模拟器`
  - 因为后续的 `FDex2` 要求 `Android 4.4+`
  - 此处用的是： `夜神模拟器` = `Nox App Player`
    -

    - 去模拟安卓 `4.4.2`

        - `夜神模拟器` 已自带 `root`
- `Hook框架`
    - 用于后续安装插件去导出 `dex` 文件
    - 此处选用： `XPosed框架`
        - 在安卓中安装 `XPosed` 框架
            - 之前已安装 `XPosed框架`
            -

- 要破解的安卓apk
    - 比如 `v3.4.8` 的小花生的apk：

- 

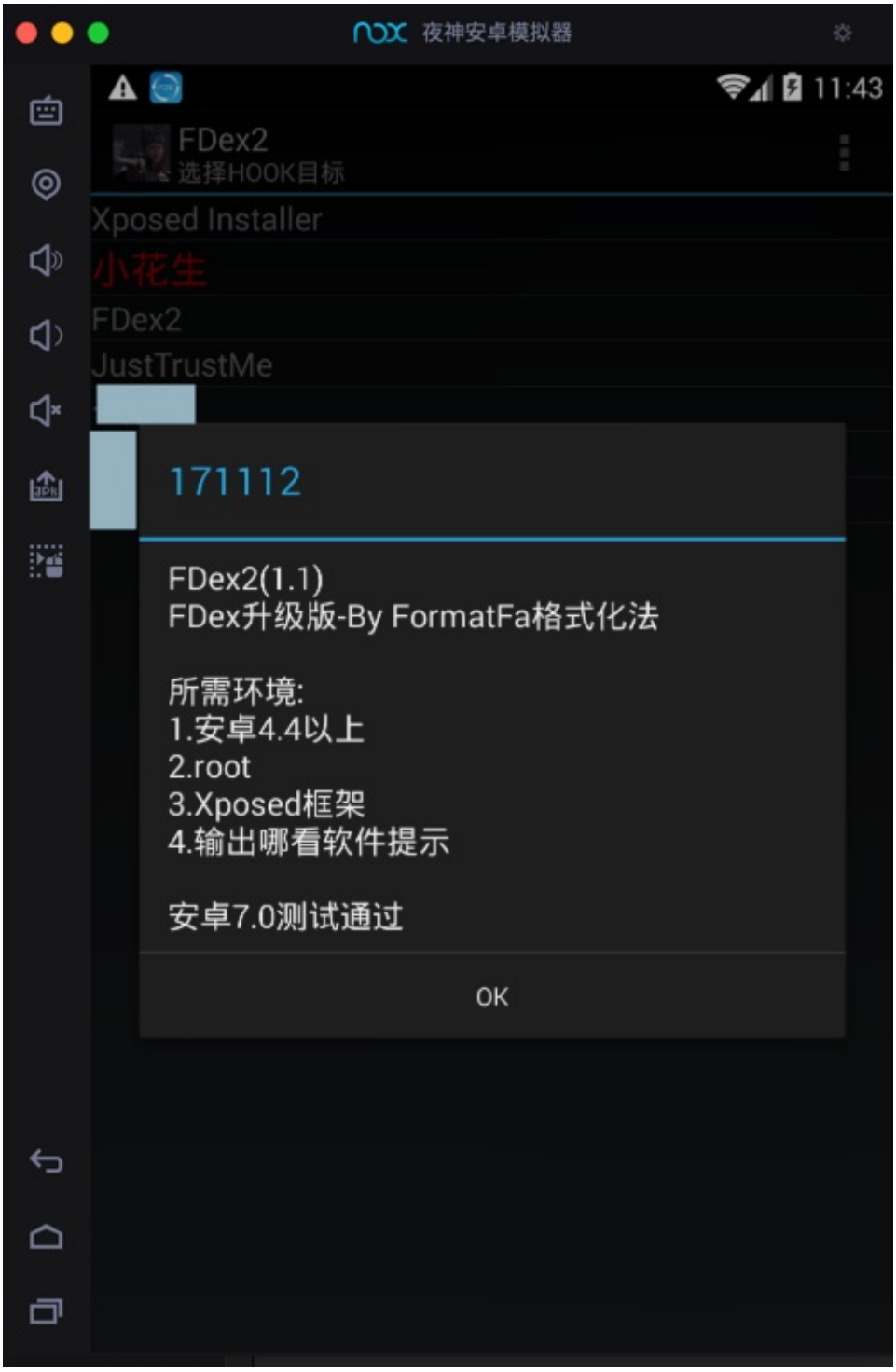
- 下载好 `FDex2` 的 `apk`
  - 从这里下载到 `FDex2_1.1.apk`
    - 

下面以 `夜神模拟器` + `XPosed框架` 为例解释如何操作。

详细步骤

- 安装要破解的apk
  - 把要破解的 `apk` 安装到 `夜神安卓模拟器` 中：
  -
- 安装和激活 `FDex2`
  - 安装 `FDex2` 安装到夜神模拟器中
  - 去XPosed里勾选 `激活` FDex2：

-
- 注意：
  - 其会提示： `XPosed模块列表已更新，重启后更改将生效`
  - 所以为了使 `FDex2生效` ， 记得去重启 `XPosed` 。
- 此处的版本是 `1.1` ：

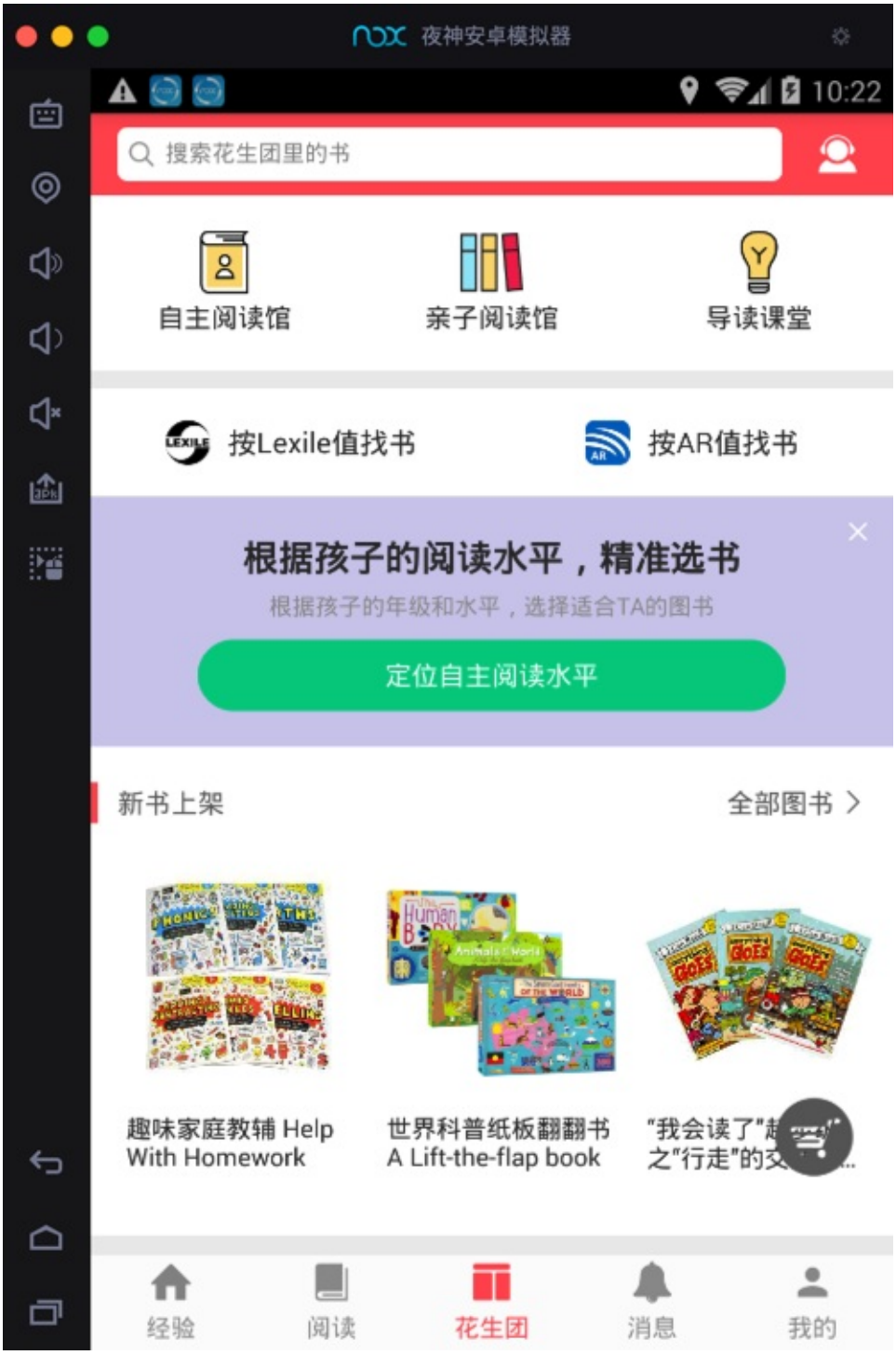


- `FDex2` 中设置要处理的app
  - 然后再去打开FDex2，点击此处要破解的app：小花生

-
- 会提示设置成功：
- > 设置保存成功，请重新打开目标软件，hook包名：com.huili.readingclub
- > dex输出目录：/data/data/com.huili.readingclub

-
- 运行要破解的app
  - 正常去打开和运行要破解的app

-
- 注意：
  - 其实只要打开了：



- 稍等几秒，即可
  - 此处内部 `FDex2` 已经去导出app的所有文件到对应的目录了：`/data/data/com.huili.readingclub`
- 但是为了更保险，此处再去：随意点击和切换页面，也点击到了要破解的页面，感觉会更好

- 拷贝出已导出的 `dex` 等文件
  - 所以接着去对应路径
    - 之前 `FDex2` 设置app时已提示的输出路径
      - `/data/data/com.huili.readingclub`
  - 找dex文件（和其他相关项目文件）：

-
- 并拷贝出来即可

最终拷贝出我们要的dex文件：

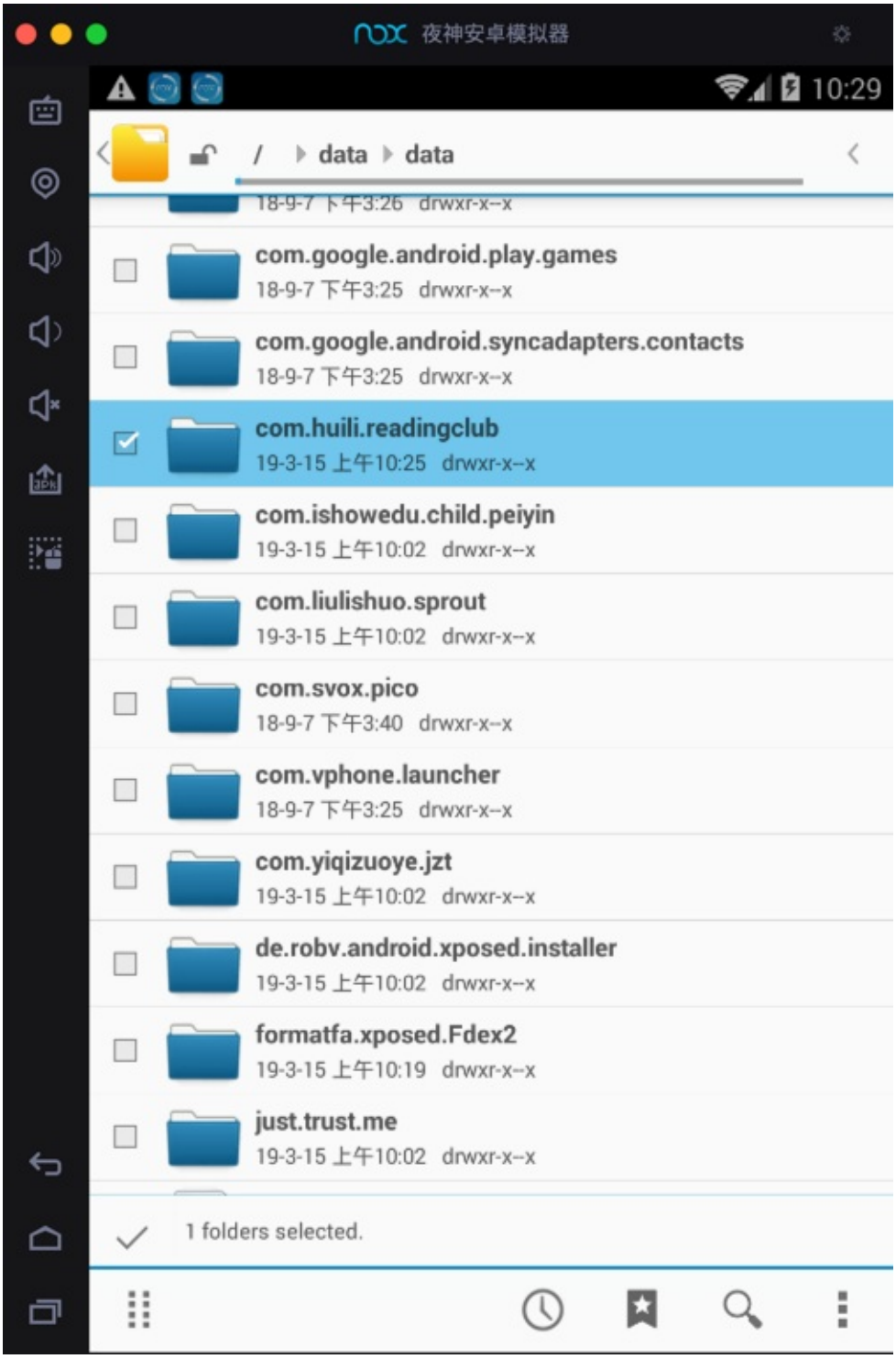
```
➜  v3.4.8 ll
total 81656
-rw-------  1 crifan  staff   1.1M  3 19 14:05 com.huili.readingclub1166288.dex
-rw-------  1 crifan  staff    12M  3 19 14:04 com.huili.readingclub13088280.dex
-rw-------  1 crifan  staff   1.4M  3 19 14:04 com.huili.readingclub1461452.dex
-rw-------  1 crifan  staff   187K  3 19 14:04 com.huili.readingclub191572.dex
-rw-------  1 crifan  staff   2.7M  3 19 14:04 com.huili.readingclub2847840.dex
-rw-------  1 crifan  staff   3.8M  3 19 14:04 com.huili.readingclub3986968.dex
-rw-------  1 crifan  staff   8.3M  3 19 14:04 com.huili.readingclub8725900.dex
-rw-------  1 crifan  staff   8.4M  3 19 14:04 com.huili.readingclub8825612.dex
```

注意和说明

如何从夜神模拟器中导出文件到Mac中

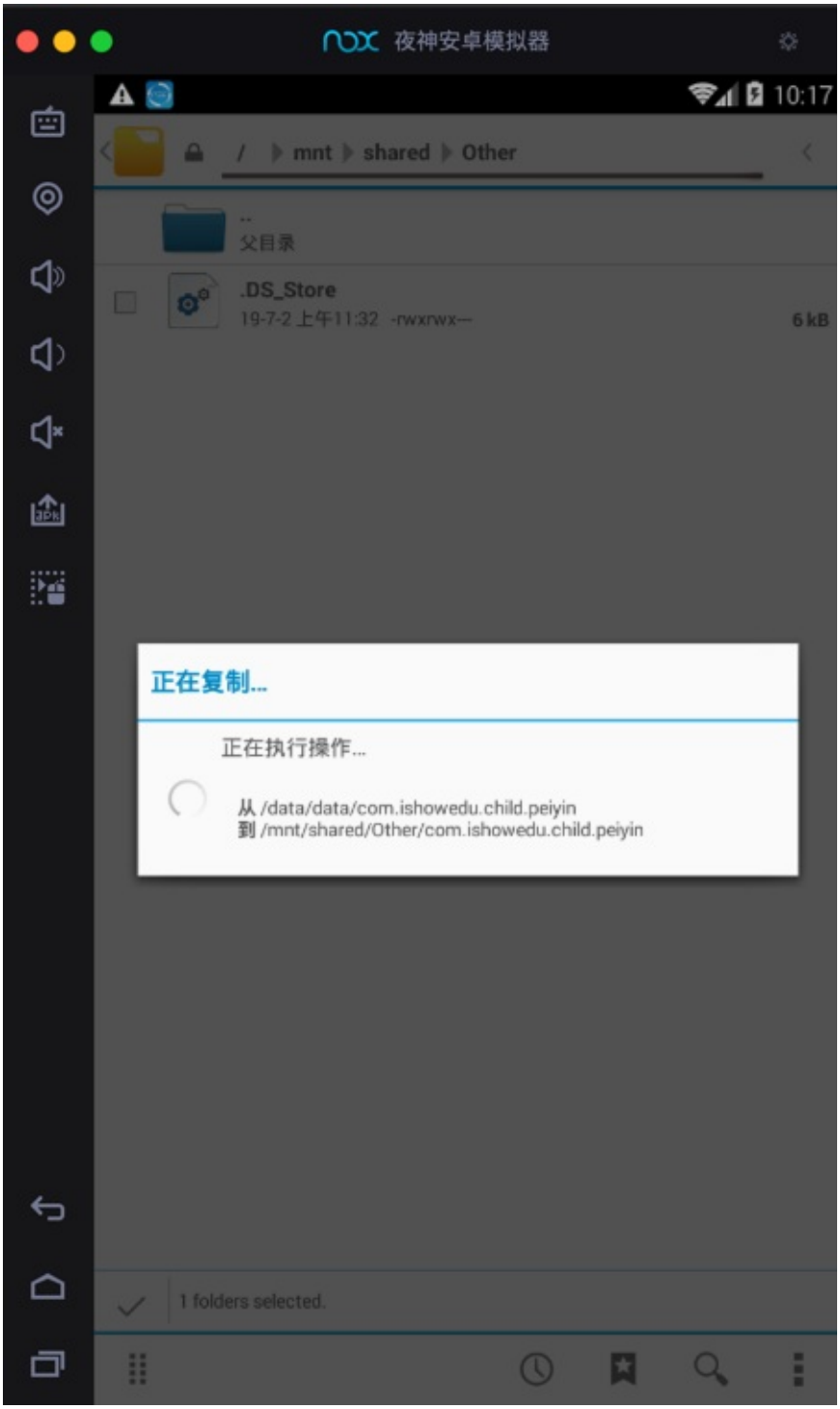
此处使用夜神模拟器自带的 `文件管理器` ：

- 详细步骤：
  - 先把导出的文件拷贝到共享目录
    1. 勾选 `/data/data/` 下面的 `com.huili.readingclub`

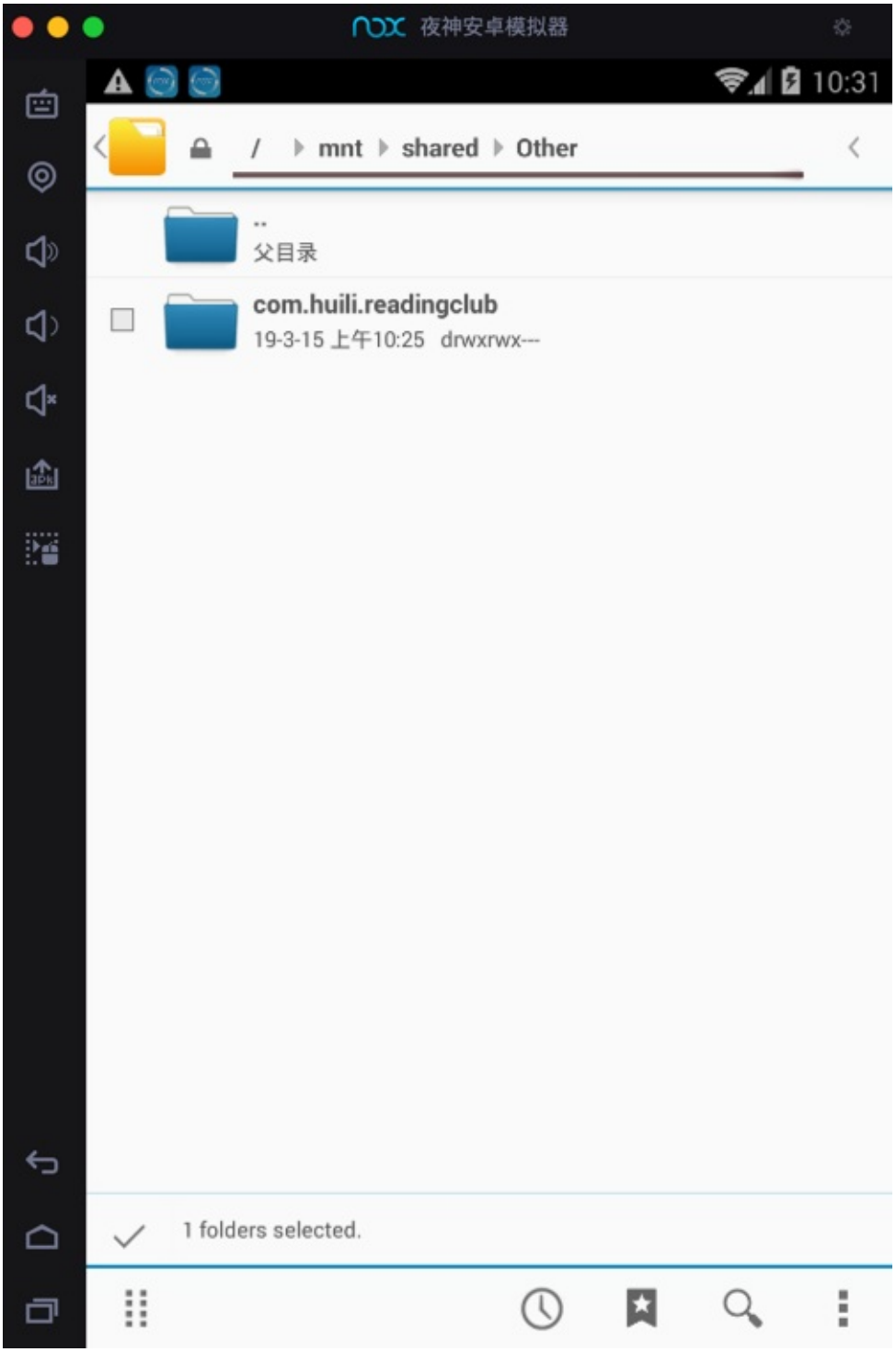


2. 切换到夜神用于和电脑共享的文件夹 `/mnt/shared/Other` 中去 `操作` -> `粘贴选择项`

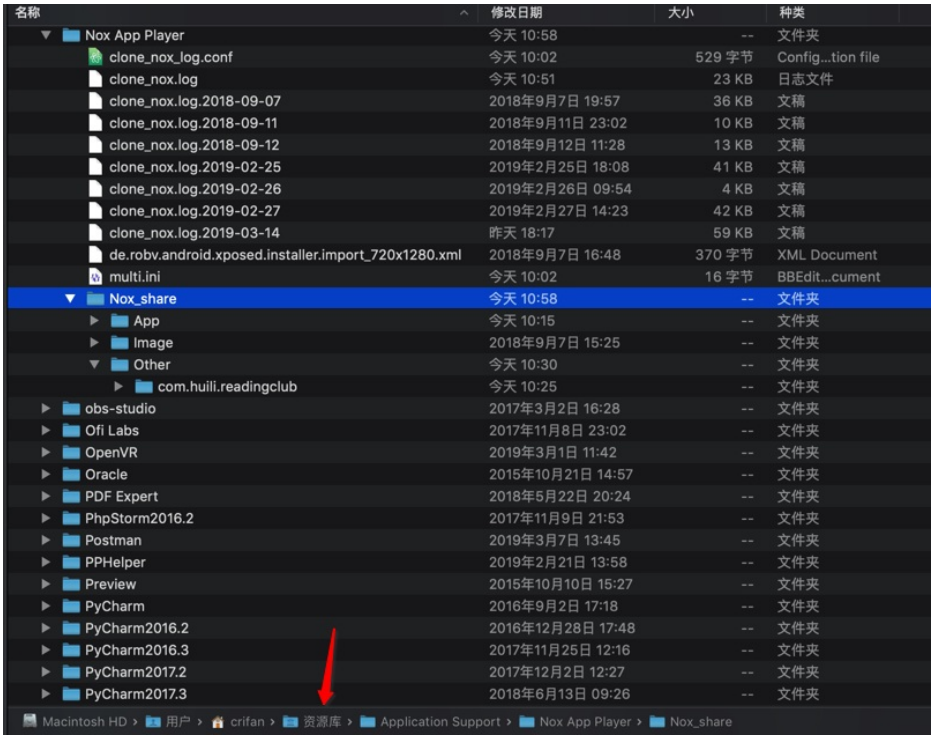
-
夜神安卓模拟器
10:15
/ mnt shared Other
..
父目录
.DS_Store
19-7-2 上午11:32 -rwxrwx---
6 kB
操作
属性
刷新
新建文件夹
新建文件
全部选择
取消全选
粘贴选择项
移动选择项
删除选择项
压缩选择项
创建链接
添加到书签
添加快捷方式
取消
1 folders selected.
- 然后会开始复制和粘贴：



3. 复制粘贴后，无法立即看到已拷贝的文件。回到上一级目录，再重新进来即可看到：

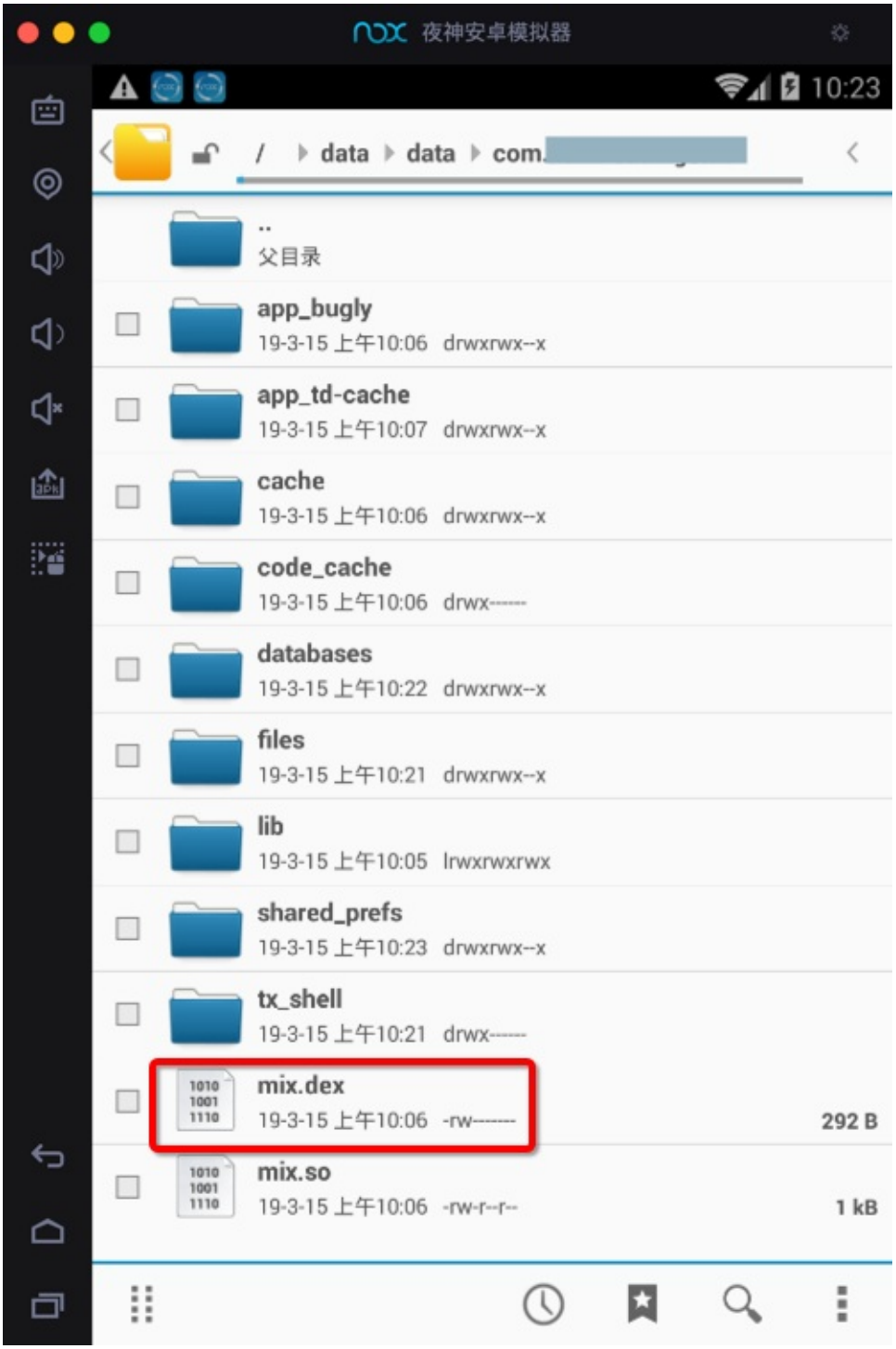
- 

- 再去到PC（此处Mac）中找到共享根目录，并拷贝文件
  - 夜神模拟器和PC的共享根目录
    - `Windows`：
      - 据说是 `C:\Users\{USERNAME}/Nox_share/Other`
    - `Mac`：
      - `/Users/{用户名}/Library/Application Support/Nox App Player/Nox_share`
      - 举例：
        - `/Users/crifan/Library/Application\ Support/Nox\ App\ Player/Nox_share`
          - 说明：其中空格需要 \ 转义

- 

没看到我们要导出的dex文件

1. 有时候没有看到导出文件，则可以重新多试试几次，即可。
2. 有些app（的有些版本），导出的文件中，没有我们希望的（多个dex文件中的某个）包含了app业务逻辑的dex文件
   - 说明该apk采用了更加高级的加固，无法导出我们要的dex文件
   - 举例：
     - 比如小花生的v3.6.9导出的只有一个无效的292B的dex文件：mix.dex



crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2020-01-16 22:43:11

2.1 dex转java

`jadx` 可以直接从 `dex` 导出 `java` 源码

下载 `jadx`

- 从这里下载最新版的 `jadx`
  - 比如：
    - jadx-0.9.0.zip
  - 解压后得到：
    - `jadx` ： 命令行工具
    - `jadx-gui` ： 带图形界面的
      - 双击即可运行

此处想要用 `jadx` 去从 `jar` 中导出代码，有两种方式：

- 用 `jadx` 的命令行直接导出代码
- 用 `jadx-gui` 查看代码，也可以再导出代码

下面详细介绍

`jadx` 直接导出代码

切换到要导出代码的目录，已有dex文件要导出，则可以直接运行：

语法：

```
bin/jadx dex_file.dex -d output_folder
```

举例：

```
jadx-0.9.0/bin/jadx dex_file.dex -d .

../../../../../../reverse_engineering/jadx/jadx-1.0.0/bin/jadx ../../apk_to_dex/com.ishowedu.child.peiyin/com.ishowedu.chi
ld.peiyin8392664.dex -d com.ishowedu.child.peiyin8392664_java

../../../../../../reverse_engineering/jadx/jadx-1.0.0/bin/jadx ../dex_to_jar/com.ishowedu.child.peiyin9201516-dex2jar.jar -d .
```

