I.J. Computer Network and Information Security, **2014, 4, 63-71**
Published Online March 2014 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2014.04.08

CBC and Interleaved CBC Implementations of PACTS Cryptographic Algorithm

J. John Raybin Jose
Department of Information Technology, Bishop Heber College (Autonomous), Tiruchirappalli, India.
raybinjose@yahoo.com

E. George Dharma Prakash Raj
School of Computer Science, Engineering & Technology, Bharathidasan University, Tiruchirappalli, India.
georgeprakashraj@yahoo.com

***Abstract* —** PACTS (Parallelized Adaptive Cipher with Transposition and Substitution) is a new class of Symmetric Cryptographic Algorithm designed using traditional techniques to efficiently utilize the parallel computing capabilities of the modern computing systems. It overcomes the performance inconsistencies prevalent in conventional cryptographic algorithms when they are implemented in different computing systems with different processing capabilities. The size of the key and the plain text blocks of PACTS are each 1024-bits. The adaptive nature of this algorithm is achieved by incorporating flexibility in the size of the key and plain text sub-blocks and the number of rounds. Level of Intra-packet parallelization, variety in grain size and the required security strength are achieved by suitably deciding the sub-block size. Flow of the algorithm is made dynamic by determining the execution steps at runtime using the value of the key. In spite of these advantages PACTS always produces the same cipher text block for a particular plain text block when the same key is used. CBC mode along with 2-way and 4 way Interleaved CBC modes are employed to overcome this problem. The performance of the PACTS in ECB, CBC and Interleaved CBC modes are analyzed with implementations in shared memory parallel computing environment using Open MP, Java Threads and MPI.

***Index Terms* —** Symmetric Block Cipher; Parallel Cryptography; Adaptive Cryptography; Cipher Block Chaining; Interleaving; Transposition; Substitution.

I. INTRODUCTION

Traditional cryptographic algorithms focus only on the complexity of the algorithm and the strength and the secrecy of the key [1]. We face variety of challenges in efficiently implementing these crypto systems as new trends and technologies have crept into modern communication and computing systems. Conventional symmetric cryptographic algorithms such as DES, IDEA, RC6, Blowfish and AES are developed before the year 2000 when computers were built around single 32, 16 or even 8 bits processors. But now, Cryptographic algorithms are executed much faster on modern computers. The present day computing systems and that of future are not that of single core 32-bits desktops, but of multi-cored chips and multiprocessor machines whose processing capacities are 64 or 128 or more bits. Parallelizing the cryptographic algorithms is the only means to utilize these systems productively [2].

Nowadays there is a sharp increase in the rate of encryptions and decryptions carried out per unit time as the amount of information passing through communication networks have increased exponentially. This imposes additional overhead in the information exchange and may cause congestion. A way out of this trouble is to develop a new class of parallel algorithms that can reduce the time required for encryption and decryption without diminishing the security strength.

Cryptographic algorithms are broadly classified as Symmetric and Asymmetric Key Algorithms. Symmetric key algorithms use only one key and they rely on the secure distribution and management of the key, which is used both for encryption and decryption. Symmetric key algorithms are further divided into stream ciphers and block ciphers. Stream ciphers encrypt the bytes of the message one at a time but block ciphers take a number of bytes and encrypt them as a single unit. In Asymmetric Key Algorithms a public key and a private key are used. The public key is used to encrypt the information at the sending end and it is available to all; whereas the private key is known only to the receiver and it is used to decrypt the information [3]. Symmetric Block Ciphers are involved in this work.

Substitution and transposition techniques are employed in PACTS. Substitution replaces each character by another, whereas transposition permutes the characters in a block of data. Substitution causes confusion in a cipher and adds more complexity to the algorithm in finding a relationship between the key and the cipher text from one side and the key and the plaintext in another side. Transposition causes diffusion and makes sure that there is no local relationship between the statistics of the characters in the plaintext and the cipher text [4, 5].

Adaptive Cryptography is a trend in Cryptography, which focuses on attaining flexibility in the cryptographic implementations by dynamically varying the algorithmic flow and the choice of the key and the plain-text sub-blocks. Adaptive Cryptographic techniques can be classified as (i) Inter-Algorithmic Adaptive Techniques and (ii) Intra-Algorithmic Adaptive Techniques.

Inter-Algorithmic Adaptation is achieved by employing different algorithms [6], whereas Intra-Algorithmic Adaptation instills dynamism within the same algorithm. Intra-Algorithmic Adaptation is employed in this work.

Parallel Cryptography is a recent development, which deals with implementation of cryptographic algorithms in modern parallel computing environments. Parallelism accelerates processing by simultaneous execution of multiple tasks. Implicit parallelism is achieved by the inherent resources and techniques in the processing hardware. Explicit parallelism is extracted by the external arrangements and codes by utilizing the available parallel hardware resources efficiently. Techniques used for explicit parallelism can be categorized as (i) Per-Connection Parallelism (ii) Per-Packet Parallelism and (iii) Intra-Packet Parallelism.

Per-connection parallelism is a method in which each connection is given its own thread or process that runs exclusively on one processor. This is the most common method of parallelization, and requires no modification to the existing algorithm or the software implementation. The per-connection parallelization method makes no attempt to fully utilize modern architectures.

In Per-packet parallelism connections disperse their packet processing load over multiple processors, wherein each packet is treated individually. Many current algorithms lend themselves well to this kind of parallelization, but, no cryptographic software implementing this per-packet parallelism is available.

Intra-packet parallelism is the most difficult type of parallelism, as it depends on algorithm design. It also requires changes in the implementation of the cryptographic algorithm, depending no longer on the flexibility of the hardware or operating system upon which it is run [2], [7]. Intra-packet parallelism is employed in PACTS.

This paper is planned as follows. Section II gives the related works, Section III depicts the ECB implementation of PACTS, Section IV gives the CBC implementation of PACTS, Section V deals with its Interleaved CBC implementation, and Section VI concludes the paper.

II. RELATED WORKS

Efforts to parallelize existing cryptographic algorithms have been pursued by many researchers from year 2000 onward. The prominent of these efforts can be classified broadly as Hardware or Software Parallel Cryptographic Implementations involving several technical approaches beneath them as shown in Fig 1.

