Journal Of Harmonized Research (JOHR)

Journal Of Harmonized Research in Engineering
2(1), 2014, 42-48

ISSN 2347 – 7393

Original Research Article

SECURE DATABASE USING GPGPU

Arvind Pawar, Avadhut Sonawane, Nikhil Jamdar,Siddharth Shingavi

BE,Comp, Sinhgad College of Engineering Pune- 41

Abstract- Database maintains huge records that may contain sensitive information such as credit card number, passwords, medical reports, email-ids, etc. whose confidentiality must be protected. Encryption is a one way to ensure confidentiality of database records. Database Encryption when performed on CPU takes more time and thus degrades the performance. The Graphics Processing Units (GPU) that supports parallel computing can be used to enhance the performance of cryptographic operations on database records. The symmetric block cipher cryptographic algorithms like Advanced Encryption Standard (AES) that are generally used to encrypt databases, can be implemented on GPU. The GPGPU implementation gives huge performance gain in AES cryptography. We have also proposed implementation of AES using dynamic parallelism in Kepler architecture for getting more performance gain.

Keyword *- Graphics Processing Unit; Database Security; Dynamic Parallelism; Advanced*

Introduction

Nowadays, the use of personal and sensitive data on network has been rapidly increased. The security of this data is a big issue. Traditionally, databases were not completely encrypted due to performance related issues.

For Correspondence:
arvindpawar4444ATgmail.com
Received on: February 2014
Accepted after revision: February 2014

To increase the performance, General Purpose Graphics Processing Unit (GPGPU) could be used instead of CPU. The GPGPU is a currently emerging new technology that allows the parallel computation ability of GPU processors for general purpose computation [1]. For computations that supports single instruction multiple data execution mode, GPU

shows outstanding performance improvement. Recently, one of the largest email service provider worldwide, faced security breach that

made them suffered the exposure of email-ids and passwords of its email customers [2].Hackers were looking for additional email addresses to send spam or scam messages. Access to email addresses could lead to more serious breaches involving banking and shopping sites. That's because many sites uses email to reset passwords.

The 128 bit AES symmetric block cipher algorithm could be used to secure database records. Some modes of AES algorithm supports parallelism and thus can be implemented on GPU to achieve better performance than conventional CPU implementation.

Proposed System

The architecture involves basically three modules namelydecision module, Host AES and GPU AES. The parameter for decision module is the size of data to be encrypted or decrypted. If size of data is large, then GPU AES module will be invoked else Host AES module is invoked.

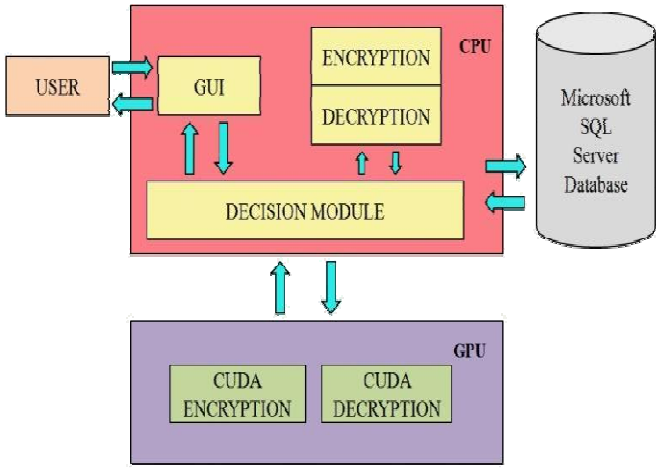

Fig. 1Proposed System

The user will enter the query on provided user interface. Based on the type of query, the decision for encryption or decryption of records will be taken. If it is a insert query, the record attribute values are extracted and send for encryption on GPU. Thereafter, these encrypted values are replaced with actual plain values within record and then the insert query with modified record is forwarded to database engine for execution.

