PERFORMANCE EVALUATION OF BLOWFISH ALGORITHM ON SUPERCOMPUTER IMAN1

Mahmoud Rajallah Asassfeh[1], Mohammad Qatawneh[1] and Feras Mohamed AL-Azzeh[2]

[1]Department of Computer Science-King Abdullah II School for information technology, University of Jordan, Amman-Jordan,
[2]Department of computer information systems, Alzaytoonah University of Jordan, Amman-Jordan.

ABSTRACT

Cryptographic applications are becoming increasingly more important in today's world of data exchange, big volumes of data need to be transferred safely from one location to another at high speed. In this paper, the parallel implementation of blowfish cryptography algorithm is evaluated and compared in terms of running time, speed up and parallel efficiency. The parallel implementation of blowfish is implemented using message passing interface (MPI) library, and the results have been conducted using IMAN1 Supercomputer. The experimental results show that the runtime of blowfish algorithm is decreased as the number of processors is increased. Moreover, when the number of processors is 2, 4, and 8, parallel efficiency achieves up to 99%, 98%, and 66%, respectively.

KEYWORDS

Blowfish; Encryption; MPI; Supercomputer

1. INTRODUCTION

As we are moving to the information society, where information can travel fast, and through various modes of communication in what is called as the global village, it has become more apparent that the same information can end up in the wrong hands either by mistake, or with the intention to harm. Hence for secure communication required cryptography algorithms. Several cryptography algorithms have proposed like AES, DES, 3DES, RC2 [18][20]. Among these algorithms is blowfish cryptography algorithm.

The encryption algorithms are usually divided into two types: Symmetric key encryption (private) and Asymmetric key encryption (public), in Symmetric key encryption or secret key encryption, only one key is used to encrypt and decrypt data, the key should be distributed before start sending between entities [16][19] The Symmetric key cryptography algorithms include Blowfish, AES, RC2, DES, 3DES, and RC5[17][18][15]

In Asymmetric key encryption or public key encryption, private key and public key are used, the Public key is used for encryption and a private key is used for decryption [2].

The aim of this paper is to implement and evaluate the performance of parallel blowfish algorithm in terms of execution time, speedup, and parallel efficiency using Message Parallel Interface (MPI) on supercomputer IMAN1. The Iman1 is the first Jordanian supercomputer with high performance computing resources. It is used not only from inside Jordan also in the region for academic purposes. It is using 2260 PlayStation3 devices. IMAN1 is Jordan's first and fastest High Performance Computing resource, funded by JAEC and SESAME. It is available for use by academia and industry in Jordan and the region [8].

Parallel and distributed computing systems are high-performance computing systems that spread out a single application over many multi-core and multi-processor computers in order to rapidly complete the task. Parallel and distributed computing systems divide large problems into smaller sub-problems and assign each of them to different processors in a typical distributed system running concurrently in parallel [9][10] [11] [12] [13][14][21].

The rest of this paper is organized as follows; Section 2 presents the related works. Section 3 reviews the blowfish algorithm, Section 4 presents the experiments and results, and Section 5 presents the conclusion.

2. RELATED WORK

One of the very important functional features of cryptographic algorithms is cypher speed, this feature is significant in case of block cyphers since they usually work on large data sets, there are many researchers and studies to increase the speed of encryption algorithms using parallel implementation.

In [1] the author describes the parallelization process of the encryption algorithm and he use eight Quad- Core(32) Intel Xeon Processors 7310 Series - 1.60 GHz, from experimental result he showed that the application of the parallel encryption algorithm for multiprocessor would considerably improve the time of the data encryption and decryption, he believed that the speed-ups received are satisfactory.

In [6] the authors demonstrated the way of implementing blowfish cryptography algorithm on graphical processing unit (GPU) which can be used for parallel computing to improve the performance of the algorithm, the experiment shows improvement in performance of GPU in encryption and decryption of large files, they observed also that even if input file size increases, average encryption and decryption time can be reduced using GPU.

