# The Poisson Binomial Mechanism for Unbiased Federated Learning with Secure Aggregation


Wei-Ning Chen
Stanford University

Ayfer Özgür
Stanford University

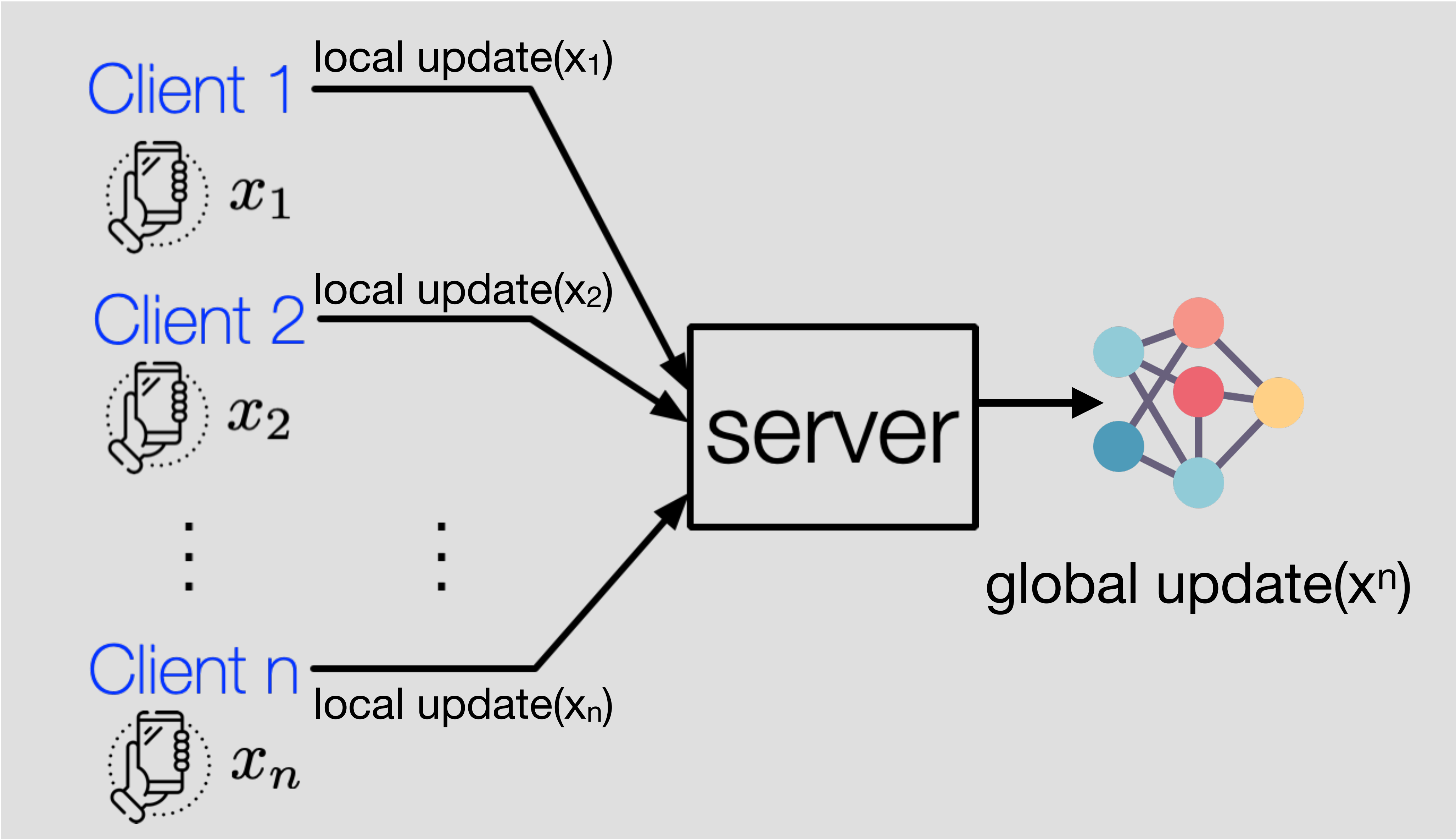
Peter Kairouz
Google Research



ICML 2022

# Federated Learning with Differential Privacy

- Objectives of private federated learning (FL)
  - Keep clients data **on device**
  - Ensure trained models differentially private (DP)

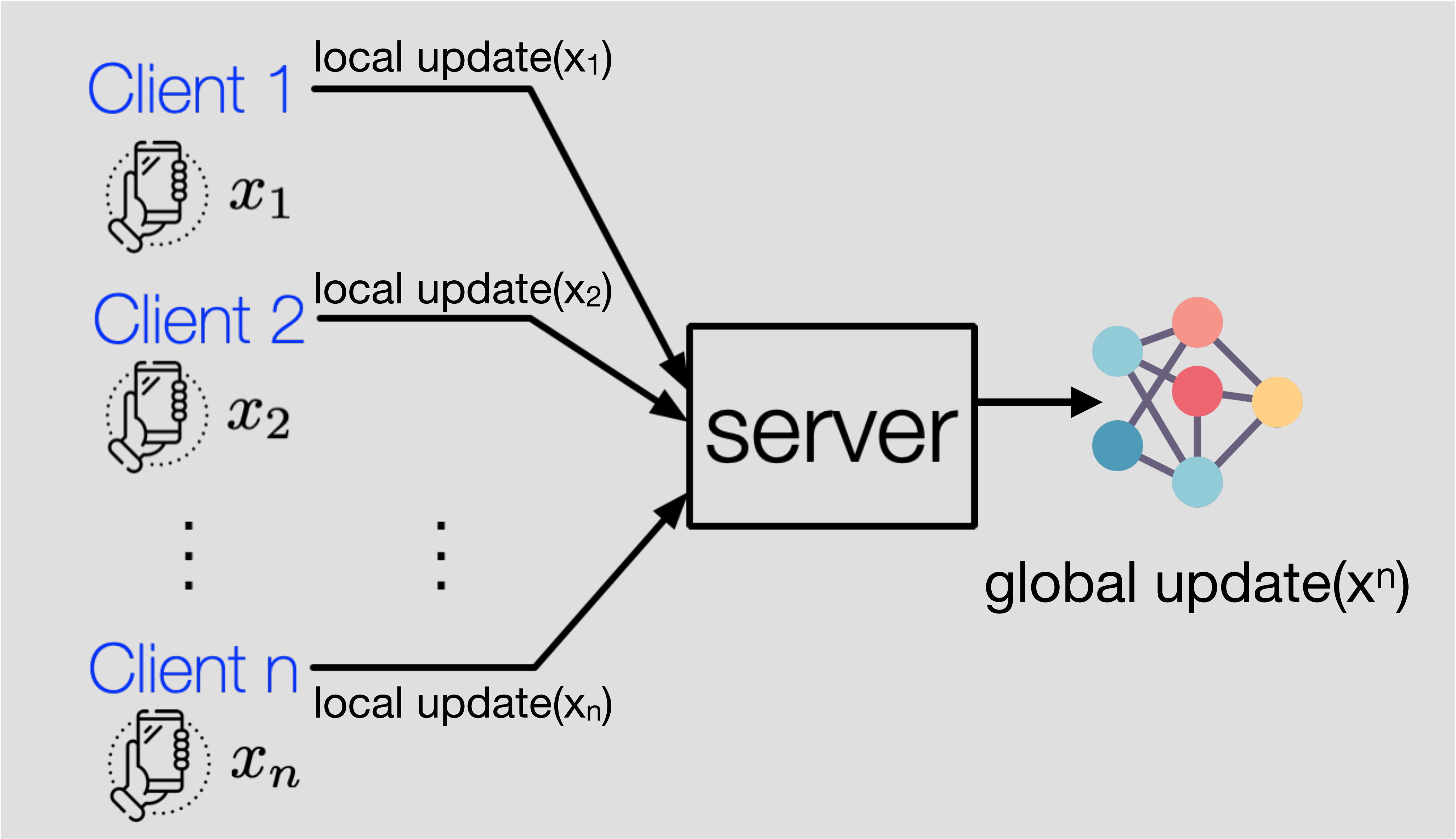



Indeed, we are mostly interested in Rènyi DP, which allows for tighter privacy accounting: $D_\alpha\left(\mathcal{M}(x_1, ..., x_n) \| \mathcal{M}(x_1', ..., x_n)\right) \leq \varepsilon$

# Federated Learning with Differential Privacy

- Objectives of private federated learning (FL)
  - Keep clients data **on device**
  - Ensure trained models differentially private (DP)

- Example: the Gaussian mechanism

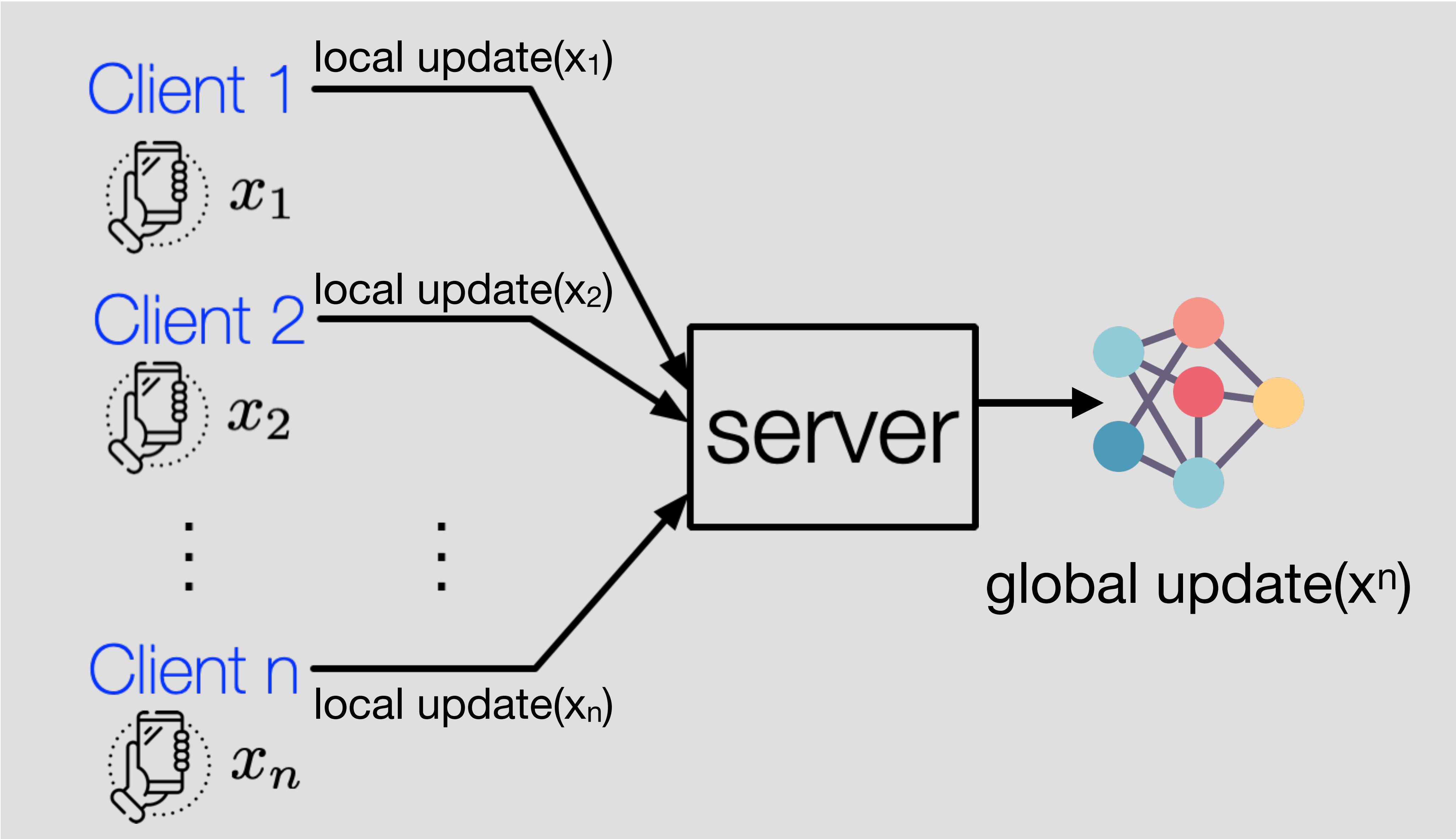



Indeed, we are mostly interested in Rènyi DP, which allows for tighter privacy accounting: $D_\alpha\left(\mathcal{M}(x_1,...,x_n)\|\mathcal{M}(x'_1,...,x_n)\right) \leq \varepsilon$

# Federated Learning with Differential Privacy

- Objectives of private federated learning (FL)
  - Keep clients data **on device**
  - Ensure trained models differentially private (DP)

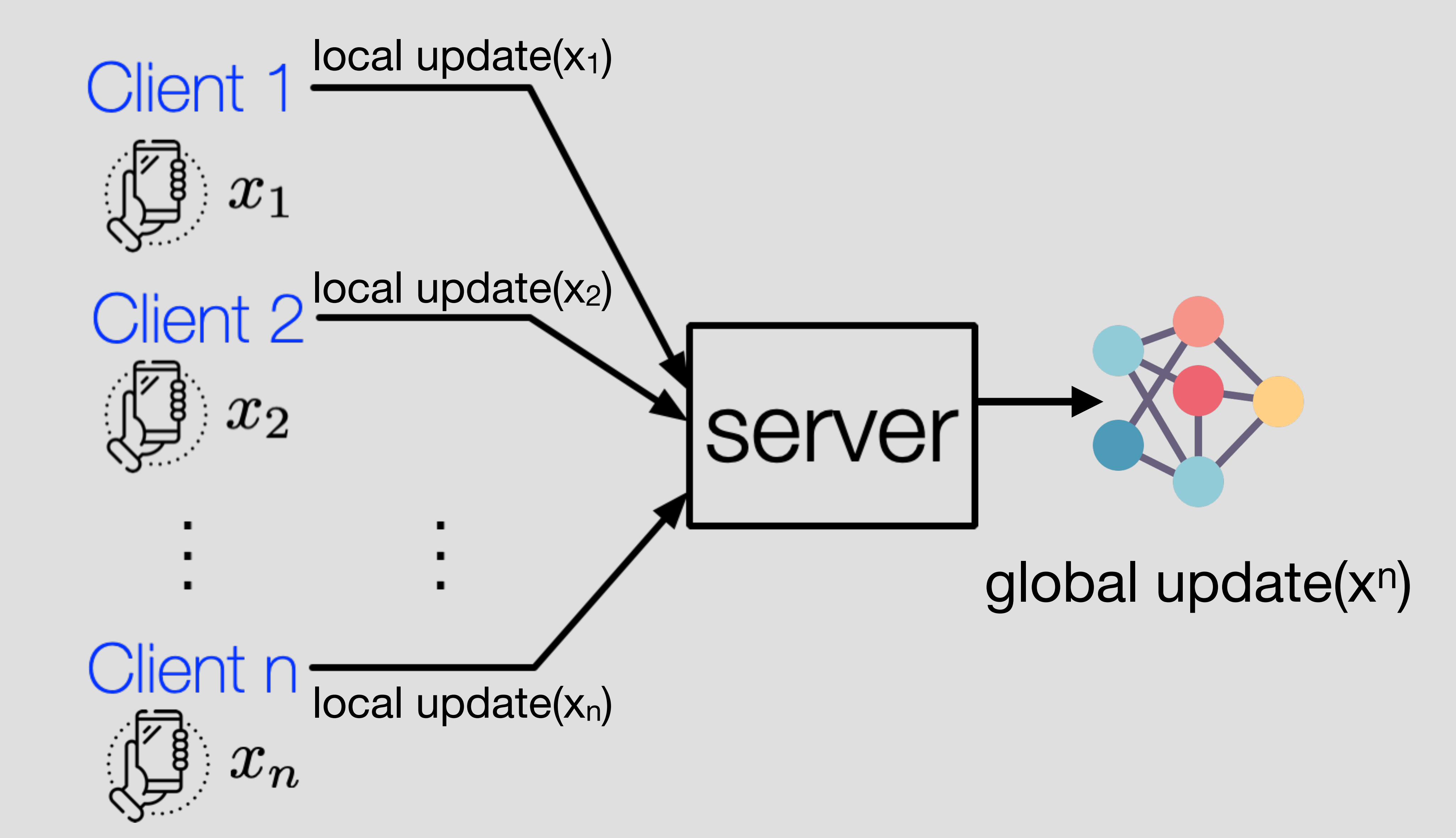
- Example: the Gaussian mechanism
  - In each round, server samples a batch of clients




Indeed, we are mostly interested in Rènyi DP, which allows for tighter privacy accounting: $D_\alpha\left(\mathcal{M}(x_1, ..., x_n) \| \mathcal{M}(x_1', ..., x_n)\right) \leq \varepsilon$

