“CRYPTOGRAPHY PROJECT”

PROJECT REPORT

Submitted for the Course: CRYPTOGRAPHY FUNDAMENTALS (CSE 1011)

By

ARPIT RATHI	**16BCI0065**
PARMEET SINGH	**16BCE0184**

Slot: F1

Name of faculty: PROF.MARIMUTHU K.

(SCHOOL OF COMPUTER SCIENCE AND ENGINEERING)

October, 2017

CERTIFICATE

This is to certify that the project work entitled “Elgamal Cryptosystem” that is being submitted by “ ***Arpit Rathi and Parmeet Singh”*** Cryptography Fundamentals(CSE 1011) is a record of bonafide work done under my supervision. The contents of this Project work, in full or in parts, have neither taken from any other source nor have been submitted for any other CAL course.

Place: VIT UNIVERSITY

Date: 26th October,2017

Signature of Students:

Arpit Rathi

Parmeet Singh

Signature of Faculty: PROF. MARIMUTHU K.

INTRODUCTION

Cryptography, or cryptology, is the practice and study of hiding information. It is sometimes called code, but this is not really a correct name. It is the science used to try to keep information secret and safe. Modern cryptography is a mix of mathematics, computer science, and electrical engineering.

When a message is sent using cryptography, it is changed (or *encrypted*) before it is sent. The method of changing text is called a "code" or, more precisely, a "cipher". The changed text is called "ciphertext". The change makes the message hard to read. Someone who wants to read it must change it back (or *decrypt* it). How to change it back is a secret. Both the person that sends the message and the one that gets it should know the secret way to change it, but other people should not be able to. Studying the cyphertext to discover the secret is called "cryptanalysis" or "cracking" or sometimes "code breaking".

Public key cryptography (PKC) is an encryption technique that uses a paired public and private key (or asymmetric key) algorithm for secure data communication. A message sender uses a recipient's public key to encrypt a message. To decrypt the sender's message, only the recipient's private key may be used.

ABSTRACT

ElGamal encryption system is an asymmetric key encryption algorithm for public-key cryptography which is based on the Diffie–Hellman key exchange. The system provides an additional layer of security by asymmetrically encrypting keys previously used for symmetric message encryption.

It has two layers :

1.) SHARING OF KEYS USING DIFFIE HELLMAN :
2.) AFFINE CIPHER USING THE SHARED KEY GENERATED

2. Methodology: Experimental/ Simulation:

DETAIL PLAN :

We will study the Elgamal Cryptosystem and then implement it using java.

For the GUI, we will use Java Applet.

Then we will use that GUI created to demonstrate the Elgamal Cryptosystem.

