Challenge 1

Rishi Vardhan Majji
24B0969

August 13, 2025

1 Description

The one-time pad, as you have learnt, is perfectly secure. However, its perfect secrecy depends on the key being used exactly once. If the same key is reused, the resulting "two-time pad" is no longer secure as XORing the two ciphertexts gives you the XOR of the plaintexts, which is a lot of information about the two messages. In fact, if the messages are in natural language, it may even allow you to extract both the messages fully given their XOR. In this challenge, you'll do exactly that. ciphertext1.enc and ciphertext2.enc contain two one-time pad encrypted ciphertexts en crypted with the same key. One of them is a "flag" used in a Capture The Flag competition (hint: it starts with cs409) and the other one is a normal English sentence. Use the redundancy of the English language to figure out the entire flag. Feel free to look at encrypt.py to understand how to use the strxor function to XOR two byte objects in python.

2 Approach

As using XOR operation on the ciphertexts gives us the result of XOR operation on the original Flag and Message, using XOR on this and the part **"cs409{"** (elongated to match the length of message, we can get the first 6 letters of the message.Which would be **"Crypta"**.

Now we can anticipate what the "Crypta" thing could actually be, and its **Cryptanalysis**. Using this method,we continuously predict the flag and message by their initial parts and get it.

```
with open("ciphertext1.enc", 'rb') as f:
    c1=f.read()
```

```
with open("ciphertext2.enc", 'rb') as f:
    c2=f.read()

c = strxor(c1,c2)
```

Get the XOR of the ciphertexts which is the same as that of the flag and message, call it **c**

```
text1=b'cs409{'
text2=b''

if len(text1)< len(c):
    text1=text1.ljust(len(c))
else :
    text1=text1[:len(c)]

if len(text2)< len(c):
    text2=text2.ljust(len(c))
else :
    text2=text2[:len(c)]
```

Start with the texts you know initially and extend them to the length of the ciphertexts.

```
c3=strxor(text1,c)
c4=strxor(text2,c)
print(c3)
print(c4)
```

We are going to XOR text 1 with c (which is XOR of text1 and text2) and the same with text2. So we should get text2 as c3 and text1 as c4.We then print these and analyse what could come next in text 1 and 2 simultaneously.

The process looks somewhat like this

```
b'Crypta'
b'cs409{one_time'
b'Cryptanalysis freq'
b'cs409{one_time_pad_key_re'
b'Cryptanalysis frequently invo'
b'cs409{one_time_pad_key_reuse_compr'
b'Cryptanalysis frequently involves statistical att'
b'cs409{one_time_pad_key_reuse_compromises_security!!!'
b'Cryptanalysis frequently involves statistical attacks'
```

We slowly reach

```
text1=b'cs409{one_time_pad_key_reuse_compromises_security!!!}'
text2=b'Cryptanalysis frequently involves statistical attacks'
```