PROJECT REPORT

NETWORK AND INFORMATION SECURITY

IMAGE ENCRYPTION USING RSA ALGORITHM WITH IMAGE STEGNOGRAPHY

Team Members:-

S.No.	Name	Reg.No
1.	Hari Rajesh	17BIT0155
2.	Naga Siva	17BIT0203
3.	Ravi Sankar	17BIT0304

Under the guidance of P. Shantarajah Sir

1.ABSTRACT:-

The increasing usage and dependency on Internet have made security of data a priority and a reason of concern. Cryptographic technique is one of the principal means to protect information security. Not only has it to ensure the information confidential, but also provides digital signature, authentication, secret sub-storage, system security and other functions. Therefore, the encryption and decryption solution can ensure the confidentiality of the information, as well as the integrity of information and certainty, to prevent information from tampering, forgery and counterfeiting. Encryption and decryption algorithm's security depends on the algorithm while the internal structure of the rigor of mathematics, it also depends on the key confidentiality.

For transfer of confidential medical data, we are encrypting the medical data with RSA algorithm and then hiding the encrypted image along with a cover image and sending it to receiver. The receiver will first extract the secured medical data then decrypt it to get original image.

2.INTRODUCTION:-

Encryption is one of the principal means to grantee the security of sensitive information. It not only provides the mechanisms in information confidentiality, but also functioned with digital signature, authentication, secret sub-keeping, system security and etc. Therefore, the purpose of adopting encryption techniques is to ensure the information's confidentiality, integrity and certainty, prevent information from tampering and forgery.

We are using most secure asymmetric encryption algorithm RSA to encrypt the image.

3.LITERATURE SURVEY

[1] Xin Zhou: Cryptographic technique is one of the principal means to protect information security. Encryption and decryption algorithm's security depends on the algorithm while the internal structure of the rigor of mathematics, it also depends on the key confidentiality. This paper proposed an implementation of a complete and practical RSA encrypt/decrypt solution based on the study of RSA public key algorithm. RSA's security depends on the difficulty of integer factorization, whether it is equivalent to integer factorization has not been fully proved in theory because there is no evidence providing that to cracking RSA would definitely require for making large numbers factorization.

[2] Ramadhan J. Mstafa: Steganography is a technique that prevents unauthorized users to have access to the important data. The steganography and digital watermarking provide methods that users can hide and mix their information within other information that make them difficult to recognize by attackers. Digital watermarking is one of the most widely used applications for steganography technique. Watermarking hides information in a digital signal. Least Significant Bit Watermarking LSB is the one of the oldest and simplest algorithms that allows users to hide their information using spatial domain. The human eye cannot recognize the difference that occurs in the two first bits in each pixel. In other words, the change in the least significant bit does not affect the image's quality. For example, two pixels of an RGB image color will provide six bits for watermarking. To encode a message (100111) in RGB image needs two LSB pixels [13; 31].

RGB Pixel 1 (R: 00010101 G: 11001100 B: 11101100)

RGB Pixel2 (R: 11011111 G: 00010001 B: 11001001)

To hide the same message (100111) in a grey-scale image six LSB pixels are needed.

Pixel1: 10010101 Pixel2: 00001100 Pixel3: 11001000

Pixel4: 10011111 Pixel5: 00010001 Pixel6: 11001011

[3] Shubhi Mittal: The increasing usage and dependency on Internet has made security of data a priority and a reason of concern. The paper had taken into consideration RSA algorithm for encryption and image steganography for data hiding using LSB technique. In this paper they analyse the degree of difference