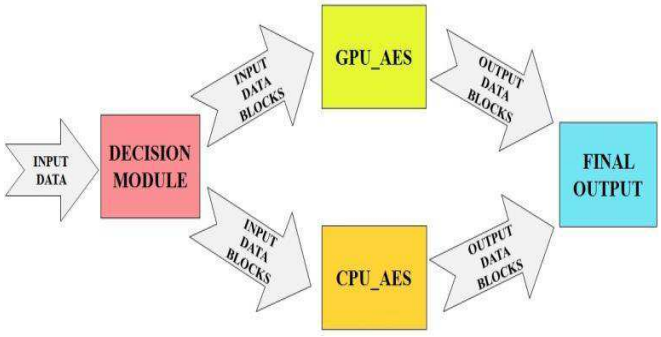

Fig. 2 Decision Module

Cuda Programming

Compute Unified Device Architecture (CUDA) is a parallel computing platform and programming model invented by NVIDIA[3]. Its programming model is based on ANSI C, extended with several keywords and constructs. The programmer writes a single source program that contains both the host (CPU) code and the device (GPU) code. These two parts are automatically separated and compiled by the CUDA compiler tool chain[6].

CUDA allows the programmer to write device code in C functions called kernels. A kernel is different from a regular function in that it is executed by many GPU threads in a Single-Instruction Multiple-Data (SIMD) fashion. Each thread executes the entire kernel once. While launching a kernel for GPU execution, number of blocks and threads per block must be specified.

Each GPU thread has a unique identification and has access to multiple GPU memories during kernel execution. Globally, all threads have read and write access to the global memory, and read-only access to the constant memory and the texture memory. Since the GPU threads cannot access host memory, data needed by a kernel must be transferred to these GPU memories before it is launched.

AES ALGORITHM

The Advanced Encryption Standard is based on the Rijndael algorithm developed by two Belgian cryptographers, Joan Daemen and Vincent Rijmen. It is a symmetric key block cipher algorithm i.e. same key is used while encrypting and decrypting data blocks. The data blocks are of 128 bit size. The key sizes comes in three variations - 128 bit key, 192 bit key or 256 bit key. The AES cipher calculation is done by applying same transformations rounds repetitively for specific no of times which is based on the key size. Each round consists of several processing steps, including one that depends on the encryption key.

The AES-128 algorithm is iterative and consists of 10 rounds. The input is a block of data and the initial key [4]. This initial key is expanded to generate set of keys for each round, before applying transformations. Each round operates on the intermediate result of the previous round and is a sequence of transformations, namely SubBytes, ShiftRows, MixColumns and AddRoundKey. The intermediate result of any step is called the state. The final round is slightly different and the output after 10 rounds is the block of encrypted data.

The AES algorithm has six mode of encryption of which Counter Mode and Electronic Code Book (ECB) supports parallelism [5]. For database encryption, counter mode is normally preferred as every record will seem differently after encryption. We implemented ECB mode of AES for database encryption on GPU as well as on CPU and computed speedup.

