[ACTF新生赛2020]crypto-des

原创

2er0!=O 于 2021-08-01 21:57:50 发布 93 收藏 1

分类专栏： Crypto wp des 文章标签： buu

本文链接：https://blog.csdn.net/m0_52727862/article/details/119304431


Crypto 同时被 3 个专栏收录

48 篇文章 1 订阅

订阅专栏


wp

54 篇文章 0 订阅

订阅专栏

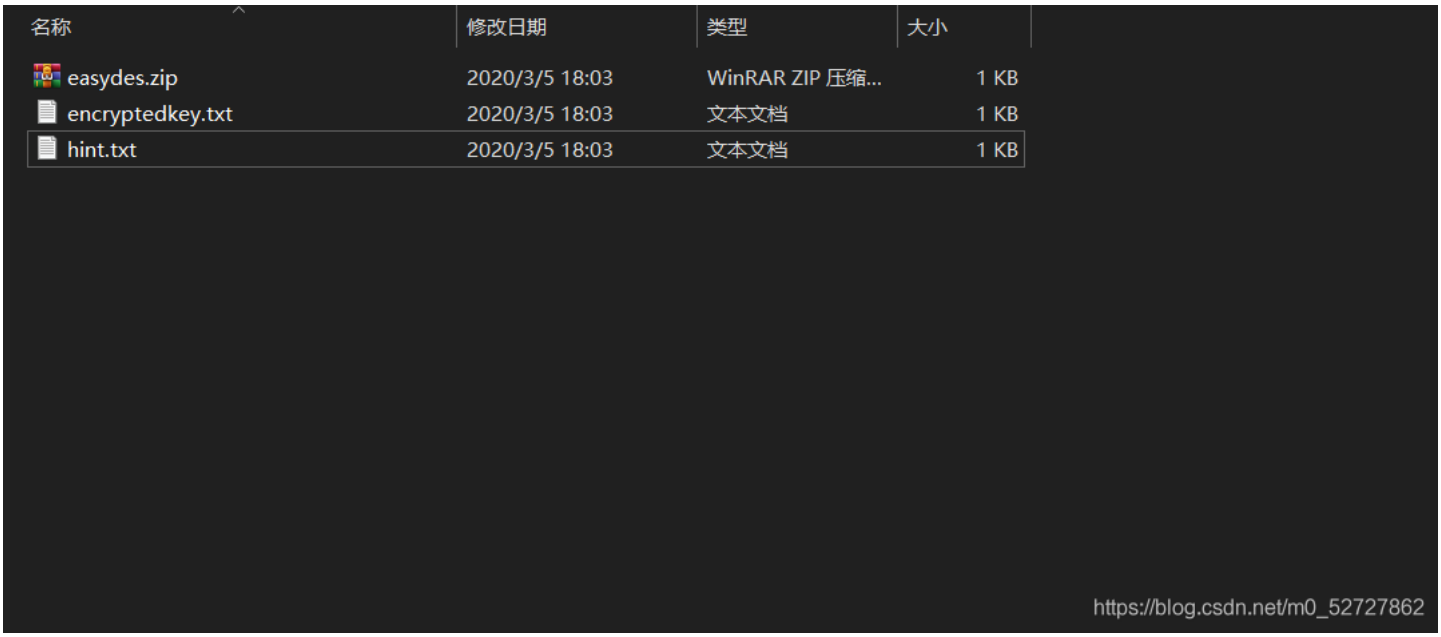

des

1 篇文章 0 订阅

订阅专栏

[ACTF新生赛2020]crypto-des

附件：



加密的 `easydes.zip` ，通过解密 `encryptedkey.txt` 以获取解密 zip 的 key

encryptedkey.txt:

```
721432389920416410000000.000000,
771353571780065040000000000000000.000000,
112586834561643540000000.000000,
67378029765916820000000.000000,
755534860921847030000000000000.000000,
4397611913739958700000.000000,
762093780286210390000000000000000.000000
```

hint.txt:

```
To solve the key, Maybe you know some interesting data format about C language?
```

