

代码审计：PbootCMS2.07内核处理缺陷导致的一个前台任意文件包含漏洞分析

P3rh4ps / 2020-05-19 09:30:38 / 浏览数 8286

挖出来之后看了下官网发现不到半个月之前更新了最新版，下下来之后发现这洞修了..我吐了
干脆直接发出来 分享下思路吧

0x00漏洞分析

漏洞发生在PbootCMS内核的模板解析函数中
为了方便看直接上一个完整的Parser代码吧

```

public function parser($file)
{
    // 设置主题
    $theme = isset($this->vars['theme']) ? $this->vars['theme'] : 'default';
    //file形式:xxxxx/../.././可以双写穿越
    $theme = preg_replace('/\.\.(\V|\\)/', '', $theme); // 过滤掉相对路径
    $file = preg_replace('/\.\.(\V|\\)/', '', $file); // 过滤掉相对路径

    if (strpos($file, '/') === 0) { // 绝对路径模板
        $tpl_file = ROOT_PATH . $file;
    } elseif (! $pos = strpos($file, '@')) { // 跨模块调用
        $path = APP_PATH . '/' . substr($file, 0, $pos) . '/view/' . $theme;
        define('APP_THEME_DIR', str_replace(DOC_PATH, '', $path));
        if (! is_dir($path)) { // 检查主题是否存在
            error('模板主题目录不存在! 主题路径: ' . $path);
        } else {
            $this->tplPath = $path;
        }
        $tpl_file = $path . '/' . substr($file, $pos + 1);
    } else {
        // 定义当前应用主题目录
        define('APP_THEME_DIR', str_replace(DOC_PATH, '', APP_VIEW_PATH) . '/' . $theme);
        if (! is_dir($this->tplPath . '/' . $theme)) { // 检查主题是否存在
            error('模板主题目录不存在! 主题路径: ' . APP_THEME_DIR);
        }
        $tpl_file = $this->tplPath . '/' . $file; // 模板文件
    }
    $note = Config::get('tpl_html_dir') ? '<br>同时检测到您系统中启用了模板子目录' . Config::get('tpl_html_dir') . ', 请核对是否是此原因导致! ': '';
    file_exists($tpl_file) ? error('模板文件' . APP_THEME_DIR . '/' . $file . '不存在! ' . $note);
    $tpl_c_file = $this->tplPath . '/' . md5($tpl_file) . '.php'; // 编译文件

    // 当编译文件不存在, 或者模板文件修改过, 则重新生成编译文件
    if (! file_exists($tpl_c_file) || filemtime($tpl_c_file) < filemtime($tpl_file) || ! Config::get('tpl_parser_cache')) {
        $content = Parser::compile($this->tplPath, $tpl_file); // 解析模板
        file_put_contents($tpl_c_file, $content) ? error('编译文件' . $tpl_c_file . '生成出错! 请检查目录是否有可写权限! '); // 写入编译文件
        $compile = true;
    }
    //tplPath:PbootCMS/template
    ob_start(); // 开启缓冲区,引入编译文件
    $rs = include $tpl_c_file;
    if (! isset($compile)) {
        foreach ($rs as $value) { // 检查包含文件是否更新,其中一个包含文件不存在或修改则重新解析模板
            if (! file_exists($value) || filemtime($tpl_c_file) < filemtime($value) || ! Config::get('tpl_parser_cache')) {
                $content = Parser::compile($this->tplPath, $tpl_file); // 解析模板
                file_put_contents($tpl_c_file, $content) ? error('编译文件' . $tpl_c_file . '生成出错! 请检查目录是否有可写权限! '); // 写入编译文件
            }
            ob_clean();
            include $tpl_c_file;
            break;
        }
    }
    $content = ob_get_contents();
    ob_end_clean();
    return $content;
}

```

简单讲一下重点

```
// 设置主题
$theme = isset($this->vars['theme']) ? $this->vars['theme'] : 'default';
//file形式:xxxxx/../../../可以双写穿越
$theme = preg_replace(pattern: '/\.\.(\w|\\|\/)/', replacement: '', $theme); // 过滤掉相对路
$file = preg_replace(pattern: '/\.\.(\w|\\|\/)/', replacement: '', $file); // 过滤掉相对路

if (strpos($file, needle: '/') === 0) { // 绝对路径模板
    $tpl_file = ROOT_PATH . $file;
}
```

