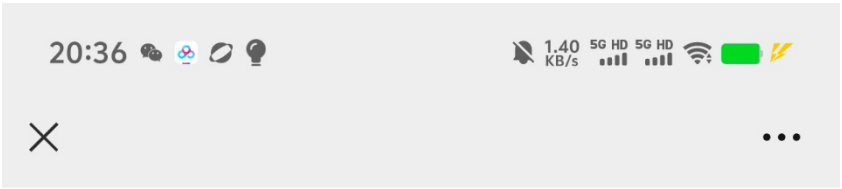
1.二维码的瘦身

用程序创建一个随机的版本号为 10 的二维码

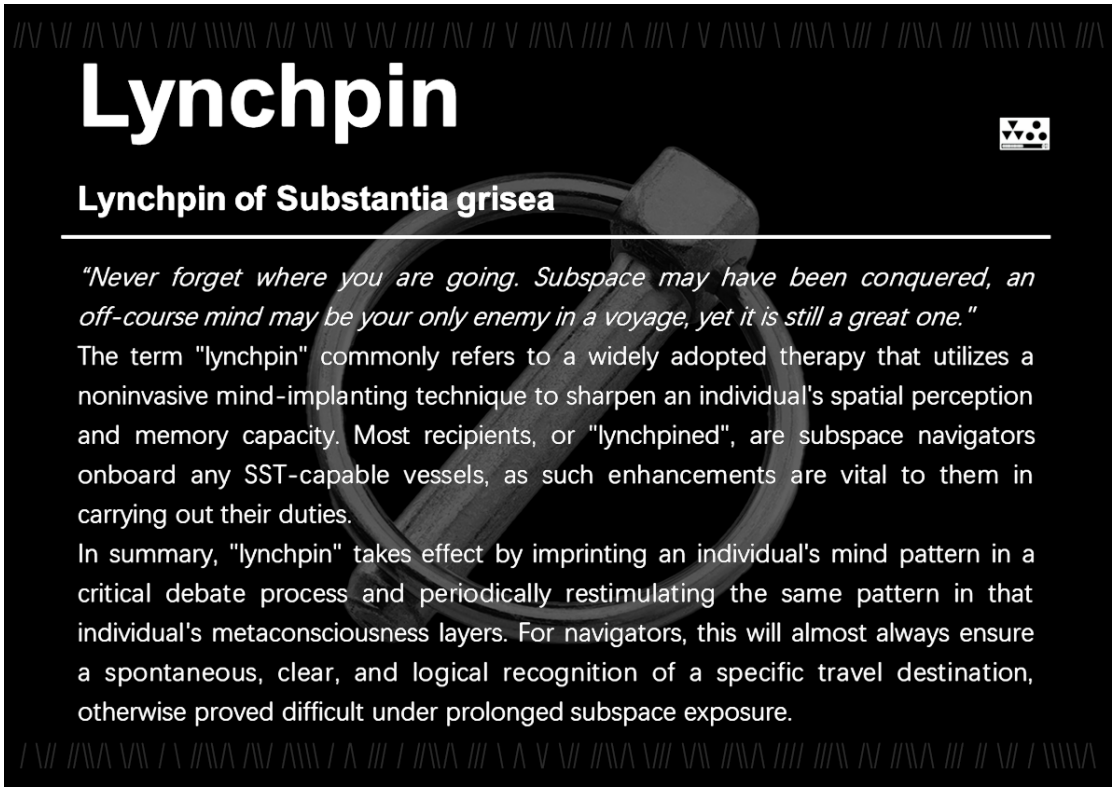
再 illustrator 中把题目给的对应位置覆盖上去

扫码即可



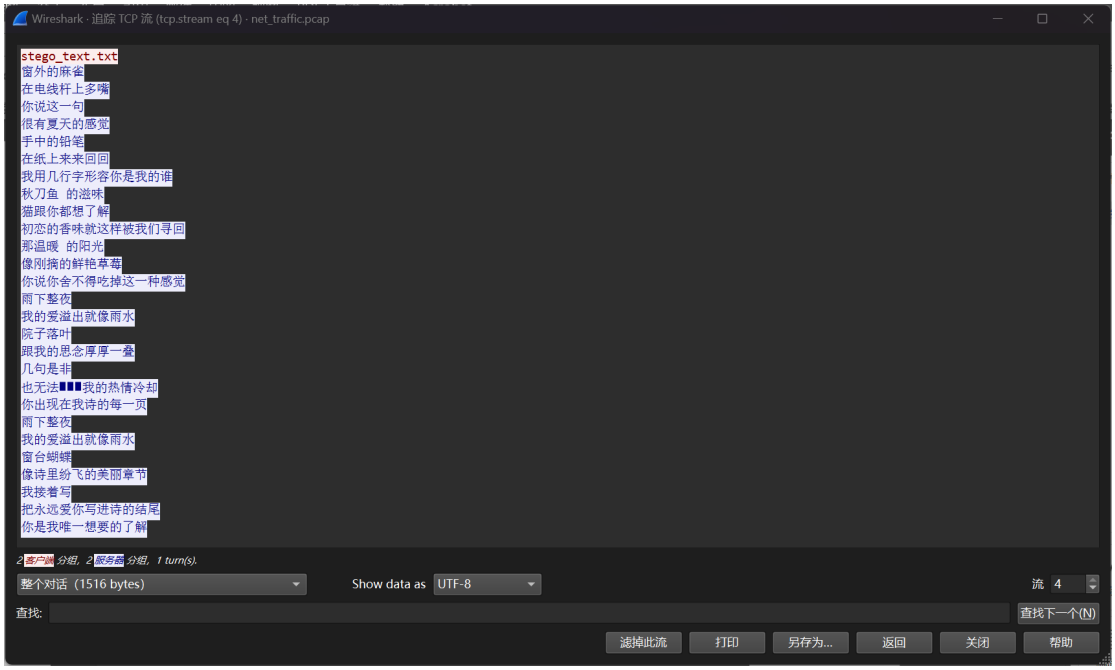
fductf{mGF95iiMjf1bdXyR8XXMz0CvHH3CaFBlYOQuy2LGpdTUE1mNfKsDwP0Fq4FrX0DzFDEshZ9Im9ke

