PES University, Bengaluru

VI Sem

Session: January – May, 2020

CRYPTOGRAPHY PROJECT

UE17EC341

"IMPLEMENTATION OF RSA ALGORITHM TO ENCRYPT AND DECRYPT IMAGE"

Submitted by

6th B

PRATHIMA CHOWDARY(PES1201700776)

SHREYA V DEEXIT(PES1201701648)

V SAISRI(PES1201701763)

INTRODUCTION:

Image security is an utmost concern in case web attacks become more serious. The Image encryption and decryption has applications in internet communication, military communication, medical imaging, multimedia systems, telemedicine, etc. To make the data secure from various attacks the data must be encrypted before it is transmitted. The government, financial institution, military, hospitals are deals with confidential images about their patient, financial status, geographical areas, enemy positions. Most of this information is now collected and stored on electronic computers and transmitted over the network. If these all the confidential images about enemy positions, patient and geographical areas are get in the wrong hands such a security could lead to declination of war, wrong treatment etc. Protecting the confidential images is the legal requirement. So has to make a strong encryption for an image so that it can't be hacked easily. And the perfection in the original image can be obtained after decrypting it.

RSA is an algorithm which is used to provide the encryption and authentication system. This was developed in 1977 by Ron Rivest, Adi Shamir, and Leonard Adleman. This algorithm is the most commonly used encryption and authentication algorithm. The RSA algorithm is one of the first public key cryptosystems, and it is widely used for secure data transmission. In such a cryptosystem, the encryption key is a public one and the decryption key differs which is kept secret. In RSA, this asymmetry is based on the product of two large prime numbers, the factoring problem. The RSA encrypt key encrypts the image, so that it converts into cipher text format and it will be stored as a text file. The opposite method of encryption, the reverse process is computed by another one decryption key of RSA algorithm and it decrypts the image from the cipher text. Finally it will discover the resultant image by the decryption techniques.

Asymmetric key cryptography:

Asymmetric cryptography is used for encryption and decryption algorithm pairs. With public key cryptography, keys work in pairs of matched public and private keys. Public key cryptography, also called asymmetric key cryptography which is using a pair of keys for encryption and decryption. With public key cryptography, keys work in pairs of matched public and private keys. The cryptography technique is using the secret message transfer from one place to another place over the networks. The cryptography technique requires some algorithms for encrypting the data.

