2013 13th International Conference on Control, Automation and Systems (ICCAS 2013)
Oct. 20-23, 2013 in Kimdaejung Convention Center, Gwangju, Korea

Loop-based RSA Key Generation Algorithm using String Identity

Norhidayah Muhammad[1]*, Jasni Mohamad Zain[1], Md Yazid Mohd Saman[2]
University Malaysia Pahang, 26600, Pahang, Malaysia,
(Tel : +60199828284; E-mail: mrs.hidayah@yahoo.com.my), (Tel : +6095492113; E-mail: jasni@ump.edu.my)
University Malaysia Terengganu, 21300, Terengganu, Malaysia,
(Tel : +6096683316;E-mail: yazid@umt.edu.my)

Abstract— This paper will propose i-RSA algorithm, this algorithm is focus on key generation algorithm. Enhancement of this algorithm is user identity can be used as a public key such as email address. Previous algorithm was successful used email identity as a public key, but not all type of email can used as a public key. So we propose i-RSA algorithm that can produces 66.6% compared to previous algorithm (46.67%) email can be a string public key. The differences between i-RSA and previous algorithm are looping process in key generation, to get new value of p and q parameter, looping process will stop when value of k is equal to 1, and the email can be a public key. RSA algorithm and CRC 32 hash function will be explain in preliminaries section. Detail explanations of i-RSA algorithm in propose algorithm section. In future, i-RSA algorithm can be improved, so that 100% of email can be a public key.

Keywords— RSA algorithm; i-RSA algorithm; CRC32 hash function

1. INTRODUCTION

Information security also known as a computer security is the approach to protect information from unauthorized access, stole, disruption, inspection, modification or whatever manipulation of information[1]. The purpose of adopting encryption techniques is to ensure the information confidentiality, integrity and certainty. Cryptographic techniques can be divided into symmetric and asymmetric. In symmetric cryptosystems, the same key is used for the encryption or decryption and this key need to be secure and must be shared between the sender and the receiver[2]. These cryptosystems are very fast and easy to use, but a problem occurs when the key is stolen. Adversary will be able to decrypt the data, since the same key is used for both encryption and decryption.

In asymmetric or public-key cryptosystems, two different keys are necessary: the public and the private keys. With the receiver's public key, the sender encrypts the message and sends it to the receiver who decrypts the message with his private key[3]. The most popular and most widely used public-key cryptosystem is the RSA whose security depends on the difficulty of discovering the private key.

Several techniques have been proposes to enhance RSA algorithm. Hence, the present paper proposes i-RSA algorithm to generate the cryptographic keys required by participants to secure communication using their identities, which is similar to identity based encryption scheme (IBE). In 1984, Shamir[4] proposed public key encryption scheme in which the public key can be an arbitrary string. The basic idea IBE was to use trusted third party called Certificate Authority (CA) to provide trusted public key to the various participants on demand. To setup the hierarchical infrastructure for numerous CA's extra overhead was required. We proposed enhancement of previous algorithm and call it i-RSA it's standing for RSA algorithm using string user identity. The reason why we enhance the previous algorithm is because the previous algorithm cannot use all type of user identity as a public key. Previous algorithm can only used several of email address as a public key and if user email address cannot be a public key, user needs to create another email address. So upgrade this algorithm with one step of looping process in key generation algorithm. i-RSA algorithm shows a better result compared to the previous algorithm. The paper is organized as follows; The preliminaries of RSA algorithm and CRC32 hash function, and the result comparison between proposes algorithm and previous algorithm[5]. The simulation results for CRC32 are addressed in paper which proves the effectiveness of this proposal.

2. PRELIMINARIES OF RSA AND HASH FUNCTION

2.1 RSA Algorithm

RSA is named according to its inventors Ron Rivest, Adi Shamir and Adleman Leonard. RSA supports encryption and digital signature. The keys must be made in such a way that the decryption key may not be easily deduced from the public encryption key[6].

