**FEP專案Fortify漏洞掃描程式修改說明**

2022/08/19

Copyright the SYSCOM Computer Engineering Co.2022, All Rights Reserved

Revision History

|  |  |  |
| --- | --- | --- |
| Date | Author | Remark |
| 2021/08/19 | YU, Richard | Initial Version |
| 2021/09/05 | YU, Richard | 補充[Order By語句後的${}](#_Order_By語句後的${})章節  新增[关系运算符${}](#_关系运算符${})章節  新增[其他](#_其他)章節 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# Catalog

[Catalog 3](#_Toc113264338)

[Concept 5](#_Toc113264339)

[A01 Broken Access Control 6](#_Toc113264340)

[Path Manipulation(Critical) 6](#_Toc113264341)

[Access Control: Database (High) 7](#_Toc113264342)

[Access Specifier Manipulation (High) 7](#_Toc113264343)

[Cross-Site Request Forgery (Low) 7](#_Toc113264344)

[A02 Cryptographic Failures 8](#_Toc113264345)

[Insecure Transport: Weak SSL Protocol (Critical) 8](#_Toc113264346)

[Key Management: Hardcoded Encryption Key (Critical) 8](#_Toc113264347)

[Weak Encryption (Critical) 8](#_Toc113264348)

[Insecure Randomness (High) 8](#_Toc113264349)

[Key Management: Empty Encryption Key (High) 8](#_Toc113264350)

[Key Management: Hardcoded Encryption Key (High) 8](#_Toc113264351)

[Privacy Violation: Heap Inspection (High) 9](#_Toc113264352)

[Weak Encryption: Insecure Mode of Operation (High) 9](#_Toc113264353)

[Key Management: Null Encryption Key (Low) 9](#_Toc113264354)

[A03 Injection 10](#_Toc113264355)

[SQL Injection: MyBatis Mapper (Critical) 10](#_Toc113264356)

[Like語句後的${} 10](#_Toc113264357)

[Order By語句後的${} 10](#_Toc113264358)

[关系运算符${} 11](#_Toc113264359)

[=語句後的${} 11](#_Toc113264360)

[Call Store Procedures SQL中的${currentSchema} 11](#_Toc113264361)

[動態表名中的${} 12](#_Toc113264362)

[FEPTXN${tableNameSuffix} 12](#_Toc113264363)

[FEPLOG${tableNameSuffix} 15](#_Toc113264364)

[其他 16](#_Toc113264365)

[A04 Insecure Design 17](#_Toc113264366)

[A05 Security Misconfiguration 18](#_Toc113264367)

[Spring Boot Misconfiguration: Actuator Endpoint Security Disabled (Critical) 18](#_Toc113264368)

[Password Management: Password in Configuration File (High) 18](#_Toc113264369)

[Spring Boot Misconfiguration: DevTools Enabled (High) 20](#_Toc113264370)

# Concept

本文檔旨在闡述，本專案的源代碼在進行Fortify掃描後，程式修改說明。

# A01 Broken Access Control

## Path Manipulation(Critical)

fep-frmcommon新增程式CleanPathUtil，如下：

|  |
| --- |
| package com.syscom.fep.frmcommon.util; import java.util.HashMap; import java.util.Map;  */\*\*  \* 用來解決Fortify Path Manipulation Issues  \*  \** ***@author*** *Richard  \*/* public class CleanPathUtil {  private static final Map<Character, Character> *whiteChars* = new HashMap<>();   static {  for (int i = 0; i <= 126; i++) {  *whiteChars*.put((char) i, (char) i);  }  }   private CleanPathUtil() {  }   public static String cleanString(String aString) {  StringBuilder sb = new StringBuilder();  for (int i = 0; i < aString.length(); i++) {  char charAt = aString.charAt(i);  Character character = *whiteChars*.get(charAt);  if (character != null) {  sb.append(character);  }  }  return sb.toString();  } } |

