# A New Private Information Retrieving Protocol With Low Computation Complexity *

Sergey Bezzateev
bsv@aanet.ru
with
Alexandra Afanasyeva and Vladimir Balakirsky

Saint Petersburg State University of Aerospace Instrumentation

Russia

Theory Days at Saka
October 25-27, 2013
Saka, Estonia

# Outline

- Introduction
- Overview of previous schemes
- New scheme description
- Short numerical example
- Competitive analysis
- Conclusion

# Anonymity problems

1. Who?
2. Wherefrom?
3. What?

Absolute anonymity isn't practically achievable, online or offline.
("Ten Immutable Laws Of Security from Microsoft , version 2.0", Law 9)

# Anonymity. Possible Solutions

1. Who? Blind Signature
2. Wherefrom? Onion Protocol
3. **What? Private Information Retrieval Protocol**

The private information retrieval (PIR) concept was proposed by Chor, Goldreich, Kushilevitz and Sudan in 1995.
Authors first considered the problem of anonymity between data owner and data consumer, from the point of view of the user's security.

**Problem:** user would like receive some data from database without revealing its interest to database owner.

[B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan. Private information retrieval. In Proc. 36th Annual IEEE Symp.Foundation Comp. Sci., pp. 41-50, 1995.]

# Simple but not fine decision

Copy whole database,
but...
this leads to a large communication cost.

Server holds $n$-bit string $\boldsymbol{x} = (x_1 x_2 \ldots x_i \ldots x_n)$,
User wishes to retrieve $x_i$ and keeps $i$ private, without asking all $n$-bits.

Two parts of PIR problem by attacker model considered.
1. **Information-theoretic approach.**
Attacker has unlimited computation power, and the security of system is based on insufficiency of information received by attacker.
It was proved by Chor, Goldreich, Kushilevitz and Sudan in 1995 that there is no solution of PIR problem in information-theoretic approach with communication less then $n$ bit without replication of database.
**Modified problem statement:**
- $r$ servers hold replicas of $n$-bit string $\boldsymbol{x} = (x_1 x_2 \dots x_i \dots x_n)$,
- user wishes to retrieve $x_i$ and keep $i$ private, without asking all $n$-bits.

Servers can't interact, and user can send different requests to different servers.
**Solution:**
- During pre-processing database of $n$-bit length increase to $N$ bit.

$\boldsymbol{x} = (x_1 x_2 \dots x_i \dots x_n) \rightarrow (y_1 y_2 \dots y_N)$,
- $r$ copies of pre-processed database is stored on $r$ servers.

User asks result of some function $f$ of $(y_1 y_2 \dots y_N)$ from each server and calculate value $x_i$, that he is interested.
No one server using its own request can receive any information about calculated value $x_i$ and its index $i$.

2.**Computational approach**
Attacker is restricted to perform only polynomial-time computations and the privacy is based on the fact that requested position $i$ is computationally hidden from attacker.
Under polynomial-time attacker model single-server solutions were proposed.
**Solution:**
User has to construct function which is calculated from all database and to find the value and position interested to the user anyone should to know some secret value, or he has to solve some hard problem. As we assume that attacker is polynomial-time restricted, so he can not solve it.

# Complexity of PIR schemes

Complexity of PIR schemes includes two components:

1. **Computation complexity**
   - Server's computational complexity. Server's costs for calculating answer on the user's query by whole database.
   - User's computational complexity. User's cost for calculation bit $x_i$ from server's answer.
2. **Communication complexity**
   - Server's communication complexity. Network overhead from server to user.
   - User's communication complexity. Network overhead from user to server.

**Database transformation with monomials.[1]**

For any index number $i \in [0, N]$ it is exists unique vector $E(i) \in \mathbb{F}_2^m$ with Hamming weight $w$ and associated with it monomial $z_{i_1} \cdot z_{i_2} \cdot \ldots \cdot z_{i_w}$ obtained as multiplication of a subset of $w$ elements of the set $Z = \{z_1, z_2, \ldots z_m\}$ where $m$ and $w$ are the minimal integers such that $\binom{m}{w} \geq N$.

$$\boldsymbol{x} = (x_1, \ldots, x_N) \to F_X(z_1, \ldots, z_m) = \sum_{i=1}^{N} x_i \prod_{E(i)_l = 1} z_l,$$

where $E(i)_l$ is the $l$-th coordinate of $E(i)$.

