# Optimized Private Information Retrieval Protocol Using Graphics Processing Unit With Reduced Accessibility

**Dissertation**

submitted in partial fulfillment of the requirements
for the degree of

**Master of Technology, Computer Engineering**

by

**Sandip B. Bansode**
**Roll No: 121022007**

under the guidance of

**Prof. S. B. Mane**

Department of Computer Engineering and Information Technology
College of Engineering, Pune
Pune - 411005.

June 2012

# DEPARTMENT OF COMPUTER ENGINEERING AND INFORMATION TECHNOLOGY, COLLEGE OF ENGINEERING, PUNE

# CERTIFICATE

This is to certify that the dissertation titled

## Optimized Private Information Retrieval Protocol Using Graphics Processing Unit With Reduced Accessibility

has been successfully completed

By

**Sandip B. Bansode**
(121022007)

and is approved for the degree of

**Master of Technology, Computer Engineering**.

**Prof. S. B. Mane,**
Guide,
Department of Computer Engineering
and Information Technology,
College of Engineering, Pune,
Shivaji Nagar, Pune-411005.

**Dr.Jibi Abraham,**
Head,
Department of Computer Engineering
and Information Technology,
College of Engineering, Pune,
Shivaji Nagar, Pune-411005.

Date : _____

# Abstract

Database outsourcing as a service is a new trend emerging in the computing industry instead of managing database in-house. This introduces several security issues related to database. One of the important security requirement is privacy. A Private Information Retrieval protocol (PIR) allows user to retrieve an element from the database in such way that identity of that element is not get revealed by database server. In this project work, we present the optimized lattice based PIR schema. This proposed schema is based on use of different techniques like General Purpose Graphics Processing Unit (GPGPU) and cache like technology to reduce accessibility. The latest proposed PIR scheme deal with user privacy, use of GPGPU but left to focus on data privacy, data confidentiality and accessibility cost. In our approach our system focusing on issues like user privacy, data confidentiality and accessibility cost by using cache and computational overhead by using GPGPU. Thus searching time of proposed PIR protocol is minimized to number of records in the cache. Thus our proposed design improves the overall performance of PIR protocol.

# Acknowledgments

I express my sincere gratitude towards my guide **Prof. S.B. Mane** for his constant help, encouragement and inspiration throughout the project work. Without his invaluable guidance, this work would never have been a successful one. I would also like to thank *Dr.Jibi Abraham* for their valuable suggestions and helpful discussions. Last, but not the least, I would like to thank Prof.Suraj Sawant.

**Sandip B. Bansode**
College of Engineering, Pune

# Contents

**Abstract** **ii**

**Acknowledgments** **iii**

**List of Figures** **vii**

**List of Tables** **vii**

**1 Introduction** **1**

1.1 Overview . . . . . 1
1.2 What is Private Information Retrieval . . . . . 3
1.3 Graphics Processing Unit (GPU) . . . . . 4
1.4 Motivation . . . . . 4
1.5 Problem Definition . . . . . 5
1.6 Objective . . . . . 5

**2 Literature Survey** **6**

**3 Proposed System Design** **8**

3.1 Work Flow . . . . . 9
3.2 Overview of PIR Protocol . . . . . 11
3.2.1 Request Generation . . . . . 11
3.2.2 Answer Encoding . . . . . 12
3.2.3 Information Extraction . . . . . 12

**4 Implementation** **14**

4.1 Basic description of the scheme . . . . . 14
4.2 Reply Generation using GPGPU . . . . . 16
4.3 Key Management Issue . . . . . 17
4.3.1 Necessity of Key Management . . . . . 17
4.3.2 Ways to Achieve Key Management . . . . . 18

4.3.3 Limitations of Our Approach . . . . . 20
4.4 Achieving Data Confidentiality . . . . . 20
4.4.1 Overview of AES Algorithm . . . . . 20
4.4.2 Advantages of AES . . . . . 21
4.5 Package Description . . . . . 21
4.5.1 NTL . . . . . 22
4.5.2 Crypto++ . . . . . 22
4.5.3 CUDA Drivers . . . . . 22
4.6 Specification For Implementation . . . . . 23
4.6.1 Hardware Requirements . . . . . 23
4.6.2 Software Requirements . . . . . 24
4.7 Screen-shot of Implemented System . . . . . 24
4.7.1 Service Provider . . . . . 24
4.7.2 Organization . . . . . 24
4.7.3 Client . . . . . 26

**5 Experiments and Results** **27**

**6 Conclusion and Future Work** **31**
6.1 Conclusion . . . . . 31

**A Practical Approach of PIR** **33**

**B Paper Publications** **36**
B.1 Paper Acceptance notification for ICETCIT 2012 - India . . . . . 36

**Bibliography** **37**

# List of Figures


1.1 An Overview of Service Provider Model . . . . . 2
1.2 PIR Schema . . . . . 3

3.1 System Design . . . . . 9

4.1 Scheme Overview . . . . . 15
4.2 Key Manipulation . . . . . 19
4.3 Key Management Overview . . . . . 19
4.4 Block Matrix Multiplication . . . . . 23
4.5 Service Provider . . . . . 24
4.6 Organization . . . . . 25
4.7 Client . . . . . 26

5.1 Result Comparison of Data Rate v/s No. of Files (Carlos-Aguilar Approach) . . . . . 28
5.2 Result Comparison of Data Rate v/s No. of Files (Proposed System) 29

A.1 Query Generation . . . . . 33
A.2 Reply Generation . . . . . 34
A.3 Reply Extraction . . . . . 35

# List of Tables

5.1 Performance of PIR on CPU . . . . . 27
5.2 Performance of PIR on GPU . . . . . 27
5.3 Comparison with Existing Schema . . . . . 30

# Chapter 1

# Introduction

## 1.1 Overview

Data outsourcing is a new emerging trend for the data management in the computing industry. The data outsourcing is database as a service in which third party database administrator provide a service to the customer as per their requirement. It is an efficient solution instead of traditional in house database management system. Data outsourcing services provide the outsourcing solution across a large range of platforms and technologies at competitive prices. In the outsourced database model, the customer completely rely on the external servers (third party service provider) for the storage, maintenance and extracting of the data. This introduces the several research challenges and security issues related to the database and thus become one of the most active topics in research community. The server at the provider's side is not trusted in this scenario, so the ultimate goal that client want is to use their outsourced database as in house one. In this model the different parameter related to the security such as leak of their sensitive information have to consider. This introduces several additional challenges related to the privacy preserving for client query as well as data which is outsourced. Overall, although security requirements are different between real-world applications, the following requirements are most noteworthy:

**Data confidentiality:** Outsiders and even the server's operators (database administrators) are not able to see the client's outsourced data contents in any case (including when the client's queries are performed on the server).

**User privacy:** Clients do not want the server to know about their queries and the returned results.

**Data privacy:** Clients are not allowed to get more information than what they are querying on the server.

**Authentication and data integrity:** Clients must be ensured that data returned from the un-trusted server has originated from the data owner and has not been tampered with. The above security requirements are different from the traditional database security issues and will in general influence the performance, usability and scalability of the outsourced database model.
There are four main service provider (SP) model based on client-server architecture [?]:

- **UP-DP model:** In this case data owners are the SPs. They charge the client according to their use of data, so the SP provider deals with data privacy (DP) and client deals with user privacy (UP).
- **UP-nDP model:** The scenario of this model is same as UP-DP model only difference is that the data stored is in public in nature. Here client deals with user privacy but SP don't deal with data privacy.

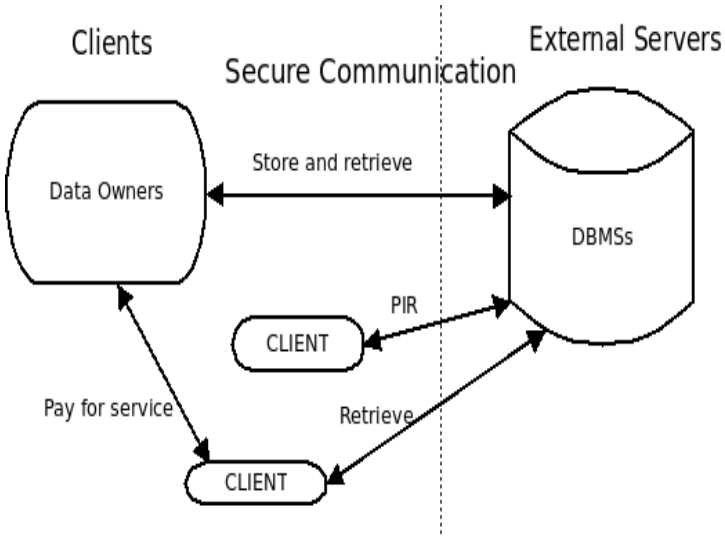


Figure 1.1: An Overview of Service Provider Model

- **DC-UP model:** In this model, the data owner outsources its data to external database server. Here the data owner concerned with data confidentiality (DC) and user privacy (data owner becomes a client).
- **DC-UP-DP model:** In this model data owners outsource their data and charge clients for using their information. This is the most complex model in terms of security issues. The data owner is concerned about both the data confidentiality and data privacy with respect to both the external database

server and its clients. The client, in turn, is concerned about the user privacy with respect to both the data owner and the server.

