International Journal of Technical Research and Applications e-ISSN: 2320-8163,
www.ijtra.com Special Issue 28 (August, 2015), PP. 1-6

USING HYPERCUBE INTERCONNECTION NETWORKS FOR RSA CRYPTOGRAPHY

[1,2]Kamal Jadidy Aval, [1]Masumeh Damrudi*
[1]Department Department of Computer Science,
Firoozkooh Branch, Islamic Azad University, Firoozkooh, Iran
[2]Department of Computer System and Communication,
Faculty of Computing,
Universiti Teknologi Malaysia,
81310, Skudai, Johur Bahru, Malaysia
*m.damrudi@gmail.com

***Abstract*— The RSA (Rivest, Shamir and Adleman) is one of the most preferred algorithms for asymmetric cryptography. Modular exponentiation is the time consuming part of RSA algorithm. It's a common way to use parallel processing to achieve faster results in terms of speed. In this study, a new method exploiting parallel processing is proposed for RSA cryptography. This parallel method employs the hypercube interconnection network that has been attempted for the first time in this paper. The theoretical foundations show that the encrypted data resulting from the proposed method and the original RSA are the same and the method can be employed to make the RSA parallel which leads to speedup.**

***Index Terms*— Parallel processing; cryptography; RSA; hypercube**

I. INTRODUCTION

Nowadays, almost all of our activities are tightly coupled to the computers in variety of forms from controlling room temperature to using GPS navigators or sending emails. Due to the necessity of speed in computations, parallel processing of various operations has become a prevalent science. Security is one of the most important parts of mentioned activities, and cryptography is the most popular way of having secured information. Developing the field of parallel processing in the cryptography, researchers launched to issue new parallel methods for cryptography algorithms such as RSA. Almost all state of the art parallel approaches in the field of cryptography are summarized in [1], which can be used as a reference for further studies. To the best of authors' knowledge, none of these approaches are exploiting Interconnection Networks (INs) for their solutions.

This paper presents a new method employing hypercube interconnection network for RSA algorithm, which improves results in terms of speed. Hypercube is one of the common and popular INs [2-4]. As a technical specification, a proper IN should keep the network diameter as short as possible and at the same time minimize the topological network cost [5-7]. The hypercube interconnection network has this advantage due to its natural attractive properties [5]. This paper presents the CRSA(Cube based RSA) algorithm to indicate that using hypercube to make RSA parallel is applicable.

The rest of this article is organized as follows. Firstly, we describe the hypercube interconnection network in the next section. Then, the RSA as a public cryptography is explained. Section 4 provides the cube based RSA method. Afterwards, theoretical foundations are presented. Finally, the conclusion of the work is drawn.

2. Hypercube Interconnection Network

One of the most important factors with high impact on network performance and its power consumption is topology [6]. Intrinsic properties of the hypercube such as strong connectivity, regularity, topological symmetry, and recursive constructions size have made it attractive [3]. Because of the hypercube's ability to emulate a various of other frequently used networks and its elegant topological properties, it is one of the most popular interconnection networks or parallel

computer systems [8]. Cube is one of the most practical INs among existing interconnection networks. Some practical approaches on this interconnection network are explained in the following.

The cube interconnection network has been widely used in a variety of parallel systems such as Intel iPSC[9], the Connection Machine CM-2 [10], the nCUBE [9, 11], and SGI Origin 2000 [12, 13]. The hypercube is a loosely coupled parallel processor according to the binary *n*-cube network. An *n*-cube is composed of 2^n processors that are identical. The hypercube architecture, which is demonstrated in Fig. 1, has some limitations. Primarily, the *k*-dimensional hypercube has $N=2^k$ vertices, so its structure is restricted to have exactly 2^k nodes [14].

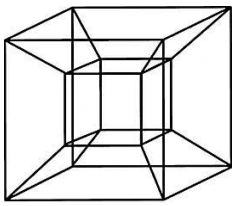

Fig. 1 Hypercube interconnection network

Two reasons for the ability of hypercube to perform many computations at high speed are the linear bisection width and the logarithmic diameter [9, 15]. The communication diameter is logarithmic in terms of the number of processor elements, which is the same as the tree, pyramid, and mesh-of-trees. The bisection width of the hypercube is linear in terms of the number of processor elements, which is a significant improvement over the bisection width for the mesh, pyramid, and mesh of trees. Therefore, it is possible to move large amounts of data quite efficiently. Any two disjoint sub-cubes of size *n/2* are connected by exactly *n/2* communication links. That is, the bisection width of a hypercube at the size of *n* is $\Theta(n)$ [16]. In the hypercube architecture of size *N*, the degree and diameter are the same, which is $log_2\ N$ [15, 16] and because of this specification, a good balance between the complexity of the network topology and the communication speed is achieved [14].

