java基于asm插桩之idea插件鉴权逻辑探索实战

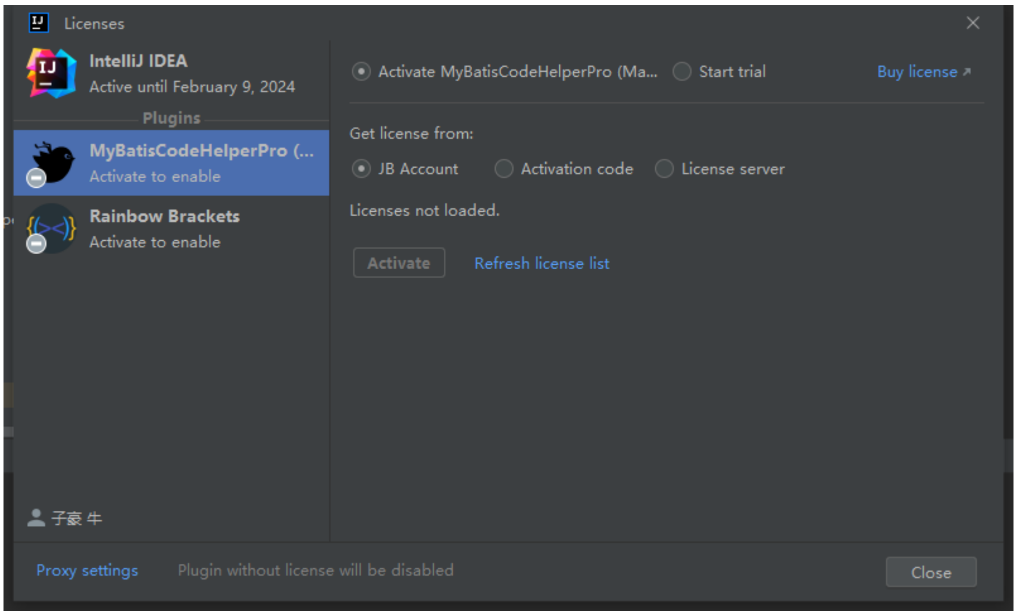
引言

1. 本次分享仅限用于学习和研究目的，不得将本次分享的内容用于商业或非法用途。
2. 理论→分析→实战。
3. idea如何离线对插件进行鉴权。
4. 插桩技术如何敢于idea插件鉴权机制。

依附于idea的插件鉴权体系

有些插件使用的是idea的收费体系。

即使在没有网络的环境中，idea也会弹出license鉴权弹窗。


Licenses

IntelliJ IDEA
Active until February 9, 2024

Plugins

MyBatisCodeHelperPro (...
Activate to enable

Rainbow Brackets
Activate to enable

子豪 牛

[✓ Activate MyBatisCodeHelperPro (Ma...] Start trial Buy license ↗

Get license from:

[✓ JB Account] Activation code License server

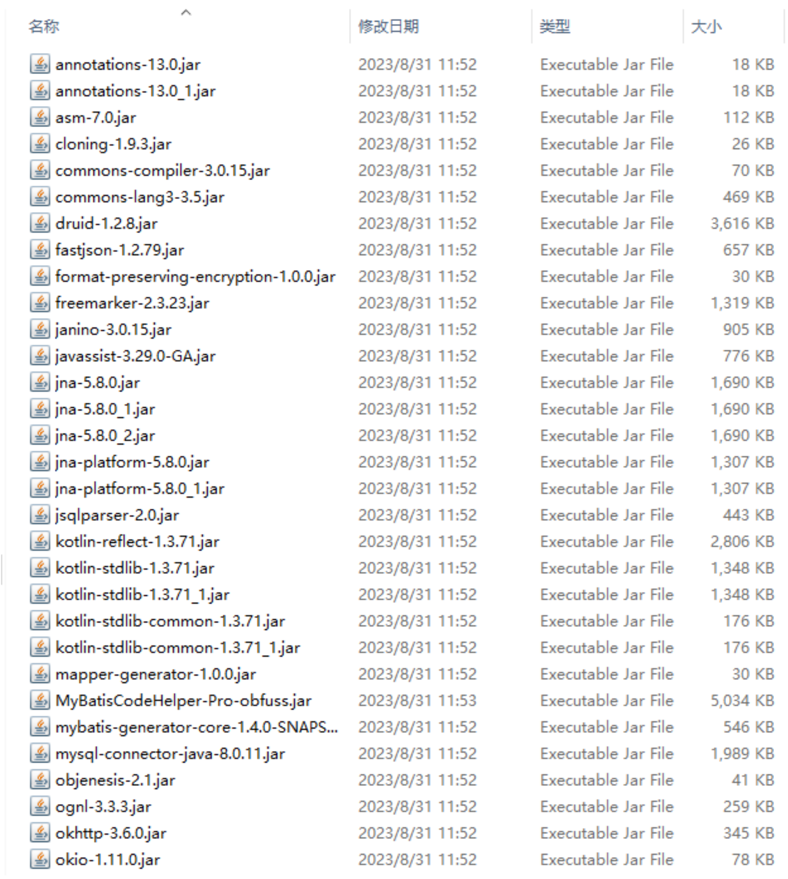
Licenses not loaded.

Activate Refresh license list

Proxy settings Plugin without license will be disabled Close

在没有网络的环境中，idea也能检测到这个插件需要license，说明与之相关的元数据就在插件安装包中。

看一下安装包结构



名称	修改日期	类型	大小
annotations-13.0.jar	2023/8/31 11:52	Executable Jar File	18 KB
annotations-13.0_1.jar	2023/8/31 11:52	Executable Jar File	18 KB
asm-7.0.jar	2023/8/31 11:52	Executable Jar File	112 KB
cloning-1.9.3.jar	2023/8/31 11:52	Executable Jar File	26 KB
commons-compiler-3.0.15.jar	2023/8/31 11:52	Executable Jar File	70 KB
commons-lang3-3.5.jar	2023/8/31 11:52	Executable Jar File	469 KB
druid-1.2.8.jar	2023/8/31 11:52	Executable Jar File	3,616 KB
fastjson-1.2.79.jar	2023/8/31 11:52	Executable Jar File	657 KB
format-preserving-encryption-1.0.0.jar	2023/8/31 11:52	Executable Jar File	30 KB
freemarker-2.3.23.jar	2023/8/31 11:52	Executable Jar File	1,319 KB
janino-3.0.15.jar	2023/8/31 11:52	Executable Jar File	905 KB
javassist-3.29.0-GA.jar	2023/8/31 11:52	Executable Jar File	776 KB
jna-5.8.0.jar	2023/8/31 11:52	Executable Jar File	1,690 KB
jna-5.8.0_1.jar	2023/8/31 11:52	Executable Jar File	1,690 KB
jna-5.8.0_2.jar	2023/8/31 11:52	Executable Jar File	1,690 KB
jna-platform-5.8.0.jar	2023/8/31 11:52	Executable Jar File	1,307 KB
jna-platform-5.8.0_1.jar	2023/8/31 11:52	Executable Jar File	1,307 KB
jsqlparser-2.0.jar	2023/8/31 11:52	Executable Jar File	443 KB
kotlin-reflect-1.3.71.jar	2023/8/31 11:52	Executable Jar File	2,806 KB
kotlin-stdlib-1.3.71.jar	2023/8/31 11:52	Executable Jar File	1,348 KB
kotlin-stdlib-1.3.71_1.jar	2023/8/31 11:52	Executable Jar File	1,348 KB
kotlin-stdlib-common-1.3.71.jar	2023/8/31 11:52	Executable Jar File	176 KB
kotlin-stdlib-common-1.3.71_1.jar	2023/8/31 11:52	Executable Jar File	176 KB
mapper-generator-1.0.0.jar	2023/8/31 11:52	Executable Jar File	30 KB
MyBatisCodeHelper-Pro-obfuss.jar	2023/8/31 11:53	Executable Jar File	5,034 KB
mybatis-generator-core-1.4.0-SNAPS...	2023/8/31 11:52	Executable Jar File	546 KB
mysql-connector-java-8.0.11.jar	2023/8/31 11:52	Executable Jar File	1,989 KB
objenesis-2.1.jar	2023/8/31 11:52	Executable Jar File	41 KB
ognl-3.3.3.jar	2023/8/31 11:52	Executable Jar File	259 KB
okhttp-3.6.0.jar	2023/8/31 11:52	Executable Jar File	345 KB
okio-1.11.0.jar	2023/8/31 11:52	Executable Jar File	78 KB

