INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH & DEVELOPMENT

ISSN 2278 – 0211 (Online)

Modified Double Mod RSA Tested with Brute Force Attack

Abhishek Gupta
M.Tech Scholar, Dehradun Institute of Technology, Dehradun, India
Vishal Sharma
Assistant Professor, Dehradun Institute of Technology, Dehradun, India

Abstract:
Rivest Shamir and Adlemen (RSA) cryptosystem, an asymmetric cryptosystem uses single modulus function for encryption and decryption with different key, hence asymmetric. The key to decryption is using private key that can be created from the public key and modulus using modulus inverse, extended Euclidean method. We modified the RSA using double modulus function to increase the security. After which we applied brute force attack which is more prevalent in the insecure network. There were 3 transmissions between the sender and receiver for the same message. We tested the RSA security using brute force attack on final transmission where the receiver is getting the final message. We got around 12 character match from the original plain text. Which forces, further security in RSA needs to be implemented. We defined keys of size 32 bits and modulus in range from 16-32 bits.

Keywords: *RSA, DOUBLE RSA, Frequency analysis, Brute force attack*

1. Introduction

Internet was the invention in the field of ICT for purpose of exchange of information and shortens the spaces. The network was build to transfer Gigabytes of data. But it led to rise of eavesdropper who using known plaintext, known cipher text can easily decrypt the cipher text generated. To prevent easy decryption asymmetric key algorithm was designed by Rivest Shamir and Adlemen (RSA) [1]. The security of RSA lies in the difficulty in factoring the large number (modulus). The prime numbers are generated randomly and tested using Rabin miller primality test [2]. If a good algorithm survives for long that's will be a miracle, if one can factor the modulus into prime numbers then it will be breach of security. In the RSA there is no authentication of the sender sending the confidential information. Though generating a factor belongs to a NP Hard problem, but with high computing power processor it doesn't seems difficult. To enhance the security further we proposed a technique of Double RSA. A methodology defined with cooperation of the sender and the receiver. Double RSA technique enhances the security of the message by using double modulus function which makes it difficult to decrypt. But we also propose 3 brute force attacks that may be possible on the double RSA which makes it though a weak implementation of security [3]. The following paper is organized as section 2 describes about the research made in this field. In section 3 we describe brief about RSA, brute force attack, frequency analysis. In section 4 we write about the algorithm used to implement Double RSA and methodology undertook to perform the brute force attack. In section 5 we write about the conclusion that we derive from the research work and possible future enhancement in improving this technique.

2. Literature Survey

H. Orman et.al in [5] explains how to determine the length of an asymmetric key as a function of a symmetric key strength. He also explained how varying the sizes of large integers change the time to use algorithms for exchange. Gene Tsudik in [6] explained about message authentication with one-way hash function, which we used modifying it with double modulo. In [7] characterizes the attacks on RSA into 4 categories, they are elementary attacks by misuse, low private exponent attack, low public exponent attack and attack on the implementation. RSA is the asymmetric cryptography system. The security of RSA public key cryptosystem is based on the assumption that factoring of a large number (modulus) is difficult. In RSA if one can factor modulus into its prime numbers then the private key is also detected and hence the security of the cryptosystem is broken. The Subset-Sum cryptosystem (Knapsack Cryptosystem) is also an asymmetric cryptographic technique. The Merkle-Hellman system is based on the subset sum problem (a special case of the knapsack problem): given a list of numbers and a third number, which is the sum of a subset of these numbers, determine the subset. In general, this problem is known to be NP-complete. However, if the set of numbers (called the knapsack) is super increasing, that is, each element of the set is greater than the sum of all the numbers before it, the problem is `easy' and solvable

in polynomial time with a simple greedy algorithm. So in this paper a Modified Subset-Sum over RSA Public key cryptosystem (MSSRPKC) is presented which is secure against Mathematical and brute-force attacks on RSA as well as Shamir attacks. This paper also presents comparison between MSSRPKC and RSA cryptosystems in respect of security and performance.

3. Background

This section briefly describes our motivation to take up this study as mode of research. RSA algorithm finds its application in generating cipher text difficult to decrypt. It was used as a means to send keys to receiver when asymmetric algorithm works in conjunction with symmetric cryptosystem. We were dragged by the security of RSA but there exist several brute force attacks to RSA. Then comes Double RSA, enhancing the security of RSA by using double modulus. Before announcing a robust and versatile cryptosystem it is mandatory to review it thoroughly with arbitrary cases. So we try to run some brute force attack on the cipher text generated during double RSA algorithm.