Parallelization with RISC Processors by HoWon Kim et al., introduced a special-purpose microprocessor known as crypto processor. It had a 32-bit RISC processor block and a coprocessor block dedicated to SEED and TDES [8]. Pionteck et al., in their work presented a hardware design of AES with reconfigurable encryption/decryption engines which supports all key lengths [9].

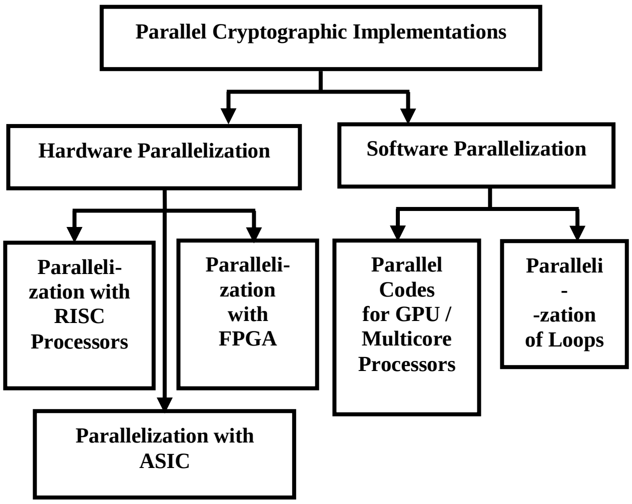
Parallel Cryptographic Implementations
Hardware Parallelization
Software Parallelization
Paralleli-zation with RISC Processors
Paralleli-zation with FPGA
Parallel Codes for GPU / Multicore Processors
Paralleli- -zation of Loops
Parallelization with ASIC

Figure 1: Classification of Parallel Implementation of Cryptographic Algorithms

An Application-Specific Integrated Circuit (ASIC) is an Integrated Circuit customized for a particular use. ASICs provide robust operation and much of the overhead involved is reduced. The works carried out on ASIC implementation of DES, 3DES, IDEA and AES by S. Mukherjee et al., T. Ichikawa et al., and B. Weeks et al., are prominent in this category [10]-[12].

Field Programmable Gate Array (FPGA) logic cells are reconfigurable platforms that provide low cost, high performance implementations of Block Ciphers. Several standard algorithms such as DES, TDES, and AES are parallelized using FPGAs by Swankoski et al., Kotturi, et al., and Chi-Wu, et al. [13]-[15].

Multi-core Processors and Graphical Processing Units (GPUs) are used to parallelize the existing Cryptographic algorithms to enhance their performance. CUDA programming is used to parallelize algorithms in GPU and OpenMP is used to extract parallelism from Multi-core Processors [16]-[19]. Praveen Dongara et al., implemented several symmetric cryptographic algorithms in ECB, CBC and interleaved CBC modes [20]. Similar works with CBC and Interleaved CBC modes were also carried out on AES by Zadia Codabux-Rossan et al. [21] and Ashokkumar et al. [22].

The most time-consuming elements of source code of cryptographic algorithms without including the I/O functions are loops, they are parallelized for all the popular cryptographic algorithms such as DES, Triple DES, IDEA, AES, RC5, Blowfish, GOST and LOK191 by Burak et al. in standard modes of operations such as ECB, CBC, CFB, OFB and CTR modes [23] [24].

Even though all the efforts to parallelize the existing conventional cryptographic algorithms with hardware and software techniques had given better results, they cannot be fully parallelized or implemented efficiently in present day computing systems. The dependency problems and the inability to efficiently modularize the sections of the algorithms hover around and haunt the parallelization. Thus a path for the new class of cryptographic algorithms that is devoid of these problems is set in.

III. ECB IMPLEMENTATION OF PACTS

In our previous work we have developed the Parallel Adaptive Cipher with Transposition and Substitution Techniques (PACTS) and implemented in ECB mode [25].PACTS is a symmetric block cipher with block length and key size each of 1024 bits. The sub-block size of key and the plain text is varied based on the computing environment used. The flow of the algorithm is decided dynamically by deriving the control information from the key. The granularity of the algorithm is decided by forming sub-blocks of various sizes in the range 2n where n=3 to 8. The processing resources available and the security strength required are used to decide the number of rounds and size of the sub-blocks. This is depicted in fig 2. The key is applied directly in the first round but the key transformations for successive rounds are achieved by rotating resultant key after the first round to the left or right by the value determined by its first eleven bits.

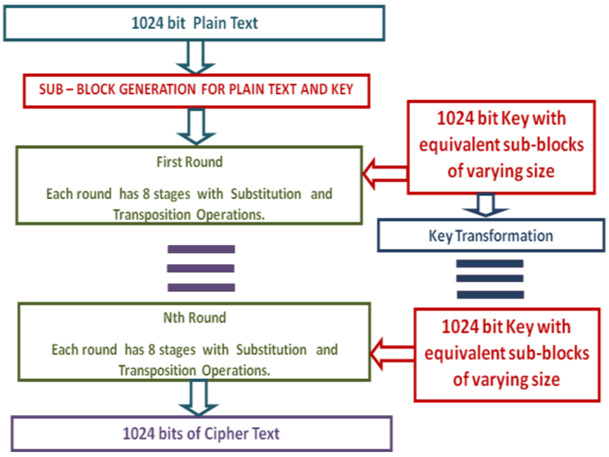
1024 bit Plain Text
SUB – BLOCK GENERATION FOR PLAIN TEXT AND KEY
First Round
Each round has 8 stages with Substitution and Transposition Operations.
1024 bit Key with equivalent sub-blocks of varying size
Key Transformation
Nth Round
Each round has 8 stages with Substitution and Transposition Operations.
1024 bit Key with equivalent sub-blocks of varying size
1024 bits of Cipher Text

Figure 2. General Block Diagram of PACTS

Each round has eight stages and the 1024 bits key is transformed for use in each stage as depicted in fig 3. In the first stage XOR operation is performed with the key sub-blocks and the plain text sub-blocks in a pattern decided by the initial and the final bits of the key. If both these bits are of same value then XOR operation is performed directly, otherwise the plain text bits are reversed before the operation. The key bits are then rotated to right or left by b/2 positions within the key sub-blocks so that it can be used in the next stage. Here ‘b’ refers to the number of bits in each key sub-block. The direction of rotation is determined by the parity of the key. If the parity is odd the bits are rotated to the right, otherwise they are rotated to the left. In the second stage key sub-block patterns are substituted with the plaintext sub-blocks.

