FB.RANGE\_ARRAY\_OFFSET

Метод вызывается с параметром массива и параметром смещения, но смещение выходит за пределы. Это приведет к исключению IndexOutOfBoundsException во время выполнения.

Таблица №1 – Основные характеристики детектора FB.RANGE\_ARRAY\_OFFSET

|  |  |
| --- | --- |
| Категория критичности | Minor |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Runtime Exception Potential |
| CWE | --- |
| CVE | --- |
| Возможные последствия | * ArrayIndexOutOfBoundsException * Повреждение памяти * Непредсказуемое поведение |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Потенциально небезопасное | Корректная конструкция |
| public static void incorrectTest() {  // Некорректно: отсутствие проверки offset/length  byte[] data = {1, 2, 3, 4, 5};  int offset = 3;  int length = 10; // Превышает размер массива    try {  processData(data, offset, length); // Опасный вызов  } catch (ArrayIndexOutOfBoundsException e) {  System.out.println("Caught ArrayIndexOutOfBoundsException");  }  } | public static void correctTest() {  // Корректно: проверка границ перед обработкой  byte[] data = {1, 2, 3, 4, 5};  int offset = 3;  int length = 10;    if (offset >= 0 && length >= 0 && offset + length <= data.length) {  processDataSafely(data, offset, length);  } else {  System.out.println("Invalid offset/length parameters");  }    // Альтернатива с корректировкой параметров  int safeOffset = Math.max(0, offset);  int safeLength = Math.min(length, data.length - safeOffset);  if (safeLength > 0) {  processDataSafely(data, safeOffset, safeLength);  }  } |