强网杯”部分题目Writeup

原创

合天网安实验室 于 2018-03-29 20:56:00 发布 237 收藏

点击上方“合天智汇”，选择“置顶公众号”

有内涵的干货文章第一时间送达！

本文作者：tinyfisher

投稿活动：重金悬赏 | 合天原创投稿等你来！

周末参加了强网杯，虽然只做出了一些题目，收获还是蛮大的，记录一下解题过程和思路，Writeup如下：

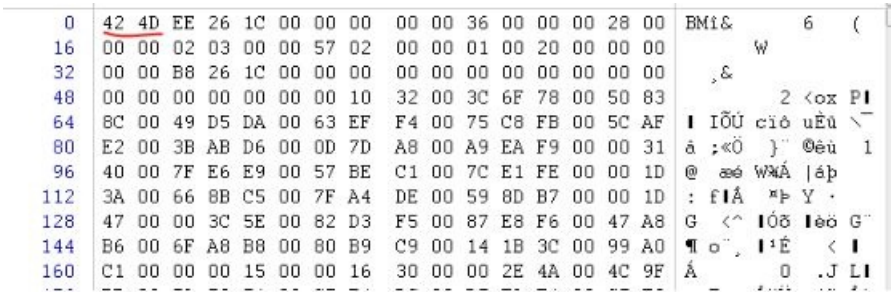
1

Welcome

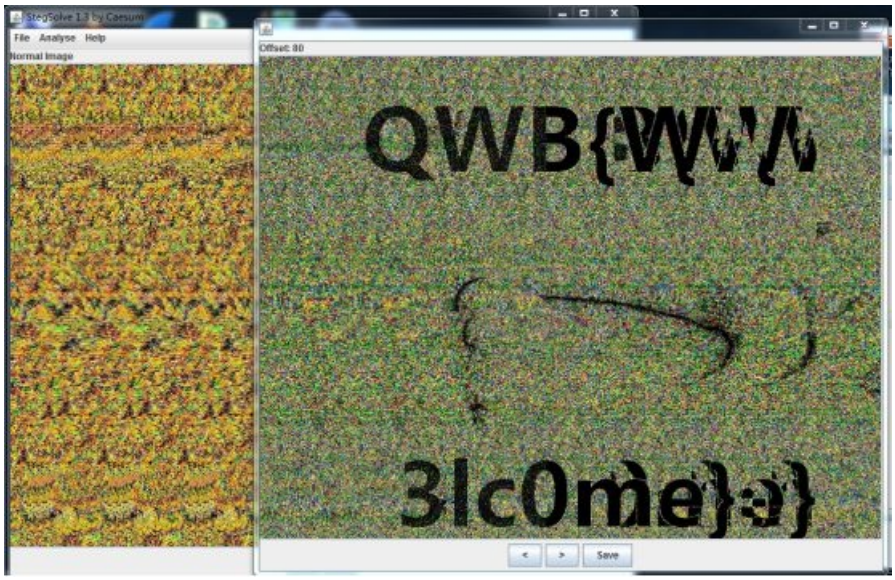
题目描述：

解题思路：

首先下载文件，用winhex看看文件头为424D，判断文件为bmp文件：



尝试用notepad打开看看文件内容中是否有flag，没有发现；然后binwalk一下未发现图片中有隐藏文件；再尝试用stegsolve打开，stereogram不断设置offset，发现图片有一些异常，当offset为80时，出现flag



2

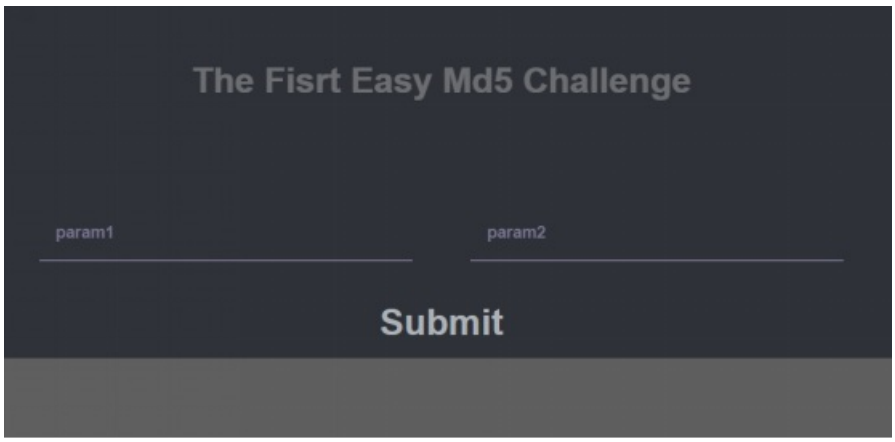
web签到

题目描述：

解题思路：

这题还是蛮有意思的，虽说是签到，考察的点很好

第一关：



看一下源代码：

很基础的==弱类型判断，要使得param1!=param2并且md5(param1)==md5(param1)