修改方法，使用CleanPathUtil.cleanString對path進行clean，修改範例如下：

|  |  |
| --- | --- |
| 修改前 | File file = new File(\_BatchOutputPath + strFileName); |
| 修改後 | File file = new File(**CleanPathUtil.*cleanString*(**\_BatchOutputPath + strFileName**)**); |

## Access Control: Database (High)

待定。

## Access Specifier Manipulation (High)

修改方法，使用ReflectionUtils.makeAccessible，修改範例如下：

|  |  |
| --- | --- |
| 修改前 | field.setAccessible(true);  ReflectionUtils.setField(field, entity, substr);  field.setAccessible(false); |
| 修改後 | ReflectionUtils.makeAccessible(field);  ReflectionUtils.setField(field, entity, substr);  ~~field.setAccessible(false);~~ |

|  |  |
| --- | --- |
| 修改前 | Method method = processorCls.getDeclaredMethod(methodName, argsCls);  method.setAccessible(true); |
| 修改後 | Method method = processorCls.getDeclaredMethod(methodName, argsCls);  ReflectionUtils.makeAccessible(method); |

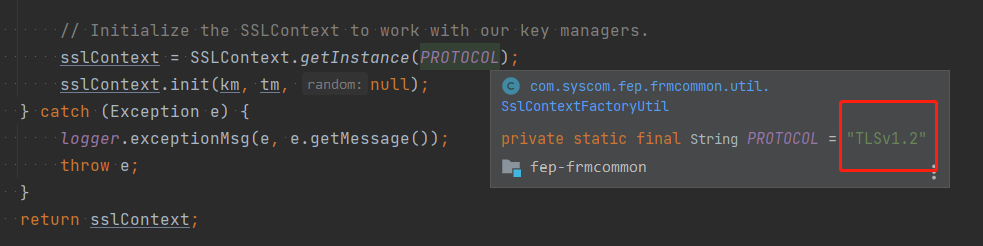
## Cross-Site Request Forgery (Low)

待定。

# A02 Cryptographic Failures

## Insecure Transport: Weak SSL Protocol (Critical)

修改方法，SSL Protocol由原本的TLS改為TLSv1.2。



## Key Management: Hardcoded Encryption Key (Critical)

修改方法，刪除atmgw-key.pem文件。

## Weak Encryption (Critical)

待定(照.NET翻寫的程式)。

## Insecure Randomness (High)

待定(Third-party javascript library)。

## Key Management: Empty Encryption Key (High)

待定(Third-party javascript library)。

## Key Management: Hardcoded Encryption Key (High)

修改方法，刪除atmgw-certificate.jks文件。

更新atmgw-certificate.p12文件。

## Privacy Violation: Heap Inspection (High)

待定。

## Weak Encryption: Insecure Mode of Operation (High)

待定。

## Key Management: Null Encryption Key (Low)

待定。

# A03 Injection

## SQL Injection: MyBatis Mapper (Critical)

### Like語句後的${}

修改方法一，使用CONCAT函式，修改範例如下：

|  |  |
| --- | --- |
| 修改前 | ALARM\_NO LIKE '%${alarmNo}%' |
| 修改後 | ALARM\_NO LIKE CONCAT('%', CONCAT(#{alarmNo}, '%')) |

修改方法二，使用<bind />，修改範例如下：

|  |  |
| --- | --- |
| 修改前 | ALARM\_NO LIKE '%${alarmNo}%' |
| 修改後 | <bind name="alarmNo" value="'%'+alarmNo+'%'"/>  ALARM\_NO LIKE #{alarmNo} |

### Order By語句後的${}

Order By中的欄位名，修改方法，使用DbHelper. isSqlInOrderByValid函式，修改範例如下：

|  |  |
| --- | --- |
| 修改前 | <if test="orderBy != null and orderBy != '' ">  ORDER BY ${orderBy}  </if> |
| 修改後 | <if test="@com.syscom.fep.common.util.DbHelper@isSqlInOrderByValid(orderBy)">  ORDER BY ${orderBy}  </if> |

