Private information retrieval

Slides adapted from here, here, here
Last class: FSS & DPF

- Function secret sharing: allows a dealer to split a function \( f \) into function shares \( f_i \) such that for any input \( x \), \( f(x) = \sum_{i} f_i(x) \), where \( f_i \) are succinct and secret.

- Distributed point functions: a special function that can be efficiently shared

  - Define a point function \( f_{\alpha, \beta} : \{0,1\}^n \rightarrow \mathbb{G} \) for \( \alpha \in \{0,1\}^n \) and \( \beta \in \mathbb{G} \), where \( f(\alpha) = \beta \), and \( f(x) = 0 \) for \( x \neq \alpha \).

- Setting: multiple servers with some collusion threshold, each holding a copy of the full dataset.
Private information retrieval (PIR)

“Can a user query a database without the database learning the query?”

Many potential applications: DNS lookup, keyword searching, etc.
PIR definitions

• Database $x$ of $n$ bits

• Query$(1^n, i) \rightarrow q$, Answer$(x, q) \rightarrow a$, Decode$(a) \rightarrow x_i$

• **Correctness:** client gets the bit that it wants

  • $\forall n \in \mathbb{N}, \forall i \in [n], \forall x \in \{0,1\}^n$, 
    
    $Pr[\text{Decode}(a) = x_i : q \leftarrow \text{Query}(1^n, i), a \leftarrow \text{Answer}] = 1$

• **Privacy:** server should not learn anything about client’s bit

  • $\forall n \in \mathbb{N}, \forall i, i' \in [n], \{q \leftarrow \text{Query}(1^n, i)\} \approx_c \{q \leftarrow \text{Query}(1^n, i')\}$
PIR via DPFs

DPF can efficiently share the point function $f_{i,1}$

(Eval($q_0$) $\oplus$ Eval($q_1$) = $e_i$, which is a vector where $i$th index is 1, and 0 everywhere else)

Client receives $a_0 = \langle x, q_0 \rangle$, $a_1 = \langle x, q_1 \rangle$

$x_i = a_0 + a_1$
Single server PIR

- Recap: the DDH problem

- Let $\mathbb{G}$ be a cyclic group of prime order $q$ generated by $g \in \mathbb{G}$

  - Challenger computes $\alpha, \beta, \gamma \leftarrow \mathbb{Z}_q, u \leftarrow g^\alpha, v \leftarrow g^\beta, w_0 \leftarrow g^{\alpha \beta}, w_1 \leftarrow g^\gamma$

  - $(u, v, w_0) = (g^\alpha, g^\beta, g^{\alpha \beta})$ is a Diffie-Hellman tuple

  - Challenger gives $(u, v, w_b)$ to the adversary where $b \leftarrow \{0, 1\}$

  - Hard for adversary to guess $\hat{b} = b$

- An extra property: given a DH tuple $(u, v_1, w_1)$, a tuple $(u, v_2, w_2)$, then $(u, v_1 \cdot v_2, w_1 \cdot w_2)$ is a DH tuple if and only if $(u, v_2, w_2)$ is a DH tuple
Single server PIR

- Server holds database $x \in \{0,1\}^n$
- Client inputs an index $i \in \{1, \ldots, n\}$
- To query an index $i$
  - Client prepares $n$ triples where the $i$th tuple is a non-DH tuple
    - Constructs $g^a$, $g^{b_j}$, $g^{c_j}$, for $j = 1, \ldots, n$
    - $c_j = ab_j$ for $j \neq i$, otherwise choose random $c_j$
  - Server computes and sends $\prod x_i g^{b_j}$ and $\prod x_i g^{c_j}$ (dot product)
  - If $(g^a, \prod x_i g^{b_j}, \prod x_i g^{c_j})$ is a DH tuple, then $x_i = 0$, otherwise $x_i = 1$

$O(n)$ query, $O(1)$ answer
Single server PIR with better communication

- Tradeoff between query length and answer length
- Restructure the database and view is as a matrix of size $\sqrt{n} \times \sqrt{n}$
- Bit $i$ is represented $(i_1, i_2)$, an element in the matrix
- Client constructs a PIR query with index $i_2$
- Server applies PIR on each row, returns one column (matrix multiplication)
- Client chooses the $i_1$-th item

$O(\sqrt{n})$ query, $O(\sqrt{n})$ answer
Even better communication

• Insight: though the answer is of length $\sqrt{n}$, the client only needs one element.

• Idea: can view the answer to the query as another database and run a second PIR on this DB!

• Recursion results in a complexity that is asymptotically smaller than $n^\epsilon$ for every constant $\epsilon > 0$

• Tradeoff is more compute
**ORAM**

- Memory contents changes with every query
- One client $\rightarrow$ one server
- Reads and writes
- Server process in $\text{polylog}(n)$

**PIR**

- Public, static DB
- Multiple clients $\rightarrow$ one server
- Traditionally only for reads
- Linear server work per query
PIR is still expensive

- Communication cost
  - Two-server PIR: $O(\log n)$
  - Single-server PIR: $\text{polylog}(n)$ from public key crypto assumptions

- Computation cost
  - Batching: batch multiple queries together in a single scan
  - Preprocessing: by offloading some work in a separate preprocessing phase, and by storing extra information, the “online” cost of a retrieval is less than a linear scan
Today’s reading: Pung
Next time: Vuvuzela

- A very different approach to anonymous messaging
- No longer using a database abstraction
- Do not need to use heavy crypto -> much more scalable
- Network traffic & dead drop access patterns leak information
  - Same chain of servers used to shuffle traffic & add cover traffic (all but one can be compromised)
  - Differential privacy offers a scalable way hiding metadata (albeit weaker)