Secure Transmission by Signal Encryption and Encoding

PROJECT REPORT

Submitted for Course: Digital Signal Processing (ECE2006)

By

Abhas Mathur	16BEC0010
Sanskar Biswal	16BEC0403

Slot: C2

Faculty: Prof. Sudhakar M.S

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

November 2018

CERTIFICATE

This is to certify that the project work entitled “Secure Transmission by Signal Encryption and Encoding” that is being submitted by “ *GROUP-3*” DIGITAL SIGNAL PROCESSING (ECE2006) is a record of bonafide work done under my supervision. The contents of this Project work, in full or in parts, have neither been taken from any other source nor have been submitted for any other CAL course.

Place : Vellore
Date : 30/10/2018

Signature of Students:
Abhas Mathur (16BEC0010)
Sanskar Biswal (16BEC0403)

Signature of Faculty:

ACKNOWLEDGEMENTS

This report would not have been possible without the essential and gracious support of Prof.Sudhakar. His willingness to motivate us contributed tremendously to my report. We also would like thank him for showing me some examples related to the topic of report.

Besides we would like to thank the authorities of VIT, Vellore for providing us the positive environment and facilities to complete this project and report.

ABSTRACT

Encryption is a way to secure and verify data that are traded through public communication channels in the presence of intruder party called antagonists. Consequently, the transmitted or stored message can be converted to unreadable form except for intended receivers. The decryption techniques allow intended receiver to reveal the contents of previously encrypted message via secrete keys exchanged exclusively between transmitter and receiver. The encryption and decryption techniques can be applied equally to a message in any form such as text, image, audio or video.

The current scope of the project is limited to testing the method of RSA encryption on text data. The run-time analysis of this data is further conducted to analyze the efficiency of the algorithm being implemented.

The observations indicate that the time for execution is relatively low for data in the range of 2-256 bytes. Beyond that, an exponential increase in encryption time is observed. All simulations in the current project were done in MATLAB.

OBJECTIVE

- To develop a model of RSA encryption for data signals and convert them into transmittable data packets.
- Further analyze the *'time of encryption' v/s 'size of data'*.
- Draw conclusions on the relation between the observed time for encryption and size of the inbound data-stream.

METHODOLOGY

- The input message signal will follow a random scramble pattern for data rearrangement before transmission. On the receiver end, only the accurate key can be used to unscramble data. A double-key encryption method will be used.
- This method is termed as RSA cryptography algorithm.

RSA cryptography

1. RSA involves a public key and private key. The public key can be known to everyone; it is used to encrypt messages. Messages encrypted using the public key can only be decrypted with the private key. The keys for the RSA algorithm are generated the following way:
    1. Two large random prime nos. '*p*' and '*q*' are chosen.
    2. n = pq ; n is the modulus of
    3. ϕ(n) = (p-1)*(q-1) . Here ϕ(n) is called a totient.
    4. An integer 'e' such that ; 1<e< ϕ(n) is chosen. 'e' is the public key.
    5. For some integer k, 'd' is chosen ;
    6. d = (1+k ϕ(n)/e). 'd' is the private key

