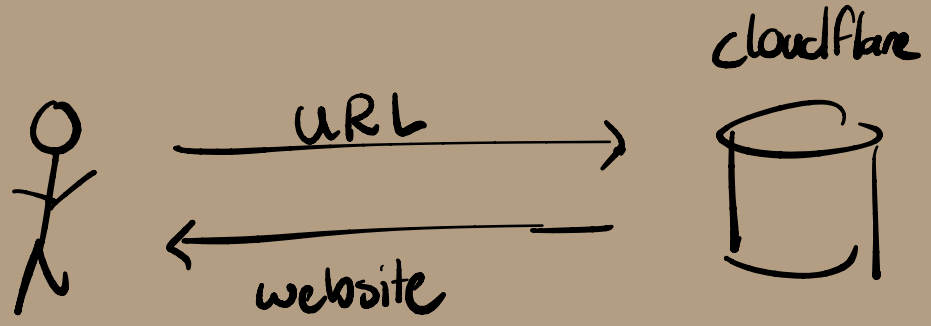
# Private Information Retrieval



We've seen:

1. Secure Multi (2) - Party Computation

$\underline{A(x_1)}$ $\qquad\qquad$ $\underline{B(x_2)}$

$\longleftrightarrow$

$\longleftarrow f(x_1, x_2) \longrightarrow$

Security: "A, B only learn $f(x_1, x_2)$"

2. $\underline{\text{Curator}(D)}$ $\xleftarrow{\quad q \quad}$ $\underline{\text{Analyst}(q)}$

$\xrightarrow{\quad r \quad}$

Security: "Analysts view is insensitive to all individuals"

| MPC | DP |
|---|---|
| trust no one | trust curator |
| leak exact answer | apx answer for privacy |
| cryptographic sec. | weaker numeric security |
| $(+2^{-\lambda}, \lambda \approx 128)$ | $(\times e^{\varepsilon}, \varepsilon \approx \frac{1}{10}$ in theory$)$ |

not very efficient

in practice!

$\leftarrow e^{\varepsilon} \approx 9\underline{\underline{M}}$

For Lookup Hints, Apple uses a privacy budget with epsilon of 4, and limits user contributions to two donations per day. For emoji, Apple uses a privacy budget with epsilon of 4, and submits one donation per day. For QuickType, Apple uses a privacy budget with epsilon of 8, and submits two donations per day.

For Health types, Apple uses a privacy budget with epsilon of 2 and limits user contributions to one donation per day. The donations do not include health information itself, but rather which health data types are being edited by users.

For Safari, Apple limits user contributions to 2 donations per day. For Safari domains identified as causing high energy use or crashes, Apple uses a single privacy budget with epsilon of 4. For Safari Auto-play intent detection, Apple uses a privacy budget with epsilon of 8.

Source: Apple's "Differential Privacy" overview

Rest of the course:
special cases of MPC.

Suppose...

Alice $\longrightarrow$ Google

DATA $D$

query $q$

wants $f(q, D)$

cost?

w/ MPC: $\sim |C_f| > |D|$

ideal: $\sim |q| + |f(q, D)|$

Simplest Case

Alice$(i)$ ← index

Google$(a)$ ← large, public array

$a[i] \longleftarrow$ $\longrightarrow$ not $i$

Is this OT?

No - $a$ is public

Is this important?

Yes. Most web browsing: $i$ is URL, $a$ is websites

Is $i$ sensitive?