In [7] the authors present high throughput blowfish architecture , it integrate pipeline technique to break critical path delay and increase speed , the result show that the architecture of blowfish algorithm provide better performance due to parallel execution of the algorithm on FPGA.

In our research we implemented the blowfish algorithm on parallel platform (supercomputer) which is different from the above researches in it is architecture and the number of processors used.

3. BLOWFISH ALGORITHM

in 1993 Bruce Schneier, one of the world's leading cryptologists, designed the Blowfish algorithm and made it available in the public domain, blowfish is a variable length key, blowfish is also a block cipher with a length of 64 bit , and has not been cracked yet, it can be used in hardware applications due to its compactness [3][5][7]. There are two parts for this algorithm; a part that addresses the expansion of the key and apart that addresses the encryption of the data.

3.1. Key Expansion

The key Expansion of blowfish algorithm begins with the P-array and S-boxes with the utilization of many sub-keys, which requires pre-computation before data encryption or decryption. The P-array consists of eighteen 4 byte sub-keys: P1, P2…P17, P18.

Blowfish with keys up to 448 bits length is transformed into several sub-key arrays.
There are 256 entries for each of the four 32-bit S-boxes:

S1, 0, S1,1,....., S1,255
S2, 0, S2,1,....., S2,255
S3, 0, S3,1,....., S3,255
S4, 0, S4,1,....., S4,255

Below the steps of how to generate the subkeys:

- Initialize the P-array and four S-boxes with a fixed string, this string consist of hexadecimal digits of π.
- The first element in P-array (P1) XORed with the first 32 bits of the key, and the second element in P-array (P2)XORed with the second 32 bits of the key ,repeated this until all the elements in P-array are XORed with the key bits.
- Encrypt all zero string by blowfish algorithm using sub keys described in step (1, 2).
- Change P1 and P2 with the output of step (3).
- Using the modified sub keys encrypt the output of step (3).
- Change P3 and P4 with the output of step (5).

This process is continued, until the entire of the P-array and 4 s-boxes are changed [5].

3.2. Encryption/Decryption Process

Figure 1 illustrates the Blowfish algorithm

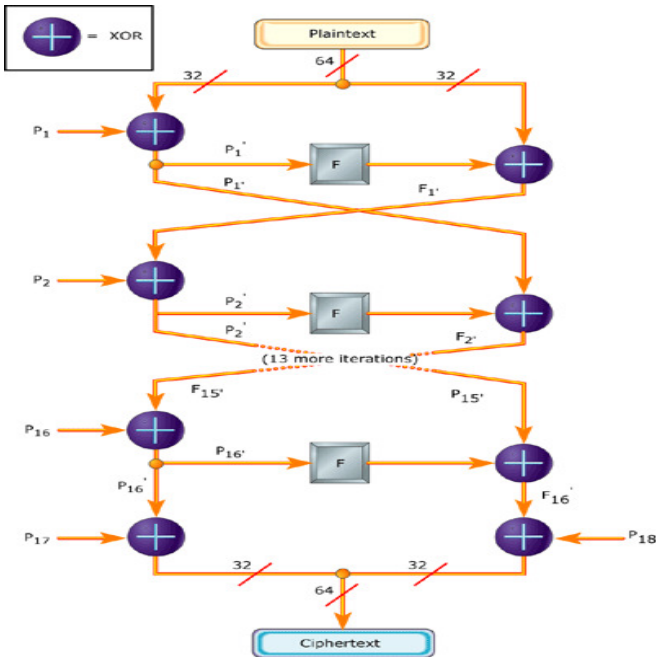

Figure 1. Blowfish algorithm [4]

Applying the blowfish cryptography algorithm, the encryption process of the message“HI world” is passed through the following stages as shown in Fig 1. As follows:

- The blowfish is a block cypher algorithm of size 64 bit for the block.
- The message “HI world” consists of 7 characters + 1 space which equal to 8 byte (64 bit).

- First divide the message into 32 bits, the left 32 bits which represent “Hi w” are XORed with the first element of P-array (P1) (which generated from key expansion) and create a value called P1'.

