

<<Java安全编码标准>>

图书基本信息

书名：<<Java安全编码标准>>

13位ISBN编号：9787111428183

10位ISBN编号：7111428188

出版时间：2013-6

出版时间：机械工业出版社

作者：Fred Long,Dhruv Mohindra,Robert C. Seacord,Dean F. Sutherland,David Svoboda

译者：计文柯,杨晓春

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

<<Java安全编码标准>>

内容概要

《java安全编码标准》是java安全编码领域最权威、最全面、最详细的著作，java之父james a. gosling推荐。

不仅从语言角度系统而详细地阐述java安全编码的要素、标准、规范和最佳实践，而且从架构设计的角度分析了java api存在的设计缺陷和可能存在的安全风险，以及应对的策略和措施。

可以将本书作为java安全方面的工具书，根据自己的需要，找到自己感兴趣的规则进行阅读和理解，或者在实际开发中遇到安全问题时，根据书中列出的大致分类对规则进行索引和阅读，也可以通读全书的所有规则，系统地了解java安全规则，增强对java安全特性、语言使用、运行环境特性的理解。

本书能指导java软件工程师设计出高质量的、安全的、可靠的、强大的、有弹性的、可用性和可维护性高的软件系统。

<<Java安全编码标准>>

作者简介

Fred Long 英国Aberystwyth大学计算机科学系高级讲师和教学主任。
主要讲授形式方法、Java、C++和C的编程模式以及与编程相关的安全问题的课程。
他是英国计算机协会中威尔士分会的主席，自1992年以来在软件工程研究所（SEI）担任客座研究员。
最近正在研究如何在Java中探查安全性漏洞。

Dhruv Mohindra 印度Persistent系统工程有限公司的高级软件工程师。
曾研发了广泛应用于企业服务器的监控软件。
曾在SEI的CERT项目工作，并致力于在编程社区中提高对安全问题的警觉性。
曾任职于卡内基·梅隆大学，拥有信息安全策略与管理硕士学位和印度Pune大学计算机工程学士学位。

Robert C. Seacord 资深计算机安全专家和作家。
在计算机安全、历史系统改造以及基于组件的软件工程等领域具有极深的造诣。
目前管理卡内基·梅隆大学SEI的CERT在安全编码领域的创新项目。
拥有Rensselaer Polytechnic学院计算机科学学士学位。

Dean F. Sutherland CERT高级软件安全工程师，编译器后端技术专家组高级专家。
拥有卡内基·梅隆大学博士学位。
曾担任职业软件工程师，在Tartan公司工作超过14年。

David Svoboda CERT软件安全工程师，资深Java开发工程师，在Java开发领域拥有13年的开发经验。
是卡内基·梅隆大学的一系列软件开发项目的主要开发者，这些项目涉及从层级芯片建模到社会组织仿真再到自动机器学习等多个方面。