Yes. Consider

WebMD: $i$ leaks health information

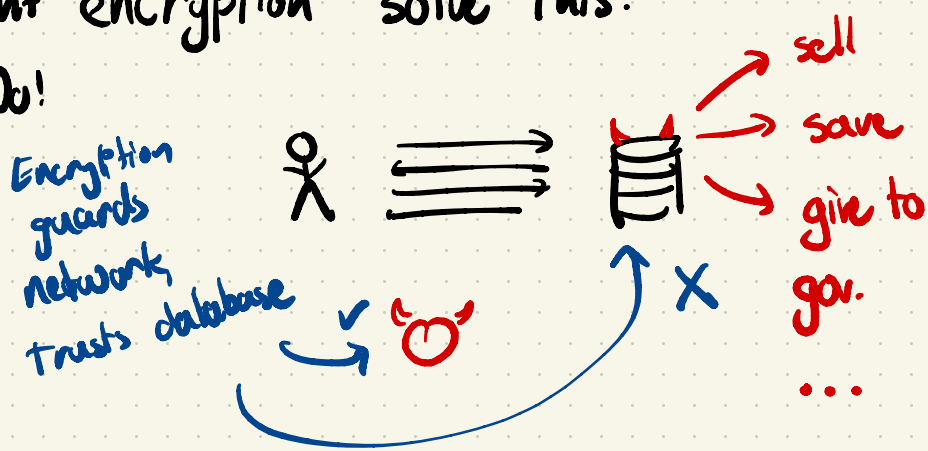
Stock prices: $i$ leaks financial info

Weather: $i$ leaks location

...

Doesn't encryption solve this?

No!




Core Q: Can we use a database without leaking queries to it...

Trivial Answer:

Yes! Download DB!

huge communication cost

In sublinear communication?

Information theoretically? No....

Idea: if you don't send $a[j]$, you learn that $j \neq i$.

→ and ECCs don't solve this...

Two ways to circumvent...

Multiple non-colluding servers

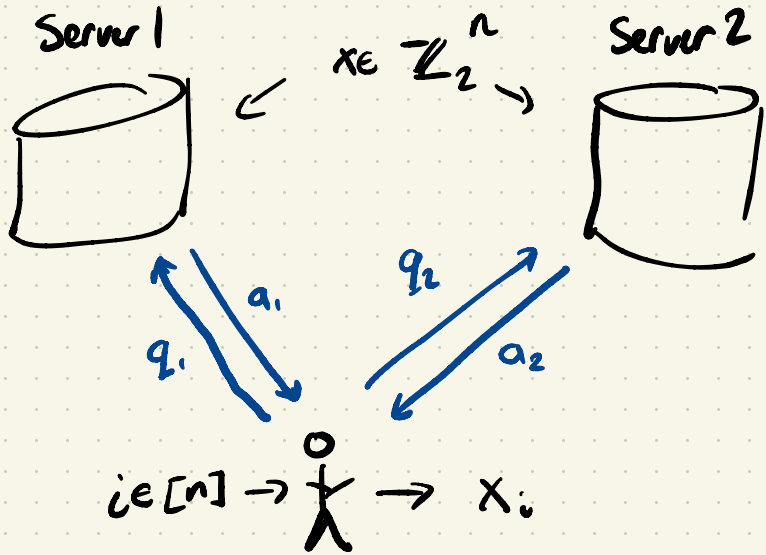
- [CGKS '95]
- two non-colluding servers

Computational assumptions

- [KO '97]

We'll see both today

# 2 Server PIR (Setup)




1. Security depends on non-collusion.

Simplifications (WLOG)

- Array values are bits
- Array keys are integers (not keywords)

# Formally

2-server PIR is 3 algs:

- $\text{Query}(n, i) \to q_0, q_1$
- $\text{Answer}(x, q) \to a$
- $\text{Construct}(a_0, a_1) \to x_i$

Properties:

- Correctness: $\forall n \in \mathbb{N} \ \forall i \leq n \ \forall x \in \mathbb{Z}_2^n$

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

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

$$\{q_b : q_0, q_1 \leftarrow \text{Query}(n, i)\}$$
$$\approx \quad \{q_b : q_0, q_1 \leftarrow \text{Query}(n, i')\}$$

How did we capture non-collusion?

By including only <u>one</u> query ($q_b$) in each distribution.

