链滴

《阿里巴巴 Java 开发手册》涉及安全部分讲解

作者：nanolikeyou

原文链接：https://ld246.com/article/1527928533067

来源网站：链滴

许可协议：署名-相同方式共享 4.0 国际 (CC BY-SA 4.0)

《阿里巴巴Java开发手册》可以作为编码风格、规范、要约的最佳实践，在阅读过程中，有一部分质方面的规范涉及安全，提取出来分享下。

0.**【推荐】类成员与方法访问控制从严**:

java的访问控制可以通过private、protect关键字，避免被外部反射机制调用实现。

1） 如果不允许外部直接通过 new 来创建对象，那么构造方法必须是 private。

如果构造方法是private，则外部new 时,编译器检查为The constructor PrivateTool() is not visible 需要设置为一定的权限，default则为package访问权限

```
public class PrivateTool {

private PrivateTool() {

// TODO Auto-generated constructor stub

}

}
```

2） 工具类不允许有 public 或 default 构造方法。

这样的工具类不够优雅

3） 类非 static 成员变量并且与子类共享，必须是 protected。

保持类与子类的成员变量的权限一致

4） 类非 static 成员变量并且仅在本类使用，必须是 private。

5） 类 static 成员变量如果仅在本类使用，必须是 private。

6） 若是 static 成员变量，必须考虑是否为 final。

7）类成员方法只供类内部调用，必须是 private。

8） 类成员方法只对继承类公开，那么限制为 protected。

以上比较详细的说明了关键字的使用方法。

1.**【推荐】避免 Random 实例被多线程使用，虽然共享该实例是线程安全的，但会因竞争同一 seed 导致的性能下降。**

java.util.Random 的实例或者 Math.random（）的seed唯一（），来自于系统时间的通过线性同余式产生，所以在获取大量的值时，攻击者可以预测seed值，继而计算出随机数。推荐使用java.security SecureRandom

2.**【推荐】在并发场景下，通过双重检查锁（double-checked locking）实现延迟初始化的优 化题隐患(可参考 The "Double-Checked Locking is Broken" Declaration)，推荐解 决方案中较为单一种（适用于 JDK5 及以上版本），将目标属性声明为 volatile 型***

参考https://race604.com/java-double-checked-singleton/?from=timeline

3.**【推荐】接口入参保护，这种场景常见的是用于做批量操作的接口。**

对接口入参必须进行校验，包括对外提供的开放接口，不管是 RPC/API/HTTP 接口。敏感权限入口。

【强制】对 trace/debug/info 级别的日志输出，必须使用条件输出形式或者使用占位符的方 式。

日志系统不使用占位符的话，可能发生CRLF注入的风险。

4.**【参考】可以使用 warn 日志级别来记录用户输入参数错误的情况，避免用户投诉时，无所适 从注意日志输出的级别，error 级别只记录系统逻辑出错、异常等重要的错误信息。如非必 要，请不要**

此场景打出 error 级别。

考虑到twitter的案例，遵循合法合规要求，需要避免在日志中记录银行卡、身份证、密码、token、cookie等敏感信息。参考案例：http://www.anquan.us/static/bugs/wooyun-2014-055359.html

5. **安全规约**

【强制】隶属于用户个人的页面或者功能必须进行权限控制校验。

目前的订单越权、遍历等web安全问题增多，需要对与此类平行权限问题予以重点考虑。

6. **【强制】用户敏感数据禁止直接展示，必须对展示数据进行脱敏。**

同时也需要对打码字段定义统一的标准，如果A应用对打码158****9119，而B应用只显示前7位，称打码不严，也是一种信息安全隐患。

7. **【强制】用户输入的 SQL 参数严格使用参数绑定或者 METADATA 字段值限定，防止 SQL 注入，禁止字符串拼接 SQL 访问数据库。**

参数绑定是ORM的说法，可以通过强制类型语言的特性避免一些恶意输入，底层采用JDBC的PrepareStatement预定义sql功能，本意是为了提高效率，因为mysql给字符串转移，所以达到了防止sql注的目的，参考https://blog.csdn.net/xieyuooo/article/details/10732375?utm_source=itdadao&um_medium=referral

8. **【强制】用户请求传入的任何参数必须做有效性验证。 说明：忽略参数校验可能导致： page size 过大导致内存溢出 恶意 order by 导致数据库慢查询 任意重定向 SQL 注入 反序列化注入 正则输入源串拒绝服务 ReDoS 说明：Java 代码用正则来验证客户端的输入，有些正则写法验证普用户输入没有问题， 但是如果攻击人员使用的是特殊构造的字符串来验证，有可能导致死循环的结果。**

攻击人员的思路同开发迥然不同，会考虑破坏系统机密性、完整性、可用性的恶意触发点。通过污点析的技术判断代码污染源即是考虑到一切来源于用户的输入都是不可信的。

9. **【强制】禁止向 HTML 页面输出未经安全过滤或未正确转义的用户数据。**

实际使用中，可以使用json格式工具类往页面输出内容。xss的场景很复杂，采用过滤和转义手段时要采用组织内部的安全api或者专业的esapi，避免自己实现。

10. **【强制】表单、AJAX 提交必须执行 CSRF 安全过滤**

csrf、cors跨域资源配置不当的安全问题逐步增多，目前对于表单提交的csrf需要强烈关注，其实logincsrf，查询、删除时的csrf也需要适当处理。

11. **【强制】在使用平台资源，譬如短信、邮件、电话、下单、支付，必须实现正确的防重放限制，数量限制、疲劳度控制、验证码校验，避免被滥刷、资损。**

验证码可以在短信网关限制单人单天不超过一定的次数，通过要建立短信模板，避免攻击者越权自定短信内容带来安全风险。验证码可以采用6位数字+字母的组合，并及时销毁。

12. **【推荐】发贴、评论、发送即时消息等用户生成内容的场景必须实现防刷、文本内容违禁词过滤风控策略。**

文本、图片等内容安全已经有了多种开放服务，可以快速接入保证风控效果。

13. **【强制】sql.xml 配置参数使用：#{}，#param# 不要使用${} 此种方式容易出现 SQL 注入。**

实践中遇到的问题是有业务会模糊查询、动态sql，这种情况需要进行过滤。