FB.UUF\_UNUSED\_FIELD

Детектор определяет ошибку, в ходе которой частный метод никогда не вызывается. Хотя вполне возможно, что метод будет вызван посредством отражения, более вероятно, что метод никогда не будет использоваться и его следует удалить.

Таблица №1 – Основные характеристики детектора

|  |  |
| --- | --- |
| Категория критичности | MAJOR |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Ошибка разработчика |
| CWE | — |
| CVE | — |
| Возможные последствия | Наличие неиспользуемых методов, особенно с модификатором private, может привести к излишней сложности в коде, снижению читаемости и поддерживаемости. |
| Комментарий | Если метод действительно не используется в коде, рекомендуется удалить его. |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Некорректное написание | Корректное написание |
| public static void incorrectTest() {  class MyClass {  private void unusedMethod() {  System.out.println("...");  }  public void doSomething() {  System.out.println("...");  }  }  MyClass myClass = new MyClass();  myClass.doSomething();  } | public static void correctTest() {  class MyClass {  public void doSomething() {  System.out.println("...");  }  }  MyClass myClass = new MyClass();  myClass.doSomething();  } |

Методы устранения уязвимости при срабатывании детектора:

1. Удалите неиспользуемый частный метод. Если метод нигде не вызывается внутри класса, его удаление упростит код.
2. Проверьте, не был ли метод предназначен для вызова из другого метода класса. Убедитесь, что логика программы реализована корректно и все необходимые методы вызываются.
3. Рассмотрите возможность удаления кода внутри неиспользуемого метода и его переноса в вызывающий метод (если такой существует). Это может упростить структуру кода.