CS208: Applied Privacy for Data Science
Other Distributed DP Models: Shuffling and MPC

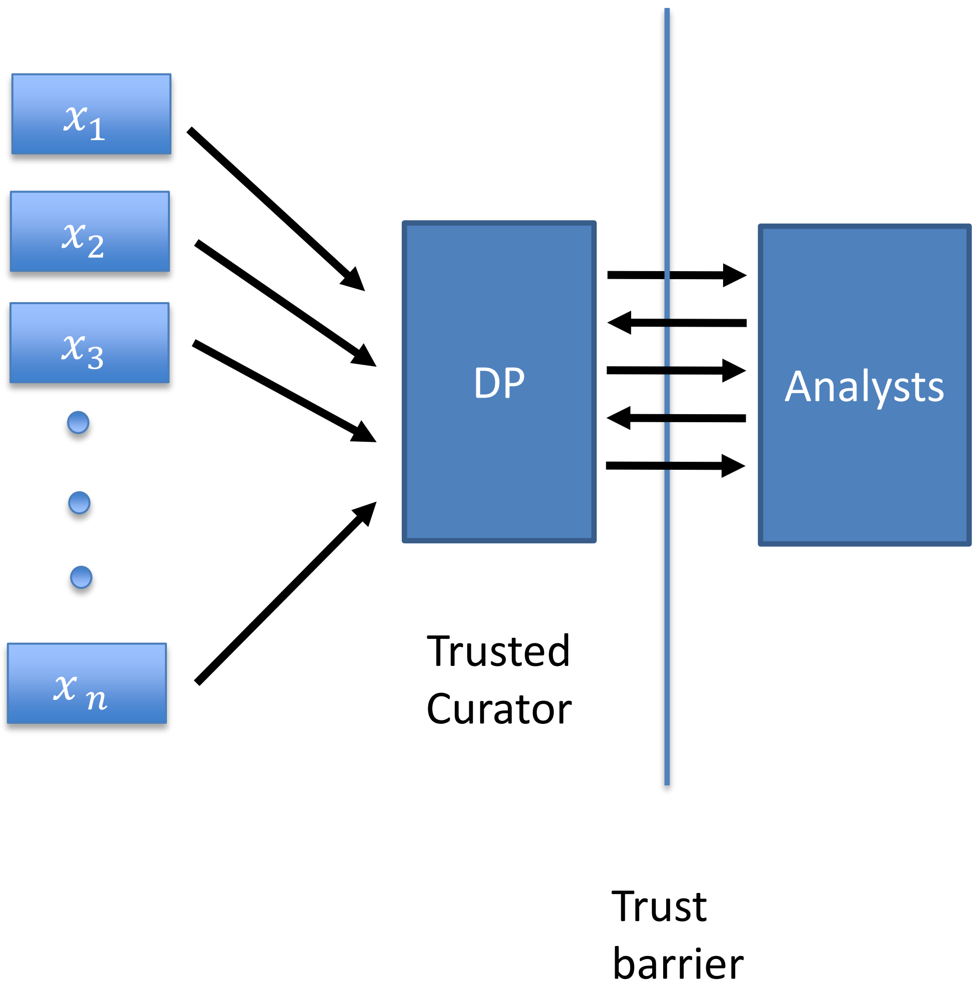
School of Engineering & Applied Sciences
Harvard University

