RSA ALGORITHM WITH A NEW APPROACH ENCRYPTION AND DECRYPTION MESSAGE TEXT BY ASCII

Ahmad Steef[1], M. N. Shamma[2] and A. Alkhatib[3]

[1]Department of Mathematics, Al Baath University- Homs- Syria.
[2]Basic Sciences department- the Mechanical and Electrical Engineering- Damascus University- Syria.
[3]Department of mathematics & Faculty sciences - Al Baath University- Homs- Syria.

ABSTRACT

In many research works, there has been an orientation to studying and developing many of the applications of public-key cryptography to secure the data while transmitting in the systems, In this paper we present an approach to encrypt and decrypt the message text according to the ASCII(American Standard Code for Information Interchange) and RSA algorithm by converting the message text into binary representation and dividing this representation to bytes(8s of 0s and 1s) and applying a bijective function between the group of those bytes and the group of characters of ASCII and then using this mechanism to be compatible with using RSA algorithm, finally, Java application was built to apply this approach directly.

KEYWORDS

Cryptography, RSA Algorithm, ASCII(American Standard Code for Information Interchange), Binary Number Systems, Java Language.

1. INTRODUCTION

Public-Key Cryptography[1][2] is the most common major factor for security and protection of Systems. It is a strong technique to secure the data transmitting within Systems. Public-Key Cryptography uses two roles(encrypting and decrypting functions). Encrypting function encrypts massage and converts it to cipher(gibberish) "C", Decrypting Function is applied to "C" to coming back to original message "M". decryption and encryption functions are inverse each other and use distinct keys(numbers)[1]. In a special case, the RSA Algorithm[1][2], the most strong common applications of Public-Key Cryptography which published by Rivest, Shamir and Adelman 1978. It depends on the difficult problem in mathematics which is "factorization problem". This Algorithm was built depending on exponential function E(Encryption Function)has inverse D(Decryption Function), but it is difficult to discovering it, because this depends on the " P = NP Problem[2]" is one of the biggest open problems in both mathematics and computer science. RSA Algorithm is used to encrypt and decrypt text according to represent alphabets (A, B, …….Z) as decimal numbers(from 0 ..to 25)[2], but what about if the text message includes numbers and some characters like: &@!*....? what about encryption and decryption text message by RSA according to ASCII(American Standard Code for Information Interchange)[3]. This paper focused on cryptography to secure the message while transmitting in the systems, so, protecting

DOI:10.5121/ijcis.2015.5403

confidentiality and integrity of information. In this paper we will illustrate the use of RSA to encrypt and decrypt message which is to be transmitted from sender to receiver by a new approach related of ASCII. It's easy to convert any character in ASCII into associated number but How can we convert big numbers to the associated text by ASCII?. The answer of all inquiries was included in this paper, and a numerical example applied to illustrating the basic idea in applicable way, finally, Java application was built to apply this approach directly.

2. RSA ALGORITHM.

RSA Algorithm[1][2] is the most strong common applications of Public-Key Cryptography which published by Rivest, Shamir and Adelman 1978. It uses two distinct keys(two numbers), public-key which possible to be known to every one and the other is private-key which is secured and not allowed to exchange between the sender and reciver. This algorithm described as following:

1- Choose two distinct large random prime numbers p and q.
2- Compute $n = p * q$,
3- Compute Euler's function of n : $\emptyset(n) = (p-1) * (q-1)$.
4- Choose an integer e such that $1 < e < \emptyset(n)$ and $gcd(e, \emptyset(n)) = 1$.,
5- Compute d such that: $e.d = 1(mod\ \emptyset(n))$.

The number e is the public – key and d is the private –key. Let M is a message need to be encrypted and get cipher C, so, in the first the message M represented as blocks of numbers every value of block must be less than the number "n" [2]. For any block number M_i ; $i \in I$,such that: $M_i \in Z_n = \{0,1,2,3, \dots \dots n-1\}$, the Encryption role of this algorithm is :$C_i \equiv {M_i}^e (mod\ n)$. and, the Decryption role is :$M_i \equiv {C_i}^d (mod\ n)$.

Note: M divided into blocks as form: $(M_1 | M_2 | \dots \dots M_i | \dots \dots)$and Cipher C will be as form: $(C_1 | C_2 | \dots \dots C_i | \dots \dots)$.

