**Intellij IDEA JRebel 5.6.3 手動破解方法**

2015-02-04 17:06:33 | 分類：[讀書筆記](http://blog.tongbor.com/article/list/1) | 標籤： [idea](http://blog.tongbor.com/search/query?k=idea) [jrebel](http://blog.tongbor.com/search/query?k=jrebel) [破解](http://blog.tongbor.com/search/query?k=%E7%A0%B4%E8%A7%A3) | 有 1428 人流覽

在系統開發調試過程中，因為需要不停地修改代碼導致需要不停地發佈系統，而等待系統發佈完成是個很“漫長而痛苦”的過程。有什麼辦法可以讓修改的代碼在不需要重新發佈系統的情況下馬上生效呢？JRebel就是一個可以解決此問題的元件，它是一個支援java應用熱部署的JVM外掛程式。有了JRebel，就可以為程式師節省很多寶貴時間。

JRebel就是一個好的東西，但是它不能被免費擁有。10人的團隊使用JRebel一年，官網售價為**$4150**，你沒有看錯，是$不是￥！

JRebel位元組碼是經過混淆處理的，google了一下這方面的資料，得到一些啟發，嘗試破解最新版，破解成功。

破解過程大致都一樣，主要的檔是jrebel.jar、jrebel.lic。可以通過官網線上申請14天的使用註冊碼，來生成jrebel.lic。

請勿留郵箱求最新版，自己動手豐衣足食，這裡僅提供思路和方法。

**01. 準備工具**

反編譯工具：[jd-gui](http://jd.benow.ca/)

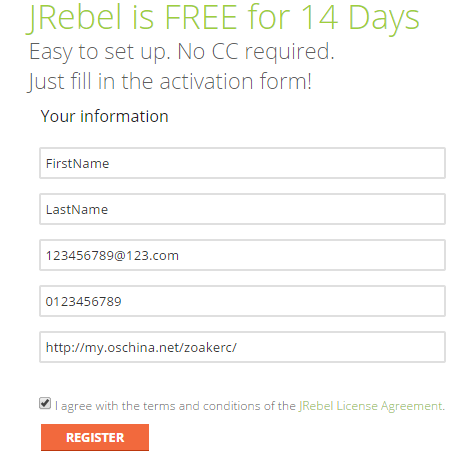
編輯Java位元組碼的類庫：[Javassist](http://www.javassist.org/)

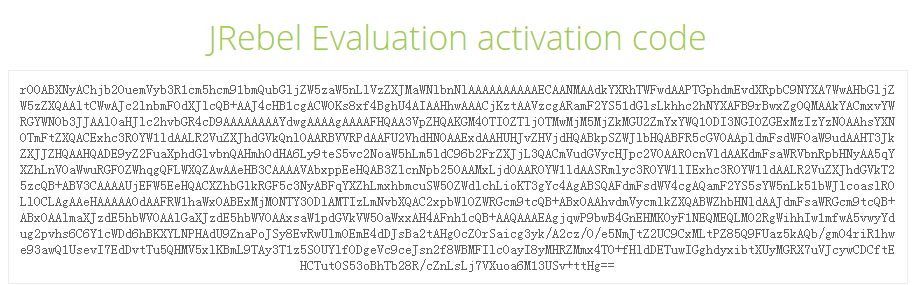
IDEA外掛程式 JRebel：[JRebel Plugin For IDEA](http://plugins.jetbrains.com/plugin/4441?pr=idea&showAllUpdates=true)

JRebel試用註冊碼線上申請：[JRebel is FREE for 14 Days](https://zeroturnaround.com/software/jrebel/download/)

**02. 申請試用註冊碼**

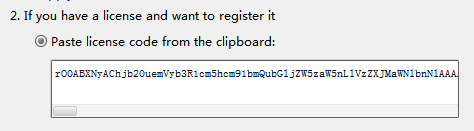
打開申請註冊碼的位址，隨便填寫申請資訊，申請成功後會生成一個試用的註冊碼，先保存起來。

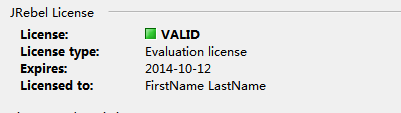




**03. 註冊JRebel生成license.lic文件**

IDEA中安裝JRebel完成後，打開settings -> IDE settings -> JRebel -> Open activation dialog -> Paste license file from the clipboard，將上面的試用註冊碼粘貼到這裡。確定後會顯示註冊成功，但是有限制時間。