安装包里就是一些jar包

说明插件是通过提供一批jar来实现的，既然是插件，一定就有启动入口，有入口肯定就有主jar。

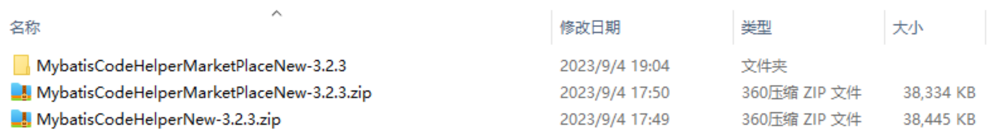
寻找主jar，一眼就能看出`MyBatisCodeHelper-Pro-obfuss.jar`是目标jar，因为他名字含有obfuss，都告我我们这个jar混淆了

插件化开发，加载插件一般都有固定的切入点作为入口，比如：java-spi、实现特定结构、类添加特定注解、元数据文件。

最常见的是通过元数据文件来定义插件信息，探索一下，很容易找到文件是`META-INF/plugins.xml`。

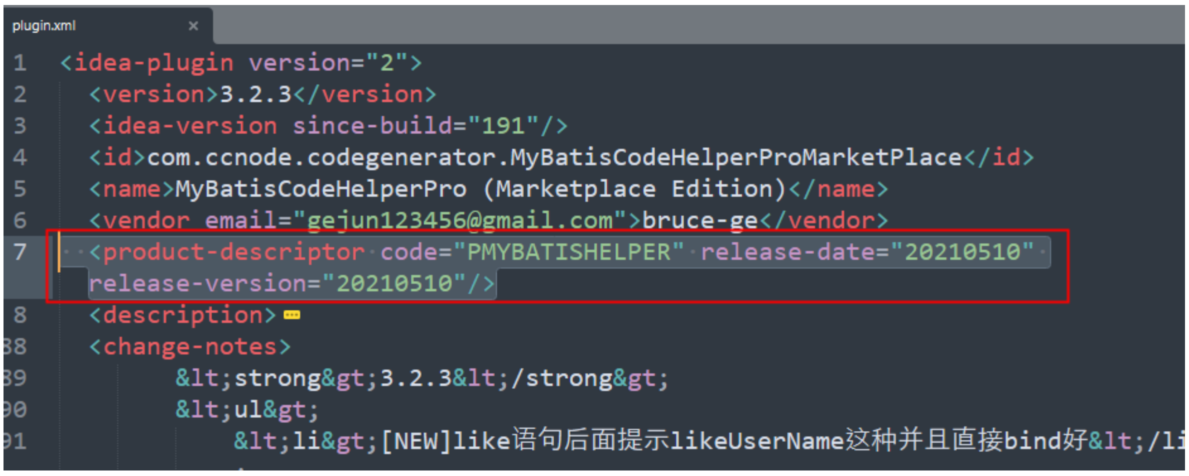
查看此文件内容，很难看出哪个xml节点与收费相关。

很幸运的是，`MyBatisCodeHelper-Pro`也有不依赖插件市场的版本。



名称	修改日期	类型	大小
MybatisCodeHelperMarketPlaceNew-3.2.3	2023/9/4 19:04	文件夹	
MybatisCodeHelperMarketPlaceNew-3.2.3.zip	2023/9/4 17:50	360压缩 ZIP 文件	38,334 KB
MybatisCodeHelperNew-3.2.3.zip	2023/9/4 17:49	360压缩 ZIP 文件	38,445 KB

对比一下两个版本的插件元数据文件，发现插件市场版本多了`product-descriptor`节点。



plugin.xml

```
<idea-plugin version="2">
  <version>3.2.3</version>
  <idea-version since-build="191"/>
  <id>com.ccnode.codegenerator.MyBatisCodeHelperProMarketPlace</id>
  <name>MyBatisCodeHelperPro (Marketplace Edition)</name>
  <vendor email="gejun123456@gmail.com">bruce-ge</vendor>
  <product-descriptor code="PMYBATISHELPER" release-date="20210510"
  release-version="20210510"/>
  <description>
  <change-notes>
        &lt;strong&gt;3.2.3&lt;/strong&gt;
        &lt;ul&gt;
            &lt;li&gt;[NEW]like语句后面提示likeUserName这种并且直接bind好&lt;/li
```

将此节点删除，重新打包成jar包，安装修改后的插件。

安装后，idea不再弹出license鉴权弹窗，但插件无法使用。

说明插件自身也有鉴权机制，只是将鉴权入口交给了idea。

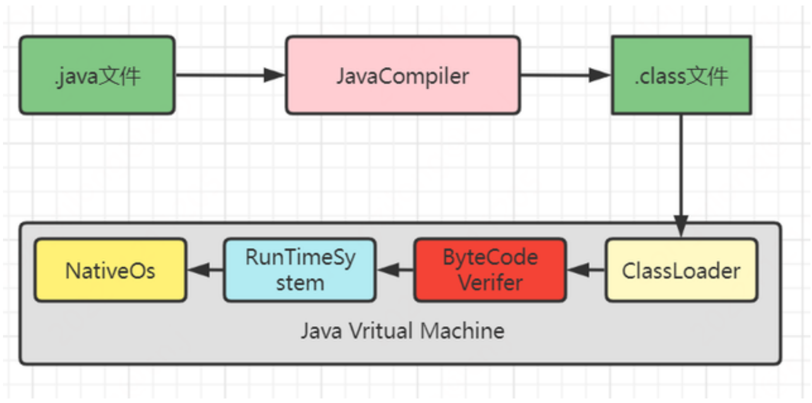
字节码

字节码是一种中间状态的二进制文件，是由源码编译过来的，可读性没有源码的高。

cpu并不能直接读取字节码，在java中，字节码需要经过JVM转译成机器码之后，cpu才能读取并运行。

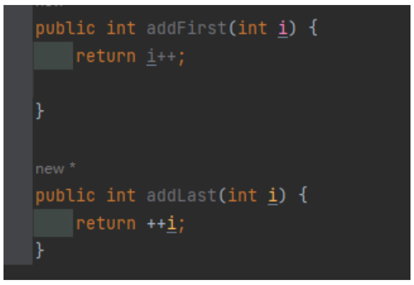
使用字节码的好处：一处编译，到处运行。java就是典型的使用字节码作为中间语言，在一个地方编译了源码，拿着.class文件就可以在各种计算机运行。

JVM屏蔽/封装了底层OS的差异。



从字节码角度分析i++和++i的差异

代码



```
public int addFirst(int i) {
    return i++;

}

new *
public int addLast(int i) {
    return ++i;
}
```

```
// access flags 0x1
public addFirst(I)I
  // parameter  i
 L0
  LINENUMBER 16 L0
  ILOAD 1
  IINC 1 1
  IRETURN
 L1
  LOCALVARIABLE this Lcom/dapeng/flow/FlowApplication; L0 L1 0
  LOCALVARIABLE i I L0 L1 1
  MAXSTACK = 1
  MAXLOCALS = 2

// access flags 0x1
public addLast(I)I
  // parameter  i
 L0
  LINENUMBER 21 L0
  IINC 1 1
  ILOAD 1
  IRETURN
 L1
  LOCALVARIABLE this Lcom/dapeng/flow/FlowApplication; L0 L1 0
  LOCALVARIABLE i I L0 L1 1
  MAXSTACK = 1
  MAXLOCALS = 2
```

字节码增强

如果我们不想修改源码，但是又想加入新功能，让程序按照我们的预期去运行，可以通过编译过程和加载过程中去做相应的操作。

简单来讲就是：将生成的.class文件修改或者替换称为我们需要的目标.class文件。

由于字节码增强可以在完全不侵入业务代码的情况下植入代码逻辑，所以可以用它来做一些酷酷的事，比如下面的几种常见场景：

1、动态代理

2、热部署

3、调用链跟踪埋点

4、动态插入log(性能监控)