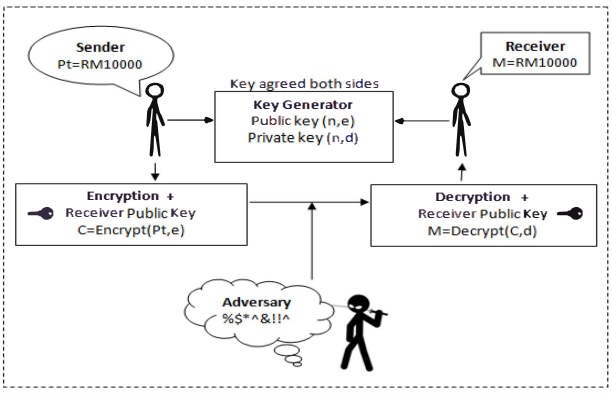

Fig. 1. RSA encryption algorithm

The challenge to factorize an integer large number in two prime numbers makes this cryptosystem secure[7]. Actually, keys of 1024 bits to 2048 bits are commonly used. RSA cryptosystem uses the mode n, the smallest non-negative complete the remaining lines of operation, where n is the product of two different primes p and q[8].

RSA encryption keys are public, while the decryption keys are not, so only the person with the correct decryption key can decipher an encrypted message. RSA algorithm consists of three major steps: Key generation, encryption and decryption.

RSA encryption algorithm can be simply described as follow:

<Key Generation >

- Take the random prime number p, q, and n=p*q.
- Set phi (n)= (p-1) * (q-1).
- Taken an integers e which co-prime for phi(n),where 1<e<n.
- Solution of the variant d by the equation d*e=1 (mod phi(n)), by Modular inverses theorem[9].
- Encrypt (e, n) as a public key.
- Decrypt (d, n) as the private key.

< Encryption >

- $C=Pt^e$ mod n.(c is cipher text)(Pt is plain text)

< Decryption >

- $M=C^d$ mod n. (M is decrypt message)

2.2 CRC Hash Function

A Cyclic Redundancy Check (CRC) or A Cyclic Redundancy Check (CRC) or polynomial code checksum is a hash function designed to detect accidental changes to raw computer data[5]. CRC also known as the CRC code or just CRC. CRCs are so called because the check (data verification) code is a redundancy (it adds zero information to the message) and the algorithm is based on cyclic codes. The term CRC may refer to the check code or to the function that calculates it, which accepts data streams of any length as input but always outputs a fixed-length code. Its computation resembles a polynomial long division operation in which the quotient is discarded and the remainder becomes the result. The length of the remainder is always less than the length of the divisor (called the generator polynomial), which therefore determines how long the result can be CRC32 takes as input a string of any length and gives output of 32 bit.

3. PROPOSED KEY GENERATION ALGORITHM

In this section, RSA cryptographic keys generation algorithm (i-RSA) is proposes. Figure 2 showed the flow of i-RSA key generation algorithm. Email address should be converting to hash function. Any known hash function can be use for this algorithm such as adler32, CRC32, SHA-1 etc. The first step in key generation is choose two random prime number for p and q, after that calculate n = p*q. Next step is calculate value of modulo phi(n)=(p-1)*(q-1). User input their email id as a parameter e, and get decimal value for email id using CRC hash function h(id)= hash (e). After that test either GCD (h(id),phi(n)) = 1 or not. If value of GCD=1, so that email id can be a public key, and if GCD is not equal to one, the process will be looping from the first step to get new value of modulo until GCD equal to one.

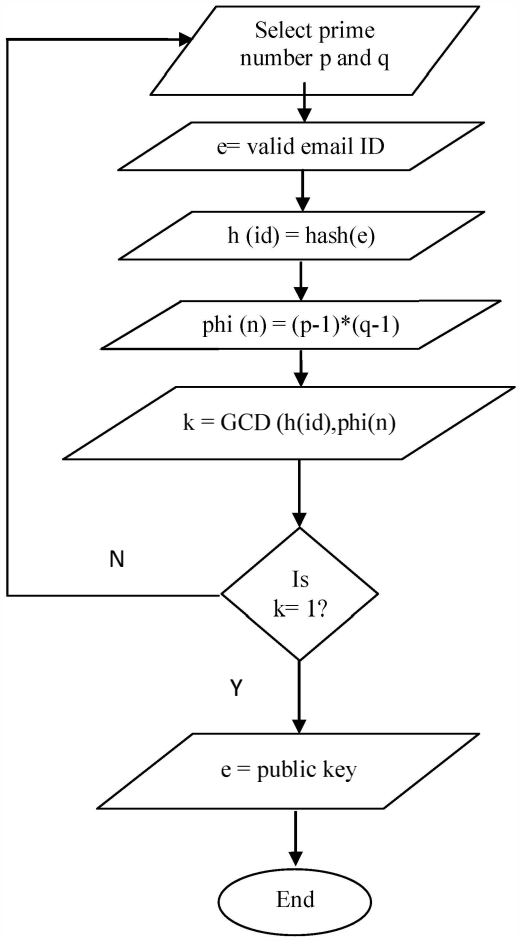

Fig. 2. Show flow chart of i-RSA algorithm key generation using hash function CRC 32.

