Symmetrically Private Information Retrieval (Extended Abstract)

Sanjeev Kumar Mishra[1] and Palash Sarkar[2]

[1] Cisco Systems(India) Private Limited
Prestige Waterford, No 9, Brunton Road
Bangalore-560 025. India
sanjeevm@cisco.com
[2] Centre for Applied Cryptographic Research
Department of Combinatorics and Optimization
University of Waterloo, 200 University Avenue West
Waterloo, Ontario, Canada N2L 3G1
(On leave from Indian Statistical Institute, Calcutta)
psarkar@cacr.math.uwaterloo.ca, palash@isical.ac.in

Abstract. In this paper we present a single-round, single-server symmetrically private information retrieval scheme, in which privacy of user follows from intractability of the quadratic residuacity problem and the privacy of the database follows from the XOR assumption for quadratic residues introduced in this paper. The communication complexity of the proposed scheme for retrieving one bit can be made $O(n^{\epsilon})$, for any $\epsilon > 0$, where n is the number of bits in the database. We extend the protocol to a block retrieval scheme which is specially efficient when the number of records in the database is equal to the size of each record.

Keywords: *Private Information retrieval (PIR), Symmetrically Private Information Retrieval (SPIR), Quadratic Residuacity Assumption (QRA), Probabilistic Encryption, Cryptographic Protocols.*

A Introduction

In the age of Internet accessing remote database is common and information is the most sought after and costliest commodity. In such a situation it is very important not only to protect information but also to protect the identity of the information that a user is interested in. Consider the case, when an investor wants to know value of a certain stock, but is reluctant to reveal the identity of that stock, because it may expose his future intentions with regard to that stock. The scheme or protocol which facilitates a user to access database and receive desired information without exposing the identity of information was first introduced by Chor et al. [2] in 1995 under the name of *Private Information Retrieval*. It was also independently studied by Cooper and Birmen [4] in context of implementing an anonymous messaging service for mobile user.

B. Roy and E. Okamoto (Eds.): INDOCRYPT 2000, LNCS 1977, pp. 225–236, 2000.

A Private Information Retrieval (PIR) scheme allows a user to retrieve bits from a database (DB) while ensuring that the database does not learn which bits were retrieved. The database is modeled as an array of bits x held by a server, and the user wants to retrieve the bit x_i for some i, without disclosing any information about i to the server. We denote number of bits (records) in database by n. The communication complexity (i.e., the number of bits exchanged between user and DB), of such a protocol is denoted by $C(n)$. Also the exchange of information between the user and the DB may be interactive or non-interactive. In the first case the protocol is single-round while in the second case it is multi-round.

A trivial PIR scheme consists of sending the entire database to the user, resulting in $C(n) = n$. Clearly, such a scheme provides information theoretic privacy. Any PIR scheme with $C(n) < n$ is called non trivial. Chor et al. [2] proved that under the requirement of information theoretic security, and involvement of single database in the protocol, trivial PIR is the only possible solution. A way to overcome this impossibility result is to consider $k > 1$ servers, each holding a copy of the database x. Chor et al. [2] presented a $k > 1$ server scheme with communication complexity $O(n^{\frac{1}{k}})$. This was improved by Ambainis [1] to a $k > 2$ server PIR scheme and having a communication complexity of $O(n^{\frac{1}{(2k-1)}})$.

Another way of getting non-trivial PIR schemes is to lower the privacy requirement from information theoretic privacy to computational privacy. Chor and Gilboa [3] presented multi-server computationally private information retrieval schemes in which the privacy of user is guaranteed against the computationally bounded servers. Kushilevitz and Ostrovsky [7] showed that under the notion of computational privacy one can achieve nontrivial PIR protocol even with a single server. In particular, [7] show that assuming the hardness of quadratic residuacity problem, one can get single database PIR protocol with communication complexity $O(n^{\epsilon})$ for any $\epsilon > 0$.

While protecting the privacy of user, it is equally important that user the should not be allowed to learn more than the amount of information that he/she pays for. This is refered to as database privacy and the corresponding protocol is called a Symmetrically Private Information Retrieval (SPIR) protocol. However, the database is assumed to be honest and carries out its part of the protocol correctly. Gertner et. al. [6] presented general transformations of PIR schemes in multi server setting satisfying certain properties into SPIR schemes, with logarithmic overhead in communication complexity. Kushilevitz and Ostrovsky [7] noted that in a setting of single server, their PIR protocol can be converted into a SPIR protocol employing zero knowledge techniques to verify the validity of query. The problem is that the use of zero knowledge techniques imply that the resulting protocol is no longer a single round protocol. Thus the question of getting single-server, *single-round* nontrivial SPIR protocol was left open.