3. RSA as a public cryptography

Security services take the important role of secure transferring confidential information such as master cards' passwords through unreliable networks. The prominent way to provide this service is using cryptographic algorithms.

"The development of public-key cryptography is the greatest and perhaps the only true revolution in the entire history of cryptography" [17] and at the same level, RSA was one of the first great advances in asymmetric key cryptography, which has been widely used for years in many applications such as electronic commerce and trading protocols [18-20]. Public-key cryptography has high flexibility that can be applied to most of the cryptographic systems being used these days [21]. On the other hand, RSA is one of the safest standard algorithms for providing security in networks based on public-key [22]. This cryptography is secure although it is not as fast as symmetric cryptography.

Public-key was invented mostly to solve the certain problem of secret key distribution [21, 23]. Secret key distribution is a tough problem. Managing a secret key is only feasible for a small network. Secret key cryptography, by itself, is not an option for a network of millions of users; that is why it cannot support internet e-commerce [24]. It has a big disadvantage on key exchange and limited on the number of keys, which is up to n^2 in symmetric key, where *n* is the number of participants in the group. A great number of keys are needed to be maintained in a group. While in Public-key cryptography, the number of keys that are needed to be maintained in the group is only $2n$ [21].

RSA is widely used because it is in an acceptable level of utilization. Unlike symmetric algorithms, the problem is that this algorithm is not fast enough; that is why many applications use it just to exchange keys. This well known algorithm is applied to many systems for encryption and decryption [25] such as the ones from CISCO®, therefore, making it faster can help many applications to run rapidly and more secure.

4. Cube based RSA (CRSA)

Hypercube topology is used as the parallel processing architecture in CRSA. A hypercube, which is also called *n*-cube is created based on cubes. The RSA cryptosystem[26] is described in almost all cryptographic papers on RSA, which in brief is as following:

$n=pq$

$\varphi=(p-1)(q-1)$

$e<n$, $gcd(e, \varphi)=1$

$d=e^{-1} \bmod \varphi$

Let m be the plaintext. It must be divided into blocks smaller than n. $C_x=m_x^e \bmod n$ is the result of encryption and $m_x=C_x^d \bmod n$ is the result of decryption. The block number is shown by x. As the core calculation part of RSA, exponentiation is a time-consuming operation, especially in presence of greater keys.

In the proposed design, in addition to an *n*-cube an extra node is also needed as the coordinator. From technological point of view, the architecture can work as a cryptographic coprocessor, which collaborates with the CPU. In this case, the coordinator can be the CPU itself. Digital embedded systems do have a main processor, and this coprocessor is used to perform cryptographic operations faster and with a higher level of throughput. The number of nodes in the coprocessor can be different due to the speed needs of the application. However, it cannot be less than eight illustrating one cube. Using more nodes, the more parallelization will be gained and at the same time employment of more processor elements for smaller data and key lengths should be avoided. There is a tradeoff between the number of processor elements, the data size and the key length. The parallel calculation increases the speed of operations, which is multiplication. A hypercube with 16 nodes has been chosen to describe the solution, although the equations are provided in general form. The following definitions are used for the CRSA algorithm. It should be considered that each PE has received its input value(s) from prior PE(s) and sends the results to the subsequent PEs. This "send and receive" is formalized into the following equations where variable i present the index of PEs.

(1) $\forall i, i\in\{i \bmod 8=0\}$; p_i computes $p_{i-8} \times p_{i-8} \bmod n$

(2) $\forall i, i\in\{i \bmod 8=1\}$; p_i computes $p_{i-1} \times p_{i-8} \bmod n$

(3) $\forall i, i\in\{i \bmod 8=2\}$; p_i computes $p_{i-2} \times p_{i-8} \bmod n$

Except for i=10 p_i computes $p_{i-2} \times p_{i-2} \bmod n$

(4) $\forall i, i\in\{i \bmod 8=3\}$; p_i computes $p_{i-1} \times p_{i-2} \times p_{i-8} \bmod n$

(5) $\forall i, i\in\{i \bmod 8=4\}$; p_i computes $p_{i-4} \times p_{i-8} \bmod n$

