XX系统源代码安全审计报告

**—— 安全团队**

目录

[一. 项目详情 2](#_Toc8748231)

[1.1 审计对象 2](#_Toc8748232)

[1.2 项目详情 2](#_Toc8748233)

[二. 漏洞详情 3](#_Toc8748234)

[2.1 SQL注入漏洞 3](#_Toc8748235)

[三. 审计结果及修复建议 4](#_Toc8748236)

# 项目详情

## 审计对象

|  |  |  |  |
| --- | --- | --- | --- |
| 业务名称 |  | 代码项目名称 |  |
| 项目负责人 |  | 项目负责人邮箱 |  |

## 项目详情

|  |  |
| --- | --- |
| 项目描述 |  |
| 项目版本\* |  |
| 开发语言 |  |
| 项目框架 |  |
| GitLab仓库地址 |  |
| 测试地址 |  |
| 测试账号 |  |

注：项目版本号必须填写

# 漏洞详情

## SQL注入漏洞

**漏洞描述：**

SQL注入攻击是web开发中最常见的一种安全漏洞。由于程序没有过滤用户的输入，攻击者通过控制提交的参数注入恶意SQL语句，服务器端在处理时将攻击者的输入作为原始SQL查询语句的一部分执行，导致改变了程序原始的SQL查询逻辑，额外地执行了攻击者构造的SQL查询语句。

**漏洞级别：高**

**缺陷代码位置：**

JavaVulnerableLab/src/main/java/org/cysecurity/cspf/jvl/controller/LoginValidator.java，第52行

**缺陷代码分析：**

在LoginValidator.java文件中，存在SQL语句拼接，且拼接的参数user、pass来自于用户请求request。

|  |
| --- |
| protected void processRequest(HttpServletRequest request, HttpServletResponse response)  throws ServletException, IOException {      **String user=request.getParameter("username").trim();**  **String pass=request.getParameter("password").trim();**  try  {  Connection con=new DBConnect().connect(getServletContext().getRealPath("/WEB-INF/config.properties"));  if(con!=null && !con.isClosed())  {  ResultSet rs=null;  Statement stmt = con.createStatement();  **rs=stmt.executeQuery("select \* from users where username='"+user+"' and password='"+pass+"'");**  if(rs != null && rs.next()){  HttpSession session=request.getSession();  session.setAttribute("isLoggedIn", "1");  session.setAttribute("userid", rs.getString("id"));  session.setAttribute("user", rs.getString("username"));  session.setAttribute("avatar", rs.getString("avatar"));  Cookie privilege=new Cookie("privilege","user");  response.addCookie(privilege);  if(request.getParameter("RememberMe")!=null)  {  Cookie username=new Cookie("username",user);  Cookie password=new Cookie("password",pass);  response.addCookie(username);  response.addCookie(password);  }  response.sendRedirect(response.encodeURL("ForwardMe?location=/index.jsp"));  }  else  {  response.sendRedirect("ForwardMe?location=/login.jsp&err=Invalid Username or Password");  }    }  }  catch(Exception ex)  {  response.sendRedirect("login.jsp?err=something went wrong");  }    } |

**修复建议：**

1. 对输入的参数进行过滤或转义；
2. 采用预编译的方式进行查询。

# 审计结果及修复建议

本次对【XX系统】的源代码安全审计，已发现安全问题的修复建议如下：

|  |  |  |
| --- | --- | --- |
| **发现问题** | **漏洞级别** | **修复建议** |
| 报表统计SQL注入 | **高** | 1、使用参数化查询构造后台查询语句；  2、对前端录入的参数进行特殊字符过滤处理； |
| 登录暴力破解 | **中** | 登录过程中，只提交表单参数userName和password，并未有其他安全设计限制尝试登录，通过穷举用户名和密码的方式，可得到正确的登录密码。 |
| 会话欺骗攻击 | **低** | 修改权限分配机制，建议采用后台验证。避免前端内容可被篡改风险。 |
| … | … | … |