FB.SBSC\_USE\_STRINGBUFFER\_CONCATENATION

Детектор определяет ошибку, в ходе которой кажется, что этот метод строит строку, используя конкатенацию в цикле. На каждой итерации строка преобразуется в StringBuffer/StringBuilder, добавляется и преобразуется обратно в строку. Это может привести к квадратичной стоимости числа итераций, поскольку растущая строка копируется заново на каждой итерации.

Таблица №1 – Основные характеристики детектора

|  |  |
| --- | --- |
| Категория критичности | MAJOR |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Ошибка разработчика |
| CWE | — |
| CVE | — |
| Возможные последствия | Сбой программы при попытке модификации неизменяемого кортежа, некорректная работа функций, ожидающих изменяемую структуру данных |
| Комментарий | Невозможность добавить элемент в кортеж вызывает сбой там, где ожидался список |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Некорректное написание | Корректное написание |
| public static void incorrectTest() {  String result = "";  for (int i = 0; i < 1000; i++) {  // На каждой итерации создается новая строка!  result += i + ",";  }  System.out.println(result);  } | public static void correctTest() {  StringBuilder result = new StringBuilder();  for (int i = 0; i < 1000; i++) {  result.append(i).append(",");  }  System.out.println(result);  } |

Методы устранения уязвимости при срабатывании детектора:

1. Замените конкатенацию строк (+=) на использование StringBuilder (или StringBuffer для потокобезопасности).
2. Создайте экземпляр StringBuilder (или StringBuffer) перед началом цикла.
3. Внутри цикла используйте метод append() для добавления строк или других данных к StringBuilder/StringBuffer.
4. После завершения цикла вызовите метод toString() у StringBuilder/StringBuffer, чтобы получить итоговую строку.