即可转换出源代码到当前目录下，输出有：

- resources
- sources
  - 有你要的源码

转换速度还是不错的。

举例

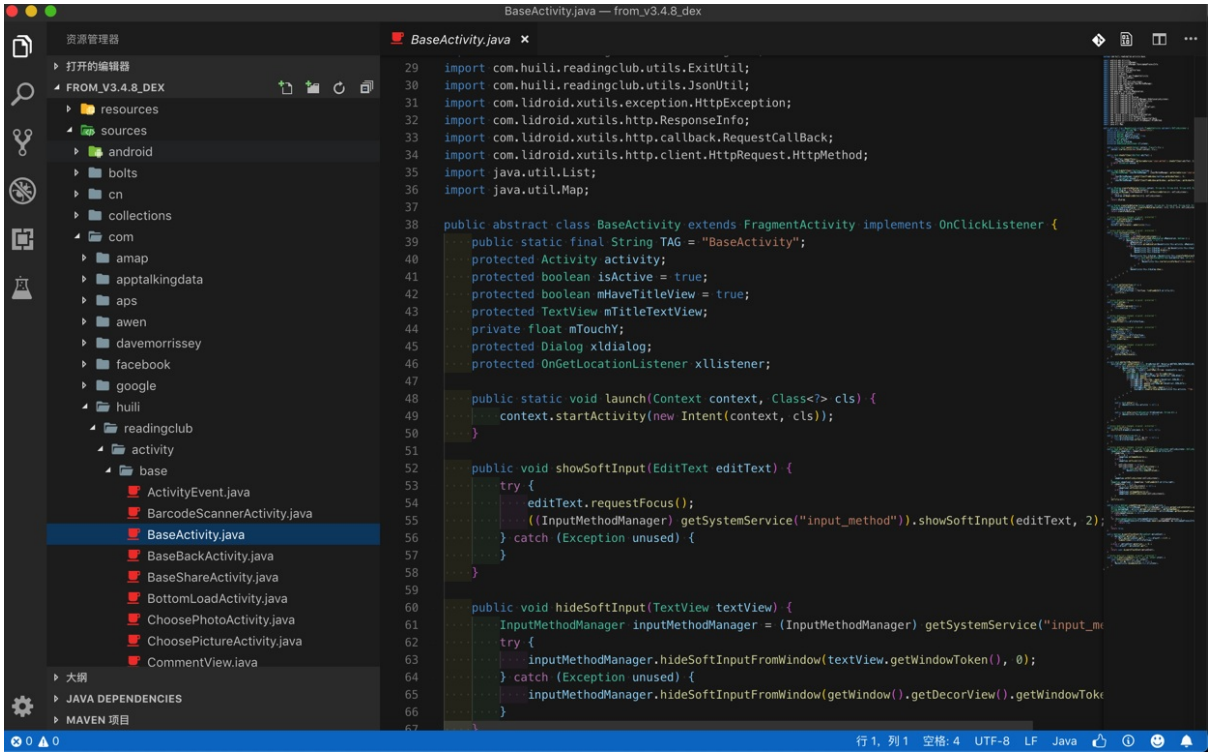
```
from_v3.4.8_dex /Users/crifan/dev/dev_tool/android/reverse_engineering/jadx/jadx-0.9.0/bin/jadx ../../../../../xiaohuashen
g/app_hook_dump_dex/FDex2/v3.4.8/com.huili.readingclub8825612.dex -d .
...
申间很多错误
```

```
...
WARN  - Found 75 references to unknown classes
ERROR - 6 errors occurred in following nodes:
ERROR -   Method: android.support.v4.provider.FontsContractCompat.getFontFromProvider(android.content.Context, android.sup
port.v4.provider.FontRequest, java.lang.String, android.os.CancellationSignal):android.support.v4.provider.FontsContractCo
mpat$FontInfo[]
ERROR -   Method: cn.addapp.pickers.util.LogUtils.getTraceElement():java.lang.String
ERROR -   Method: cn.jiguang.a.a.b.c.a(android.os.Message):void
ERROR -   Method: cn.jiguang.d.b.f.a(int):boolean
ERROR -   Method: cn.jiguang.d.d.m.a(android.content.Context, boolean):java.util.List java.io.File
ERROR -   Method: cn.jiguang.g.e.a(java.lang.String, java.util.Map):cn.jiguang.g.e
WARN  - 2299 warnings in 454 nodes
ERROR - finished with errors
```

转换后:

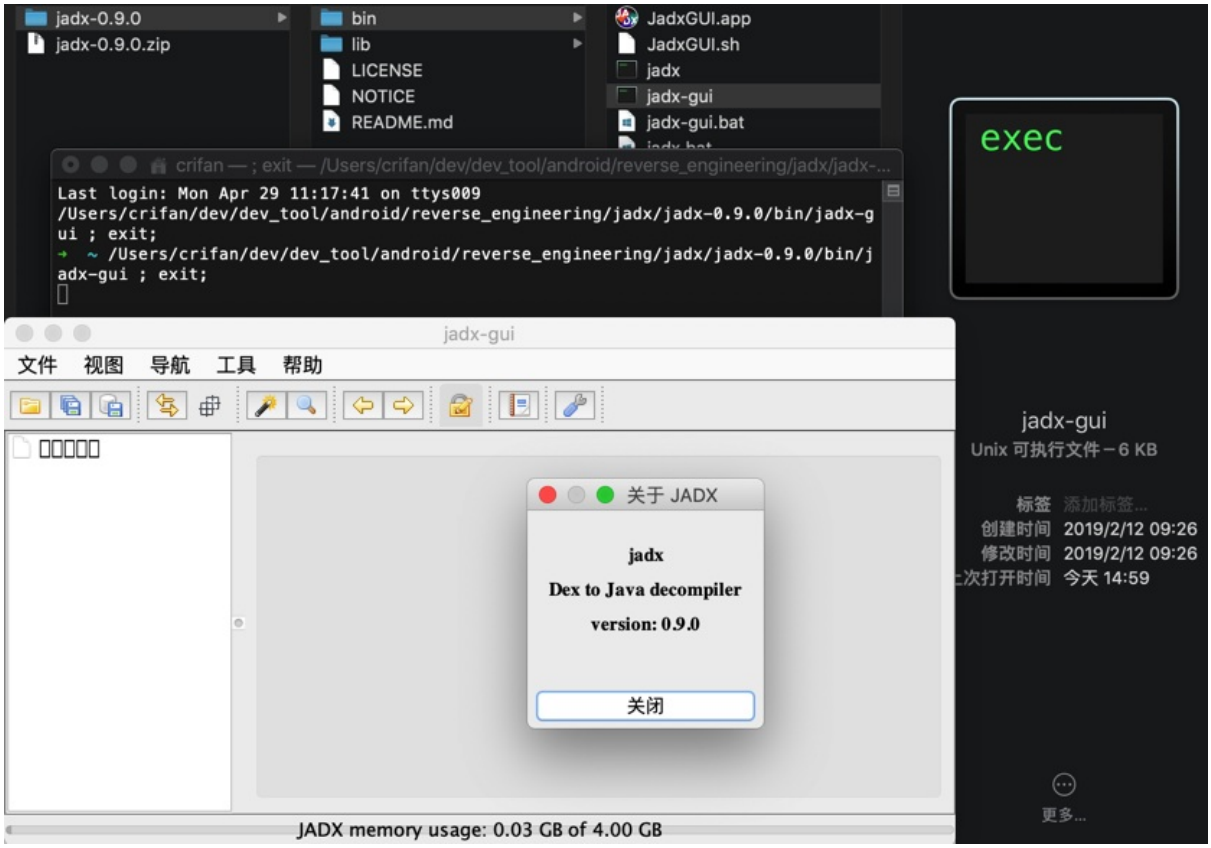
```
➜  from_v3.4.8_dex ll
total 0
drwxr-xr-x   3 crifan  staff    96B  4 29 15:29 resources
drwxr-xr-x  13 crifan  staff   416B  4 29 15:30 sources
```

转换后的代码用VSCode去打开的效果:

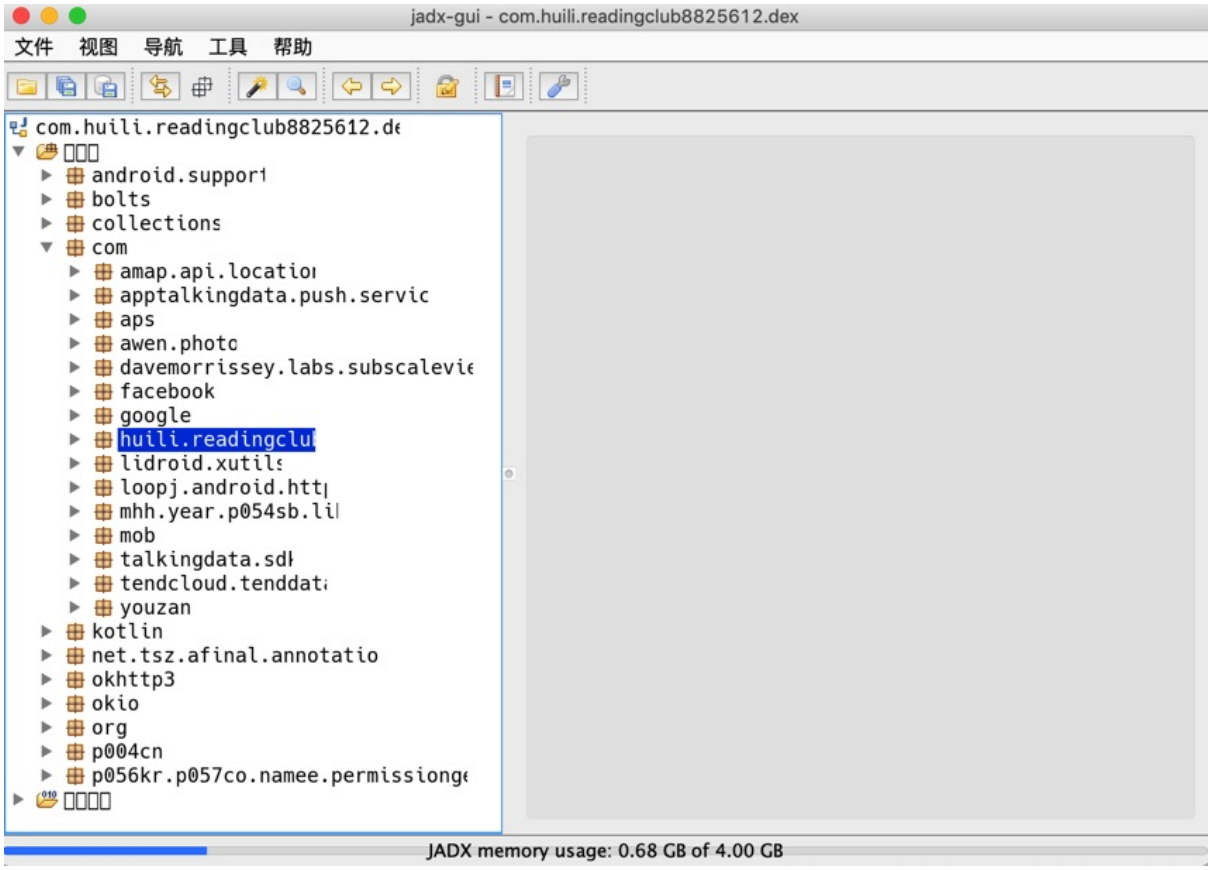


`jadx-gui` 查看和导出代码

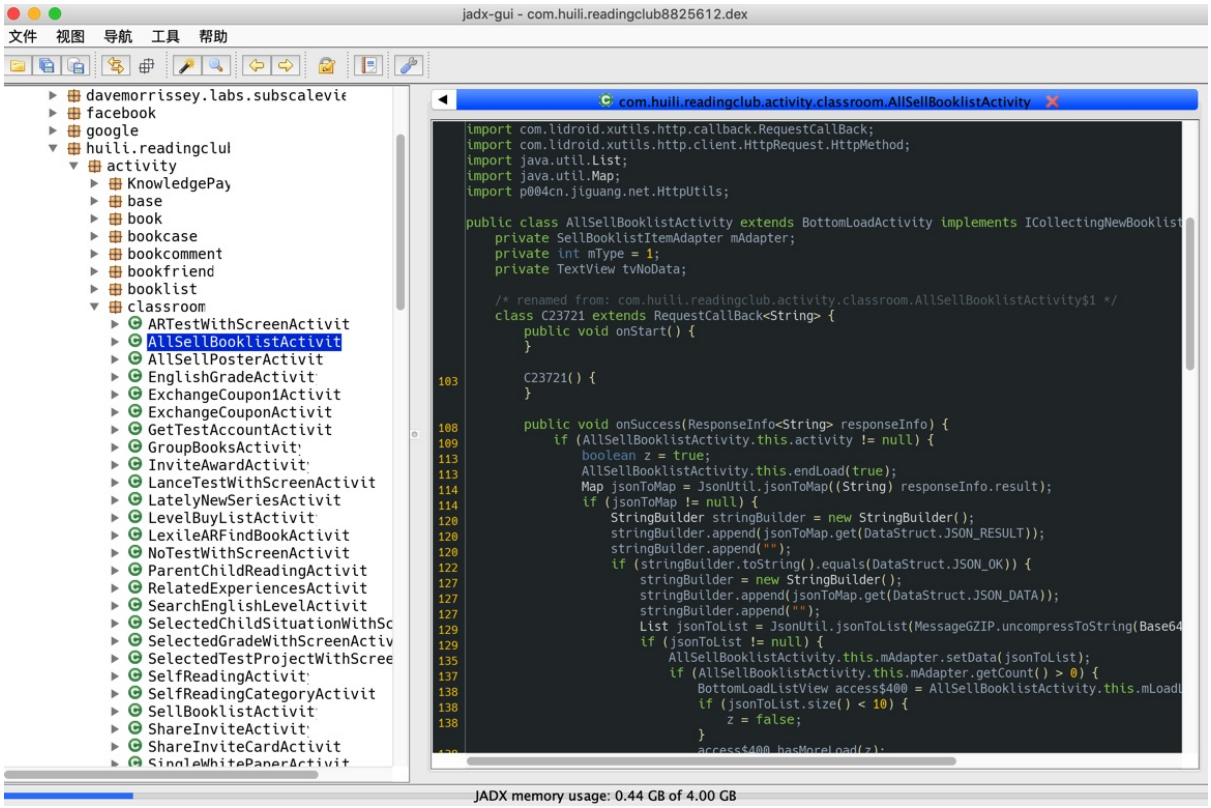
双击 `jadx-gui` 即可运行:



然后去打开对应的jar文件：`com.huili.readingclub8825612-dex2jar.jar`，即可看到包含了app业务逻辑的代码结构和包名：

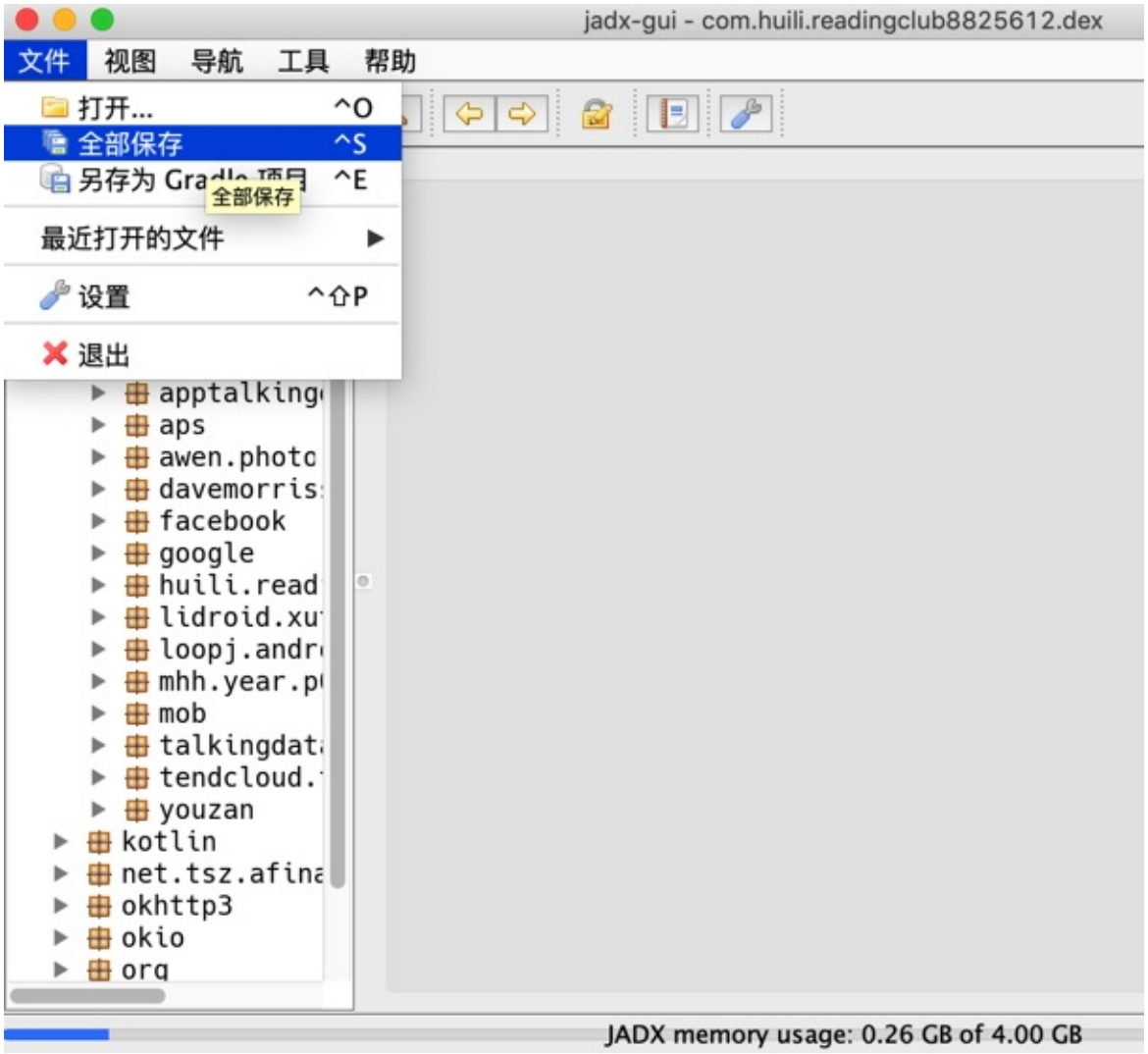


然后展开后可以看到详细的代码：

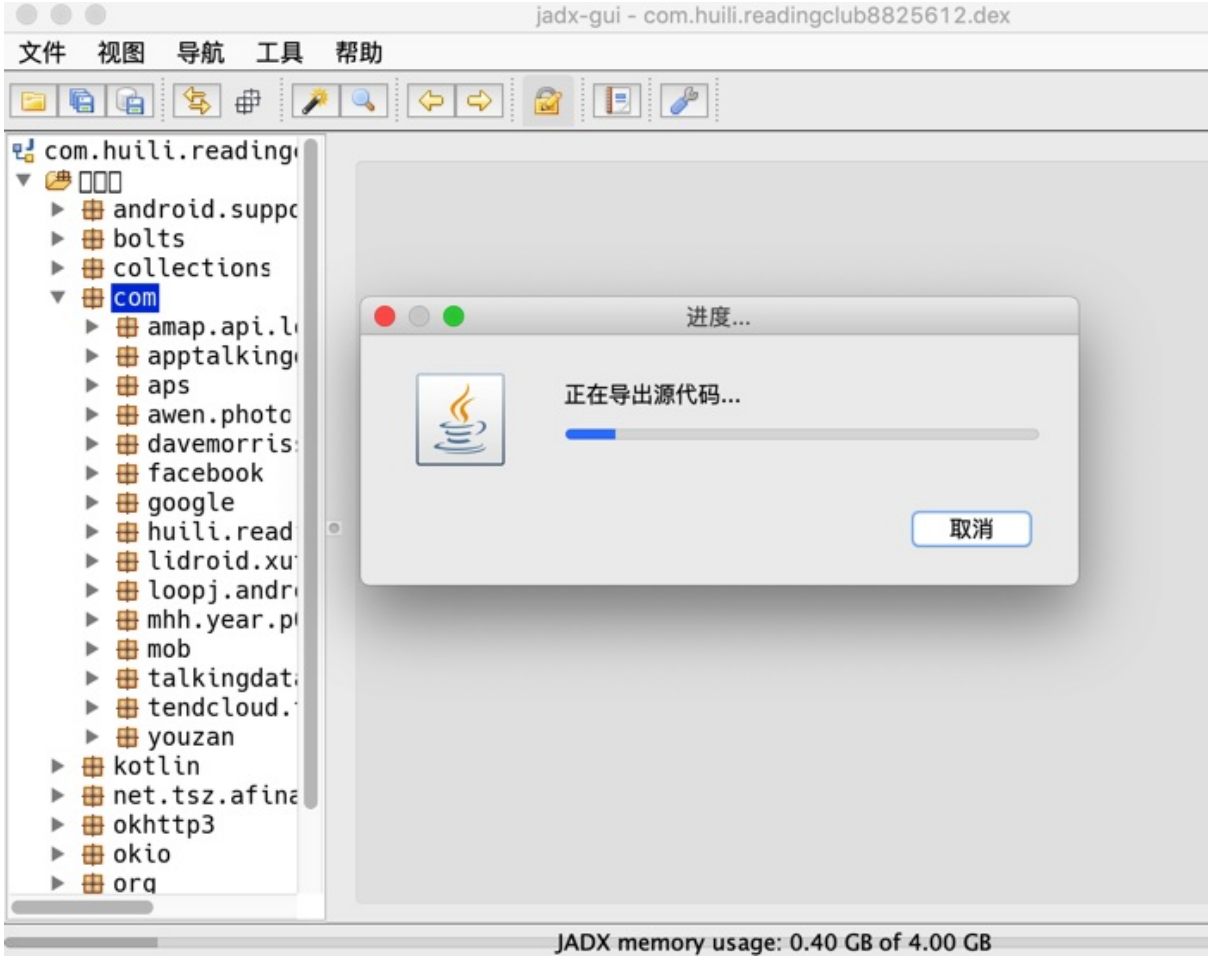


然后如果想要导出全部代码，则可以去：

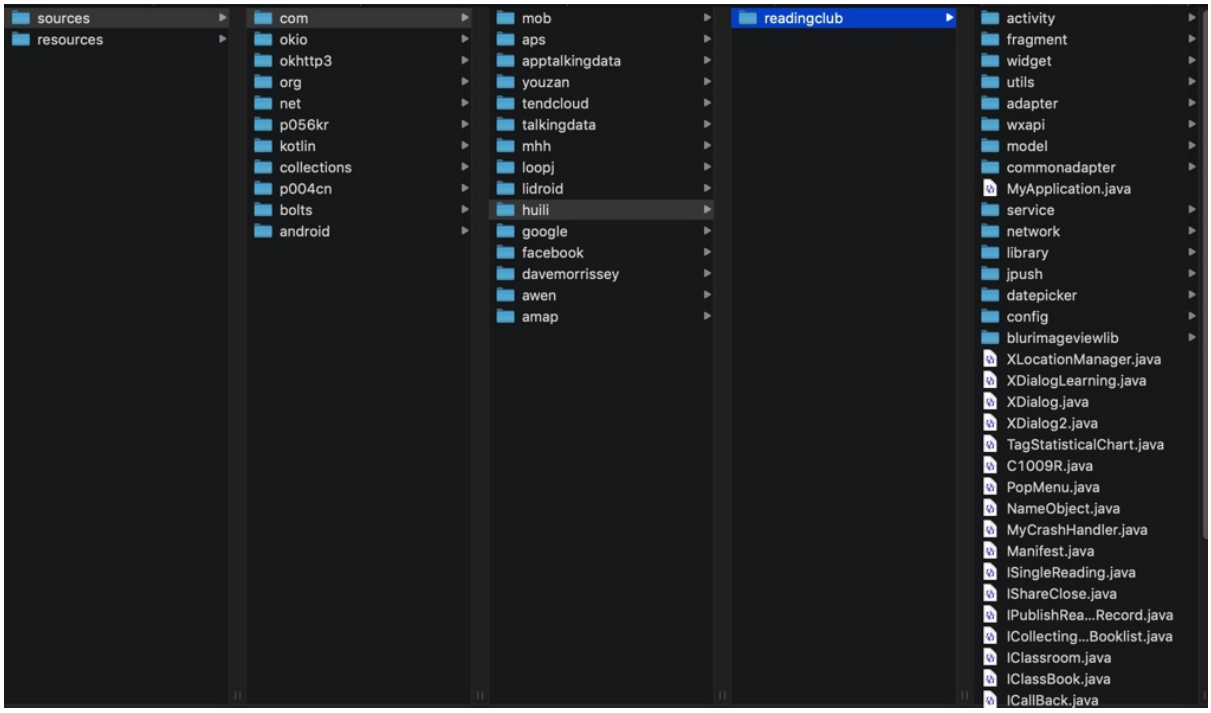
```
File -> Save All
```


jadx-gui - com.huili.readingclub8825612.dex
文件
视图
导航
工具
帮助
打开...
^O
全部保存
^S
另存为 Gradle 项目
^E
全部保存
最近打开的文件
设置
^⇧P
退出
apptalking
aps
awen.photo
davemorris
facebook
google
huili.read
lidroid.xu
loopj.andr
mhh.year.p
mob
talkingdat
tendcloud.
youzan
kotlin
net.tsz.afina
okhttp3
okio
org
JADX memory usage: 0.26 GB of 4.00 GB

然后稍等片刻:



即可在导出的 `sources` 文件夹中找到你要的源码：



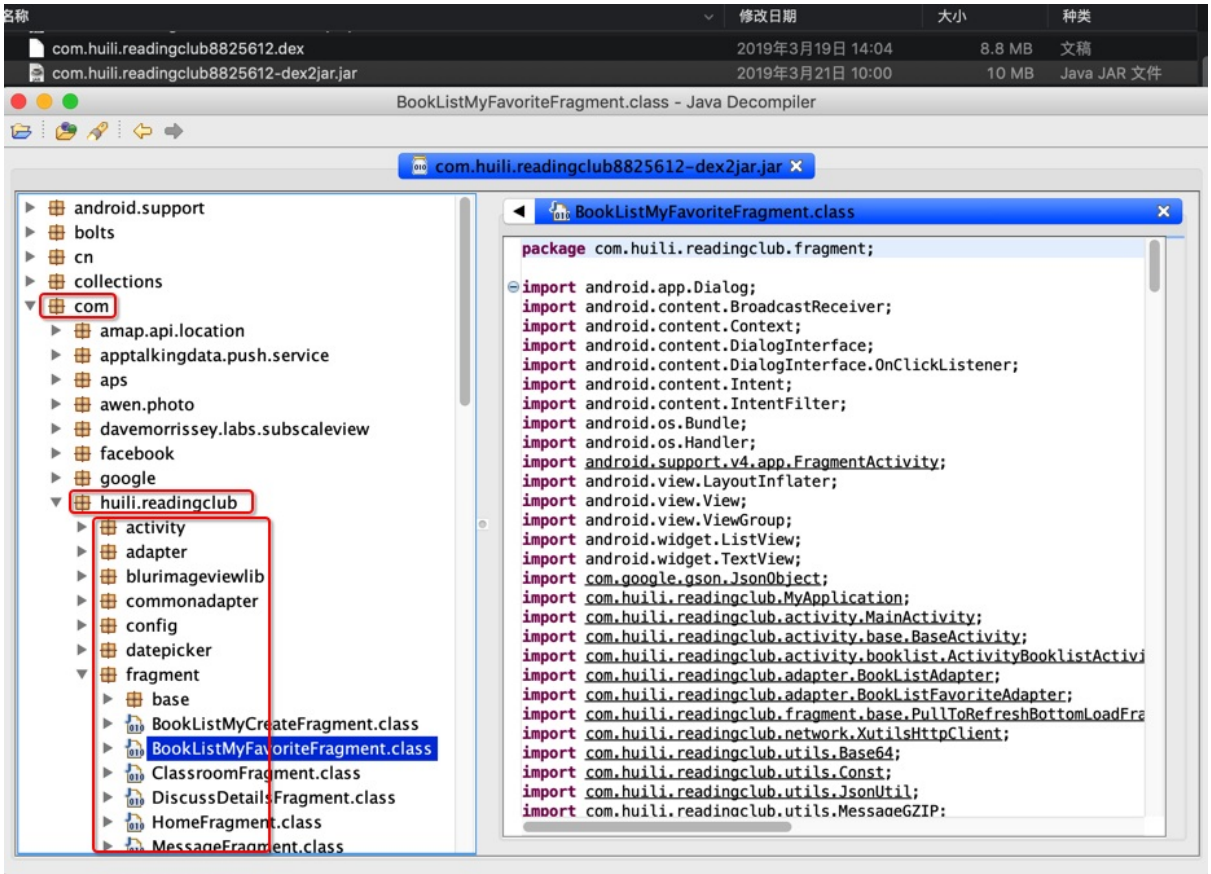
具体过程详见：

反编译器 Jadx

备注和说明

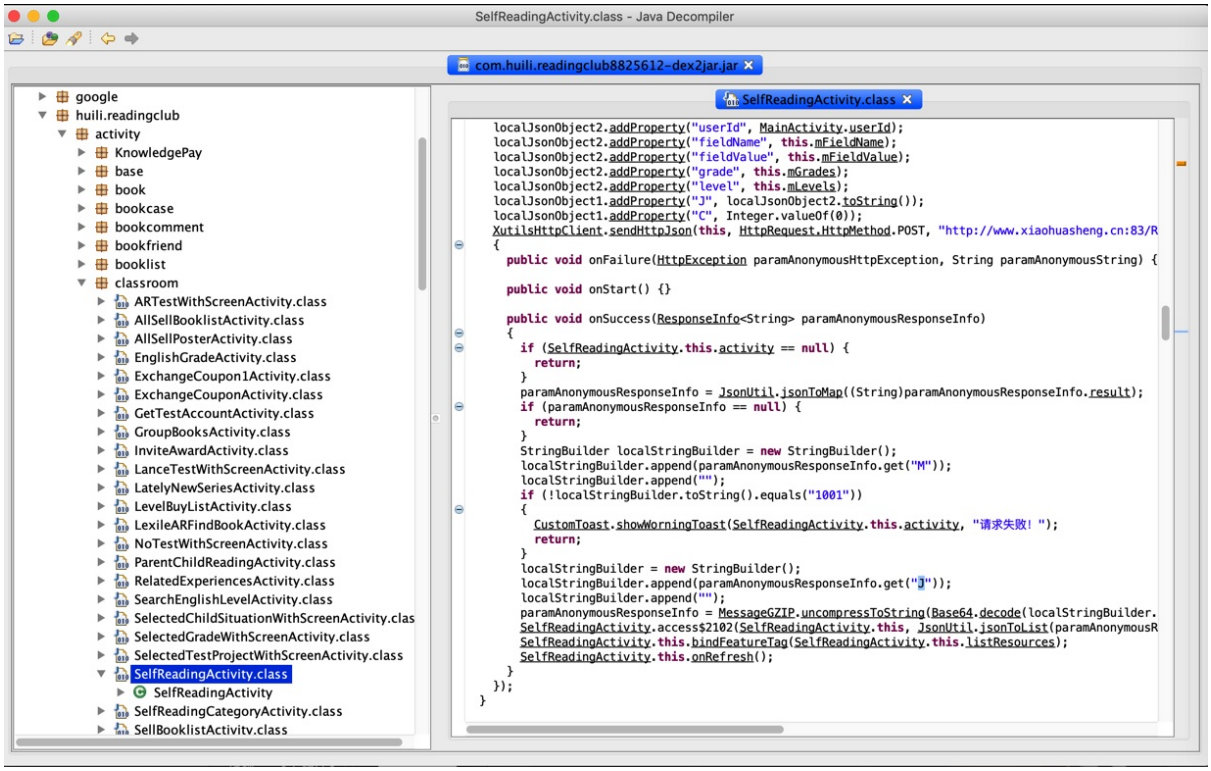
用 `JD-GUI` 打开同一个 `jar` 的效果

另外，用 `JD-GUI` 打开同一个jar的效果：



其中找到了我们之前需要的app相关的业务逻辑的代码：

`/com/huili/readingclub/activity/classroom/SelfReadingActivity.class`

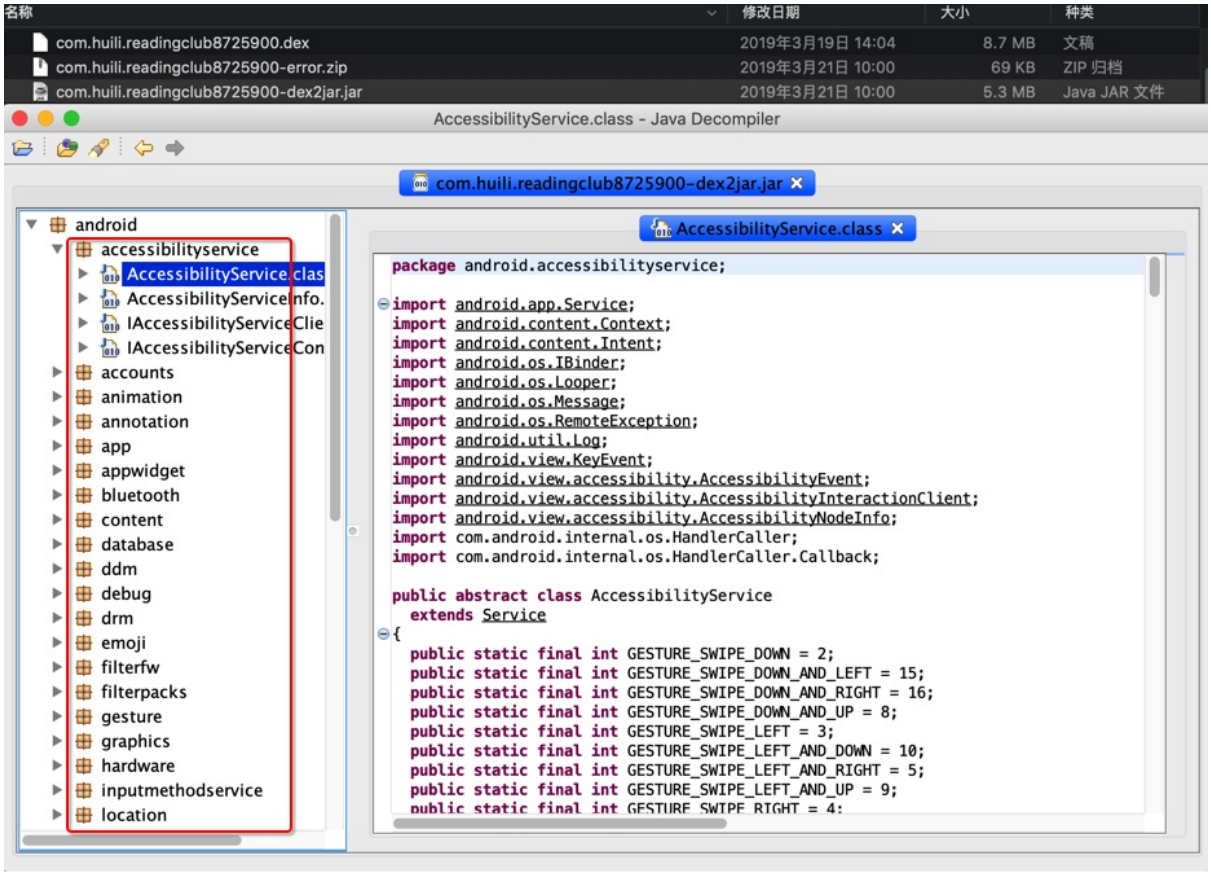


其中 `onSuccess` 中就是我们希望得到的对于 `J` 字段解密的逻辑。

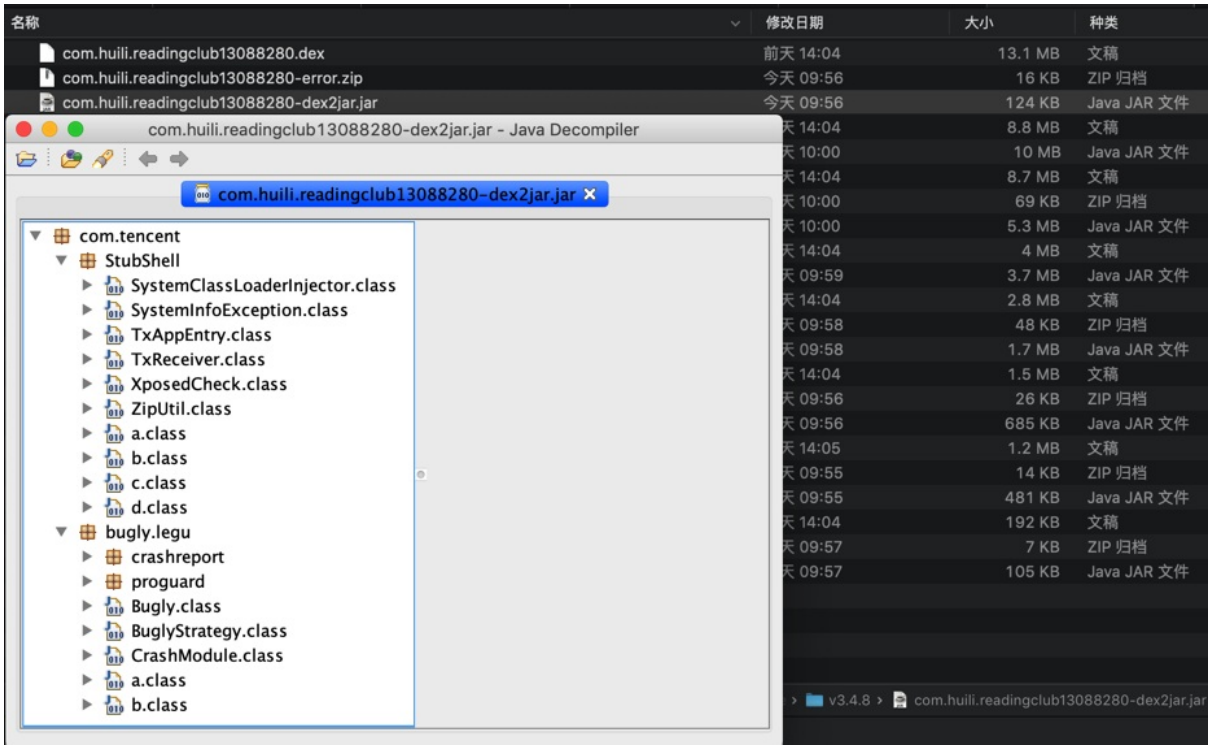
其他无效的jar转换出jar的效果

如前一步所述，从多个 `dex` 可以转换出多个 `jar`

而这些无效的、没有包含app业务逻辑的 `jar` ，去用一些反编译工具打开后的效果是：



其他的一些，比如腾讯乐固加密了的，最终转换出来的jar，去打开后只能看到腾讯乐固的代码：



注意事项

`jadx` 不能从 `jar` 导出 `java` ，否则会报错

举例：

```
../../../../../reverse_engineering/jadx/jadx-1.0.0/bin/jadx ../dex_to_jar/com.ishowedu.child.peiyin9201516-dex2jar.jar -d .

INFO  - loading ...
INFO  - converting to dex: com.ishowedu.child.peiyin9201516-dex2jar.jar ...
ERROR - jadx error: Error load file: ../dex_to_jar/com.ishowedu.child.peiyin9201516-dex2jar.jar
jadx.core.utils.exceptions.JadxRuntimeException: Error load file: ../dex_to_jar/com.ishowedu.child.peiyin9201516-dex2jar.j
ar
    at jadx.api.JadxDecompiler.loadFiles(JadxDecompiler.java:138)
    at jadx.api.JadxDecompiler.load(JadxDecompiler.java:102)
    at jadx.cli.JadxCLI.processAndSave(JadxCLI.java:32)
    at jadx.cli.JadxCLI.main(JadxCLI.java:18)
Caused by: jadx.core.utils.exceptions.DecodeException: java class to dex conversion error:
 dx exception: Translation has been interrupted
    at jadx.core.utils.files.InputFile.loadFromJar(InputFile.java:191)
    at jadx.core.utils.files.InputFile.searchDexFiles(InputFile.java:82)
    at jadx.core.utils.files.InputFile.addFilesFrom(InputFile.java:40)
    at jadx.api.JadxDecompiler.loadFiles(JadxDecompiler.java:136)
    ... 3 common frames omitted
Caused by: jadx.core.utils.exceptions.JadxException: dx exception: Translation has been interrupted
    at jadx.core.utils.files.JavaToDex.convert(JavaToDex.java:63)
    at jadx.core.utils.files.InputFile.loadFromJar(InputFile.java:182)
    ... 6 common frames omitted
Caused by: java.lang.RuntimeException: Translation has been interrupted
    at com.android.dx.command.dexer.Main.processAllFiles(Main.java:614)
    at com.android.dx.command.dexer.Main.runMultiDex(Main.java:365)
    at com.android.dx.command.dexer.Main.runDx(Main.java:286)
    at jadx.core.utils.files.JavaToDex.convert(JavaToDex.java:49)
    ... 7 common frames omitted
Caused by: java.lang.InterruptedException: Too many errors
    at com.android.dx.command.dexer.Main.processAllFiles(Main.java:606)
    ... 10 common frames omitted
```

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2020-09-05 09:45:45

2.2.1 dex转换出jar

思路

从前面的 `app导出dex` ，我们已得到：（一个或）多个dex文件，而这么多 `dex` ，其中只有一个是真正包含了安卓app的业务逻辑的 `dex` 文件。

我们后续会（经过转换和尝试而）找到该 `dex` ，然后用 `dex2jar` 等工具从 `dex` 转换出 `jar` 文件

准备

- 下载 `dex2jar`
  - 从dex2jar的下载页面下载到最新版本的 `dex2jar`
    - 比如：dex-tools-2.1-SNAPSHOT.zip
      - 解压后得到： `d2j-dex2jar.sh`

详细步骤

比如前面的 `v3.4.8` 版本的某安卓apk，经过前面一步导出了多个 `dex` 文件后，此处接着去用 `dex2jar` 去分别转换出 `jar` ：

```
d2j-dex2jar.sh -f dex_file.dex
```

期间，很多个 `dex` 转换 `jar` 的期间都会报错。

经过尝试最终找到了，转换不仅不报错且转换出来的 `jar` 还是我们希望的有效的包含了app业务逻辑的 `jar` ：

```
➜  v3.4.8 /Users/crifan/dev/dev_tool/android/reverse_engineering/dex-tools/dex-tools-2.1-SNAPSHOT/d2j-dex2jar.sh -f com.hu
ili.readingclub3986968.dex
dex2jar com.huili.readingclub3986968.dex -> ./com.huili.readingclub3986968-dex2jar.jar
➜  v3.4.8 /Users/crifan/dev/dev_tool/android/reverse_engineering/dex-tools/dex-tools-2.1-SNAPSHOT/d2j-dex2jar.sh -f com.hu
ili.readingclub8825612.dex
dex2jar com.huili.readingclub8825612.dex -> ./com.huili.readingclub8825612-dex2jar.jar
```

经后续确定，此处：

- 从： `8.4MB` 的 `com.huili.readingclub8825612.dex`
- 转换出的： `10MB` 的 `com.huili.readingclub8825612-dex2jar.jar`

就是我们要的，包含了app的业务逻辑的代码。

说明和提示

apk改名zip解压得到的 `classes.dex` 去转换一般无法得到有效的 `jar`

比如之前的 `v3.6.9` 版本的某安卓apk，改名为 `zip` 再解压得到的 `classes.dex` ，此处用 `dex2jar` 去转换：

```
➜  classes.dex pwd
/Users/crifan/dev/dev_root/company/naturling/projects/crawl_data/小花生app/xiaohuasheng/decoded dex/v3.6.9/classes.dex
➜  classes.dex /Users/crifan/dev/dev_tool/android/reverse_engineering/dex-tools/dex-tools-2.1-SNAPSHOT/d2j-dex2jar.sh -f c
lasses.dex
```

```
dex2jar classes.dex -  ./classes-dex2jar.jar
➜  classes.dex ll
total 26000
-rw-------  1 crifan  staff   212K  3 25 10:20 classes-dex2jar.jar
-rwxr-xr-x@ 1 crifan  staff    12M  1 25 17:43 classes.dex
```

虽然转换过程并没有报错，但是才得到200多KB的jar

-》最终经确认，其中没有包含业务逻辑代码，不是我们需要的jar，是无效的jar

多个 `dex` 转换 `jar` 的期间会报错

比如：

```
➜  v3.4.8 /Users/crifan/dev/dev_tool/android/reverse_engineering/dex-tools/dex-tools-2.1-SNAPSHOT/d2j-dex2jar.sh -f com.hu
ili.readingclub1166288.dex
...
GLITCH: 0000 Lcom/android/internal/telephony/uicc/VoiceMailConstants;.getVoiceMailTag(Ljava/lang/String;)Ljava/lang/String;
  zero-width instruction op 0xf4
Detail Error Information in File ./com.huili.readingclub1166288-error.zip
Please report this file to one of following link if possible (any one).
    https://sourceforge.net/p/dex2jar/tickets/
    https://bitbucket.org/pxb1988/dex2jar/issues
    https://github.com/pxb1988/dex2jar/issues
    dex2jar@googlegroups.com

➜  v3.4.8 /Users/crifan/dev/dev_tool/android/reverse_engineering/dex-tools/dex-tools-2.1-SNAPSHOT/d2j-dex2jar.sh -f com.hu
ili.readingclub13088280.dex
...
GLITCH: 009f Lcom/tencent/bugly/legu/proguard/z;.a(Ljava/lang/Thread;Ljava/lang/String;Ljava/lang/String;Ljava/lang/String;
)V  zero-width instruction op 0xf8
Detail Error Information in File ./com.huili.readingclub13088280-error.zip


➜  v3.4.8 /Users/crifan/dev/dev_tool/android/reverse_engineering/dex-tools/dex-tools-2.1-SNAPSHOT/d2j-dex2jar.sh -f com.hu
ili.readingclub1461452.dex
...
GLITCH: 0000 Lcom/google/android/util/SmileyResources;.getSmileys()Lcom/google/android/util/AbstractMessageParser$TrieNode;
  zero-width instruction op 0xf4
WARN: can't get operand(s) for sub-double/2addr, out-of-range or not initialized ?
WARN: can't get operand(s) for int-to-float, out-of-range or not initialized ?
WARN: can't get operand(s) for return-wide, out-of-range or not initialized ?
WARN: can't get operand(s) for move-exception, out-of-range or not initialized ?
WARN: can't get operand(s) for move-exception, out-of-range or not initialized ?
Detail Error Information in File ./com.huili.readingclub1461452-error.zip


➜  v3.4.8 /Users/crifan/dev/dev_tool/android/reverse_engineering/dex-tools/dex-tools-2.1-SNAPSHOT/d2j-dex2jar.sh -f com.hu
ili.readingclub191572.dex
...
GLITCH: 0006 Lcom/android/okhttp/internal/tls/OkHostnameVerifier;.verifyHostName(Ljava/lang/String;Ljava/lang/String;)Z |
zero-width instruction op=0xee
Detail Error Information in File ./com.huili.readingclub191572-error.zip

➜  v3.4.8 /Users/crifan/dev/dev_tool/android/reverse_engineering/dex-tools/dex-tools-2.1-SNAPSHOT/d2j-dex2jar.sh -f com.hu
ili.readingclub2847840.dex
...
GLITCH: 0006 Lsun/misc/Unsafe;.unpark(Ljava/lang/Object;)V | zero-width instruction op=0xf8
Detail Error Information in File ./com.huili.readingclub2847840-error.zip
➜  v3.4.8 /Users/crifan/dev/dev_tool/android/reverse_engineering/dex-tools/dex-tools-2.1-SNAPSHOT/d2j-dex2jar.sh -f com.hu
ili.readingclub8725900.dex
...
GLITCH: 0000 Landroid/widget/ZoomControls;.setOnZoomOutClickListener(Landroid/view/View$OnClickListener;)V | zero-width in
struction op=0xf4
GLITCH: 0000 Landroid/widget/ZoomControls;.setZoomSpeed(J)V | zero-width instruction op=0xf4
WARN: can't get operand(s) for move-result-object, wrong position ?
WARN: can't get operand(s) for move-result-object, wrong position ?
WARN: can't get operand(s) for move-result-object, wrong position ?
WARN: can't get operand(s) for move-object/16, out-of-range or not initialized ?
WARN: can't get operand(s) for shr-int/2addr, out-of-range or not initialized ?
```

```
WARN: can't get operand(s) for move/16, out-of-range or not initialized ?
WARN: can't get operand(s) for move-result-object, wrong position ?
WARN: can't get operand(s) for move/16, out-of-range or not initialized ?
WARN: can't get operand(s) for move-result, wrong position ?
WARN: can't get operand(s) for cmpl-float, out-of-range or not initialized ?
WARN: can't get operand(s) for move-result-object, wrong position ?
WARN: can't get operand(s) for sput-boolean, out-of-range or not initialized ?
WARN: can't get operand(s) for move-result-object, wrong position ?
WARN: can't get operand(s) for move-result-object, wrong position ?
WARN: can't get operand(s) for move-result-object, wrong position ?
WARN: can't get operand(s) for move-object/from16, out-of-range or not initialized ?
WARN: can't get operand(s) for move-object/from16, out-of-range or not initialized ?
WARN: can't get operand(s) for move-object/from16, out-of-range or not initialized ?
WARN: can't get operand(s) for move-object/from16, out-of-range or not initialized ?
WARN: can't get operand(s) for move-object/from16, out-of-range or not initialized ?
WARN: can't get operand(s) for sput-boolean, out-of-range or not initialized ?
WARN: can't get operand(s) for sput-boolean, out-of-range or not initialized ?
WARN: can't get operand(s) for move-result-object, wrong position ?
WARN: can't get operand(s) for aput-char, out-of-range or not initialized ?
WARN: can't get operand(s) for mul-float, out-of-range or not initialized ?
WARN: can't get operand(s) for move-result-object, wrong position ?
WARN: can't get operand(s) for move-result-object, wrong position ?
WARN: can't get operand(s) for move-result-object, wrong position ?
WARN: can't get operand(s) for move-wide/16, out-of-range or not initialized ?
WARN: can't get operand(s) for move-result-object, wrong position ?
WARN: can't get operand(s) for mul-int/2addr, out-of-range or not initialized ?
WARN: can't get operand(s) for aput-char, out-of-range or not initialized ?
WARN: can't get operand(s) for aput-char, out-of-range or not initialized ?
WARN: can't get operand(s) for move-result-object, wrong position ?
WARN: can't get operand(s) for sput-byte, out-of-range or not initialized ?
WARN: can't get operand(s) for aget-byte, out-of-range or not initialized ?
WARN: can't get operand(s) for and-int/2addr, out-of-range or not initialized ?
WARN: can't get operand(s) for move/from16, out-of-range or not initialized ?
WARN: can't get operand(s) for iput-boolean, out-of-range or not initialized ?
WARN: can't get operand(s) for iput-boolean, out-of-range or not initialized ?
WARN: can't get operand(s) for move-result-object, wrong position ?
WARN: can't get operand(s) for cmpg-float, out-of-range or not initialized ?
Detail Error Information in File ./com.huili.readingclub8725900-error.zip
Please report this file to one of following link if possible (any one).
    https://sourceforge.net/p/dex2jar/tickets/
    https://bitbucket.org/pxb1988/dex2jar/issues
    https://github.com/pxb1988/dex2jar/issues
    dex2jar@googlegroups.com
java.util.IllegalFormatConversionException: d  java.lang.String
    at java.util.Formatter$FormatSpecifier.failConversion(Formatter.java:4302)
    at java.util.Formatter$FormatSpecifier.printInteger(Formatter.java:2793)
    at java.util.Formatter$FormatSpecifier.print(Formatter.java:2747)
    at java.util.Formatter.format(Formatter.java:2520)
    at java.util.Formatter.format(Formatter.java:2455)
    at java.lang.String.format(String.java:2940)
    at com.googlecode.d2j.smali.BaksmaliDumpOut.s(BaksmaliDumpOut.java:68)
    at com.googlecode.d2j.smali.BaksmaliCodeDumper.visitFilledNewArrayStmt(BaksmaliCodeDumper.java:248)
    at com.googlecode.d2j.node.insn.FilledNewArrayStmtNode.accept(FilledNewArrayStmtNode.java:19)
    at com.googlecode.d2j.smali.BaksmaliDumper.accept(BaksmaliDumper.java:569)
    at com.googlecode.d2j.smali.BaksmaliDumper.baksmaliCode(BaksmaliDumper.java:544)
    at com.googlecode.d2j.smali.BaksmaliDumper.baksmaliMethod(BaksmaliDumper.java:482)
    at com.googlecode.d2j.smali.BaksmaliDumper.baksmaliMethod(BaksmaliDumper.java:428)
    at com.googlecode.dex2jar.tools.BaksmaliBaseDexExceptionHandler.dumpMethod(BaksmaliBaseDexExceptionHandler.java:148)
    at com.googlecode.dex2jar.tools.BaksmaliBaseDexExceptionHandler.dumpTxt0(BaksmaliBaseDexExceptionHandler.java:126)
    at com.googlecode.dex2jar.tools.BaksmaliBaseDexExceptionHandler.dumpZip(BaksmaliBaseDexExceptionHandler.java:135)
    at com.googlecode.dex2jar.tools.BaksmaliBaseDexExceptionHandler.dump(BaksmaliBaseDexExceptionHandler.java:92)
    at com.googlecode.dex2jar.tools.Dex2jarCmd.doCommandLine(Dex2jarCmd.java:120)
    at com.googlecode.dex2jar.tools.BaseCmd.doMain(BaseCmd.java:290)
    at com.googlecode.dex2jar.tools.Dex2jarCmd.main(Dex2jarCmd.java:33)
```

其中注意到：

- 如果报错，会有把错误信息导出到名为xxx-error.zip的压缩文件中。
  - 比如：`com.huili.readingclub2847840-error.zip`
    - 可供后续分析使用

- 后续经过确认，这些报错的，往往是没有包含app业务逻辑的，不是我们要的dex，所以可以忽略这些错误

上述所有的 `dex` 转换为 `jar` 之后：

```
➜  v3.4.8 ll
total 125288
-rw-------  1 crifan  staff   469K  3 21 09:55 com.huili.readingclub1166288-dex2jar.jar
-rw-r--r--  1 crifan  staff    14K  3 21 09:55 com.huili.readingclub1166288-error.zip
-rw-------  1 crifan  staff   1.1M  3 19 14:05 com.huili.readingclub1166288.dex
-rw-------  1 crifan  staff   121K  3 21 09:56 com.huili.readingclub13088280-dex2jar.jar
-rw-r--r--  1 crifan  staff    16K  3 21 09:56 com.huili.readingclub13088280-error.zip
-rw-------  1 crifan  staff    12M  3 19 14:04 com.huili.readingclub13088280.dex
-rw-------  1 crifan  staff   669K  3 21 09:56 com.huili.readingclub1461452-dex2jar.jar
-rw-r--r--  1 crifan  staff    25K  3 21 09:56 com.huili.readingclub1461452-error.zip
-rw-------  1 crifan  staff   1.4M  3 19 14:04 com.huili.readingclub1461452.dex
-rw-------  1 crifan  staff   103K  3 21 09:57 com.huili.readingclub191572-dex2jar.jar
-rw-r--r--  1 crifan  staff   7.0K  3 21 09:57 com.huili.readingclub191572-error.zip
-rw-------  1 crifan  staff   187K  3 19 14:04 com.huili.readingclub191572.dex
-rw-------  1 crifan  staff   1.6M  3 21 09:58 com.huili.readingclub2847840-dex2jar.jar
-rw-r--r--  1 crifan  staff    47K  3 21 09:58 com.huili.readingclub2847840-error.zip
-rw-------  1 crifan  staff   2.7M  3 19 14:04 com.huili.readingclub2847840.dex
-rw-------  1 crifan  staff   3.5M  3 21 09:59 com.huili.readingclub3986968-dex2jar.jar
-rw-------  1 crifan  staff   3.8M  3 19 14:04 com.huili.readingclub3986968.dex
-rw-------  1 crifan  staff   5.1M  3 21 10:00 com.huili.readingclub8725900-dex2jar.jar
-rw-r--r--  1 crifan  staff    68K  3 21 10:00 com.huili.readingclub8725900-error.zip
-rw-------  1 crifan  staff   8.3M  3 19 14:04 com.huili.readingclub8725900.dex
-rw-------  1 crifan  staff   9.5M  3 21 10:00 com.huili.readingclub8825612-dex2jar.jar
-rw-------  1 crifan  staff   8.4M  3 19 14:04 com.huili.readingclub8825612.dex
```

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2020-01-16 22:31:27

2.2.2 jar转换出java

准备

- 前面一步从 `dex` 转换得到的 `jar` 文件
    -
- 选择合适的反编译器： `procyon`
    - 用于从 `jar` 转换出 `java` 源代码
    - 网上很多人提到了用的比较广的： `JD-GUI`
        - 经过实测，基本够用，但不够完美
    - 后续自己测试了多个其他的反编译器
    - 最终结论如下：
        - `Jadx` > `Procyon` > `CFR` > `JD-GUI`
            - 详见：常见反编译器对比 · 安卓应用的安全和破解
            - 注：此处 `jadx` 其实是，直接从 `dex` 转 `java` ，而其他几个反编译器是从 `jar` 转 `java` ，略有不同。请使用时稍微注意区别。
    - 所以此处选用： `procyon`
        - 关于反编译效果更好的 `jadx` ，详见前面章节。

详细步骤

比如用procyon反编译器从jar中导出java源码

语法：

```
java -jar /path/to/your/procyon-decompiler-0.5.36.jar -jar jav_file.jar -o output_folder
```

举例：

```
java -jar  /Users/crifan/dev/dev_tool/android/reverse_engineering/Procyon/procyon-decompiler-0.5.36.jar -jar ../../dex_to_
jar/com.ishowedu.child.peiyin8392664-dex2jar.jar -o com.ishowedu.child.peiyin8392664_java
```

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2020-01-16 22:30:12

常见安卓破解工具

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:23:50

从app导出dex

下面整理关于从 `运行中的安卓app` 导出 `dex` 文件的一些 `hook` 工具。

从app导出dex的文件的命名

折腾 `FDex2` 等工具，从运行中的app导出 `dex` 文件时，看到很多文件名都是类似于

`com.huili.readingclub8825612.dex`

对这类文件名，并没注意到有何特别。

后来从别人的帖子中，突然意识到：

这些hook工具的生成dex的代码中，对应的命名规则应该是：

`packageName` + `fileSize` .dex

所以此处的

`com.huili.readingclub8825612.dex`

对应着就是：

- `packageName` ： `com.huili.readingclub`
- `fileSize` ： `8825612`
  - = `8.4MB`
    - 就是dex文件本身的大小

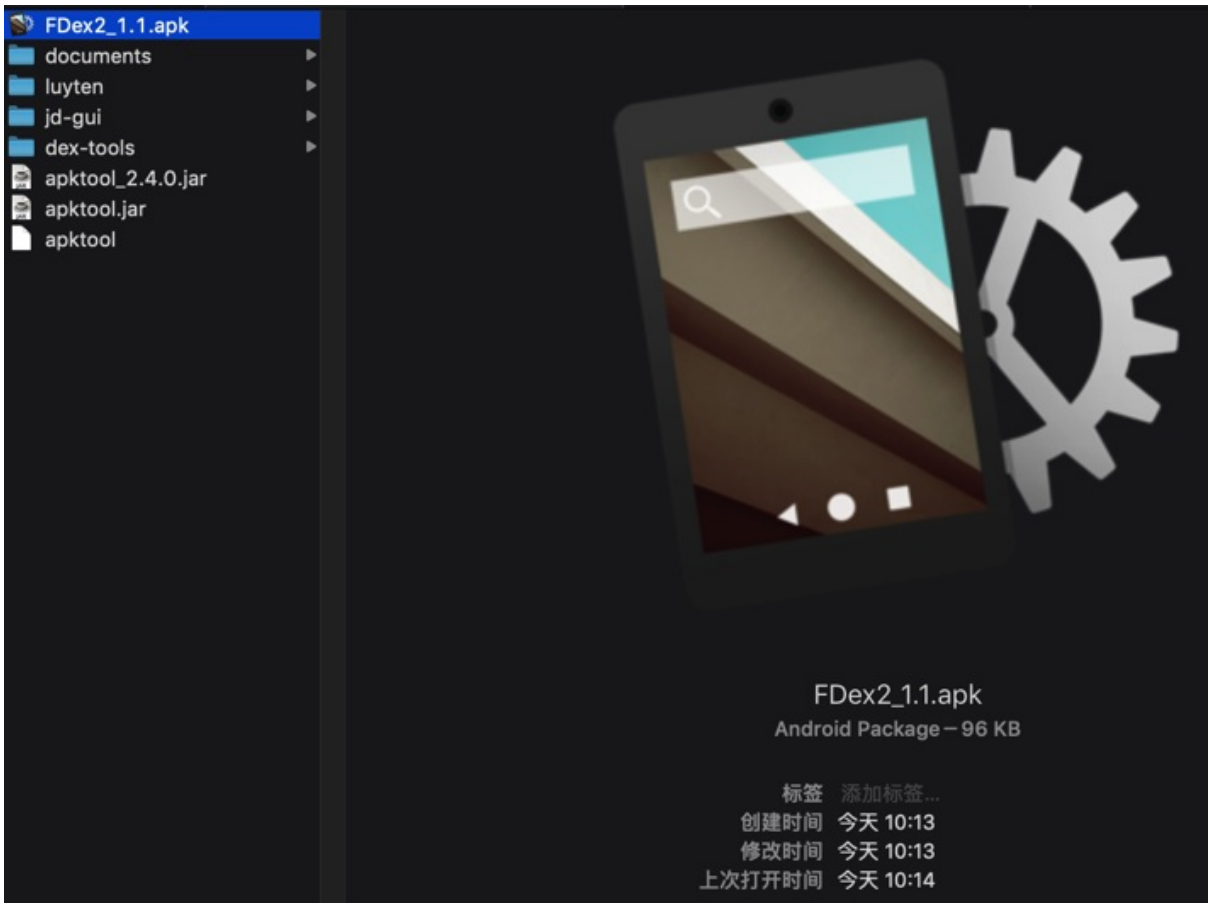
crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:14:57

FDex2

用来从 `运行中的安卓app` 中导出 `dex` 文件的工具。

- 下载地址
  - 百度网盘
    - 链接: https://pan.baidu.com/s/1ITF8CN96bxWpFwv7J174lg 提取码: 3e3t
  - CSDN
    - 脱壳工具 FDex2-CSDN下载

下载得到 `96KB` 的apk文件： `FDex2_1.1.apk`



crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:00:46

DumpDex

用来从 `运行中的安卓app` 中导出 `dex` 文件的工具。

- 官网地址
  - WrBug/dumpDex: 一款Android脱壳工具，需要xposed支持, 易开发已集成该项目

不要从DeveloperHelper下载

对于想要去下载dumpDex的话，建议去：

https://github.com/WrBug/dumpDex/releases

而不要去作者所说的，去下载

集成了DumpDex的DeveloperHelper

-> 否则我此处会导致夜神模拟器中的app运行时产生崩溃，从而无法导出想要的dex文件

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:00:43

drizzleDumper

听别人提到过，自己没用过。

- 功能
  - 一款基于内存搜索的Android脱壳工具
    - 可以从运行中的安卓app中，利用 `ptrace机制` ，导出dex文件
- github主页
  - DrizzleRisk/drizzleDumper: drizzleDumper是一款基于内存搜索的Android脱壳工具
- 机制和原理
  - root设备之后，通过ptrace附加需要脱壳的apk进程，然后在脱壳的apk进程的内存中进行dex文件的特征搜索，当搜索到dex文件时，进行dex文件的内存dump
- 使用步骤
  - 将 \armeabi 下的drizzleDumper push进手机
  - 进入shell，赋给可执行权限
  - 运行drizzleDumper [包名] [等待时间,默认为0]
  - 运行需要脱壳程序
- 使用举例

```
$ adb push F:\drizzleDumper /data/local/tmp
$ chmod 755 drizzleDumper
$ ./drizzleDumper xyz.sysorem.crackme
```

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:00:33

DexExtractor

- 功能：
  - 用于破解邦邦加密的安卓 `dex` 文件提取器
- github主页
  - lambdalang/DexExtractor
- 使用说明
  - 4.4的虚拟机 编译好了libdvm。
  - 代码github上，脱梆梆的壳，别的没测试
  - 把dexdump出来，然后base64解码下，然后odex2dex，没了
  - system.img 有空上传
    - 作者编译好的镜像文件 `system.img` 的下载地址
      - system-arm_md5_6395c2f1451dbbed027d7293ab39a6e7.img.tar.gz
  - 启动模拟器加上sdcard
- 注意
  - apk没有写权限的反编译了加上write就好了。
- 支持
  - 梆梆加固
  - 爱加密（新版本没事）
  - 其他，暂时没测试

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:00:22

InDroid

- 主页
  - romangol/InDroid: Dalvik vm Instrumentation OS
- 作者
  - GoSSIP小组
- 功能
  - 基于Dalvik VM的插桩分析框架
- 原理
  - 直接修改AOSP上的Dalvik VM解释器，在解释器解释执行Dalvik字节码时，插入监控的代码，这样就可以获取到所有程序运行于Dalvik上的动态信息，如执行的指令、调用的方法信息、参数返回值、各种Java对象的数据等等。InDroid只需要修改AOSP的dalvik vm部分代码，编译之后，可直接将编译生成的新libdvm.so刷入任何AOSP支持的真机设备上

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:00:51

dex转jar

此处整理一些从 `dex` 文件转换出 `jar` 文件的工具。

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:20:21

dex2jar

主页

- 官网
  - dex2jar | Penetration Testing Tools
- 其他主页/镜像
  - github
    - pxb1988/dex2jar: Tools to work with android .dex and java .class files
  - BitBucket
    - https://bitbucket.org/pxb1988/dex2jar
  - SourceForge
    - https://sourceforge.net/p/dex2jar
  - Google Code
    - https://code.google.com/p/dex2jar

功能

- 用于处理 `安卓` 的 `.dex` 文件和 `java` 的 `.class` 文件的一系列的工具
  - 核心和常用功能
    - 从 `dex` 文件导出 `jar` 文件
  - 一系列的工具，包括
    - `dex-reader/writer`：读写 `dex` （Dalvik Executable）文件
      - 具有和 `ASM` 类似的轻量级的API接口
    - `d2j-dex2jar`：把 `dex` 文件转换为 `class` 文件（=jar压缩包文件= `jar` 包= `jar` 文件）
    - `smali/baksmali`：反汇编 `dex` 转换出 `smali` 文件，从`smali` 文件中汇编出 `dex` 文件
      - 和smali/baksmali虽然语法相同，但不太一样的是，此处支持描述中包含 `"Lcom/dex2jar\t\u1234;"` 这类文字描述
    - 其他一些工具
      - d2j-decrypt-string

用法和举例

```
d2j-dex2jar Usage Example
root@kali:~# d2j-dex2jar /usr/share/metasploit-framework/data/android/apk/classes.dex
dex2jar /usr/share/metasploit-framework/data/android/apk/classes.dex -  classes-dex2jar.jar
```

最基本的用法：

- 从 `apk` 中转换出 `jar`：
  - `sh dex-tools/dex-tools-2.1-SNAPSHOT/d2j-dex2jar.sh -f ~/path/to/apk_to_decompile.apk`
- 从 `dex` 中转换出 `jar`：
  - `sh dex-tools/dex-tools-2.1-SNAPSHOT/d2j-dex2jar.sh -f ~/path/to/dex_to_decompile.dex`