2 什么是 cimbar 码
用 camerafilecopy（扫 cimbar 码的软件扫）
得到一张图片



上下对应的是莫斯密码
输到工具里翻译即可

3FDUKindergarten
进入 sh 再用 find 找到 flag 就行，实例现在打不开所以没有截图
4Net_traffic
把每个流都看一遍
发现 tcp.stream eq 4 中有七里香



从 stego 看出是一种隐写，看原始数据有一堆重复的没用字符，了解到是零宽字符隐写

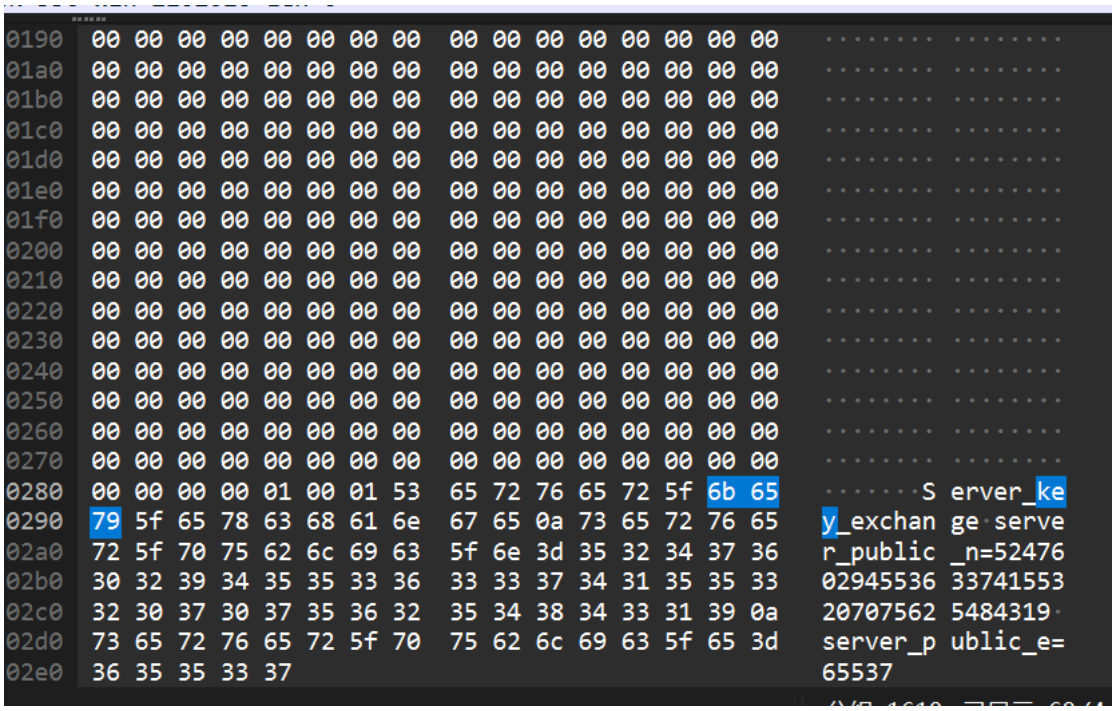
Unicode Steganography with Zero-Width Characters (330k.github.io)

5Enc_net_traffic

注意到是 tls v1.2

会有 key exchange 信息

直接用 wireshark 过滤器搜索“key”



追踪流

根据题目说法 使用 RSA 做 key 交换

先将 n 质因数分解,网上随便搜一个就行 factordb.com

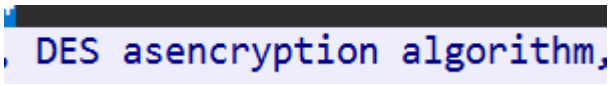
掏出 ai 写的 rsa 计算工具

得到预主密钥 16348233491274573437

和 client random 和 server random 计算出 master 3643242706250008610

```
74 6f 74 61 6c 3a 2a 81  53 70 a6 39 fe 00 2a 81   total:*. Sp.9..*
53 70 a6 39 fe 00 bb 27  08 36 1d 16 66 69 fb 72   Sp.9...' .6..fi.
14 ca d9 38 2b ac bc 56  ce 65 96 af bc 30 e3 17   ...8+..V .e...0.
fa 8c a6 e3 98 d6 e6 3e  24 9a 94 e5 b5 ad 2a 81   .......> $.....*
53 70 a6 39 fe 00 2a 81  53 70 a6 39 fe 00 fa d7   Sp.9..*. Sp.9...
8b d6 a2 2f b0 b1 cb 3b  09 a3 f0 5f a2 0c 37 77   .../...; ..._..7
34 a3 6c 17 22 55 2a 81  53 70 a6 39 fe 00 c2 15   4.l."U*. Sp.9...
08 c0 2a 91 15 93 4f f3  bd a5 e6 9e f8 7f f4 22   ..*...O. .......
6a 12 90 32 63 fc 17 11  ad 00 93 a4 3f 15 4f 78   j..2c... ....?.O
d4 f4 fa 88 49 ac 15 6c  e5 1b 7a ad 2c 4d f5 09   ....I..l ..z.,M.
8f b1 db 18 19 8c 2a 81  53 70 a6 39 fe 00 2a 81   ......*. Sp.9..*
53 70 a6 39 fe 00 41 f3  da 4f c7 69 4d 39 a1 de   Sp.9..A. .O.iM9.
6c db e7 a2 52 b7 b5 59  db e9 20 b6 99 36 2a 81   l...R..Y .. ..6*
```

可以看出加密的数据从 2a 开始



询问 ai 编写用 des 自动解密的工具

发现 ai 不靠谱

安装 `pycryptodome`

首先，您需要安装这个库。可以使用以下命令：

bash Copy Code

```
pip install pycryptodome
```

DES解密示例

以下是一个简单的示例，展示如何使用`pycryptodome`进行DES解密：

python Copy Code

```python
from Crypto.Cipher import DES
import binascii

# DES密钥（必须是8字节）
key = b'12345678'
# 要解密的密文（需要是8字节的倍数）
ciphertext = binascii.unhexlify('your_hexadecimal_ciphertext_here')

# 创建DES解密对象
des = DES.new(key, DES.MODE_ECB)

# 解密
plaintext = des.decrypt(ciphertext)

# 输出解密后明文
print("Decrypted plaintext:", plaintext.decode('utf-8').rstrip('\x00'))  # 去掉填充的空字符
```

