Replication Is Not Needed: Single Database, Computationally-Private Information Retrieval

(EXTENDED ABSTRACT)

Eyal Kushilevitz[*] Technion

Rafail Ostrovsky[†] Bellcore

Abstract

We establish the following, quite unexpected, result: replication of data for the computational Private Information Retrieval problem is not necessary. More specifically, based on the quadratic residuosity assumption, we present a *single database*, computationally-private information-retrieval scheme with $O(n^{\epsilon})$ communication complexity for any $\epsilon > 0$.

1 Introduction

1.1 Problem Statement and History

Private Information Retrieval (PIR) schemes allow a user to retrieve information from a database while maintaining the privacy of the queries from the database. More formally, we view the data as an n-bit string x from which the user wishes to obtain the bit x_i while keeping the index i private from the database. The notion of PIR schemes was introduced by Chor, Goldreich, Kushilevitz and Sudan [CGKS-95] in an *information theoretic* setting. This means that the queries asked by the user give no information whatsoever about i. In this setting they proved the following results:

- Every (information theoretic) PIR scheme with a single-database requires $\Omega(n)$ bits of communication (this matches the trivial upper bound in which the user just asks for a copy of the entire database in order to hide which particular bit it is interested in).
- A way to get around the above impossibility result so as to reduce the communication complexity to be sub-linear (in n) is by assuming that the data is **replicated** in several sites which are assumed not to communicate with one another. Indeed, [CGKS-95] show that if the data is replicated $k \geq 2$ times (i.e., in k replicated copies of the database) then there are PIR schemes with sub-linear communication complexity. For example, there is a scheme with communication complexity $O(n^{1/3})$ for two non-communicating database copies. Schemes with even smaller communication complexity for bigger k are also given in [CGKS-95] and some further improvements appear in [Amb-97]. In other words, replication of data seems to be a crucial ingredient for (information theoretic) PIR.

Recently, Chor and Gilboa [CG-97], Ostrovsky and Shoup [OS-97], and Itkis [I-96], considered the notion of *Computational PIR* (cPIR) schemes. In such schemes the databases are restricted to perform only polynomial-time computations and the privacy of the user's requests is relaxed so that the identity of i is only computationally hidden from the databases. Naturally, the privacy in this case has to rely on some intractability assumption. In this setting, the above mentioned works show upper bounds which are

[*]Dept. of Computer Science, Technion, Haifa, Israel. e-mail: eyalk@cs.technion.ac.il; http: // www.cs.technion.ac.il/ ~eyalk; Part of this research was done while visiting DIMACS center at Rutgers University. Supported by MANLAM Fund 120-857 and by the Fund for Promotion of Research at the Technion.

[†]Bell Communications Research, MCC 1C-365B, 445 South Street, Morristown, New Jersey 07960-6438. e-mail: rafail@bellcore.com.

much better than what is known for (information theoretic) PIR schemes. For example, in [CG-97] it is shown, based on the existence of one-way functions,[1] that for every $0 < c < 1$ there is a cPIR scheme for $k = 2$ databases with communication complexity $O(n^c)$. Again, all known cPIR schemes (as is the case with all PIR schemes) assume that the data is represented in *several* databases, that do not communicate with each another.

In this paper, we consider the following question:

IS THE ASSUMPTION OF HAVING SEVERAL NON-COMMUNICATING DATABASES REQUIRED FOR *cPIR* SCHEMES AS WELL ?

1.2 Our Main Result

Surprisingly, we show that in the computational setting one can get rid of the assumption about the replication of data which is in the heart of all previous (PIR and cPIR) solutions. More precisely, we prove the following:

MAIN THEOREM (Informal Statement) For every $c > 0$ there exists a `single`-database cPIR scheme with communication complexity $O(n^c)$, assuming the hardness of deciding *quadratic residuosity.*

Moreover, our scheme has the following additional properties:

1. The data is stored in the database in its plain form (i.e. the string x itself). Hence there is no need in pre-processing, storing auxiliary information, or coordination between different users. (In particular this allows at the same time retrieving information which do not require privacy and hence has a smaller communication complexity cost.)

2. The scheme is a single-round query-answer protocol (as opposed to multi-round protocols), which is the common communication pattern in the context of databases.

[1] A one-way function is a function that can be efficiently computed but cannot be inverted in polynomial time. One could make stronger assumptions, e.g., the existence of a function that cannot be inverted in time $2^{\sqrt{n}}$, and introduce a corresponding security parameter. We will do it later in the technical part of this paper but for simplicity we will avoid doing it in the introduction. In fact, the exact security of all cPIR schemes can be analyzed, similar to the approach taken in [L-96].

These additional properties are also shared by [CGKS-95, Amb-97, CG-97] whereas [OS-97, I-96] are multi-round schemes that use a special representation for the data.

