Lecture 3: Private Information Retrieval

MIT - 6.893

Fall 2020

Henry Corrigan-Gibbs

Plan

* Recap: Preliminaries
* PIR: What it is, why it's amazing
* Stretch break
* Constructions
  - Two-server PIR
  - One-server PIR

Logistics

* HW1 due this Friday 9/18 @ 5pm Boston via Gradescope
  ↳ You must use Latex template
* OH: W 3-4:30pm on Zoom (link on Piazza)
* Please give feedback on psets
* Anonymous feedback form

Recap: Fundamental Primitives

$$\begin{tikzcd}
\text{PRPs} \arrow[r, bend left, "\text{switching lemma}"] & \text{PRFs} \arrow[l, bend left, "\text{Luby-Rackoff (Feistel)}"] \arrow[r, bend left, "\text{counter mode}"] & \text{PRGs} \arrow[l, bend left, "\text{GGM tree}"] \arrow[r, bend left, "\text{immediate}"] & \text{OWFs} \arrow[l, bend left, "\text{G.L., HILL}"]
\end{tikzcd}$$

→ All imply each other.
→ None imply key exchange.

- If $P = NP$, none exist.
- If $P \neq NP$, ?

Asymptotic view:
security parameter λ
efficient = $poly(\lambda)$
small = $negl(\lambda)$

Concrete view
efficient = runs on your computer in reasonable time budget
small $\leq 2^{-128}$.

A "perfect" research result

Adopted from talk by Dan Spielman, who adopted it from someone else (I think)

1) Has a beautiful theory
2) Works in practice
3) Solves a problem that people care about.

⇒ It's a rare piece of work that meets this rubric. But aim high.

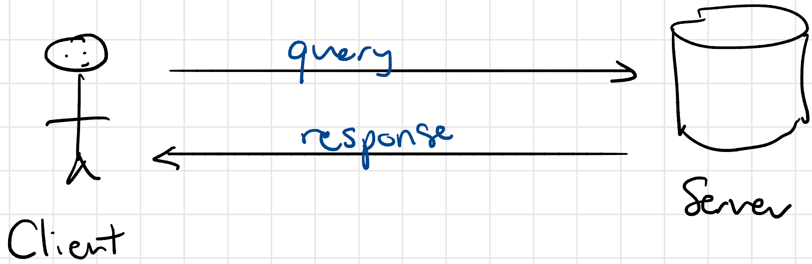
Today

- One of my favorite "almost perfect" ideas in crypto.
- Lots of activity, more ongoing even today.
  ↳ Will cover recent results next week.
- A classic crypto result: seems impossible, then is simple.

Bad news: For reasons we'll see, it's not quite practical yet.

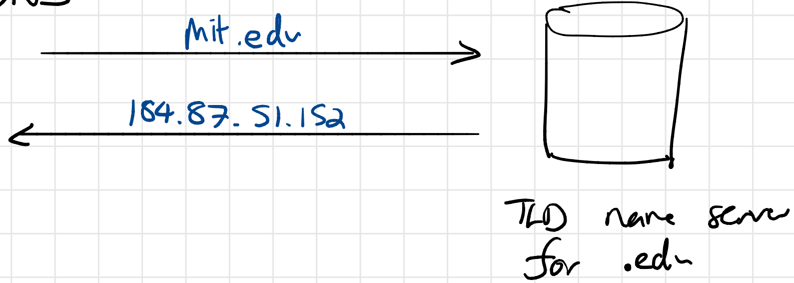
Private Information Retrieval

Every day on the Internet

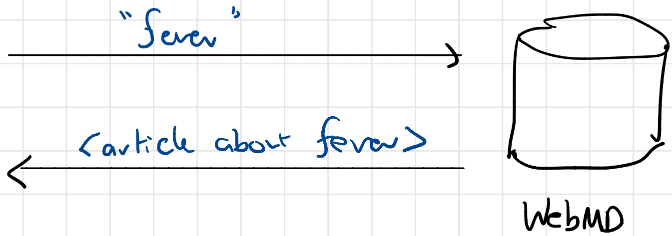


Examples

1. DNS



2.