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:19:44

Enjarify

- 主页
  - Github
    - 旧：google/enjarify
    - 新：Storyyeller/enjarify
- 下载
  - Releases · Storyyeller/enjarify
- 功能
  - 把安卓的 `Dalvik` 字节码转换为Java的 `字节码`
    - 和 `dex2jar` 类似
- 特点
  - Google 官方开源的
  - 用 `Python 3` 写的
  - 比 `dex2jar` 更新且更好
- 用法
  - `python3 -O -m enjarify.main yourapp.apk`
  - 已设置好环境变量后
    - `enjarify yourapp.apk`
    - `enjarify classes2.dex`
    - `enjarify yourapp.apk -o yourapp.jar`

为何不用 `dex2jar` ？

- `dex2jar`
  - 比较旧了
  - 大部分情况是转换没问题
  - 但是
    - 有时候
      - 模糊特性时
      - 边缘情况时
    - 转换
      - 会报错
      - 甚至不报错但生成的是错误的结果
- `Enjarify`
  - 最新设计的工具
  - 支持绝大多数（尽可能多的）情况
    - 包括有些dex2jar会出错的情况
    - 且额外支持
      - Unicode的类名
      - 用作多种类型的常量
      - 隐式转换
      - 正常执行流程中的异常处理
      - 引用了非常多的常量的类
      - 名字非常长的方法
      - 捕获函数后的异常处理
      - 错误类型的静态初始变量

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:19:48

Dedexer

- 主页
  - Dedexer user's manual
- 下载
  - dedexer - Browse Files at SourceForge.net
- 作用
  - 把dex转换为类似于汇编的格式
    - 目的：从dex文件创建类似于Jasmin的源代码
- 用法
  - `java -jar ddx.jar -o -D -d <destination directory> <source>`
  - `java -jar ddx1.5.jar -o -D -d c:\dex\gen c:\dex\classes.dex`
- 举例

```
D:\WINDOWS\system32
java -jar ddx1.5.jar -o -D -d c:\dex\gen c:\dex\classes.dex
Processing com/eoeandroid/market/MarketActivity$2
Processing com/eoeandroid/market/MarketActivity$1
```

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:19:40

反编译器

此处主要讨论java反编译器，更主要是的和安卓相关的java的反编译器，其中尤其涉及到dex转出jar包后的，如何从jar包反编译出java源代码的相关反编译器。

作用

- 从 `jar文件` = `jar包` 转换出 `java源代码`
  - 用于
    - 在GUI图形界面工具中查看java代码
    - 命令行工具中直接导出整个项目所有的java代码文件

总体结论

- 网上目前提到的最多的要数：`JD-GUI`
- 自己的经验和心得
  - `JD-GUI` 不太够用：
    - 有些文件转换会出错
    - 且代码逻辑也不够清晰
  - 目前自己
    - 以后尽量用 `Jadx`
      - 特点
        - 代码转换不仅不出错
        - 关键是代码逻辑更加清晰和易懂
          - 大大提升代码质量，使得代码更加易读
      - 用来
        - 直接用jadx打开未加固的apk
          - 即可查看和导出java源代码
        - 打开（用FDex2从加固了的apk导出的）dex文件
          - 查看和导出java源代码
    - 其次考虑用：`Procyon`
      - 或基于Procyon的GUI工具：`Luyten`
      - 特点
        - 代码转换不出错的
      - 用来
        - 查看代码：用基于Procyon的Luyten去查看代码
        - 导出代码：用Procyon去从jar反编译转换出java源代码

常用安卓相关java反编译器

网上有很多安卓相关的java反编译器，大致整理如下：

- 比较老旧的
  - Jad
    - 不再维护了，源码仓库已关闭
    - 不支持Java 5+
  - Java DeObfuscator

        - https://sourceforge.net/projects/jdo/
        - JDO is a Java DeObfuscator that works on class files directly. JDO contains a simple and easy to use GUI that makes automatic deobfuscation of Java projects a one-click operation!
    - JODE
        - http://jode.sourceforge.net/
        - a java package containing a decompiler and an optimizer for Java. This package is freely available under the GNU GPL. It hasn't been updated for quite some time.
    - AndroChef
        - http://www.androiddecompiler.com/
        - =
        - http://www.neshkov.com/ac_decompiler.html
        - 只支持Windows平台
    - Candle
        - https://github.com/bradsdavis/candle-decompiler
        - by Brad Davis, developer of JBoss Cake, is an early but promising work in progress= is far away from being feature complete
- 相对新的工具
    - JD-GUI
        - 官网地址：
            - http://jd.benow.ca/
            - -》
            - http://java-decompiler.github.io
        - is an Decompiler, which comes with its own GUI. All is licensed under GPLv3. Like CFR the source for the decompiler itself, is not published, but you have the right to decompile the bionaries. And the binaries are under an OpenSource-License (CFR is under the MIT-license and JD Core is under the GPLv3 license)
        - 转换效果：经常会报错
            - 更准确的说是：对于LINQ/DLR的树编译器产生的代码，会不支持，会报错
                - -》解析后的代码中，包含/ *Error* / // Byte code 这种代码
    - CFR
        - 官网
            - CFR - yet another java decompiler.
            - http://www.benf.org/other/cfr/
        - 特点
            - 支持java 9/10/12等
        - by Lee Benfield is well on its way to becoming the premier Java Decompiler. Lee and I actually work for the same company and share regression tests. We're engaged in a friendly competition to see who can deliver a better decompiler. Based on his progress thus far, there's a very good chance he will win--at least on decompiling obfuscated code :).
    - Krakatau
        - https://github.com/Storyyeller/Krakatau
        - by Robert Grosse, written in Python, includes a robust verifier. It focuses on translating arbitrary bytecode into valid Java code, as opposed to reconstructing the original code.
    - Fernflower
        - https://github.com/JetBrains/intellij-community/tree/master/plugins/java-decompiler/engine
        - https://github.com/fesh0r/fernflower
        - an analytical Java decompiler
    - Cavaj
        - https://cavaj-java-decompiler.jaleco.com/
    - Procyon
        - mstrobel / Procyon — Bitbucket

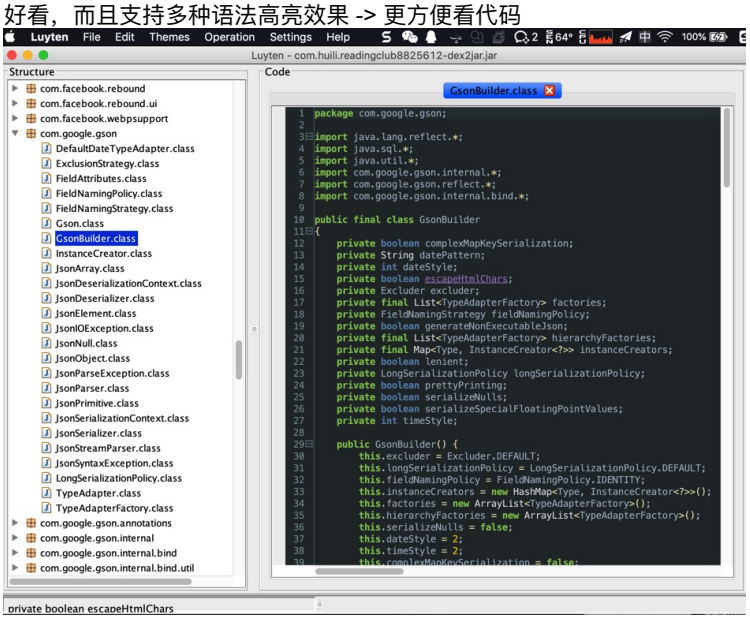
        - https://bitbucket.org/mstrobel/procyon
            - 官网使用文档
                - mstrobel / Procyon / wiki / Java Decompiler — Bitbucket
                - https://bitbucket.org/mstrobel/procyon/wiki/Java%20Decompiler
        - 重点关注Java 5之后的特性支持
            - 这些是其他很多反编译工具不支持的，会报错的
        - 更详细的解释
            - 枚举声明
            - 枚举和字符串的switch表达式
                - 目前只测试支持javac 1.7
            - 局部类Local classes
                - 匿名和带名字的都支持
            - 注解/标注Annotations
            - Java 8的Lambdas和方法引用(比如 :: 操作符)
        - -》对很多人了来说，比较关注：支持java8
        - Procyon本身是命令行工具
        - 基于Procyon的带GUI图形界面的工具
            - SecureTeam Java Decompiler
                - http://www.secureteam.net/Java-Decompiler.aspx
                - A JavaFX-based decompiler front-end with fast and convenient code navigation. Download it, or launch it directly from your browser.
            - Luyten
                - https://github.com/deathmarine/Luyten
                - An open source front-end by deathmarine
            - Bytecode Viewer
                - https://github.com/Konloch/bytecode-viewer
                - -》
                - https://bytecodeviewer.com
                - an open source Java decompilation, disassembly, and debugging suite by @Konloch. It can produce decompiled sources from several modern Java decompilers, including Procyon, CFR, and FernFlower.
            - Helios
                - https://github.com/samczsun/Helios
                - similar to Bytecode Viewer. But is a completely new and independent project, which uses SWT instead of Swing.
            - Enigma
                - http://www.cuchazinteractive.com/enigma/
                - Originally used to deobfuscate Minecraft versions. Uses Procyon internally.
                - 作者已不再维护
    - Jadx
        - 也支持从jar查看java代码
            - 和导出全部代码
                - 导出方式还有2种
                - 保存全部代码
                - 和以Gradle的方式导出源码
                - 如果打开的是apk文件
                - 则导出了dex对应的java源码外，还有assets等资源和其他文件
                - -》更加利于你得到更接近apk的原始代码的项目结构
            - 同时还支持直接打开apk
                - 查看apk中的各种文件
                - 包括java源代码

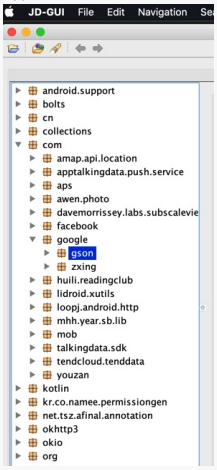
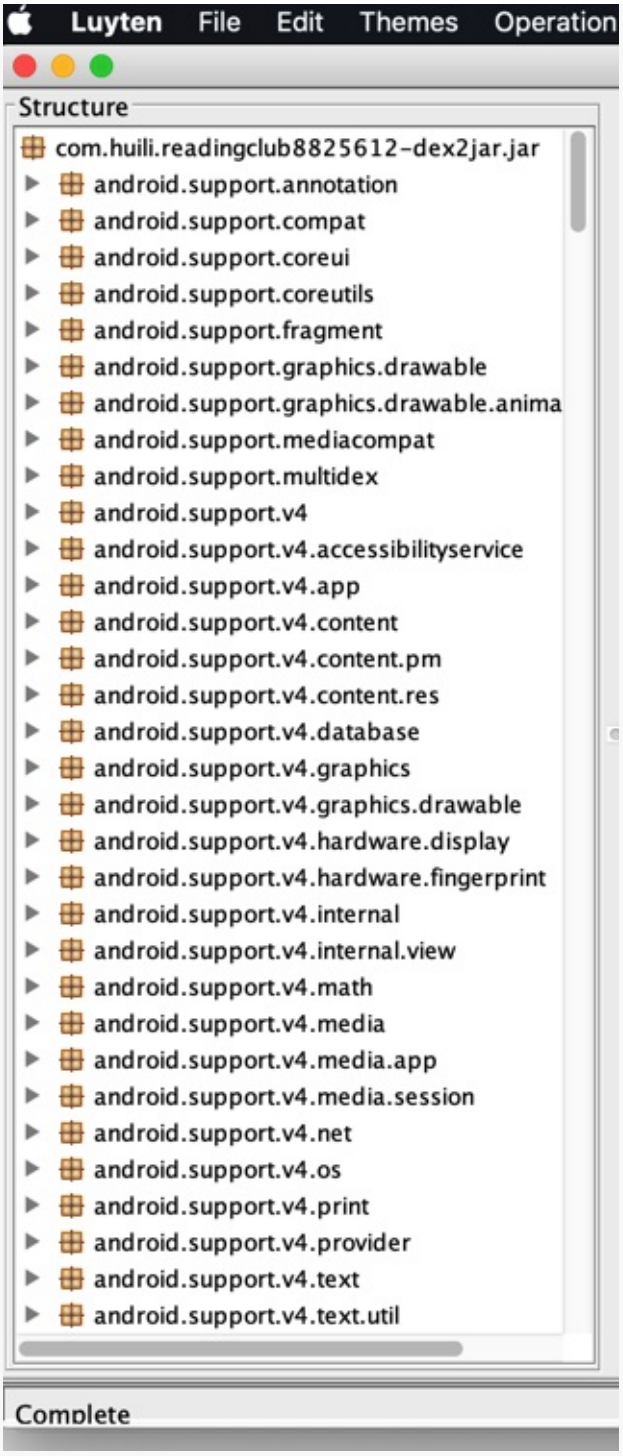
crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:19:33

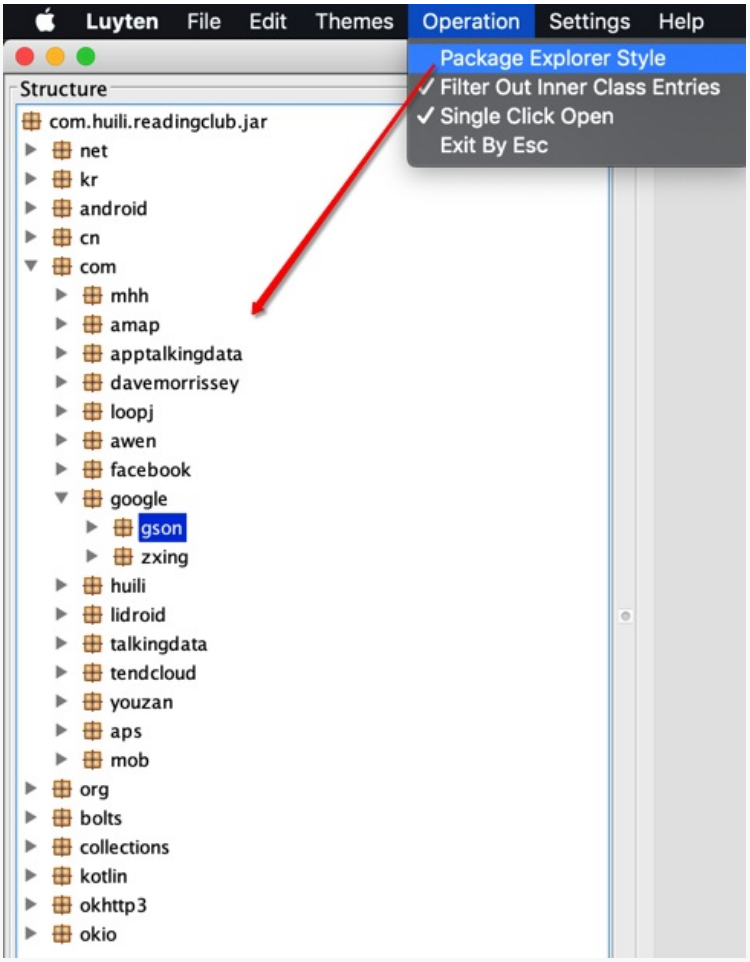
常见反编译器对比

下面就去对比常见的安卓的java反编译器的效果和优缺点。

`JD-GUI` vs (基于 `Procyon` 的) `Luyten`

对比项	JD-GUI	Luyten
流行程度	比较广->大家用的比较多	一般->相对使用的人不是很多
jar转java的准确率	高 ->jd-gui代码转换出错Error	很高 ->Luyten中可正确解析显示源码：
显示：整体UI界面和代码高亮	比较简单和朴素->不够炫酷和好看	好看，而且支持多种语法高亮效果 -> 更方便看代码 
		（默认）平铺直叙，直接显示的 ->层次结构不够清晰

显示： 包/类的 显示结构	以目录的树状结构显示的 -> 层次比较清晰  	 后记： 后来发现，把默认勾选的选项 `Operation -> Package Explorer Style` 取消勾选，也可以按照树状显示类和包名了：



`JD-GUI` VS `CFR` VS `Procyon` VS `Jadx`

对于同一个jar包：`com.huili.readingclub.jar`

用不同工具：

- JD-GUI
- CFR
- Procyon
- Jadx

去导出java源代码后，转换导出的效果，尤其是准确性，是否出错，是不一样的。

转换的细节是否完美

jd-gui的细节不够好的地方：

static函数：

```
static
{
  lock = new ReentrantLock();
}
```

而CFR、Procyon、Jadx转换结果可以更完整：

CFR的：

```
static {
    userTokens = new HashMap();
    lock = new ReentrantLock();
    condition = lock.newCondition();
}
```

Procyon

```
    static {
        ModelSecurity.userTokens = new HashMap<String, String>();
        lock = new ReentrantLock();
        condition = ModelSecurity.lock.newCondition();
    }
```

Jadx

```
public class ModelSecurity {
    private static final Condition condition = lock.newCondition();
    private static final Lock lock = new ReentrantLock();
    private static HashMap<String, String> userTokens = new HashMap();
}
```

很明显从：

- CFR的： `userTokens = new HashMap();`
- Procyon的： `ModelSecurity.userTokens = new HashMap<String, String>();`
- Jadx的： `private static HashMap<String, String> userTokens = new HashMap();`

可以看出：

- `Procyon` 能识别static变量，细节转换的很完美和精确
- 而 `Jadx` ： 更进一步识别出是private的static类型的变量，更加准确。

对于细节转换的结论：

- `JD-GUI` ： 细节不够好
- `CFR` ： 细节基本满足要求
- `Procyon` ： 细节完美转换
- `Jadx` ： 不仅完美且代码变量和结构更合理

转换是否出错及代码逻辑清晰度

getToken函数转换效果

比如： `com.huili.readingclub.model.ModelSecurity` 的 `getToken`

JD-GUI某函数转换报错得不到源码

都是错误代码：

```
  /* Error */
  private static String getToken(String paramString)
  {
    // Byte code:
    //   0: aload_0
    //   1: invokestatic 81    com/huili/readingclub/utils/StringUtil:isNullOrEmpty    (Ljava/lang/String;)Z
    //   4: ifeq +5 -> 9
    //   7: aconst_null
    //   8: areturn
    //   9: aload_0
```

```
    //   10: invokestatic 87    java/lang/Integer:parseInt    (Ljava/lang/String;)I
    //   13: iconst_1
    //   14: if_icmpge +5 -> 19
    //   17: aconst_null
    //   18: areturn
    //   19: getstatic 22    com/huili/readingclub/model/ModelSecurity:userTokens    Ljava/util/HashMap;
    //   22: aload_0
......
```

CFR某函数转换报错但有源码

转换报错：

summary.txt

```
com.huili.readingclub.model.ModelSecurity
---------------------------
getToken(java.lang.String )
  Loose catch block
run()
  Loose catch block
```

代码中有报错：

```
    /*
     * WARNING - Removed back jump from a try to a catch block - possible behaviour change.
     * Loose catch block
     * Enabled aggressive block sorting
     * Enabled unnecessary exception pruning
     * Enabled aggressive exception aggregation
     * Lifted jumps to return sites
     */
    private static String getToken(final String string2) {
        ......
        lock.lock();
        Runnable runnable = new Runnable((String)charSequence){
            ......
            /*
             * WARNING - Removed back jump from a try to a catch block - possible behaviour change.
             * Loose catch block
             * Enabled aggressive block sorting
             * Enabled unnecessary exception pruning
             * Enabled aggressive exception aggregation
             * Lifted jumps to return sites
             */
            @Override
            public void run() {
                Throwable throwable2222;
```

Procyon某函数完美转换无错误：

```
//
// Decompiled by Procyon v0.5.34
//
    private static String getToken(String s) {
        if (StringUtil.isNullOrEmpty(s)) {
            return null;
        }
        ...
        sb.append("AyGt7ohMRlgECZG7kiIU0IUJBII7S#N");
        ModelSecurity.lock.lock();
        try {
            try {
                new Thread(new Runnable() {
```

```
                @Override
                public void run() {
                    ModelSecurity.lock.lock();
                    try {
                        try {
                            final StringBuilder sb = new StringBuilder();
                            ...
                            sb.append("/");
```

Jadx某函数完美转换外，还保持代码变量和结构更合理 -> 能识别常量定义和引用

```
    private static String getToken(final String str) {
        if (StringUtil.isNullOrEmpty(str) || Integer.parseInt(str) < 1) {
            return null;
        }
        ......
        stringBuilder.append(MyConfig.SECRET_KEY);
        final String md5 = StringUtil.md5(stringBuilder.toString());
        lock.lock();
        try {
            new Thread(new Runnable() {
                public void run() {
                    ModelSecurity.lock.lock();
                    try {
                        StringBuilder stringBuilder = new StringBuilder();
                        ...
                        stringBuilder.append(HttpUtilKt.PATHS_SEPARATOR);
```

其中包括 `stringBuilder.append` 的参数是

常量定义 `MyConfig.SECRET_KEY`

而不是之前代码的常量的值：

`"AyGt7ohMR!gECZG7kiIU0IUJBII7S#N"`

`getMD5Str` 的 `char` 的 `list` 转换效果

另外，再去对比：

Jadx导出的代码的逻辑和结构，非常清晰：

char的list很清楚，以及赋值语句：

```
cArr2[i] = cArr[(b >> 4) & 15];
```

也容易看懂

```
    public static final String getMD5Str(String str) {
        char[] cArr = new char[]{'0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'a', 'b', 'c', 'd', 'e', 'f'};
        try {
            byte[] bytes = str.getBytes();
            MessageDigest instance = MessageDigest.getInstance("MD5");
            instance.update(bytes);
            char[] cArr2 = new char[(r1 * 2)];
            int i = 0;
            for (byte b : instance.digest()) {
                int i2 = i + 1;
                cArr2[i] = cArr[(b >> 4) & 15];
                i = i2 + 1;
```

```
            cArr2[i2] = cArr[b & 15];
        }
        return new String(cArr2);
    } catch (Exception unused) {
        return null;
    }
}
```

而不是像：

Procyon（的Luyten）的代码：

```
public static final String getMD5Str(final String s) {
    final char[] array2;
    final char[] array = array2 = new char[16];
    array2[0] = '0';
    array2[1] = '1';
    array2[2] = '2';
    array2[3] = '3';
    array2[4] = '4';
    array2[5] = '5';
    array2[6] = '6';
    array2[7] = '7';
    array2[8] = '8';
    array2[9] = '9';
    array2[10] = 'a';
    array2[11] = 'b';
    array2[12] = 'c';
    array2[13] = 'd';
    array2[14] = 'e';
    array2[15] = 'f';
    try {
        final byte[] bytes = s.getBytes();
        final MessageDigest instance = MessageDigest.getInstance("MD5");
        instance.update(bytes);
        final byte[] digest = instance.digest();
        final int length = digest.length;
        final char[] array3 = new char[length * 2];
        int i = 0;
        int n = 0;
        while (i < length) {
            final byte b = digest[i];
            final int n2 = n + 1;
            array3[n] = array[b >> 4 & 0xF];
            n = n2 + 1;
            array3[n2] = array[b & 0xF];
            ++i;
        }
        return new String(array3);
    }
    catch (Exception ex) {
        return null;
    }
}
```

作为一个char的list，还是没有清晰的表达出来

更不像是CFR的：

```
public static final String getMD5Str(String object) {
    char[] arrc;
    char[] arrc2 = arrc = new char[16];
    arrc2[0] = 48;
    arrc2[1] = 49;
    arrc2[2] = 50;
```

```
        arrc2[3] = 51;
        arrc2[4] = 52;
        arrc2[5] = 53;
        arrc2[6] = 54;
        arrc2[7] = 55;
        arrc2[8] = 56;
        arrc2[9] = 57;
        arrc2[10] = 97;
        arrc2[11] = 98;
        arrc2[12] = 99;
        arrc2[13] = 100;
        arrc2[14] = 101;
        arrc2[15] = 102;
        object = object.getBytes();
        char[] arrc3 = MessageDigest.getInstance("MD5");
        arrc3.update((byte[])object);
        object = arrc3.digest();
        int n2 = ((byte[])object).length;
        arrc3 = new char[n2 * 2];
        int n3 = 0;
        for (int i2 = 0; i2 < n2; ++i2) {
            byte by = object[i2];
            int n4 = n3 + 1;
            arrc3[n3] = arrc[by >> 4 & 15];
            n3 = n4 + 1;
            arrc3[n4] = arrc[by & 15];
        }
        try {
            object = new String(arrc3);
            return object;
        }
        catch (Exception exception) {
            return null;
        }
    }
```

连char的list都不够明显，只是char的int数值。

更更不像是 `JD-GUI` 导出的源码：

```
    public static final String getMD5Str(String paramString)
    {
      char[] arrayOfChar = new char[16];
      char[] tmp6_5 = arrayOfChar;
      tmp6_5[0] = 48;
      char[] tmp12_6 = tmp6_5;
      tmp12_6[1] = 49;
      char[] tmp18_12 = tmp12_6;
      tmp18_12[2] = 50;
      char[] tmp24_18 = tmp18_12;
      tmp24_18[3] = 51;
      char[] tmp30_24 = tmp24_18;
      tmp30_24[4] = 52;
      char[] tmp36_30 = tmp30_24;
      tmp36_30[5] = 53;
      char[] tmp42_36 = tmp36_30;
      tmp42_36[6] = 54;
      char[] tmp49_42 = tmp42_36;
      tmp49_42[7] = 55;
      char[] tmp56_49 = tmp49_42;
      tmp56_49[8] = 56;
      char[] tmp63_56 = tmp56_49;
      tmp63_56[9] = 57;
      char[] tmp70_63 = tmp63_56;
      tmp70_63[10] = 97;
      char[] tmp77_70 = tmp70_63;
      tmp77_70[11] = 98;
      char[] tmp84_77 = tmp77_70;
      tmp84_77[12] = 99;
      char[] tmp91_84 = tmp84_77;
```

```
    tmp91_84[13] = 100;
    char[] tmp98_91 = tmp91_84;
    tmp98_91[14] = 101;
    char[] tmp105_98 = tmp98_91;
    tmp105_98[15] = 102;
    tmp105_98;
    try
    {
      paramString = paramString.getBytes();
      Object localObject = MessageDigest.getInstance("MD5");
      ((MessageDigest)localObject).update(paramString);
      paramString = ((MessageDigest)localObject).digest();
      int i = paramString.length;
      localObject = new char[i * 2];
      int j = 0;
      int k = 0;
      while (j < i)
      {
        int m = paramString[j];
        int n = k + 1;
        localObject[k] = ((char)arrayOfChar[(m >> 4 & 0xF)]);
        k = n + 1;
        localObject[n] = ((char)arrayOfChar[(m & 0xF)]);
        j++;
      }
      paramString = new String((char[])localObject);
      return paramString;
    }
    catch (Exception paramString) {}
    return null;
  }
```

连char的list不仅不够明显，不仅只是char的int数值，而且还有多余的赋值，影响代码逻辑的理解，以及对应的数据赋值：

```
localObject[k] = ((char)arrayOfChar[(m >> 4 & 0xF)]);
```

都很晦涩难懂。

对于几种反编译抓换代码的出错程度和代码逻辑是否完美的结论是

- `JD-GUI`：某函数转换报错得不到源码
- `CFR`：某函数转换报错但有源码
- `Procyon`：某函数完美转换无错误
- `Jadx`：某函数完美转换外还能识别常量定义和代码结构更清晰

`JD-GUI` vs `CFR` vs `Procyon` vs `Jadx` 对比总结

下面从整体总结和对比这几个反编译的反编译的效果：

java反编译器	JD-GUI	CFR	Procyon	Jadx
转换出错程度	比较多	少许	很少	极少
出错状态和相关信息	会显示：`/* Error */ // Byte code`	输出转换期间出错的地方到文件：`summary.txt`	会显示：`This method could not be decompiled. Original Bytecode`	
转换出的代码	不是很好，	部分细节转换的略有瑕疵，但是还	完美转换细节 代码中字符数组定义等内容可以正确转换，但是逻辑不清晰	不仅转换完美无错，而且代码逻辑更准确和完整 代码中的常量

的代码的质量	细节不够好	是可以看到基本代码逻辑的	比如只是能转换出常量解析后的字符串值，而无法识别出是常量的引用	的引用都能完美还原出来代码中字符数组定义等内容可以完美转换
转换出的文件是否有标识	无	有，顶部有标识： Decompiled with CFR 0.141 并且还标出哪些转换出错： Could not load the following classes: android.app.Dialog android.content.Context android.content.res.Resources android.view.View android.view.View$OnClickListener android.widget.TextView	有，顶部有标识： Decompiled by Procyon v0.5.34	无

所以最终结论就是：

- 以后尽量用 `Jadx`
  - 直接用jadx打开未加固的apk
    - 即可查看和导出java源代码
  - 打开（用FDex2从加固了的apk导出的）dex文件
    - 查看和导出java源代码
- 其次考虑用 `Procyon` （或基于Procyon的GUI工具 `Luyten` ）
  - 查看代码：用基于Procyon的 `Luyten` 去查看代码
  - 导出代码：用Procyon去从jar反编译转换出java源代码

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:17:37

常见反编译器

下面整理一些安卓相关的反编译器。

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:17:24

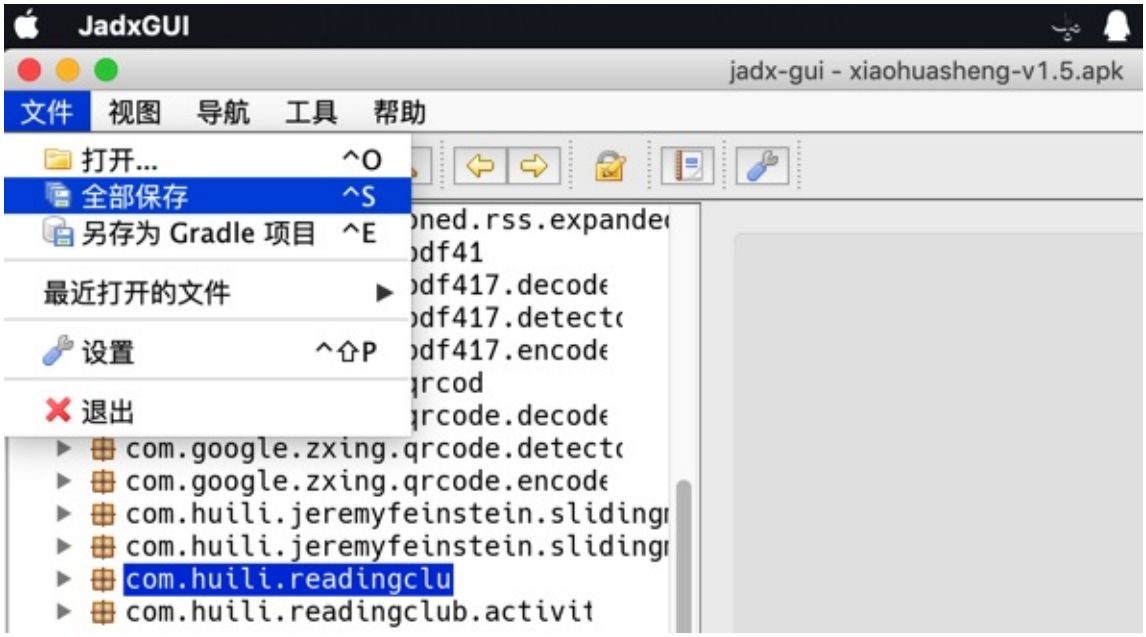
jadx

- 主页
  - skylot/jadx: Dex to Java decompiler
- 功能
  - 从 `dex` 或 `apk` 文件中转换出 `java` 源代码的反编译器
- 两种模式/版本
  - 命令行版本= `command line version` ： `jadx`
  - 图形界面版本= `GUI` = `graphical version` ： `jadx-gui` = `JadxGUI`
    - -> 注意：很多人往往把 `jadx-gui` 简称为 `jadx`
- 截图
  -

