### 实训22 对DVWA系统进行手工SQL注入实训

**实训目的：**

通过本实训，掌握手工SQL注入的基本方法。

#### 任务1测试和分析页面的功能

1．点击“SQL Injection”，这里有一个输入框。根据页面的提示，可以在输入框中输入用户的ID。我们输入“1”之后，点击“Submit”按钮。发现它返回了关于这个User的信息，如图11-10所示。它返回三行数据，第一行是我们输入的内容。第二行是用户名，第三行是用户别名。同时，在浏览器的地址栏，我们发现URL这里有个id=1，这是不是就是我们输入的User ID呢？我们在输入框再输入“2”，发现URL变成了id=2 。我们可以得出这里传进去的id的值是我们可以控制的。我们在输入框中输入的内容，会通过id传进到服务器。

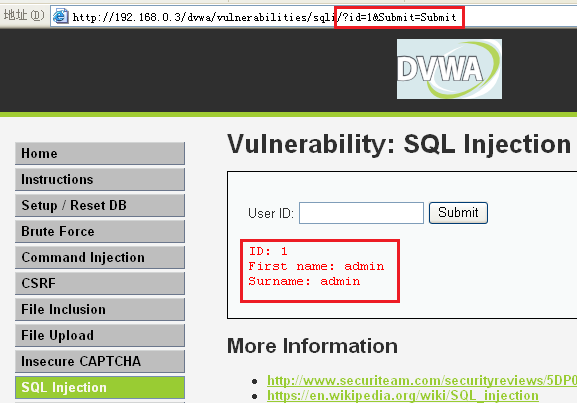


图11-10 返回User信息

#### 任务2对参数进行测试

1．对id这个参数进行测试，查看一下它是否存在sql注入漏洞。在输入框里面输入“1'”。发现这里报错了，说sql语句出现了语法错误，如图11-11所示。

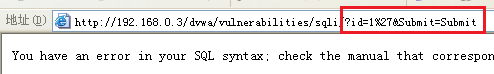


图11-11 SQL语法错误

可以进行这样一个猜测：这个id是被两个“'”包住的，查询语句可能是这样的：

select firstname,surname from users where id = '1';

当我们在1之后加一个引号，那么查询语句会变成这样：

select firstname,surname from users where id = '1'';

可以看到单引号数目不平衡，最后一个引号没被闭合。

2．多种办法可以消除引号没有被闭合的问题，下面就简单介绍一下。

方法一：可以在原来的基础上再继续输入一个引号，也就是“1''”。这时我们的查询语句为：

select firstname,surname from users where id = '1''';

方法二：使用“#”符号来注释后面的单引号，这时查询语句将会变成：

select firstname,surname from users where id = '1'#';

方法三：使用“-- ”。这里注意了“-- ”后面有一个空格。在URL当中，我们可以使用“+”来代替“--”后面的空格。这时查询语句将会变成：

select firstname,surname from users where id = '1'--+';