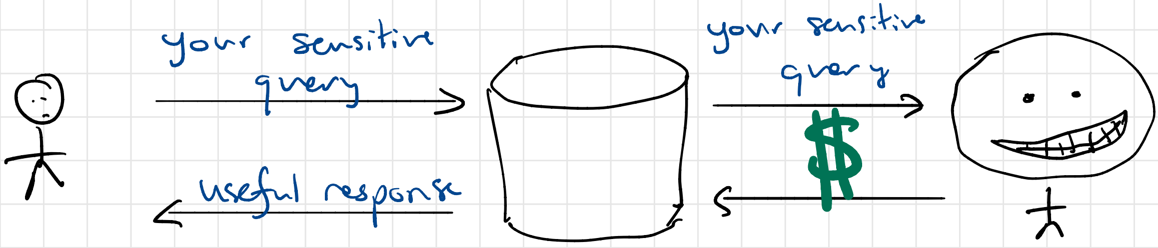


3. Many more....

- Searching Google
- Looking news articles
- Fetching data from social media networks
- Looking for Airbnb properties

Notice: The client's query can be sensitive!
It can leak:
- what website you're visiting
- your health conditions
- your travel plans
- political interests
- ...

↳ Today, server learns all of these things!



Evil website

Question: "Can you query a database without the database learning your query?"

Trivial answer: "Yes, just download the entire database."

DB server doesn't learn your query.

Still, this is unsatisfying.

Let's ask a better question...

Better Question:

"Can you query a database without the database learning your query...

... with communication sublinear in the database size?

Answer: Unconditionally, no. [CGKS '95]

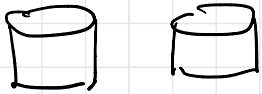
We won't prove this, but there is a clean info. theoretic argument in the original PIR paper.

What do we do when we are stuck in life?

Option I

Change the model. (Rom, etc.)

What if we have two non-colluding copies of the DB?



"two-server PIR" [CGKS 95]

Option II

Make assumptions!

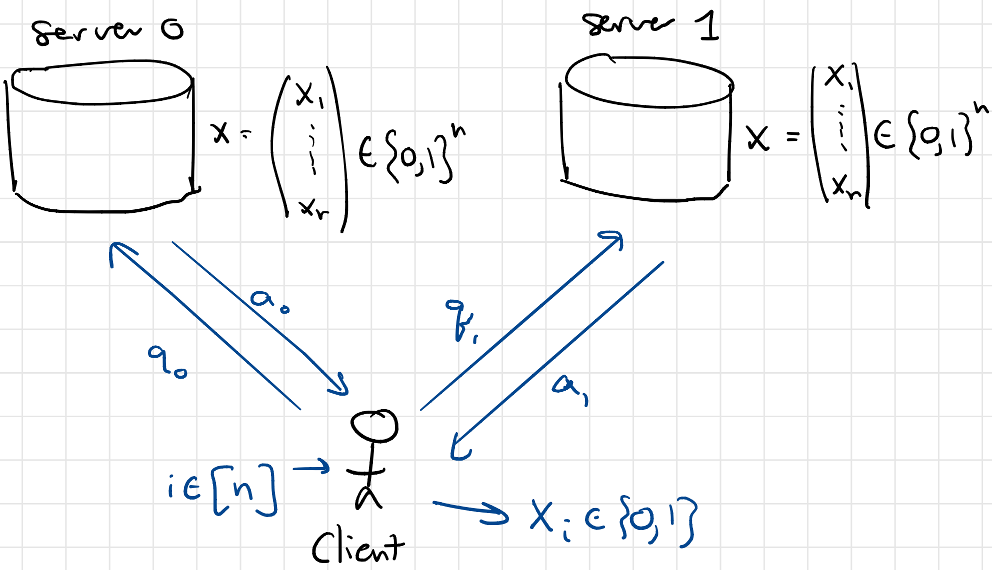
Under basic "public-key" assumptions (DDH, Paillier, ...) can build non-trivial PIR w/ 1 server

"single-server PIR" [Kushilevitz & Ostrovsky '97]

(k-server PIR uses k DB replicas)

Two-server PIR

Model



Important! Security only holds if servers do not collude (i.e., one of the two servers is honest).