# Federated Learning with Differential Privacy

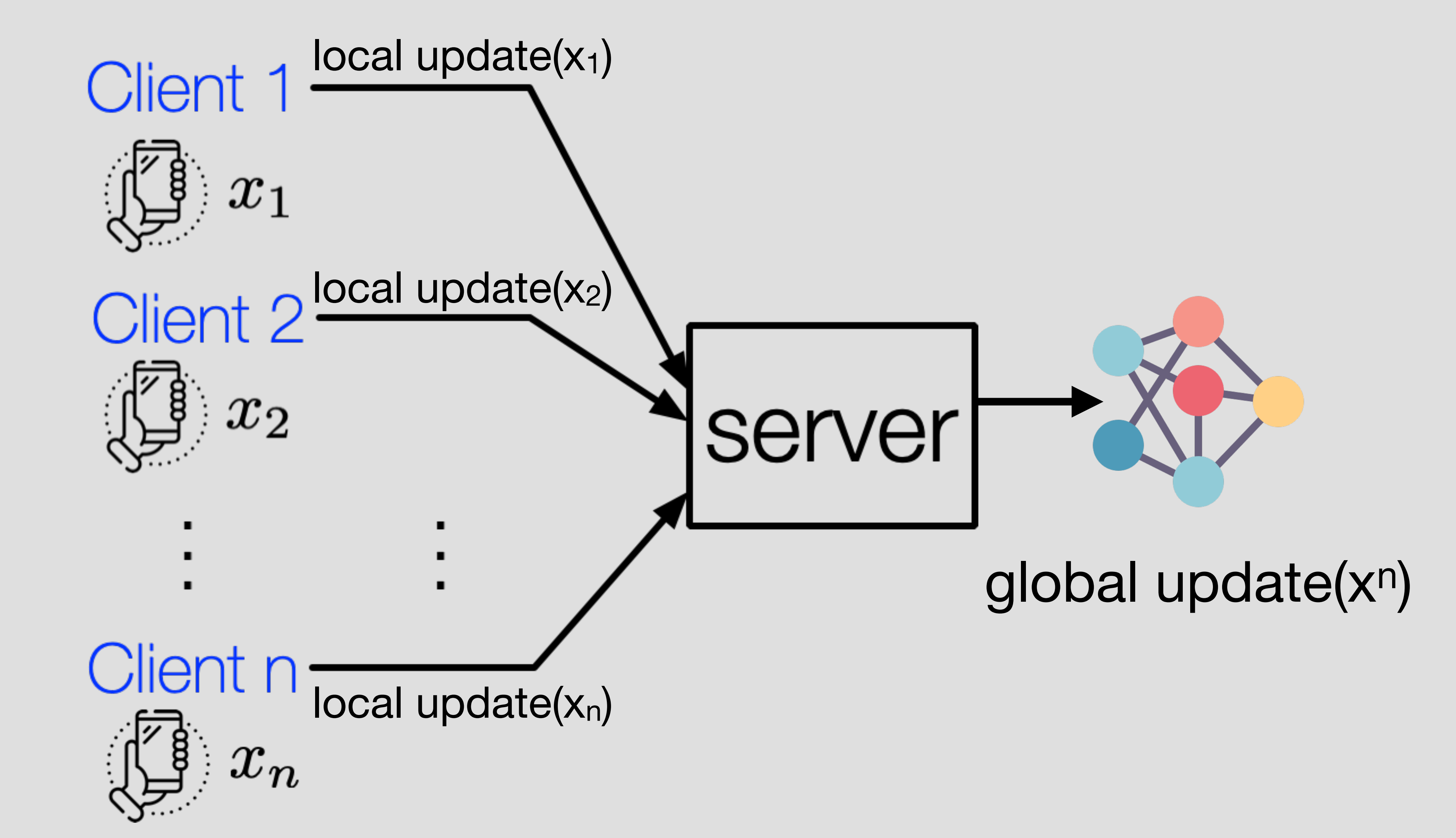
- Objectives of private federated learning (FL)
  - Keep clients data **on device**
  - Ensure trained models differentially private (DP)




- Example: the Gaussian mechanism
  - In each round, server samples a batch of clients
  - Each client computes a (clipped) local model update (e.g. a local gradient)

Indeed, we are mostly interested in Rènyi DP, which allows for tighter privacy accounting: $D_\alpha\left(\mathcal{M}(x_1,...,x_n)\|\mathcal{M}(x'_1,...,x_n)\right) \leq \varepsilon$

# Federated Learning with Differential Privacy




- Objectives of private federated learning (FL)
  - Keep clients data **on device**
  - Ensure trained models differentially private (DP)
- Example: the Gaussian mechanism
  - In each round, server samples a batch of clients
  - Each client computes a (clipped) local model update (e.g. a local gradient)
  - Server computes the average of all local updates and adds Gaussian noise satisfying DP[1]:

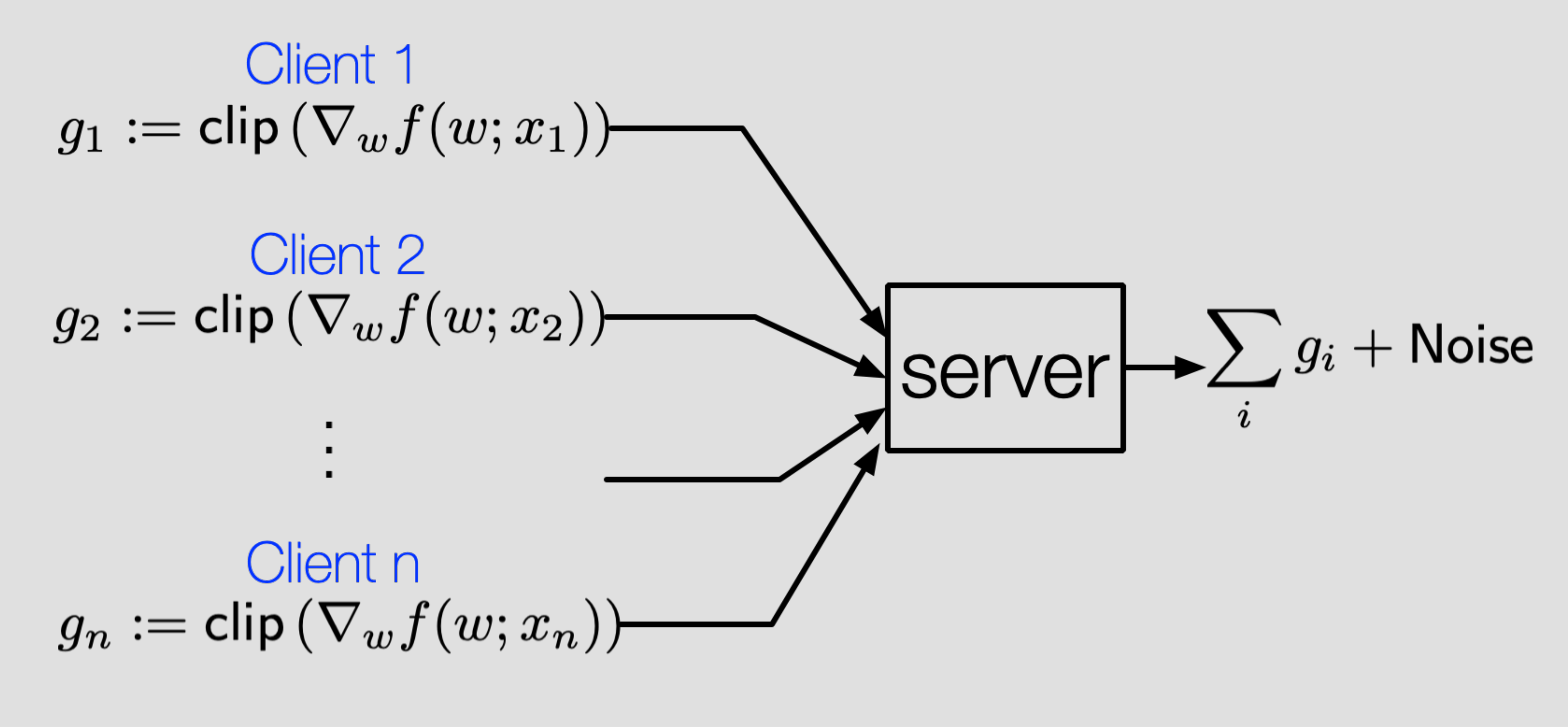
$$\forall \mathcal{S}, \mathbb{P}\left\{\text{update}(x_1, x_2, ..., x_n) \in \mathcal{S}\right\} \leq e^{\varepsilon}\mathbb{P}\left\{\text{update}(x_1', x_2, ..., x_n) \in \mathcal{S}\right\} + \delta$$

  - Server updates the global model

Indeed, we are mostly interested in Rènyi DP, which allows for tighter privacy accounting: $D_\alpha\left(\mathcal{M}(x_1, ..., x_n) \| \mathcal{M}(x_1', ..., x_n)\right) \leq \varepsilon$

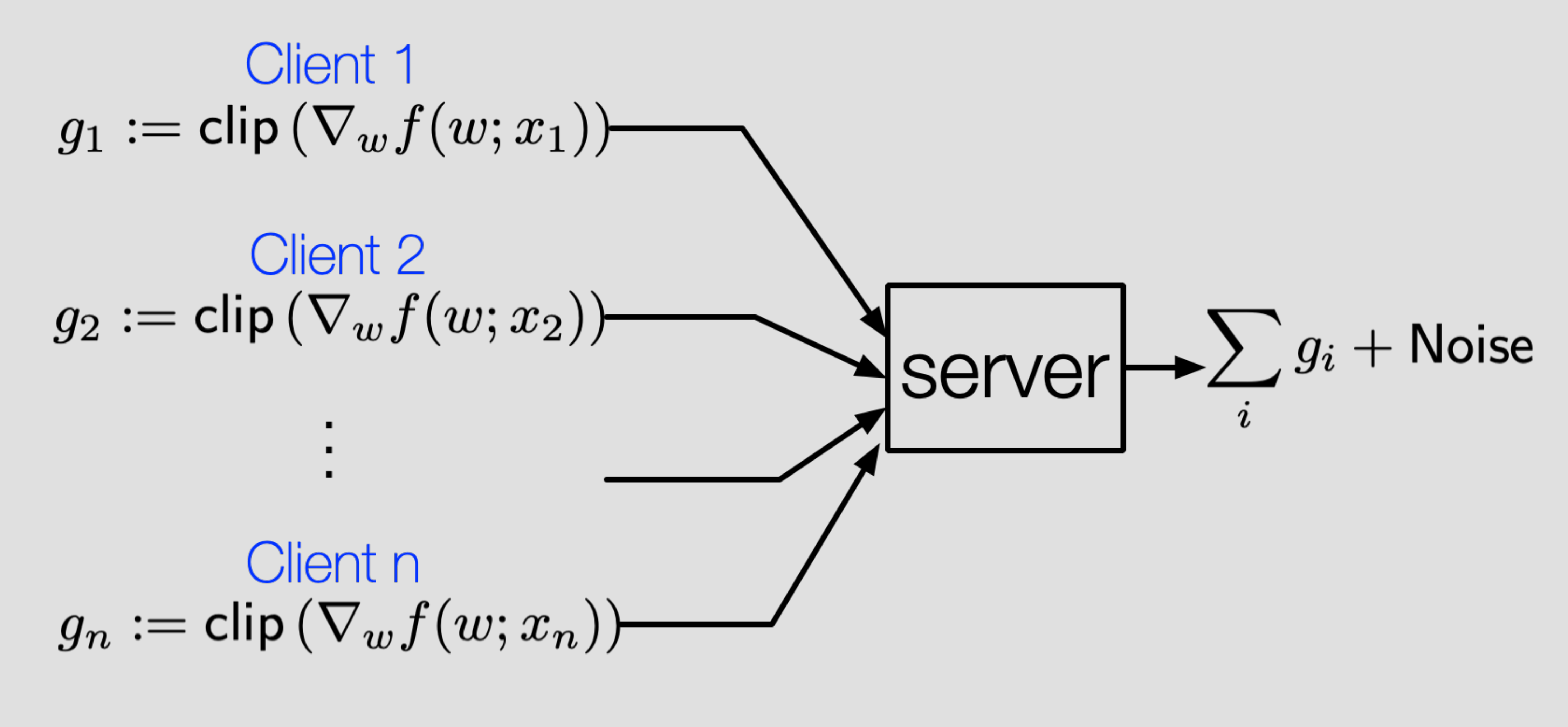
# Federated Learning with Differential Privacy

- FL with **central** differential privacy (DP)



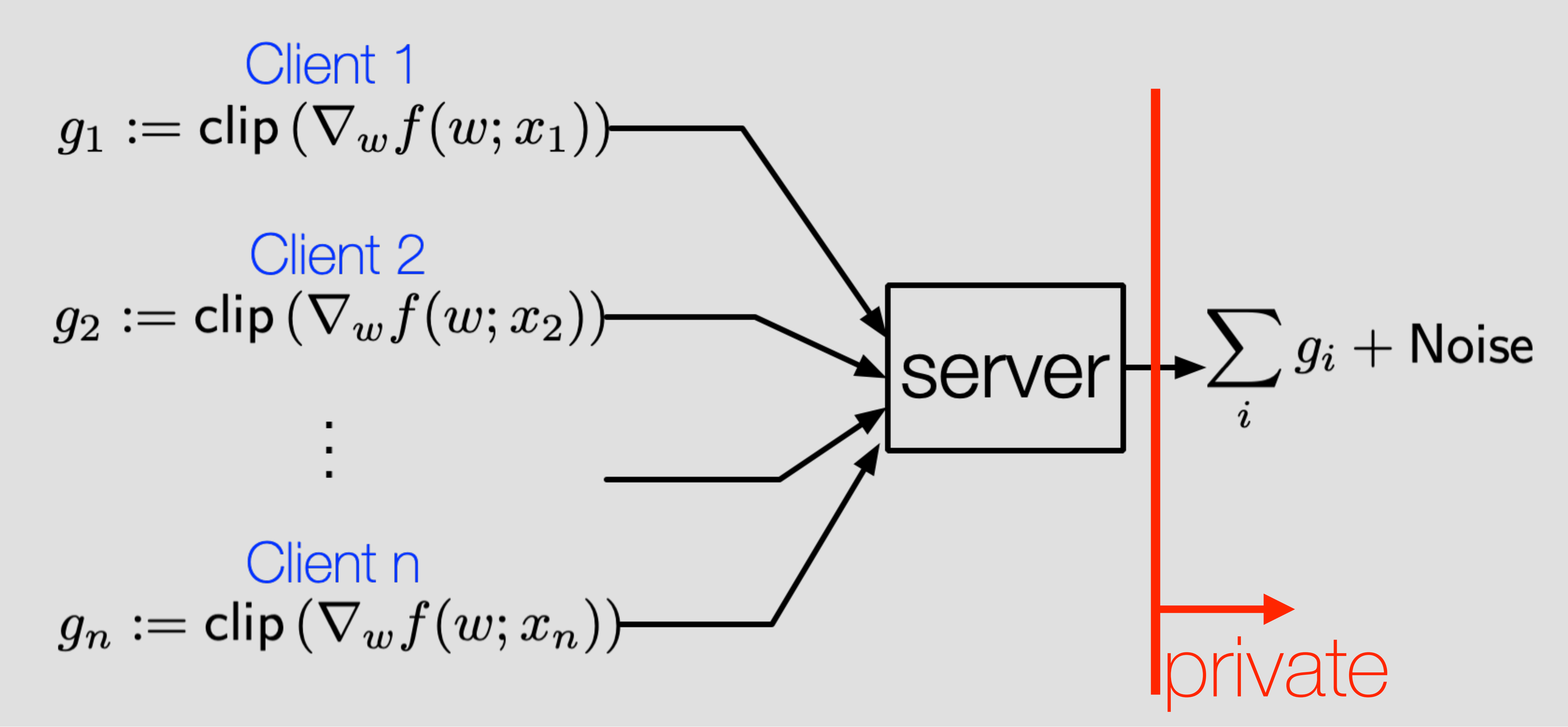
# Federated Learning with Differential Privacy

- FL with **central** differential privacy (DP)
  - Server collects local model updates and perturbs them



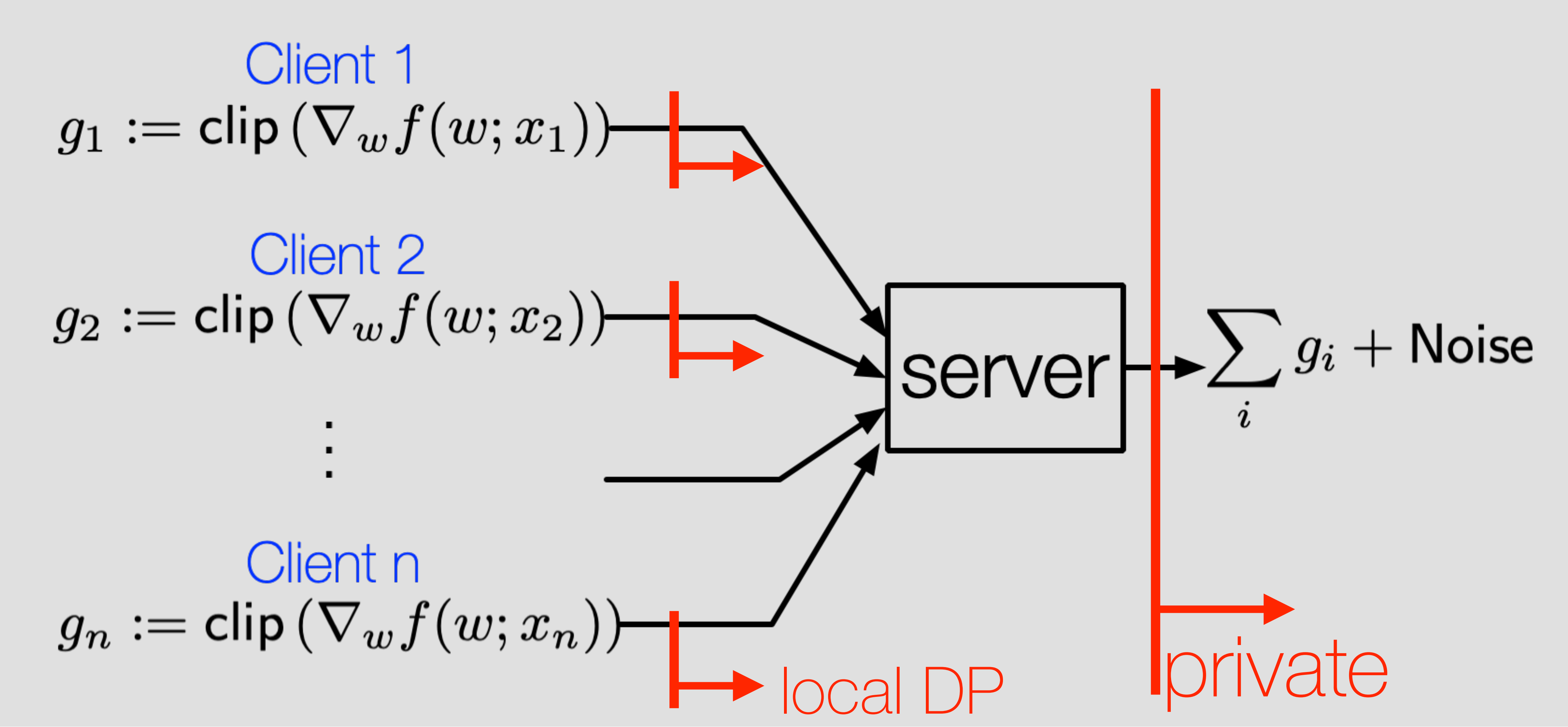
# Federated Learning with Differential Privacy

