Volume 6, Issue 7, July 2016 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Accelerating Hybrid Cryptographic Algorithm Using GPU

Soumya Unnikrishnan
M.tech Scholar, Department of CSE
Amrita Vishwa Vidyapeetham (University)
Bengaluru, Karnataka, India

Tirumale K.Ramesh
Professor, Department of CSE
Amrita Vishwa Vidyapeetham (University)
Bengaluru, Karnataka, India

Abstract— Data security is foremost important in on-line transaction especially when the exchanged data contains sensitive information. Data encryption is perceived as one of the good technique to avoid unauthorized use of such sensitive information during on-line transaction. There are many encryption algorithms in practice and Advanced Encryption System (AES) is one of the latest encryption algorithms which possess few niche advantages over its predecessors. However, AES cannot ensure completely safe information exchange as the safety depends mainly on the 'key' which is being transmitted along with message. If the key is known to an intruder, he or she can decrypt the message using the AES algorithm. To avoid such situation, this paper combined the AES algorithm with a hash algorithm- Message Digest5 (MD5) to enhance the information security of online transactions. The hash algorithm will generate a hash value from the key of the encrypted message. It is unique to a message in transaction and is irreversible. If the intruder makes any attempt to retrieve the original message during the transaction without authentication, the hash value of the key will get modified when it reaches the receiver. By comparing the hash values of the key at sender and receiver, a prediction can be made whether any loss of integrity happened to the message during transaction. However, the combination algorithm will require increased processing time. To reduce the computation time, the hybrid algorithm is parallelized using graphic processing unit (GPU).

Keywords: Cryptography, HPC, CUDA, AES, MD5.

I. INTRODUCTION

The information security has become one of the most significant problems in data communication. So it becomes an inseparable part of data communication. The security of data transmission is a vital problem in communication networks. A communication system is reliable as long as it provides high level of security. Usually, users exchange personal sensitive information or important documents. In this case; security, integrity, authenticity and confidentiality of the exchanged data should be provided over the transmission medium. Nowadays, internet multimedia is very popular; a significant amount of data is exchanged every second over a non secured channel, which may not be safe. Therefore, it is essential to protect the data from attackers. To protect the data; various cryptographic techniques can be used.

There are mainly three types of encryption: Symmetric, Asymmetric and Hybrid encryption. In symmetric encryption the same key is used for both encryption and decryption. Everyone who obtains knowledge of the key can transform the cipher text back to plain text. If you want to preserve confidentiality, you must protect your key and keep it a secret. Therefore, symmetric encryption is also called private or secret key encryption. Symmetric key encryption algorithms are significantly faster than asymmetric encryption algorithms, which makes symmetric encryption an ideal candidate for encrypting large amounts of data. Examples of popular symmetric encryption algorithms include Twofish, Serpent, AES, 3DES, Blowfish etc.

The real time application scenario where this work is useful is explained as follows. An organization where two employees working in same project, need to exchange information each other. Sometimes the data must be crucial and have to protect from an outsider. One employee of the project is in India and other employee is onsite (overseas). Daily Indian employee needs updates from onsite employee. So large file exchanges (like detailed design of the project) are happening. But these files must be protected between these two employees only. If both the employees agreed on a symmetric key encryption and a shared secret key, the message can be send in an encrypted form. If an intruder from the same company (insider attack) who is intentionally outsourcing the project details to the competitor company, he can also agreed with onsite employee for the shared secret key and can decrypt the crucial data and outsource the project details. So for overcoming this type of situation, one can use different cryptographic methods like symmetric encryption, asymmetric encryption, hashing etc. In this paper, a hybrid cryptographic algorithm is proposed that makes use of both symmetric algorithm and hashing. Both the employees have some set of keys and they will agree on the available set of keys previously. The onsite employee will send the details in an encrypted form using symmetric encryption and key used is any key from the available set of keys. Along with the ciphered text the onsite employee will send the hash of the key to the employee who is working in India. The employee who is working in India also have the same set of keys and he will produce the hashes of the keys

and once the key and encrypted message arrived from onsite employee, he will compare the hash values of the keys available with him and received hash value and he can find the key used for encryption from the available set. If an intruder tries to hack the data in between from the network, the intruder will get only the encrypted data and hash value of the key. Using the hash value of the key the intruder cannot decipher the data and hence the data is safe between the legitimate users.

