

secret

文件拿到手看到的只是各种？（不可见字符）

[illegible]

先改一下缩进，方便审计：

```

1 <?php
2 if (!defined("FFFDCCAFAF"))
3 {
4     define("FFFDCCAFAF", __FILE__);
5     global $0,$1,$2,$3,$4,$5,$6,$7,$8,$9,$10,$11,$12,$13,$14,$15,$16,$17,$18,$19,$20,$21,$22,$23,$24,$25,$26,$27,$28,$29,$30,$31,$32,$33,$34,$35,$36,$37,$38,$39,$40,$41,$42,$43,$44,$45,$46,$47,$48,$49,$50,$51,$52,$53,$54,$55,$56,$57,$58,$59,$60,$61,$62,$63,$64,$65,$66,$67,$68,$69,$70,$71,$72,$73,$74,$75,$76,$77,$78,$79,$80,$81,$82,$83,$84,$85,$86,$87,$88,$89,$90,$91,$92,$93,$94,$95,$96,$97,$98,$99,$100,$101,$102,$103,$104,$105,$106,$107,$108,$109,$110,$111,$112,$113,$114,$115,$116,$117,$118,$119,$120,$121,$122,$123,$124,$125,$126,$127,$128,$129,$130,$131,$132,$133,$134,$135,$136,$137,$138,$139,$140,$141,$142,$143,$144,$145,$146,$147,$148,$149,$150,$151,$152,$153,$154,$155,$156,$157,$158,$159,$160,$161,$162,$163,$164,$165,$166,$167,$168,$169,$170,$171,$172,$173,$174,$175,$176,$177,$178,$179,$180,$181,$182,$183,$184,$185,$186,$187,$188,$189,$190,$191,$192,$193,$194,$195,$196,$197,$198,$199,$200,$201,$202,$203,$204,$205,$206,$207,$208,$209,$210,$211,$212,$213,$214,$215,$216,$217,$218,$219,$220,$221,$222,$223,$224,$225,$226,$227,$228,$229,$230,$231,$232,$233,$234,$235,$236,$237,$238,$239,$240,$241,$242,$243,$244,$245,$246,$247,$248,$249,$250,$251,$252,$253,$254,$255,$256,$257,$258,$259,$260,$261,$262,$263,$264,$265,$266,$267,$268,$269,$270,$271,$272,$273,$274,$275,$276,$277,$278,$279,$280,$281,$282,$283,$284,$285,$286,$287,$288,$289,$290,$291,$292,$293,$294,$295,$296,$297,$298,$299,$300,$301,$302,$303,$304,$305,$306,$307,$308,$309,$310,$311,$312,$313,$314,$315,$316,$317,$318,$319,$320,$321,$322,$323,$324,$325,$326,$327,$328,$329,$330,$331,$332,$333,$334,$335,$336,$337,$338,$339,$340,$341,$342,$343,$344,$345,$346,$347,$348,$349,$350,$351,$352,$353,$354,$355,$356,$357,$358,$359,$360,$361,$362,$363,$364,$365,$366,$367,$368,$369,$370,$371,$372,$373,$374,$375,$376,$377,$378,$379,$380,$381,$382,$383,$384,$385,$386,$387,$388,$389,$390,$391,$392,$393,$394,$395,$396,$397,$398,$399,$400,$401,$402,$403,$404,$405,$406,$407,$408,$409,$410,$411,$412,$413,$414,$415,$416,$417,$418,$419,$420,$421,$422,$423,$424,$425,$426,$427,$428,$429,$430,$431,$432,$433,$434,$435,$436,$437,$438,$439,$440,$441,$442,$443,$444,$445,$446,$447,$448,$449,$450,$451,$452,$453,$454,$455,$456,$457,$458,$459,$460,$461,$462,$463,$464,$465,$466,$467,$468,$469,$470,$471,$472,$473,$474,$475,$476,$477,$478,$479,$480,$481,$482,$483,$484,$485,$486,$487,$488,$489,$490,$491,$492,$493,$494,$495,$496,$497,$498,$499,$500,$501,$502,$503,$504,$505,$506,$507,$508,$509,$510,$511,$512,$513,$514,$515