INFORMATION SECURITY PLAYCOLOR CIPHER

Lakshmi Pravallika Somisetty | *lsh28@umsystem.edu*

School of Compter Science and Engineering, University of Missouri-KansasCity

Abstract

An innovative cryptographic mechanism based on play color cipher is used for encryption and decryption of message by using RSA algorithm for key transmission. This method is used in-order to prevent Brute force attacks and Birthday attacks while sending a secret message. In order to provide security for transmission key we use RSA algorithm which is public algorithm available for anyone. We can substitute any character and number in plain text with a color block and will receive cipher text as color blocks at the end. And by decrypting the cipher text we will get our original message. In this project we use red, green, blue colors which provides 1,67,77,216 color combinations and transmits image securely by color block substitution of character, number and special symbols forming a Unicode.

Keywords: Play Color Cipher, Substitution, Combinations, Permutations, Unicode, RSA Algorithm

1 INTRODUCTION

1.1 Encryption: Encryption means securing digital data using one or more digital techniques along with a key and password. There are many kinds of algorithms which can be used to encrypt the information. The algorithm we use in this paper is RSA algorithm.

Encryption is of two types they are symmetric encryption and asymmetric encryption.

- **Symmetric Encryption:** It is an encryption technique which uses a single key for whole encryption and decryption process. This key only shared between user and receiver. The purpose of symmetric encryption is to secure secret or sensitive information. It's used most important industries such as defense, aerospace, banking, fitness care, and different industries wherein securing a person's, business', or organization's sensitive information is of the utmost importance.
  Symmetrically encrypted information can be accessed by anyone who knows the secret key. Some of the symmetric encryptions include Data Encryption Standard, Triple Data Encryption Standard, Advanced Encryption Standard, International Data Encryption Algorithm (IDEA), TLS/SSL protocol.

- **Asymmetric Encryption:** It is an encryption technique which uses two keys namely private key and public key for encryption and decryption process. This key only shared between user and receiver. In this encryption, two unique key pairs are used to encrypt data asymmetrically. Asymmetric encryption can be done using manually or automatically. Here only unique public

keys are exchanged between user and sender secretly, hence it is more secure than symmetric encryption.

Some of the other examples of this asymmetric encryption is Rivest Shamir Adleman, Digital Signature Standard, Digital Signature Algorithm, Elliptical Curve Cryptography, Diffie-Hellman exchange method, TLS/SSL protocol. Asymmetric encryption can also be used for digital signatures. It uses the private key to sign the files or information.

1.2 Key length: The length of key plays a key role in the algorithm and which supports different key lengths. In general, the more the key size is the security is high but it has its limitations. In theory the security increases exponentially with key length in symmetric key (Sastry V.U.K, 2006).

2. CRYPTOGRAPHIC ALGORITHMS

2.1 Secret key cryptography:

Secret key cryptography uses only one key for encryption and decryption process. It is also called as symmetric encryption i.e.; same key is used for converting plain text to cipher text and vice versa. It either uses block cipher or stream cipher for encryption (Stallings, 2008).

Stream Cipher: It is a single bit at a time cipher which usually works on feedback mechanism where key continuously changes. There are two kinds of stream ciphers. One is Self-synchronising which uses each bit in key as a function of previous keys.

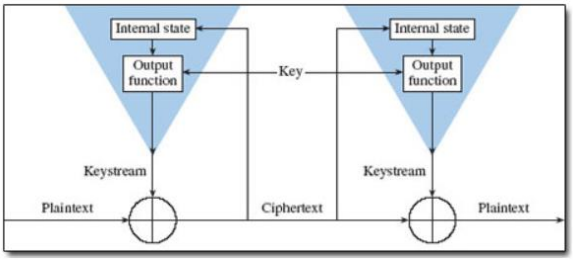


Fig-1: Self-synchronising stream cipher

And other is Synchronous which generates keystream independent of previous keys by using same key generation from sender and receiver.