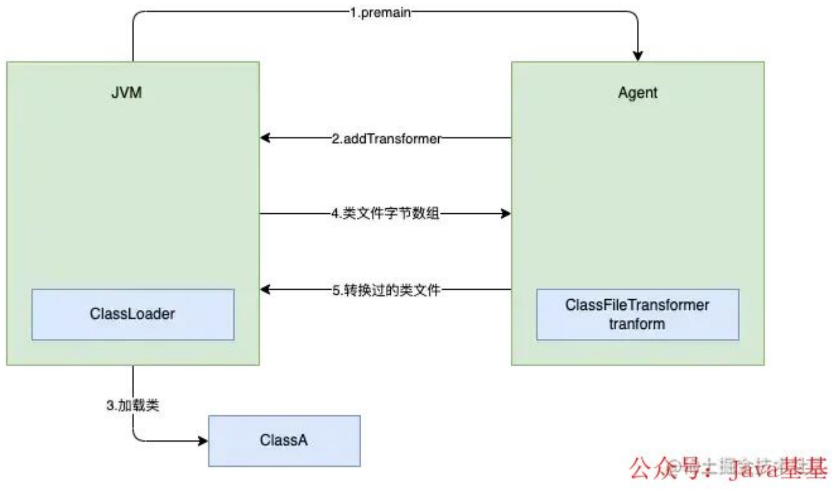
5、测试代码覆盖率跟踪

6、修改特定方法的入参、出参、执行逻辑。

javaagent

javaagent是JVM提供的可以在运行时修改类字节码的机制。
支持在类加载时修改类字节码，或者是在运行时修改类字节码后重载类。
流程：

1. JVM调用agent的premain方法进行初始化
2. agent将自己的 ClassFileTransformer 注册到JVM中
3. JVM加载类时，将字节码交给agent的 ClassFileTransformer
4. ClassFileTransformer 返回修改后的字节码数据
5. JVM用 ClassFileTransformer 返回的数据进行类加载



idea插件插桩实战

dhook

本次分享使用的javaagent框架是基于开源组件dhook二次开发的。
支持以下功能：

1. 可以通过配置文件指定切入点和执行逻辑，支持正则表达式
2. 支持记录方法入参和出参
3. 支持修改方法入参
4. 支持修改方法出参
5. 支持打印方法调用堆栈
6. 支持代理方法
7. 支持直接方法字节码

配置文件

```
{
// 是否打印JVM加载的所有类
  "printAllClassName": false,
  // 是否将dhook修改过的类，导出到磁盘上
  "dumpTransformedClass": true,
  // 指定导出哪些类
  "dumpClasses": [

  ],
  // 类名以这些开头的类，直接跳过，加快启动速度
  "skipClassPrefixes": [
    "java/",
    "javax/",
    "jdk/",
    "com/keven1z/",
    "shaded/"
  ],
  // 插桩逻辑
  "hooks": [
    {
    // 是否启用此规则
      "enable": true,
      // 类名匹配，以~开头表示使用正则表达式匹配
      "className": "com/ccnode/codegenerator/af/a/b",
      // 方法签名匹配，以~开头表示使用正则表达式匹配
      "desc": "~.*Z",
      // 方法名匹配，以~开头表示使用正则表达式匹配
      "method": "~.*",
      // 是否记录方法参数
      "logArgs": true,
      // 是否记录方法返回值
      "logReturnValue": false,
      // 修改方法入参
      "parameters": null,
      // 修改方法返回值
      "returnValue": "true",
      // 打印方法堆栈
      "printStackTrace": false,
      // 方法代理
      "proxyMethod": null,
      // 替换方法体字节码
      "byteCodes": null
    }
  ]
}
```

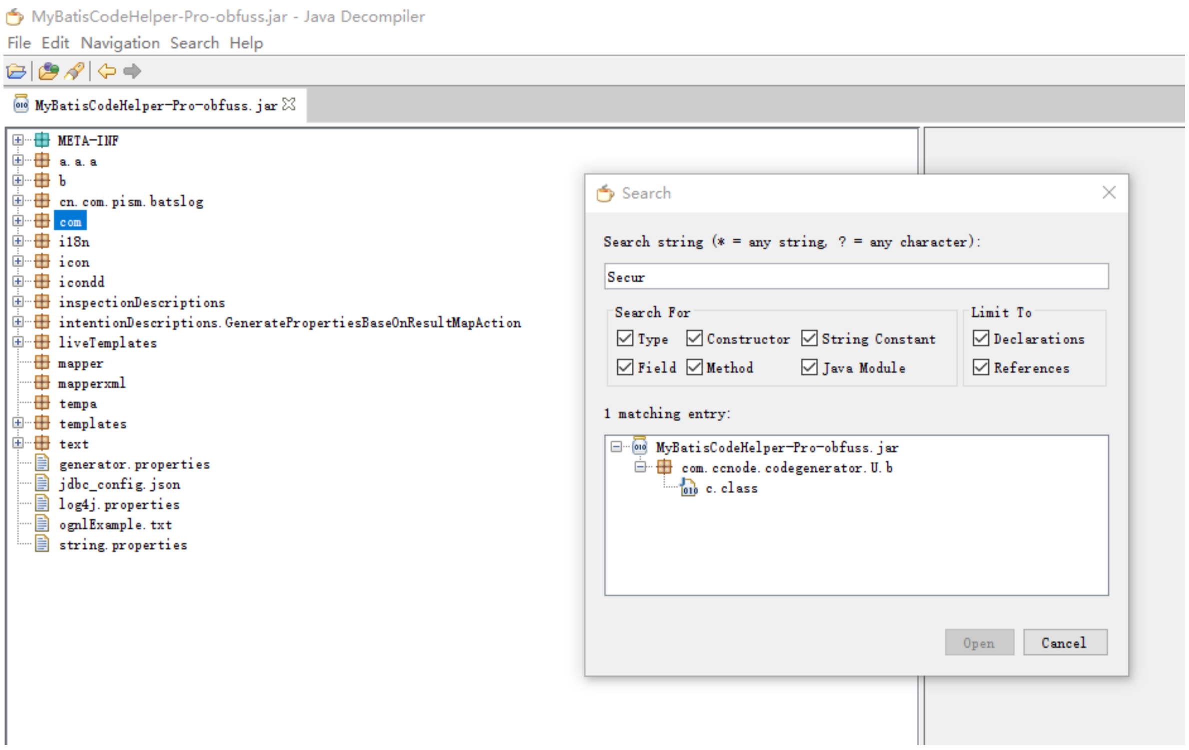
实战

本次分享使用`MyBatisCodeHelper-Pro`的插件市场版作为案例进行实战，其他插件基本类似。

使用`jd-gui`反编译`MyBatisCodeHelper-Pro-obfuss.jar`

基于license进行授权，通常会使用RSA加密。

搜索jar中使用RSA的类



```
MyBatisCodeHelper-Pro-obfuss.jar
  META-INF
  a.a.a
  b
  cn.com.pism.batslog
  com
  i18n
  icon
  icondd
  inspectionDescriptions
  intentionDescriptions.GeneratePropertiesBaseOnResultMapAction
  liveTemplates
  mapper
  mapperxml
  tempa
  templates
  text
  generator.properties
  jdbc_config.json
  log4j.properties
  ognlExample.txt
  string.properties
```

```
Search string (* = any string, ? = any character):
Secur
Search For: [✓ Type] [✓ Constructor] [✓ String Constant] [✓ Field] [✓ Method] [✓ Java Module]
Limit To: [✓ Declarations] [✓ References]
1 matching entry:
MyBatisCodeHelper-Pro-obfuss.jar
  com.ccnode.codegenerator.U.b
    c.class
```

