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

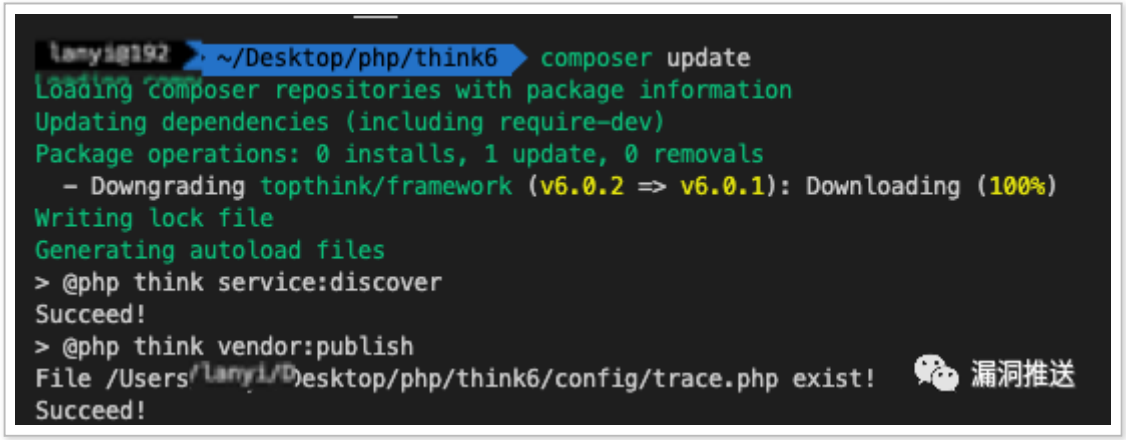
编辑程序根目录下的 composer.json 文件, 改成如红框所示

然后执行

-

```
compose update
```

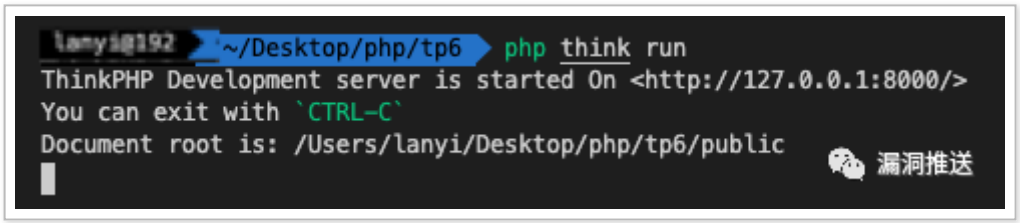
如图所示, 即是降级成功



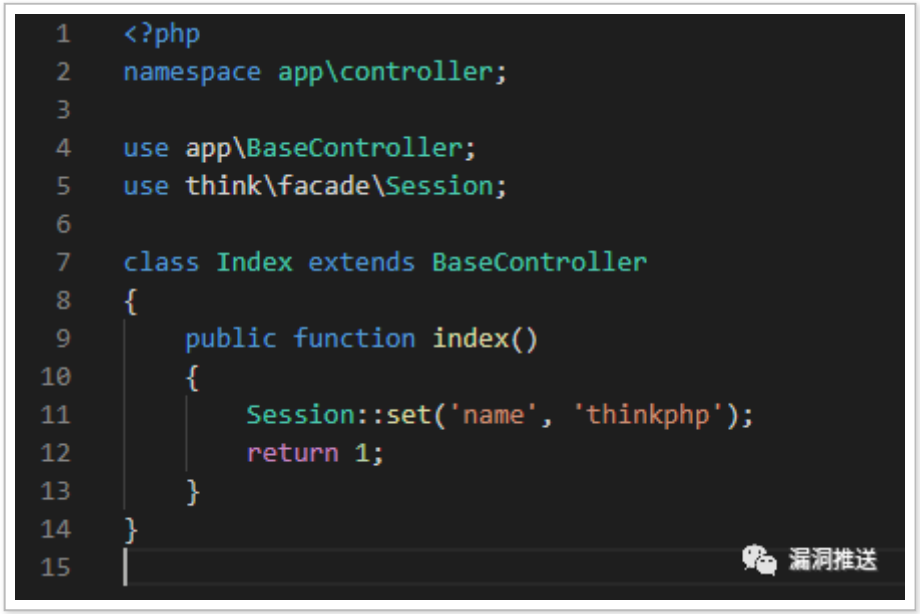
然后是非常重要的一步, 很多复现的文章都没有写, 就因为这个浪费了我一两个小时的时间.

