International Journal of Applied Engineering Research
ISSN 0973-4562 Volume 10, Number 11 (2015) pp. 27943-27950

http://www.ripublication.com

Resolving LPN Problem In Vector Processing Architectures

N. Hari Ram Kumar
Department of ADC, School of Computing, SASTRA University,Thanjavur, India.
E-mail: hnadaraj@live.com

Abstract

In this project, an algorithm to solve the problem of Learning Parities with Noise (LPN Problem), which is one of the predominant hard problems used for designing cryptographic Systems, has been resolved in Vector Processing Architectures. Several algorithms have been proposed to solve this LPN problem. Recent advancements solve this problem in a cluster architecture with more than 32 cores of Intel processors. The speed up achieved has been improved from the previous methods. However, in the system proposed in this project, we aim to solve the problem in vector processing architectures like NVidia Tesla. They make the use of the Graphic Processing Unit (GPU) of the systems to perform the computations. The use of GPU for the general purpose computing makes it possible to solve any hard problem at amazingly shorter time. With such processing model, peak performance has been achieved while executing the brute force LPN algorithm using Cuda implementation.

Keywords: LPN, Parity Problem, Cryptography, RFID, GPU, CUDA

Introduction

Now-a-days cryptographic solutions are embedded in economic devices like RFID tags. There are lot of cryptographic primitives to inject security in those applications. The main goal of cryptography is to develop low level cryptographic primitives with high intractability features which are really hard to break the security of the modelled system. Learning Parities with Noise is one such protocols with very limited computing power. The AES standard, which is known for its relative safety, is a conventional technique in these cryptographic systems. However, it is really expensive to implement AES. The LPN can be used as a good alternative as its intractability is proven.

Several cryptographic protocols uses LPN Problem in their security system. Most of the algorithms that solve the LPN problem require very large amount of memory and space [1]. But the practical applicability of the LPN problem is in memory constrained environments like the RFID tags. In such cases Brute force scheme is the only attack that can be applied in order to perform crypt analysis on the system. Lot of existing systems, solve this LPN problem in cluster environments, using combinatoric

objects. The parallel version using combinatoric objects will cause deviated results if the inputs provided are unordered [2]. Hence, dividing the task into smaller sub tasks and executing each subtasks in faster processing architectures like NVidia Tesla will help to increase the efficiency of the algorithm.

This project concentrates on resolving the brute force LPN algorithm in processors that supports NVidia Graphic Processing Unit [3]. The implementation part is being done by using CUDA by NVidia. This report will be provided with a detailed description on the LPN problem and the technologies used. The results based on the execution is given ultimately and the system is concluded.

Learning Parities In The Presence of Noise Problem

The concept of Parity Learning in the presence of errors and its intractability feature are being explained in the section below.

The LPN Problem

Let us consider a pxq matrix A, where p is a polynomial of size q. There is also a p bit vector z and a noise parameter $\boldsymbol{\eta \in (0, 1/2)}$. Our goal is to find a q bit vector x such that $\boldsymbol{|Ax\text{-}z| \leq \eta p}$ in a scenario, where there is a sender and a receiver, let us consider that a common secret key k of n bits is shared between the sender and receiver. The first stage of the secure communication starts with the sender sending a random challenge c which belongs in $\boldsymbol{\{0, 1\}^n}$to the receiver [4]. The Parity bit, let us say z, can be calculated by both the sender and the receiver as **c.k (mod2)**. The receiver will send this calculated parity bit to the sender. The sender will accept the receiver if and only if the parity bit of the receiver matches with the senders own calculated parity bit. The Parity bit in these cases will be either zero or one which is very easy for an attacker outside the system to guess the parity bit though there is a probability of 1/2(which can be represented as 2^{-1}). If the sender sends n challenges to the receiver, then the probability of the likely event of guessing the parity bit will changes to 2^{-n}. If an eavesdropper is known of all the challenge and responses between the sender and the receiver, it is very easy to regenerate the secret key k used by them through the Gaussian Elimination method [5]. In order to avoid such scenario, a noise parameter is introduced into the system. The receiver can deliberately introduce some error in the response with the probability of η. The receiver will be authenticated if the total number of correct responses is greater than or equals the value $\boldsymbol{(1-\eta) * n}$.

