测试总结报告

一、 测试概述

项目	内容
测试对象	WordPress 6.4 内容管理系统（文章管理、评论模块）
测试时间	2025年11月
测试环境	WordPress 6.4 + Docker + MySQL + Edge浏览器
测试人员	王慧琴
测试目的	验证CMS基础功能完整性，评估安全防护能力

二、 测试范围

已测内容

- 文章管理：发布/编辑/删除、标题边界值、草稿保存
- 评论功能：正常提交、XSS安全测试、边界长度
- 用户权限：角色访问控制

未测内容

- 性能测试（无压力测试工具）
- 插件兼容性测试
- 多浏览器兼容性测试

三、 测试方法与工具

类型	方法	工具
功能测试	黑盒测试、等价类划分、边界值分析	Excel
安全测试	手工注入XSS payload	Charles抓包、浏览器开发者工具
缺陷管理	按模板记录复现步骤与证据	Excel、截图工具

四、 测试用例执行统计

模块	用例数	通过	未通过	失败率
用户登录	7	7	0	0

文章发布	12	12	0	0
文件上传	7	7	0	0
用户权限	7	7	0	0
评论功能	6	5	1	16.67%
搜索功能	2	2	0	0

五、 缺陷详情

高危缺陷（1个）

项目	内容
标题	评论模块存储型XSS漏洞
严重程度	高危
现象	评论框输入 `<script>alert('1')</script>`，提交后页面弹出警告框，且每次刷新都触发
根因分析	后端渲染评论内容时未进行HTML实体编码，恶意脚本原样插入DOM
证据	Elements面板显示script标签作为DOM节点插入（未转义）
修复方案	添加 `wp_kses_post` 过滤器对评论内容进行白名单过滤
验证结果	修复后payload显示为纯文本，不再执行脚本

六、 遗留风险

- 未配置HTTPS，Cookie明文传输（生产环境需修复）
- 未测试高并发场景下的数据库死锁

七、 个人收获与改进

维度	具体内容
技术能力	掌握Charles抓包分析、浏览器开发者工具定位DOM注入、XSS漏洞验证流程
测试思维	认识到安全测试需专项构造恶意payload，不能依赖功能用例覆盖
流程规范	建立了"发现→记录→跟踪→验证"的缺陷管理习惯
待改进	前期环境搭建耗时过长，下次应先写Docker-compose脚本；需补充性能测试技能