这里对传入路径的过滤并不严格，可以双写绕过
再往下跟一下

```
// 当编译文件不存在，或者模板文件修改过，则重新生成编译文件
if (! file_exists($tpl_c_file) || filemtime($tpl_c_file) < filemtime($tpl_file) || ! Config::get(item: 'tpl_parser_cache')) {
    $content = Parser::compile($this->tplPath, $tpl_file); // 解析模板
    file_put_contents($tpl_c_file, $content) ?: error(string: '编译文件' . $tpl_c_file . '生成出错! 请检查目录是否有可写权限! '); // 写入
    $compile = true;
}

//tplPath:PhootCMS/template
ob_start(); // 开启缓冲区,引入编译文件
$rs = include $tpl_c_file;
```

当模板文件不在缓存中的时候，会读取\$tpl_file中的内容，然后写入缓存文件中并且包含。
也就是说，当parser函数的参数可以被控制的时候，就会造成一个任意文件包含。
所以，要找一个可控参数的parser调用
经过简单寻找，就可以发现前台控制器TagController中的index方法，完美符合我们的要求
上代码：

```
public function index()
{
    // 在非兼容模式接受地址第二参数值
    if (defined('RVAR')) {
        $_GET['tag'] = RVAR;
    }

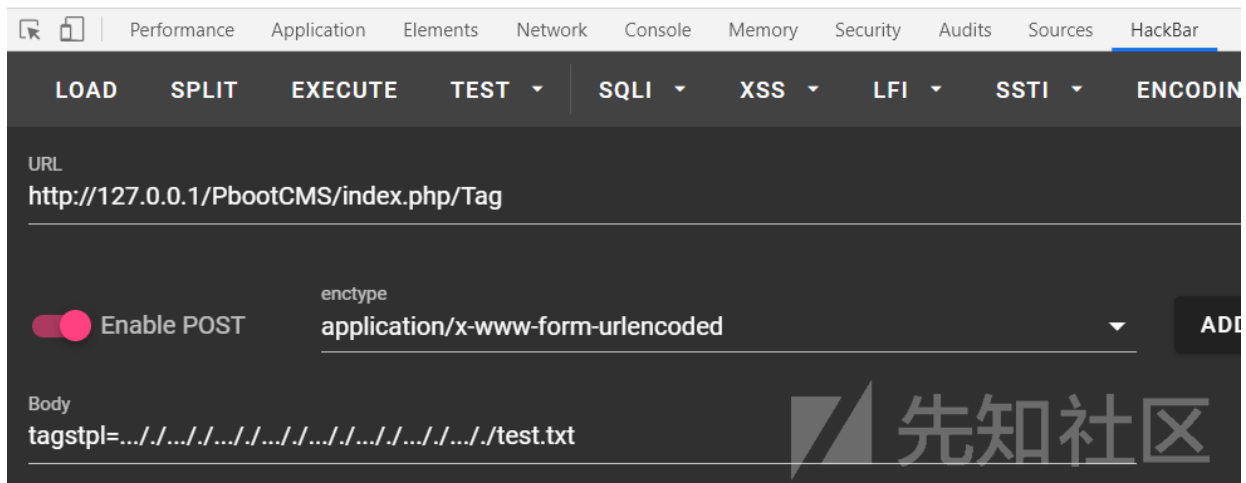
    if (! get('tag')) {
        _404('您访问的页面不存在，请核对后重试! ');
    }
    $a=get('tag');
    $tagstpl = request('tagstpl');
    if (! preg_match('/^[w\-\.\.]+$', $tagstpl)) {
        $tagstpl = 'tags.html';
    }
    $content = parent::parser($this->htmlDir . $tagstpl); // 框架标签解析
    $content = $this->parser->parserBefore($content); // CMS公共标签前置解析
    $content = $this->parser->parserPositionLabel($content, 0, '相关内容', homeurl('tag' . get('tag'))); // CMS当前位置标签解析
    $content = $this->parser->parserSpecialPageSortLabel($content, - 2, '相关内容', homeurl('tag' . get('tag'))); // 解析分类标签
    $content = $this->parser->parserAfter($content); // CMS公共标签后置解析
    $this->cache($content, true);
}
```