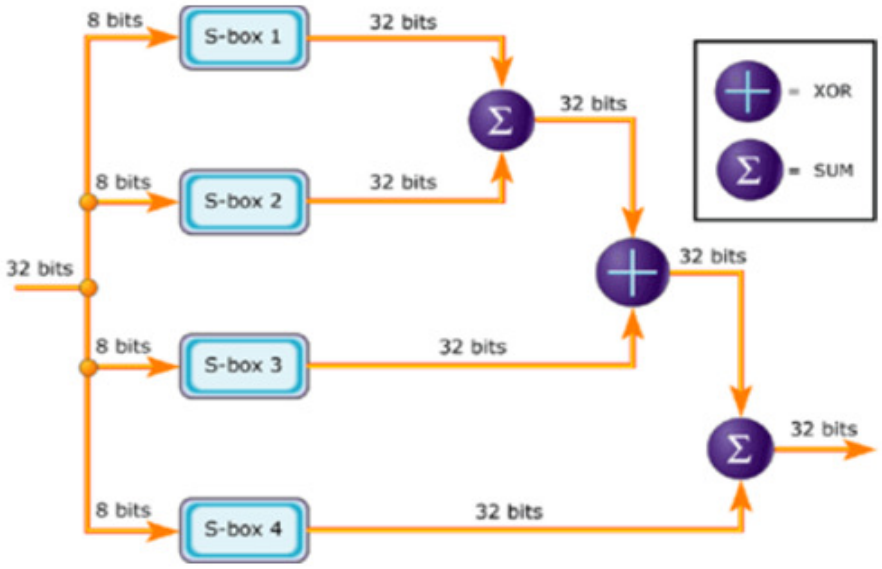

Figure 2. Graphical of function (F) [4].

- Run the result in point 3 (P1') through a transformation function called F In which the 32 bit is divided into 4 bytes each one uses as indices to a value in the S-boxes a ,b ,c ,d as shown in Fig 2.
- The first two values from S-box 1, 2 are added to each other then Xored with the third value from S-box3 .
- The result is added to the value from S-box4 to produce 32 bit.
- The output of the transformation function F is XORed with the “right” 32 bits of the message “orld” to produce F1'.
- F1' replace the left half of the message and P1' replace the right half.
- The process is repeated 15 more times with successive members of the P-array.

The resulting P16' and F16' after 16 round are then XO Red with the last two entries in the p-array (entries P17 and P18) and recombined to produce the 64 bit cipher text of the message “HI world “, a graphical representation of the function (F) appears in Fig 2.

4. EXPERIMENTS AND RESULTS

In order to evaluate the performance of the blowfish algorithm in parallel platform experiment method was used ,the experiment conducted on IMAN1 super computer, and on sequential platform as a reference to compare the results from parallel platform with it , Fig .3 illustrates the design stages of the research which consist of the following**:**

- Programming the blowfish encryption algorithm using C language.
- Convert the sequential C program into parallel using MPI library standard.

- Implement the algorithm on supercomputer IMAN1 and conduct the experiment by changing the number of processors and apply different plaintext size then take the maximum processor time.
- We repeat the experiment many times for every number of processors then we took the average and record it in table .2
- We calculate the speed up and efficiency from runtime recorded in table .2
- The results are evaluated in term of encryption time, speed up and parallel efficiency.