As per this project work concern our implementation follows DC-UP model. In the DC-UP model, it is necessary the stored data to be encrypted because the data owner can employ traditional encryption techniques to protect their data more efficiently. To satisfy the user privacy requirement, private information retrieval (PIR)-like protocols are employed. Sole purpose of this project work is to implement Private Information Retrieval protocol to ensure user privacy and improve the effectiveness of PIR by considering different security parameters.

## 1.2 What is Private Information Retrieval

A PIR scheme is a protocol in which a client retrieves a tuple out of n from replicated database in such way that retrieved tuple should not get revealed at the service provider side. In this project work,we are focusing on single database PIR which is non-replicated. In PIR protocol, client can retrieve an element of index i from non-replicated database. For this purpose it uses a PIR query generation algorithm with input i and sends resulting query to database. As shown in the figure below [12].

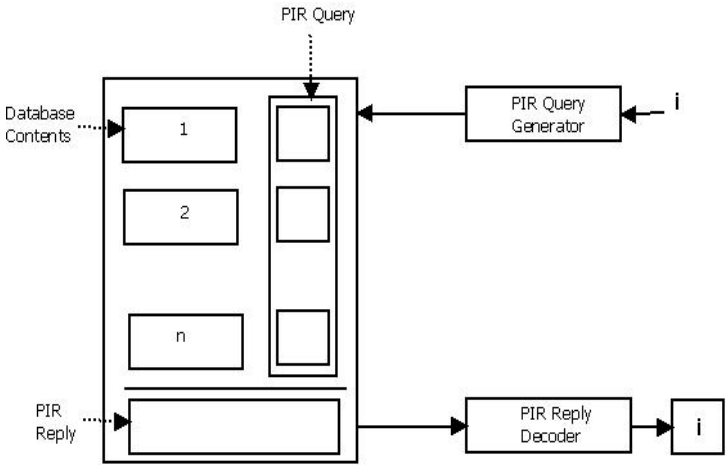


Figure 1.2: PIR Schema

Then it combines its elements to the query which is generated using reply generation algorithm and obtains a result which is send back to the client. Finally client decodes the result using reply decoding algorithm. The protocol is said to be

correct if client get the expected element and said to be private if the information of that record is not get revealed at the service provider side.

## 1.3 Graphics Processing Unit (GPU)

GPUs are useful at manipulating computer graphics. They are most effective while dealing with the large blocks of data which have to process in parallel instead of general purpose CPUs. They are used for general purpose scientific and engineering computing. GPU run faster than multi-core CPUs. The details of the GPU can be found at [15][19].

## 1.4 Motivation

In order to retrieve an element from a database server, user send a request in the form of query pointing out which element the user wants. The database server sends back the requested element. The user would like to keep the information private that he retrieved from a database server. A Private Information Retrieval (PIR) is a protocol, in which user retrieves a record privately out of n records from the database server. The database may be replicated or non-replicated. In this case we are considering the only single database PIR. From this non-replicated single database we have to retrieve a record privately in such way that record shouldn't get known to the database server. The database includes such as electronic book, media databases, and bioinformatics etc which are required to keep confidential. For this a trivial solution is the downloading the whole database to the client side and retrieving a record that user is interested in, but this is not the feasible solution for large data set. It is a big challenge to achieve the efficient PIR protocol that give the surety of good response time and covers the maximum security issues in the real life. To reduce the communication and computational cost a number of researches have been performed in the area of PIR. Chor et al.[1] suggest basic assumptions for PIR. First assumption is, PIR is not possible without downloading the entire data set to the client side. Second assumption is entire database must be read.

In this scenario the time required to compute the record is equal to searching time on entire database. User privacy is also getting violated if one of the records is not read by client while processing. During last two decades, there is a number of research have done in the privacy protection in the single database PIR. Single database PIR scheme is not computationally efficient as they have to process

each and every record in the database.Our proposed model is based on the recent proposal Lattice Based approach [2]. It introduces the noise based protocols. The cost per bit in data set is vector addition. It is much cheaper than modular multiplication used in number theory scheme.

## 1.5 Problem Definition

To improve the performance of PIR and to overcome the existing problem, we address the new technology that is use of General Purpose Graphics Processing Units (GPGPU). This is used on the server side in our proposed model. Also we are focusing to reduce the accessibility of our protocol if the same record is requested by the client again. The user privacy is the key point in the PIR protocol. In our approach we are also focusing on the confidentiality, data privacy, computational overhead and reduced accessibility cost. Finally we are trying to achieve the efficient PIR protocol in terms of computational cost, accessibility cost with different security parameters.

## 1.6 Objective

To achieve the efficient and improved performance of Private Information Retrieval(PIR) Protocol with the help of different parallelizing techniques like use of General Purpose Graphics Processing Unit and improve the PIR protocol in terms of computational cost, accessibility cost and considering different security issues like data confidentiality, user privacy.

# Chapter 2

# Literature Survey

In the private information retrieval protocol the database may be replicated or non-replicated. In this scenario we are focusing on the single database which is not replicated. An especially interesting sub-field of research in PIR protocols, called single-database PIR [25], deals with the schemes that allow a user to retrieve privately an element of a non-replicated database. A number of researches have done in case of replicated database. In case of non replicated single database Kushilevitz and Ostrovsky [4] stated that there is no need of replication of data to solve the computational PIR while in the earlier research papers, the problem of computational PIR is solved by the replication of data and they also showed the efficiency in terms of communication. They used same attacker model that was developed by Chor and Gilboa [5].

Julien Stern presented a generic construction in 1998 [6], in his paper he generated a new idea of zero knowledge protocol. This scheme is building block of development of other schemes. This was the first protocol with acceptable communication performance. After that Cachin et al.[7] proposed a scheme of single database PIR with polylogarithmic communication complexity. This construction is based on the $\phi-$ hiding assumption and also stated that query size is not dependent on the total number of elements in the database.

Gentry and Ramzan proposed a scheme [8] in 2005. This scheme is similar to Lipmaas Scheme [9] which is good in communication point of view. Gentry et al. try to improve the communication complexity. For retrieving a larger block of data from database, this scheme achieve constant rate. A study of various problems related to PIR and further research in single database is done by Yan-Cheng Chang [10]. He proposed a scheme having logarithmic communication complexity on the server side. This scheme is also used for implementing 1 out of N, l-bit string oblivious transfer.

Now the current single database scheme shows the improvement in the computational complexity. The new scheme [11] uses the general purpose graphics processing unit (GPGPU). We are able to develop a fast PIR scheme that can be used for many applications. Carlos Anguila Melchor [12] showed the improve performance for linear algebra scheme which are different for CPU(central processing unit) and GPU implementation. Carlos Aguilar and Philipee Gaborit in 2007 [13] presented an overview, usability and the trends that are used till now in single database private information retrieval protocol. In this paper they have presented the different approaches to obtain the PIR and comparison between them. The computational performance is improved by two orders of magnitude when compared to the existing approaches, which is shown by Carlos Anguilar et al. [2] and also showed that practical usability of PIR is not based on communication performance. Also present the comparison between Lipmaas schema and Gentry Schema in terms of computation.

The burning problem today is communication and computational complexity along with different security issues in the PIR protocol. In our approach we are using the GPGPU (general purpose graphics processing unit) and cache like technology to reduce accessibility of protocol so that we can overcome the existing problem of PIR.

# Chapter 3

# Proposed System Design

Our proposed model is based on lattice based PIR scheme [2]. The PIR scheme we propose is based on concept of controlled noise addition, database cache like technology, encrypted data and use of GPGPU. This proposed design which basically shows the extension of High-speed single database PIR [12]. In this system the database is treated as a set of files which are stored on the external service provider are publicly accessible. This system design successfully achieve the user privacy and achieve less computational overhead by using the concept General Purpose computing on Graphics Processing Unit (GPGPU). In our system design we can achieve the data privacy, user privacy, data confidentiality and the different security parameters with the less accessibility cost (in case where same record is requested again by the client) and computational overhead with the help of database caching like technology and use of GPGPU respectively. The proposed design consists of three components: client, service provider and data owner. The data owner who outsources its data to the external service provider is responsible for data management, data storage. The client is able to access the data from the external service provider. The detail scenario can be easily understood by observing the proposed system architecture. The proposed design is arranged in to three parts and the working flow of our design is explained below:

**Client:** The clients are actually the data users, who read, download and access the data as per their privileges. The client may be a database authorized employee in the organization.

**Service Provider:** In the scenario of database outsourcing the external service provider is not trusted so that we have to keep our data on service provider in such way that, the private information should not get revealed by the service provider

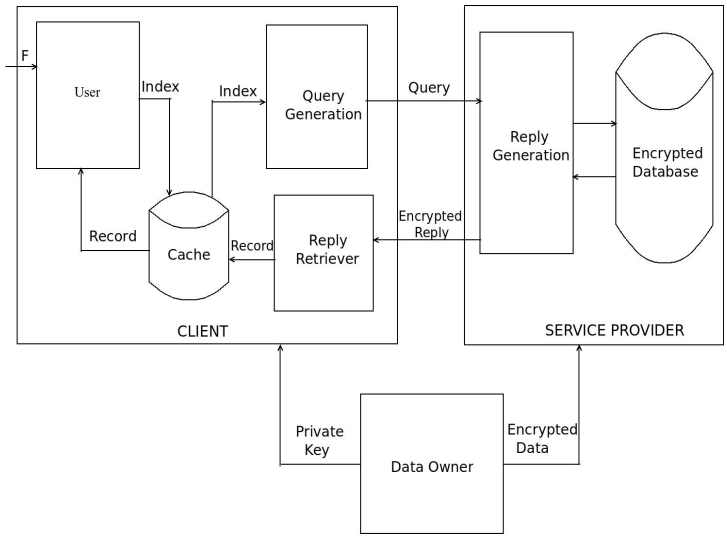


Figure 3.1: System Design

itself.