Cryptography With The LPN Problem

The modern cryptography uses light weight cryptographic protocols in order tobuild the security of the system. The security of the system depends on theintractability property of the protocols used. The LPN is one such hard problem [7]. The intractability of the LPN problem is proven. Hence, LPN can be used as anefficient protocol to build a cryptosystem.

The Cryptography possible with this LPN problem includes

- MAC
- Digital Signatures
- Public Key Encryption

The Parameters in the LPN problem

The algorithm that solves the LPN problem will have the following parameters

- Dimension
- Noise Rate
- Number of Samples

The Number of Samples Parameter

The last parameter mentioned here is the Number of Samples. This particular parameter is a crucial one because, in several other approaches that solve the LPN problem require a very large number of samples which creates a space problem. In case the samples are reduced in number, the unlikely event of increased time of execution will occur. Hence, should there be any system that solves the LPN problem efficiently, then that should use large number of samples and execute very faster.

GPU Computing

The traditional CPU executes only one instruction at a time. Later, several researches, like introducing multi cores, have been conducted in order to improve the speed at which the CPU executes. Recently, the Graphic Processing Unit, which is used for accelerating the video in a system is being looked as a separate execution unit like the CPU. The General Purpose Computing on Graphic Processing Unit (GPGPU) became more popular and a lot of applications have been accelerated by using this concept [8].

Not every operation can be performed by the GPU. GPU cannot stand alone for processing. CPU intensive operations are passed on to the GPU for execution. The difference between the execution of the CPU and the GPU can be well understood from the figure 1.

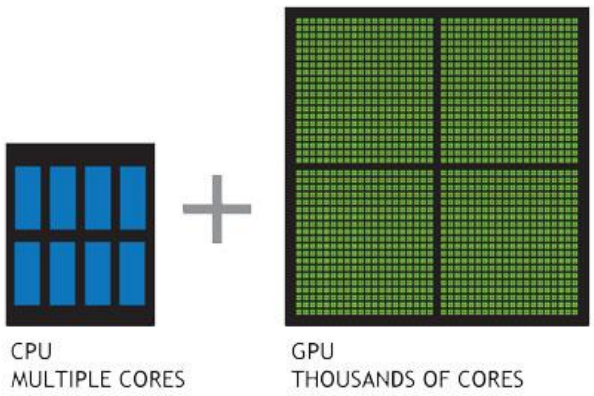

Figure 1: CPU vs GPU

The GPU will have an enormous amount of coresto exploit massively parallel operations. The Programming Language for developing GU applications are CUDA, OPENGL, etc. CUDA was developed by NVidia themselves.

In GPU computing, the execution is similar to SIMD format. For our convenience, let us consider the addition of two thousand numbers. The CPU execution will be sequential. The time required is huge. The parallel version with 100 cores can perform this by the master sending the array of 20 numbers to each core for the addition and the results are collected towards the end of execution. Thus, the time required is reduced to just the addition time of twenty numbers plus the communication overhead. Unlikely, in GPU, there will be thousands of cores. Thus the execution time is similar to the addition of one or two numbers. Moreover, the communication time is also not very large.

The GPGPU programming techniques rely on big data sets for efficient performance. The different threads of execution interact through each other by using Off-chip memory. The GPU applications restrict not only video types but data of any kind can be stored. Not only the processing power of the GPU but also the bandwidth of the graphic memory and the bandwidth of the system bus that is connected to the GPU plays an important role in determining the speed at which the data are sent in and out of the GPU device. [9]

The great processing power possessed by the GPU has got noticed by the researchers. They have developed wide variety of models and simulations which the traditional CPU could not handle. This has broken down the fact that the GPU is used only for receiving graphics in the computer. Scientific simulations require nearly hundreds of repeated calculations and modeling which can be provided by the GPGPU.On the darker side, the password generating tools can also exploit the power of GPGPU and generate the passwords using Brute Force method within few seconds.