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

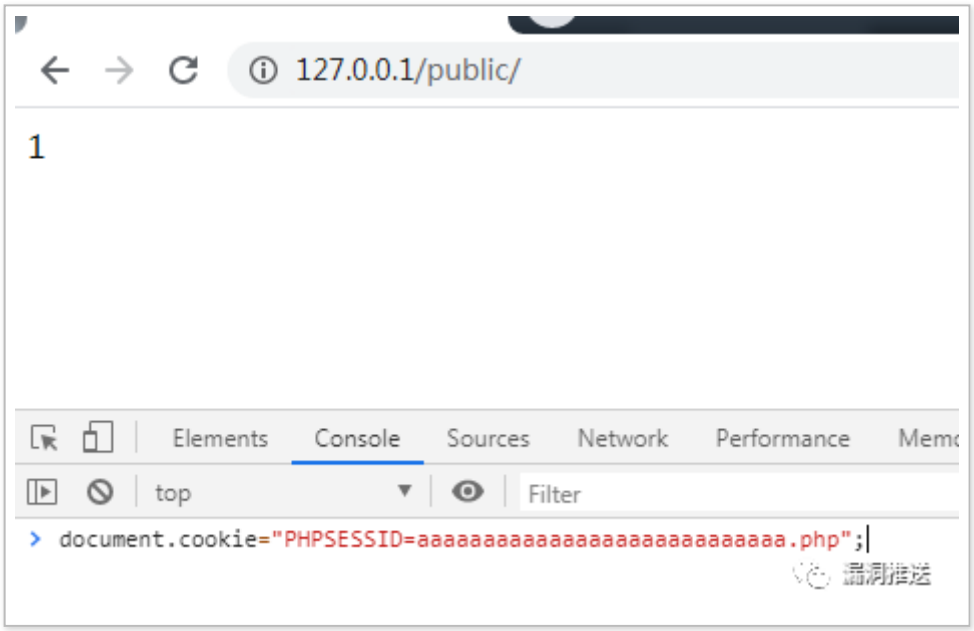
终端运行 php think run 启动调试环境



app\controller\Index.php 中添加测试代码



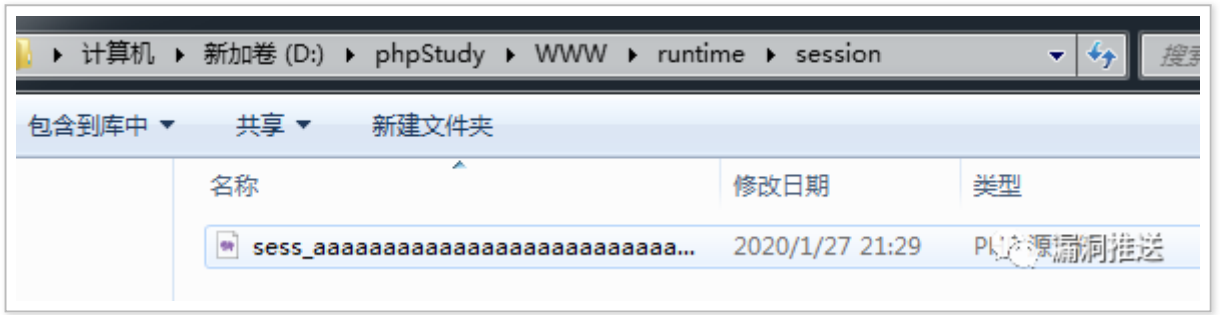
在浏览器 f12 console 里面执行 poc, 然后刷新页面



-

```
document.cookie="PHPSESSID=aaaaaaaaaaaaaaaaaaaaaaaaaaa.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

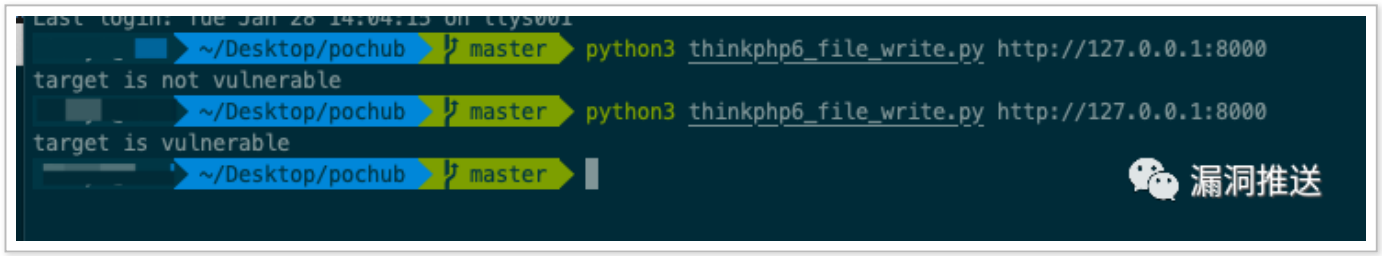
-

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

Python 漏洞检测 POC

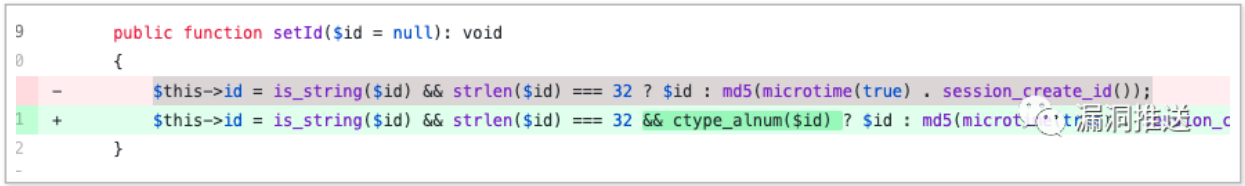
-

