**Cryptography**

In this project a message is encrypted and decrypted using IEEE paper: Security using colours and Armstrong Numbers for providing secure communication between end users. To make it more appealing, a picture message is encrypted and decrypted instead of a text file (text is even more easy to encrypt or decrypt).

The language used here is Java.

**Idea**

In the present world scenario, it is difficult to transmit data from one place to another place securely. This is because hackers are more powerful nowadays. To ensure secured data transmission there are several techniques being followed. One among them is cryptography which is the practice and study of hiding information.

Encryption is the transformation of data into some unreadable form. Its purpose is to ensure privacy by keeping the information hidden from anyone for whom it is not intended. Decryption is the reverse of encryption; it is the transformation of encrypted data back into some intelligible form. Encryption and decryption require the use of some secret information, usually referred to as a key. The data to be encrypted is called as plain text. The encrypted data obtained because of encryption process is called as cipher text.

**Types of Cryptographic Algorithms**

The three types of algorithms are depicted as follows

1. Secret Key Cryptography (SKC): Uses a single key for both encryption and decryption.

2) Public Key Cryptography (PKC): Uses one key for encryption and another for decryption

3) Hash Functions: Uses a mathematical transformation to irreversibly "encrypt" information.

**RGB Colour Model**

Any colour is the combination of three primary colours Red, Green and Blue in fixed quantities. A colour is stored in a computer in form of three numbers representing the quantities of Red, Green and Blue, respectively. This representation is called RGB representation which is used in computers to store images in BMP, JPEG and PDF formats. Here each pixel is represented as values for Red, Green and Blue. Thus, any colour can be uniquely represented in the three-dimensional RGB cube as values of Red, Green and Blue.

The RGB colour model is an additive model in which Red, Green and Blue are combined in various ways to produce other colours. By using appropriate combination of Red, Green and Blue intensities, many colours can be represented. Typically, 24 bits are used to store a colour pixel. This is usually apportioned with 8 bits each for red, green, and blue, giving a range of 256 possible values, or intensities, for each hue.

**Armstrong number** is a **number** that is equal to the sum of cube of its digits. For example, 0, 1, 153, 370, 371 and 407 are the **Armstrong numbers**.

**Approach**

The existing techniques involve the use of keys involving prime numbers and the like. As a step further ahead let us considers a technique in which Armstrong numbers and colours are used. Further a combination of substitution and permutation methods is used to ensure data security.

The substitution process is performed by assigning the ASCII equivalent to the characters. Permutation process is performed by using matrices and Armstrong number.

In this technique the first step is to assign a unique colour for each receiver. Each colour is represented with a set of three values.

**Initial Preparations**

1. A key is known to both the sender and the receiver.

This key is used to generate a numeric key by applying permutation on an array of Armstrong numbers. This is level 1 of encryption/decryption.

1. Now this numeric key is used to get the base intensity of 3 colours RGB. (Unique colour assignment for each receiver)

This base intensity is in the range of 0-255 and serve as starting number in a square matrix of side 16. This is level 2 of encryption/decryption.

Now, the initial preparations for both encryption and decryption are done.

**Encryption**

1. We open an image file using FileInputStream of java.io package.
2. We take one data unit (may be a character in case of text or a binary digit in case of image) at time (call it a) and encrypt it in the following manner.

* Take one character from the numeric key (call it b) and perform a xor b (call it c).
* Take c and break it into upper nibble (call it row) and lower nibble (call it column).
* Take one matrix from RGB and use its base to find which number exists at this row and column (call it d).
* Convert d into char and store it in the output file which is opened using FileOutputStream of java.io package.

1. In this way, encryption is done.

**Decryption**

Here we perform the steps of encryption in reverse order.

1. We open the encrypt image file using FileInputStream of java.io package.
2. We take one data unit (may be a character in case of text or a binary digit in case of image) at time (call it a) and encrypt it in the following manner.

* Take one matrix from RGB and use its base and a to find row (call it upper nibble) and column (call it lower nibble) corresponding to a.
* Combine this lower and upper nibble to form one number (call it b).
* Take one character from the numeric key (call it c) and perform b xor c (call it d).
* Convert d into char and store it in the output file which is opened using FileOutputStream of java.io package.

1. In this way, decryption is done.

**Conclusion**

In this way, a two-level encryption is achieved by using Armstrong numbers and colours with the help of IEEE paper. The technique of Secret Key Cryptography is used to secure data.