**Data Owner:** The organization is the database owner. The data owner outsource its data to the third party (service provider). For this purpose organization has to take care of its data. So that organization has to consider all the security issues, in order not reveal the information by the service provider.

## 3.1 Work Flow

The following are the steps showing working flow of our proposed system.

- The user (client) gets the list of file names with their indexes which are stored on the third party service provider. Here F which is input to the user represents the name of file list with the index. Our database consists of normal set of files whose contents are in encrypted format. This database being outsourced by the data owner to the third party service provider.
- After receiving the name of files, the user select one of the index that he is interested in. Then the user (client) tries to check whether the record that he is interested in is present in the cache (that is received folder on the client side which contains the records which are retrieved from the server) or not.
- If the record is present in the cache[3] (received folder) then he retrieves that

record from it. Hence user needs not to move to the main database which is present to the server side and reduces the communication time.

As in the PIR protocol, to maintain the user privacy the user has to visit each and every record and if he fails to do that then server who is not trusted one will try to find out that user is not interested in those record. So no user privacy will preserve.

- If the record is not found in the cache (received folder) then it will send that index of the record to the query generation module.
- Then query generation module creates a query which is represented in the matrix format. The detail description of the query generation matrix is given by Carlos in his paper [2].
- Once the query is generated, the query is sent to the database service provider. To obtain a reply, the database inserts each and every element into the matrix which is obtained by query generation with the multiplicative operation and sums all the rows. The details of the reply generation algorithm given by Carlos et al. [2]. In this case, as the database is in encrypted form so we get the reply generation which is in encrypted format.
- After that, this reply is sent to the response retriever which is present on the client side. The response retriever extracts the information with the help of information extraction algorithm given in Lattice based scheme [2].
- The extracted information treated on the decryption process. In this scenario the algorithm used for encryption and decryption like AES [14]. They are symmetric in nature; we have to send the private key to the client in a secure manner.
- After the decoding and the decryption process we get the actual record in which the user is interested. The entry of that record is stored in the cache first and then finally that record is sent to the user.

In this way user get the record that he is interested in by considering the security parameters like user privacy, data privacy, confidentiality with minimum accessibility cost and computational overhead.

## 3.2 Overview of PIR Protocol

Basically the proposed system design based on lattice based PIR scheme, using NTRU-like approach. The basic idea of our PIR protocol start from secret random matrix M of [N,2N] of the rank N over the field of prime number. Different set matrices are obtained by multiplying the matrix M with different invertible random matrices. Once the matrices are generated, they are disturbed in a such way that, we obtain the softly disturbed matrices(SDMs) and hardly disturbed matrices(HDMs) by the introduction of noise. In this scenario the client send the request to the server for a particular element from the database. Client actually send the set of different matrices that consist of different SDMs and one HDM. The database insert each of its element in the corresponding matrix with a multiplicative operation, and sum all the rows of resulting matrices to obtain database reply. In this case, the reply vector contain the noise, for that client is able to understand unmodified columns of matrices, so that client able to remove the noise and retrieve the element that he/she wants from the database.
Our proposed design of PIR protocol is based on the three algorithm [2]

**1) Request Generation**

**2) Answer Encoding**

**3) Information Extraction**

### 3.2.1 Request Generation

The scheme has three global integer parameters: 2N, the dimension of the lattice and special parameters p and q. The database is described as a set of n elements, and we note $i_0$ the index of the database element the user is interested in. To obtain a PIR request, the user will follow:

- Note $l_0 = \lceil \log(\text{n} \times \text{N}) \rceil + 1$ and set q as $2^{2l_0}$ and p as a prime larger than $2^{3l_0}$.
- Generate A and B, two random matrices over Z/pZ such that A is invertible, and note M = $[A|B]$.
- For each $i \epsilon \{1.....N\}$ compute a matrix $M_i''$=$[A_i|B_i]$ by multiplying M by random invertible matrix $P_i$.

- Generate the random scrambling matrix as a N × N random diagonal matrix over Z/pZ.
- For each $i\epsilon\,\{1.....N\}\,\backslash i_0$ generate the soft noise matrix $D_i$, a N×N random matrix over$\{-1,1\}$ and compute the soft disturb matrix $M_i''$=$[A_i|B_i+D_i\Delta]$
- Generate $D_{i_0}$ , the hard noise matrix, by:
  – generating a soft noise matrix,
  – multiplying each diagonal term by q.
- Compute the hard disturbed matrix $M_{i_0}'$=$[A_{i_0}|B_{i_0}+D_{i_0}\Delta]$.
- Choose the random permutation columns $\rho(.)$ and compute $M_i$=$\rho(M_i')$ for $i\epsilon\,\{1.....N\}$.
- Send the ordered set $\{M_1....M_n\}$ to the database.

### 3.2.2 Answer Encoding

To answer to the PIR reply the database follows. The result is a vector V of dimension 2N over Z/pZ. We will suppose that each element has exactly the number of bits that can be encoded into a matrix ($l_0 \times N$ bits).

- Split each database element $m_i$ in N $l_0$ -bit integers $\{m_i1....m_iN\}$
- For each $i\epsilon\,\{1.....n\}$ construct the vector $v_i = \sum_{j=1}^{N} m_{ij}M_{ij}$ where $M_{ij}$ denotes the j-th row of $M_i$.
- Return $V = \sum_{j=1}^{N} v_i$

### 3.2.3 Information Extraction

To extract the information from the database reply, the user(client) will operate in two phases. First phase is to recover the noise included in the vector and the second is to unscramble and filter out this noise to extract the information.

- Compute the non-permuted noisy vector $V' = \rho^{-1}(V)$.
- Retrieve $E = V'_D - V'_U A^{-1}B$, the scrambled noise, $V'_U$ and $V'_D$ being resp. the undisturbed and disturbed halves $V'$.
- Compute the unscrambled noise $E' = E\Delta^{-1}$.

- For each $e'_j$ in $E' = [e'_1...e'_n]$, compute $e''_j = e'_j - \epsilon$ with $\epsilon := e'_j$ mod q if $e'_j$ mod q÷ 2 and $\epsilon := e'_j$ mod q - q else.
- For each $j\epsilon\{1.....n\}$, compute $m_{i_{0_j}} = e''_j q^{-1}$.

To recover the noise, the reverse permutation of column has to be done. Then it will use the N first coordinates of vector and the matrix M to obtain last N coordinates(which are disturbed). That last N coordinates should be free of noise. Then it will use the same values to extract noise which are inserted in these coordinates.This noise is combination of set of soft and hard noise. To eliminate the noise, firstly we have to eliminate scrambling noise matrix, then filter out the soft noise and then to obtain the database sub-elements of $m_{i_0}$ divide each coordinate by the factor of hard noise.

The above three algorithm shows the basic description of PIR protocol. To achieve the improved performance of this basic PIR protocol, the fourth stage of implementation that is added in this protocol is the parallel operations. This parallel operation is done on the matrix multiplication on the server side to reduce the computational overhead which was observed in the basic PIR operation.

In this protocol we have to visit each and every record in the database to achieve the user privacy. If one of the record is left to visit by the client, then database server will try to find out the record that is being retrieved by the user. Suppose if client request a particular record more number of times from the database server then for each time client has to visit each and every record for each iteration. In this case the communication cost (between client and server) and accessibility cost is more and probably same for each iteration. Hence to reduce accessibility cost, our system design deals with the cache like structure on the client side. It keeps the copy of the files that is being visited(retrieved) by the client, so if client request particular record again then firstly it checks the entry of the same record in cache is present or not. If that record is present then it sends the message to the client otherwise that file request has to be sent to the request generation module which is present on the client side.

# Chapter 4

# Implementation

The implementation of single-database PIR schema we proposed can run both in a CPU and GPU too. It uses CUDA (compute unified device architecture), nVIDIAs library for general purpose computing on graphics processing unit. The results shows that the computation time required for our protocol in a CPU is more than the computation time required in a GPU. The lattice based PIR methodologies allow to process database contents faster than previous implementation by the several orders of magnitude, specially when use of parallelized GPUs (graphics processing unit).

