FB.DM\_USELESS\_THREAD

Этот метод создает поток без указания метода запуска либо путем наследования от класса Thread, либо путем передачи объекта Runnable. Таким образом, это не приносит ничего, кроме траты времени.

Таблица №1 – Основные характеристики детектора FB.DM\_USELESS\_THREAD

|  |  |
| --- | --- |
| Категория критичности | Minor |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Performance/Concurrency Issue |
| CWE | --- |
| CVE | --- |
| Возможные последствия | Нагрузка на систему из-за создания лишних потоков; утечка ресурсов (незавершенные потоки); сложность отладки и управления потоками; потенциальные deadlock-и при неправильном управлении |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Потенциально небезопасное | Корректная конструкция |
| public static void incorrectTest() {  // Плохо: создание потока без реальной необходимости  new Thread(() -> {  System.out.println("Этот поток не делает ничего полезного");  }).start();  // Плохо: создание потока, который сразу завершается  new Thread(() -> {}).start();  // Плохо: создание потока без сохранения ссылки (нельзя контролировать)  for (int i = 0; i < 5; i++) {  new Thread(() -> {  System.out.println("Бесполезный поток #" + Thread.currentThread().getId());  }).start();  }  } | public static void correctTest() {  // Хорошо: использование пула потоков  ExecutorService executor = Executors.newFixedThreadPool(2);  // Полезная работа в потоке  executor.submit(() -> {  System.out.println("Полезная задача 1 выполняется в пуле потоков");  });  // Еще одна полезная задача  executor.submit(() -> {  System.out.println("Полезная задача 2 выполняется в пуле потоков");  });  // Завершение работы пула  executor.shutdown();  } |