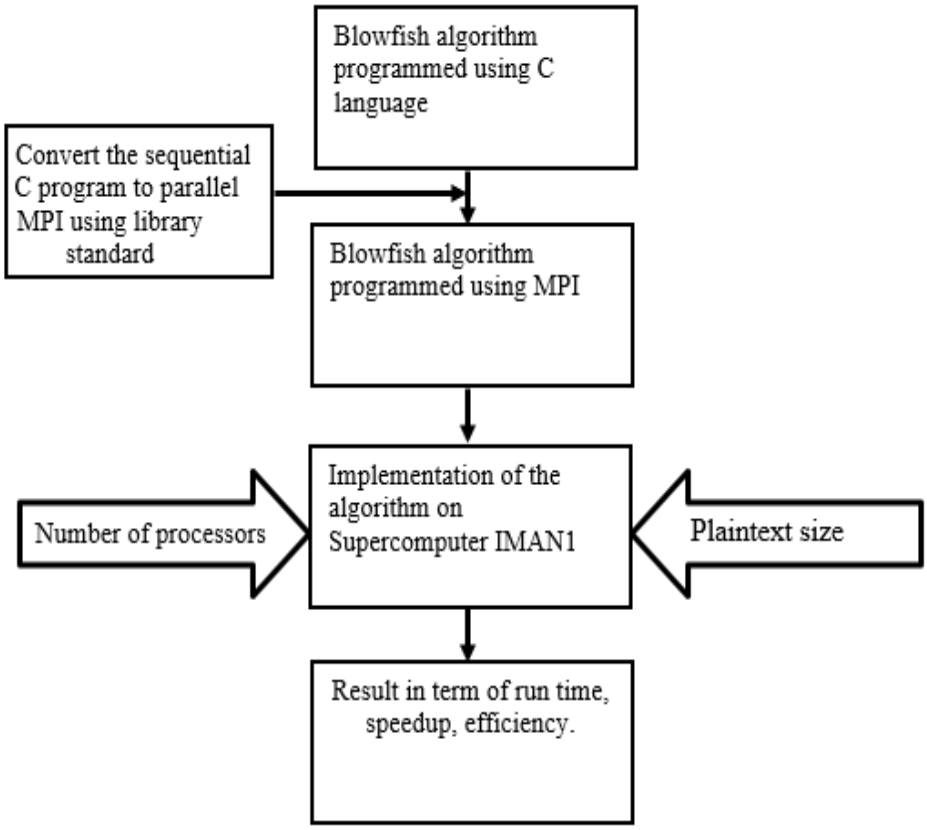

Figure 3. The design stages of the research.

The hardware and software specification with implementation parameters are listed in table 1 .

Table 1. The Hardware and Software specifications

Hardware specification	Intel core i5-4200u CPU @1.6GHZ 2.4GHZ, RAM 4 GB
Software specification	Scientific Linux 6.4 with open MPI, C, C++ complier
Data size (Kbyte)	4000, 8000, 16000, 40000, 80000, 160000
Number of processors	1,2,3,4,8,16,32,64,128

Table 2. Encryption time for blowfish algorithm on IMAN1Super Computer

	Time of encryption(s)								
Plaintextsize(Mbyte)	p=1	p=2	p=3	p=4	p=8	p=16	p=32	p=64	p=128
4	0.222	0.112	0.0779	0.062	0.0408	0.0327	0.0533	0.05	0.013
8	0.444	0.224	0.151	0.118	0.084	0.0793	0.136	0.111	0.084
16	0.888	0.454	0.305	0.241	0.163	0.1884	0.1558	0.1522	0.119
40	2.24	1.122	0.76	0.586	0.402	0.4	0.507	0.353	0.331
80	4.44	2.23	1.502	1.194	0.809	0.73	0.631	0.73	0.643
160	8.886	4.445	2.999	2.34	1.64	1.317	1.15	1.26	1.25

4.1. Encryption Time Evaluation

Figure4. Shows the run time for the blowfish algorithm, using 1 processor (sequential), when the plaintext size increased the time of encryption is increased.