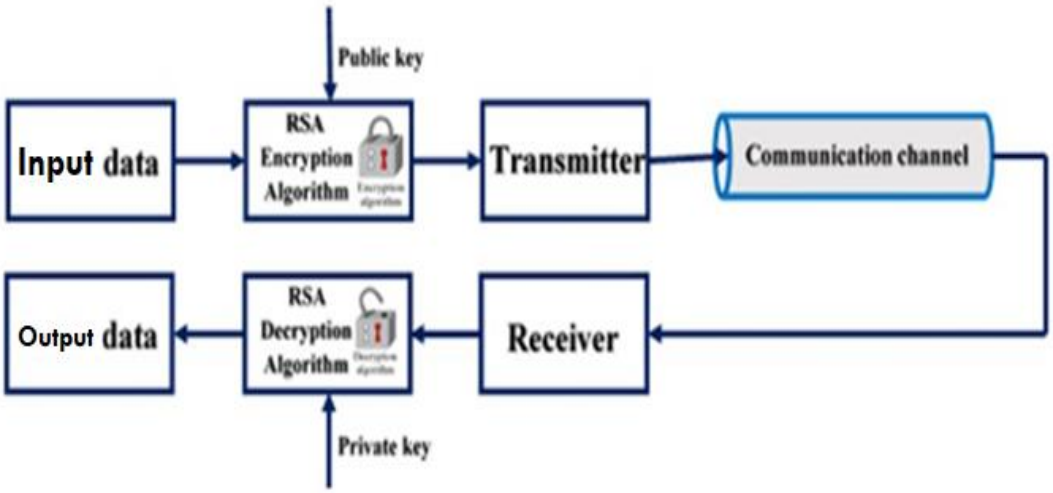
Encrypting Messages

1. Given (n,b) and (n,a) as computed above
2. To encrypt bit pattern, m, compute:
   $x = m^e \bmod(n)$
   (i.e., remainder when me is divided by n)
3. To decrypt received bit pattern, c, compute:
   $m = x^d \bmod(n)$
   (i.e., remainder when cd is divided

Decrypting Messages

1. At A, m can be retrieved by, $m = c^d \bmod n$. The private key 'd' is known by A.
2. The original prime nos. may be recovered by, $c^{ed} = m \bmod(pq)$
3. The message 'm' is reversed by the padding scheme and the message M is retrieved by A

THE BLOCK DIAGRAM



CODES

Three variants of Code were designed and tested.

- Custom built code for RSA Encryption and Decryption
- Use of Java Library for RSA Implementation
- Analysis of computational time for Java format of execution

Custom Code for RSA Implementation:

```
clc;
clear all;
disp('RSA algorithm');
p=input('Enter the prime no. for p: ');
q=input('Enter the prime no. for q: ');
n=p*q;
fprintf('\nn=%d',n);
phi=(p-1)*(q-1);
fprintf('\nphi(%d) is %d',n,phi);
val=0;
cd=0;
while(cd~=1||val==0)
n1=randint(1,1,n);
e=randint(1,1,[2 n1]);
 val=isprime(e);
 cd=gcd(e,phi);
 end
val1=0;
d=0;
while(val1~=1);
d=d+1;
val1=mod(d*e,phi);
end
fprintf('\nd=%d',d);
fprintf('\nPublic key is (%d,%d)',e,n);
fprintf('\nPrivate key is (%d,%d)',d,n);
m=input('\nEnter the message: ','s');
m1=m-0;
disp('ASCII equivalent of message ');
disp(m1);
over=length(m1);
 o=1;
```

```
 while(o<=over);
    m=m1(o);
    diff=0;
    if(m>n);
        diff=m-n+1;
    end
    m=m-diff;

qm=dec2bin(e);
len=length(qm);
c=1;
xz=1;
while(xz<=len)
    if(qm(xz)=='1')
      c=mod(mod((c^2),n)*m,n);
    elseif(qm(xz)=='0')
        c=(mod(c^2,n));
    end
    xz=xz+1;
end
c1(o)=c;
qm1=dec2bin(d);
len1=length(qm1);
nm=1;
xy=1;
while(xy<=len1)
    if(qm1(xy)=='1')
       nm=mod(mod((nm^2),n)*c,n);
    elseif(qm1(xy)=='0')
        nm=(mod(nm^2,n));
    end
     xy=xy+1;
end
nm=nm+diff;
nm1(o)=char(nm);
o=o+1;
 end


o=1;
fprintf('\nThe encrypted message is \n');
while(o<=over)
   fprintf('\t%d',c1(o));
   o=o+1;
```

```
end
o=1;
fprintf('\nThe decrypted mes in ASCII is \n');
while(o<=over)
fprintf('\t%d',nm1(o));
o=o+1;
end
fprintf('\nThe decrypted message is: ');
disp(nm1);
fprintf('\n');
```