Following this exchange manipulation is carried out between the key sub-blocks. This is achieved by swapping the odd numbered sub-blocks with the next higher order even numbered sub-blocks. In the third stage, rotation of bits within the sub-blocks is carried out based on the value pointed by the pattern of bits in the key sub-block. The Intra sub-block rotation is carried for each key sub-block based on the initial key value bits in each sub-blocks.

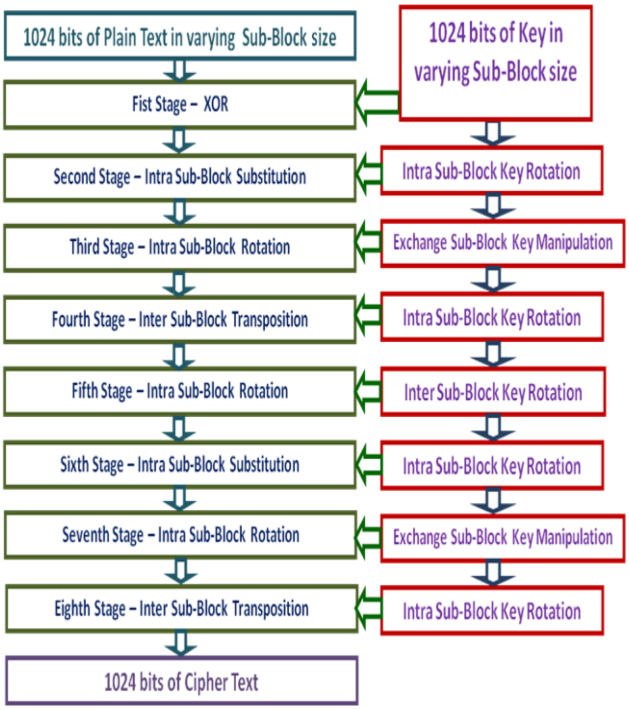
1024 bits of Plain Text in varying Sub-Block size
1024 bits of Key in varying Sub-Block size
Fist Stage – XOR
Second Stage – Intra Sub-Block Substitution
Intra Sub-Block Key Rotation
Third Stage – Intra Sub-Block Rotation
Exchange Sub-Block Key Manipulation
Fourth Stage – Inter Sub-Block Transposition
Intra Sub-Block Key Rotation
Fifth Stage – Intra Sub-Block Rotation
Inter Sub-Block Key Rotation
Sixth Stage – Intra Sub-Block Substitution
Intra Sub-Block Key Rotation
Seventh Stage – Intra Sub-Block Rotation
Exchange Sub-Block Key Manipulation
Eighth Stage – Inter Sub-Block Transposition
Intra Sub-Block Key Rotation
1024 bits of Cipher Text

Figure 3. Stages in each round of PACTS

In the fourth stage inter sub-block transpositions are carried out based on the bit pattern in the key. An inter sub-block rotation is performed for the key before the fifth stage and an intra sub-block rotation is carried out for data sub-blocks in the fifth stage. The key manipulation is performed with the key sub-blocks as it is done before the second stage. The sixth stage is the Intra sub-block substitution round. Then the key undergoes exchange sub-block key manipulation once again, as it is done just before the third stage. In the seventh stage sub-blocks are yet again rotated within themselves to the left or right based on the initial few bits of each key sub-block. The key bits then experience Intra sub-block rotation before they are utilized for Inter sub-block transposition in the eighth and the final stage of each round. If more rounds are used in the algorithm an Inter Sub-Block Key Rotation operation is performed after the eighth stage, so that it can be used for the next round. Brief algorithmic depiction of PACTS with single round is given as follows:

Input : 1024 bit plain text block

Output : 1024 bit cipher text block

Sub-Block Generation:

1. Run environment identification routine to identify the number of processors/cores ‘p’, their data handling capacities ‘c’ and clock speed ‘s’ to divide the 1024 bits key and the 1024 bits plain text into sub-blocks of ‘b’ bits.
2. if p==1&&c < 16 bits&&s ≤ 10 MHz then b=8 bits.
3. else if p == 1 && c ≥ 16 bits && c < 32 bits && s > 10 MHz && s ≤ 100 MHz then b = 16 bits.
4. else if p == 1 && c ≥ 32 bits && c < 64 bits && s > 100MHz && s ≤ 1000 MHz then b = 32 bits.
5. else if p≤4&&c ≥ 64 bits&&s >1GHz then b=64 bits.
6. else if p>4&&c≥64 bits&&s >3GHz then b=256 bits
7. else display “resources unsuitable for PACTS”.

Steps in Single Round of PACTS :

8. XOR key sub-blocks with data sub-blocks.
9. Intra sub-block rotation on key sub-blocks.
10. Intra sub-block substitution on data sub-blocks.
11. Exchange sub-block operation on key sub-blocks.
12. Intra sub-block rotation on data sub-blocks.
13. Intra sub-block rotation on key sub-blocks.
14. Inter sub-block transposition on data sub-blocks.
15. Inter sub-block rotation on key sub-blocks.
16. Intra sub-block rotation on data sub-blocks.
17. Intra sub-block rotation on key sub-blocks.
18. Intra sub-block substitution on data sub-blocks.
19. Exchange sub-block operation on key sub-blocks.
20. Intra sub-block rotation on data sub-blocks.
21. Intra sub-block rotation on key sub-blocks.
22. Inter sub-block transposition on data sub-blocks.

