Task Parallelism using Distributed Computing for Encryption and Decryption

Ms. Pallavi S. Shendekar[*], Prof. Vijay S. Gulhane

[*] Department of Information Technology, Sipna College of Engineering and Technology, Amravati
[*] Department of Information Technology, Sipna College of Engineering and Technology, Amravati

Abstract- In today's world internet is being used by almost everyone. Numerous file exchanges take place online including many official documents. These files require some sort of security mechanisms while being transmitted over the internet. Technology has done a great deal for changing the way we live and do business today. We can see the use of computers from the vegetable shop to large scale businesses. In this fast moving world we need something essential for fast computation. Along with the popular use of computer, information security has also become one of the problems which need to be solved. Many security issues like the malware authors, information leakage, endangerment and unauthorized exploitation need to be taken into account. To control this, crypto-security is necessary. More Applications started to use Advanced Encryption Standard (AES). However, Since AES on large blocks is computationally intensive and largely byte-parallel. This paper presents the parallel implementation of AES algorithm in a distributed environment for data conversion which provides flexibility & performance improvement in terms of speed-up. System will parallel perform the encryption and decryption process in a distributed environment and the performance analysis shows improvement in terms of execution time**.**

Index Terms- *Distributed computing, parallel processing, data conversion, distributed programming, Advanced Encryption Standard Algorithm.*

I. INTRODUCTION

Early day's people used paper to record the data. With the evolution of computer this has been changed, we started using computers to store data. Instead of letters we started using email, securing information has also increased with this .Where cryptography has become mandatory. For the protection of data transmission over insecure channels two types of cryptographic systems are used: Symmetric and Asymmetric cryptosystems,[1]. Symmetric cryptosystems such as Data Encryption Standard (DES) and Advanced Encryption Standard (AES) uses an identical key for both to encrypt the plain text and decrypt the cipher text. Asymmetric cryptosystems such as Rivest-Shamir-Adleman (RSA) & Elliptic Curve Cryptosystem (ECC) uses different keys for encryption and decryption. Symmetric cryptosystem is more suitable to encrypt large amount of data with high speed.

To replace the old Data Encryption Standard, in Sept 12 of 1997, the National Institute of Standard Technology (NIST) required proposals to what was called Advanced Encryption Standard (AES) [2]. Many algorithms were presented originally with researches from 12 different nations. On October 2nd 2000, NIST has announced the Rijndael algorithm is the best in security, performance, efficiency, implement ability, & flexibility. The Rijndael algorithm was developed by Joan Daemen of Proton World International and Vincent Rijmen of Katholieke University at Leuven. It became effective as a Federal government standard on May 26, 2002 after approval by the Secretary of Commerce. It is available in many different encryption packages. AES is the first publicly accessible and open cipher approved by the National Security Agency (NSA) for top secret information. So, it has broad applications, such as smart cards and cell phones, WWW servers and automated teller machines, and digital video recorders. Numerous architectures have been proposed for the hardware implementations of the AES algorithm.

To estimate the speedup of a tightly coupled system on a single application, we use a model of parallel computation introduced by Ware[3]. We define a as the fraction of work in the application that can be processed in parallel. Then we make a simplifying assumption of a two-state machine; that is, at any instant either all processors are operating or only one processor is operating. Consider the condition user having 10000 document to process and each having large data in this case project need to employee the system which will transfer processing over the network system and save output on the server or main system. So the proposed system will aimed at parallel processing of provided task.

In Distributed Computing approach, it is followed to assign a job to a processor if it is idle. The focus is now on how to optimize re-sources to decrease the energy consumption by volumes of computing equipments to deal with green and sustainability issues[4]. Various hardware and software architectures are used for distributed computing. At a lower level, it is necessary to interconnect multiple CPUs with some sort of network, regardless of whether that network is printed onto a circuit board or made up of loosely-coupled devices and cables. At a higher level, it is necessary to interconnect processes running on those CPUs with some sort of communication system.

II. LITERATURE SURVEY

Various algorithm and models have been proposed, mostly heuristic in nature, as the optimal solution often requires future knowledge and is computationally intensive. Multiple tasks can be executed simultaneously in a given message-passing system. To achieve good performance, the system has to recognize efficiently free subsystems of various sizes for incoming tasks. Parallel processing is a form of computation in which many calculations are carried out simultaneously,[5] operating on the principle that large problems can often be divided into smaller ones, which are then solved concurrently ("in parallel"). There are several different forms of parallel computing: bit-level, instruction level, data, and task parallelism. As power consumption (and consequently heat generation) by computers has become a concern in recent years,[6] parallel computing has become the dominant paradigm in computer architecture, mainly in the form of multicore processors.[7] Parallel computers can be roughly classified according to the level at which the hardware supports parallelism, with multi-core and multi processor computers having multiple processing elements within a single machine, while clusters, MPPs, and grids use multiple computers to work on the same task. Specialized parallel computer architectures are sometimes used alongside traditional processors, for accelerating specific tasks.