找到了1个类，查看内容，发现其中确实是执行RSA加密的代码。

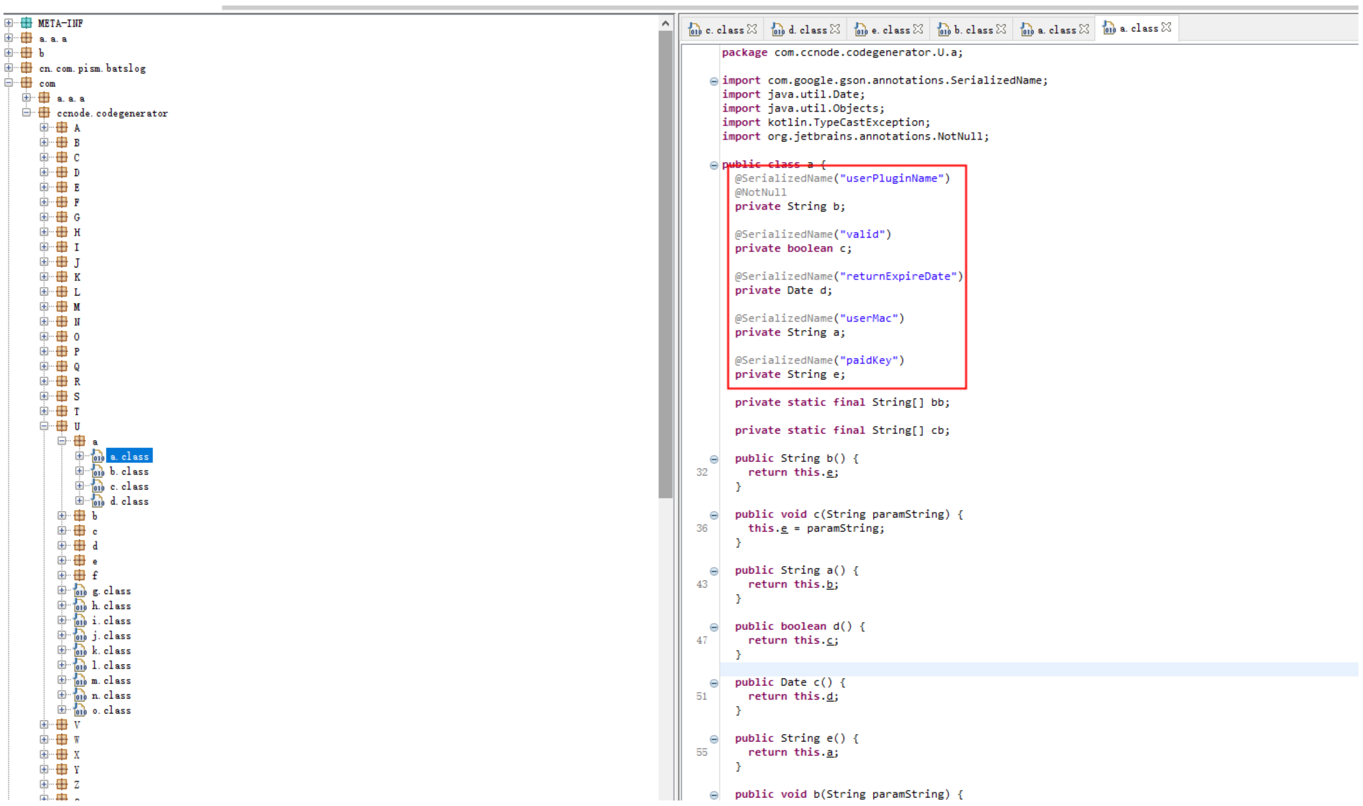
如果直接对RSA加密进行破解，首先需要探索出license的结构，其次需要修改class中的公钥。

这里采取更简单地方式。

判断插件是否授权，结果一定是个boolean值。

能否找到一个方法，修改其返回的boolean值，就能让插件认为已经授权了。

在类`com.ccnode.codegenerator.U.b.c`附近的包里探索一下，看有没有可疑的地方。



```
package com.ccnode.codegenerator.U.a;

import com.google.gson.annotations.SerializedName;
import java.util.Date;
import java.util.Objects;
import kotlin.TypeCastException;
import org.jetbrains.annotations.NotNull;

public class a {
  @SerializedName("userPluginName")
  @NotNull
  private String b;

  @SerializedName("valid")
  private boolean c;

  @SerializedName("returnExpireDate")
  private Date d;

  @SerializedName("userMac")
  private String a;

  @SerializedName("paidKey")
  private String e;

  private static final String[] bb;

  private static final String[] cb;

  public String b() {
    return this.e;
  }

  public void c(String paramString) {
    this.e = paramString;
  }

  public String a() {
    return this.b;
  }

  public boolean d() {
    return this.c;
  }

  public Date c() {
    return this.d;
  }

  public String e() {
    return this.a;
  }

  public void b(String paramString) {
```

确实找到了几处可疑的地方。因为需要将license等信息持久化，此插件选择使用注解来保持序列化的字段名不变。

实际搜索时，发现`com.ccnode.codegenerator.U`包下好几个类都有可疑代码，无法直接确定是哪个类。

下面使用插桩技术进行探索，直接将这个包下所有返回bool值得方法都进行插桩，打印出参和入参。

用网上开源的dHook组件进行二次开发，移除了在线功能，只保留离线功能，并对功能点进行了增强，修复了缺陷。

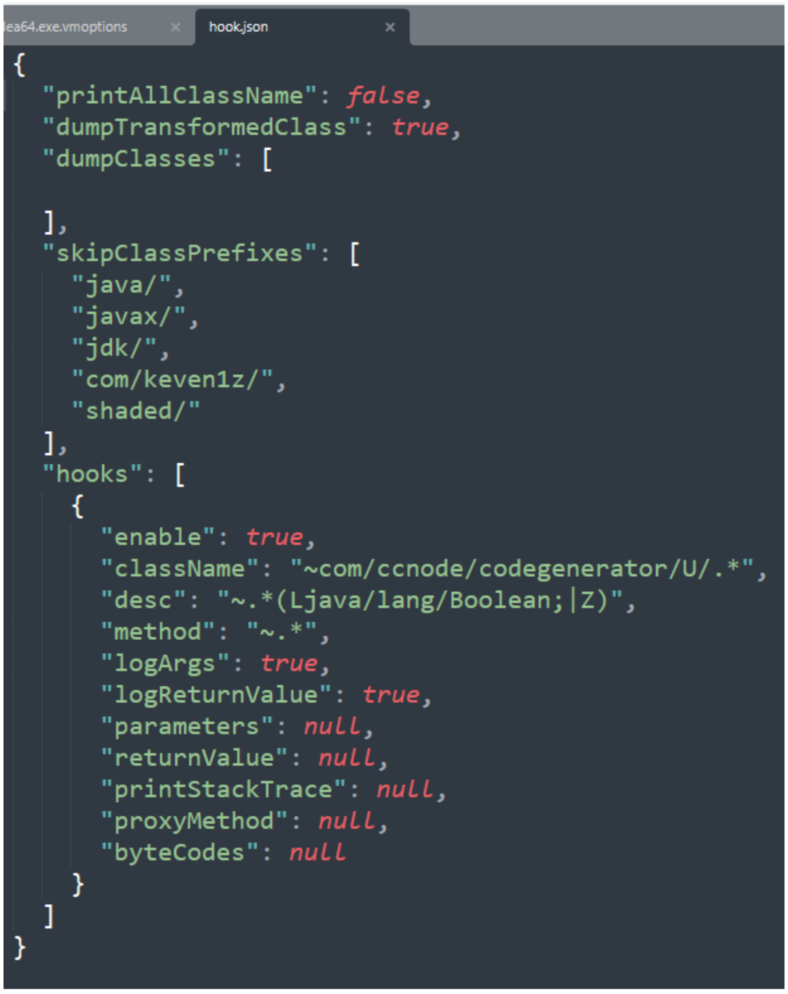
dHook会读取jar包同级的hook.json配置文件

编写配置文件，对上述方法进行插桩。

将dhook以javaagent的形式添加到idea64.exe.vmoptions中（一定要修改idea安装包下的vmoptions文件）

`-javaagent:D:\app\idea-hook\dHook.jar`

在cmd中调用idea.bat启动idea，观察日志



```json
{
  "printAllClassName": false,
  "dumpTransformedClass": true,
  "dumpClasses": [

  ],
  "skipClassPrefixes": [
    "java/",
    "javax/",
    "jdk/",
    "com/keven1z/",
    "shaded/"
  ],
  "hooks": [
    {
      "enable": true,
      "className": "~com/ccnode/codegenerator/U/.*",
      "desc": "~.*(Ljava/lang/Boolean;|Z)",
      "method": "~.*",
      "logArgs": true,
      "logReturnValue": true,
      "parameters": null,
      "returnValue": null,
      "printStackTrace": null,
      "proxyMethod": null,
      "byteCodes": null
    }
  ]
}
```