Non-essential simplifications

* DB is an array of bits (can extend to handle longer rows)

* DB lookup is by index (can implement a key-value map)

later this week

More formally:

Two-server PIR consists of three eff algs:

$$\text{Query}(1^n, i) \to (q_0, q_1)$$

$$\text{Answer}(x, q_b) \to a$$

$$\text{Reconstruct}(a_0, a_1) \to x_i$$

Properties

1. Correctness: Client gets the bit it wants.

$$\forall n \in \mathbb{N},\ \forall i \in [n],\ \forall x \in \{0,1\}^n$$

$$\Pr\left[\text{Reconstruct}(a_0, a_1) = x_i : \begin{array}{l} (q_0, q_1) \leftarrow \text{Query}(1^n, i) \\ a_0 \leftarrow \text{Answer}(x, q_0) \\ a_1 \leftarrow \text{Answer}(x, q_1) \end{array}\right] = 1.$$

2. No single server learns anything about client's bit.

$$\forall n \in \mathbb{N} \quad \forall i, i' \in [n], \quad \forall \beta \in \{0,1\}$$

Can be $\equiv$ for perfect privacy.

$$\{q_\beta : (q_0, q_1) \leftarrow \text{Query}(1^n, i)\} \approx_c \{q_\beta : (q_0, q_1) \leftarrow \text{Query}(1^n, i')\}$$

Non-collusion is captured by our requirement that the marginal distributions are indist.

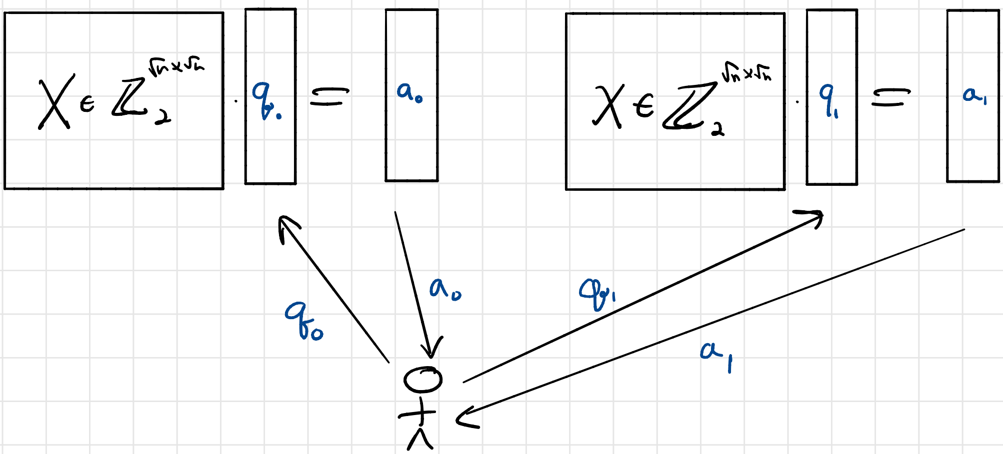
→ In info-theoretic setting (explain), (q_0, q_1) will leak secret index i.

Stretch Break

Two-server PIR scheme with $O(\sqrt{n})$ communication.

↑ Already very non-trivial.

Idea: View database as a matrix in $\mathbb{Z}_2^{\sqrt{n} \times \sqrt{n}}$.