Ostrovsky and Shoup [OS-97] consider the question of private *writing* into the database, both in the information-theoretic and computational setting, provided that there are several non-communicating sites. Note that in the single-database model (i.e. without replication) if the data is stored in its plain form then private writing is impossible (since the database can easily observe the changes in x).

On the other hand, if the data is stored in a single database and is encrypted, with a key that is held by the user (secretly from the database), then the model becomes the *oblivious RAM* model [G-87, Ost-90, GO-96]. (For more detailed discussion see [OS-97]). We point out however that there are some major differences between the two models; hence, transforming any of the oblivious RAM solutions into a cPIR scheme yields schemes that do not satisfy the above-mentioned features of PIR schemes. That is, in the oblivious RAM solutions the data is stored in a special *encrypted* form; it requires a pre-processing initialization step (for encrypting the data); it requires the user (the "cpu" in the terminology of RAM) to keep some state information (in particular the encryption key); and finally if there are several users it requires them to coordinate their acts so that each of them will have access to the encryption key and still this key must be hidden from the database. If the database obtains this key (e.g., by pretending to be a user) the privacy is lost. Thus, the *oblivious RAM* techniques do not provide a solution to the single-database computational PIR problem.

Finally, let us mention that the *quadratic residuosity assumption* (QRA) is a widely used assumption in cryptography. Starting from the pioneering work of Goldwasser and Micali [GM-84] and later in numerous other papers; for example, see [BBS-86, BC-86, BDMP-91].[2] It remains an important open problem to find a single database efficient cPIR scheme that relies on weaker intractability as-

[2] To the best of our knowledge, the fastest known algorithm for deciding the quadratic residuosity modulo N problem is to first factor N (and then the problem is easy). The best algorithm for factoring known to-date runs in (heuristic) time $e^{\tilde{O}(N^{1/3})}$ (for further details see an excellent survey by Odlyzko [Odl-95]).

sumptions.

Organization: In Section 2 we provide some necessary definitions. Section 3 includes a simple scheme whose communication complexity is still not as small as guaranteed above. Later in Section 4 we use this simple scheme to construct a more involved scheme with the desired communication complexity. Finally, in Section 5 we show several extensions and generalizations of our result.

2 Preliminaries

In this section we give some definitions and facts that are used in this paper. In Section 2.1 we include some facts from number-theory and state the quadratic residuosity assumption, and in Section 2.2 we define computational PIR. We provide formal definitions of this notion which are essential for later providing a rigorous proof of privacy for our schemes. (The reader may prefer to read first the description of the schemes (Sections 3 and 4) and return to the definitions before reading the proofs of privacy.)

2.1 Quadratic Residuosity Assumption

We shall use in this work the intractability of the **Quadratic Residuosity** problem. This problem was first used in a cryptographic setting by Goldwasser and Micali [GM-84], and since then has found many other (cryptographic) applications.[3] Below we recall this problem. (The reader is referred, for example, to [BS-96] for a good reference on number theory and to [M-90] for a reference on the quadratic residuosity problem.)

Let N be a natural number. Define

$$Z_N^\star = \{x | 1 \leq x \leq N, gcd(N,x) = 1\}.$$

The quadratic residuosity predicate is defined as follows: $Q_N(y) = 0$ if $\exists w \in Z_N^\star$ such that $w^2 = y \bmod N$ and $Q_N(y) = 1$ otherwise. We say that y is QR (quadratic residue) mod N if $Q_N(y) = 0$ (i.e. y is a "square") and we say that y is a QNR (quadratic non-residue) if $Q_N(y) = 1$ (i.e., y is a "non-square"). The problem is considered "hardest" when N is a product of two distinct primes of equal length $k/2$; thus the "hard set" (indexed by k) is

$$H_k = \{N | N = p_1 \cdot p_2 \textit{ where } p_1, p_2 \textit{ are } k/2\textit{-bit primes}\}$$

If the factorization of $N \in H_k$ is known, computing $Q_N(y)$ can be done in $O(|N|^3)$ time. Let $\left(\frac{y}{N}\right)$ denote the Jacobi symbol. Recall that for all N (i.e., even for $N \in H_k$) the value of $\left(\frac{y}{N}\right)$ can be computed in time polynomial in $|N|$ even without knowing the factorization of N. For $N \in H_k$, if $\left(\frac{y}{N}\right) = -1$ then y is always a QNR while among the y's such that $\left(\frac{y}{N}\right) = +1$ exactly half are QNRs and half are QRs. Thus we consider only $N \in H_k$ and only y's such that $\left(\frac{y}{N}\right) = +1$. Thus, define $Z_N^{+1} \triangleq \{y \in Z_N^\star | \left(\frac{y}{N}\right) = 1\}$. For any x, y in Z_N^{+1} their product xy is a QNR if exactly one of them is a QNR; that is, $Q_N(xy) = Q_N(x) \oplus Q_N(y)$. Finally, note that to pick a random QR in Z_N^{+1} we just need to pick a random $r \in Z_N^\star$ and compute r^2 (for doing this one clearly does not need to know the factorization of N).