,$516,$517,$518,$519,$520,$521,$522,$523,$524,$525,$526,$527,$528,$529,$530,$531,$532,$533,$534,$535,$536,$537,$538,$539,$540,$541,$542,$543,$544,$545,$546,$547,$548,$549,$550,$551,$552,$553,$554,$555,$556,$557,$558,$559,$560,$561,$562,$563,$564,$565,$566,$567,$568,$569,$570,$571,$572,$573,$574,$575,$576,$577,$578,$579,$580,$581,$582,$583,$584,$585,$586,$587,$588,$589,$590,$591,$592,$593,$594,$595,$596,$597,$598,$599,$600,$601,$602,$603,$604,$605,$606,$607,$608,$609,$610,$611,$612,$613,$614,$615,$616,$617,$618,$619,$620,$621,$622,$623,$624,$625,$626,$627,$628,$629,$630,$631,$632,$633,$634,$635,$636,$637,$638,$639,$640,$641,$642,$643,$644,$645,$646,$647,$648,$649,$650,$651,$652,$653,$654,$655,$656,$657,$658,$659,$660,$661,$662,$663,$664,$665,$666,$667,$668,$669,$670,$671,$672,$673,$674,$675,$676,$677,$678,$679,$680,$681,$682,$683,$684,$685,$686,$687,$688,$689,$690,$691,$692,$693,$694,$695,$696,$697,$698,$699,$700,$701,$702,$703,$704,$705,$706,$707,$708,$709,$710,$711,$712,$713,$714,$715,$716,$717,$718,$719,$720,$721,$722,$723,$724,$725,$726,$727,$728,$729,$730,$731,$732,$733,$734,$735,$736,$737,$738,$739,$740,$741,$742,$743,$744,$745,$746,$747,$748,$749,$750,$751,$752,$753,$754,$755,$756,$757,$758,$759,$760,$761,$762,$763,$764,$765,$766,$767,$768,$769,$770,$771,$772,$773,$774,$775,$776,$777,$778,$779,$780,$781,$782,$783,$784,$785,$786,$787,$788,$789,$790,$791,$792,$793,$794,$795,$796,$797,$798,$799,$800,$801,$802,$803,$804,$805,$806,$807,$808,$809,$810,$811,$812,$813,$814,$815,$816,$817,$818,$819,$820,$821,$822,$823,$824,$825,$826,$827,$828,$829,$830,$831,$832,$833,$834,$835,$836,$837,$838,$839,$840,$841,$842,$843,$844,$845,$846,$847,$848,$849,$850,$851,$852,$853,$854,$855,$856,$857,$858,$859,$860,$861,$862,$863,$864,$865,$866,$867,$868,$869,$870,$871,$872,$873,$874,$875,$876,$877,$878,$879,$880,$881,$882,$883,$884,$885,$886,$887,$888,$889,$890,$891,$892,$893,$894,$895,$896,$897,$898,$899,$900,$901,$902,$903,$904,$905,$906,$907,$908,$909,$910,$911,$912,$913,$914,$915,$916,$917,$918,$919,$920,$921,$922,$923,$924,$925,$926,$927,$928,$929,$930,$931,$932,$933,$934,$935,$936,$937,$938,$939,$940,$941,$942,$943,$944,$945,$946,$947,$948,$949,$950,$951,$952,$953,$954,$955,$956,$957,$958,$959,$960,$961,$962,$963,$964,$965,$966,$967,$968,$969,$970,$971,$972,$973,$974,$975,$976,$977,$978,$979,$980,$981,$982,$983,$984,$985,$986,$987,$988,$989,$990,$991,$992,$993,$994,$995,$996,$997,$998,$999,$1000,$1001,$1002,$1003,$1004,$1005,$1006,$1007,$1008,$1009,$1010,$1011,$1012,$1013,$1014,$1015,$1016,$1017,$1018,$1019,$1020,$1021,$1022,$1023,$1024,$1025,$1026,$1027,$1028,$1029,$1030,$1031,$103
```