Each point $E(i)$ corresponds to exactly one term $x_i \prod_{E(i)_l = 1} z_l$, thus, the PIR problem is reduced to the problem of evaluating:

$$F_X(E(i)) = x_i, \forall i \in \{1, \ldots, N\}.$$

[1] A. Beimel, Y. Ishai, E. Kushilevitz, and J. F. Raymond. Breaking the barrier for information-theoretic private information retrieval. In Proc. of the 43rd IEEE Symp. on Foundations of Comp.Sc. (FOCS), 2002, pp. 261-270. [2] D.Woodruff and S.Yekhanin. A geometric approach to information-theoretic private information retrieval. SIAM J. Comput., vol. 37, no. 4, 2007, pp. 1046-1056.

**Woodruff–Yekhanin's scheme[2]**
The main idea is:

- to use derivative $f'(\lambda) = \frac{\partial f}{\partial \lambda} = \sum_{i \geq 1} i a_i \lambda^{i-1}$ of the polynomial $f(\lambda) = \sum_{i \geq 0} a_i \lambda^i$ over a finite field and the partial derivatives $\frac{\partial F_X}{\partial z_l}$ of $F_X(z_1, \ldots, z_m)$.
- to use a partial derivative of $F_X(z_1, \ldots, z_m)$ and their values at point $Z' = P + \lambda_h V, P = E(i)$, $\lambda_h$ are distinct and nonzero elements of $\mathbb{F}_q$ . $V$ - random vector from $\mathbb{F}_q^m$, $r$ is number of servers.

$$f'(\lambda_h) = \sum_{l=1}^{m} \frac{\partial F_X}{\partial z_l}\bigg|_{Z=P+\lambda_h V} \cdot V_l.$$

User wants to retrieve $F_X(P)$.

**PIR protocol for Woodruff–Yekhanin's scheme[2]**
Number of servers is $r$.

$\mathcal{U}$ : Selects random $V \in \mathbb{F}_q^m$.

$\mathcal{U} \to \mathcal{S}_h$ : $P + \lambda_h V$ , $h \in [1, r]$.

$\mathcal{S}_h \to \mathcal{U}$ : $F(P + \lambda_h V)$; $\left.\frac{\partial F_X}{\partial z_l}\right|_{Z=P+\lambda_h V}$, $l \in \{1, \ldots, m\}$.

$\mathcal{U}$ : Reconstructs $f(\lambda) = F_X(P + \lambda V)$ from $\{f(\lambda_h)\}$ and $\{f'(\lambda_h)\}$.

**General case for bit retrieving**

In general bit retrieving scheme let's suppose that our database $X = \{x_1, x_2, \ldots, x_n\}$ , where $x_i \in GF(2)$.

We will use the following mapping for positions of the database:

$$i --- > u_i = (u_i^{(0)} u_i^{(1)} \ldots u_i^{(l-1)}), u_i^{(j)} \in GF(2), wt(u_i) = w, \left( \begin{array}{c} l \\ w \end{array} \right) \geq n.$$

Where $w + 1$ - number of servers from which we can obtain responds.

Each vector $u_i$ corresponds to the monomial $m_i = z_{i_1} z_{i_2} \cdots z_{i_w}$.

Now we can define the following mapping for our database:

$$X --- > F(z_0, z_1, \ldots, z_{l-1}) = \sum_{i=1}^{n} x_i m_i =$$

$$x_1 z_0 z_1 \cdots z_{w-1} + x_2 z_0 z_1 \cdots z_{w-2} z_w + x_n z_{l-w+1} z_{l-w+2} \cdots z_l.$$

Now we should chouse field extension $m$.

This value directly depends from the number of servers $w$ from which we can obtain responds.

