ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 10, October 2016

Implementing the Information Security using Modified RSA Algorithm with the Help of N Prime Number

Md Wasim[1], Prof. Dr Pranam Paul[2]

MCA Final Year Student, Narula Institute of Technology, Agarpara, West Bengal, India[1]

HOD, Dept. of Computer Application, Narula Institute of Technology, Agarpara, West Bengal, India[2]

ABSTRACT: In any communication, security is the most important issue in today's world. Lots of data security and data hiding algorithms have been developed in the last decade, which worked as motivation for the research. The scenario of present day of information security system includes confidentiality, authenticity, integrity, non-repudiation. This present work focus is enlightening the technique to secure data or message with authenticity and integrity. With the growth of internet and network, the need for secure data transmission become more and more essential and important, as security is a major concern in the internet world. Data likely to be kept hide from all people except from the authorized user cannot be sent in plain text. So the plain text should be codified by the process of encryption.
Each type of data has its own features; therefore different techniques should be used to protect confidential data from unauthorized access. Here we modify the RSA algorithm with N prime number. The earlier RSA algorithm is use to encrypt the text with the help of two prime number, now the modify RSA algorithm can also be used to encrypt the plain text with the help of N prime number. In this algorithm encryption is done on the binary file so it can be applicable for any type of data such as text as well as multimedia. Here the same idea of cryptography is working (i.e. using key conversion of plain text into cipher text called encryption and the reverse means cipher text to plain text called decryption).

KEYWORDS: Cryptography, Encryption, Decryption, Plain Text (Pt), Cipher Text (Ct), Asymmetric key.

I. INTRODUCTION

The rapid growth of computer networks allowed larger files, such as digital image, text to be easily transmitted over the internet. Data encryption is widely used to ensure security of those data. Here we modify the RSA algorithm with n prime number. It is an Asymmetric key algorithm i.e. both encryption and decryption will have different key.
For encryption and decryption different keys have to be generated. At first, the plain text has been converted into its binary form and then the binary file is divided into some blocks. Using block size we will encrypt the message with the help of public key and we will decrypt the message with the help of private key.

II. RELATED WORK

In[3] the author used two prime number to generated the encryption and decryption key. In [7] the author used perfect square number to calculate the difference between two numbers and calculated the number of bits required to represent them. In [8] the author emphasized on division method where how many times division method will be applied is calculated. In [6] author used primer number from where basic concept of this algorithm is obtained. Each author has shown different ways of strengthening security to data. In this algorithm encryption and decryption process are performed on binary data. All data which is under stable by the computer is finally converted into binary bits. So it can be implemented for any data type encryption process. Therefore that encryption technique can be used for text encryption, image encryption etc.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 10, October 2016

III. ALGORITHMS

In this section, Key generation is discussed in section III.1. In the section III.2 and III.3 discussed about the algorithm of encryption and decryption respectively.

A. KEY GENERATION

1. Choose up to n large prime number to generate the key for encryption and decryption.
Let P=11, Q=13,R=17,S=19,T=23(I am taking 5 small prime number)
2. Calculate N=P*Q*R*S*T
We have N= 1062347
3. Select public key (i.e. encryption key) E such that it not the factor of e=(P-1)*(Q-1)*(R-1)*(S-1)*(T-1) and it should be less than e.
e=(11-1)*(13-1)*(17-1)*(19-1)*(23-1)
e=760320, the factor of 760320 are 2,2,2,2,2,2,2,2,2,3,3,3,5,11.Thus we have to choose E such that none of the factor of E 2,3,5 and 11. As a few example we cannot choose E as 4(Because it has 2 as a factor),15(Because it has 5 as a factor),33(Because it has 11 as a factor).
Let us choose E as 29.
4. To Select the private key (i.e. the decryption key) D such that the following equation is true:
(D*E) mod (n1-1)*(n2-1)(n3-1)(n4-1)….(nt-1)=1
Let us substitute the value of P, Q, R, S and T in the equation
We have (D*29) mod (11-1)*(13-1)*(17-1)*(19-1)*(23-1)=1
That is (D*29) mod (10)*(12)*(16)*(18)*(22)=1
That is (D*29) mod 760320=1
After some calculation, let us take D= 393269
(393269*29) mod 760320=11404801 mod 760320=1, which is what we wanted.