This paper is focusing on a hybrid cryptographic algorithm for enhancing security using AES and MD5. A combination of two algorithms as hybrid will take more time for encryption/decryption. So in order to reduce the processing time, graphic processing unit is used .NVIDIA CUDA [8] is used to program the proposed algorithm and enhance the GPU [7].

Further the paper is organized as follows: Section-II: describes the related work carried in this area. Section III: proposes a new method of hybrid cryptosystem and also providing methodology to accelerate the proposed algorithm by exploiting the advantages of GPU. In section IV, implementation details is included and Section V is discussing about results and performance analysis. Finally, section VI gives the summarized aspect of the work and the possible future enhancement of the current work.

II. RELATED WORK

Different cryptographic algorithms for data encryption and their acceleration using GPU are studied. Genetic algorithm approach of some cryptographic methods also surveyed and it is very complex in designing phase. This section gives a summary of the background work.

In terms of encryption and decryption various studies are done to enhance the performance of cryptographic algorithms (both for symmetric and asymmetric algorithms) exploring GPU by modifying implementation and utilizing different kinds of GPU.

Traditional GPU architecture is used for achieving parallelism [1]. It is easy to begin with and portable. But data are stored in floating point format in the memory model of OpenGL .So it is not suitable for AES computation, which requires massive numerical calculation. The key used for encryption is vulnerable to attack if it is placed in a network. The feasibility of using GPU for cryptographic processing in symmetric key cryptography is discussed in [2]. The advantage of this paper is that it is comparing the performance of an OpenGL-based implementation with general CPU based implementation. Unavailability of the bitwise logical operation in the programmable shaders is a drawback of this paper. So CPU keeps loaded during all the execution. In paper[3],authors proposed and developed a optimized cryptographic algorithm for wireless sensor network using the technique of genetic programming.This method is highly secure, fast and computationally efficient. Moreover this methodology is consuming less energy and increasing the autonomy of embedded networking devices.

The paper [4] presented a new algorithm that simplifies the creation and expansion process of the encryption key of the AES algorithm. Cryptanalysis is very difficult, ie,the algorithm is highly secure. But the algorithm will take time if the data size is larger.

The paper [5] implemented the AES algorithm using OpenCL.

Throughput is little bit higher compared to serial implementation. The algorithm is using same key for encryption and decryption and the key is not secured in a network. Authenticity of sender cannot be proved by AES algorithm in its design is a major drawback.

The paper [6] is comparing the advantages of MD5 and SHA- 1.MD5 hashing is very easy to do and it is very difficult to revert.MD5 can produce the digest very fast and security is also ensuring.

The literature review covers the main aspects of the paper, which can be divided into two essential parts: security and parallelism. ECB mode of symmetric encryption is regarded as one of the best approach for parallelism and within the symmetric encryption algorithms; AES will provide high performance in the case of both security and speed. Cryptographic hash algorithms are practically impossible to invert. They have been called as the workhorses of modern cryptography. This work proposes to use MD5 algorithm to produce the digest of the key. Many hash algorithms are there in cryptographic world, but MD5 is simple compared to other hash algorithms and it will take less time by utilizing the advantages of GPU.

By using hybrid encryption, secure and convenient key exchange can be achieved while ensuring fast and efficient encryption of large amounts of data. And by using GPU, the performance can be improved in terms of memory and CPU utilization and speed of our cryptographic programs.

Hybrid encryptions can provide high security, however hybrid encryptions are not popular nowadays because of their complexity and low speed while executing on CPU. Usage of GPU will enhance the performance of hybrid encryption. By using AES and MD5 for hybrid encryption on GPU, high level of security can be ensured for sensitive data without compromising speed and CPU utilization. The authenticity of the sender also can be ensured using this method.

