JarvisOJ - Writeup（5.31更新）

转载

weixin_30767835 于 2018-05-31 21:37:00 发布 106 收藏
文章标签： 数据结构与算法 python
原文链接：http://www.cnblogs.com/L1B0/p/9119129.html

> **此篇用来记录我在jarivsOJ上的一些题解，只给解题思路，不放flag**

Misc

0x01 You Need Python(300)

> 题目有两个文件，一个py文件，另一个是经过编码的key

文件key

首先看到提示**key_is_here_but_do_you_know_rfc4042**，baidu一下了解到是**utf9编码**，于是将这个文件解码，得到一串字符（md里对下划线要转义才能正常显示，偷个懒所以我这里贴出的字符是**没转义过**的，请不要**直接**拿去进行下一步的操作以免**出错**）

```
_____*((//++_-_%___)**((%(-))+_______+(%+_____+_______%+_-(_//(_%_)))))+__*(((_________/)+_%+_____-
(_________//_____))**((_____+_____)+_______+_________%))+_____(((_________//+_______%)+(_____-_))**
((+_)+_______-(_______//)))+_____*((+_______-(_______//-_%%))**(_____+_____+_____))+__*(+_______-(//-
_________%_____%__))**(_________-____+_______)+(+____)**(_________%%+_+_)+(_-)*((____//____-
_____%____%)+_________)**(_____-(_______//_______+_________%)+_)+(_+
(_______%_______)*+)_________+_______(((_________%_______)_+_____-(_________//_________))_)+
(_____/)(((____-+_____)(_+_))**)+*((+_______-_)**_____)+___*(((+_-_/_+-_______%_____%__)*
(-+_____/+_______%_____))**__)+(//)*(((_________%%+_+_)%_____)+_______-_)**+__*
((_____/(_____%))+_)((_________%_______)+_+)+_//+_____+_________/___
```

对下划线计数，运算符正常运算可以得到**52870021310743315l3**，一开始以为这就是真正的key，然后死活解不出来，后来尝试ascii解码得到**I_4m-k3y**，这才是正确的key

文件flag.py

marshal.loads是反序列化操作，用uncompyle2即可解决。

脚本如下

```
#参考链接：https://www.aliyun.com/jiaocheng/475933.html

import uncompyle2
import marshal, zlib, base64

co = marshal.loads(zlib.decompress(base64.b64decode('eJxtVP9r21YQvyd/ieWm66Cd03QM1B8C3pggUuzYCSWstHSFQijy

f = open('test','w')
uncompyle2.uncompyle('2.7.3',co,f)
```

拿到源码之后就很好做了，运行源码可以知道我们需要输入**key**和**flag**，然后经过**encrypt函数**的加密和**cipherText**比较，而key我们已经拿到了，简单分析一下encrypt函数可以发现，连爆破都不用，直接赋值就好了。。

解题脚本如下

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
__Author__ = "LB@10.0.0.55"

import utf9
import hashlib
from libnum import n2s
from string import printable

def get_key():

    a = open('key','r').read()
    print type(a)
    key = utf9.utf9decode(a)
    print key

    exp = ''
    num = 0
    for i in key:
        if i == '_':
            num += 1
        elif num != 0:
            exp += str(num) + i
            num = 0
        else:
            exp += i
    exp += str(num)

    return n2s(eval(exp))

def sha1(string):
    return hashlib.sha1(string).hexdigest()


def calc(strSHA1):
    r = 0
    for i in strSHA1:
        r += int('0x%s' % i, 16)
    return r
```

```
def decrypt(key):
    print key
    keySHA1 = sha1(key)
    print keySHA1
    intSHA1 = calc(keySHA1)
    print intSHA1
    r = ''

    flag = '-185-147-211-221-164-217-188-169-205-174-211-225-191-234-148-199-198-253-175-157-222-135-240-
    flag = flag.split('-')[1:]
    flag = [ eval('-'+i) for i in flag ]
    print len(flag),flag

    for i in range(33):
        r += chr(-int('0x%s' % keySHA1[i % 40], 16)+intSHA1+flag[i])
        intSHA1 = calc(sha1(r[:i + 1])[:20] + sha1(str(intSHA1))[:20])
    print len(r),r

def main():

    key = get_key()
    decrypt(str(key))

main()
```

Crypto

Reverse

Pwn