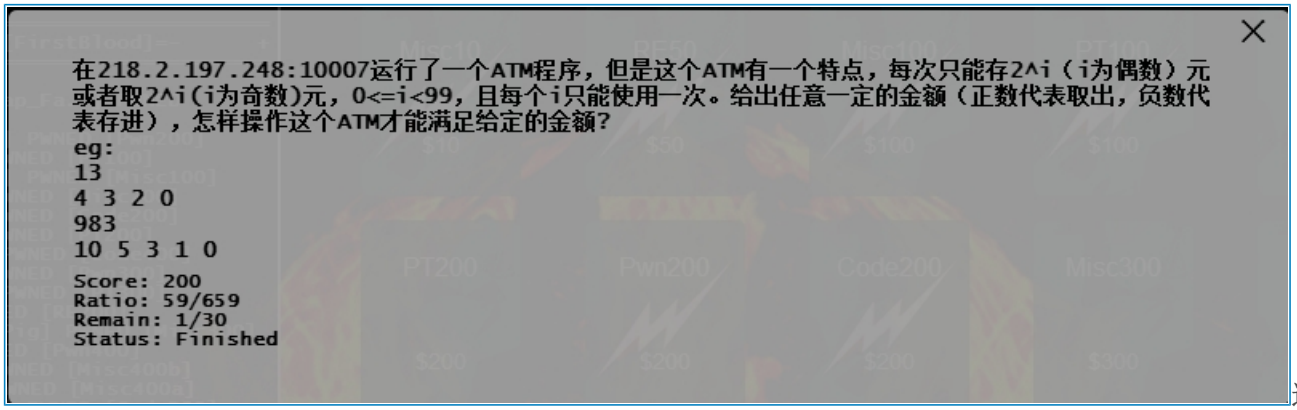
SCTF-2014 writeup（部分）

转载

weixin_33963594 于 2014-12-10 15:39:05 发布 66 收藏
文章标签： json python
原文链接：http://blog.51cto.com/cugou/1588334

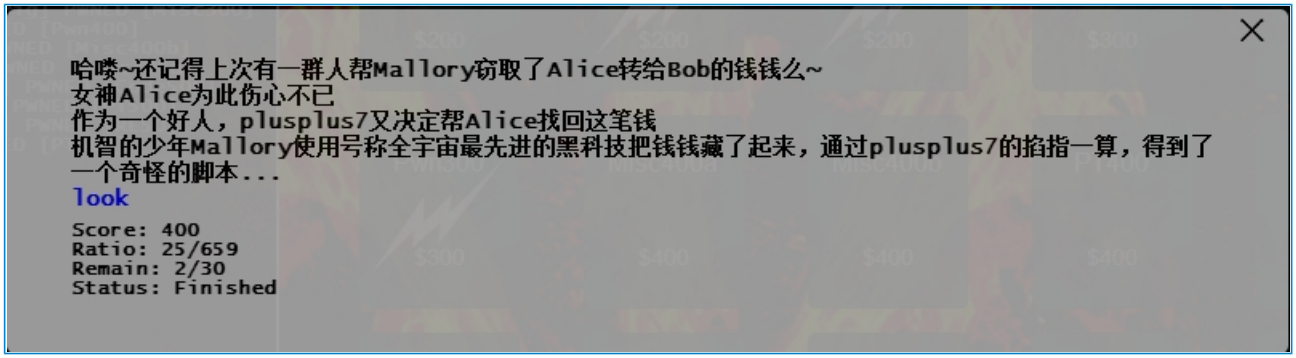
code200:



这个题目看着描述很复杂，实际上把数字转成二进制，输出二进制位上是 1 的位置序号就可以了。需要注意的是奇偶位是有符号的，如果最高位序号是奇数，需要在前面补一个偶数位。

编程的任务等待以后python练习的时候补吧。

code400:



```
import json
import hashlib
import os
import base64
from Crypto.Cipher import AES

fp = open("secret.json", "r")
secret = json.load(fp)
fp.close()

if type(secret["the answer to life the universe and everything"]) != type(u"77"):
    destroy_the_universe()

answer = hashlib.sha1(secret["the answer to life the universe and everything"]).hexdigest()[0:16]
key = hashlib.sha1(secret["Don't google what it is"]).digest()[0:6]

if ord(key[4])*(ord(key[5])-5) != 17557:
    destroy_the_universe()

keys = ["hey"+key[2]+"check"+key[3]+"it"+key[0]+"out",
        "come"+key[1]+"on"+key[4]+"baby"+key[5]+"~~!"]
answer = AES.new(keys[1], AES.MODE_ECB).encrypt(AES.new(keys[0], AES.MODE_ECB).encrypt(answer))

if base64.b64encode(answer) == "fm2knkCBHPuhCQHYE3spag==":
    fp = open("%s.txt" % hashlib.sha256(key).hexdigest(), "w")
    fp.write(secret["The entrance to the new world"])
    fp.close()
```

很明显这个题目是要爆破的。首先计算出k[4]和k[5],对17557做因式分解：97*181。

因而存在两组可能k[4]=97, k[5]=186;k[4]=102, k[5]=181。需要分别爆破。

这里有个可以优化的地方，就是在对应一组k[4]和k[5]的情况下keys[1]只有256种可能，可以先解出keys[1]对应的256个答案的解，也就是破解出最外面那一层AES，而后keys[0]对应的DES进行爆破的时候，结果只要查keys[1]对应的256长度的答案表就可以了。虽然算法复杂度依然是255*255*255*255，但最外面那一层没有计算AES，速度要快一些。