- FL with **central** differential privacy (DP)
  - Server collects local model updates and perturbs them
  - Sounds great, *unless the server is not trusted…*



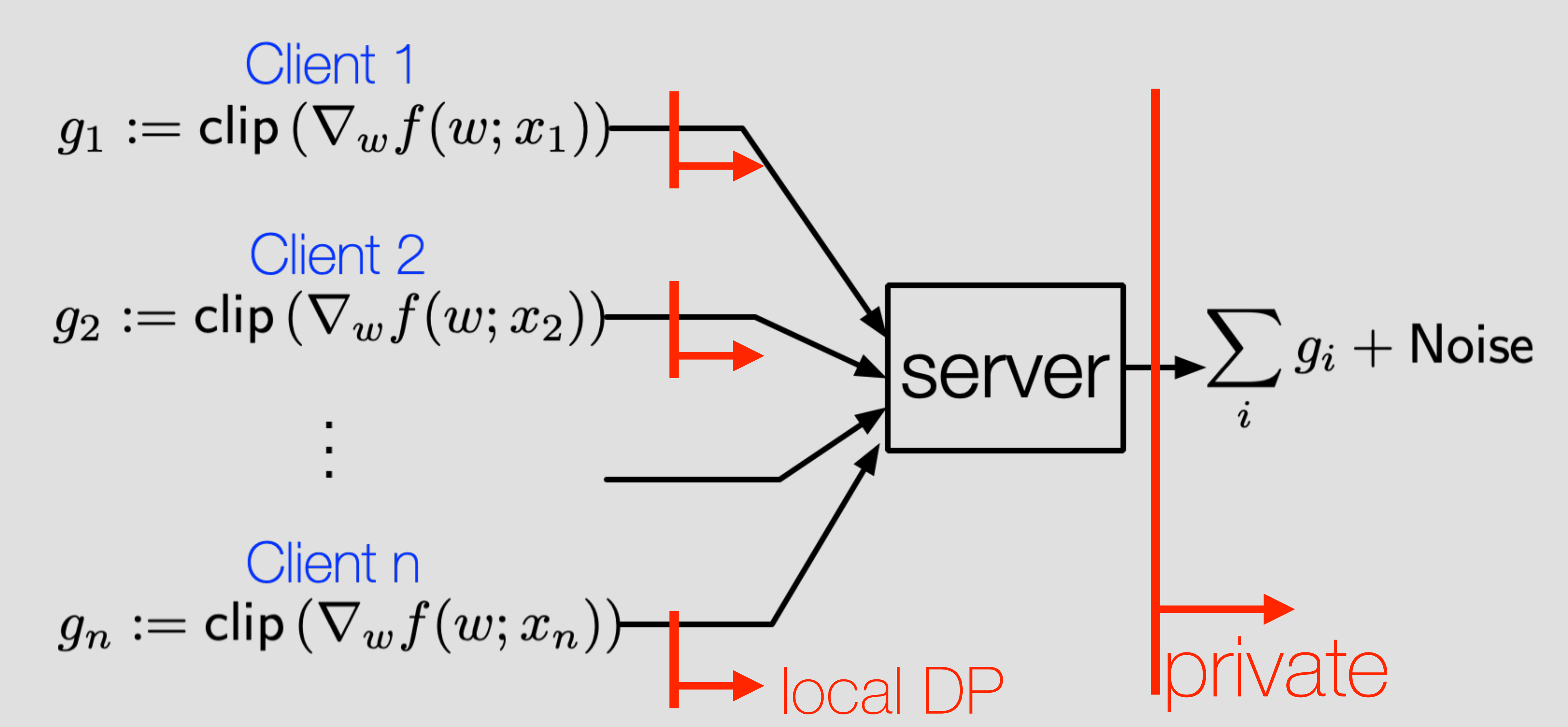
# Federated Learning with Differential Privacy

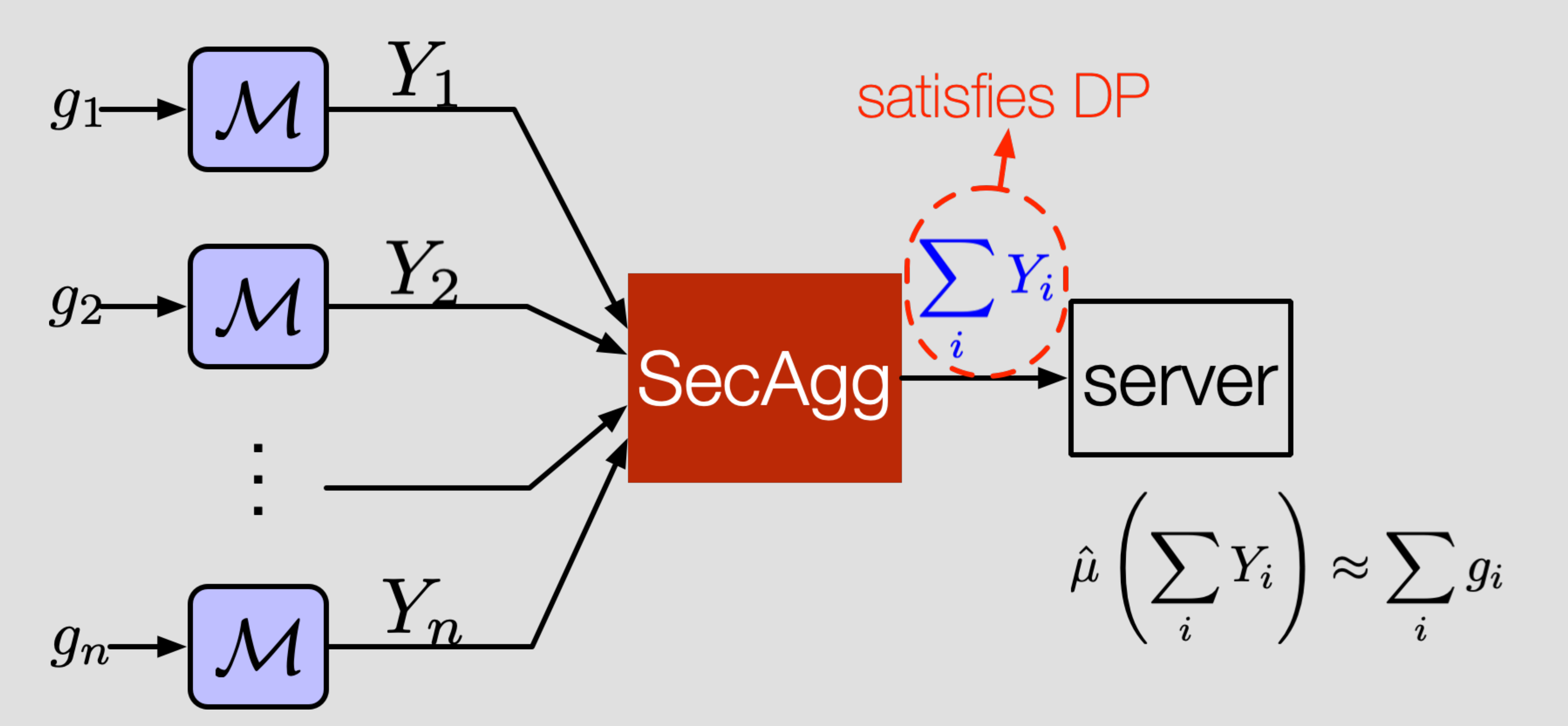
- FL with **central** differential privacy (DP)
  - Server collects local model updates and perturbs them
  - Sounds great, *unless the server is not trusted…*
- FL with **local** DP
  - Strongest privacy guarantees
  - Poor utility compared to central DP



# Federated Learning with Differential Privacy

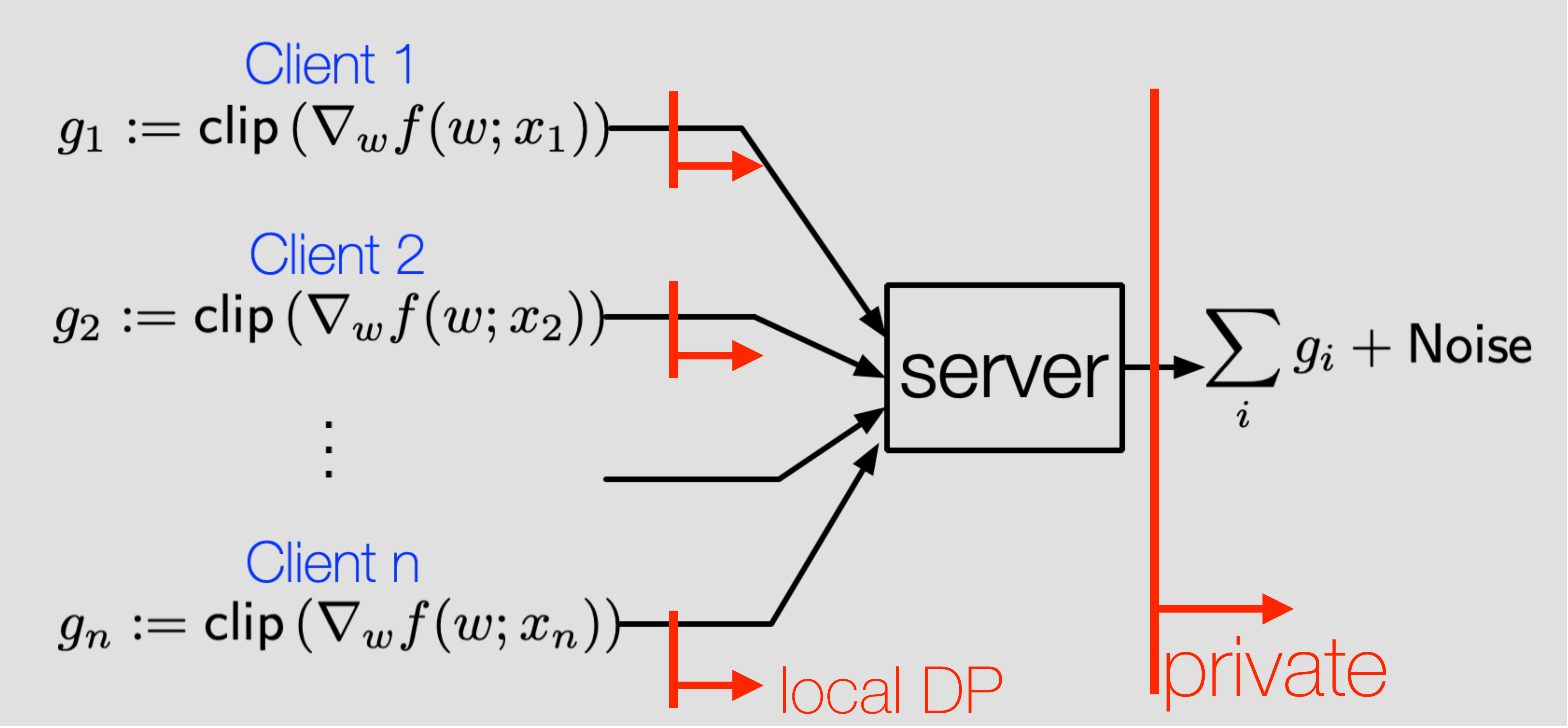
- FL with **central** differential privacy (DP)
  - Server collects local model updates and perturbs them
  - Sounds great, *unless the server is not trusted…*
- FL with **local** DP
  - Strongest privacy guarantees
  - Poor utility compared to central DP
- FL with **distributed** DP

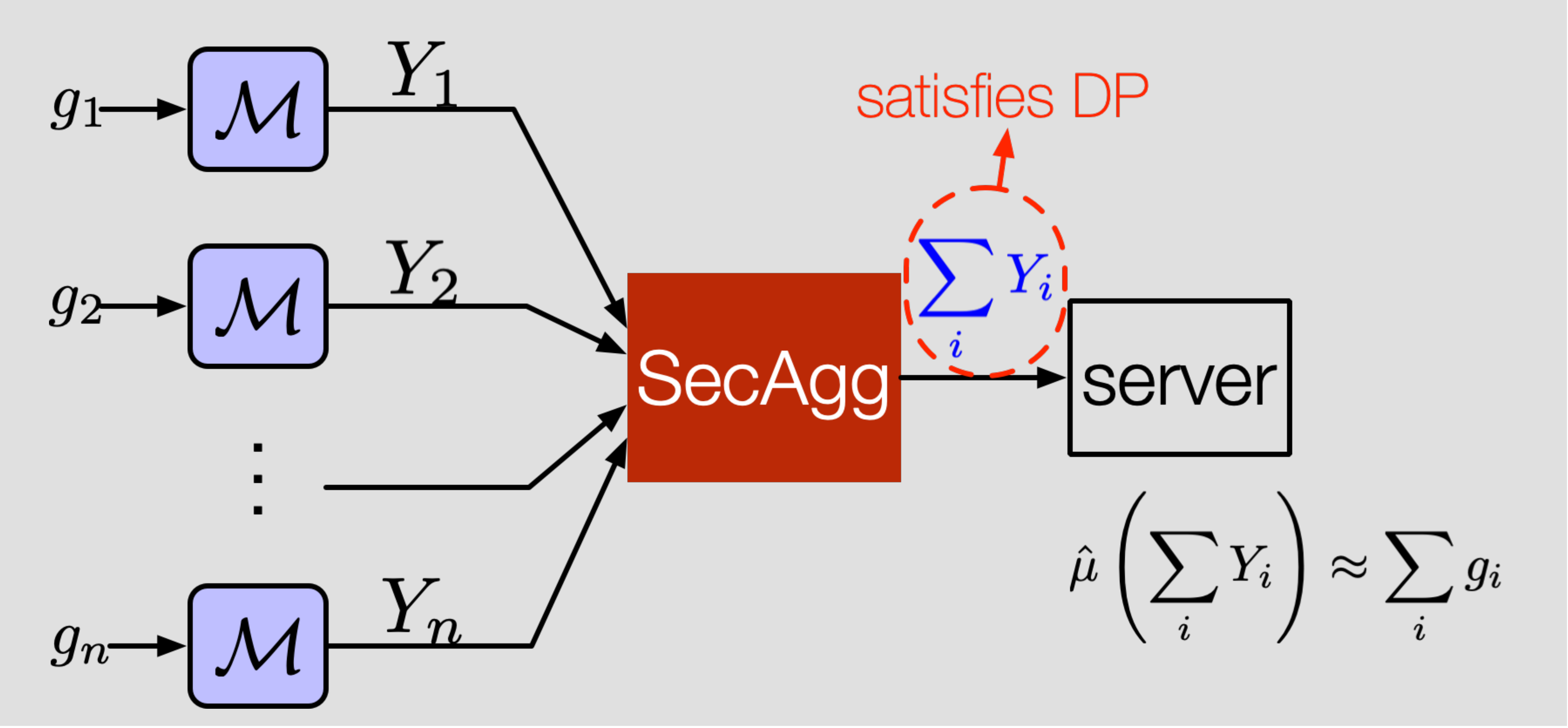


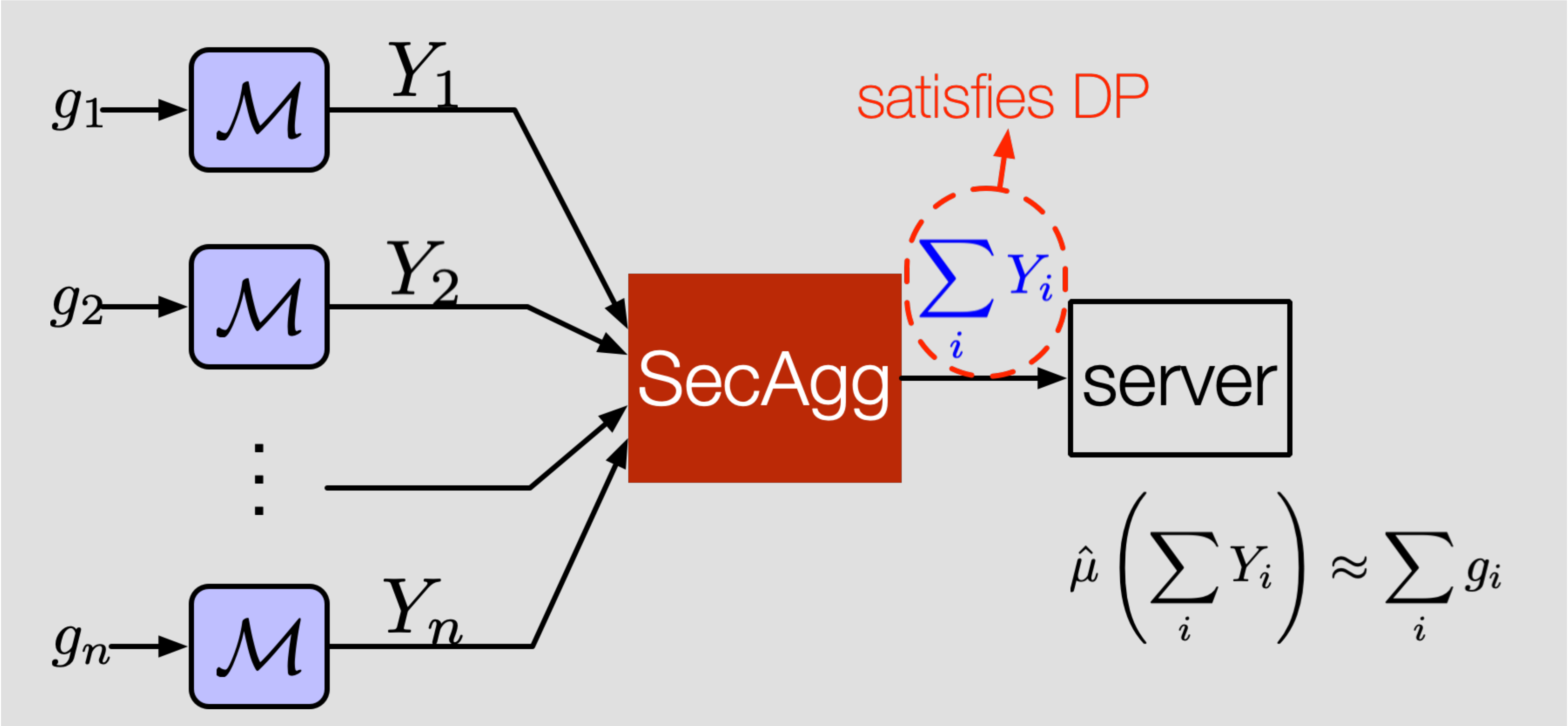
# Federated Learning with Differential Privacy