Order By欄位名後的ASC/DESC，修改方法，使用<choose/>，修改範例如下：

|  |  |
| --- | --- |
| 修改前 | ORDER BY TASK\_NAME ${direction} |
| 修改後 | ORDER BY TASK\_NAME  <choose>  <when test="direction.toUpperCase() == 'ASC'">  ASC  </when>  <when test="direction.toUpperCase() == 'DESC'">  DESC  </when>  </choose> |

### 关系运算符${}

修改方法，使用RelationalOperators，將原本傳入的參數**@Param("operator") String operators**，改為**@Param("operators") RelationalOperators operators**，同時XML中修改範例如下：

|  |  |
| --- | --- |
| 修改前 | WHERE NPSCHK\_EFFECTDATE  ${operator} #{effDate,jdbcType=VARCHAR} |
| 修改後 | WHERE NPSCHK\_EFFECTDATE  ${operators.operator} #{effDate,jdbcType=VARCHAR} |

### =語句後的${}

修改方法，$直接改為#，同時如果有單引號，則刪除，修改範例如下：

|  |  |
| --- | --- |
| 修改前 | EVENT\_ALARM\_NO = '${alarmNo}' |
| 修改後 | EVENT\_ALARM\_NO = #{alarmNo} |

### Call Store Procedures SQL中的${currentSchema}

首先，在com.syscom.fep.mybatis.configuration.DataSourceConfiguration添加如下方法：

|  |
| --- |
| */\*\*  \* 從mybatis-config.xml中獲取<property name="currentSchema" />值  \* 此方法給以下地方使用：  \* 1. UserDefineExtMapper.xml中call SP的方法  \*  \** ***@return*** *\*/* public static String getPropCurrentSchema() {  SqlSessionFactory factory = SpringBeanFactoryUtil.*getBean*("fepSqlSessionFactory");  String currentSchema = (String) factory.getConfiguration().getVariables().get("currentSchema");  return currentSchema; } |

然後修改方法，將${currentSchema}直接改為${@com.syscom.fep.mybatis.configuration.DataSourceConfiguration@getPropCurrentSchema()}，範例如下：

|  |  |
| --- | --- |
| 修改前 | call ${currentSchema}.GETSTAN(  #{ID, mode=OUT, jdbcType=INTEGER},  #{INTERVAL, mode=OUT, jdbcType=INTEGER}  ) |
| 修改後 | call ${@com.syscom.fep.mybatis.configuration.DataSourceConfiguration@getPropCurrentSchema()}.GETSTAN(  #{ID, mode=OUT, jdbcType=INTEGER},  #{INTERVAL, mode=OUT, jdbcType=INTEGER}  ) |