## 4.1 Basic description of the scheme

In the above chapter of System Design we have discuss with the three algorithms which are necessary to gain lattice based PIR protocol. In this section we give an overview of PIR scheme. The bottleneck in the PIR scheme is at reply generation not in the query generation and reply extraction. So in this section we are discussing the reply generation only. The use of GPGPU is on the reply generation phase. In this PIR scheme the implementation of query generation and reply extraction is straight forward as discussed in the chapter of system design. The formal descriptions of three phases are available in [2].The database at the service provider side is described as set of n elements. Each element $a_i$ (for $i\epsilon\{1.....n\}$) is divided into $l_0$-bit sub-elements and represented as a matrix $A_i$.

$$A(i) = \begin{pmatrix} a_{i,1,1} & .. & a_{i,1,N} \\ : & : & : \\ a_{i,L,1} & .. & a_{i,L,N} \end{pmatrix}$$

Here in implementation N and $l_0$ are two security parameters (they are set to 48 and 16). $S_{max}$ is the size of largest element in database and L is parameter that is set to $L = [Smax/(N \times l_0)]$. If the database contain an element which is smaller than $L \times N \times l_0$ then end of matrix is filled up with the standard padding techniques. In our implementation we use the zero padding techniques at end of matrix in such situations.

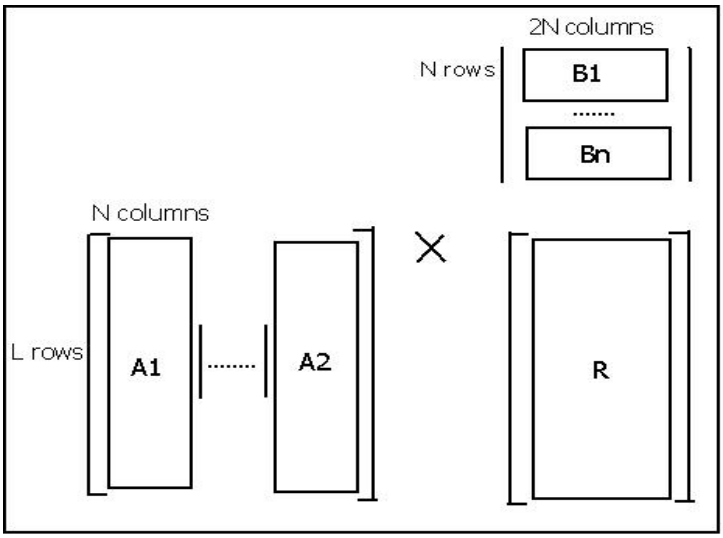


Figure 4.1: Scheme Overview

To retrieve an element from the database server, user generates a query which is formed of n matrices (B1,..Bn) one for each database element. Each matrix is of dimension $N \times 2N$ with scalars in Z/pZ and p is $3l_0$-bit prime. If the user wants to retrieve an element $a_{i_0}$ from database, user generates a query in such way that the $i_0$th matrix has special property , so because of that property that matrix is not get identified as special to the service provider. This property ensures that the user is able to extract the desired element from the server reply. The user sends the query to the service provider which hosts the database of organization. To generate a reply, the server multiplies the column concatenation of the database element matrices and row concatenation of query matrices as shown in above figure. The resulting matrix that is R is a $L \times 2N$ matrix with $3l_0$ bit scalars and is six times larger than the matrix of database element. Here the query size is $n \times N \times 2N \times 3l_0$ which for given parameters it is 28n Kbytes.

## 4.2 Reply Generation using GPGPU

The GPGPU is the general purpose graphics processing unit.It is specialized electronic circuit which is designed for rapidly manipulate and change memory in such way so as to accelerate the computational process. GPU acts as a co-processor. Basically it is used to accelerate the performance of CPU for general purpose scientific and engineering computing. CPU - consist of 4-8 cores and GPU consists of hundreds of small cores. In our scheme the bottleneck point is the reply generation. To overcome this bottleneck we are focusing on the use of graphics processing unit which help the CPU to perform that task faster. GPGPGU applications make use of graphic processing units (GPU) as massively parallel processor. Considering both the cases of speed and bandwidth due to their parallel architecture, use of GPU is an attractive choice for complex computational task than traditional CPUs.

In our approach we specially deal with use of GPU for multiplication process which is complex and time consuming in PIR scheme. For this purpose we specially focus on use of CUDA (Compute Unified Device Architecture) is parallel computing architecture. It is C programming language on GPUs. It requires no knowledge of graphics APIs or GPU programming. CUDA and the nvcc compiler enable the user to write programs in C which will run on an ordinary host computer (Windows, Linux, etc...) with the additional capability of running SIMT code on an attached CUDA-enabled GPU device. In CUDA the terms host and device refer to the ordinary computer and graphic card respectively. The term SIMT means single instruction, multiple thread and refers to a model wherein many threads running in parallel all execute the same instruction at the same time, but on data specific to that thread. We refer the reader to the NVIDIA CUDA C Programming Guide [18]for a detailed overview of the CUDA programming model and hardware. CUDA devices run many threads simultaneously. Blocks of threads may be one-, two- or three-dimensional, as the programmer prefers. Since matrices are two-dimensional, we will use two-dimensional blocks. Set of different blocks forming a grid. A kernel that allows host code to offload matrix multiplication to the GPU.

```
__global__ void MatMulKernel(Matrix A, Matrix B, Matrix C) {
// Each thread computes one element of C
// by accumulating results into Cvalue
float Cvalue = 0;
int row = blockIdx.y * blockDim.y + threadIdx.y;
int col = blockIdx.x * blockDim.x + threadIdx.x;
if(row > A.height || col > B.width) return;
```

```
for (int e = 0; e < A.width; ++e)
Cvalue += A.elements[row * A.width + e] *
B.elements[e * B.width + col];
C.elements[row * C.width + col] = Cvalue;
}
```

The first line contains the -global- keyword declaring that this is an entry-point function for running code on the device. The declaration float Cvalue = 0 sets aside a register to hold this float value where we will accumulate the product of the row and column entries.The next two lines help the thread to discover its row and column within the matrix. It is a good idea to make sure you understand those two lines before moving on. The if statement in the next line terminates the thread if its row or column place it outside the bounds of the product matrix. This will happen only in those blocks that overhang either the right or bottom side of the matrix. The next three lines loop over the entries of the row of A and the column of B (these have the same size) needed to compute the (row, col)-entry of the product, and the sum of these products are accumulated in the Cvalue variable. Matrices A and B are stored in the device's global memory in row major order, meaning that the matrix is stored as a one-dimensional array, with the first row followed by the second row, and so on. Thus to find the index in this linear array of the (i, j)-entry of matrix A, for example, we compute ($i \times width$ of A) to find the starting index of the ith row, and then add j to go to the jth entry in that row. Finally, the last line of the kernel copies this product into the appropriate element of the product matrix C, in the device's global memory.

This approach has been proven to run 100 times faster than CPU computation for a 190 bit modulus [12], it only runs 2-3 times faster for 1024 bits modulus [12].The case is very different for protocol that we have implemented. The basic operation is multiplication of $l_0$-16 bit scalar by $3l_0$ which can be efficiently encoded through SIMD instructions on which GPU are specialized. Each of these operations can be executed inside single stream processor and large number of these processor in modern GPUs results in significant performance improvement.

## 4.3 Key Management Issue

### 4.3.1 Necessity of Key Management

In our system design, there are three entities Client, Service provider and Organization. The organization that wanted to outsource its data to the service provider

so that the task of management and storage and service to customer can be handled by service provider only. But there are many challenges that are appear while focusing on this scenario, first is the privacy of our data that is being stored and the next is the integrity, confidentiality,etc. For example, if organization has hundreds of latest different songs. The market value of that songs is quite large enough and organization want to store all these songs to the service provider. In outsourcing scenario the service provider is not trusted one, so service provider can make the replica of these songs and can sell these songs that is copy right violation. For this purpose the organization has to take some provision so that the server itself should not read the data.

### 4.3.2 Ways to Achieve Key Management

One way to achieve this is to kept all the data in the encrypted format. In implementation we have consider the different (block cipher) encryption algorithm and depending on the performance we have to choose one of them. Our protocol deals with the single database which is not replicated. So organization encrypt its entire data set by the single master key only. If there are different data sets that are present on service provider which are related to different clients then we have to consider the different keys for encryption with respect to each client.

If we consider the different keys for different client for the same database, then it becomes a tedious job to the organization to encrypt whole database again and again with the different keys with respect to different clients. This solution is not feasible one to achieve the security of organization's data.

So we have considered only one master key that is used to encrypt the whole database and storing that encrypted data on the service provider side. In this situation how to securely transfer the key to the different customer by the organization by considering all the attack that is possible to achieve that key by the attacker. The solution for this can be use of Diffie-Hellman Key exchange algorithm [26]. But this algorithm is susceptible to the Man-In-Middle attack. For this case we have decided to break the key in to two part, but we have to create a key in a such way that both the half part of the key is dependent on each other. That is if we have half part of key then we can able to generate the next half part of the original key. This scenario is applied in proposed protocol to solve the problem of key management.

The Key of length n is divided in two parts that is KEY1 and KEY2 each of length n/2. This key is distributed as shown in the below diagram to different

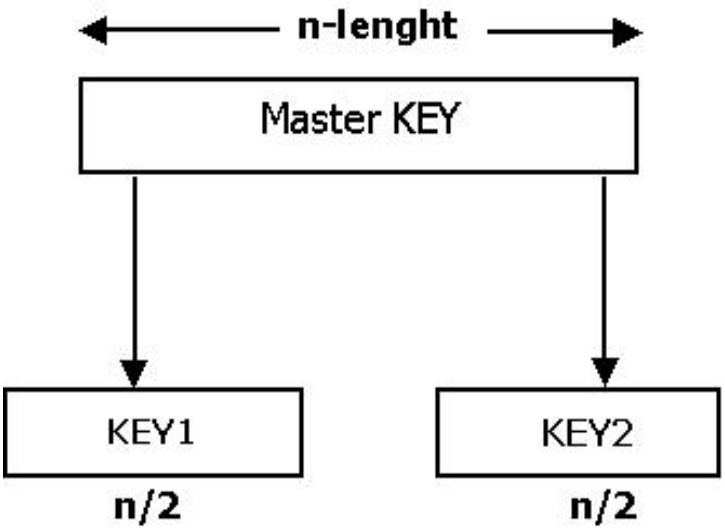


Figure 4.2: Key Manipulation

customer.

