DEREF\_OF\_NULL.COND

Означает, что разыменуется указатель, который был сравнен с NULL, при некоторых условиях, которые в точности не могут быть оценены анализатором (т.е. соответствующее значение атрибута отлично от истины и отражает только возможность).

Таблица №1 – Основные характеристики детектора

|  |  |
| --- | --- |
| Категория критичности | Normal |
| Надежность | High |
| Источник | SvEng |
| Тип ошибки | Ошибка разработчика |
| CWE | — |
| CVE | — |
| Возможные последствия | Может привести к ошибке выполнения, если условие, при котором указатель равен null, всё же выполнится во время работы программы. |
| Комментарий | Рекомендуется явно проверять указатели на null в условиях, где анализатор не может достоверно гарантировать их ненулевое значение, особенно перед разыменованием. |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Некорректное написание | Корректное написание |
| public static void incorrectTest(String maybeNull) {  // возможен NPE  if (maybeNull != null || maybeNull.length() > 0) {  System.out.println(maybeNull.length());  }  } | public static void correctTest(String maybeNull) {  if (maybeNull != null && maybeNull.length() > 0) { // корректная проверка  System.out.println(maybeNull.length());  }  } |

Методы устранения уязвимости при срабатывании детектора:

1. Измените условное выражение так, чтобы оно гарантировало, что разыменование происходит только при ненулевом значении. Используйте логическое AND (&&) вместо OR (||), если требуется, чтобы значение было не null и удовлетворяло другим условиям.
2. Если условие проверки на null и другое условие связаны логическим OR, рассмотрите разделение логики. Обработайте случай null отдельно (например, с помощью return или другого действия) перед выполнением операций, требующих ненулевого значения.
3. Если значение может быть null при определенных условиях, предусмотрите обработку этих условий до того, как будет предпринята попытка разыменования. Это может включать использование значения по умолчанию, пропуск операции или выброс исключения.
4. Используйте Optional (в Java) или аналогичные типы для явного представления возможности отсутствия значения. Это заставляет явно проверять наличие значения перед его использованием.