FB.RE\_POSSIBLE\_UNINTENDED\_PATTERN

Детектор определяет ошибку, в ходе которой вызывается строковая функция и "." или "|" передается параметру, который принимает регулярное выражение в качестве аргумента.

Таблица №1 – Основные характеристики детектора

|  |  |
| --- | --- |
| Категория критичности | MAJOR |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Ошибка разработчика |
| CWE | CWE185 |
| CVE | — |
| Возможные последствия | После возникновения ошибки FB.RE\_POSSIBLE\_UNINTENDED\_PATTERN выполнение программы может привести к неожиданным результатам при обработке строк, так как использование символов, таких как ".", "|" в контексте регулярных выражений может быть интерпретировано неверно. |
| Комментарий | Ошибка указывает на возможную ошибку в регулярном выражении, где символы ".", "|" могут быть использованы неправильно, что может повлиять на логику обработки данных. |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Некорректное написание | Корректное написание |
| public static void incorrectTest() {  String str = "abc";  var flag = str.matches("a.c"); // true  } | public static void correctTest() {  String str = "abc";  var flag = str.matches("a\\.c"); // false  } |

Методы устранения уязвимости при срабатывании детектора:

1. Проверьте, действительно ли символы . или | должны интерпретироваться как метасимволы регулярных выражений в данном контексте.
2. Если вы намеревались использовать . как литеральную точку, замените его на \\..
3. Если вы намеревались использовать | как литеральную вертикальную черту, замените его на \\|.
4. Будьте внимательны и экранируйте другие специальные символы регулярных выражений (\*, +, ?, ^, $, {, }, [, ], ( ,), \) обратным слешем \ (например, \\*, \+, \?, \^, \$, \\{, \\}, \\[, \\], \\(, \\), \\\\).
5. Используйте строковые литералы с необработанными строками (raw string literals), если они поддерживаются в вашем языке (в Java такой возможности нет до версии 15, и даже там это не напрямую связано с регулярными выражениями, но может улучшить читаемость при большом количестве обратных слешей). В Java вам нужно использовать двойные обратные слеши.
6. Тестируйте свои регулярные выражения с различными входными данными, чтобы убедиться, что они работают так, как вы ожидаете.