Network Security

Programming Assignment #1

Problem #1 (Breaking of a Classical Cipher)

The python script (**encrypt.py**) provided along with this assignment implements an encryption algorithm (cipher), that takes a plaintext message file as input and produces an encrypted file as the output. As a sample to demonstrate the working of the cipher, a **plaintext.txt** and its corresponding **ciphertext.txt** is also provided. The key that has been used to produce this sample ciphertext is **bits@f463.** As part of this problem of the assignment you are supposed to do the following tasks.

1. Explain the encryption algorithm implemented by the python script **encrypt_classical.py**.
2. Write the corresponding decryption script which will take output generated by the encryption script and generate the same plaintext as the output that was given to the encryption script as the input, considering both the encryption and decryption scripts are run with the same key.
3. You are supposed to submit a CRACK code (**crack_classical.py**) which when supplied with a ciphertex generated by the cipher should produce the corresponding plaintext. Note that, producing complete plaintext may not be feasible. So, even if your code is able to generate the partially correct plaintext, it is fine. In one sense, we are performing a ciphertext only attack to get the plaintext. It would be good if you use python language to write your code. However, you can convert the python script provided herewith into a C program and then write the CRACK in C.

Problem #2 (Multi Time Pad is not Secure)

In class, we discussed that OTP has perfect security but when the same key used multiple times it could lead to an insecure cipher.

1. Write the encryption (encrypt_mtp.py) and decryption (decrypt_mtp.py) and a key generation (key_mtp.py) program for MTP cipher. Assume the plaintext is encoded as ASCII characters.
2. Develop a CRACK code (CRACK.py) which when supplied with a number of ciphertext (say 10) generated by the cipher using the same key, should produce the corresponding plaintext for all the ciphers. Note that, producing complete plaintext may not be feasible. So, even if your code is able to generate the partially correct plaintext, it is fine. In one sense, we are performing a ciphertext only attack to get the plaintext. It would be good if you use python language to write your code. However, you can convert the python script provided herewith into a C program and then write the CRACK in C.

Submission Details

1. Submit a zipped file with the name ***studentID_assignment1.zip*** through Nalanda
2. The zipped file should contain two different directories namely **Assignment_1a and Assignment_1b.** Submit the problem 1 and 2 in the **Assignment_1a and Assignment_1b** directories respectively.
3. The **Assignment_1a** directory should contain **report_1a.pdf**, **decrypt_classical.py**, **crack_classical.py** for the tasks 1, 2 and 3 of the problem #1. The **crack_classical.py** file submitted by you should take the ciphertext as the input from the file with name "**ciphertext.txt**". The output (plaintext) generated by your code should be written in the file with name "**recoveredtext.txt**".
4. The **Assignment_1b** directory should contain **report_1b.pdf, encrypt_mtp.py, decrypt_mtp.py, key_mtp.py** for the task 1 of problem 2 and **crack_mtp.py** for the task 2 of the problem 2. The **crack_mtp.py** file submitted by you should take the ciphertext as the input from the file with name "**ciphertext.txt**". The output (plaintext) generated by your code should be written in the file with name "**recoveredtext.txt**".

* **BORROWING or COPYING the source code will lead to heavy punishment.**