Java Library Format Implementation of RSA

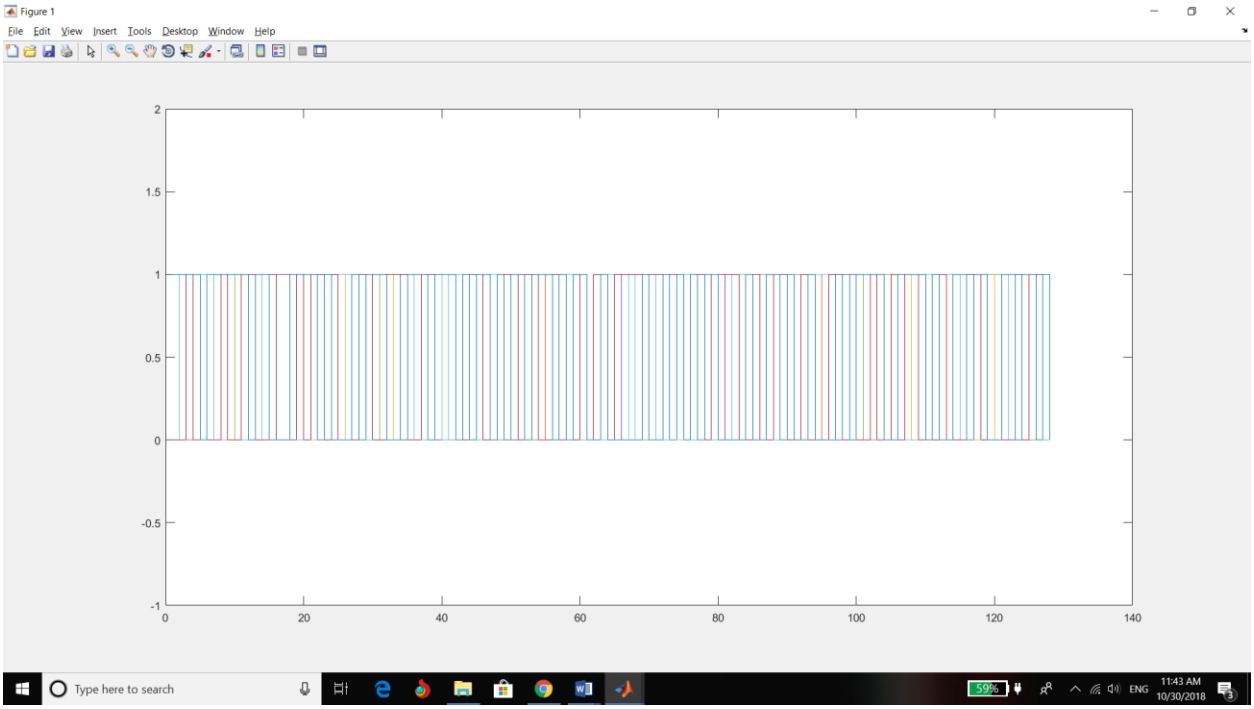
```
import javax.crypto.Cipher;


plaintext = 'Text Message Data';

cipher = Cipher.getInstance('RSA');
keygen = java.security.KeyPairGenerator.getInstance('RSA');
keyPair = keygen.genKeyPair();

%Encrypt
cipher.init(Cipher.ENCRYPT_MODE, keyPair.getPrivate());
plaintextUnicodeVals = uint16(plaintext);
plaintextBytes = typecast(plaintextUnicodeVals, 'int8');
ciphertext = cipher.doFinal(plaintextBytes);
bDAT = decimalToBinaryVector(typecast(ciphertext,'uint8'));
%DAT = decimalToBinaryVector(typecast(plaintext,'uint8'));

%Signal Processing
figure;
x1 = stairs(bDAT);
%x2 = stairs(DAT);
ylim([-1 2]);

% Decrypt
cipher.init(Cipher.DECRYPT_MODE, keyPair.getPublic());
decryptedBytes = cipher.doFinal(ciphertext);
decryptedText = char(typecast(decryptedBytes, 'uint16'))
```