- FL with **central** differential privacy (DP)
  - Server collects local model updates and perturbs them
  - Sounds great, *unless the server is not trusted…*
- FL with **local** DP
  - Strongest privacy guarantees
  - Poor utility compared to central DP
- FL with **distributed** DP
  - Clients locally perturb their own model updates
  - Server aggregates local updates via cryptographic MPC such as secure aggregation (SecAgg)
  - Privacy does not rely on the trust to the server

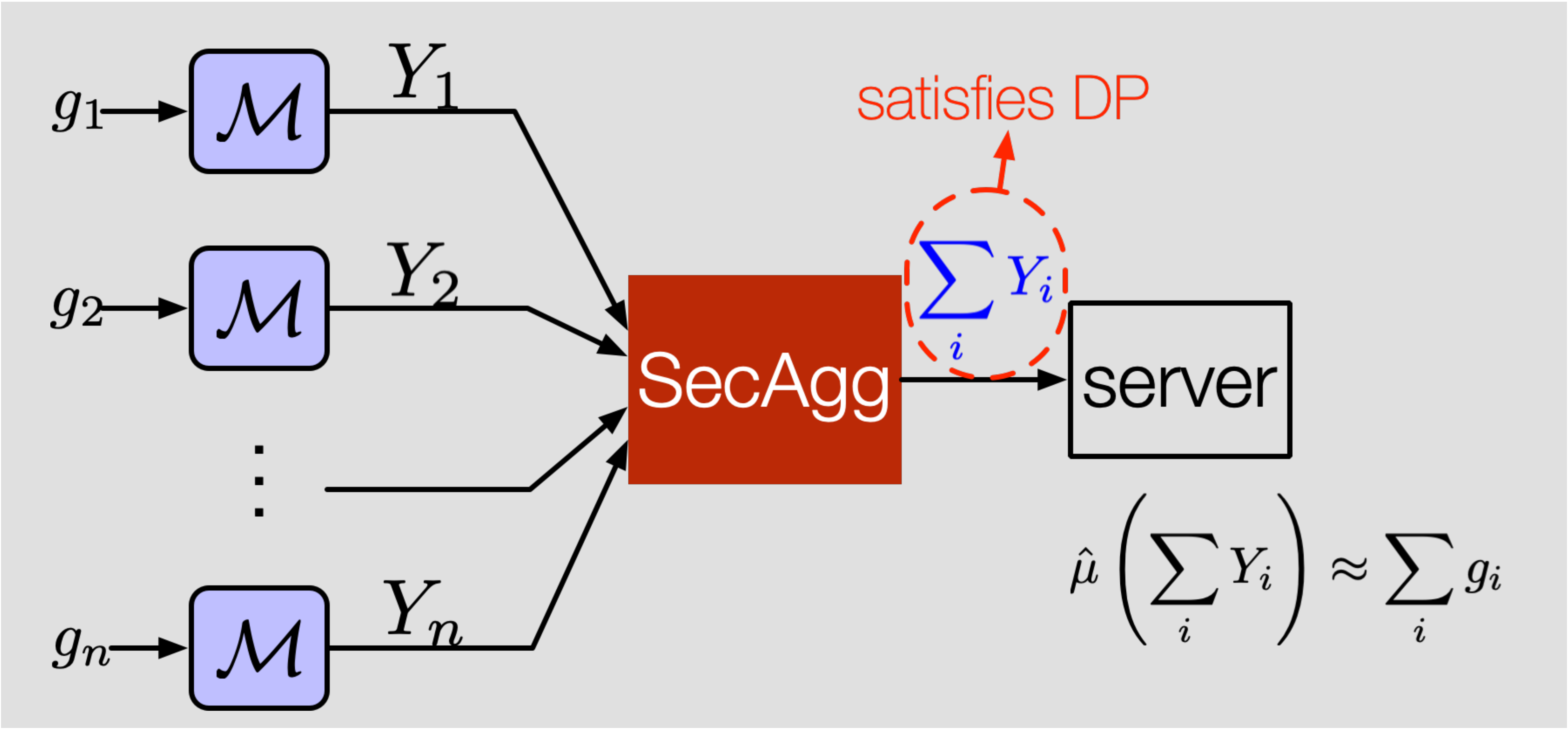


# Constraints on private FL with SecAgg

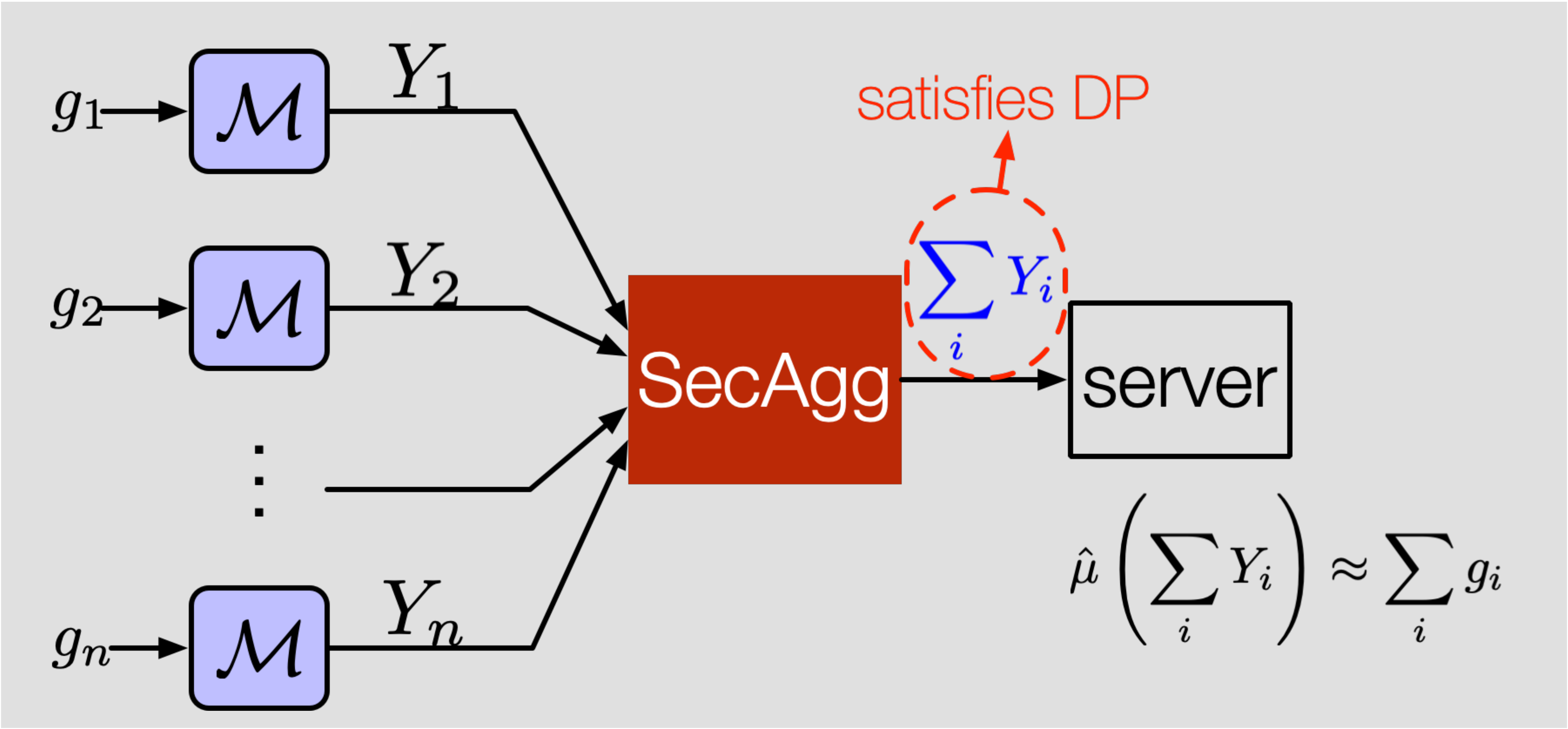


# Constraints on private FL with SecAgg



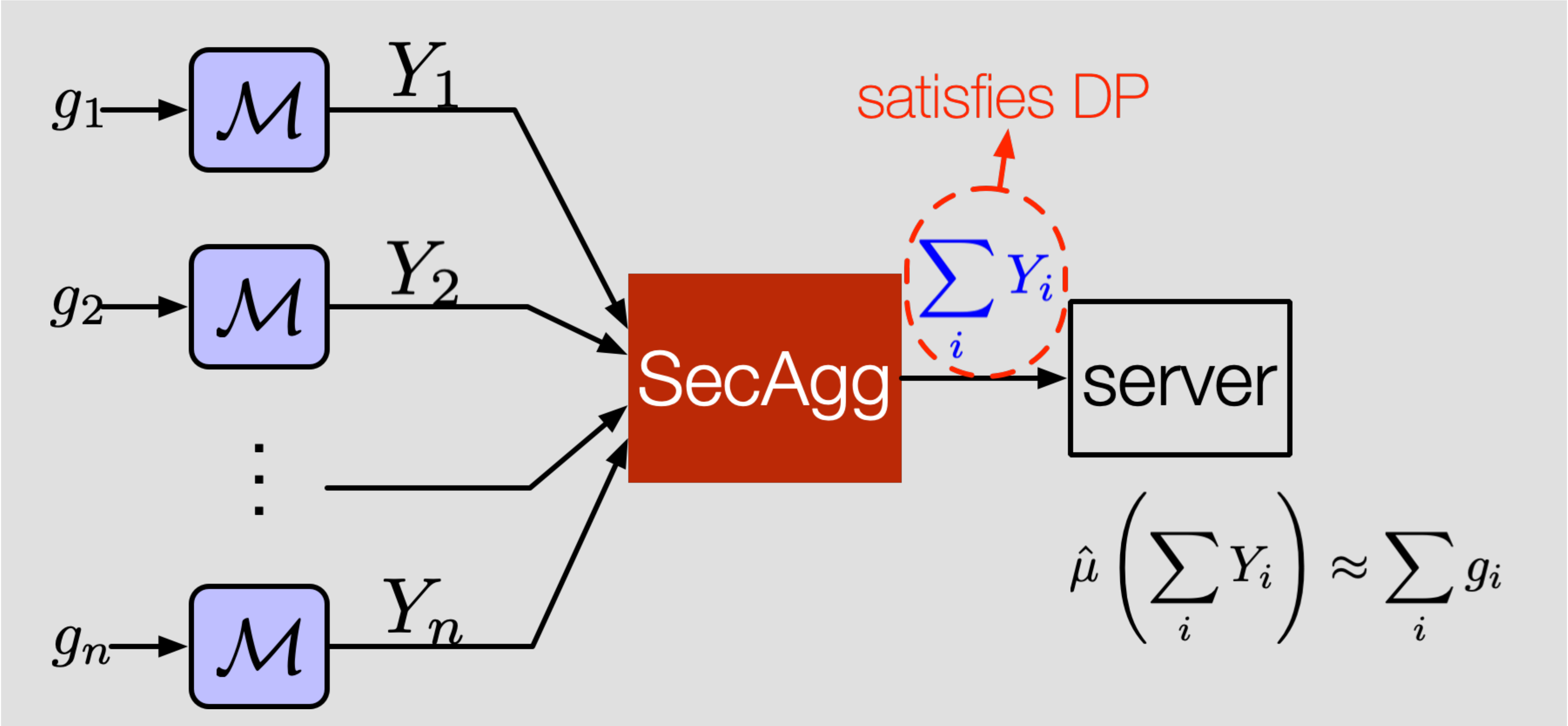

- The local randomizer needs to be **linear** over a **finite field**

# Constraints on private FL with SecAgg




- The local randomizer needs to be **linear** over a **finite field**
- An **unbiased** estimator is preferred

# Constraints on private FL with SecAgg

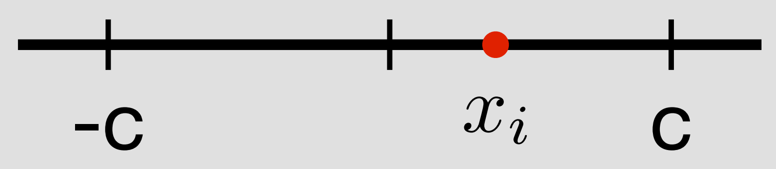



- The local randomizer needs to be **linear** over a **finite field**
- An **unbiased** estimator is preferred
- Less communication in high privacy regime (with small $\varepsilon$)

# Previous solutions of distributed DP

- Previous solutions with SecAgg and DP

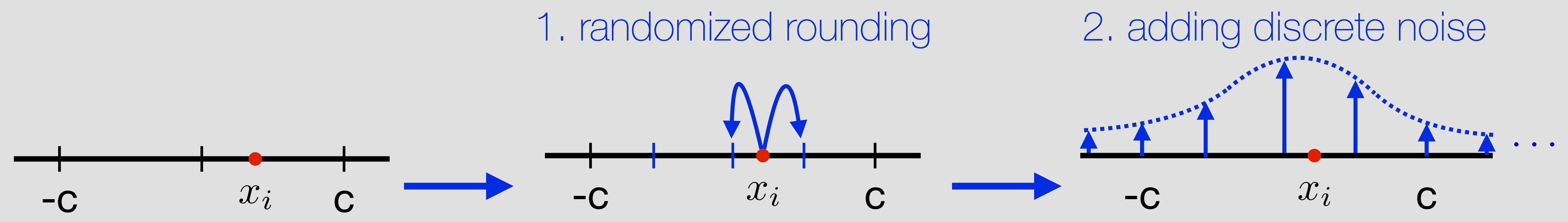
- [ ] Over a finite field
- [ ] Linear
- [ ] Unbiased
- [ ] Communication $\searrow \varepsilon$



# Previous solutions of distributed DP

- Previous solutions with SecAgg and DP
  - (stochastically) round local updates
  - perturb with discrete local noise

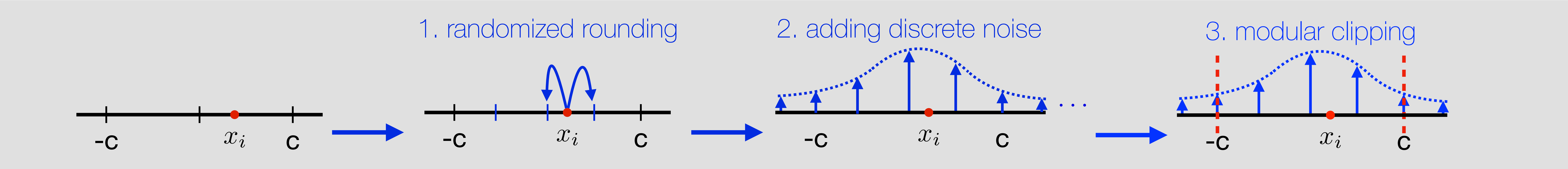
- ☐ Over a finite field
- ☐ Linear
- ☐ Unbiased
- ☐ Communication $\searrow \varepsilon$



# Previous solutions of distributed DP

- Previous solutions with SecAgg and DP
  - (stochastically) round local updates
  - perturb with discrete local noise
  - map to a finite field by modular clipping
  - examples: binomial [1], distributed discrete Gaussian [2], Skellam[3]

