

基础知识:Java的破解和反破解之道 PDF转换可能丢失图片或格式，建议阅读原文

https://www.100test.com/kao_ti2020/145/2021_2022__E5_9F_BA_E7_A1_80_E7_9F_A5_E8_c104_145134.htm java字节码能够很容易被反编译大家都晓得啦，今天下午我为了得到一个心仪已久的j builder opentools（昨天1.0 Released，新鲜出炉！但只能用14天，这怎么行~@@#!@#!#@!@#%%^@，少说也要140天嘛！），于是我不惜放下其他工作，研究了一把该软件加密方法的破解和反破解，结合以前的一些经验，作文一篇与大家共飨，并不是鼓励大家... 破解之道：如今市面上的java obfuscator很多(可以从google分类中列出)比较著名的有4thpass的产品（呵呵，胖友们！KBrowser都用过了吧），不要钱的可以用JODE（JODE即是Obfuscator也是Decompiler，还提供源程序，推荐初用者使用），一般来说代码扰乱器工作原理有三种，最初级的有比如Jbuilder自带的（缺省情况下该项功能关闭），能把私有变量和方法的名称用乱码代替；稍微高级一点的能把公开变量和方法也能用乱码代替，通常是输入你要扰乱的jar和一个脚本（用来控制保留部分，否则你的主程序也不能执行了），有些不用乱码代替变量名，而是直接用Java的关键字，读者可能会感到疑惑，其实这种工具是绕过了Java编译器的限制，输入class或jar（不是源程序），扰乱后输出class或Jar，当然JVM还是能够执行的！如果用一般的方法这些类文件是没法经过反编译后修改再用javac或jikes编译的，破解之道是先用反编译软件发编译出来（所有非法变量名加前缀），然后依样画葫芦用同样的扰乱器生成修改好的类文件以达到目的；再高级一点就不是用常规方法了，一

些是针对市面上出现反编译软件做一些陷阱，使得这些反编译软件不能工作；还有一种是自己做一个Java编译器（JDK里也有一个java实现的编译器），在符合JVM规范的前提下乱编译，一般用在applet上，使得一般的反编译软件根本就解释不了。目前我只能搞搞中等扰乱的程序。最著名的反编译器有JAD1.58e是用C写的，前台还有一个Delphi的界面。用它可以用很快的编译，我顺便写了一个perl小程序，可以批量编译上千个class。对一些提供license.key（包含授权信息的加密文件）的软件，一般这种文件会采用DES,RAS和CRC校验而且一般是二进制的（即使有时输出成BASE64编码），直接修改文件是浪费时间的，你可以先反编译通过阅读源程序来探究解密过程，如果过程是可逆的，那么你自己实现一个加密过程，可以很容易的生成你自己想要的license key；如果过程不可逆也不是就搞不定了，有些强度不大的加密算法还是可以用暴力破解法来搞定，还有一种情况是对数字加密（一般指过期时间）如果你能修改这个过期时间那么你就可以多用一会儿了，用数学方法描述一下：假设集合X是明文包含的元素集合，Y是X经过算法后的映射，包含密文元素，如果有存在两个算法A和B，能使得 $\{Y - A \rightarrow X\} = \{Y - B \rightarrow X\}$ ，A算法可逆，但B算法是不可逆的，生产方用A的逆算法加密授权信息（X:String）到（Y:byte[]），并在软件中用B算法解密，这样你就搞不定了，但如果集合X的元素是有限的，假设只有0-9（new Date().getTime()格式），那么算法B就称为不可逆但不可靠的，因为你通过一个样本（一般都会给你评价版的license啦！），是可以得到某些Y集合中元素在X集合中的逆映射的，这样你可以直接用这张映射表来修改license了...

反破解之道：如果是做产品或提供演示程序，加密还是有好处的，加密的软件可以用上面提到的JODE，一般都是对编译好的class文件进行扰乱，因为并不是所有的符号都需要扰乱，如果你开发的是一个类库，或者某些类需要动态装载，那些公共API就必须保留符号不变，这样别人才能使用你的类库。先编写脚本对那些需要保留的符号名称进行配置，某些扰乱器能够调整字节码的顺序，使反编译更加困难。如果你用的代码扰乱器能保证别人不能通过反编译来修改或代替你的class，那么你还得注意不要用不可靠的加密算法。我自己写了一个不可逆且可靠的算法，正在申请专利中.... 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com