Parallel computer programs are more difficult to write than sequential ones,[8] because concurrency introduces several new classes of potential software bugs, of which race conditions are the most ommon. Communication and synchronization between the different subtasks are typically some of the greatest obstacles to getting good parallel program performance.

A. Parallel Programming Model

Parallel Programming Models typically falls into one of several categories: Shared memory model, Thread model, Distributed memory model, Data parallel model, Hybrid model, SPMD and MPMP model.

Shared Memory Model

In this programming model, task shares a common address space, which they read and write to asynchronously. Various mechanisms such as locks or semaphores may be used to control access to the shared memory.

Thread Model

This programming model is a type of shared memory programming. In the threads model of parallel programming, a single "heavy weight" process can have multiple "light weight", concurrent execution paths.

Distributed Memory Model

A set of tasks that use their own local memory during computation. Multiple tasks can reside on the same physical machine and/or across an arbitrary number of machines. Tasks exchange data through communications by sending and receiving messages.

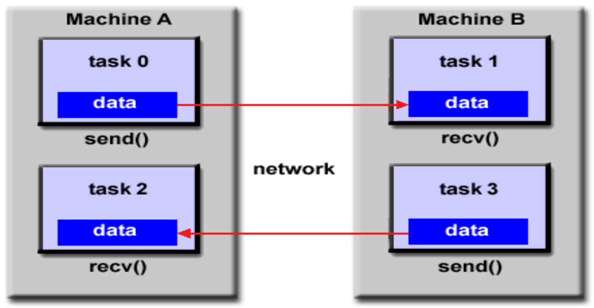

Fig.1 Distributed Memory Model

Data Parallel Model

Data Parallel Model is also referred to as Partitioned Global Address Space (PGAS) model. In this model address space is treated globally. The data set is typically organized into a common structure, such as an array or cube.

Hybrid Model

A hybrid model combines more than one of the above programming models. A common example of a hybrid model is the combination of the message passing model (MPI) with the threads model (OpenMP).

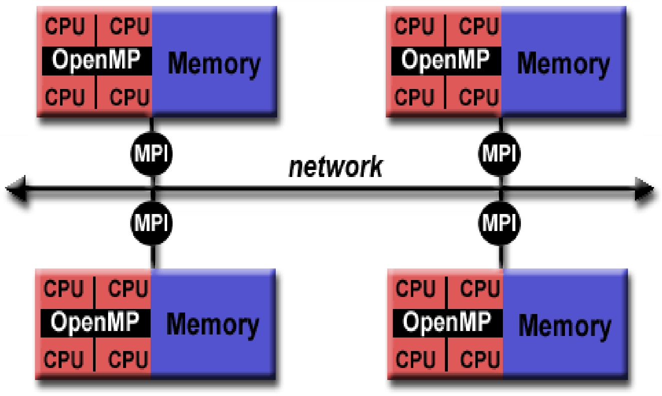

Fig.2 Hybrid Model

Single Program Multiple Data (SPMD)

SPMD is high level programming model that can be built upon any of the above programming model. This program can be thread, message passing, data parallel or hybrid. The SPMD model, using message passing or hybrid programming, is probably the most commonly used parallel programming model for multi-node clusters.

Multiple Program Multiple Data (MPMD)

Similar to SPMD, MPMD is a high level Programming model. MPMD applications are not as common as SPMD application.

Another basic aspect of distributed computing architecture is the method of communicating and coordinating work among concurrent processes. Through various message passing protocols, processes may communicate directly with one another, typically in a master/slave relationship. Alternatively, a "database-centric" architecture can enable distributed computing to be done without any form of direct inter-process communication, by utilizing a shared database.

PROPOSED ARCHITECTURES

In the proposed architectures the main factors are the designing the distributed system and parallel processing through a specific problem domain. Here the problem domain is known and well defined, the environment in which the system run is also well defined. Figure 2 shows the basic block diagram of the system architecture. Actual task processing needs a series of steps to be performed. These series of processes are simultaneously executed on different client machines. Basically here we are distributing the no. of files on to the network through the shared database.