这个配置插桩后看不到相关日志，扩大插桩范围，对 com/ccnode/codegenerator 包下所有的方法进行插桩

```json
{
  "enable": true,
  "className": "~com/ccnode/codegenerator/.*",
  "desc": "~.*",
  "method": "~.*",
  "logArgs": false,
  "logReturnValue": true,
  "parameters": null,
  "returnValue": null,
  "printStackTrace": null,
  "proxyMethod": null,
  "byteCodes": null
}
```

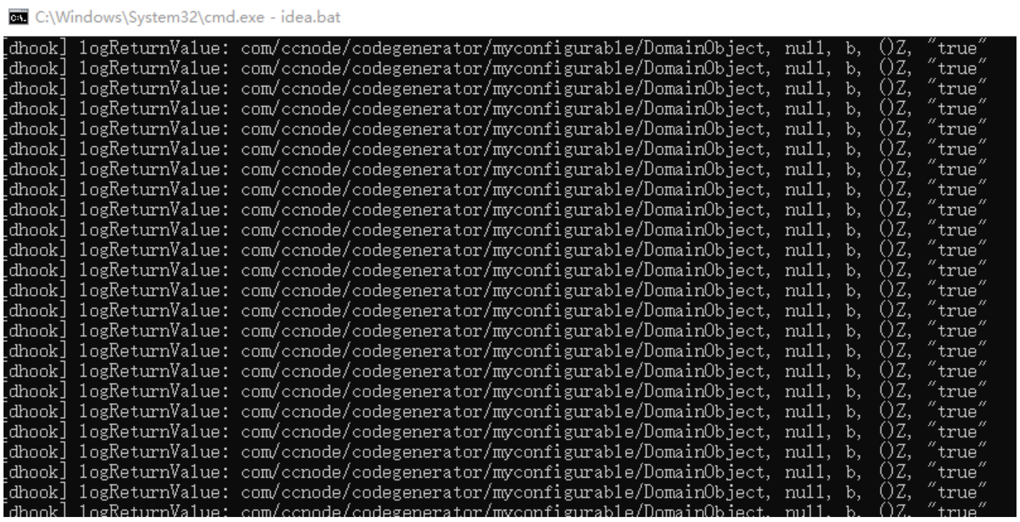
插桩后启动失败，类加载失败，字节码错误

说明dHook在有些情况处理并不完美

在报错信息中发现包`com.ccnode.codegenerator.myconfigurable`很可疑，查看类发现里面类名都没混淆

缩小插桩范围，只对出错的方法进行插桩

```json
{
  "enable": true,
  "className": "com.ccnode.codegenerator.myconfigurable.DomainObject",
  "desc": "()Z",
  "method": "b",
  "logArgs": false,
  "logReturnValue": true,
  "parameters": null,
  "returnValue": null,
  "printStackTrace": null,
  "proxyMethod": null,
  "byteCodes": null
}
```


C:\Windows\System32\cmd.exe - idea.bat

```
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/DomainObject, null, b, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/DomainObject, null, b, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/DomainObject, null, b, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/DomainObject, null, b, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/DomainObject, null, b, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/DomainObject, null, b, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/DomainObject, null, b, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/DomainObject, null, b, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/DomainObject, null, b, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/DomainObject, null, b, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/DomainObject, null, b, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/DomainObject, null, b, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/DomainObject, null, b, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/DomainObject, null, b, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/DomainObject, null, b, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/DomainObject, null, b, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/DomainObject, null, b, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/DomainObject, null, b, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/DomainObject, null, b, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/DomainObject, null, b, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/DomainObject, null, b, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/DomainObject, null, b, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/DomainObject, null, b, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/DomainObject, null, b, ()Z, "true"
```

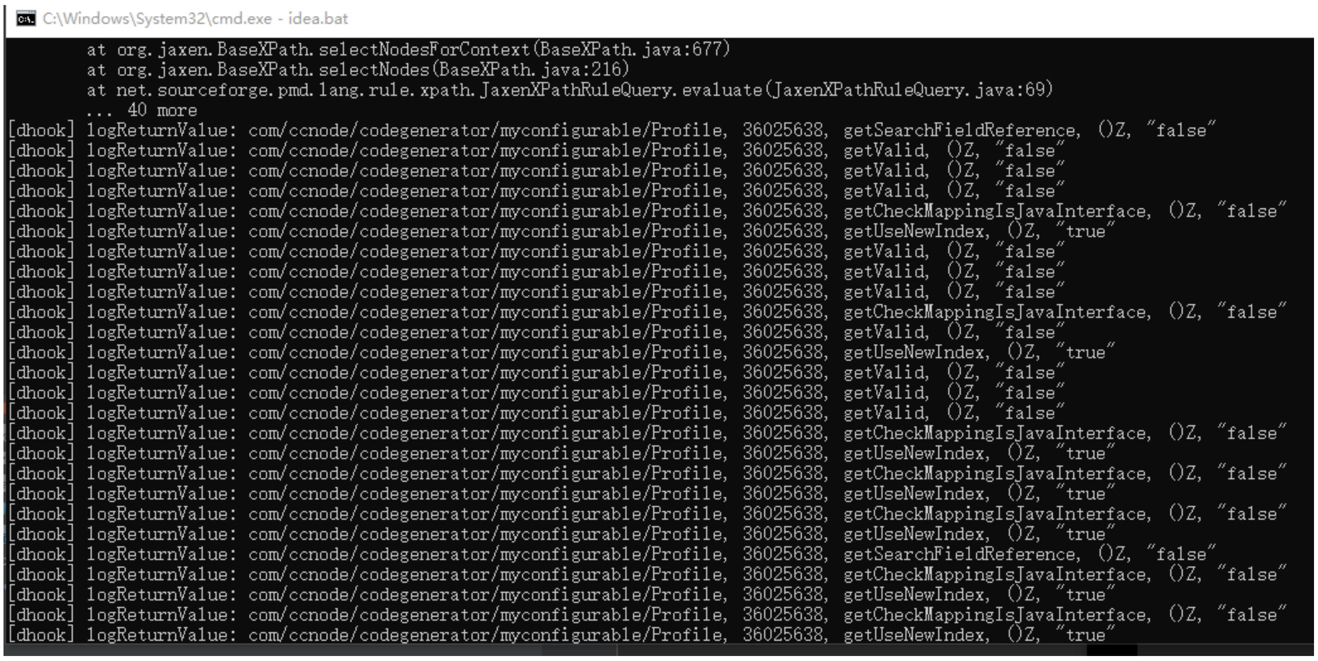
插桩成功，有很多日志输出，记录了方法返回值，说明插桩是成功的，可以放心的调整配置文件了

查看相同包下的类， 发现`com.ccnode.codegenerator.myconfigurable.Profile`类比较可疑

对其返回boolean的方法进行插桩，查看返回值

```
{
      "enable": true,
      "className": "com.ccnode.codegenerator.myconfigurable.Profile",
      "desc": "()Z",
      "method": "~.*",
      "logArgs": false,
      "logReturnValue": true,
      "parameters": null,
      "returnValue": null,
      "printStackTrace": null,
      "proxyMethod": null,
      "byteCodes": null
    }
```

发现相当可疑的方法调用



C:\Windows\System32\cmd.exe - idea.bat

```
        at org.jaxen.BaseXPath.selectNodesForContext(BaseXPath.java:677)
        at org.jaxen.BaseXPath.selectNodes(BaseXPath.java:216)
        at net.sourceforge.pmd.lang.rule.xpath.JaxenXPathRuleQuery.evaluate(JaxenXPathRuleQuery.java:69)
        ... 40 more
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/Profile, 36025638, getSearchFieldReference, ()Z, "false"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/Profile, 36025638, getValid, ()Z, "false"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/Profile, 36025638, getValid, ()Z, "false"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/Profile, 36025638, getValid, ()Z, "false"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/Profile, 36025638, getCheckMappingIsJavaInterface, ()Z, "false"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/Profile, 36025638, getUseNewIndex, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/Profile, 36025638, getValid, ()Z, "false"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/Profile, 36025638, getValid, ()Z, "false"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/Profile, 36025638, getValid, ()Z, "false"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/Profile, 36025638, getCheckMappingIsJavaInterface, ()Z, "false"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/Profile, 36025638, getValid, ()Z, "false"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/Profile, 36025638, getUseNewIndex, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/Profile, 36025638, getValid, ()Z, "false"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/Profile, 36025638, getValid, ()Z, "false"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/Profile, 36025638, getValid, ()Z, "false"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/Profile, 36025638, getCheckMappingIsJavaInterface, ()Z, "false"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/Profile, 36025638, getUseNewIndex, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/Profile, 36025638, getCheckMappingIsJavaInterface, ()Z, "false"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/Profile, 36025638, getValid, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/Profile, 36025638, getCheckMappingIsJavaInterface, ()Z, "false"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/Profile, 36025638, getUseNewIndex, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/Profile, 36025638, getSearchFieldReference, ()Z, "false"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/Profile, 36025638, getCheckMappingIsJavaInterface, ()Z, "false"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/Profile, 36025638, getUseNewIndex, ()Z, "true"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/Profile, 36025638, getCheckMappingIsJavaInterface, ()Z, "false"
[dhook] logReturnValue: com/ccnode/codegenerator/myconfigurable/Profile, 36025638, getUseNewIndex, ()Z, "true"
```