(6) $\forall i, i\in\{i \bmod 8=5\}$; p_i computes $p_{i-1} \times p_{i-4} \times p_{i-8} \bmod n$

(7) $\forall i, i\in\{i \bmod 8=6\}$; p_i computes $p_{i-2} \times p_{i-4} \times p_{i-8} \bmod n$

Except for i=14 p_i computes $p_{i-2} \times p_{i-2} \times p_{i-4} \bmod n$

(8) $\forall i, i\in\{i \bmod 8=7\}$; p_i computes $p_{i-1} \times p_{i-2} \times p_{i-4} \bmod n$

if i<N-1; p_i computes $p_{i-1} \times p_{i-2} \times p_{i-4} \times p_{i+8} \bmod n$

The above equations apply to the architectures that include more than one cube. The equations for the first cube are as following that, according to the algorithm, should be calculated before above equations.

(9) $\forall i, i\in\{1,4\}$; p_i computes $p_0 \times p_0 = A \times A$

(10) *if* $i=2$; p_i receive and send $p_0=B$

(11) *if* $i=3$; p_i computes $p_1 \times p_1$

(12) *if* $i=5$; p_i computes $p_1 \times p_4$

(13) *if* $i=6$; p_i computes $p_2 \times p_4$

For brevity, we represent one multiplication together with one modulus as MulMod. About half of the processor elements, need one MulMod block while the other half need two MulMod blocks. In this example, PEs indexed 1, 3, 4, 5, 6, 8, 9, 10, and 12 perform only one MulMod while the others carry out two MulMod operations except PEs indexed 0 and 2 that do not perform any MulMod and just pass through the values, and PE7, which performs three multiplications. These PEs are demonstrated in Fig. 2. This figure shows the scheme

of the CRSA and processor element indexing in the hypercube topology.

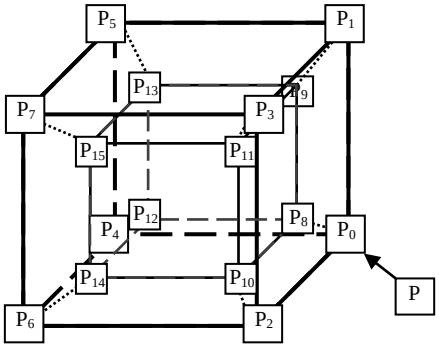

Fig. 2 Processor elements in 2-Cube of hypercube architecture

In Fig. 2 the processor P is considered as the coordinator. The structure of one MulMod block is shown in Fig. 3.

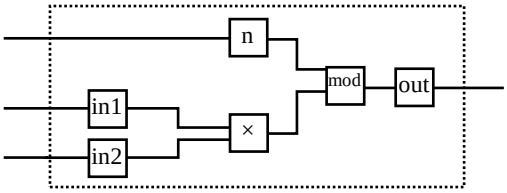

Fig. 3 MulMod structure

Two inputs are multiplied and the modulus of the results to *n* is transmitted as output. The previously presented equations illustrate the operations inside and between the processor elements.

5. Theoretical foundations

Theoretical bases are discusses in the following. It is proved that the results of CRSA are the same as RSA. The following equations demonstrate the values employed by these PEs to calculate the equations. The key is divided into the value e_d as following to perform parallel operations for each part.

The value e_d, which is fed into the *p* (coordinator) to get the results from previous equations is:

(14) $e_d = 10 + 3 \times 2^2 \sum_{i=3,j=4}^{i=2^{N/8}-1} i \; (i = 2j - 1, j = 2j)$

The values e_r and e_o are computed using the following formula:

(15) $e_o = e \; div \; e_d$

(16) $e_r = e \bmod e_d$

The pseudo code of CRSA algorithm for two cubes is described as following:

CRSA Algorithm for two cubes

$e_o = e \; div \; e_d$
$e_r = e \bmod e_d$
$A = m^{eo} \bmod n$
$B = m^{er} \bmod n$
in par; p_1, p_4, p_8: $A \times A \pmod n$, p_2:B
in par; $p_3, p_5, p_9, p_{10}, p_{12}$: $A^2 \bmod n \times A^2 \bmod n \pmod n$, p_6: $A^2 \bmod n \times B \pmod n$
in par; p_{11}, p_{13}, p_{14}: $A^4 \bmod n \times A^4 \bmod n \times A^4 \bmod n \pmod n$
p_{15}: $A^{12} \bmod n \times A^{12} \bmod n \times A^{12} \bmod n \pmod n$
p_7: $A^{36} \bmod n \times A^4 \bmod n \times A^4 \bmod n \times A^2 \bmod n \times B \pmod n$

In the first step, the processor element *p*, calculates $m^{e_o} \bmod n$ using constant length nonzero windows of sliding window technique [27] and sends it as the input of processor element p_0 to send it to processor elements p_1, and p_4. Then the result of equation $m^{e_r} \bmod n$, which is retrieved from the results of $m^{e_o} \bmod n$ in *p*, is sent to processor element p_1 to send it as input of processor elements p_2. The outputs of *p* are *A* and *B*, which are computed as following.