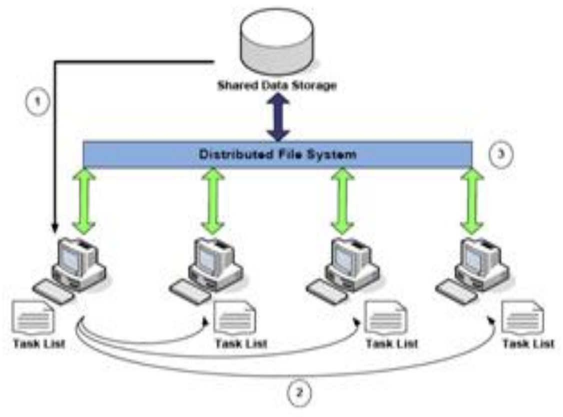

Fig. 3 System Architecture

AES Algorithm

Advanced Encryption Standard (AES) is a variant of Rijndael cipher algorithm, a symmetric block cipher which translates the plaintext into cipher text in blocks. This algorithm has the fixed input block size of 128 bits and the key size of 128, 192, 256 bits. The algorithm's operations are performed on a two-dimensional array of bytes called the State**.** The State consists of four rows of bytes, each containing nb bytes calculated by taking ratio of the total number of data (input) bits and 32. (128/32=4). The Cipher (encrypt) or Inverse Cipher (decrypt) operations are then conducted on this State array, after which the final value is copied to the output array. The AES algorithm, unlike DES does not form a fiestal network, where encryption and decryption processes are similar. AES processes data blocks of 128 bits, using cipher keys with lengths of 128, 192, and 256 bits.

The Encryption and decryption process consists of a number of different transformations applied consecutively over the data block bits, in a fixed number of iterations, called rounds (Nr), which depends on the length of the key used. The Middle round will undergo Nr-1 iterations.

The Encryption process consists of 4 phases. They are:

1. Key Expansion
2. Initial Round
   - a. AddRoundKey
3. Middle Rounds
   - a. Substitute Bytes
   - b. Shift Rows
   - c. Mix Columns
   - d. Add Round Key
4. Final Round
   - a. Substitute Bytes
   - b. Shift Rows
   - c. Add Round Key

The phases of Decryption are (for the particular cipher text):

1. Key expansion
2. Initial round
   - a. Add Round Key
3. Middle Round
   - a. Inverse Shift Rows
   - b. Inverse Substitute Bytes
   - c. Add Round Key
   - d. Inverse Mix Columns
4. Final Round
   - a. Inverse Shift Rows
   - b. Inverse Substitute Bytes
   - c. Add Round Key

The algorithm for encryption/decryption process is shown in figure 4

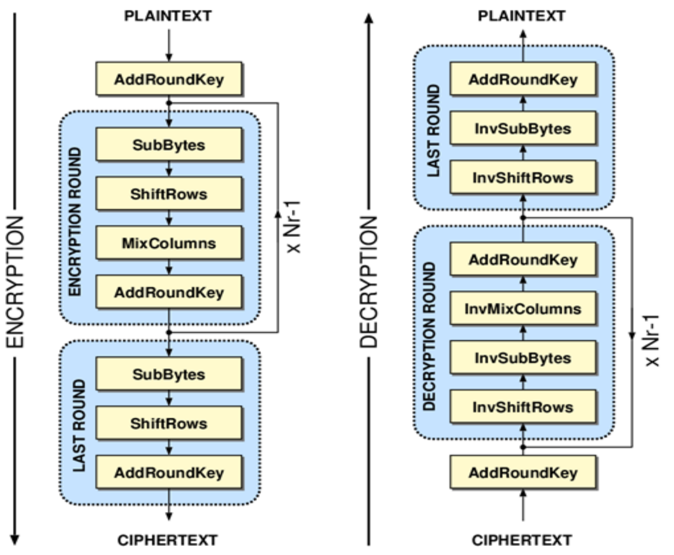

Fig. 4 AES Algorithm

Key Expansion:

Key expansion takes the input key of 128, 192 or 256 bits and produces an expanded key for use in the subsequent stages. The expanded key’s size is related to the number of rounds to be performed. For 128-bit keys, the expanded key size is 352 bits. For 192 and 256 bit keys, the expanded key size is 624 and 960 bits. It is the expanded key that is used in subsequent phases of the algorithm. During each round, a different portion of the expanded key is used in the AddRoundKey step.

AddRoundKey:

During this stage of the algorithm, the message is combined with the state using the appropriate portion of the expanded key.

Sub Bytes:

During this stage, the block is modified by using an 8-bit substitution, or SBox. This is a non-linear transformation used to help avoid attacks based on algebraic manipulation.

Shift Rows:

This stage of the algorithm shifts cyclically shifts the bytes of the block by certain offsets. Blocks of 128 and 192 bits leave the first 32-bits alone, but shift the subsequent 32-bit rows of data by 1, 2 and 3 bytes respectively.