传入parser的参数，是通过request接收的参数\$tagstpl和\$this->htmlDir拼接的，因为已经知道在函数内部可以出现目录穿越，所以前面的路径不管怎么拼接都无所谓啦。
这样就完成了整个攻击链，TagController->parser->双写绕过->文件读取->文件写入->文件包含

00x1 漏洞验证

因为是windows搭的环境，就不读/etc/passwd了，读一下D盘根目录的文件吧

success



成功
再包含个phpinfo试试

127.0.0.1/PbootCMS/index.php/Tag

PHP Version 7.3.4

System	Windows NT DESKTOP-QHF49CP 10.0 build 18362 (W
Build Date	Apr 2 2019 21:50:57
Compiler	MSVC15 (Visual C++ 2017)
Architecture	x64
Configure Command	cscrip /nologo configure.js "--enable-snapshot-build' pdo-oci=c:\php-snap-build\deps_aux\oracle\x64\insta snap-build\deps_aux\oracle\x64\instantclient_12_1\sd enable-com-dotnet=shared" "--without-analyzer" "--v
Server API	CGI/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	C:\WINDOWS
Loaded Configuration File	D:\phpstudy_pro\Extensions\php\php7.3.4nts\php.ini
Scan this dir for additional .ini files	(none)

Performance Application Elements Network Console Memory Security Audits Sources HackBar

LOAD SPLIT EXECUTE TEST SQLI XSS LFI SSTI ENCODING HAS

URL
http://127.0.0.1/PbootCMS/index.php/Tag

Enable POST enctype application/x-www-form-urlencoded ADD HEADER

Body
tagstpl=../../../../../../../../../../../../test.php

先知社区

也是可以的

漏洞验证完成

本来是想再看看有没有什么组合利用的姿势，毕竟文件包含这种洞本身利用的灵活度还是蛮高的

不过既然最新版已经修了 就不多看了

00x2 修复方案

官方的修复非常简单粗暴，没有对内核中导致漏洞的根本原因过滤不充分进行修改，而是直接对TagController的正则进行了修改，强制限制了后缀名为html。

看一下前后对比

2.07:

```
if (! preg_match( pattern: '/^[\\w\\-\\.\\_\\/]+$/', $tagstpl)) {  
    $tagstpl = 'tags.html';  
}
```

先知社区

2.08:

```
37 .....  
38 ..... $tagstpl = request('tagstpl');  
39 ..... if (! preg_match('/^[\\w]+\\.html$/',$tagstpl)) {  
40 .....     $tagstpl = 'tags.html';  
41 ..... }
```

先知社区

[上一篇： 蚁剑改造过WAF系列（二）](#)[下一篇： SSTI模板注入\(Python+J...](#)

6 条回复



坏虾

2020-05-19 14:03:11

很棒的分析。

 0 [回复Ta](#)

Gh0stF

2020-05-19 17:18:09

既然没有对内核做出调整，那是否可以说只要能再次找到可控参数的parser调用，依旧可以造成漏洞呢

 0 [回复Ta](#)

P3rh4ps

2020-05-19 17:57:40

[@Gh0stF](#) 是的

 0 [回复Ta](#)

撕袜方丈

2020-06-29 02:03:20

```
// 解析模板文件
public function parser($file)
{
    // 设置主题
    $theme = isset($this->vars['theme']) ? $this->vars['theme'] : 'default';

    $theme = preg_replace_r('{\.\.(\./|\\\\)}', '', $theme); // 过滤掉相对路径
    $file = preg_replace_r('{\.\.(\./|\\\\)}', '', $file); // 过滤掉相对路径

    if (strpos($file, '/') === 0) { // 绝对路径模板
        $tpl_file = ROOT_PATH . $file;
    } elseif (!! $pos = strpos($file, '@')) { // 跨模块调用
```

先知社区

实际上已经在V2.0.8修复了

 0 [回复Ta](#)

撕袜方丈

2020-06-29 02:04:15

[@撕袜方丈](#) 这里的新的preg_replace_r函数就是新增的自定义函数，功能就是循环replace

👍 0 回复Ta



P3rh4ps

2020-11-02 22:01:24

@[撕袜方丈](#) 是这样的 是我写的时候疏忽了 感谢师傅指正

👍 0 回复Ta

[登录](#) 后跟帖

[RSS](#) | [关于社区](#) | [友情链接](#) | [社区小黑板](#) | [举报中心](#) | [我要投诉](#)