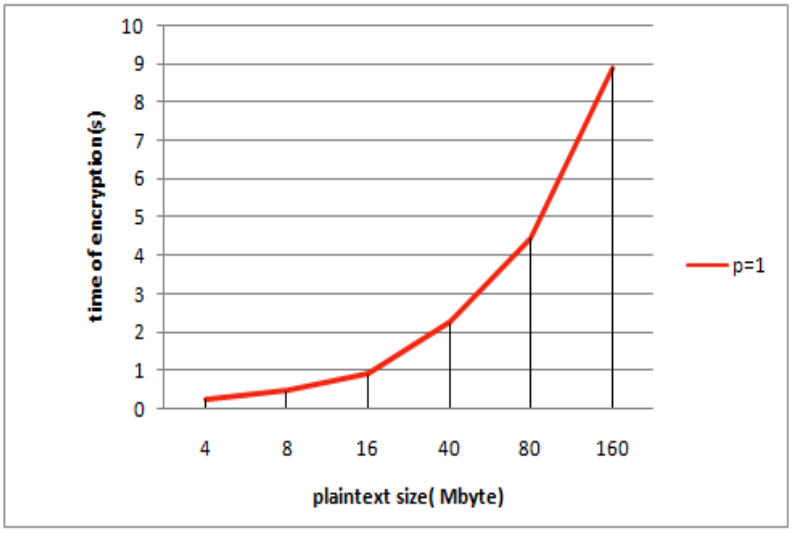

Figure 4. The time of encryption for sequential platform for different plaintext size.

Figure5. illustrates the encryption time according to different number of processors, we chose 6 different data size from 4 Mbyte up to 160 Mbyte which cover small and large data size, we note from the figure the following:

- When the number of processors increases the encryption time decrease, due to the data distribution among the processors, and this is clear when we are moving from 2 -4-8-16-32 processors.
- When we use 64 and 128 processors the time of encryption decrease slightly and sometimes remain the same, or more than 32 processors this is due to the increase in communication overhead, and it became more than the computation overhead.

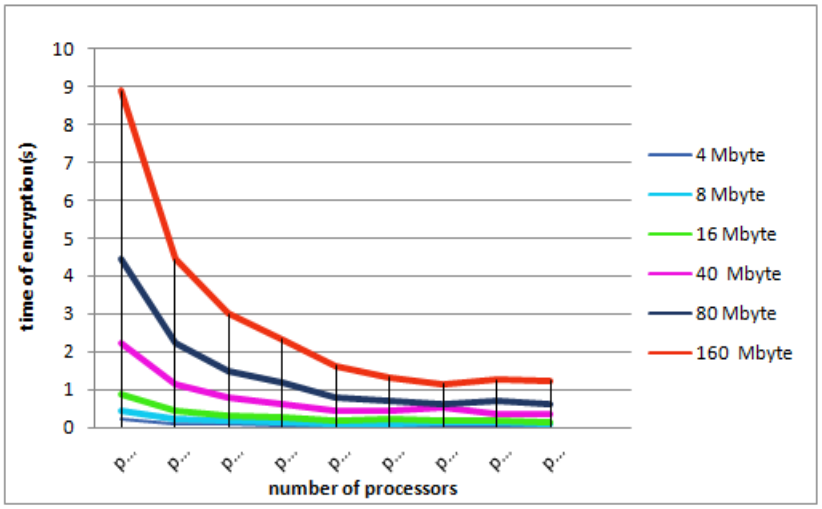

Figure 5. The time of encryption for different number of processors for different plaintext size

Figure 6, and 7 show the encryption time for different plaintext size 4,8,16 Mbyte and 40, 80,160 Mbyte for different number of processors.

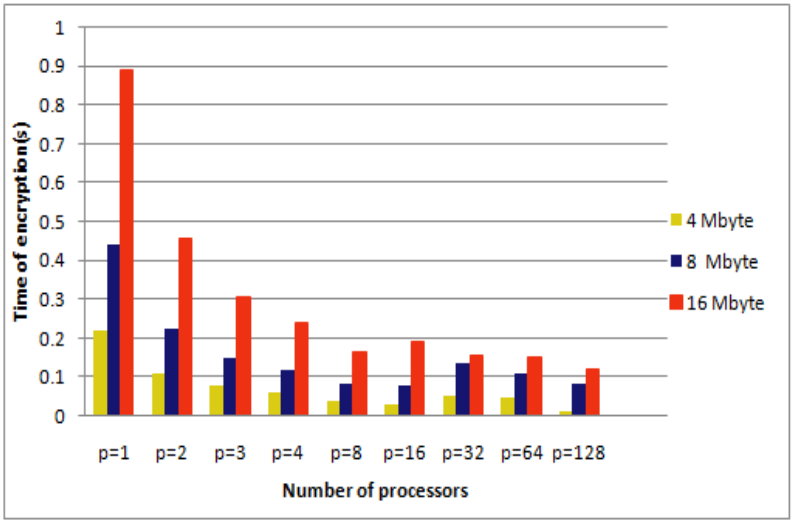

Figure 6. The time of encryption for different number of processors for 4, 8, 16 Mbyte.