### 動態表名中的${}

#### FEPTXN${tableNameSuffix}

首先，(通過MyBatis Generator)在FeptxnMapper.xml中創建如下的<sql />。

|  |
| --- |
| <sql id="chooseTableName">  <!--  WARNING - @mbg.generated  This element is automatically generated by MyBatis Generator, do not modify.  -->  <choose>  <when test="tableNameSuffix == '01'">  FEPTXN01  </when>  <when test="tableNameSuffix == '02'">  FEPTXN02  </when>  <when test="tableNameSuffix == '03'">  FEPTXN03  </when>  <when test="tableNameSuffix == '04'">  FEPTXN04  </when>  <when test="tableNameSuffix == '05'">  FEPTXN05  </when>  <when test="tableNameSuffix == '06'">  FEPTXN06  </when>  <when test="tableNameSuffix == '07'">  FEPTXN07  </when>  <when test="tableNameSuffix == '08'">  FEPTXN08  </when>  <when test="tableNameSuffix == '09'">  FEPTXN09  </when>  <when test="tableNameSuffix == '10'">  FEPTXN10  </when>  <when test="tableNameSuffix == '11'">  FEPTXN11  </when>  <when test="tableNameSuffix == '12'">  FEPTXN12  </when>  <when test="tableNameSuffix == '13'">  FEPTXN13  </when>  <when test="tableNameSuffix == '14'">  FEPTXN14  </when>  <when test="tableNameSuffix == '15'">  FEPTXN15  </when>  <when test="tableNameSuffix == '16'">  FEPTXN16  </when>  <when test="tableNameSuffix == '17'">  FEPTXN17  </when>  <when test="tableNameSuffix == '18'">  FEPTXN18  </when>  <when test="tableNameSuffix == '19'">  FEPTXN19  </when>  <when test="tableNameSuffix == '20'">  FEPTXN20  </when>  <when test="tableNameSuffix == '21'">  FEPTXN21  </when>  <when test="tableNameSuffix == '22'">  FEPTXN22  </when>  <when test="tableNameSuffix == '23'">  FEPTXN23  </when>  <when test="tableNameSuffix == '24'">  FEPTXN24  </when>  <when test="tableNameSuffix == '25'">  FEPTXN25  </when>  <when test="tableNameSuffix == '26'">  FEPTXN26  </when>  <when test="tableNameSuffix == '27'">  FEPTXN27  </when>  <when test="tableNameSuffix == '28'">  FEPTXN28  </when>  <when test="tableNameSuffix == '29'">  FEPTXN29  </when>  <when test="tableNameSuffix == '30'">  FEPTXN30  </when>  <when test="tableNameSuffix == '31'">  FEPTXN31  </when>  </choose> </sql> |

然後在sql中引用，修改範例如下：

|  |  |
| --- | --- |
| 修改前 | FEPTXN${tableNameSuffix} |
| 修改後 | <include refid="com.syscom.fep.mybatis.mapper.FeptxnMapper.chooseTableName"/> |

範例二如下：

|  |  |
| --- | --- |
| 修改前 | FEPTXN${argsMap.tableNameSuffix} |
| 修改後 | <bind name="tableNameSuffix" value="argsMap.tableNameSuffix"/>  <include refid="com.syscom.fep.mybatis.mapper.FeptxnMapper.chooseTableName"/> |

#### FEPLOG${tableNameSuffix}

首先，(通過MyBatis Generator)在FeplogMapper.xml中創建如下的<sql />。

|  |
| --- |
| <sql id="chooseTableName">  <!--  WARNING - @mbg.generated  This element is automatically generated by MyBatis Generator, do not modify.  -->  <choose>  <when test="tableNameSuffix == '1'">  FEPLOG1  </when>  <when test="tableNameSuffix == '2'">  FEPLOG2  </when>  <when test="tableNameSuffix == '3'">  FEPLOG3  </when>  <when test="tableNameSuffix == '4'">  FEPLOG4  </when>  <when test="tableNameSuffix == '5'">  FEPLOG5  </when>  <when test="tableNameSuffix == '6'">  FEPLOG6  </when>  <when test="tableNameSuffix == '7'">  FEPLOG7  </when>  </choose> </sql> |

然後在sql中引用，修改範例如下：

|  |  |
| --- | --- |
| 修改前 | FEPLOG${tableNameSuffix} |
| 修改後 | <include refid="com.syscom.fep.mybatis.ems.mapper.FeplogMapper.chooseTableName"/> |

範例二如下：

|  |  |
| --- | --- |
| 修改前 | FEPLOG${argsMap.tableNameSuffix} |
| 修改後 | <bind name="tableNameSuffix" value="argsMap.tableNameSuffix"/>  <include refid="com.syscom.fep.mybatis.ems.mapper.FeplogMapper.chooseTableName"/> |

#### 其他

使用<choose />，修改範例如下：

|  |  |
| --- | --- |
| 修改前 | SELECT \* FROM ${table} |
| 修改後 | SELECT \* FROM  <choose>  <when test="table.toUpperCase() == 'WKPOSTDTL'">  WKPOSTDTL  </when>  <when test="table.toUpperCase() == 'HKWKPOSTDTL'">  HKWKPOSTDTL  </when>  <when test="table.toUpperCase() == 'MOWKPOSTDTL'">  MOWKPOSTDTL  </when>  </choose> |

