- Scribes?
- lecture recording

Secure aggregation

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

Assume all clients have D-H public key $h_i = g^{s_i}$

each pair of clients i,j can compute shared key $k_{i,j}$

Basic protocol:

each client sends to server: $y_i = x_i + \underbrace{\sum_{i<j} k_{i,j} - \sum_{i>j} k_{i,j}}_{mask_i}$

$$\sum_i mask_i = 0$$

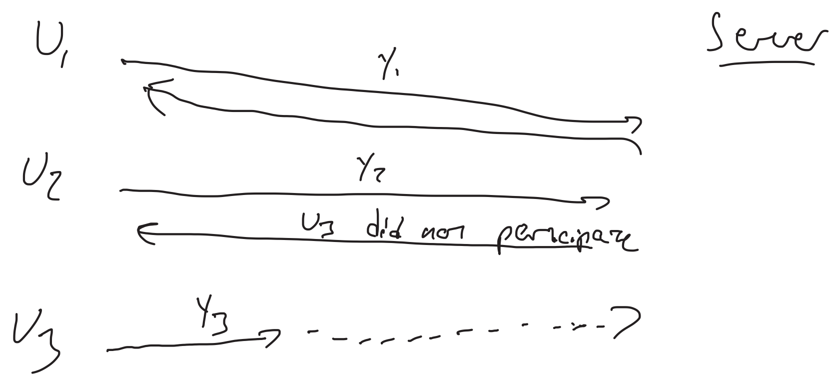
$$\Rightarrow \sum_i y_i = \sum x_i$$

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?

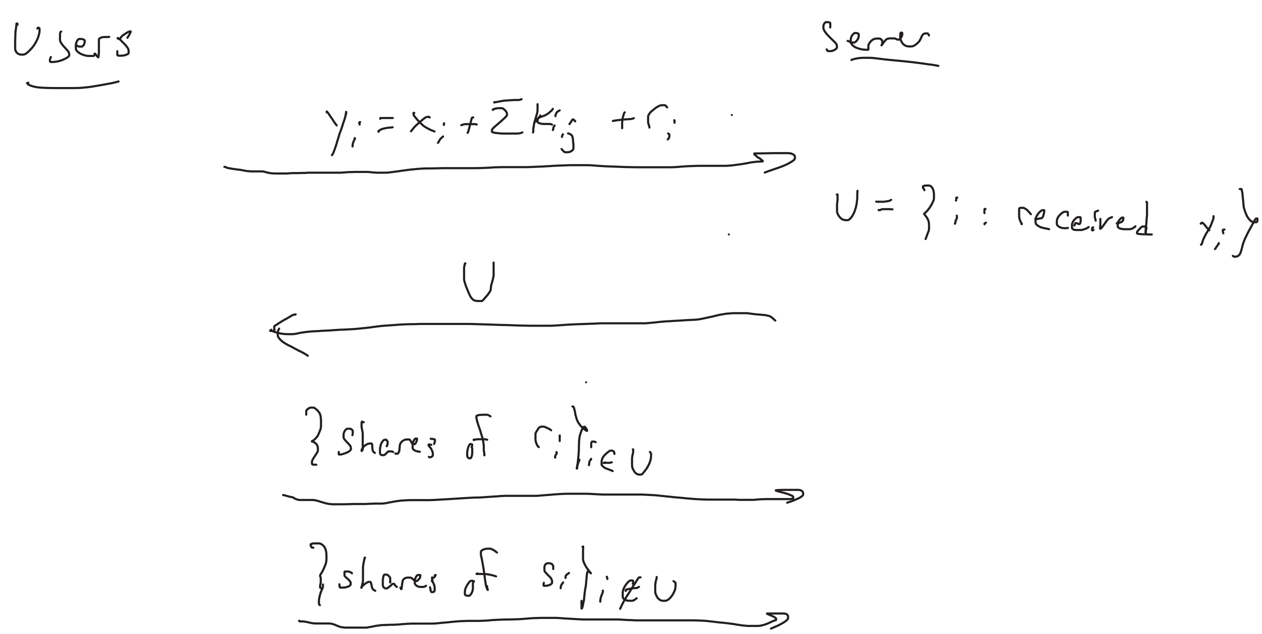
- malicious server can break privacy of users
- delayed messages break privacy

