"Traditional Cryptography: A Mathematical overview"

Sonal Sarnaik
Department of MCA
Marathwada Institute of technology, Aurangabad, Maharashtra, India

Nilesh Jaybhay
Second year, Department of MCA
Marathwada Institute of technology, Aurangabad, Maharashtra, India

Rutuja Sontakke
Second year, Department of MCA
Marathwada Institute of technology, Aurangabad, Maharashtra, India

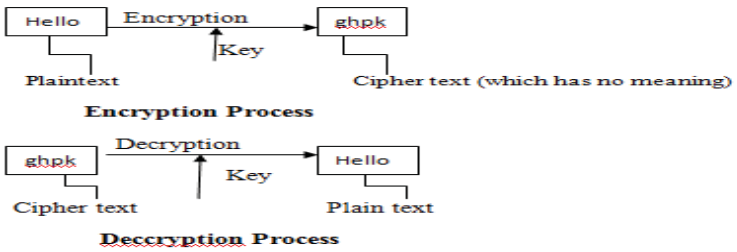
Abstract - Cryptography is the mathematical system which is used to provides privacy and authentication to the data. Now a day's Modern Cryptography is used everywhere for security of data, For example in Banking sector where thousands of transaction takes place every minute , Military where security and authentication of the data is very important , Social websites where lots of personal data of people must be secure etc. To understand the working of Modern cryptography we first need to understand the base of cryptography i.e. we need to understand Traditional cryptography. This paper gives the description of Modular arithmetic, Congruence, Modular multiplicative inverse, Shift cipher and its implementation in C++, Affine cipher its implementation in C++.

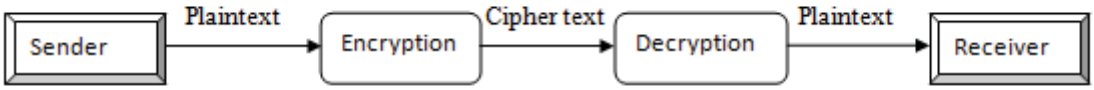
Keyword: Modular arithmetic, modular inverse, shift cipher, affine cipher.

I. INTRODUCTION

Now a day's exchange of information plays a very important role, as everything has become digital. We exchange information, we do many online transactions, online shopping, exchange official and personal information online, in all these examples security plays very important role. This role is played by cryptography. Cryptography is the study of mathematical system involving two kinds of security problems, privacy and authentication. Cryptography is divided in two parts, Traditional cryptography and modern cryptography[2][3][5][6][7][8][12].

Security to the data is provided by converting original text(called as plain text) into cipher text (which is not understand by anyone) at sender and that cipher text is send to receiver where cipher text is converted into original text. The conversion of plain text to cipher text is done by applying encryption key to it and similarly plain text is retrieved from cipher text by applying decryption key to it[3][6][7][12].





Overall Communication

A cryptosystem is a pair of algorithm that takes a key and converts plain text to cipher text and back. The first one who used this technique was Julies Caeser ,as he did not trust his messenger to pass official messages between officers and governors ,so he created a system in which each original character is replaced by the character by three positions ahead of the roman alphabet[5][6][7][8].

Example:-(For English alphabet)

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
d	e	f	g	h	i	j	k	l	m	n	o	P	q	r	S	T	u	v	W	x	y	Z	a	b	c

Shifted Text by three positions

Original text

By suing above chart original message “MEETATNIGHT”, will be encrypted as,

Original text:-	M	E	E	T	A	T	N	I	G	H	T
Encrypted text:-	p	h	h	w	d	W	Q	l	j	k	w

Encrypted message will be “phhwdwqljkw”, which will not be understood by anyone. In this example, original text is replaced by other text which is three positions ahead of it, is called as key. Cryptosystem can be categorized into two types, Symmetric and asymmetric[5][6][7][8][11][12]. Symmetric cryptosystem uses a shared secret key which is used for encryption process as well as for decryption process. for example DES (Data Encryption Standard)[6], AES(Advance Encryption Standard)[15],Blowfish[1][10]. Asymmetric key cryptography uses public key for encryption and a private key for decryption, there is no need for a shared secrete key. Public key is known to all so any one can encrypt data and send to the receiver , where receiver will apply its own private key and will decrypt the data. Examples of this are Diffie-Hellman , RSA and Elgamal [6][7][8][10][12][13][14].