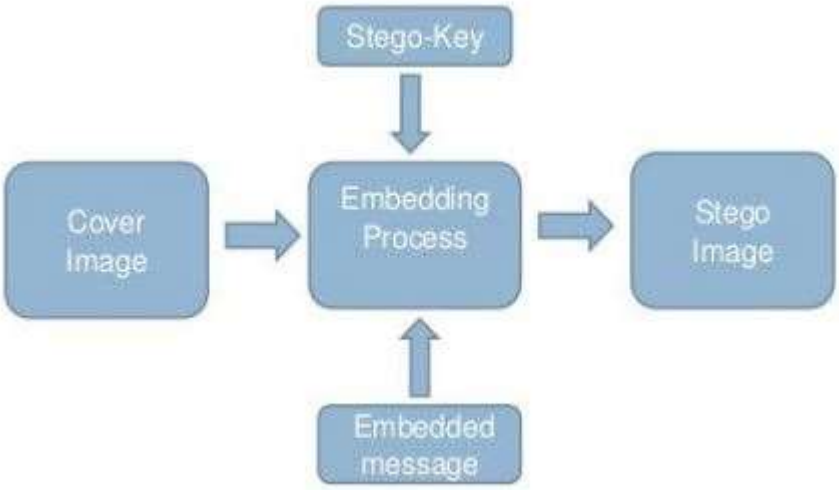
when steganography and cryptography are implemented separately and compare them on various parameters when both are implemented simultaneously for providing dual layer protection. The algorithm intends of implementing LSB technique and RSA data security algorithm, the most imperative thing is to ensure that the size of original image and stego image must be equivalent and same goes for plain text and cipher text used in RSA.

4.METHODOLOGY

4.1 RSA Algorithm:-

1. Selecting two large primes at random 'p', q'
2. computing their system modulus n = p.q $\phi(n) = (p-1)(q-1)$
3. selecting at random the encryption key e where $1<e< \phi(n)$, $gcd(e,\phi(n)) =1$ Select e , such that e is relatively prime to $\phi(n)$
4. solve following equation to find decryption key d $e.d \equiv 1 \mod \phi(n)$ and $0 \leq d \leq n$ ($d \equiv e -1 \mod \phi(n)$)
5. publish their public encryption key : KU={e,n}
6. keep secret private decryption key : KR={d,p,q}

4.2 Steganography:-

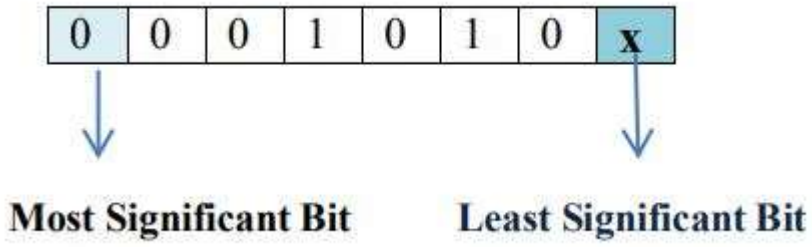


In image steganography a cover image is converted into a stego image. A cover image is an image that is used as an innocent looking tool for communicating secret information, that information or message is embedded into the cover

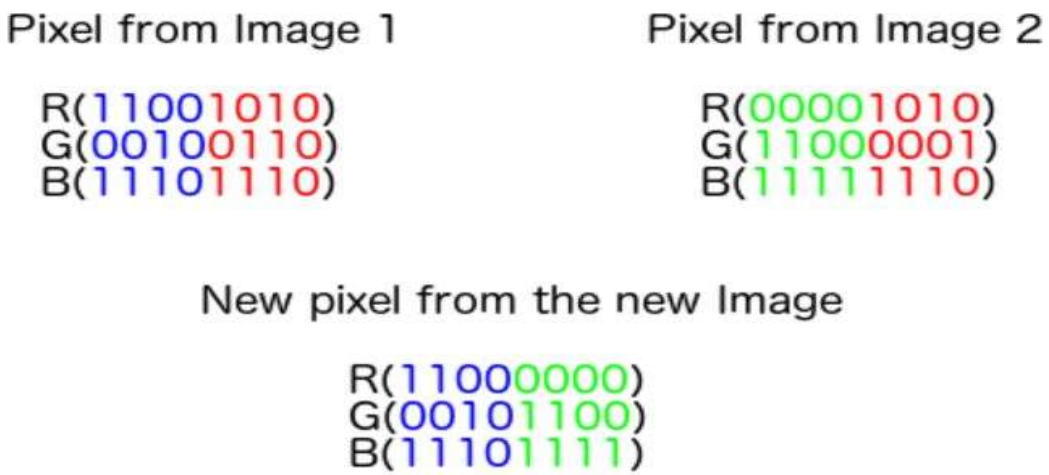
image which is then called a stego image. It is a preferred technique as the image used as a media is less susceptible to detection as exchanging images has become a very common phenomenon. Also, the text messages hidden in the images cause little distortion as a single bit of the entire byte is altered and image has multiple redundant bits which make it easier to hide the information in it.