实际上直接暴力也可以。

```
import hashlib
import base64
from Crypto.Cipher import AES
k4=102
k5=181
kkk = hashlib.sha1("42").hexdigest()[0:16]
for k0 in range(255, 256):
    for k1 in range(0, 256):
        for k2 in range(0, 256):
            for k3 in range(0, 256):
                keys = ["hey"+chr(k2)+"check"+chr(k3)+"it"+chr(k0)+"out",
                        "come"+chr(k1)+"on"+chr(k4)+"baby"+chr(k5)+"~~!"]
                #print len(keys[0]), len(keys[1])
                answer = AES.new(keys[1], AES.MODE_ECB).encrypt(AES.new(keys[0], AES.MODE_ECB).encrypt(kkk)
                if base64.b64encode(answer) == "fm2knkCBHPuhCQHYE3spag==":
                    print "%s.txt" % hashlib.sha256(chr(k0)+chr(k1)+chr(k2)+chr(k3)+chr(k4)+chr(k5)).hexdig
                    exit();
```

经过测试，完成一次255*255*255的暴力，2分钟左右。服务器上要更快些，开到16个进程，很快可以直接爆出hash文件名：

http://download.sycsec.com/code/code400/5bd15779b922c19ef9a9ba2f112df1f2dbb0ad08bbf9edac27a28a0f3ba753f4.txt

没想到打开这个txt，里面还有一层加密：

```
Welcome to the new world!
Mallory把Alice给Bob的钱藏在了一个神秘的地方，还把地址加密，然后带着密钥，坐黄金之心离开了！
没有密钥的plusplus7用黑科技复原了一部分明文，然后打麻将去了，所以剩下的就交给你啦！
P.S.听说您很会暴力破解：）

====== Base64格式密文 ======
Or18/xSC2xW5pT7BLbIE7YPGLwWytbZsxupMp4w6iaa0QvtYZUMefkf43wmzR36MekHm23wgI4buIJLGk7m7gTq9fP8UgtsVuaU+wS2yBO2

====== 藏宝地址 ======
*****n**M****H***j***Wx*******d***********h*****3****=*******==******t**F**M**f***hM***********3***H*w**J
```

分析了一下，密文和部分破解的明文都是320个字节，以为是异或，但怎么尝试都不对。最后注意点放到明文的“==”上面，第一段连续的“=*******==”出现的时候，假设这里*都是=，那么实际上是一段连续的========，看明文，最后的“=*******=****”也应该是连续的======，那么把这两部分对应的密文进行比较，发现是一样的。这里实际上运气比较好，第一段连续的“=*******==”结束位置正好是64个字符处。这样可以发现，明文实际上是以64个字符分割的，列对齐的。处理后得到：

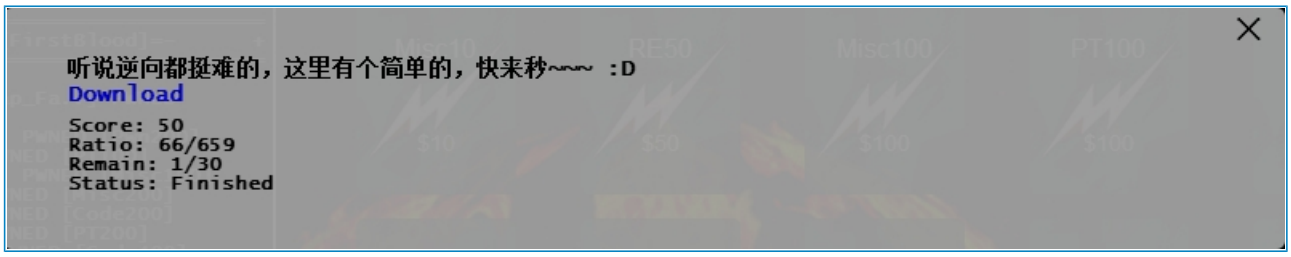
```
*****n**M****H***j***Wx*******d***********h*****3****=********==
******t**F**M**f***hM***********3***H*w**J********=**==********
U******E**95**V*c*N****5**t*M*****J*c*Q*****c*h5**0******==*==**
**NUR******************X2*u*H**Y***********G**P****==**********
*0*****************************f***5****OX*********=*******=****
```

很明显，每一列只有一个字符不是*，组合起来是一个base64字符串，解码后就是flag。

实际上这个题目还是比较人性化的，每一行后面都加了许多====。给解密提供了线索。

re50:

文件见附件。



文件是x86_64ELF，IDA查看，很简单。下面是等效代码。

```
int main()
{
    char s1[] = "Jr3gFud6";
    char s2[16];
    char passwd[16];
    int len;

    printf("input your password:");
    scanf("%s", passwd);
    len = strlen(passwd);
    if(len != 9)    #这里按照ida的字节抄过来。
        return 0;

    memset(s2, 0, 16);
    for(int i=0; i<len; i++)
        s2[i] = passwd[i]+3;

    if(! strcmp(s1, s2))
        printf("The flag is SCTF{%s}\n", passwd);
    return 0;
}
```

解码就是s1每个字符-3。

misc100:

此题可以学习到C++，异常处理throw--catch的反汇编分析。所以单独开帖写一下。

转载于:https://blog.51cto.com/cugou/1588334