We are now ready to state the Quadratic Residuosity Assumption,[4] which informally states that there is no family of polynomial-size circuits for computing the predicate $Q_N(y)$ that can do significantly better than "guessing". Formally,

Quadratic Residuosity Assumption (QRA): For every constant c , and every family of polynomial-size circuits $C_k(\cdot,\cdot)$, there exists an integer K such that for all $k > K$

$$\mathbf{Prob}_{N \in_R H_k;\ y \in_R Z_N^{+1}} \left(C_k(N,y) = Q_N(y)\right) < \frac{1}{2} + \frac{1}{k^c} \ ,$$

where $N \in_R H_k;\ y \in_R Z_N^{+1}$ denotes the experiment of first drawing two $k/2$-bit primes p_1, p_2 and compute $N = p_1 \cdot p_2$, and then drawing y uniformly at random from Z_N^{+1}.

In [GM-84] it is shown that the quadratic residuosity problem is random self-reducible, and that even given polynomially many (in $|N|$) quadratic non-residues $y_1, \ldots, y_{poly(|N|)}$ it is still as difficult to compute $Q_N(x)$ as without y's.

[3] The problem was originally considered (as one of the four main arithmetic problems) by Gauss [G-01].

[4] This is a *non-uniform* version of the assumption. A uniform version (which is a weaker assumption) refers to algorithms instead of families of circuits. We will provide the details of the uniform definition in the full version of this paper.

2.2 Computational PIR

We are now ready to define a single database computational PIR. We shall follow similar definitions from [CG-97, OS-97] and use the terminology of interactive Turing Machines [GMR-85]. A computational PIR scheme is a protocol for two players, a user $\mathcal{U}$ and a database $\mathcal{DB}$, which are limited to probabilistic polynomial-time computations. They are formally defined as follows:

- The "database", $\mathcal{DB}$, is a probabilistic polynomial-time interactive Turing machine with a read-only *input* tape, a read-only *random* tape, a *work*-tape, a write-only *output* tape and two *communication*-tapes: a write-only communication tape for sending massages to $\mathcal{U}$ and a read-only communication tape for receiving messages from $\mathcal{U}$.
- The "user", $\mathcal{U}$, is a probabilistic polynomial-time interactive Turing machine with a read-only *input* tape, a read-only *random* tape, a *work* tape, a write-only *output* tape and two *communication* tapes: a write-only communication tape to send messages to $\mathcal{DB}$ and a read-only communication tape to read messaged from $\mathcal{DB}$.

A computational PIR (cPIR) protocol allows the user to "read" (i.e., retrieve) a bit x_i from the database string x. Formally, the database, $\mathcal{DB}$, is given on its input tape a security parameter, 1^k, and an n-bit input string x. The user, $\mathcal{U}$, has on its input tape the same (as $\mathcal{DB}$) security parameter 1^k and an instruction "read(i)" where $1 \leq i \leq n$. The scheme proceeds as follows:

1. The user, $\mathcal{U}$, performs a polynomial time computation (utilizing its input tape, its random tape and its work tape) and then writes a single message ("query") $q(i)$ on its write-only communication tape (i.e., it sends $q(i)$ to the database).
2. The database reads its read-only communication tape (to get the query $q(i)$ from the user), performs a polynomial time computation (utilizing his input tape its work tape and its random tape) and then writes a single message (an "answer") $a(x, q(i))$ on its write-only communication tape.
3. The user reads the "answer" $a(x, q(i))$ sent by the database.[5] Then, $\mathcal{U}$ performs a polynomial-time computation using the $\mathcal{DB}$ answer and its work tape (which contains information stored from Step 1). At the end of this computation $\mathcal{U}$ outputs a single bit on its output tape (which should be x_i).

The communication complexity of the cPIR problem is measured as a function of n (the length of the data string x) and the security parameter k. We say that the cPIR protocol has communication complexity $CC(n,k)$ if for any data string x of length n and security parameter k, and any request i, it is always the case that $|q(i)| + |a(x,q(i))| \leq CC(n,k)$. A cPIR scheme must satisfy both *correctness* and *privacy* constraints:

- **Correctness** – For any input length n, and for any n-bit input string x given to $\mathcal{DB}$ (i.e., the content of the database), for any security parameter 1^k and for any user request instruction "read(i)", the output of $\mathcal{U}$ must be x_i.
- **Privacy** – Informally, computational privacy means that the database cannot distinguish, with non-negligible probability, the difference between any two distributions of requests $q(i)$ and $q(j)$. Formally, for all constants c, for all database length n, for any two $1 \leq i, j \leq n$, and for all polynomial-size families of circuits $C_k(\cdot)$ there exists an integer K such that for all $k > K$

$$\begin{aligned} | \quad & \left[\mathbf{Prob}_{\textit{coins-of-}\mathcal{U}}(C_k(q(i)) = 1)\right] \\ - \quad & \left[\mathbf{Prob}_{\textit{coins-of-}\mathcal{U}}(C_k(q(j)) = 1)\right] \quad | \\ & < \frac{1}{(\max(k,n))^c} \stackrel{\triangle}{=} \epsilon(k,n) \ . \end{aligned}$$

  (In our context usually $k \ll n$; e.g., $k = n^\alpha$ for some $0 < \alpha < 1$.)