Mix Columns:

This stage takes the four bytes of each column and applies a linear transformation to the data. The column is multiplied by the coefficient polynomial c(x) = 3x3+x2+x+2 (modulo x4+1). This step, in conjunction with the Shift Rows step, provides diffusion in the original message, spreading out any non-uniform patterns. At the end of the algorithm, the original message is encrypted. To decrypt the cipher text, the algorithm is essentially run in reverse, however, if the key used for Key Expansion is not known, a brute-force attack on the cipher could take thousands of years.

The decryption implementation results are similar to the encryption implementation. The key expansion module is modified in the reverse order. In which last round key is treated as the first round and decreasing order follows.

Model Implemented

Implementing the system is to fairly offer flexibility on memory access in distributed environment using parallel processing. Adapting an algorithm, that has a computational complexity that consists in simple byte operations (AND, OR, XOR, Shifting, 32 bit adding etc.) is the starting point of our scientific research. The source has two main entries: Encrypt () and Decrypt (). These functions take in a plain text / cipher text 128-bit source block, and output an encrypted/decrypted 128-bit block.

At a higher level, Encrypt () and Decrypt () functions take in a 128-bit key and a variable length message. They split the message into 128-bit blocks, appropriately pad the block when the message size is a multiple of 128 bits, and pass the blocks to the Encrypt () and Decrypt () functions. This naturally leads to a highly parallel implementation. The thread perform the Encrypt () and Decrypt () functions in parallel. Each thread works on a subset of the data, so there are no dependencies between threads. This assumes that the cipher is used in parallel-friendly modes. Since the purpose of any cipher is to quickly encrypt incoming data, the performance metric we picked is the run-time of the algorithm.

RESULT

The results were achieved by running a random data set through the encryption and decryption modules 5 times

S. No	Input file size(KB)	Normal Encryption time(ms)	Normal Decryption time (ms)	Parallel Encryption time(ms)	Parallel Decryption time (ms)
1	23	30.6846	34.62628	1.1432	1.276
2	58	30.786	34.551	3.777	4.231
3	115	32.1342	36.0249	7.8651	8.673
4	457	39.4578	44.113	9.2004	11
5	914	78.4082	88.131	15.221	17.8576

CONCLUSION

In this paper, an efficient way to enhance the encryption/ decryption using parallel processing in distributed environment is proposed. This report presents the most efficient, currently known approaches in encryption and decryption of text with AES on parallel processing to achieve great speed. If the amount of data is large, the encryption/decryption time required is greatly reduced, if it runs on a parallel processing in a distributed environment.

Future work will include efficient implementations of other common symmetric and asymmetric algorithms. Parallel implementations of hashing and public key algorithms may also be implemented, in order to create a complete cryptographic framework in a distributed environment.

REFERENCE

[1] Piotr Bilski , Wiesław Winiecki, 2010 ,Multi-core implementation of the symmetric cryptography algorithms in the measurement system, Measurement 43 (2010) 1049–1060, Elsevier. 10.1016/j.measurement.2010.03.002

[2] Behrouz A. Forouzan, De Anza College Cryptography and Network Security (McGraw-Hill,2007)

[3] Wang Y. and Morris R. (1985) IEEE Trans. Computing, 34(3), 204-217. Stone H.S. (1977) IEEE Trans of Software Engineering, 3(1), 95-93.

[4] Joshi E. International Journal of Computer Applications, 1(18), 0975 - 8887.

[5] Gottlieb, Allan; Almasi, George S. (1989). Highly parallel computing. Redwood City, Calif.: Benjamin/Cummings.ISBN 0-8053-0177-1.

[6] S.V. Adve et al. (November 2008). "Parallel Computing Research at Illinois: The UPCRC Agenda" (PDF). Parallel@Illinois, University of Illinois at Urbana-Champaign. faster."

[7] Asanovic, Krste et al. (December 18, 2006). "The Landscape of Parallel Computing Research: A View from Berkeley"(PDF). University of California, Berkeley. Technical Report No. UCB/EECS-2006-183.

[8] Hennessy, John L.; Patterson, David A., Larus, James R. (1999).Computer organization and design : the hardware/software interface(2. ed., 3rd print. ed.). San Francisco: Kaufmann.ISBN 1-55860-428-6.

AUTHORS PROFILE

Ms. Pallavi S. Shendekar received her BE degree in Information Technology from Amravati University / India. She is working towards her Master in Information Technolgy from Amravati University. She has also attended various national and international Conferences. Her research is focused on Data Processing, Neural networks.

Prof. Vijay. S. Gulhane is Associate Professor of Department of Computer Science & Engineering at Sipna College of Engineering Technology, Amravati. He did his B.E in Computer Science & Engg. & M.E. in Computer Science & Engg. Amravati. He is working towards his PhD in Computer Science and Engineering from Amravati University. He has 16 years of experience & published 46 papers in International Journal and Conference also. He is life member of I.S.T.E.; Fellow of I.E.T.E. and I.E. (India).