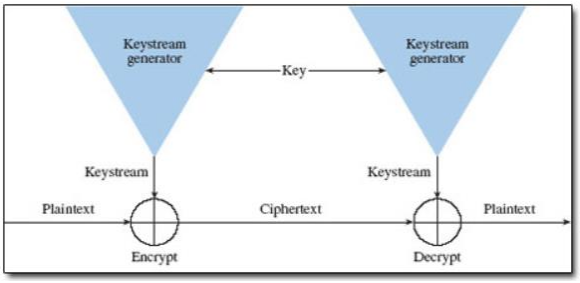


Fig-2: Synchronising stream cipher

Block Cipher: It uses a block of text at a time. A plain text will be converted into cipher text using a same key and this cipher text will be converted into stream cipher using the same key. Feistel cipher is widely used for block cipher encryption which uses substitution, permutation and key expansion. Block cipher uses several modes of operation such as

- Electronic Code Book mode
- Cipher Block Chaining mode
- Cipher Feedback mode
- Output Feedback mode
- Counter mode

2.2 Public-key cryptography:

It is published by Martin Hellman and Whitfield Diffie in 1976. It uses two keys namely public key and Private key which is unknown to anyone. Any key can be used first to encrypt information irrespective of order. It is also called as asymmetric encryption.

2.3 Hash Functions:

It is also a one-way encryption also called as Message Digest. Depends on plain text, hash value is defined which is impossible to find out. Hash functions are also designed so that small changes in the input produce significant differences in the hash value. Some of Hash functions being used are:

- Message Digest Algorithm
- SHA-1 Algorithm
- SHA-256/512 Algorithm
- eD2K and many other

3.RSA ALGORITHM

RSA Algorithm is published in 1977 developed by Ron Rivest, Adi Shamir, and Leonard Adleman. It is an example of asymmetric encryption where it generates a public key by multiplying two random large prime numbers. Also, private key is generated using same prime numbers. Here encryption is done using public key and decryption is done using private key, where public key is shared with everyone and private key is secret.

The security of RSA relies on the difficulty of multiplying the product of two large prime numbers, and the huge solution of its multiplication. Whether it is as difficult as the factoring problem is an open question. Till now there is no method to solve this problem yet it is more securable but it is a slow algorithm compared to other. Hence, it cannot be used to directly encrypt the data and is also used share the keys. So, it can be used for huge encryptions.

This algorithm uses four steps. They are:

- Key generation
- Key distribution
- Encryption
- Decryption

A basic principle behind RSA is the observation that it is practical to find three very large positive integers e, d, and n, such that for all integers m (with $0 \leq m < n$). It is very hard to find d. we can also change the two large prime numbers when we need a new function.

$(m^e)^d = m(\mathrm{mod} n)$

The public key is represented by the integers n and e, and the private key by the integer d and m represents the message.

3.1 Key generation:

- Choose two distinct prime numbers p and q. p and q should be random and should be similar in magnitude. Where p and q must be kept secret.Now compute n=p*q
  - n is used as the modulus for both the public and private keys and the length is called as key length and expressed in bits. n is the part of public key.
- Compute λ(n), λ(p), λ(q)
  - Where λ(n) is least factor of λ(p), λ(q)
  - Both p and q are prime, $\lambda(p) = \varphi(p) = p - 1$, and likewise $\lambda(q) = q - 1$. Hence λ(n) = LCM (p − 1, q − 1). λ(n) must be kept secret.
- Choose an integer e such that $1 < e < \lambda(n)$
  - e is released as part of the public key. e, λ(n) are co-prime and e is part of public key.
- d as $d \equiv e^{-1} \pmod{\lambda(n)}$; that is, d is the modular multiplicative inverse of e modulo λ(n)

3.2 Key distribution:

Suppose Ram wants to send information to Bheem and they use RSA Algorithm. Ram should know the public key to convert plain text to cipher text i.e., encryption and Bheem must use his private key to convert cipher text to plain text i.e., decrypt the message.

Bheem transmits his public key (n, e) to Ram via a reliable, but not necessarily secret, route. Bheem's private key (d) is never distributed.

3.3 Encryption:

After Ram obtains Bheem’s public key, he can send a message M to Bheem.

To do it, he first turns M into an integer m such that $0 \leq m < n$ by using an agreed-upon reversible protocol known as a padding scheme. He then computes the ciphertext c, using Bheem's public key e,

$$C=(m^e) \pmod{n}$$

This can be done reasonably quickly, even for very large numbers, using modular exponentiation.

3.4 Decryption:

Bheem can recover m from C by using his private key exponent d by computing

$$c^d = (m^e)^d = m(\text{mod} n)$$

Given m, he can recover the original message M by reversing the padding scheme.

4. ALGORITHM USED IN PAPER

DES algorithm in general uses a 64 bit-key in which 8 bits are used for parity checking, hence only 56 bits are being used and gives 2^{56} permutations. AES algorithm depends on rounds of encryption hence provides few permutations only.

