FB.NP\_NULL\_INSTANCEOF

Этот тест экземпляра всегда будет возвращать значение false, поскольку проверяемое значение гарантированно будет нулевым. Хотя это и безопасно, следует убедиться, что это не является признаком какого-то недопонимания или какой-либо другой логической ошибки.

Таблица №1 – Основные характеристики детектора FB.NP\_NULL\_INSTANCEOF

|  |  |
| --- | --- |
| Категория критичности | Minor |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Логическая ошибка |
| CWE | CWE-476 |
| CVE | --- |
| Возможные последствия | Бессмысленная проверка, возможное скрытие ошибок в логике программы |
| Комментарий (опционально) | Убедитесь, что проверка instanceof не избыточна и корректна. |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Потенциально небезопасное | Корректная конструкция |
| public static void incorrectExample(Object obj) {  if (obj == null) {  System.out.println("Объект null");  }  // Бессмысленная проверка, так как obj может быть null  if (obj instanceof String) {  System.out.println("Это строка: " + obj);  }  } | public static void correctExample(Object obj) {  if (obj == null) {  System.out.println("Объект null");  } else if (obj instanceof String) { // Проверка типа только для ненулевых объектов  System.out.println("Это строка: " + obj);  }  } |

Методы устранения уязвимости при срабатывании детектора

1. Проверка на null перед instanceof: Убедитесь, что проверка типа выполняется только для ненулевых объектов.
2. Логический анализ: Удалите избыточные проверки instanceof для заведомо null значений.
3. Рефакторинг условий: Объедините условия null-проверки и проверки типа, если это логически оправдано.