For ECB mode implementation of PACTS in Personal Computers, single round execution is sufficient as it provides the required security level. Use of inter sub-block and intra sub-block transpositions, substitutions and shift operations makes PACTS more communication intensive rather than computation intensive. PACTS is implemented in shared memory architecture using MPI, OpenMP and Java Threads programming with different sub-block sizes and compared with sequential results. The speedup of various combinations of executions are analyzed and compared and the results are shown in Table 1.

TABLE I ECB MODE IMPLEMENTATION OF PACTS (SINGLE ROUND)

SUB-BLOCK SIZE	SPEEDUP IN ECB MODE					
	MPI		OpenMP		JAVA Threads	
	ENC	DEC	ENC	DEC	ENC	DEC
8 bits	1.73	1.82	3.22	3.30	3.09	3.15
16 bits	2.93	3.01	3.57	3.64	3.36	3.42
32 bits	3.46	3.53	3.66	3.75	3.59	3.65
64 bits	3.60	3.68	3.84	3.93	3.76	3.81
128 bits	3.65	3.72	3.87	3.94	3.80	3.85
256 bits	3.73	3.79	3.92	3.97	3.86	3.89

ECB Mode: Electronic Code Book Mode
ENC : Encryption DEC : Decryption

The performance of cryptographic algorithms in parallel computing environment is shown using speedup. Speedup is the ratio of the time taken by the serial implementation of the algorithm to that of its parallel implementation and is denoted by Sp = Ts/Tp. Where ‘Tp’ denotes the parallel execution time and ‘Ts’ denotes the sequential execution time.

All the parallel implementations provided similar variations in their output. When the sub-block size is kept small the speedup is low, but it gradually increased linearly when the sub-block size is increased. The decryption process provided better speedup than the encryption process because most of the values and the decisions computed for the encryption stages are made available to the decryption stages. A comparative representation of the performance of encryption algorithm using MPI, OpenMP and Java threads are shown in Fig. 4 and that of the decryption algorithm is shown in Fig 5.

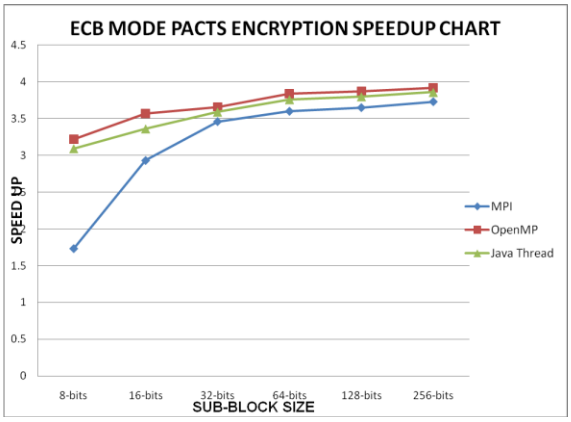

Figure. 4. Performance of ECB mode Encryption of PACTS

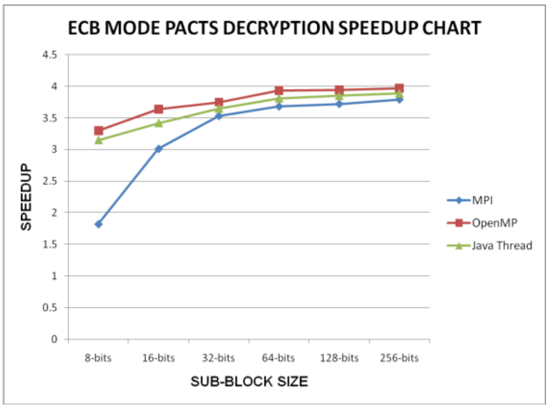

Figure. 5. Performance of ECB mode Decryption of PACTS

IV. CBC IMPLEMENTATION OF PACTS

In ECB mode of PACTS, a plain text block always produces the same cipher text block, when the same key is used. Cipher Block Chaining (CBC) mode is used to overcome this problem. CBC mode ensures that even if the same plain text block is repeated again and again it yields totally different cipher text blocks in the output. In CBC mode result of the encryption of the previous block are fed back into the encryption of the current block. As there is no feedback available for the first block of the plaintext a random block of text known as Initialization Vector (IV) is used in the first step of encryption. The encryption process in CBC mode is shown in Fig. 6 and the decryption process in Fig. 7.

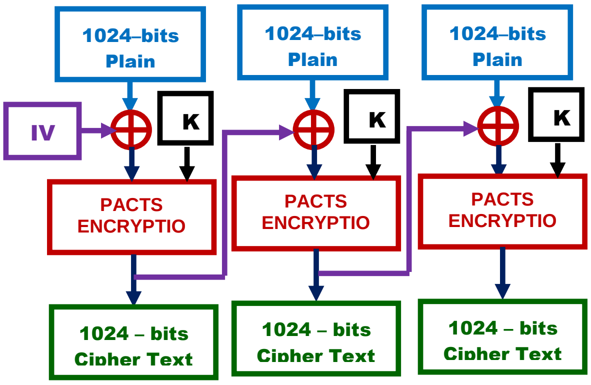

Figure. 6: CBC mode Encryption of PACTS

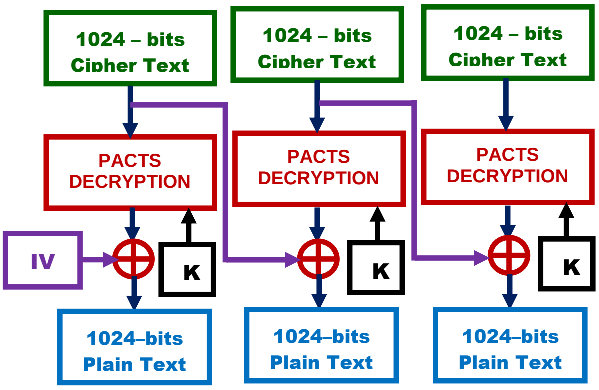

Figure. 7: CBC mode Decryption of PACTS

The decryption is just the reverse of the encryption, except that the feedback of the previous level is readily available for decryption, whereas in encryption it is not. The speedup results of CBC mode implementation of PACTS in MPI, OpenMP and Java Threads are given in Table II.