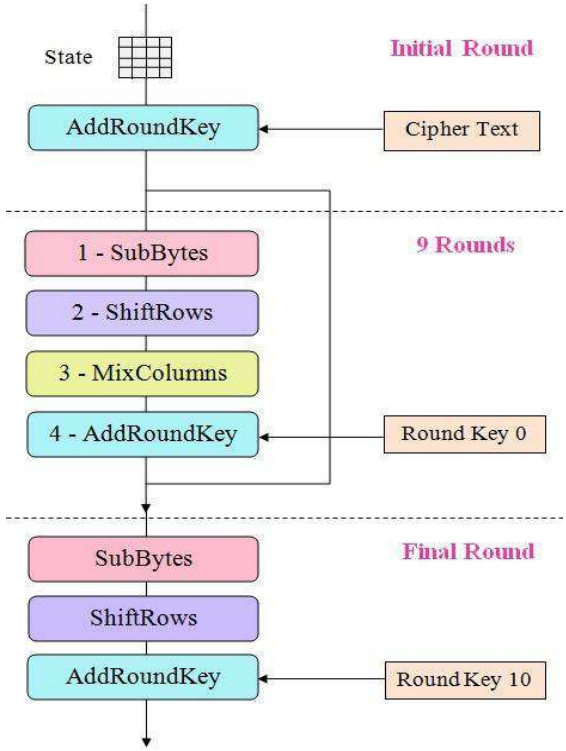

Fig.3 Overview of 128 bit AES algorithm

1. AES IMPLEMENTATION ON GPU

Before inserting records within database, 16 byte data blocks are formed out of these records, column-wise. If necessary, padding is done. Thereafter, these well-formed 16 byte data blocks are transferred to GPU for encryption. The number of data blocks decides the number of threads to be created. Each thread in kernel then performs operations on its corresponding data block logically at same time. The Key Expansion is performed on CPU, as its computations are inherently serial. The sub-functions of AES are implemented as simple device functions. After execution of kernel, the data blocks are copied back to CPU. The AES round key and lookup tables are stored within constant memory rather than slow global memory to optimize the execution of AES on GPU. A data structure is maintained to reconstruct the records from these data blocks after encryption.

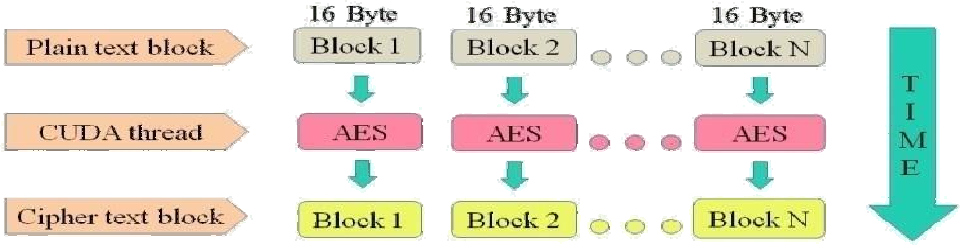

Fig. 4 Logical Execution of CUDA AES

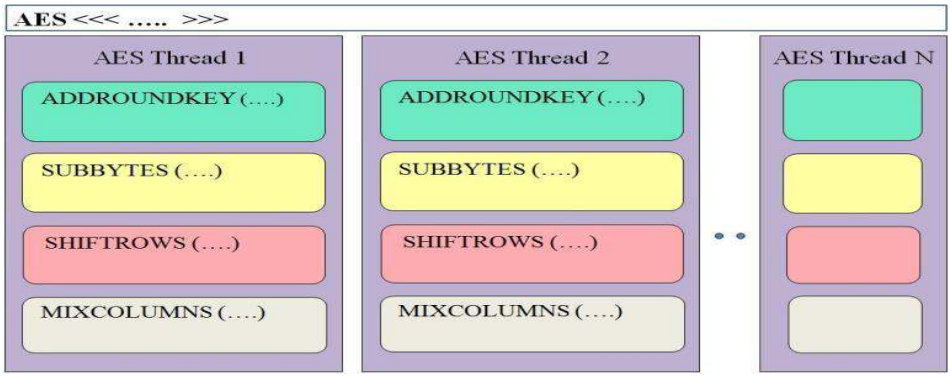

Fig. 4 Simple AES Kernel

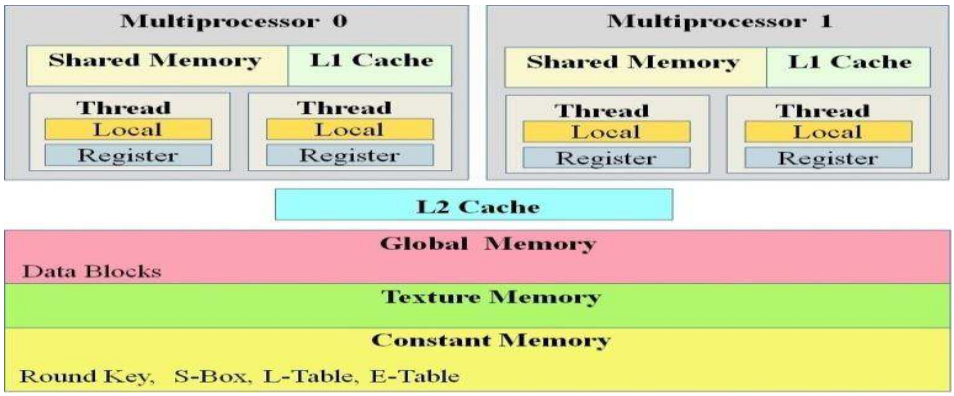

Fig. 5 Memory used on GPU

Dynamic Parallelism

NVIDIA's new Kepler GK110 GPU's has introduced dynamic parallelism technology that enables CUDA kernel to launch other kernels using CUDA runtime API[7]. Till now, AES sub-functions were implemented as device functions on GPU due to lack of dynamic parallelism support. Now on this new dynamic parallelism supported GPU's, we propose to implement these sub-functions as CUDA kernel and launch these sub-function kernels from parent AES kernel.