修改配置文件，将getValid方法返回值改为true

插桩成功，但依然没有绕过鉴权。

修改配置文件，打印getValid方法的调用堆栈

得到调用栈，发现用处不大

```
[dhook] logArgs: com/ccnode/codegenerator/myconfigurable/Profile, 362358827, getValid,
()Zjava.lang.RuntimeException: printStackTrace

        at com.ccnode.codegenerator.myconfigurable.Profile.getValid(Profile.kt)
        at com.ccnode.codegenerator.S.a.getLanguagesToInject(a.java:28)
        at
com.intellij.psi.impl.source.tree.injected.InjectedLanguageManagerImpl.processInPlaceInjectorsFor
(InjectedLanguageManagerImpl.java:442)
        at
com.intellij.psi.impl.source.tree.injected.InjectedLanguageUtilBase.probeElementsUp(InjectedLangu
ageUtilBase.java:246)
        at
com.intellij.psi.impl.source.tree.injected.InjectedLanguageUtilBase.enumerate(InjectedLanguageUti
lBase.java:176)
        at
com.intellij.psi.impl.source.tree.injected.InjectedLanguageUtilBase.enumerate(InjectedLanguageUti
lBase.java:146)
        at
com.intellij.psi.impl.source.tree.injected.InjectedLanguageUtilBase.hasInjections(InjectedLanguag
eUtilBase.java:540)
        at
com.intellij.util.xml.impl.GenericValueReferenceProvider.getReferencesByElement(GenericValueRefer
enceProvider.java:42)
        at
com.intellij.psi.impl.source.resolve.reference.ReferenceProvidersRegistryImpl.getReferences(Refer
enceProvidersRegistryImpl.java:182)
        at
com.intellij.psi.impl.source.resolve.reference.ReferenceProvidersRegistryImpl.mapNotEmptyReferenc
esFromProviders(ReferenceProvidersRegistryImpl.java:163)
        at
com.intellij.psi.impl.source.resolve.reference.ReferenceProvidersRegistryImpl.doGetReferencesFrom
Providers(ReferenceProvidersRegistryImpl.java:142)
        at
com.intellij.psi.impl.source.resolve.reference.ReferenceProvidersRegistry.getReferencesFromProvid
ers(ReferenceProvidersRegistry.java:43)
        at
com.intellij.psi.impl.source.xml.XmlTagDelegate.getReferencesImpl(XmlTagDelegate.java:163)
        at
com.intellij.psi.impl.source.xml.XmlTagDelegate.getDefaultReferences(XmlTagDelegate.java:129)
        at com.intellij.psi.impl.source.xml.XmlTagImpl.getReferences(XmlTagImpl.java:94)
        at
com.intellij.psi.PsiReferenceServiceImpl.doGetReferences(PsiReferenceServiceImpl.java:37)
        at
com.intellij.psi.PsiReferenceServiceImpl.getReferences(PsiReferenceServiceImpl.java:26)
        at
com.intellij.psi.search.SingleTargetRequestResultProcessor.processTextOccurrence(SingleTargetRequ
estResultProcessor.java:32)
        at
com.intellij.psi.impl.search.PsiSearchHelperImpl$5.lambda$execute$0(PsiSearchHelperImpl.java:962)
        at
com.intellij.psi.impl.search.LowLevelSearchUtil.processTreeUp(LowLevelSearchUtil.java:85)
        at
com.intellij.psi.impl.search.LowLevelSearchUtil.lambda$processElementsAtOffsets$0(LowLevelSearchU
til.java:176)
        at
com.intellij.psi.impl.search.LowLevelSearchUtil.processOffsets(LowLevelSearchUtil.java:203)
        at
```

```
com.intellij.psi.impl.search.LowLevelSearchUtil.processElementsAtOffsets(LowLevelSearchUtil.java:
175)
        at
com.intellij.psi.impl.search.PsiSearchHelperImpl$5.execute(PsiSearchHelperImpl.java:958)
        at
com.intellij.psi.impl.search.PsiSearchHelperImpl$2.processInReadAction(PsiSearchHelperImpl.java:2
83)
        at
com.intellij.psi.impl.search.PsiSearchHelperImpl$2.processInReadAction(PsiSearchHelperImpl.java:2
74)
        at
com.intellij.openapi.application.ReadActionProcessor.lambda$process$0(ReadActionProcessor.java:11
)
        at
com.intellij.openapi.application.impl.ApplicationImpl.runReadAction(ApplicationImpl.java:941)
        at com.intellij.openapi.application.ReadAction.compute(ReadAction.java:68)
        at
com.intellij.openapi.application.ReadActionProcessor.process(ReadActionProcessor.java:11)
        at
com.intellij.psi.impl.search.PsiSearchHelperImpl.lambda$processCandidates$18(PsiSearchHelperImpl.
java:919)
        at
com.intellij.psi.impl.search.PsiSearchHelperImpl.lambda$processVirtualFile$11(PsiSearchHelperImpl
.java:555)
        at
com.intellij.openapi.application.impl.ApplicationImpl.tryRunReadAction(ApplicationImpl.java:1154)
        at
com.intellij.psi.impl.search.PsiSearchHelperImpl.processVirtualFile(PsiSearchHelperImpl.java:534)
        at
com.intellij.psi.impl.search.PsiSearchHelperImpl.lambda$processPsiFileRoots$7(PsiSearchHelperImpl
.java:405)
        at
com.intellij.psi.impl.search.PsiSearchHelperImpl.lambda$processFilesConcurrentlyDespiteWriteActio
ns$9(PsiSearchHelperImpl.java:476)
        at
com.intellij.openapi.application.impl.ReadMostlyRWLock.executeByImpatientReader(ReadMostlyRWLock.
java:174)
        at
com.intellij.openapi.application.impl.ApplicationImpl.executeByImpatientReader(ApplicationImpl.ja
va:215)
        at
com.intellij.psi.impl.search.PsiSearchHelperImpl.lambda$processFilesConcurrentlyDespiteWriteActio
ns$10(PsiSearchHelperImpl.java:475)
        at
com.intellij.concurrency.ApplierCompleter.execAndForkSubTasks(ApplierCompleter.java:136)
        at
com.intellij.openapi.application.impl.ApplicationImpl.tryRunReadAction(ApplicationImpl.java:1154)
        at
com.intellij.concurrency.ApplierCompleter.lambda$wrapInReadActionAndIndicator$1(ApplierCompleter.
java:92)
        at
com.intellij.openapi.progress.impl.CoreProgressManager.lambda$executeProcessUnderProgress$12(Core
ProgressManager.java:608)
        at
com.intellij.openapi.progress.impl.CoreProgressManager.registerIndicatorAndRun(CoreProgressManage
r.java:683)
        at
com.intellij.openapi.progress.impl.CoreProgressManager.computeUnderProgress(CoreProgressManager.j
ava:639)
        at
com.intellij.openapi.progress.impl.CoreProgressManager.executeProcessUnderProgress(CoreProgressMa
nager.java:607)
        at
com.intellij.openapi.progress.impl.ProgressManagerImpl.executeProcessUnderProgress(ProgressManage
rImpl.java:60)
        at
com.intellij.concurrency.ApplierCompleter.wrapInReadActionAndIndicator(ApplierCompleter.java:104)
        at com.intellij.concurrency.ApplierCompleter.lambda$compute$0(ApplierCompleter.java:83)
        at
com.intellij.openapi.application.impl.ReadMostlyRWLock.executeByImpatientReader(ReadMostlyRWLock.
java:174)
        at
com.intellij.openapi.application.impl.ApplicationImpl.executeByImpatientReader(ApplicationImpl.ja
va:215)
        at com.intellij.concurrency.ApplierCompleter.compute(ApplierCompleter.java:83)
        at java.base/java.util.concurrent.CountedCompleter.exec(CountedCompleter.java:754)
        at java.base/java.util.concurrent.ForkJoinTask.doExec(ForkJoinTask.java:373)
        at
java.base/java.util.concurrent.ForkJoinPool$WorkQueue.topLevelExec(ForkJoinPool.java:1182)
        at java.base/java.util.concurrent.ForkJoinPool.scan(ForkJoinPool.java:1655)
        at java.base/java.util.concurrent.ForkJoinPool.runWorker(ForkJoinPool.java:1622)
        at java.base/java.util.concurrent.ForkJoinWorkerThread.run(ForkJoinWorkerThread.java:165)
```

