

Pinpoint分析报告

项目名称	java_src-2018-11-16-10-25-14
首次分析时间	2018年11月16日 10:25:54
最近分析时间	2018年11月16日 10:25:54
报告范围	单次分析
本次分析时间	2018年11月16日 10:25:54
报告类型	已标注缺陷报告

目录

1. 缺陷统计

2. 漏洞报告

2.1 空指针解引用

2.2 “.”或“|”用于正则表达式

2.3 正则表达式DOS (ReDOS)

3. 未标注漏洞摘要

40
最新发现缺陷数

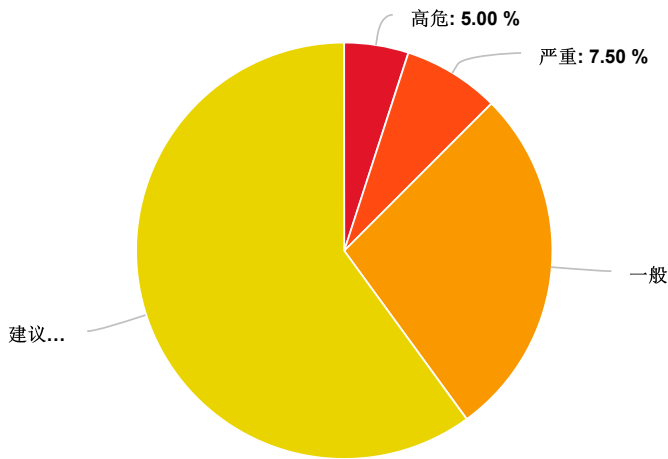
0
较上次未检出的缺陷

40
总缺陷

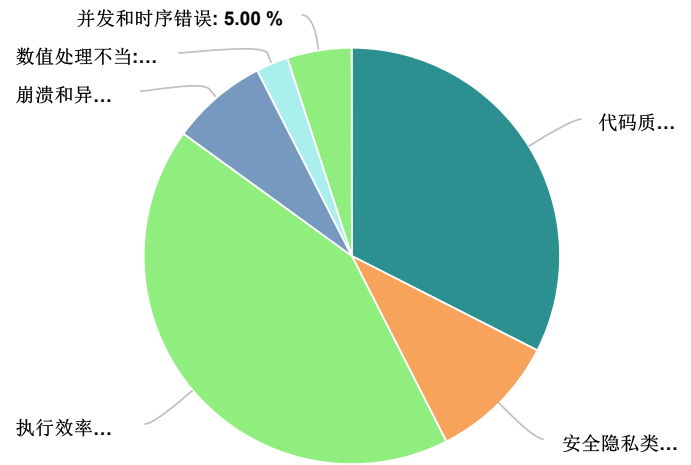
合格
项目评级

10617
代码行数

严重程度统计



漏洞分类统计



● 高危 - 2
● 建议改进 - 24

● 严重 - 3

● 一般 - 11

● 代码质量和风格 - 13

● 安全隐私类缺陷 - 4

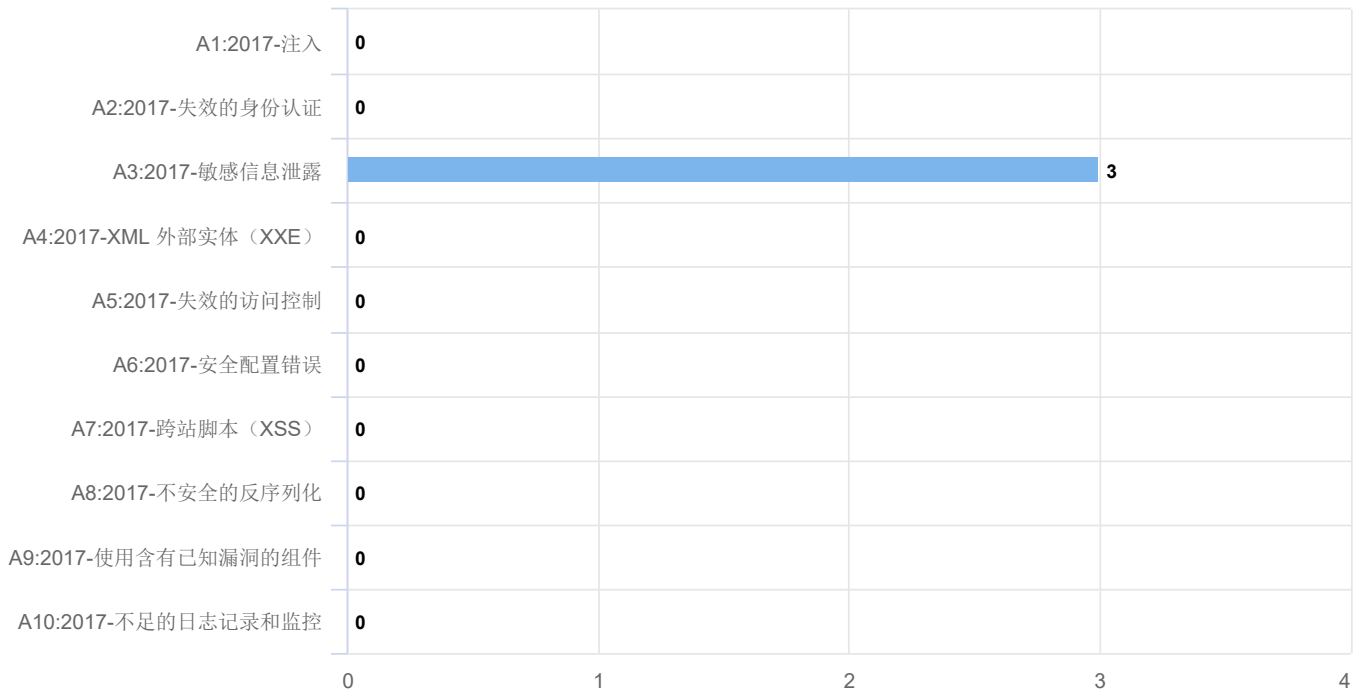
● 执行效率低下 - 17

● 崩溃和异常错误 - 3

