

# 关于强网拟态题目 EasyFilter

之前有人问过这道题目，大概的代码是这样：

```
<?php
    ini_set("open_basedir","./");
    if(!isset($_GET['action'])){
        highlight_file(__FILE__);
        die();
    }
    if($_GET['action'] == 'w'){
        @mkdir("./files/");
        $content = $_GET['c'];
        $file = bin2hex(random_bytes(5));
        file_put_contents("./files/".$file,base64_encode($content));
        echo "./files/".$file;
    }elseif($_GET['action'] == 'r'){
        $r = $_GET['r'];
        $file = "./files/".$r;
        include("php://filter/resource=$file");
    }
}
```

我当时简单看了下代码，甚至翻了下PHP底层的源码，没找到解决方法，后来忙别的事就给忘了。

今天看到星球有同学发了答案，我又翻了下PHP底层的源码，原来是我之前弄错了。简单写一篇小文字记录一下。

处理php://filter的逻辑是在这个文件中：ext/standard/php\_fopen\_wrapper.c

php\_stream\_url\_wrap\_php 函数：

```
1  php_stream * php_stream_url_wrap_php/php_stream_wrapper *wrapper, const char
   *path, const char *mode, int options,
2                                     zend_string **opened_path,
   php_stream_context *context STREAMS_DC) /* {{{ */
3  {
4      int fd = -1;
5      int mode_rw = 0;
6      php_stream * stream = NULL;
7      char *p, *token = NULL, *pathdup;
8      zend_long max_memory;
9      FILE *file = NULL;
10 #ifdef PHP_WIN32
11     int pipe_requested = 0;
12 #endif
13
14     if (!strncasecmp(path, "php://", 6)) {
15         path += 6;
16     }
17 }
```

```

18     if (!strcasecmp(path, "temp", 4)) {...}
19     if (!strcasecmp(path, "memory")) {...}
20     if (!strcasecmp(path, "output")) {...}
21     if (!strcasecmp(path, "input")) {...}
22     if (!strcasecmp(path, "stdin")) {
23         ...
24     } else if (!strcasecmp(path, "stdout")) {
25         ...
26     } else if (!strcasecmp(path, "stderr")) {
27         ...
28     } else if (!strcasecmp(path, "fd/", 3)) {
29         ...
30     } else if (!strcasecmp(path, "filter/", 7)) {
31         /* Save time/memory when chain isn't specified */
32         if (strchr(mode, 'r') || strchr(mode, '+')) {
33             mode_rw |= PHP_STREAM_FILTER_READ;
34         }
35         if (strchr(mode, 'w') || strchr(mode, '+') || strchr(mode, 'a')) {
36             mode_rw |= PHP_STREAM_FILTER_WRITE;
37         }
38         pathdup = estrndup(path + 6, strlen(path + 6));
39         p = strstr(pathdup, "/resource=");
40         if (!p) {
41             zend_throw_error(NULL, "No URL resource specified");
42             efree(pathdup);
43             return NULL;
44         }
45
46         if (!(stream = php_stream_open_wrapper(p + 10, mode, options,
opened_path))) {
47             efree(pathdup);
48             return NULL;
49         }
50
51         *p = '\0';
52
53         p = php_strtok_r(pathdup + 1, "/", &token);
54         while (p) {
55             if (!strcasecmp(p, "read=", 5)) {
56                 php_stream_apply_filter_list(stream, p + 5, 1, 0);
57             } else if (!strcasecmp(p, "write=", 6)) {
58                 php_stream_apply_filter_list(stream, p + 6, 0, 1);
59             } else {
60                 php_stream_apply_filter_list(stream, p, mode_rw &
PHP_STREAM_FILTER_READ, mode_rw & PHP_STREAM_FILTER_WRITE);
61             }
62             p = php_strtok_r(NULL, "/", &token);
63         }
64         efree(pathdup);
65
66         if (EG(exception)) {
67             php_stream_close(stream);
68             return NULL;
69         }
70
71         return stream;
72     } else {
73         return NULL;

```

```
74 | }
75 | ...
76 | }
```

前面那些if判断就不说了，都是用来处理 `php://` 中其他的path。

我们看到对 `filter/` 的处理。

第一个关键点在第38到44行，在 `pathdup` 变量中找到 `resource=` 这个关键字，如果没有这个关键字，直接报错退出。

第二个关键点在第46行，将 `resource=` 关键字后面的字符串传给 `php_stream_open_wrapper`，用于读取下一个流。这实际上算一个递归的过程。

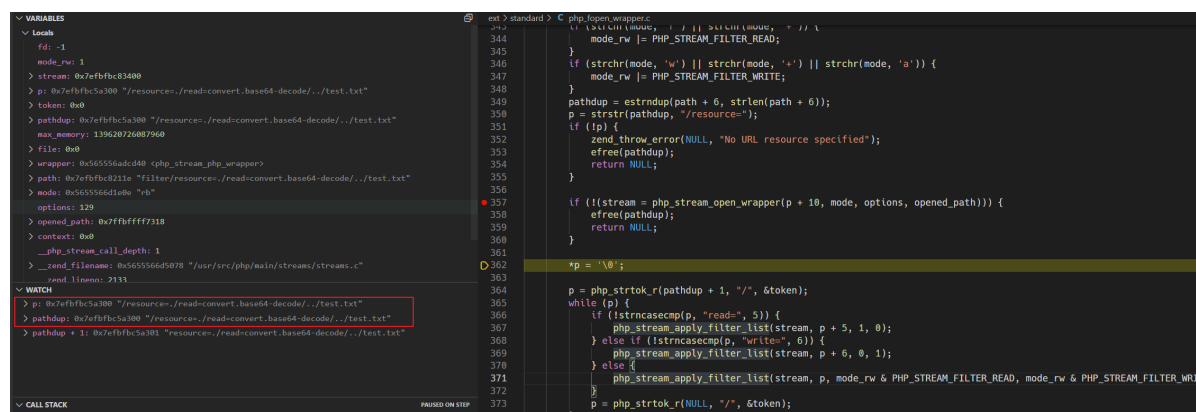
第三个关键点是第51行，`*p = '\0'`。

第四个关键点是第53到63行，用 `/` 分割 `pathdup` 变量，并遍历这些分割后的部分，判断是否是 `read=`、`write=` 等，对第二步中读取到的文件内容进行filter处理。

这几个关键点里，第三个点当时我弄错了，也怪我自己没有动态调试。我当时以为这个 `p` 指向的是 `resource=` 后面的那个字符，所以这里将 `p` 设置成 `\0` 以后相当于 `pathdup` 这个字符串就被截取了。

所以我认为第四步是从 `php://filter/` 后开始查找，查找到 `resource=` 为止（因为等号后面就是 `\0`）。而在本题里，这部分用户并不可控。

实际上今天动态调试我发现，在第三步的时候，`p` 指向的地址实际上和 `pathdup` 相同：



所以将 `*p` 设置为 `\0`，堵上的是 `php://filter/` 这段字符串的最后一个字符。

而后面查找 `read=` 和 `write=` 的操作是从 `filter/` 后的第一个字符开始的，那么自然可以查找到用户可控的地方。

所以，我们最后通过 `php://filter/resource=./read=convert.base64-decode/./test.txt` 这样输入，控制了 `read=` 操作的值是 `convert.base64-decode`，而这一部分又作为路径，成功读取到文件。

## 参考

本文题目来自于这个帖子：<https://t.zsxq.com/AYbmQfl>

本文里涉及的PHP调试方法，可以参考这2篇帖子：<https://t.zsxq.com/aMZRNnz>、<https://t.zsxq.com/7iYVRZr>