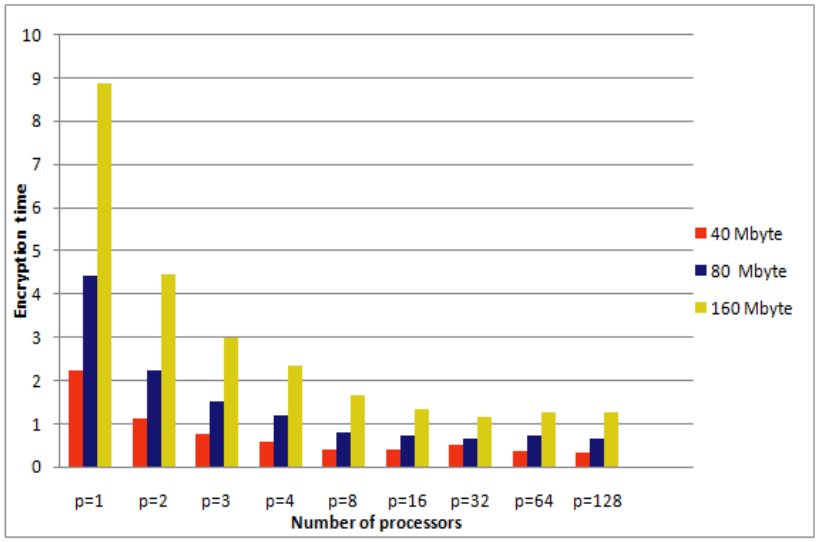

Figure 7. The time of encryption for different number of processors for 40, 80, 160 Mbyte

4.2. SPEEDUP EVALUATION

The speedup is the ratio between the sequential time and the parallel time Figure8. Illustrate the speed up to 3 different plaintext size 8 M byte, 40 M byte, 160M byte,
From Fig 8, we note the following:

- The speed up increase when the number of processors increase, and the increments is the same for all plaintext size for processors 2-8.
- When the number of processors =16 the speed up for 8 Mbyte and 40 M byte is the same while the speed up for 160 Mbyte is the best.
- When the number of processors = 32, 64,128 the speed up is the best for 160 Mbyte plaintext and after that 40 Mbyte then 8 Mbyte, respectively.
  So the speed up on the large size of data is better when use number of processor 32, 64,128.

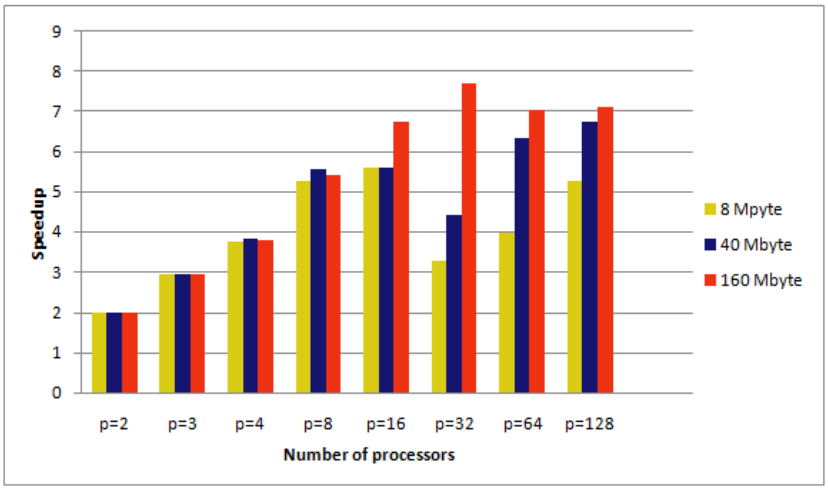

Figure 8. The speedup of the blowfish algorithm on different number of processors and different plaintext size

4.3. PARALLEL EFFICIENCY EVALUATION

Parallel Efficiency is the ratio between the speed up and the number of processors. Fig.9 shows the parallel efficiency of blowfish algorithm on different plaintext size 8 M byte, 40 M byte, 160 M byte on different number of processors.