$(w + 1)m \leq 2^m - 2.$

**Main Idea**

For each server $S_i$ we will chouse primitive element $\alpha_i \in GF(2^m)$ with coset $A_i = \{\alpha_i, \alpha_i^2, \ldots, \alpha_i^{2^{m-1}}\}$ such that for any different servers $S_i$ and $S_j$ , $A_i \bigcap A_j = \emptyset$. For each primitive element $\alpha_i$ with coset $A_i = \{\alpha_i, \alpha_i^2, \ldots, \alpha_i^{2^{m-1}}\}$ let construct a basis $\mathcal{B}_i$ consists of $m$ consecutive powers of these element

$$\mathcal{B}_i = \{\alpha_i, \alpha_i^2, \ldots, \alpha_i^m\}.$$

Now let recalculate variables $\widehat{z}_0, \ldots, \widehat{z}_{l-1}$ that will be sending to server $S_i$ as following:

$$\widehat{z}_j = z_j + \sum_{k=1}^{m} c_{kj}\alpha_i^k, \text{ where } c_{kj} \in GF(2), j = 0, \ldots, l-1.$$

Therefore on sever $S_i$ the function $F(\widehat{z}_0, \widehat{z}_1, \ldots, \widehat{z}_{l-1})$ will calculated

$$\begin{aligned}
&F(\widehat{z}_0, \widehat{z}_1, \ldots, \widehat{z}_{l-1}) = \sum_{i=1}^{n} x_i m_i \\
&= x_1\widehat{z}_0\widehat{z}_1\cdots\widehat{z}_{w-1} + x_2\widehat{z}_0\widehat{z}_1\cdots\widehat{z}_{w-2}\widehat{z}_w + \ldots + x_n\widehat{z}_{l-w+1}\widehat{z}_{l-w+2}\cdots\widehat{z}_l \\
&= x_1 z_0 z_1\cdots z_{w-1} + x_2 z_0 z_1\cdots z_{w-2} z_w + \ldots + x_n z_{l-w+1} z_{l-w+2}\cdots z_l \\
&+ \sum_{j=1}^{wm-1} b_j\alpha_i^j \prod_{l=1}^{f(1\le f<w)} z_{j_l} + b_{wm}\alpha_i^{wm},
\end{aligned}$$

where $b_j \in GF(2)$.

Therefore on each server we obtain function $F$ as a function of $z_0, \ldots, z_{l-1}$ and $\alpha_i$. By the veritable $\alpha_i$ this function can be rewritten as

$$\begin{aligned} F(\alpha_i, z_0, \ldots, z_{l-1}) &= f_{mw}\alpha_i^{mw} + f_{mw-1}\alpha_i^{mw-1} + \ldots + f_1\alpha_i + x_1 z_0 z_1 \cdots z_{w-1} \\ &+ x_2 z_0 z_1 \cdots z_{w-2} z_w + \ldots + x_n z_{l-w+1} z_{l-w+2} \cdots z_l. \end{aligned}$$

Thus we obtain polynomial $F$ of degree of $\alpha_i$ not greater than $mw$ with the same coefficients for any primitive elements $\alpha_i$. It is means that request will be same for all servers and can be represented as a summa of polynomial from some variable $y$ and monomials $m_i = z_{i_1} z_{i_2} \cdots z_{i_w}$:

$$\begin{aligned} F(y, z_0, \ldots, z_{l-1}) &= f_{mw}y^{mw} + f_{mw-1}y^{mw-1} + \ldots + f_1 y + x_1 z_0 z_1 \cdots z_{w-1} \\ &+ x_2 z_0 z_1 \cdots z_{w-2} z_w + \ldots + x_n z_{l-w+1} z_{l-w+2} \cdots z_l. \end{aligned}$$

# Main Idea

By Lagrange interpolation procedure it is possible to calculate the value of $x_1 z_0 z_1 \cdots z_{w-1} + x_2 z_0 z_1 \cdots z_{w-2} z_w + \ldots + x_n z_{l-w+1} z_{l-w+2} \cdots z_l$ when we have at least $mw + 1$ values of this function in $wm + 1$ different points.

- By using $w$ different primitive elements $\alpha_i$ we obtain $w$ values of function $F$ .
- And by using $m - 1$ powers for each element $\alpha_i$ i.e. $\{\alpha_i^2, \ldots, \alpha_i^{2^{m-1}}\}$ we obtain $m - 1$ additional values $F(\alpha_i^{2^j}, z_0, \ldots, z_{l-1}) = F(\alpha_i, z_0, \ldots, z_{l-1})^{2^j}, j = 1, \ldots, m - 1$ for each primitive element.

