HANDLE\_LEAK.EXCEPTION

Генерируется, когда ресурс не освобождается в случае возникновения исключения. Если ресурс не освобождается при обработке исключений, это может привести к утечке этого ресурса при возникновении ошибки.

Таблица №1 – Основные характеристики детектора

|  |  |
| --- | --- |
| Категория критичности | Normal |
| Надежность | Unknown |
| Источник | Roslyn;SvEng |
| Тип ошибки | Ошибка разработчика |
| CWE | CWE404 |
| CVE | — |
| Возможные последствия | Может привести к утечке системных ресурсов (например, файловых дескрипторов, потоков), особенно при повторяющихся ошибках, что влияет на стабильность и производительность приложения. |
| Комментарий | Рекомендуется использовать конструкции, гарантирующие освобождение ресурсов даже при исключениях, например, блоки finally или try-with-resources, чтобы обеспечить корректное завершение работы с ресурсами. |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Некорректное написание | Корректное написание |
| public static void incorrectTest() {  FileInputStream fis = null;  try {  fis = new FileInputStream("test.txt");  // возможное исключение  int data = fis.read();  System.out.println(data);  } catch (IOException e) {  // здесь исключение обработано, но ресурс не закрыт!  System.err.println("Ошибка чтения файла: " + e.getMessage());  // Если исключение произошло ДО fis.close() — файл останется открытым  }  } | public static void correctTest() {  try (FileInputStream fis = new FileInputStream("test.txt")) {  int data = fis.read();  System.out.println(data);  } catch (IOException e) {  System.err.println("Ошибка чтения файла: " + e.getMessage());  }  } |