B. ALGORITHMS FOR ENCRYPTION

Step 01: Fist convert the each character of plain text into its corresponding ASCII value, after calculating the each character of ASCII value ,convert these value into 8 bit binary and store in a file1.
Step 02: Choose up to n large prime number where (n=1, 2, 3…. Up to t).
Step 03: Then calculate the Modified RSA modulus by multiplying them together say N.
Step 04: After calculating the modulus by multiplying n prime number, we will find how many bit required represent the value of Modified RSA modulus say it x.
Step 05: Choose the Block size (it should be same in encryption and decryption).
Step 06: Select public key such that (i.e. encryption key) E such that it not the factor of (n1-1)*(n2-1)(n3-1)(n4-1)….(nt-1)
Step 07: Read the selected block size from file1 and Convert the selected block size into decimal.
Step 08: Using the public key E encrypts the message (Ct=Pt^E mod N) and convert the output into binary and represent the binary bit into x no of bit and store it into file2 and count the no of bit if the no of bit is divisible by 8 then do not need to add dummy bit at the end of the file2 otherwise add the dummy bit at the end of the file.
Step 09: Select the 8 bit from the file2, convert into decimal, generate the symbol and store it into file3.
Step 10: Sent file3 to the receiver.

C. ALGORITHM FOR DECRYPTION

Step 1:Read each character from file3, find its ASCII value, and convert each ASCII value into 8 bit binary and store in a file4.
Step 2: Choose the block size say it bl. (it should be same in encryption and decryption).
Step 3: Find the no of bit required to represent the value of N say it x.
Step 4: Select the private key (i.e. the decryption key) D such that the following equation is true:
(D*E) mod (n1-1)*(n2-1)(n3-1)(n4-1)….(nt-1)=1
Step 5: Read the x no of bit from the file4 and convert it into decimal.
Step 6: After converting the x no of bit into decimal, decrypt the message by using (Pt=Ct^D mod N).

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 10, October 2016

Step 7: Then convert the result of step 5 into binary and represent this binary bit into bl (block size) no of bit and store in a file5.
Step 8: Now read 8bit data from file5, convert it into decimal and store it in a file6 (file6 will be our plain text).

IV. EXAMPLE

We have elaborate our project work through an illustrative example. Suppose we want to encrypt the plain text ‘‘Encryption MRSA”

A. EXAMPLE OF ENCRYPTION

Step 1: Fist each character of the plain text is converted into its corresponding ASCII value, after calculating the ASCII value convert these value into 8 bit binary and store in a file1.

E=>069=>01000101	n=>110=>01101110	c=>099=>01100011
r=>114=>01110010	y=>121=>01111001	p=>112=>01110000
t=>116=>01110100	i=>105=>01101001	o=>111=>01101111
n=>110=>01101111	sp(space)=>32=>00100000	M=>077=>01001101
R=>082=>01010010	S=>083=>01010011	A=>65=>01000001

File1 contain the binary stream following
01000101011011100110001101110010011110010111000001110100011010010110111101101110001000000100110101010010010100110100000 1

Step 2: Choose up to n large prime number where (n=1, 2, 3…. t).
P=3, Q=5, R=7, S=11(I am taking 4 small prime numbers to find the value of N).
Step 3: Then calculate the MRSA modulus by multiplying them together say N.
N=3*5*7*11=1155

Step 4: After calculating the Modified RSA modulus by multiplying n prime number, we will find how many bit required to represent the value of Modified RSA modulus N say it x.
N=1155-1=1154
x=11bits is required to represent the value of N
Step 5: Choose the block size say it Bl(it should be same in encryption and decryption).
Bl=4.
Step 6: Select public key such that (i.e. encryption key) E such that it not the factor of (n1-1)*(n2-1)(n3-1)(n4-1)….(nt-1)
e=(P-1)*(Q-1)*(R-1)*(S-1)
e=(3-1)*(5-1)*(7-1)*(11-1)=2*4*6*10=480
We will select the encryption key E=17
Step 7: Read the selected block size from file1 and Convert the selected block size into decimal.
0100=04
0101=05
0110=06
1110=14
In this way we can calculate all the remaining value.