April 9, 2025

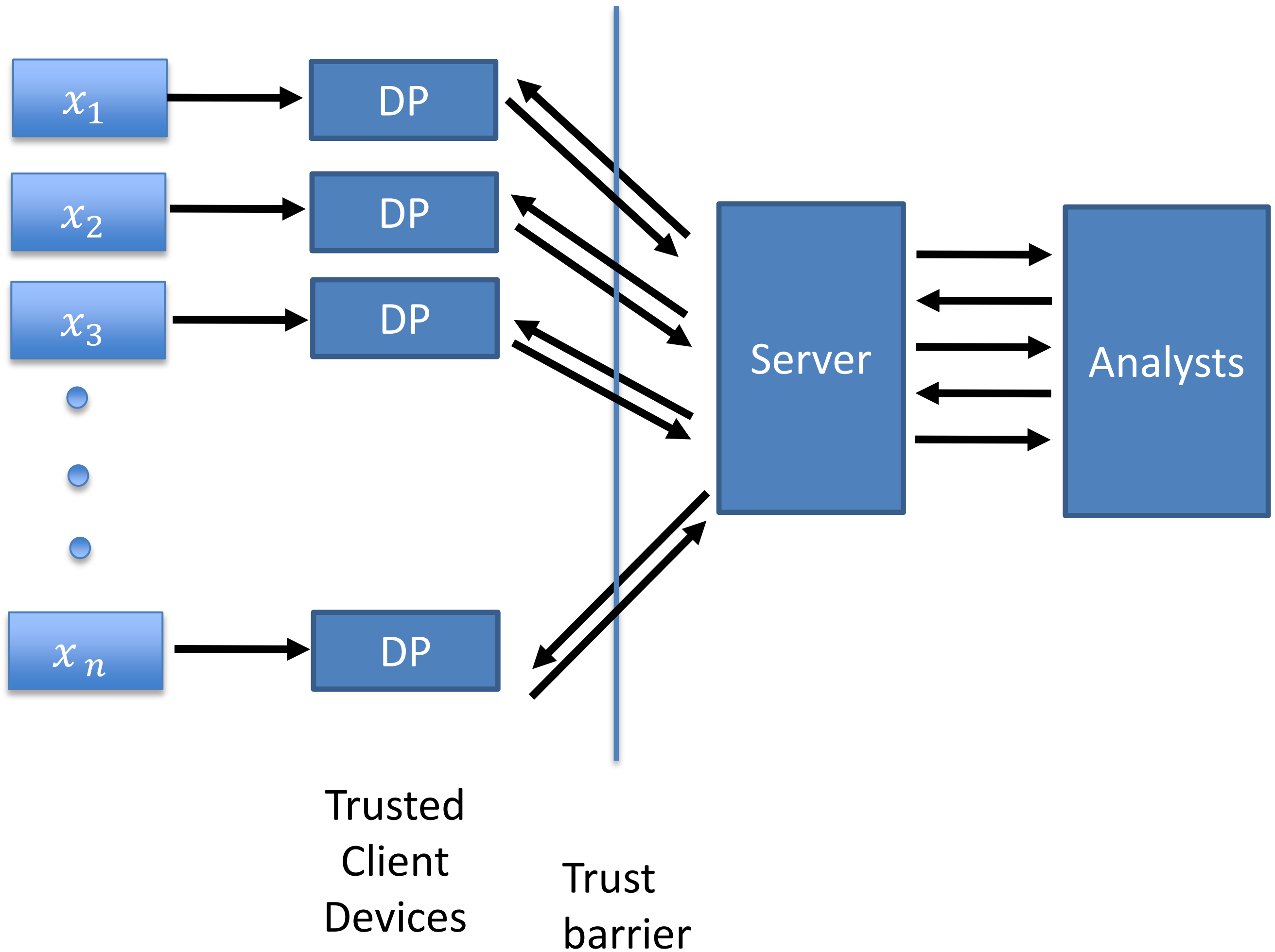
Housekeeping

- No pset due this week, hw8b due Fri 4/18.
- Project Deadlines
  - Detailed project description: Fri 4/11
  - Full project paper: Wed 4/30
  - Revision of paper: Thu 5/8
  - Poster session: Thu 5/8, 9am-12pm in the SEC.
  - 3 late days per group on project deadlines.
  - Come to OH to discuss!
- Charles River Symposium on Privacy (CRiSP) Fri in SEC