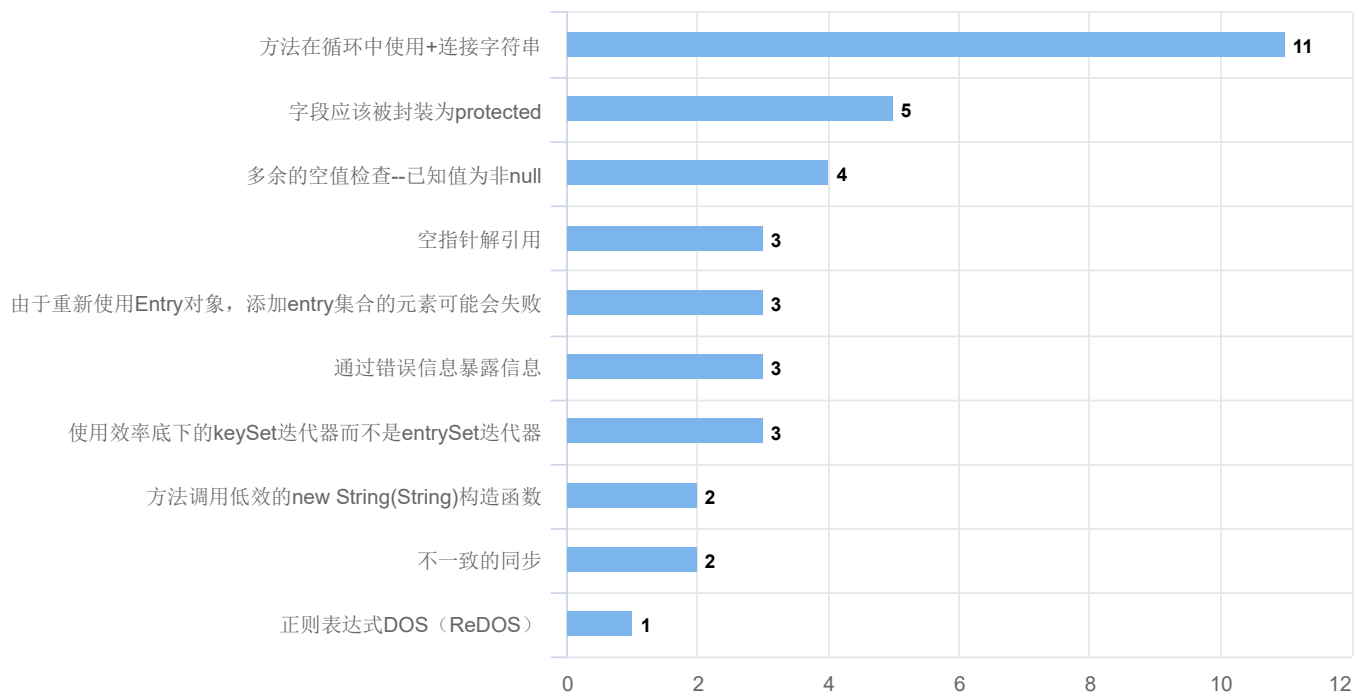
● 数值处理不当 - 1

● 并发和时序错误 - 2

OWASP 十大严重安全漏洞



数量最多的十大漏洞类型



CWE/SANS 排名前25的严重安全漏洞

TOP: 1 数据库命令注入 CWE-89	0
TOP: 2 操作系统命令注入 CWE-78	0
TOP: 3 缓冲区溢出 CWE-120	0
TOP: 4 跨站脚本 CWE-79	0
TOP: 5 缺少关键功能的身份验证 CWE-306	0
TOP: 6 缺少授权 CWE-862	0
TOP: 7 使用硬编码的凭证 CWE-798	0
TOP: 8 缺少敏感数据的加密 CWE-311	0
TOP: 9 危险类型的文件无限制上传 CWE-434	0
TOP: 10 在安全决策中依赖不可信输入 CWE-807	0
TOP: 11 用不必要的特权执行 CWE-250	0
TOP: 12 跨站请求伪造 CWE-352	0
TOP: 13 限制目录的路径名限制不当 CWE-22	0
TOP: 14 没有完整性检查的代码下载 CWE-494	0
TOP: 15 授权不正确 CWE-863	0
TOP: 16 来自不受信任的功能 CWE-829	0
TOP: 17 关键资源的权限分配不正确 CWE-732	0