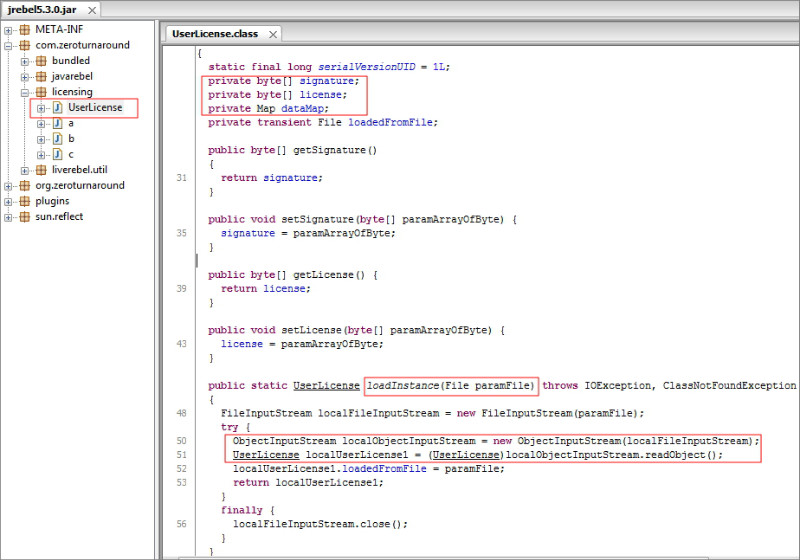
關閉IDEA，將JRebel生成的license.lic和安裝好的jrebel.jar提取出來，先放到C盤根目錄。

jrebel.lic所在路徑：C:\Users\當前用戶名\.jrebel\jrebel.lic

jrebel.jar所在路徑：C:\Users\當前用戶名\.IntelliJIdea13\config\plugins\jr-ide-idea\lib\jrebel\jrebel.jar

**04. 反編譯並查看UserLicense.java**

用反編譯工具jd-gui打開jrebel.jar，找到類：**com.zeroturnaround.licensing.UserLicense**，這是用戶license所對應的類。此類同時還提供了分別從URL，byte陣列以及檔中反序列化(deserialization)UserLicense物件的三個重載方法。



**5. JRebel license檔的結構**

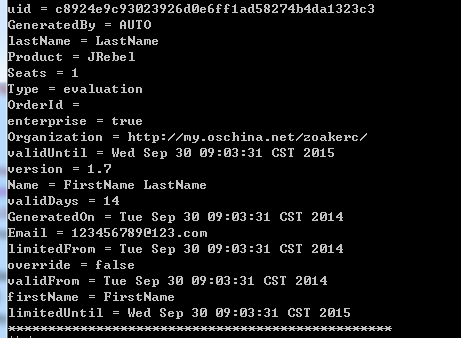
從上面的UserLicense截圖可以看到，UserLicense類有5個成員變數，其中第一個是static，最後一個是transient，這兩個變數在物件序列化(serialization)時是不會寫入檔的，所以用戶license主要包括：signature、license和dataMap。

使用類中的方法loadInstance(File paramFile)測試附件中的license檔jrebel.lic，發現反系列化(deserialization)後UserLicense物件中的signature和license有值，而dataMap的值為null。經過分析發現，把license反序列化到dataMap物件中，即可以得到用戶License：

[?](http://my.oschina.net/zoakerc/blog/309381)

|  |  |
| --- | --- |
|  | UserLicense userLicense = UserLicense.loadInstance(new File("c:/jrebel.lic"));  ObjectInputStream ois = new ObjectInputStream(new ByteArrayInputStream(userLicense.getLicense()));  dataMap  = (Map)ois.readObject(); |

經過上面的處理，既可以清楚的看到保存在dataMap中的license資訊：



**06. 創建新license文件**

icense資訊是存放在dataMap中的，所以可以讀取dataMap並修改值，最簡單的方法就是直接變成full版本，或者設置成永久試用。

試用版的dataMap的key並不全，部分key暫時不清楚其作用，還有一些隱藏key。

[?](http://my.oschina.net/zoakerc/blog/309381)

|  |  |
| --- | --- |
|  | //設置啟動時的列印注釋  dataMap.put("Comment","\*\*\* Use for study only! \*\*\*");  //設置為商業full版本  dataMap.put("commercial","true");  //還有其他的一些隱藏欄位  //...... |

