Privacy Enhancing Technologies

Private Information Retrieval

Florian Tramèr

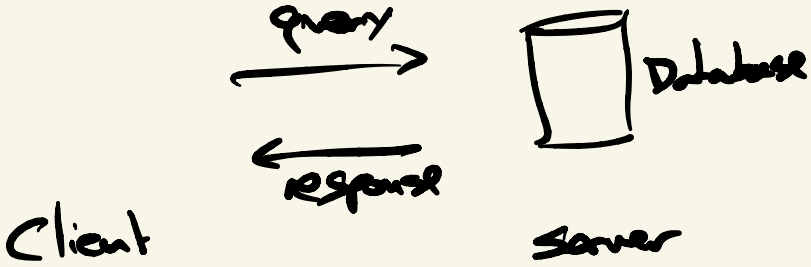
PIR

HCG: "an almost perfect research problem"

Dan Spielman: A perfect research problem

1. It has a nice theory ✓
2. It works in practice (✓)
3. It solves a problem people care about (✓)

The goal: Build a private Google search



Q: Can we do this with privacy for the client?

Trivial PIR: For every query, download the DB

Sublinear DB search: the communication between client and server is $o(|DB|)$

↳ $O(\sqrt{N})$

$O(\log N)$

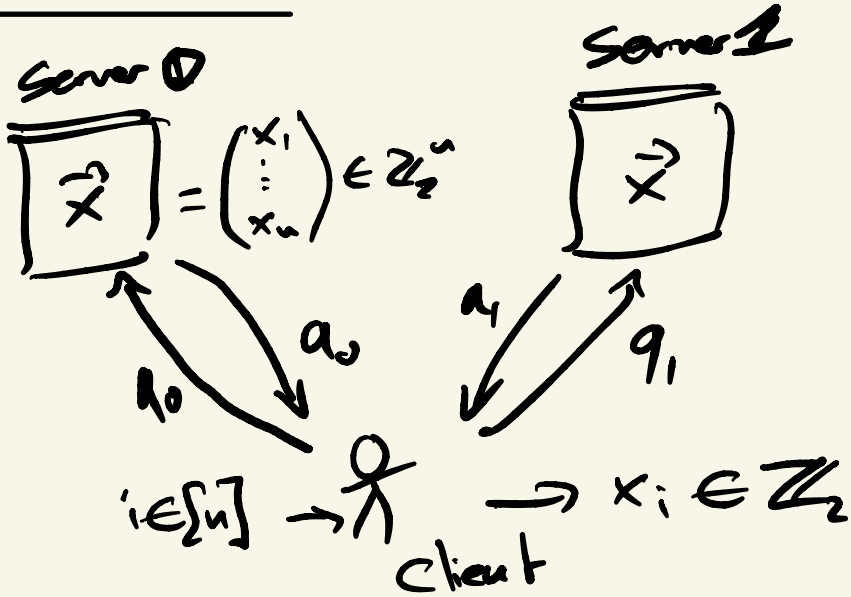
This is impossible ! (information theoretically)

Solution:

1) Assumption: we have 2 or more non-colluding servers

2) Assumption: Public Key crypto

2-Server PIR



Privacy guarantee: even if both servers are malicious but don't collude the client's query is hidden

Def: 2-Server PIR

- $q_0, q_1 \leftarrow \text{Query}(i)$
- $a_b \leftarrow \text{Answer}(\vec{x}, q_b)$ for $b \in \{0,1\}$
- $x_i \leftarrow \text{Reconstruct}(a_0, a_1)$

Properties

- Correctness: $\forall n \in \mathbb{N}, i \in [n], \vec{x} \in \mathbb{Z}_2^n$

$$P_r\left[\text{Reconstruct}(a_0,a_1) = x_i \;:\; \begin{array}{l} q_0,q_1 \leftarrow \text{Query}(i) \\ a_0 \leftarrow \text{Answer}(q_0) \\ a_1 \leftarrow \text{Answer}(q_1) \end{array}\right]$$

- Security: $\forall n \in \mathbb{N}, i,i' \in [n], b \in \{0,1\}$

$$\{q_b : q_0,q_1 \leftarrow \text{Query}(i)\} \overset{c}{\equiv} \{q_b : q_0,q_1 \leftarrow \text{Query}(i')\}$$