We provide the first solution to this problem by modifying the basic PIR scheme of [7] to ensure that privacy of the database is maintained. Our scheme introduces two new steps (preprocessing and postprocessing) in the database computation but does not increase the number of rounds. There is a communication overhead in the communication from the user to the database, but in

the recursive scheme this is offset by the postprocessing step which effectively decreases the number of bits sent by the database to the user. In fact, for just PIR scheme the preprocessing step is not required and hence the total communication complexity is K times less than that of [7]. The preprocessing step of database computation empower the DB to restrict the user to get information from one column of the matrix formed from the database, while the postprocessing computation prevents the user to get more than one bit from the column selected in preprocessing step. Thus the user is constrained to get exactly a single bit of information from the database for each invocation of protocol. The proof of user privacy is based on the intractibility of the quadratic residuacity problem and the proof of database privacy requires a new assumption which we call the XOR assumption. In the XOR assumption we assume that if $x, y \in Z_N^{+1}$ and $w = x \oplus y$, then from w it is not possible to get any information about the quadratic character of x and y even if the user is computationally powerful enough to determine quadratic residuacity in Z_N^{+1}.

We extend the basic scheme into an efficient (in terms of communication complexity) protocol for SPIR by allowing the database to perform a depth first search on matrices of progressively smaller sizes. As a result we are able to prove the following (see [7] for a similar result on PIR protocol).

Theorem: For every ϵ, there exists a single-server, single-round computational SPIR scheme with communication complexity $O(n^{\epsilon})$, where user privacy is based on QRA and database privacy is based on the XOR assumption.

We extend the bit retrieval scheme into a block retrieval scheme which is specially efficient when the number of records is equal to the size of a record.

Remark: Although we will present our schemes based on the Quadratic Residuacity Assumption, it can be based on more general assumptions. Following the approach of Mann [8], we can replace Quadratic Residuacity Predicates with any Homomorphic Trapdoor Predicates.

B Preliminaries

Informally stating, a PIR scheme is a collection of three algorithms, the users query generation algorithm $\mathcal{A}_{\mathcal{Q}}$, database answer computation algorithm $\mathcal{A}_{\mathcal{D}}$ and users reconstruction algorithm $\mathcal{A}_{\mathcal{R}}$, such that it satisfies the following requirements:

Correctness: In every invocation of the scheme the user retrieves the desired bit provided the user's query is correct.

User Privacy: In every invocation of the scheme the server does not gain any information about the index of the bit that the user retrieves.

PIR schemes can be classified into two groups on the basis of privacy they provide to the user.

Information Theoretic Privacy : Given the query to server or database, he cannot gain any information about the index the user is interested in, *regardless of his computational power.*

Computational Privacy : Here the server is assumed to be computationally bounded, say a polynomial size circuit. Thus for computational privacy, the probability distribution of the queries the user sends to database when he is interested in i_{th} bit, and the probability distribution of the queries when he is interested in i'_{th} bit, should be computationally indistinguishable to the server.

A symmetrically private information retrieval (SPIR) scheme is a PIR scheme satisfying an additional requirement, **the privacy of database**. Privacy can again be considered to be information theoretic or computational. Informally, computational data privacy requires, for each execution, there exists an index i, such that the probability distributions on the user's view are computationally indistinguishable for any two databases x, y such that $x_i = y_i$. That is a computationally bounded user does not receive information about more than a single bit of the database, no matter how he forms his query of given length.

For the formal definitions of PIR protocols refer to Chor et. al. [2], Kushilevitz and Ostrovsky [7] and Mann [8]. For the formal definitions of SPIR protocols see Gertner et. al. [6], Crescenzo et. al. [5], Mishra [9]. Also Mishra [9] contains a bibliography on this and related security problems.

C The Basic Scheme

In this section we introduce a basic SPIR scheme for bit retrieval. We add two new and important steps - the preprocessing and the postprocessing steps - to the database computation in the protocol of [7]. The preprocessing step restricts user access to a particular column and the postprocessing step allows the user to get only a single bit from the column. This scheme in itself is not efficient, but it makes the main ideas clear which will be used in the efficient bit retrieval scheme based on a recursive database computation. In the following, by a QNR we will mean a quadratic non residue whose Jacobi symbol is 1. Also by Z_N^{+1} we denote the set of all elements in Z_N^* whose Jacobi symbol is 1.

For clarity, we use DB to denote server or database program which handles the database. We view the n-bit database string x to be an $(s \times t)$ matrix of bits denoted by D. The user is interested in retrieving the i_{th} bit x_i of the database x, which is the $(a, b)_{th}$ entry in matrix D, where, $(a, b) = getIndex(i, t)$. The algorithm $getIndex(,)$ is defined as follows.

$getIndex(i, t)$ {
  if $t \mid i$, then $t_1 = \frac{i}{t}$ and $t_2 = t$
  else $t_1 = \lfloor \frac{i}{t} \rfloor + 1$ and $t_2 = i \bmod t$.
  return (t_1, t_2).
}

Query Generation (by user) :

1. User generates two $\frac{K}{2}$ bit prime numbers p and q, such that $p \equiv q \equiv 3 \pmod 4$. Here K is the security parameter.