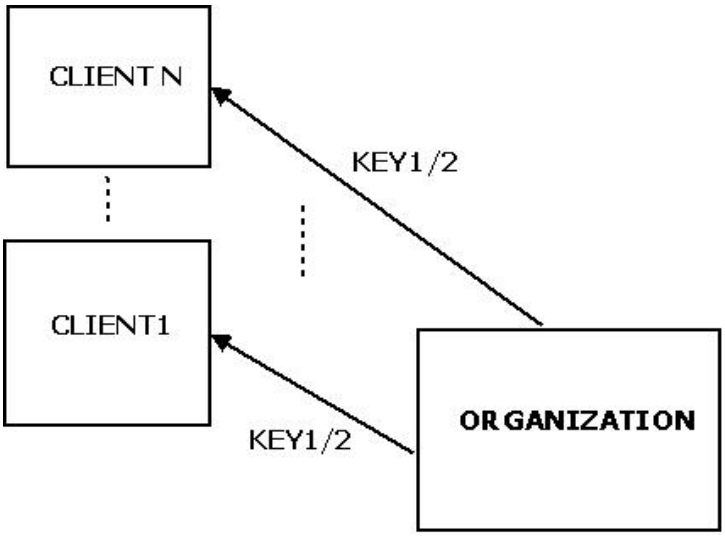


Figure 4.3: Key Management Overview

In the implementation we have consider the rest part of sub-key is generated on the respective client side. So according to that we have create a logic on the client side so that by using any sub-key we can generate the rest sub-key and properly concatenating these we can get the master key that is use for decryption process. In this process of key distribution by organization if attacker tries to attack in transit process of key exchange, it will only get half part of key, so it has to recover the rest half part to get the actual master key but it is difficult to get

the logic of that because it is implemented on the client side only. In the above diagram one master key is divided only in two sub-keys. This approach emerges some limitations which we have to deal with this key management issue.

### 4.3.3 Limitations of Our Approach

- As in key management solution only two sub-keys are produced and used in the entire process of communication. There is chances of getting the same sub-key to more than one client so the probability of getting same sub-key is greater than 0.5.
- Also, let client1 get the sub-key1 and client2 get the same sub-key1. If the client1 is not loyal and becomes attacker then it can attack in the process of key exchange of organization and client2 and grab the encrypted data which is in transit and try to decrypt it with the help of half sub-key1 and rest of the half key it will generate on its side. So that there is chances that client1 can access client2 data which is in transit.

## 4.4 Achieving Data Confidentiality

As we have discussed in the above section of key management that how the master key is distributed among the different clients. As we have deal with block cipher encryption algorithm in our implementation. The algorithm we have used is symmetric in nature. That is only one key is used for both encryption and decryption process.

In this implementation of proposed system the data confidentiality is achieved by symmetric encryption algorithm like AES (Advanced Encryption Standard). AES is algorithm capable of using 128,192 and 256 bit. AES performs encryption (and the reverse, decryption) which is a series of well-defined steps that can be followed as a procedure.AES has a fixed block size of 128 bits [14].

### 4.4.1 Overview of AES Algorithm

1) KeyExpansion
2) Initial Round

- AddRoundKey each byte of the state is combined with the round key using bitwise xor.

3)Rounds

- SubBytes  a non-linear substitution step where each byte is replaced with another according to a lookup table.
- ShiftRows  a transposition step where each row of the state is shifted cyclically a certain number of steps.
- MixColumns  a mixing operation which operates on the columns of the state, combining the four bytes in each column.
- AddRoundKey

4)Final Round (no Mix Columns)

- SubBytes
- ShiftRow
- AddRoundKey

### 4.4.2 Advantages of AES

- AES doesn't use a Feistel network and is fast in both software and hardware.
- Advanced Encryption Standard not only assures security but also improves the performance in a variety of settings such as smart cards, hardware implementations etc.
- AES is federal information processing standard and there are currently no known non-brute-force direct attacks against AES.
- AES is strong enough to be certified for use by the US government for top secret information.

## 4.5 Package Description

In implementation of this proposed system we have come around the different packages which are necessary for our system to run.
1) NTL (Number Theory Library)
2) Crypto++ Library
3) CUDA toolkit, Device Driver and cuda SDK

### 4.5.1 NTL

In our implementation the number theory library is used on the client side. NTL is written and maintained by Victor Shoup. NTL [22] is a high-performance, portable C++ library providing data structures and algorithms for manipulating signed, arbitrary length integers, and for vectors, matrices, and polynomials over the integers and over finite fields. NTL provides a clean and consistent interface to a large variety of classes representing mathematical objects. It provides a good environment for easily and quickly implementing new number-theoretic algorithms, without sacrificing performance.

### 4.5.2 Crypto++

Crypto++ Library is a C++ class library of cryptographic schemes. It is freely available. Crypto++ provides complete cryptographic implementations. This library deals with the different cryptographic algorithm given at [24].On Linux, Crypto++[24] is named libcryptopp. Crypto++ is also called as CryptoPP, libcrypto++, and libcryptopp. It is a free and open source C++ class library of cryptographic algorithm.

### 4.5.3 CUDA Drivers

In this implementation we have focused on use of GPU so that we require some drivers and GPU enabled devices. Here CUDA toolkit, device driver and CUDA sdk is required and that can be downloaded from [27]. The detail explanation is given in the manual that we get while downloading these requirements.

In our implementation ./install script gives us necessary steps required to run the proposed system. It also gives the installation of basic packages that are primarily required for the system. With the help of the script, it is possible to compile client or server with or without CUDA to check the CPU and GPU performance.

As we have discussed, the server is get compiled in the server directory during the installation process. PIRServer configfile [port]is the command used to run the basic server on both CPU or GPU. Here the configfile is the configure file that deals with list of the file names that are present on the service provider. The server use these files (database elements) to form the matrices $A_1....A_n$. After running the same command, it loads the list of file names and wait for the connection with client. The client is initiated by running the command PIRClient. The client is

get compiled in the client directory. Then it asks for the server IP (default IP 127.0.0.1), after that it connects to the server and client receives the list of file names from the service provider. Then client chooses an index for a file and client generates the query for that element using NTL. After the query is generated([2])it is sent to the server. Once the query is received on server side, the server starts

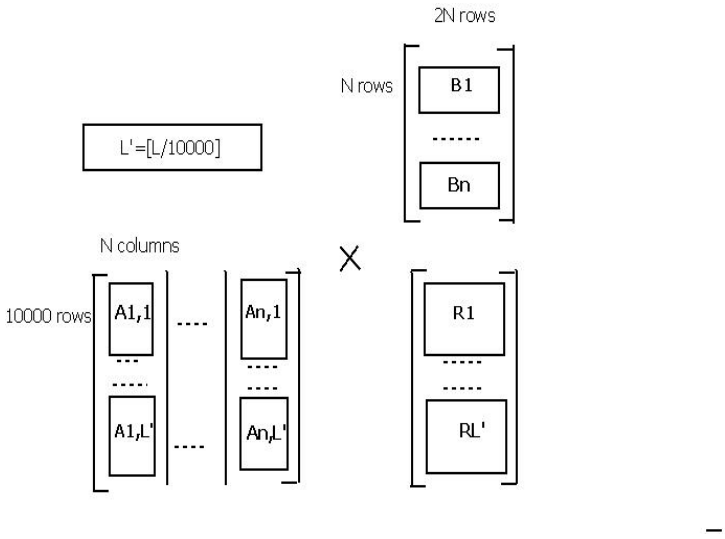


Figure 4.4: Block Matrix Multiplication

reply computation. In this scenario if there are more than two clients and initiating one after another then the client who initiate first the server provide the service to that client first and then it moves to other one. In order to optimize the cache usage we have used block matrix multiplication algorithm. The elements of database are split in matrix block of 10000 rows. Server iteratively performs GPU to compute $A_i1 \times B_i$and add the obtain results respectively. After performing the n iterations server get reply $R_1$ and send this reply to client. The same procedure is followed by the server until and unless it obtains all the reply and finally reply is send to the client. On the client side it decodes the reply and extract the element by using reply extraction algorithm[2].