```
https://github.com/lanyi1998/pochub/blob/master/thinkphp6_file_write.py
```



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

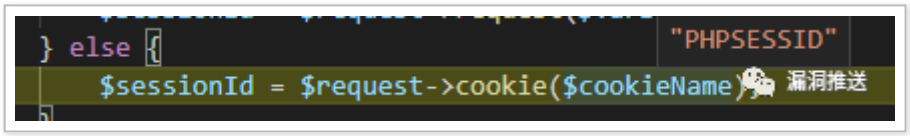


-

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

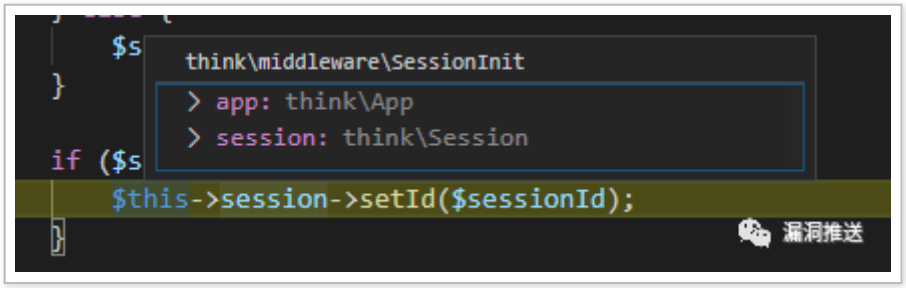
找到在 vendor/topthink/framework/src/think/middleware/SessionInit.php 中

第一步:



从 cookie 中取出 phpsessid 的值, 赋值给 sessionId

第二步:



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

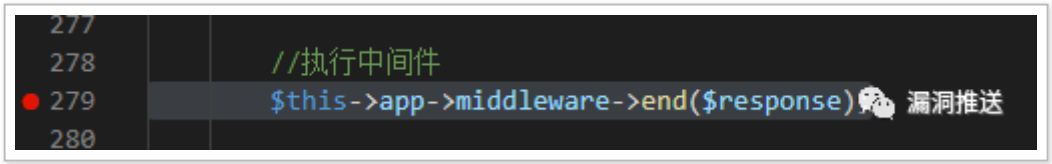
第三步:

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

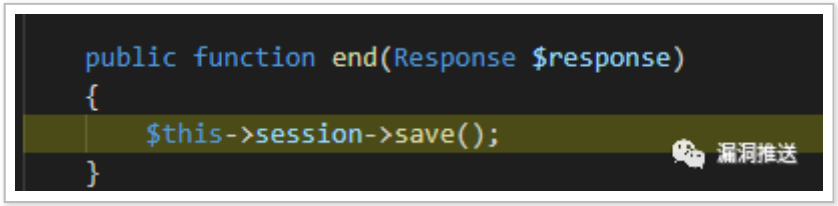
看 thinkphp 文档

tp6 会在请求结束的时候进行 session 写入

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



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



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

```php
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;
}
```


跟进到红框所示的 write 方法, 会执行
vendor\topthink\framework\src\think\session\driver\File.php 中

```php
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    }
```

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

```php
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

```php
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_cre
}
```


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

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()
```