Traditional cryptography is based on symmetric key cryptography, where sender and receiver both uses the same key for encryption and for decryption. So the total security of the data will depend on the type of the key, Length of the key, way to share secret key[6][7].It works on two basic components,

1. Substitution:-In this method one letter is replaced by other letter.
2. Transposition:-In this method letters are arranged in different orders.

Substitution method and transposition method can be mono alphabetic (one original letter will substitute/transposed by other letter) and can be poly alphabetic (one original letter will substitute/transposed by many letters)[5][6][7].

Two different methods used in traditional cryptography to encrypt and decrypt data. Such as

1. Shift cipher
2. Affine Cipher

Before elaborating these concepts one should know some mathematical concepts[5][6][7].

II.MODULAR ARITHMETIC

In mathematics, **modular arithmetic** is a system of arithmetic where numbers "wrap around" upon reaching a certain value—the modulus. This is shown in below example[5][6][8].

Example: - a mod m ---------- (2.1)

Where m is 12 and a is random value stars from 1

Cycle A	Cycle B	
1 mod 12=1	13 mod 12=1	→ Cycle stars again
2 mod 12=2	14 mod 12=2	
3 mod 12=3	15 mod 12=3	
4 mod 12=4	16 mod 12=4	
5 mod 12=5	17 mod 12=5	
6 mod 12=6	18 mod 12=6	
7 mod 12=7	19 mod 12=7	
8 mod 12=8	20 mod 12=8	
9 mod 12=9	21 mod 12=9	
10 mod 12=10	22 mod 12=10	
11 mod 12=11	23 mod 12=11	
12 mod 12=0	24 mod 12 =0	
	25 mod 12=1	→ Cycle stars again
	26 mod 12=2	

III. CONGRUENCE

Modular arithmetic can be handled mathematically by introducing a congruence relation on the integers that is compatible with the operations of integers: addition, subtraction, multiplication. For a positive integer m, two integers *a* and *b* are said to be **congruent modulo** *n* ,i.e[5].

$$\mathbf{a \equiv b \bmod m} \quad \text{----------} \quad (3.1)$$

This can be said only if

i. m divides (b-a)

ii. a mod m = b mod m.

Example:-

105 mod 13 , 508 mod 13

i. (b-a)/m
(508-105)/13
=403/13
=31

ii. a mod m = 105 mod 13 = 1
b mod m =508 mod 13 =1

So,105 mod 13 = 508 mod 13

As i and ii both conditions are satisfied so we can say that **105 ≡ 508 mod 13.**

IV. MODULAR MULTIPLICATIVE INVERSE

In modular arithmetic modular multiplicative inverse of an integer a modulo m is an integer a^{-1} such that[5][8][12] ,

$$\mathbf{a * a^{-1} \equiv 1 \bmod m} \quad \text{----------} \quad \mathbf{(4.1)}$$

Where a^{-1} can be found by ,

$$\mathbf{a^{-1} = a^{\varphi(m)-1} \bmod m} \quad \text{----------} \quad \mathbf{(4.2)}$$

a^{-1} can be find out by using Euler's function ϕ(p) .if m is prime then directly can apply the formula but if m is composite then convert it in the multiples of prime, for example:

25=5 * 3, where 5 and 3 are prime numbers.