只好自己查 Des 加密解法写了一个

from pyDes import *

data=[0x2a, 0x81,
0x53, 0x70, 0xa6, 0x39, 0xfe, 0x00, 0x2a, 0x81,
0x53, 0x70, 0xa6, 0x39, 0xfe, 0x00, 0xbb, 0x27,
0x08, 0x36, 0x1d, 0x16, 0x66, 0x69, 0xfb, 0x72,
0x14, 0xca, 0xd9, 0x38, 0x2b, 0xac, 0xbc, 0x56,
0xce, 0x65, 0x96, 0xaf, 0xbc, 0x30, 0xe3, 0x17,
0xfa, 0x8c, 0xa6, 0xe3, 0x98, 0xd6, 0xe6, 0x3e,
0x24, 0x9a, 0x94, 0xe5, 0xb5, 0xad, 0x2a, 0x81,

0x53, 0x70, 0xa6, 0x39, 0xfe, 0x00, 0x2a, 0x81,
0x53, 0x70, 0xa6, 0x39, 0xfe, 0x00, 0xfa, 0xd7,
0x8b, 0xd6, 0xa2, 0x2f, 0xb0, 0xb1, 0xcb, 0x3b,
0x09, 0xa3, 0xf0, 0x5f, 0xa2, 0x0c, 0x37, 0x77,
0x34, 0xa3, 0x6c, 0x17, 0x22, 0x55, 0x2a, 0x81,
0x53, 0x70, 0xa6, 0x39, 0xfe, 0x00, 0xc2, 0x15,
0x08, 0xc0, 0x2a, 0x91, 0x15, 0x93, 0x4f, 0xf3,
0xbd, 0xa5, 0xe6, 0x9e, 0xf8, 0x7f, 0xf4, 0x22,
0x6a, 0x12, 0x90, 0x32, 0x63, 0xfc, 0x17, 0x11,
0xad, 0x00, 0x93, 0xa4, 0x3f, 0x15, 0x4f, 0x78,
0xd4, 0xf4, 0xfa, 0x88, 0x49, 0xac, 0x15, 0x6c,
0xe5, 0x1b, 0x7a, 0xad, 0x2c, 0x4d, 0xf5, 0x09,
0x8f, 0xb1, 0xdb, 0x18, 0x19, 0x8c, 0x2a, 0x81,
0x53, 0x70, 0xa6, 0x39, 0xfe, 0x00, 0x2a, 0x81,
0x53, 0x70, 0xa6, 0x39, 0xfe, 0x00, 0x41, 0xf3,
0xda, 0x4f, 0xc7, 0x69, 0x4d, 0x39, 0xa1, 0xde,
0x6c, 0xdb, 0xe7, 0xa2, 0x52, 0xb7, 0xb5, 0x59,
0xdb, 0xe9, 0x20, 0xb6, 0x99, 0x36, 0x2a, 0x81,
0x53, 0x70, 0xa6, 0x39, 0xfe, 0x00, 0x2a, 0x81,
0x53, 0x70, 0xa6, 0x39, 0xfe, 0x00, 0x45, 0x99,
0x24, 0xa4, 0x10, 0xc8, 0xbf, 0xdd, 0xe7, 0xe1,
0x13, 0x57, 0xc0, 0x01, 0x52, 0x9a, 0xc0, 0x51,
0xf9, 0x6f, 0xeb, 0x68, 0xb1, 0xf5, 0xe9, 0xab,
0xff, 0x77, 0x91, 0xa2, 0x5a, 0x5d, 0x5b, 0x8c,
0xb0, 0x3f, 0xab, 0x25, 0xe4, 0xc7, 0x7f, 0xb6,
0x63, 0x5d, 0x9c, 0x40, 0x75, 0xbb, 0x90, 0xcb,
0x58, 0x8c, 0xee, 0x47, 0x86, 0x49, 0x8f, 0xdb,
0xe3, 0x78, 0x35, 0x5a, 0xd6, 0x4f, 0x62, 0x45,
0xf7, 0x8b, 0xec, 0x70, 0x1d, 0xa2, 0xe7, 0x27,
0xd5, 0xc3, 0xb0, 0xec, 0x45, 0xf9, 0x86, 0x89,
0x43, 0x8f, 0xf3, 0xee, 0x29, 0x53, 0x08, 0xb0,
0x7a, 0xe5, 0x83, 0x85, 0xe2, 0x2e, 0xf1, 0x6e,
0x26, 0x3c, 0x02, 0x36, 0xbd, 0x61, 0xc9, 0xfb,
0x74, 0xfc, 0x5d, 0x30, 0xd4, 0x9c, 0x84, 0x03,
0xf6, 0x2f, 0xa7, 0x96, 0x01, 0x5b, 0xf8, 0x98,
0x3d, 0x8d, 0xee, 0x61, 0x32, 0x48, 0x36, 0x07,
0x2a, 0x38, 0xb4, 0xac, 0xdb, 0x5d, 0x1d, 0x6b,
0xb1, 0xab, 0x8b, 0x1a, 0x3a, 0xf0, 0xa6, 0x0b,
0x9b, 0x5e, 0x50, 0xe1, 0x16, 0xed, 0x68, 0xea,
0xb7, 0x96, 0x61, 0x8a, 0xc9, 0x84, 0x9d, 0x12,
0x8c, 0x84, 0x78, 0x67, 0xc8, 0xcb, 0x19, 0x95,
0x8c, 0x98, 0x25, 0xa3, 0xe2, 0x6c, 0x91, 0x0a,
0x87, 0x2b, 0x1d, 0x96, 0x62, 0xb4, 0x96, 0x56,
0x25, 0xa8, 0x05, 0x5c, 0xcb, 0x6e, 0x4e, 0xd1,