TOP: 18 潜在危险功能的使用 CWE-676	0
TOP: 19 使用存在风险的加密算法 CWE-327	0
TOP: 20 错误的缓冲区大小计算 CWE-131	0
TOP: 21 对过度验证尝试的不当限制 CWE-307	0
TOP: 22 URL重定向到不受信任的站点 CWE-601	0
TOP: 23 不受控制的格式字符串 CWE-134	0
TOP: 24 整数溢出或绕回 CWE-190	0
TOP: 25 没加盐的单向哈希的使用 CWE-759	0

0

[高危] 空指针解引用

- 缺陷描述
- 漏洞与风险
 - 服务下线程序崩溃
 - 任意代码执行
- 漏洞利用威胁
- 缓解与预防
- 演示实例
 - 漏洞代码示例 1
 - 漏洞代码修复示例 1
 - 漏洞代码示例 2
 - 漏洞代码修复示例 2
 - 漏洞代码示例 3
 - 漏洞代码修复示例 3
- 参考资料

缺陷描述

尝试使用空指针访问数据将导致运行时错误。当程序取消引用某个预期为有效但结果为空值的指针，就会发生空指针取消引用。发生空指针取消引用缺陷通常是由于错误处理或争用情况无效，并通常会导致程序异常中止。在 C/C++ 代码中对指针取消引用之前，必须对其进行检查以确认它不为空值。