III. PROPOSED METHODOLOGY

An approach for authenticity of data is proposed here. Hybrid encryption is proposed for improving the security and authenticity of data without compromising the performance of the system. In our hybrid approach we are using AES encryption for data encapsulation and MD5 hashing for key encapsulation.MD5 algorithm compliments AES algorithm by generating the digest value for the key used for encryption. On sender side, the cipher text of original

message is produced using a key. In a network if this key is compromised, the attacker can easily decrypt the data and also modify the data and can send duplicated data to the receiver. So MD5 hashing algorithm is also used along with the symmetric key approach. The key of the sender is hashed using MD5 and produce a digest value. The sender will send the key, hashed key and cipher text to recipient and the receiver will first generate the hash value of the key and compared with the digest value sent by the sender. if both the digest value are same, the receiver can conclude that the message is not tampered and it is from reliable sender. Hence we can prove the authenticity of the sender in a communication network. The proposed architecture can be better explained by the Fig:(1) shown below.

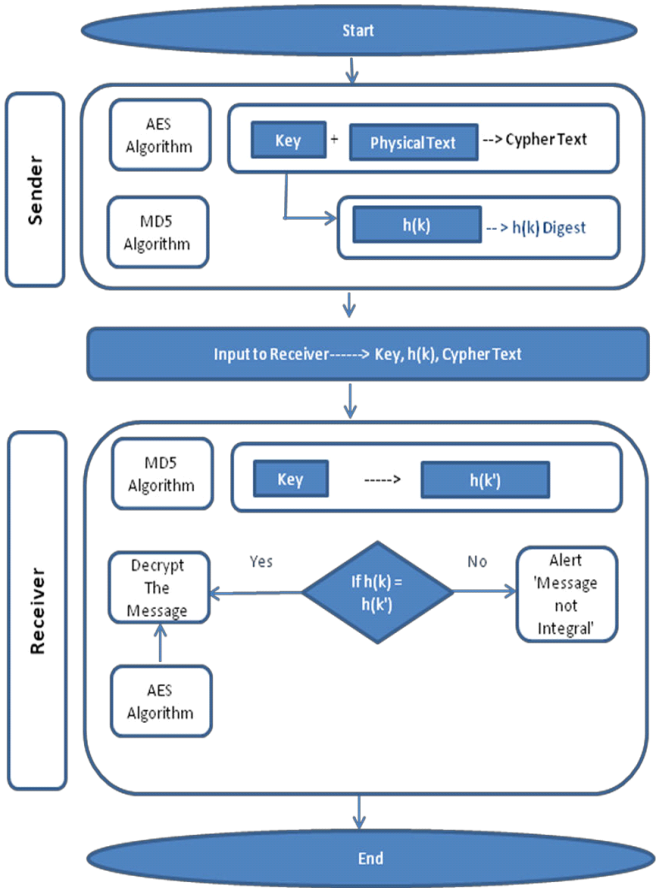


Fig (1): Proposed Hybrid Encryption Scheme

Incase CPU is used in isolation for this encryption/decryption, there will be an overhead in order to encrypt and decrypt data. In order to reduce this overhead due to encryption and decryption we can use GPU. Here we can offload the heavy computations i.e., encryption and decryption to GPU, and CPU can be utilized for some other operations.

IV. IMPLEMENTATION DETAILS

The algorithms used in this proposed architecture i.e., AES and MD5 require several steps. In case CPU is used in isolation for this hybrid algorithm, an additional overhead will be there for data manipulation, in order to encrypt and decrypt. Offloading these operations to GPU is used to reduce this overhead, and CPU time will be utilized for some other operations.

There are mainly three modes of implementation for the hybrid algorithm. They are as follows

1. CPU Mode: Serial version of proposed hybrid algorithm implemented using C. Both key digest generation and encryption on CPU.
2. GPU Mode: Parallel version of proposed hybrid algorithm implemented using CUDA. Both key digest generation and encryption on GPU.
3. CPU-GPU Mode: Key digest generation on CPU and encryption on GPU.