3. ADDRESSED PROBLEM

Suppose we have message text and want to encrypt it by RSA algorithm depending on ASCII, first, we will represent it as numbers by convert every character of the message text into associated number depending on ASCII, then we will apply RSA algorithm on those numbers which presented in, but when we want to coming back to original message, there will be problem because there are characters in ASCII represented as decimal numbers of two digits and the others represented as numbers in a three digits, for example the character "a" is represented as 97 and "v" is represented as 118, and " %" is represented as 37, so, for example the "av%" converted to the number "1189737" but according to the number"1189737" we can't discover which message associated by ASCII even if we forwarded to dividing it's digits because we don't know the right mechanism for dividing!, not only for that, but also we want to represent the text as not big decimal numbers as possible as we can, to be able to apply RSA algorithm on that whole text directly in an efficient way.

In the following section we will illustrate our approach to be able to coming back to the original message and how we will use RSA algorithm with this approach.

4. PROPOSED SOLUTION AND ASSOCIATED APPROACH WITH RSA.

We know that ASCII table[3] consists of 256 characters and the representation of them as numbers from 0 to 255, so, suppose the group A is equal to $Z_{255} = \{0,1,2 \dots \dots 255\}$, and let's define the function f as form: $f: A \to \{0,1\}^8$. It's easy to know the number of elements of $\{0,1\}^8$ is $2^8 = 256$. Indeed, f is bijective function and so, it has inverse function. We can represent every dicimal number in A as a binary number by using binary and decimal number systems principles[5], and every number consists of 8 of 0s and 1s.

(we can add some 0s to the left of that representation if the number of digits is not 8), the same, we can represent every character as binary representation and every binary consists of 8 of 0s and 1s.

Now, we will reivew our approch as algorithm, so suppose the message M which we want to encrypt it, this algorithm illustrates the encryption role and get Cipher "C"(gibberish) as form:

1- Convert every character of M to binary representation(every character must consists of 8 of 0s and 1s) from left to right.
2- Convert whole representation(S) of M to decimal representation M_1 .
3- Apply RSA encrypting function on the number M_1 and get number M_2 .
4- Convert the number M_2 to binary representation(but the number of digits must divided by 8...we can do that by adding zeros to the left of representaion)
5- Divide that representation into bytes(8s of 0s and 1s) from left to right.
6- Convert every byte in step 5 into asochiated character in ASCII from left to right and this creates cipher text C.

To come back the original message "M", the following algorithm illustrates that:

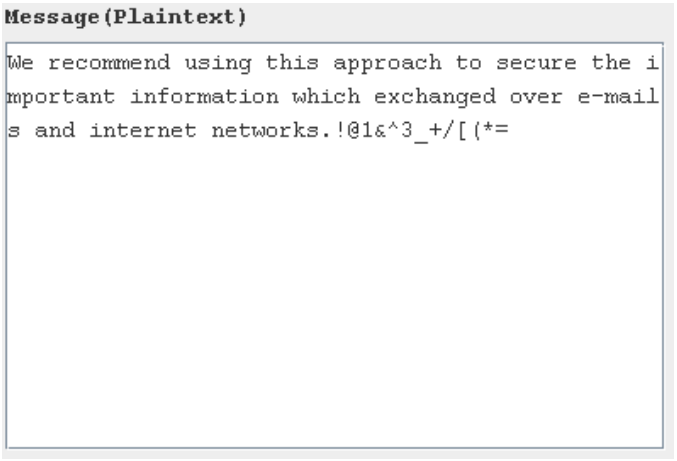
1- Convert every character of C to binary representation(every character must consists of 8 of 0s and 1s) from left to right.
2- Convert whole representation of C to decimal representation M_2 .
3- Apply RSA decrypting function on the number M_2 and will get M_1.
4- Convert the number M_1 to binary representation(S)(but the number of digits must divided by 8...we can do that by adding zeros to the left of representaion)
5- Divide that representation into bytes(8s of 0s and 1s) from left to right.
6- Convert every byte in step 5 into asochiated character in ASCII from left to right and this creates the original message text M.