# Requests generation for $j$-th bit retrieving.

In our protocol we use the scrambling matrix $C$ as a binary random matrix of size $[m \times l]$ of elements $c_{kj}$.
Matrix $B_i$ of size $[m \times m]$ consists of the elements of basis :

$$B_i = (\alpha_i \alpha_i^2 \ldots \alpha_i^m).$$

Now if we want to obtain $j-$ th bit $x_j$ we should send to server $S_i$ the following request:

$$R_i = U_j + B_i C.$$

where $U_j = (z_0 \alpha^0 z_1 \alpha^0 \ldots z_{l-1} \alpha^0)$ and $(z_0 z_1 \ldots z_{l-1}) = u_j$. Now we can rewrite $R_i$ as:

$$R_i = \left( \quad z_0 \alpha^0 + \sum_{k=1}^{m} c_{k1} \alpha_i^k \quad z_1 \alpha^0 + \sum_{k=1}^{m} c_{k2} \alpha_i^k \quad \ldots \quad z_{l-1} \alpha^0 + \sum_{k=1}^{m} c_{kl} \alpha_i^k \quad \right),$$

where $c_{kj}$ is the element of $k$-th row and $j$-th column in matrix $C$.

Each server calculate $F(R_i)$ over the field $GF(2^m)$ :

$$F(R_i) = F(\alpha_i) = f_{mw}\alpha_i^{mw} + f_{mw-1}\alpha_i^{mw-1} + \ldots + f_1\alpha_i + x_j$$

Therefore we obtain from $w+1$ servers $w+1$ values $F(\alpha_i), i = 1, \ldots, w+1$ of the function $F(y), \deg F(y) \leq mw$ . By using properties of cosets $A_i, i = 1, \ldots w$ in the field $GF(2^m)$ we obtain the $m(w+1)$ values of this function in $m(w+1)$ different points.

By using Lagrange interpolation procedure for $m(w+1)$ values of the function $F(y))$ in $m(w+1)$ different points we can find coefficient $x_j$.

For illustration of our new solution let's consider toy example.
Database is $X = \{x_1, x_2, \ldots, x_6\} = \{1, 0, 1, 1, 0, 0\}$ is stored on $w + 1 = 3$ servers.
Mapping of bit position is $i- > u_i$

| $i$ | $u_i$ | monomial($m_i$) |
|---|---|---|
| 1 | 1100 | $z_0z_1$ |
| 2 | 1010 | $z_0z_2$ |
| 3 | 0110 | $z_1z_2$ |
| 4 | 1001 | $z_0z_3$ |
| 5 | 0101 | $z_1z_3$ |
| 6 | 0011 | $z_2z_3$ |

$X- > F(z_0, z_1, z_2, z_3) = \sum_{i=1}^{6} x_i \cdot m_i = z_0z_1 + z_1z_2 + z_0z_3.$

In current example $GF(2^4)$ will be used with primitive polynomial $g(x) = x^4 + x + 1$.

| $i$ | $\alpha^i$ |
|---|---|
| 0 | 0001 |
| 1 | 0010 |
| 2 | 0100 |
| 3 | 1000 |
| 4 | 0011 |
| 5 | 0110 |
| 6 | 1100 |
| 7 | 1011 |
| 8 | 0101 |
| 9 | 1010 |
| 10 | 0111 |
| 11 | 1110 |
| 12 | 1111 |
| 13 | 1101 |
| 14 | 1001 |

Table: Table of elements $GF(2^4)$

To retrieve the second bit we should use $z_0 = 1, z_1 = 0, z_2 = 1, z_3 = 0$ and therefore request vector is $u_2 = (z_0 z_1 z_2 z_3) = (1010)$.
In our new approach we will use the elements from the field $GF(2^m)$.
Therefore our request vector $u_2$ becomes the matrix $U_2$ of the size $[4 \times 4]$:

$$U_2 = (z_0 \alpha^0 z_1 \alpha^0 z_2 \alpha^0 z_3 \alpha^0) = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

We use anywhere here the interpretation of the field element $\alpha^j$ as a binary matrix-column with number of the rows equal to the field extension.
For each server $S_i$ we will calculate the "salted" request :

$$R_i = U_2 + B_i C.$$

Scrambling random matrix $C = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$

For server $S_1$ request $R_1 = U_2 + B_1 C = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} + B_1 C$

Where $B_1 = \{\alpha\alpha^2\alpha^3\alpha^4\} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$.