Step 8: Using the public key E encrypt the message (Ct=Pt^E mod N) and convert the output into binary and represent the binary bit into x no of bit , store it into file2 and count the no of bit if the no of bit is divisible by 8 then do not need to add dummy bit at the end of the file2 otherwise add the dummy bit at the end of the file.

Ct=Pt^E mod N=04^17 mod 1155=>709=>01011000101
Ct=Pt^E mod N=05^17 mod 1155=>080=>00001010000
Ct=Pt^E mod N=06^17 mod 1155=>426=>00110101010
Ct=Pt^E mod N=14^17 mod 1155=>119=>00001110111
Ct=Pt^E mod N=06^17 mod 1155=>426=>00110101010

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 10, October 2016

In this way we can calculate all the remaining value.

File2 contain the binary as follow before addition of dummy bit.

010110001010000101000000110101010000011101110011010101001100000000011101110000100010110101110111
0000110000011001110111000000000000000011101110000101100010100110101010011000001100011010101000011
1000010011010101000001110111010001011010000000000000101100010110000100111000010100000100010110100
001010000011000000000101100010100000000001

No of bit in file2 is not divisible by 8 so we need to add 6 dummy bit

File2 contain the binary as follow before addition of dummy bit

010110001010000101000000110101010000011101110011010101001100000000011101110000100010110101110111
0000110000011001110111000000000000000011101110000101100010100110101010011000001100011010101000011
1000010011010101000001110111010001011010000000000000101100010110000100111000010100000100010110100
0010100000110000000001011000101000000000001000000

Step 8: Select the 8 bit from the file2, convert into decimal, generate the symbol and store it into file3.

01011000=088
10100001=161
01000000=064
11010101=213

In this way we can calculate the remaining value.

File3 Will contain the following cipher text

X¡@ÕsTÀÂ-wÜ p±MS5C,,ÕtZ ,'
´(0Š @

Step 10.Sent file3 (i.e. cipher text) to the receiver.

B. EXAMPLE OF DECRYPTION

File3 is containing the following cipher text

X¡@ÕsTÀÂ-wÜ p±MS5C,,ÕtZ ,'
´(0Š @

Step 1: Read each character from file3, find its ASCII value, and convert each ASCII value into 8 bit binary and store in a file4.

X □ 088 □01011000
¡ □161 □10100001
@ □064 □01000000
Õ □213 □11010101

In this way we can find all the remaining value.

Now the file4 will contain the following binary data

010110001010000101000000110101010000011101110011010101001100000000011101110000100010110101110111
0000110000011001110111000000000000000011101110000101100010100110101010011000001100011010101000011
1000010011010101000001110111010001011010000000000000101100010110000100111000010100000100010110100
0010100000110000000001011000101000000000001000000

Step 2: Choose the block size say it bl.

bl=4

Step 3: Find the no of bit required to represent the value of N say it x.