<<Java安全编码标准>>

书籍目录

《java安全编码标准》

译者序

序

前言

致谢

第1章 概述1

1.1 错位的信任1

1.2 注入攻击2

1.3 敏感数据泄露3

1.4 效能泄露5

1.5 拒绝服务6

1.6 序列化8

1.7 并发性、可见性和内存8

1.8 最低权限原则14

1.9 安全管理器15

1.10 类装载器16

1.11 小结16

第2章 输入验证和数据净化 (ids) 17

规则17

风险评估概要17

.2.1 ids00-j净化穿越受信边界的非受信数据18

2.2 ids01-j验证前标准化字符串26

2.3 ids02-j在验证之前标准化路径名28

2.4 ids03-j不要记录未经净化的用户输入31

2.5 ids04-j限制传递给zipinputstream的文件大小33

2.6 ids05-j使用ascii字符集的子集作为文件名和路径名35

2.7 ids06-j从格式字符串中排除用户输入37

2.8 ids07-j不要向runtime.exec()?方法传递非受信、未净化的数据38

2.9 ids08-j净化传递给正则表达式的非受信数据41

2.10 ds09-j如果没有指定适当的locale, 不要使用locale相关方法处理与locale相关的数据44

2.11 ids10-j不要拆分两种数据结构中的字符串45

2.12 ids11-j在验证前去掉非字符码点50

2.13 ids12-j在不同的字符编码中无损转换字符串数据51

2.14 ids13-j在文件或者网络i/o两端使用兼容的编码方式53

第3章 声明和初始化 (dcl) 56

规则56

风险评估概要56

3.1 dcl00-j防止类的循环初始化56

3.2 dcl01-j不要重用java标准库的已经公开的标识59

3.3 dcl02-j将所有增强for语句的循环变量声明为final类型60

第4章 表达式 (exp) 63

规则63

风险评估概要63

4.1 exp00-j不要忽略方法的返回值63

4.2 exp01-j不要解引用空指针65

<<Java安全编码标准>>

4.3 exp02-j使用两个参数的arrays.equals()方法来比较两个数组的内容67

4.4 exp03-j不要用相等操作符来比较两个基础数据类型的值67

4.5 exp04-j确保使用正确的类型来自动封装数值72

4.6 exp05-j不要在一个表达式中对同一变量进行多次写入73

4.7 exp06-j不要在断言中使用有副作用的表达式76

第5章 数值类型与运算 (num) 78

规则78

风险评估概要78

5.1 num00-j检测和避免整数溢出79

5.2 num01-j不要对同一数据进行位运算和数学运算85

5.3 num02-j确保除法运算和模运算中的除数不为088

5.4 num03-j使用可容纳无符号数据合法取值范围的整数类型89

5.5 num04-j不要使用浮点数进行精细计算90

5.6 num05-j不要使用非标准化数92

5.7 num06-j使用strictfp修饰符确保跨平台浮点运算的一致性94

5.8 num07-j不要尝试与nan进行比较97

5.9 num08-j检查浮点输入特殊的数值98

5.10 num09-j不要使用浮点变量作为循环计数器100

5.11 num10-j不要从浮点字元构造bigdecimal对象101

5.12 num11-j不要比较或者审查以字符串表达的浮点数值102

5.13 num12-j确保将数值转换成较小类型时不会产生数据丢失或曲解103

5.14 num13-j转换基本整数类型至浮点类型时应避免精度损失107

第6章 面向对象 (obj) 110

规则110

风险评估概要110

6.1 obj00-j只有受信子类能对具有不变性的类和方法进行扩展111

6.2 obj01-j声明数据成员为私有并提供可访问的封装器方法116

6.3 obj02-j当改变基类时, 保存子类之间的依赖关系118

6.4 obj03-j在新代码中, 不要混用具有泛型和非泛型的原始数据类型124

6.5 obj04-j为可变类提供复制功能, 并通过此功能允许将实例传递给非受信代码128

6.6 obj05-j在返回引用之前, 防御性复制私有的可变的类成员132

6.7 obj06-j对可变输入和可变的内部组件创建防御性复制136

6.8 obj07-j不允许敏感类复制其自身138

6.9 obj08-j不要在嵌套类中暴露外部类的私有字段141

6.10 obj09-j比较类而不是类名称143

6.11 obj10-j不要使用公有静态的非final变量144

6.12 obj11-j小心处理构造函数抛出异常的情况146

第7章 方法 (met) 153

规则153

风险评估概要153

7.1 met00-j验证方法参数154

7.2 met01-j不要使用断言验证方法参数156

7.3 met02-j不要使用弃用的或过时的类和方法157

7.4 met03-j进行安全检测的方法必须声明为private或final158

7.5 met04-j不要增加被覆写方法和被隐藏方法的可访问性160

7.6 met05-j确保构造函数不会调用可覆写的方法161

7.7 met06-j不要在clone()中调用可覆写的方法163

<<Java安全编码标准>>

7.8 met07-j不要定义类方法来隐藏基类或基类接口中声明的方法165

7.9 met08-j确保比较等同的对象能得到相等的结果167

7.10 met09-j定义了equals()方法的类必须定义hashCode()方法174

7.11 met10-j实现compareTo()方法时遵守常规合约176

7.12 met11-j确保比较中的关键码是不可变的178

7.13 met12-j不要使用析构函数182

第8章 异常行为 (err) 187

规则187

风险评估概要187

8.1 err00-j不要消除或忽略可检查的异常187