0xee, 0x8d, 0xcb, 0x16, 0x3b, 0xce, 0x3b, 0x41,
0x4a, 0xe9, 0xf1, 0xd4, 0x36, 0x6a, 0xbf, 0x26,
0x78, 0x2c, 0xe1, 0xfc, 0x83, 0x88, 0x6c, 0x2a,
0xcd, 0x56, 0x95, 0x32, 0xad, 0x51, 0xe9, 0x26,
0x0b, 0xa1, 0xfd, 0x28, 0x67, 0xd5, 0x81, 0x27,
0x5e, 0xe2, 0x1d, 0xc2, 0xf6, 0xf0, 0x55, 0xc3,
0x9d, 0x4c, 0x10, 0x84, 0xc8, 0x4b, 0x24, 0xfc,
0x87, 0xc1, 0x48, 0x9a, 0x57, 0xe7, 0xe7, 0x58,
0x6a, 0xe1, 0x0a, 0x4b, 0x81, 0xb0, 0xd9, 0x1b,
0xb2, 0x66, 0x15, 0xa8, 0x34, 0x23, 0xc2, 0x7b,
0x14, 0x71, 0xec, 0x06, 0x27, 0xf8, 0x27, 0xab,
0x39, 0x0e, 0xf3, 0x83, 0x67, 0xa8, 0x6a, 0x4c,
0x1d, 0x91, 0x0e, 0x48, 0x8d, 0xcf, 0xe9, 0x73,
0x26, 0xd9, 0x40, 0x68, 0xb5, 0x72, 0xce, 0x79,
0x04, 0xa9, 0x50, 0x57, 0x3e, 0x63, 0x35, 0x78,
0xf7, 0x1c, 0xd3, 0x27, 0x9b, 0xe4, 0x32, 0xdf,
0x08, 0xd0, 0x84, 0x8f, 0xb2, 0x83, 0x14, 0xa0,
0x3d, 0xc0, 0x41, 0xc6, 0x81, 0x2d, 0x87, 0xc7,
0x42, 0xe7, 0xca, 0xba, 0x30, 0x91, 0xf0, 0x07,
0x5f, 0x27, 0x7a, 0x79, 0xa4, 0x00, 0x8a, 0xb4,
0xea, 0xe5, 0xba, 0x75, 0x0f, 0x1d, 0x02, 0x22,
0x37, 0x0d, 0x9a, 0x03, 0x66, 0xdc, 0xd0, 0x5f,
0x62, 0xe5, 0x99, 0x34, 0x11, 0x7b, 0x79, 0x59,
0x91, 0xdd, 0x74, 0xe4, 0x6a, 0xe3, 0xa9, 0xbc,
0x70, 0x4e, 0x31, 0x70, 0xc2, 0x7b, 0x64, 0xd1,
0x8b, 0xc0, 0xb5, 0xcd, 0x80, 0x65, 0x34, 0x04,
0x51, 0x25, 0xb2, 0x82, 0x04, 0x0b, 0xbf, 0x8e,
0xd2, 0x11, 0x69, 0x2f, 0x6f, 0x71, 0xda, 0x38,
0xd5, 0x16, 0x45, 0x8d, 0x89, 0x96, 0x33, 0x0b,
0x7b, 0xab, 0x26, 0x33, 0x9f, 0x49, 0x30, 0x4b,
0x67, 0x70, 0x9d, 0xe4, 0xf4, 0xbb, 0x12, 0x4f,
0xc0, 0x56, 0x3d, 0xfd, 0xac, 0x78, 0xb1, 0x03,
0xf1, 0x41, 0x4d, 0x11, 0xbc, 0x05, 0xbd, 0x1d,
0xd4, 0x63, 0x4f, 0x9e, 0x29, 0xc9, 0x9c, 0xa2,
0x7e, 0xc2, 0xce, 0x99, 0xde, 0xa6, 0xd1, 0xd9,
0x89, 0x64, 0x08, 0xf0, 0xfd, 0x68, 0x28, 0xe4,
0xc5, 0x86, 0xcc, 0x2c, 0x90, 0x97, 0x56, 0x73,
0x27, 0x14, 0x87, 0xae, 0xf1, 0x42, 0x1f, 0x77,
0x45, 0xc8, 0x6a, 0x23, 0x82, 0xbc, 0x9a, 0x67,
0x38, 0x9c, 0x2f, 0xe7, 0x9a, 0x76, 0x80, 0x84,
0xf9, 0x53, 0xb9, 0x56, 0xc2, 0x34, 0xeb, 0x72,
0x2c, 0xb0, 0x58, 0x0c, 0x13, 0xb5, 0x7d, 0xb4,
0x1c, 0xd7, 0xc8, 0x0d, 0x94, 0xae, 0x3a, 0xc8,
0x72, 0x5f, 0x34, 0x5c, 0xc6, 0x8b, 0x82, 0x4f,

