PHP伪协议

lorexxar / 2016-11-08 12:30:00 / 浏览数 4265 安全技术 漏洞分析 顶(0) 踩(0)

最近php伪协议的各种神奇妙用好像突然又常常提到了，php中支持的伪协议有下面这么多

```
file:// — ■■■■■■■■■
http:// — ■■ HTTP(s) ■■
ftp:// — ■■ FTP(s) URLs
php:// — ■■■■■■■/■■■■I/O streams■
zlib:// — ■■■■
data:// — ■■■RFC 2397■
glob:// — ■■■■■■■■■■■■
phar:// — PHP ■■
ssh2:// — Secure Shell 2
rar:// — RAR
ogg:// — ■■■■
expect:// — ■■■■■■■■
```

今天着重研究php://

首先先把官方文档贴上来
http://php.net/manual/zh/wrappers.php.php

有两个比较重要的配置在php.ini中，allow_url_fopen
和allow_url_include会影响到fopen等等和include等等函数对于伪协议的支持，而allow_url_include依赖allow_url_fopen，所以allow_url_fopen不开启的话，allow_url_i

php://是用来访问各个输入、输出流的，除了`php://stdin, php://stdout ■ php://stderr`

php://input

php://input代表可以访问请求的原始数据，简单来说POST请求的情况下，php://input可以获取到post的数据。

比较特殊的一点，enctype="multipart/form-data" 的时候 php://input 是无效的。

php://output

php://output 是一个只写的数据流， 允许你以 print 和 echo 一样的方式 写入到输出缓冲区。

php://filter

这篇文章的关键在于讨论php://filter,事实上，这也是我们常常使用的一个伪协议，在任意文件读取，甚至getshell的时候都有利用的机会。

`php://filter` 是一种元封装器， 设计用于数据流打开时的筛选过滤应用。 这对于一体式（all-in-one）的文件函数非常有用，类似 readfile()、 file() 和 file_get_contents()， 在数据流内容读取之前没有机会应用其他过滤器。

事实上，在include函数的使用上，经常会造成任意文件读取漏洞，而file_get_contents()和file_put_contents()这样函数下，常常会构成getshell等更严重的漏洞。

php://filter 目标使用以下的参数作为它路径的一部分。 复合过滤链能够在一个路径上指定。详细使用这些参数可以参考具体范例。

文档里是这么写的

```
■■ ■■
resource=<■■■■■■■■>      ■■■■■■■■■■■■■■■■■■■■■■■■■■
read=<■■■■■■■■>          ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■|■■■■■
write=<■■■■■■■■>     ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■|■■■■■
<■■■■■■■■■■>         ■■■■■■ read= ■ write= ■■■■ ■■■■■■■■■■■■■■■■■■■■
```

我们举一个例子，这是平时我们用来任意文件读取的payload

```
php://filter/read=convert.base64-encode/resource=upload.php
```

这里读的过滤器为convert.base64-encode，就和字面上的意思一样，把输入流base64-encode。
resource=upload.php，代表读取upload.php的内容

下面仔细研究下关于过滤器的问题

过滤器

先贴文档，不因为自己的翻译小问题接锅（・ω・）ノ
http://php.net/manual/zh/filters.php

转换过滤器

http://php.net/manual/zh/filters.convert.php

convert.* 过滤器是php5.0.0以后添加的。

base64

convert.base64-encode和 convert.base64-decode使用这两个过滤器等同于分别用 base64_encode()和 base64_decode()函数处理所有的流数据。
convert.base64-encode支持以一个关联数组给出的参数。如果给出了 line-length，base64 输出将被用 line-length个字符为 长度而截成块。如果给出了 line-break-chars，每块将被用给出的字符隔开。这些参数的效果和用 base64_encode()再加上 chunk_split()相同。

当然，这里的过滤器不止运用于php://filter，所以文档中给出的例子是这样的

```
<?php
$fp = fopen('php://output', 'w');
stream_filter_append($fp, 'convert.base64-encode');
fwrite($fp, "This is a test.\n");
fclose($fp);
/* Outputs:  VGhpcyBpcyBhIHRlc3QuCg==  */

$param = array('line-length' => 8, 'line-break-chars' => "\r\n");
$fp = fopen('php://output', 'w');
stream_filter_append($fp, 'convert.base64-encode', STREAM_FILTER_WRITE, $param);
fwrite($fp, "This is a test.\n");
fclose($fp);
/* Outputs:  VGhpcyBp
         :  cyBhIHRl
         :  c3QuCg==  */

$fp = fopen('php://output', 'w');
stream_filter_append($fp, 'convert.base64-decode');
fwrite($fp, "VGhpcyBpcyBhIHRlc3QuCg==");
fclose($fp);
/* Outputs:  This is a test.  */
?>
```

quoted-printable

convert.quoted-printable-encode和 convert.quoted-printable-decode使用此过滤器的 decode 版本等同于用 quoted_printable_decode()函数处理所有的流数据。没有和 convert.quoted-printable-encode相对应的函数。
convert.quoted-printable-encode支持以一个关联数组给出的参数。除了支持和 convert.base64-encode一样的附加参数外，convert.quoted-printable-encode还支持布尔参数 binary和 force-encode-first。 convert.base64-decode只支持 line-break-chars参数作为从编码载荷中剥离的类型提示。