Central DP

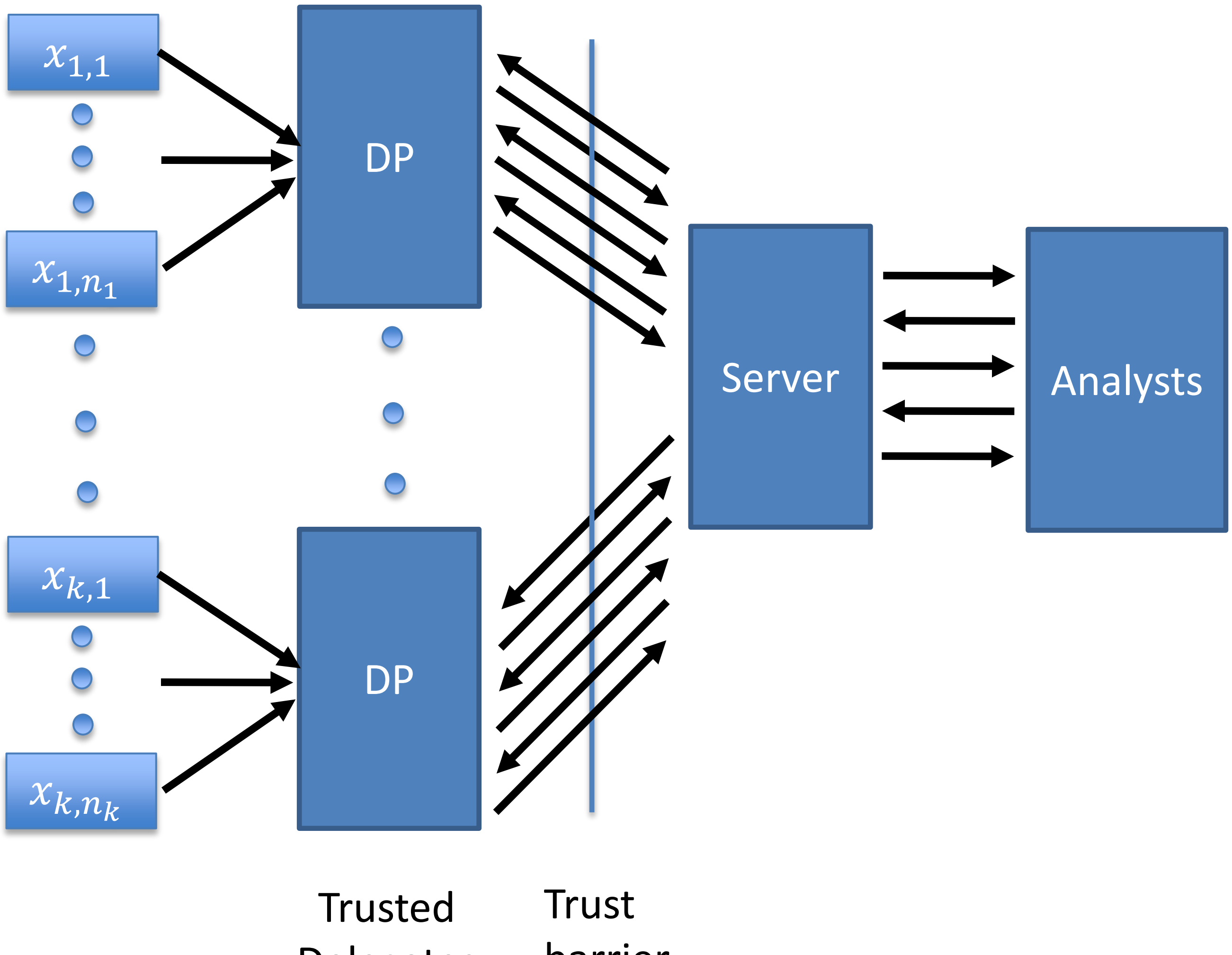


Local DP



x_1
x_2
x_3
x_n
DP
DP
DP
DP
Server
Analysts
Trusted Client Devices
Trust barrier

Federated DP

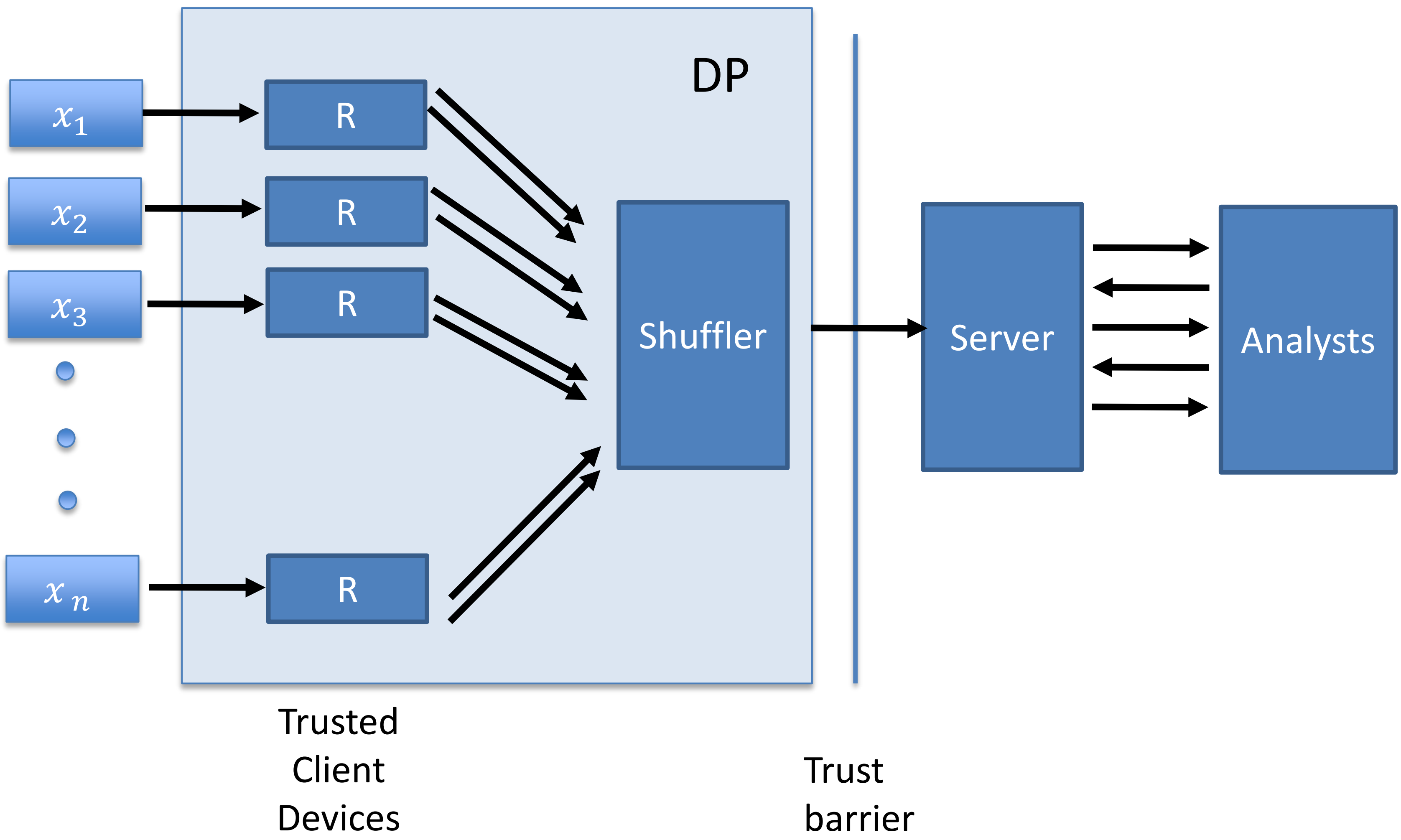


Other Models

- Can we get the “best of both worlds”?
  - Privacy protections like the local model
  - Accuracy like the central model

- Two approaches
  - The shuffle model
  - Using cryptography (secure multiparty computation)

Shuffle DP



x_1
x_2
x_3
x_n
R
R
R
R
DP
Shuffler
Server
Analysts
Trusted Client Devices
Trust barrier

Binary Sum with Shuffle DP

- Suppose each $x_i \in \{0,1\}$ and R = (weak) randomized response

$$R(x_i) = \begin{cases} \mathrm{Ber}(1/2) & \text{w.p. } p = o(1) \\ x_i & \text{w.p. } 1-p \end{cases}$$

Analyzing the privacy of client i:

Accuracy: error $O(\sigma[S]) = O\left(\frac{\sqrt{\ln(1/\delta)}}{\varepsilon}\right)$. No dependence on n!