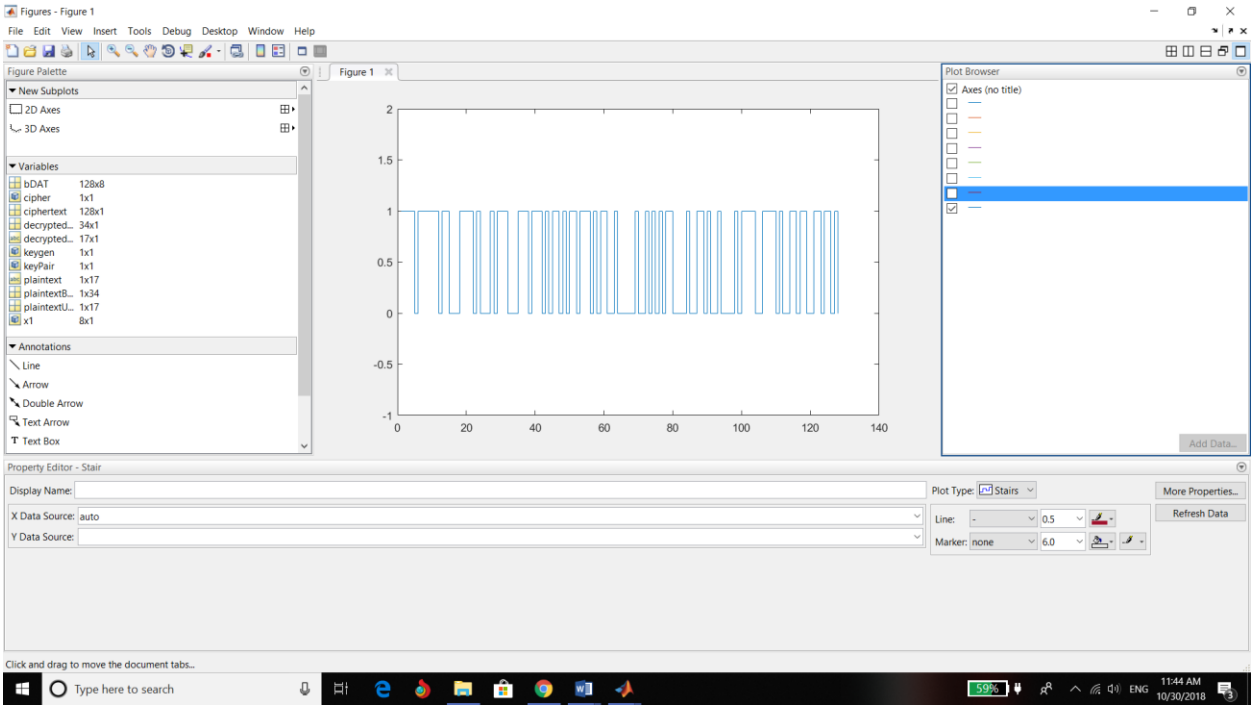
Execution Time Analysis on Encryption

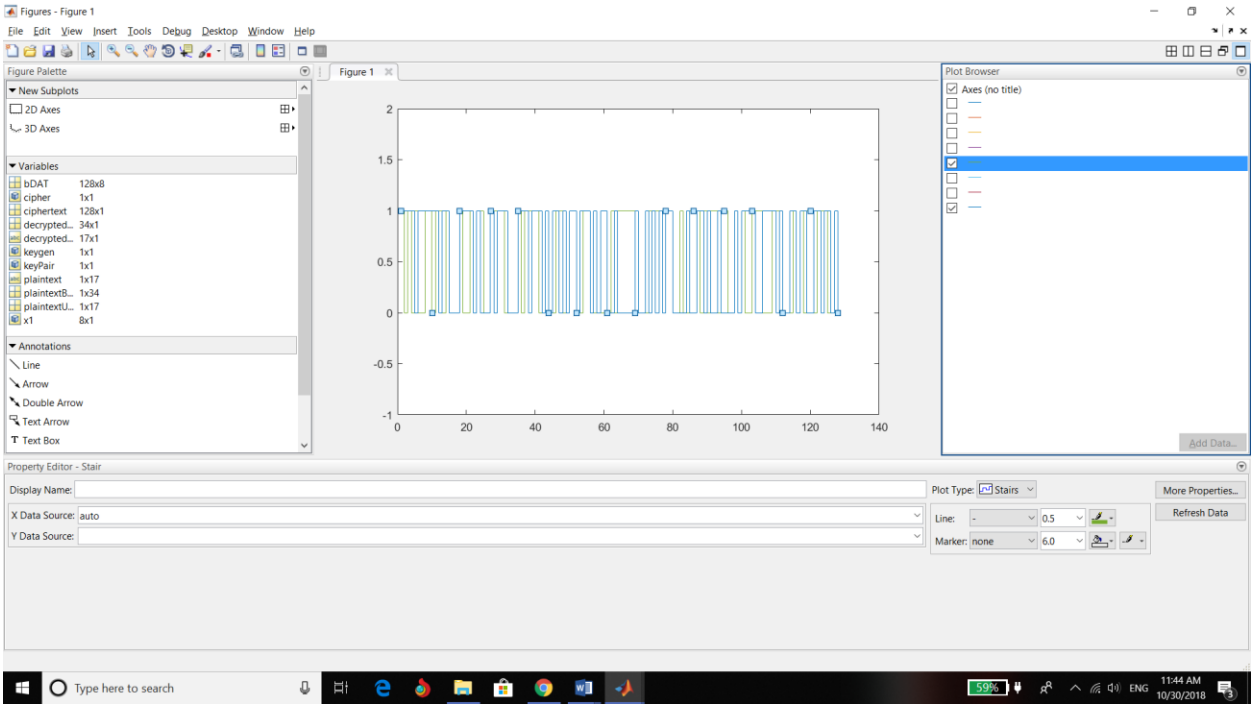
```
clc;
clear all;
close all;

import javax.crypto.Cipher
% Setup Cipher Modules
cipher = Cipher.getInstance('RSA');
keygen = java.security.KeyPairGenerator.getInstance('RSA');
keyPair = keygen.genKeyPair();
%Generate Different Word Lengths
d = []; % Store one set of data
s = []; % Store all data sets
t_set = [];
for i = 0:5
    n = 2^i;
    for j = 0:n
        d = [d 'a'];
    end
    plaintext = d;
    tic
    cipher.init(Cipher.ENCRYPT_MODE, keyPair.getPrivate());
    plaintextUnicodeVals = uint8(plaintext);
    plaintextBytes = typecast(plaintextUnicodeVals,
'int8');
    ciphertext = cipher.doFinal(plaintextBytes);
    bDAT =
decimalToBinaryVector(typecast(ciphertext,'uint8'));
    t = toc;
    t_set = [t_set t];
end

plot(t_set);
```

