USE\_AFTER\_RELEASE

Указывает на ситуацию, когда ресурс (часто это память или системный ресурс) используется после того, как он был освобожден или удален.

Таблица №1 – Основные характеристики детектора

|  |  |
| --- | --- |
| Категория критичности | Normal |
| Надежность | High |
| Источник | SvEng |
| Тип ошибки | Ошибка разработчика |
| CWE | CWE415;CWE416 |
| CVE | — |
| Возможные последствия | Использование освобожденного ресурса может привести к неопределенному поведению программы, включая сбои, повреждение данных, утечки памяти и нарушение безопасности. |
| Комментарий | Необходимо тщательно отслеживать жизненный цикл ресурсов, используя инструменты для автоматического освобождения памяти или другие механизмы контроля, чтобы предотвратить использование после освобождения. |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Некорректное написание | Корректное написание |
| public static void incorrectTest() {  FileInputStream fileInputStream = null;  try {  fileInputStream = new FileInputStream("test.txt");  fileInputStream.close();  // Попытка использовать после закрытия — ошибка  fileInputStream.read();  } catch (IOException e) {  throw new RuntimeException(e);  }  } | public static void correctTest() {  try (FileInputStream fileInputStream = new FileInputStream("test.txt")) {  int data = fileInputStream.read();  System.out.println(data);  } catch (IOException e) {  throw new RuntimeException(e);  }  } |

Методы устранения уязвимости при срабатывании детектора:

1. Не используйте ресурс после его освобождения. Убедитесь, что после вызова метода, освобождающего ресурс (например, close(), delete(), free()), никакие операции с этим ресурсом больше не выполняются.
2. Установите ссылку на ресурс в null после освобождения. После освобождения ресурса присвойте переменной, которая на него ссылалась, значение null. Это поможет предотвратить случайное повторное использование и может вызвать исключение NullPointerException при попытке доступа, облегчая отладку.
3. Используйте RAII (Resource Acquisition Is Initialization) или аналогичные паттерны. В Java используйте try-with-resources для автоматического закрытия ресурсов.