2. User chooses t numbers at random $y_1, \ldots, y_t \in Z_N^*$, such that y_b is a QNR and y_j $(j \neq b)$ is a QR.

3. User chooses s numbers at random $\gamma_1, \ldots, \gamma_s \in Z_N^*$, such that γ_a is a QNR

and γ_j $(j \neq a)$ is a QR.
4. User sends $N, y_1, \ldots, y_t$ and $\gamma_1, \ldots, \gamma_s$ to DB. The factorization of N is kept secret.
Database Computation :
1. ***Preprocessing*** *(Column Control):* For each bit $D(r,\tau)$ of matrix D, a K bit number $\psi(r,\tau)$ is randomly chosen as follows. If $D(r,\tau) = 0$ then $\psi(r,\tau)$ is a QR and if $D(r,\tau) = 1$, then $\psi(r,\tau)$ is a QNR. Denote by $\psi(r,\tau,\kappa)$ the κ_{th} bit of $\psi(r,\tau)$. DB now forms K matrices of size $s \times t$, $D_\kappa : \kappa = 1, 2, \ldots, K$ as follows: $D_\kappa(r,\tau) = \psi(r,\tau,\kappa)$ Thus for the original matrix D, DB has formed K matrices D_κ of same size. The database can always generate random QR's without knowing the factorization of N. The primes p, q are chosen to be $p, q \equiv 3 \bmod 4$. This allows the DB to also randomly generate QNR's in Z_N^{+1}.
2. For each of the K matrices D_κ, DB computes for every row r of D_κ, a number $z(r,\kappa)$ as follows: $z(r,\kappa) = \nu_r \left(\prod_{l=1}^{t} (y_l)^{D_\kappa(r,l)}\right)$ where ν_r is a randomly chosen QR by DB. Thus, DB computes $s \times K$, numbers $z(r,\kappa)$ where each $z(r,\kappa)$ is a K-bit number. For, $1 \leq c \leq K$, denote by $z(r,\kappa,c)$ the c_{th} bit of $z(r,\kappa)$.
3. ***Post Processing*** *(Row Control):* DB forms K matrices $\Delta_\kappa, 1 \leq \kappa \leq K$, such that $\Delta_\kappa(r,c) = z(c,\kappa,r)$. Now, for each of the K matrices Δ_κ $(1 \leq \kappa \leq K)$, the database DB, computes for every row r $(1 \leq r \leq K)$ a number $\zeta(r,\kappa)$ as follows : $\zeta(r,\kappa) = \nu_r \left(\prod_{l=1}^{s} (\gamma_l)^{\Delta_\kappa(r,l)}\right)$ where ν_r is a randomly chosen QR by DB. Thus for each of the K matrices Δ_κ, DB computes K numbers $\zeta(1,\kappa)$,$\zeta(2,\kappa)$,...,$\zeta(s,\kappa)$, where each of these numbers in itself a K-bit number. DB sends these K^2 numbers (a total of K^3 bits) to user.
Bit Reconstruction : User retrieves the desired bit as follows:
1. Observe that $\zeta(r,\kappa)$ is a QR iff $\Delta_\kappa(r,a)$ is 0 (see Lemma 1). Thus determining the quadratic character of the K^2 numbers, $\zeta(r,\kappa)$, gives the user the bits $\Delta_\kappa(1,a), \Delta_\kappa(2,a), \ldots, \Delta_\kappa(K,a)$.
2. From the construction of matrix Δ_κ, we see that, $z(a,\kappa,r) = \Delta_\kappa(r,a)$, and further $z(a,\kappa) = z(a,\kappa,1), \ldots, z(a,\kappa,K)$ for all $1 \leq \kappa \leq K$. Thus for all $1 \leq \kappa \leq K$, user gets $z(a,\kappa)$.
3. The quantity $z(a,\kappa)$ is a QR iff $D_\kappa(a,b)$ is 0 (see Lemma 1). Thus by determining the quadratic characters of K numbers $z(a,\kappa)$ $(1 \leq \kappa \leq K)$, user gets the bits $D_1(a,b), \ldots, D_K(a,b)$. ¿From the construction of matrices D_κ, it is clear that $D_\kappa(a,b) = \psi(a,b,\kappa)$, and further $\psi(a,b) = (a,b,1), \ldots, \psi(a,b,K)$. Using Lemma 1, $\psi(a,b)$ is a QR iff $D(a,b)$ is 0. Thus user gets the bit $D(a,b)$.
Remark: The security parameter K must satisfy $K \geq \max\{K_0, \quad poly(\log n)\}$ where n is the number of bits in database, K_0 is the smallest number such that encryption scheme under consideration (encryption by QR's and QNR's here) is secure. The $poly(\log n)$ factor comes because, DB is assumed to be resourceful enough to do a computation of $O(n)$.
Correctness of the protocol follows from the following Lemma which is not difficult to prove.