0x1b, 0x81, 0x5b, 0x4d, 0x3b, 0x66, 0xaa, 0x07,
0xdf, 0xdc, 0x52, 0x79, 0x59, 0x38, 0x99, 0x4f,
0x62, 0x61, 0x4a, 0x5b, 0xe6, 0x6c, 0x4e, 0xd0,
0xb4, 0xc5, 0x95, 0xb0, 0x7e, 0xff, 0x40, 0x2c,
0x5a, 0x4d, 0x69, 0x78, 0xbb, 0xf5, 0xc3, 0x34,
0x13, 0x2e, 0xad, 0x8d, 0xae, 0xb7, 0x6b, 0x4b,
0xd6, 0x6c, 0xb1, 0xed, 0xc2, 0xa5, 0xd8, 0xe2,
0x31, 0x5e, 0x03, 0x3b, 0xaa, 0x82, 0x4f, 0x0d,
0xf6, 0x25, 0x64, 0x32, 0x9b, 0x46, 0xa4, 0xa8,
0x78, 0xec, 0x91, 0x2a, 0x52, 0xb2, 0xf0, 0x0d,
0xc6, 0x92, 0x63, 0x4e, 0x2b, 0x85, 0xd1, 0x8c,
0x48, 0x65, 0x27, 0xb3, 0x21, 0x14, 0xb4, 0x47,
0xca, 0xbb, 0x65, 0x92, 0x08, 0xe6, 0xd0, 0xd8,
0xc0, 0x40, 0xc1, 0xf5, 0x84, 0x32, 0x5d, 0x0d,
0x97, 0x74, 0x4d, 0x1f, 0x94, 0x78, 0x63, 0x45,
0x5e, 0xb9, 0xba, 0xb2, 0x11, 0x65, 0x47, 0x98,
0x86, 0xcd, 0xf2, 0x9e, 0xc3, 0x3c, 0xf8, 0x2f,
0xf3, 0x77, 0x41, 0x20, 0x94, 0x5a, 0xd7, 0xd4,
0x39, 0xe0, 0x17, 0x5b, 0xfc, 0xbc, 0xd6, 0x80,
0xdc, 0xc3, 0x73, 0x2c, 0x76, 0x79, 0xf0, 0x5f,
0x1e, 0x10, 0xc3, 0xe0, 0x24, 0x91, 0xc3, 0x9f,
0x0a, 0x66, 0x74, 0x90, 0x49, 0xd7, 0xbe, 0xcc,
0x19, 0x7f, 0x52, 0xdc, 0x8a, 0x15, 0xc9, 0x02,
0xd1, 0x00, 0x7a, 0xe7, 0xf6, 0x57, 0x9f, 0x56,
0xa3, 0x0c, 0x6c, 0xc4, 0x51, 0xc8, 0xe7, 0xfd,
0x2b, 0x69, 0xfa, 0xae, 0x13, 0x83, 0xa7, 0x6e,
0x85, 0x4d, 0xf6, 0xf9, 0x9d, 0x2a, 0xd4, 0x3a,
0x88, 0x5d, 0x3b, 0x75, 0x1e, 0x15, 0x9c, 0x73,
0x74, 0x96, 0xeb, 0x1a, 0x82, 0x98, 0x27, 0x52,
0x43, 0xf3, 0x7b, 0xf7, 0xd2, 0x4a, 0x4f, 0x1b,
0xba, 0x3a, 0x45, 0x3d, 0x62, 0x85, 0xe7, 0xe1,
0x13, 0x57, 0xc0, 0x01, 0x52, 0x9a, 0xc4, 0xf6,
0xe6, 0x8b, 0x64, 0x52, 0xb7, 0x98, 0xd9, 0x24,
0x80, 0x3c, 0xf1, 0xef, 0xa4, 0xcb, 0x65, 0x52,
0xff, 0x5e, 0xb1, 0x11, 0xf8, 0x5b, 0x88, 0x7c,
0x9f, 0xf7, 0x50, 0x9c, 0x66, 0xad, 0x0c, 0x73,
0x2c, 0xf0, 0xc5, 0x3e, 0x2f, 0xb1, 0xff, 0x52,
0xec, 0x25, 0x42, 0x40, 0xeb, 0xcf, 0x99, 0xb2,
0x5e, 0x3c, 0x17, 0x5f, 0xaa, 0x1d, 0x9a, 0x2a,
0x7a, 0x2d, 0x7c, 0x5c, 0xe4, 0xf8, 0xe8, 0x95,
0x63, 0x31, 0xa5, 0x89, 0x8a, 0x8d, 0x09, 0xea,
0xa4, 0x97, 0x4a, 0xea, 0xf5, 0x49, 0x4b, 0x5d,
0x6f, 0x20, 0xc7, 0x59, 0xdf, 0x41, 0x89, 0x3a,
0x7d, 0xb8, 0x65, 0x9d, 0xfc, 0x9f, 0x84, 0x3e,

0xcf, 0x89, 0xa5, 0x44, 0xbe, 0x93, 0xe5, 0x33,
0x92, 0x9e, 0x97, 0x75, 0x57, 0xde, 0x93, 0x57,
0x1b, 0xc1, 0xd5, 0x9e, 0xa2, 0xf9]
master=3643242706250008610 从 master 的 16 进制长度可以看出是普通的 des
master=master.to_bytes(8,"little")
k=des(master,des,pad=None,padmode=PAD_PKCS5)
decrypted=k.decrypt(data)
print(decrypted)
IV 值若不提供 会从 master 中截取,从 pydes 的源代码中可以看出

ALICE'S ADVENTURES IN WONDERLAND\r\n\r\n Lewis Carroll\r\n\r\n THE MILLENNIUM FULCRUM EDITION 3.0\r\n\r\n\r\n\r\n\r\n\r\n CHAPTER I\r\n\r\n Down the Rabbit-Hole\r\n\r\n\r\n Alice was beginning to get very tired of sitting by her sister\r\non the bank, and of having nothing to do: once or twice she had\r\npeeped into the book her sister was reading, but it had no\r\npictures or conversations in it, `and what is the use of a book,'\r\nthought Alice `without pictures or conversation?'\r\n\r\nflag{423101ef-cc3d-462f-ab92-f1035a96ce28}\r\n So she was considering in her own mind (as well as she could,\r\nfor the hot day made her feel very sleepy and stupid), whether\r\nthe pleasure of making a daisy-chain would be worth the trouble\r\nof getting up and picking the daisies, when suddenly a White\r\nRabbit with pink eyes ran close by her.\r\n\r\n There was nothing so VERY remarkable in that; nor did Alice\r\nthink it so VERY much out of the way to hear the Rabbit say to\r\nitself, `Oh dear! Oh dear! I shall be late!' (when she thought\r\nit o\xc6\x16#_\x14\xf6\xaf\x17>;[\x04&\xdc\xda\xbf\xb9\xde\xea\xf7kt\xa4\xd0J\r\xab/\xf8l\xc5\x0e"

flag{423101ef-cc3d-462f-ab92-f1035a96ce28}

6Alice 与 Bob 的小纸条

字母替换的密码

编写程序翻译

```
def replace_letters(input_string):
    #
    mapping = {
        'a':'x','b': 'a', 'c': 'e', 'd': 'c', 'e': 'l', 'f': 'w','g':'j','h':'q',
        'i': 'r', 'j':'p','k': 'b', 'l': 'h', 'm': 'd', 'n': 'm',
        'o': 'k','p':'s', 'q': 'o', 'r': 'f', 's': 'y', 't': 'i',
        'u': 's','v':'g','w': 't', 'x': 'n','y':'v', 'z': 'u'
    }#字母要自己尝试

    # 替换字母并用 '_' 代替其余字母
    # 替换字母并用 '_' 代替其余字母
    result = ""
    for char in input_string:
```

```
            lower_char = char.lower()  # 将字符转换为小写以进行映射
            if lower_char in mapping:
                # 根据原字符的大小写，选择替换后的字符的大小写
                if char.isupper():
                    result += mapping[lower_char].upper()
                else:
                    result += mapping[lower_char]
            elif char in [' ', ',']:  # 保留空格和逗号
                result += char
            else:
                result += '_'  # 用 '_' 代替其余字母

        return result

# 读取输入文件和写入输出文件
input_file_path = "C:/Users/30104/Desktop/新建 文本文档 (4).txt"
output_file_path = 'output.txt'  # 输出文件路径

# 读取文件内容
with open(input_file_path, 'r', encoding='utf-8') as file:
    content = file.read()

# 处理内容
result = replace_letters(content)

# 写入输出文件
with open(output_file_path, 'w', encoding='utf-8') as file:
    file.write(result)

print("处理完成，结果已写入到", output_file_path)
```