RSA algorithm comes under asymmetric cryptography

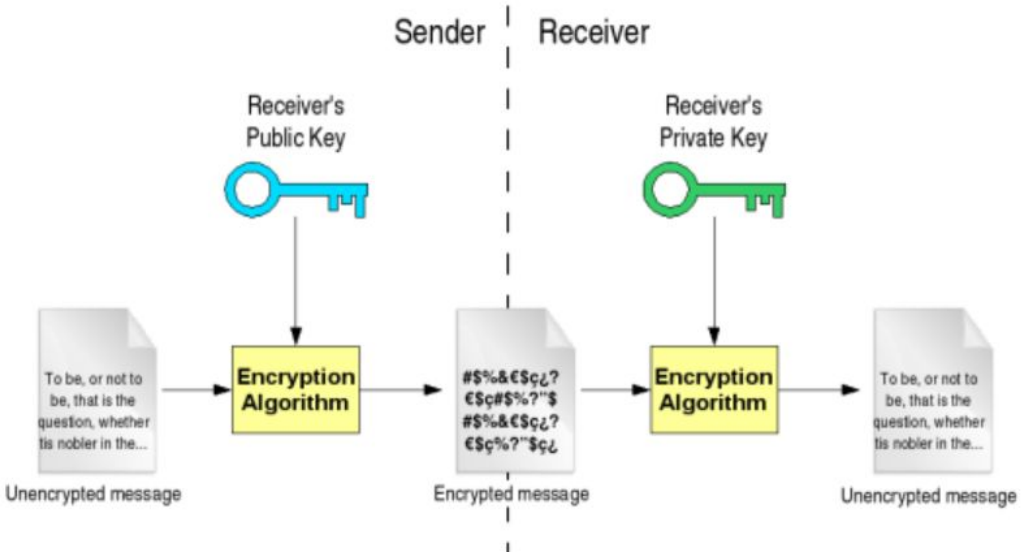
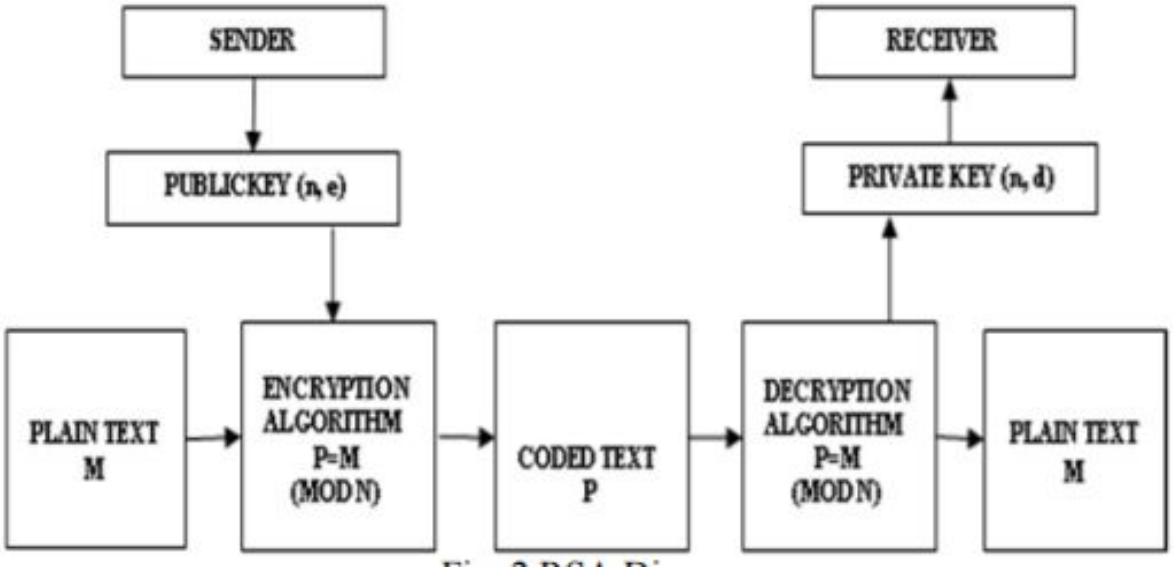


IMAGE CRYPTOGRAPHY METHODOLOGY BY RSA:

The RSA is a cryptographic algorithm which is used to encrypt and decrypt the data. This algorithm was developed in 1977 by Ron Rivest, Adi Shamir, and Leonard Adleman. RSA cryptosystem is also known as the public-key cryptosystem. RSA is normally used for secure data transmission. The encryption is starting on the RSA algorithm with the selection of two large prime numbers, along with an auxiliary value, as the public key. The prime numbers are kept secret. The public key is used to encrypt a message, and the private key is used to decrypt a message or information. The RSA algorithm is encrypt the original image and decrypts the image by the different keys



RSA ALGORITHM:

RSA is an algorithm used in the modern computer environment to encrypt and decrypt the data in transform. The RSA algorithm is also called an asymmetric cryptographic algorithm. Asymmetric cryptosystem means two different keys are used in the encryption and decryption. In the two keys one key is used for encryption and the second key is used for decryption. This RSA algorithm is also called as the public key cryptography. Because one of the secret keys can be given to everyone which means public. The other key must be kept private. The RSA algorithm consists of three major steps in encryption and decryption. The steps are following as,

1) Key Generation

2) Encryption

3) Decryption

A. Key generation

The key generation is the first step of RSA algorithm. The RSA involves a public key and a private key. On those keys the public key can be known to everyone and it is used for encrypting messages. Messages encrypted with the public key can decrypt using the private key. The keys for the RSA algorithm is generated by the following steps,

1) First choose the two distinct prime numbers p and q.