Related Works

Several researches have been done in solving the LPN problem. Due to the recent advancements in Wireless technologies, the cryptographic devices have become economic and portable. In such cases, the memory involved in the system is very low. As LPN problem is possessed with ease of computation that can be an effective security protocol for those devices [10]. Only brute force search is applicable in such memory constrained environment. Ivan Texido et.al proposed a method in which they parallelized the enhanced algorithm to solve this LPN problem in environments where memory is constrain.

There are several cryptographic lightweight primitives which uses shared key authentication protocols. Those primitives may be like block ciphers or one-way hash functions. Only such protocols help in the deployment of memory constrained resources like smart cards. [11]

In a system proposed by Gilbert, the HB+ authentication scheme allows the protocol to be implemented in RFID tags. The security of the system lies in the assumed intractability of the protocol used.A system is said to be secure if it is built with a hard problem. The intractability property of the system maintains the hardness of the problem LPN is a hard problem as well as it is lightweight to implement.In another system proposed by Heyse, the features of the HB+ authentication protocol is still improved. Lot of enhancements have been proposed in order to improve the LPN

problem for cryptography. A variant of LPN problem, "Ring LPN" was proposed from which a new variant called LAPIN authentication protocol was proposed.

LPN problem was solved by using fast correlation attacks. The time required to solve was sub exponential. Later, several improvisation over the algorithm was developed which took care to reduce the time taken to solve the parity problem. In those cases, that have considered the input samples required as very small in number.

An efficient scheme would be the one which works on large set of inputs and still reduce the time taken to solve. In those cases, memory is surely constrain until Ivan Texido et al proposed a method in which they consider an enhanced algorithm that solves the LPN problem.

In Brute force technology, the input samples are considered one by one. If noise occur, then the input is flipped [12]. This process is continued till the end of the input.

Later, the brute force scheme was enhanced in a manner that the input is not given one by one. The input is split into two subsets based on a probability of the occurrence of the error. Then there will be two sets of input. Those input samples will be arranged in an increasing order of the hamming weight. Then the LPN problem is solved with those input. This method help to achieve the result faster with huge data samples.

The idea proposed by Ivan Texido involves implementing the enhanced LPN algorithm in parallel architecture. The implementation is carried out by using MPI using C. The architecture used for implementation consists of a cluster with 32 Intel cores. We have an access to Oracle which produces samples at random that are distributed probabilistically. The oracles has a lot of samples. One of those is the secret key which we have to find. It is possible for us to query the oracle. On each query, the oracle picks a response and also a noise associated with each sample. [13]