4.3 LSB (Least Significant Bit):-

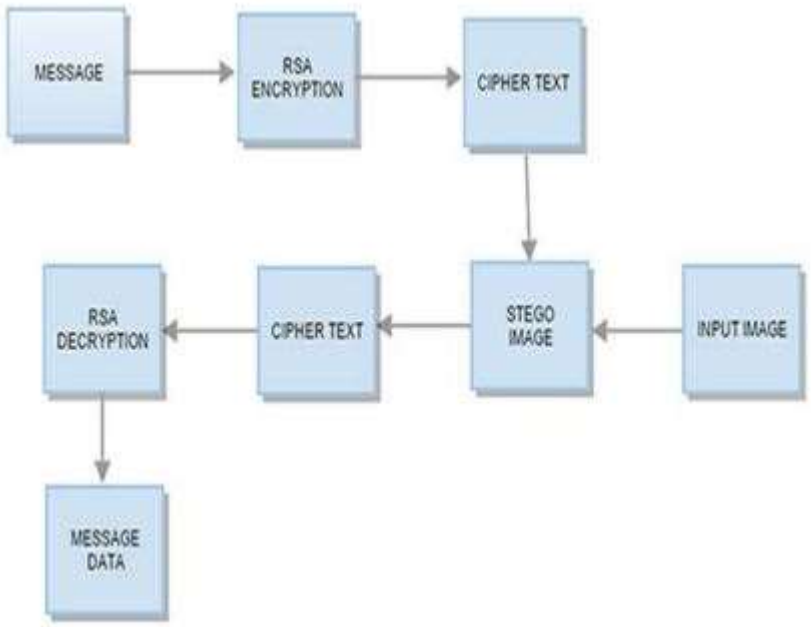
LSB i.e. least significant bit technique is a simple and effective technique that can be used for implementing image steganography. The technique replaces the least significant bit (the right-most bit) or the 8th bit of the byte with the bit of the information that is intended to be embedded in the image.



In the additive colour model 3 primary RGB (Red, Green, and Blue) colours are combined in various proportions in order to make multiple different colours for example, on adding green and blue



4.4 Working model:-



1. Encrypt the image using RSA algorithm
2. Merge the encrypted image with cover image using steganography
3. Send the stego image to the receiver.
4. Receiver would extract the encrypted image from stego image.
5. Receiver decrypts encrypted image using his private key.
6. Receiver views original image.

5.PLATFORM:-

Software Requirements:

1. Python 3.7.3
2. OpenCV Library
3. PIL(Python Imaging Library)

6.Code:-

Input images first is cover image second is image to be transmitted.


```
#RSA algorithm
import random
import os,sys
import numpy
from PIL import Image
jpgfile = Image.open("img2.jpg")
#jpgfile.show()
print (jpgfile.bits, jpgfile.size, jpgfile.format)
```

```
row,col = jpgfile.size
pixels = jpgfile.load()
primes=[]
def getprimes(n):
    for num in range(2,n+ 1):
        i = 2
        for i in range(2, num):
            if(num % i == 0):
                i = num
                break;
    # If the number is prime then print it.
        if(i != num):
            primes.append(num)
import random
getprimes(100)

p = primes[random.randrange(1,24)]
q = primes[random.randrange(1,24)]
print("choosen prime numbers are:")
print (" p=",p," q=", q)
n = p*q
mod=n
phi=(p-1)*(q-1)
e=17
def modInverse(a, m) :
    a = a % m;
```

```
    for x in range(1, m) :
        if ((a * x) % m == 1) :
            return x
    return 1
d=modInverse(e,phi)
print("Public Key",phi,e)
print("Private Key",p,q,d)
def encrypt(pt,e,n):
    return pow(pt,e)%n
#Encrypting the image to be sent using RSA algorithm
enc = [[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 = encrypt(r,e,n)
        g1 = encrypt(g,e,mod)
        b1 = encrypt(b,e,mod)
        enc[i][j] = [r1,g1,b1]
print (pixels[row-1,col-1])
img = numpy.array(enc,dtype = numpy.uint8)
img3 = Image.fromarray(img,"RGB")
img3.save("img4.jpg")

#Encrypted image which would be hidden in transmission
img3
```

```
(190, 181, 176)
```