REMARKS: Notice, again, that the above definition is *non-uniform*.[6] Additionally, note that our definition of computational privacy, as in [CG-97, OS-97], is defined in terms of indistinguishability of

[5] A more general definition may allow a more involved interactive protocol between $\mathcal{U}$ and $\mathcal{DB}$, consisting of polynomial (in k and n) number of rounds, where each "round" consists of a single message sent from $\mathcal{U}$ to $\mathcal{DB}$ and a single message back to $\mathcal{U}$. However, since all our protocols are single-round protocols, we shall use this simpler formulation.

[6] We postpone the uniform definitions until the full version of the paper, but remark that all our results can be extended to the uniform model as well.

user's requests. A computationally equivalent definition could consider indistinguishability of two *sequences* of requests; i.e., where the user has a sequence $i_1, \ldots, i_t$ of indices it wishes to retrieve from x. In addition, as in the case of "probabilistic encryption", we can also define computational privacy in terms of "semantic privacy" (see [GM-84]), where, informally, semantic privacy means that whatever a polynomially-bounded adversary A can compute from the queries sequence, it can compute without it. As with probabilistic encryption, one can show that computational privacy in terms of indistinguishability implies semantic privacy as well.

3 The Basic Scheme

In this section we present a single database cPIR scheme whose communication complexity is $O(n^{0.5+c})$, for any $c > 0$. This is not as low as promised in our main theorem, but note that it is already better than the $\Omega(n)$ lower bound for the information theoretic case [CGKS-95]. In the next section we will use this scheme as a starting point for our more efficient schemes.

For describing our basic scheme we view the database x as a $s \times t$ matrix of bits, denoted M. The user, is interested in retrieving privately the bit x_i of the database which is the (a, b) entry of the matrix M. The basic scheme works as follows:

1. The user starts by picking at random a k-bit number $N \in H_k$ (i.e., it picks at random two $k/2$-bit primes and multiplies them). The user sends N to $\mathcal{DB}$ (but keeps its factorization secret).

2. The user chooses uniformly at random t numbers $y_1, \ldots, y_t \in Z_N^{+1}$ such that y_b is a QNR and y_j, for $j \neq b$, is a QR. It sends these t numbers to $\mathcal{DB}$ (total of $t \cdot k$ bits).

3. The database, $\mathcal{DB}$, computes for every row r a number $z_r \in Z_N^\star$ as follows: It first computes (in $Z_N^\star$)
$$w_{r,j} = \begin{cases} y_j^2 & \text{if } M_{r,j} = 0 \\ y_j & \text{if } M_{r,j} = 1 \end{cases}$$
and then it computes
$$z_r = \prod_{j=1}^{t} w_{r,j}.$$
The observation here is that if $j \neq b$ then $W_{r,j}$ is always a QR, while if $j = b$ then $W_{r,j}$ is QR iff $M_{r,j} = 0$ (and it is a QNR otherwise). Therefore, z_r is a QR iff $M_{r,b} = 0$ (and it is a QNR otherwise).

4. $\mathcal{DB}$ sends $z_1, \ldots, z_s$ to the user (total of $s \cdot k$ bits).

5. The user considers only the number z_a, corresponding to the row of M which contains the bit it is interested in. This number is a QR iff $M_{a,b} = 0$ (and it is a QNR otherwise). Since $\mathcal{U}$ knows the factorization of the number N it can efficiently check whether z_a is a QR and by this retrieve the bit $M_{a,b}$.

Correctness: Follows immediately from the description of the scheme. ■

Communication Complexity: The communication in this scheme consists of $s + t + 1$ k-bit numbers $(N, y_1, \ldots, y_t, z_1, \ldots, z_s)$. Pick $s = t = \sqrt{n}$ and the communication complexity is:
$$(2\sqrt{n} + 1) \cdot k.$$
Therefore, even if we use the weak assumption that the security parameter, k, equals n^c, for some constant $c > 0$, we get a communication complexity of $n^{\frac{1}{2}+c}$. This is already better than the $\Omega(n)$ lower bound proved in [CGKS-95] for the single-database case in the information theoretic setting! ■

Privacy: The proof is by contradiction. Suppose, towards a contradiction, that for some indices i and i' the database can distinguish the queries on index i from the queries on index i'. We will use this assumption to construct a circuit to compute the quadratic residuosity predicate. In our matrix notation we denote $i = (a, b)$ and $i' = (a', b')$. Obviously, by the description of the protocol $b \neq b'$ (as otherwise the protocol works in an identical way and there is no way to distinguish). The fact that the database can distinguish these two indices implies that there exists an algorithm (more precisely, a family of polynomial-time circuits), B, such that if B gets queries from the distribution of queries generated by the user on index i then B outputs 1 with some probability p, while if B gets queries from the distribution generated on index i' then B outputs 1 with probability, say, $p + \epsilon$.