- Sublinear communication:

$$|\text{Query}(i)| + |\text{Answer}(\vec{x}, q_0)| + |\text{Answer}(\vec{x}, q_1)| = o(n) + o(1)$$

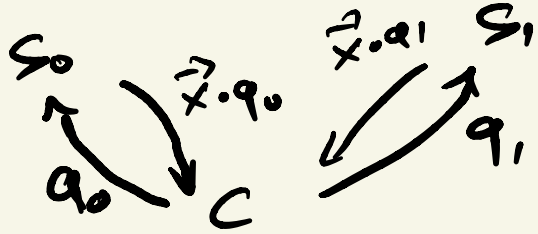
A strawman scheme:

$$x_i = \vec{x} \cdot e_i = \sum_{j=1}^{n} x_j \cdot (e_i)_j$$

($e_i = [0, 0, \dots, 0, \underset{\text{i-th position}}{1}, 0, \dots, 0]$)

Idea: hide e_i using "secret sharing" (additive)

$e_i = q_0 \oplus q_1$, where $q_0 \in \mathbb{Z}_2^n$



$$q_0 \oplus q_1 = x_i$$

$\hookrightarrow \langle \vec{x}, q_0 \rangle \oplus \langle \vec{x}, q_1 \rangle$

$= \langle \vec{x}, q_0 \oplus q_1 \rangle$

$= \langle \vec{x}, e_i \rangle = x_i$

$$\left(\bigoplus_j x_j (q_0)_j \right) \oplus \left(\bigoplus_j x_j (q_1)_j \right) =$$

$$\bigoplus_j x_j \left((q_0)_j \oplus (q_1)_j \right) = \bigoplus_j x_j (e_i)_j$$

$$= x \cdot e_i$$

Where's the progress? $|q_0| + |q_1| = 2|X|$

trivial PIR: Client downloads n bits

Strawman: Client uploads $2n$ bits
and downloads 2 bits

Load-balancing for $O(\sqrt{n})$ comm.

Idea: View the database as a square

$$\vec{X} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \Rightarrow X \in \mathbb{Z}_2^{\sqrt{n} \times \sqrt{n}}$$

$$\begin{bmatrix} x_1 & x_2 & \cdots & x_{\sqrt{n}} \\ x_{\sqrt{n}+1} & \cdots & & \\ & \ddots & & x_n \end{bmatrix}$$

The protocol:

write $i \in [n]$ as $(i,j) \in [\sqrt{n}] \times [\sqrt{n}]$

- Query(i,j) : $q_0 \xleftarrow{\$} \mathbb{Z}_2^{\sqrt{n}}$

  $q_1 \leftarrow q_0 \oplus e_j$

- Answer(X, q_b) : $a_b \leftarrow X \cdot q_b \in \mathbb{Z}_2^{\sqrt{n}}$ (X: $\sqrt{n} \times \sqrt{n}$, q_b: $\sqrt{n}$)

- Reconstruct(a_0, a_1) : $x_i \leftarrow (a_0 \oplus a_1)_j$

- Communication:
  - * upload: $|q_0| + |q_1| = 2\sqrt{n}$ bits
  - * download: $|a_0| + |a_1| = 2\sqrt{n}$ bits

  total: $O(\sqrt{n})$

- Security: q_0 is uniformly random in $\mathbb{Z}_2^{\sqrt{n}}$

  q_1 is as well (one-time pad)