两边都是==弱类型判断，这里说一下==和===的区别:

要使$a == $b，只需要类型转换后 $a 等于 $b即可；要使$a === $b，则不但需要 $a 等于 $b，并且需要它们的类型也相同。可以明确的看到，==会在比较的时候进行类型转换的比较。

如果比较一个数字和字符串或者比较涉及到数字内容的字符串，则字符串会被转换为数值并且比较按照数值来进行。

绕过方式1：

param1=240610708，param2=QNKCDZO，这两个参数不相等；

md5('240610708') 的结果是：0e462097431906509019562988736854

md5('QNKCDZO') 的结果是：0e830400451993494058024219903391

由于是==，0e462097431906509019562988736854在比较的时候会做类型转换成数字，而0e开头代表科学计数法，所以无论0e后面是什么，0的多少次方还是0，这样就可以绕过。本地测试：

```php
<?php
var_dump(md5('240610708')==md5('QNKCDZO'));
?>
```

```
C:\Users\TinyP\Desktop>php 1.php
bool(true)
```

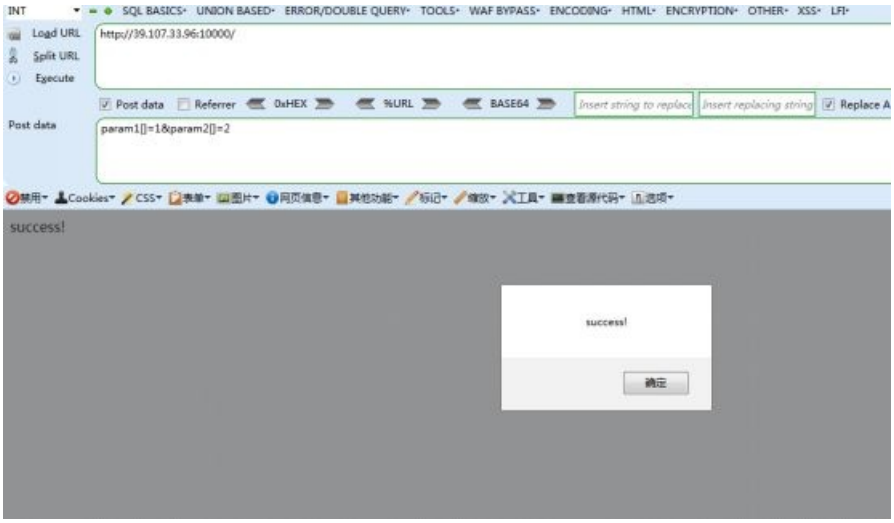
绕过方式2：

param1[]=1¶m2[]=2

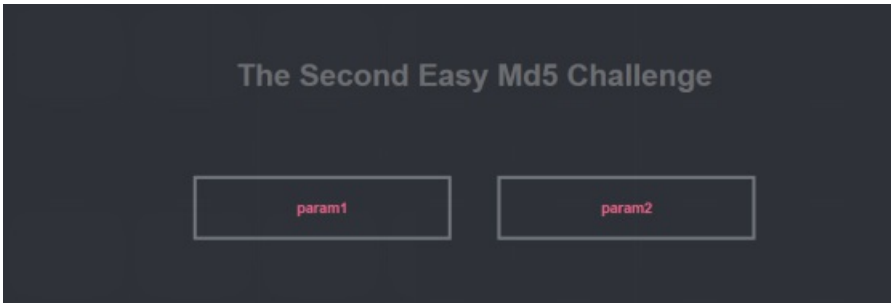
这里param1和param2都是数组，值不相等，但是md5（数组）会报错，返回null，因此