The whole process of i-RSA algorithm consists of three parts: key generation, encryption and decryption. This proposes method is focused on key generation algorithm. i-RSA will be used user identity as public key. For previous algorithm we did a test either the value of k is equal to one or not by using the same value of phi(n), and if value of k is equal to one, so email id can be a public key, and if k is not equal to one, the email id cannot be a public key[5]. While the method proposed doing a looping process on the parameters p and q and create a different phi(n) value, and calculate until the value of k is equal to one. After key generation processes settle, we continue to the next step which is encryption process. For encryption, the process is the same as an original RSA algorithm, and also for decrypt process.

Table 1 i-RSA algorithm explanation

<Key Generation>

1. Select two parameters, namely p and q, these two parameters must be prime number.
   p= random prime number
   q= random prime number
2. Compute n by using following formula.
   n = p * q
3. Compute phi (n).
   phi (n) = (p – 1) * (q – 1)
4. Input the user valid email Id.
   e = valid email id
5. Convert email id to hash function.
   h(id) = hash(E)
6. Choose the exponent k if :
   GCD (h(id), phi (n)) = 1
   Else, looping the process 1 to 6 and until
   k=1
7. Compute private key exponent d through following formula
   d = e ^ -1 mod (phi (n))

Thus the public key consists of public key exponent e and n. And private key consists of private key exponent d and n.

Public key: (n, e)
Public key: (n, d)

<Encryption>

8. Sender used public key to encrypt the plain text.
   C = m ^ e mod (n)

<Decryption>

9. Receiver used private key to decrypt the cipher text.
   M = C ^ d mod (n)

Table 2 shows an example of the data used for the experiment of i-RSA algorithm. We did an experiment by using email id as a dataset. For the algorithm simulation proposes algorithm is also using email id as a dataset. Fifteen data used in this simulation. These data were compared between the proposed i- RSA algorithm and previous algorithms. Table 2 shows the result of previous algorithm and i-RSA algorithm. Enhancement process in i-RSA show an improvement of email can be a public key. It's better than the previous algorithm.

Figure 3 show the graph of comparison result between i-RSA algorithm and previous algorithm. The green color represent value of public key and the purple color is non public key. Previous algorithm show just 46.67% for public key and 53.33% for non public key, its mean non public key is more than public key for previous algorithm. While i-RSA algorithm shows the value of the public key is 66.67% and for non public key is 33.33%. So the result of this experiment show the proposed algorithm i-RSA can produced more public key from email address compare previous algorithm. This result may be further improved, if larger user email address is considered.

