Private Information Retrieval

- Motivations
- 2-Server PIR
- Single Server PIR
- Modern Results

Motivation

"Search"

Database of
Medical Conditions
"Advertisement Platform"

Query: Peanut Allergies
Peanut
Response: Webpage

- Database server learns your query and can sell this information to Ad networks or health providers.
- Could lead to more targeted ads, higher premiums, targeted pricing

No privacy!!!

If only we could query databases without the server learning any information about our query!

We can with Private Information Retrieval!
First (trivial construction)

- The server could just send the whole database!
- Issue: Communication is linear!
- Question: Can we obtain sublinear communication without the server learning our query?

Not Quite! — if \(\text{comm} < |\text{DB}|\), then we lose info about some entry \(j\); an unbounded server could distinguish which entry!

Getting around the issue!

**Noncollusion [CGKST95]:**
- Replace single server with 2 or more non-colluding servers.
- Not allowed to communicate!
- info-theoretic

**Computational Bound [K097]:**
- Assume the server is a computationally bounded adversary
- rely on cryptographic primitives
Two Server PIR

- We'll refer to the servers as $S_0, S_1$, and treat the database $DB \in \{0,1\}^N$ (i.e. Entry $j \in [N] = DB[j] \in \{0,1\}$)

A two server PIR scheme is a tuple of eff algs:

- **Query** $(N, i) \rightarrow (q_0, q_1)$: $N = |DB|$ and $i$ is the query index. $q_0, q_1$ are query strings sent to $S_0, S_1$ respectively.

- **Answer** $(DB, q) \rightarrow a$: $DB \in \{0,1\}^N$, $q$ is the query string from the client and $a$ is the server response.

- **Reconstruct** $(a_0, a_1) \rightarrow b$: $a_0, a_1$ are responses from $S_0, S_1$ and $b$ is the reconstructed entry.

\[
\begin{align*}
\text{Total Comm:} & \quad |a_0| + |a_1| + \\
& \quad |q_0| + |q_1|
\end{align*}
\]

\[
(q_0, q_1) \leftarrow \text{Query}(N, i)
\]

\[
b \leftarrow \text{Reconstruct}(a_0, a_1)
\]

\[\text{should be } DB[i]\]
Correctness:
\[ \forall N \in \mathbb{N}, \ 0 \leq i \leq N, \ \text{DB} \in \{0, 1\}^N, \]
\[ \Pr \left[ b = \text{DB}[i] \quad \left| \begin{array}{l}
(q_0, q_i) \in \text{Query}(N, i) \\
q_0 \in \text{Answer}(\text{DB}, q_0) \\
q_i \in \text{Answer}(\text{DB}, q_i) \\
b \leftarrow \text{Reconstruct}(q_0, q_i)
\end{array} \right. \right] = 1 \]

Security:
\[ \forall \text{poly-bounded } N \in \mathbb{N}, \ \text{DB} \in \{0, 1\}^N, \ \text{pairs } (i, j) \in [N]^2, \ b \in \{0, 1\}, \]
the following distributions are indistinguishable,
\[ \left\{ \text{Query} (\text{DB}, i)[b] \right\} \approx \left\{ \text{Query} (\text{DB}, j)[b] \right\} \]
which could be perfect or computational

**Note:** \( b \) selects only one of the server queries.

**Warmup:** A trivial construction!

**ith basis vector**

\[ \text{Query}(N, i): \text{Sample } q_0 \in \{0, 1\}^N \text{ and } q_i \leftarrow q_0 \oplus e_i. \text{ Output } (q_0, q_i) \]

\[ \text{Answer}(\text{DB}, q): \text{Output } a \leftarrow \langle \text{DB}, q \rangle \leftarrow \text{inner product} \]

\[ \text{Reconstruct}(q_0, q_i): \text{Output } b \leftarrow q_0 \oplus q_i \]
Correctness: $a_0 \oplus a_i = \langle DB, a_0 \rangle \oplus \langle DB, a_i \rangle$

$= \langle DB, a_0 \oplus a_i \rangle$

$= \langle DB, e_i \rangle = DB[i]$

Security: Separately, $a_0$ and $a_i$ are uniformly random $N$ bit vectors. Thus, the distributions will be identical!

**Issue:** Communication is still linear! (swapped direction)
(upload vs download)
(with just sending DB

CGKS Construction

- Assume $N = n^2$ is a perfect square for simplicity.

\[
\begin{bmatrix}
DB \\
N
\end{bmatrix} \rightarrow n
\]

\[
\begin{bmatrix}
e_0 \\
e_1
\end{bmatrix} = \begin{bmatrix}
0 \\
0 \\
0 \\
\vdots \\
0
\end{bmatrix}
\]

Query $(N, (i, j))$: Sample $q_0 \in \mathbb{F}_q, 1^n$ and $q_i = q_0 \oplus e_j$. Output $(q_0, q_i)$

Answer $(DB, q)$: Compute vector $a \leftarrow DB \cdot q$. Output $a$.