继续调整配置文件，对com/ccnode/codegenerator/myconfigurable包下所有返回boolean的方法进行插桩

```
 {
      "enable": true,
      "className": "~com/ccnode/codegenerator/myconfigurable/.*",
      "desc": "()Z",
      "method": "~.*",
      "logArgs": true,
      "logReturnValue": false,
      "parameters": null,
      "returnValue": null,
      "printStackTrace": false,
      "proxyMethod": null,
      "byteCodes": null
    }
```

发现每次点代码生成，都有如下方法调用（mybatis generate files）

```
[dhook] logArgs: com/ccnode/codegenerator/myconfigurable/DomainObject, null, b, ()Z
```

打印此方法的调用栈， 查看相关类，发现com.ccnode.codegenerator.af.a.b包比较可疑

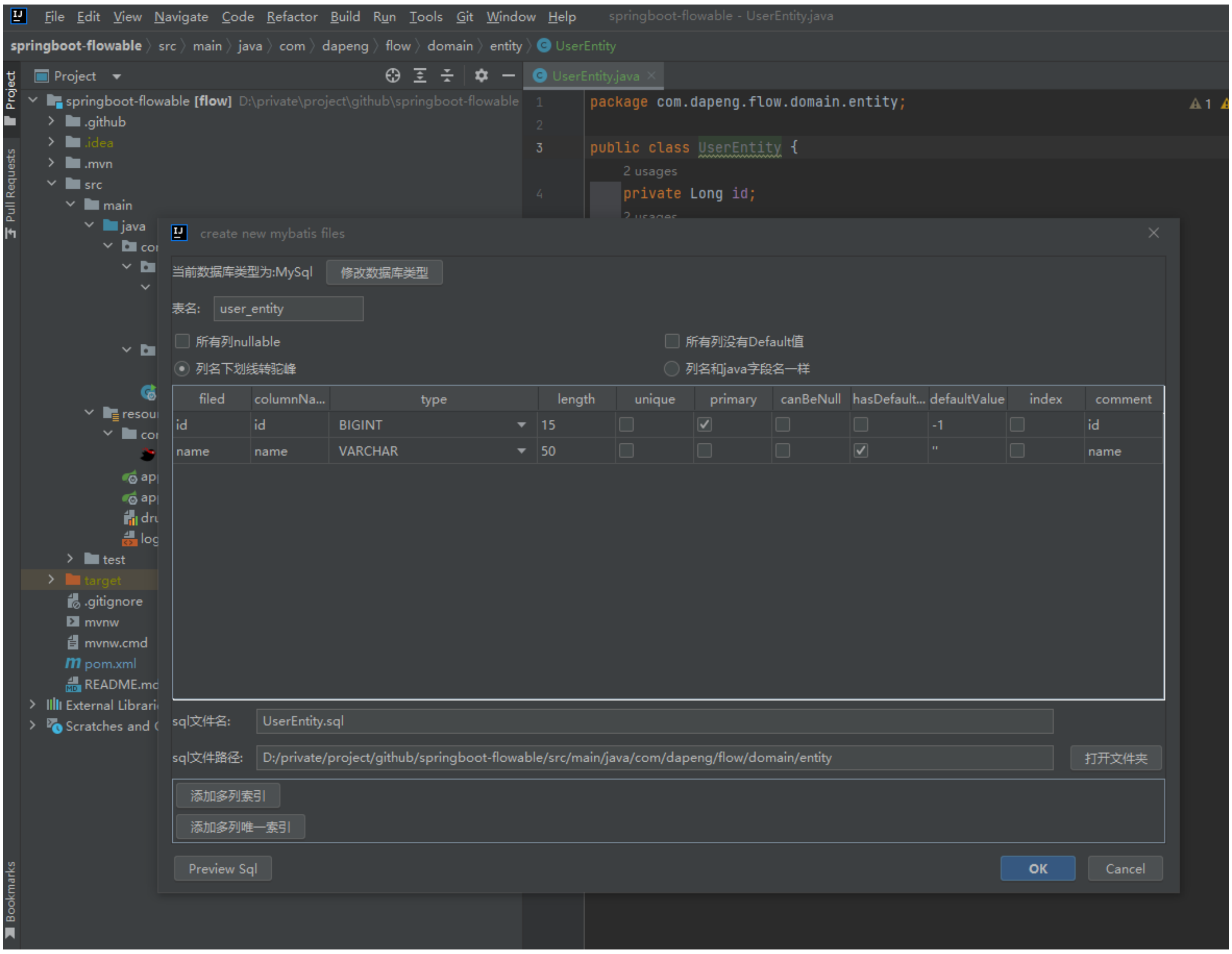
```
java.lang.RuntimeException: printStackTrace
	at com.ccnode.codegenerator.myconfigurable.DomainObject.b(DomainObject.java)
	at com.ccnode.codegenerator.af.a.b.a(b.java:59)
	at com.ccnode.codegenerator.ah.P.invoke(P.java:35)
	at 
com.intellij.codeInsight.actions.CodeInsightAction.lambda$actionPerformedImpl$0(CodeInsightAction
.java:68)
	at 
com.intellij.codeInsight.actions.CodeInsightAction.lambda$actionPerformedImpl$1(CodeInsightAction
.java:74)
	at 
com.intellij.openapi.command.impl.CoreCommandProcessor.executeCommand(CoreCommandProcessor.java:2
19)
	at 
com.intellij.openapi.command.impl.CoreCommandProcessor.executeCommand(CoreCommandProcessor.java:1
74)
	at 
com.intellij.openapi.command.impl.CoreCommandProcessor.executeCommand(CoreCommandProcessor.java:1
55)
	at 
com.intellij.codeInsight.actions.CodeInsightAction.actionPerformedImpl(CodeInsightAction.java:65)
	at 
com.intellij.codeInsight.actions.CodeInsightAction.actionPerformed(CodeInsightAction.java:40)
	at 
com.intellij.openapi.actionSystem.ex.ActionUtil.doPerformActionOrShowPopup(ActionUtil.java:315)
	at 
com.intellij.openapi.actionSystem.ex.ActionUtil.lambda$performActionDumbAwareWithCallbacks$4(Acti
onUtil.java:294)
	at 
com.intellij.openapi.actionSystem.ex.ActionUtil.performDumbAwareWithCallbacks(ActionUtil.java:337
)
	at 
com.intellij.openapi.actionSystem.ex.ActionUtil.performActionDumbAwareWithCallbacks(ActionUtil.ja
va:294)
	at com.intellij.openapi.actionSystem.ex.ActionUtil.invokeAction(ActionUtil.java:515)
	at com.intellij.ui.popup.ActionPopupStep.performAction(ActionPopupStep.java:232)
	at com.intellij.ui.popup.ActionPopupStep.lambda$onChosen$1(ActionPopupStep.java:220)
	at 
com.intellij.openapi.application.TransactionGuardImpl.performActivity(TransactionGuardImpl.java:1
05)
	at 
com.intellij.openapi.application.TransactionGuardImpl.performUserActivity(TransactionGuardImpl.ja
va:94)
	at com.intellij.ui.popup.AbstractPopup.lambda$dispose$18(AbstractPopup.java:1543)
	at 
com.intellij.util.ui.EdtInvocationManager.invokeLaterIfNeeded(EdtInvocationManager.java:113)
	at com.intellij.ide.IdeEventQueue.ifFocusEventsInTheQueue(IdeEventQueue.java:180)
	at 
com.intellij.ide.IdeEventQueue.executeWhenAllFocusEventsLeftTheQueue(IdeEventQueue.java:133)
	at 
com.intellij.openapi.wm.impl.FocusManagerImpl.doWhenFocusSettlesDown(FocusManagerImpl.java:164)
	at com.intellij.ui.popup.AbstractPopup.dispose(AbstractPopup.java:1540)
	at com.intellij.ui.popup.WizardPopup.dispose(WizardPopup.java:162)
	at com.intellij.ui.popup.list.ListPopupImpl.dispose(ListPopupImpl.java:326)
	at 
com.intellij.ui.popup.PopupFactoryImpl$ActionGroupPopup.dispose(PopupFactoryImpl.java:266)
	at com.intellij.openapi.util.ObjectTree.runWithTrace(ObjectTree.java:129)
```

```
        at com.intellij.openapi.util.ObjectTree.executeAll(ObjectTree.java:159)
        at com.intellij.openapi.util.Disposer.dispose(Disposer.java:219)
        at com.intellij.openapi.util.Disposer.dispose(Disposer.java:207)
        at com.intellij.ui.popup.WizardPopup.disposeAllParents(WizardPopup.java:266)
        at com.intellij.ui.popup.list.ListPopupImpl.handleNextStep(ListPopupImpl.java:434)
        at com.intellij.ui.popup.list.ListPopupImpl._handleSelect(ListPopupImpl.java:406)
        at com.intellij.ui.popup.list.ListPopupImpl.handleSelect(ListPopupImpl.java:361)
        at
com.intellij.ui.popup.PopupFactoryImpl$ActionGroupPopup.handleSelect(PopupFactoryImpl.java:278)
        at
com.intellij.ui.popup.list.ListPopupImpl$MyMouseListener.mouseReleased(ListPopupImpl.java:618)
        at java.desktop/java.awt.AWTEventMulticaster.mouseReleased(AWTEventMulticaster.java:298)
        at java.desktop/java.awt.Component.processMouseEvent(Component.java:6648)
        at java.desktop/javax.swing.JComponent.processMouseEvent(JComponent.java:3392)
        at
com.intellij.ui.popup.list.ListPopupImpl$MyList.processMouseEvent(ListPopupImpl.java:694)
        at java.desktop/java.awt.Component.processEvent(Component.java:6413)
        at java.desktop/java.awt.Container.processEvent(Container.java:2266)
        at java.desktop/java.awt.Component.dispatchEventImpl(Component.java:5022)
        at java.desktop/java.awt.Container.dispatchEventImpl(Container.java:2324)
        at java.desktop/java.awt.Component.dispatchEvent(Component.java:4854)
        at java.desktop/java.awt.LightweightDispatcher.retargetMouseEvent(Container.java:4948)
        at java.desktop/java.awt.LightweightDispatcher.processMouseEvent(Container.java:4575)
        at java.desktop/java.awt.LightweightDispatcher.dispatchEvent(Container.java:4516)
        at java.desktop/java.awt.Container.dispatchEventImpl(Container.java:2310)
        at java.desktop/java.awt.Window.dispatchEventImpl(Window.java:2802)
        at java.desktop/java.awt.Component.dispatchEvent(Component.java:4854)
        at java.desktop/java.awt.EventQueue.dispatchEventImpl(EventQueue.java:781)
        at java.desktop/java.awt.EventQueue$4.run(EventQueue.java:730)
        at java.desktop/java.awt.EventQueue$4.run(EventQueue.java:724)
        at java.base/java.security.AccessController.doPrivileged(AccessController.java:399)
        at
java.base/java.security.ProtectionDomain$JavaSecurityAccessImpl.doIntersectionPrivilege(Protectio
nDomain.java:86)
        at
java.base/java.security.ProtectionDomain$JavaSecurityAccessImpl.doIntersectionPrivilege(Protectio
nDomain.java:97)
        at java.desktop/java.awt.EventQueue$5.run(EventQueue.java:754)
        at java.desktop/java.awt.EventQueue$5.run(EventQueue.java:752)
        at java.base/java.security.AccessController.doPrivileged(AccessController.java:399)
        at
java.base/java.security.ProtectionDomain$JavaSecurityAccessImpl.doIntersectionPrivilege(Protectio
nDomain.java:86)
        at java.desktop/java.awt.EventQueue.dispatchEvent(EventQueue.java:751)
        at com.intellij.ide.IdeEventQueue.defaultDispatchEvent(IdeEventQueue.java:918)
        at com.intellij.ide.IdeEventQueue.dispatchMouseEvent(IdeEventQueue.java:840)
        at com.intellij.ide.IdeEventQueue._dispatchEvent(IdeEventQueue.java:763)
        at com.intellij.ide.IdeEventQueue.lambda$dispatchEvent$6(IdeEventQueue.java:450)
        at
com.intellij.openapi.progress.impl.CoreProgressManager.computePrioritized(CoreProgressManager.jav
a:791)
        at com.intellij.ide.IdeEventQueue.lambda$dispatchEvent$7(IdeEventQueue.java:449)
        at
com.intellij.openapi.application.TransactionGuardImpl.performActivity(TransactionGuardImpl.java:1
13)
        at com.intellij.ide.IdeEventQueue.performActivity(IdeEventQueue.java:624)
        at com.intellij.ide.IdeEventQueue.lambda$dispatchEvent$8(IdeEventQueue.java:447)
        at
```

```
com.intellij.openapi.application.impl.ApplicationImpl.runIntendedWriteActionOnCurrentThread(ApplicationImpl.java:881)
        at com.intellij.ide.IdeEventQueue.dispatchEvent(IdeEventQueue.java:493)
        at
java.desktop/java.awt.EventDispatchThread.pumpOneEventForFilters(EventDispatchThread.java:207)
        at
java.desktop/java.awt.EventDispatchThread.pumpEventsForFilter(EventDispatchThread.java:128)
        at
java.desktop/java.awt.EventDispatchThread.pumpEventsForHierarchy(EventDispatchThread.java:117)
        at java.desktop/java.awt.EventDispatchThread.pumpEvents(EventDispatchThread.java:113)
        at java.desktop/java.awt.EventDispatchThread.pumpEvents(EventDispatchThread.java:105)
        at java.desktop/java.awt.EventDispatchThread.run(EventDispatchThread.java:92)
```