(17) $A = m^{e_o} \bmod n$

(18) $B = m^{e_r} \bmod n$

p_0 receives *A* and sends it to p_1 and p_4. Afterwards, p_0 sends *B* to p_2. Then equations (9) to (13) are performed for the first cube. Following these instructions, for the inner cube, equations (1) to (8) are done. The output of p_7 is the encryption of *m*. These operations are carried out in parallel. Fig. 4, specifies which PEs work in parallel in each step.

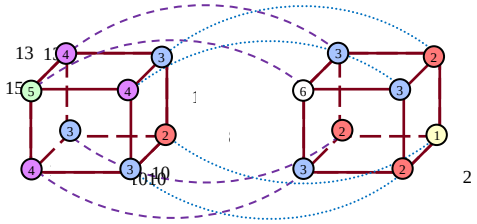

Fig. 4 CRSA algorithm on hypercube architecture using the second view of scheme

In the above figure, each PE has a colour and number, which shows which PEs are operating in parallel during each step. The flows of operations are started from *PE0* and are finished in *PE15*, although the results are in *PE7*.

No matter how many nodes are employed in the architecture; the result of CRSA will always be obtained from *PE7*. CRSA method has improved the modular exponentiation process as following to do faster computations:

$r=e \bmod e_d$

$m^e = m^{e/ed} \times \ldots \times m^{e/ed+r}$

$m^e = m^r \prod_{i=1}^{e_d} m^{e/e_d}$

The sequence of operations, which are being done to carry out RSA encryption are presented in the following. Using equation (14) we have:

$$(19) \qquad e_d = 10 + 3 \times 2^2 \sum_{\substack{i=2^{\frac{N}{8}}-1 \\ i=3,j=4}} i \,(i = 2j-1, j = 2j)$$

$$= 10+3\times 2^2(3)=46$$

According to a known fact in number theory,

$(a\times b) \bmod n = ((a \bmod n) \times (b \bmod n)) \bmod n$

According to the results of p_7:

$C = A^{36} \bmod n \times A^4 \bmod n \times A^4 \bmod n \times A^2 \bmod n \times B \pmod n$

$= ((A^{46}) \bmod n \times B \bmod n) \bmod n$

$= ((A)^{46} \bmod n \times B \bmod n) \bmod n$

It is derived from equations 15 and 16 that:

$= ((m^{eo} \bmod n)^{46} \bmod n \times (m^{er} \bmod n) \bmod n) \bmod n$

Utilizing equation 19:

$= ((m^{eo} \bmod n)^{ed} \times (m^{er} \bmod n)) \bmod n$

$= ((m^{eo})^{ed} \times m^{er}) \bmod n$

$= ((m^{eo})^{ed})^{+er} \bmod n$

And using equations 15 and 16:

$= m^e \bmod n$ □

This shows that the result coming out of CRSA is the same as main RSA.

II. Conclusion

This paper presents a new method for RSA encryption with hypercube interconnection network, which improves the speed. The main innovation in this paper is the use of hypercube interconnection network to encrypt messages. The theoretical analysis shows that the encryption resulting from CRSA and RSA are the same. In the future work, it should be analyzed to see how CRSA is better than other methods in terms of speedup using computational complexity and simulation results.

References

[1] M. Damrudi and N. Ithnin, "State of the Art Practical Parallel Cryptographic Approaches," *Australian Journal of Basic and Applied Sciences*, vol. 5, pp. 660-677, 2011.

[2] S. Dobrev, *et al.*, "Black hole search in common interconnection networks," *Networks*, vol. 47, pp. 61-71, 2006.

[3] D. K. Guo, *et al.*, "KCube: A novel architecture for interconnection networks," *Information Processing Letters*, vol. 110, pp. 821-825, Sep 2010.

[4] N. Kapoor, *et al.*, "A genetic algorithm for finding the pagenumber of interconnection networks," *Journal of Parallel and Distributed Computing*, vol. 62, pp. 267-283, Feb 2002.