REMRK (1): we can stop at step(4)in encryption algorithm above and get cipher as a dicimal number, but in this case we will begin from step(4)in the decryption algorithm mentioned above.

REMRK (2): within applying this approch we have to be attantion in conditions of RSA algorithm wich related of message when represented as dicimal number because of the RSA not applicable if this number isn't less than the modulo n (n=p*q) (see [2]).

5. NUMERICAL EXAMPLE.

Suppose the message M as following:



And we want to apply our approach with RSA, so, in the beginning let's build the RSA requirements$(p, q, n\ e, d, \emptyset(n))$ as mentioned above.

By using Miller–Rabin Test algorithm [2][4], we generated a 600-bits prime numbers p and q using java language[4][5] and get the following:

p=368166876332410021820655766545518792801657530238122706751384778240042778538461783870579961419016808201088329579401454411226957952898605246763846960198591381699175558181954423680935 1.

q=271867940257160144799268272235976309694414039178006499884876265232914021114468180061170641218156523178466332440217980586197709849237029353231432585315328291036135994902620997906195 9.

n=p*q=100092770339404915095673588919015806094269106785965013167567222713830562213232789743817326355526845720378796064871153364854291987320979265303230176808960953480042856110102903249172263946823829341628768305886551852057174997356577742468864641126802282282262244313854572630034076738036034391558203350478162209554367664324531491865426026769622041340025808209957860 9.

So, $\emptyset(n) = (p-1) * (q-1) =$ 100092770339404915095673588919015806094269106785965013167567222713830562213232789743817326355526845720378796064871153364854291987320979265303230176808960953480042856110102903249171623912007239771462148381847770356954678925787161613262228380083329325482609314349922822027396903406656479729538583915480737541752232029724536212319912107096886729786941232788370730 0.

Select number e ; $1 < e < \emptyset(n)$, such that: $gcd(e, \emptyset(n)) = 1$, let take e= 11;

And the number d(inverse product) will be :

d=45496713790638597770760722235916275497395048539075005985257828506286619187833
08624718969379796674805471763457494143334766104181241862693796832864440040731606
72746752777319501476896192687276018077937340173567168344070308602630528006028 2
85627310604238855731506522692191830634956093934763513426629052491244337160105 46
80566073692363236850440394226304278330856230591.

So, the public- key is (e,n) and private-key is (d, n).Now, let's apply our approach which mentioned above step by step, to encrypt message M.

1- The binary representation of the message (character by character) is: S=
010101110110010100100000011100100110010101100011011011110110110101101101
011001010111011100110010000100000011101010111001101101001011011001100111
001000000111010001101000011010010111001100100000011000010111000001110000
011100100110111101100001011000110110100000100000011101000110111100100000
011100110110010101100011011101010111001001100101001000000111010001101000
011001010010000001101001011011010111000001101111011100100111010001100001
011011100111010000100000011010010110111001100110011011111011100100110110 1
011000010111010001101001011011110110111000100000011101110110100001101001
011000110110100000100000011001010111100001100011011010000110000101101110
011001110110010101100100001000000110111101110110011001010111001000100000
011001010010110101101101011000010110100101101100011100110010000001100001
011011100110010000100000011010010110111001110100011001010111001001101110
011001010111010000100000011011100110010101110100011101110110111101110010
011010110111001100101110001000010100000000110001001001100101111000110011
010111110010101100101111010110110010100000101010001111 01.

2- The decimal representation of S is:
M_1=67477990192598109094026434793800128271623759125622767016961491405796 0
88274383644369970131807949813858022588176538096721730314658378073281265 6
11145286262073549835300274884544221576823895398534324930035218598971627 6
56472563528346662621303268839116735117580407292136276840880384768590448 9
44662517840792486937386332916393 57.