# A04 Insecure Design

# A05 Security Misconfiguration

## Spring Boot Misconfiguration: Actuator Endpoint Security Disabled (Critical)

待定

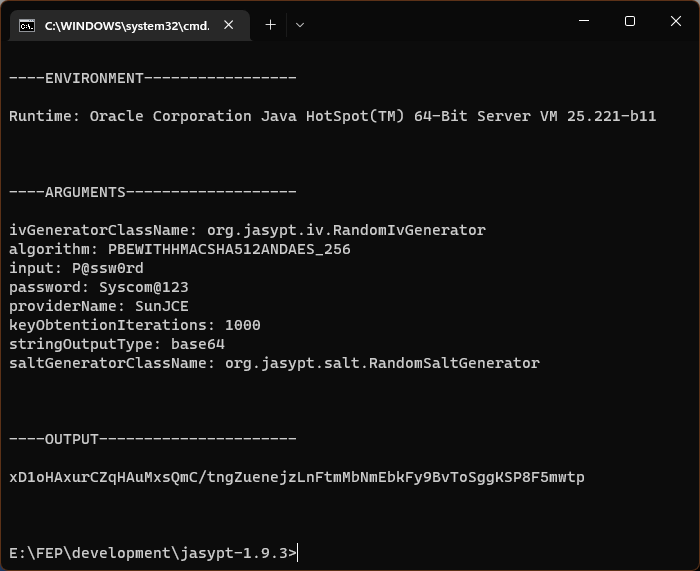
## Password Management: Password in Configuration File (High)

首先訪問[jasypt](http://www.jasypt.org/download.html)，下載jasypt-1.9.3工具，解壓後放置在任意folder下，例如C:\ jasypt-1.9.3。然後打開cmd分別執行如下的命令：

產出db設定中加密後的密碼字串，其中原始密碼為P@ssw0rd：

|  |
| --- |
| C:\ jasypt-1.9.3\bin\encrypt.bat input=**P@ssw0rd** password=Syscom@123 algorithm=PBEWITHHMACSHA512ANDAES\_256 keyObtentionIterations=1000 saltGeneratorClassName=org.jasypt.salt.RandomSaltGenerator providerName=SunJCE stringOutputType=base64 ivGeneratorClassName=org.jasypt.iv.RandomIvGenerator |

