FB.DM\_NUMBER\_CTOR

Детектор определяет ошибку, в ходе которой использование new Integer(int) гарантированно всегда приводит к созданию нового объекта, тогда как Integer.valueOf(int) позволяет кэшировать значения, выполняемые компилятором, библиотекой классов или JVM. Использование кэшированных значений позволяет избежать выделения объектов, и код будет работать быстрее.

Таблица №1 – Основные характеристики детектора

|  |  |
| --- | --- |
| Категория критичности | MAJOR |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Ошибка разработчика |
| CWE | — |
| CVE | — |
| Возможные последствия | Создание объектов числовых обёрток через конструкторы (new Integer(int), new Double(double)) приводит к избыточному потреблению памяти и дополнительной нагрузке на сборщик мусора, поскольку каждый вызов создаёт новый объект вместо использования кэшированных значений. |
| Комментарий | Рекомендуется использовать фабричные методы, такие как Integer.valueOf(int), Double.valueOf(double), которые позволяют повторно использовать уже созданные объекты и сокращают накладные расходы, улучшая производительность программы. |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Некорректное написание | Корректное написание |
| public static void incorrectTest() {  // Всегда создается новый объект  // deprecated since version 9  Integer a = new Integer(10);  System.out.println(a);  } | public static void correctTest() {  // Использует кэшированные объекты  Integer a = Integer.valueOf(10);  System.out.println(a);  } |

Методы устранения уязвимости при срабатывании детектора:

1. **Замените конструкторы на valueOf():** Вместо использования new Integer(int), new Long(long) и т.д., используйте соответствующие статические методы valueOf(): Integer.valueOf(int), Long.valueOf(long), Short.valueOf(short), Byte.valueOf(byte), Character.valueOf(char).
2. **Обратите внимание на автоупаковку:** В большинстве случаев Java автоматически выполняет упаковку при присваивании примитивного значения переменной типа-обертки (например, Integer a = 10;). В таких случаях нет необходимости явно вызывать конструктор или valueOf(). Однако, если вы явно вызываете конструктор, замените его на valueOf().
3. **Проверьте весь код:** Найдите все места, где используются конструкторы числовых оберток, и замените их на valueOf().
4. **Исключения:** Для типов Float и Double кэширование не так эффективно, и разница между конструктором и valueOf() может быть менее значительной. Тем не менее, следование единообразному стилю и использование valueOf() даже для них считается хорошей практикой.
5. **Удалите устаревший код:** Обратите внимание, что конструкторы числовых оберток были объявлены устаревшими (deprecated) начиная с Java 9, что является еще одной причиной для перехода на valueOf() или автоупаковку.