**Network Security**

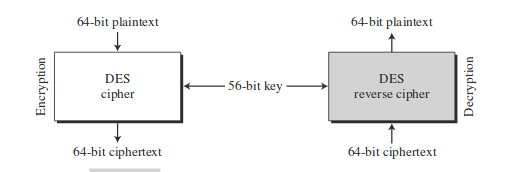
**Assignment -2-**

Hala Ali

Roll No:187556

at first : we should depend on the openssl library to implement DES Algorithm , because this library includes the required file ”des.h” and “rand.h”

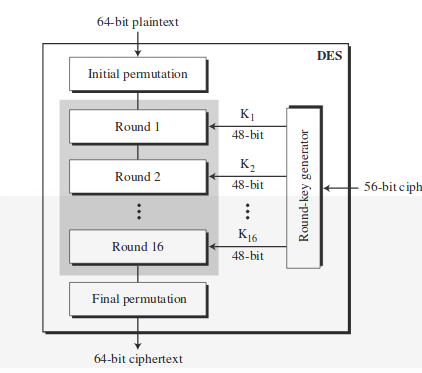
as shown in the following figure :At the encryption site, DES takes a 64-bit plaintext and creates a 64-bit ciphertext; at the decryption site, DES takes a 64-bit ciphertext and creates a 64-bit block of plaintext. The same 56-bit cipher key is used for both encryption and decryption.

Key length is 56 bits because each 8 bits the least signification bit is used as parity bit so the resulted key length is 56 bits

encryption process is made of

two permutations (P-boxes), which we call initial and final permutations, and sixteen rounds.

Each round uses a different 48-bit round key generated from the cipher key according to predefined algorithm



Key Generation

The round-key generator creates sixteen 48-bit keys out of a 56-bit cipher key.

However, the cipher key is normally given as a 64-bit key in which 8 extra bits are the parity bits, which are dropped before the actual key-generation process

the implementation code

**at client side**

we should included the required files in openssl library

#include <openssl/des.h>

#include <openssl/rand.h>

then initialization of seed using a fixed predefined series

DES\_cblock seed = {0xFE, 0xDC, 0xBA, 0x98, 0x76, 0x54, 0x32, 0x10};

By using a specific "seed" value, the same key will be generated on each run of the code (assuming a deterministic [P](https://en.wikipedia.org/wiki/PRNG)RNG is used).

//The key schedule is an expanded form of the key; it is used to speed the encryption process

DES\_key\_schedule keysched;

RAND\_seed (const void \*buf, int num); mixes the num bytes at buf into the PRNG state, here num=entropy that is an estimate of how much randomness is contained in buf

RAND\_seed(seed, sizeof(DES\_cblock));

// DES\_string\_to\_key: Convert a string to a DES key. Use something like PKCS5\_PBKDF2\_HMAC\_SHA1() to create key from passwords.

DES\_string\_to\_key("hala",&key);

DES\_set\_key:Setup a des key schedule from a key

DES\_set\_key((C\_Block \*)key, &keysched);

After creation the key we will encrypt the data before sending at client side and decrypt data after receiving using the function DES\_ecb\_encrypt function

note: Encrypt/decrypt a block using DE, here using ECB mode

Parameters:

the first argument :input data to encrypt

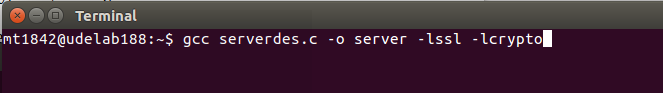
the second argumentoutput: data after encryption

the third argumentoutput: key schedule to use

the forth argumentoutput:encp argument if non zero, encrypt. if zero, decrypt.

Actually the same procedure will use at the server side because the same key must be used by both client and server(symmetric key) .

To compile c code with open ssl



Execution