md5(param1)==md5(param1)，也就是null==null，也可以绕过。

综上，可以构造数据或者md5 0e开头的字符串绕过，无需md5碰撞：



第二关：



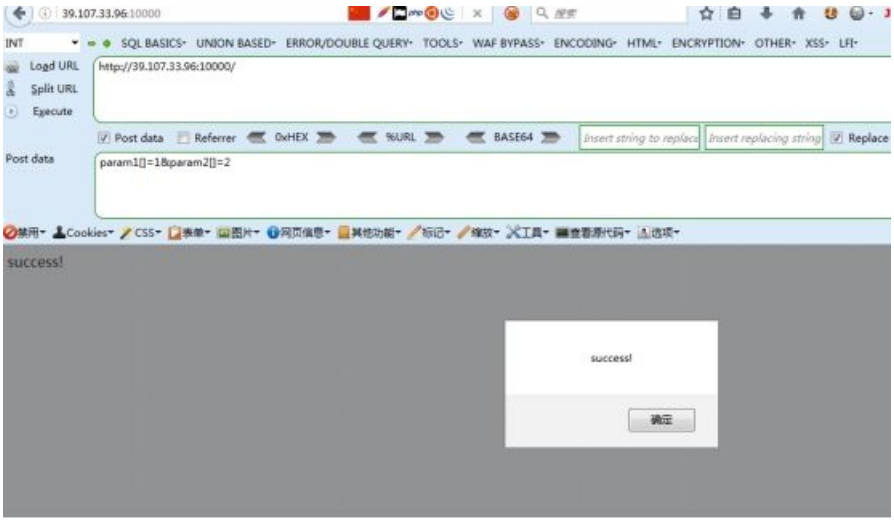
看一下源代码：

```
<h2>The Second Easy Md5 Challenge</h2>
<!--
    if($_POST['param1']!==$_POST['param2'] && md5($_POST['param1'])===md5($_POST['param2'])){
        die("success!");
    }
-->
<span class="input input--kuro">
```

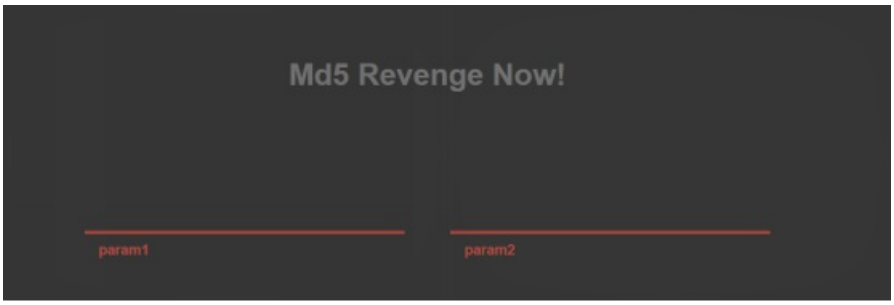
这里param1!==param2并且md5(param1)===md5(param1)，两边都是===判断，和第一关的==弱类型判断不一样，此时0e462097431906509019562988736854!== 0e830400451993494058024219903391，因为这里不做类型转换，当做字符串处理。这里只能用数组绕过，md5（数组）会报错，返回null，null===null

payload：

param1[]=1¶m2[]=2



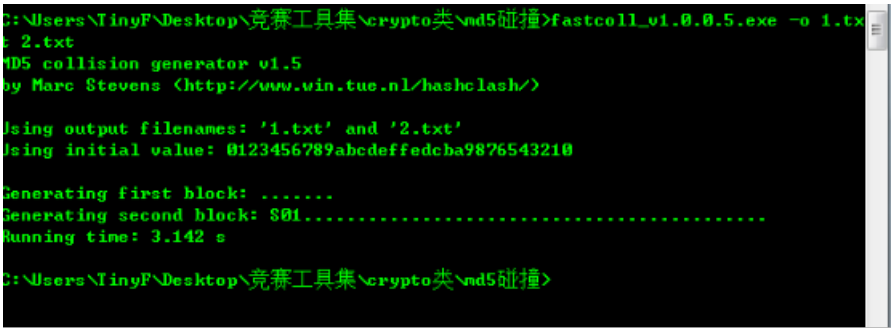
第三关：



看一下源代码：

```
<h2>Md5 Revenge Now!</h2>
<!--
if((string)$_POST['param1']!==(string)$_POST['param2'] && md5($_POST['param1'])===md5($_POST['param2'])){
    die("success!");
}
-->
```

