FB.RV\_ABSOLUTE\_VALUE\_OF\_RANDOM\_INT

Детектор определяет ошибку, в ходе которой код генерирует случайное целое число со знаком, а затем вычисляет абсолютное значение этого случайного целого числа. Если число, возвращаемое генератором случайных чисел, равно Integer.MIN\_VALUE, то результат также будет отрицательным (поскольку Math.abs(Integer.MIN\_VALUE) == Integer.MIN\_VALUE). (Та же проблема возникает и для длинных значений).

Таблица №1 – Основные характеристики детектора

|  |  |
| --- | --- |
| Категория критичности | MAJOR |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Ошибка разработчика |
| CWE | CWE-440 |
| CVE | — |
| Возможные последствия | Ошибка может вызвать некорректное поведение алгоритмов, например, выход за границы массива или неверное распределение значений в структурах данных. |
| Комментарий | Во избежание проблем вместо Math.abs(hashCode()) лучше использовать конструкцию, исключающую Integer.MIN\_VALUE. |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Некорректное написание | Корректное написание |
| public static void incorrectTest(Random random) {  int number = random.nextInt();  // Возможен Integer.MIN\_VALUE:  int absNumber = Math.abs(number);  } | public static void correctTest(Random random) {  int number = random.nextInt();  int absNumber = (number == Integer.MIN\_VALUE) ? 0 : Math.abs(number);  } |

Методы устранения уязвимости при срабатывании детектора:

1. **Проанализируйте использование абсолютного значения случайного числа.** Убедитесь, что отрицательное значение, особенно Integer.MIN\_VALUE (или Long.MIN\_VALUE для nextLong()), не приведет к ошибкам в дальнейшей логике (например, выход за границы массива).
2. **Обработайте крайний случай Integer.MIN\_VALUE (или Long.MIN\_VALUE).** После получения случайного числа проверьте, равно ли оно минимальному значению. Если да, присвойте абсолютному значению 0 или другое безопасное положительное значение.
3. **Используйте Math.abs() с предварительной корректировкой.** Перед вычислением абсолютного значения можно добавить небольшое положительное число (например, 1) к случайному числу. Это сдвинет Integer.MIN\_VALUE в отрицательную область, где взятие абсолютного значения даст положительный результат (хотя и изменит распределение).
4. **Генерируйте случайные числа в нужном диапазоне.** Если вам требуется случайное неотрицательное число в определенном диапазоне, используйте методы Random.nextInt(int bound) или Random.nextLong(long bound), чтобы избежать генерации отрицательных чисел в первую очередь.
5. **Если требуется случайное число в диапазоне, включающем ноль, и вы хотите избежать Integer.MIN\_VALUE, можно генерировать случайное число в диапазоне [0, Integer.MAX\_VALUE] и при необходимости умножать его на -1 с определенной вероятностью (но это не гарантирует отсутствие Integer.MIN\_VALUE при исходной генерации).**