By the construction, in the first case the distribution consists of a number N uniformly picked from H_k and then t numbers from Z_N^{+1} which are all QRs except for the number in position b which is a QNR. The second distribution is similar except that the QNR is located in position b'.

We now describe (a circuit) C that, on input N, y (chosen according to the "hard distribution"; i.e., $N \in_R H_k$; $y \in_R Z_N^{+1}$) computes the quadratic residuosity predicate with probability at least $1/2 + \epsilon/2$. The idea is to construct a sequence as expected by the distinguisher B.

1. Pick $t-2$ random QRs and place them in all the positions except b and b'.

2. Pick at random one of the positions b or b'. Place y in this position and another random QR in the other.

3. Run B on the above sequence. If the chosen position is b' output the same value as B; if the chosen position is b flip B's output.

Let us first compute the probability that C outputs 1 on input N, y such that y is a QR modulo N. In this case, no matter what position is chosen in Step 2 the input to B is a sequence of t random QRs in Z_N^{+1}. On such an input there is some probability q that B outputs 1. The probability that C outputs 1 in this case is $1/2 \cdot q + 1/2 \cdot (1-q) = 1/2$. Now we compute the probability that C outputs 1 on input N, y such that y is a QNR modulo N. In this case the sequence generated contains a single QNR either in position b or in position b'. By the assumptions on B the probability that C outputs 1 in this case is $1/2 \cdot (p+\epsilon) + 1/2 \cdot (1-p) = 1/2 + \epsilon/2$. Note that this is slightly different than what we need (since on a QR we are supposed to output 1 with probability less than $1/2$). This is easily solved by adding a first step in which with probability $\epsilon/4$ we output 0 and stop. ■

To conclude, we have just proved the following theorem:

Theorem 1: Under the QRA, for every $c > 0$, there exists a cPIR scheme with communication complexity $O(n^{0.5+c})$.

4 The Recursive Scheme

In this section we define a more complicated computational-PIR scheme, with lower communication complexity. We emphasize that the new (recursive) scheme, can still be implemented in a single round (that is, the user sends a query, gets an answer, and retrieve from it the desired bit).

The main observation is as follows: consider Step 4 of the basic scheme. In this step the user sends s k-bit numbers $z_1, \ldots, z_s$ to the user. The user is only interested in one of these numbers, z_a, however it cannot tell this information (the index a) to $\mathcal{DB}$ as this will violate the privacy constraint. It is therefore natural to view these k numbers as a $s \cdot k$ bit string and let the user and database use a cPIR scheme, so that the user will get the k bits of this string which are of interest for him. More formally, denote by $\mathcal{S}_1$ the basic scheme defined above. Now, the scheme $\mathcal{S}_\ell$, is recursively defined as follows. Denote by n_ℓ the number of bits we have when executing $\mathcal{S}_\ell$ (on the top level, denoted L, we have $n_L = n$). Let t_ℓ be the number of columns in the matrix used by the scheme $\mathcal{S}_\ell$ to represent the string (and $s_\ell = n_\ell / t_\ell$). We obtain $\mathcal{S}_\ell$ by replacing Step 4 of the basic scheme by:

4*. $\mathcal{U}$ and $\mathcal{DB}$ execute k times the scheme $\mathcal{S}_{\ell-1}$. In each of these executions the user gets one of the bits of z_a out of a string of length $n_{\ell-1} = k \cdot s_\ell$ held by the database. After these k executions, the user has the number z_a that allows him reconstructing the desired bit, as in Step 5 of the basic scheme.

Note that although the string from which we retrieve bits becomes smaller in the recursion, we still need to use the *same* security parameter 1^k given as an input (typically, k will be chosen as a function of n, the size of the original database). On the other hand, Step 1 can be executed only once, and we can use the same N in all levels of the recursion.

The next step is to observe that this new version of Step 4 can be further improved. This is because when applying k times the scheme $\mathcal{S}_{\ell-1}$ for retrieving the number z_a out of the information held by $\mathcal{DB}$ it is clear that in the first use of $\mathcal{S}_{\ell-1}$ the user will retrieve the most significant bit of z_a, in the second use of $\mathcal{S}_{\ell-1}$ it will retrieve the second most significant bit of z_a and so on. Hence in each of these executions the database can be smaller in a factor of k. Thus we get:

4**. $\mathcal{U}$ and $\mathcal{DB}$ execute k times the scheme $\mathcal{S}_{\ell-1}$. In the d-th execution the user gets the d-th most significant bit of z_a out of a string of length $n_{\ell-1} = s_\ell$ held by the database (that contains the d-th bit of each of $z_1, \ldots, z_{s_\ell}$). After these k executions, the user has the number z_a that allows him reconstructing the desired bit, as in Step 5 of the basic scheme.