8.2 err01-j不能允许异常泄露敏感信息192

8.3 err02-j记录日志时应避免异常196

8.4 err03-j在方法失败时恢复对象先前的状态197

8.5 err04-j不要在finally程序段非正常退出201

8.6 err05-j不要在finally程序段中遗漏可检查异常202

8.7 err06-j不要抛出未声明的可检查异常205

8.8 err07-j不要抛出runtimeexception、exception或throwable209

8.9 err08-j不要捕捉nullpointerexception或任何它的基类210

8.10 err09-j禁止非受信代码终止jvm216

第9章 可见性和原子性 (vna) 219

规则219

风险评估概要219

9.1 vna00-j当需要读取共享基础数据类型变量时，需要保证其可见性219

9.2 vna01-j保证对一个不可变对象的共享引用的可见性222

9.3 vna02-j保证对于共享变量的组合操作是原子性的225

9.4 vna03-j即使每一个方法都是相互独立并且是原子性的，也不要假设一组调用是原子性的230

9.5 vna04-j保证串联在一起的方法调用是原子性的235

9.6 vna05-j保证在读写64位的数值时的原子性239

第10章 锁 (lck) 241

规则241

风险评估概要241

10.1 lck00-j通过私有final锁对象可以同步那些与非受信代码交互的类242

10.2 lck01-j不要基于那些可能被重用的对象进行同步246

10.3 lck02-j不要基于那些通过getClass()返回的类对象来实现同步249

10.4 lck03-j不要基于高层并发对象的内置锁来实现同步252

10.5 lck04-j即使集合是可访问的，也不要基于集合视图使用同步253

10.6 lck05-j对那些可以被非受信代码修改的静态字段，需要同步进入255

10.7 lck06-j不要使用一个实例锁来保护共享静态数据256

10.8 lck07-j使用相同的方式请求和释放锁来避免死锁258

10.9 lck08-j在异常条件时，保证释放已经持有的锁266

10.10 lck09-j不要执行那些持有锁时会阻塞的操作270

10.11 lck10-j不要使用不正确形式的双重锁定检查惯用法273

10.12 lck11-j当使用那些不能对锁策略进行承诺的类时，避免使用客户端锁定277

第11章 线程api (thi) 282

规则282

风险评估概要282

11.1 thi00-j不要调用thread.run()282

<<Java安全编码标准>>

- 11.2 thi01-j不能调用threadgroup方法284
- 11.3 thi02-j通知所有等待中的线程而不是单一线程287
- 11.4 thi03-j始终在循环中调用wait()和await()方法292
- 11.5 thi04-j确保可以终止受阻线程295
- 11.6 thi05-j不要使用thread.stop()来终止线程300
- 第12章 线程池 (tps) 304
 - 规则304
 - 风险评估概要304
 - 12.1 tps00-j使用线程池处理流量突发以实现降低性能运行304
 - 12.2 tps01-j不要使用有限的线程池来执行相互依赖的任务307
 - 12.3 tps02-j确保提交至线程池的任务是可中断的312
 - 12.4 tps03-j确保线程池中正在执行的任务不会失败而不给出任何提示315
 - 12.5 tps04-j使用线程池时, 确保threadlocal变量可以重新初始化318
- 第13章 与线程安全相关的其他规则 (tsm) 323
 - 规则323
 - 风险评估概要323
 - 13.1 tsm00-j不要使用非线程安全方法来覆写线程安全方法323
 - 13.2 tsm01-j不要让this引用在创建对象时泄漏326
 - 13.3 tsm02-j不要在初始化类时使用后台线程332
 - 13.4 tsm03-j不要发布部分初始化的对象336
- 第14章 输入输出 (fio) 342
 - 规则342
 - 风险评估概要342
 - 14.1 fio00-j不要操作共享目录中的文件343
 - 14.2 fio01-j使用合适的访问权限创建文件351
 - 14.3 fio02-j发现并处理与文件相关的错误352
 - 14.4 fio03-j在终止前移除临时文件354
 - 14.5 fio04-j在不需要时关闭资源357
 - 14.6 fio05-j不要使用wrap()或duplicate()创建缓存, 并将这些缓存暴露给非受信代码361
 - 14.7 fio06-j不能在一个单独的inputstream上创建多个缓存区封装器364
 - 14.8 fio07-j不要让外部进程阻塞输入和输出流367
 - 14.9 fio08-j对读取一个字符或者字节的方法, 使用int类型的返回值370
 - 14.10 fio09-j不要使用write()方法输出超过0 ~ 255的整数372
 - 14.11 fio10-j使用read()方法保证填充一个数组373
 - 14.12 fio11-j不要将原始的二进制数据作为字符数据读入375
 - 14.13 fio12-j为小端数据的读写提供方法376
 - 14.14 fio13-j不要在受信边界之外记录敏感信息379
 - 14.15 fio14-j在程序终止时执行正确的清理动作381
- 第15章 序列化 (ser) 387
 - 规则387
 - 风险评估概要387
 - 15.1 ser00-j在类的演化过程中维护其序列化的兼容性388
 - 15.2 ser01-j不要偏离序列化方法的正确签名390
 - 15.3 ser02-j在将对象向信任边界之外发送时, 需要签名并且封装敏感对象392
 - 15.4 ser03-j不要序列化未经加密的敏感数据397
 - 15.5 ser04-j不要允许序列化和反序列化绕过安全管理器401
 - 15.6 ser05-j不要序列化内部类实例404