Lemma 1. *Let $x = [x_1, x_2, \ldots, x_t]$ be a bit array of length t and let $\chi_1, \ldots, \chi_t$ be chosen such that $\chi_1, \ldots, \chi_{i-1}, \chi_{i+1}, \ldots, \chi_t$ are QR's and χ_i is a QNR. Let $y = \prod_{l=1}^{t} (\chi_l)^{x_l}$. Then y is a QR iff $x_i = 0$.*

Privacy of User : Suppose there exists a family of polynomial time circuits C_n that can distinguish between two query distributions Q_i and $Q_{i'}$ (for two indices i and i' of the database) with probability larger than $\frac{1}{n^\ell}$ for some integer ℓ. Then following [7], we can construct another family of polynomial time circuit C'_n from C_n, which will, on input N and $y \in Z_N^{+1}$ compute the quadratic residuacity predicate with probability at least $\frac{1}{2} + \frac{1}{8 \cdot n^\ell}$.

Privacy of database for *Honest but Curious User* is easy to prove and so here we consider the case of a *Dishonest User* A dishonest user can deviate from the protocol to possibly gain any extra information in the following ways:

1. N is a product of more than two primes. It is not clear that the user can gain extra information by using such N. Hence we will assume that N is indeed a product of two primes.

2. Assuming that N is a product of two primes, the numbers that the user sends to DB must be in Z_N^{+1}. This is necessary since the DB can perform this computation and reject a query not confirming to this specification. Hence the only way a dishonest user can cheat is to send more than one QNR's in each query set. We now argue that this provides the user with no information at all, i.e., even if one query set has two QNR's then the user does not get even one bit of the database.

Note that even if the user chooses the QR's and QNR's with some "special" properties, this will not help since in the computations of $z(r, \kappa)$ and $\zeta(r, \kappa)$, the multiplication by a randomly chosen ν_r will destroy these properties. Similar to Lemma 1, we have

Lemma 2. *Let $x_1, \ldots, x_t$ be in $QR \bigcup QNR$ and $b_1, \ldots, b_t$ be a bit string and a number z is computed as: $z = \prod_{l=1}^{t} (x_l)^{b_l}$. Suppose $x_{i_1}, \ldots, x_{i_s}$ are QNR's and rest are QR's, then z is a QR iff $b_{i_1} \oplus \ldots \oplus b_{i_s} = 0$.*

The problem is establishing database security is that we cannot base it on QRA, since the user is capable of determining quadratic residuacity in Z_N^{+1}. Indeed the user is required to determine quadratic residues for the protocol to work. However, we will see later that if the user sends more than one QNR's in his query set then he receives an element z in Z_N, which is the XOR of some randomly chosen elements $x_1, \ldots, x_t$ in Z_N^{+1}. We would like to argue that from z the user gets no information about the individual quadratic characters of any of the x_i's even though he knows the factorization of N. We make this requirement more formal in the following manner.

XOR Assumption : Let $z = x_1 \oplus \ldots \oplus x_t$, where $x_1, \ldots, x_t \in Z_N^{+1}$. Let $X = \{x_1, \ldots, x_t\}$ and A be an arbitrary subset of X. Then we assume that for N sufficiently large $Prob(A \subseteq QR, X - A \subseteq QNR \mid z) \in [\frac{1}{2^t} - \delta(K), \frac{1}{2^t} + \delta(K)]$, where K is the number of bits required to represent N and $\delta(K)$ goes to zero as K increases.

A formal proof of the XOR assumption looks difficult to obtain. Some experiments were conducted to verify the XOR assumption for $t = 2$ and small values of N. We briefly mention the important observations.

1. From simulation it is observed that $\delta(K)$ depends on $\frac{N - 2^{K-1}}{2^{K-1}}$. As this ratio increases from 0 to an upper bound of $\eta(K) > 0$, the value of $\delta(K)$ falls rapidly.

Further the upper bound $\eta(K)$ decreases exponentially with increasing K.
2. The nearer the ratio $\frac{p}{q}$ of two prime factors of N is to 1, the smaller is the value $\delta(K)$.
3. XOR Assumption can be generalized to any probabilistic encryption scheme, there are some supportive evidences which we are unable to present here for the lack of space.