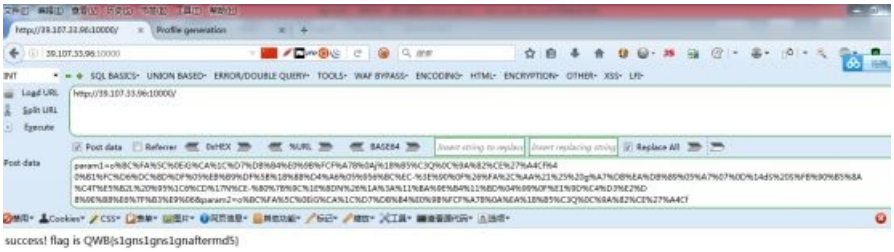
这里两边都是强判断===，并且强制转换为string类型进行比较，想了很久，只能通过md5碰撞绕过去，早知道第三关这样，前面几关也都可以用md5碰撞绕过。首先用fastcoll生成2个md5一致的文件：



然后将这两个文件的内容通过url编码传进去即可：

Payload：

```
param1=o%BC%FA%5C%0EiG%CA%1C%D7%DB%B4%E0%9B%FCF%A78%0Aj%18%B5%C3Q%0C%9A%
0%B1%FC%D6%DC%8D%DF%05%EB%B9%DF%5B%18%88%D4%A6%05%956%BC%EC-
%3E%90%0F%26%FA%2C%AA%21%25%20g%A7%DB%EA%DB%89%05%A7%07%0D%14dS%20S%FB%
%80%7B%9C%1E%8DN%26%1A%3A%11%BA%9E%B4%11%BD%04%99%0F%E1%9D%C4%D3%E2%D
8%9E%B8%E6%7F%B3%E9%06¶m2=o%BC%FA%5C%0EiG%CA%1C%D7%DB%B4%E0%9B%FCF%A78%
```



3

streamgame1

题目介绍：

题目给了一个算法和一个key：

```
'''
from flag import flag
assert flag.startswith("flag{")
assert flag.endswith("}")
assert len(flag)==25
'''
def lfsr(R,mask):
    output = (R << 1) & 0xffffff
    i=(R&mask)&0xffffff
    lastbit=0
    while i!=0:
        lastbit^=(i&1)
        i=i>>1
    output^=lastbit
    return (output,lastbit)


R=int(flag[5:-1],2)
mask    =   0b10100011000100011100

f=open("key","ab")
for i in range(12):
    tmp=0
    for j in range(8):
        (R,out)=lfsr(R,mask)
        tmp=(tmp << 1)^out
    f.write(chr(tmp))
f.close()
```

这里没有看具体的算法，因为flag长度25位，格式为flag{}，那么中间长度就是19位，而密文key也很短：

key

Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
0	55	38	F7	42	C1	0D	B2	C7	ED	E0	24	3A					U8÷BÁ ²Çíà$:

尝试直接用爆破进行测试，按照题目的算法遍历flag，和key的每一位进行比较，如果匹配，那么该字符就是flag的一部分：

```
def lfsr(R,mask):
    output = (R << 1) & 0xffffff
    i=(R&mask)&0xffffff
    lastbit=0
    while i!=0:
        lastbit^=(i&1)
```

```
        i=i>>1
    output^=lastbit
    return (output,lastbit)

#R=int(flag[5:-1],2)
mask    =   0b10100011000100011100
#print f1.read()
for R in range(0,0b11111111111111111111):

    tmpr=R
    for i in range(12):
        tmp=0
        for j in range(8):
            (R,out)=lfsr(R,mask)
            tmp=(tmp << 1)^out
        if i==0:
            if tmp==0x55:
                pass
            else:
                break
        if i==1:
            if tmp==0x38:
                pass
            else:
                break
        if i==2:
            if tmp==0xF7:
                pass
            else:
                break
        if i==3:
```

```
        if tmp==0x42:
            pass
        else:
            break
    if i==4:
        if tmp==0xC1:
            print tmpr
        else:
            break
```

从0到0b1111111111111111111遍历flag，然后带入到加密算法进行计算，根据key的二进制值，比较每一位是否相等，如果第一位相等继续比较第二位，如果不相等则继续遍历，比较各4-5位左右，如果都相等，差不多可以判断遍历成功，tmpr的二进制形式就是flag

