UNCHECKED\_FUNC\_RES.USER.STRICT

Предупреждение UNCHECKED\_FUNC\_RES.USER указывает на то, что результат функции не проверяется перед использованием. Это может привести к непредсказуемому поведению программы, если функция возвращает ошибку или неожиданное значение. Например, если функция должна возвращать указатель на объект, но происходит ошибка и возвращается NULL, а затем этот NULL используется без проверки - это приведет к ошибке. «STRICT» указывает на строгий уровень проверки.

Таблица №1 – Основные характеристики детектора

|  |  |
| --- | --- |
| Категория критичности | Normal |
| Надежность | Unknown |
| Источник | SvEng |
| Тип ошибки | Ошибка разработчика |
| CWE | — |
| CVE | — |
| Возможные последствия | Игнорирование возвращаемых значений может привести к сбоям, утечкам ресурсов или некорректному поведению программы, даже в сложных или косвенных сценариях. |
| Комментарий | Даже при кажущейся незначительности, все возвращаемые значения функций следует проверять для повышения надежности и устойчивости кода. |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Некорректное написание | Корректное написание |
| public static void incorrectTest() {  // Используем результат без проверки  String data = fetchDataFromServer();  System.out.println(data.toUpperCase()); // NullPointerException  } | public static void correctTest() {  String data = fetchDataFromServer();  if (data != null) {  System.out.println(data.toUpperCase());  } else {  System.out.println("Data not available");  }  } |

Методы устранения уязвимости при срабатывании детектора:

1. Всегда проверяйте возвращаемое значение. Независимо от кажущейся надежности функции, всегда явно проверяйте ее возвращаемое значение перед его использованием.
2. Обрабатывайте все возможные результаты. Учитывайте все потенциальные значения, которые может вернуть функция (успех, различные типы ошибок, особые состояния) и реализуйте обработку для каждого из них.
3. Используйте строгие проверки. Применяйте строгие условия проверки (например, != null, сравнение с конкретными кодами ошибок) вместо неявных или предположительных проверок.