3- Applying RSA Encryption on M_1 and get M_2as following:

$$M_2 = M_1^e (mod\ n) =$$

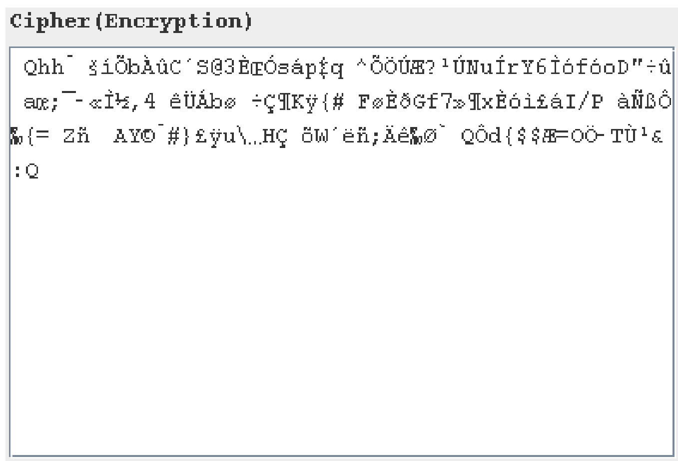
863062826742439700286535333905757029720207366910955251723582169330998889
334187101465152792779967024367063523002136275498607043397142939850644957
851666561154751260571623481264893147279833000162802646942400233819241566
533597070669145406550404565169899133453077460458404471104791143022106721
525602611893089331489685531217659064283488939741765624469536510261947451
3.

4- binary representation of M_2 (with 0s which added to the left) is:
10000000010100010110100001101000100010000000000110011010111011011101010 1
000011010110001011000000111111011010000111011010001010011010000000011001 1
110010001000110011010011011100111110000101110000100001110111000100000101
010111101101010111010110110110101100011000111111101110011101101001001110
011101011100110101110010010110010011011011001100111100110110011011110011
011011110100010000100010111101111111101100011111011000011001110000111011
101011111001011010101011110011001011110100101100001101001001000011101010

1101110011000001011000101111100000001000111110111110001111011011001001011
1111111011110110010001100000111010001101111100011001000111100000001000111
0110011000110111101110111011011001111000110010001111001111101100101000011
1110000101001001001011110101000001111111111000001101000111011111111010100
1000100101111011001111010000000100101101011110001000011100000001010100001
0101100110101001100010000001000110111110110100011111111111011101010101011100
1000010101001000110001111000110111111010101010111101101001110101111110001
0011101111000100111010101000100111011000100110000001111101010001110101100
0110010001111011001001000010010011000110001111010100111111010110100101100
010101001101100110111001001001100011101001010001.

5- Divide the representation in step (4) into 8s from "**10000000**" to the last byte which is"**01010001**"

6- Convert every byte from "**10000000**" to "**01010001**" into associated character in ASCII and get the cipher C(more gibberish)as form:



NOTE: in cipher above there are some characters disappear.

Now, suppose the receiver wants to do decryption to knows the original message M !, our decryption approach illustrates the mechanisem as the following:

1- The binary representation of the cipher C (character by character) is the same as in step (4)above in example(every representation of character must be byte): L=

2- 1000000001010001011010000110100010001000000000001100110101110110111010101
0000110101100010110000000111110110100001110110100010100110100000000110011
1100100010001100110100110111001111100001011100001000011101110001000000101
0101111011010101110101101101101011000110001111111011100111011010010011110
0111010111001101011100100101100100110110110011001111001101100110111100111
0110111101000100001000101111011111111011000111110110000110011100001110111
1010111110010110101010111100110010111101001011000011010010010000111010110
1101110011000001011000101111100000001000111110111110001111011011001001011
1111111011110110010001100000111010001101111100011001000111100000001000111
0110011000110111101110111011011001111000110010001111001111101100101000011
1110000101001001001011110101000001111111111000001101000111011111111010100
1000100101111011001111010000000100101101011110001000011100000001010100001

0101100110101001100010000010001101111101101000111111111101110101010111001000010101001000110001111000110111110101010101111011010011101011111100010011101111000100111010101000100111011000100110000001111101010001110101000110010001111011001001000010010011000110001111010100111111010110100101100101010011011001101110010010011000111010010100 01..