TABLE II CBC MODE IMPLEMENTATION OF PACTS (SINGLE ROUND)

SUB-BLOCK SIZE	SPEEDUP IN CBC MODE OF PACTS					
	MPI		OpenMP		JAVA Threads	
	ENC	DEC	ENC	DEC	ENC	DEC
8 bits	**1.18**	**1.76**	**1.67**	**3.24**	**1.53**	**3.08**
16 bits	**1.46**	**2.95**	**1.78**	**3.59**	**1.68**	**3.37**
32 bits	**1.66**	**3.46**	**1.83**	**3.71**	**1.76**	**3.61**
64 bits	**1.71**	**3.62**	**1.88**	**3.89**	**1.82**	**3.76**
128 bits	**1.78**	**3.68**	**1.92**	**3.90**	**1.84**	**3.81**
256 bits	**1.85**	**3.74**	**1.97**	**3.93**	**1.88**	**3.84**

CBC Mode: Cyber Block Chaining Mode
ENC : Encryption DEC : Decryption

The performance of CBC mode encryption of PACTS is reduced considerably because of the dependencies caused by feedback of the ciphertext to the next level. The decryption is not affected as the feedback to the next level is readily available. The performance graphs of encryption and decryption of PACTS in CBC mode is given in Fig. 8 and Fig. 9.

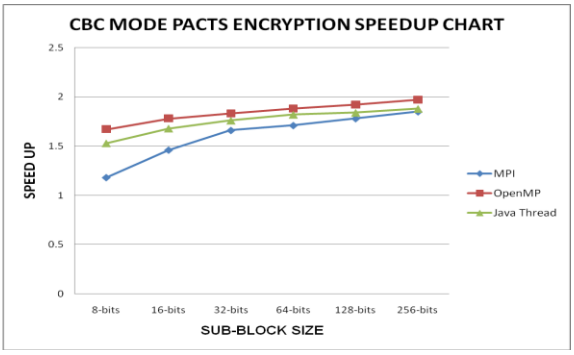

Figure. 8: Performance of CBC mode Encryption of PACTS

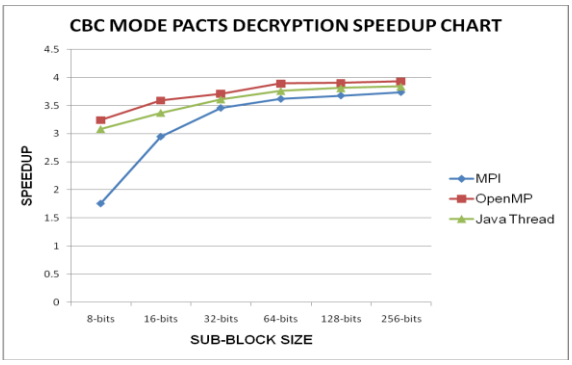

Figure. 9: Performance of CBC mode Decryption of PACTS

V. INTERLEAVED CBC IMPLEMENTATIONS OF PACTS

The encryption in CBC mode depends on the encryption of the previous sub-blocks. This makes it difficult to parallelize encryption. The solution to this problem is to interleave multiple encryption blocks. Interleaving can be done in n-ways, wherein the 2-way and 4-way interleaving are adopted in this work.

A. Two-Way Interleaved CBC mode of PACTS

Two-Way interleaving is the next immediate improvement to the CBC implementation. In two-way interleaving the output of the first encryption sub-block is feedback to the third stage and the second to fourth stage and so on. The two Initialization Vectors are required to start the encryption and the decryption processes. The structure of two-way interleaving for encryption is shown in Fig. 10 and that of decryption in Fig. 11.

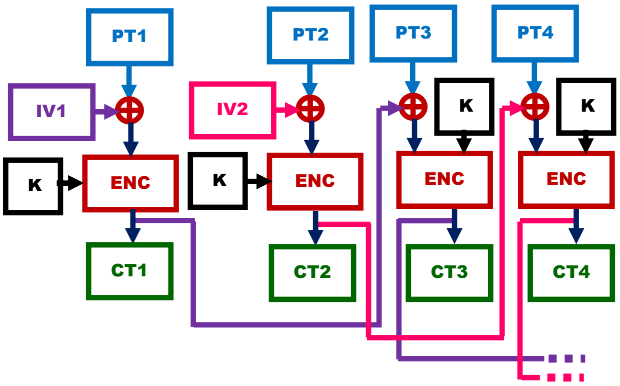

Figure. 10: Two-Way Interleaved CBC mode Encryption of PACTS

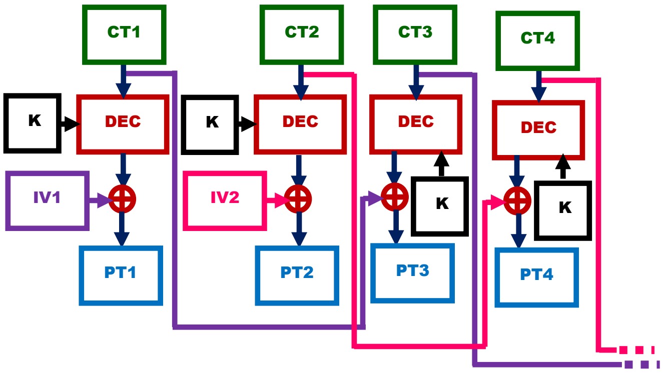

Figure. 11: Two-Way Interleaved CBC mode Decryption of PACTS

The decryption process in two-way interleaved CBC mode of PACTS is the reverse of the encryption process. The ciphertext feedback of the previous level is also readily available in each block of decryption, whereas in encryption it is not so. The speedup results of two-way interleaved CBC mode implementation of PACTS in MPI, OpenMP and Java Threads are given in Table III.

