Symmetric Encryption

In this lab, you will be writing your own cryptographic library to decrypt a substitution cipher, and using existing cryptographic libraries to experiment with a symmetric encryption called AES and a classic encryption named ceaser cipher.

1.1 Substitution Cipher (5 points)

Files
1. *sub_key.txt*: key
2. *sub_ciphertext.txt*: ciphertext

sub_key.txt contains a permutation of the 26 upper-case letters that represents the key for a substitution cipher. Using this key, the **i**th letter in the alphabet in the plaintext has been replaced by the ith letter in sub_key.txt to produce ciphertext in *sub_ciphertext.txt*. For example, if the first three letters in your sub_key.txt are ZDF..., then all As in the plaintext have become Zs in the ciphertext, all Bs have become Ds, and all Cs have become Fs. The plaintext we encrypted has only upper-case letters, numbers and spaces. Numbers and spaces in the plaintext were not encrypted. They appear exactly as they did in the plaintext.

Submit the plaintext, which is obtained using the key *sub_key.txt* to decrypt *sub_ciphertext.txt*, in the file ***solution01.txt***

Cryptographic Library

For the following tasks, we recommend PyCrypto, an open-source crypto library for python. PyCrypto can be installed using pip with *sudo pip install pycrypto* or by going to their website at https://www.dlitz.net/software/pycrypto/.

1.2 AES encryption (5 points)

Files
1. *aes_key.hex*: key
2. *aes_iv.hex*: initialization vector
3. *aes_ciphertext*.hex: ciphertext

aes_key.hex contains a 256-bit AES key represented as an ascii string of hexadecimal values. *aes_iv.hex* contains a 128-bit Initialization Vector in a similar representation. We encrypted a sentence using AES in CBC mode with this key and IV and wrote the resulting ciphertext (also stored in hexadecimal) in *aes_ciphertext*.hex.

Decrypt the ciphertext using the provided information and submit the plaintext in ***solution02.txt**.*

1.3 AES: Breaking A Weak AES Key (5 points)

Files

1. *aes_weak_ciphertext.hex*: ciphertext

As with the last task, we encrypted a sentence using 256-bit AES in CBC and stored the result in hexadecimal in the file *aes_weak_ciphertext.hex*. For this task, though, we haven't supplied the key. All we'll tell you about the key is that it is 256 bits long and its 251 most significant (leftmost) bits are all 0's. The initialization vector was set to all 0s. First, find all plaintexts in the given key space. Then, you will review the plaintexts to find the correct plaintext and the corresponding key.

Find the key of the appropriate plaintext and submit it as a hex string in **Solution03.hex**.

1.4 Breaking Caeser cipher (10)

Write a program in python than will decipher a Caeser cipher without knowing the key. Your program must be able to find the key and the correct plain text without any human intervention.