Privacy Amplification by Shuffling

$$R(x_i) = \begin{cases} \mathrm{Ber}(1/2) & \text{w.p. } p = \frac{c \ln(1/\delta)}{\varepsilon^2 n} \\ x_i & \text{w.p. } 1-p \end{cases}$$

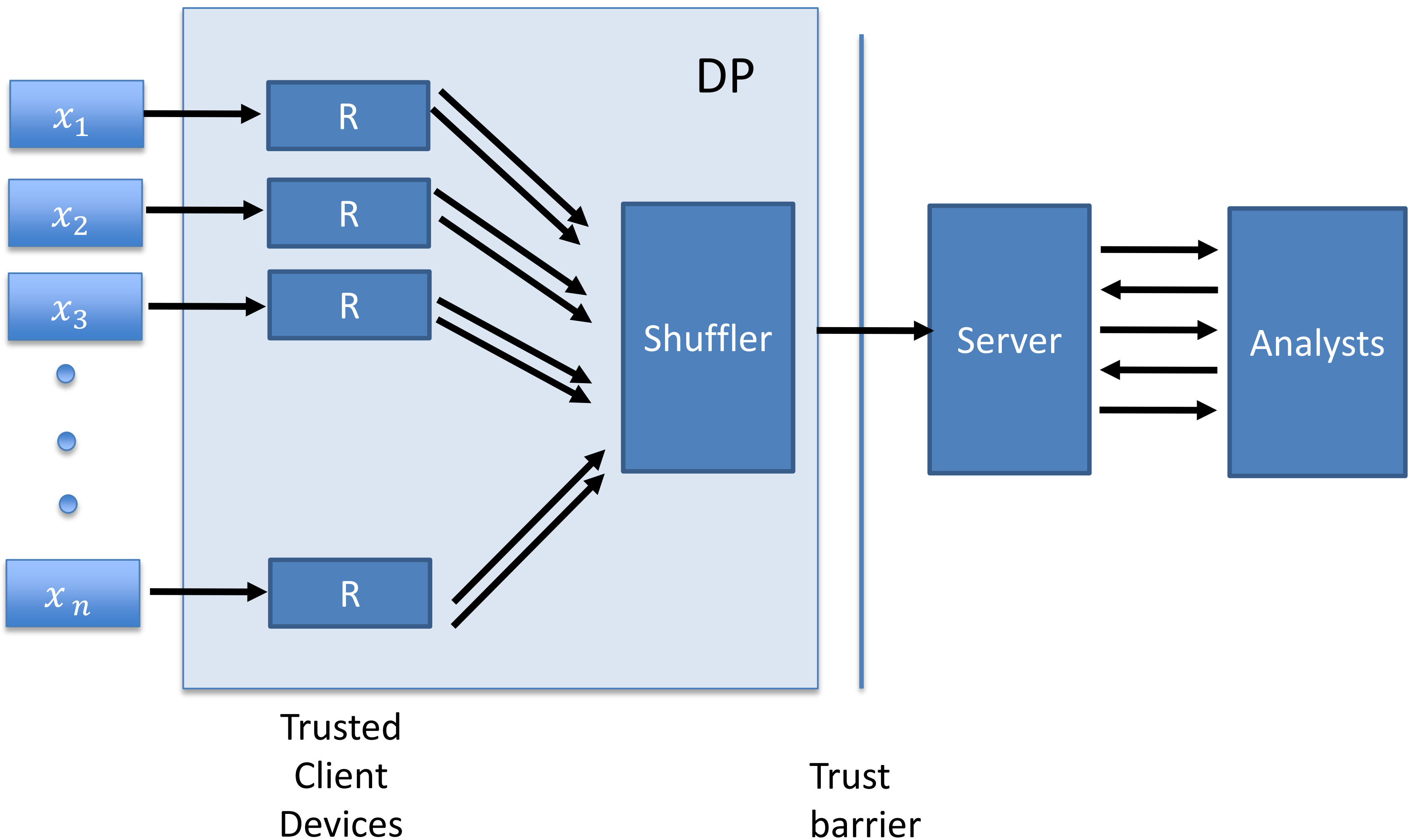
- Note that R is only $\varepsilon_0 = \ln\left(\frac{1-p/2}{p/2}\right) \approx \ln\left(\frac{\varepsilon^2 n}{\ln(1/\delta)}\right)$-DP.