Finally, if we look at any level j of the recursive scheme, then in all executions of $\mathcal{S}_j$ we are interested in the *same* index of the strings from which we retrieve (this index depends on i, the index of x retrieved in the upper level of the scheme, and on j). Therefore we also make the following modification:

- For each level j of the recursion, $\mathcal{U}$ sends a single query (consisting of t_j numbers). This query serves in *all* invocations of $\mathcal{S}_j$ as the message of Step 2 of the scheme.

Correctness: Follows immediately from the description of the scheme. ■

Communication Complexity: First, we choose, for every ℓ, the same value $t_\ell = n^{1/(L+1)}$ as our parameter. This implies, that if on the top level we use the scheme $\mathcal{S}_L$ on the database of size $n_L = n$ then by induction (on $\ell = L, L-1, \ldots, 1$):

$$n_\ell = n^{\frac{\ell+1}{L+1}} \ .$$

Next, if we look at the recursion and "open" it then we observe that, in the upper level, $\mathcal{U}$ sends a query to $\mathcal{DB}$ which prepares k strings based on this query (but does not send any bit); then, in the next level, $\mathcal{U}$ again sends a query, based on it $\mathcal{DB}$ prepares from each string it had k new strings (all together k^2 strings) and so on. Finally in the last level of the recursion the user again sends a query based on which the database prepares k new strings for each string it had in the previous level (all together k^L strings of length $n^{1/(L+1)}$). With this view we already see that even the recursive scheme can be implemented as a single query-answer round scheme. It is also quite simple now to analyze the communication complexity: for each of the L levels of the recursion the user sends $n^{1/(L+1)}$ numbers of k bits each. The database replies only in the last level when it has to send k^L strings of length $n^{1/(L+1)}$. All together, the communication complexity is

$$n^{1/(L+1)} \cdot (k^L + L \cdot k).$$

If we use, the scheme $\mathcal{S}_L$ for $L = O(\sqrt{\log n / \log k})$ (this value of L makes $n^{\frac{1}{L+1}}$ and k^L equal) we get a complexity of $2^{O(\sqrt{\log n \cdot \log k})}$. For example, if $k = n^c$ we get complexity of $n^{O(\sqrt{c})}$ and if $k = \log^c n$ we get complexity[7] of $2^{O(\sqrt{\log n \log\log n})}$. ■

Privacy: The proof is similar to the proof of the basic scheme although it is slightly more delicate. Again, we assume towards a contradiction that there exists a distinguisher which for some i and i' can distinguish the distribution of queries $q(i)$ from the distribution of queries $q(i')$ with ϵ probability. If we "open" the recursion, as is done above, then we observe that the query sent by the user is just a number N followed by a (fixed-length) sequence of numbers $z_1, z_2, \ldots, z_\beta$ from Z_N^{+1} (by the above calculations $\beta = L \cdot n^{1/(L+1)}$). For an index i denote by I_i the subset of $\{1, \ldots, \beta\}$ of all positions that contain QNRs (by the protocol, for every index i the subset I_i is fixed and it is independent of the random choices made by the user). Moreover, as in the previous proof, i and i' must be such that $I_i \neq I_{i'}$ (as otherwise, if in all steps of the recursion i and i' belong to the same column of the matrix, then by the definition of the protocol the user behaves the same on both i and i' and they are indistinguishable).

As before, we denote by B the distinguisher between $q(i)$ and $q(i')$. That is, B is a circuit such that if B gets queries from the distribution of queries generated by the user on index i then B outputs 1 with some probability p, while if B gets queries from the distribution generated on index i' then B outputs 1 with probability, say, $p + \epsilon$. We construct a circuit C to compute the quadratic residuosity predicate. C on input N, y works as follows:

1. Pick at random one of the subsets I_i or $I_{i'}$. In every position not in the chosen set put a random QR from Z_N^{+1}. In every position which in the

[7] As mentioned, the currently best known algorithm for solving the quadratic residuosity problem runs in time $2^{\tilde{O}(N^{1/3})}$. Hence, it might be reasonable to use $k = \log^c n$, for sufficiently large constant c, as a security parameter. The standard assumption, that the problem cannot be solved in polynomial time, allows using only $k = n^c$, for arbitrarily small constant c.

chosen set put the product of y and a random QR.

2. Run B on the above sequence. If the chosen set is $I_{i'}$ output the same value as B; if the chosen position is I_i flip B's output.

The observation is that if y is a QR then the whole sequence consists of β numbers which are all QRs. On the other hand, if y is a QNR then the resulted sequence is distributed as are the queries in $q(i)$ or $q(i')$ (depending on the choice made at Step 1). With this observation the calculation of probabilities proceeds exactly as in the proof for the basic scheme. ■

To conclude, we have just proved the following theorem:

Theorem 2: Under the QRA, for every $c > 0$, there exists a cPIR scheme with communication complexity $O(n^c)$.