发现好多类似base64的东西，但是并不是所有的直接解都有意义，

PI

You

帮我写一个脚本，要求输入两个字符串，并将第一个字符串中与第二个字符串相同的字符用第二个字符串逆序的相同位置的字符替换



ChatGPT

当你提到脚本，我假设你是指使用脚本语言。以下是一个使用 Python 编写的简单脚本，用于实现你描述的要求：

python

Copy code

```
def replace_with_reversed(original_str, replacement_str):
    replaced_chars = []

    for char in original_str:
        if char in replacement_str:
            reversed_char = replacement_str[len(replacement_str) - replacement_str.index(char) - 1]
            replaced_chars.append(reversed_char)
        else:
            replaced_chars.append(char)

    return ''.join(replaced_chars)

# 输入两个字符串
str1 = input("请输入第一个字符串：")
str2 = input("请输入第二个字符串：")
```

```
def replace_with_reversed(original_str, replacement_str):
    replaced_chars = []

    for char in original_str:
        if char in replacement_str:
            reversed_char = replacement_str[len(replacement_str) - replacement_str.index(char) - 1]
            replaced_chars.append(reversed_char)
        else:
            replaced_chars.append(char)

    return ''.join(replaced_chars)

# 输入两个字符串
str1 = input("请输入第一个字符串：")
str2 = input("请输入第二个字符串：")

# 替换并输出结果
result = replace_with_reversed(str1, str2)
print("替换后的字符串：", result)
```

```
}  
  
$?=??"strtr");  
$?=??"strrev");  
$?=??"eval");  
$?=??"gzuncompress");  
$?=??"preg_replace");  
$?=??" /bb612e36177b7793d9eb7de0ed259273/e");  
$?=??"base64_decode");  
  
function ????(&$?????)
```

对else同理:

```
else
{
global $?, $??, $???, $????, $?????, $?????!, $????????, $?????????, $?????????!, $???????????;
$?????DEL?????,$?????????????, $?????????????????, $?????????????DEL?????DEL?;
$?????=?( "c3RydHI=");
$????=?("c3RycmV2");
$???=?("wXwhbA==","ZcCWw");
$????=?("x3p1bmRvbXByxXRn","ZzGRNsxn");
$????=?("eHJlZl9vZXBsVWNI","cVvryYe");
$????=?("F2JinJEyZGM2MGk3Yjk3OGNcOWViN2RlMTVcmjU5MjQzF2U=", "LcGKtKF");
$????=?("KiFZZTK0X2Rlk29kZB==","YiQBmK");
}
```

简单替换一下后观察这个四个? 的函数（以下称fun2，两个? 的函数称fun1）：

[illegible]

这是在干嘛，让gpt帮忙审计，大概理解，是对fun2输入的参数赋值，选中的内容与赋值有关；

先对这一段解一下，看看这段是什么样的代码：

```

1  <?php
2  $?=?("rXhwbG9krQ==","ZjPanpr");
3  $??=?("emNsb3Nl","ZkSjMJe");
4  $????=?("rmlsrXNpemU=","ZqnuhOFr");
5  $????=?("CnJlYWQ=","ZGFOAPC");
6  $????=?("g3Vig3Rj","cjINUyg");
7  $????=?("em9welw4=","ZszKe");
8  $????DEL=?("nl5fYXJyYXk=","awJxQOGln");
9  $????DEL=?("awQ1","bRZIHVa");
10 $????DEL=?("IhJlZ19tEXRjaA==","chEYHI");
11 $?=?=__FILE__;
12 if($????DEL?/?(.+?)\(/",$?, $???)
13 {
14     $????=?[1];
15 }
16 else
17 {
18     $????=?;
19 }
20 $????=?($?, "rb");
21 $????=?($?, $?($?));
22 $?($?);
23 $????($?, -22)==$?($?DEL?($?));
24 $????=@$?($?($?));
25 $????=time();
26 $????=?("U3l1bmNvbXB?yUXNz","ZlupU");
27 ?>