关于quoted_printable_decode()在php.net上的解释是将 quoted-printable 字符串转换为 8-bit 字符串，原谅我没怎么看懂

字符串过滤器

string.*是用来处理各个字符串的，比较像python的string模块

string.rot13

rot13，很好理解

```
<?php
$fp = fopen('php://output', 'w');
stream_filter_append($fp, 'string.rot13');
fwrite($fp, "This is a test.\n");
/* Outputs:  Guvf vf n grfg.   */
?>
```

toupper

变大写，也同样很好理解

```
<?php
$fp = fopen('php://output', 'w');
stream_filter_append($fp, 'string.toupper');
fwrite($fp, "This is a test.\n");
/* Outputs:  THIS IS A TEST.   */
?>
```

tolower

这回是小写

```
<?php
$fp = fopen('php://output', 'w');
stream_filter_append($fp, 'string.tolower');
fwrite($fp, "This is a test.\n");
/* Outputs:  this is a test.   */
?>
```

string.strip_tags

string.strip_tags（自 PHP 5.0.0 起）使用此过滤器等同于用 strip_tags()函数处理所有的流数据。可以用两种格式接收参数：一种是和 strip_tags()函数第二个参数相似的一个包含有标记列表的字符串，一种是一个包含有标记名的数组。

strip_tags()返回给定的字符串 str 去除空字符、HTML 和 PHP 标记后的结果。

```
<?php
$fp = fopen('php://output', 'w');
stream_filter_append($fp, 'string.strip_tags', STREAM_FILTER_WRITE, "<b><i><u>");
fwrite($fp, "<b>bolded text</b> enlarged to a <h1>level 1 heading</h1>\n");
fclose($fp);
/* Outputs:  <b>bolded text</b> enlarged to a level 1 heading   */

$fp = fopen('php://output', 'w');
stream_filter_append($fp, 'string.strip_tags', STREAM_FILTER_WRITE, array('b','i','u'));
fwrite($fp, "<b>bolded text</b> enlarged to a <h1>level 1 heading</h1>\n");
fclose($fp);
/* Outputs:  <b>bolded text</b> enlarged to a level 1 heading   */
?>
```

压缩过滤器

zlib.* 压缩过滤器自 PHP 版本 5.1.0起可用，在激活 zlib的前提下。也可以通过安装来自 » PECL的 » zlib_filter包作为一个后门在 5.0.x版中使用。此过滤器在 PHP 4 中 不可用。

zlib.deflate和 zlib.inflate是主要的两个用法

```
<?php
$params = array('level' => 6, 'window' => 15, 'memory' => 9);

$original_text = "This is a test.\nThis is only a test.\nThis is not an important string.\n";
echo "The original text is " . strlen($original_text) . " characters long.\n";

$fp = fopen('test.deflated', 'w');
stream_filter_append($fp, 'zlib.deflate', STREAM_FILTER_WRITE, $params);
fwrite($fp, $original_text);
fclose($fp);

echo "The compressed file is " . filesize('test.deflated') . " bytes long.\n";
echo "The original text was:\n";
/* Use readfile and zlib.inflate to decompress on the fly */
readfile('php://filter/zlib.inflate/resource=test.deflated');

/* Generates output:

The original text is 70 characters long.
The compressed file is 56 bytes long.
The original text was:
This is a test.
This is only a test.
This is not an important string.
```

```
*/
?>
```

加密过滤器

mcrypt.和 mdecrypt.使用 libmcrypt 提供了对称的加密和解密。

格式为 mcrypt.ciphername，其中 ciphername是密码的名字，将被传递给 mcrypt_module_open()。有以下五个过滤器参数可用:

参数 是否必须 默认值 取值举例
mode 可选 cbc cbc, cfb, ecb, nofb, ofb, stream
algorithms_dir 可选 ini_get('mcrypt.algorithms_dir') a lgorithms 模块的目录
modes_dir 可选 ini_get('mcrypt.modes_dir') modes 模块的目录
iv 必须 N/A 典型为 8，16 或 32 字节的二进制数据。根据密码而定
key 必须 N/A 典型为 8，16 或 32 字节的二进制数据。根据密码而定

细节自己研究文档吧
http://php.net/manual/zh/filters.encryption.php

上一篇：MySQL在渗透测试中的应用 下一篇：【原创】中间件漏洞检测(Middl...
1. 1 条回复


master 2016-11-09 07:01:18

好了，别在我师傅 面前说我厉害，你再这样下去，我的几个师傅，估计要把我赶出去了。

0 回复Ta