4

streamgame2

题目描述：

其实和streamgame 2没什么区别，只是长度变了：

```
from flag import flag
assert flag.startswith("flag{")
assert flag.endswith("}")
assert len(flag)==27

def lfsr(R,mask):
    output = (R << 1) & 0xffffff
    i=(R&mask)&0xffffff
    lastbit=0
    while i!=0:
        lastbit^=(i&1)
        i=i>>1
    output^=lastbit
    return (output,lastbit)



R=int(flag[5:-1],2)
mask=0x100002

f=open("key","ab")
for i in range(12):
    tmp=0
    for j in range(8):
        (R,out)=lfsr(R,mask)
        tmp=(tmp << 1)^out
    f.write(chr(tmp))
f.close()
```

还是遍历，只不过flag长度变成了27位，去掉“flag{}”6位，因此中间长度为21位，也就是0-0b111111111111111111111修改一下长度就可以

```
def lfsr(R,mask):
    output = (R << 1) & 0xffffff
    i=(R&mask)&0xffffff
    lastbit=0
    while i!=0:
        lastbit^=(i&1)
        i=i>>1
    output^=lastbit
    return (output,lastbit)
#R=int(flag[5:-1],2)
mask=0x100002
#print f1.read()
for R in range(0,0b111111111111111111111):
    tmpr=R
    for i in range(12):
        tmp=0
        for j in range(8):
```

```
        (R,out)=lfsr(R,mask)
        tmp=(tmp << 1)^out
    if i==0:
        if tmp==0xB2:
            pass
        else:
            break
    if i==1:
        if tmp==0xE9:
            pass
        else:
            break
    if i==2:
        if tmp==0x0E:
            pass
        else:
            break
    if i==3:
        if tmp==0x13:
            pass
        else:
            break
    if i==4:
        if tmp==0xA0:
            print tmpr
        else:
            break
```

5

streamgame4

题目描述：

换汤不换药，虽说是1024X1024，但flag长度还是固定的21位：

```
from flag import flag
assert flag.startswith("flag{")
assert flag.endswith("}")
assert len(flag)==27

def nlfsr(R,mask):
    output = (R << 1) & 0xffffff
    i=(R&mask)&0xffffff
    lastbit=0
    changesign=True
    while i!=0:
        if changesign:
            lastbit &= (i & 1)
            changesign=False
        else:
            lastbit^=(i&1)
        i=i>>1
    output^=lastbit
    return (output,lastbit)

R=int(flag[5:-1],2)
mask=0b110110011011001101110

f=open("key","ab")
for i in range(1024*1024):
    tmp=0
    for j in range(8):
        (R,out)=nlfsr(R,mask)
        tmp=(tmp << 1)^out
    f.write(chr(tmp))
f.close()
```

因此还是遍历：

```
def nlfsr(R,mask):
    output = (R << 1) & 0xffffff
    i=(R&mask)&0xffffff
    lastbit=0
    changesign=True
    while i!=0:
        if changesign:
            lastbit &= (i & 1)
            changesign=False
        else:
```

```
            lastbit^=(i&1)
        i=i>>1
    output^=lastbit
    return (output,lastbit)

#R=int(flag[5:-1],2)
mask=0b11011001101100110110
#print f1.read()
for R in range(0,0b11111111111111111111):
    tmpr=R
    for i in range(12):
        tmp=0
        for j in range(8):
            (R,out)=nlfsr(R,mask)
            tmp=(tmp << 1)^out

        if i==0:
            if tmp==0xD1:
                pass
            else:
                break
        if i==1:
            if tmp==0xD9:
                pass
            else:
                break
        if i==2:
            if tmp==0x40:
                pass
            else:
                break
```

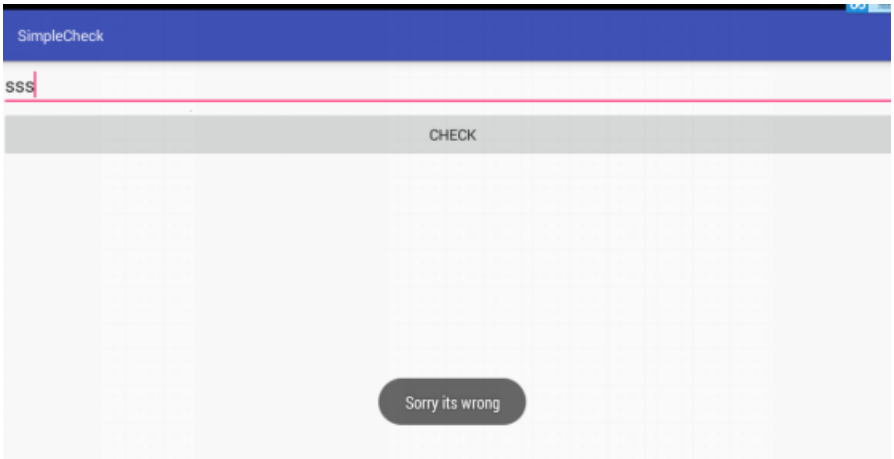
```
    if i==3:
        if tmp==0x43:
            pass
        else:
            break
    if i==4:
        if tmp==0x93:
            print tmpr
        else:
            break
```

