**Exercise 2 – Mta Crypto**

**Objective**

Practice synchronization between consumer and producer threads in the same process.

**Overview**

Write a multithreaded application that illustrates a password hacking using brute force technique. There should be 1 thread that generates a randomized plain password, encrypts it using a randomized key and “sends” the encrypted buffer to X decrypter threads, the decrypter threads compete to decrypt the encrypted password by trying to guess the key.

**Detailed description**

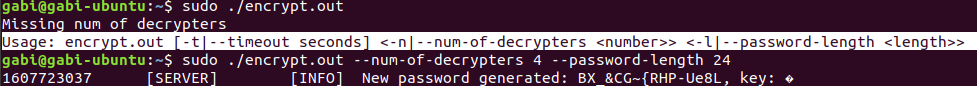
**General**

Following flags should be supported:

*-n|--num-of-decrypters* – will determine how many decrypter threads will be created

*-l|--password-length* – number of characters that will be encrypted, the more characters will be encrypted the harder it will be for the decrypters

*-t|--timeout(optional)* – time is seconds until server regenerates a password if it didn’t receive correctly decrypted password



**Encrypter(Server)**

Create a single thread that will use global data structures (to your choice) to generate a random printable\* password in the length specified by the user, encrypts it using a random and secret key and send the encrypted password (without the key) to the decrypters, the encrypter thread will then wait for 1 of 2 things:

1) A decrypted password received from one of the decrypters

2) Timeout (configured by the user) expires

If the password is correct or the time out expired a new password should be generated, encrypted and sent to the decrypters.

If an incorrect password received, it should be dropped and encrypter thread shall continue waiting for 1 of the 2 things specified above.

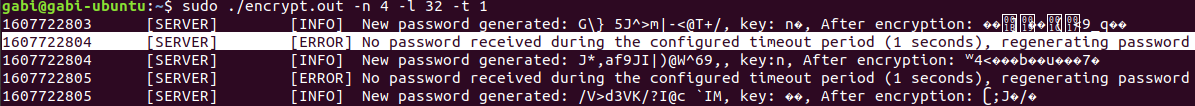
**Decrypter(Client)**

Decrypter threads will compete between themselves, once receiving an encrypted password from the encrypter, they should immediately start trying to decrypt it by generating random keys, on every random key the decrypter shall decrypt the encrypted password and if it is printable send it to the encrypter.

Upon newly received password the decrypter shall **immediately** (at most one spare iteration allowed) discard the old password and start working on the new received password.



With –t option:



**Notes**

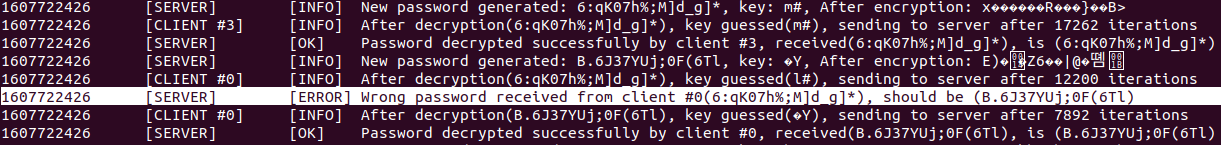
\* Printable: everything that isprint() (man 3 isprint) accepts as printable

\* Password length must be multiplication of 8 and key length must be 1/8 from the password length

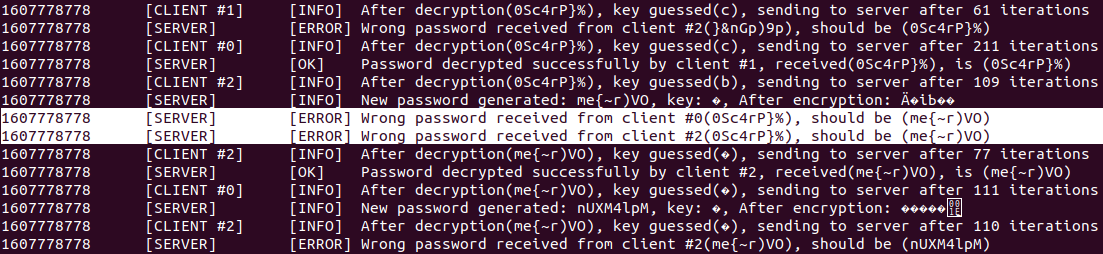
\* Make sure to see that CPU utilizes 100%

\* Make sure to handle failure scenarios such as:

Wrong guessed password:



Out of order passwords:



\* For encrypting the data and generating random data, use the provided utility libraries (<https://github.com/gavrielk/LinuxCourseCodePub/blob/master/mta_crypt_lib/>), install it using the following command (Ubuntu 18 or 20 is assumed):

$ sudo dpkg --install mta-utils-dev.deb