4. RSA

RSA cryptosystem was proposed by Rivest Shamir and Adlemen as asymmetric cryptosystem. The key to the security provided by RSA lies in the modulus (N) generated from the multiplication of two large prime numbers tested using Rabin miller test. Then we generate

$$\emptyset(n) = (P-1) * (Q-1)$$

Where P and Q are two large prime number. The process of generating the public key is to generate a public key in such a manner that greatest common divisor (GCD) of public key and $\emptyset$ is 1. The private key to be generated for decryption should be multiplicative inverse of the public key. This step of finding the multiplicative inverse drags RSA to a class of NP Hard problem [4]. The RSA algorithm designed in this research is responsive to 16 bits integers. The use of integers (generated by ASCII value of the characters) to encrypt and decrypt makes its algorithm more secure. The value of the public key should be found within the range 1 and $\emptyset(n)$ and public key and $\emptyset(n)$ should be co-prime. The private key is generated keeping in mind that it should be a multiplicative inverse of the public key which is main essence of the algorithm.

5. Brute Force Attack

A brute force attack is a process of guessing based on some initial hints provided to solve problem. The result is derived eventually. The brute force attack on RSA was using some public key and modulus. Since using modulus we can derive $\emptyset(n)$ using Euler totient function. Having $\emptyset(n)$ we can generate a table of multiplication modulo in which the column defines the public key and the row define the private key which is the multiplicative inverse of the public key [4] . Another possible brute force attack was using frequency analysis. Since single attack not always reveals the information, we can combine the two. This process can be explained as, when we derive the private key from public key and modulus and apply the key on the cipher text, we may get some of the characters. When we apply frequency analysis based on some standard frequency of the alphabets we can get the entire plain text back.

6. Algorithm Details

6.1. Double RSA

Double RSA was implemented with the cooperation of the sender and the receiver of the data. The procedure for implementing double RSA has assumed that (e, d) pairs of keys and modulus N belongs to the receiver and (g,h) pairs of keys and modulus N1 belong to sender. Where e and g are public keys and d and h are private keys.

- The sender of the data encrypt the message (M) using the public key of the receiver.
  $C(s) = M^g \ mod \ N$
  Where C(s) is the cipher text generated by the sender.
- The receiver on receiving the cipher text starts behaving as sender and encrypts it using the sender's public key.
  $C(r) = M^g mod \ N1$
- Where C(r) is the cipher text generated by the receiver behaving as sender.
- The sender on receiving the cipher text from the receiver decrypts it using his own private key. The receiver on receiving partially decrypted text decrypts further using his own private key and if he manages to get all text back the sender is authenticated and confidentiality of the text is maintained.

6.2. Brute force Attack

The brute force attack is based on guessing based on some early assumptions or results. Though double RSA is secure than RSA still it possess vulnerability due to availability of modulus, public keys and parts of cipher text in unsecure network. There are possibilities of brute force attack on the double RSA that has been mentioned below.

- Brute force 1: In the very first transmission of the cipher text by the sender to the receiver, the eavesdropper obtains the public key of the receiver and the modulus.
  Using the public key and applying brute force method to generate private key as e X_N d=1 that is using e d can be generated by multiplication modulo N.
  Using the modulus N and e we can verify the d as ø(n)=ø(e)*ø(d)= N-(e+d-1), where ø(n) is the Euler's totient function.
- Brute force 2: In the second transmission from the receiver to the sender we have two cipher text C(s) and C(r) in out that will be available to the eavesdropper. So now hacker has two modulus N, N1 and two public keys e and g. Now, if the eavesdropper generates an arbitrary private key h' using g public key and apply on obtained cipher text C(r), generates a text T. The eavesdropper can surely compare T and C(s). If it returns true we have successfully generated the sender private key else repeat the process.
- Brute force 3: At this stage the eavesdropper has both (public key, private key) pairs of the sender and the receiver. He can validate his pairs using brute force 3. In the final transmission of the partially decrypted cipher text. The eavesdropper is made available C2(s), second cipher text generated by the sender. Now he can apply the receiver's private key to obtain the original message.

There is a possibility that we are not with the complete message, but we are with characters that matches the original message, so we can apply frequency analysis algorithm to obtain full correct original message.

