FB.DM\_DEFAULT\_ENCODING

Детектор определяет ошибку, в ходе которой вызов метода, который выполняет преобразование байта в строку (или строку в байт) и предполагает, что кодировка платформы по умолчанию подходит. Это приведет к тому, что поведение приложения будет различаться на разных платформах. Используйте альтернативный API и явно укажите имя набора символов или объект Charset.

Таблица №1 – Основные характеристики детектора

|  |  |
| --- | --- |
| Категория критичности | MAJOR |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Ошибка разработчика |
| CWE | — |
| CVE | — |
| Возможные последствия | Использование кодировки по умолчанию может привести к непредсказуемым результатам при работе с текстом на разных платформах, поскольку кодировка может различаться, что вызовет проблемы с интерпретацией и сохранением данных. |
| Комментарий | Для обеспечения корректности работы с текстовыми данными следует всегда явно указывать кодировку, используя Charset или указав название кодировки, чтобы избежать зависимостей от платформы. |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Некорректное написание | Корректное написание |
| public static void incorrectTest() {  String str = "Hello, world!";  // без явной кодировки  byte[] bytes = str.getBytes();  // без указания кодировки  String decodedStr = new String(bytes);  System.out.println(decodedStr);  } | public static void correctTest() {  String str = "Hello, world!";  // с явной кодировкой  byte[] bytes = str.getBytes(StandardCharsets.UTF\_8);  // с явной кодировкой  String decodedStr = new String(bytes, StandardCharsets.UTF\_8);  System.out.println(decodedStr);  } |

Методы устранения уязвимости при срабатывании детектора:

1. **Явно указывайте кодировку:** При использовании методов для преобразования между байтами и строками (например, String.getBytes(), конструкторы String из массива байтов, методы InputStreamReader, OutputStreamWriter и т.д.), всегда передавайте объект Charset (например, StandardCharsets.UTF\_8) или строковое имя кодировки в качестве аргумента.
2. **Используйте StandardCharsets:** Класс java.nio.charset.StandardCharsets предоставляет константы для наиболее распространенных кодировок (UTF-8, UTF-16, ISO-8859-1 и др.). Используйте их для обеспечения типобезопасности и читаемости кода.
3. **Определите подходящую кодировку:** Выберите кодировку, которая наилучшим образом подходит для обрабатываемых данных. UTF-8 является хорошим универсальным вариантом для большинства текстовых данных.
4. **Проверьте существующий код:** Найдите все места в коде, где происходит преобразование байтов в строки или наоборот, и убедитесь, что кодировка указана явно.
5. **Будьте внимательны к внешним источникам данных:** Если вы получаете данные из внешних источников (файлы, сеть и т.д.), убедитесь, что вы знаете их кодировку и используете ее при чтении. При записи данных также явно указывайте кодировку.