2) For security purposes, the integers p and q should be chosen, and it should be the similar bit-length. Prime integers can be efficiently found by primality testing.

3) Then compute the n value, n = pq.

4) n is used as the modulus for both the public and private keys. Its length, usually expressed in bits, is the key length.

5) Compute $\varphi(n) = \varphi(p)\varphi(q) = (p - 1)(q - 1) = n - (p + q - 1)$, where φ is Euler's totient function. This value is kept private.

6) Choose an integer e such that $1 < e < \varphi(n)$ and $\gcd(e, \varphi(n)) = 1$; i.e., e and $\varphi(n)$ are co-prime. e is then released as the public key. e has a short bit-length and small Hamming weight results in more efficient encryption. However, much smaller values of e have been shown to be less secure in some settings.

7) Determine d as $d \equiv e^{-1} \pmod{\varphi(n)}$; i.e., d is the modular multiplicative inverse of e (modulo $\varphi(n)$). This is stated as, solve the d given $d \cdot e \equiv 1 \pmod{\varphi(n)}$. This is computed using the extended Euclidean algorithm. Using the pseudo code in the Modular integers section, inputs a and n correspond to e and $\varphi(n)$, respectively.

8) d value is kept as the private key. The public key consists of the modulus n and the public key e. The private key has the modulus n and the private key d, and it is kept secret. p, q, and $\varphi(n)$ values are kept secret, because they can be used to calculate d.