7. Simulation Environment

Double RSA was implemented using two famous tools Eclipse IDE and Matlab 2011a. MATLAB was responsible for backend computation of key, cipher text and storing them into the file. MATLAB program files were combined into jar file and were added in the build path of eclipse IDE. The frontend design was made in JAVA using JAVASE 1.7. Java was used, as more security can be implemented using JAVA. The frontend is responsible of taking user input and sending it to the respective matlab file and executing it. The result is captured by the java IDE and is displayed on the applet created in JAVA.

S. no	Algorithm Variable	Value
1.	Modulus(N and ø)	16-32 bits
2.	e, d, g, h (variable)	32 bits
3.	Text 1000 characters	

Table 1: Simulation Environment Details

8. Results

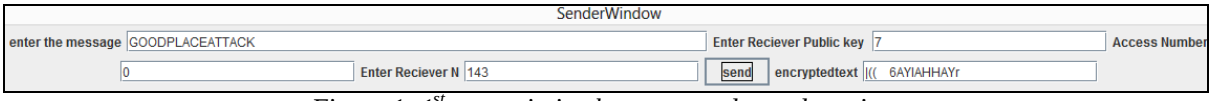

Figure 1: 1st transmission between sender and receiver

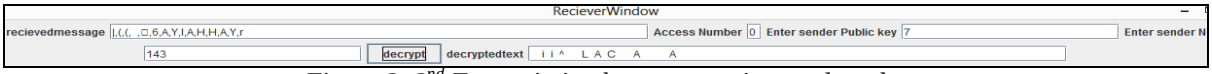

Figure 2: 2nd Transmission between receiver and sender

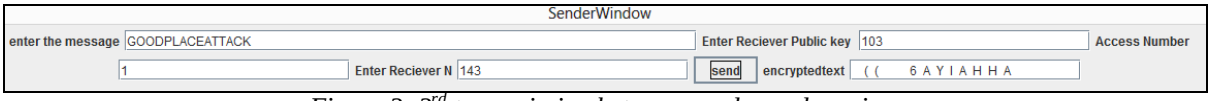

Figure 3: 3rd transmission between sender and receiver

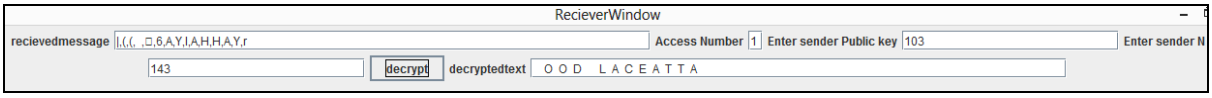

Figure 4: The final message after double RSA with 3 character missing

Figure 5: Brute force attack on the cipher text generated by the DOUBLE RSA

9. Conclusion

We were strongly determined about the security of the message generated using RSA. But with use of Euler totient function, extended Euclidean, it became easy to generate the private key given the public key and modulus. we modified the RSA using double modulo to enhance the security. Double RSA can be implemented in various ways, of which one is mentioned in this research. We started to test double modulo RSA for keep information confidential in the unsecure network. We applied brute force at 1st and 2nd stage of transmission and gathered the public and private key pairs which were applied in the 3rd stage where the original message was to be received by the receiver. When eavesdropper applied the brute force attack based initial set of values we got 12 characters match from 15 character plain text. This was further tested on 1000 character plain text from which 982-986 characters got matched, which shows that deficiency in RSA still prevails in spite enhancing it with double modulo function. This study can be taken as a thought to enhance RSA further using multiple modulo till we get 20-30% character match from the brute force attack.

10. References

1. J. Quisquater and C. Couvreur, "Fast Decipherment algorithm for RSA public–key cryptosystem", Electronic Letters, pp. 905-907, 1982.
2. G.D Sutter, "Modular Multiplication and Exponentiation Architecture for Fast RSA cryptosystem based on Digital Serial Computation", IEEE transaction on Industrial Electronics, pp. 3101-3109, 2011.
3. Michael Ben, "On the cryptographic security of single RSA bits", Proceedings of the fifteenth annual ACM symposium on Theory of computing", pp. 421-430, 1983.
4. Robert S Boyer, "PROOF CHECKING THE RSA PUBLIC KEY ENCRYPTION ALGORITHM", The American Mathematical Monthly, pp. 181-189, 1984.
5. H.Orman, et.al, "Determining Strengths for Public Keys Used for Exchanging Symmetric Keys", RFC3766.
6. Gene Tsudik, "Message authentication with one-way hash functions", ACM SIGCOMM Computer Communication Review, pp. 29-38, 1992.
7. Dan Boneh, "Twenty Years of Attacks on the RSA Cryptosystem", www.ams.org/notices/199902/boneh.pdf