When color cipher is used and encrypted with RSA algorithm, for three colors namely Red, Green and Blue it provides 1,67,77,216 permutations due to the color range and scaling factor. In play color cipher we need to transmit key for security of algorithm. It uses RSA algorithm for key transmission which is a public key algorithm. It uses same technique as of play fair cipher 5*5 matrix where we need to select a key at first. Then we have encrypted the key and get output and needs to check color for the value by multiplying alphabet number and color. By substituting this we will get cipher text as color (Prof. K. Ravindra, 2010).

4.1 Encryption:

1.RSA consists of 128-bit key which is divided into 4 parts K1, K2, K3, K4.

2. Read the plain text and convert into UTF. Now create a block of plain text named C1.

3. Apply initial permutation to C1 with key K4. The output will be cipher text C2.

4. Now apply permutation for C2 with K4 and it will be C3.

5. Substitute the characters in C3 with color matrix formed (process is discussed above).

6. Now cipher text and key can be transmitted to receiver in the form of color blocks by using RSA algorithm.

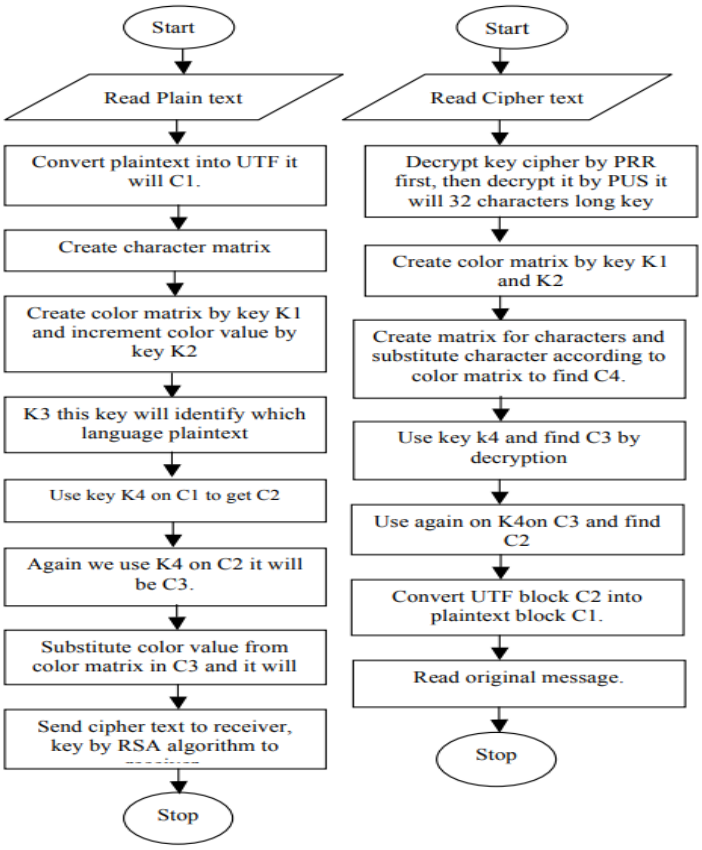
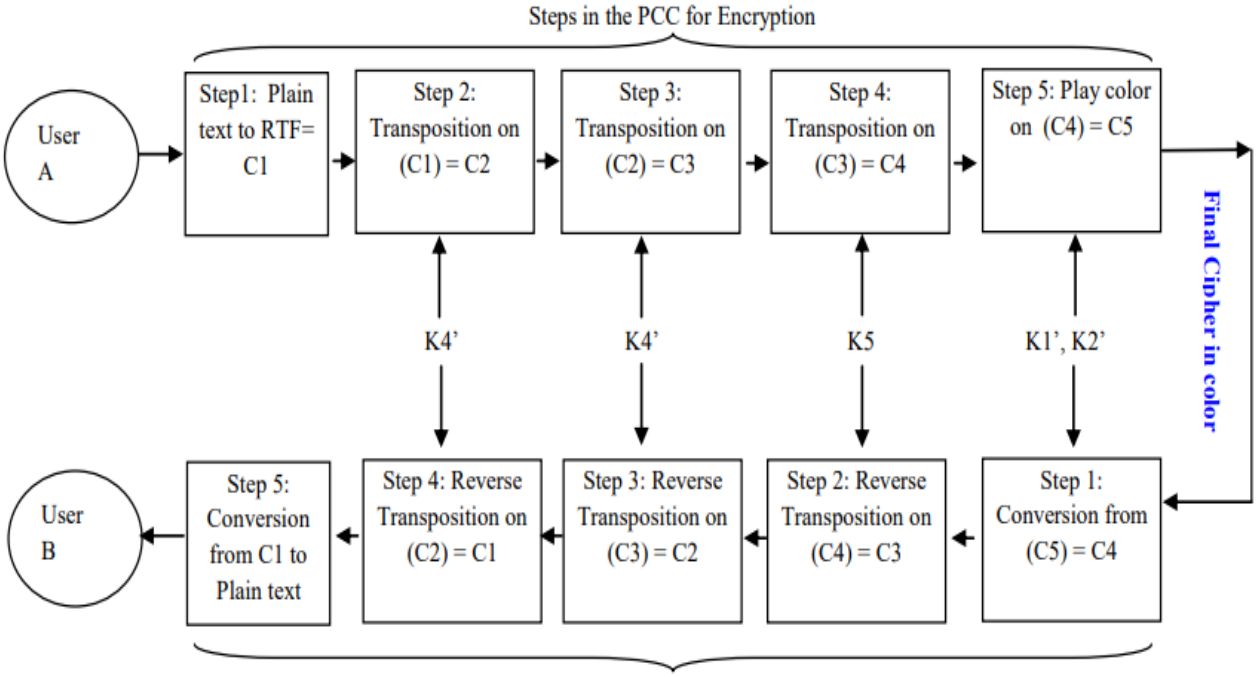