From Fig. 8 we note that the parallel efficiency for blowfish algorithm is the best when the number of processors = 2,3,4,8 then the parallel efficiency decrease when we increase the number of processors from 16-128.

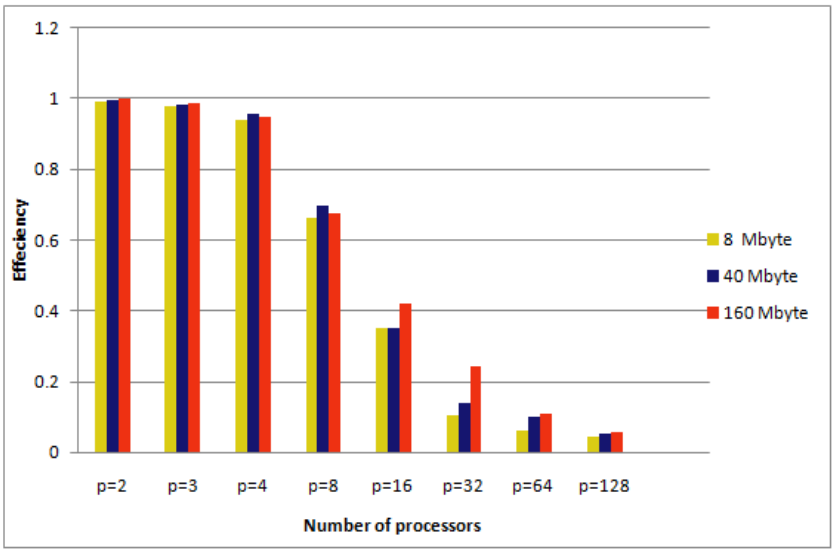

Figure 9. The Efficiency of the blowfish algorithm on different number of processors and different plaintext size

5. CONCLUSION

This paper represents the performance evaluation of blowfish algorithm in the parallel platform, the algorithm is implemented using MPI library, and the experiment is performed on an IMAN1 supercomputer. The experimental results show that the run time of blowfish algorithm is decreased when increasing the number of processors, also the speed up increased when the number of processors increased, it achieved the best value when the number of processors=32 for a plaintext size of 160 Mbyte, more over the parallel efficiency is the best when the number of processors is 2,4,8 it achieve up to 99% , 98% , 66%,respectively ,while in number of processors 16 , 32, 64, 128 the parallel efficiency achieve up to 42% ,24% ,10% ,5% respectively.

The evaluation leads us to the fact that implementing blowfish algorithm in parallel structure will increase the speed of algorithm and this can be done by implementing the blowfish on multiprocessors platforms or on hardware using hardware description language which uses parallelism in the execution of the algorithm.

REFERENCES

[1] Burak, D., 2015. Parallelization of an encryption algorithm based on a spatiotemporal chaotic system and a chaotic neural network. Procedia Computer Science, 51, pp.2888-2892.

[2] Elminaam, D.S.A., Kader, H.M.A. and Hadhoud, M.M., 2008. Performance evaluation of symmetric encryption algorithms.IJCSNS International Journal of Computer Science and Network Security, 8(12), pp.280-286.

[3] Nie, T., Song, C. and Zhi, X., 2010, April. Performance evaluation of DES and Blowfish algorithms.In Biomedical Engineering and Computer Science (ICBECS), 2010 International Conference on (pp. 1-4). IEEE.

[4] Gatliff, B., 2003. Encrypting data with the Blowfish algorithm. Embedded Systems Programming, 16(8), pp.28-35.

[5] Schneier, B., 1993, December. Description of a new variable-length key, 64-bit block cipher (Blowfish). In International Workshop on Fast Software Encryption (pp. 191-204). Springer, Berlin, Heidelberg.