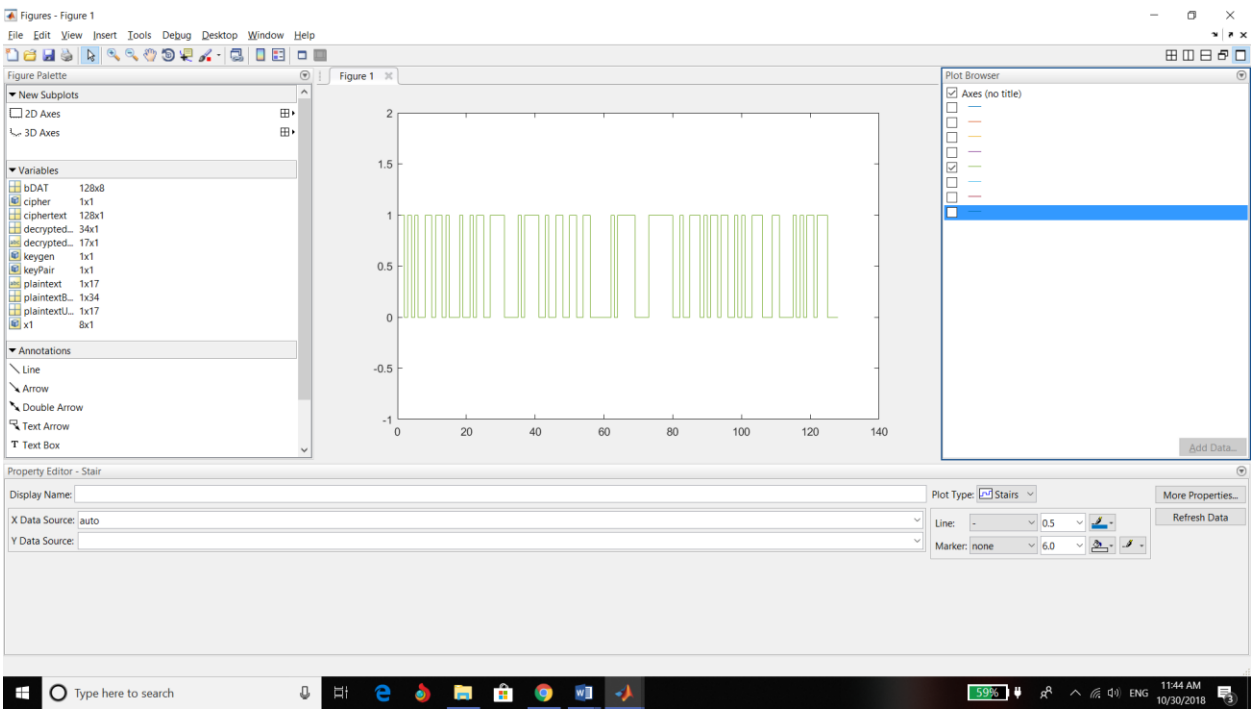
OBSERVATION

Java Encryption Signal:

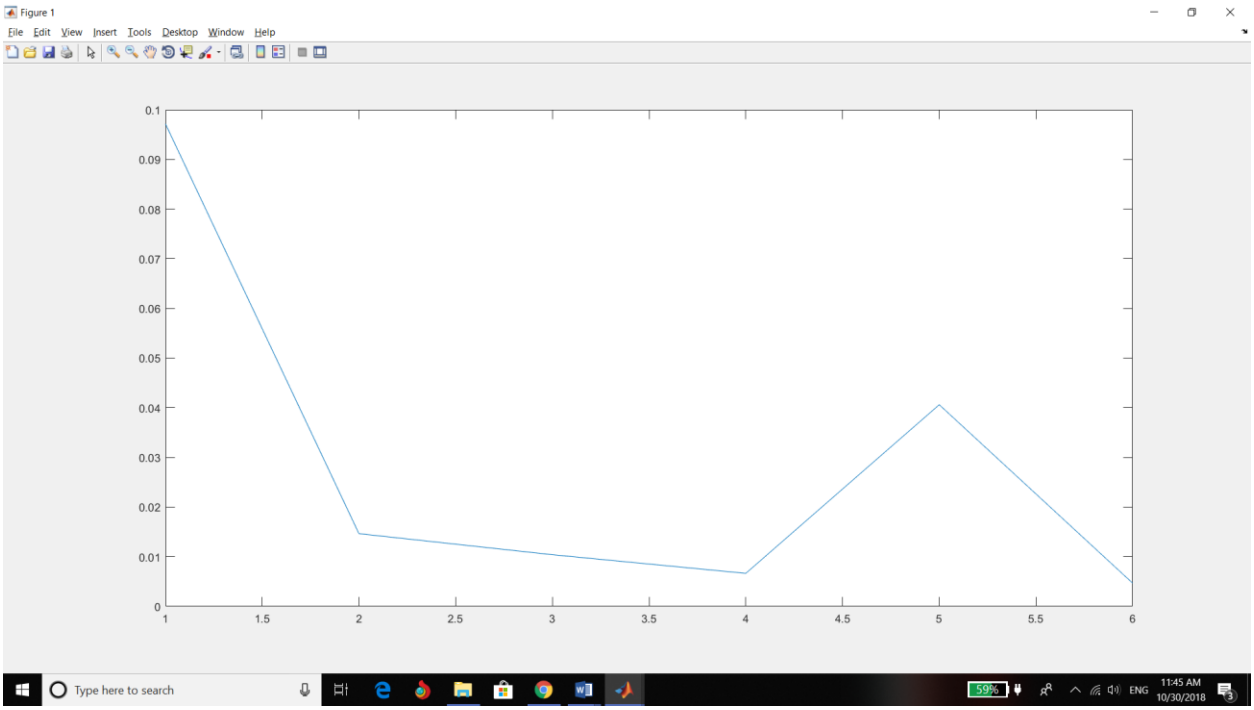





Figures - Figure 1
File Edit View Insert Tools Debug Desktop Window Help
Figure Palette
New Subplots
2D Axes
3D Axes
Variables
bDAT 128x8
cipher 1x1
ciphertext 128x1
decrypted... 34x1
decrypted... 17x1
keygen 1x1
keyPair 1x1
plaintext 1x17
plaintextB... 1x34
plaintextU... 1x17
x1 8x1
Annotations
Line
Arrow
Double Arrow
Text Arrow
Text Box
Figure 1
Plot Browser
Axes (no title)
Add Data...
Property Editor - Stair
Display Name:
Plot Type: Stairs
More Properties...
Refresh Data
X Data Source: auto
Y Data Source:
Line: - 0.5
Marker: none 6.0
Type here to search
59%
ENG
11:44 AM
10/30/2018


Figures - Figure 1
File Edit View Insert Tools Debug Desktop Window Help
Figure Palette
New Subplots
2D Axes
3D Axes
Variables
bDAT 128x8
cipher 1x1
ciphertext 128x1
decrypted... 34x1
decrypted... 17x1
keygen 1x1
keyPair 1x1
plaintext 1x17
plaintextB... 1x34
plaintextU... 1x17
x1 8x1
Annotations
Line
Arrow
Double Arrow
Text Arrow
Text Box
Figure 1
Plot Browser
Axes (no title)
Add Data...
Property Editor - Stair
Display Name:
Plot Type: Stairs
More Properties...
Refresh Data
X Data Source: auto
Y Data Source:
Line: - 0.5
Marker: none 6.0
Type here to search
59%
ENG
11:44 AM
10/30/2018

Execution Time Analysis:



CONCLUSION

The first graph output is the final encoded file post encryption. It is observed that the data is initially encoded to binary and based on RSA public and private keys, they are shifted and multiplexed, which results in an encrypted and undecipherable form.

The time analysis is with respect to the (2^n) bytes of data, with n plotted on x-axis and computation time on y-axis.

We thus can conclude that provided the data-size is in the range of 1-1kB, RSA method is very optimal and fast. However, there is an observed exponential increase with increased data size.

REFERENCES

,

[1]Jingli Zheng, Zhengbing Hu, Chuiwei Lu, ―A Light-weight Symmetric Encryption Algorithm Based on Feistel Cryptosystem Structure‖, IJCNIS, Vol. 7, No. 1, December 2014, pp. 16-23.

[2]Mamta. Juneja, and Parvinder S. Sandhu, ―A Review of Cryptography Techniques andImplementation of AES for Images‖, International Journal of Computer Science and Electronics Engineering (IJCSEE) Volume 1, Issue 4 (2013) ISSN 2320-401X; EISSN 2320-4028.

[3]Koumal Kaushik and Suman, ―An Innovative Approach for Video Steganography‖, IJCNIS, Vol. 7, No. 11, October 2015, pp. 72-79.

[4]Anupam Mondal and Shiladitya Pujari, ―A Novel Approach of Image Based Steganography Using Pseudorandom Sequence Generator Function and DCT Coefficients‖, IJCNIS Vol. 7, No. 3, February 2015 pp.42-49.

[5]Habutsu T., Nishio Y., Sasase I., and Morio S., ―A secret key cryptosystem by iterating chaotic map,‖ Lect. Notes comput. Sci, Advances in Cryptology-EuroCrypt'91, 1991, vol. 547, page(s): 127-140