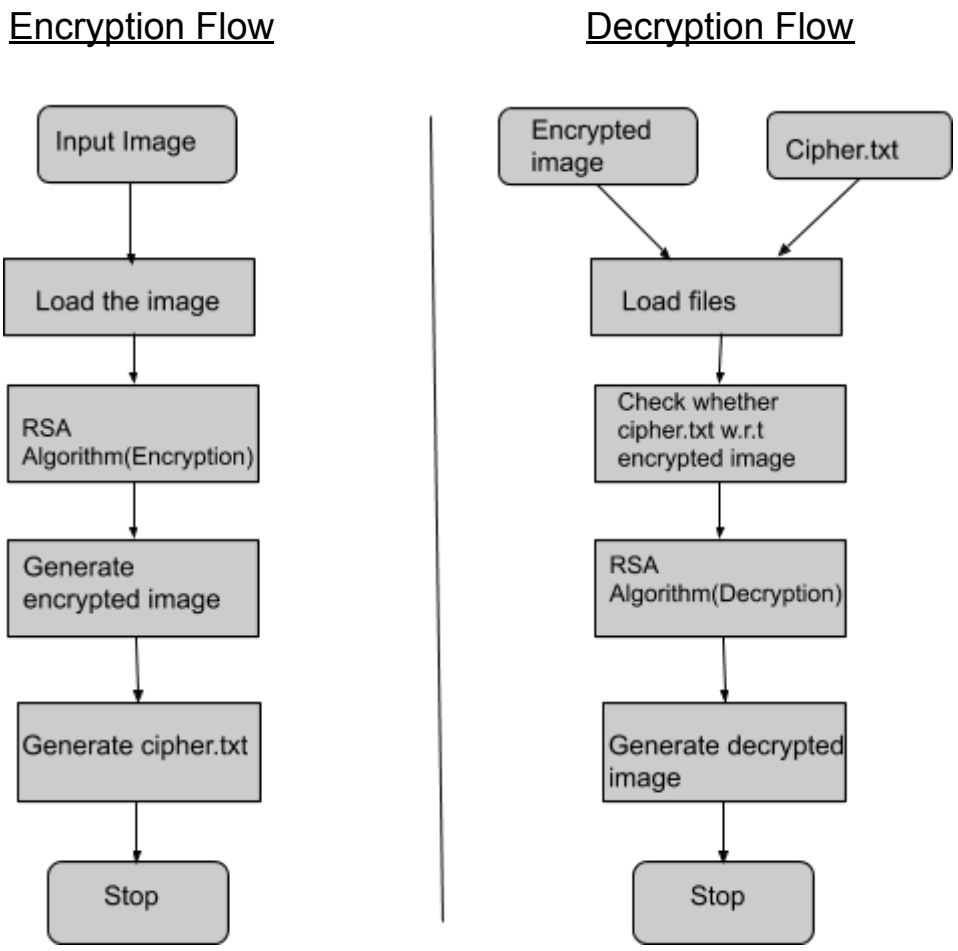
B. Encryption

Alice transmits her public key (n, e) to Bob and keeps the private key d secret. Bob then wishes to send the message M to Alice. So, first turns M into an integer m, such that $0 \leq m < n$ and $\gcd(m, n) = 1$. Then it computes the cipher text c. This can be done efficiently, even the numbers are 500- bit numbers, it is using the Modular exponentiation. Bob then transmits c to Alice. At least nine values of m will yield a cipher text c equal to m.

C. Decryption

Alice can recover m from c by using her private key exponent d via computing. Given m, she can recover the original message M by reversing the padding scheme.

BLOCK DIAGRAM:



CODES:

Encryption(At the sender):

```
from tkinter import *
from tkinter.filedialog import askopenfile
from PIL import Image
import random
import os,sys
import numpy
top = Tk()
top.geometry("200x200")

def k():
    file = askopenfile(mode ='r', filetypes =[('Python Files', '*.*')])
    print(file.name)
    img=Image.open(file.name)
    row,col = img.size
    pixels=img.load()
    row1 = 1000003
    phi = [0 for x1 in range(row1)]
    occ = [0 for x1 in range(row1)]
    primes = []
    phi[1] = 1

    for i in range(2,1000001):
            if(phi[i] == 0):
                    phi[i] = i-1
                    primes.append(i)
                    #j = 2*i
                    for j in range (2*i,1000001,i):
                            if(occ[j] == 0):
                                    #print ("inside if2")
                                    occ[j] = 1
                                    phi[j] = j
                                    #print (phi[j])

                                    phi[j] = (phi[j]*(i-1))//i
                                    #print(phi[j])
                                    #j = j + i

    p = primes[random.randrange(1,167)]
    q = primes[random.randrange(1,167)]
```

```
    print(p," ", q)
    n = p*q
    mod = n
    phin1 = phi[n]
    print("phin1",phin1)
    phin2 = phi[phin1]
    print("phin2",phin2)
    e = primes[random.randrange(1,9000)]
    print("e",e)
    mod1 = phin1
    def power1(x,y,m):
            ans=1
            while(y>0):
                    if(y%2==1):
                            ans=(ans*x)%m
                    y=y//2
                    x=(x*x)%m
            return ans
    d = power1(e,phin2-1,mod1)
    enc = [[0 for x in range(row)] for y in range(col)]
    dec = [[0 for x in range(row)] for y in range(col)]
    for i in range(col):
            for j in range(row):
                    r,g,b = pixels[j,i]
                    r1 = power1(r+10,e,mod)
                    g1 = power1(g+10,e,mod)
                    b1 = power1(b+10,e,mod)
                    enc[i][j] = [r1,g1,b1]
    print (pixels[row-1,col-1])
    ik=numpy.array(enc)
    img = numpy.array(enc,dtype= numpy.uint8)
    print(ik.shape)
    print("-----------------------------------------------")
    print("img=",img)
    print("-----------------------------------------------")
    #print("ik",len(ik))
    a_file = open(r"C:\Users\Shreya\Desktop\test1.txt", "w")
    for r in ik:
        numpy.savetxt(a_file, r)
```

```
a_file = open(r"C:\Users\Shreya\Desktop\test1.txt", "w")
for r in ik:
    numpy.savetxt(a_file, r)

a_file.close()
img1 = Image.fromarray(img,"RGB")
I = numpy.asarray(img1)
print("-----------------------------------------------")
print("i=",I)
print("-----------------------------------------------")
print("ENcryption done!!")
img1.show()
img1.save(r"C:\Users\Shreya\Desktop\enc.jpg")
original_array = numpy.loadtxt(r"C:\Users\Shreya\Desktop\cipher.txt").reshape(col,row,3)
h=list(original_array)
j=numpy.array(h,dtype= numpy.uint8)
```