CODE :

```
import java.applet.*;
import java.net.*;
import java.awt.*;
public class CRYPTOGRAPHY extends java.applet.Applet
{
   Button b=new Button("ENCRYPT");
   Button b1=new Button("DECRYPT");
   TextField secretNumber=new TextField(10);
   TextField secretNumber1=new TextField(10);
   TextArea message=new TextArea();
   TextArea encryptedMessage=new TextArea();
   TextField publicKey=new TextField(10);
   Label l1=new Label("PUBLIC KEY");
   Label l2=new Label("SECRET NUMBER r");
   Label l3=new Label("SECRET NUMBER s");
   public void init()
   {
      add(b);
      add(b1);
      add(secretNumber);
```

```
add(secretNumber1);
add(publicKey);
add(message);
add(encryptedMessage);
add(l1);
add(l2);
add(l3);
setLayout(null);
publicKey.setSize(100,20);
publicKey.setLocation(150,50);
secretNumber.setSize(100,20);
secretNumber.setLocation(150,100);
secretNumber1.setSize(100,20);
secretNumber1.setLocation(150,140);
l1.setSize(120,20);
l1.setLocation(20,50);
l2.setSize(120,20);
l2.setLocation(20,100);
l3.setSize(120,20);
l3.setLocation(20,140);
message.setSize(500,400);
message.setLocation(20,200);
encryptedMessage.setSize(500,400);
encryptedMessage.setLocation(550,200);
b.setSize(70,20);
b.setLocation(20,620);
b1.setSize(70,20);
```

```
    b1.setLocation(120,620);
    setSize(1200,1000);
}

public String encrypt(int r,int s,int a,String message)
{
    //Generation of Private Keys
    int ka=(int)Math.pow(a,r)%26;
    int kb=(int)Math.pow(a,s)%26;
    //Generation of Shared Secret Key
    int sk1=(int)Math.pow(ka,s)%26;
    int sk2=(int)Math.pow(kb,r)%26;
    String mes=message.toLowerCase();
    String answer="";
    String template="abcdefghijklmnopqrstuvwxyz";
    for(int i=0;i<message.length();i++)
    {
        if(Character.isLetter(mes.charAt(i)))
        {
            int xx=template.indexOf(mes.charAt(i));
            int crypt=(xx*sk1)%26;
            answer=answer+template.charAt(crypt);
        }
        else
            answer=answer+mes.charAt(i);
    }
    return answer;
```

```
}

public String decrypt(int r,int s,int a,String message)
{
    //Generation of Private Keys
    int ka=(int)Math.pow(a,r)%26;
    int kb=(int)Math.pow(a,s)%26;
    //Generation of Shared Secret Key
    int sk1=(int)Math.pow(ka,s)%26;
    int sk2=(int)Math.pow(kb,r)%26;
    String mes=message.toLowerCase();
    String answer="";
    String template="abcdefghijklmnopqrstuvwxyz";
    int sk2Inverse=modInverse(sk2,26);
    for(int i=0;i<message.length();i++)
    {
        if(Character.isLetter(mes.charAt(i)))
        {
            int xx=template.indexOf(mes.charAt(i));
            int crypt=(xx*sk2Inverse)%26;
            answer=answer+template.charAt(crypt);
        }
        else
            answer=answer+mes.charAt(i);
    }
    return answer;
}
```

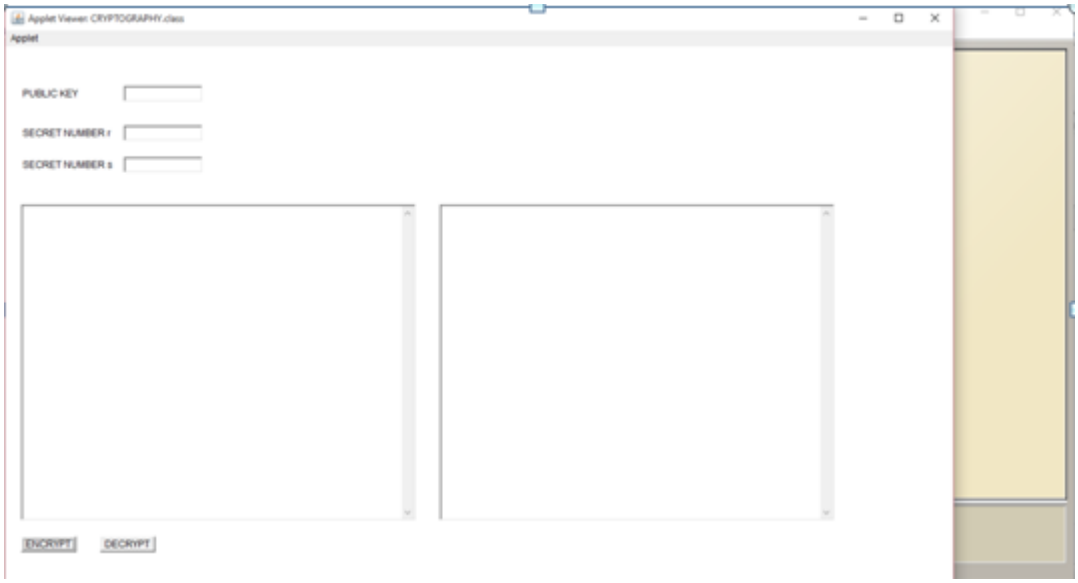
```
int modInverse(int a, int m)
{
    a = a%m;
    int ans=0;
    for (int x=1; x<m; x++)
    {
        if ((a*x) % m == 1)
        {
            ans=x;
            break;
        }
    }
    return ans;
}
public boolean action(Event e,Object j)
{
    if(e.target instanceof Button)
    {
        if(j.equals("ENCRYPT"))
        {
            //String message="message";
            int r=Integer.valueOf(secretNumber.getText());
            int s=Integer.valueOf(secretNumber1.getText());
            int a=Integer.valueOf(publicKey.getText());

            String message1=message.getText();
```