`jadx-gui` 使用举例

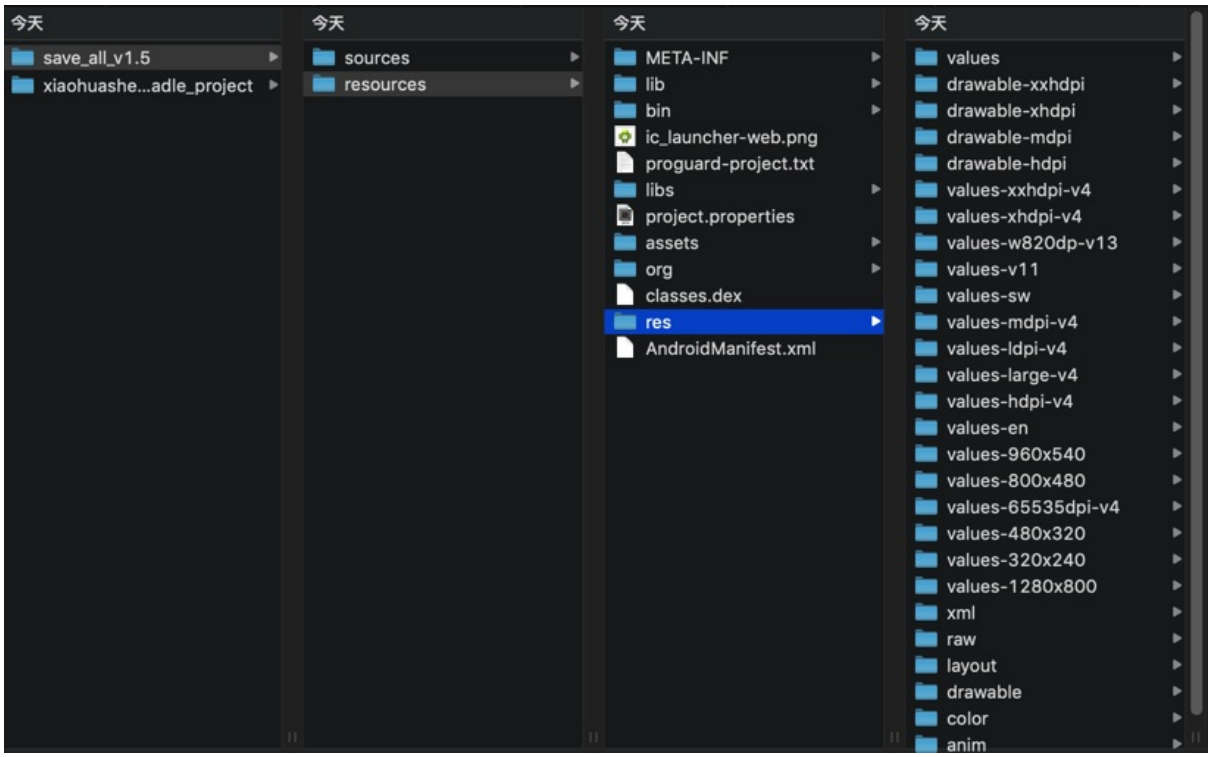
用 `jadx-gui` 导出全部代码

`文件 -> 全部保存`

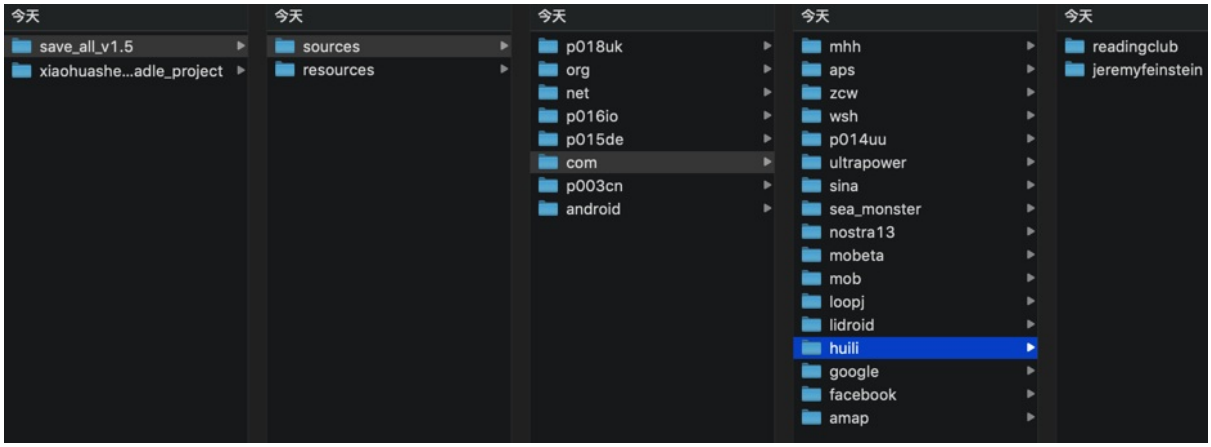


即可下载到：

- 各种资源： `resources`
- 源码： `sources`
  - 其中的 `sources` ，和 `文件 -> 另存为Gradle项目` 所导出的代码是一样的



其中就有我们希望的app的业务逻辑的代码：



从 `jadx-gui` 打开的结构看出是否加固和是哪家的加固

对于某个安卓的apk，用 `jadx-gui` 打开不同版本的apk的效果是：

- v1.5
  -
- v3.4.8

- v3.6.9

。

-》

可以看出apk是否被加固以及用了何种加固方案：

- `v1.5`： 没有被加固
- `v3.4.8`： 加固方案 qihoo奇虎360
- `v3.6.9`： 加固方案 腾讯乐固legu

`jadx` 的help帮助信息=语法参数

```
➜  jadx_cmd_exported /xxx/jadx/jadx-0.9.0/bin/jadx --help

jadx - dex to java decompiler, version: 0.9.0

usage: jadx [options] (input file) (.apk, .dex, .jar or .class)
options:
  -d, --output-dir                    - output directory
  -ds, --output-dir-src               - output directory for sources
  -dr, --output-dir-res               - output directory for resources
  -r, --no-res                        - do not decode resources
  -s, --no-src                        - do not decompile source code
  -e, --export-gradle                 - save as android gradle project
  -j, --threads-count                 - processing threads count (default: 2)
  --show-bad-code                     - show inconsistent code (incorrectly decompiled)
  --no-imports                        - disable use of imports, always write entire package name
  --no-replace-consts                 - don't replace constant value with matching constant field
  --escape-unicode                    - escape non latin characters in strings (with \u)
  --respect-bytecode-access-modifiers - don't change original access modifiers
  --deobf                             - activate deobfuscation
  --deobf-min                         - min length of name, renamed if shorter (default: 3)
  --deobf-max                         - max length of name, renamed if longer (default: 64)
```

```
  --deobf-rewrite-cfg                - force to save deobfuscation map
  --deobf-use-sourcename             - use source file name as class name alias
  --cfg                              - save methods control flow graph to dot file
  --raw-cfg                          - save methods control flow graph (use raw instructions)
  -f, --fallback                     - make simple dump (using goto instead of 'if', 'for', etc)
  -v, --verbose                      - verbose output
  --version                          - print jadx version
  -h, --help                         - print this help
Example:
  jadx -d out classes.dex
```

常见问题

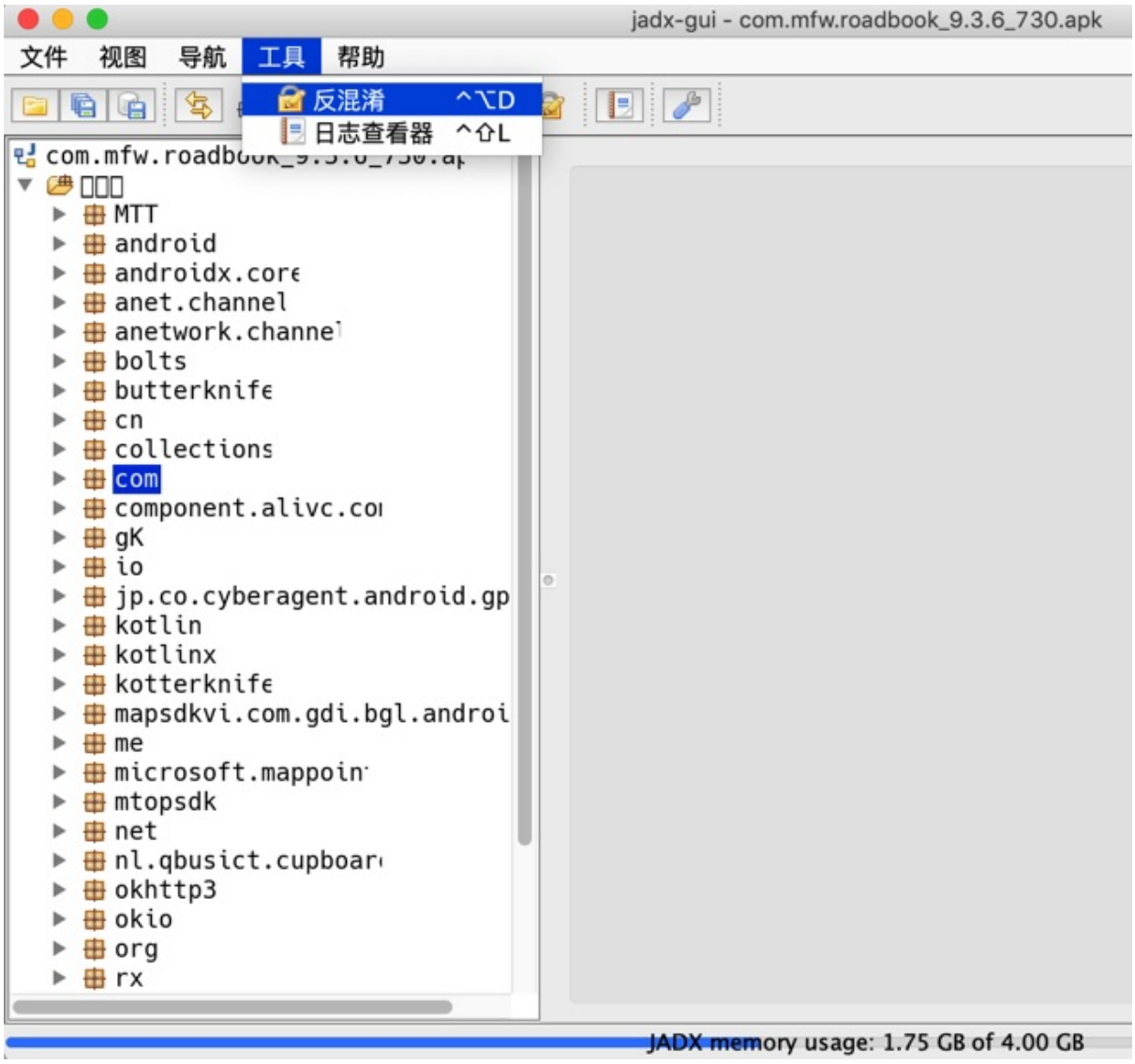
Jadx中如何反混淆 `deobfuscation`

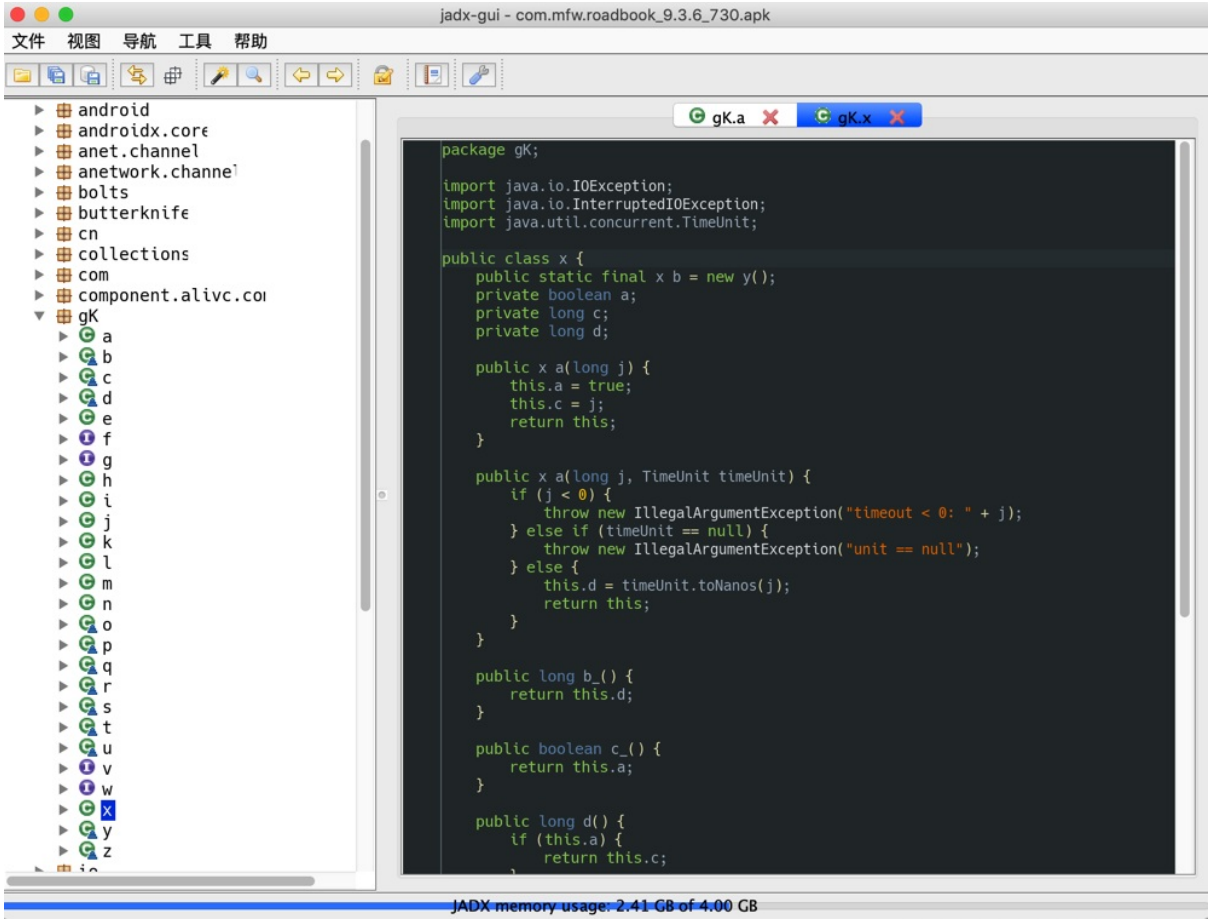
此处暂时没有找到，反混淆前后对比效果明显的例子。

暂时只能随便找了个效果不明显的，用于解释如何开启和关闭反混淆。

如下：

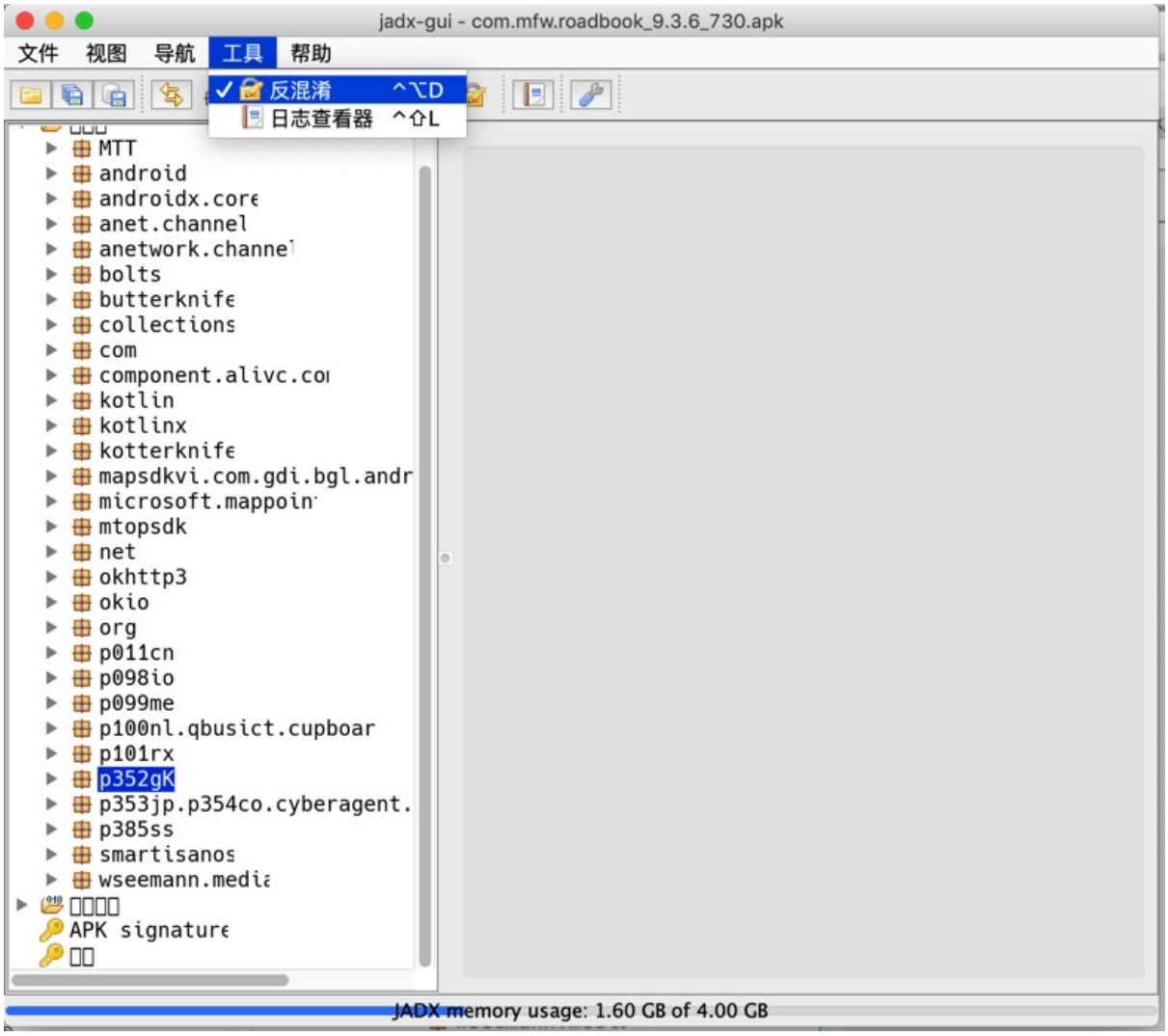
不带反混淆：



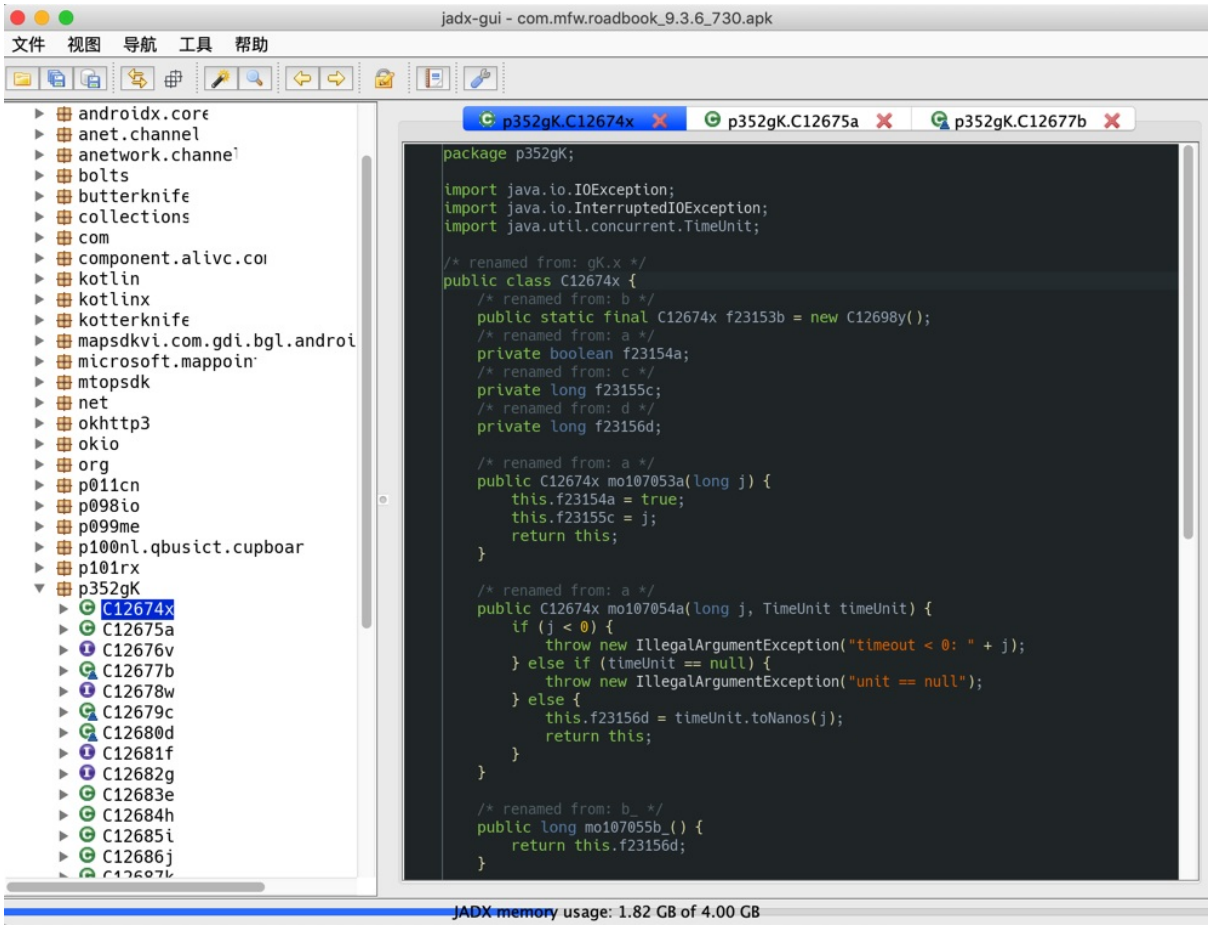


都是a,b,c,d,j,等变量名

启用反混淆：



之前的gK,io等，就反混淆了：



变量名改为了：f23154a，f23155c，虽然反混淆后的效果很一般，但是至少比a,b,c更容易看懂一些。

jadx转换出错：java.lang.OutOfMemoryError

如果用 `jadx` 转换代码期间出错：

```
java.lang.OutOfMemoryError: GC overhead limit exceeded
    at jadx.core.dex.visitors.blocksmaker.BlockProcessor.computeDominators(BlockProcessor.java:189)
    at jadx.core.dex.visitors.blocksmaker.BlockProcessor.processBlocksTree(BlockProcessor.java:52)
    at jadx.core.dex.visitors.blocksmaker.BlockProcessor.visit(BlockProcessor.java:42)
    at jadx.core.dex.visitors.DepthTraversal.visit(DepthTraversal.java:27)
    at jadx.core.dex.visitors.DepthTraversal.lambda$visit$1(DepthTraversal.java:14)
    at jadx.core.dex.visitors.DepthTraversal$$Lambda$10/469590976.accept(Unknown Source)
    at java.util.ArrayList.forEach(ArrayList.java:1249)
    at jadx.core.dex.visitors.DepthTraversal.visit(DepthTraversal.java:14)
    at jadx.core.ProcessClass.process(ProcessClass.java:32)
    at jadx.api.JadxDecompiler.processClass(JadxDecompiler.java:292)
    at jadx.api.JavaClass.decompile(JavaClass.java:62)
    at jadx.api.JadxDecompiler.lambda$appendSourcesSave$0(JadxDecompiler.java:200)
    at jadx.api.JadxDecompiler$$Lambda$13/1425454633.run(Unknown Source)
    at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1142)
    at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:617)
    at java.lang.Thread.run(Thread.java:745)
...
```

且同时伴有：

- CPU占用率很高
- 内存消耗也很大
  - 比如此处JadxCLI占用了4G的内存
    -

就是典型的： `OOM` = `Out Of Memory` 的问题了。

解决办法，有两种：

- 增加JVM最大内存
  - 逻辑：修改 `jadx` 脚本，增大 `-Xmx` 的值
  - 步骤：
    - 编辑 `jadx-0.9.0/bin/jadx` ，找到 `DEFAULT_JVM_OPTS` 的配置，修改其中 `-Xmx` 的值
    - 比如把此处的
    - `DEFAULT_JVM_OPTS='"-Xms128M" "-Xmx4g"'`
    - 改为：
    - `DEFAULT_JVM_OPTS='"-Xms128M" "-Xmx6g"'`
    - 即表示，把JVM最大内存，从之前的 `4G` ，增大到 `6G`
    - 这样就运行 `jadx` 使用更多的内存，从而降低或消除 `OOM` 的问题了
- 减少线程数
  - 逻辑：通过 `-j N` ，N=1/2之类，减少进程数，从而降低内存占用，减少OOM的概率
  - 步骤：
    - 在命令行运行jadx时，传递 `-j` 参数，指定线程数，比如
      - `jadx -d output_folder -j 1 your_apk.apk`
  - 缺点：
    - 处理速度会有所降低
      - 因为默认 `4` 线程处理，反编译等速度会比较快
      - 线程数减少后，反编译等速度可能会有所影响

说明：

- 一般反编译小的不复杂的 `apk` 或 `dex` ，不会遇到 `OOM` 问题
- 反编译比较大型的，比较复杂的 `apk` 或 `dex` ，才可能会遇到 `OOM`
  - 比如之前 用jadx反编译马蜂窝 遇到了 `OOM`

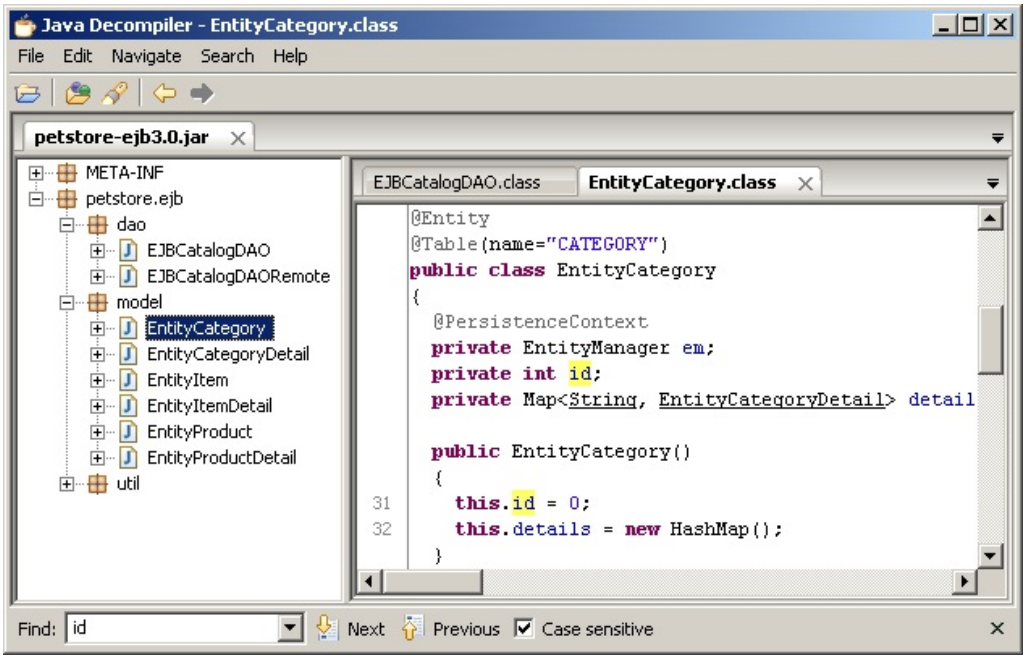
crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:16:02

CFR

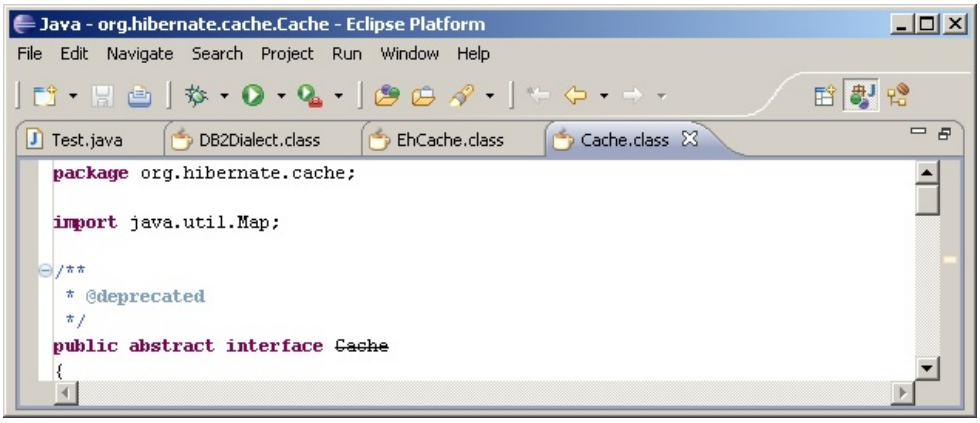
- `CFR` = `Class File Reader`

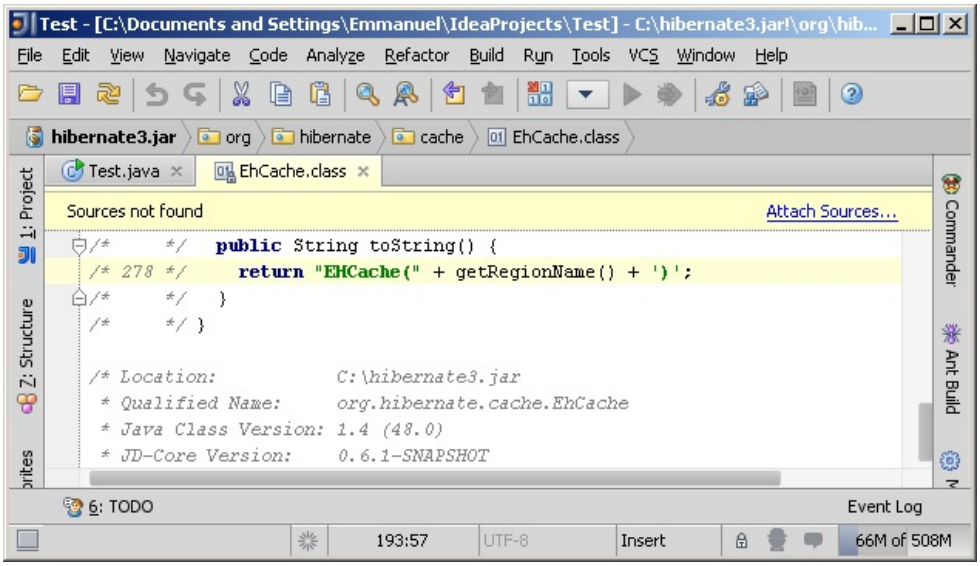
crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:15:32

JD-GUI

- 主页
  - Java Decompiler
- JD Project=Java Decompiler project
  - `JD-Core`
    - 是什么：`一个库`
    - 功能：从一个或多个.class文件中重构java源代码
    - 用途：
      - 可用于恢复丢失的源代码
      - 可用于查看jar包的java源码=查看JRE（Java运行时）库的源码
    - 特点：
      - java 5中最新的功能
        - 注释annotations
        - generics 或 枚举类
    - 说明：
      - JD-GUI内置包含了JD-Core
      - JD-Eclipse内置包含了JD-Core
  - `JD-GUI`
    - 是什么：一个独立的带图形界面的`程序`
    - 作用：显示查看jar包的java源代码
      - 注：`jar包`=内部包含了很多`.class`文件的，被压缩打包成`jar`
    - 举例
      - 

    - github主页
      - java-decompiler/jd-gui: A standalone Java Decompiler GUI
    - 下载
      - http://java-decompiler.github.io
      - -》
      - Releases · java-decompiler/jd-gui
      - -》
      - 比如：

        - `Mac`：
            - jd-gui-osx-1.4.1.tar
    - 插件
        - JD-Eclipse：Eclipse的插件
            - 举例：
            - 

        - JD-IntelliJ：IntelliJ IDEA的插件
            - 举例：
            - 

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:16:38

Procyon

- bitbucket官网
  - mstrobel / Procyon — Bitbucket

Luyten vs Procyon

从jar包导出代码：

- Luyten
  - 带GUI图形界面：可直接查看代码
  - 也可用来Save all导出全部代码
    - 但是速度比较慢
      - 当代码很大时：几十分钟还没导出完毕
- Procyon
  - 不带界面，只是一个jar包：
    - `procyon-decompiler-0.5.34.jar`
  - 可直接用来从你的jar包导出源码
    - 当代码很大时：速度很快，1分钟左右即可搞定

用Procyon从jar导出java源码

下面介绍用Procyon作为命令行工具去导出一个jar包文件为java源代码的过程：

下载Procyon的jar包

比如 procyon-decompiler-0.5.34.jar

用procyon的jar包从你的jar包转换出java代码

语法：

`java -jar /path/to/procyon-decompiler-0.5.34.jar -jar your_to_decompile.jar -o outputFolderName`

举例：

```
java -jar  /Users/crifan/dev/dev_tool/android/reverse_engineering/Procyon/procyon-decompiler-0.5.34.jar -jar com.huili.rea
dingclub8825612-dex2jar.jar -o com.huili.readingclub8825612

java -jar  /Users/crifan/dev/dev_tool/android/reverse_engineering/Procyon/procyon-decompiler-0.5.36.jar -jar ../../dex_to_
jar/com.ishowedu.child.peiyin8392664-dex2jar.jar -o com.ishowedu.child.peiyin8392664_java
```

基于 `Procyon` 的 `Luyten` 中的菜单中参数是来自 `procyon-decompiler` 命令行的参数

从 `procyon-decompiler` 的help帮助信息是：

```
procyon-decompiler -?
Usage:  main class [options] type names or class/jar files
  Options:
```