把dataMap列化到檔中就可以生成了一個新的license檔。

可以通過下面的步驟進行序列化UserLicense

[?](http://my.oschina.net/zoakerc/blog/309381)

|  |  |
| --- | --- |
|  | ByteArrayOutputStream bos = new ByteArrayOutputStream();  ObjectOutputStream oos = new ObjectOutputStream(bos);  oos.writeObject(dataMap);  byte[] licenseBuf = bos.toByteArray();  userLicense.setLicense(licenseBuf);  //簽名：userLicense.setSignature(signature);  ObjectOutputStream oos = new ObjectOutputStream(new BufferedOutputStream(new FileOutputStream("c:/jrebel-crack.lic")));  oos.writeObject(userLicense); |

**07. 破解signature驗證**

大家都知道，非對稱加密演算法主要是兩個用途：一是簽名（用自己的金鑰簽名，對方用自己的公開金鑰驗簽），二是加密（用對方的公開金鑰加密，對方用自己的金鑰解密）。要破解簽名，我能想到無非有以下的幾種方式：

a> 獲取對方的金鑰（幾乎不可能）

b> 瞭解license檔的結構，然後偽造簽名。比如：自己產生一對公開金鑰和私密金鑰，然後用自己的金鑰對license進行簽名，把簽名資訊放入到license中，同時用自己的公開金鑰替換原有的公開金鑰

c> 找到驗證簽名的地方，讓驗簽結果始終為true（偽造的license也可以通過驗簽），這種方式的難度在於找到驗證簽名的地方。

破解就是採用這種方式，也是目前破解JRebel最簡單的方式。

言歸正傳，簡單查看了使用者UserLicense的dataMap的結構，接下來就需要查找有哪些類使用了UserLicense類（驗證簽名肯定需要用到這個類）。最簡單的方法是：把jrebel.jar全部反編譯處理，Windows大小寫敏感，會提示是否更改，點全部重命名即可，然後查找關鍵字**UserLicense**

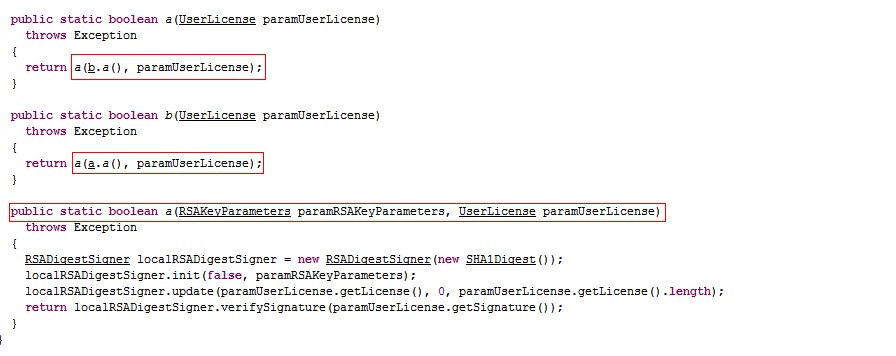
查找的結果只有幾個類使用到了UserLicense類，再在這幾個類中查找關鍵字**getSignature()**，經過篩選，發現最終只剩下1個類（對於不的版本，經過混淆處理後類名不一樣）。

對於jrebel 5.2.2是：com.zeroturnaround.javarebel.Av

對於jrebel 5.6.2是：com.zeroturnaround.javarebel.tP

對於jrebel 5.6.3是：com.zeroturnaround.javarebel.tR

接下來就是讓驗簽的那個方法返回true即可。如果細心摸索，會發現所有返回boolean方法都要調用一個最基本的方法，那麼我們只需要修改這個方法返回true即可，也就是修改用到了getSignature()得到UserLicense實體類的簽名的那個方法。



對於Class檔的修改，這時候就需要用到javassit或者cglib或者ASM。這裡使用javassit方法對Class檔進行修改，其他修改Class方法研究中。

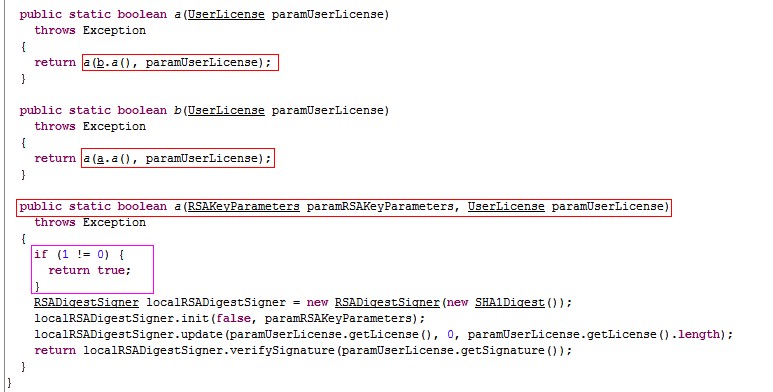
**08. 使用javassit修改並生成class檔**

下載javassit的jar包，在Netbeans之類的IDE中新建一個Java項目，並導入javassit.jar。新建一個Main.java，以jrebel 5.6.3為例。