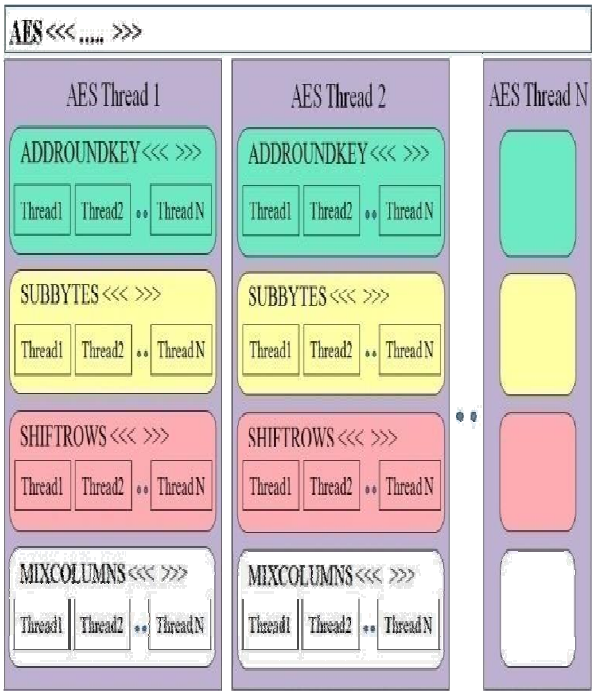

Fig. 6 AES using Dynamic Parallelism

Experimental Results

Processor: Intel Core i5 M 460, 2.53 GHz

GPU-1 : GeForce GT 420M,96 CUDA cores, 1 GHz

GPU-2 : Tesla S2050, 448 CUDA cores, 1.15 GHz

TABLE -I
AES Encryption Results on GeForce GPU

Data Size	Core i5 M 460 Sequential (sec)	GeForce GT 420M GPU (sec)	Speed up
16 KB	0.015	0.001	15.0
32 KB	0.018	0.001	18.0
64 KB	0.038	0.003	12.6
128 KB	0.075	0.005	15.0
256 KB	0.158	0.010	15.8
512 KB	0.304	0.019	16.0
1 MB	0.612	0.038	16.1
2 MB	1.255	0.076	16.5
4 MB	2.304	0.152	15.1
8 MB	4.773	0.303	15.7
16 MB	9.094	0.610	14.9
32 MB	17.815	1.210	14.7
64 MB	35.605	2.419	14.7
128 MB	72.684	4.852	14.9
256 MB	150.884	9.695	15.5
512 MB	299.998	20.793	14.4

The total time required for AES encryption or decryption increases with the increase in data size. The parallel AES implementation on GPU achieves massive speedup over the serial AES implementation on CPU. The decryption shows more speedup than encryption operation. The GPU-1 gives an approximate speedup of 15.3x for encryption and 17.1x for decryption respectively. When the data size is less, then CPU gives better performance than GPU. The most effective use of GPU resources are when the data size is sufficiently large.

Data Size	Core i5 M 460 Sequential (sec)	Tesla S2050 GPU (sec)	Speedup
16 KB	0.015	0.0003	50.0
32 KB	0.018	0.0003	60.0
64 KB	0.038	0.0004	95.0
128 KB	0.075	0.0008	93.7
256 KB	0.158	0.0012	131.6
512 KB	0.304	0.0024	126.6
1 MB	0.612	0.0044	139.0
2 MB	1.255	0.0084	149.4
4 MB	2.304	0.0165	139.6
8 MB	4.773	0.0331	144.1
16 MB	9.094	0.0656	138.6
32 MB	17.815	0.1302	136.8
64 MB	35.605	0.2598	137.0
128 MB	72.684	0.5185	140.1
256 MB	150.884	1.0365	145.5
512 MB	299.998	2.0721	144.7