TABLE III 2-WAY INTERLEAVED CBC MODE IMPLEMENTATION OF PACTS (SINGLE ROUND)

SUB-BLOCK SIZE	SPEEDUP IN 2-WAY ICBC MODE OF PACTS					
	MPI		OpenMP		JAVA Threads	
	ENC	DEC	ENC	DEC	ENC	DEC
8 bits	1.29	1.78	2.05	3.26	1.93	3.10
16 bits	1.76	2.97	2.17	3.60	2.08	3.39
32 bits	2.05	3.48	2.25	3.72	2.16	3.62
64 bits	2.11	3.64	2.31	3.89	2.23	3.78
128 bits	2.26	3.69	2.40	3.91	2.34	3.82
256 bits	2.38	3.76	2.47	3.94	2.41	3.86

ICBC Mode: Interleaved Cyber Block Chaining Mode
ENC : Encryption DEC : Decryption

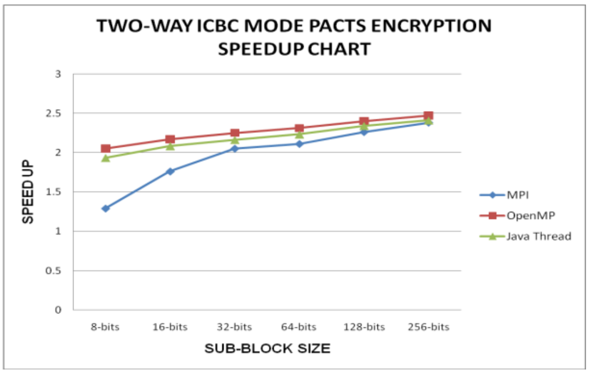

Figure. 12: Performance of 2-Way ICBC mode Encryption of PACTS

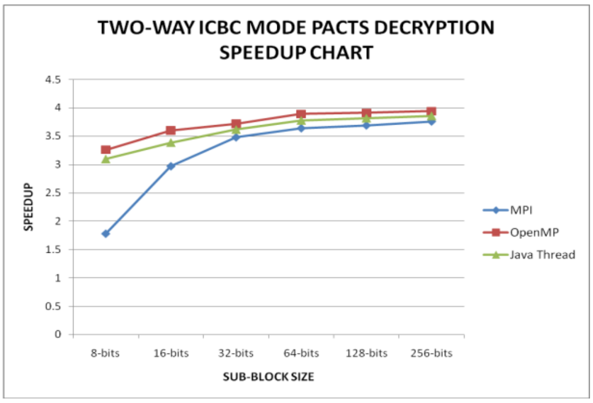

Figure. 13: Performance of 2-Way ICBC mode Decryption of PACTS

The performance of two-way interleaved CBC implementation is found to be better than the CBC implementation and it is illustrated in Fig. 12 and Fig. 13. The additional processes or threads that handle the two-ways of encryption and decryption separately are responsible for this enhancement.

B. Four-Way Interleaved CBC mode of PACTS

In four-way interleaving the output of the first encryption sub-block is feedback to the fifth and that of second to sixth, third to seventh, fourth to eighth stages and so on. In four-way interleaving, four Initialization Vectors are required to start the encryption and the decryption processes. In order to enhance the efficiency of execution of 4-way ICBC mode implementation of PACTS in parallel computing environments, the number of processes or threads used for implementing the encryption or the decryption algorithms is increased by four times. Implementation of 4-way interleaved CBC technique considerably increases the complexity because of the additional operations that has to be performed, for every level of interleaving. The structure of four-way interleaving of PACTS for encryption is shown in Fig. 14 and that of decryption in Fig. 15.

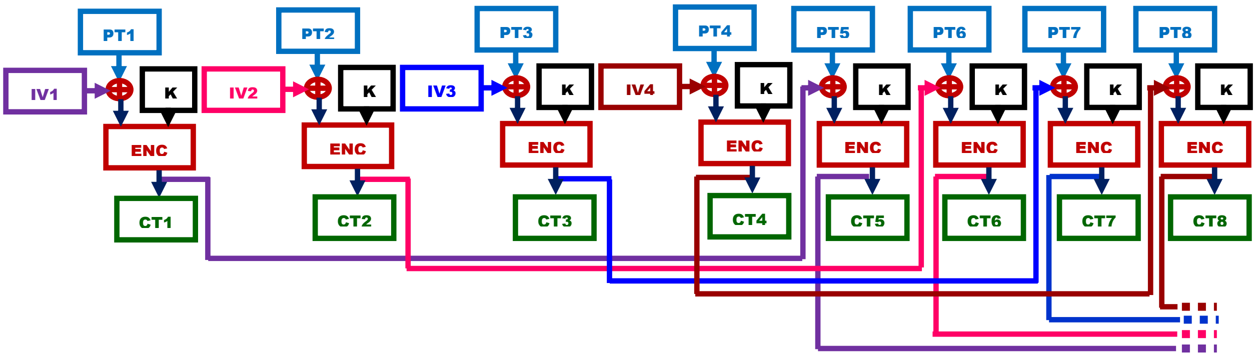

Figure. 14: Four-Way Interleaved CBC mode Encryption of PACTS

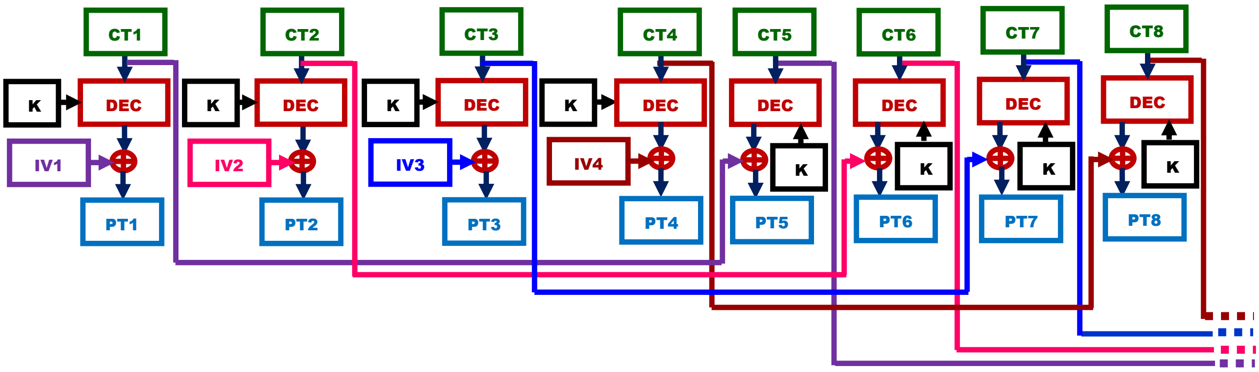

Figure. 15: Four-Way Interleaved CBC mode Decryption of PACTS

TABLE IV 4-WAY INTERLEAVED CBC MODE IMPLEMENTATION OF PACTS (SINGLE ROUND)