We consider three cases which can occur from the possible nature of query sent by user:
First set contains more than one QNR : Let the first set in the query sent by user contain p many QNR's at positions $b_1, \ldots, b_p$. Then Lemma 2 implies that in the reconstruction phase a number $z(a, \kappa)$, $(1 \leq \kappa \leq K)$ obtained by user is a QR iff $D_\kappa(a, b_1) \oplus \ldots \oplus D_\kappa(a, b_p) = 0$. Thus user is able to reconstruct $\psi(a, b_1) \oplus \ldots \oplus \psi(a, b_p)$.
Second set contains more than one QNR : Let the second set of numbers in the query sent by user contains q many QNR's at positions $a_1, a_2, \ldots, a_q$. Again using Lemma 2 in post processing computation, we see that a number $\zeta(r, \kappa)$ received by user is a QR iff $\Delta_\kappa(r, a_1) \oplus \ldots \oplus \Delta_\kappa(r, a_q) = 0$, $(1 \leq r \leq K)$, and $(1 \leq \kappa \leq K)$. Thus user will be able to reconstruct $z(a_1, \kappa) \oplus z(a_2, \kappa) \oplus \ldots \oplus z(a_m, \kappa)$.
Both sets contain more than one QNR : Let first set contain more than one QNR's at positions $b_1, \ldots, b_p$ and the second set contain QNR's at positions $a_1, a_2, \ldots, a_q$. Then a number $\xi(r, \kappa)$ received by user is a QR iff $\Delta_\kappa(r, a_1) \oplus \ldots \oplus \Delta_\kappa(r, a_q) = 0$, $(1 \leq r \leq K)$, and $(1 \leq \kappa \leq K)$. Thus user will be able to reconstruct $z(a_1, \kappa) \oplus z(a_2, \kappa) \oplus \ldots \oplus z(a_q, \kappa)$ $(1 \leq \kappa \leq K)$.
Thus in all the three cases user will be able to reconstruct only XOR's of more than one numbers, and the XOR assumption says that from the XOR of two or more numbers from the set Z_N^{+1}, it is not possible to know anything about the quadratic characters of the constituent numbers. Hence if user sends more than one QNR's in any set of numbers in his query, he fails to get even one bit of the database x.
Communication complexity : Total communication from user to database DB in this scheme is $(1 + t + s)$ K-bit numbers $(N, y_1, \ldots, y_t, \gamma_1, \ldots, \gamma_s)$. while database returns K^2 K-bit numbers $\zeta_(r, \kappa)$ $(1 \leq r \leq K, 1 \leq \kappa \leq K)$ obtained after the postprocessing to user. Thus the communication complexity is $(1 + t + s + K^2) \cdot K$ bit, which can be minimized by choosing $t = s = \sqrt{n}$, and the communication complexity is: $(2\sqrt{n} + 1 + K^2) \cdot K$. Under similar assumptions on the security parameter Kushilevitz and Ostrovsky [7] obtained the communication complexity $(2 \cdot \sqrt{n} + 1) \cdot K$ for their basic PIR scheme. A closer look reveals that, with an extra communication of K^3 bit over the basic PIR scheme presented by Kushilevitz and Ostrovsky [7], we have successfully obtained a SPIR scheme. Even with the weaker assumption on security parameter, i.e.; $K = n^\epsilon$ for some constant $\epsilon > 0$, we get a communication complexity $O(n^{\frac{1}{2}+\epsilon})$, provided $\epsilon < \frac{1}{4}$. Thus we have proved the following theorem :

Theorem 1. *For every $\frac{1}{4} > \epsilon > 0$, there exists a single-server, single-round SPIR protocol with communication complexity $O(n^{\frac{1}{2}+\epsilon})$ where user privacy is based on QRA and the database privacy is based on the XOR assumption.*

D Iterative Bit SPIR Scheme

In this section we develop an improved scheme using the ideas developed in our basic scheme and we manage to bring down the communication complexity. We essentially achieve this by allowing the DB to do a recursive computation (see also [7]). We put stress on the fact that the scheme involves only a single round of communication and security of database as well as user remain preserved.

As before, we view the n-bit database string x to be an $(s \times t)$ matrix of bits denoted by D. The i_{th} bit x_i of the database x is $(a_1, b_1)_{th}$ entry in matrix D, where $(a_1, b_1) = getIndex(i, t)$.

Query Generation :

1. User generates two $\frac{K}{2}$ bit prime number p and q with $p, q \equiv 3 \bmod 4$. Calculate $N = pq$.

2. User calculates t such that $t^{L+1} = n$, where L is the depth of recursion of the database computation. The value of L is chosen such that communication complexity is minimized (as we will see later).

3. User generates a sequence of the pair of indices (a_j, b_j) as follows.

for $j \leftarrow 1$ **to** $L - 1$ $\{(a_{j+1}, b_{j+1}) = getIndex(a_j, t)\}$

The pair (a_j, b_j) correspond to the row and column index of the relevant bit in the matrices in j_{th} round of DB computation. Also define $s_j = \frac{n}{t^j}$ and $t_j = t$ for $1 \leq j \leq L$. (s_j, t_j) are number of rows and columns in each matrix in the j_{th} round of DB computation.

4. User generates an $L \times t$ matrix y, where for $1 \leq \sigma \leq L$, $1 \leq \beta \leq t$:
$y(\sigma, \beta)$ is a QNR if $\beta = b_\sigma$, else it is a QR. Clearly each row of y contains exactly one QNR.