Father of suspect in Georgia school shooting arrested__The father of a __yearold boy accused of killing four people at a high school in the US state of Georgia has been arrested__Colin Gray, __, is facing four charges of involuntary manslaughter, two counts of seconddegree murder and eight of cruelty to children, said the Georgia Bureau of Investigation GBI__GBI Director Chris Hosey said on Thursday evening the charges were directly connected to his sons actions and allowing him to possess a weapon__The son, Colt Gray, is accused of killing two teachers and two students in Wednesdays shooting at Apalachee High School in Winder, near Atlanta__He is due in court on Friday charged as an adult with four counts of murder__Authorities are investigating whether Colin Gray bought the ARstyle weapon as a gift for his son in December ___, law enforcement sources told CBS News, the BBCs US partner__In May ___, the FBI alerted local police to online threats about a school shooting, associated with an email address linked to the suspect__A sheriffs deputy went to interview the boy, who was __ at the time__His father told police he had guns in the house, but his son did not have unsupervised access to

them, the FBI said in a statement on Wednesday__Officials say the threats were made on Discord, a social media platform popular with video gamers, and contained images of guns__The accounts profile name was in Russian and translated to the surname of the attacker who killed __ people at Sandy Hook Elementary School in Connecticut in _____A police incident report describing last years interview with the boy and his father was released on Thursday__In the report, a deputy described the boy as reserved and calm and said he assured me he never made any threats to shoot up any school__They said he claimed to have deleted his Discord account because it was repeatedly hacked__Colin Gray also told police his son was getting picked on at school and had been struggling with his parents separation__Police records reveal that the boys mother and father were in the process of divorcing, and he was staying with his father during the split__The teen often hunted with his father, who told police he had photographed his son with a deers blood on his cheeks__The boys maternal grandfather told the New York Times he partly blames the tumultuous home life after Mr Grays split from his daughter___I understand my grandson did a horrendous thing theres no question about it, and hes going to pay the price for it, Charlie Polhamus told the newspaper__My grandson did what he did because of the environment that he lived in, he added__During the news conference on Thursday, Barrow County Sheriff Jud Smith said all nine of those injured were expected to make a full recovery__Several victims had already left hospital, he said__The flag begins with fductf_ Then comes the left brace_ Contents inside two braces are WrodFerqeuncy, which indicates you should analyse word frequency to solve this problem_ And dont forget the right brace_ That is, fductf, left brace, WrodFerqeuncy, and right brace__Students Mason Schermerhorn and Christian Angulo, both __, and teachers Richard Aspinwall, __, and Christina Irimie, __, died in the attack__Witnesses said the suspect left an algebra lesson on Wednesday morning only to return later and try to reenter the classroom__Some students went to open the locked door, but apparently saw the weapon and backed away__Witnesses said they then heard a barrage of ____ gunshots_ Two school police officers quickly challenged the boy and he immediately surrendered__These are not the first charges against the parents of a suspect in a school shooting__In April, the parents of a Michigan teenager who killed four students with a gun they bought for him just days before the shooting were sentenced for their role in the attack__James and Jennifer Crumbley were both found guilty of manslaughter and each sentenced to __ to __ years in prison__The case was widely reported to be the first time the parents of a child who had carried out a mass shooting were held criminally liable__

红字为 flag

7Jeff Dean 笑话

直接对给出的 n 分解

75260681471021614550282490981109532540064768189519963003788853449628148238491094583307996632151408412567215312977037223220187737015088671579053426702046623459412581502446523522085346430389150310811736933663200863622291

得到 pq

```
from Crypto.Util.number import long_to_bytes
import gmpy2
n
=752606814710216145502824909811095325400647681895199630037888534496281
482384910945833079966321514084125672153129770372232201877370150886715790
534267020466234594125815024465235220853464303891503108117369336632008636
2291
p=8193423899118349
q=918549832129642631217357209306156789342787361268206093941796483383789
484542708798597295010966934663904150064134172470988102552209051513911732
043909262052289347076966590217809538667560583127015378375359
c=256864680473574148724330122982850798107083352523479960192526887872058007
105600473949115369178375312035118863980916595994907094041490961553985080
322065358054909582906998184723825819020580338318407731878539345535582513
7
```

```
e=65537
phin =(p-1)*(q-1)
print(gmpy2.invert(e, phin))
x=gmpy2.invert(e, phin)
t=c
print(t)
print(gmpy2.powmod(t,x,n))
flag=long_to_bytes(gmpy2.powmod(t,x,n))
print(flag)
flag1=str(flag,'gbk')
print(flag1)
```

```
b'fductf{small_prime_factor_in_rsa_is_dangerous_F270BA33AA791B45}'
fductf{small_prime_factor_in_rsa_is_dangerous_F270BA33AA791B45}
```