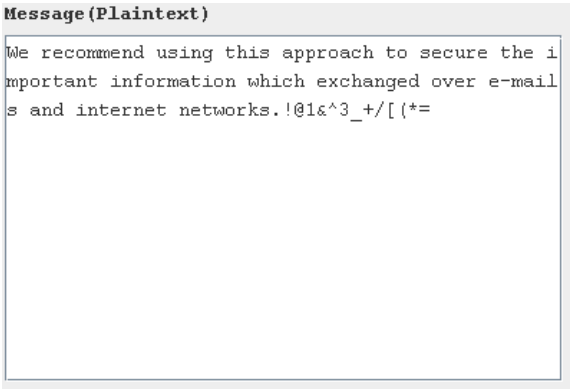
3- The dicimal representation of L is M_2= 863062826742439700286535333905757029720207366910955251723582169330998889334187101465152792779967024367063523002136275498607043397142939850644957851666561154751260571623481264893147279833000162802646942400233819241566533597070669145406550404565169899133453077460458404471104791143022106721525602611893089331489685531217659064283488939741765624469536510261947451 3.

4- Applying RSA Decryption on M_2 and get M_1 the same as in the step (2) in encryption algorithm above as following:
$M_1 = M_2^d (mod\ n) =$ 674779901925981090940264347938001282716237591256227670169614914057960882743836443699701318079498138580225881765380967217303146583780732812656111452862620735498353002748845442215768238953985343249300352185989716276564725635283466626213032688391167351175804072921362768408803847685904489446625178407924866937386332916 39357.

5- Convert M_1 into binary representation(adding zeros to the left of representation to be the length divided by 8) and we will get S the same in the first step in encryption algorithm above as following: S= 010101110110010100100000011100100110010101100011011011110110110101101101011001010110111001100100001000000111010101110011011010010110111001100111001000000111010001101000011010010111001100100000011000010111000001110000011100100110111101100001011000110110100000100000011101000110111100100000011100110110010101100011011101010111001001100101001000000111010001101000011001010010000001101001011011010110101110000011011110111001001110100011000010110111001110100001000000110100101101110011001100110111101110010011011010110000101110100011010010110111101101110001000000111011101101000011010010110001101101000000100000011001010111100001100011011010000110000101101110011001110110010101110010000100000011011110111011001100101011100100000000110010100101101011011010110000101101001011011000111001100100000011000010110111001100100000010000001101001011011100111010001100101011100100110111001100101011101000010000001101110011001010111010001110111011011110111001001101011011100110010111000100001010000000011000100100110010111100011001101011111100101011001011110101101100101000000101010001111101.

6- Divide the last representation into bytes and you able to do that because the number of digits is divided by 8.

7- Convert from left to right every byte in the last representation into associated character in ASCII and will be get the exactly message "M" :

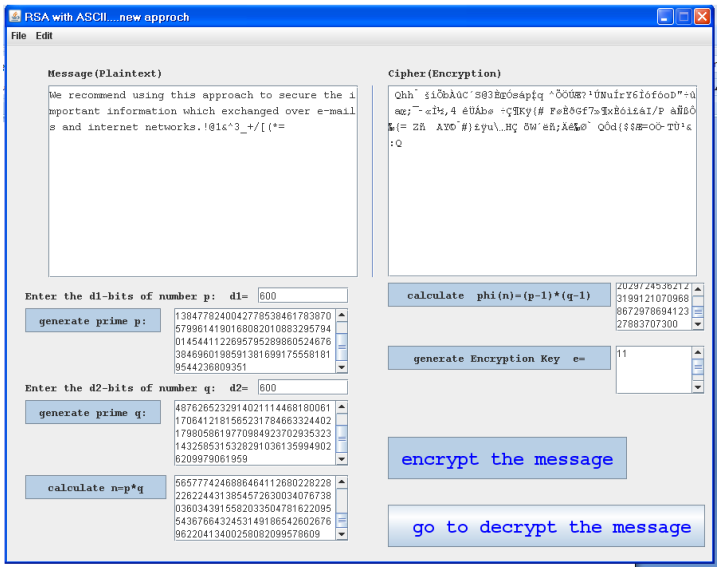


NOTE: We can see that decryption approach is absolutely true and will lead to the exactly original message, because it was built from last step to first step in the proposed encryption approach in this paper(it is the inverse of encryption approach step by step!).