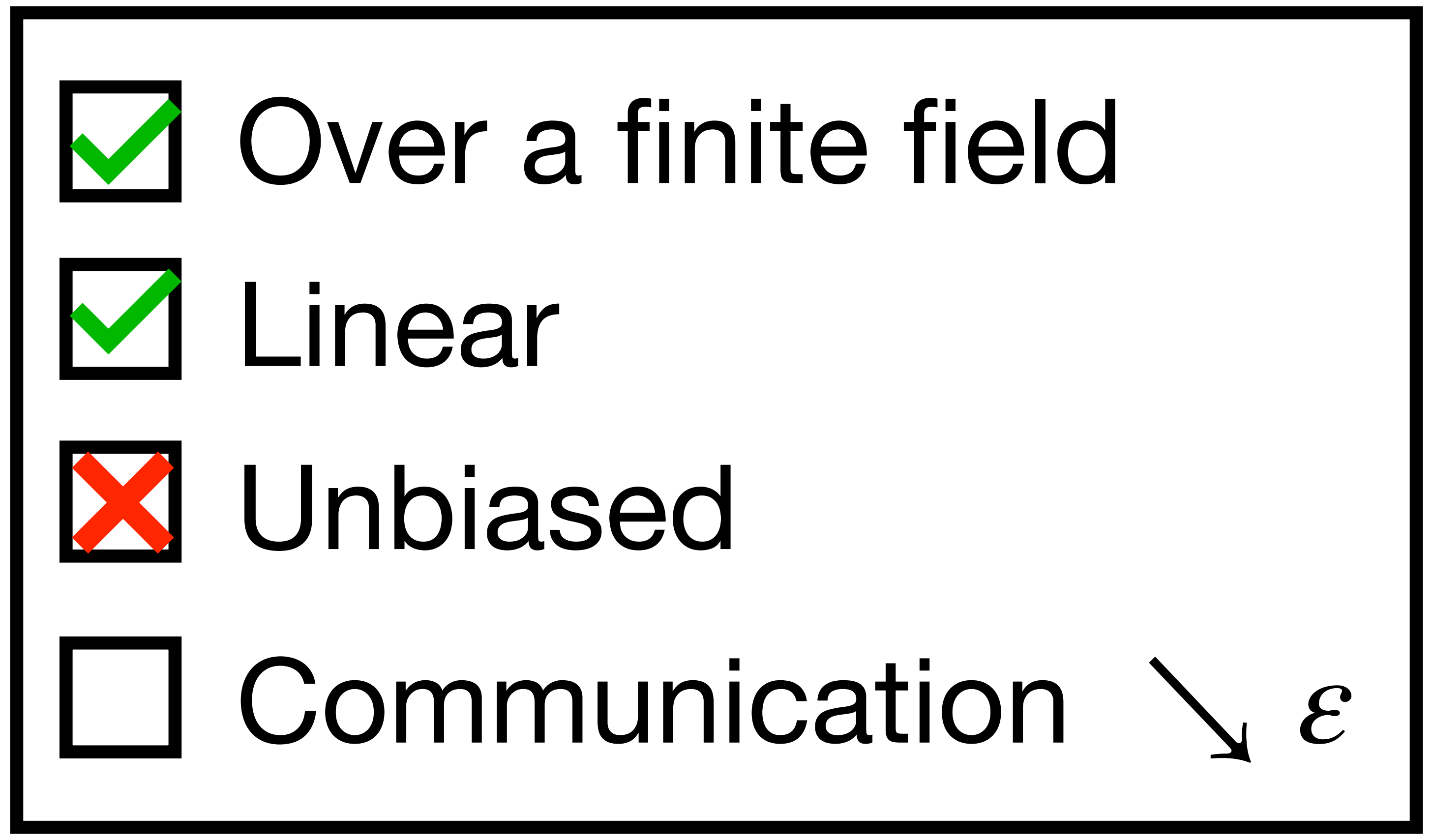
- [x] Over a finite field
- [x] Linear
- [ ] Unbiased
- [ ] Communication ↘ $\varepsilon$



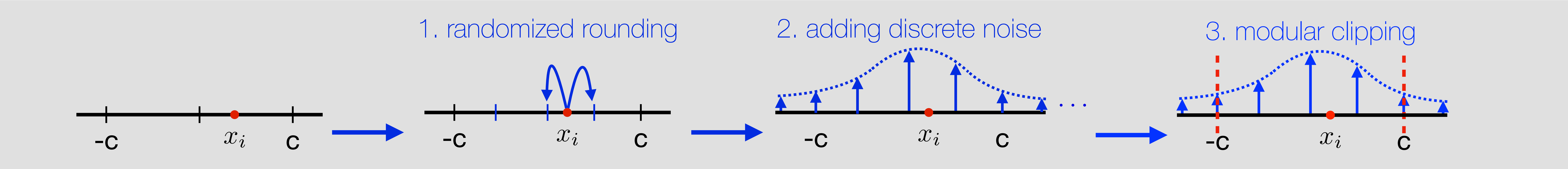

[1] Suresh Ananda Theertha, et al. "cpSGD: Communication-efficient and differentially-private distributed SGD." NeurIPS 2018.
[2] Peter Kairouz, et al. "The distributed discrete gaussian mechanism for federated learning with secure aggregation." ICML 2021.
[3] Naman Argawal, et al. "The skellam mechanism for differentially private federated learning." NeurIPS 2021.

# Previous solutions of distributed DP

- Previous solutions with SecAgg and DP
  - (stochastically) round local updates
  - perturb with discrete local noise
  - map to a finite field by modular clipping
  - examples: binomial [1], distributed discrete Gaussian [2], Skellam[3]



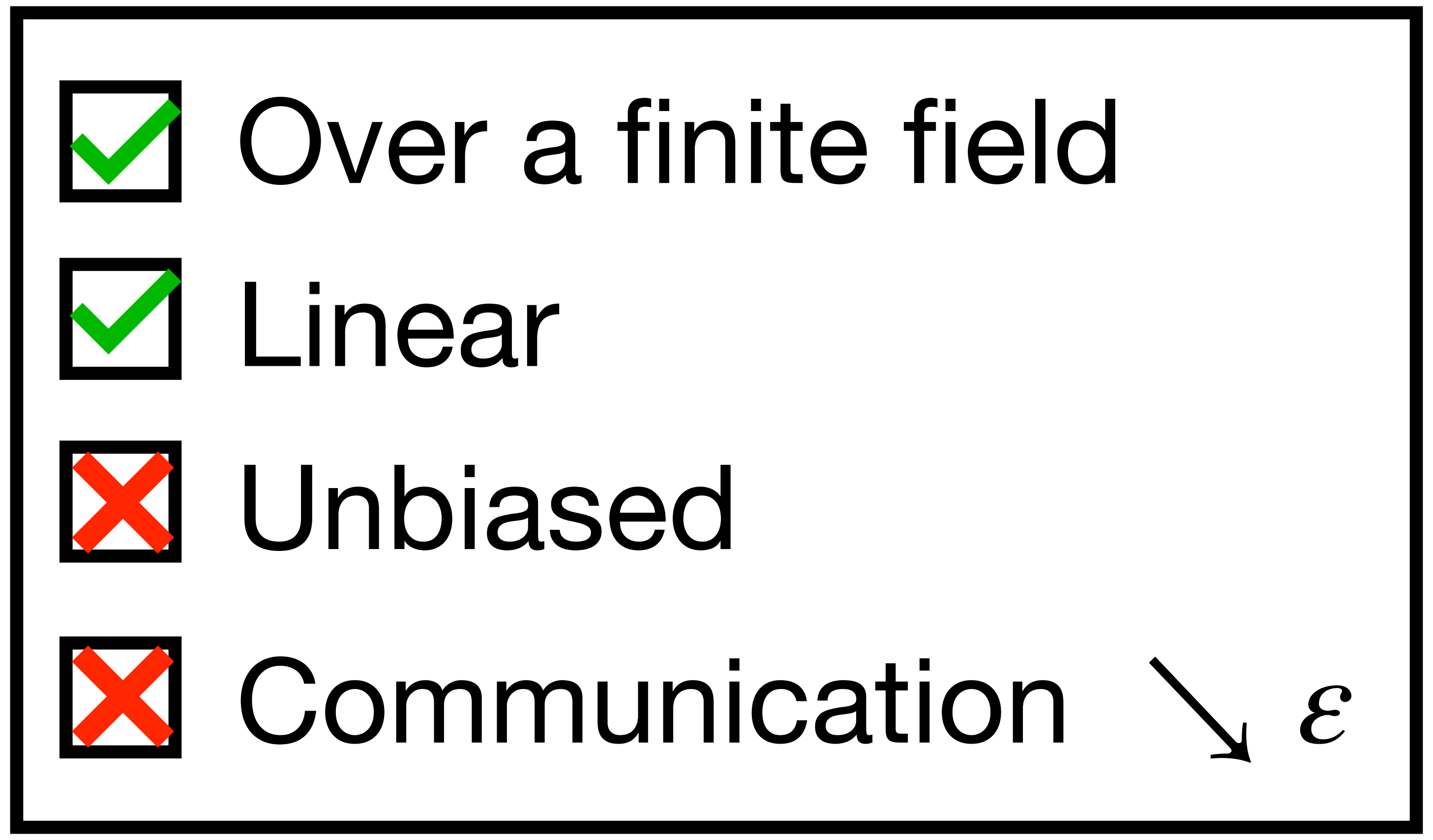

- Potential issues
  - the modular clipping introduces **bias**



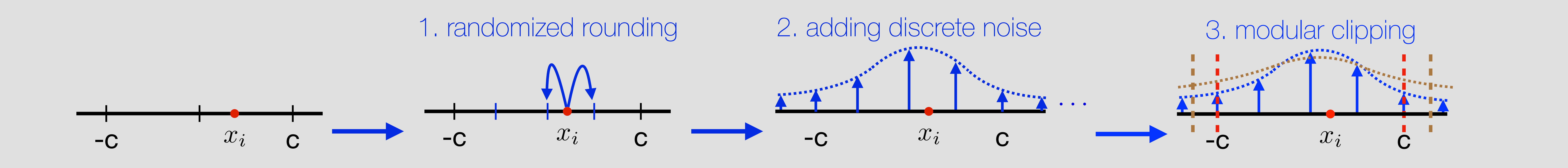

[1] Suresh Ananda Theertha, et al. "cpSGD: Communication-efficient and differentially-private distributed SGD." NeurIPS 2018.
[2] Peter Kairouz, et al. "The distributed discrete gaussian mechanism for federated learning with secure aggregation." ICML 2021.
[3] Naman Argawal, et al. "The skellam mechanism for differentially private federated learning." NeurIPS 2021.

# Previous solutions of distributed DP

- Previous solutions with SecAgg and DP
  - (stochastically) round local updates
  - perturb with discrete local noise
  - map to a finite field by modular clipping
  - examples: binomial [1], distributed discrete Gaussian [2], Skellam[3]




- Potential issues
  - the modular clipping introduces **bias**
  - the **higher privacy**, the larger variance of the noise, resulting in **higher communication** cost
  - Communication cost $\rightarrow \infty$ as $\varepsilon \rightarrow 0$



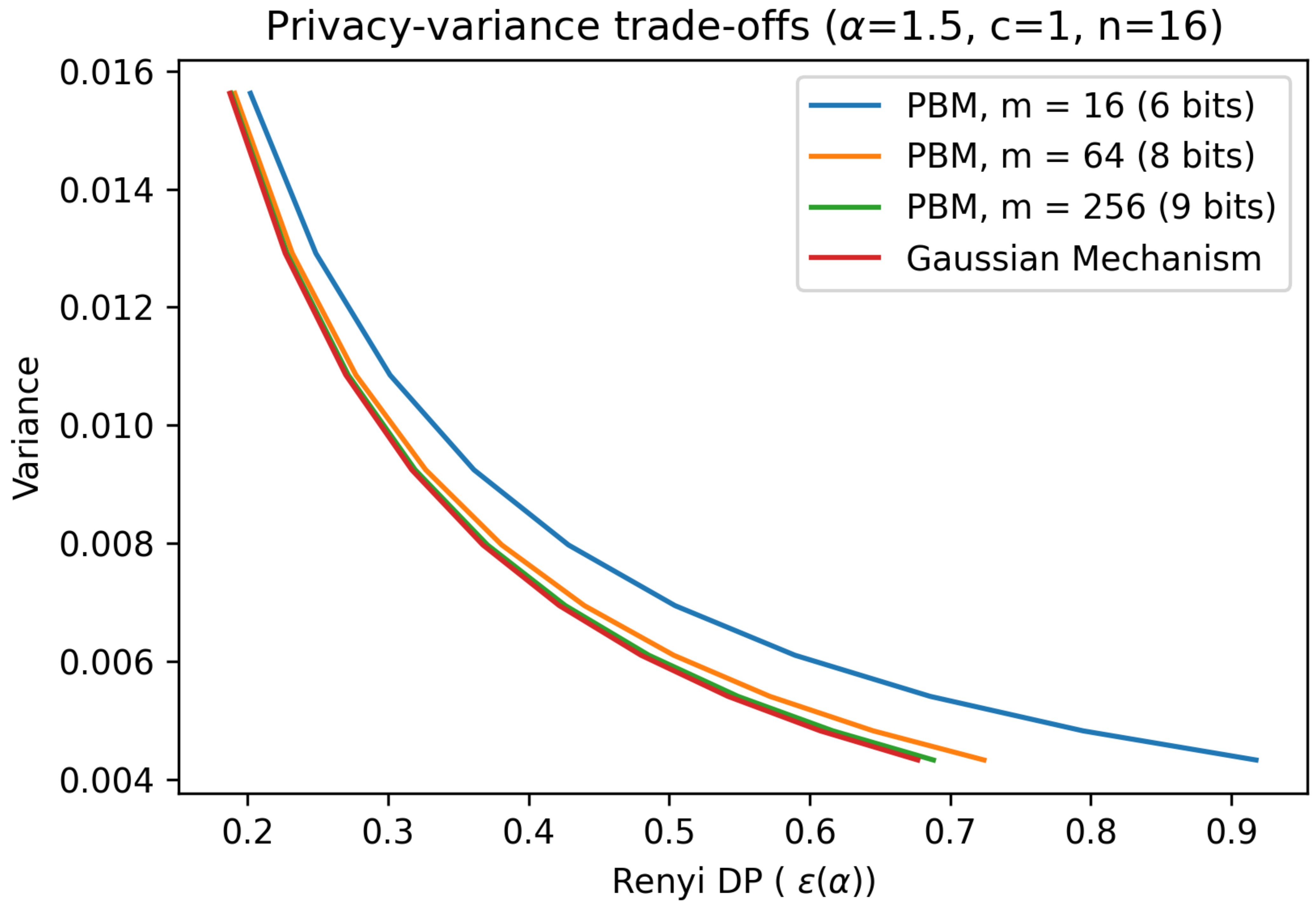

[1] Suresh Ananda Theertha, et al. "cpSGD: Communication-efficient and differentially-private distributed SGD." NeurIPS 2018.
[2] Peter Kairouz, et al. "The distributed discrete gaussian mechanism for federated learning with secure aggregation." ICML 2021.
[3] Naman Argawal, et al. "The skellam mechanism for differentially private federated learning." NeurIPS 2021.

# Our contributions

- We propose the Poisson-binomial mechanism (PBM), which
  - yields an **unbiased** estimate of the mean
  - has communication **decreasing** with $\varepsilon$

# Our contributions

- We propose the Poisson-binomial mechanism (PBM), which
  - yields an **unbiased** estimate of the mean
  - has communication **decreasing** with $\varepsilon$
  - achieves order-optimal privacy-accuracy trade-off
  - allows for numerically computing the **exact** privacy loss
  - converges to the performance of centralized Gaussian



# The scalar Poisson binomial mechanism (sPBM)

**Algorithm (scalar PBM)**

**Parameters:** $m \in \mathbb{N}$, $\theta \leq 0.1$

For client $i$:

1. Re-scale $x_i$ to $p_i \in [\frac{1}{2} - \theta, \frac{1}{2} + \theta]$
2. Draw $Y_i \sim \text{Binom}(m, p_i)$

- 1-d mean estimation problem
  - Client i holds $x_i \in [-c, c]$
  - Server estimates $\mu = \frac{1}{n} \sum_i x_i$

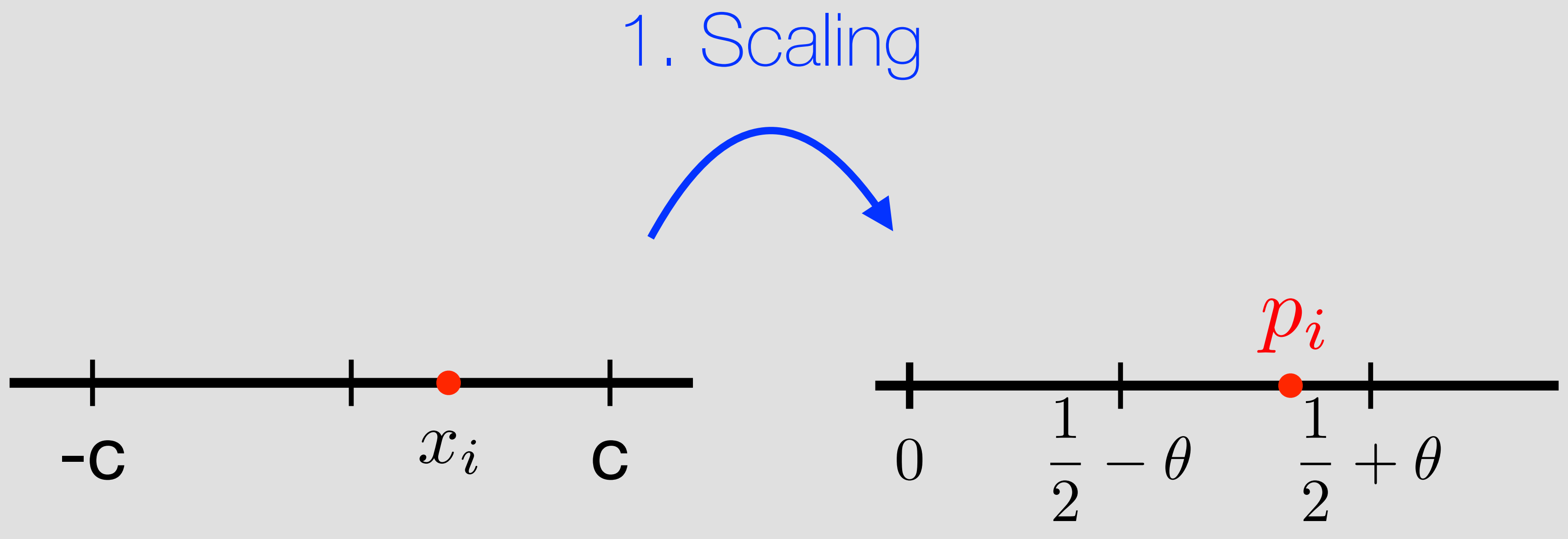
# The scalar Poisson binomial mechanism (sPBM)

**Algorithm (scalar PBM)**

**Parameters:** $m \in \mathbb{N}$, $\theta \leq 0.1$

For client $i$:

1. Re-scale $x_i$ to $p_i \in [\frac{1}{2} - \theta, \frac{1}{2} + \theta]$
2. Draw $Y_i \sim \text{Binom}(m, p_i)$




- 1-d mean estimation problem
  - Client i holds $x_i \in [-c, c]$
  - Server estimates $\mu = \frac{1}{n} \sum_i x_i$

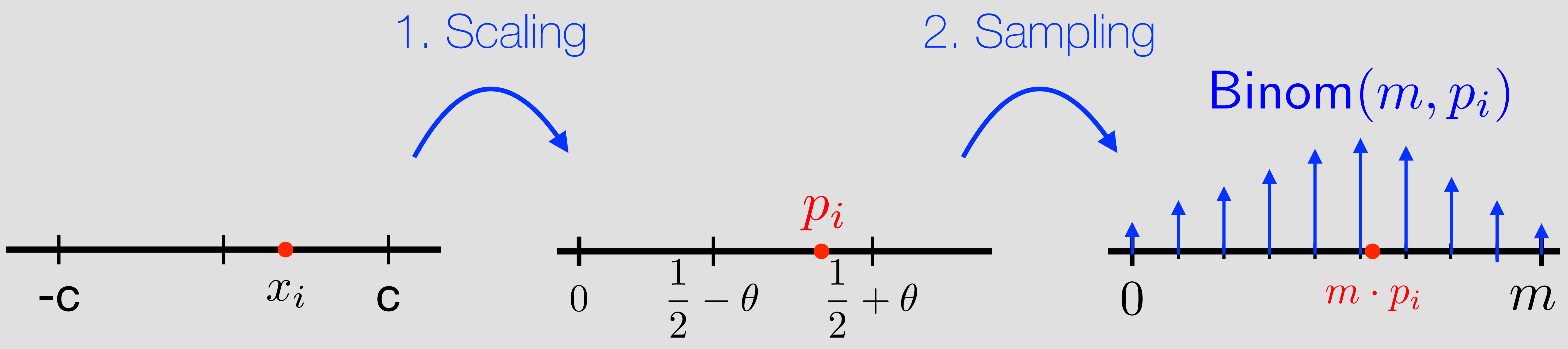
# The scalar Poisson binomial mechanism (sPBM)

**Algorithm (scalar PBM)**

**Parameters:** $m \in \mathbb{N}$, $\theta \leq 0.1$

For client $i$:

1. Re-scale $x_i$ to $p_i \in [\frac{1}{2} - \theta, \frac{1}{2} + \theta]$
2. Draw $Y_i \sim \text{Binom}(m, p_i)$




- 1-d mean estimation problem
  - Client i holds $x_i \in [-c, c]$
  - Server estimates $\mu = \frac{1}{n} \sum_i x_i$

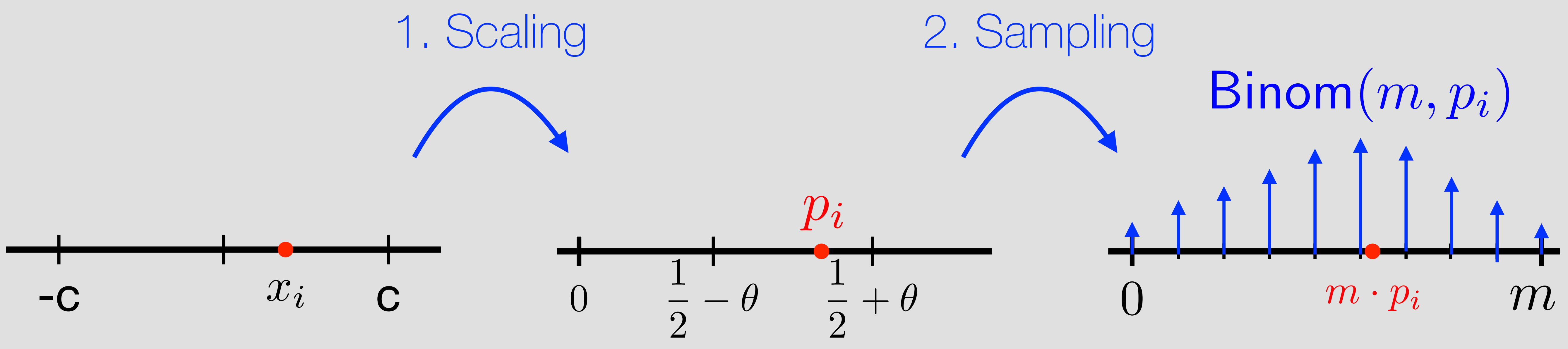
# The scalar Poisson binomial mechanism (sPBM)

**Algorithm (scalar PBM)**

**Parameters:** $m \in \mathbb{N}$, $\theta \leq 0.1$

For client $i$:

1. Re-scale $x_i$ to $p_i \in [\frac{1}{2} - \theta, \frac{1}{2} + \theta]$
2. Draw $Y_i \sim \text{Binom}(m, p_i)$




- 1-d mean estimation problem
  - Client i holds $x_i \in [-c, c]$
  - Server estimates $\mu = \frac{1}{n} \sum_i x_i$
- $\hat{x}_i := c(Y_i - m/2)/m\theta$ yields an unbiased estimator on $x_i$

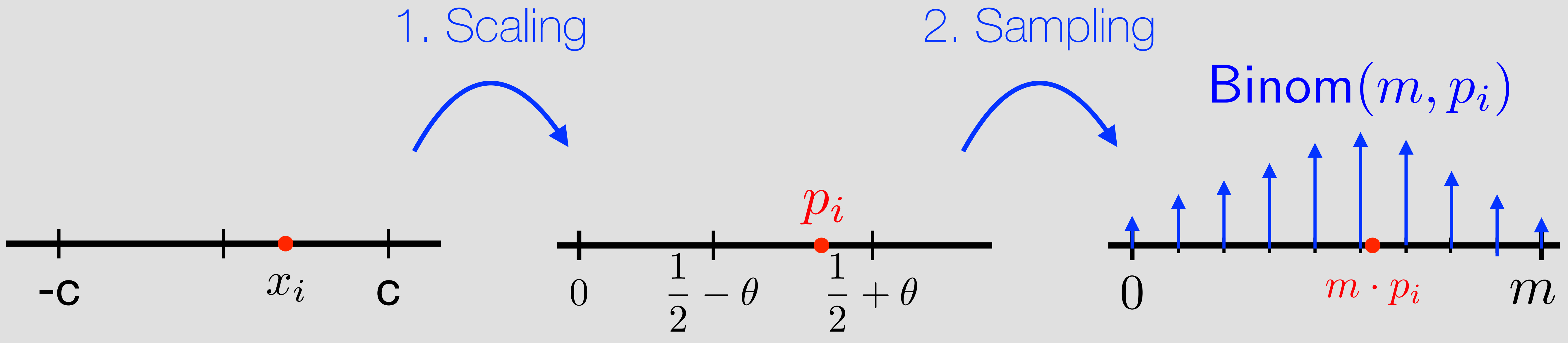
# The scalar Poisson binomial mechanism (sPBM)

**Algorithm (scalar PBM)**

**Parameters:** $m \in \mathbb{N}$, $\theta \leq 0.1$

For client $i$:

1. Re-scale $x_i$ to $p_i \in [\frac{1}{2} - \theta, \frac{1}{2} + \theta]$
2. Draw $Y_i \sim$ Binom$(m, p_i)$




- 1-d mean estimation problem
  - Client i holds $x_i \in [-c, c]$
  - Server estimates $\mu = \frac{1}{n} \sum_i x_i$
- $\hat{x}_i := c(Y_i - m/2)/m\theta$ yields an unbiased estimator on $x_i$
- $Y_i$ at most $m$, so $m$ dictates the communication cost

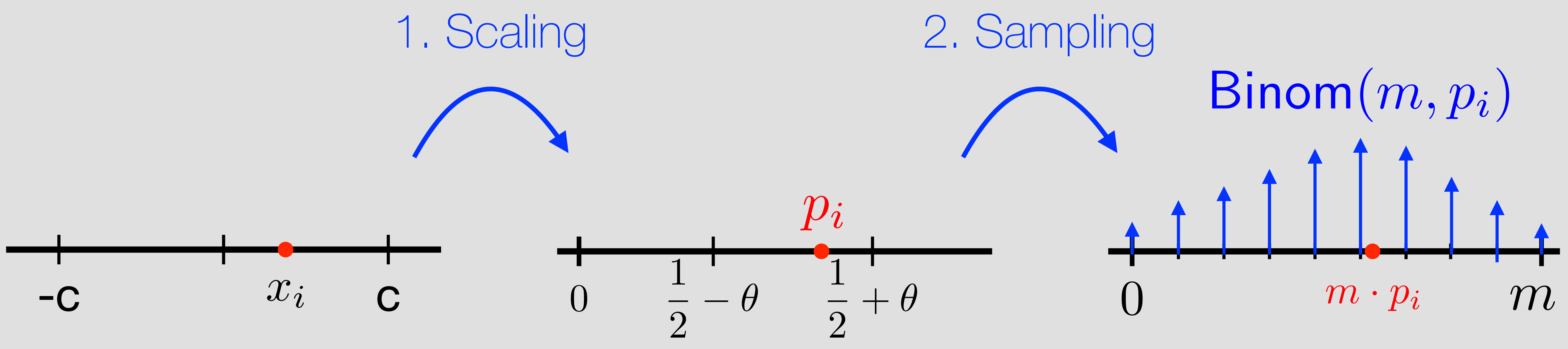
# The scalar Poisson binomial mechanism (sPBM)

**Algorithm (scalar PBM)**

**Parameters:** $m \in \mathbb{N}$, $\theta \leq 0.1$

For client $i$:

1. Re-scale $x_i$ to $p_i \in [\frac{1}{2} - \theta, \frac{1}{2} + \theta]$
2. Draw $Y_i \sim \text{Binom}(m, p_i)$




- 1-d mean estimation problem
  - Client i holds $x_i \in [-c, c]$
  - Server estimates $\mu = \frac{1}{n} \sum_i x_i$
- $\hat{x}_i := c(Y_i - m/2)/m\theta$ yields an unbiased estimator on $x_i$
- $Y_i$ at most $m$, so $m$ dictates the communication cost
- Higher privacy $\rightarrow$ decreasing $m$ and $\theta$

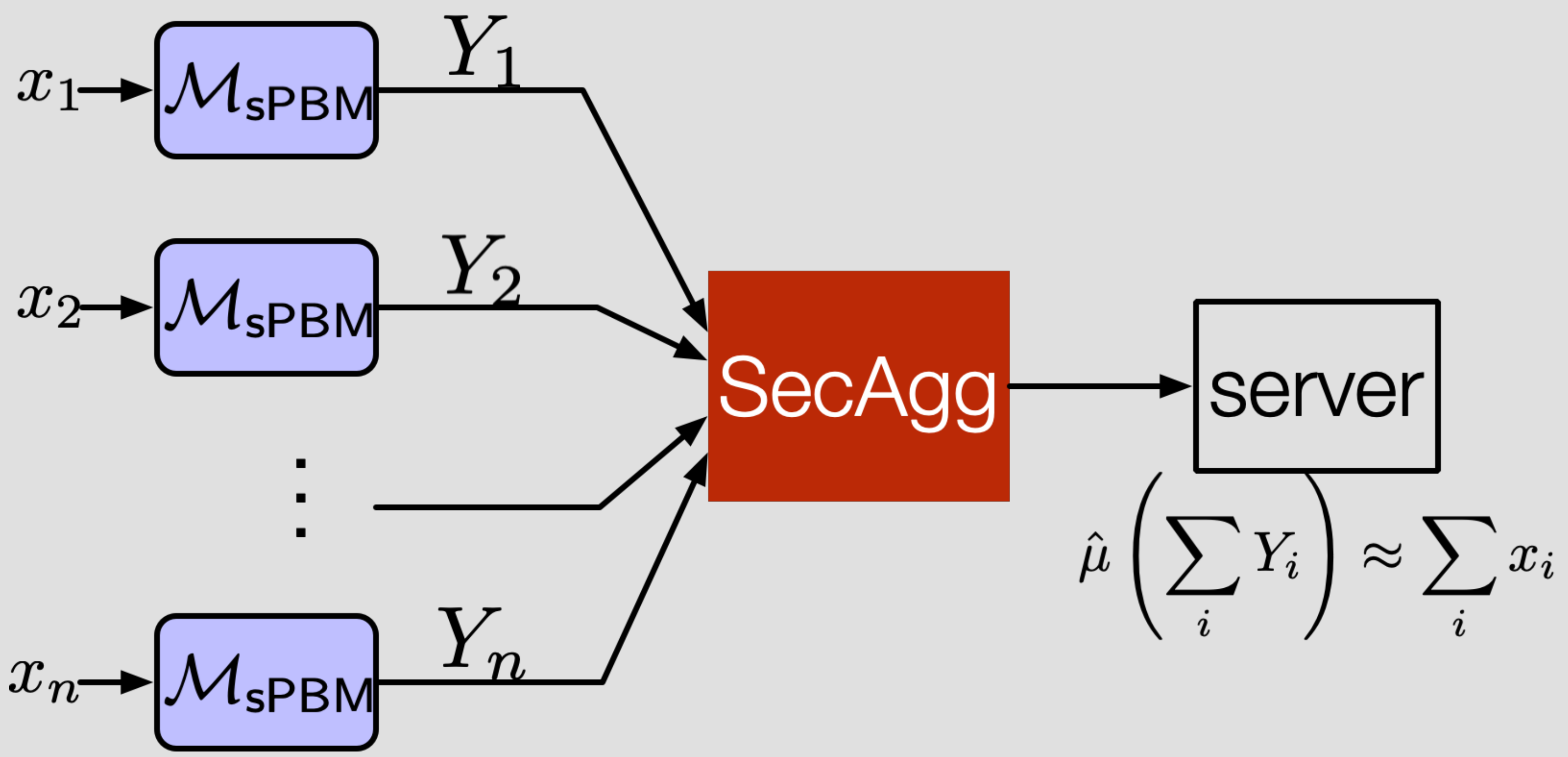
# Mean Estimation with sPBM and SecAgg

**Algorithm (sPBM)**

**Parameters:** $m \in \mathbb{N}$, $\theta \leq 0.1$

For client $i$:

1. Re-scale $x_i$ to $p_i \in [\frac{1}{2} - \theta, \frac{1}{2} + \theta]$
2. Draw $Y_i \sim$ Binom$(m, p_i)$




- 1-d mean estimation problem
  - Client i holds $x_i \in [-c, c]$
  - Server estimates $\mu = \frac{1}{n}\sum_i x_i$

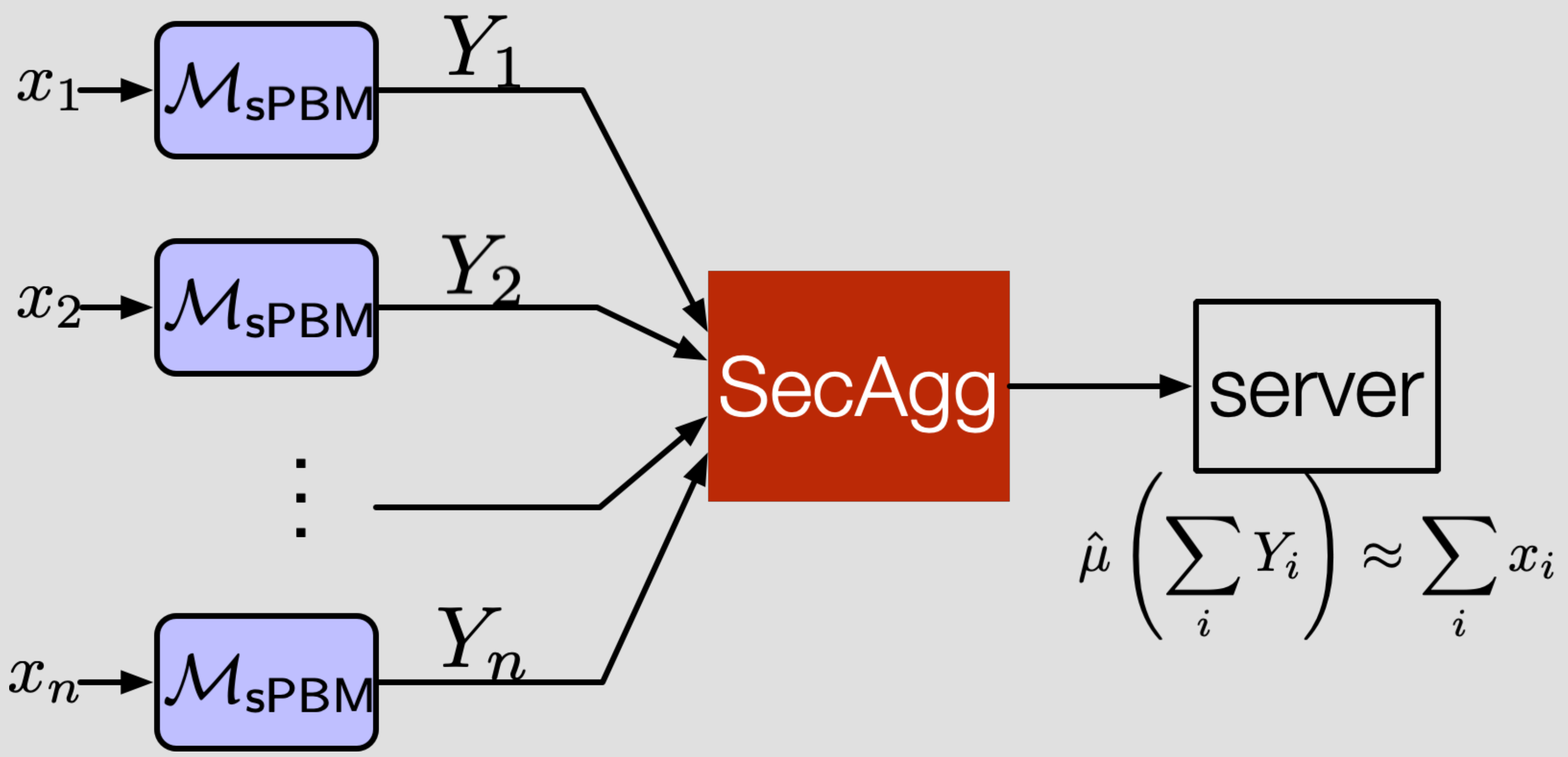
# Mean Estimation with sPBM and SecAgg

**Algorithm (sPBM)**

**Parameters:** $m \in \mathbb{N}$, $\theta \leq 0.1$

For client $i$:

1. Re-scale $x_i$ to $p_i \in [\frac{1}{2} - \theta, \frac{1}{2} + \theta]$
2. Draw $Y_i \sim \text{Binom}(m, p_i)$




- 1-d mean estimation problem
  - Client i holds $x_i \in [-c, c]$
  - Server estimates $\mu = \frac{1}{n}\sum_i x_i$
- Performance guarantees
  - $\hat{\mu} \triangleq \frac{c}{m\theta}\left(\sum_i Y_i - m/2\right)$ is an unbiased estimator with

$$\text{MSE}(\hat{\mu}) \leq \frac{c^2}{4nm\theta^2}$$

  - Per-client communication: log(m+1)+log(n) bits
  - Satisfies $\varepsilon(\alpha)$-DP for $\varepsilon(\alpha) \geq \Omega\left(\alpha m\theta^2/n\right)$

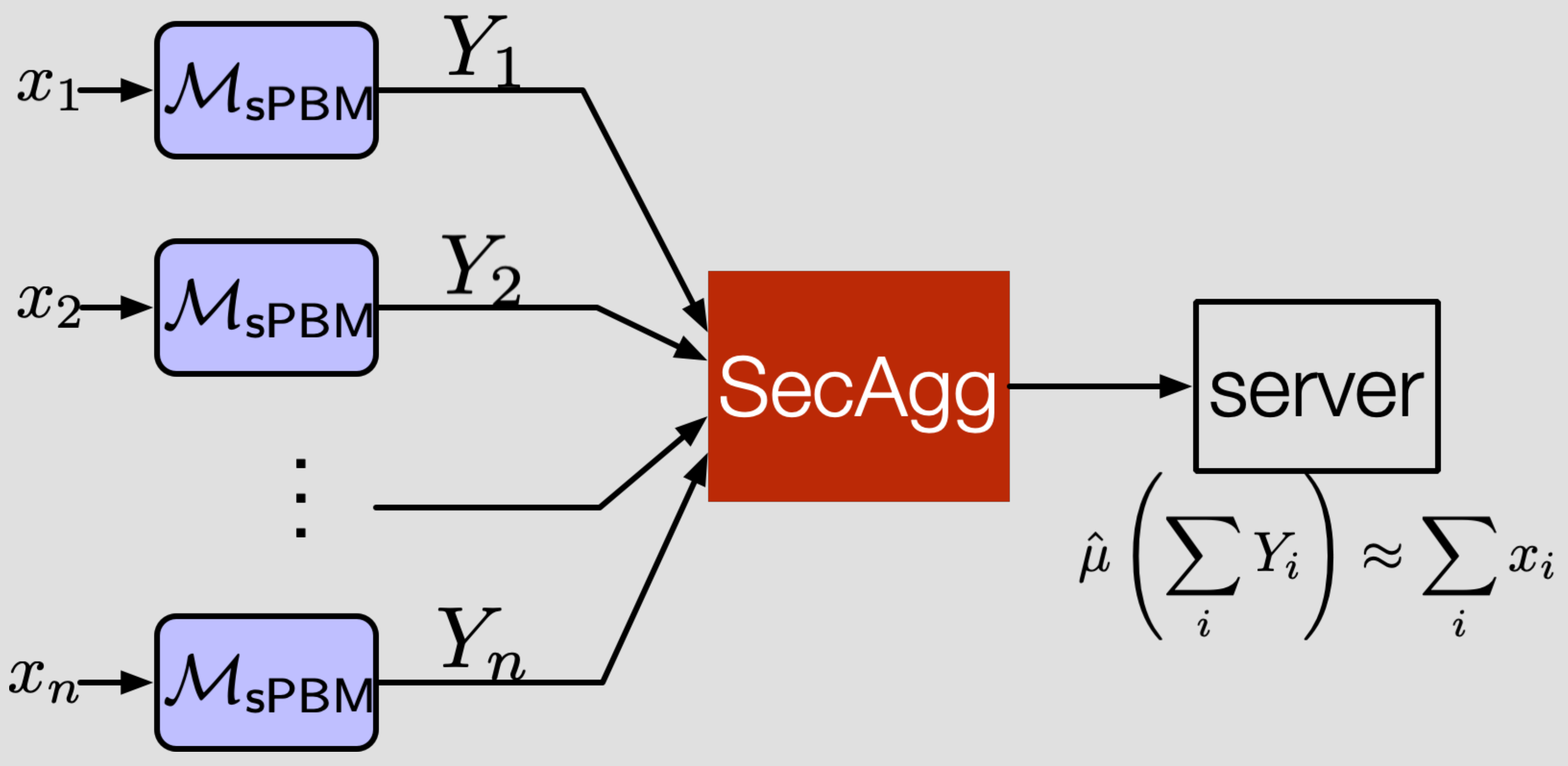
# Mean Estimation with sPBM and SecAgg

**Algorithm (sPBM)**

**Parameters:** $m \in \mathbb{N}$, $\theta \leq 0.1$

For client $i$:

1. Re-scale $x_i$ to $p_i \in [\frac{1}{2} - \theta, \frac{1}{2} + \theta]$
2. Draw $Y_i \sim \text{Binom}(m, p_i)$




- 1-d mean estimation problem
  - Client i holds $x_i \in [-c, c]$
  - Server estimates $\mu = \frac{1}{n}\sum_i x_i$
- Performance guarantees
  - $\hat{\mu} \triangleq \frac{c}{m\theta}\left(\sum_i Y_i - m/2\right)$ is an unbiased estimator with

$$\text{MSE}(\hat{\mu}) \leq \frac{c^2}{4nm\theta^2}$$

  - Per-client communication: log(m+1)+log(n) bits
  - Satisfies $\varepsilon(\alpha)$-DP for $\varepsilon(\alpha) \geq \Omega\left(\alpha m\theta^2/n\right)$

- Key properties of sPBM
  - **linear,** so compatible with SecAgg
  - $(m, \theta)$ jointly characterizes the three-way trade-off of privacy, communication, and accuracy.
  - Communication (dictated by $m$) decreases with $\varepsilon$

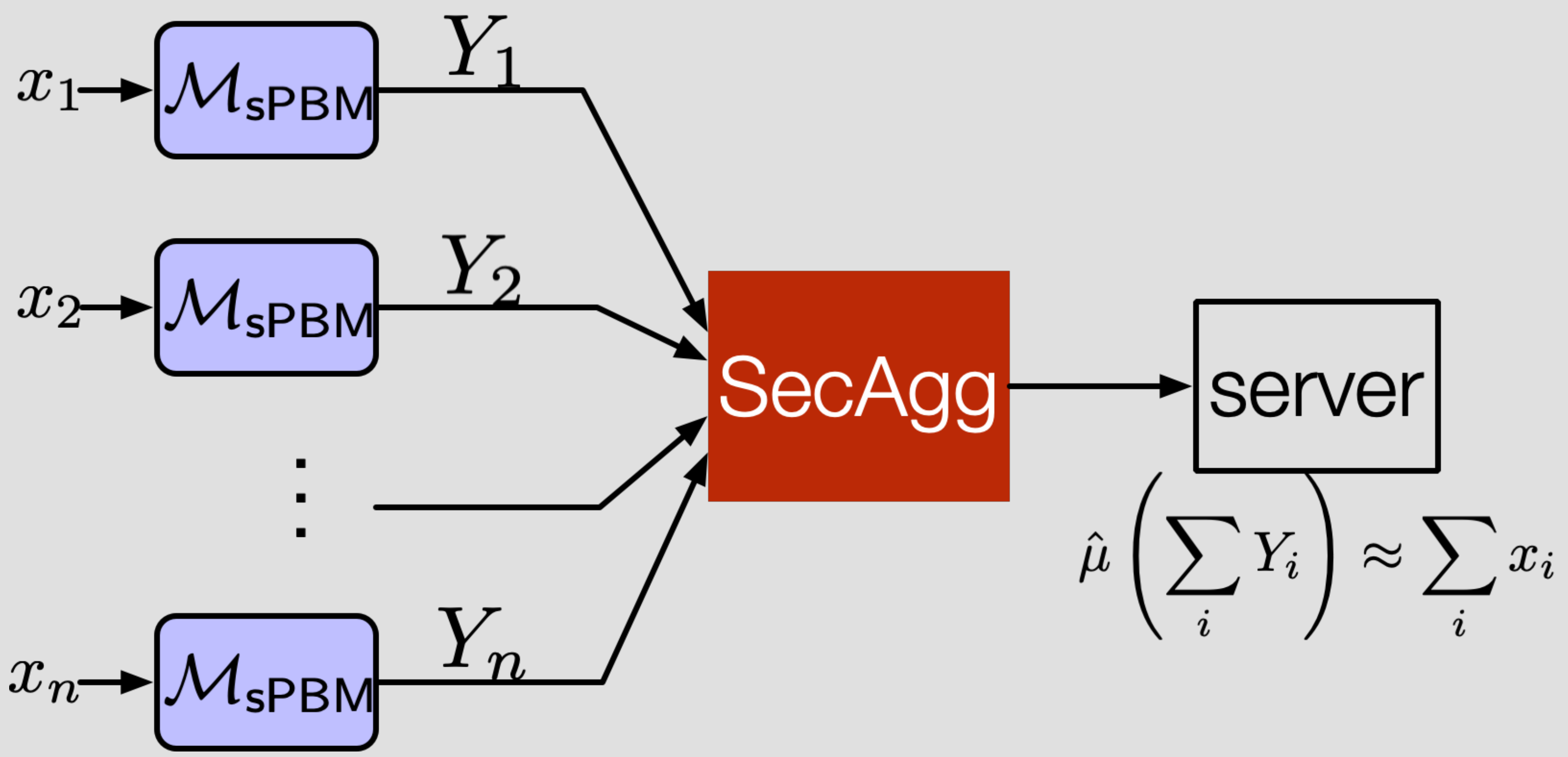
# Privacy of sPBM

**Algorithm (sPBM)**

**Parameters:** $m \in \mathbb{N}$, $\theta \leq 0.1$

For client $i$:

1. Re-scale $x_i$ to $p_i \in [\frac{1}{2} - \theta, \frac{1}{2} + \theta]$
2. Draw $Y_i \sim \text{Binom}(m, p_i)$




- Privacy of sPBM

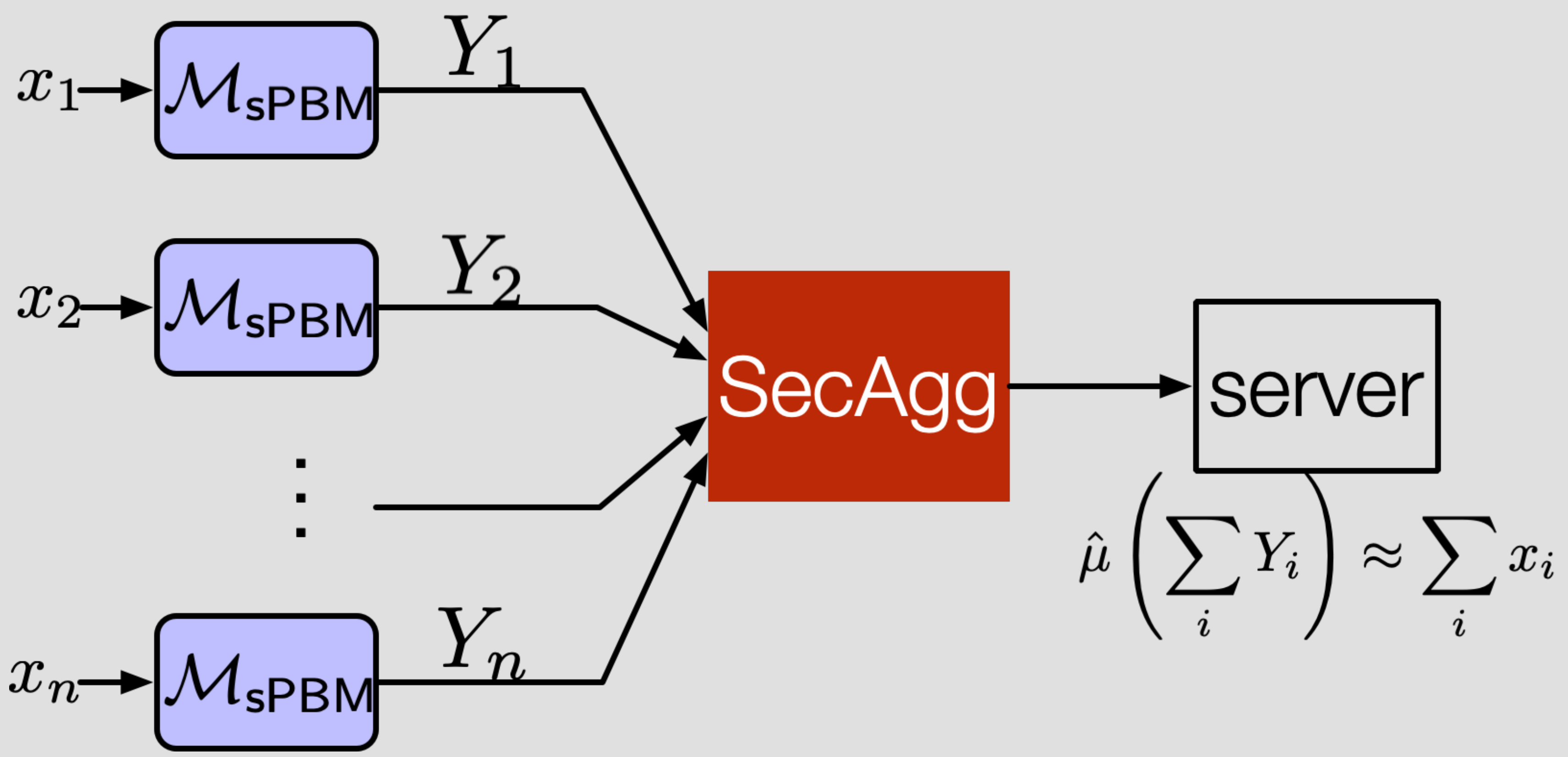
$$\varepsilon(\alpha) = \max_{p_1', p_1, \ldots, p_n} \mathsf{D}_\alpha\left(\sum_{i \in [n]} \text{Binom}(m, p_i) \middle\| \text{Binom}(m, p_1') + \sum_{i \in [2:n]} \text{Binom}(m, p_i)\right)$$

# Privacy of sPBM

**Algorithm (sPBM)**

**Parameters:** $m \in \mathbb{N}$, $\theta \leq 0.1$

For client $i$:
1. Re-scale $x_i$ to $p_i \in [\frac{1}{2} - \theta, \frac{1}{2} + \theta]$
2. Draw $Y_i \sim \text{Binom}(m, p_i)$




- Privacy of sPBM

$$\varepsilon(\alpha) = \max_{p'_1, p_1, \ldots, p_n} \mathsf{D}_\alpha\left(\sum_{i \in [n]} \text{Binom}(m, p_i) \middle\| \text{Binom}(m, p'_1) + \sum_{i \in [2:n]} \text{Binom}(m, p_i)\right)$$

1. By the (quasi) convexity of divergence, maximum occurs when $p_i \in \left\{\frac{1}{2} - \theta, \frac{1}{2} + \theta\right\}$
2. Decompose the sum via data-processing inequalities
3. Bound the divergence with the sub-Gaussian norm of the likelihood ratio.

