Jordan University of Science & Technology
Department of Network Engineering and Security
NES416- Network Programming
Programming Assignment 4

Due Date: see course website

Description:

You are required to implement a server program that responds to both TCP and UDP requests from the corresponding clients. The service provided by the server is simply a substitute cipher/cryptosystem. The cipher replaces one character to another based on some key. The input message (plaintext) can be of any size but composed only of small-letter (**abcdefghijklmnopqrstuvwxyz**). The output (ciphertext) has the same size and composed also of small-letter characters. The key is limited to 7 characters (also small-letter). You need to treat each digit as a number according to its position in the alphabet. For example, a=0, b=1, …,z=26. To encrypt a message, you add a plaintext character with the corresponding key character (mod 26). The key might be repeated to match the length of the plaintext message. In the decryption, you subtract the corresponding key character from the ciphertext character (mod 26). Again the key might be repeated as necessary. For example, if the key is "nesdept" the process goes as follows:

Plaintext: hellostudents
Key: nesdept**nesdept**
Ciphertext: uidoshmhhwqxh

In the decryption, the reverse is performed so you get the original massage.

The server should wait for requests on the same port for both TCP and UDP requests. The TCP part MUST be concurrent. However, the UDP part must be iterative. Furthermore, you are requested to use host and service names in you code, rather that the IP and port numbers. That is, you need to edit some configuration files (see lecture recordings) and add necessary lines to map between your server local IP address and some hostname, and also to map between your server port number and some service name. For this assignment, the server hostname is ***nes416_crypto_server*** and the service name is ***nes416_port***.

The client part (TCP or UDP), will display the following menu to the user to select from:

1. Encrypt a message
2. Decrypt a message
3. Exit
4. Listing files in the current directory on the server

If the user selects 1, the **encryption** must be implemented by the **client**. The client asks the user to enter a plaintext message (NULL terminated) and a key. Then it encrypts the message and sends it to the server. The server, which implements the decryption, reads the encrypted message, asks the user to enter the key, then sends the plaintext message to the client. Then, the client prints the received message on standard output together with the IP address and port number of the server.

If the user selects 2, the **decryption** must be implemented by the **client**. The client asks the user to enter a ciphertext message (NULL terminated) and a key. Then it decrypts the message and sends it to the server. The server, which implements the encryption in this case, reads the decrypted message, asks the user to enter the key, then sends the corresponding ciphertext message to the client. Then, the client prints the received message on standard output together with the IP address and port number of the server.

The connection between your client and server should stay open, so that the client can repeat the operation again. You should also add a logout or stop type of a command so that the client can tell the server that it is finished and the session should end. For example, if the user enters --, the client and server program terminates. Note that the server displays the IP address and port number of each client request

Output Sample:

For example, running the code should produce something similar to this output:

```
Client>
Enter a msg: hellostudents
Enter a key= nesdept
Sent Encrypted msg: uidoshmhhwqxh
Received plaintext msg: hellostudents from the server(print the
IP address and port number of the server)
Enter a msg: --
Exiting...thank you

Server> waiting for client's request
Received ciphertext uidoshmhhwqxh from client:(print the IP
address and port number of the client)
Enter Key: nesdept
Sent hellostudents to the client
Waiting for clients request
Received -- from client:(print the IP address and port number of
the client) Exiting…
```

Hints:

- ☐ Ask questions as early as possible.
- ☐ DO NOT use the header file "unp.h" from the book
- ☐ Submit your source code for both the client and the server, and some running sample of your code as one zipped file whose name is your student ID number
- ☐ Your programs should be compiled and run without any single error or warning.
- ☐ Comment and **error-check** you code
- ☐ NO CHEATING/COPYING is allowed
- ☐ Your program for the client needs to take two arguments that specify the **hostname** and the **service name** of the server. Your program for server needs to take an argument that specifies the **service name** to work on
- ☐ Don't use the bind() function on the clients
- ☐ Test your code by running the two clients simultaneously.
- ☐ Don't leave Zombies behind.
- ☐ **Make sure to handle interrupted system calls in your code**