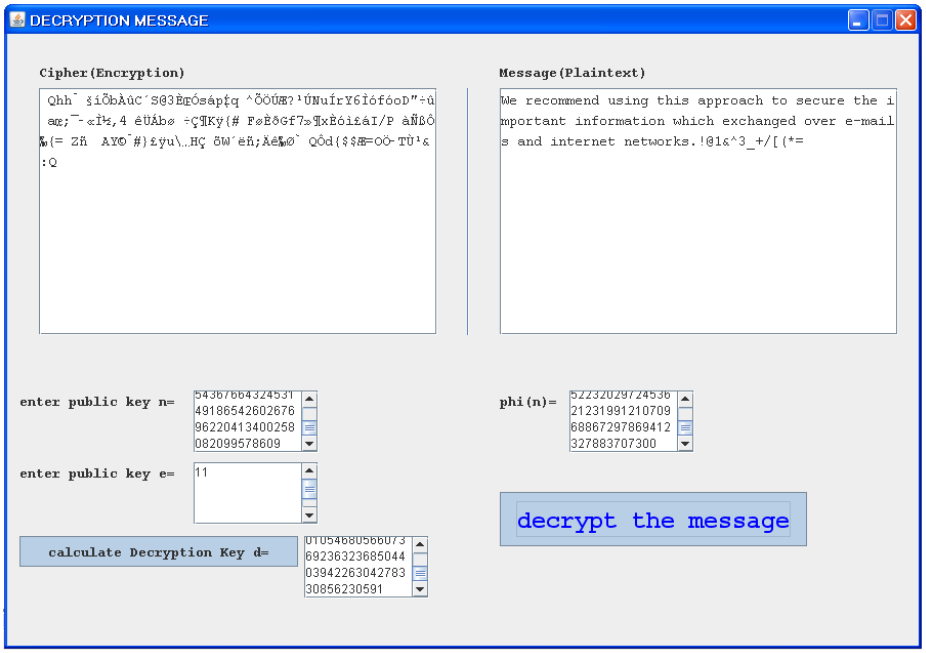
6. JAVA APPLICATION FOR PROPOSED APPROACH

Depending on java language[4][5] we build an application to implimination our approach. In this application we can generate d1,d2-bits primes numbers p and q depending on Miller–Rabin Test algorithm[4][2], and we can directly apply RSA algorithm with our approach to encrypt and decrypt message text according to ASCII. Also, this application can tell us directly if the RSA algorithm is applicable with this approach or not(just when we put the message text wich we want incrypt it).

Figure(1) reviews the results of the numerical example mentioned above by our application, and the figure(2) appeares when the "go to decrypt the message" button is preesed to come back the original message according to the private key "d":



Figure(1): java application for encryption approach deppending on RSA and ASCII



Figure(2): java application for decryption approach deppending on RSA and ASCII

7. CONCLUSION AND FUTURE PLAN

This paper describes a new approach for encryption and decryption message text using RSA algorithm depending on ASCII, and java application to implementation this approach. We recommend using this approach to secure the important information which exchanged over e-mails and internet networks, even, secure the gibberish passwords.

Since the RSA algorithm will be slow when the message text is big, so, we recommend to use this approach after dividing the message into blocks and apply this approach on the every block in fast way while transmitting in the networks ...block by block, so we can build java applet to do that and this is will be our future plan with what the relationship between the modulo "n" in RSA and the number representation of the message to allow us dividing into blocks automatically in an efficient way.

REFERENCES

[1] William, Stalling,(2014), "Cryptography and Network Security-Principles and Practice", Sixth Edition, Pearson Education, Inc
[2] Song, Y. Yan,(2013), "computational number theory and modern cryptography", 1st, Wiley.
[3] website Link: http://www.asciitable.com/ . On Thu Dec 24 20:38:46 , 2015.
[4] David, Bishop, (2003), "Introduction to Cryptography with Java Applets",Jones and Bartlett Publisher.
[5] Y. Daniel Liang,(2013), "Introduction to Java Programming " Ninth Edition, Pearson Education..

AUTHORS

Ahmad Steef is a PhD Candidate in mathematics at Al-Baath University, Homs, Syria.

Mohammad Nour Shamma is a professor in mathematics at Damascus University- Basic Sciences department- the Mechanical and Electrical Engineering- Syria. His Research interests include mathematical analysis, Cryptography.

Abdulbaset Alkhatib is a professor in mathematics at Al-Baath University- Homs, Syria. His Research interests include Linear algebra, Cryptography.