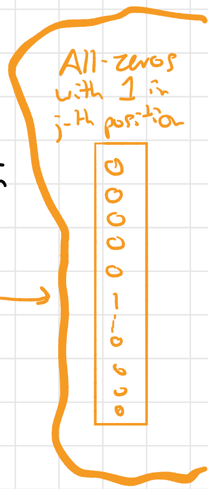


Client wants to read bit $(i,j) \in [\sqrt{n}] \times [\sqrt{n}]$

$\text{Query}(1^n, i, j) \to (q_0, q_1)$

Sample random $q_0, q_1 \in \mathbb{Z}_2^{\sqrt{n}}$ s.t.

$q_0 + q_1 = e_j \in \mathbb{Z}_2^{\sqrt{n}}$



$\text{Answer}(x, q) \to X \cdot q \in \mathbb{Z}_2^{\sqrt{n}}$

$\text{Reconstruct}(a_0, a_1) \to (a_0 + a_1)_i = x_{ij}$

↑ Select value from ith position

(1) Correctness

$$a_0 + a_1 = (Xq_0 + Xq_1)_i$$
$$= (X(q_0 + q_1))_i$$
$$= (X e_j)_i$$
$$= x_{ij}$$

✓

(2) Security

q_0 is a uniform random vector (independent of (i,j)).

↳ Same for q_1. ✓

Notice: No computation assumptions here!

Efficiency:

Upload: $2\sqrt{n}$ bits

Download: $4\sqrt{n}$ bits.

$\Rightarrow$ Total: $4\sqrt{n}$ bits. ✓

Single-server PIR

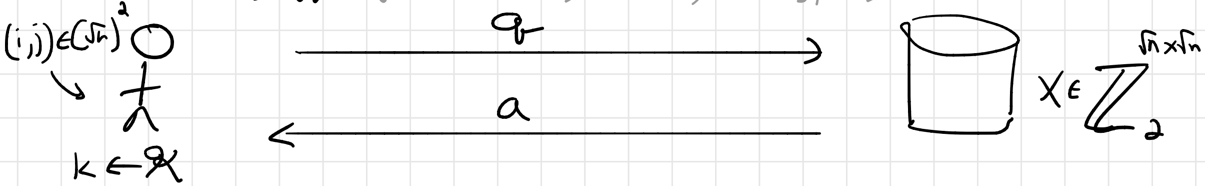
Linearly homomorphic encryption scheme:

$$E(k, m_0) + E(k, m_1) = E(k, \underbrace{m_0 + m_1}_{\text{mod } 2})$$

Can build from QR, DDH, LWE, ...

Idea: Client sends encryption of its query vector rather than using secret sharing.

$$E(k, e_j) = \big(E(k,0), \ldots, E(k,1), E(k,0), \ldots \big)$$



$$a \leftarrow X \cdot q$$
$$= X \cdot \mathrm{Enc}(k, e_j)$$
$$= \mathrm{Enc}(k, X \cdot e_j)$$
$$= \mathrm{Enc}(k, \vec{x}_j)$$

By adding ciphertexts

Output $x_{ij} \leftarrow (\mathrm{Dec}(k, a))_i$

$$= \mathrm{Dec}(k, \mathrm{Enc}(k, \vec{x}_j))$$

$$= x_{ij}$$

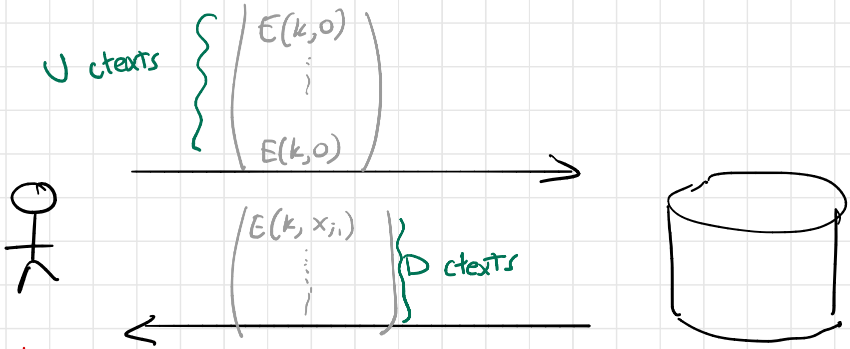
Communication: $2\sqrt{n}$ ciphertexts.

