FB.STCAL\_INVOKE\_ON\_STATIC\_CALENDAR\_INSTANCE

Детектор определяет ошибку, в ходе которой несмотря на то, что JavaDoc не содержит никаких намеков на это, календари по своей сути небезопасны для многопоточного использования. Детектор обнаружил вызов экземпляра Calendar, полученный через статическое поле. Это выглядит подозрительно.

Таблица №1 – Основные характеристики детектора

|  |  |
| --- | --- |
| Категория критичности | MAJOR |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Ошибка разработчика |
| CWE | CWE366 |
| CVE | — |
| Возможные последствия | Использование общего статического экземпляра Calendar в многопоточной среде может привести к некорректному поведению программы из-за одновременного изменения состояния объекта разными потоками. |
| Комментарий | Вместо использования статического Calendar рекомендуется создавать новые экземпляры при каждом обращении или использовать ThreadLocal<Calendar>. Также можно заменить Calendar на java.time API (LocalDateTime, ZonedDateTime), который является потокобезопасным. |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Некорректное написание | Корректное написание |
| private static final Calendar calendar = Calendar.getInstance();  public static void incorrectTest() {  // Потенциальная проблема  int year = calendar.get(Calendar.YEAR);  System.out.println(year);  } | public static void correctTest() {  int year = LocalDate.now().getYear();  System.out.println(year);  } |

Методы устранения уязвимости при срабатывании детектора:

1. Найдите все места, где вызываются методы экземпляра статического объекта Calendar.
2. Рассмотрите возможность создания нового экземпляра Calendar в каждом месте использования.
3. Используйте ThreadLocal<Calendar>, если создание нового экземпляра при каждом использовании является неэффективным или если требуется сохранять состояние Calendar в пределах одного потока.
4. Рекомендуемый подход: Замените java.util.Calendar на классы из пакета java.time. java.time API предоставляет потокобезопасные и более удобные классы для работы с датой и временем.
5. Если по какой-либо причине необходимо использовать статический Calendar, обеспечьте внешнюю синхронизацию при каждом обращении к нему. Однако этот подход является менее предпочтительным из-за потенциальных проблем с производительностью и сложности управления синхронизацией.