FB.JLM\_JSR166\_LOCK\_MONITORENTER

Детектор определяет ошибку, в ходе которой метод выполняет синхронизацию объекта, реализующего java.util.concurrent.locks.Lock. Такой объект блокируется/разблокируется с помощью метода Acquire()/release(), а не с помощью конструкции Synchronized (...).

Таблица №1 – Основные характеристики детектора

|  |  |
| --- | --- |
| Категория критичности | MAJOR |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Ошибка разработчика |
| CWE | — |
| CVE | — |
| Возможные последствия | Использование конструкции synchronized на объекте, реализующем интерфейс java.util.concurrent.locks.Lock, может привести к путанице и неправильному управлению блокировками. |
| Комментарий | Для объектов, реализующих Lock, следует использовать явные методы блокировки и разблокировки (lock()/unlock()), а не синхронизированные блоки, чтобы сохранить ясность и корректность управления параллельным доступом. |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Некорректное написание | Корректное написание |
| public static void incorrectTest() {  class Test {  private final Lock lock = new ReentrantLock();  public void method() {  // синхронизация на объекте Lock  synchronized (lock) {  System.out.println("Executing critical section");  }  }  }  } | public static void correctTest() {  class Test {  private final Lock lock = new ReentrantLock();  public void method() {  // явная блокировка  lock.lock();  try {  // ...  System.out.println("Executing critical section");  } finally {  // разблокировка  lock.unlock();  }  }  }  } |

Методы устранения уязвимости при срабатывании детектора:

1. **Замените synchronized блоком на явное использование lock() и unlock():** Вместо synchronized (lock) используйте lock.lock() для получения блокировки и поместите критическую секцию в блок try, а вызов lock.unlock() в блок finally, чтобы гарантировать освобождение блокировки даже при возникновении исключений.
2. **Удалите synchronized:** Если синхронизация на объекте Lock была добавлена ошибочно, и управление блокировками должно осуществляться только через методы Lock API, удалите блок synchronized.
3. **Проверьте документацию и примеры использования Lock:** Обратитесь к документации java.util.concurrent.locks.Lock и примерам ее использования, чтобы понять правильный способ работы с блокировками этого API.
4. **Убедитесь в консистентности использования механизмов синхронизации:** Выберите один подход к управлению параллельным доступом (либо встроенная синхронизация через synchronized, либо явные блокировки через Lock API) и последовательно используйте его в своем коде для конкретного объекта блокировки.