- General amplification thm: if R is ε_0-DP, then $M(x_1, \dots, x_n) = \mathrm{Shuffle}\big(R(x_1), \dots, R(x_n)\big)$ is (ε, δ)-DP with relation as above

Shuffle vs. Central DP

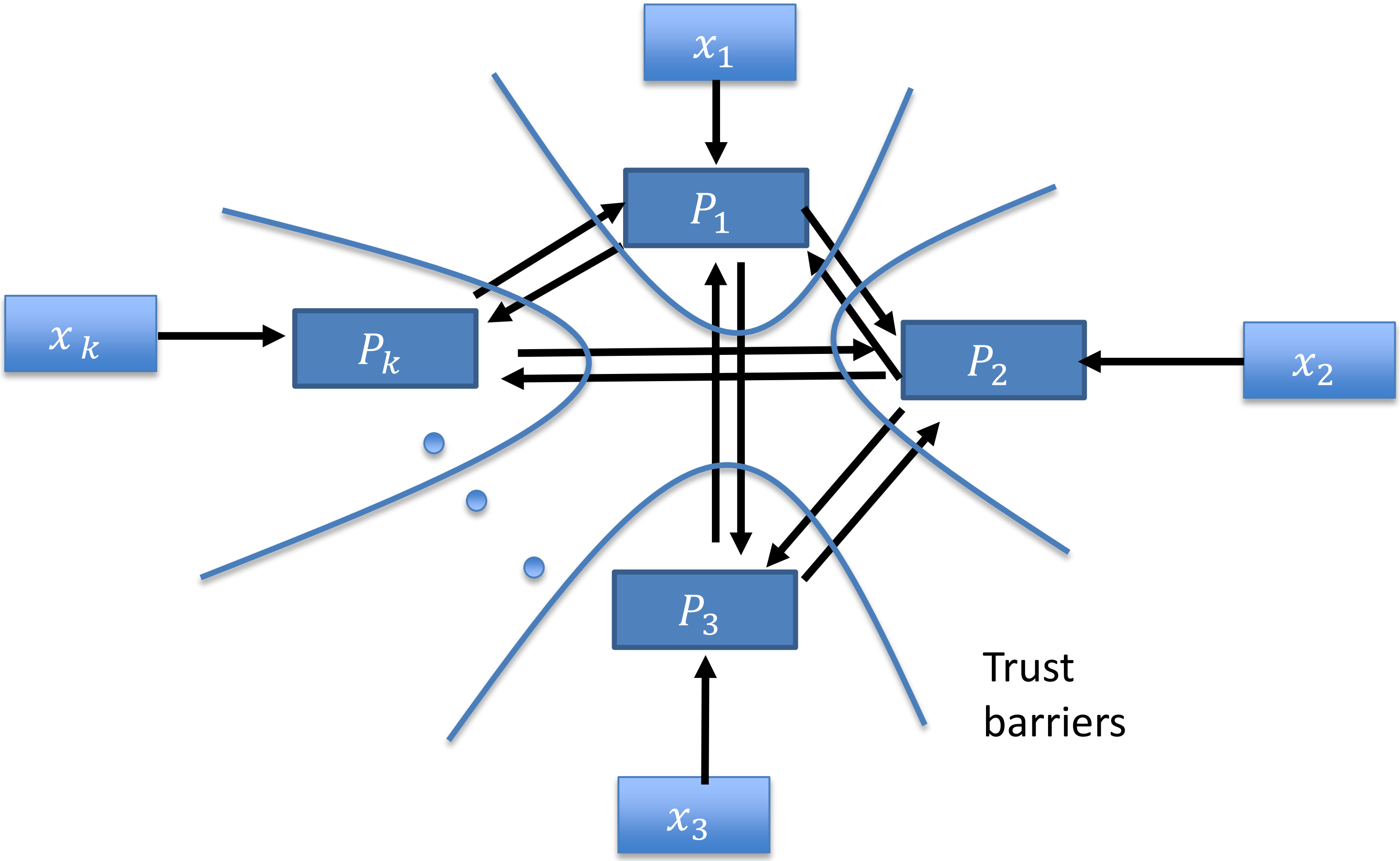
- There is a multi-message shuffle-DP protocol with error $O(1/\varepsilon)$, matching the central model.
- For other problems, shuffle seems to give accuracy strictly between local and central.
  - E.g. best known error for histograms: $O\left(\frac{\ln(1/\delta)}{\varepsilon^2}\right)$.
  - Don't know matching upper & lower bounds for most problems, especially for multi-message shuffle protocols.
- Q: trust considerations for shuffle model?

