FB.ME\_ENUM\_FIELD\_SETTER

Этот общедоступный метод, объявленный в общедоступном перечислении, безоговорочно устанавливает поле перечисления, поэтому это поле может быть изменено вредоносным кодом или случайно из другого пакета. Хотя изменяемые поля перечислений могут использоваться для ленивой инициализации, открывать их для внешнего мира — плохая практика. Стоит рассмотреть возможность удаления этого метода или объявления его частным для пакета.

Таблица №1 – Основные характеристики детектора FB.ME\_ENUM\_FIELD\_SETTER

|  |  |
| --- | --- |
| Категория критичности | Minor |
| Надежность | Unknown |
| Источник | SpotBugs |
| Тип ошибки | Design Flaw |
| CWE | --- |
| CVE | --- |
| Возможные последствия | * Нарушение инвариантов enum * Непредсказуемое поведение * Проблемы с безопасностью |

Таблица №2 – Примеры срабатывания детектора и устранения уязвимости

|  |  |
| --- | --- |
| Потенциально небезопасное | Корректная конструкция |
| public static void incorrectTest() {  // Некорректно: enum с изменяемым полем и сеттером  enum Status {  ACTIVE("A"), INACTIVE("I");    private String code;    Status(String code) {  this.code = code;  }    public void setCode(String code) { // Опасный сеттер  this.code = code;  }    public String getCode() {  return code;  }  }    Status.ACTIVE.setCode("X"); // Изменение enum!  System.out.println("Modified enum: " + Status.ACTIVE.getCode());  } | public static void correctTest() {  // Корректно: неизменяемый enum  enum Status {  ACTIVE("A"), INACTIVE("I");    private final String code; // final поле    Status(String code) {  this.code = code;  }    public String getCode() {  return code;  }  }    System.out.println("Proper enum: " + Status.ACTIVE.getCode());    // Альтернатива для сложных случаев  enum AdvancedStatus {  ACTIVE {  @Override  public String getDisplayName() {  return "Active";  }  },  INACTIVE {  @Override  public String getDisplayName() {  return "Inactive";  }  };    public abstract String getDisplayName();  }  } |