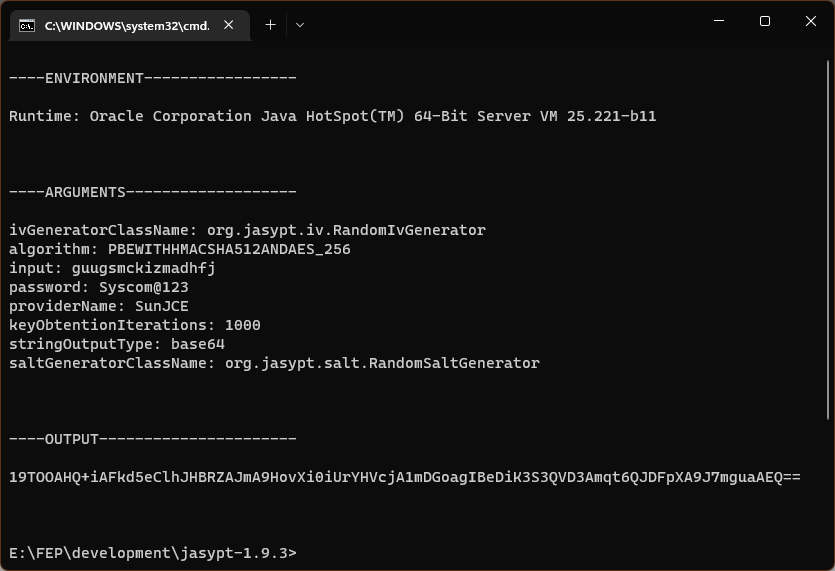
例如下圖，記錄產出的OUTPUT字串，注意每次產出的加密字串不相同。



產出mail設定中加密後的密碼字串，其中原始密碼為guugsmckizmadhfj：

|  |
| --- |
| C:\ jasypt-1.9.3\encrypt.bat input=**guugsmckizmadhfj** password=Syscom@123 algorithm=PBEWITHHMACSHA512ANDAES\_256 algorithm=PBEWITHHMACSHA512ANDAES\_256 keyObtentionIterations=1000 saltGeneratorClassName=org.jasypt.salt.RandomSaltGenerator providerName=SunJCE stringOutputType=base64 ivGeneratorClassName=org.jasypt.iv.RandomIvGenerator |

例如下圖，記錄產出的OUTPUT字串，注意每次產出的加密字串不相同。



\fep\pom.xml的<properties></properties>中定義如下內容：

|  |
| --- |
| <jasypt-spring-boot-starter-version>3.0.4</jasypt-spring-boot-starter-version> |

\fep-frmcommon\pom.xml的<dependencies></dependencies>內添加jasypt-spring-boot-starter，如下內容：

|  |
| --- |
| <dependency>  <groupId>com.github.ulisesbocchio</groupId>  <artifactId>jasypt-spring-boot-starter</artifactId>  <version>${jasypt-spring-boot-starter-version}</version>  </dependency> |

修改properties檔中相關的password設定，修改範例如下，其中ENC(...)中的內容則是上述jasypt工具產出的加密後的字串：

|  |  |
| --- | --- |
| 修改前 | spring.datasource.XXX.password=P@ssw0rd |
| 修改後 | spring.datasource.XXX.password=ENC(rq7spHpmsd5nMmO25S2AWak+p5X/ueOaOR0LUD4gI5H3u3KsQlI+actC9wU4zr2R) |

Mail的設定，修改如下，其中ENC(...)中的內容則是上述jasypt工具產出的加密後的字串：

|  |  |
| --- | --- |
| 修改前 | spring.fep.XXX.mailsscode=guugsmckizmadhfj |
| 修改後 | spring.fep.XXX.mailsscode= ENC(3SM3px7gSO1uXkn8maceH+ruzkzWt6iFEeeSLiKub7Bk5iDBvspd8+6nqRWITw8DSXogtkPmgdA36hY6y/XCIg==) |

## Spring Boot Misconfiguration: DevTools Enabled (High)

在\fep-web\pom.xml中，spring-boot-devtools的scope為runtime，所以在build版時不會包進war檔中。

## Unreleased Resource: Streams

New File之後記得要把file關掉

請注意是檔案類型的都要在finally進行close

BufferedReader、BufferedWriter、OutputStreamWriter、File、FileInputStream

範例：

public void processFile(String fName) throws FileNotFoundException,

IOException {

FileInputStream fis;

try {

fis = new FileInputStream(fName);

int sz;

byte[] byteArray = new byte[BLOCK\_SIZE];

while ((sz = fis.read(byteArray)) != -1) {

processBytes(byteArray, sz);

}

}

finally {

if (fis != null) {

safeClose(fis);

}

}

}

public static void safeClose(FileInputStream fis) {

if (fis != null) {

try {

fis.close();

} catch (IOException e) {

log(e);

}

}

}

## Null Dereference

請先用判斷式確認參照的物件是否是空的，否則會發生null point exception

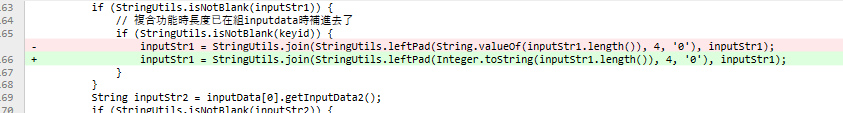
Foo foo = null;

If(foo != null) {

foo.setBar(val);

