FB.SWL\_SLEEP\_WITH\_LOCK\_HELD

Детектор определяет ошибку, в ходе которой метод вызывает Thread.sleep() с удержанной блокировкой. Это может привести к очень низкой производительности и масштабируемости или к взаимоблокировке, поскольку другие потоки могут ожидать получения блокировки. Гораздо лучше вызвать функцию wait() для блокировки, которая снимает блокировку и позволяет запускаться другим потокам.

Таблица №1 – Основные характеристики детектора

|  |  |
| --- | --- |
| Категория критичности | MAJOR |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Ошибка разработчика |
| CWE | — |
| CVE | — |
| Возможные последствия | Вызов Thread.sleep() при удержании блокировки может привести к снижению производительности, ухудшению масштабируемости и возможным взаимоблокировкам. Другие потоки, ожидающие освобождения блокировки, могут оставаться в состоянии ожидания дольше, чем необходимо, что замедлит работу программы. |
| Комментарий | Вместо Thread.sleep() рекомендуется использовать wait(), так как он освобождает блокировку и позволяет другим потокам выполняться. |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Некорректное написание | Корректное написание |
| private static final Object lock = new Object();  public static void incorrectTest() throws InterruptedException {  synchronized (lock) {  // ...  Thread.sleep(5000);  }  } | private static final Object lock = new Object();  public static void correctTest() throws InterruptedException {  synchronized (lock) {  // ...  lock.wait();  }  } |

Методы устранения уязвимости при срабатывании детектора:

1. **Использование wait() вместо Thread.sleep():** Вместо того чтобы вызывать Thread.sleep() внутри синхронизированного блока, используйте метод wait() объекта блокировки. wait() приостанавливает текущий поток и **освобождает** удерживаемую блокировку, позволяя другим потокам получить доступ к синхронизированному ресурсу. Поток, вызвавший wait(), переходит в состояние ожидания до тех пор, пока другой поток не вызовет метод notify() или notifyAll() для этого же объекта блокировки, либо не истечет необязательный тайм-аут.
2. **Перемещение вызова Thread.sleep() за пределы синхронизированного блока:** Если логика работы позволяет, рассмотрите возможность выполнения операции, требующей задержки, после освобождения блокировки. Это предотвратит блокировку других потоков на время сна.