5. User randomly chooses s_L numbers $\gamma_1, \ldots, \gamma_{s_L} \in Z_N^*$, such that γ_{a_L} is a QNR and γ_j $(j \neq a_L)$ is a QR.

6. User sends N, the matrix y and the numbers $\gamma_1, \ldots, \gamma_{s_L}$ to the DB. The factorization of N is kept secret.

Database Computation :

Database DB performs a $L + 2$ round of computation in three phases. First phase is the preprocessing round of basic scheme, while the third phase is the post processing round. The middle step is an L round recursive computation.

1. ***Pre Processing*** *(Column Control) :* As in the basic scheme DB, forms K matrices $D_\kappa(\alpha, \beta)$ from the original matrix $D(\alpha, \beta)$. Again the user requires exactly one bit from each of the matrices D_κ's.

2. ***Iterative Computation:*** The database DB performs a L round recursive computation according to following algorithm:
For each D_κ $(1 \leq \kappa \leq K)$ formed in the preprocessing round perform the call $DFSCompute(D_\kappa, s, t, y_1, 1)$.
The algorithm $DFSCompute$ is described below:
$DFSCompute(M, s, t, y_l, l)\{$
/* • y_l is the l_{th} row of the matrix of numbers sent by user. Note $y_l[i] = y(l, i)$ is a QR if $i \neq b_l$ and $y_l[i]$ is a QNR if $i = b_l$.
• M is an $s \times t$ matrix of bits and we want to retrieve the bit $M[a_l, b_l]$.

• l denotes the level of the recursion. */

1. Set for $1 \leq i \leq s$, $z[i] = \nu_i \left(\prod_{j=1}^{t} (y_l[j])^{M[i,j]} \right)$ where ν_i for each i is a QR chosen by DB uniformly at random.

/* Each $z[i]$ is a K-bit string. For $1 \leq j \leq K$, let $z[i,j]$ denote the j_{th} bit of $z[i]$. We require the string $z[a_l]$ */

2. For $1 \leq j \leq K$, form K matrices M_j, where each M_j is an $\frac{s}{t} \times t$ matrix formed from the column vector, $z[*,j] = z[1,j], \ldots, z[s,j]$ by breaking $z[*,j]$ into t-bit blocks and arranging the blocks in a row wise fashion.

/* The string $z[a_l]$ is distributed over the K matrices M_j, i.e., the string $z[a_l]$ is equal to $M_1[a_{l+1}, b_{l+1}], \ldots, M_K[a_{l+1}, b_{l+1}]$, where $(a_{l+1}, b_{l+1}) = getIndex(a_l, t)$. */

3. $for(1 \leq j \leq K)\{$
$if(l < L-1)\ DFSCompute(M_j, \frac{s}{t}, t, y_{l+1}, l+1)$
$else\ PostCompute(M_j, \frac{s}{t}, t, y_L, \gamma)$
$\}$
$\}$

The routine $PostCompute(\cdot)$ is the postprocessing step and is described below:

3. ***Post Processing****(Row Control):*

$PostCompute(M, s, t, y, \gamma)\{$

/* • As in $DFSCompute$ M is an $s \times t$ matrix of bits and we want to retrieve the bit $M[a,b]$, where the index a and b is hidden in the y and γ.

• $y[j]$ $(1 \leq j \leq t)$ is an array of t numbers (L_{th} row of the matrix y sent by user). $y[j]$ is a QNR if $j = b$ else it is a QR.

• $\gamma[j]$ $(1 \leq j \leq s)$ is an array of s numbers (γ sent by user). $\gamma[j]$ is a QNR if $j = a$ else it is a QR. */

1. Set for $1 \leq i \leq s$, $z[i] = \nu_i \left(\prod_{j=1}^{t} (y[j])^{M[i,j]} \right)$, where ν_i for each i is a QR chosen by DB uniformly at random.

/* Each $z[i]$ is a K-bit string. For $1 \leq j \leq K$, let $z[i,j]$ denote the j_{th} bit of $z[i]$. We require the string $z[a]$ */

2. Set $M'[i,j] = z[j,i]$ for $1 \leq j \leq s$, $1 \leq i \leq K$.

/* M' is an $K \times s$ matrix of bits. */

3. Set for $1 \leq i \leq K$, $\zeta[i] = \nu_i \left(\prod_{j=1}^{s} (\gamma[j])^{M'[i,j]} \right)$ where ν_i for each i is a QR chosen by DB uniformly at random.

4. Output the strings $\zeta[1], \ldots, \zeta[K]$. These are sent to the user.

$\}$

Reconstruction Phase and Correctness :

We show that from the output of $PostCompute(\cdot)$, it is possible to reconstruct the $(a_1, b_1)_{th}$ bit of matrix D i.e., i_{th} bit of database x. This will establish the correctness of protocol and also provide the method using which user can reconstruct the i_{th} bit of database x. (We assume that the user's query is properly formed.)