[?](http://my.oschina.net/zoakerc/blog/309381)

|  |  |
| --- | --- |
|  | import javassist.ClassPool;  import javassist.CtClass;  import javassist.CtMethod;  public class Main {  public static void main(String[] args) throws Exception {  ClassPool pool = ClassPool.getDefault();  //取得需要反編譯的jar檔，設定路徑  pool.insertClassPath("c:/jrebel.jar");  //取得需要反編譯修改的檔，注意是完整路徑  CtClass cc1 = pool.get("com.zeroturnaround.javarebel.tR");  //5.6.2 CtClass cc1 = pool.get("com.zeroturnaround.javarebel.tP");  CtClass[] params = new CtClass[2];  //方法對應的參數  params[0] = pool.get("com.zeroturnaround.bundled.org.bouncycastle.crypto.params.RSAKeyParameters");  params[1] = pool.get("com.zeroturnaround.licensing.UserLicense");  //取得需要修改的方法  CtMethod method = cc1.getDeclaredMethod("a", params);  //插入修改項，我們讓他直接返回true(注意：根據方法的具體返回值返回，因為這個方法返回值是boolean，所以直接return true;)  method.insertBefore("if(1!=0)return true;");  //寫入保存  cc1.writeFile();  //jar uvf jrebel.jar com/zeroturnaround/javarebel/tR.class |

成功運行後，會在你新建的專案目錄下生成對應包名的class檔，使用jd-gui打開生成的**tR.class**後會發現我們的代碼已經添加成功。



**09. 替換jar包中的class文件**

替換jar 包下面的class 檔，很多人會想到直接用winrar 打開替換，在一般的情況下，是可行的，但是如果說這個jar 的代碼經過混淆後，會有大小寫不同，檔案名是相同的，在windos下檔案名是不區分大小寫的。如果直接用winrar替換的話，你會發現，替換的並非是你想替換的那個檔。

現在有2中可以行的方案：

a> 在Linux下把jar包解壓，替換，打成jar。這樣比較麻煩。

b> 可以直接用Java jar 工具來替換。

jar uvf test.jar test.class

這樣會直接把test.class 直接添加到jar包的根目錄。

jar uvf test.jar com/test/test.class

這樣就可以替換相應目錄的class檔了。

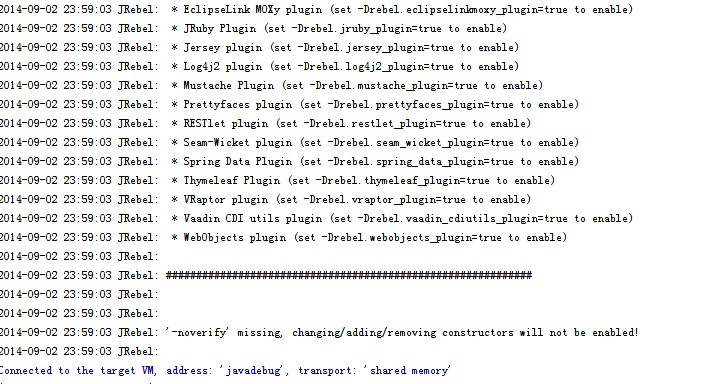
這裡值得注意的是  test.class 必須放在com/test 檔下，要和jar的路徑對應起來，不然會找不到這個檔或目錄，jar包和com資料夾的上級在同一個目錄。

回到我們生成的tR.class的最頂層包資料夾，使用CMD命令**jar uvf jrebel.jar com/zeroturnaround/javarebel/tR.class**替換掉原jar包中的tR.class

**10. 替換idea外掛程式中對應檔**

將我們生成的jrebel.jar和jrebel.lic替換掉原來的檔，生成的jrebel.lic也可以拷貝一份到jrebel.jar所在目錄。

重新啟動IDEA，會發現jrebel已經破解成功！



來自hobby的博客  <http://my.oschina.net/zoakerc/blog/309381>