空指针解引用检查器查找那些对空指针或可能为空的指针进行取消引用的实例。

漏洞与风险

服务下线程序崩溃

空指针取消引用通常会导致服务下线，程序崩溃、退出或重启。这些问题通常是因为无效的异常处理而发生的。即使处理了异常，让程序回复正常状态也是非常困难的。

任意代码执行

少数情况下空指针解引用会触发任意代码执行漏洞。

漏洞利用威胁

严重

缓解与预防

要避免该漏洞，你应该：

- 对所有将返回值的函数进行空值检查
- 确保所有外部输入都经过验证
- 明确地对变量进行初始化
- 确保对不同寻常的异常进行正确处理

演示实例

漏洞代码示例 1

```
1 void reassign(int *argument, int *p) {
2     if (goodEnough(argument)) return;
3     *argument = *p;
4 }
5
```



```

6 void npd_check_call_must(int *argument) {
7     int *p = getValue();
8     if (p != 0) {
9         *p = 1;
10    }
11    reassign(argument, p);
12 }

```

虽然在第 8 行对 *p 进行了空值检查，它在之后被传递到函数 reassign，而在此，根据第 11 行的条件语句的结果，可能在不进行空值检查的情况下就对其取消引用。该类型漏洞会产生无法预料的意外结果。

漏洞代码修复示例 1

```

1 void reassign(int *argument, int *p) {
2     if (goodEnough(argument)) return;
3     *argument = *p;
4 }
5
6 void npd_check_call_must(int *argument) {
7     int *p = getValue();
8     if (p != 0) {
9         *p = 1;
10    }
11    if (p != 0) reassign(argument, p);
12 }

```

在经修复的代码版本中，在第 11 行加入了另一次空值检查。

漏洞代码示例 2

在这个例子里程序从用户那里获取一个 IP 地址，验证它的合法性并查询它的主机名，保存在缓存区中。

```

1 void host_lookup(char *user_supplied_addr){
2     struct hostent *hp;
3     in_addr_t *addr;
4     char hostname[64];
5     in_addr_t inet_addr(const char *cp);
6
7     // routine that ensures user_supplied_addr is in the right format for conversion
8     validate_addr_form(user_supplied_addr);
9     addr = inet_addr(user_supplied_addr);
10    hp = gethostbyaddr( addr, sizeof(struct in_addr), AF_INET);
11    strcpy(hostname, hp->h_name);
12 }

```

如果 IP 地址看起来是正常的，但是没能获取主机名，那么 gethostbyaddr() 这个调用就会返回 NULL。由于这段代码没有检验 gethostbyaddr 的返回值，因此在 11 行调用 strcpy 的时候就会引发空指针解引用。

漏洞代码修复示例 2

```

1 void host_lookup(char *user_supplied_addr){
2     struct hostent *hp;
3     in_addr_t *addr;
4     char hostname[64];
5     in_addr_t inet_addr(const char *cp);
6
7     // routine that ensures user_supplied_addr is in the right format for conversion
8     validate_addr_form(user_supplied_addr);
9     addr = inet_addr(user_supplied_addr);
10    hp = gethostbyaddr( addr, sizeof(struct in_addr), AF_INET);
11    if (hp != 0) strcpy(hostname, hp->h_name);
12 }

```

修复方式为在 11 行加入额外的检查逻辑。

漏洞代码示例 3

```
1 public static String readConfigFromFile(File file) {
2     try {
3         return com.google.common.io.Files.toString(file);
4     } catch (java.io.IOException e) {
5         return null;
6     }
7 }
8
9 public static void main(String[] args) {
10    if (args.length < 0) {
11        return;
12    }
13    String config = readConfigFromFile(args[0]);
14    // config可能为空指针。如为空导致main退出。
15    String[] lines = config.split("\n");
16    ...
17 }
```