8est-your-nc

nc 一下实例

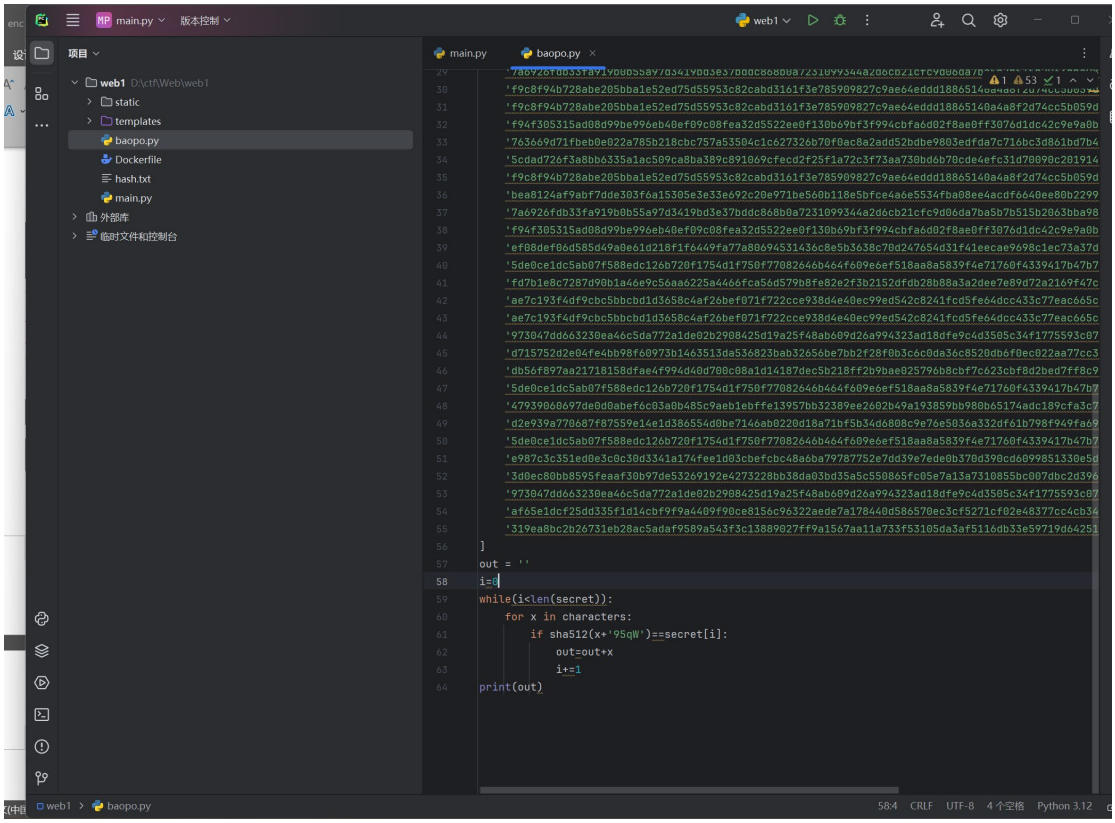
9 草率的毕业设计

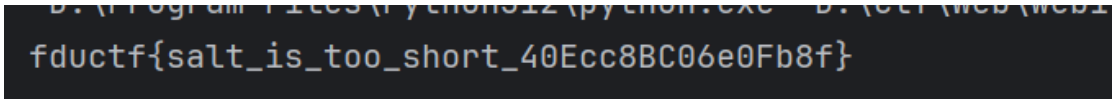
```
D:\ctf\Crypto\tool\hashcat-6.2.6>hashcat.exe -a 3 -o out -m 1700 fd7b1e8c7287d90b1a46e9c56aa6225a4466fca56d579b8fe82e2f3
b2152dfdb28b88a3a2dee7e89d72a2169f47cec1c7124d273bea22f6abb854d3fe36980e2 ?a?a?a?a
```

```
af65e1dcf25dd335f1d14cbf9f9a4409f90ce8156c96322aede7a178440d586570ec3cf5271cf02e48377cc4cb341c2760e996960ce314849a29e86ec8ad87cd:f95qW
5de0ce1dc5ab07f588edc126b720f1754d1f750f77082646b464f609e6ef518aa8a5839f4e71760f4339417b47b7ca3fae96b59c79a0ce4a95d488bf7af47c3a:095qW
fd7b1e8c7287d90b1a46e9c56aa6225a4466fca56d579b8fe82e2f3b2152dfdb28b88a3a2dee7e89d72a2169f47cec1c7124d273bea22f6abb854d3fe36980e2:E95qW
```

每次的盐都是一样的

用程序挨个爆破第一个字母





10 Baby64 用 IDA 打开

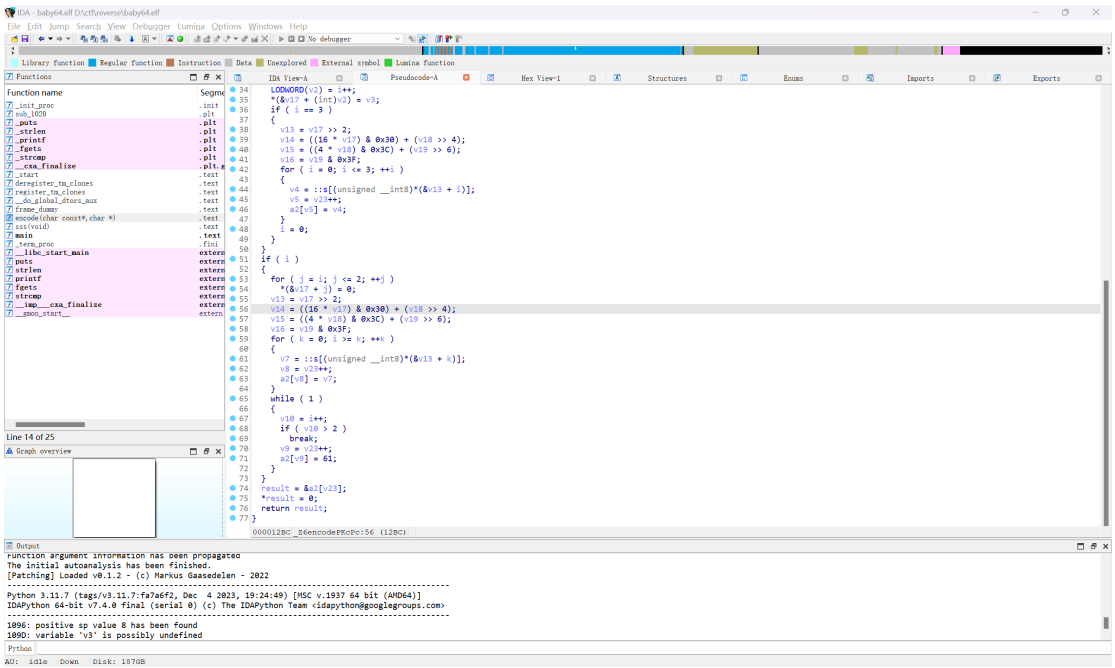
```c
int __fastcall main(int argc, const char **argv, const char **envp)
{
  char s1[8]; // [rsp+0h] [rbp-60h] BYREF
  __int64 v5; // [rsp+8h] [rbp-58h]
  __int64 v6; // [rsp+10h] [rbp-50h]
  __int64 v7; // [rsp+18h] [rbp-48h]
  __int64 v8; // [rsp+20h] [rbp-40h]
  char s[8]; // [rsp+30h] [rbp-30h] BYREF
  __int64 v10; // [rsp+38h] [rbp-28h]
  __int64 v11; // [rsp+40h] [rbp-20h]
  __int64 v12; // [rsp+48h] [rbp-18h]
  __int64 v13; // [rsp+50h] [rbp-10h]

  *(_QWORD *)s = 0LL;
  v10 = 0LL;
  v11 = 0LL;
  v12 = 0LL;
  v13 = 0LL;
  *(_QWORD *)s1 = 0LL;
  v5 = 0LL;
  v6 = 0LL;
  v7 = 0LL;
  v8 = 0LL;
  printf("Enter your flag: ");
  fgets(s, 40, _bss_start);
  encode(s, s1);
  if ( !strcmp(s1, k) )
    puts("Correct!");
  else
    puts("Incorrect!");
  return 0;
}
```

main 函数没提到什么

只是说用 encode 编码了输入的 s

Encode



我直接丢给 ai

解释到这是一个 base64 编码的程序

查看 strings

```
ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/
YnQ0Z2Qnf111bX4eNWMeaiA1W2QpYU8nbWJycE9xanU8Dh==
```