Architecture Diagram

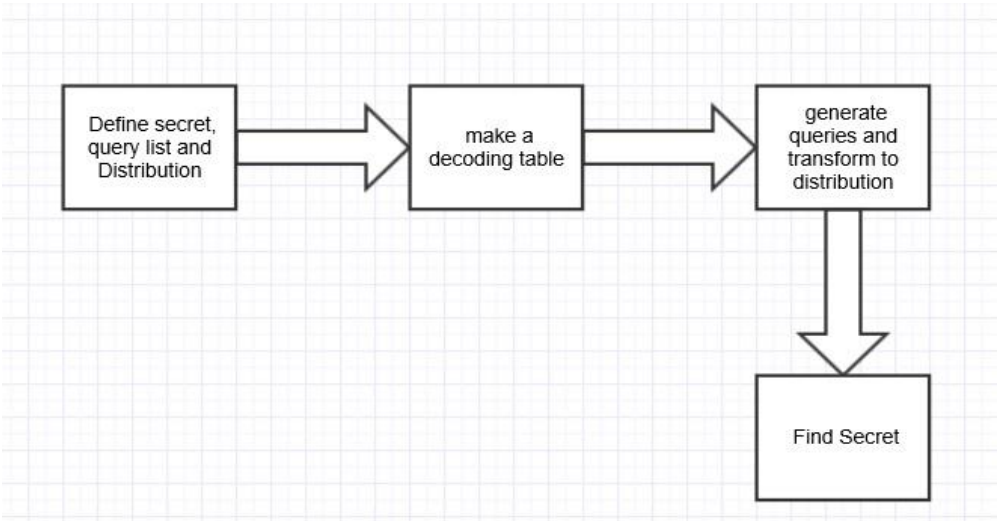

Figure 2: Architecture

The system architecture is shown above in the figure 2. The system starts with defining the list of keys. The input samples should be distributed probabilistically with the values of the noise. The distribution should be given in the system. Then a decoding table is provided in order to filter out the secret key. Here is where the flipping of the input samples will occur. Then queries are passed into the oracle and with the response for each query, the secret key is determined. In case there is no noise involved in the system, then finding the secret key is like decoding a binary linear code with the use of Gaussian Elimination. In whatever cases, the secret key would be a vector.

Proposed System

Now-a-days cryptographic solutions are embedded in economic devices like RFID tags. There are lot of cryptographic primitives to inject security in those applications. The main goal of cryptography is to develop low level cryptographic primitives with high intractability features which are really hard to break the security of the modelled system. Learning Parities with Noise is one such protocols with very limited computing power. The AES standard, which is known for its relative safety, is a conventional technique in these cryptographic systems. However, it is really expensive to implement AES. The LPN can be used as a good alternative as its intractability is proven.

Several cryptographic protocols uses LPN Problem in their security system. Most of the algorithms that solve the LPN problem require very large amount of memory and space. But practical applicability of the LPN problem is in memory constrained environments like RFID tags. In such cases Brute force scheme is the only attack that can be applied in order to perform cryptanalysis on the system. Lot of existing systems, solve this LPN problem in cluster environments, using combinatorial objects. The parallel version using combinatorial objects will cause deviated results if the inputs provided are unordered. Hence, dividing the task into smaller sub tasks and executing each sub task in faster processing architectures like NVidia Tesla will help to increase the efficiency of the algorithm.

The Graphic processing Unit if used for the general purpose computing, then a lot of simulations which seems to be a challenge to the researches can be solved very easily. Multiple processes can be executed at the same time. In a very large architectures, the number of cores at which the kernel function operates will be few thousands. In a GPGPU execution, multiple kernel functions may be execute from the whole architecture. This is however limited to one kernel function per grid, which is actually the group of blocks. Blocks are nothing but the group of threads of execution. Thus, each block and thread will possess a unique id so that they could be used for calling the methods. The particular thread or block has to execute the particular job is achieved by using the id.

The system proposed here consists of the brute force algorithm to solve the LPN in memory constrained environment. The algorithm will take the inputs of the number of samples, the noise parameter and the dimension. When the number of samples is large and the memory is constrained, and also the execution time should be faster, then the

only method which will lead us to success is the Brute Force methodology. The output which we will obtain is the secret key. There will be a lot of samples. The sender and receiver will indulge in a communication. In order to improve the security, the receiver will acknowledge that he has received the secret key to the sender along with several noised results. The noise is included wantonly into the responses. In case, the noise is not present, then any other person, who is in reach of the request and responses between the sender and the receiver, can easily decode the secret key by using Gauss Elimination method. This is similar to solving the linear system of equations. Thus in this algorithm, each sample is considered. It is flipped if the noise is present. Then it will check for the noise rate. If the samples matches with the noise rate, then the secret vector can be produced.