The proposed hybrid algorithm is successfully implemented in CUDA and the details are listed below step by step.

STEP 1: Declare variables and allocate memory on GPU

STEP 2: Copy data from host to device

STEP 3: Calling kernel

In AES algorithm the kernel variables are called as follows:

Retrieve the maximum number of threads that can be invoked in the graphics card. If (input size < maximum number of threads), then we take them to be half that size, but if it was greater we give the maximum value.

Number of Blocks =Data size in bytes/Number of threads.

Blocks = (Data Size/16*Number of thread)
In MD5 algorithm the kernel variables are called as follows: Number of threads=64;
Number of blocks= (Data size/Number of thread) STEP 4: Decide the granularity as 16 bytes per thread for AES and 1 byte per thread for MD5.
STEP 5: Declare the input and output variables stored in shared memory. We have to declare the tables for S-boxes and RCon matrices and also expanded keys which are stored in constant memory.
STEP 6: Add offset variable in the encryption function. Index = blockIdx.x * blockDim.x + threadIdx.x
STEP 7: Finally after encryption copy data from device to host.

V. RESULTS AND ANALYSIS

The experiments performed in order to prove the performance and efficiency of hybrid algorithm on multi-core machines. It is clearly understood from the obtained results that hybrid encryption / decryption can be accelerated on multi-core machines if parallel implementation is used. NVIDIA CUDA framework is used to implement the proposed algorithm in parallel and accelerate the performance. However, it is obvious that the performance will be increased significantly in the presence of more number of processors/threads.

For evaluation of results and testing, a test environment, design of three testing modes and different threads per block allocation are presented. Text files of different sizes are used for the test; 1KB, 2KB, 10KB, 512KB, 1MB, 10MB, 50MB, 100MB, and 1GB.

Three different implementations of the hybrid algorithms use the same set of files and same keys.

At first case, a hybrid mechanism running in CPU only mode (only use CPU for key generation and encryption) is designed. This is the serial implementation of the proposed work. And at the second test case, CPU-GPU mode (use CPU for key generation and key digest generation, and GPU for encryption and decryption) is used. Here MD5 algorithm is in serial mode and AES algorithm is parallelized to run on GPU. At last GPU mode (only using GPU for key generation and key digest generation and encryption) of hybrid mechanism is used. Comparison is done between the three test cases and speed up is calculated. The test result will be showed in this section in TABLE I.

Table I: Performance comparison of three modes of implementation of hybrid algorithm.

File Size	CPU Hybrid Time(m s)	# of blocks	# of thre ads	GPU Hybrid Time(m s)	CPU-GPU Mode Hybr id time(ms)	Speed Up
1KB	0.0728	16	64	0.0706	0.0614	1.18
2KB	0.192	32	64	0.15	0.11	1.75
10KB	0.87	160	64	0.42	0.38	2.28
512K B	4.56	8192	64	0.87	0.80	5.70
1MB	12.72	16384	64	2.4	1.87	6.80
10MB	50.138	163840	64	9.46	7.32	6.85
50MB	206.11	819200	64	38.67	30.11	6.85
100M B	381.37	1638400	64	71.02	55.43	6.88
1GB	3127.6	16777216	64	578.13	434.38	7.2

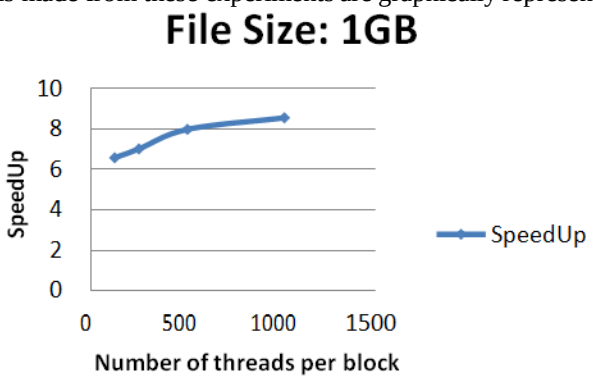
When designing high performance systems, scalability is of great importance. This is particularly important in the case of cryptographic applications as use cases may vary from encrypting small files to encrypting bulk files. While measuring scalability, we must consider the different number of threads available per block in an existing GPU with increase in file size.