→ $(q_0, q_1)$ can (and will!) leak $i$.

# 2 Server PIR with $O(\sqrt{n})$ comm.

1. View $x$ as $\sqrt{n} \times \sqrt{n}$ matrix.
2. Balance query size & response size.

Reframing:

$$x \in \mathbb{Z}_2^n \implies X \in \mathbb{Z}_2^{\sqrt{n} \times \sqrt{n}}$$

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

Let $e_i \in \mathbb{Z}_2^{\sqrt{n}}$ denote the vector that is 1 only at index $i$.

Query$(n, (i,j)) \to (q_0, q_1)$:

$$q_0 \xleftarrow{\$} \mathbb{Z}_2^{\sqrt{n}}$$

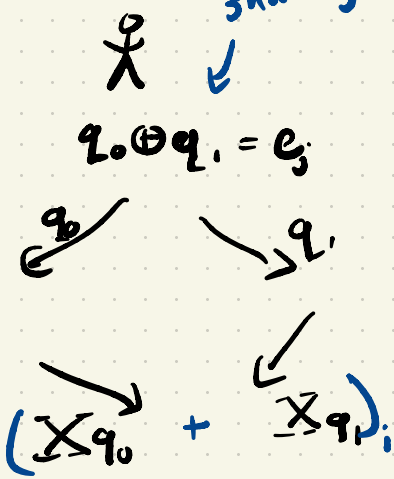
$$q_1 \leftarrow e_j \oplus q_0$$

Answer$(X, q) \to a$:

$$a \leftarrow Xq$$

Construct$(a_0, a_1) \to x_{i,j}$

$$x_{i,j} \leftarrow (a_0 + a_1)_i$$



Correctness:

$$(a_0 + a_1)_i = (\underline{X} q_0 + \underline{X} q_1)_i$$
$$= (\underline{X}(q_0 + q_1))_i$$
$$= (\underline{X} e_j)_i$$
$$= \underline{X}_{i,j}$$

Privacy:

$q_0$ is uniformly random (independent of $i, j$)

$q_1 = e_j + q_0$ is also uniformly random (without $q_0$)

Costs:

upload: $q_0, q_1$ : $\Theta(\sqrt{q})$

download: $a_0, a_1$ : $\Theta(\sqrt{q})$

server time: $\Theta(n)$

client time: $\Theta(\sqrt{n})$

Moving to 1-server PIR....

Replace secret-sharing $(q_0 + q_1 = e_j)$ with homomorphic encryption $(E(k, e_j))$

# 1-Server PIR

Recall <u>homomorphic</u> encryption:

$$E(k, m_1) + E(k, m_2) = E(k, m_1 + m_2)$$

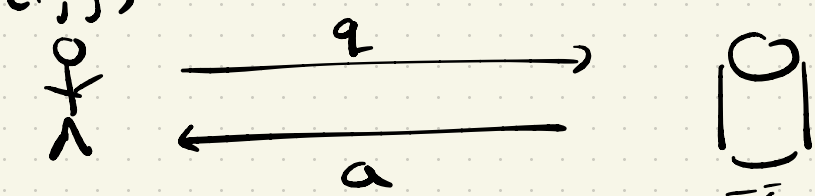
→ various constructions (e.g. our favorite PRF)

element-wise

$$q \leftarrow E(k, e_j) = (E(k,0), \dots, E(k,1), \dots, E(k,0))$$

index $j$.

$(i, j)$




$a \leftarrow \underline{X} q$

$$\left( a_\ell \leftarrow \sum_{i=0}^{\sqrt{n}} \underline{X}_{\ell,i} \cdot q_i \right)$$

output

$\underline{X}_{i,j} = D(k, a)_i$

correct?

$= D(k, X \cdot E(k, e_j))_i$

$= D(k, E(k, Xe_j))_i$

$= (Xe_j)_i$

$= \underline{X}_{i,j}$ ✓

Private?