Therefore $R_1 = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix} = \{\alpha^7\alpha^{12}\alpha^8\alpha^5\}$

Similarly for servers $S_2$ and $S_3$ we get $B_2 = \{\alpha^3\alpha^6\alpha^9\alpha^{12}\}$ and $B_3 = \{\alpha^7\alpha^{14}\alpha^6\alpha^{13}\}$.

And therefore we obtain

$$R_2 = \{\alpha^4\alpha^{14}\alpha^{13}\alpha^2\}$$

and

$$R_3 = \{\alpha^5\alpha^3\alpha^3\alpha\}.$$

Each generated request is sent to appropriate server.

Each server calculates $F(R_i)$ over the field $GF(2^4)$.
$F(R_1) = F(\alpha^7\alpha^{12}\alpha^8\alpha^5) = \alpha^7\alpha^{12} + \alpha^{12}\alpha^8 + \alpha^7\alpha^5 = \alpha^9$.
$S_2$ calculates $F(R_2) = F(\alpha^4\alpha^{14}\alpha^{13}\alpha^2) = \alpha^4\alpha^{14} + \alpha^{14}\alpha^{13} + \alpha^4\alpha^2 = \alpha^7$.
$S_3$ calculates $F(R_3) = F(\alpha^5\alpha^3\alpha^3\alpha) = \alpha^5\alpha^3 + \alpha^3\alpha^3 + \alpha^5\alpha = \alpha^8$.
Each server $S_i$ sends its reply to the user.

## 2-nd bit retrieving procedure.

User has precomputed Lagrange coefficients:
$\{\lambda_{0,1}, \lambda_{0,2}, \lambda_{0,3}, \lambda_{0,4}, \lambda_{0,6}, \lambda_{0,7}, \lambda_{0,8}, \lambda_{0,9}, \lambda_{0,12}\} =$
$\{\alpha, \alpha^{11}, \alpha^{12}, \alpha^{6}, \alpha^{0}, \alpha^{4}, \alpha^{0}, \alpha^{4}, \alpha^{11}\}$
He calculates additional points for polynomial interpolation:
From $S_1$ one point is received $F(\alpha) = \alpha^9$, then
$(F(\alpha))^2 = F(\alpha^2) = \alpha^3, (F(\alpha))^4 = F(\alpha^4) = \alpha^6, (F(\alpha))^8 = F(\alpha^8) = \alpha^{12}$.
From $S_2$ one point is received $F(\alpha^3) = \alpha^7$, then $\left(F(\alpha^3)\right)^2 = F(\alpha^6) =$
$\alpha^{14}, \left(F(\alpha^3)\right)^4 = F(\alpha^{12}) = \alpha^{13}, \left(F(\alpha^3)\right)^8 = F(\alpha^9) = \alpha^{11}$.
From $S_3$ one point is received $F(\alpha^7) = \alpha^8$.
There are 9 points of the polynomial $F(x)$ and it is enough to interpolate $F(0) = x_2$ using Lagrange coefficients.

$$x_2 = F(0) = \prod \lambda_{0,i} \cdot F(\alpha^i) = 0$$

.

For competitive comparison of proposed scheme with existing solution all significant parameters are presented in the following table.

| Parameters | Woodruff-Yekhanin Scheme | Our solution |
|---|---|---|
| Communication complexity | $O(r^2 \log_2 r N^{1/(2r-1)})$ | $O(N^{\frac{1}{r}})$ |
| Storage complexity | $N$ | $N$ |
| Computation complexity: - server side | $O(r^2 N^{2r/(2r-1)})$ | $O(rN)$ |
| - client side | $O(r^2 N^{1/(2r-1)})$ | $O(r^2)$ |

Table: Comparison

where $r$ is number of servers and $N$ is database size.

THANK YOU FOR YOUR ATTENTION!