## 4.6 Specification For Implementation

### 4.6.1 Hardware Requirements

- Dual-Core AMD Opteron(tm) Processor 2212
- RAM 1GB

- nVIDIA GTX 280

### 4.6.2 Software Requirements

- Windows NT/Linux/Mac
- C++
- Number Theory Library
- Crypto++ library

## 4.7 Screen-shot of Implemented System

### 4.7.1 Service Provider

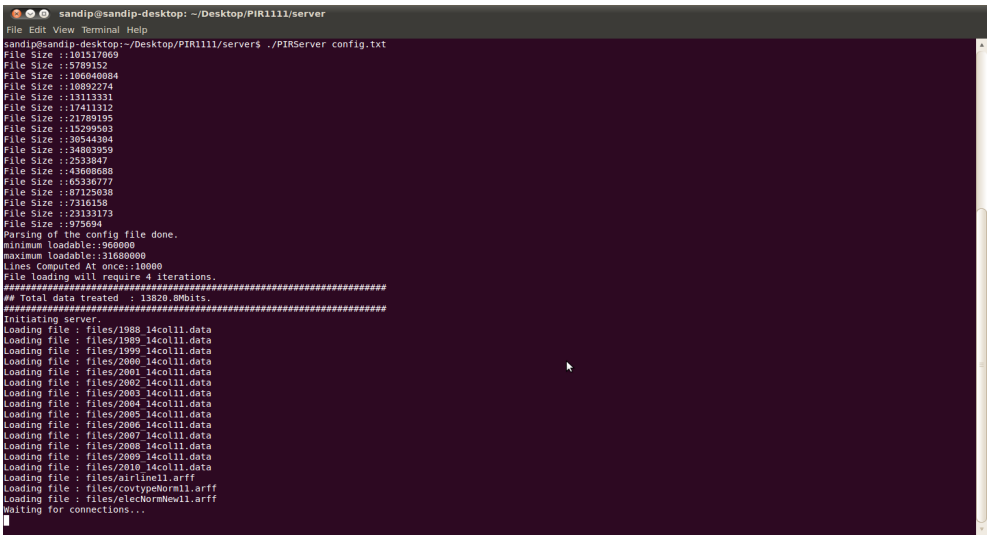


Figure 4.5: Service Provider

In the above screen-shot of server, we can observe the step by step execution of server.

### 4.7.2 Organization

We can see that the organization has to initiate permanantly. So if client initiate then it can send the sub-key in secure manner.

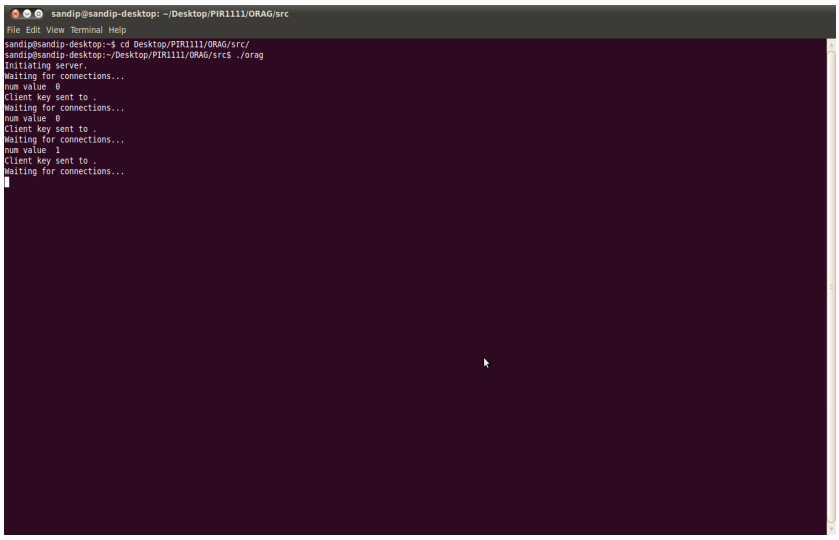


Figure 4.6: Organization

### 4.7.3 Client

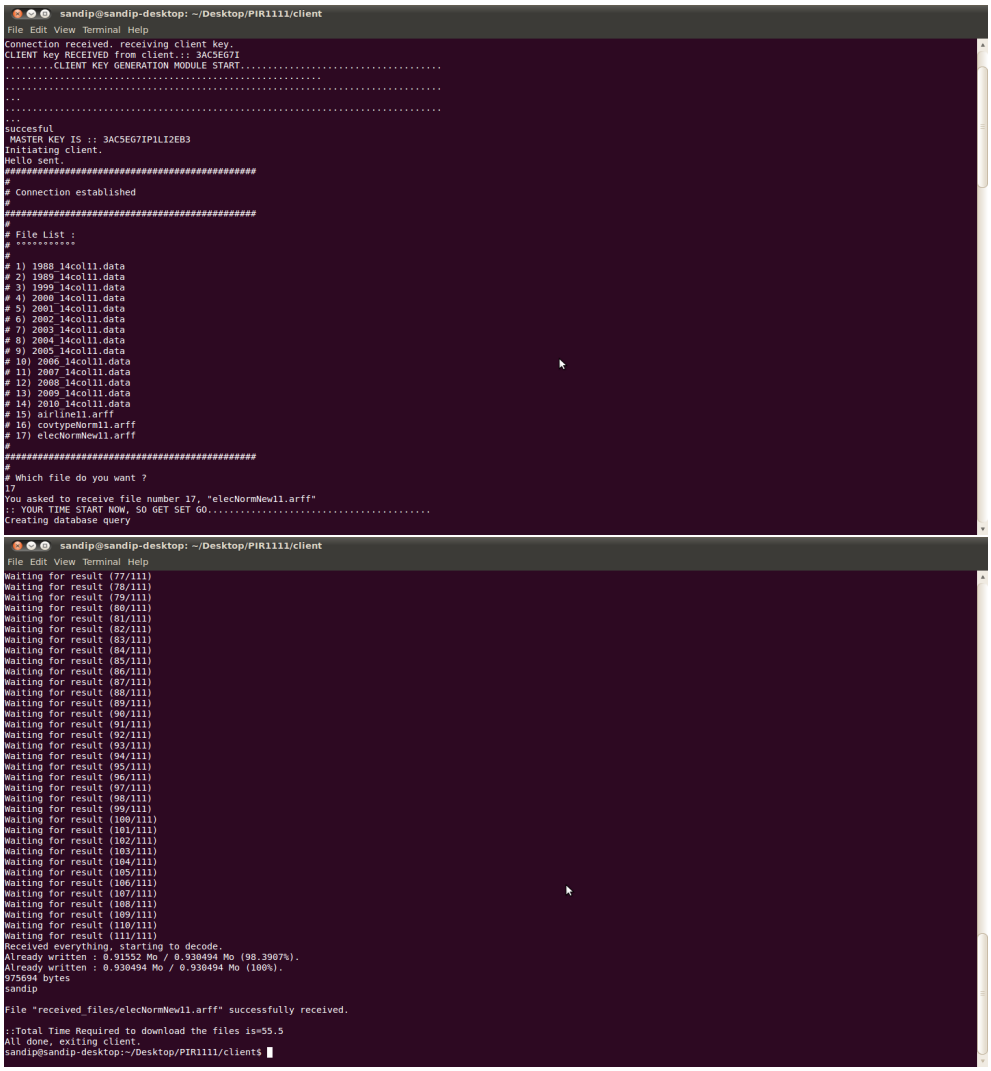


Figure 4.7: Client

# Chapter 5

# Experiments and Results

In this implementation of system, the experimental results are mainly based on the comparision of total time required to downlaod an element from the database on the client side with respect to CPU and GPU, the accessibility cost in case when client request the same element again from the database, user privacy and data confidentiality.

**1) Comparision analysis of with respect to time and increase in database sizes**

Table 5.1: Performance of PIR on CPU

| File Size(MB) | T.C.Time(sec) | T.D.Time(sec) | Data Rate(MB/s) |
|---|---|---|---|
| 10 | 319.829 | 390.566 | 45.5718 |
| 50 | 320.134 | 643.944 | 45.5285 |
| 100 | 313.504 | 958.716 | 46.4913 |
| 150 | 311.936 | 1279.17 | 46.7249 |
| 200 | 312.791 | 1593.08 | 46.5972 |
| 250 | 312.838 | 1877.39 | 46.5902 |

Table 5.2: Performance of PIR on GPU

| File Size(MB) | T.C.Time(sec) | T.D.Time(sec) | Data Rate(MB/s) |
|---|---|---|---|
| 10 | 42.4508 | 113.539 | 343.343 |
| 50 | 42.1197 | 366.723 | 346.043 |
| 100 | 39.6966 | 686.347 | 367.165 |
| 150 | 41.5095 | 1018.57 | 351.129 |
| 200 | 40.0291 | 1316.29 | 364.115 |
| 250 | 41.5172 | 1615.4 | 351.064 |

Here T.C Time is Total Computation Time, T.D.Time is Total Downloading Time. The above table gives the information about the performance of the proposed system on the CPU and GPU respectively. The total computation time

is the time required to perform the reply generation on CPU and GPU. From this table we have concluded that the total computation time required for reply generation is more on CPU than GPU. The data rate is the rate at which the information is transferred from service provider to the client. From this table we have observed that the data transferring rate is more in case of GPU than CPU.

Hence the total time required to download the element from the main database is less in case of GPU than CPU in our proposed system. The speed up can be calculated as the execution time on sequential algorithm divided by execution time on parallel algorithm (S=$T_s/T_p$ where $T_s$ is execution time on serial and $T_p$ is execution time on parallel). The speedup that we have obtained using GPU is between 7-8.