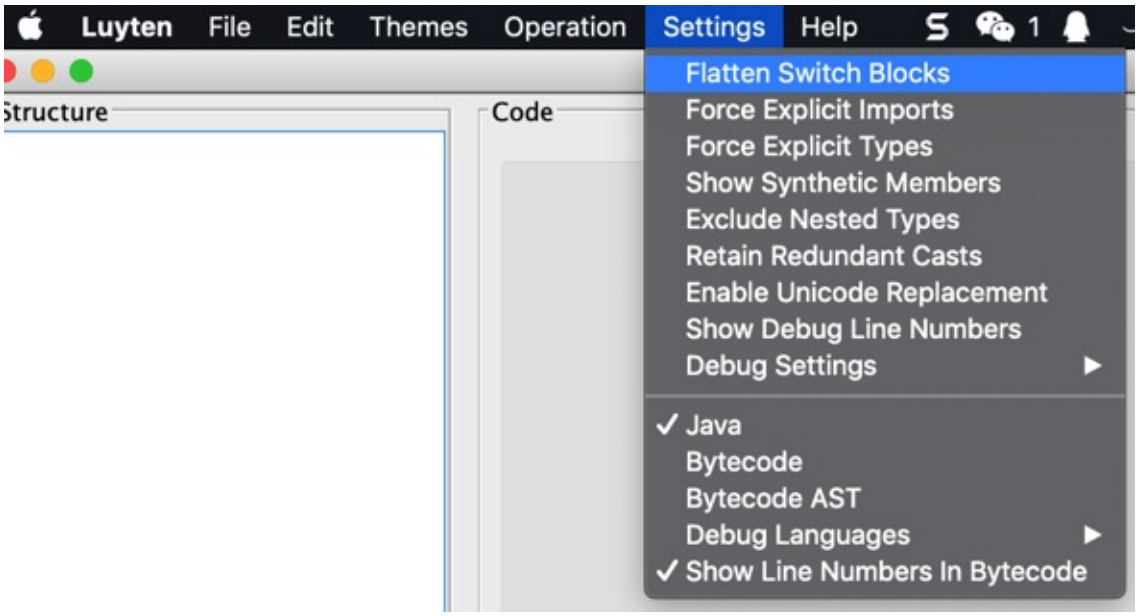
```
-b, --bytecode-ast
   Output Bytecode AST instead of Java.
   Default: false
-ci, --collapse-imports
   Collapse multiple imports from the same package into a single wildcard
   import.
   Default: false
-cp, --constant-pool
   Includes the constant pool when displaying raw bytecode (unnecessary with
   -v).
   Default: false
-dl, --debug-line-numbers
   For debugging, show Java line numbers as inline comments (implies -ln;
   requires -o).
   Default: false
    --disable-foreach
   Disable 'for each' loop transforms.
   Default: false
-eml, --eager-method-loading
   Enable eager loading of method bodies (may speed up decompilation of
   larger archives).
   Default: false
-ent, --exclude-nested
   Exclude nested types when decompiling their enclosing types.
   Default: false
-ei, --explicit-imports
   [DEPRECATED] Explicit imports are now enabled by default.  This option
   will be removed in a future release.
   Default: false
-eta, --explicit-type-arguments
   Always print type arguments to generic methods.
   Default: false
-fsb, --flatten-switch-blocks
   Drop the braces statements around switch sections when possible.
   Default: false
-fq, --force-qualified-references
   Force fully qualified type and member references in Java output.
   Default: false
-?, --help
   Display this usage information and exit.
   Default: false
-jar, --jar-file
   [DEPRECATED] Decompile all classes in the specified jar file (disables
   -ent and -s).
-lc, --light
   Use a color scheme designed for consoles with light background colors.
   Default: false
-lv, --local-variables
   Includes the local variable tables when displaying raw bytecode
   (unnecessary with -v).
   Default: false
-ll, --log-level
   Set the level of log verbosity (0-3).  Level 0 disables logging.
   Default: 0
-mv, --merge-variables
   Attempt to merge as many variables as possible.  This may lead to fewer
   declarations, but at the expense of inlining and useful naming.  This feature is
   experimental and may be removed or become the standard behavior in future releases.
   Default: false
-o, --output-directory
   Write decompiled results to specified directory instead of the console.
-r, --raw-bytecode
   Output Raw Bytecode instead of Java (to control the level of detail, see:
   -cp, -lv, -ta, -v).
   Default: false
-ec, --retain-explicit-casts
   Do not remove redundant explicit casts.
   Default: false
-ps, --retain-pointless-switches
   Do not lift the contents of switches having only a default label.
   Default: false
-ss, --show-synthetic
   Show synthetic (compiler-generated) members.
   Default: false
```

```
    -sm, --simplify-member-references
       Simplify type-qualified member references in Java output [EXPERIMENTAL].
       Default: false
    -sl, --stretch-lines
       Stretch Java lines to match original line numbers (only in combination
       with -o) [EXPERIMENTAL].
       Default: false
    -ta, --type-attributes
       Includes type attributes when displaying raw bytecode (unnecessary with
       -v).
       Default: false
        --unicode
       Enable Unicode output (printable non-ASCII characters will not be
       escaped).
       Default: false
    -u, --unoptimized
       Show unoptimized code (only in combination with -b).
       Default: false
    -v, --verbose
       Includes more detailed output depending on the output language (currently
       only supported for raw bytecode).
       Default: false
        --version
       Display the decompiler version and exit.
       Default: false
    -ln, --with-line-numbers
       Include line numbers in raw bytecode mode; supports Java mode with -o
       only.
       Default: false
```

从这些支持的参数中，可以推断：

基于 `Procyon` 的 `Luyten` 中的菜单中的选项，都是对应着这些参数的。

比如： `Settings -> Flatten Switch Blocks`



对应着此处参数 `--flatten-switch-blocks` ：

```
    -fsb, --flatten-switch-blocks
       Drop the braces statements around switch sections when possible.
       Default: false
```

其他以此类推，都是一样的逻辑：把命令行工具的参数，用菜单选项的形式展示和支持出来了。

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2020-01-16

22:30:19

Luyten

- 主页
  - deathmarine/Luyten: An Open Source Java Decompiler Gui for Procyon
- 功能
  - Java Decompiler Gui for Procyon
  - 基于Procyon的带图形界面的Java反编译器
- 界面
  -
- 下载
  - Release Luyten v0.5.4 · deathmarine/Luyten

Luyten的语法高亮的不同主题的效果

- 默认：Default = Default-Alt

-
- Dark

- Eclipse

-
- Visual Studio

- IntelliJ

。

下载和使用Luyten去查看和导出java代码

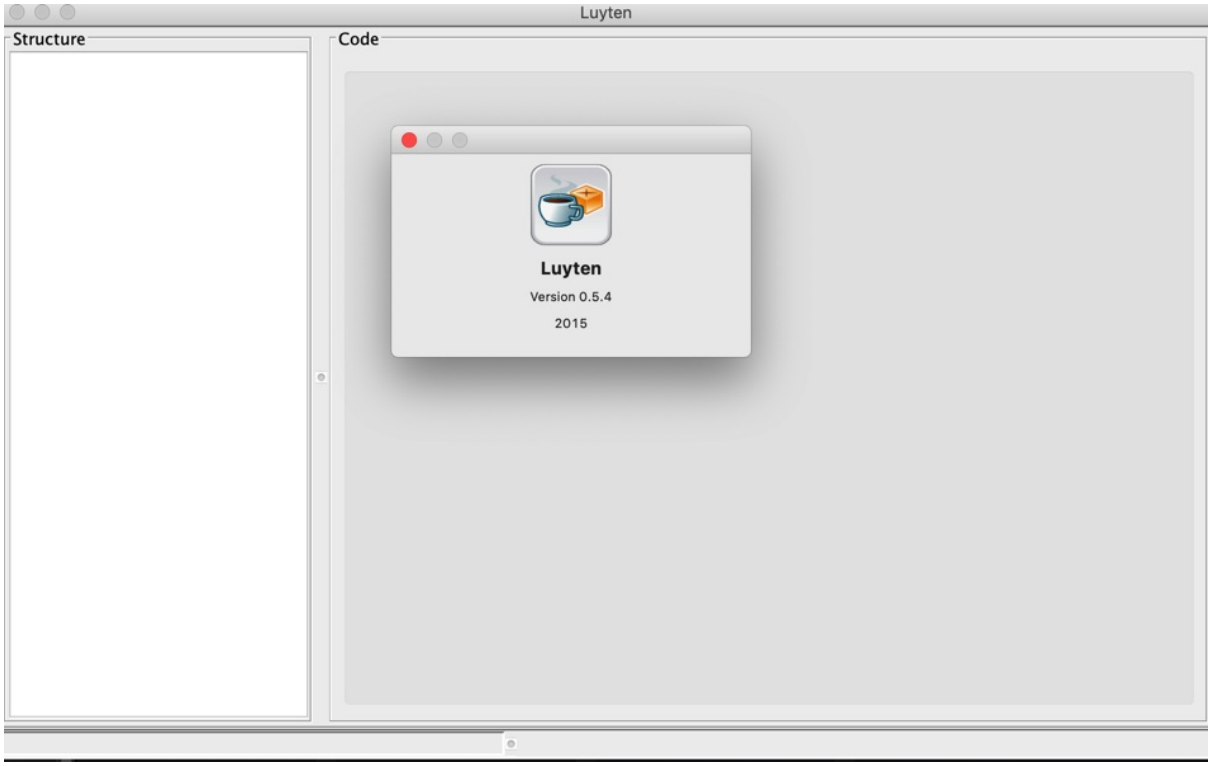
去Releases · deathmarine/Luyten下载最新版本，比如Mac版的：

luyten-OSX-0.5.4.zip

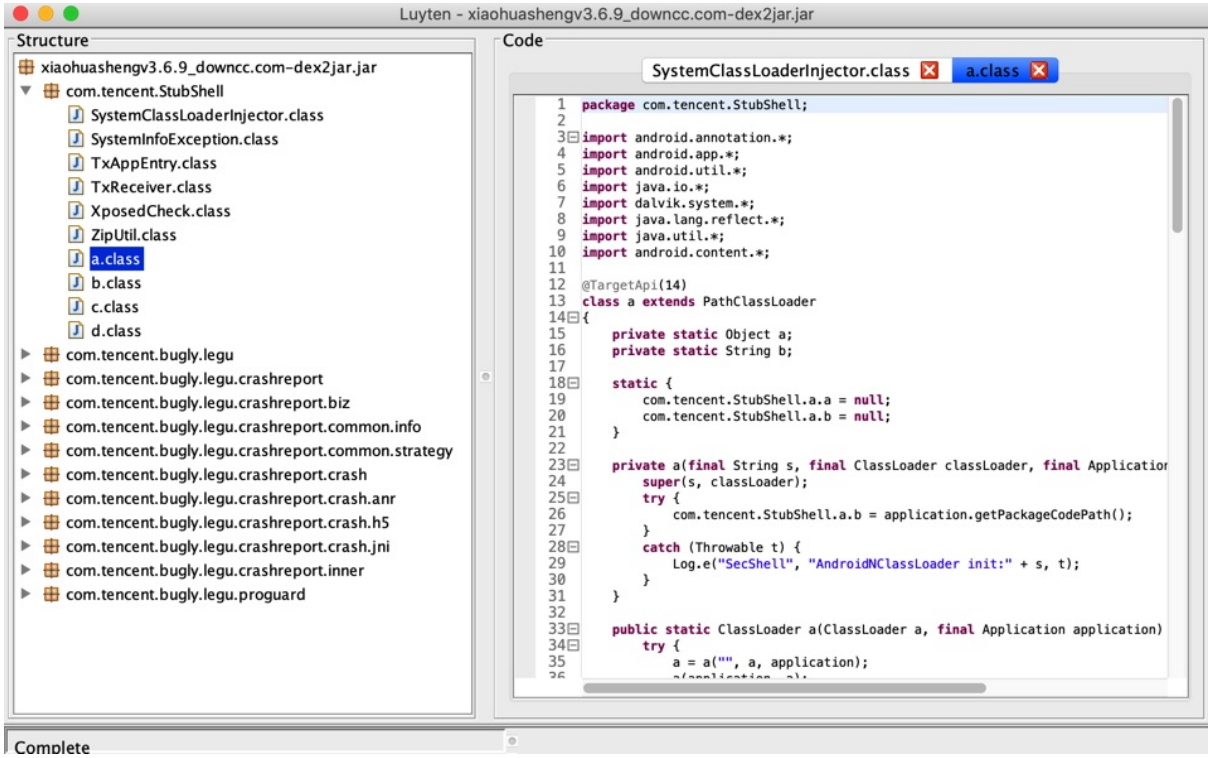
解压得到 `Luyten.app` ：

luyten-OSX-0.5.4.zip
Luyten.app
Luyten.app
应用程序 – 4.6 MB
标签 添加标签...
创建时间 2019年2月5日 星期二 15:48
修改时间 2019年2月5日 星期二 15:48
版本 0.5.4

运行后：

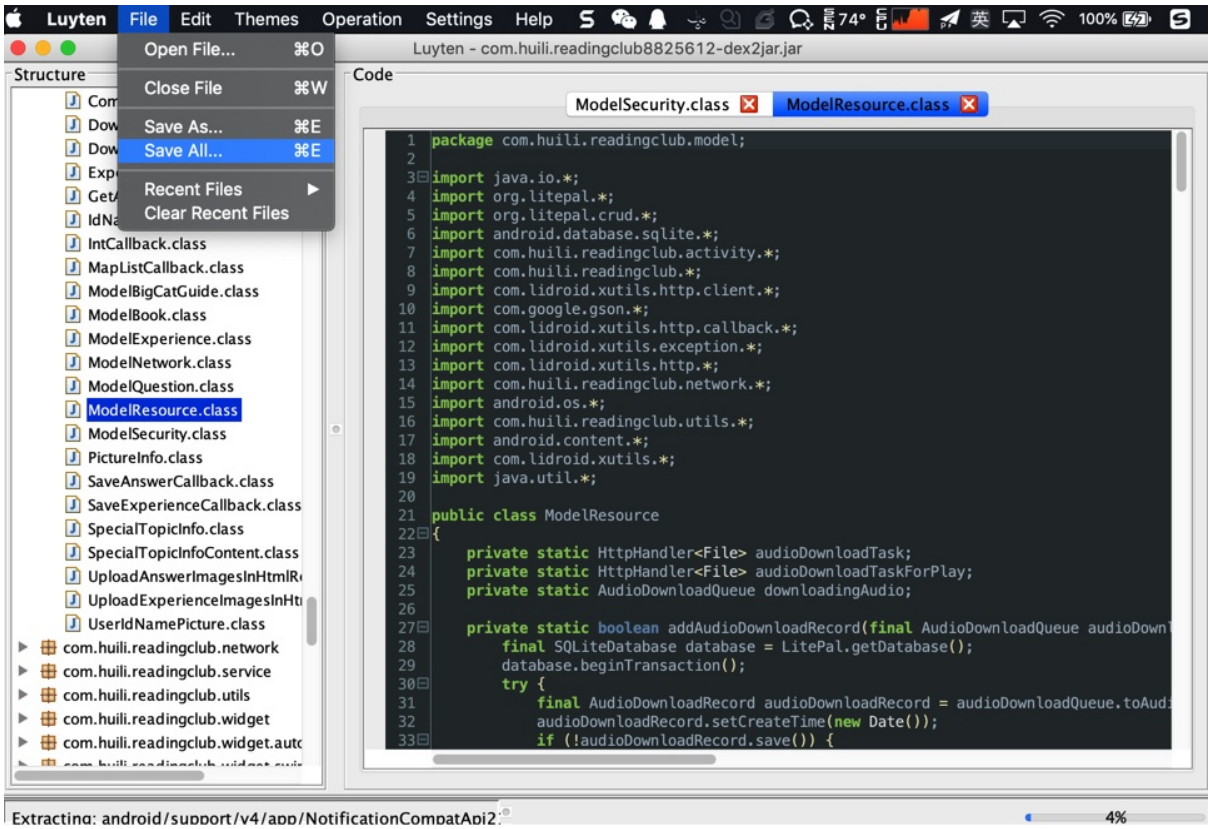

Luyten
Structure
Code
Luyten
Version 0.5.4
2015

然后去把 `jar` 包拖进来即可看到代码：



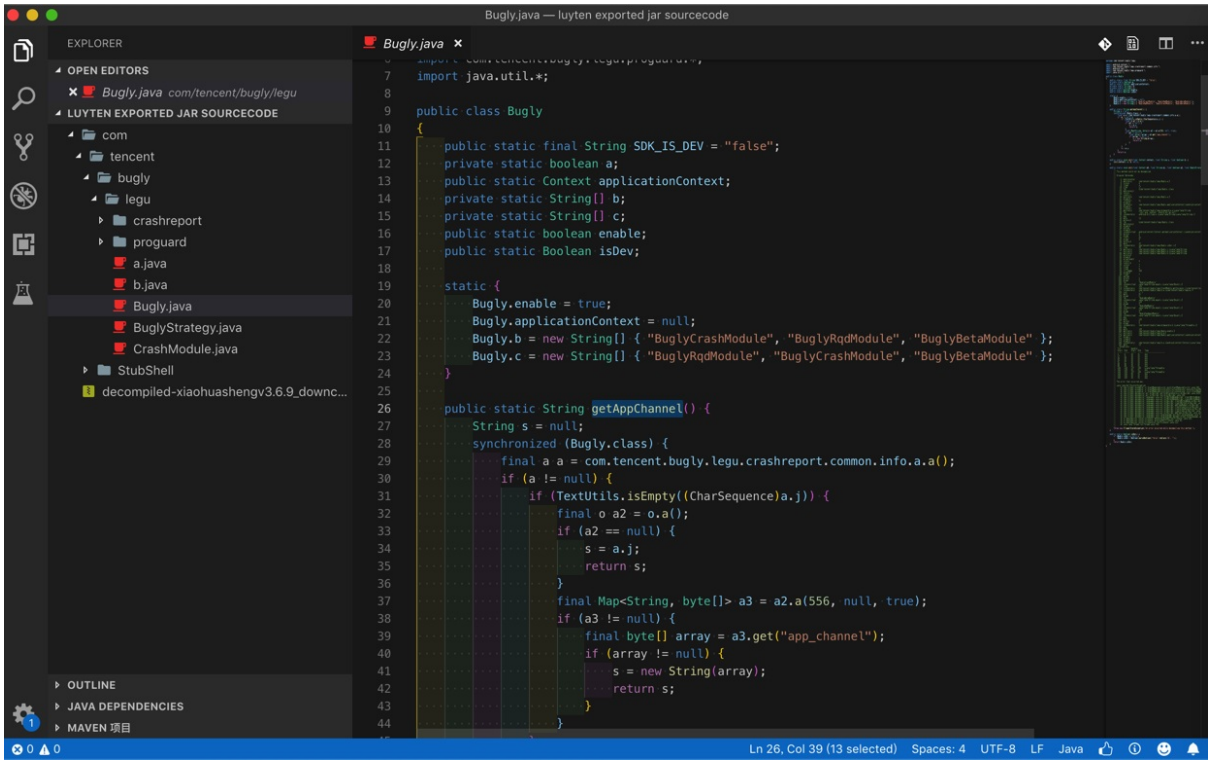
导出所有代码的步骤：

`File -> Save All -> decompiled-xxx.zip`

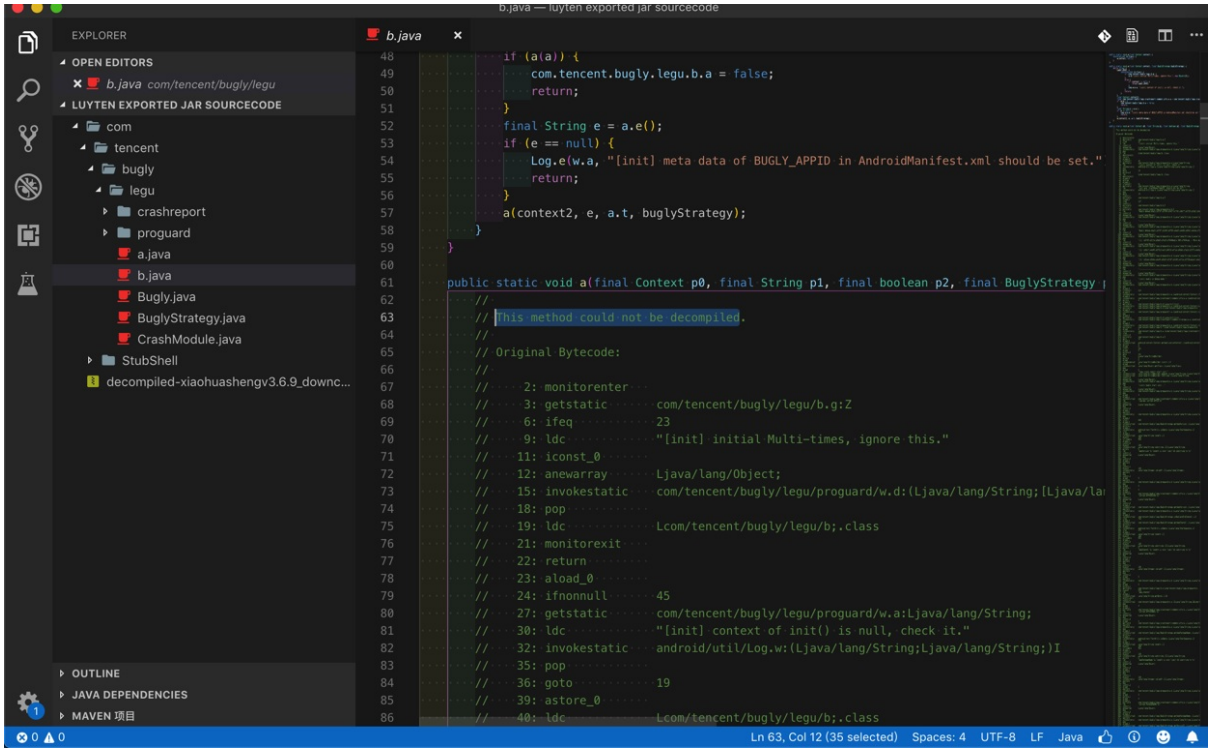


解压后，即可用VSCode等工具方便查看代码了。

之前 `JD-GUI` 、 `CFR` 等工具转换出错的代码，此处是可以正确解析的：



但是也还是有部分代码无法正确解析：



crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2020-01-16 21:52:05

Krakatau

- 主页
  - Storyyeller/Krakatau: Java decompiler, assembler, and disassembler
- 功能
  - java的反编译器、汇编器、反汇编器
    - 如果要java反编译：则需要安装Java JDK
    - 如果不需要java反编译，只是汇编和反汇编，则可以不用安装Java JDK
      - 但是为了更好的测试结果，最好安装Java JDK
- 特点
  - 全部是 `Python` 写的
  - 支持 `Java 10`

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:17:08

Fernflower

- 主页
  - 官网
    - intellij-community/plugins/java-decompiler/engine at master · JetBrains/intellij-community
  - 非官方 GitHub
    - fesh0r/fernflower: Unofficial mirror of FernFlower Java decompiler (All pulls should be submitted upstream)
- 功能
  - Java的反编译器
    - 从Java的 `class` 反编译出java `源代码`
- 说明
  - IntelliJ IDEA 内置
- 用法
  - `java -jar fernflower.jar [-<option>=<value>]* [<source>]+ <destination>`
  - 举例
    - `java -jar fernflower.jar -hes=0 -hdc=0 c:\Temp\binary\ -e=c:\Java\rt.jar c:\Temp\source\`
    - `java -jar fernflower.jar -dgs=1 c:\Temp\binary\library.jar c:\Temp\binary\Boot.class c:\Temp\source\`

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:15:41

GDA

- `GDA` = `GJoy Dex Analysizer` =GDA反编译器
- 主页
  - GDA主页-功能强大的交互式Android反编译分析工具
- 下载
  - GDA下载 最新版
- 功能
  - 一款强大而轻便的交互式反编译器，也是一款综合性逆向分析利器
    - 支持分析apk，dex，odex，oat类型文件
    - 支持python脚本以及方法签名制作与识别
  - 工具包含三个由作者独立完成的高速解析引擎
    - 反编译引擎
    - apk壳检测引擎
    - 恶意行为检测引擎
  - 再加上作者独创的使用了字节码直接转java伪代码的解析方式
    - 无需转换成smali汇编后在做反编译
      - 大大提升了解析速度
- 特点
  - 无需安装java环境和android环境就可以使用
    - 不是依赖其他的（很多安卓反汇编的、java的）库
  - 分析速度快、体积小、内存占用少
- 截图
  -

-

-

- GDA vs JEB

。

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:15:44

Dare

- 主页
  - Dare Homepage
- 下载
  - Dare downloads
- 功能
  - 将 `apk` 文件反编译为Java的 `class` 文件
- 用法
  - `dare -d apkoutput WeChat_462.apk`
- 截图
  - 输出

```
for (localObject = "unknown error"; ; localObject = "null param or 0 length")
  while (true)
  {
    this.eLD = ((String)localObject);
    localObject = new java/lang/StringBuilder;
    String str1 = "errorCode: ";
    ((StringBuilder)localObject).<init>(str1);
    int i = this.dUB;
    localObject = ((StringBuilder)localObject).append(i);
    String str2 = "\t msg: ";
    localObject = ((StringBuilder)localObject).append(str2);
    str2 = this.eLD;
```

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:15:37

JAD

- `JAD` = `the fast JAva Decompiler`
- 主页
  - Home Page of Jad - the fast Java decompiler
    - 作者个人主页？：KPD Home Page
- 下载
  - Download Jad
  - JAD Java Decompiler Download Mirror
- 说明
  - 比较老旧的反编译器，已不再更新和维护
    - 不建议继续使用
      - 换用最新的，比如 `jadx`

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:15:49

其他破解类工具

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:21:36

apktool

- `apktool` 是啥：一个用来逆向工程第三方的二进制的安卓app的工具
  - 即：逆向apk的工具
    - 输入：`apk` 文件
    - 输出：各种安卓 `资源文件`
  - 还可以用来：
    - 构建 `自动化` 工作 -> 取代重复的手动工作
      - 比如自动编译apk等工作
    - 给原有apk `添加额外东西`
      - 比如反编译出apk后
        - 再本地化localizing
        - 添加其他功能
    - `分析` 原有apk的功能和逻辑
- 如何安装
  - Apktool - How to Install
- github主页
  - iBotPeaches/Apktool: A tool for reverse engineering Android apk files
- 官网
  - Apktool - A tool for reverse engineering 3rd party, closed, binary Android apps
  - 官方提示
    - 请不要用apktool用于盗窃破解apk

举例

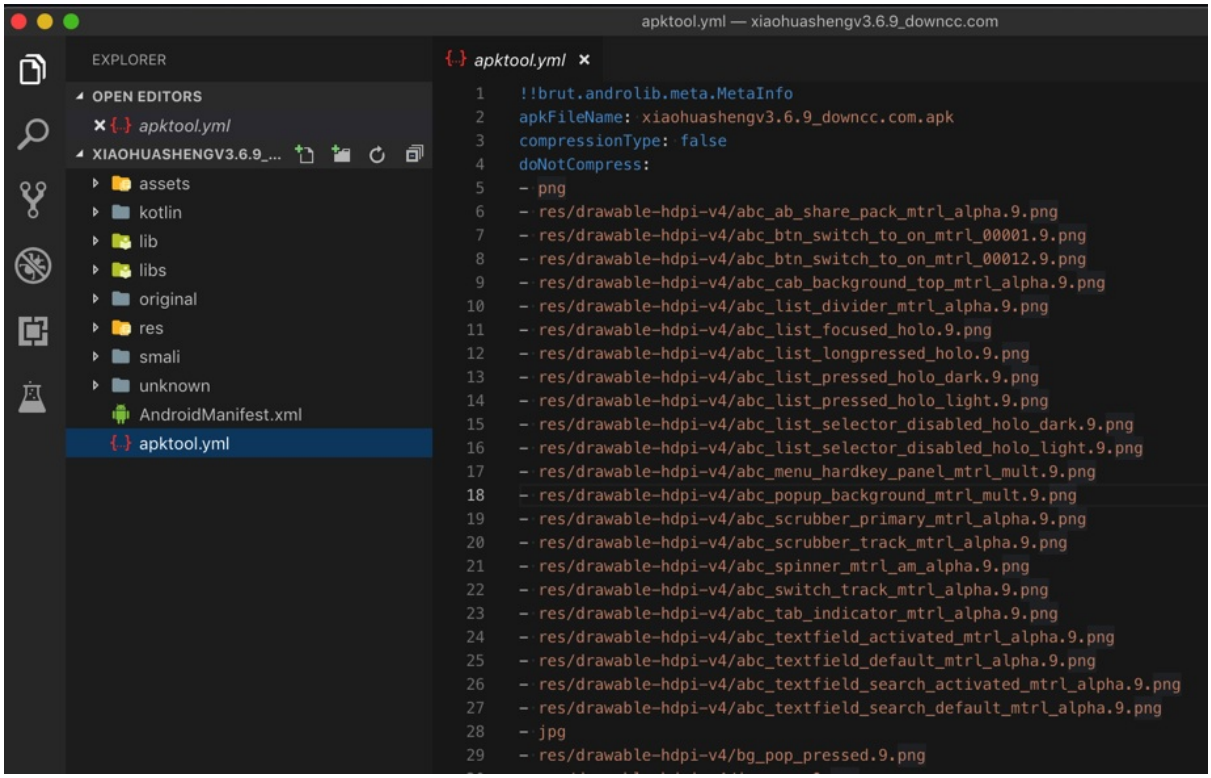
如何用apktool反编译某apk

下面举例说明，如何用apktool去反编译apk：

```
➜  apk ll
total 51280
-rw-r--r--@ 1 crifan  staff    25M  3 14 09:00 xiaohuashengv3.6.9_downcc.com.apk
➜  apk apktool d xiaohuashengv3.6.9_downcc.com.apk
I: Using Apktool 2.4.0 on xiaohuashengv3.6.9_downcc.com.apk
I: Loading resource table...
I: Decoding AndroidManifest.xml with resources...
S: WARNING: Could not write to (/Users/crifan/Library/apktool/framework), using /var/folders/46/2hjxz38n22n3ypp_5f6_p__000
00gn/T/ instead...
S: Please be aware this is a volatile directory and frameworks could go missing, please utilize --frame-path if the defaul
t storage directory is unavailable
I: Loading resource table from file: /var/folders/46/2hjxz38n22n3ypp_5f6_p__00000gn/T/1.apk
I: Regular manifest package...
I: Decoding file-resources...
I: Decoding values */* XMLs...
I: Baksmaling classes.dex...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...
➜  apk ll
total 51280
drwxr-xr-x  12 crifan  staff   384B  3 14 13:39 xiaohuashengv3.6.9_downcc.com
-rw-r--r--@  1 crifan  staff    25M  3 14 09:00 xiaohuashengv3.6.9_downcc.com.apk
➜  apk cd xiaohuashengv3.6.9_downcc.com
➜  xiaohuashengv3.6.9_downcc.com ll
total 160
-rw-r--r--    1 crifan  staff    63K  3 14 13:39 AndroidManifest.xml
-rw-r--r--    1 crifan  staff    14K  3 14 13:39 apktool.yml
```

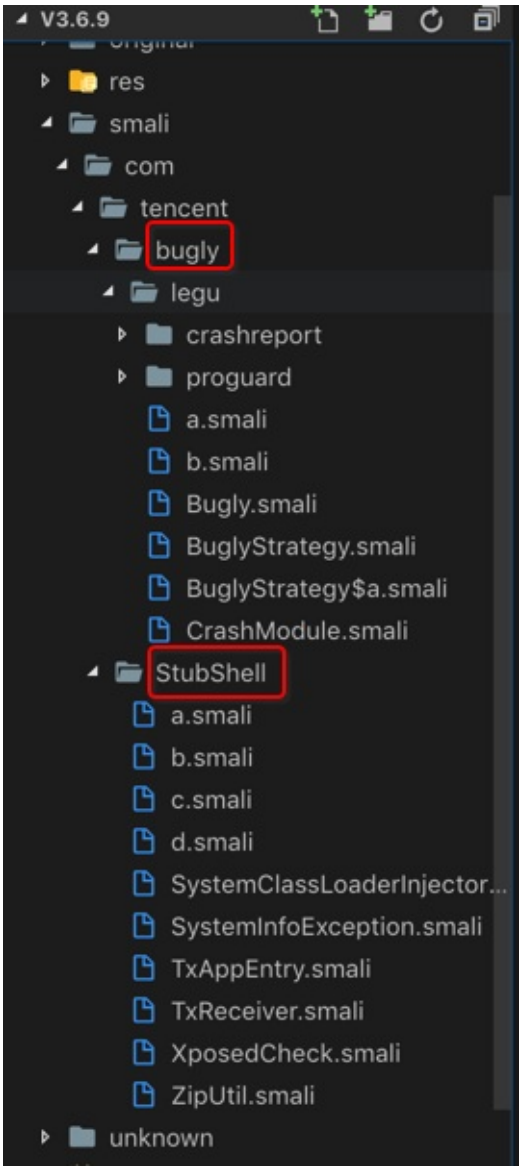
```
drwxr-xr-x   10 crifan  staff   320B  3 14 13:39 assets
drwxr-xr-x    8 crifan  staff   256B  3 14 13:39 kotlin
drwxr-xr-x    9 crifan  staff   288B  3 14 13:39 lib
drwxr-xr-x    3 crifan  staff    96B  3 14 13:39 libs
drwxr-xr-x    4 crifan  staff   128B  3 14 13:39 original
drwxr-xr-x  143 crifan  staff   4.5K  3 14 13:39 res
drwxr-xr-x    3 crifan  staff    96B  3 14 13:39 smali
drwxr-xr-x   10 crifan  staff   320B  3 14 13:39 unknown
```