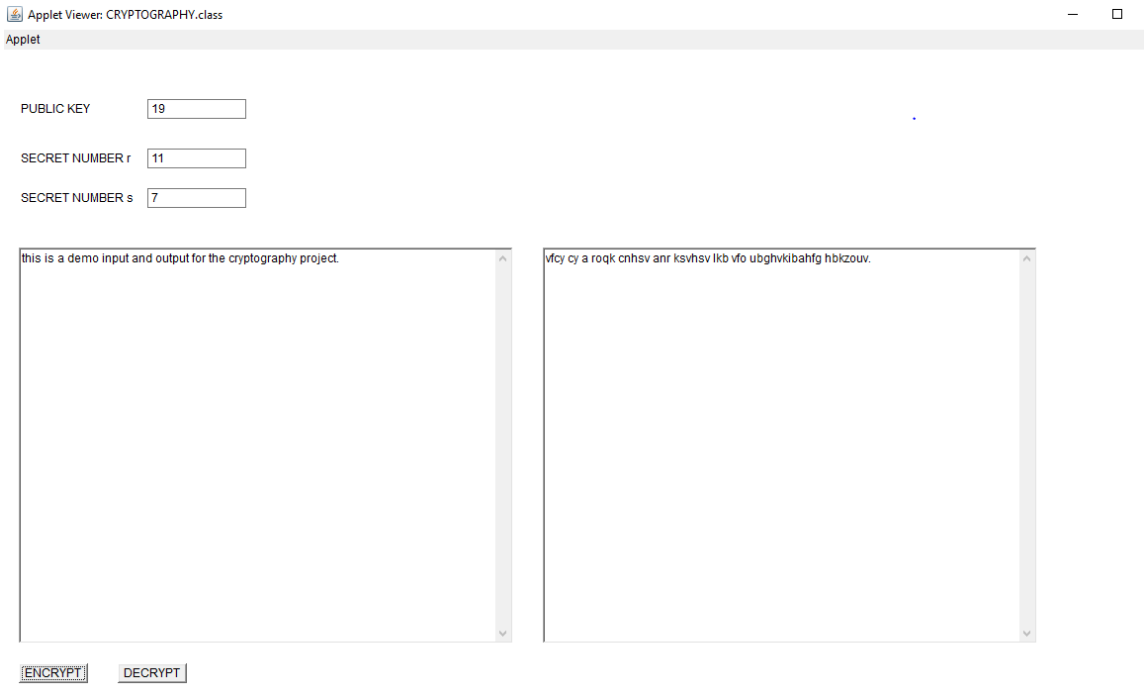
```
            String em=encrypt(r,s,a,message1);
            encryptedMessage.setText(em);
        }
        if(j.equals("DECRYPT"))
        {
            //String message="message";
            int r=Integer.valueOf(secretNumber.getText());
            int s=Integer.valueOf(secretNumber1.getText());
            int a=Integer.valueOf(publicKey.getText());
            String message1=encryptedMessage.getText();
            String em=decrypt(r,s,a,message1);
            message.setText(em);
        }
    }
    return true;
  }
}
```

3. Results and Discussions:

GUI CREATED :



OUTPUT :



4. Conclusion:

THE CODE RAN SUCCESSFULLY GENERATING THE REQUIRED OUTPUT.

5. REFERENCES:

- www.wikipedia.com
- www.google.com
- www.youtube.com