

期望类需要满足的条件

- 不能被反序列化
- 存在setter方法
- setter方法操作的是全局变量

如何快速找到这个类？ codeql。

1、创建数据库

```
arch -x86_64 ./codeql database create
/Users/pen4uin/Workspace/CodeAnalysis/codeql/databases/dubbo/dubbo2_7_21 --
language="java" --command="mvn clean install -Dmaven.test.skip=true --file
pom.xml" --source-
root=/Users/pen4uin/Workspace/VulnerabilityResearch/dubbo/dubbo-dubbo-2.7.21 --
overwrite
```

- 坑: 这里需要删除子模块 `dubbo-serialization-protobuf`，否则会报错生成数据库失败

2、编写符合以上条件的ql规则

The screenshot shows the CodeQL IDE interface. On the left, a query rule is defined in a file named `isNotDeserializable`. The rule defines a predicate `isNotDeserializable(Class c)` that checks if a class is not deserializable based on its source type and qualified name. It also defines two classes: `SetterMethod` and `GlobalVariable`, which extend `Method` and `Field` respectively. The `SetterMethod` class checks for a setter method with a specific signature. The `GlobalVariable` class checks for a static field. On the right, the results of the query are displayed in a table with 4 results. The results are:

#	q
1	ConfigUtils
2	ApplicationModel
3	AdaptiveCompiler
4	ChannelHandlers

最终得到期望类: ConfigUtils

```
41 public class ConfigUtils {
42
43     private static final Logger logger = LoggerFactory.getLogger(ConfigUtils.class);
44     private static Pattern VARIABLE_PATTERN = Pattern.compile(
45         "\\$\\s*\\{?\\s*([\\._0-9a-zA-Z]+)\\s*\\}?"
46     );
47     private static volatile Properties PROPERTIES;
```

完整的ql规则

```
import java

predicate isNotDeserializable(Class c) {
    not exists (Interface i |
        c.getASourceSupertype*() = i.getSourceDeclaration() and
        i.hasQualifiedName("java.io", "Serializable")
    )
}

class SetterMethod extends Method {
    SetterMethod() {
        this.getName().indexOf("set") = 0 and
        this.getName().length() > 3 and
        this.getNumberOfParameters() = 1 and
        this.getReturnType() instanceof VoidType
    }
}

class GlobalVariable extends Field {
    GlobalVariable() {
        this.isStatic()
    }
}

class Query extends Class {
    Query() {
        isNotDeserializable(this) and
        exists (SetterMethod sm, GlobalVariable gv |
            sm.getDeclaringType() = this and
            sm.getBody().getAStmt().(ExprStmt).getExpr().(AssignExpr).getDest().
            (VarAccess).getVariable() = gv
        )
    }
}

from Query q
select q
```