上面显示出来的结果和输入1时一样。到这里我们就可以确定：

（1）漏洞的参数是“id”。

（2）漏洞的类型是字符型。

#### 任务3构造payload

确认漏洞之后，就可以构造payload。payload就是一段恶意代码，以便能够获得数据库里面的数据。

1.分析字段数

分析字段数，有两种方法。

方法一：用order by 语句

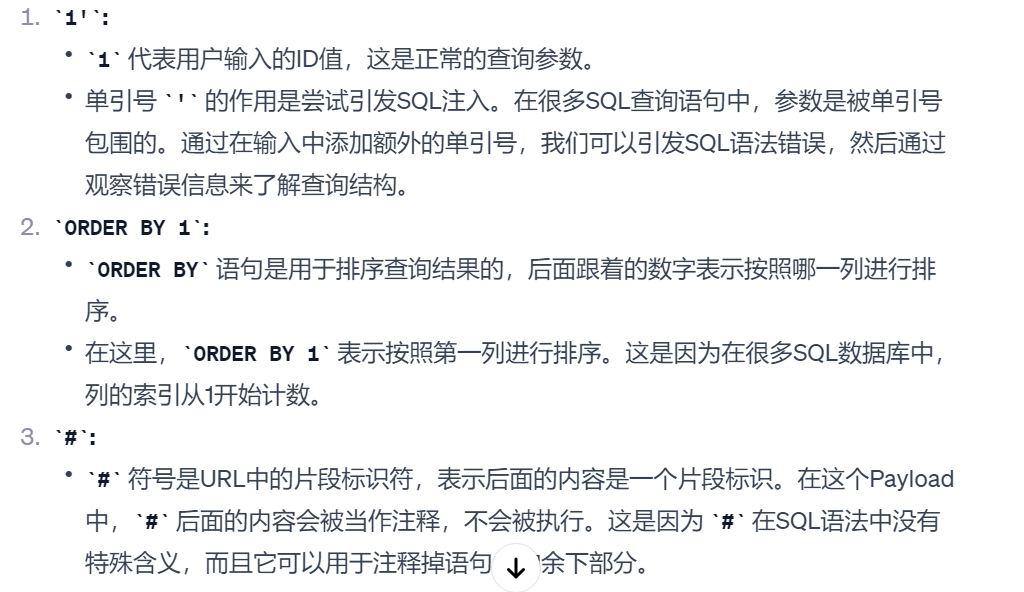
分析字段数的原因是，我们后面需要用union select语句来获得敏感数据。根据order by知识如果后面跟着的数字超出了字段数时，就会报错，通过这个我们可以确定字段数。构造的payload如下：

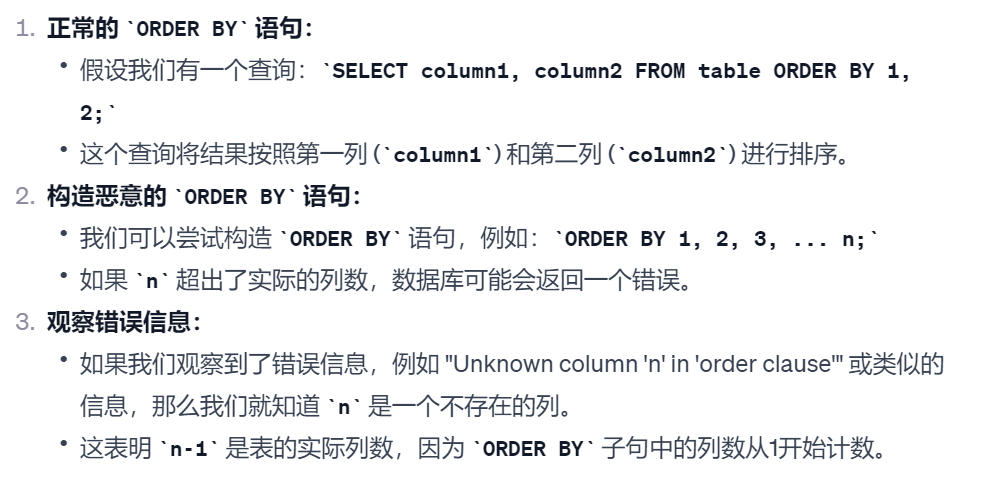
1' order by 1#

1' order by 2#

1' order by 3#

当输入到3的时候，发现它报错了，也就是说字段数为2。





方法二：用union select来猜测字段数。

因为当字段数不对应的时候，它是会发生报错的，构造以下查询语句：

1' union select 1#

1' union select 1,2#

1' union select 1,2,3#

* **UNION SELECT** 用于合并两个查询的结果集。
* 在 **UNION SELECT** 语句中，**1,2** 表示在新的查询中选择两个常量值作为新的列

可以发现，只有输入“1'union select 1,2 #”的时候没有报错，也就是说字段数为2。同时，我们也注意到，返回的内容中多了三条数据，其实这就是union select出来的数据，如图11-12所示。

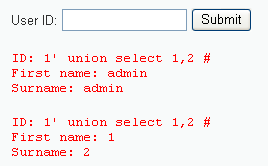


图11-12 联合查询返回结果

2. 获取信息

字段数为2，说明数据列有两列。可以通过union select语句查出两个数据。接下来，我们来获取所需要的数据库里面的信息。

（1）获取当前数据库名和当前用户名

构造数据库查询语句如下所示：

1' union select database(),user()#

database()将会返回当前网站所使用的数据库名字，user()将会返回进行当前查询的用户名。可以看到当前使用的数据库为：dvwa，当前的用户名：root@localhost，如图11-13所示。

