Network Security Assignment

Assignment_1b

Encrypt_MTP Algorithm:

Input in the algorithm:

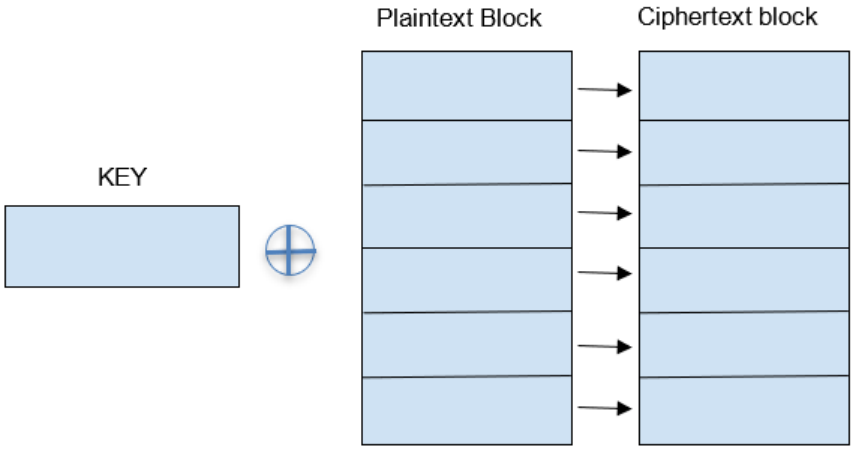
- Plain text file in .txt format

Output of the algorithm:

- Hexadecimal format file

Algorithm:

- Plain text file taken as *input.txt* and split into sentences from each full stop ``. '' and stores it in the list_sentences.
- Now max length of sentence in the sentences length is calculated.
- A Key of length max_length is generated randomly which is composed of the alphabet. Also, the generated key is written back in the .txt file for decryption_mtp uses.
- Convert the key in Bitvector i,e Key_bitVEctor
- Now for each sentence in the list_sentences.
    - sentence converted in Sentence_BitVector
    - XOR the Sentence_BitVector and Key_bitVEctor and convert into hex_string format and write back to "*output_file.txt".*



Decrypt_MTP Algorithm:

Input in the algorithm:

- encrypted file from *output.txt* which is in hex format.
- Key is also taken input from a file which is stored during encryption in the *mtp_key.txt* file.

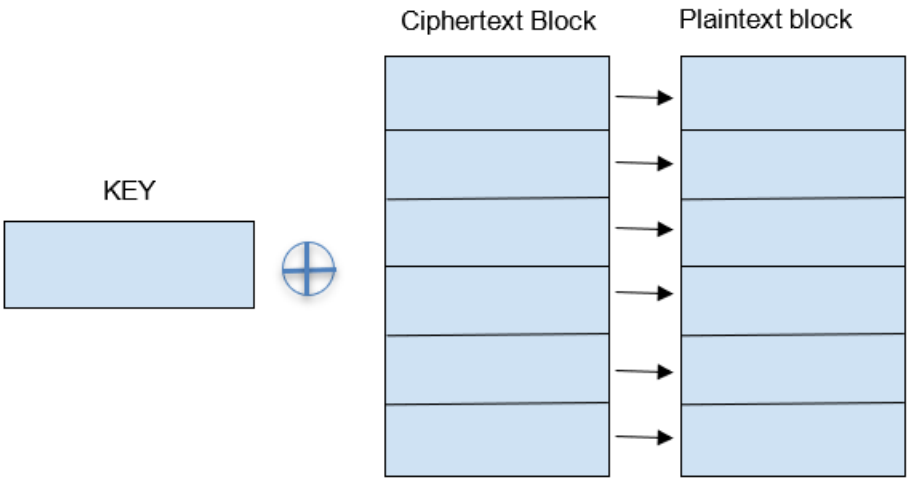
Output of the algorithm:

- Finally Decrypted text is stored in the *Decrypted_output_file.txt* file.

Algorithm:

- Key is read from file
- Also, the encrypted file from *output.txt* is read line by line:
    - At each line of encrypted file of input, it finds length of line
    - Key is taken the length equal the length of individual line of encrypted line
    - And now the encrypted line is XOR with the KEY and converted back to Ascii text and stored to the output file name "`Decrypted_output_file.txt".`

- Finally Decrypted file is generated.



Crack_MTP Algorithm:

Input in the algorithm:

- encrypted file from *output.txt* which is in hexadecimal format.

Output of the algorithm:

- Finally Decrypted text is stored in the *Decrypted_output_file.txt* file.

Algorithm:

- Encrypted file reads from the file name 'cipher.txt'
- Convert the encrypted file from hexadecimal to binary representation
- Create an empty Plaintext is initialized with "_". Which is use later to writing the decrypting text alphabet
- Frequency analysis is performed for getting the finally decrypted file:
    - It XOR at one line of column to their next line if the result of the XOR is any alphabet then definitely one of them is **blank space (ASCII code 32)**.
    - To Find out which one of them is **blank space (ASCII code 32)** , perform two separate frequency corresponding to that on each of the line in **ciphertexts** file:
        - Take the ciphertext value of the first line at that column and perform XORing operation corresponding to remaining each line at that column position and count as count1
        - Do the similar steps for the second line column value ciphertext as count2 and compare it.
        - In this way we get one of them which has greater count value.
        - Now take the line corresponding to greater count value and take that corresponding column ciphertext value and XOR it with space bitvalue (i,e `0b100000`) we get the key value for the column value position.
        - Once we get the Key at that column value, position XOR it with all the lines of ciphertext that value from only the particular column value ciphertext and store it in the ***Plaintext.***
    - Repeat this for all the columns and at last we get Plaintext for the all-key alphabet which we find out and remaining are replaced with "_".
- Finally, Plaintext is written back to the "`recovered.txt`"