FB.RE\_CANT\_USE\_FILE\_SEPARATOR\_AS\_REGULAR\_EXPRESSION

В выделенным детектором участке кода используется File.separator, однако на некоторых операционных системах лучше использовать регулярные выражения. Это не удастся на платформах Windows, где File.separator представляет собой обратную косую черту, которая интерпретируется в регулярном выражении как escape-символ. Среди других опций можно использовать File.separatorChar=='\\' ? "\\\\" : File.separator вместо File.separator.

Таблица №1 – Основные характеристики детектора FB.RE\_CANT\_USE\_FILE\_SEPARATOR\_AS\_REGULAR\_EXPRESSION

|  |  |
| --- | --- |
| Категория критичности | Minor |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Regex Syntax Error |
| CWE | CWE-185 |
| CVE | --- |
| Возможные последствия | Некорректная работа регулярных выражений |
| Комментарий (опционально) | Особенно проблематично для кросс-платформенного кода. |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Потенциально небезопасное | Корректная конструкция |
| public static void incorrectUsage() {  String path = "C:\\temp\\file.txt";  boolean isTemp = path.matches(".\*" + File.separator + "temp" +  File.separator + ".\*");  // На Windows выбросит исключение из-за неэкранированных \  System.out.println("Is temp file (incorrect): " + isTemp);  }  } | public static void correctUsage() {  String path = "C:\\temp\\file.txt";    // Способ 1: Pattern.quote()  boolean safe1 = path.matches(".\*" + Pattern.quote(File.separator) + "temp" +  Pattern.quote(File.separator) + ".\*");    // Способ 2: Явное экранирование  String sep = File.separator.equals("\\") ? "\\\\" : File.separator;  boolean safe2 = path.matches(".\*" + sep + "temp" + sep + ".\*");    // Способ 3: Использование File API  boolean safe3 = new File(path).getParent().endsWith(File.separator + "temp");    System.out.println("Is temp file (safe1): " + safe1);  System.out.println("Is temp file (safe2): " + safe2);  System.out.println("Is temp file (safe3): " + safe3);  } |

Методы устранения уязвимости при срабатывании детектора

1. Использование Pattern.quote();
2. Явное экранирование обратной косой черты;
3. Использование File API вместо регулярных выражений;
4. Использование константных разделителей (если известна ОС).