Fig-3: Encryption & Decryption flowchart



4.2 Decryption:

1. Receiver gets the encrypted key and decrypts it using receiver private key and sender's public key.
2. Receiver gets the key after decryption as 32-bit long. Which then can be divided into 4 parts K1(1-15 bits), K2 (16-22 bits), K3(23rd bit), K4(24-32 bits) respectively.
3. Now from character matrix color blocks can be substituted and C3 can be found.
4. Use K4 in the same matrix and find out C2.
5. Similar to reverse encryption, apply K4 to C2 which gives C1.
6. C1 is in the form UTF and is converted to plain text.

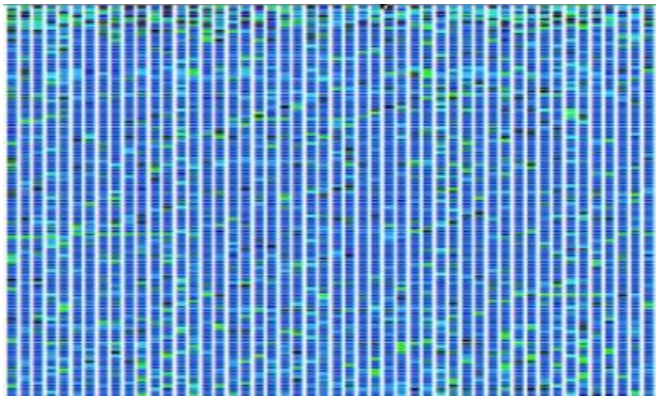
5. CODE USED

```
from PIL import Image
import math

#Encodes string to pixel image.
def encode_text(text, path):
    HEXD = text.encode(encoding="utf_8").hex() #Converts text to utf 8 hex format
    LGTH = 1
    HIGT = 1
    TARR = [HEXD[i:i+32].ljust(32, "0") for i in range(0, len(HEXD), 32)]

    #Calculate image width
    if(len(str(HEXD)) < 32):
        LGTH = math.ceil(len(str(HEXD))/6)
        HIGT = 1
    elif(len(str(HEXD)) >= 32):
        LGTH = 32
        HIGT = math.ceil(len(TARR)/6)

    HEXT = [HEXD[i:i+6].ljust(6, "0") for i in range(0, len(HEXD), 6)] #Creates array of hex data in 6-character color format

    OIMG = Image.new("RGB", (LGTH,HIGT))

    PIXL = []

    for i, v in enumerate(HEXT):
        CVAL = HEXT[i]
        RGBV = tuple(int(CVAL[i:i+2], 16) for i in (0, 2, 4)) #Converts hex color value to RGB
        PIXL.append(RGBV)

    OIMG.putdata(PIXL)
    OIMG.save(path+HEXT[0]+".png")

    print("File '"+HEXT[0]+".png' saved!")

#Encodes hex value to pixel image.
def encode_hex(data, path):
```

```
    TARR = [data[i:i+32].ljust(32, "0") for i in range(0, len(data), 32)]

    #Calculate image width
    if(len(str(data)) < 32):
        LGTH = math.ceil(len(str(data))/6)
        HIGT = 1
    elif(len(str(data)) >= 32):
        LGTH = 32
        HIGT = math.ceil(len(TARR)/4)

    HEXT = [data[i:i+6].ljust(6, "0") for i in range(0, len(data), 6)]

    OIMG = Image.new("RGB", (LGTH,HIGT))

    PIXL = []

    for i, v in enumerate(HEXT):
        CVAL = HEXT[i]
        RGBV = tuple(int(CVAL[i:i+2], 16) for i in (0, 2, 4))
        PIXL.append(RGBV)

    OIMG.putdata(PIXL)
    OIMG.save(path+HEXT[0]+".png")

    print("File '"+HEXT[0]+".png' saved!")

codes image to hex value.
def decode_file(file):
    FIMG = Image.open(file)
    FDAT = list(FIMG.convert('RGB').getdata())
    HDAT = ""

    for r, g, b in FDAT:
        HDAT += '{:02x}{:02x}{:02x}'.format(r, g, b) #Converts RGB color values to HEX

    print(FDAT[0][0])
    print("Decoded hex output: "+HDAT)
    return HDAT

nverts decoded output to UTF-8 text.
def convert_hex(hex):
    CONV = hex.replace(" ","") #Removes space delimiters
    OUTP = bytes.fromhex(CONV).decode('utf-8') #Converts hex to UTF-8 string

    print("Converted output: "+OUTP)
    return OUTP
```