Shuffle DP



x_1
x_2
x_3
x_n
R
R
R
R
DP
Shuffler
Server
Analysts
Trusted Client Devices
Trust barrier

Secure Multiparty Computation



Requirement: At end of protocol, each party P_i learns $f_i(x_1, \ldots, x_n)$ and nothing else!

Secret Sharing

Privately do the following:

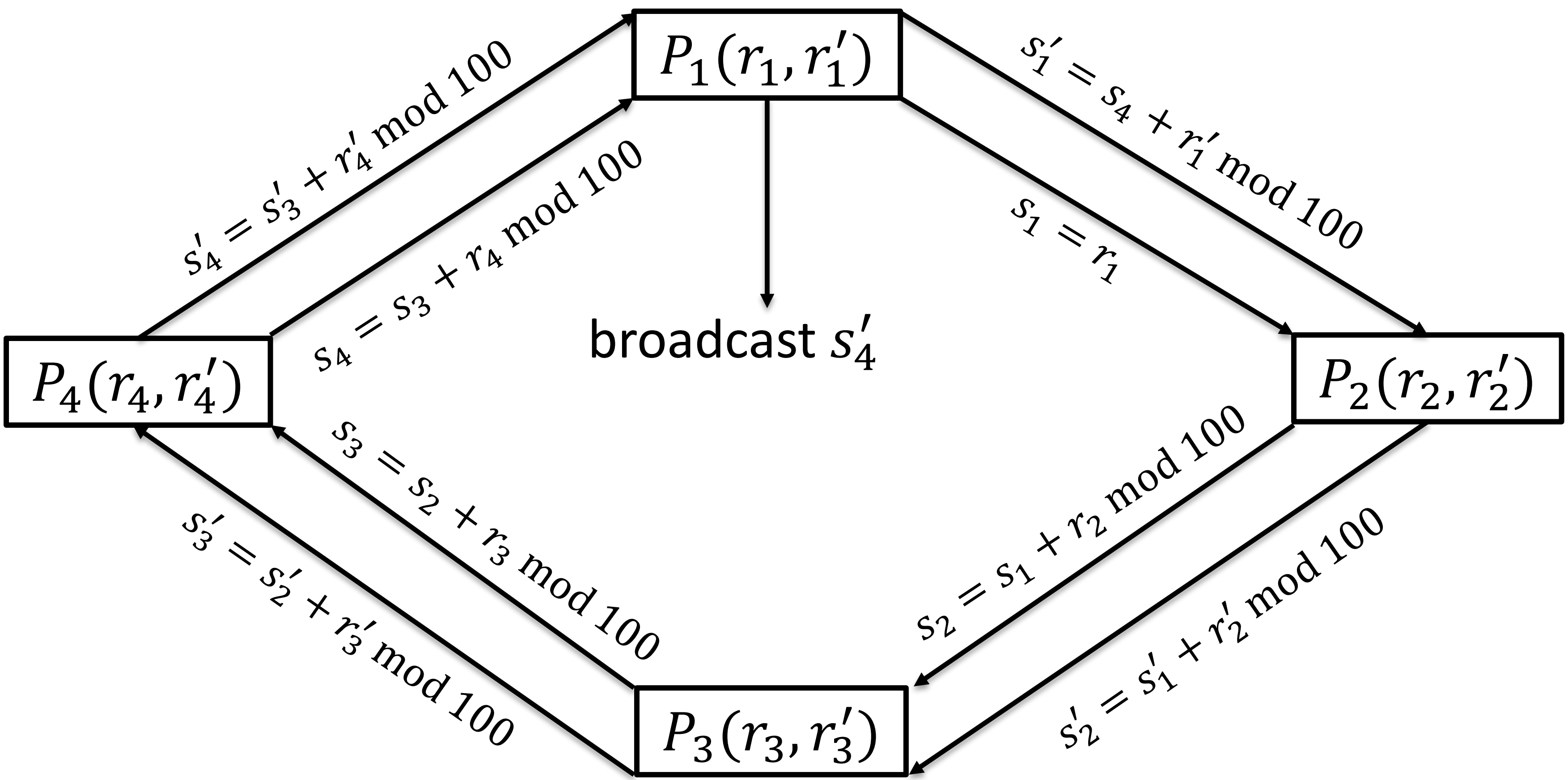
1. Write down $x_i = (\text{your height in inches}) - 45$
2. Choose a uniformly random $r_i \in \{0,1,\ldots,99\}$.
3. Let $r_i' = (x_i + 100 - r_i) \bmod 100$.
   (mod 100 = keep only last 2 digits)

