FB.MWN\_MISMATCHED\_NOTIFY

Детектор определяет ошибку, в ходе которой метод вызывает Object.notify() или Object.notifyAll() без явной блокировки объекта. Вызов notify() или notifyAll() без удержания блокировки приведет к выдаче исключения IllegalMonitorStateException.

Таблица №1 – Основные характеристики детектора

|  |  |
| --- | --- |
| Категория критичности | MAJOR |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Ошибка разработчика |
| CWE | — |
| CVE | — |
| Возможные последствия | Вызов Object.notify() или Object.notifyAll() без удержания соответствующей блокировки приведет к выбросу исключения IllegalMonitorStateException, что может нарушить корректное выполнение программы и привести к некорректной работе механизма синхронизации потоков. |
| Комментарий | Для предотвращения данной ошибки вызовы notify() и notifyAll() следует осуществлять внутри блока synchronized, удерживающего монитор соответствующего объекта. |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Некорректное написание | Корректное написание |
| public static void incorrectTest() {  class Test {  private final Object lock = new Object();  public void method() {  // без захвата блокировки  lock.notify();  }  }  new Test().method();  } | public static void correctTest() {  class Test {  private final Object lock = new Object();  public void method() {  synchronized (lock) {  lock.notify();  }  }  }  new Test().method();  } |

Методы устранения уязвимости при срабатывании детектора:

1. **Оберните вызовы notify() и notifyAll() в synchronized блок:** Убедитесь, что вызов notify() или notifyAll() происходит внутри блока synchronized, который синхронизирован на том же объекте, который используется для wait().
2. **Проверьте владение монитором:** Перед вызовом notify() или notifyAll() убедитесь, что текущий поток владеет монитором соответствующего объекта.
3. **Следуйте паттерну wait()/notify()/notifyAll():** Все три метода должны использоваться в связке и с соблюдением правил владения монитором. Обычно wait() вызывается внутри цикла, проверяющего условие, и notify()/notifyAll() вызываются после изменения этого условия.
4. **Используйте более высокоуровневые средства синхронизации (предпочтительно):** Пакет java.util.concurrent предоставляет более гибкие и безопасные средства синхронизации, такие как Lock, Condition, CountDownLatch, Semaphore и другие, которые часто предпочтительнее низкоуровневых wait()/notify()/notifyAll(). Рассмотрите возможность их использования вместо явной работы с мониторами объектов.