There is an oracle which has the secret key. One can make any number of queries to the oracle in order to get the secret key. In each query, we will send along the noise rate and the other parameters. The samples will be calculated for the noise rate. If the match with the calculated noise rate occurs, then the secret key is revealed. Each input sample is noised at a particular probability to avoid the key to be provided in the hand of the attackers. In case, the input is not error, then it would be very easier for any third party attackers. They, simply with their access to the request response pairs can achieve the secret key. This is more or less very similar to decoding binary linear code with the use of Gaussian Elimination. Thus a decoding table should be created in our system. The parity function is generated and the inputs are assigned with the parity. Also, they are distributed using standard distributions like Poisson. Finally, the Kernel function is called in order to execute the search for the secret key which will be attained in few seconds.

Results and Analysis

Thus the brute force algorithm to solve the LPN problem in vector processingarchitecture has been executed. Experimental results show that the proposedsystem achieves a speed up of about 98% than the linear code.The number of GPU cores with which the system has been run is 32 cores. Thiscode is also tested in 48 cores of GPU. Thus by increasing the number of GPUcores, the time taken to execute can be reduced. This also takes a large numberof input samples unlike the other method.

Conclusion and Future Enhancement

The algorithm to solve the LPN problem can be enhanced in such way that the communication overhead is reduced. The implementation can be carried out in an environment which contains both MPI and CUDA. Thus a massive level of parallelism can be exploited. In MPI methodology, different functions can be executed simultaneously by different processors. So a lot of scenario involves in assigning the tasks to the processor. GPU performs Single Instruction Multiple Data type of parallelism very efficiently. In case, MPI and CUDA are used together,

different functions are sent to different GPU and each GPU performs the function at amazing speed.

Reference

[1] Ivan Texido, MPI based implementation of an enhanced algorithm to solve the LPN problem in memory constrained environments, Parallel Computing 40, 102-112 (2014).

[2] Apple Baum, B., Barak, B., Widgeon Public-key cryptography from divergent assumptions. In: Schulman, L.J. (ed.) 42nd ACM STOC, pp. 171180. ACM Press (2010)

[3] Kopka, H., Daly P.W., Noise-tolerant learning, the parity problem, and the statistical query model, Addison-Wesley, Reading, MA, 1999.

[4] Banerjee, A., Peikert, C., Rosen, A Pseudorandom functions and lattices, Cryptology ePrint Archive, Report 2011/401 (2011), http://eprint.iacr.org/.

[5] Goldreich, O., Goldwasser, S., Micali How to construct random functions, Journal of the ACM 33, 792807 (1986)

[6] Gilbert, H., Robshaw, M., Sibert, H.An active attack against hb+ - a provably secure lightweight authentication protocol, Vol.2, pp.60-74, summer 1998.

[7] Gilbert, H., Robshaw, M.J.B., Seurin, Y How to Encrypt with the LPN Problem, vol. 5126, pp. 679690. Springer, Heidelberg (2008)

[8] Hastad, J., Impagliazzo, R., Levin, L.A., Luby, M., A pseudorandom generator from any one-way function,SIAM Journal on Computing 28(4), 13641396 (1999)

[9] Heyse, S., Kiltz, E., Lyubashevsky, V., Paar, C., Pietrzak, and K.: An efficient authentication protocol based on ring-lpn, manuscript, 2011.

[10] Michael Alekhnovich, "More on Average Case vs Approximation Complexity", Computational Complexity, vol 20, issue 4, pp 755-786, 2011.

[11] Avrim Blum, Merrick L. Furst, Jeffrey C. Jackson, Michael J. Kearns, Yishay Mansour and Steven Rudich, "Weakly learning DNF and characterizing statistical query learning using Fourier analysis", STOC, pp 253-262, 1994.

[12] Frank Thomson Leighton and Michael T. Goodrich, "Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing", STOC, ACM, pp 23-25, 1994.

[13] Nigel P. Smart, "Advances in Cryptology" - EUROCRYPT 2008, 27th Annual Conference on the Theory and Applications of Cryptographic Techniques, Lecture Notes in Computer Science, vol 4965, pp 13-17, 2008.