Secure aggregation

- large number of clients, each holding $x_i$
- server wants to compute $\sum x_i$

Assume all clients have D-H public key $h_i = g^{s_i}$
Each pair of clients $i,j$ can compute shared key $K_{ij}$

Basic protocol:
- each client sends to server: $y_i = x_i - \sum_{i < j} K_{ij} - \sum_{i > j} K_{ij}$

  $\sum_{i} \text{mask}_i = 0$

  $\Rightarrow \sum_i y_i = \sum_i x_i$

Problem?
If server fails to receive a single $y_i$ value, entire protocol breaks down

Add recovery step
- Every client shares its D-H secret $s_i$ w/ other clients using $t$-out-of-$n$ secret sharing scheme
- Server can request shares of $s_i$ for any $y_i$ value it did not receive
Problem?

- Malicious server can break privacy of users.
- Delayed messages break privacy.

Solution

Users apply individual masks:

\[ y_i = x_i + \sum_{j<i} k_{ij} - \sum_{j<i} k_{ij} + c_i \]

Users also secret share \( c_i \) among the other clients.

Users

\[ y_i = x_i + \sum k_{ij} + c_i \]

\[ U = ?; i : \text{receives } y_i \]

Server

\[ U \]

? shares of \( c_i ; i \in U \)

? shares of \( s; i \in U \)
Many clients, multiple servers, n servers, secure against n-1 semi-honest servers, each client has input \( x_i \); want to compute \( \sum_i x_i \).

**Simple protocol:**

Client sends \( x_i \) to servers \( S_1, S_2, S_3 \).

Challenge: enforce that client's input satisfies some predicate

\[ s_1 \land (x_1) \] \( \) want to verify that \( C(x) = 1 \)

\[ s_2 \land (x_2) \] \( \)

\[ s_3 \land (x_3) \] \( \)

**Option 1:** Server can run MPC protocol evaluating \( C(x) \) to check if result is 1.

**Option 2:** Client can locally evaluate \( C(x) \), and determine the value on every wire of the circuit.

Servers can verify correctness by secretly evaluating a \( O(1) \)-depth circuit.
Option 3:

Client evaluates $C(x)$, $M$ = # multi. gates in $C$

- let $F$ be a poly. s.t. $F(t) =$ value on left input wire of gate $t$, for $1 \leq t \leq M$
- let $g$ be s.t. $g(t) =$ value on right input wire of gate $t$, for $1 \leq t \leq M$
- let $h = F \cdot g$

// $h(t) =$ value on output wire of gate $t$

- provide $\{x\}, \{F(x)\}, \{g(x)\}, \{h\}$

Servers

- generate $\{f\}$ and $\{g\}$ locally

$$f_i = \sum_{j=0}^{M} F(j) \cdot g_j$$

$$h_i = \sum_{j=0}^{M} f_i \cdot g_j$$

$$h(x) = \sum_i h_i \cdot x^i$$

$$\{h(x)\} = \sum \{h_i\}$$

Claim:

1) if client is honest, then $\hat{F} = F$, $\hat{g} = g$, $\hat{F} \cdot \hat{g} = h$

2) if $h \neq 0$ then client shared is incorrect, then $\hat{F} \cdot \hat{g} \neq h$

Servers then check whether

$$P(t) = \text{Tr}(\hat{F}(t) \cdot g(t) - h(t)) \equiv 0$$

i.e., check if $P(\hat{r}) = 0$ at a random $r$
If \( P(c) \neq 0 \), then \( P \circ \mathcal{L}(r) = 0 \) \( \leq \frac{\deg(P)}{\lvert r \rvert} \)

- Server agrees on \( r \)
- Each server locally computes \( \hat{r} \cdot \hat{h}(r) \), \( \hat{f}(r) \), \( \Delta r \cdot \hat{g}(r) \)
- Client will also share Beamer triple \( (a), (b), (c) \), \( c = ab + \delta \)
- Servers use Beamer triple to compute \( \Delta c \cdot \hat{f}(r) \cdot \hat{g}(r) + \delta \)
- Check that \( \int (\hat{f}(r) \cdot \hat{g}(r) - h(r)) = 0 \)

\[ + (\hat{f}(r) \cdot \hat{g}(r) - h(r)) + \delta = 0 \]