漏洞代码修复示例 3

```
1 public static String readConfigFromFile(File file) {
2     try {
3         return com.google.common.io.Files.toString(file);
4     } catch (java.io.IOException e) {
5         return null;
6     }
7 }
8
9 public static void main(String[] args) {
10    if (args.length < 0) {
11        return;
12    }
13    String config = readConfigFromFile(args[0]);
14    if (config == null) {
15        System.err.println("Can't read config from " + args[0]);
16        System.exit(1);
17    }
18    String[] lines = config.split("\n");
19    ...
20 }
```

参考资料

1. <https://cwe.mitre.org/data/definitions/476.html> ↩

漏洞报告 (1 - 3)

缺陷位置:

src/main/java/zuo/biao/apijson/JSONObject.java: 411

标注: 确认

时间: 2018-11-16 10:25:54

可信度: 95%

缺陷ID: d1d990de49bc84dee0db733cf694e7d3

触发步骤:

src/main/java/zuo/biao/apijson/JSONObject.java

```
402         Log.e(TAG, "put value == null >> return null;");
403         return null;
404     }
405     if (StringUtil.isEmpty(key, true)) {
406         Class<?> clazz = value.getClass();
407         if (clazz == null || clazz.getAnnotation(MethodAccess.class) == null) {
408             throw new IllegalArgumentException("puts StringUtil.isNotEmpty(key, true)
409             == false" +
410             " && clazz == null || clazz.getAnnotation(MethodAccess.class) ==
411             null" +
412             " \n key为空时仅支持 类型被@MethodAccess注解 的value !!!" +
413             " \n 如果一定要这么用, 请对 " + clazz.getName() + " 注解!" +
```

1 这里比较了clazz和null, 说明clazz可能为空指针

src/main/java/zuo/biao/apijson/JSONObject.java

```
406         Class<?> clazz = value.getClass();
407         if (clazz == null || clazz.getAnnotation(MethodAccess.class) == null) {
408             throw new IllegalArgumentException("puts StringUtil.isNotEmpty(key, true)
409             == false" +
410             " && clazz == null || clazz.getAnnotation(MethodAccess.class) ==
411             null" +
412             " \n key为空时仅支持 类型被@MethodAccess注解 的value !!!" +
413             " \n 如果一定要这么用, 请对 " + clazz.getName() + " 注解!" +
414             " \n 如果是类似 key[]: {} 结构的请求, 建议用 putsAll(...) !");
415         }
416         key = value.getClass().getSimpleName();
```

2 调用clazz的java.lang.Class.getName方法(使用可疑的空指针)

缺陷位置:

src/main/java/zuo/biao/apijson/server/AbstractParser.java: 1077

标注: 确认

时间: 2018-11-16 10:25:54

可信度: 98%

缺陷ID: 551386e3a9a5cb4d8d4f43e32105174e

触发步骤:

src/main/java/zuo/biao/apijson/server/AbstractParser.java

```
1061         break;
1062     }
1063 }
1064
1065 //逐层到达targetKey的直接容器JSONObject parent
1066 if (keys != null && keys.length > 1) {
1067     for (int i = 0; i < keys.length - 1; i++) { //一步一步到达指定位置parentPath
1068         if (parent == null) { //不存在或路径错误(中间的key对应value不是JSONObject)
1069             break;
1070         }
1071     }
1072 }
1073 }
1074
1075 if (parent != null) {
```

1 这里比较了keys和null, 说明keys可能为空指针

src/main/java/zuo/biao/apijson/server/AbstractParser.java

```
1072     }
1073 }
1074
1075 if (parent != null) {
```

```

1076         Log.i(TAG, "getValueByPath >> get from queryResultMap >> return
parent.get(keys[keys.length - 1]);");
1077         target = parent.get(keys[keys.length - 1]); //值为null应该报错
NotExistException, 一般都是id关联, 不可为null, 否则可能绕过安全机制
1078         if (target != null) {
1079             Log.i(TAG, "getValueByPath >> getValue >> return target = " + target);
1080             return target;
1081         }

