FB.TQ\_MAYBE\_SOURCE\_VALUE\_REACHES\_ALWAYS\_SINK

Значение, которое помечено как возможно не являющееся экземпляром значений, обозначенных квалификатором типа, и это значение гарантированно будет использоваться таким образом, который требует значений, обозначенных этим квалификатором типа.

Значение аннотировано потенциально не являющимся экземпляром значений определяемых квалификатором типов, хотя оно гарантированно будет использоваться значением, которое должно определяться этим квалификатором.

Таблица №1 – Основные характеристики детектора FB.TQ\_MAYBE\_SOURCE\_VALUE\_REACHES\_ALWAYS\_SINK

|  |  |
| --- | --- |
| Категория критичности | Minor |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Unsafe Data Flow |
| CWE | --- |
| CVE | --- |
| Возможные последствия | Утечка конфиденциальных данных в ненадежные каналы вывода; нарушение инкапсуляции при передаче изменяемых объектов; несанкционированный доступ к данным через публичные API; нарушение контрактов безопасности при передаче значений |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Потенциально небезопасное | Корректная конструкция |
| public static void incorrectTest() {  // 1. Передача непроверенных пользовательских данных  String userInput = getUserInput(); // @MaybeSource  processSensitiveData(userInput); // @AlwaysSink  // 2. Передача изменяемой коллекции  List<String> mutableList = new ArrayList<>();  mutableList.add("data");  storeImmutableData(mutableList); // Потенциальная модификация  // 3. Непроверенный URL  String untrustedUrl = request.getParameter("url");  openSecureConnection(untrustedUrl);  } | public static void correctTest() {  // 1. Валидация входных данных  String userInput = sanitizeInput(getUserInput());  processSensitiveData(userInput);  // 2. Защитное копирование  List<String> mutableList = new ArrayList<>();  mutableList.add("data");  storeImmutableData(new ArrayList<>(mutableList));  // 3. Проверка URL  String untrustedUrl = validateUrl(request.getParameter("url"));  openSecureConnection(untrustedUrl);  } |

Методы устранения уязвимости при срабатывании детектора.

1. Валидация входных данных;
2. Защитное копирование;
3. Использование иммутабельных объектов;
4. Явные контракты с аннотациями;
5. Контроль потока данных.