FB.RCN\_REDUNDANT\_NULLCHECK\_WOULD\_HAVE\_BEEN\_A\_NPE

Здесь проверяется значение, чтобы определить, является ли оно нулевым, но это значение не может быть нулевым, поскольку ранее оно было разыменовано, и если бы оно было нулевым, при предыдущем разыменовании возникло бы исключение нулевого указателя. По сути, этот код и предыдущее разыменование расходятся во мнениях относительно того, может ли это значение быть нулевым. Либо проверка избыточна, либо предыдущее разыменование ошибочно

Таблица №1 – Основные характеристики детектора FB.RCN\_REDUNDANT\_NULLCHECK\_WOULD\_HAVE\_BEEN\_A\_NPE

|  |  |
| --- | --- |
| Категория критичности | Minor |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Redundant Code |
| CWE | CWE-476 |
| CVE | --- |
| Возможные последствия | Избыточный код |
| Комментарий (опционально) | Оптимизация проверок. |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Потенциально небезопасное | Корректная конструкция |
| public static void redundantCheck(String input) {  char firstChar = input.charAt(0); // NPE здесь, если input null  if (input != null) { // FB.RCN\_REDUNDANT\_NULLCHECK\_WOULD\_HAVE\_BEEN\_A\_NPE  System.out.println("First char: " + firstChar);  }  } | public static void correctCheck(String input) {  // Вариант 1: Проверка до разыменования  if (input != null && !input.isEmpty()) {  char firstChar = input.charAt(0);  System.out.println("First char: " + firstChar);  }    // Вариант 2: Допускаем NPE, если контракт метода запрещает null  char firstChar = input.charAt(0);  System.out.println("First char: " + firstChar);  }  } |