r_i and r_i' are secret shares of x_i:

- Each one reveals nothing about x_i
- But with both, can reconstruct x_i:

Group MPC Example: Height Sum

Protocol for calculating $\sum_i x_i$ at your table:



Pseudocode

1. Each party P_i secret-shares their input x_i into (r_i, r_i')

2. Party P_1 sets $s_1 = r_1$, sends to P_2.

3. For $i = 2, \ldots, n$, party P_i does the following:
   - Receive s_{i-1} from P_{i-1}.
   - Send the value $s_i = s_{i-1} + r_i \bmod m$ to P_{i+1} (P_1 if $i = n$)

4. Party P_1 receives s_n from P_n, sends $s_1' = s_n + r_1' \bmod m$ to P_2

5. For $i = 2, \ldots, n$, party P_i does the following:
   - Receive s_{i-1}' from P_{i-1}.
   - Send the value $s_i' = s_{i-1}' + r_i \bmod m$ to P_{i+1} (P_1 if $i = n$)

6. Party P_1 broadcasts result s_n'

Properties

- Claim: No one learns anything other than $\sum_i x_i \bmod m$.

- Proof idea: In addition to the broadcast result $s'_n = \sum_i x_i \bmod m$ and their own input, party P_i for $1 < i < n$ sees:

Reflection Questions

Discuss with group and in Google form (via yellkey)

1. Identify at least one benefit of this protocol for computing a sum “with privacy”.

2. Identify at least one limitation or assumption of this protocol.

MPC is Always Possible (in theory)

Theorem (1980’s): Assume that secure cryptography exists. Then for all polynomial-time computable functions $f_1, \dots, f_n$ (even randomized), there is a polynomial-time secure MPC protocol with security against:

- All feasible (e.g. polynomial-time) adversaries
- Even if they deviate from the protocol
- Even if they control $n - 1$ parties

DP+MPC

Applying Secure MPC to f_1=any central DP algorithm, we get a protocol Π

- Accuracy of central DP
- Privacy of local DP against feasible adversaries A
  - Even ones that deviate from protocol
  - And corrupt up to $n-1$ parties

Q: Why aren't we done?

A:

Ways to make MPC more efficient

- Focus on specific functionalities (e.g. summation without noise)
- Restrict to passive (“honest but curious”) adversaries
- Restrict sizes of coalitions (“threshold adversaries”)
- Use trusted hardware (secure enclaves, Intel SGX)

PETs: DP vs. Crypto

Model	Utility	Privacy	Who Holds Data?
Centralized Differential Privacy	statistical analysis of dataset	individual-specific info	trusted curator
Local or Federated Differential Privacy	statistical analysis of dataset	individual-specific info	original users (or delegates)
Secure Multiparty Computation	any query desired	everything other than result of query	original users (or delegates)
Fully Homomorphic (or Functional) Encryption	any query desired	everything (except possibly result of query)	untrusted server