5 Extensions and Generalizations

In this section we briefly mention several extensions of the PIR problem (we defer the details to the full version of the paper). In all these extensions PIR schemes (and more specifically single-database PIR schemes, as the one provided by the current work) can serve as building blocks for more powerful protocols.

5.1 Additional Security for the User

Consider, for example, the site that maintains all the US patents and suppose that some user wishes to read a certain patent without the database getting any information about the identity of the patent that the user retrieves. Such a user can use a PIR scheme for this operation. However, a malicious database operator can mount the following attack to infer whether the user is after the j-th patent in the database: the database operator can replace the j-th patent/entry in its database with "garbage" and then, after executing the PIR scheme, see if the user "protests". In this case, the database operator can deduce that the j-th patent was the patent that the user wanted to retrieve.

This experimentation attack can be defeated if the user can obtain a certificate that the database provided wrong information, which can then be used to "sue" the database operator. To this end, the database can sign (using a variety of representation-efficient signatures) his messages, which can then be used to prove the database misbehavior in case of unexpected data.

In addition, In case when the database data comes from a trusted third party, the database entries can be augmented, in a natural way, with an authentication mechanism, where the user can read not only the entry but the authentication as well.

Moreover, the database operator can publish (say, in the New-York Times) a root of a commitment tree (based on cryptographic hash-functions), and then the user can retrieve (using PIR schemes), not just an entry, but an entire (logarithmic length) path from the root to the desired entry, where again the database signs all the messages that it sends to the user. Thus, the user gets a communication-efficient certificate that the data is in correspondence with the committed value, or a certificate that the database has either committed to a wrong data or mis-behaved during the protocol execution. Notice that all this can be done in one round.

5.2 Additional Security for the Database

In a PIR scheme, the database should have no idea which bit (or a sequence of bits) the user retrieves. Yet, if this is a financial enterprise, it also wishes to make sure that the user does not get, say two entries, for the price of one. This question was originally posed by Gertner, Ishai, Kushilevitz and Malkin [GIKM-97] mainly for information-theoretic PIR (with multiple databases), but the same concern applies in a single database computation setting as well. We remark that in the single-database computational-PIR setting we can employ zero-knowledge techniques to handle this problem as well. In particular, our main protocol can be modified in such a way that when the user follows the protocol as prescribed it gets only a single bit, and then the user can prove to the database, in a zero-knowledge fashion that his query is well-formed (e.g., it has the "correct" number of QRs and QNRs etc). The communication complexity of such a protocol is proportional to the original query size plus the communication complexity of a zero-knowledge proof, which is polynomially related to the query size (both for interactive and non-interactive zero-knowledge). Since our protocol requires $O(n^{\epsilon})$ bits of communication for arbitrary $\epsilon > 0$, the resulting protocol also needs only $O(n^{\epsilon'})$ communication.

5.3 Membership Queries

Suppose that the database contains n ℓ-bit "strings" $s_1, \ldots, s_n$. The user has some string s and he wishes to find an address i such that $s_i = s$ or to learn that such an address does not exist, without revealing to the database the value of s. This problem is a generalization of the PIR problem because, given x, the database can create a list of all entries j such that $x_j = 1$ (each such entry is a string of length $\ell = \log n$). Then, the user can retrieve the bit x_i by checking if i is one of these strings.

This problem can be solved by appropriately combining the use of *perfect hash-functions* to hash the strings into a hash table in a 1-1 manner and PIR schemes for reading values from this hash-table. In particular, note that the database can choose a hash function h that is 1-1 on the strings $s_1, \ldots, s_n$ and send its name (which is "short") to the user. Now the strings can be stored indexed by their hash-value, hence reducing the problem to PIR. We remark that this problem, and some extensions of it, was also independently studied and solved (using the same techniques) by Chor, Gilboa and Naor [CGN-97].

6 Conclusions and Open Problems

We have established that in the computational setting, replication of data is not necessary in order to retrieve, in a private and communication-efficient manner, information from a database. With the additional security for the database (i.e., where the user gets only one bit out of n bits) the PIR model can be re-formulated as one-out-of-n Oblivious Transfer protocol, a generalization of the well-known one-out-of-two Oblivious Transfer ($\binom{2}{1}$-OT) primitive [R-81, EGL-85] which plays an important role in cryptographic protocol design. Hence, we hope that our single-database computational PIR will be useful for the design of other cryptographic protocols.