**2) Result Comparison of Our Approach with Carlos-Aguilar Approach**
The results that we have drawn is based on the number of files and the data rate with their respective sizes. In result analysis, we have tested the system of Carlos-Aguilar for 2-files,10-files and 100-files of 1MB, 5MB and 10MB file sizes on AMD-Opteron having 4GB RAM and having graphics card of GeForce GTX 280. The same statics that are also applied on our approach. We get the following results.

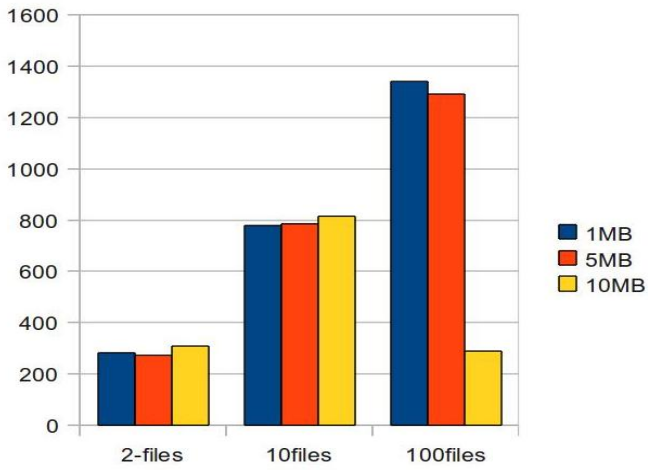


Figure 5.1: Result Comparison of Data Rate v/s No. of Files (Carlos-Aguilar Approach)

on X-axis Number of files and on Y-axis Data Rate(MB/s) The above comparison shows that, in case of GPU the optimal value is attained when the file size increases to some megabytes. In our approach basically the data which are store

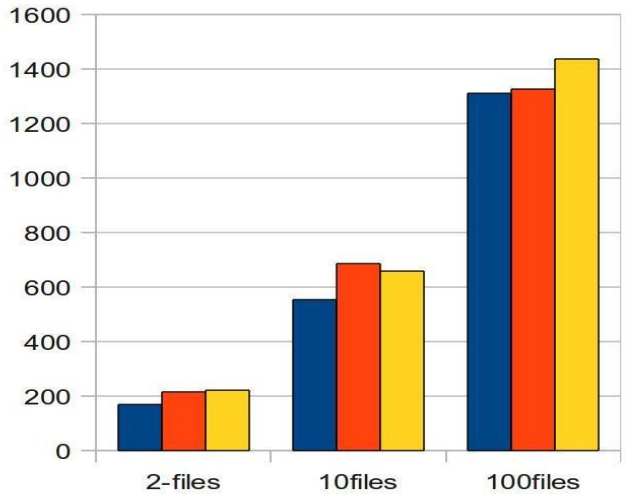


Figure 5.2: Result Comparison of Data Rate v/s No. of Files (Proposed System)

on the service provider side is in encrypted and with zip format, so the size of files on service provider in our approach is less than in case of Calos-Aguilar system. Hence there is change observable in both scenario particulary in case of 100 files of 10MB. Thus our system tries to degarde the computational cost at the server side and increases the processing throughput in terms of data rate.

**3) Accessibility Cost**

- The accessibilty cost in our implementation is related to the access to the database which is on service provider side. To reduce the cost of access to main database, this system maintains the cache like structure (received folder) on the client side, in which the visited record is maintained. In traditional approach of PIR as mentioned above for each request, the client has to visit each and every records in database. If client requests the same record again then instead of visiting the main database it can visit the received folder. Let the main database on the serice provider contains N elements. If client request a ith record out of N record then accessibility cost for traditional approach is O(N). In our proposed system the worst accessibility cost is number of records in the received folder that is m (let m records in received folder $m < N$).

**4) User Privacy**

- To achieve the user privacy our proposed system totally based on three algorithm that are discussed in the above sections which are basically deal

with lattice based approach. In PIR protocol to obtain particular element from database, it has to visit every entry from the database. Proposed protocol is said to be correct if user get the correct element from service provider and said to be private if it is not get reveled at service provider side. We have manually checked for different elements that user gets the same element that he/she wants by checking the contents of element.

**5) Comparison with Existing schema of single database PIR**

Table 5.3: Comparison with Existing Schema

| Approaches | User Privacy | Min Computational Cost | Min Accessibility cost | Data Confidentiality |
|---|---|---|---|---|
| High Speed Single db PIR | √ | √ | × | × |
| Improved Performance of PIR | √ | × | × | √ |
| Our Approach | √ | √ | √ | √ |

The above table shows the comparison between different approaches which basically deal with single database PIR. In above table the parameters that are achieved by proposed system is shown by tick.

# Chapter 6

# Conclusion and Future Work

## 6.1 Conclusion

This implementation of Private Information Retrieval protocol gives user privacy as well as data confidentiality with the help of encryption algorithm like AES algorithm. This implementation also shows need of use of parallel processing architecture to increase speed of execution of protocol, which will lead to practical usability of PIR schemes in real world. It also reduces the accessibility cost with the help of cache like technique, so that request of repeated records can be handled on the client side. As this scheme deal with the use of single database which is non-replicated, so this proposed system handles the key management issue with some pros and cons. Thus our proposed system design increases overall performance of PIR protocol.

# Future Work

- As per discussion regarding key management issue, the new better different techniques will be applied to solve this issue in better way, so that key can be exchanged between multiple clients and the organization securely without any attack.
- This implementation deals with the testing of only text files with their increase in size. So this protocol will be able to test on the different types of databases.
- Use of new GPUs that are arising in market day by day will lead to better performance.

# Appendix A

# Practical Approach of PIR

In the above sections we have discussed about the new approach that we have used for the traditional Private Information Protocol. Now we are going to discuss about how the traditional PIR practically works. We are going to discuss the toy example that is showing how this PIR protocol is executing step by step and their intermediate steps can be checked out with the help of this toy example. We give here a toy example of our protocol. Let database be a database with n = 2 elements of 8 bits each, and suppose N = 2, and $i_0 = 2$.

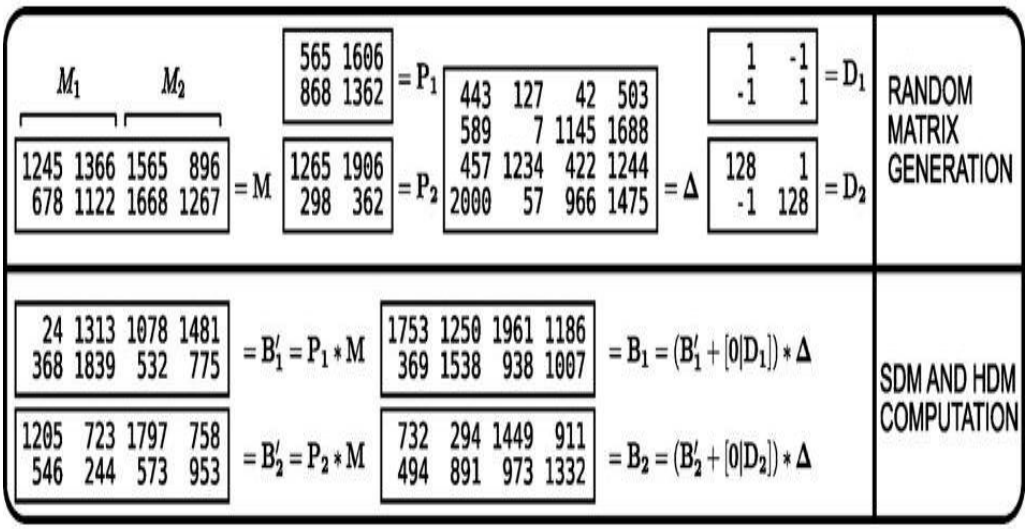


Figure A.1: Query Generation

The user first sets $l_0 = \lceil \log(\text{n} \times \text{N}) \rceil + 1 = 4$ and set q as $2^{l_0-1}$=128 and p as a prime larger than $2^{3l_0} < 2053$. He then generates A and B, two random $2 \times 2$ matrices over GF(2053), A being invertible After that he generates two

invertible matrices P1 and P2, the noise scrambling diagonal matrix $\Delta$, one soft noise matrix D1 and one hard noise matrix D2. Then the user computes M1 and M 2 multiplying M by P1 and P2. The first of these matrices is soft disturbed by adding the results of the soft noise matrix D1 to its third and fourth columns and multiplying the result to the right by $\Delta$ which results in M1. M2 is computed similarly. Finally, the user chooses a random permutation of the four columns and applies it both to $M_1$ and $M_2$ to obtain the final query $\{M_1, M_2, p\}$.