Example:-

5^{-1}=?, Where m=26

26=13 * 2,

So P_1=13 ,p_2=2

$$\omega(m)=\omega(p_1-1) *\omega(p_2-1)$$
$$=\omega(13-1) * \omega(2-1)$$
$$=12 * 1$$
$$\omega(m) =12$$

putting in equation (4.2)

$$a^{-1} = a^{\omega(m)-1} \bmod m$$
$$5^{-1} = 5^{12-1} \bmod 26$$
$$5^{-1}=21$$

Putting 5^{-1} in equation (4.1)

$$a * a^{-1} \equiv 1 \bmod m$$
$$5*5^{-1} \bmod 26 \equiv 1 \bmod 26$$
$$5*21 \bmod 26 \equiv 1$$
$$105 \bmod 26 \equiv 1$$

1≡1 equation (2) satisfied.

V.SHIFT CIPHER(CAESAR CIPHER)

In cryptography, a Caesar cipher, also known as a Caesar's cipher, the shift cipher, Caesar's code or Caesar shift, is one of the simplest and most widely known encryption techniques[5][6][8][12]. In this technique a text is replaced by other text which is on k^{th} position from the original text .K is called as key .As we have 26 alphabets in our alphabetic system so we can shift a particular text from 1 by 26^{th} position. General rule for encryption and decryption is shown bellow. Where p is plain text, C is cipher text ,k is key used to encrypt and decrypt data, x is original data ,y is decrypted data.

Let $P = C = K = Z_{26}$ $(0 \le k \le Z_{26})$

Encryption $(e_k(x))$=

Decryption $(d_k(y))$ =

Z_{26}

Example:-

Let original text is “ENCRYPTION” and key is 3, So first find out its index position as we have to shift a key by 3 positions.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
T	U	V	W	X	Y	Z												
19	20	21	22	23	24	25												

Encryption process

Original text	**E**	**N**	**C**	**R**	**Y**	**P**	**T**	**I**	**O**	**N**
x	4	13	2	17	24	15	19	8	14	13
$x + k$	7	16	5	20	27(1)	18	22	11	17	16
$x + k \bmod 26$	7	16	5	20	1	18	22	11	17	16
Decrypted Text	**H**	**Q**	**F**	**U**	**B**	**S**	**W**	**L**	**O**	**Q**

Decryption process

Received text	**H**	**Q**	**F**	**U**	**Y**	**S**	**W**	**L**	**O**	**Q**
y	7	16	5	20	1	18	22	11	17	16
$y-k$	4	13	2	17	-2	15	19	8	14	13
$(y-k) \bmod 26$	4	13	2	17	24	15	19	8	14	13
Retrieved Text	**E**	**N**	**C**	**R**	**Y**	**P**	**T**	**I**	**O**	**N**

Fig.5.1. Implementation of Shift Cipher in C++

```
#include<iostream.h>
#include<conio.h>
#include<stdlib.h>
#include<string.h>
#include<stdio.h>
class Shift
{       public:
        void encrypt()
        {
        cout<<"\n\n ENCRYPTION ";
        char a[]={'a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p',
                  'q','r','s','t','u','v','w','x','y','z'};
        char str[50];
        int len,rem,key,i,j;
        cout<<"\n\n Enter a string : ";//Enter plaintext
        gets(str);
        cout<<"\n\n Enter a key for encryption : ";//Enter
secret key;
        cin>>key;
        cout<<"\n\n Ciphertext after encyption : ";
        for(i=0;i<26;i++)//Encryption
        {
        for(j=0;j<26;j++)
        {
        if(a[j]==str[i])
        {
        int num=j+key;
        rem=num%26;
        str[i]=a[rem];
        cout<<str[i];
        break;
        } } } }
        void decrypt()
        {
        cout<<"\n\n DECRYPTION ";
        char a[]={'a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p',
                  'q','r','s','t','u','v','w','x','y','z'};
        char str[50];
        int len,rem,key,i,j;
        cout<<"\n\n Enter a string : ";//Enter ciphertext
        gets(str);
        cout<<"\n\n  Enter key for decryption : ";//Enter
secret key
        cin>>key;
        cout<<"\n\n Plaintext after decryption : ";
for(i=0;i<26;i++)//Decryption
{       for(j=0;j<26;j++)
        {
        if(a[j]==str[i])
        {
        int num;
        num=j-key;
        if(num<0)
        {num=26-(-num);
        }
        str[i]=a[num];
        cout<<str[i];
        break;
        }}}};
void main()
{
        clrscr();
        Shift sh;
        int choice;
        char ch;
        do
        {
        cout<<"\n\n Select Operation do u want to perform ";
        cout<<"\n\n [1]Encryption \n\n [2]Decryption\n";
        cout<<"\n\n Enter your choice : ";
        cin>>choice;

        switch(choice)
        {
                case 1:sh.encrypt();
                        break;
                case 2:sh.decrypt();
                        break;
                default:cout<<"\n\n Invalid choice ";
        }
        cout<<"\n\n Do you want to continue [y/n]?";
        cin>>ch;
        }while(ch=='y' || ch=='Y');
        getch();
}
```