Reconstruct $(a_0, a_i)$: Output $b = (a_0 \oplus a_i)[i]$
Correctness: \( a_0 \oplus a_1 = DB \cdot q_0 \oplus DB \cdot q_1 \)
\[ = DB \cdot (q_0 \oplus q_1) \]
\[ = DB \cdot e_j = DB_j \text{ "jth col of DB"} \]

\( DB_j [i] = DB[i, j] \).

Security: Similarly, both \( q_0 \) and \( q_1 \) are uniformly random vectors.

Communication: \( O(\sqrt{N}) \) bits! (upload = download)

Can we do better in the two-server setting?

- When considering unbounded servers (info-theoretic), [DG15] obtain \( O(\sqrt{\log\log N / \log N}) = O(\sqrt{\log N}) \) comm from locally-decodable codes.

\[ DB \rightarrow C \]

[BG11, 15]

- In computationally bounded setting, distributed point functions (constructed from PRGs) gives us a 2-server PIR with \( O_\lambda(\log N) \) communication.

Expand \( \begin{array}{c} K_0 \text{ keys} \\ t_0 \end{array} \)

Expand \( \begin{array}{c} K_1 \\ t_1 \end{array} \)

\( t_0 \oplus t_1 = c_i \)
Computational Single Server PIR

- [k097] leverages linearly-homomorphic encryption (E(-gama, etc)

\[ \forall k \in \mathcal{K}, \forall m_0, m_1 \in \mathcal{M}, \]
\[ \text{Enc}(k, m_0) + \text{Enc}(k, m_1) = \text{Enc}(k, m_0 + m_1) \]

\text{Query}(N, (i, j)): \ k \leftarrow \mathcal{K}. \ Output \ query \ q' \leftarrow (\text{Enc}(k, e_1[i]), \text{Enc}(k, e_2[j]), \ldots, \text{Enc}(k, e_n[n]))

\subseteq \text{vector of ciphertexts}

\text{Answer} (DB, q): \ Output \ a \in DB \cdot q.

\[
\begin{bmatrix}
\mathcal{C}_1 \\
\vdots \\
\mathcal{C}_n
\end{bmatrix}
= \begin{bmatrix}
\mathcal{C}_1 \\
\vdots \\
\mathcal{C}_n
\end{bmatrix}
+ \begin{bmatrix}
0 \\
\vdots \\
0
\end{bmatrix} + \ldots
\]

\text{Linear Homomorphism}

\[
\begin{bmatrix}
\mathcal{C}_1 + \mathcal{C}_2 + \ldots \\
\mathcal{C}_1 + \ldots \\
0 + \mathcal{C}_2 + \ldots \\
\ldots
\end{bmatrix}
\leq n \text{ additions}
Reconstruct \((k, a)\): Decrypt ciphertexts,

\[ u \leftarrow (\text{Dec}(k, a_1), \text{Dec}(k, a_2), \ldots, \text{Dec}(k, a_n)) \]

Output \(u[i]\).

**Correctness:** Follows from linear homomorphism of \(\text{Enc}\),

\[
a = DB \cdot a = \left[ \sum_t DB_{1,t} \cdot a_t \right] = \left[ \begin{array}{c}
\text{Enc}(k, DB_{1,j}) \\
\text{Enc}(k, DB_{2,j}) \\
\vdots \\
\text{Enc}(k, DB_{n,j})
\end{array} \right]
\]

**Security:** Follows from semantic security of \(\text{Enc}\) notably that the \(\text{Enc}\) of an \(e_j\) should be computationally indistinguishable from \(e_j \neq j\).

**Comm:** \(|\text{Ciphertext}| = O(\lambda)\) so overall comm is \(O(\lambda n) = O(\lambda \sqrt{n})\)

Can we do better?
Idea: Recursion! Notice the client only needs 1 ciphertext, so why not treat the server response as another database DB'. $N = n^3$

\[ \text{Client} \rightarrow \text{Server} \]

\[ q = \text{Enc}(k, e_j) \]

\[ q' = \text{Enc}(k, e_i) \]

\[ a' \]

\[ q' = a' \]

Can he recurse further?
Issue: $|\text{Entry}| = 1$ bit

$\downarrow$

$|\text{Ciphertext}| = \lambda$ bit

So, everytime we recurse we blow up communication by a factor of $\lambda$. $|a| = \lambda^2 n^2$, $|a'| = \lambda^2 n$

Solution: Leverage a Lin-hom scheme which

- good rate: $\frac{|m|}{|c|}$ is closer to 1
- large message spaces

[Damgard- Jurik]

Best Comp Single Server: polylog($n$) comm

[CM99, Lip05]

From QR, BHT, LWE assumptions