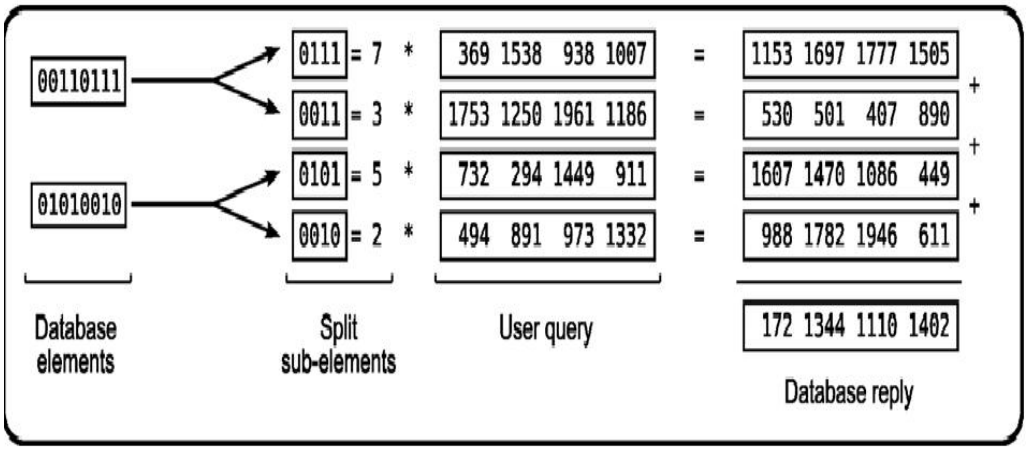


Figure A.2: Reply Generation

The database has two 8-bit elements, each is split in two 4-bit sub-elements. This sub-elements will be represented as integers in the interval $i\epsilon\{0.....15\}$ in above fig. When the database receives the query $\{M_1, M_2, p\}$, it multiplies the two matrices rows by the sub-elements and sends back the sum of all results. Upon reception of the database reply, the user inverts the column permutation. He then deducts from the first columns the undisturbed vector associated to the database reply and uses it to recover the scrambled noise. Using $\Delta - 1$ he unscrambles the noise and through the Euclidean algorithm he recovers the databases sub-elements encoded in the answer and reconstructs $m_{i_0}$.

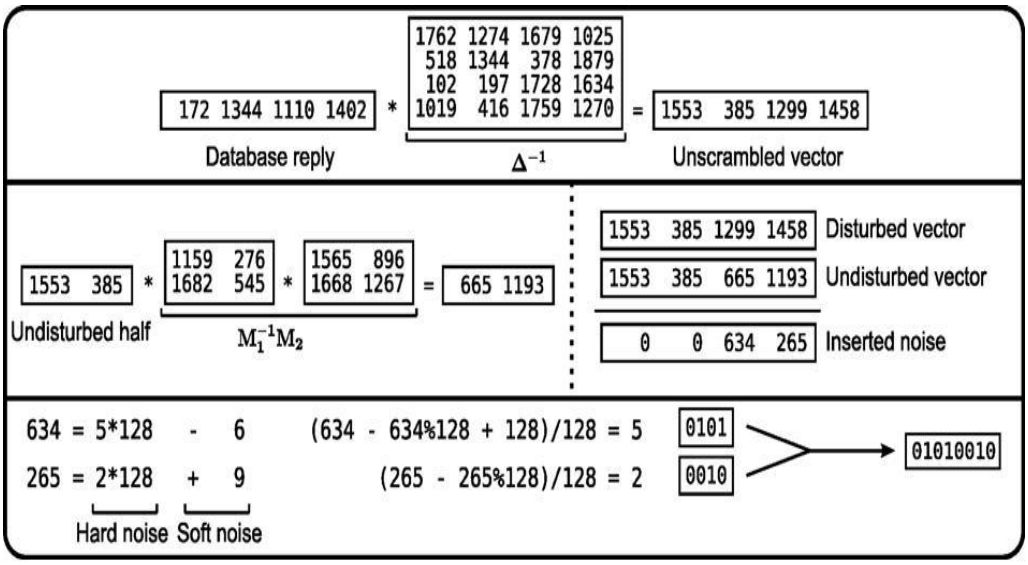


Figure A.3: Reply Extraction

As per shown in above example one can retrieve database index by performing all operations in three stages of given Lattice based Private Information Retrieval.

# Appendix B

# Paper Publications

In this section detail overview of paper publication is given which includes paper title, abstract, conference details.

## B.1 Paper Acceptance notification for ICETCIT 2012 - India

**Title of Paper:**
Optimized Private Information Retrieval Using Graphics Processing Unit and Cache Technology.
**Conference Details:**
International Conference on Emerging Trends in Computer and and Information Technology (ICETCIT 2012 ) http://www.icetcit.com

# Bibliography


[1] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan, Private information retrieval, *in Foundations of Computer Science,1995. Proceedings., 36th Annual Symposium on*, oct 1995,pp. 4150.

[2] C. Aguilar Melchor and P. Gaborit. A Lattice Based Computationally-Efficient Private Information Retrieval Protocol. In *Western European Workshop on Research in Cryptology (WEWoRC2007), Bochum, Germany. Book of Abstracts, pages 50 54, Extended version available on* IACR eprints http://eprint.iacr.org/2007/446, 2007.

[3] T. Harder and A. Buhmann, Value complete, column com plete, predicate complete magic words driving the design of cache groups, *VLDB JOURNAL*, pp. 101007, 2007.

[4] E. Kushilevitz and R. Ostrovsky, Replication is not needed: single database, computationally-private information retrieval, in *Foundations of Computer Science, 1997. Proceedings., 38th Annual Symposium on*, oct 1997, pp. 364373.

[5] B. Chor and N. Gilboa, Computationally private information retrieval (extended abstract), in Proceedings of the twenty ninth annual ACM symposium on Theory of computing,ser. STOC 97. New York, NY, USA: ACM, 1997, pp. 304313. [Online]. Available: `http://doi.acm.org/10.1145/258533.258609`

[6] J. P. Stern, U. C. Group, B. Maxwell, and P. D. Levant, A new and efficient all-or-nothing disclosure of secrets protocol. Springer-Verlag, 1998, pp. 357371.

[7] C. Cachin, S. Micali, and M. Stadler, Computationally private information retrieval with polylogarithmic communication, in *Advances in Cryptology* EUROCRYPT 99, ser. Lecture Notes in Computer Science, J. Stern, Ed. Springer Berlin / Heidelberg, 1999, vol. 1592, pp. 402414, 10.1007/3-540-48910-X 28. [Online]. Available:`http://dx.doi.org/10.1007/3-540-48910-X28`

[8] C. Gentry and Z. Ramzan, Single-database private information retrieval with constant communication rate, *In Proceedings of the 32nd International Colloquium on Automata, Languages and Programming.* Springer-Verlag, 2005, pp.803815.

[9] H. Lipmaa, An oblivious transfer protocol with log-squared communication, 2004.

[10] Y. cheng Chang, Single database private information retrieval with logarithmic communication, Tech. Rep., 2004.

[11] C. Melchor and P. Gaborit, A fast private information re- trieval protocol, in *Information Theory, 2008. ISIT 2008. IEEE International Symposium on*, july 2008, pp. 18481852.

[12] C. A. Melchor, High-speed single-database pir implementation.

[13] Single-Database Private Information Retrieval Protocols :Overview, Usability and Trends, Rapport de recherche, 2007. [Online]. Available: http://hal.inria.fr/inria-00134415

[14] A. Nadeem and M. Y. Javed, A performance comparison of data encryption algorithms, 2005 *International Conference on Information and Communication Technologies*, pp. 8489,2005. [Online]. Available: http://ieeexplore.ieee.org/lpdocs

[15] W. mei W. Hwu, *GPU Computing Gems Emerald Edition*, 1st ed.San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2011.

[16] D. H. Bailey, High performance computing at nasa, *Performance Computing*, vol. 1, no. 2, pp. 238241, 1994.[Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5455096

[17] C. Aguilar Melchor and P. Gaborit. A Fast Private Information Retrieval Protocol. In *The 2008 IEEE International Symposium on Information Theory (ISIT08), Toronto, Ontario, Canada.* To appear. IEEE Computer Society Press,2008.

[18] N. Corp. NVIDIA CUDA Compute Unified Device Architecture Programming Guide 1.1. http://developer.download.nvidia.com/compute/cuda/1 1/NVIDIA CUDA Programming Guide 1.1.pdf, 2007.

[19] S. Fleissner. GPU-accelerated montgomery exponentiation. In *7th International Conference on Computational Science(ICCS07), Beijing, China*, volume 4487 of Lecture Notes in Computer Science, pages 213220. Springer, 2007.

[20] W. Gasarch and A. Yerukhimovich. Computational inexpensive PIR, 2006. Draft available online at http://www.cs.umd.edu/arkady/pir/pirComp.pdf.

[21] Ian Goldberg. Improving the Robustness of Private Information Retrieval. In *The 2007 IEEE Symposium on Security and Privacy (SP'07), Oackland, CA, USA, pages 131148.IEEE Computer Society Press*, 2007.

[22] V. Shoup. NTL: A library for doing Number Theory. http://www.shoup.net/ntl/, 2007.

[23] R. Sion and B. Carbunar. On the Computational Practicality of Private Information Retrieval. In *14th ISOC Network and Distributed Systems Security Symposium (NDSS07), San Diego, CA, USA*, 2007.

[24] Crypto++: An open source cryptographic library http://www.cryptopp.com/wiki/

[25] Giovanni Di Crescenzoy Tal Malkinz Rafail Ostrovskyx: Single Database Private Information Retrieval Implies Oblivious Transfer in EUROCRYPT 2000, LNCS 1807, pp. 122-138, 2000.

[26] Diffie Hellman Key exchange alorithm:http://en.wikipedia.org/wiki/Diffie

[27] CUDA :http://developer.nvidia.com/cuda-downloads