LAB ASSIGNMENT - 6

1. Blowfish Algorithm

Follow the given algorithm below, implement a code for Encryption and Decryption using Python. This algorithm belongs to Efficient Data Hiding in Audio.

Following to Instructions -

- This algorithm is a 64-bit block cipher with a variable length key.
- Requires less memory.
- Sequence Spread Spectrum (SSS) is one such method that spreads the signal encryption and decryption.
- It requires a 32 bit microprocessor at a rate of one byte for every 26 clock cycles.
- A variable length key block cipher up to 448 bits.
- Blowfish contains 16 rounds. Each round consists of XOR operation and function. Each round consists of key expansion and data encryption.
- Key expansion generally used for generating initial contents of one array and data encryption uses a 16 round feistel network.
- 64 bit Plain text is taken and divided into two 32 bits data and at each round the given key is expanded & stored in 18 p-array and gives a 32 bit key as input and XOR ed with previous round data.
- The function F its own functionality is to divide a 32-bit input into four bytes and use those as indices into an S-array.
- The lookup results are then added and XOR ed together to produce the output. ● At the 16th round there is no function .The output of this algorithm should be 64 bit cipher text.

ALGORITHM STEPS:

Divide X into two 32-bit halves XL and XR
For i=1 to 16:
XL = XL Å Pi
XR = F (XL) Å XR
Swap XL and XR
End for
Swap XL and XR (Undo the last swap.)
XR = XR Å P17
XL = XL Å P1
Recombine XL and

Output X (64-bit data block: cipher text)

For decryption, the same process is applied, except that the sub-keys Pi must be supplied in reverse order. The nature of the Feistel network ensures that every half is swapped for the next round (except, here, for the last two sub-keys P17 and P18)

2. Write an algorithm for RC4, RC5, RC6 and implement a code through Python.

3. IDEAAlgorithm

3.1 . Encrypt the following message using IDEA algorithm

Message: 1111 1011 1101 1010

Key: 10101001110111110110010111000011

3.2. Generate the key for decryption from the following encryption key.

Key: 10101001110111110110010111000011