Decryption(At the receiver):

```
import tkinter as tk
from tkinter import *
from tkinter.filedialog import askopenfile
from PIL import Image
import random
import os,sys
import numpy
import cv2 as cv
import imageio
root= tk.Tk()
canvas1 = tk.Canvas(root, width = 300, height = 300)
canvas1.pack()
label1 = tk.Label(root, text= "choose the image", fg='green', font=('helvetica', 12, 'bold'))
canvas1.create_window(150, 150, window=label1)

file2 = askopenfile(mode ='r', filetypes =[('Python Files', '*.*')])
print(file2.name)
img=Image.open(file2.name)
row,col = img.size

canvas2 = tk.Canvas(root, width = 300, height = 300)
canvas2.pack()
label2 = tk.Label(root, text= "choose the text file", fg='green', font=('helvetica', 12, 'bold'))
canvas2.create_window(150, 150, window=label2)

file1 = askopenfile(mode ='r', filetypes =[('Python Files', '*.*')])
original_array = numpy.loadtxt(file1.name).reshape(col,row,3)
nu=list(original_array)
h=list(original_array)
j=numpy.array(h,dtype= numpy.uint8)
op = numpy.asarray(img)

comparison = j[0][0] == op[0][0]
equal_arrays = comparison.all()
if(equal_arrays):
    print("recieved text and image file are matching")

row1 = 1000003
phi = [0 for x1 in range(row1)]
occ = [0 for x1 in range(row1)]
primes = []
```

```
phi[1] = 1
for i in range(2,1000001):
    if(phi[i] == 0):
        phi[i] = i-1
        primes.append(i)
        #j = 2*i
        for j in range (2*i,1000001,i):
            #print("j ",j)
            #print(j)
            if(occ[j] == 0):
                #print ("inside if2")
                occ[j] = 1
                phi[j] = j
                #print (phi[j])
            phi[j] = (phi[j]*(i-1))//i


p = 409
q = 677
print(p," ", q)

n = p*q
mod = n

phin1 = phi[n]
print("phin1",phin1)
phin2 = phi[phin1]
print("phin2",phin2)
e = 45247
print("e",e)
mod1 = phin1
dec = [[0 for x in range(row)] for y in range(col)]
def power1(x,y,m):
    ans=1
    while(y>0):
        if(y%2==1):
            ans=(ans*x)%m
        y=y//2
        x=(x*x)%m
    return ans
d = power1(e,phin2-1,mod1)


for i in range(col):
        for j in range(row):
                r,g,b = nu[i][j]
                r1 = power1(r,d,mod)-10
                g1 = power1(g,d,mod)-10
                b1 = power1(b,d,mod)-10
                dec[i][j] = [r1,g1,b1]
img2 = numpy.array(dec,dtype = numpy.uint8)
img3 = Image.fromarray(img2,"RGB")
img3.show()
root.mainloop()
```

APPLICATIONS OF IMAGE CRYPTOGRAPHY:

Core banking is a set of services provided by the group of networked bank branches. Bank customers may access their funds and perform the simple transactions from the member branch offices. The major issue in core banking is the authenticity of the customer. An unavoidable hacking of the databases on the Internet, it is always quite difficult to trust the information on the Internet. To solve this problem of authentication proposing an algorithm based on image processing and image cryptography. The internet multimedia applications are becoming popular. The valuable multimedia content such as the image is vulnerable to unauthorized access while in storage and during transmission over a network.