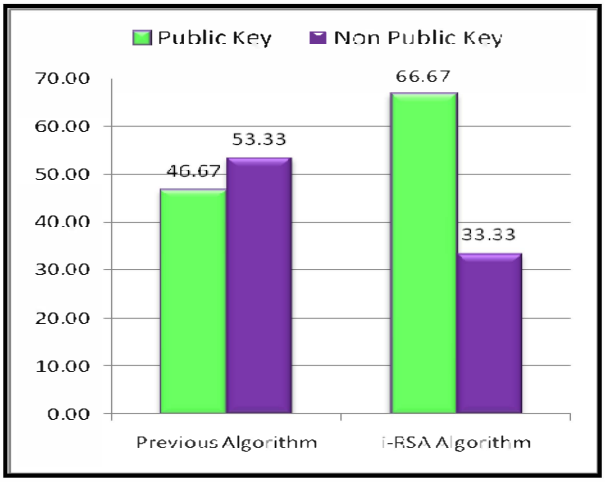

Fig. 3.Result of comparison between previous algorithm and i-RSA algorithm

Table 2 Dataset of email identification

Email	*h(id)*	*Decimal (h(id))*	*i-RSA Algorithm GCD(h(id)), phi(n)*	*Result*	*Previous Algorithm GCD(h(id)), phi(n)*	*Result*
affendi.hashim@gmail.com	3f107afe	1058044670	2	Non public key	2	Non public key
siti_masrina@yahoo.com	99b279b8	2578610616	12	Non public key	12	Non public key
sameh.shenoda@ci.menofia.edu.eg	82edae41	2196614721	1	Public key	7	Non public key
ivanmanurung@rocketmail.com	b8bdb3ac	3099440044	3	Non public key	3	Non public key
yayank@usu.ac.id	f4fc1dfb	4110163451	1	Public key	1	Public key
marischaelveny@ymail.com	60beab5d	1623108445	1	Public key	1	Public key
apank_abank@yahoo.co.id	270165e2	654403042	2	Non public key	2	Non public key
gayoayu@yahoo.co.id	0d489287	222859911	1	Public key	1	Public key
aini@tmsk.uitm.edu.my	94f425e9	2499028457	1	Public key	1	Public key
asma@fskik.upsi.edu.my	f6b6cc0e	4139174926	2	Non public key	2	Non public key
sufi_anto@usu.ac.id	030aabbd	51030973	1	Public key	1	Public key
mehmood99_ahmed@Yahoo.com	fcec5467	4243346535	1	Public key	1	Public key
vani_hardi@yahoo.com	4b247a0d	1260681741	1	Public key	9	Non public key
maizan@unisza.edu.my	0f41e5e5	255976933	1	Public key	1	Public key
ringorbnsrg@students.usu.ac.id	9ea01461	2661291105	1	Public key	3	Non public key

CONCLUSION

The key certificates are used to authenticate the user's key pair. So certificate does work as important role in secure communication but to issue the certificate is a big challenge and it also increases the overhead due to the increasing cost. Therefore, previous algorithm create a RSA algorithm using identity as a public key, but the result for previous method can be improve by using i-RSA algorithm, because of i-RSA algorithm doing a looping process for modulo until the email can be transform as public key. Comparison has done between previous algorithm and i-RSA algorithm and the result show i-RSA algorithm can produce a better result. In future i-RSA algorithm can be improved to produce all email address can be a public key

ACKNOWLEDGEMENT

This study is funded by Skim Latihan Akademik Ipta (Slai) under the Malaysia Ministry of Higher Education (MOHE) and Universiti Sultan Zainal Abidin (UniSZA) Malaysia. The authors would like to acknowledge all contributors, and others who have helped and greatly assisted in the completion of the study.

REFERENCES

[1] W. Diffie and E.M. Hellman, "New Directions in Cryptography," *IEEE Transactions on Information Theory*, vol. 22, pp. 644-654, 1976.

[2] C.C. Tan, W. Haodong, Z. Sheng, and L. Qun, "IBE-Lite: A Lightweight Identity-Based Cryptography for Body Sensor Networks," *IEEE Transactions on Information Technology in Biomedicine*, vol. 13, pp. 926-932, 2009.

[3] N. Anane, M. Anane, H. Bessalah, M. Issad, and K. Messaoudi, "RSA Based Encryption Decryption of Medical Images," in *Proc. 7th International Multi-Conference on Systems Signals and Devices (SSD)* 2010, pp. 1-4.

[4] A. Shamir, "identity based cryptosystems and signature schemes," *Advances in Cryptology-Crypto* vol. 196, pp. 47-53, 1984.

[5] S. Tripathi, G.P. Biswas, and S. Kisan, "Cryptographic keys generation using identity," in *Proc. 3rd International Conference on Advances in Recent Technologies in Communication and Computing (ARTCom 2011),* 2011, pp. 148-151.

[6] E. Milanov, *The RSA Algorithm*: 2009.

[7] G.B. Agnew, R.C. Mullin, I.M. Onyszchuk, and S.A. Vanstone, "An Implementation for a Fast Public-Key Cryptosystem," *Journal of Cryptology* vol. Volume 3, pp. 1991.

[8] S. Wang and G. Liu, "File Encryption and Decryption System Based on RSA Algorithm," in *Proc. 2011 International Conference on Computational and Information Sciences (ICCIS),* 2011, pp. 797-800.

[9] Y.S. Yan, *Number Theory for Computing*: Springer, 2002.