Input:

Output:



Cryptanalysis

- In English, we have ten numbers and 26 characters so we will get key of 26^{32} that have $1.9*10^{45}$ permutations when we perform one encryption per second it will take $6*10^{35}$ years to do an attack. Whereas, for ten numbers we will get key 10^{32} when executed one encryption per microsecond 3.1*1029 years to do an attack.
- We transmit key of 23 bit using RSA algorithm it's key length is 92 bits and key space is $2^{92}=5.0*10^{27}$ key. When one value is done for 10^{-3}seconds then time required for key space

is $1585*10^{14}$ years. If we perform one encryption per micro second it takes $2.4*10^{16}$ years and for 10^6 years it takes 1842.6 years (gaitonde, 2012).

- When we apply these two logics for 18 decillion number of colors it is impossible for anyone to perform Brute force attack or Man-in-Middle attack.

6.ADVANTAGES AND DISADVANTAGES

6.1 Advantages:

- It is very easy to implement and is safe and secure for transmitting confidential data.
- Brute force attack and Man-in-Middle attack doesn't occur.
- Overcomes limitation of many other ciphers.
- Bulk data capacity and high correlation among pixels can be done easily.
- Transportation cost through this channel is very low and speed is very fast.

6.2 Disadvantages:

- Not much reliable
- Takes more time

Conclusion

We developed a play color cipher using substituting technique where we used RSA algorithm for key transmission to transfer information securely. We used RSA algorithm in order to provide more security for secret key transmission due to its maximum no. of permutations. We use a 32-bit long key for encryption and decryption process. In the beginning we convert plain text to UTF file as any kind of programming language can be used and UTF provides a better and faster algorithm than RTF.

This method also prevents Brute force attacks, Birthday attacks and many other which can be proved by performing cryptanalysis. A stronger cipher using 18 Decillion colors and their enormous permutations of color combinations is built. It also takes some decades to crack such kind of cipher.

REFERENCES

Andrey Bogdanov, D. K. (2011). Biclique Cryptanalysis of the Full AES. *Crypto 2011 cryptology conference.* Santa Barbara, California.

Dinesh Sharma, R. P. (2017). Colour Based Cryptography. *International Research Journal of Engineering & Technology*.

gaitonde, A. (2012). Color Coded Cryptography. *International Journal of Scientific & Engineering Research*, Volume-3, Issue-7.

Pritha Johar, S. E. (2012). A Novel Approach to Substitution Play Color Cipher. *International Journal of Next Generation Computer Application*, Volume1.

Prof. K. Ravindra, D. K. (2010). A block cipher generation using color substitution. *International Journal of Computer Applications*, (p. Volume1).

Sastry V.U.K, S. U. (2006). A large block cipher using modular arithmetic inverse of a key matrix and mixing of the key matrix and the plain text. *Journal of Computer Science.*

Stallings, W. (2008). *Cryptography and Network Security, principle and Practice, 5th edition.*