压缩包尝试爆破并没有结果。。。。。

解密脚本：

```
import struct

s = [721432389920416410000000.000000,771353571780065040000000000000000.000000,112586834561643540000000.000000,673
78029765916820000000.000000,755534860921847030000000000000.000000,4397611913739958700000.000000,762093780286210390
00000000000000.000000]
a = ''
b = ''
for i in s:
    i = float(i)
    a += struct.pack('<f',i).hex()        #小端
print(a)

for j in s:
    i = float(i)
    b += struct.pack('>f',i).hex()        #小端
print(b)

# -*- coding: utf-8 -*-
import binascii
#16进制整数转ASCii编码字符串
a = 0x496e74657265737472696e67204964656120746f20656e6372797074
a1 = hex(a) #转换成相同的字符串即'0x665554'
a2 = a1[2:] #截取掉'0x'
a3 = binascii.a2b_hex(a2) #转换成ASCii编码的字符串
print(a3)

b = 0x747079727470797274707972747079727470797274707972747079727470797274707972
b1 = hex(b) #转换成相同的字符串即'0x665554'
b2 = b1[2:] #截取掉'0x'
b3 = binascii.a2b_hex(b2) #转换成ASCii编码的字符串
print(b3)
```

运行得到：

```
496e74657265737472696e67204964656120746f20656e6372797074
747079727470797274707972747079727470797274707972747079727470797274707972
b'Interestring Idea to encrypt'
b'tpyrtpyrtpyrtpyrtpyrtpyrtpyr'
```

解得key：

```
Interestring Idea to encrypt
```

解压得到加密脚本：

```
import pyDes
import base64
from FLAG import flag
deskey = "********"
DES = pyDes.des(deskey)
DES.setMode('ECB')
DES.Kn = [
   [1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0,
0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0],
   [1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 1,
0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0],
   [0, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0,
1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0],
   [1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0,
1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1],
   [0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0,
0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1],
   [0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 0,
0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0],
   [0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1,
0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0],
   [0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0,
0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0],
   [1, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1,
1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0],
   [0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0,
0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0],
   [0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1,
1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1],
   [0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1,
0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0],
   [1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1,
1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0],
   [1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1],
   [1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0,
1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1],
   [1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1,
0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1]
  ]
cipher_list = base64.b64encode(DES.encrypt(flag))
#b'vrkgBqeK7+h7mPyWujP8r5FqH5yyVlqv0CXudqoNHVAVdNO8ML4lM4zgez7weQXo'
```

加密文件是一个 DES 题目，加密的16轮子密钥都给出来了，直接base64解码后，decrypt()解密就行了。

```
import pyDes
import base64
from Crypto.Util.number  import*
deskey  = "********"
DES = pyDes.des(deskey)
DES.setMode('ECB')
DES.Kn = [
   [1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0,
0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0],
   [1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 1,
0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0],
   [0, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0,
1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0],
   [1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0,
1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1],
   [0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0,
0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1],
   [0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 0,
0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0],
   [0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1,
0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0],
   [0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0,
0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0],
   [1, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1,
1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0],
   [0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0,
0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0],
   [0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1,
1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1],
   [0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1,
0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0],
   [1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1,
1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0],
   [1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1],
   [1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0,
1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1],
   [1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1,
0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1]
  ]
# cipher_list = base64.b64encode(DES.encrypt(flag))

k = b'vrkgBqeK7+h7mPyWujP8r5FqH5yyVlqv0CXudqoNHVAVdNO8ML4lM4zgez7weQXo'
data = base64.b64decode(k)
# print(data)
flag = DES.decrypt(data)
print(flag)
```

运行得到：

```
b'actf{breaking_DES_is_just_a_small_piece_of_cake}'
```