6

simplecheck

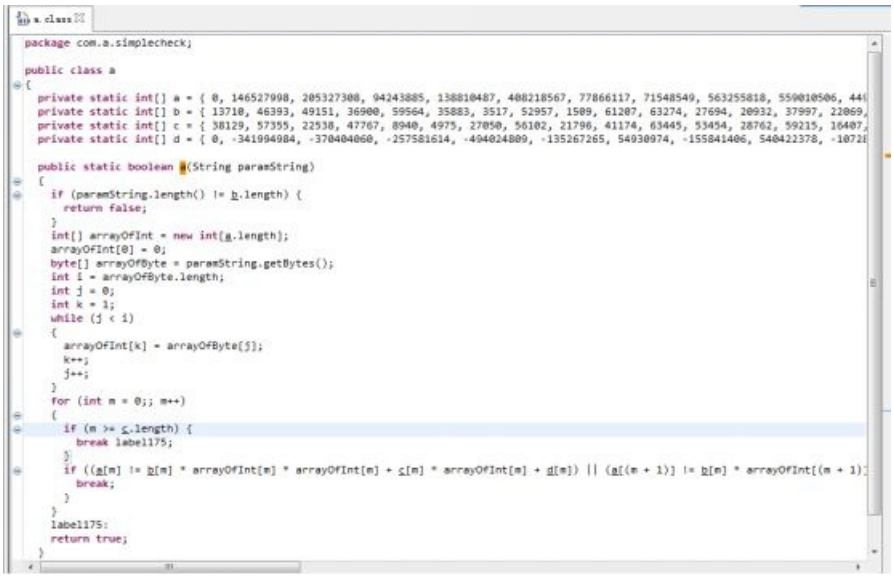
题目描述：

题目给了一个apk，运行下试试：



要输入flag，错误返回“sorry its wrong“

反编译apk，看一下关键代码：



```
package com.a.simplecheck;

public class a
{
  private static int[] a = { 0, 146527908, 205327308, 94243885, 138810487, 408218567, 77866117, 71548549, 563255818, 559010506, 44(
  private static int[] b = { 13710, 46393, 49151, 36900, 59564, 35883, 3517, 52957, 1509, 61207, 63274, 27694, 20932, 37997, 22069,
  private static int[] c = { 38129, 57355, 22538, 47767, 8940, 4975, 27050, 56102, 21796, 41174, 63445, 53454, 28762, 59215, 16407,
  private static int[] d = { 0, -341994984, -370404060, -257581614, -494024809, -135267265, 54930974, -155841406, 540422378, -10728

  public static boolean a(String paramString)
  {
    if (paramString.length() != b.length) {
      return false;
    }
    int[] arrayOfInt = new int[a.length];
    arrayOfInt[0] = 0;
    byte[] arrayOfByte = paramString.getBytes();
    int i = arrayOfByte.length;
    int j = 0;
    int k = 1;
    while (j < i)
    {
      arrayOfInt[k] = arrayOfByte[j];
      k++;
      j++;
    }
    for (int m = 0;; m++)
    {
      if (m >= c.length) {
        break label175;
      }
      if ((a[m] != b[m] * arrayOfInt[m] * arrayOfInt[m] + c[m] * arrayOfInt[m] + d[m]) || (a[(m + 1)] != b[m] * arrayOfInt[(m + 1)]
        break;
      }
    }
    label175:
    return true;
  }
```

