FB.NP\_METHOD\_PARAMETER\_RELAXING\_ANNOTATION

Метод всегда должен реализовывать контракт метода, который он переопределяет. Таким образом, если метод принимает параметр который помечен как @Nullable, не следует переопределять этот метод в подклассе методом, где этот параметр равен @Nonnull. Это нарушает контракт, согласно которому метод должен обрабатывать нулевой параметр.

Таблица №1 – Основные характеристики детектора FB.NP\_METHOD\_PARAMETER\_RELAXING\_ANNOTATION

|  |  |
| --- | --- |
| Категория критичности | Minor |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Static Analysis False Positive |
| CWE | CWE476 |
| CVE | --- |
| Возможные последствия | * Ложные предупреждения анализатора * Избыточные проверки на null * Усложнение кода |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Потенциально небезопасное | Корректная конструкция |
| public static void incorrectTest() {  // Некорректно: противоречивые аннотации  class Processor {  void process(@Nullable String input) {  if (input != null) {  save(input.toLowerCase()); // Анализатор может предупредить  }  }    void save(@Nonnull String data) {  System.out.println("Saving: " + data);  }  }    new Processor().process(null);  } | public static void correctTest() {  // Корректно: согласованные аннотации  class ProperProcessor {  void process(@Nullable String input) {  if (input == null) {  return;  }  save(input.toLowerCase()); // Без предупреждений  }    void save(@Nonnull String data) {  System.out.println("Saving: " + data);  }    // Или явное преобразование  void alternativeProcess(@Nullable String input) {  save(input != null ? input.toLowerCase() : "");  }  }    ProperProcessor processor = new ProperProcessor();  processor.process("test");  processor.process(null);  } |