The image processing applications have been commonly found in the Military communication, Forensics, Robotics, Intelligent systems etc

MERITS AND DEMERITS OF IMAGE CRYPTOGRAPHY:

A. Merits

One advantage to encryption is that it separates the security of data from the security of the device where the data is transmitted over the Internet. And the advantages to implementing encryption include the pain that comes with data breach disclosures, the provision of strong protection for intellectual property. The people should keep in mind the standard email is not secure and is in fact tantamount to writing sensitive information on postcards. The encrypted data that can only be read by a system or user who has the key to unencrypted the data means the system or user is authorized to read the data. Encrypted data cannot be accessed by the third parties. The encryption comes with the numerous advantages that need to protect the data. And some other benefit is there in using Image Cryptography. There are,

1) Peace of Mind 2) Identity Theft Protection 3) Safe Decommissioning of Computer 4) Unauthorized Access Protection 5) Compliance with Data Protection Acts

B. Demerits

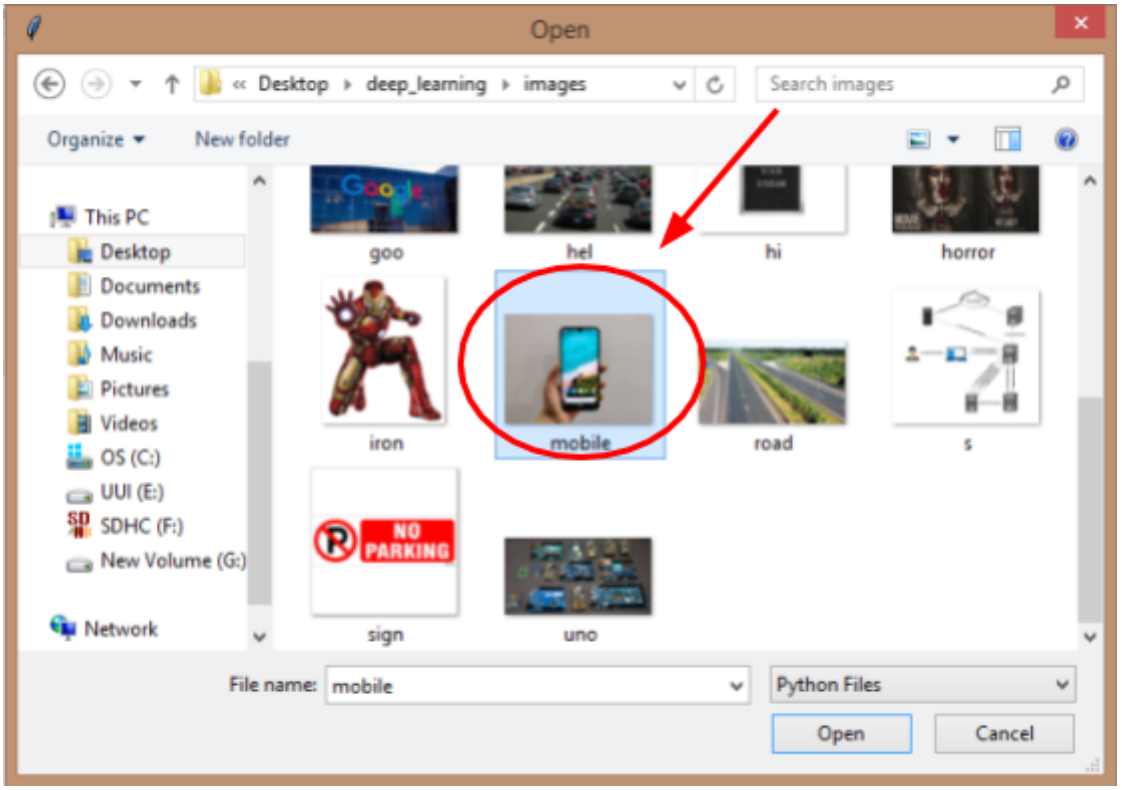
Encryption is a very complex technology. One big disadvantage of encryption is related with keys are that the security of data becomes the security of the encryption key. The data is lost effectively if it loses the keys. Encrypting data and creating the keys necessary to encrypt and decrypt the data is computationally expensive. The systems performing are heavy and take the available resources in computation. One of the common drawbacks of traditional full-disk encryption solutions are reduction of overall performance of the system deployment key pitfall is that a poor encryption implementation could result in the false of security when in fact it is wide open to attack.