The next experiment is based on allocating different number of thread per block. The maximum numbers of available CUDA threads are 1024 and if we are utilizing the maximum available threads per block the performance is increasing. The test results of two larger files with different number of thread per block are summarized in the TABLE II.

Table II: Performance comparison of larger text files with different number threads allocation.

File Size	CPU Hybrid Time (ms)	# of blocks	# of threads	GPU Hybrid Time (ms)	CPU-GPU Mode Hybrid Time (ms)	SpeedU p
100MB	381.37	819200	128	63.24	55.11	6.92
		409600	256	57.10	53.41	7.14
		204800	512	54.95	47.91	7.96
		102400	1024	48.56	45.29	8.42
1GB	3127.6	8388608	128	501.66	476.23	6.57
		4194304	256	488.26	446.23	7.01
		2097152	512	455.64	391.92	7.98
		1048576	1024	416.88	365.80	**8.55**

The performance comparisons made from these experiments are graphically represented as:



VI. CONCLUSION AND FUTURE WORK

A hybrid cryptographic security algorithm is implemented successfully in GPU using NVIDIA CUDA. The proposed approach improves the reliability and ensures the integrity and authenticity of the sender. The parallelization achieved a speed up of 8.5X approximately.

The proposed work has can be applied to Internet Banking so as to increase the security and protect the system from SQL attacks. It can also be applied in smart device security to ensure authenticity and integrity.

Future scope of work may include performing audio or video file encryption using same hybrid algorithm. Further, new hash algorithm such as KECCAK can be incorporated instead of MD5 to enhance security features. Also, the use of different available graphics card can be explored to analyse the difference in performance. Further, the energy efficiency (i.e., by calculating the amount of power dissipation) of the proposed parallelized hybrid algorithm can also be studied in depth as part of future work.

REFERENCES

[1] Deguang Le,Jinyi Chang,Xingdou Gou,Ankang Zhang, Conglan Lu "Parallel AES algorithm for fast Data Encryption on GPU", in 2nd International Conference on Computer Engineering and Technology,IEEE, 16-18 April 2010,pp:V6-1-V6-6.

[2] D.L.Cook, J.Ionnidis, A.D.Keromytis,and J.Luck.CryptoGraphics:"Secret Key Cryptography Using graphics Cards".In Proceedings of Cryptographer's Track at the RSA Conference,San Francisco, USA, Feb 2005. pp.334-350.

[3] Semente, R.S,Salazar,A.O,Oliveira,F.D.M;"CRYSEED:An automatic 8-bit cryptographic algorithm developed with genetic programming", In Proceedings of IEEE international conference I2MTC May 2014.pp.1065- 1068.

[4] Sliman Arrag, Abdellatif Hamdoun, Abderrahim tragha and Salah eddine Khamlich; "Replace AES Key Expansion Algorithm By Modified Genetic Algorithm", Applied Mathematical Sciences, Vol.7, 2013, no.144,7161-7171.

[5] Xingliang Wang, Xiachao Li,Mei Zou,Jun Zhou; "AES finalists implementation for GPU and multi-core CPU based on OpenCL", IEEE International Conference on ASID, June 2011, pp.38-42.

[6] Putri Ratna A.A,Dewi Purnamasari P, Shaugi A,Salman M; "Analysis and comparison of MD5 and SHA-1 algorithm implementation in simple-O authentication based security system", IEEE International Conference on Quality in Research,June 2013.pp 99-104.

[7] http://www.nvidia.in/object/gpu-computing-in.html

[8] https://en.wikipedia.org/wiki/**CUDA**