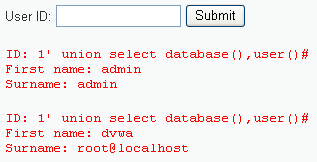
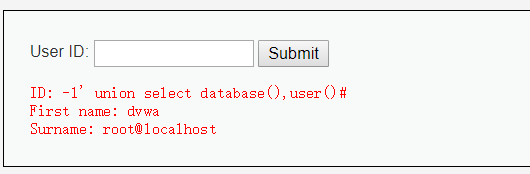


图11-13 返回数据库名和当前用户名

有时候，后面的select语句会限制输出的行数，我们可以让原数据库查询无效，也就是输入无效的id，使得原数据库查询不返回结果。输入如下：

-1' union select database(),user()#

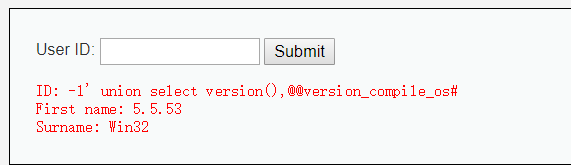
这样就只会返回我们需要的数据了。



version() 获取当前数据库版，@@version\_compile\_os获取当前操作系统。输入如下：

-1' union select version(),@@version\_compile\_os#

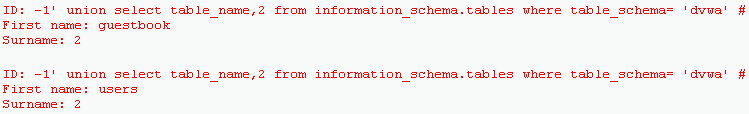
（2）获取当前的用户表

根据上面的信息，我们知道当前数据库名为dvwa，接下来我们要获得表名和字段名以及内容。

当你有不懂的字会怎么办呢？对了，查字典。那么Mysql有没有类似于字典的东西呢？答案是肯定的，就是information\_schema，这是一个包含了mysql数据库所有信息的“字典”，本质上是一个database，存放着其他各个数据的信息。

在information\_schema里，有tables表和columns表对我们来说非常有用。tables这个表存放的是关于数据库中所有表的信息，里面有个字段叫table\_name，还有个字段叫做table\_schema。其中table\_name是表名，table\_schema表示的是这个表所在的数据库。对于columns，它有column\_name，table\_schema，table\_name。回想一下，我们拥有的信息是数据库名。也就是说我们可以构造这样的payload来从数据库里获取一些信息。构造的查询语句如下：

-1' union select table\_name,2 from information\_schema.tables where table\_schema= 'dvwa' #

图11-14 返回表名

爆出来两个表，如图11-14所示，我们对users表更感兴趣。不是说还有一个columns表么？所以我们还需要table\_name以及table\_schema来查column\_name。这次我们构造的payload如下：

-1' union select column\_name,2 from information\_schema.columns where table\_schema= 'dvwa' and table\_name= 'users' #

这里简单说一下，倘若不指定数据库名为dvwa'，若是其他数据里面也存在users表的话，则会出现很多混淆的数据。当然，在这里直接使用下面的语句也是可以成功的。

在这些返回的数据中，我们看到了user，password。这是我们最希望看得的字段，我们再次修改payload：

-1' union select user,password from users #

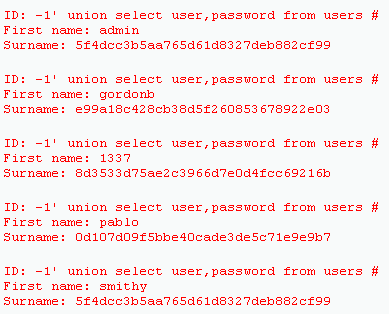


图11-15 返回用户名和密码

我们爆出所有的用户名和密码值，如图11-15所示。这密码好像有点奇怪，数一数是32位。应该是经过md5加密的。我们需要找一些破解md5值的网站来进行破解可以了。我们对“gordonb”用户来进行破解，md5密文为：e99a18c428cb38d5f260853678922e03。对密文进行解密得到密码是“abc123”。