Single server PIR

Linearly homomorphic encryption

$$Enc(k, m_1) \text{ "+" } Enc(k, m_2) = Enc(k, m_1 \text{ "+" } m_2)$$

Strawman

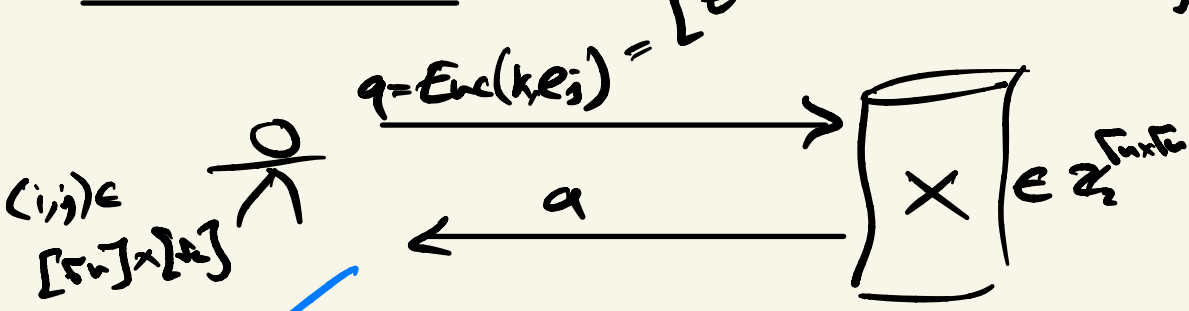
Query: $\vec{q} = Enc(k, e_j) = [Enc(k,0), \ldots, Enc(k,1), \ldots, Enc(k,0)]$

Answer: $\vec{x} \cdot \vec{q}$

$$= Enc(k,0) \cdot x_1 + \ldots + Enc(k,1) \cdot x_j + \ldots + Enc(k,0) \cdot x_n$$

$$= Enc(k, x_j)$$

Load balancing



$(Dec(a))_i$

$= (Dec(Enc(x_j)))_i$

$= (x_j)_i = x_{ij}$

$a = X \cdot q$

$= X \cdot Enc(e_j)$

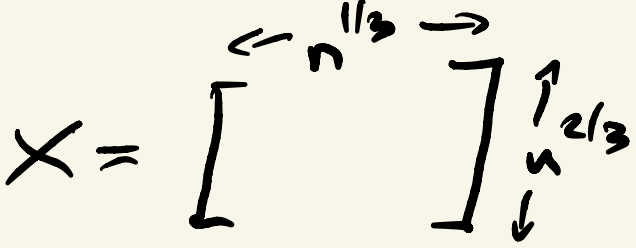
$= Enc(X \cdot e_j)$

$= Enc(x_j)$

Communication: $|q| + |a|$ ciphertexts

$= O(\sqrt{n}\lambda)$ bits

Recursive PIR:



$(i,j,k) \in [n]^3$

$q_0 = Enc(e_j)$

$q_1 = Enc(e_i)$

$X \cdot q_0 = a_0$ (dim $n^{2/3}$)

$\begin{bmatrix} & n^{1/3} & \\ n^{1/3} & A & \end{bmatrix} \cdot q_1 = a_1$ (dim $n^{1/3}$)

$n^{1/3}$ ciphertext

$$q_0 = X \cdot Enc(e_j) = Enc(X \cdot e_j) = Enc(x_j)$$

x_j: column j of X, $n^{2/3}$ entries

a_0: written as a matrix

$$q_1 = a_0 \cdot Enc(e_i) = Enc(a_0 \cdot e_i)$$

$$= Enc(Enc(X_{ij}))$$

With $O(\log n)$ recursion steps you can get $O(\text{polylog } n)$ comm.

This is "optimal". Why?

"indexing" into a Db of size n needs $\log n$ bit

Can we do recursive PIR in the 2-server protocol? **No!** Why?

But we can get polylog(n) comm. (see HW)

Computational Complexity

All schemes we saw have the server do $O(n)$ work

Inherent, unless you change how you count / the model

- Batching: k queries on a DB of size n in $O((\lambda \log k)n)$ time

- Pre-processing the DB
  - Some linear work <u>once</u> and then answer many queries with sublinear work
    - 2-server: $O(\text{Tm poly}(\log n))$ "online" computation
    - <u>1</u>-server: $O(\text{poly} \log n)$ computation per query after a linear pre-proc.

- $Enc(m_1)$ $Enc(m_2)$ $\Rightarrow$ $Enc(m_1 + m_2)$
- $Enc(m_1)$ $\Rightarrow$ $Enc(2m_1)$
- $Enc(m_1)$ $\Rightarrow$ $Enc(K \cdot m_1)$ in $\log K$ steps