<<Java安全编码标准>>

- 15.7 ser06-j在反序列化时，对私有的可变的组件进行防御性复制405
- 15.8 ser07-j不要对实现定义的不可变因素使用默认的序列化格式406
- 15.9 ser08-j在从拥有特性的环境中进行反序列化之前最小化特权410
- 15.10 ser09-j不要从readobject()方法中调用可以被覆写的方法413
- 15.11 ser10-j在序列化时，避免出现内存和资源泄漏414
- 15.12 ser11-j防止覆盖外部化的对象415
- 第16章 平台安全性 (sec) 417
 - 规则417
 - 风险评估概要417
 - 16.1 sec00-j不要允许特权代码块越过受信边界泄露敏感信息417
 - 16.2 sec01-j不要在特权代码块中使用污染过的变量420
 - 16.3 sec02-j不要基于非受信源进行安全检查422
 - 16.4 sec03-j不要在允许非受信代码装载任意类之后装载受信类424
 - 16.5 sec04-j使用安全管理器检查来保护敏感操作426
 - 16.6 sec05-j不要使用反射来增加类、方法和字段的可访问性429
 - 16.7 sec06-j不要依赖于默认的由urlclassloader和java.util.jar提供的自动化签名检查434
 - 16.8 sec07-j当编写一个自定义的类装载器时调用基类的getpermissions()方法437
 - 16.9 sec08-j定义基于原生方法的封装器438
- 第17章 运行环境 (env) 441
 - 规则441
 - 风险评估概要441
 - 17.1 env00-j不要签名只执行非特权操作的代码441
 - 17.2 env01-j将所有安全敏感的代码置于单独一个jar包中，并且在签名之后封装它443
 - 17.3 env02-j不要信任环境变量的值446
 - 17.4 env03-j不要赋予危险的权限组合448
 - 17.5 env04-j不要关闭字节码验证功能451
 - 17.6 env05-j不要部署一个被远程监视的应用452
- 第18章 其他 (msc) 457
 - 规则457
 - 风险评估概要457
 - 18.1 msc00-j在交换安全数据时使用sslsocket而不是socket457
 - 18.2 msc01-j不要使用空的无限循环461
 - 18.3 msc02-j生成强随机数462
 - 18.4 msc03-j不要硬编码敏感信息464
 - 18.5 msc04-j防止内存泄漏466
 - 18.6 msc05-j不要耗尽堆空间473
 - 18.7 msc06-j当一个遍历正在进行时，不要修改它对应的集合477
 - 18.8 msc07-j防止多次实例化单例对象481
- 术语表490
- 参考资源497

<<Java安全编码标准>>

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>