SUB-BLOCK SIZE	SPEEDUP IN 4-WAY ICBC MODE OF PACTS					
	MPI		OpenMP		JAVA Threads	
	ENC	DEC	ENC	DEC	ENC	DEC
8 bits	**1.47**	**1.80**	**2.35**	**3.28**	**2.24**	**3.13**
16 bits	**2.16**	**2.99**	**2.64**	**3.62**	**2.53**	**3.40**
32 bits	**2.62**	**3.50**	**2.78**	**3.73**	**2.69**	**3.66**
64 bits	**2.78**	**3.65**	**2.89**	**3.91**	**2.82**	**3.79**
128 bits	**2.81**	**3.70**	**3.02**	**3.93**	**2.90**	**3.83**
256 bits	**3.02**	**3.77**	**3.16**	**3.95**	**3.11**	**3.88**

ICBC Mode: Interleaved Cyber Block Chaining Mode
ENC : Encryption DEC : Decryption

A considerable improvement is seen in the performance of four-way interleaved CBC mode implementation of PACTS when compared with the simple CBC and two-way Interleaved CBC modes. This is shown in Table IV and in Fig. 16 and Fig. 17.

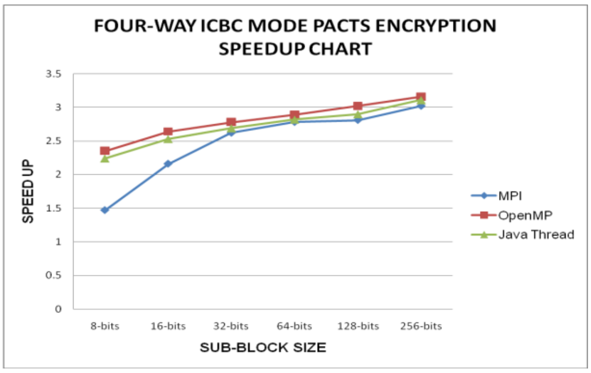

Figure. 16: Performance of 4-Way ICBC mode Encryption of PACTS

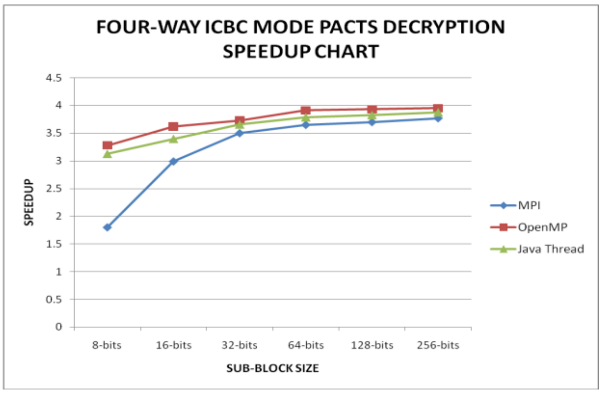

Figure. 17: Performance of 4-Way ICBC mode Decryption of PACTS

Increasing the level of the interleaving in CBC mode of PACTS enhances the parallel performance, but it also increases the number of Initialization Vectors required and the complexity of implementations. Even though the encryptions are made to perform better, it cannot reach the level of ECB mode implementations because of the dependency issues involved with the feedback of the ciphertext from the previous stage. The decryption processes does not suffer such drawbacks and they perform well in parallel executions, as the ciphertext of the previous stage is available well in advance before the beginning of the processes in the current level.

VI. CONCLUSION

PACTS is an adaptive cryptographic algorithm that provides better security strength and performance in parallel computing environments. It requires 5.7 X 10288 years to break this cipher with brute force attack. PACTS is a dynamic algorithm as its granularity and execution are decided during runtime using the bit patterns in the key. As the general reversible techniques are used, this algorithm is scalable. The algorithm is exclusively designed for software implementations and to avoid dependency problems in the parallel processing environments. PACTS is a communication intensive block cipher with Inter-block operations incurring more communication cost than Intra-block operations.

When executed in parallel computing environments the performance of PACTS in ECB mode is found to be better. But it always produces the same ciphertext for a particular plaintext when the same key is used. Although CBC mode is employed to alleviate this problem, its decryptions support parallelization, whereas its encryptions do not. The issue faced in parallelization of CBC mode encryptions is solved to some extent with two-way and four-way Interleaved CBC implementations.

REFERENCES

[1] Jeffrey Hoffstein, Jill Pipher, Joseph H. Silverman, “An Introduction to Mathematical Cryptography”, Springer International Edition, Springer (India) Pvt. Ltd., New Delhi, 2008.

[2] Eric C. Seidel, Joseph N. Gregg, “Preparing Tomorrow’s Cryptography: Parallel Computation via Multiple Processors, Vector Processing, and Multi-Cored Chips”, Research Paper, May 2003.

[3] William Stallings, “Cryptography and Network Security-Principles and Practice”, 5th Edition, Dorling Kindersley (India) Pvt. Ltd., licensees of Pearson Education, 2011.

[4] Menezes A.J., Van Oorschot P.C., Vastone S.A., “Handbook of Applied Cryptography”, CRC Press, 1996.

[5] Schneier B., “Applied Cryptography: Protocols, Algorithms, and Source Code in C”, Second Edition, Wiley & Sons, 1995.

[6] Suman Khakurel, Prabhat Kumar Tiwary, Niwas Maskey, Gitanjali Sachdeva, “Security Vulnerabilities in IEEE 802.11 and Adaptive Encryption Technique for Better Performance”, IEEE Symposium on Industrial Electronics and Applications, Penang, Malaysia, 2010.