N=1155-1=1154

x=11 bits is required to represent the value of N

Step 4: Select the private key (i.e. the decryption key) D such that the following equation is true:

(D*E) mod (n1-1)*(n2-1)*(n3-1)*(n4-1)….(nt-1)=1

=(D*17)mod(3-1)*(5-1)*(7-1)*(11-1)
=(D*17)mod (2)*(4)*(6)*(10)
=(D*17)mod 480

If we put D=113, we will get as a remainder 1

(113*17) mod 480=1

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 10, October 2016

Step 5: Read the x (according to my example the value of x is 11 bit) no of bit from the file4 and convert it into decimal.
01011000101=709
00001010000=080
00110101010=426
00001110111=119
00110101010=426
In this way we can calculate all the remaining value.
Step 6: After converting the x no of bit into decimal, decrypt the message by using (Pt=ct^D mod N).
Pt=ct^E mod N=0709^113 mod 1155=>04
Pt=ct^E mod N=0080^113 mod 1155=>05
Pt=ct^E mod N=0426^113 mod 1155=>06
Pt=ct^E mod N=0119^113 mod 1155=>14
Pt=ct^E mod N=0426^113 mod 1155=>06
In this way we can calculate all the remaining value.
Step 7: Then convert the result of step 6 into binary and represent this binary bit into bl (block size .i.e. 4) no of bit and store in a file5.
04=0100
05=0101
06=0110
14=1110
06=0110
In this way we can calculate all the remaining value.

Now the file2 will contains the following binary data
0100010101101110011000110111001001111001011100000111010001101001011011110110111000100000010011010101001001010011010000001

Step 8: Now read 8bit data from file5, convert it into decimal, generate the symbol and store it in a file6 (file6 will be our plain text).

01000101□069□E	01101110□110□n	01100011□**0**99□c
01110010□114□r	01111001□121□y	01110000□112□p
01110100□116□t	01101001□105□i	01101111□111□o
01101110□110□n	00100000□032□sp	01001101□077□M
01010010□082□R	01010011□083□S	01000001□065□A

The file6 (plain text) will contain the following
Encryption MRSA

V. RESULT ANALYSIS

In this algorithm encryption is perform on binary data. All data which is under stable by the computer is finally converted into binary bits. So it can be implemented for any data type. Therefore that encryption technique can be used for text encryption, image encryption i.e., multimedia encryption process. The specialty of this algorithm is that, if we change the key then the encryption and decryption time will be different in each time.

A. CHOOSE KEY
As before we say the Key used to be in Encryption and Decryption process. The restriction of choosing encryption key E is 1< E <e and for the decryption it should be (D*E) mod e=1

B. SIZE AND TIME COMPARATIVE REPORT ON ENCRYPTION
This algorithm has been implemented on number of data files varying types of content and sizes of wide range, shown in Table 1.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 10, October 2016

TABLE 1
Size and Time Comparative Table of encryption

FILE NAME	FILE SIZE(IN BYTE)	ENCRYPTION FILE SIZE(IN BYTE)	ENCRYPTION TIME(in sec)	ENCRYPTION TIME/BYTE
Msg1	1024	2304	0.0102990	0.00001006
Msg2	2036	4581	0.02133390	0.0001048
Msg3	4098	9221	0.03789700	0.00000925
Msg4	6147	13831	0.05233300	0.00000851
Msg5	10245	23052	0.07766600	0.00000758
Msg6	20490	46103	0.12692900	0.00000619

The following graph, Fig V.2.1 shows the comparison between file size and encryption time per byte.

