FB.IIL\_PATTERN\_COMPILE\_IN\_LOOP\_INDIRECT

Метод создает одно и то же регулярное выражение внутри цикла, поэтому оно будет компилироваться на каждой итерации. Было бы более оптимально предварительно скомпилировать это регулярное выражение, используя Pattern.compile вне цикла.

Таблица №1 – Основные характеристики детектора FB.IIL\_PATTERN\_COMPILE\_IN\_LOOP\_INDIRECT

|  |  |
| --- | --- |
| Категория критичности | Minor |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Performance Issue |
| CWE | --- |
| CVE | --- |
| Возможные последствия | * Многократная компиляция одного и того же регулярного выражения * Снижение производительности * Избыточное использование памяти |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Потенциально небезопасное | Корректная конструкция |
| public static void incorrectTest() {  String[] inputs = {"test1", "test2", "test3"};  String regex = "test\\d"; // Одно и то же выражение    // Некорректно: косвенная компиляция в цикле через метод  for (String input : inputs) {  if (matchesPattern(input, regex)) { // Компиляция при каждом вызове  System.out.println("Match: " + input);  }  } } | public static void correctTest() {  String[] inputs = {"test1", "test2", "test3"};  String regex = "test\\d";    // Корректно: однократная компиляция  Pattern pattern = Pattern.compile(regex);  for (String input : inputs) {  if (pattern.matcher(input).matches()) {  System.out.println("Match: " + input);  }  }  } |