[7] Thomas Rauber, Gudula Runger, “Parallel Programming –for Multicore and Cluster Systems”, International Edition, Springer (India) Pvt. Ltd. New Delhi, 2010.

[8] HoWon Kim, YongJe Choi, Kyoil Chung, and HeuiSu Ryu, "Design and Implementation of a Private and Public Key Crypto Processor and Its Application to Security System," proceedings of the 3rd International Workshop on Information Security Applications, pp. 515 – 531, Jeju, Korea, 2002,

[9] Pionteck, T., Staake T., Stiefmeier T., Kabulepa L. D., Glesner M., “Design of reconfigurable AES encryption/decryption engine for mobile terminals”, in the proceedings of the International Symposium on Circuits and Systems, 2004.

[10] Sourav Mukherjee, Bidhudatta Sahoo, “A survey on hardware implementation of IDEA Cryptosystems” Information Security Journal: A Global Perspective, Vol. 20, Nr. 4-5, pp 210-218, 2011.

[11] Tetsuya Ichikawa, Tomomi Kasuya, and Mitsuru. Matsui. “Hardware evaluation of the AES finalists.” In Proc. Third Advanced Encryption Standard Candidate Conference, pages 279–285, USA, 2000.

[12] Bryan Weeks, Mark Bean, Tom Rozylowicz, and Chris Ficke. “Hardware performance simulations of Round 2 Advanced Encryption Standard algorithms”. In Proc. Third Advanced Encryption Standard Candidate Conference, USA, 2000.

[13] Swankoski E. J., Brooks R. R., Narayanan V., Kandemir M., and Irwin M. J., “A Parallel Architecture for Secure FPGA Symmetric Encryption”, proceedings of the 18th International Parallel and Distributed Processing Symposium, Santa Fe, New Mexico, 2004.

[14] Kotturi D., Seong-Moo Y., Blizzard J., “AES crypto chip utilizing high-speed parallel pipelined architecture” proceedings of the IEEE International

Symposium on Circuits & Systems, 2005.

[15] Chi-Wu H., Chi-Jeng C., Mao-Yuan L., Hung-Yun T., "The FPGA Implementation of 128-bits AES Algorithm Based on Four 32-bits Parallel Operation", First International Symposium on Data, Privacy, and E-Commerce, 2007.

[16] Chonglei, M., J. Hai and J. Jennes, "CUDA-based AES Parallelization with fine-tuned GPU memory utilization", proceedings of the IEEE International Symposium on Parallel and Distributed Processing, Workshops and Ph. D. Forum, pp19-23, 2010.

[17] Julian Ortega, Helmuth Tefeffiz, Christian Treffiz, "Parallelizing AES on Multicores and GPUs", IEEE International Conference on Electro/Information Technology, Mankato, US, pp. 1-5, 2011.

[18] Li, H. and J. Z. Li, "A new compact dual-core architecture for AES encryption and decryption", Canadian Journal of Electrical and Computer Engineering, pp 209-213, 2008.

[19] Hua Li., "A parallel S-box architecture for AES byte substitution", paper presented at IEEE sponsored International Conference on Communication, Circuits and Systems, Chengdu, China, 2004.

[20] Praveen Dongara, T. N. Vijaykumar, Accelerating Private-key cryptography via Multithreading on Symmetric Multiprocessors. In Conference Proceedings of the IEEE International Symposium on Performance Analysis of Systems and Software, pp 58-69, 2003.

[21] Zadia Codabux-Rossan, M. Razvi Doomum, "AES CCMP Algorithm with N-Way Interleaved Cipher Block Chaining", University of Mauritius Research Journal, Volume – 15, pp 527-544, 2009.

[22] S. Ashokkumar, K. Karuppasamy, Balaji Srinivasan, V.Balasubramanian "Parallel Key Encryption for CBC and Interleaved CBC" International Journal of Computer Applications, Volume 2–No. 1, 2010.

[23] Bielecki W., Burak D., "Parallelization of Standard Modes of Operation for Symmetric Key Block Ciphers", Image Analysis, Computer Graphics, Security Systems and Artificial Intelligence Applications Vol 1, Bialystok 2005.

[24] Bielecki W., Burak D., "Parallelization of Symmetric Block Ciphers", Computing, Multimedia and Intelligent Techniques special issue on Live Biometrics and Security, Vol. 1, Czestochowa University of Technology, June 2005.

[25] J. John Raybin Jose, Dr. E. George Dharma Prakash Raj, "PACTS – A Communication Intensive Symmetric Block Cipher for Parallel Computing Environments" in the proceedings of IEEE International Conference on Research and Development Prospects on Engineering and Technology, Nagapattinam, India, April 2013.

Authors Profile:

J. John Raybin Jose has completed his Masters Degree in Applied Physics, Computer Applications and Master of Philosophy in Physics, Computer Science. He is working as Assistant Professor and Head, Department of Information Technology, Bishop Heber College (Autonomous), Tiruchirappalli, Tamil Nadu, India. He has 18 years of teaching and 12 years of research experience. Currently he is pursuing his doctoral degree at Bharathidasan University, Tiruchirappalli, India, under the guidance or Dr. E. George Dharma Prakash Raj. His areas of research include Information Security, Mobile Computing and High Performance Computing.

Dr. E. George Dharma Prakash Raj completed his Masters Degree in Computer Science and Masters of Philosophy in Computer Science in the years 1990 and 1998. He completed his Doctoral degree in Computer Science in the year 2008. He has around 23 years of Academic experience and 15 years of Research experience in the field of Computer Science. Currently he is working as Assistant Professor in the School of Computer Science, Engineering and Technology at Bharathidasan University, Tiruchirappalli, India. He has published several papers in International Journals and Conferences related to Computer Science. He is Editorial Board Member, Reviewer and Programme Committee Member in many International Journals and Conferences. He has convened many National and International Conferences related to Computer Science.

How to cite this paper: J. John Raybin Jose, E. George Dharma Prakash Raj,"CBC and Interleaved CBC Implementations of PACTS Cryptographic Algorithm", IJCNIS, vol.6, no.4, pp.63-71, 2014. DOI: 10.5815/ijcnis.2014.04.08