然而直接把下面的东西复制到 cyberchef 是一堆乱码

于是看到

```
1  __int64 sss(void)
2  {
3    __int64 result; // rax
4    char v1; // [rsp+1h] [rbp-5h]
5    int i; // [rsp+2h] [rbp-4h]
6
7    for ( i = 0; i <= 62; i += 2 )
8    {
9      v1 = s[i];
10     s[i] = s[i + 1];
11     result = i + 1;
12     s[result] = v1;
13   }
14   return result;
15 }
```

他显然是把一个字符串奇数和偶数调换

则有可能是

```
YnQ0Z2Qnf111bX4eNWMeaiA1W2QpYU8nbWJycE9xanU8Dh==
```

调换或者

字符集调换

经尝试为字符集调换

编写程序翻译

```
#include <iostream>
#include <string>
#include <vector>

static const std::string original_base64_table =
    "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/";
// 原始 Base64 字符表

// 调换奇数和偶数位置
std::string swap_positions(const std::string &input) {
```

```
    std::string swapped = input;
    for (size_t i = 0; i + 1 < swapped.size(); i += 2) {
        std::swap(swapped[i], swapped[i + 1]);
    }
    return swapped;
}

// 生成自定义 Base64 字符表
std::string create_custom_base64_table() {
    return swap_positions(original_base64_table);
}

// 反向查找表
std::vector<int> create_base64_decode_map(const std::string &table) {
    std::vector<int> map(256, -1);
    for (size_t i = 0; i < table.size(); ++i) {
        map[static_cast<unsigned char>(table[i])] = i;
    }
    return map;
}

// Base64 解码函数
size_t base64_decode(const std::string &input, std::vector<unsigned char> &output) {
    std::string custom_base64_table = create_custom_base64_table();
    std::vector<int> decode_map = create_base64_decode_map(custom_base64_table);
    size_t output_len = 0;

    for (size_t i = 0; i < input.size(); i += 4) {
        int a = decode_map[static_cast<unsigned char>(input[i])];
        int b = decode_map[static_cast<unsigned char>(input[i + 1])];
        int c = decode_map[static_cast<unsigned char>(input[i + 2])];
        int d = decode_map[static_cast<unsigned char>(input[i + 3])];

        if (input[i + 2] == '=') {
            output.push_back((a << 2) | (b >> 4));
            break;
        } else if (input[i + 3] == '=') {
            output.push_back((a << 2) | (b >> 4));
            output.push_back((b << 4) | (c >> 2));
            break;
        } else {
            output.push_back((a << 2) | (b >> 4));
            output.push_back((b << 4) | (c >> 2));
            output.push_back((c << 6) | d);
```

```
            }
            output_len += 3;
        }

        return output_len;
}

int main() {
        std::string base64_input;

        // 输入需要解码的 Base64 字符串
        std::cout << "Enter the Base64 string: ";
        std::getline(std::cin, base64_input);

        std::vector<unsigned char> decoded_output;
        size_t decoded_length = base64_decode(base64_input, decoded_output);

        // 输出解码结果
        std::cout << "Decoded string: ";
        for (size_t i = 0; i < decoded_length; ++i) {
            std::cout << decoded_output[i];
        }
        std::cout << std::endl;

        return 0;
}
```

```
Enter the Base64 string:YnQ0Z2Qnf111bX4eNWMeaiA1W2QpYU8nbWJycE9xanU8Dh==
 Decoded string: fductf{M4in_1s_n0t_the_fir3t_0ne}
```

11Easypython

给出的文件是一个 pyc 程序 linux file 命令即可看出

用 pycdc 反编译

```
def chall():
    key = b'FDUCTF{2024}'
    flag = input('Please input the flag: ').encode()
    chcek_data = bytes.fromhex('f9ecf8f97b8f96baecc607004ec26724623cee86ece84a4581f8c063')
    s = mm(key)
    encrypted_data = nn(s, flag)
    if encrypted_data == chcek_data:
        print('Congratulations! You got the flag!')
        return None
    None('Sorry, the flag is not correct!')

if __name__ == '__main__':
    chall()
    return None
```

没有完全编译成功

但是通过 ai 可以看出使用的加密算法

代码解释

1. KSA (Key Scheduling Algorithm):
   - 在 `mm` 函数中，首先初始化一个状态数组 `s`，然后通过循环根据给定的 `key` 生成一个伪随机排列。
   - `j` 是一个索引，用于在 `s` 中进行交换。
2. PRGA (Pseudo-Random Generation Algorithm):
   - 在 `nn` 函数中，生成与输入数据等长的伪随机字节流，然后使用 XOR 操作对原始数据进行加密。
3. **挑战函数**:
   - 在 `chall` 函数中，获取用户输入并与预定义的加密数据进行比较，根据结果输出相应信息。

KSA 和 PRGA 都是可逆的 直接让 ai 写出解密代码就行

```python
def mm(key):
    N = 256
    s = list(range(N))
    j = 0
    key_length = len(key)
    for i in range(N):
        j = (j + s[i] + key[i % key_length]) % N
        s[i], s[j] = s[j], s[i]
    return s


1 个用法
def nn(s, data):
    n = len(data)
    keystream = []
    i = j = 0

    for _ in range(n):
        i = (i + 1) % 256
        j = (j + s[i]) % 256
        s[i], s[j] = s[j], s[i]
        keystream.append(s[(s[i] + s[j]) % 256])

    encrypted_data = bytes([data[k] ^ keystream[k] for k in range(n)])
    return encrypted_data
```

解密函数如上图

```
1 个用法
def decrypt(flag, key):
    s = mm(key)
    decrypted_data = nn(s, flag)
    return decrypted_data


1 个用法
```

得到 flag

```
"D:\Program Files\Python312\python.exe" D:\ctf\r
Decrypted flag: fductf{Y0ur4r3g00d@tPyth0n!}
```