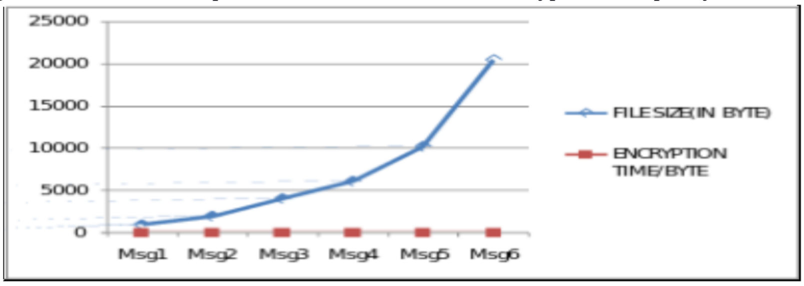


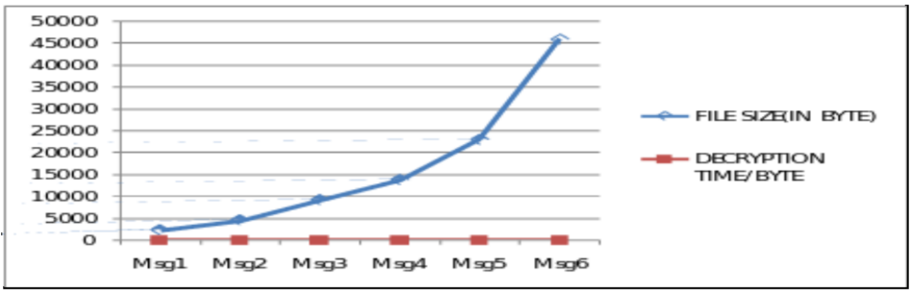
Fig V.2.1 File Size vs Encryption time per byte

- Sky Blue line indicates the file size in bytes.
- Light Pink line indicates the encrypted file size in byte.

From the above figure V.2.1 we can say that if the size of the will increase then the encryption time will also increase In tabel2 we show the decryption time with decrypted file size.

TABLE 2
V.2 Size and Time Comparative Report on decryption

FILE NAME	FILE SIZE(IN BYTE)	DECRYPT FILE SIZE(IN BYTE)	DECRYPTION TIME(in sec)	DECRYPTION TIME/BYTE
msg1.txt	2304	1024	0.02775200	0.00001006
Msg2.txt	4581	2036	0.05143700	0.00001123
Msg3.txt	9221	4098	0.08527400	0.00000925
Msg4.txt	13831	6147	0.09784700	0.00000707
Msg5.txt	23052	10245	0.15174700	0.00000658
Msg6.txt	46103	20490	0.25603700	0.00000555



The following graph, Fig V.2.2 shows the comparison between file size and decryption time per byte.
Fig V.2.2 File Size vs Decryption time per byte

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 10, October 2016

- Sky Blue line indicates the file size inbytes.
- Light Pink line indicates the decrypted file size in byte.

From the above figure V.2.2 we can say that when the size of the file is increases then the decryption time also increase.

C. *SECURITY*

The security of RSA public key encryption algorithm is mainly based on the integer factorization problem, which can be described as:

Given integer n as the product of 2 distinct prime numbers p and q,
find p and q.

If the above problem could be solved, the RSA encryption is not secure at all. This is because the public key {n, e} is known to the public. Anyone can use the public key {n, e} to figure out the private key {n,d} using these steps:

- Compute p and q by factorizing n.
- Compute m = (p-1)*(q-1).
- Compute d such that d*e mod m = 1 to obtain the private key {n,d}

If n is small, the integer factorization problem is easy to solve by testing all possible prime numbers in the range of (1, n).

For example, given 35 as n, we can list all prime numbers in the range of (1, 35): 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, and 31, and try all combinations of them to find 5 and 7 are factors of 35.

As the value of n gets larger, the integer factorization problem gets harder to solve. But it is still solvable with the use of computers. For example, the RSA-100 number with 100 decimal digits, or 330 bits, has been factored by Arjen K. Lenstra in 1991:

```
RSA-100 = 15226050279225333605356183781326374297180681149613
80688657908494580122963258952897654000350692006139

p*q = 37975227936943673922808872755445627854565536638199
* 40094690950920881030683735292761468389214899724061
```

If you are using the above RSA-100 number as n, your private key is not private any more.