```

用同样的方法再替换审计一下，好多似乎是无用的步骤，

注意这一行：

```

10 $??DEL??DEL?="preg_match";
11 $??=__FILE__;
12 if($preg_match("/(.*?)\\(/",fclose,eval))
13 {
14     $???=eval[1];
15 }
16 else
17 {
18     $???=fclose;
19 }
20 $???=fopen($???, "rb");
21 $???=fread($???,filesize($???));
22 fclose($???);
23 substr($??.-22)==substr(md5(substr(substr($???,0,-32).
24 $???=@gzuncompress(base64_decode($???));
25 substr=time();
26 $???="gzuncompress";
27 // $???="eval";
28 // $???="gzuncompress";
29 // $???="preg_replace";
30 // $???="strtr";
31 // $???="/bb612e36177b7793d9eb7de0ed259273/e";
32 // $???="strrev";
33 // $???="base64_decode";
34 ?>
35

```

与主函数中调用fun2输入的函数有关联


```

41  global $?, $??, eval, gzuncompress, preg_replace, strtr,
    $????????????????, $????????????????DEL?, $??
42
43  $???="eval";
44  $????="gzuncompress";
45  $?????="preg_replace";
46  $??????="strtr";
47  $????????="/bb612e36177b7793d9eb7de0ed259273/e";
48  $????????????="strrev";
49  $????????????????="base64_decode";
50
51  dec1="eNrzzTepAgADZwFr";
52  $????????=fun2(dec1);
53
54  @preg_replace(/bb612e36177b7793d9eb7de0ed259273/e, eval."(
    +B12yEkK?SXPNws9r?pUi/BO+M?IbUeLRFLDELQwFfEA0t?gAgFWtuK
    3L?f/Hh27/3?re9//8Nv?f/79P/yT?D9/7o3/7?e7/770//?/o++

```

但有什么用呢？

看最后还有一大串base64，而且执行它的代码跟fun2中的很相似，猜测是同种，

直接对这一大串解密时发现中间有两个未知的量插进来了；

在更改一下变量名和fun2函数内的代码，使其便于审计：

```

7  $????????????????="base64_decode";
8
9  function fun2(&$input2)
10 {
11     global $?, $??, $???, $????, $?????, $?????, $?????, $?????, $?????,
12     $????????????????, $????????????????DEL?, $????????DEL?, $????????DEL?, $????
13
14     $????????DEL????????DEL?="die";
15
16     $input2 = @gzuncompress(base64_decode(eNrzzTepAgADZwFr)); //$enc1 = Mo4z
17
18     return "7"; //$enc1 = 7
19 }
20 else

```

```

8 $??=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=?=????

