FB.STCAL\_STATIC\_SIMPLE\_DATE\_FORMAT\_INSTANCE

Детектор определяет ошибку, в ходе которой как указано в JavaDoc, DateFormats по своей сути небезопасны для многопоточного использования. Совместное использование одного экземпляра за границами потоков без надлежащей синхронизации приведет к нестабильному поведению приложения.

Таблица №1 – Основные характеристики детектора

|  |  |
| --- | --- |
| Категория критичности | MAJOR |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Ошибка разработчика |
| CWE | CWE366 |
| CVE | — |
| Возможные последствия | Использование статического экземпляра SimpleDateFormat в многопоточной среде без синхронизации может привести к непредсказуемым результатам, таким как некорректное форматирование или разбор дат, возникновение исключений и искажение данных. |
| Комментарий | Вместо статического SimpleDateFormat рекомендуется использовать ThreadLocal<SimpleDateFormat> или заменить его на java.time.format.DateTimeFormatter, который является потокобезопасным и предпочтителен в современных Java-приложениях. |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Некорректное написание | Корректное написание |
| public static void incorrectTest() {  class Test {  private static final SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd");  // ...  }  } | public static void correctTest() {  class Test {  private static final DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd");  // ...  }  } |

Методы устранения уязвимости при срабатывании детектора:

1. **Использование ThreadLocal<SimpleDateFormat>:** Вместо статического экземпляра SimpleDateFormat создайте ThreadLocal, который будет предоставлять уникальный экземпляр SimpleDateFormat для каждого потока. Это гарантирует, что каждый поток работает со своим собственным экземпляром, избегая проблем с конкурентным доступом.
2. **Переход на java.time.format.DateTimeFormatter:** В современных Java-приложениях рекомендуется использовать классы из пакета java.time, введенного в Java 8. Класс DateTimeFormatter является потокобезопасным и предоставляет более гибкие возможности для форматирования и разбора дат и времени.