这题需要让函数a返回true，传递的参数paramString为flag，需要我们逆出flag，算法大概的意思：

首先定义了一些数组a，b，c，d

往下看代码

```
if (paramString.length() != b.length) {
    return false;
}
```

这里说明了flag的长度需要等于b数组的长度，也就是34，再往下看：

```
int[] arrayOfInt = new int[a.length];
  arrayOfInt[0] = 0;
  byte[] arrayOfByte = paramString.getBytes();
  int i = arrayOfByte.length;
  int j = 0;
  int k = 1;
  while (j < i)
  {
    arrayOfInt[k] = arrayOfByte[j];
    k++;
    j++;
}
```

这里new了一个新数组arrayOfInt，arrayOfInt[0] = 0;然后将flag赋值到arrayOfInt[1]- arrayOfInt[34]，也就是说数组arrayOfInt，第一位为0，后面34位为flag。

再往下看关键代码：

```
for (int m = 0;; m++)
    {
      if (m >= c.length) {
        break label175;
      }
      if ((a[m] != b[m] * arrayOfInt[m] * arrayOfInt[m] + c[m] * arrayOfInt[m] + d[m]) || (a[(m + 1)] != b[m] * arrayOfInt[(m + 1)] * arrayOfInt[(m + 1)] + c[m] * arrayOfInt[(m + 1)] + d[m])) {
        break;
      }
}
```

m从0到34进行遍历，要使得if ((a[m] != b[m] * arrayOfInt[m] * arrayOfInt[m] + c[m] * arrayOfInt[m] + d[m]) || (a[(m + 1)] != b[m] * arrayOfInt[(m + 1)] * arrayOfInt[(m + 1)] + c[m] * arrayOfInt[(m + 1)] + d[m]))为假

由于if里面是||，也就是0||0才为0，转换一下这个条件就是：

a[m] == b[m] * arrayOfInt[m] * arrayOfInt[m] + c[m] * arrayOfInt[m] + d[m]

且

a[(m + 1)] == b[m] * arrayOfInt[(m + 1)] * arrayOfInt[(m + 1)] + c[m] * arrayOfInt[(m + 1)] + d[m]

明白了关键函数，就可以尝试利用爆破区爆破flag：

a= [0, 146527998, 205327308, 94243885, 138810487, 408218567, 77866117, 71548549, 563255818, 559010506, 449018203, 576200653, 307283021, 467607947, 314806739, 341420795, 341420795, 469998524, 417733494, 342206934, 392460324, 382290309, 185532945, 364788505, 210058699, 198137551, 360748557, 440064477, 319861317, 676258995, 389214123, 829768461, 534844356, 427514172, 864054312]

b= [13710, 46393, 49151, 36900, 59564, 35883, 3517, 52957, 1509, 61207, 63274, 27694, 20932, 37997, 22069, 8438, 33995, 53298, 16908, 30902, 64602, 64028, 29629, 26537, 12026, 31610, 48639, 19968, 45654, 51972, 64956, 45293, 64752, 37108]

c=[38129, 57355, 22538, 47767, 8940, 4975, 27050, 56102, 21796, 41174, 63445, 53454, 28762, 59215, 16407, 64340, 37644, 59896, 41276, 25896, 27501, 38944, 37039, 38213, 61842, 43497, 9221, 9879, 14436, 60468, 19926, 47198, 8406, 64666]

d=[0, -341994984, -370404060, -257581614, -494024809, -135267265, 54930974, -155841406, 540422378, -107286502, -128056922, 265261633, 275964257, 119059597, 202392013, 283676377, 126284124, -68971076, 261217574, 197555158, -12893337, -10293675, 93868075, 121661845, 167461231, 123220255, 221507, 258914772, 180963987, 107841171, 41609001, 276531381, 169983906, 276158562]

```
flag=""
for m in range(1,34):
    for f1 in range(32,127):
```

```
        if((a[m] == b[m-1] * f1 * f1 + c[m-1] * f1 + d[m-1]) and (a[m] == b[m] * f1 * f1 + c[m] * f1 + d[m])):
            flag+=chr(f1)
            break
        else:
            pass
#print len(c)
print flag+"}"
```

投

稿

评

优

2018年第一季度原创投稿评优结果将在4月份公布啦！

届时将评选出三个奖项，共计15名原创作者！

积极参与奖

最具文采奖

最佳作者奖

丰厚大礼等着大家哟，快来积极参与投稿吧！

重金悬赏 | 合天原创投稿等你来！（点击了解投稿详情）

合天智汇

网址：*www.heetian.com*

电话：4006-123-731

长按图片，据说只有颜值高的人才能识别哦→