[5] M. Abdullah, *et al.*, "The chained-cubic tree interconnection network," *International Arab Journal of Information Technology*, vol. 8, pp. 334-343, 2011.

[6] M. Hoseiny Farahabady, *et al.*, "Some topological and combinatorial properties of WK-recursive mesh and WK-pyramid interconnection networks," *journal of Systems Architecture*, vol. 54, pp. 967-976, 2008.

[7] N. G. Kini, *et al.*, "Torus Embedded Hypercube Interconnection Network: A Comparative Study," *International Journal of Computer Applications*, vol. 1, pp. 29-31, 2010.

[8] L. Yamin, *et al.*, "Efficient communication in metacube: a new interconnection network," in *Proceedings International Symposium on Parallel Architectures, Algorithms and Networks. I-SPAN'02*, Place of Publication: Los Alamitos, CA, USA; Makati City,

Metro Manila, Philippines. Country of Publication: USA., 2002.

[9] H. El Rewini and M. Abd El Barr, *Advanced Computer Architecture and Parallel Processing*. USA: John Wiley & Sons Publishing, 2005.

[10] L. W. Tucker and G. G. Robertson, "Architecture and applications of the Connection Machine," *Computer*, vol. 21, pp. 26-38, 1988.

[11] J. P. Hayes and T. Mudge, "Hypercube supercomputers," *Proceedings of the IEEE*, vol. 77, pp. 1829-1841, Dec. 1989 1989.

[12] C. R. Inc and S. G. Inc, "Origin2000 employs SSI," *Computer Dealer News*, vol. 14, p. 1, 1998.

[13] D. Jiang and J. P. Singh, "A methodology and an evaluation of the SGI Origin2000," *SIGMETRICS Perform. Eval. Rev.*, vol. 26, pp. 171-181, 1998.

[14] R. K. Katare and N. S. Chaudhari, "Study of topological property of interconnection networks and its mapping to sparse matrix model," *International Journal of Computer Science and Applications*, vol. 6, pp. 26-39, 2009.

[15] B. Parhami, *Introduction to Parallel Processing Algorithms and Architectures*. Santa Barbara, California: Kluwer Academic Publishers, 2002.

[16] R. Miller and L. Boxer, *Algorithms Sequential and Parallel: A Unified Approach*, Second Edition ed. Hingham, Massachusetts, 2005.

[17] W. Stallings, *Cryptography and Network Security Principles and Practices,*, Fourth ed. USA: Prentice Hall, 2006.

[18] W. Fan, *et al.*, "Parallelization of RSA Algorithm Based on Compute Unified Device Architecture," in *9th International Conference on Grid and Cooperative Computing (GCC)* Nanjing, China, 2010, pp. 174-178.

[19] G. Perin, *et al.*, "Montgomery modular multiplication on reconfigurable hardware: Fully systolic array vs parallel implementation," in *VI Southern Programmable Logic Conference (SPL)*, Ipojuca 2010, pp. 61-66.

[20] P. Lara, *et al.*, "Parallel modular exponentiation using load balancing without precomputation," *Journal of Computer and System Sciences*, vol. 78, pp. 575-582, 2012.

[21] T. Teerakanok and K. Kamolphiwong, "Accelerating asymmetric-key cryptography using Parallel-key Cryptographic Algorithm (PCA)," in *6th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology(ECTI-CON)*, Pattaya, Chonburi, Thailand, 2009, pp. 812-815.

[22] H. Thapliyal and M. B. Srinivas, "VLSI Implementation of RSA Encryption System using Ancient Indian Vedic Mathematics," in *Proceedings of the Micro technologies of The New Millennium(VLSI Circuits and Systems)*, Sevilla, Spain, 2005, pp. 888-892.

[23] P. Thorsteinson and G. G. Arun Ganesh, *.NET Security and Cryptography*. USA: Pearson Education Inc., Prince Hall, 2004.

[24] H. X. Mel and D. Baker, *Cryptography Decrypted*. USA: Addison Wesley, 2001.

[25] S. Sepahvandi, *et al.*, "An Improved Exponentiation Algorithm for RSA Cryptosystem," in *International Conference on Research Challenges in Computer Science, ICRCCS '09.* , Shanghai, 2009, pp. 128-132.

[26] R. L. Rivest, *et al.*, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM* vol. 21, pp. 120-126, February 1978.

[27] C. K. Koc, "High-Speed RSA Implementation," RSA Data Security, Inc., Redwood City, CA, USA1994.

[1] .