FB.FE\_FLOATING\_POINT\_EQUALITY

Детектор определяет ошибку, в ходе которой операция сравнивает два значения с плавающей запятой на предмет равенства. Поскольку вычисления с плавающей запятой могут включать округление, вычисленные значения float и double могут быть неточными. Для значений, которые должны быть точными, например денежных значений, рассмотрите возможность использования типа с фиксированной точностью, такого как BigDecimal. Для значений, которые не должны быть точными, рассмотрите возможность сравнения на равенство в некотором диапазоне, например: if ( Math.abs(x - y) < .0000001 ).

Таблица №1 – Основные характеристики детектора

|  |  |
| --- | --- |
| Категория критичности | MAJOR |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Ошибка разработчика |
| CWE | — |
| CVE | — |
| Возможные последствия | Сравнение чисел с плавающей запятой (float и double) на точное равенство может привести к ошибкам из-за потери точности при вычислениях, что может вызвать неожиданные логические ошибки. |
| Комментарий | Используйте BigDecimal для финансовых расчетов или сравнивайте числа с погрешностью, используя Math.abs(x - y) < epsilon, где epsilon — допустимая разница. |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Некорректное написание | Корректное написание |
| public static void incorrectTest() {  double a = 0.1 \* 3;  double b = 0.3;  // сравнение значений с плавающей запятой на точное равенство  if (a == b) {  System.out.println("Значения равны");  } else {  System.out.println("Значения не равны");  }  } | public static void correctTest() {  double a = 0.1 \* 3;  double b = 0.3;  final double EPSILON = 1e-9;  // проверяем разницу значений в пределах допустимой погрешности  if (Math.abs(a - b) < EPSILON) {  System.out.println("Значения равны");  } else {  System.out.println("Значения не равны");  }  } |

Методы устранения уязвимости при срабатывании детектора:

1. **Замените == или != на сравнение с эпсилон:** Для большинства случаев, когда не требуется абсолютная точность, сравнивайте абсолютную разницу между двумя числами с плавающей запятой с небольшим пороговым значением (эпсилон). Выбор значения эпсилон зависит от конкретной задачи и требуемой точности.
2. **Используйте BigDecimal для точных расчетов:** Если вам необходима высокая точность, особенно при работе с денежными значениями или в финансовых приложениях, используйте класс BigDecimal. При создании BigDecimal из double или float рекомендуется использовать конструктор, принимающий String, чтобы избежать потери точности на этапе создания объекта. Для сравнения BigDecimal используйте метод equals() (для точного равенства значения и масштаба) или compareTo() (для численного равенства).
3. **Избегайте прямого сравнения результатов последовательных операций:** Если вы сравниваете результат серии вычислений с плавающей запятой с константой, рассмотрите возможность сравнения с эпсилон.
4. **Проверьте весь код:** Найдите все места, где выполняется прямое сравнение чисел с плавающей запятой, и примените один из рекомендуемых подходов.
5. **Будьте внимательны при работе с библиотеками:** Некоторые библиотеки могут предоставлять свои собственные методы для сравнения чисел с плавающей запятой с учетом погрешности. Изучите документацию используемых библиотек.