得到项目目录文件：



其中有：

- 最基本的：`AndroidMenifest.xml`
  -
- 得不到我们要的 `dex` 文件
- 可得到：和app业务逻辑相关代码的smali文件
  - 想要得到最终java源码的话
    - 需要再去找 `smali转java` 的工具才可以

        - 此处即使不去转换得到java源码
            - 也可以从smali文件的目录结构和文件名，大概能看出app内部的类/文件名了
    - 而用apktool转换apk得到smali源码，是有前提的：apk没有加固
        - 加固了的apk反编译后只能看到被加固后的目录结构，看不到app业务逻辑代码和结构
            - 举例：某个被腾讯乐固legu加了密的apk，反编译后看不到原始类名和目录结构，只能看到腾讯乐固的目录结构
            - 

用apktool反编译安卓版微信6.6.7

后来去从这里下载到 `安卓版微信v6.6.7` ，然后去用 `apktool` 去反编译：

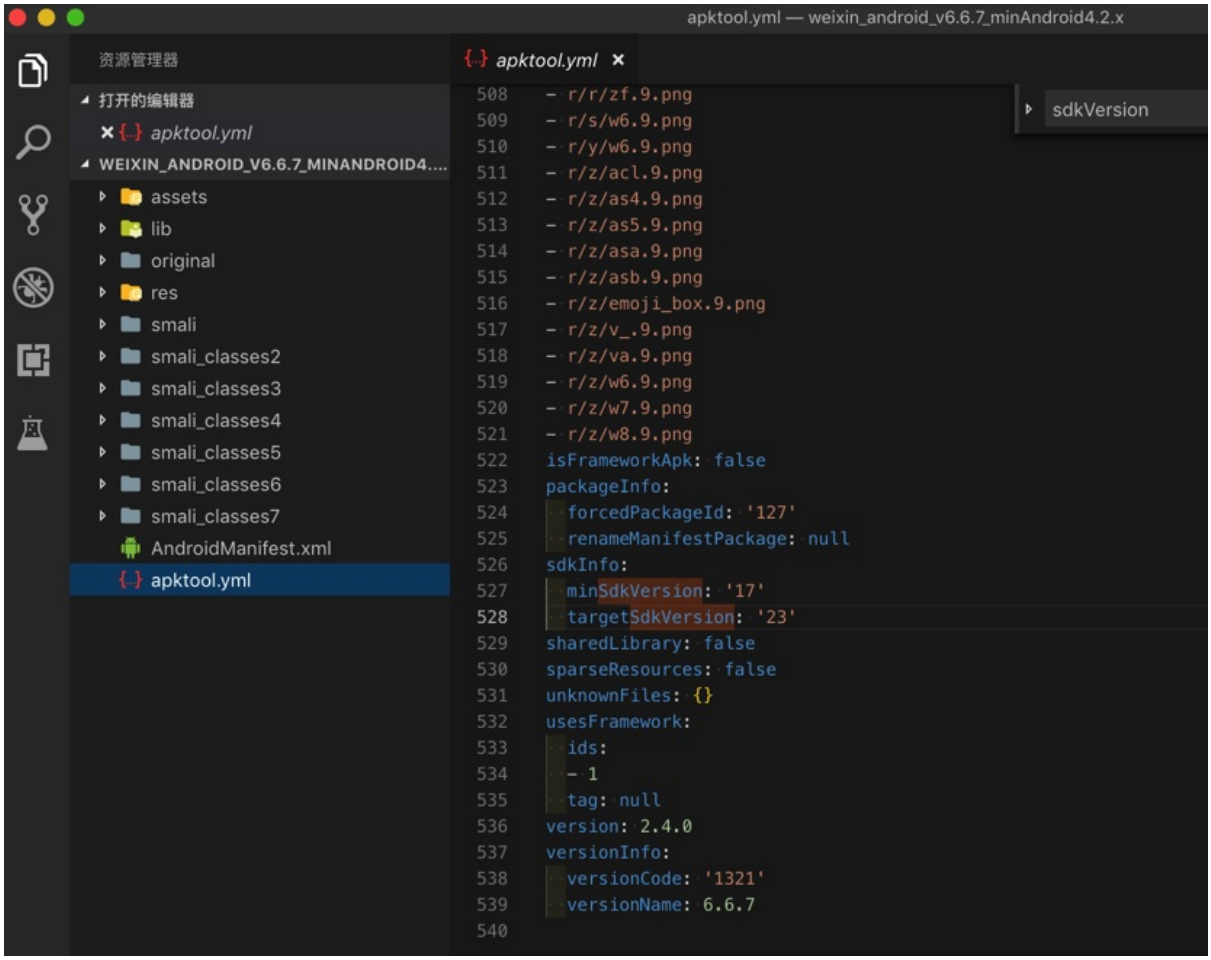
```
➜  apktool ./apktool d /Users/crifan/dev/dev_tool/android/apk/weixin/weixin_android_v6.6.7_minAndroid4.2.x.apk
I: Using Apktool 2.4.0 on weixin_android_v6.6.7_minAndroid4.2.x.apk
I: Loading resource table...
I: Decoding AndroidManifest.xml with resources...
S: WARNING: Could not write to (/Users/crifan/Library/apktool/framework), using /var/folders/46/2hjxz38n22n3ypp_5f6_p__000
00gn/T/ instead...
S: Please be aware this is a volatile directory and frameworks could go missing, please utilize --frame-path if the defaul
t storage directory is unavailable
I: Loading resource table from file: /var/folders/46/2hjxz38n22n3ypp_5f6_p__00000gn/T/1.apk
I: Regular manifest package...
I: Decoding file-resources...
```

```
I: Decoding values */* XMLs...
I: Baksmaling classes.dex...
I: Baksmaling classes2.dex...
I: Baksmaling classes3.dex...
I: Baksmaling classes4.dex...
I: Baksmaling classes5.dex...
I: Baksmaling classes6.dex...
I: Baksmaling classes7.dex...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...
➜  apktool ll
total 65480
-rwxr-xr-x@  1 crifan  staff   2.3K  3 14 11:26 apktool
-rw-r--r--@  1 crifan  staff    16M  3 14 11:29 apktool.jar
-rw-r--r--@  1 crifan  staff    16M  3 14 11:29 apktool_2.4.0.jar
drwxr-xr-x  15 crifan  staff   480B  4 30 17:35 weixin_android_v6.6.7_minAndroid4.2.x
```

然后去看看输出的信息：

```
➜  apktool cd weixin_android_v6.6.7_minAndroid4.2.x
➜  weixin_android_v6.6.7_minAndroid4.2.x ll
total 440
-rw-r--r--    1 crifan  staff   202K  4 30 17:35 AndroidManifest.xml
-rw-r--r--    1 crifan  staff   8.5K  4 30 17:35 apktool.yml
drwxr-xr-x   78 crifan  staff   2.4K  4 30 17:35 assets
drwxr-xr-x    3 crifan  staff    96B  4 30 17:35 lib
drwxr-xr-x    4 crifan  staff   128B  4 30 17:35 original
drwxr-xr-x  118 crifan  staff   3.7K  4 30 17:35 res
drwxr-xr-x    8 crifan  staff   256B  4 30 17:35 smali
drwxr-xr-x    8 crifan  staff   256B  4 30 17:35 smali_classes2
drwxr-xr-x   10 crifan  staff   320B  4 30 17:35 smali_classes3
drwxr-xr-x   12 crifan  staff   384B  4 30 17:35 smali_classes4
drwxr-xr-x   10 crifan  staff   320B  4 30 17:35 smali_classes5
drwxr-xr-x   12 crifan  staff   384B  4 30 17:35 smali_classes6
drwxr-xr-x    8 crifan  staff   256B  4 30 17:35 smali_classes7
➜  weixin_android_v6.6.7_minAndroid4.2.x cat AndroidManifest.xml
➜  weixin_android_v6.6.7_minAndroid4.2.x cat apktool.yml
!! brut.androlib.meta.MetaInfo
apkFileName: weixin_android_v6.6.7_minAndroid4.2.x.apk
compressionType: false
doNotCompress:
- arsc
- png
- sec
- conf
- dat
- txt
- data
- assets/infowindow_bg.9.png
- m4a
- wav
- assets/xwalk-command-line
- jpg
- mp3
- assets/wxa_library/local/IGNORE
- r/a/w6.9.png
- apk
- r/a7/v3.9.png
...
- r/q/aa2.9.png
- r/q/emoji_grid_item_bottom.9.png
- r/q/emoji_grid_item_left.9.png
- r/q/emoji_grid_item_middle.9.png
- gif
...
- r/r/ark.9.png
- r/r/emoji_app_msg_mask.9.png
- r/r/emoji_bottombar_bg.9.png
- r/r/emoji_box.9.png
- r/r/emoji_grid_item_fg_normal.9.png
- r/r/emoji_grid_item_fg_pressed.9.png
```

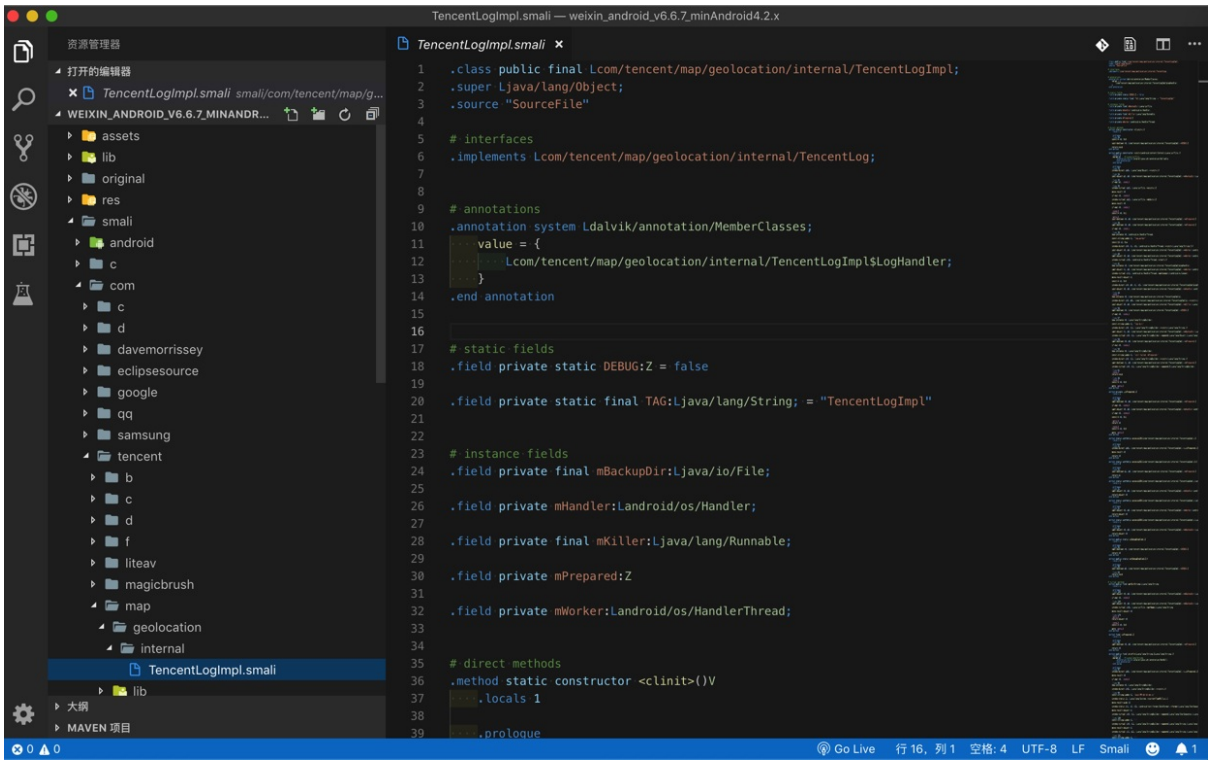
```
- r/r/tenpay_keybg.9.png
- r/r/tenpay_keyitem_bottom.9.png
- r/r/v3.9.png
...
- r/z/w8.9.png
isFrameworkApk: false
packageInfo:
  forcedPackageId: '127'
  renameManifestPackage: null
sdkInfo:
  minSdkVersion: '17'
  targetSdkVersion: '23'
sharedLibrary: false
sparseResources: false
unknownFiles: {}
usesFramework:
  ids:
  - 1
  tag: null
version: 2.4.0
versionInfo:
  versionCode: '1321'
  versionName: 6.6.7
```



可以看出一些版本方面的信息：

- `minSdkVersion: '17'` ： 最低安卓版本 `17`
  - 17对应：安卓 `4.2` ， `4.2.2`
- `targetSdkVersion: '23'` ： 目标安卓版本 `23`
  - 23对应：安卓 `6.0`
- `versionName: 6.6.7` ： 微信版本号 `6.6.7`

以及有很多的smali代码：



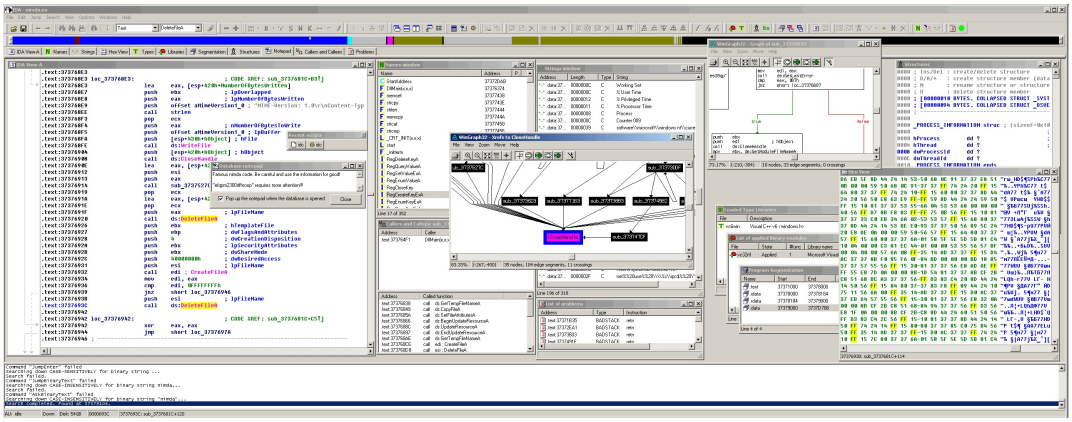
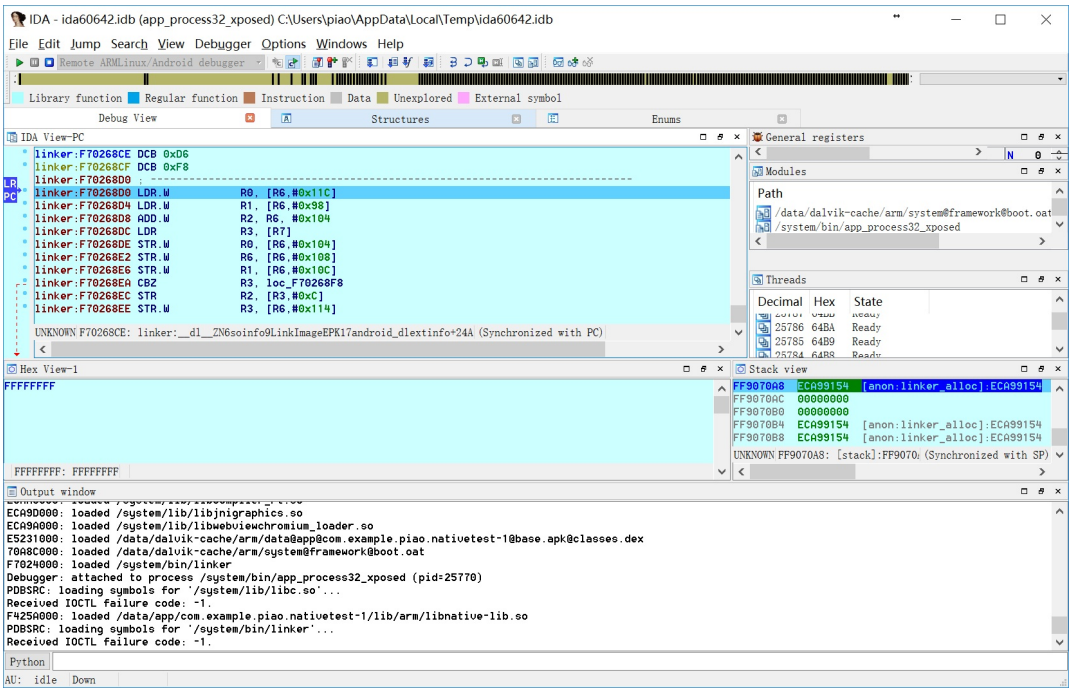
crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:21:23

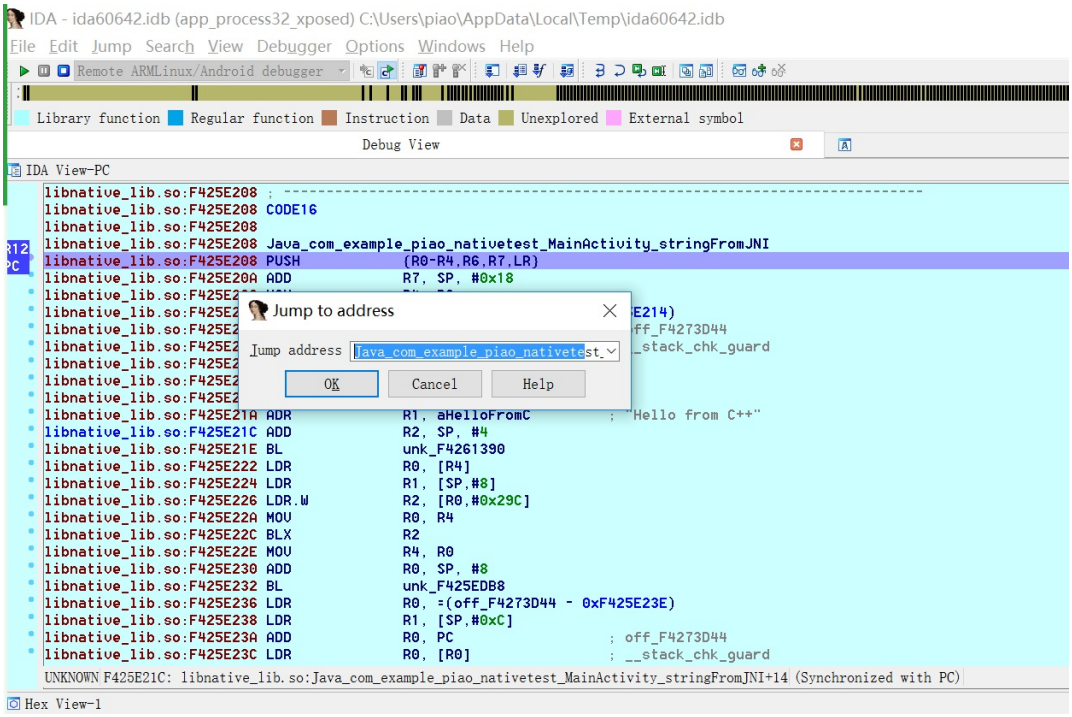
smali/baksmali

- github主页
  - JesusFreke/smali: smali/baksmali
- 功能
  - smali/baksmali is an assembler/disassembler for the dex format used by dalvik, Android's Java VM implementation.
- 语法
  - The syntax is loosely based on Jasmin's/dedexer's syntax, and supports the full functionality of the dex format (annotations, debug info, line info, etc.)

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:21:56

IDA

- `IDA` = `IDA Pro`
- 主页
  - IDA: About
- 下载
  - IDA Support: Download Center
  - 注：`IDA Pro` 是商业收费软件，请支持和购买正版
- 功能
  - 逆向工程的利器
    - 支持对Android的静态分析与动态调试
      - 动态调试so文件
- 截图
  - 官网
    - 
  - 调试安卓dex
    - 



crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:21:26

radare2

- 主页
  - Github
    - radare/radare2: unix-like reverse engineering framework and commandline tools
  - 官网
    - radare
- 功能和特点
  - 一个开源的逆向工程和二进制分析框架
  - 功能非常强大
    - 反汇编、分析数据、打补丁、比较数据、搜索、替换、虚拟化等等
    - 强大的静态或动态分析、十六进制编辑以及溢出漏洞挖掘
    - 具备超强的脚本加载能力
  - 可以运行在几乎所有主流的平台
    - GNU/Linux, Windows, *BSD, iOS, OSX, Solaris
- 组成
  - 是由一系列的组件构成
    - `rahash2`： 各种密码算法， hash算法
    - `rabin2`： 查看文件格式
    - `ragg2` / `ragg2cc`： 用于更方便的生成shellcode
    - `rax2`： 用于数值转换
    - `rasm2`： 反汇编和汇编
    - `radiff2`： 对文件进行 diff
  - `radare2`： 整合了上面的工具
- 典型用途
  - 参加CTF
  - 逆向工程
  - 漏洞挖掘
  - 分析恶意软件（如溯源）

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:21:31

安卓XML破解

此处介绍安卓的xml文件的破解相关的工具。

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-04-28 15:52:03

AXMLPrinter2

- 主页
  - GoogleCode
    - android4me - Google Code
  - Github
    - limpoxe/android4me: Automatically exported from code.google.com/p/android4me
- 下载
  - AXMLPrinter2.jar
- 作用
  - 将安卓的二进制的 `AXML` 转换为可读的 `xml` 文件
- 提示
  - 最近更新都是2008年，11年前的了
    - 看来是很老旧的工具了
- 用法
  - `java -jar AXMLPrinter2.jar xxx.xml output.xml`
  - `java -jar ~/dev/ApkTOOL/AXMLPrinter2.jar AndroidManifest.xml > main.xml`

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-04-28 16:19:14

其他辅助类工具

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:21:09

安卓调试器

在安卓反编译期间，会涉及在没有Java源码的情况下去调试安卓apk，就涉及到一些安卓调试工具。现整理如下。

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:20:32

AndBug

- 主页
  - swdunlop/AndBug: Android Debugging Library
- 作用
  - 在没有源代码的情况下，调试android上的java程序
- 功能
  - 支持断点、call stack查看、查看class、method等信息
- 截图
  -

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:20:26

Android-OpenDebug

- 主页
  - iSECPartners/Android-OpenDebug: Make any application debuggable
- 功能
  - 是一个Cydia Substrate的插件
    - 所以前提是要先安装 `Cydia Substrate`
  - 可以使得任何一个安卓程序可以被调试
    - 就有了分析和破解的可能
- 下载
  - Android-OpenDebug APK下载
- 安装
  - `adb install Android-OpenDebug.apk`
  - 或直接安装apk

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:20:29

二进制编辑器

在安卓反编译期间，可能会涉及编辑一些二进制的文件，比如 `dex` 文件等。

所以也会涉及到一些二进制编辑器，整理如下：

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-04-28 11:45:31

010editor

- 主页
  - 010 Editor - Professional Text/Hex Editor with Binary Templates
- 下载
  - SweetScape Software Inc - Download 010 Editor
- 功能：
  - 二进制(16进制)编辑器
    - 查看和编辑二进制文件
      - 用来辅助破解apk：编辑dex文件
- 截图
  -

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2020-01-16 22:55:47

EverEdit

- 首页
  - 首页 | EverEdit
- 下载
  - 下载 | EverEdit
- 功能
  - 专为国人设计的文本编辑器
    - 身躯小巧，性能卓越，自定义功能完善，丰富的主题和脚本，完美的编码、大字符集字符显示，无论您是哪个级别的码农，EverEdit都会给您带来不一样的体验！
- 特点
  - 多点编辑
    - 只需要轻按一下Alt+F3，她就会替你选择所有的同名变量；或者按Ctrl+D逐个向下选择。
  - 超强编码
    - 准确的探测文件的编码，一软在手，万码无忧！
  - Emmet/Zencoding
    - 完美支持Emmet,支持Tab一键展开那一大坨代码！
  - 完美Markdown
    - 内置markdown的预览，实时渲染，并排放置视图，一边改一边看，智能回车键和大纲！
  - 代码片段
    - 用Tab或者Shift+Tab在多个编辑点之间来回跳转。同名位置的引用，也会在修改时同时发生相同的变更。
  - 超大文件
    - 用较少的内存异步打开巨大文件！打开文件时，不会阻塞界面而且您可以随时取消该操作！
  - 二进制编辑
    - 内置二进制编辑器，瞬间打开任意大小的文件。并且可以进行查找和替换。同时对找到的字符串和被修改的位置进行高亮显示。
  - 文档地图
    - 文档地图以缩略图的形式显示出当前文档的整体外观；您可以拖放当前区域进行滚动和定位，甚至可以完全用它替换滚动条！
  - 函数列表
    - 列出安当前文件内所有的符号(类、函数、变量、宏等)，每个符号都会用一个恰当图标进行标示！
- 支持平台
  - Windows
- 截图

-

-

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-04-28 12:02:00

ClassyShark

- 主页
  - google/android-classyshark: Binary analysis of any Android/Java based app/APK/game
- 功能
  - 单独的二进制程序
    - 用来浏览和查看apk信息
      - 类的接口和成员
      - dex个数和依赖
    - 支持输入格式
      - 库
        - `dex`
        - `aar`
        - `so`
      - 可执行文件
        - `apk`
        - `jar`
        - `class`
      - 安卓二进制XML
        - `AndroidManifest`
        - 资源文件
        - 布局文件
- 截图
  -

-

-

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:20:41

APK Analyzer

- `Android Studio` 在 `v2.2` 之后自带 `APK Analyzer`
- 作用和功能
  - (重要)直观的看到apk中各个文件的大小(比如DEX，resource等等)
    - 我们可根据文件大小信息，减小apk的大小
  - (重要)学习大企业app的命名规范和目录架构规范，还可以查看大公司app使用了什么技术和第三方框架
  - 了解DEX文件的组成
  - 快速查看APK的版本信息（例如androidmanifest.xml等也可以看到）
  - 直接比较两个APK的信息，有对比才有伤害
- 下载
  - Android Studio
    - Download Android Studio and SDK tools
- 截图
  -

-
-
-

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:20:38

Fino

- `Fino` = `FINO`
- 主页
  - sysdream/fino: Android small footprint inspection tool
- 功能
  - An Android Dynamic Analysis Tool
- 特点
  - 资源占用少

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:20:51

Introspy-Android

- 主页
  - GitHub
    - iSECPartners/Introspy-Android: Security profiling for blackbox Android
  - 网站
    - Introspy-Android
- 功能
  - 帮助分析安卓app运行期间的行为
    - 以便于找到可能存在的安全问题
- 提示
  - 是个 `Cydia Substrate` 插件
    - 所以前提是先安装 `Cydia Substrate`

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:20:55

redexer

- 主页
  - plum-umd/redexer: The Redexer binary instrumentation framework for Dalvik bytecode
  - The Redexer Dalvik bytecode instrumentation platform
- 文档
  - Tutorial: using redexer to implement logging in Android apps
- 功能
  - Dalvik 字节码（用于安卓APP）分析框架
  - 它是一套基于OCaml的实用工具
    - 帮助程序员解析、操作Dalvik虚拟机。
- 作者
  - Redexer由来自马里兰大学帕克分校的PLUM组织开发完成
  - 主要作者是：Jinseong Jeon，Kristopher Micinski以及Jeff Foster

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:21:13

SmaliViewer

- `SmaliViewer` = `SV`
- 功能
  - 是一款免费的APK分析软件
  - 针对Android移动智能设备应用APK文件进行逆向
  - 用于对移动应用软件代码的分析，
  - 采用多种方法来对疑似样本进行筛选判定，如
    - 证书信息
    - 敏感SP号码信息
    - Android Manifest的权限信息
    - 函数流程图（CFG）
    - 字串表，资源文件信息
    - 敏感行为信息
    - 动态行为
  - 等等综合进行判定
    - 无论从分析的深度还是广度来看，都是一款能够满足用户需求的产品，使您在APK分析的过程中，更加得心应手。
- 下载
  - AndroidDevTools - Android开发工具 Android SDK下载 Android Studio下载 Gradle下载 SDK Tools下载
  - SmaliViewer.Zip
- 截图
  -

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:21:17

其他综合类工具

下面整理一些，去集成了，和安卓反编译和破解相关的，多个别的工具，典型的如 `dex2jar` 、 `jd-gui` 、 `apktool` 等，实现综合性的功能的一些工具。

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:22:34

ByteCode Viewer

- `Bytecode Viewer` = `BCV`
- 主页
  - 官网
    - Bytecode Viewer - Java & Android APK Reverse Engineering Suite/Tool
  - github
    - Konloch/bytecode-viewer: A Java 8 Jar & Android APK Reverse Engineering Suite (Decompiler, Editor, Debugger & More)
- 功能
  - 概述：一款基于Java 8的Jar和APK的反编译工具包
    - 包含反编译、编辑、调试等众多工具
      - 具体包含
        - 基于图形界面的：
          - 轻量级的Java的字节码查看工具
          - Java反编译器
          - 字节码编辑器
          - Smali汇编器
          - Baksmali汇编器
          - APK编辑器
          - Dex编辑器
          - APK反编译器
          - Dex反编译器
          - Procyon的Java反编译器
          - Krakatau
          - CFR的Java反编译器
          - FernFlower的Java反编译器
          - DEX2Jar
          - Jar2DEX
          - Jar-Jar
        - 以及
          - Hex查看器
          - 代码搜索器
          - 调试器
        - 等等
    - 还支持插件系统
      - 允许你与加载的类文件进行交互
        - 举例
          - 你可以写一个字符串的反混淆工具，恶意代码搜索器，或者其他的一些你所能想到的东西
- 主要特性
  - 在Bytecode Viewer的编译 / 反编译工具中集成了Krakatau
  - 集成了Smali / BakSmali－现在，你可以通过smali来编辑类文件和dex文件了
  - 支持APK / DEX－使用了Dex2Jar和Jar2Dex，可以轻松加载并保存APK文件
  - Java反编译器－Bytecode Viewer的反编译工具中集成了FernFlower，Procyon和CFR
  - 字节码编译器－CFIED的修改版
  - 十六进制查看器－由JHexPane驱动
  - 每一个反编译器 / 编辑器 / 查看器都是可以进行切换的，你可以选择每一个操作面板上所显示的元素组件
  - 功能完整的搜索系统－可以搜索字符串，函数，以及变量等信息
  - 系统完全支持使用Groovy脚本

- 截图

  -

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:22:14

Android Decompiler

- 主页
  - dirkvranckaert/AndroidDecompiler: Decompile any APK
- 功能
  - 集成了多种工具去实现反编译apk得到java代码
- 集成了哪些工具
  - `Dex2Jar` : Version 0.0.9.15
  - `android-apktool` : Version 1.5.2
  - `JD-Core-Java` : Version 1.2
  - `Artistic Style (astyle)` : Version 2.04
- 支持平台
  - Mac
  - Unix类系统
- 用法

```
usage: decompileAPK.sh [options] <APK-file>

options:
 -o,--output <dir>     The output directory is optional. If not set the
                          default will be used which is 'output' in the
                          root of this tool directory.
 --skipResources    Do not decompile the resource files
 --skipJava        Do not decompile the JAVA files
 -f,--format        Will format all Java files to be easier readable.
               However, use with CAUTION  This option might change
               line numbers
 -p,--project        Will generate a Gradle-based Android project for you
 -h,--help        Prints this help message

parameters:
 APK-file               A valid APK file is required as input
```

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:22:06

decompile-apk

- 主页
  - venshine/decompile-apk: Decompile APK（反编译APK）
- 功能
  - 集成了众多工具去从apk中反编译出java源码
    - 集成了哪些工具
      - `Apktool`：v2.2.4
      - `dex2jar`：v2.1
      - `jd-gui`：v1.4.0
      - `jadx`：v0.6.1
      - `android-classyshark`：v8.0

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:22:31

Android-Crack-Tool For Mac

- 主页
  - Jermic/Android-Crack-Tool: Android crack tool For Mac
  - https://github.com/Jermic/Android-Crack-Tool
- 下载
  - https://github.com/Jermic/Android-Crack-Tool/releases
- 介绍
  - 本软件集成了Android开发中常见的一些编译/反编译工具,方便用户对Apk进行逆向分析,提供Apk信息查看功能
- 功能
  - 反编译APK
  - 重建APK
  - 签名APK
  - 优化APK
  - DEX2JAR（APK2JAR）
  - JDGUI
  - 提取DEX
  - 提取XML
  - Class to smail
  - Apk信息查看
  - Unicode转换
- 截图