Many problems remain open. How do we reduce the intractability assumptions? Impagliazzo and Rudich [IR-89] show that implementing oblivious transfer based on general one-way functions (i.e. without trapdoor) is hard to do using black-box reductions. However, we do not know how to achieve communication-efficient cPIR even assuming general one-way trapdoor permutations (while standard $\binom{2}{1}$-OT can be implemented based on any trapdoor one-way permutations [GMW-87]). How can this be done in cPIR setting? Basing cPIR solutions on other algebraic problems could also be of interest; for example, based on our paper, Man [M-97] shows how to replace the quadratic residuosity problem in our protocol by the shortest lattice vector problem. Can one base it on other algebraic assumptions? Additionally, even assuming that our security parameter is poly-logarithmic in n, we only achieve $2^{O(\sqrt{\log n \log\log n})}$ communication complexity. It seems plausible that, assuming poly-logarithmic security parameter, one can achieve poly-logarithmic communication complexity, but this also remains open.

Acknowledgments

We are grateful to Silvio Micali for posing to us the question of additional security for the user and to Prabhakar Raghavan for posing the question of membership queries.

We wish to thank the management of Harmon Cove Towers for the availability of the steam room; some good ideas started there.

References

[Amb-97] A. Ambainis. Upper bound on the communication complexity of private information retrieval. *Proc. of 24th ICALP*, Springer, Lecture Notes in Computer Science, Vol. 1256, pages 401–407, 1997.

[BC-86] G. Brassard, and C. Crepeau. Non-Transitive Transfer of Confidence: A Perfect Zero-Knowledge Interactive Protocol for SAT and Beyond. *Proc. of 27th FOCS*, pp. 188-195, 1986.

[BS-96] E. Bach, and J. Shallit. Algorithmic Number Theory (Volume 1). MIT Press, 1996.

[BDMP-91] M. Blum, A. De-Santis, S. Micali, and G. Persiano. Noninteractive Zero-

Knowledge. *SIAM J. on Computing*, Vol. 20, pp. 1084–1118, 1991.

[BBS-86] L. Blum, M. Blum, and M. Shub. A Simple Unpredictable Pseudo-Random Number Generator. *SICOMP*, Vol. 15, pp. 364–383, 1986.

[CG-97] B. Chor, and N. Gilboa. Computationally Private Information Retrieval. In *29th STOC*, pp. 304–313, 1997.

[CGN-97] B. Chor, N. Gilboa, and M. Naor. Private Information Retrieval by Keywords. TR CS0917, Department of Computer Science, Technion, 1997.

[CGKS-95] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan. Private Information Retrieval. In *Proc. 36th Annual IEEE Symp. Foundations Comp. Sci.*, pp. 41–50, 1995.

[EGL-85] S. Even, O. Goldreich and A. Lempel, *A Randomized Protocol for Signing Contracts*, Comm. of ACM vol. 28, 1985 pp. 637-647.

[G-01] C. F. Gauss Disquisitiones arithmeticae, 1801.

[GIKM-97] Y. Gertner, Y. Ishai, E. Kushilevitz, and T. Malkin. Protecting Data Privacy in Private Information Retrieval Schemes. Manuscript, 1997.

[G-87] O. Goldreich. Towards a Theory of Software Protection and Simulation by Oblivious RAMs. *STOC '87*.

[GMW-87] O. Goldreich, S. Micali and A. Wigderson How to Play any Mental Game. *STOC '87*.

[GO-96] O. Goldreich, and R. Ostrovsky. Software Protection and Simulation by Oblivious RAMs. *JACM*, 1996.

[GM-84] S. Goldwasser and S. Micali. Probabilistic Encryption *Journal of Computer and systems sciences* 28, 270-299, 1984.

[GMR-85] S. Goldwasser, S. Micali and C. Rackoff. The Knowledge Complexity of Interactive Proof-Systems, *SIAM J. Comput.* 18 (1989), pp. 186-208; (also in STOC 85, pp. 291-304.)

[IR-89] R. Impagliazzo and S. Rudich. On the Limitations of certain One-Way Permutations *STOC '89*, pp 44-61, 1989.

[I-96] G. Itkis. Personal communication, via O. Goldreich, June 1996.

[L-96] M. Luby. Pseudorandomness and Cryptographic Applications, Princeton Computer Science Notes, D. Hanson and R. Tarjan Editors, 1996.

[M-97] E. Man. Personal communication.

[M-90] K. McCurley. Odds and Ends from Cryptology and Computational Number Theory. in *Cryptography and Computational Number Theory*, C. Pomerance ed., AMS Proceedings of Symposia in Applied Mathematics, Vol 42, pp. 145-166, 1990.

[Odl-95] A. Odlyzko. The future of Integer Factorization. *CryptoBytes*, Vol. 1, No. 2, pp. 5-12, 1995. On-line version in **http://www.research.att.com/~amo/doc/future.of.factoring.ps**

[Ost-90] R. Ostrovsky. Software protection and simulation on oblivious RAMs. M.I.T. Ph. D. Thesis in Computer Science, June 1992.

[OS-97] R. Ostrovsky, V. Shoup Private Information Storage. In *29th STOC*, pp. 294–303, 1997.

[R-81] M. O. Rabin. How to exchange secrets by oblivious transfer. Technical Report TR-81, Harvard University, 1981.