1. Suppose the call $PostCompute(M, s, t, y, \gamma)$ outputs the strings $\zeta[1], \ldots, \zeta[K]$ and the QNR's of y and γ are the b_{th} and a_{th} position respectively. The value of $\zeta[i]$ is a QR iff $M'[i,a]$ is 0, so it is possible to reconstruct the column vector

$M'[*, a]$ which is equal to the row vector $z[a, *] = z[a]$. Again $z[a]$ is a QR iff $M[a, b]$ is 0. Thus it is possible to find $M[a, b]$. So it is possible to get the bits at level $L - 1$. Now we use backward induction on the depth of recursion.

2. Suppose the call $DFSCompute(M, s, t, y_l, l)$ produces the set of matrices $M_1, \ldots, M_K$. By induction hypothesis it is possible to get the bits $M_i[a_{l+1}, b_{l+1}]$. We show it is possible to get $M[a_l, b_l]$ from these. From the construction of $z[a_l]$ we find that it is equal to $M_1[a_{l+1}, b_{l+1}], \ldots, M_K[a_{l+1}, b_{l+1}]$ and so it is possible to get $z[a_l]$. The quantity $z[a_l]$ is a QR iff $M[a_l, b_l]$ is 0. Thus we get the bit $M[a_l, b_l]$. This proves the induction. Hence the user can get the $(a_1, b_1)_{th}$ bit of all the K matrices D_κ passed in the routines $DFSCompute(M, s, t, y_1, 1)$. Thus user gets the bits $D_1[a_1, b_1], \ldots, D_K[a_1, b_1]$, which on concatenation gives the number $\psi(a_1, b_1)$ by which DB had replaced the $(a_1, b_1)_{th}$ bit of matrix D. Again $\psi(a_1, b_1)$ is a QR iff $D(a_1, b_1)$ is 0. Thus user is able to obtain the desired bit.

Privacy of user and database can be shown similarly as in the case of basic scheme. We omit the details due to lack of space.

Communication Complexity

Communication from user to DB is (1) a K-bit number N, (2) a $L \times t$ query matrix y of K-bit numbers and (3) an array of K-bit numbers of length t. Thus total communication from user to database DB in this scheme is $(1+(L+1)\cdot t)\cdot K$. The DB returns numbers computed at the end of $PostCompute(\cdot)$ routine. We analyze the tree structure formed from the computation process of DB. The Root of the computation tree is the matrix D formed from original database x. Now in preprocessing computation DB obtains K matrices D_κ's $(1 \leq \kappa \leq K)$ from the matrix D. Each of these K matrices becomes the child of the root. Thus root node, designated at level 0 has K children (all at level 1). The call of routine $DFSCompute(\cdot)$ at l_{th} level of recursion takes a matrix at l level as input and produces K matrices as output. Thus each of the nodes at level $l < L$ has K children. Matrices at level L are input to $PostCompute(\cdot)$ which produces K numbers of K-bit each, which are returned to user. Thus for each of the K^L matrices (leaf nodes of computation tree), user receives K^2 bits. Therefore the total communication from DB to user is K^{L+2} bits. Hence the communication complexity $C(n) = (1+(L+1)\cdot t+K^{(L+1)})\cdot K$ bits, where $t = n^{\frac{1}{L+1}}$. If we choose, $L = \sqrt{\frac{\log n}{\log K}} - 1$, then $C(n) = (1 + (\sqrt{\frac{\log n}{\log K}} + 1) \cdot 2^{\sqrt{\log n \cdot \log K}}) \cdot K$. Compare this with the communication complexity $O(2^{2\cdot\sqrt{\log n \cdot \log K}})$ obtained by Kushilevitz and Ostrovsky [7] for their PIR scheme. *Thus we have a single-round SPIR scheme with the communication complexity even smaller than the PIR scheme of [7].*

Even with the weaker assumption on security parameter, i.e., $K = n^\epsilon$ for some constant $\epsilon > 0$, we get a communication complexity $O(\frac{1}{\sqrt{\epsilon}} \cdot n^{\sqrt{\epsilon}+\epsilon}) = O(n^{\sqrt{\epsilon}+\epsilon})$. which is better than the communication complexity $(n^{2\cdot\sqrt{\epsilon}})$ obtained in [7] under the same assumption. If we take $K = \log^\epsilon n$, then we get the communication complexity of $O(\sqrt{\frac{\log n}{\epsilon \cdot \log\log n}} \cdot \log^\epsilon n \cdot 2^{\sqrt{\epsilon \cdot \log n \cdot \log\log n}})$.

Hence we have proved the following theorem :

Theorem 2. *For every $\epsilon > 0$, there exists a single-server single-round SPIR protocol with communication complexity $O(n^{\epsilon})$, where user privacy is based on QRA and database privacy is based on the XOR assumption.*

E Block Retrieval SPIR Scheme

In previous section we presented scheme for retrieving a bit from a database modeled as a array of bits. But a more realistic view of a database is to assume it partitioned into blocks rather than bits. We view database x as a array of records, each of size m, having n records in total. User wants to retrieve i_{th} record. Number of records n and number of bits in a record m determine L, as $L = \lceil \frac{\log n}{\log m} \rceil - 1$. The value of L determine the recursion depth of database computation.