```

2 从keys.length中读取数据(使用可疑的空指针)

缺陷位置:

src/main/java/zuo/biao/apijson/server/AbstractSQLConfig.java: 897

标注: 确认

时间: 2018-11-16 10:25:54

可信度: 65%

缺陷ID: 6987f50dbf46867df506963bbf65fe13

触发步骤:

src/main/java/zuo/biao/apijson/server/AbstractSQLConfig.java

```

892         andList.add(i, key); //userId的优先级不能比id高 0, key);
893     } else {
894         andList.add(key); //AbstractSQLExecutor.onPutColumn里getSQL, 要保证缓存
的SQL和查询的SQL里 where 的 key:value 顺序一致
895     }
896 }
1 zuo.biao.apijson.server.AbstractSQLConfig.putWhere(String, Object, boolean)中
AbstractSQLConfig.combine可能的空指针解引用
897     combine.put("&", andList);
898 }
899     return this;
900 }
901 }

```

1 zuo.biao.apijson.server.AbstractSQLConfig.putWhere(String, Object, boolean)中 AbstractSQLConfig.combine可能的空指针解引用

[严重] 正则表达式DOS (ReDOS)

正则表达式 (regexs) 经常被Denial of Service (DOS) 攻击 (称为ReDOS) 。这是因为regex引擎在分析某些字符串时可能会花费大量的时间, 时间取决于正则表达式的定义。

例如, 对于正则表达式`^(a+)+$`, 输入"`aaaaaaaaaaaaaaaaX`" 将导致正则引擎分析65536条不同的路径。^[1]
来自OWASP引用的例子

因此, 单个请求可能会在服务器端造成大量的计算。这个正则表达式以及其它类似的问题, 是由于括号内的+ (或a*) 和括号之外的+ (或a*) 这两种不同的方式都可以被正则表达式接受。这样写的方式, 每一个+可以使用"a"这个字符。为了解决这个问题, 应该重写正则表达式来消除歧义。例如, 这可以简单地重写为:`^a+$`, 这大概是作者的意思 (任意数量的a) 。假设这就是最初的正则表达式的含义, 那么这个新的正则表达式可以快速地进行评估, 并且不受ReDOS的影响。

引用

Sebastian Kubeck's Weblog: Detecting and Preventing ReDoS Vulnerabilities

[1] OWASP: Regular expression Denial of Service

CWE-400: Uncontrolled Resource Consumption ('Resource Exhaustion')

漏洞报告

缺陷位置:

`src/main/java/zuo/biao/apijson/StringUtil.java: 322`

标注: 确认

时间: 2018-11-16 10:25:54

可信度: 45%

缺陷ID: 9da4fa86b21e269ec818d177190c28fb

触发步骤:

`src/main/java/zuo/biao/apijson/StringUtil.java`

```
317     PATTERN_ALPHA = Pattern.compile("^([a-zA-Z]+)$");
318     PATTERN_ALPHA_BIG = Pattern.compile("^([A-Z]+)$");
319     PATTERN_ALPHA_SMALL = Pattern.compile("^([a-z]+)$");
320     PATTERN_NAME = Pattern.compile("[0-9a-zA-Z_]+$"); // 已用55个中英字符测试通过
321     PATTERN_PHONE = Pattern.compile("^((13[0-9])|(15[^4,\\D])|(18[0-2,5-9])|(17[0-9]))\\d{8}$");
    1 正则表达式`^([a-zA-Z0-9_\\-\\.]+)@((\\[[0-9]{1,3}\\.[0-9]{1,3}\\.[0-9]{1,3}\\.|([a-zA-Z0-9\\-]+\\.)+)([a-zA-Z]{2,4}|[0-9]{1,3})(\\)?)$`因为包含了可能引起路径爆炸的结构`[...] + ... +`, 所以容易受到拒绝服务攻击 (ReDOS)
322     PATTERN_EMAIL = Pattern.compile("^([a-zA-Z0-9_\\-\\.]+)@((\\[[0-9]{1,3}\\.[0-9]{1,3}\\.[0-9]{1,3}\\.|([a-zA-Z0-9\\-]+\\.)+)([a-zA-Z]{2,4}|[0-9]{1,3})(\\)?)$");
323     PATTERN_ID_CARD = Pattern.compile("(^[1-9]\\d{5}(18|19|([23]\\d))\\d{2}((0[1-9])|(10|11|12))((([0-2][1-9])|10|20|30|31)\\d{3}[0-9Xx]$)|^[1-9]\\d{5}\\d{2}((0[1-9])|(10|11|12))((([0-2][1-9])|10|20|30|31)\\d{2}$)");
324     PATTERN_PASSWORD = Pattern.compile("[0-9a-zA-Z]+$");
325 }
326
```


未标注漏洞摘要 (1 - 36)

严重程度	缺陷类型	位置	时间	可信度	缺陷ID
严重	空指针解引用	src/main/java/zuo/biao/apijson/server/AbstractSQLConfig.java:844	2018-11-16 10:25:54	50%	fd0e35d
一般	不一致的同步	src/main/java/zuo/biao/apijson/server/AbstractParser.java:132	2018-11-16 10:25:54	20%	05eb2ae
一般	不一致的同步	src/main/java/zuo/biao/apijson/server/AbstractParser.java:137	2018-11-16 10:25:54	20%	fb26564
一般	使用效率底下的key Set迭代器而不是entrySet迭代器	src/main/java/zuo/biao/apijson/server/AbstractParser.java:1052	2018-11-16 10:25:54	15%	cdcafb9
一般	使用效率底下的key Set迭代器而不是entrySet迭代器	src/main/java/zuo/biao/apijson/server/AbstractSQLConfig.java:850	2018-11-16 10:25:54	15%	c0f908e
一般	使用效率底下的key Set迭代器而不是entrySet迭代器	src/main/java/zuo/biao/apijson/server/AbstractSQLConfig.java:1536	2018-11-16 10:25:54	15%	dfd26a3
一般	方法调用低效的Number构造函数, 应改为使用静态的valueOf	src/main/java/zuo/biao/apijson/server/AbstractVerifier.java:181	2018-11-16 10:25:54	15%	dc5214a
一般	方法调用低效的new String(String)构造函数	src/main/java/zuo/biao/apijson/server/AbstractObjectParser.java:332	2018-11-16 10:25:54	15%	e23d3e2
一般	方法调用低效的new String(String)构造函数	src/main/java/zuo/biao/apijson/server/AbstractSQLConfig.java:2081	2018-11-16 10:25:54	15%	3b7a140
一般	通过错误信息暴露信息	src/main/java/zuo/biao/apijson/server/AbstractParser.java:292	2018-11-16 10:25:54	30%	41390e3
一般	通过错误信息暴露信息	src/main/java/zuo/biao/apijson/server/AbstractParser.java:407	2018-11-16 10:25:54	30%	6aef0b1
一般	通过错误信息暴露信息	src/main/java/zuo/biao/apijson/server/AbstractSQLExecutor.java:370	2018-11-16 10:25:54	30%	5312773
建议改进	switch在没有缺省情况的下执行case语句	src/main/java/zuo/biao/apijson/server/AbstractSQLConfig.java:985	2018-11-16 10:25:54	10%	3e40b7a