Reducing the communication

Kushilevitz & Ostrovsky '97

Let's look more closely at our PIR scheme...

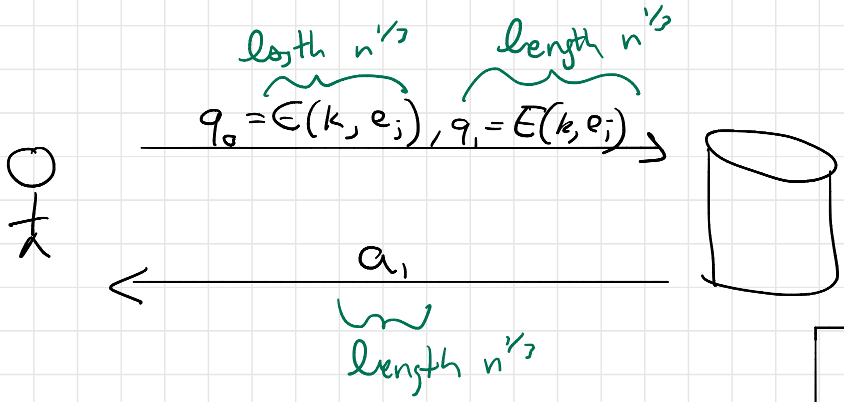
$$X = \begin{array}{cc} & U \\ D & \begin{pmatrix} \quad & \quad \\ \quad & \quad \end{pmatrix} \end{array}$$

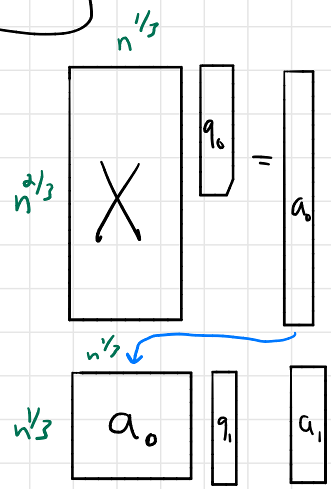


Client throws away all but one of the responses!

Idea: View answer to query as another database and run a second PIR on this DB.

$$X = \overbrace{\begin{bmatrix} \quad & \quad & \quad \\ \\ \\ \end{bmatrix}}^{n^{1/3}} \left.\vphantom{\begin{bmatrix} \\ \\ \\ \end{bmatrix}}\right\} n^{2/3}$$





Communication fell to

$\approx n^{1/3}$.

Then recurse!

The only catch is that each step of the recursion

n bits $\longrightarrow \sqrt{n}$ ciphertexts.

Under "reasonable" assumptions (QR takes $2^{\tilde{\Omega}(n^{1/3})}$ time), get $2^{O(\sqrt{\log n \log\log n})}$ communication.

↳ With more esoteric cryptosystems, (Based on Phi-hiding, Damgard-Jurik), can drop comm cost to polylog(n).

State of the art in PIR

* Two-server PIR

  Information theoretic - $n^{O\left(\sqrt{\log\log n / \log n}\right)} = n^{o(1)}$
  (Dvir & Gopi 2015)

  ↳ Do better schemes exist? With $O(\log^2 n)$ comm.?

  Computational - $O(\lambda \log n)$ ← Boyle speaking here next week!
  (Boyle, Gilboa, Ishai 2015)

  ↳ Requires only PRGs. Concretely quite efficient.

* Single-server PIR

  polylog(n) communication - from QR, DDH, LWE, ...

  (Cachin, Micali, Stadler '99, Lipmaa '05, ...)

Computational Efficiency in PIR

In all above schemes, the servers run in time linear in the DB size.

$\Rightarrow$ Linear DB scan per query.

For certain "natural" PIR schemes, this limitation is inherent. [Beimel, Ishai, Malkin '04]

$\rightarrow$ We will see some ways to reduce the computation cost at the servers. $\leftarrow$