Fig.5.2.Output of Shift cipher(encryption).

```
Enter a string : ENCRYPTION
Enter a key for encryption : 3
Ciphertext after encryption : hqfubswoq
Do you want to continue [y/n]?
```

Fig.5.3.Output of Shift cipher(Decryption).

```
Select Operation do u want to perform
[1]Encryption
[2]Decryption
Enter your choice : 2
DECRYPTION
Enter a string : hqfubswoq
Plaintext after decryption : ENCRYPTION
Do you want to continue[y/n]?
```

VI.AFFINE CIPHER

Affine cipher is similar to shift cipher, but the 'key' for the Affine cipher consists of 2 numbers, *a* and *b*. *a* should be chosen to be relatively prime to *26* (i.e. *a* should have no factors in common with *26*)[5][6][7][8][12]. For example 7 and 26 have no factors in common, so 7 is an acceptable value for *a*, however 8 and 26 have factors in common i.e 2, so 12 cannot be used for a value of *a*. b can be any value between 0 to 25.below is the rule of affine cipher where, *c* is the cipher text, *p* is the plaintext.

Let $P = C = Z_{26}$ and $k = \{(a,b) \in Z_{26} * Z_{26} : gcd(a,26)=1\}$
For k= $(a,b) \in k$,
Encryption $(e_k(x))$=
Decryption $(d_k(y))$ =
Z_{26}

Example:
Let the original text is MCAMIT, where a =3 and b=7.

We first have to check the condition for a i.e gcd(a,26)=1.

Here gcd(3,26)=1 condition satisfy, so we can use a=3 for encryption.

Encryption process

	M	C	A	M	I	T
Original text						
x	12	2	0	12	8	19
$ax + b$	43	13	7	43	31	64
$ax + b \bmod 26$	17	13	7	17	5	12
Decrypted Text	**R**	**N**	**H**	**R**	**F**	**M**

For decryption process we need to find a^{-1} by using (4.2) equation and we get $3^{-1}=9$.

Decryption process

	R	N	H	R	F	M
Received text						
y	17	13	7	17	5	12
$y - b$	10	6	0	10	-2	5
$a^{-1}(y - b) \bmod 26$	12	2	0	10	8	19
Retrieved Text	**M**	**C**	**A**	**M**	**I**	**T**

