FB.UG\_SYNC\_SET\_UNSYNC\_GET

Детектор определяет ошибку, в ходе которой класс содержит методы get и set с одинаковыми именами, где метод set синхронизируется, а метод get — нет. Это может привести к некорректному поведению во время выполнения, поскольку вызывающие метод get не обязательно увидят согласованное состояние объекта. Метод get следует сделать синхронизированным.

Таблица №1 – Основные характеристики детектора

|  |  |
| --- | --- |
| Категория критичности | MAJOR |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Ошибка разработчика |
| CWE | — |
| CVE | — |
| Возможные последствия | Несинхронизированный get()-метод в сочетании с синхронизированным set()-методом может привести к некорректному чтению данных из-за отсутствия гарантии видимости изменений между потоками. |
| Комментарий | Чтобы избежать проблем с конкурентным доступом, метод get() также следует сделать синхронизированным либо использовать volatile для переменной. Альтернативно можно применить java.util.concurrent структуры, такие как AtomicReference или ReadWriteLock. |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Некорректное написание | Корректное написание |
| public static void incorrectTest() {  class Test {  private int value;  public synchronized void setValue(int value) {  this.value = value;  }  public int getValue() {  return value;  }  }  Test test = new Test();  test.setValue(1);  System.out.println(test.getValue());  } | public static void correctTest() {  class Test {  // Volatile гарантирует видимость изменений между потоками  private volatile int value;  public void setValue(int value) {  this.value = value;  }  public int getValue() {  return value;  }  }  Test test = new Test();  test.setValue(1);  System.out.println(test.getValue());  } |