}

## Heap Inspection

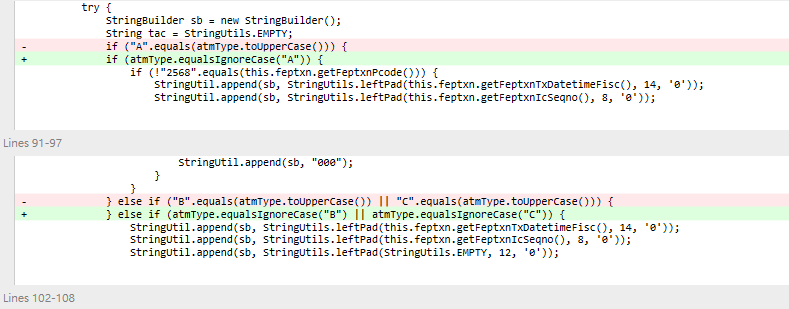


使用到敏感資料(例如密碼)之類的變數，請使用Integer.toString()，不要用String.valueOf(value)，一般數字轉字串可以使用String.valueOf(value)

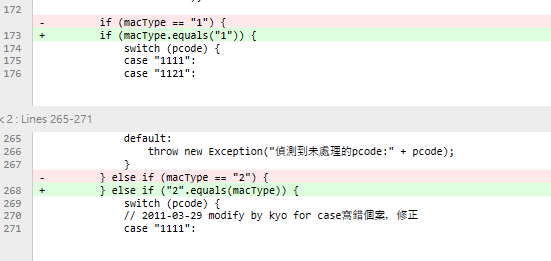
## Portability Flaw: Locale Dependent Comparison

請不要把變數進行.toUppercase或tolowercase改為全大寫或全小寫來跟文字做比對

請使用pattern.equalsIgnoreCase(""))



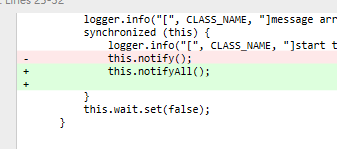
比對的時候，變數放後面，不要用==比較，必須使用equals



## Code Correctness: Call to notify()

//此問題待討論

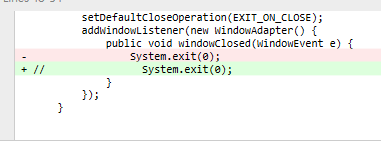
使用this.notifyAll();



## J2EE Bad Practices: JVM Termination

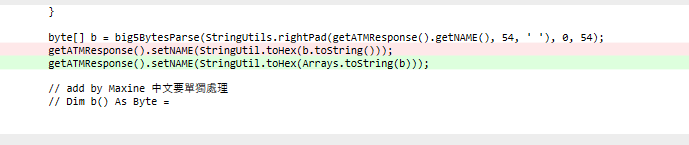
Web系統不要使用system.exit隨意中止整個系統

批次等系統按照需求可以使用此指令



## Code Correctness: toString on Array (4 issues)

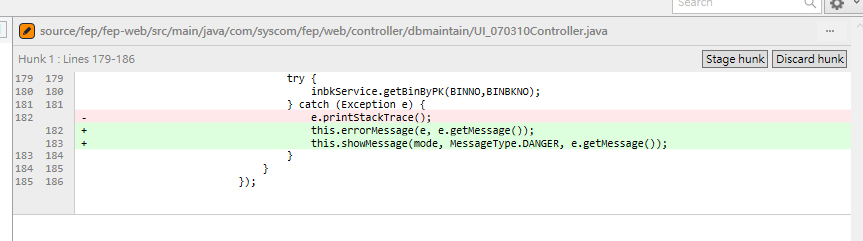
陣列請使用Arrays.toString(b)，不要使用tostring()



LOW LEVEL

System Information Leak

請不要寫e.printStackTrace();，請使用系統規範紀錄LOG寫法



**Unchecked Return Value**

**請不要直接寫mkdirs**

uI\_020061\_A.mkdirs();

改為以下寫法