查看反编译的代码，将`com.ccnode.codegenerator.af.a.b.a`的返回值修改为true

成功调起mybatis代码生成弹窗，说明license被绕过了



```java
package com.dapeng.flow.domain.entity;

public class UserEntity {
    2 usages
    private Long id;
```

create new mybatis files

当前数据库类型为:MySql [修改数据库类型]

表名: user_entity

☐ 所有列nullable　☐ 所有列没有Default值

[✓ 列名下划线转驼峰]　列名和java字段名一样

filed	columnNa...	type	length	unique	primary	canBeNull	hasDefault...	defaultValue	index	comment
id	id	BIGINT	15	☐	✓	☐	☐	-1	☐	id
name	name	VARCHAR	50	☐	☐	☐	✓	''	☐	name

sql文件名: UserEntity.sql

sql文件路径: D:/private/project/github/springboot-flowable/src/main/java/com/dapeng/flow/domain/entity

File Edit View Navigate Code Refactor Build Run Tools Git Window Help
springboot-flowable - UserEntity.java
springboot-flowable › src › main › java › com › dapeng › flow › domain › entity › UserEntity
Project
springboot-flowable [flow] D:\private\project\github\springboot-flowable
.github
.idea
.mvn
src
main
java
test
target
.gitignore
mvnw
mvnw.cmd
pom.xml
README.md
External Libraries
Scratches and Consoles
UserEntity.java
打开文件夹
添加多列索引
添加多列唯一索引
Preview Sql
OK
Cancel

总结

1. 人工搜索可疑代码进行插桩，确定目标范围
2. 大范围插桩，确定目标范围
3. 通过方法调用栈查找调用链路
4. 修改方法返回值进行探索

素材

包含dhook和插件安装包

http://172.16.6.60/share

账号/密码: share

目录 dhook

所有素材将在分享结束后24消失内删除，需要素材的尽快下载。