FB.SIO\_SUPERFLUOUS\_INSTANCEOF

Проверка типа выполняется с помощью оператора экземпляра, где можно статически определить, относится ли объект к запрошенному типу.

Таблица №1 – Основные характеристики детектора FB.SIO\_SUPERFLUOUS\_INSTANCEOF

|  |  |
| --- | --- |
| Категория критичности | Minor |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Superfluous instanceof |
| CWE | --- |
| CVE | --- |
| Возможные последствия | Избыточный код, усложняющий чтение и поддержку; незначительное снижение производительности из-за лишних проверок. |
| Комментарий | Рекомендуется удалить ненужные проверки или переработать логику, чтобы избежать лишних операций. |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Потенциально небезопасное | Корректная конструкция |
| public static void incorrectTest() {  Object obj = "Test String";    if (obj instanceof String) { // Проверка избыточна, т.к. obj уже String  String str = (String) obj;  System.out.println(str.toUpperCase());  }  } | public static void correctTest() {  Object obj = "Test String";    // Без лишнего instanceof, т.к. приведение безопасно  String str = (String) obj;  System.out.println(str.toUpperCase());  } |

Методы устранения уязвимости при срабатывании детектора.

1. Удалить избыточные instanceof, если тип объекта гарантирован;
2. Использовать полиморфизм (например, переопределение методов), чтобы избежать проверок типов;
3. Заменить instanceof на pattern matching (в Java 16+), если проверка действительно нужна;
4. Проверить логику программы, чтобы убедиться, что instanceof не является результатом ошибки в архитектуре.