FB.VO\_VOLATILE\_REFERENCE\_TO\_ARRAY

Это объявляет изменчивую ссылку на массив, что может быть не тем, что необходимо. При использовании изменчивой ссылки на массив операции чтения и записи ссылки на массив считаются энергозависимыми, но элементы массива являются энергонезависимыми. Чтобы получить изменчивые элементы массива, необходимо будет использовать один из классов атомарных массивов в java.util.concurrent (представленный в Java 5.0).

Таблица №1 – Основные характеристики детектора FB.VO\_VOLATILE\_REFERENCE\_TO\_ARRAY

|  |  |
| --- | --- |
| Категория критичности | Minor |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Concurrency Vulnerability |
| CWE | --- |
| CVE | --- |
| Возможные последствия | Видимость изменений элементов массива не гарантируется; гонки данных при модификации элементов массива; непредсказуемое поведение при параллельном доступе |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Потенциально небезопасное | Корректная конструкция |
| public static void incorrectTest() {  class ArrayHolder {  private volatile int[] array = new int[10];    public void updateArray() {  for (int i = 0; i < array.length; i++) {  array[i] = i; // Изменения элементов не гарантированно видны другим потокам  }  }  }    ArrayHolder holder = new ArrayHolder();  Thread t1 = new Thread(holder::updateArray);  Thread t2 = new Thread(() -> {  try {  Thread.sleep(10);  System.out.println("Incorrect array[5]: " + holder.array[5]);  } catch (InterruptedException e) {  Thread.currentThread().interrupt();  }  });    t1.start();  t2.start();  } | public static void correctTest() {  class ArrayHolder {  private final int[] array = new int[10];    public synchronized void updateArray() {  for (int i = 0; i < array.length; i++) {  array[i] = i; // Синхронизированная модификация  }  }    public synchronized int getValue(int index) {  return array[index]; // Синхронизированное чтение  }  }    ArrayHolder holder = new ArrayHolder();  Thread t1 = new Thread(holder::updateArray);  Thread t2 = new Thread(() -> {  try {  Thread.sleep(10);  System.out.println("Correct array[5]: " + holder.getValue(5));  } catch (InterruptedException e) {  Thread.currentThread().interrupt();  }  });    t1.start();  t2.start();  } |