TABLE II

AES Decryption Results on GeForce GPU

Data Size	**Core i5 M 460 Sequential (sec)**	**GeForce GT 420M GPU (sec)**	**Speedup**
16 KB	0.015	0.001	15.0
32 KB	0.031	0.002	15.5
64 KB	0.062	0.004	15.5
128 KB	0.117	0.007	16.7
256 KB	0.235	0.013	18.0
512 KB	0.460	0.026	17.6
1 MB	0.935	0.053	17.6
2 MB	1.825	0.105	17.3
4 MB	3.874	0.210	18.4
8 MB	6.780	0.420	16.1
16 MB	14.804	0.839	17.6
32 MB	28.520	1.675	17.0
64 MB	57.995	3.353	17.2
128 MB	113.474	6.704	16.9
256 MB	220.477	13.423	16.4
512 MB	458.532	28.231	16.2

TABLE III

AES Encryption Results on Tesla GPU

TABLE IV

AES Decryption Results on Tesla GPU

Data Size	Core i5 M 460 Sequential (sec)	Tesla S2050 GPU (sec)	Speedup
16 KB	0.015	0.0003	50.0
32 KB	0.031	0.0004	77.5
64 KB	0.062	0.0004	155.0
128 KB	0.117	0.0008	146.2
256 KB	0.235	0.0012	195.8
512 KB	0.460	0.0024	191.6
1 MB	0.935	0.0044	212.5
2 MB	1.825	0.0084	217.2
4 MB	3.874	0.0165	234.7
8 MB	6.780	0.0331	204.8
16 MB	14.804	0.0656	225.6
32 MB	28.520	0.1302	219.0
64 MB	57.995	0.2604	222.7
128 MB	113.474	0.5181	219.0
256 MB	220.477	1.0350	213.0
512 MB	458.532	2.0956	218.8

IV.CONCLUSION

In this paper, techniques to improvise the performance of database related cryptographic operations on GPU were proposed and the results were summarized. By comparing, cryptographic operations implementation on Tesla S2050 GPU with conventional CPU implementation, we got the approximate speedup of 123.2x and 187.7x for encryption and decryption respectively.The parallel computing ability of modern GPU can be used to accelerate the performance of encryption and decryption operation on huge databases. Cloud service provider can use GPUs to reduce the performance degradation cause by the cryptographic operations in their system.

We expect that the block cipher cryptographic algorithm like AES when implemented using dynamic parallelism technology, recently introduced in NVIDIA graphics card, will further improvise the performance bit more.

V.ACKNOWLEDGMENT

Free Access to the Tesla S2050 GPUwas provided by NVIDIA through cluster, for testing GPU code.

REFERENCES

7. InkyungJeun, Hyun-Chul Jung, Dongho Won and Nan Ki Lee, "Database Encryption Implementation and Analysis using Graphics Processing Unit", *IEEE Third FTRA International Conference on Mobile, Ubiquitous, and Intelligent Computing*, 2012.

8. http://abcnews.go.com/Technology/wireStory/yahoo-email-account-passwords-stolen-22305108

9. Qinjian Li, ChengwenZhong, Kaiyong Zhao, Xinxin Mei and Xiaowen Chu. "Implementation and Analysis of AES Encryption on GPU", *IEEE 14th International Conference on High Performance Computing and Communications*, 2012.

[4] Deguang Le, Jinyi Chang", Xingdou Gou, Ankang Zhang, and Conglan Lu, "Parallel AES Algorithm for Fast Data Encryption on GPU", *IEEE 2nd International Conference on Computer Engineering and Technology*, 2010.

[5] AtulKahate, *Cryptography and Network Security*, Second Edition, Tata McGraw-Hill Education.

[6] NVIDIA, *CUDA C Programming Guide.* http://docs.nvidia.com/cuda/pdf/CUDA_C_Programming_Guide.pdf

[7] NVIDIA, *Dynamic Parallelism in CUDA.* http://developer.download.nvidia.com/assets/cuda/files/CUDADownloads/TechBrief_Dynamic_Parallelism_in_ CUDA.pdf