建议改进	多余的空值检查--已知值为非null	src/main/java/zuo/biao/apijson/JSONObject.java:407	2018-11-16 10:25:54	5%	396c4b7
建议改进	多余的空值检查--已知值为非null	src/main/java/zuo/biao/apijson/SQL.java:355	2018-11-16 10:25:54	15%	c15111e
建议改进	多余的空值检查--已知值为非null	src/main/java/zuo/biao/apijson/server/AbstractObjectParser.java:234	2018-11-16 10:25:54	5%	46a220b
建议改进	多余的空值检查--已知值为非null	src/main/java/zuo/biao/apijson/server/AbstractSQLConfig.java:1781	2018-11-16 10:25:54	15%	800b8f0
建议改进	字段应该被封装为protected	src/main/java/zuo/biao/apijson/RequestRole.java:55	2018-11-16 10:25:54	15%	22595db
建议改进	字段应该被封装为protected	src/main/java/zuo/biao/apijson/StringUtil.java:633	2018-11-16 10:25:54	15%	92b082c
建议改进	字段应该被封装为protected	src/main/java/zuo/biao/apijson/server/Logic.java:27	2018-11-16 10:25:54	5%	8abb60f
建议改进	字段应该被封装为protected	src/main/java/zuo/biao/apijson/server/Logic.java:32	2018-11-16 10:25:54	15%	b919d41
建议改进	字段应该被封装为protected	src/main/java/zuo/biao/apijson/server/Logic.java:37	2018-11-16 10:25:54	15%	6046d81
建议改进	方法在循环中使用+连接字符串	src/main/java/zuo/biao/apijson/SQL.java:360	2018-11-16 10:25:54	15%	84557ed
建议改进	方法在循环中使用+连接字符串	src/main/java/zuo/biao/apijson/StringUtil.java:139	2018-11-16 10:25:54	15%	24dec43
建议改进	方法在循环中使用+连接字符串	src/main/java/zuo/biao/apijson/StringUtil.java:557	2018-11-16 10:25:54	15%	ee416ab
建议改进	方法在循环中使用+连接字符串	src/main/java/zuo/biao/apijson/StringUtil.java:660	2018-11-16 10:25:54	15%	c24f30d
建议改进	方法在循环中使用+连接字符串	src/main/java/zuo/biao/apijson/server/AbstractSQLConfig.java:502	2018-11-16 10:25:54	15%	deefef6
建议改进	方法在循环中使用+连接字符串	src/main/java/zuo/biao/apijson/server/AbstractSQLConfig.java:962	2018-11-16 10:25:54	15%	e7458d0
建议改进	方法在循环中使用+连接字符串	src/main/java/zuo/biao/apijson/server/AbstractSQLConfig.java:1255	2018-11-16 10:25:54	15%	65334ff
建议改进	方法在循环中使用+连接字符串	src/main/java/zuo/biao/apijson/server/AbstractSQLConfig.java:1399	2018-11-16 10:25:54	15%	77a92b1
建议改进	方法在循环中使用+连接字符串	src/main/java/zuo/biao/apijson/server/AbstractSQLConfig.java:1451	2018-11-16 10:25:54	15%	80b29d2

建议改进	方法在循环中使用 +连接字符串	src/main/java/zuo/biao/apijson/server/ AbstractSQLConfig.java:1539	2018-11-16 10:25:54	15%	22705e9
建议改进	方法在循环中使用 +连接字符串	src/main/java/zuo/biao/apijson/server/ RemoteFunction.java:131	2018-11-16 10:25:54	15%	94f5368
建议改进	由于重新使用Entry 对象,添加entry集 合的元素可能会失败	src/main/java/zuo/biao/apijson/server/ AbstractObjectParser.java:233	2018-11-16 10:25:54	20%	524bfeb
建议改进	由于重新使用Entry 对象,添加entry集 合的元素可能会失败	src/main/java/zuo/biao/apijson/server/ Structure.java:278	2018-11-16 10:25:54	20%	9044719
建议改进	由于重新使用Entry 对象,添加entry集 合的元素可能会失败	src/main/java/zuo/biao/apijson/server/ Structure.java:400	2018-11-16 10:25:54	20%	392f80f