Out[4]:

```
#Stegnography

from PIL import Image


def __int_to_bin(rgb):
    r, g, b = rgb
    return ('{0:08b}'.format(r),'{0:08b}'.format(g),'{0:08b}'.format(b))


def __bin_to_int(rgb):
    r, g, b = rgb
    return (int(r, 2),int(g, 2),int(b, 2))


def __merge_rgb(rgb1, rgb2):
    r1, g1, b1 = rgb1
    r2, g2, b2 = rgb2
    rgb = (r1[:4] + r2[:4],g1[:4] + g2[:4],b1[:4] + b2[:4])
    return rgb
```

```
def merge(img1, img2):
    # Check the images dimensions
    if img2.size[0] > img1.size[0] or img2.size[1] > img1.size[1]:
        raise ValueError('Image 2 should not be larger than Image 1!')
     # Get the pixel map of the two images
    pixel_map1 = img1.load()
    pixel_map2 = img2.load()

    # Create a new image that will be outputted
    new_image = Image.new(img1.mode, img1.size)
    pixels_new = new_image.load()

    for i in range(img1.size[0]):
        for j in range(img1.size[1]):
            rgb1 = __int_to_bin(pixel_map1[i, j])
            #Use a black pixel as default
            rgb2 = __int_to_bin((0, 0, 0))
            #Check if the pixel map position is valid for the second image
            if i < img2.size[0] and j < img2.size[1]:
                rgb2 = __int_to_bin(pixel_map2[i, j])
            #Merge the two pixels and convert it to a integer tuple
            rgb = __merge_rgb(rgb1, rgb2)
            pixels_new[i, j] = __bin_to_int(rgb)
    return new_image
```

```
img1=Image.open("img1.jpg")
img2=Image.open("img4.jpg")
transmit_img=merge(img1,img2)
transmit_img
```

```
def unmerge(img):
    """Unmerge an image.
    :param img: The input image.
    :return: The unmerged/extracted image.
    """

    # Load the pixel map
    pixel_map = img.load()

    # Create the new image and load the pixel map
    new_image = Image.new(img.mode, img.size)
    pixels_new = new_image.load()

    # Tuple used to store the image original size
    original_size = img.size
```

```
        for i in range(img.size[0]):
            for j in range(img.size[1]):
                # Get the RGB (as a string tuple) from the current pixel
                r, g, b = __int_to_bin(pixel_map[i, j])

                # Extract the last 4 bits (corresponding to the hidden image)
                # Concatenate 4 zero bits because we are working with 8 bit
                rgb = (r[4:] + '0000',
                       g[4:] + '0000',
                       b[4:] + '0000')

                # Convert it to an integer tuple
                pixels_new[i, j] = __bin_to_int(rgb)

                # If this is a 'valid' position, store it
                # as the last valid position
                if pixels_new[i, j] != (0, 0, 0):
                    original_size = (i + 1, j + 1)

        # Crop the image based on the 'valid' pixels
        new_image = new_image.crop((0, 0, original_size[0], original_size[1]))

        return new_image
transmit_img.save("transmitted.jpg")
transmit=Image.open("transmitted.jpg")
```

unmerge(transmit_img)

```
def decrypt(ct,d,n):
    return pow(ct,d)%n
#Now decrypt the image
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 = enc[i][j]
        r1 = decrypt(r,d,mod)
        g1 = decrypt(g,d,mod)
        b1 = decrypt(b,d,mod)
        dec[i][j] = [r1,g1,b1]
img5 = numpy.array(dec,dtype = numpy.uint8)
img6 = Image.fromarray(img5,"RGB")
img6.save("output.jpg")
```

7.References:

[1] Research and Implementation of RSA Algorithm for Encryption and Decryption Xin Zhou, Xiaofei Tang -2011 The 6th International Forum on Strategic Technology.

[2] Information Hiding in Images Using Steganography Techniques Ramadhan Mstafa ,Christian Bach

[3] PData Security using RSA Encryption Combined with Image Steganography, Shubhi Mittal, Shivika Arora, Rachna Jain