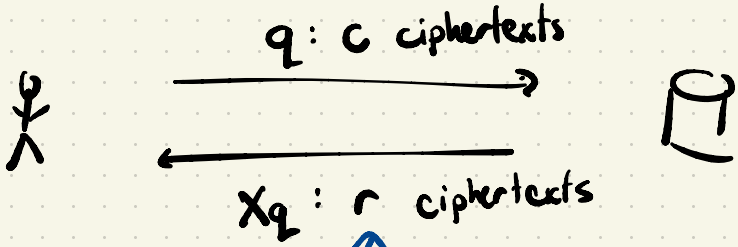
we just see ciphertexts...

# Reducing Communication further (sketch)

1. View $x$ as an $r \times c$ matrix

$$\underline{X} = \begin{array}{c} \\ r \end{array}\overset{c}{\begin{pmatrix} & \\ & \end{pmatrix}}$$

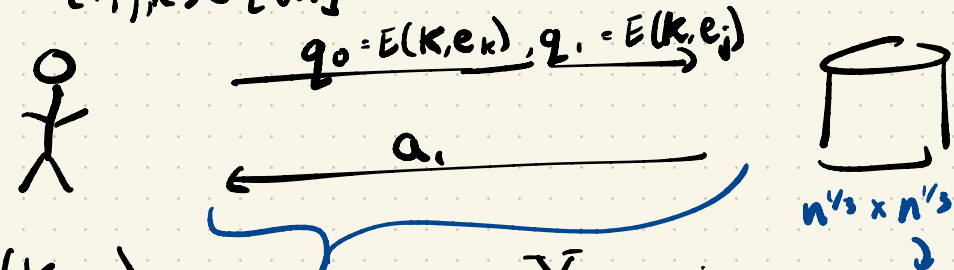
2. In 1-server PIR:




But client only looks at <u>one</u>.

Idea: view $r$ ciphertexts as a new database to do PIR on.

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




$D(K, a_1)$; $\quad a_0 \leftarrow \underline{X} q_0$ ; $\quad a_1 \leftarrow a_0 q_1$

$3n^{1/3}$ communication

$n^{2/3} \times n^{1/3}$ <u>or</u> $n^{1/3} \times n^{1/3} \times n^{1/3}$ - tensor

Technical issue!

- Surface problem: Doing PIR on $a_0$ is weird because $a_0$ has ciphertexts, not bits!
- Real problem: ciphertexts may be bigger than plaintexts (EG w/ $\mathcal{M} = \mathbb{Z}_2$, $|c| = 2\lambda$)
- Solution: Find the right encryption system

  $\Rightarrow$ e.g. Damgard-Jurik (Pallier variant)

  $\uparrow$ Homomorphic Encryption with $\mathcal{M} = \mathcal{C} = \mathbb{Z}_p$

# State of the Art PIR (communication)

- Two server
  - Information-Theoretic: $O(n^{\sqrt{\log\log n / \log n}})$ [DG '15]
  - Computational: $O(\log n)$ [GI '14] [BGI '15]
    - → Only PRF evaluations
    - → Very concretely efficient
- Single-Server
  - poly log(n) - from QR, DDH, LWE, ..
    [CMS '99] [Lip 05], ...

# Computational Efficiency of PIR

(elephant in the room)

→ The schemes above all take $\Theta(n)$ server time.

→ Impractical for the internet

→ This is unavoidable, in some sense

→ [BIM '04]: $\Omega(n)$ server work, even with multiple servers

→ How can we work around this?

1. Preprocessing
   (some server work is done before the query)
2. Batches of Queries

Stanford students have worked on this

(Dima Kogan, Henry Corrigan-Gibbs '19, '21)

A closing note...

- Cryptography asks new questions...
  - about important problems...
  - using beautiful mathematics.

Introducing all of you to this has been a pleasure and an honor.