RESULTS:

Encryption:

Tkinter interface for encryption

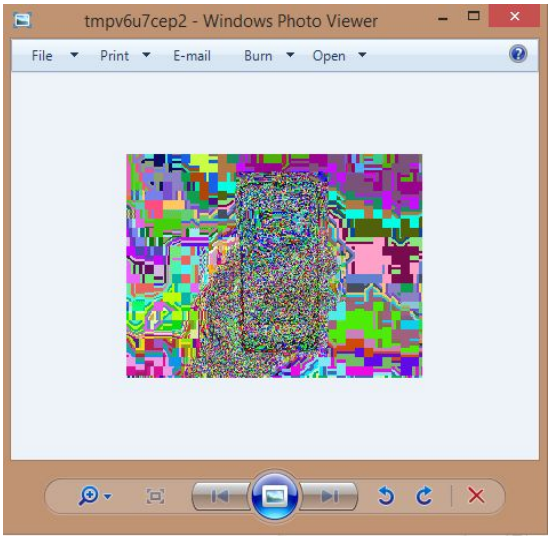
The user needs to press the encrypt button and then the browsing screen pops up.Hence the user can select any image from his desktop.



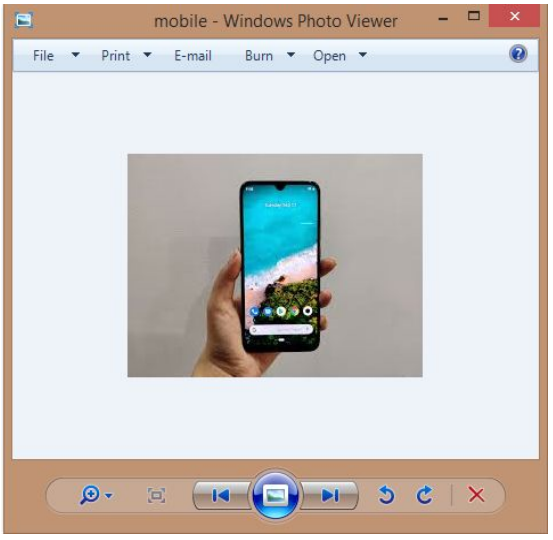
Terminal output:

```
================== RESTART: C:\Users\Shreya\Desktop\encrypt.py ==================
C:/Users/Shreya/Desktop/deep_learning/images/mobile.jpg
911   101
phin1 91100
phin2 45550
e 20201
(168, 160, 158)
(194, 259, 3)
```

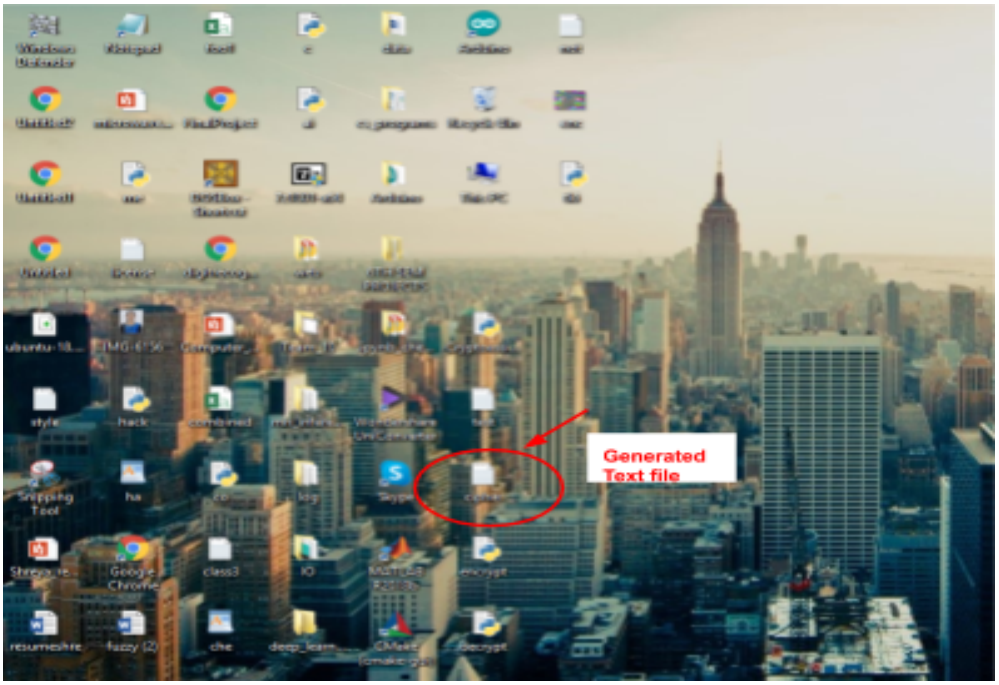
Encrypted image:



Original image:



While encrypting the image there is a text file generated.Hence the sender has to send the text file and image while sending it to the receiver



Decryption:

Tkinter interface for decryption

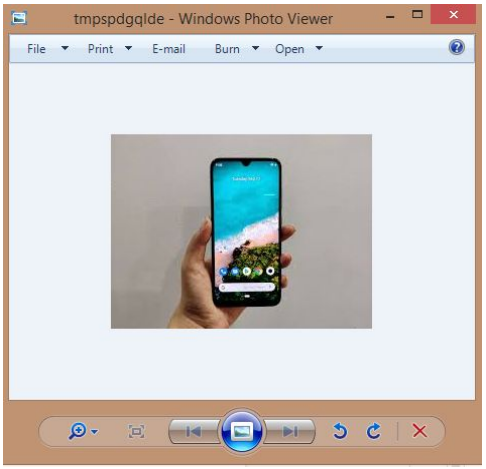
At first the “choose image” screen pops up,then the user has to choose the image to be decrypted.

Then the “choose the text file” screen pops up,and the user has to choose the text file.

Terminal output:

```
C:/Users/Shreya/Desktop/enc.jpg
recieved text and image file are matching
911   101
phin1 91000
phin2 28800
e 20201
decryption done
```

Decrypted image:



CONCLUSION:

For this, an experimental with the different raw images with the different sizes are encrypted and decrypted. The cryptography mechanism using the RSA algorithm with the public key encryption is to increase the security levels of the encrypted. Here one key is needed to encrypt and another key is needed to decrypt the image. Finally the image cryptography experiment is providing the feasibility of security to the image in network security. The data is not viewed by no one without the knowledge of cryptography.

The image consists of a secret and it is going to be encrypted. It is called as an original image that may contain the data.

The Original image is encrypted by the key which is generated by the RSA algorithm. It is converting the image into the cipher text.

Finally the cipher text is decrypted by another one decrypt key which is also generated by the RSA algorithm. And it converts the cipher text into the resultant image.

DISCUSSION:

In the digital world, the security of images has become more important as the communication has increased rapidly. All the techniques are in a real-time image encryption could only find a low level of security. Here, the image encryption algorithm proposed is efficient and highly securable with high level of security and less computation. The results of the simulation show that the algorithm has advantages based on their techniques which are applied on images. Hence it is concluded that the techniques are good for image encryption and give security in the open network.

FUTURE ENHANCEMENT:

With digital video transmission, encryption methodologies are needed that can protect digital video from attacks during transmission. So the developed algorithms can be applied on video images and web based systems

REFERENCES:

- Payal Sharma, Manju Godara, Ramanpreet Singh, Digital Image Encryption Techniques: A Review, International Journal of Computing & Business Research ISSN (Online): 2229-6166.
- Komal D Patel , Sonal Belani, Image Encryption Using Different Techniques: A Review International Journal of Emerging Technology and Advanced Engineering, ISSN 2250-2459, Volume 1, Issue 1, November 2011.
- Ambika Oad, Himanshu Yadav, Anurag Jain, A Review: Image Encryption Techniques and its Terminologies, International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249 – 8958, Volume-3, Issue-4, April 2014.

THANK YOU