As of today, the highest value of n that has been factored is RSA-678 number with 232 decimal digits, or 768 bits, factored by Thorsten Kleinjung et al. in 2009:

```
RSA-768 = 12301866845301177551304949583849627207728535695953
34792197322452151726400507263657518745202199786469
38995647494277406384592519255732630345373154826850
79170261221429134616704292143116022212404792747377
94080665351419597459856902143413

p*q = 33478071698956898786044169848212690817704794983713
76856891243138898288379387800228761471165253174308
7737814467999489
× 36746043666799590428244633799627952632279158164343
08764267603228381573966651127923337341714339681027
0092798736308917
```

As our computers are getting more powerful, factoring n of 1024 bits will soon become reality. This is why experts are recommending us [4]:

- Stop using RSA keys with n of 1024 bits now.
- Use RSA keys with n of 2048 bits to keep your data safe up to year 2030.
- Use RSA keys with n of 3072 bits to keep your data safe beyond year 2031.

http://www.herongyang.com/Cryptography/RSA-Algorithm-How-Secure-Is-RSA-Algorithm.html

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 10, October 2016

VI. CONCLUSION

My conclusion towards this algorithm is that I have tested the implementation of this algorithm and this algorithm worked correctly for the above set of values. From this we can assume that algorithm can correctly be implemented for various type and size of file. The length of the plain text is not restricted in this algorithm, so it can be applicable for any larger file. It will be secured

REFERENCES

[1] Pranam Paul, Saurabh Dutta, A K Bhattacharjee,"AnApproach to ensure Security through Bit-level Encryption with Possible Lossless Compression", International Journal ofComputer Science and Network Security", Vol. 08, No. 2, pp.
291 – 299, 2008.

[2]William Stallings, "Cryptography and network security principles and practices", 4th edition, Pearson Education, Inc.publishing as Prentice Hal, 2006.

[3]Atul Kahate, ''Cryptography and Network Security'', 4th edition, McGraw-Hill publishing company limited, 2008

[4] http://www.herongyang.com/Cryptography/RSA-Algorithm-How-Secure-Is-RSA-Algorithm.html

[5] Pranam Paul, Saurabh Dutta, A K Bhattacharjee ; "**An Approach to ensure Security through Bit-level Encryption with Possible Lossless Compression",** International Journal of Computer Science and Network Security", Vol. 08, No. 2, pp. 291 – 299, 2008.

[6]Moinak chowdhury , Prof. Dr. Pranam Paul ; **Block Based Data Encryption and ecryption Using The Distence Between Prime Numbers" ;** 2015.

[7]Shibaranjan Bhattacharyya, Prof. Dr.Pranam Paul, "An Approach to Block Ciphering using Root of Perfect Square Number", International journal of Computer Science and Network Security,ISSN: 0974 – 9616 vol-7,No.2,2015.

[8]Ayan Banrjee, Prof. Dr.Pranam Paul, "Bock Based Encryption and Decryption", International journal of Computer Science and Network Security, ISSN: 0974 – 9616 vol-7,No.2,2015.

[9] Wikipedia, the free encyclopedia.

BIOGRAPHY

Md Wasim, he is a final year student of MCA, Narula Institute of Technology under WBUT. He completed his BCA from the same college of WBUT. He completed his 10th and 12th from C.M.O High School and MD. JAN High Secondary School.

Prof. Dr Pranam Paul, ***Assistant Professor and Departmental Head, CA Department, Narula Institute of Technology (NIT), Agarpara*** had completed MCA in 2005. Then his carrier had been started as an academician from MCKV Institute of Technology, Liluah. Parallely, At the same time, he continued his research work. At October, 2006, National Institute of Technology (NIT), Durgapur had agreed to enroll his name as a registered Ph.D. scholar. Then he had joined Bengal College of Engineering and Technology, Durgapur. After that Dr. B. C. Roy Engineering College hired him in the MCA department at 2007. At the age of 30, he had got Ph.D. from National Institute of Technology, Durgapur, West Bengal. he had submitted his Ph.D. thesis only within 2 Years and 5 Months. After completing the Ph.D., he had joined Narula Institute of Technology in Computer Application Department. Parallely he continue his research work. For that, he have 39 International Journal Publications among 54 accepted papers in different areas. he also reviewer of International Journal of Network Security (IJNS), Taiwan and International Journal of Computer Science Issue (IJCSI); **Republic of Mauritius.**