FB.TQ\_ALWAYS\_VALUE\_USED\_WHERE\_NEVER\_REQUIRED

Значение, указанное как содержащее аннотацию квалификатора типа, используется в месте или местах, требующих, чтобы значение не содержало эту аннотацию.

Точнее, значение, помеченное квалификатором типа, указывающим, when=ALWAYS, гарантированно достигнет использования или применений, где тот же квалификатор типа указывает, when=NEVER.

Таблица №1 – Основные характеристики детектора FB.TQ\_ALWAYS\_VALUE\_USED\_WHERE\_NEVER\_REQUIRED

|  |  |
| --- | --- |
| Категория критичности | Minor |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Unnecessary Constant in Conditional |
| CWE | --- |
| CVE | --- |
| Возможные последствия | Ложное срабатывание условия - код может вести себя не так, как ожидал разработчик; снижение читаемости - избыточные условия усложняют понимание логики; потенциальные ошибки в логике программы - если значение действительно должно проверяться |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Потенциально небезопасное | Корректная конструкция |
| public static void incorrectTest() {  // 1. Бессмысленное условие (всегда true)  if (true) { // Найдёт FB.TQ\_ALWAYS\_VALUE\_USED\_WHERE\_NEVER\_REQUIRED  System.out.println("This will always execute");  }  // 2. Проверка финализированной константы  if (DEBUG\_MODE) { // Найдёт, если DEBUG\_MODE = false  System.out.println("Debug info"); // Этот код никогда не выполнится  }  // 3. Метод, всегда возвращающий одно значение  if (alwaysTrue()) { // Найдёт, если метод всегда возвращает true  System.out.println("Always true method");  }  } | public static void correctTest() {  // 1. Осмысленное условие  boolean shouldExecute = checkSomeCondition();  if (shouldExecute) {  System.out.println("This executes conditionally");  }  // 2. Константа заменена на изменяемое значение  boolean debugMode = System.getProperty("debug") != null;  if (debugMode) {  System.out.println("Debug info");  }  // 3. Метод с реальной логикой  if (checkCondition()) {  System.out.println("Conditional output");  }  } |

Методы устранения уязвимости при срабатывании детектора.

1. Удалите условия с константами

* Если условие всегда true - оставьте только тело блока
* Если всегда false - удалите весь блок кода

1. Замените константы на реальные проверки

* Используйте переменные, которые могут изменяться
* Добавьте реальную логику в методы-предикаты

1. Проверьте методы-предикаты

* Убедитесь, что они могут возвращать разные значения
* Если метод должен всегда возвращать одно значение - измените его сигнатуру на void