## CodeQL 闭源应用创建数据库 & SQL注入

## CodeQL 闭源应用创建数据库

对于闭源应用,无法通过maven、ant等方式构建项目,codeql也就没法创建数据库,下面根据尝试使用比较好的反编译到编译项目步骤。

### 反编译

尝试使用过jd-gui、procyon、idea的反编译包java-decompiler。jd-gui是不推荐的,在重新编译的时候问题最多,主要看看procyon和java-decompiler

#### 测试效果如下:

第一张是procyon,第二张是idea的反编译包,procyon得到的源码会由于类型没强转导致报错不可编译生成class文件

```
自
                  if (null != resources) {
49
                     while (resources.hasMoreElements())
                         final URL url = resources.nextElement();
50
    if ("jar".equals(url.getProtocol())) {
51
52
                             getFromJar(url, packageDirName, results);
53
54
                         else {
44
               if (null != resources) {
45
                  while(resources.hasMoreElements())
46
                     URL url = (URL)resources.nextElement();
47
                     if ("jar".equals(url.getProtocol())) {
48
                        getFromJar(url, packageDirName, results);
49
                     } else if ("file".equals(url.getProtocol())) {
50
                       getFromFile(classLoader, packageName, results, url);
```

在进行编译时procyon产生的错误为788个,并且部分错误会导致无法生成class文件

```
The method put(String, String) in the type Map<String,String> is not applicable for the arguments (String, Object)

786. ERROR in (at line 82)

.getTags().put("device", dataPoint.getLabels().get("Net er"));

787. ERROR in (at line 85)

.getTags().put("device", dataPoint.getLabels().get("D er"));

788. ERROR in (at line 85)

.getTags().put("device", dataPoint.getLabels().get("D er"));

788. ERROR in (at line 85)

.lists = lists.stream().filter(1 -> "Flat".equalsIgnoreCase(l.getType())).collect((Collector<? super Object, ?, tist<! '>>>)Collectors.tolist());

788. problems (788 errors)
```

idea的反编译包编译时产生219个错误,但有的虽然提示错误仍然能够编译生成 class文件。当然也会出现procyon可以编译出来文件而java-decompiler编译不成 功,所以后面可以结合使用。

#### 下载procyon.jar

使用方式如下, 指定保存的目录会自动创建。

```
1 rem 反编译class文件
2 java -jar procyon-decompiler-0.6-prerelease.jar xxxx.class -o ou t1
3 rem 反编译jar包
4 java -jar procyon-decompiler-0.6-prerelease.jar xxx.jar -o out2
```

java-decompiler, 指定保存的目录需要手动创建

```
1 rem 需要先创建保存的目录out_filename
2 java -cp "D:\IntelliJ IDEA 2019.3.5\plugins\java-decompiler\lib
```

```
\java-decompiler.jar" org.jetbrains.java.decompiler.main.decompiler.ConsoleDecompiler -dgs=true xxxxx.jar out_filename
```

### 编译

列出ecj和javac两种编译方式,推荐使用ecj,下载链接: https://mvnrepository.com/artifact/org.eclipse.jdt.core.compiler/ecj/4.6.1

```
有多个待编译的文件可以将其路径存放在文本中,使用 @file.txt 来指定, e cj 和 javac 都支持该功能
```

```
1 rem tokens表示路径级别,那么这里也就是 D:\codeql\databases\lib.zip.s
rc\demo 作为当前路径了
2 for /f tokens^=5*delims^=\ %i in ('dir/b/s D:\codeql\databases\l
ib.zip.src\demo\*.java')do @echo= .\%~j >>file.txt
```

#### 编译使用方式如下:

```
rem 使用ecj编译,指定编译时使用的java版本为8
java -jar D:\ecj-4.6.3.jar -8 demo.java
rem 使用ecj编译,批量编译,将要编译的文件路径存放在file.txt中
java -jar D:\ecj-4.6.3.jar -encoding UTF-8 -classpath xxx.jar -8
-warn:none -noExit @file.txt

rem 使用javac编译,单个java文件,-d选项指定生成.class文件的位置
"C:\jdk1.8.0_66\bin\javac.exe" -encoding UTF-8 -cp "lib\*" -d .
.\com\demo.java
rem 使用javac编译,批量编译,将要编译的文件路径存放在file.txt中
"C:\jdk1.8.0_66\bin\javac.exe" -encoding UTF-8 -cp "lib\*" -d .
@file.txt
```

ecj有些坑,在编译的时候,指定依赖包只能指定单个包,无法指定目录来自动扫描下面的所有包。这里只能将命令写入在 .cmd 中,执行 .cmd,并且需要添加 -noExit 参数,在codeql创建数据库时不加上会在ecj编译完成后直接退出导致无法成功创建。

#### 大致命令如下:

```
1.jar;./lib/xxx2.jar -8 -warn:none -noExit @file.txt
```

#### 最后通过如下来创建数据库:

```
D:\codeql.exe database create D:\codeql\databases\demo-database --language="java" --source-root=D:\codeql\demo --command="run.cm d"
```

#### 编写了对应的自动工具:

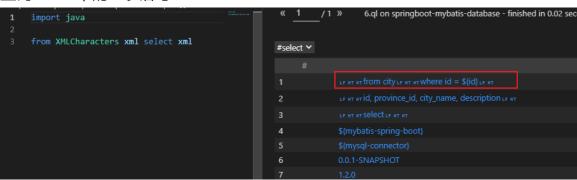
https://github.com/ice-doom/codeql compile

# SQL注入

当遇到 Mybatis XML 配置的场景,则通过如下方法(了解途径,可以阅读safe6Sec的文章):

在 codeql-cli/java/tools/pre-finalize.cmd 文件中插入 ——include "\*\*/resources/\*\*/\*.xml" , 这样创建数据库就能将 resources 下的xml文件都包含进来

#### 查询 XML 中的SQL语句



官方提供了相应的ql检测,使用 CWE-089 进行测试,https://github.com/github/codeql/pull/6319

#### 结果如下图所示:

深入学习CodeQL还得把官方提供的案例都过一遍,学!