Solution

Users apply individual masks

$$y_i = x_i + \sum_{i<j} k_{i,j} - \sum_{i>j} k_{i,j} + r_i$$

users also secret share r_i among the other clients



Prio

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 secret shares x_i

$x_i \xrightarrow{x_{i,1}} S_1$

$x_i \xrightarrow{x_{i,2}} S_2$

$x_i \xrightarrow{x_{i,3}} S_3$

Challenge: enforce that client's input satisfies some predicate

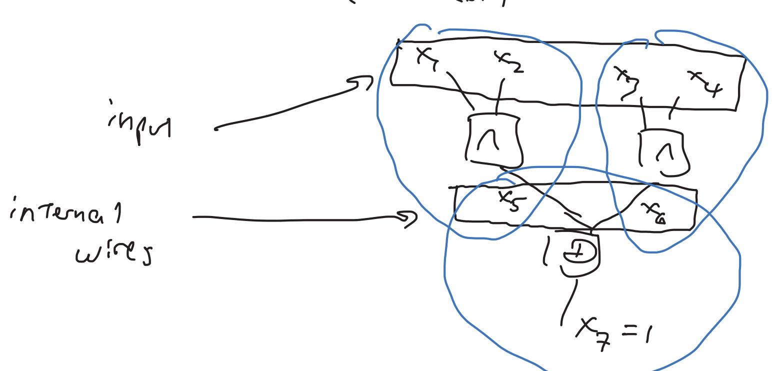
S_1 $[x]_1$ want to verify that $C(x)=1$

S_2 $[x]_2$

S_3 $[x]_3$

Option 1: Server can run MPC protocol evaluating $C(x)$ & 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 securely evaluating a $O(1)$-depth circuit

Option 3 :

Client evaluate $C(x)$, $M = \#$ mult. gates in C

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

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

- provide $\underbrace{[x],\ [f(0)],\ [g(0)],\ [h]}$

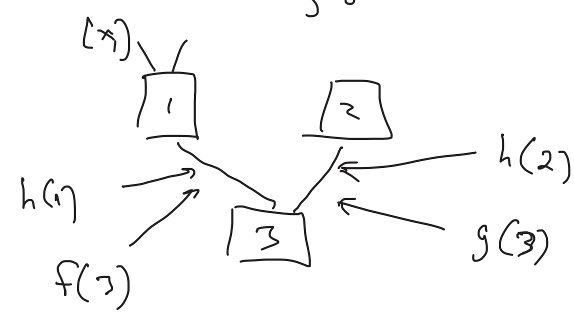
Servers

- generate $[\hat{f}]$ and $[\hat{g}]$ locally

$$f_i = \sum_{j=0}^{M} f(j) \cdot \sigma_j \qquad [f_i] = \sum_{j=0}^{M} [f(j)] \cdot \sigma_j$$

$$h(x) = \sum h_i x^i$$

$$[h(x)] = \sum x^i [h_i]$$



Claim :

1) if client is honest, then $\hat{f} = f$, $\hat{g} = g$, $\hat{f} \cdot \hat{g} = h$
2) if h that client shared is incorrect, then $\hat{f} \cdot \hat{g} \neq h$

Servers then check whether

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

i.e. check if $P(r) = 0$ at a random r

if $P(\tau) \neq 0$, then $\Pr_r [P(r) = 0] \leq \frac{\deg(P)}{|\mathbb{F}|}$

- Servers agree on r
- each server locally computes $[r \cdot h(r)]$, $[\hat{f}(r)]$, $[r \cdot \hat{g}(r)]$
- Client will also share Beaver triple
  $[a], [b], [c]$, $c = ab + \delta$
- Servers use Beaver triple to compute $[r \cdot \hat{f}(r) \cdot \hat{g}(r) + \delta]$
- check that $[r(\hat{f}(r) \cdot \hat{g}(r) - h(r))] = [0]$

$$r \cdot (\hat{f}(\tau) \cdot \hat{g}(\tau) - h(\tau)) + \delta \stackrel{?}{=} 0$$