DEREF\_OF\_NULL.ANNOT.COND

Этот детектор срабатывает, когда указатель, который был аннотирован как @Nullable или аналогично, разыменовывается после условной проверки на null. Это может быть индикатором того, что условие проверки на null не полностью покрывает все возможные случаи.

Таблица №1 – Основные характеристики детектора

|  |  |
| --- | --- |
| Категория критичности | Normal |
| Надежность | Unknown |
| Источник | SvEng |
| Тип ошибки | Ошибка разработчика |
| CWE | — |
| CVE | — |
| Возможные последствия | Такое поведение может привести к возникновению NullPointerException в случае, если не все логические пути корректно исключают null. |
| Комментарий | Рекомендуется обеспечить явную и недвусмысленную проверку всех условий перед разыменованием, либо использовать конструкции, минимизирующие риск (например, Optional или ранний выход при null). |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Некорректное написание | Корректное написание |
| public static void incorrectTest(@Nullable String input) {  if (input != null && input.length() > 0) {  // Анализатор может посчитать, что условие не гарантирует, что input не null  System.out.println(input.charAt(0));  }  } | public static void correctTest(@Nullable String input) {  if (input == null) {  System.out.println("Строка равна null");  return;  }  // разыменование происходит после явной проверки на null  System.out.println(input.charAt(0));  } |

Методы устранения уязвимости при срабатывании детектора:

1. Сделайте проверку на null исчерпывающей. Убедитесь, что все логические пути, ведущие к разыменованию, гарантируют, что значение не является null.
2. Используйте ранний выход (return) после проверки на null. Если значение оказалось null, немедленно завершите выполнение метода, чтобы избежать последующего разыменования.
3. Если возможно, реорганизуйте код так, чтобы разыменование происходило непосредственно внутри блока if (input != null).
4. Используйте Optional (в Java) или аналогичные типы для явного представления возможности отсутствия значения. Это заставляет явно обрабатывать случай null перед доступом к значению.