FB.IC\_INIT\_CIRCULARITY

Детектор определяет ошибку, в ходе которой в статических инициализаторах двух классов, на которые ссылается экземпляр ошибки, была обнаружена цикличность. Из-за такой цикличности может возникнуть множество видов неожиданного поведения.

Таблица №1 – Основные характеристики детектора

|  |  |
| --- | --- |
| Категория критичности | MAJOR |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Ошибка разработчика |
| CWE | — |
| CVE | — |
| Возможные последствия | Циклическая зависимость между статическими инициализаторами может привести к StackOverflowError или к недопустимому состоянию программы, что может вызвать проблемы с загрузкой классов, зависания или непредсказуемое поведение. |
| Комментарий | Для устранения цикличности пересмотрите зависимость между инициализаторами классов и попытайтесь устранить циклическую ссылку. Возможно, стоит использовать ленивую инициализацию или изменить структуру классов. |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Некорректное написание | Корректное написание |
| public static void incorrectTest() {  class A {  static {  System.out.println("Статический блок A");  B.getValue();  }  public static void getValue() {  System.out.println("Метод A.getValue()");  }  }  class B {  static {  System.out.println("Статический блок B");  A.getValue();  }  public static void getValue() {  System.out.println("Метод B.getValue()");  }  }  new A(); // приведет к цикличности инициализации  } | public static void correctTest() {  class A {  static {  System.out.println("Статический блок A");  }  public static void getValue() {  System.out.println("Метод A.getValue()");  }  }  class B {  static {  System.out.println("Статический блок B");  }  public static void getValue() {  System.out.println("Метод B.getValue()");  }  }  new A();  new B();  } |

Методы устранения уязвимости при срабатывании детектора:

1. **Используйте ленивую инициализацию:** Отложите инициализацию статических полей до момента первого обращения к ним. Это можно сделать, обернув инициализацию в статический метод-геттер и создавая экземпляр поля только при первом вызове этого метода.
2. **Реструктурируйте классы:** Пересмотрите зависимости между классами. Возможно, стоит выделить общие статические константы или методы в отдельный класс, чтобы разорвать прямые циклические зависимости.
3. **Избегайте прямого доступа к статическим полям других классов в статических инициализаторах:** Постарайтесь минимизировать взаимодействие между статическими инициализаторами разных классов. Если необходимо использовать информацию из другого класса, рассмотрите возможность использования статических методов вместо прямого доступа к полям.
4. **Проанализируйте граф зависимостей:** Визуализация зависимостей между классами может помочь выявить циклические ссылки в статических инициализаторах.
5. **Тестируйте загрузку классов:** Тщательно протестируйте инициализацию классов, особенно в сложных сценариях, чтобы убедиться в отсутствии проблем, связанных с циклическими зависимостями.