Fig.6.1. Implementation of Affine Cipher in C++

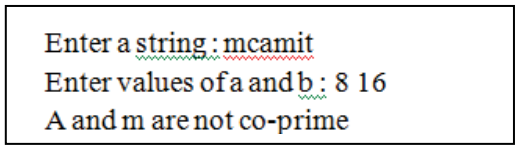
```
//Program in cpp for Affine Cipher
#include<iostream.h>
#include<conio.h>
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
void main()
{  clrscr();
  char c[]={'a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p','q','r','s',
't','u','v','w','x','y','z'};
char str[20];
int len,rem,k,i,j,a,b,m=26,x,iv,num,ans,y[20],k1=0;
cout<<"\n\n Enter a String : ";//Plaintext
gets(str);
cout<<"\n\n Enter values of a and b : ";//values for key
generation
cin>>a>>b;
for(i=2;i<m;i++)//Check for coprime condition
{
if(a%i==0 && m%i==0)
{cout<<"\n\n a and m are not co-prime ";
getch();
exit(0);
}}
len=strlen(str);
cout<<"\n\n ENCRYPTION ";
cout<<"\n\n Ciphertext after encryption : ";
for(i=0;i<len;i++)//Encryption
{
if(str[i]==32)
cout<<" ";
for(j=0;j<26;j++)
{
if(c[j]==str[i])
{
int num=a*j+b;
rem=num%26;
y[k1]=rem;
k1++;
str[i]=c[rem];
cout<<str[i];
break;
}}}
for(i=2;i<m;i++)//Modular Inverse
{
if((a*i)%m==1)
{
iv=i;
cout<<"\n\n Inverse of a is = "<<iv;
}
}
cout<<"\n\n DECRYPTION ";
cout<<"\n\n Plaintext after decryption : ";
k1=0;
for(i=0;i<len;i++)//Decryption
{
if(str[i]==32)
cout<<" ";
for(j=0;j<26;j++)
{
ans=iv*(y[k1]-b);
num=ans%26;
if(num>=0)
{
str[i]=c[num];
k1++;
}
else
{
num=26-(-num);
str[i]=c[num];
k1++;
}
cout<<str[i];
break;
}
}
getch();
}
```

Fig.6.2. Output of Affine Cipher

```
Enter a string : mcamit
Enter values for a and b : 3 7
ENCRYPTION
Ciphertext after encryption : rnhrfm
Inverse of a is = 9
DECRYPTION
Plaintext after decryption : mcamit
```

If values of a and b are not co-prime then it gives following output.

Fig.6.3. Output of Affine Cipher(if a and b are not coprime)



VII. CONCLUSION

Cryptography is very important concept used every where now a day's. In this paper we have focused on two important cipher techniques with its implementation in C++ and various mathematical concepts of traditional cryptography .It is very important to know in detail the mathematical concepts of traditional cryptography to gain knowledge of modern cryptography.

REFERENCES

[1] Bruce Schneier,*"The Blowfish Encryption Algorithm",* Dr. Dobb's Journal of Software Tools, pp. 4, 38, 40, 98, 99, 1994.
[2] Chitra Desai," *A Novel Approach for Digital Signature Scheme Based on Solving Two Hard Problems"* IJCMSA: Vol. 6, No. 3-4, July-December 2012, pp. 95– 100.
[3] Chitra G.Desai ,Rupali Bhakkad ,Sonal Sarnaik, "Identifying Quadratic Residuity Using Legendre-Jacobi Symbol".
[4] Joan Daemen and Vincent Rijmen," *Rijndael for AES".,* AES Candidate Conference, pp. 343–348, 2000.
[5] Menezes, Alfred J; van Oorschot, Paul C.; Vanstone, Scott A. (2001),*"Handbook of AppliedCryptography"*,[Online]
[6] Stinson, Douglas Robert (2006),"*Cryptography: Theory and Practice (3rd ed.)*", London: CRC Press.
[7] Whitfield Diffie and Martin E. Hellman, "*New Directions in Cryptography",* IEEE Transactions on Information Theory, 22(6):644-654, 1976.
[8] Jonathan Katz and Yehuda Lindell ,"Introduction to Modern Cryptography".
[9] Anjali Patil, Rajeshwari Goudar "A Comparative Survey Of Symmetric Encryption Techniques For Wireless Devices".
[10] Monika Agrawal, Pradeep Mishra, "A Comparative Survey on Symmetric Key Encryption Techniques".
[11] Luca Trevisan ,"Cryptography ",Lecture Notes from CS276, Spring 2009 ,Stanford University.
[12] Bruice Schineier ,"Applied Cryptography".
[13] Shafi Goldwasser, Mihir Bellare July 2008,"Lecture Notes on Cryptography".
[14] Rivest, R., A. Shamir and L. Adleman,"*A Method for Obtaining Digital Signature and Publickey Cryptosystem Communications"*ACM.21:120-126.1978 http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.40.5588.