mboxor

For this challenge, I inferred from the name that this is a multi byte xor challenge. To complete this challenge, I figured out the most likely key length by splitting up the encrypted text into n-length chunks. Now, for adjacent chunks, I XOR'ed the bytes at corresponding indices with each other (XOR'ed byte at index 0 of chunk 0 with byte at index 0 of chunk 1, etc…). If I XOR these bytes, then the resulting number will have 1's corresponding to the unmatching bits between these two bytes. If I add these unmatching bits and divide by number of bytes compared, then I should get an average number of unmatching bits per byte. Due to the way multi-byte XOR works, if my key length is correct, then the average number of unmatching bits per byte will be lowest when I have the correct key length. I created a python script (mbxor.py) to determine the key length of the key. It looked correct because 5 and multiples of 5 gave me the lowest number of unmatching bits per byte (with 5 being the most likely key length according to my python script). I verified my finding with XOR tool, and ran the following command: xortool -x ciphertext2.txt. xortool also determined that the most likely key length was 5 bytes. After this, I brute-forced all possible values for a 5 byte key (using xortool because it was tedious to code it in python) and determined which output gave me the flag. I used the following command in xortool to brute force all possible values of a 5 byte key and generate its output: xortool -x -b ciphertext2.txt. Then I used the python script find.py to find the file that had the string "flag" somewhere in it.