```

enc1的初始值是直接前面**fun1**的规则**base64**解密出来的，然后再**fun2**中再一次被解密+解压成"**Mo4z**"，

而**enc2**作为**fun2**的返回值，直接就是"**7**"。

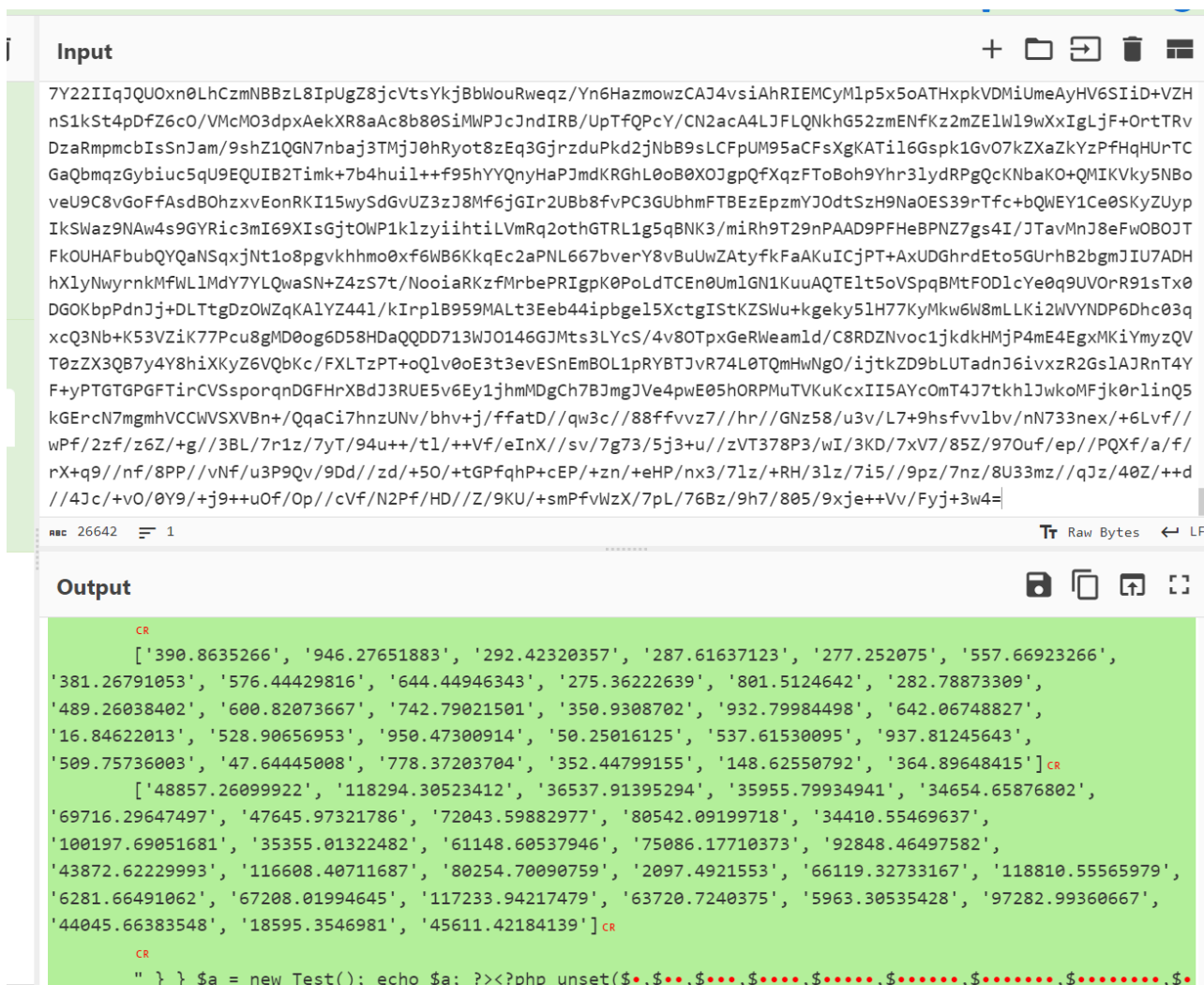
哎，这后面的两个参数不就是这个**enc2**和**enc1**吗。

```

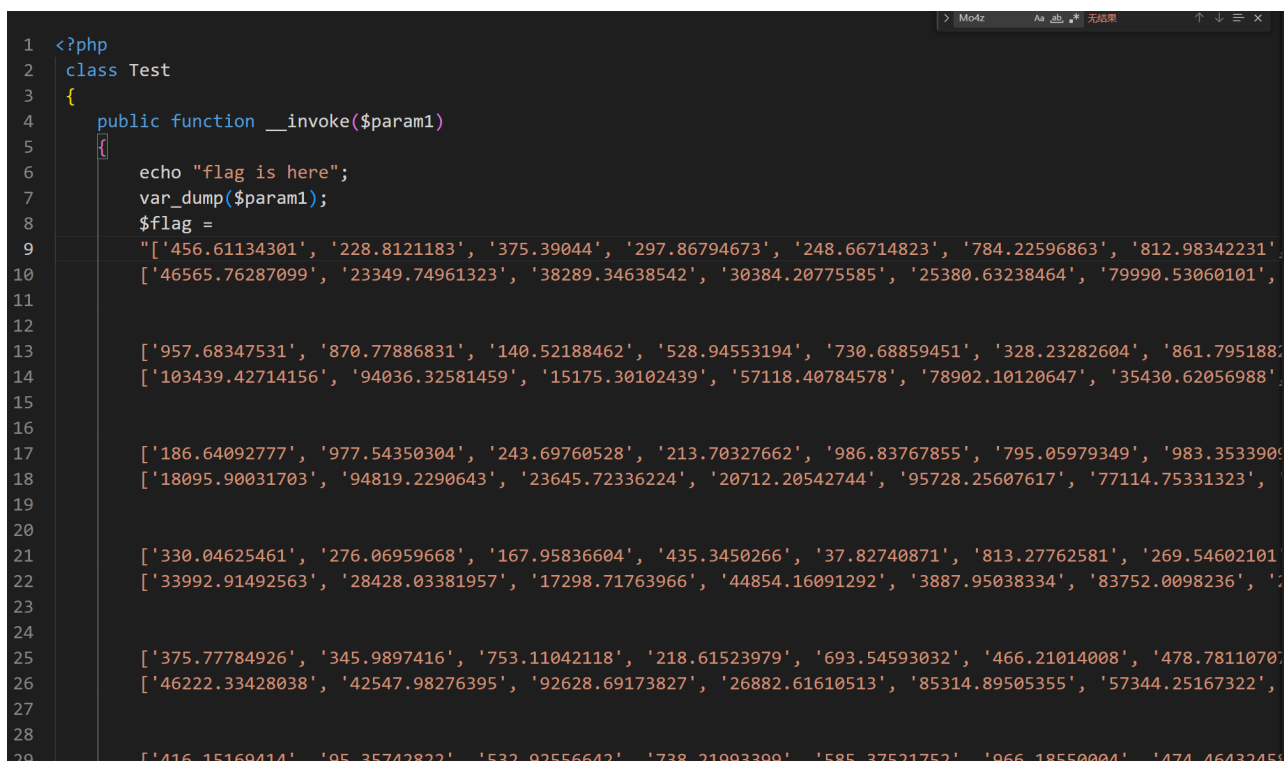
+7gph0YUrBGxThbSIEbBYWhOFSP8kFfEGn1PnxHp5agZ8Tf+1T74VAe/Oi4dtE6XeJko17h6
hik9UQcOxwvSkb6FTPQBgVYOEQu70lSynHrSMntm6WmJ0gJxt9w4YgAGTZK7082A7QIERR7J
spDYXeDZPgIEHiSQXXTa2XvQWOpZo77rwo0/hd3vmb5XskbCbd8yPbR0C2g3NpG0jXYWmpSM
+QQKNHjOw9VM2Bwm1YagZtICvAk4ssWgZo0Quq6sP1U49E+i7QDaB0a0yYygjr qH
0
1 . $enc2.$enc1.
2
3 2r10A93KEBwVpp4HCHqrumzhuMmYKM0tu7ion68qTKYhq14T1sMJdi00kpY8BsS+42aSkkMr
+g3mM1Ht0jVZyrWaUbDro7U4cUHEfyadLLlMnXXds2lECgG5u+T0k8iuvB6632U4NRjaMB9c
gxwbcSHfG1FHs07pb2/CxngcCuUzTMDoDGKi9kFu+C0Y8CP/enVQE2GRf4LQXT4wyH6wBbz6
+UrwXF8fMQaGaAyn0Inz0nMFrs2TF6R11x4MH1PNr6RqRNha3dvQA4tGfRzEiTdGHQGyZoY7
a0giAsaNNATT0szPP