For the purpose of DB computation database x held by DB is viewed as a stack of m matrices D_{μ} $(1 \leq \mu \leq m)$, where each matrix D_{μ} is an $s \times t$ matrix of bits and user wants to retrieve the bits $D_{\mu}(a_1, b_1)$. Now to retrieve the i_{th} record from database x, user generates a query following the protocol in our bit retrieval scheme, but taking the value of L defined as above. DB applies the query sent by user on all the m matrices in the stack, and send back to user the answer obtained for each of the matrices in the stack. As user can obtain i_{th} bit of each of the matrix, he will get the i_{th} record. Correctness and privacy of user and privacy of database follows from the underlying bit retrieval protocol.
The **Communication Complexity** in this scheme is $C(m, n) = (1 + (L + 1) \cdot n^{\frac{1}{L+1}} + m \cdot K^{L+1}) \cdot K$ Therefore, we proved following theorem:

Theorem 3. *There exist a block retrieval SPIR protocol with communication complexity linear in number of bits in the record m and polynomial in security parameter K, i.e., we have $O(m \cdot K^{L+2})$, where m is the number of bits in the record and $L = \lceil \frac{\log n}{\log m} \rceil - 1$, where user privacy is based on QRA and database privacy is based on the XOR assumption.*

Corollary 1. *For $n \leq m$, i.e., number of records not more than number of bits in any record, we get $L = 0$, and communication complexity: $C = (1 + n + m \cdot K) \cdot K < m \cdot (K + K)K$ i.e., $C = O(mK^2)$. For $m < n \leq m^2$), we get $L = 1$ and communication complexity $C = O(m \cdot K^3)$. In general, $n^{\frac{1}{L}} = m$, and $(L + 1) < K^(L + 1)$, thus communication complexity $C = O(m \cdot K^{L+2})$.*

F Conclusion

In this paper we have presented a single-server, single-round SPIR protocol. The communication complexity of the protocol can be made $O(n^{\epsilon})$ for any $\epsilon > 0$. Further the scheme has been extended to efficient block retrieval protocols. Some of the ideas used in the construction of SPIR protocol is based on the PIR protocol in [7]. In Mishra [9], it is shown that there exists PIR and SPIR schemes having communication complexity $O(K \log n)$ (where K is the security parameter and n is the size of the database) provided there exists probabilistic encryption schemes with certain desirable properties.

References

1. A. Ambainis. *An upper bound on the communication complexity of private information retrieval.* In Proc. of the 24th ICALP. Lecture Notes in Computer Science. vol. 1256. Springer-Verlag, New York, 1997, pp.401-407. 226
2. B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan. *Private information retrieval.* In Proceedings of 36th Annual Symposium on Foundation of Computer Science. IEEE. Milwaukee, Wisconsin, 23-25 October 1995, pp.41-50. *Journal version in JACM*, vol. 45(6), 1998, pp.965-981. 225, 226, 226, 228
3. B. Chor, and N. Gilboa. *Computationally private information retrieval* . In Proceedings of the 29th Annual Symposium on Theory of Computing, El Paso, Tex., May 4-6, 1997. ACM, New York, pp294-303. 226
4. David A. Cooper, and Kenneth P. Birman. *Preserving privacy in a network of mobile computers.* Proc. IEEE Symposium on Security and Privacy,1995, pp.26-38. 225
5. G. Di Crescenzo, T. Malkin, and R. Ostrovsky. *Single database private information retrieval implies oblivious transfer.* In Proceedings of EUROCRYPT'00, 2000. 228
6. Y. Gertner, Y. Ishai, E. Kushilevitz, and T. Malkin. *Protecting data privacy in private information retrieval schemes.* In Proceedings of the 30th Annual Symposium on Theory of Computing, Dallas, Tex., May 23-26, 1998. ACM, New York, pp.151-160. 226, 228
7. E. Kushilevitz, and R. Ostrovsky. *Replication is not needed : single database computationally-private information retrieval* . In Proceedings of 38th Annual Symposium on Foundation of Computer Science. IEEE Computer Society Press, Los Alamitos, Calif., 1997, pp. 364-373. 226, 226, 226, 226, 227, 227, 228, 228, 230, 231, 231, 232, 234, 234, 234, 235
8. E. Mann. *Private access to distributed information.* Master's thesis, Technion, Israel Institute of Technology, Hafia, 1998. 227, 228
9. S.K. Mishra. *On symmetrically private information retrieval.* MTech, Computer Science Dissertation Series, Indian Statistical Institute, Calcutta, 2000. Also available at Cryptology ePrint archive, http://eprint.iacr.org, 2000/041. 228, 228, 235