[6] Mahajan, T. and Masih, S., 2015, September. Enhancing Blowfish file encryption algorithm through parallel computing on GPU. In Computer, Communication and Control (IC4), 2015 International Conference on (pp. 1-4). IEEE.

[7] Oukili, S. and Bri, S., 2016. High Throughput Parallel Implementation of Blowfish Algorithm. Applied Mathematics & Information Sciences, 10(6), pp.2087-2092.

[8] Saadeh, M., Saadeh, H. and Qatawneh, M., 2016. Performance Evaluation of Parallel Sorting Algorithms on IMAN1 Supercomputer. International Journal of Advanced Science and Technology, 95, pp.57-72.

[9] Sleit, A., AlMobaideen, W., Qatawneh, M. and Saadeh, H., 2009. Efficient processing for binary submatrix matching. American Journal of Applied Sciences, 6(1), p.78.

[10] Qatawneh, M., 2011. Multilayer Hex-Cells: A New Class of Hex-Cell Interconnection Networks for Massively Parallel Systems. International journal of Communications, Network and System Sciences, 4(11), p.704.

[11] Qatawneh, M., 2011. Embedding Binary Tree and Bus into Hex-Cell Interconnection Network. Journal of American Science, 7(12), p.0.

[12] Mohammad, Q. and Khattab, H., 2015. New Routing Algorithm for Hex-Cell Network. International Journal of Future Generation Communication and Networking, 8(2), pp.295-306.

[13] Qatawneh, M., 2016. New Efficient Algorithm for Mapping Linear Array into Hex-Cell Network. International Journal of Advanced Science and Technology, 90, pp.9-14.

[14] Qatawneh, M., Sleit, A. and Almobaideen, W., 2009. Parallel implementation of polygon clipping using transputer. American Journal of Applied Sciences, 6(2), pp.214.

[15] Mohammd Qatawneh, Ahmad Alamoush, and Ja'far Alqatawna, 2015. Section Based Hex-Cell Routing Algorithm (SBHCR). International Journal of Computer Networks & Communications (IJCNC), 7(1).

[16] Qasem, Mais Haj, Qatawneh, Mohammad, 2018. Parallel Hill Cipher Encryption Algorithm. International Journal of Computer Applications, 179(19), pp. 16-24.

[17] Sumitra, 2013. Comparative Analysis of AES and DES security Algorithms. International Journal of Scientific and Research Publications, 3(1).

[18] Milind Mathur, Ayush kesarwani, 2013. Comparison between DES, 3DES, RC2, RC6, Blowfish and AES. Proceedings of National Conference on New Horizons in IT- NCNHIT

[19] S.Z.S. Idrus,S.A.Aljunid,S.M.Asi, 2008.'Performance Analysis of Encryption Algorithms Text Length Size on Web Browsers," IJCSNS International Journal of Computer Science and Network Security, VOL.8 No.1, pp 20-25.

[20] N. El-Fishawy,2007. Quality of Encryption Measurement of Bitmap Images with RC6, MRC6, and Rijndael Block Cipher Algorithms, International Journal of Network Security, pp.241–251.

[21] Shasi Mehlrotra seth, Rajan Mishra, 2011.Comparative Analysis of Encryption Algorithms For Data Communication, IJCST Vol. 2, Issue 2.

AUTHORS

Mahmoud Asassfeh, is an admitted PhD candidate in Computer Science in Jordan University,he received his Master degree in computer engineering from Yarmouk University,his research interests in network security and parallel computing.

Mohammad Qatawneh, is a Professor at computer science department, the University of Jordan. He received his Ph.D. in computer engineering from Kiev University in 1996. Dr. Qatawneh published several papers in the areas of parallel algorithms, networks and embedding systems. His research interests include parallel computing, embedding system, and network security.

Feras AL-Azzeh is a Professor at department of computer information systems, ALzaytoonah University of Jordan, he received his PhD in Electronic Systems and Programmable Control Systems from USSR in 1991. His research interest include IT Quality Standards and IT Arabization.