。

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:22:19

Android逆向助手

- 作者：大眼仔~旭
- 简介
  - Android 逆向助手是一款针对安卓平台的强大逆向辅助软件，功能涵盖apk反编译打包签名；dex/jar互转替换提取修复；so反编译；xml、txt加密；字符串编码等。支持直接将文件拖放到源和目标文件这，不用每次都点浏览选择。
- 支持系统：
  - WinXP、Win7，Win2003
  - 注：其它系统没有测试
- 主要功能
  - 反编译apk
  - 重新打包成apk
  - 对apk进行签名
  - 反编译dex
  - 重新打包成dex
  - dex转jar
  - dex转ddx
  - dex导出成txt
  - 反编译so
  - jar转dex
  - 提取dex
  - 替换dex
  - 修复dex
  - 加密xml转txt
  - 字符串unicode编解码
- 特别说明：
  - 源文件处支持文件或文件夹拖放
  - 必须安装 `.Net Framework 2.0` 框架
  - 部份功能依赖 `java运行环境` ，因此必须安装java
- 下载地址
  - 地址1：https://pan.baidu.com/s/1jI9L4IM
  - 地址2：Android 逆向助手 v2.2 中文版 - 大眼仔旭
- 截图

。

。

。

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:22:24

AndroidKiller

- 作者：大眼仔~旭
- 功能
  - 一款可视化的安卓应用逆向工具
    - 集Apk反编译、Apk打包、Apk签名，编码互转，ADB通信（应用安装-卸载-运行-设备文件管理）等特色功能于一身，支持logcat日志输出，语法高亮，基于关键字（支持单行代码或多行代码段）项目内搜索，可自定义外部工具；吸收融汇多种工具功能与特点
    - 打造一站式逆向工具操作体验，大大简化了安卓应用/游戏修改过程中各类繁琐工作
- 下载
  - AndroidKiller v1.2
    - 安卓 APK 反汇编工具 AndroidKiller 1.2 中文绿色免费版 - 大眼仔旭
  - AndroidKiller v1.3.1
    - AndroidKiller下载|AndroidKiller(安卓APK反汇编工具)下载 v1.3.1绿色中文版_ - pc6下载站
- 截图
  -

- 。

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:22:28

Androguard

- 主页
  - androguard/androguard: Reverse engineering, Malware and goodware analysis of Android applications
- 文档
  - Welcome to Androguard's documentation! — Androguard 3.4.0 documentation
- 功能
  - 主要用来进行静态分析安卓程序
  - 也可以用第三方库去反编译安卓
    - 支持的第三方库
      - `DAD`
      - `dex2jar + jad`
      - `DED`
- 包括多个模块/子功能
  - `androrisk.py`：该模块用于分析apk危险级别
  - `androapkinfo.py`：该模块分析apk列出其中的文件类型、权限、4大组件、是否NDK反射等信息
  - `androaxml.py`：该模块用于展示apk androidmanifest.xml
  - `androgexf.py`：该模块生成函数调用图
  - `apkviewer.py`：该模块生成指令级别的调用图
  - `androlyze.py`：该模块为交互分析环境
- 特点
  - 用 `Python` 写的
    - 支持多个平台：Linux/Windows/Mac
  - 支持多种模式
    - 命令行模式
    - 图形界面模式
    - 被当做库文件使用
- 截图
  -
  -

。

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:22:02

AFE

- `AFE` = `Android Framework for Exploitation`
- 主页
  - Github
    - appknox/AFE: Android Framework for Exploitation, is a framework for exploiting android based devices
  - 官网
    - Appknox - Mobile App Security Testing
      - 这家公司主要是做移动应用的安全测试和解决方案的
- 功能
  - is a framework for exploiting android based devices
- 说明
  - 是基于androguard的

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:22:10

其他相关工具

此处介绍，在安卓破解期间，会用到的一些相关的其他方面的工具。

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:23:46

安卓Hook框架

破解安卓期间，往往需要安装一些hook相关的插件到一些框架上，此处介绍相关的框架。

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:22:56

XPosed框架

XPosed框架是用于安装相关插件 `FDex2` ，配合破解安卓，导出运行时app的dex文件。

而之前自己折腾过这个： `XPosed框架` = `XPosed Installer`

比如安装到了 `夜神模拟器` 中的 `XPosed` ：

更详细的解释详见专门教程：

- 强大的安卓破解辅助工具：XPosed框架

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:23:03

Cydia Substrate

- 主页
  - * Cydia Substrate
- 下载
  - com.saurik.substrate.apk
- 功能
  - 和 `Xposed` 类似的框架，用来安装各种插件，实现各种功能。
    - 比如可以：
      - 安装绕过ssl检测的插件，用来破解 `ssl pinning`
        - 关于安卓的app中的https：
          - app内部启用了：
            - `SSL Pinning` = `ssl certificate pinning` = `certificate pinning`
              - = `ssl证书绑定` =证书绑定`
      - 此处也可以用来安装相关插件，导出安卓的dex文件
- 特点
  - Hook底层方法非常方便
    - 对so中的方法hook操作非常便捷
- 截图
  -

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2020-01-16 23:02:29

frida

- 主页
  - Github
    - Frida
  - 官网
    - Frida • A world-class dynamic instrumentation framework | Inject JavaScript to explore native apps on Windows, macOS, GNU/Linux, iOS, Android, and QNX
- 简介
  - 一款基于 `python` + `javascript` 的 `hook框架`
  - 可运行在Android/iOS/Linux/Win/OSX等各平台
  - 主要使用动态二进制插桩(DBI)技术
    - 将外部代码注入到现有的正在运行的二进制文件中，从而让它执行额外操作
      - 支持哪些额外操作
        - 访问进程内存
        - 在应用程序运行时覆盖函数
        - 从导入的类调用函数
        - 在堆上查找对象实例并使用
        - Hook、跟踪和拦截函数等
      - 注：调试器也能完成相应工作，不过非常麻烦，比如各种反调试
- 功能和特点
  - Scriptable
    - Inject your own scripts into black box processes. Hook any function, spy on crypto APIs or trace private application code, no source code needed. Edit, hit save, and instantly see the results. All without compilation steps or program restarts.
  - Portable
    - Works on Windows, macOS, GNU/Linux, iOS, Android, and QNX. Install the Node.js bindings from npm, grab a Python package from PyPI, or use Frida through its Swift bindings, .NET bindings, Qt/Qml bindings, or C API.
  - Free
    - Frida is and will always be free software (free as in freedom). We want to empower the next generation of developer tools, and help other free software developers achieve interoperability through reverse engineering.
  - Battle-tested
    - We are proud that NowSecure is using Frida to do fast, deep analysis of mobile apps at scale. Frida has a comprehensive test-suite and has gone through years of rigorous testing across a broad range of use-cases.
- 文档
  - Welcome | Frida • A world-class dynamic instrumentation framework
- 截图

。

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:22:53

安卓模拟器

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:22:43

夜神安卓模拟器

此处使用夜神安卓模拟器，来配合破解安卓：

- 可以在夜神模拟器的XPosed中安装 `FDex2` ，用来从运行期间的安卓app导出dex文件
- 同时夜神自带的 `文件管理器` 可以方便的导出dex文件到PC端

更详细的解释详见专门教程：

好用的安卓模拟器：夜神Nox

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:22:40

文件浏览器

想要把安卓中（导出的dex等）文件拷贝（导）出来，一般需要借助于好用的安卓浏览器。

下面整理一些好用的安卓浏览器：

夜神模拟器中自带文件浏览器

ES文件浏览器

功能很强，比较流行的安卓的文件浏览器。

MT管理器

- `MT` = `MT管理器` = `MT Manager` = `MT Manager for Android`
- 是什么：安卓中的一个文件管理器
- 常被简称为：`MT2`
  - 因为最新版本是v2
    - 比如：
      - MT浏览器_V2.5.4.apk 文件管理器
      - 文件管理神器 MT Manager v2.6.1 for Android
      - MT管理器2.0

特点

- 支持在VirtualXposed中使用MT
- 除了普通文件管理功能外，还支持APK反编译相关功能

应用简介

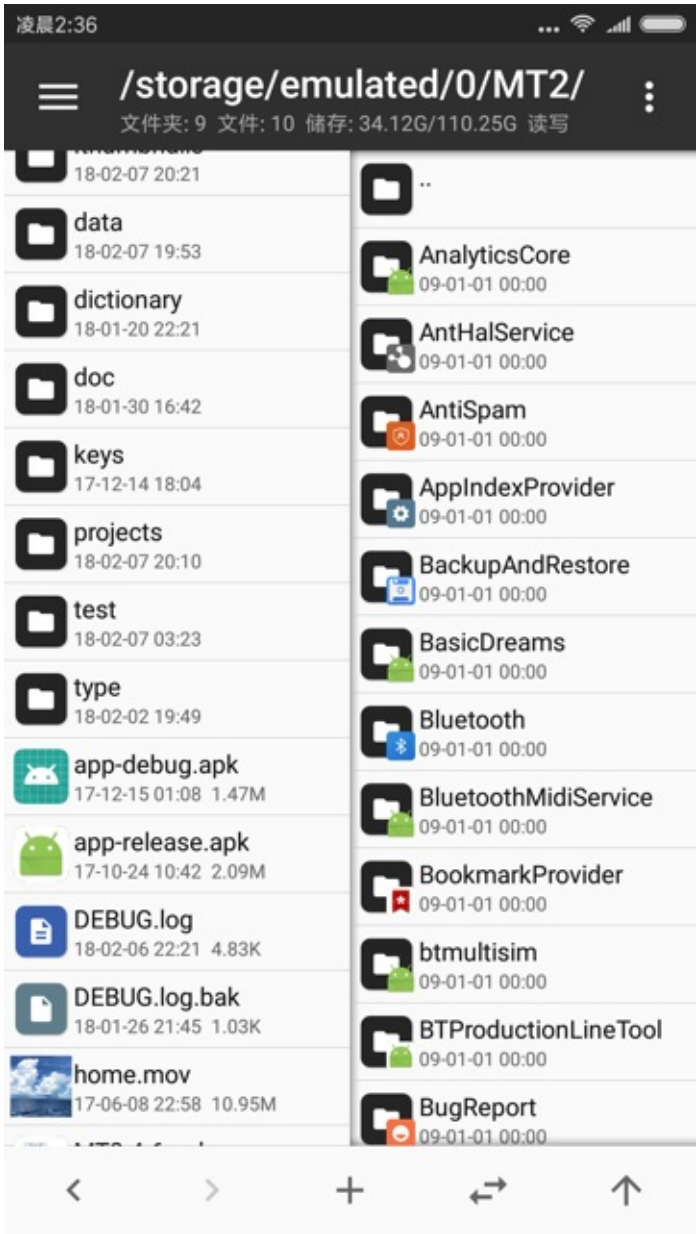
MT管理器是一款强大的文件管理工具和APK逆向修改神器。

- 如果你喜欢它的双窗口操作风格，可以单纯地把它当成文件管理器使用。
- 如果你对修改APK有深厚的兴趣，那么你可以用它做许许多多的事
  - 例如汉化应用、替换资源、修改布局、修改逻辑代码、资源混淆、去除签名校验等，主要取决于你如何使用。

主要功能

- 文件复制、移动、创建软链接、重命名、删除、创建文件(夹)，文件批量操作。获取 Root 权限后可访问系统目录，挂载文件系统为读写，修改文件权限和所有者。
- 像 WinRAR 那样打开 ZIP 格式文件，可以对 ZIP 内的文件进行删除、重命名、移动，添加/替换外部文件到 ZIP 中，无需解压后再重新打包，同时支持单独解压 ZIP 内的部分文件。
- 自带强大的文本编辑器，可以流畅编辑大文本文件，支持设置是否显示行号、开关自动换行、双指缩放字体大小、自动识别编码、代码语法高亮、自动缩进、正则搜索替换。
- 拥有图片查看、音乐播放、字体预览、执行脚本、文本对比等功能，在侧拉栏中可方便地查看存储设备、FTP连接、书签、后台、工具等。
- APK 编辑功能，主要有 DEX 编辑，ARSC 编辑，XML 编辑，APK 签名、APK 优化、APK 共存、去除签名校验、RES 资源混淆、RES 反资源混淆、翻译模式等。

截图举例

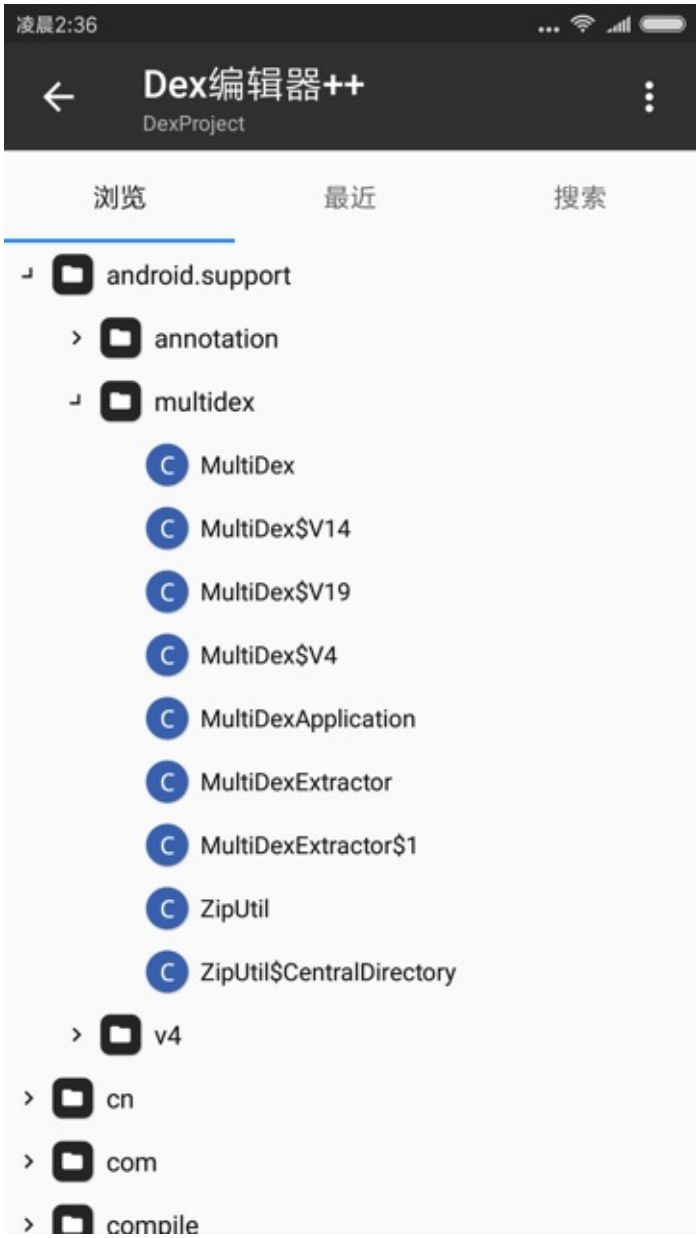

凌晨2:36
/storage/emulated/0/MT2/
文件夹: 9 文件: 10 储存: 34.12G/110.25G 读写
18-02-07 20:21
data
18-02-07 19:53
dictionary
18-01-20 22:21
doc
18-01-30 16:42
keys
17-12-14 18:04
projects
18-02-07 20:10
test
18-02-07 03:23
type
18-02-02 19:49
app-debug.apk
17-12-15 01:08 1.47M
app-release.apk
17-10-24 10:42 2.09M
DEBUG.log
18-02-06 22:21 4.83K
DEBUG.log.bak
18-01-26 21:45 1.03K
home.mov
17-06-08 22:58 10.95M
..
AnalyticsCore
09-01-01 00:00
AntHalService
09-01-01 00:00
AntiSpam
09-01-01 00:00
AppIndexProvider
09-01-01 00:00
BackupAndRestore
09-01-01 00:00
BasicDreams
09-01-01 00:00
Bluetooth
09-01-01 00:00
BluetoothMidiService
09-01-01 00:00
BookmarkProvider
09-01-01 00:00
btmultisim
09-01-01 00:00
BTProductionLineTool
09-01-01 00:00
BugReport
09-01-01 00:00

凌晨2:36
MT管理器
Bin
存储
根目录
56%
2.71G已用，2.05G可用
内部存储
30%
34.09G已用，76.17G可用
书签
MT2
/storage/emulated/0/MT2
Download
/storage/emulated/0/Quark/Download
QQfile_recv
/storage/emulated/0/tencent/QQfile_recv
工程
工具
远程管理
FTP客户端

凌晨2:36
/storage/emulated/0/MT2/
文件夹: 9 文件: 10 储存: 34.12G/110.25G 读写
.thumbnails
18-02-07 20:21
..
data
18-02-07 19:53
AnalyticsCore
09-01-01 00:00
dictionary
18-01-20 22:21
AntHalService
09-01-01 00:00
doc
AntiSpam
复制 ->
移动 ->
链接 ->
重命名
删除
压缩
属性
分享
打开方式...
添加书签
17-10-24 10:42 2.09M
09-01-01 00:00
DEBUG.log
18-02-06 22:21 4.83K
btmultisim
09-01-01 00:00
DEBUG.log.bak
18-01-26 21:45 1.03K
BTProductionLineTool
09-01-01 00:00
home.mov
17-06-08 22:58 10.95M
BugReport
09-01-01 00:00

凌晨2:36
Hex编辑器
resources.arsc *
0 1 2 3 4 5 6 7 01234567
00000 02 00 0C 00 94 1D 04 00
00008 01 00 00 00 01 00 1C 00
00010 00 FE 00 00 32 07 00 00 þ 2
00018 00 00 00 00 00 01 00 00
00020 E4 1C 00 00 00 00 00 00 ä
00028 00 00 00 00 1C 00 00 00
00030 36 00 00 00 4A 00 00 00 6 J
00038 76 00 00 00 A7 00 00 00 v §
00040 01 23 45 67 89 AB CD EF #Eg«Íï
00048 2B 01 00 00 4E 01 00 00 + N
00050 7A 01 00 00 97 01 00 00 z
00058 D3 01 00 00 F9 01 00 00 Ó ù
选择
复制
复制为...
删除
00078 18 03 00 00 42 03 00 00 B
00080 61 03 00 00 85 03 00 00 a
00088 A5 03 00 00 C7 03 00 00 ¥ Ç
00090 F8 03 00 00 19 04 00 00 ø
00098 3D 04 00 00 64 04 00 00 = d
000A0 96 04 00 00 C7 04 00 00 Ç
000A8 EB 04 00 00 1B 05 00 00 ë
000B0 4A 05 00 00 87 05 00 00 J
000B8 C3 05 00 00 F6 05 00 00 Ã ö
000C0 28 06 00 00 49 06 00 00 (I
000C8 75 06 00 00 A3 06 00 00 u £
000D0 D5 06 00 00 08 07 00 00 Õ
000D8 23 07 00 00 5C 07 00 00 # \
000E0 82 07 00 00 9F 07 00 00
000E8 BD 07 00 00 EF 07 00 00 ½ ï
Sel: 80h - C7h Len: 48h Size: 41D94h [HEX]

凌晨2:36
AndroidManifest.xml
Android Xml (反编译) *
1 <?xml version="1.0" encoding="utf-8"?>
2 <manifest xmlns:android="http://schemas.
android.com/apk/res/android"
3 package="cn.edu.njupt.cengceng"
4 platformBuildVersionCode="25"
5 platformBuildVersionName="7.1.1"
6 android:versionCode="1"
7 android:versionName="1.0">
8 <uses-sdk
9 android:minSdkVersion="14"
10 android:targetSdkVersion="25" />
11 <!-- 拥有完全的网络访问权限 -->
12 <uses-permission android:name="android.
permission.INTERNET" />
13 <application
14 android:theme="@style/AppTheme"
15 android:label="@string/app_name"
16 android:icon="@mipmap/ic_launcher"
17 android:allowBackup="true"
18 android:supportsRtl="true"
19 android:roundIcon="@mipmap/
ic_launcher_round"
20 <activity
21 android:theme
NoActionBar"
22 android:label=
23 android:name
正则表达式
区分大小写
关闭
查找 android
上个
下个
替换
全部



crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:23:43

附录

下面列出相关参考资料。

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2019-05-02 15:25:25

参考资料

- 【已解决】mac中试用FDex2去hook导出安卓app的dex等文件
- 【已解决】从不同版本的小花生apk中反编译出包含业务逻辑代码的dex和jar包源码
- 【已解决】小花生安卓app的v3.4.8版破解后找到源码中是否包含J字段的加密逻辑
- 【已解决】用apktool，dex2jar，jd-gui去反编译安卓apk查看app源码
- 【已解决】尝试破解小花生app安卓apk希望看到api返回的json中的J的解密算法得到明文
- 【已解决】如何反混淆即还原反编译后混淆的安卓代码
- 【已解决】mac中安装最新版本的安卓反编译工具：Apktool
- 【记录】从反编译安卓apk得到的java源码代码中尝试找返回json中J加密的逻辑和线索
- 【已解决】小花生app中api请求返回json的C，J，M，ST的含义和如何破解解密
- 【记录】爬取小花生app中自主阅读馆和亲子阅读馆中的有音频的绘本数据
- 【未解决】dex2jar反编译dex后jar文件包含java代码：throw new VerifyError bad dex opcode
- 【已解决】python实现java的MessageGZIP.uncompressToString即gzip的解码
- 【已解决】Python中实现java中的Base64.decode解码加密字符串
- 【已解决】从反编译小花生apk得到的包含业务逻辑代码中找到J字段解码的逻辑并用Python实现
- 【无需解决】安卓apk反编译出的smali反向破解出java原始代码
- 【已解决】安卓app如何脱壳如何破解加固
- 【已解决】找小花生app的旧版本apk并尝试能否安装使用
- 【部分解决】尝试破解安卓apk康美通去得到java源码
- 【整理】JVM参数-Xms和-Xmx参数的含义
- 【未解决】用ART，oat，dex2oat相关机制去破解新一代360、腾讯等安卓apk的加固
- 【整理】把jar包转换为java源代码的java反编译器的整理和对比
- 【已解决】用基于Procyon的Luyten反编译安卓jar包得到java源码
- 【未解决】小花生中如何得到getToken的计算逻辑以便得到正确的md5值可以正常请求接口
- 【整理】java反编译器对比：JD-GUI，CFR，Procyon，Jadx
- 【已解决】mac版JD-GUI查看并导出jar包的java源代码
- 【已解决】用java反编译器CFR从jar包导出java源代码
- 【已解决】用Procyon命令行去从jar包导出java源代码
- 【已解决】用jadx把安卓dex文件转换提取出jar包和java源代码
- 【已解决】搞懂安卓app混淆和加固常见做法和相关逻辑
- 【已解决】mac中用jadx命令行CLI从apk中导出java源码
- 【已解决】mac中用jadx从apk中导出java源码
- 【记录】从安卓的apk中解压出各种项目文件
- 【已解决】用WrBug的DumpDex从app中hook导出dex文件
- 【已解决】mac中用dex2jar反编译dex文件导出jar包文件
- 【已解决】夜神安卓模拟器中导出文件到mac电脑
- 【已解决】Nox夜神安卓模拟器中/mnt/shared对应Mac的共享目录在哪里
- 【已解决】用jadx命令行从dex文件转换出java源代码
- 【基本解决】尝试破解安卓apk马蜂窝去得到java源码
-
- 好用的安卓模拟器：夜神Nox
- Android 反编译利器，jadx 的高级技巧 - 简书
- 【手机脱壳】MT2+VXP+FDex2实现免Root脱壳 - Powered by Discuz!
- HangZhouCat/ReaverAPKTools: 逆向APK工具
- 文件管理神器 MT Manager v2.6.1 for Android-心海e站
- MT管理器2.0(bin.mt.plus) - 2.6.0 - 应用 - 酷安网
- MT管理器2.6.1公测版（第三版） - Powered by Discuz!
- 介绍 - MT管理器

- Android 逆向助手 v2.2 中文版 - 大眼仔旭
- Jermic/Android-Crack-Tool: Android crack tool For Mac
- 破解某小说App（一） - 掘金
- 最新乐加固脱壳详细教程（有图有真相） - Android安全 - 逆向未来技术社区 - Powered by Discuz!
- Android逆向助手反编译APK - 长江某菜鸟的博客 - CSDN博客
- 加固保-移动应用安全资讯
- android - Is there a way to get the source code from an APK file? - Stack Overflow
- spriteviki/Dex2oatHunter: Automatic Unpacking Tool for Android Dex Files
- Dex : Java Data Visualization
- What are .dex files in Android? - Stack Overflow
- Dalvik 可执行文件格式 | Android Open Source Project
- ART 和 Dalvik | Android Open Source Project
- How to decompile an APK or DEX file using jadx in Windows | Our Code World
- 压缩代码和资源 | Android Developers
- Android应用加固产品使用对比 - 『移动安全区』 - 吾爱破解 - LCG - LSG |安卓破解|病毒分析|破解软件|www.52pojie.cn
- S3cuRiTy-Er1C/JebScripts: Jeb public scripts
- flankerhqd/jebPlugins: Various Jeb plugins, including obfuscation restore
- enovella/jebscripts: A set of JEB Python/Java scripts for reverse engineering Android obfuscated code
- CalebFenton/simplify: Generic Android Deobfuscator
- Android 反混淆神器JEB2的使用简介 - 飞少的博客 | Jack's Blog
- 【技术分享】Android程序反混淆利器——Simplify工具 - 安全客，安全资讯平台
- [原创]JEB2反混淆神器-『Android安全』-看雪安全论坛
- Android | 使用Java Deobfuscator对JEB Decompiler反混淆_ - AppScan.IO | Janus移动安全中心
- Deobfuscating Android Triada malware – JEB Decompiler in Action
- ANDROID 逆向实例（八） — 乐固加固脱壳（2017.01） ~ and-rev
- 乐固加固脱壳实战 - faTe's Home
- Android APK脱壳--腾讯乐固、360加固一键脱壳 - 知乎
- Android APK脱壳--腾讯乐固、360加固一键脱壳 | 辉天神龙
- 强大的安卓破解辅助工具：XPosed框架
- 好用的安卓模拟器：夜神Nox
- Apktool - A tool for reverse engineering 3rd party, closed, binary Android apps
- Apktool - How to Install
- Android 各种脱壳工具使用 - 简书
- Android逆向入门流程 - 简书
- lambdalang/DexExtractor
- DexExtractor的原理分析和使用说明 - Fly20141201. 的专栏 - CSDN博客
- system-arm_md5_6395c2f1451dbbed027d7293ab39a6e7.img.tar.gz
- 使用drizzleDumper脱去某数字公司的壳 - suwenlai的博客 - CSDN博客
- 一种常规Android脱壳技术的拓展（附工具） - FreeBuf互联网安全新媒体平台
- DrizzleRisk/drizzleDumper: drizzleDumper是一款基于内存搜索的Android脱壳工具
- pxb1988/dex2jar: Tools to work with android .dex and java .class files
- dex2jar | Penetration Testing Tools
- JesusFreke/smali: smali/baksmali
- Java Decompiler
- java-decompiler/jd-gui: A standalone Java Decompiler GUI
- Releases · java-decompiler/jd-gui
- jd-gui-osx-1.4.1.tar
- Release Luyten v0.5.4 · deathmarine/Luyten
- deathmarine/Luyten: An Open Source Java Decompiler Gui for Procyon
- Releases · deathmarine/Luyten

- Bytecode Viewer - Java & Android APK Reverse Engineering Suite/Tool
- Konloch/bytecode-viewer: A Java 8 Jar & Android APK Reverse Engineering Suite (Decompiler, Editor, Debugger & More)
- Bytecode Viewer-一款基于Java 8的Android APK逆向工具包 - 安全客，安全资讯平台
- skylot/jadx: Dex to Java decompiler
- Jadx 如何反混淆deobfuscation
- 010 Editor - Professional Text/Hex Editor with Binary Templates
- SweetScape Software Inc - Download 010 Editor
- IDA: About
- IDA Support: Download Center
- 安卓逆向学习笔记 (4) - 使用IDA Pro动态调试so文件 - 蜗牛 - CSDN博客
- 安卓 APK 反汇编工具 AndroidKiller 1.2 中文绿色免费版 - 大眼仔旭
- AndroidKiller下载|AndroidKiller(安卓APK反汇编工具)下载 v1.3.1绿色中文版_ - pc6下载站
- GDA反编译器与其他android逆向工具对比 - 知乎
- GDA介绍-功能强大的交互式Android反编译分析工具
- EverEdit：带来惊喜的国产软件 - 简书
- 原创 使用AndBug调试Android Java Bytecode-『Android安全』-看雪安全论坛
- swdunlop/AndBug: Android Debugging Library
- Android安全专项-AndBug动态调试工具 - doctorq - CSDN博客
- androguard/androguard: Reverse engineering, Malware and goodware analysis of Android applications
- Welcome to Androguard’s documentation! — Androguard 3.4.0 documentation
- Android逆向之旅—Native层的Hook神器Cydia Substrate使用详解 | 尼古拉斯.赵四
- AndroidSecNotes/Android Hook 框架（Cydia篇）.md at master · JnuSimba/AndroidSecNotes
- 看雪 - 安卓黑科技之HOOK详解
- Android逆向工程工具Dare的使用方法（Mac OS X中） - qysh123的专栏 - CSDN博客
- Dare Homepage
- Dare downloads
- Google反编译新工具——Enjarify - liuweiballack的专栏 - CSDN博客
- android:tools:enjarify WooYun WiKi
- Android 逆向工具 dex2jar, enjarify 和 AXMLPrinter2 // Neurohazard
- Dedexer user's manual
- dedexer - Browse Files at SourceForge.net
- Dedexer:Dex文件反编译工具介绍 - amos_tl - ITeye博客
- 2015移动安全挑战赛（阿里&看雪主办）全程回顾（3） - Group of Software Security In Progress
- 关于Indroid的编译 · Issue #1 · romangol/InDroid
- iSECPartners/Introspy-Android: Security profiling for blackbox Android
- Introspy-Android
- 目前最全面的Android安全工具清单 - IT经理网
- 翻译Android 应用逆向工具总结-『外文翻译』-看雪安全论坛
- Infografía sobre el funcionamiento interno del robot de Android | SmallVille
- intellij-community/plugins/java-decompiler/engine at master · JetBrains/intellij-community
- fesh0r/fernflower: Unofficial mirror of FernFlower Java decompiler (All pulls should be submitted upstream)
- 【工具分享】Radare 2之旅：通过crackme实例讲解Radare 2在逆向中的应用（上） - 安全客，安全资讯平台
- radare/radare2: unix-like reverse engineering framework and commandline tools
- radare
- Radare2使用全解 - 先知社区
- radare2逆向笔记 - 有价值炮灰 - 博客园
- Android逆向之旅---Hook神器家族的Frida工具使用详解 - Android应用安全防护和逆向分析-----作者 - CSDN博客
- 浅谈 android hook 技术 - Android - 掘金
- 使用Frida框架进行hook | La0s
- Android逆向之旅-Hook神器家族的Frida工具使用详解 - 云+社区 - 腾讯云

- hookmaster/frida-all-in-one: 《FRIDA操作手册》by @hluwa @r0ysue
- 详解Hook框架frida，让你在逆向工作中效率成倍提升 - 知乎
- Android Hook工具之Frida 基础使用 - 简书
- 《Smali Viewer 用户指南》 | AVL Team
- AndroidDevTools - Android开发工具 Android SDK下载 Android Studio下载 Gradle下载 SDK Tools下载
- SmaliViewer.Zip
- Android反编译 - 简书
- Android apk分析工具：APK Analyzer - 简书
- Making the most of the APK analyzer – Android Developers – Medium
- Analyze your build with APK Analyzer | Android Developers
- 1.2.1 APK反编译工具之：Procyon，Jadx和AndroidDecompiler - Coder-Pig的猪栏 - CSDN博客
- android - decompiling DEX into Java sourcecode - Stack Overflow
- decompiler - how to use DEXtoJar - Stack Overflow
- Android反编译简单实战 - 知乎
- Android混淆（ProGuard）从0到1 - 简书
- 乐固壳分析 - bamb00 - 博客园
- Android APK 反编译实践 - 简书
- 5分钟学会基于Xposed+DumpDex的apk快速脱壳方法 - 简书
- 腾讯加固纯手工简易脱壳教程 - 『移动安全区』 - 吾爱破解 - LCG - LSG |安卓破解|病毒分析|破解软件|www.52pojie.cn
- 花生日记APP邀请注册机实战（360加固脱壳） – Silkage's Blog
- 如何反编译Android 的apk/dex/odex，获得源码 – 码农日记
- Android逆向之路---脱壳360加固 - 简书
- 26款优秀的Android逆向工程工具 - 简书
- Application Hardening - Mobile App Hardening | Promon
- Cydia Substrate使用手册 - 简书
- 看雪安全论坛 18年专注——顶级软件逆向论坛
- 一张图看懂Android编译流程 - 简书
- 原创 如何使用Xposed+JustTrustMe来突破SSL Pinning-『WEB安全』-看雪安全论坛
- 当你写爬虫抓不到APP请求包的时候该怎么办？【中级篇】 - 知乎
- tls - What is certificate pinning? - Information Security Stack Exchange

crifan.com，使用署名4.0国际(CC BY 4.0)协议发布 all right reserved，powered by Gitbook最后更新： 2020-01-16 23:03:37