```

把值带进去后解密（这里用cyberchef直接**base64**+解压）：



得到一大串有关flag的数据：



首先想到画图，分别以第一行第二行为横纵坐标，但是画出来的结果几乎是一条直线。。。

那么斜率相同

算出前四组的斜率，

102 108 97 103 -> flag

猜想正确，解出全部斜率，转化ASCII码再连成片，得到flag（我用手算的，很容易输错）：

ASCII文字

```
flag{cb933a3b--4d1e-44c4-8d8--3d0669f6b95b}
```

十六进制 (字节)

```
66 6C 61 67 7B 63 62 39 33 33 61 33 62 2D 2D 34 64 31 65 2D 34 34 63
34 2D 38 64 38 2D 2D 33 64 30 36 36 39 66 36 62 39 35 62 7D
```

二进制 (字节)

```
01100110 01101100 01100001 01100111 01111011 01100011 01100010
00111001 00110011 00110011 01100001 00110011 01100010 00101101
00101101 00110100 01100100 00110001 01100101 00101101 00110100
```

十进制 (字节)

```
102 108 97 103 123 99 98 57 51 51 97 51 98 45 45 52 100 49 101 45 52
52 99 52 45 56 100 56 45 45 51 100 48 54 54 57 102 54 98 57 53 98 125
```

flag{cb933a3b-4d1e-44c4-8d8b-3d0669f6b95b}

ezrsa

```
from Crypto.Util.number import getPrime
from secret import flag

p = getPrime(512)
print(p,pow(flag, 2, p))

# p =
4124820799737107236308837008524397355107786950414769996181324333556
9501542069800594064027673277253122386730535811486414384942123201576
65395208337575556385
# c =
1310793956350745977461620414125374748923206333620417394412326328450
7604328885680072478669016969428366667381358004059204207134817952620
014738665450753147857
```

代码很简单，但是 $e = 2$ ， $q = 1$ 限制了破解的方法：

尝试使用**Tonelli_Shanks**算法

代码参考：https://blog.csdn.net/qq_51999772/article/details/122642868

```
from Crypto.Util.number import *

r =
4124820799737107236308837008524397355107786950414769996181324333556
9501542069800594064027673277253122386730535811486414384942123201576
65395208337575556385# r = p
c =
1310793956350745977461620414125374748923206333620417394412326328450
7604328885680072478669016969428366667381358004059204207134817952620
014738665450753147857
def Legendre(n,p):
    return pow(n,(p - 1) // 2,p)
def Tonelli_Shanks(n,p):
    assert Legendre(n,p) == 1
    if p % 4 == 3:
        return pow(n,(p + 1) // 4,p)
    q = p - 1
    s = 0
```

```

while q % 2 == 0:
    q = q // 2
    s += 1
for z in range(2,p):
    if Legendre(z,p) == p - 1:
        c = pow(z,q,p)
        break
r = pow(n,(q + 1) // 2,p)
t = pow(n,q,p)
m = s
if t % p == 1:
    return r
else:
    i = 0
    while t % p != 1:
        temp = pow(t,2**(i+1),p)
        i += 1
        if temp % p == 1:
            b = pow(c,2**(m - i - 1),p)
            r = r * b % p
            c = b * b % p
            t = t * c % p
            m = i
            i = 0
    return r

result = Tonelli_Shanks(c,r)
print(long_to_bytes(result))
print(long_to_bytes(r - result))

```

改一下我们的参数，哎怎么assert断掉了

原来是p和c的值放反了

```

from Crypto.Util.number import *

c =
4124820799737107236308837008524397355107786950414769996181324333556
9501542069800594064027673277253122386730535811486414384942123201576
65395208337575556385

```

```

r =
1310793956350745977461620414125374748923206333620417394412326328450
7604328885680072478669016969428366667381358004059204207134817952620
014738665450753147857# r = p
def Legendre(n,p):
    return pow(n,(p - 1) // 2,p)
def Tonelli_Shanks(n,p):
    assert Legendre(n,p) == 1
    if p % 4 == 3:
        return pow(n,(p + 1) // 4,p)
    q = p - 1
    s = 0
    while q % 2 == 0:
        q = q // 2
        s += 1
    for z in range(2,p):
        if Legendre(z,p) == p - 1:
            c = pow(z,q,p)
            break
    r = pow(n,(q + 1) // 2,p)
    t = pow(n,q,p)
    m = s
    if t % p == 1:
        return r
    else:
        i = 0
        while t % p != 1:
            temp = pow(t,2**(i+1),p)
            i += 1
            if temp % p == 1:
                b = pow(c,2**(m - i - 1),p)
                r = r * b % p
                c = b * b % p
                t = t * c % p
                m = i
                i = 0
        return r

result = Tonelli_Shanks(c,r)
print(long_to_bytes(result))
print(long_to_bytes(r - result))

```

运行得到flag:

开始运行...

```
b'\xfaFF"\x0bxn\x93\xd1\xfd8\x91\x8d;g\x8c\xf7Wj\xcf\x8c\xde\x
3\x90aT'
b'flag{9971e255f0c020e8e57fbae75f43d7fb}'
```

运行结束。

```
flag{9971e255f0c020e8e57fbae75f43d7fb}
```