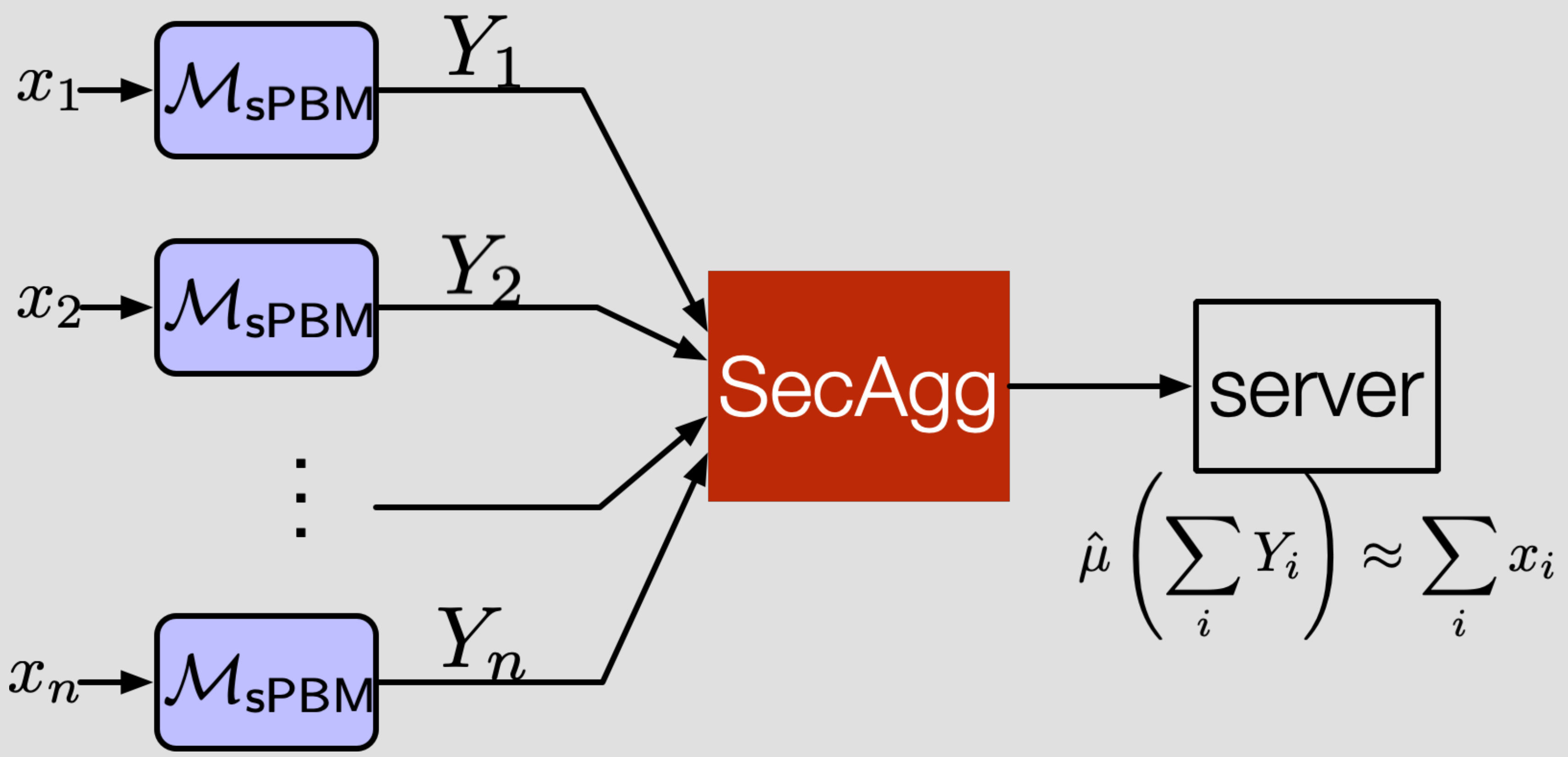
# Privacy of sPBM

**Algorithm (sPBM)**

**Parameters:** $m \in \mathbb{N}$, $\theta \le 0.1$

For client $i$:

1. Re-scale $x_i$ to $p_i \in [\frac{1}{2} - \theta, \frac{1}{2} + \theta]$
2. Draw $Y_i \sim \text{Binom}(m, p_i)$




- Privacy of sPBM

$$\varepsilon(\alpha) = \max_{p_1', p_1, \ldots, p_n} \mathrm{D}_\alpha \left( \textstyle\sum_{i\in[n]} \text{Binom}(m, p_i) \middle\| \text{Binom}(m, p_1') + \textstyle\sum_{i\in[2:n]} \text{Binom}(m, p_i) \right)$$

1. By the (quasi) convexity of divergence, maximum occurs when $p_i \in \left\{\frac{1}{2} - \theta, \frac{1}{2} + \theta\right\}$
2. Decompose the sum via data-processing inequalities
3. Bound the divergence with the sub-Gaussian norm of the likelihood ratio.

$$\varepsilon(\alpha)\text{-DP for } \varepsilon(\alpha) \ge \Omega\left(\alpha m \theta^2 / n\right)$$

# The Poisson binomial mechanism (PBM)

**Algorithm (PBM)**

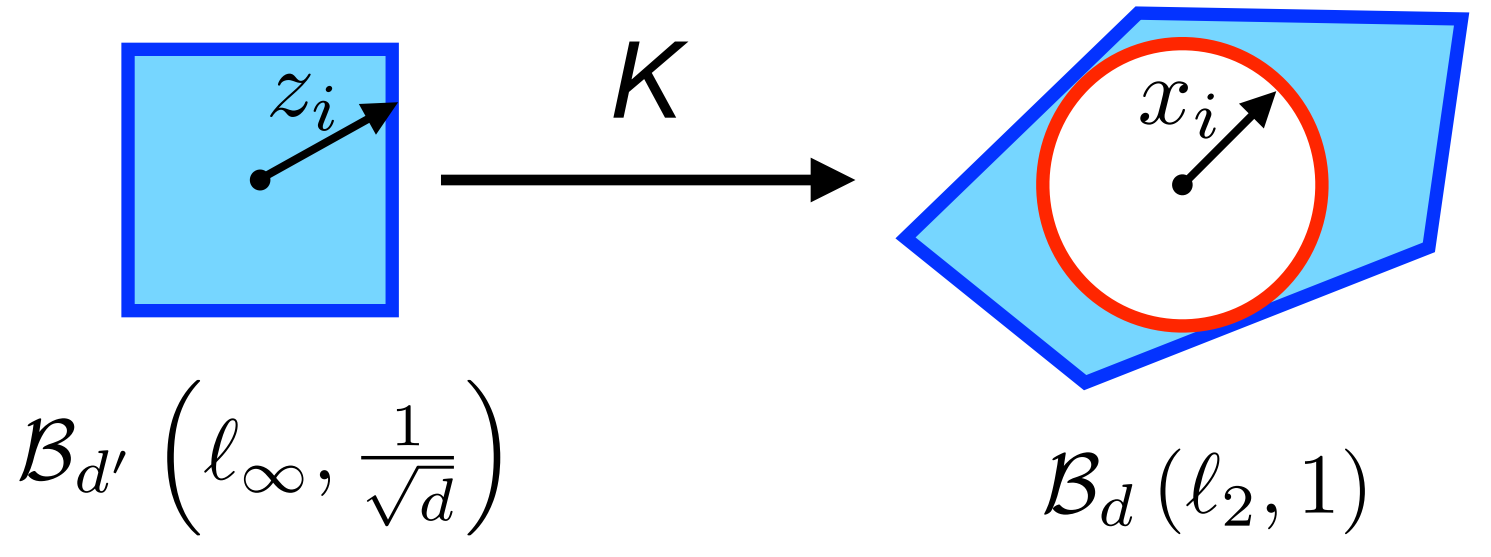
**Parameters:** $m \in \mathbb{N}$, $\theta \leq 0.1$
**Input:** $x_1, ..., x_n \in \mathbb{R}^d$, $\ell_2$-norm bound $c$

For client $i$:

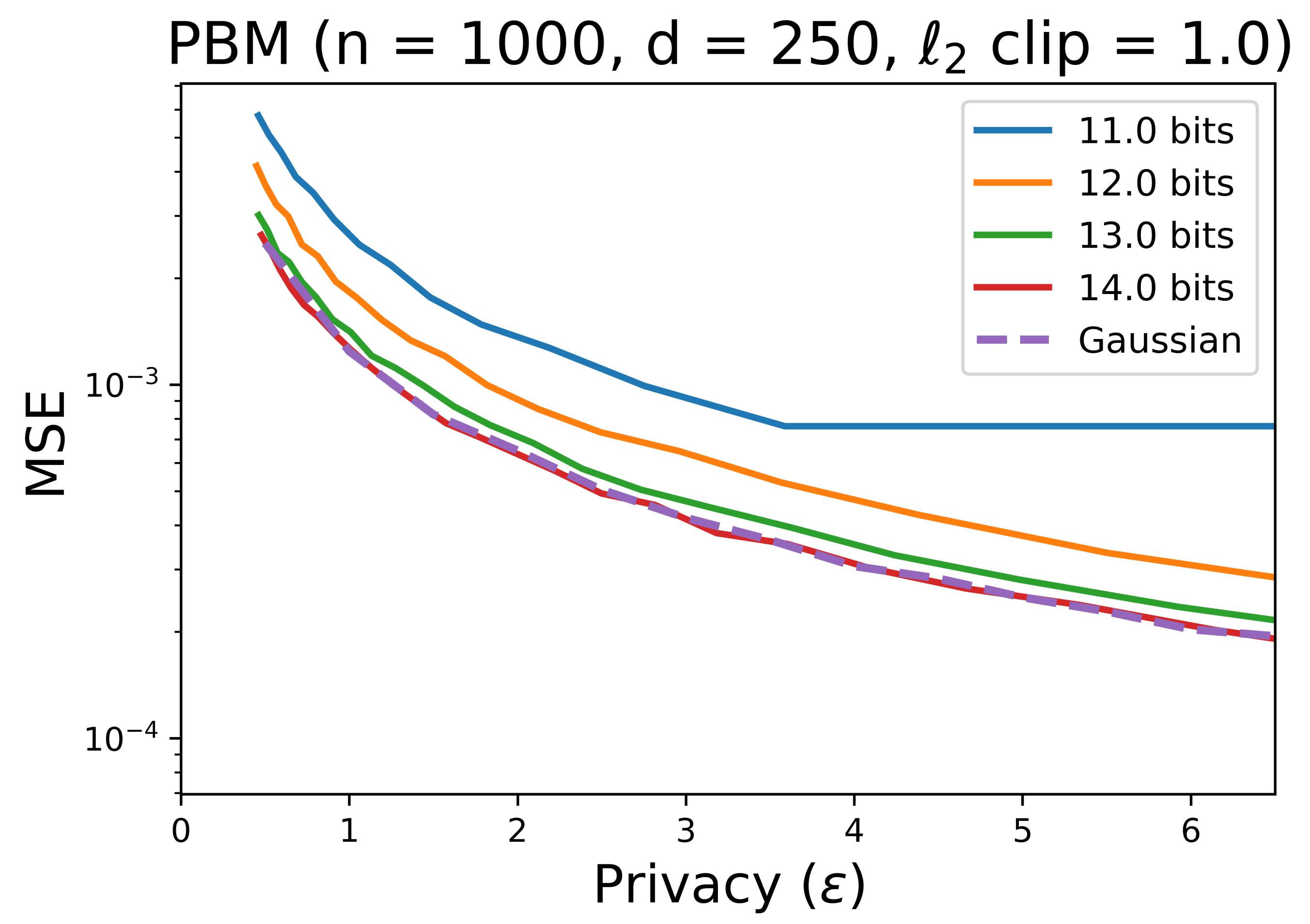
1. Compute Kashin's representation $z_i$ with $\|z_i\|_\infty = \Theta\left(\frac{K}{\sqrt{d}}\right)$
2. For each coordinate of $z_i$, apply sPBM

Server estimates $\frac{1}{n}\sum_i z_i$
Server recovers $\frac{1}{n}\sum_i x_i$ from the (estimated) Kashin's representation

**Kashin's representation**

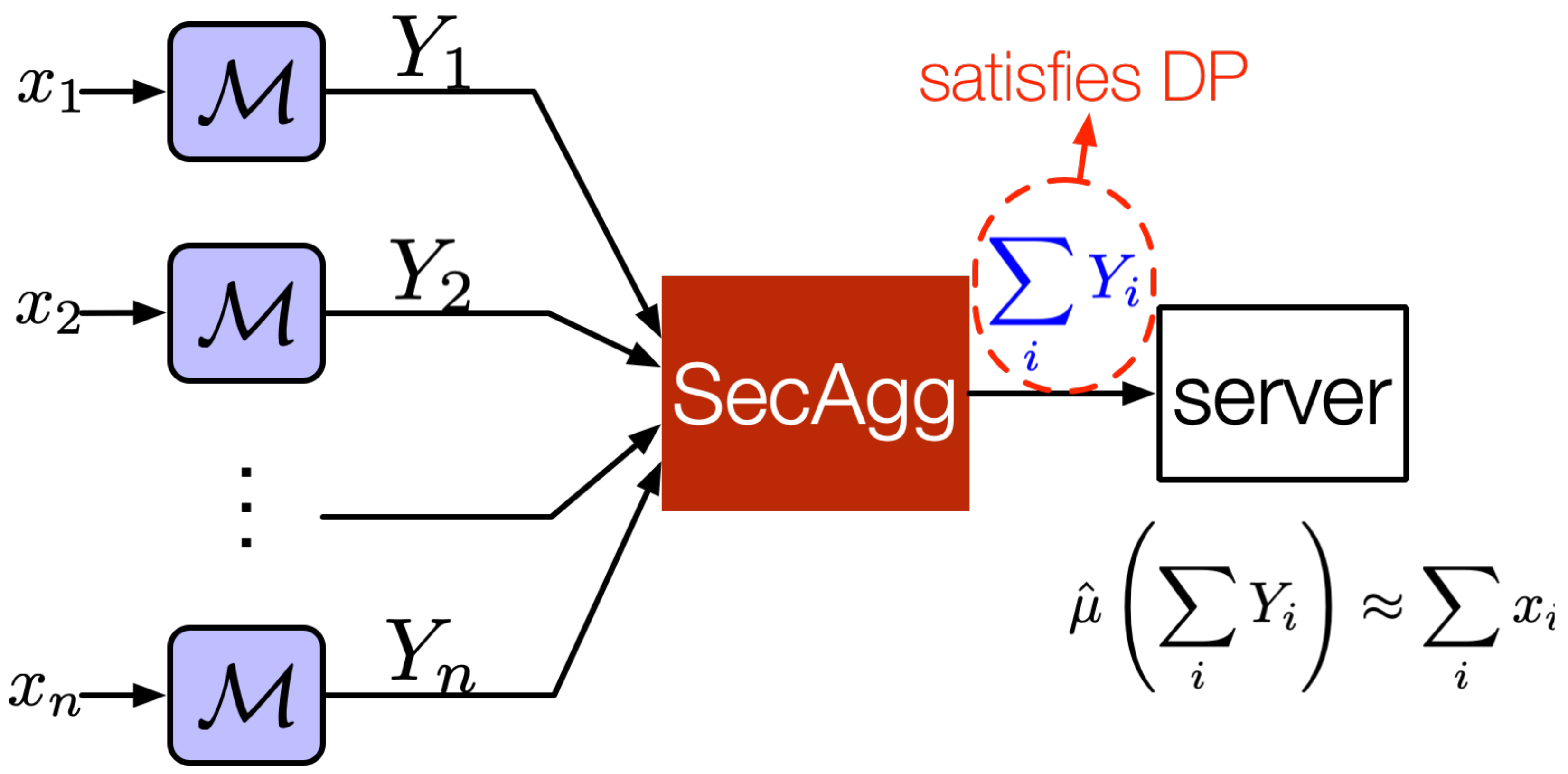



**Distributed mean estimation**



# Compare with Prior Works

**Mean estimation with SecAgg and DP**

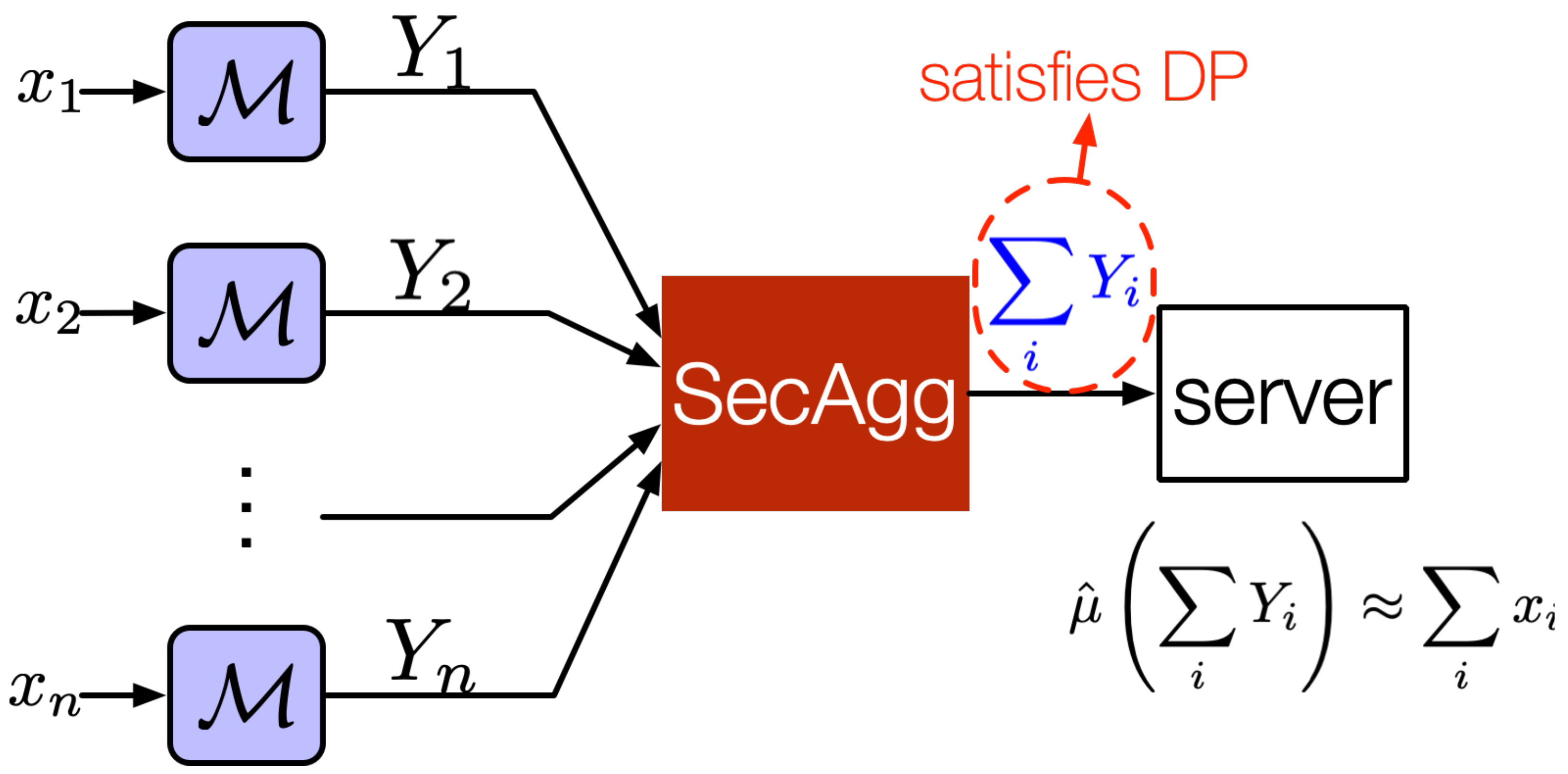




[1] Suresh Ananda Theertha, et al. "cpSGD: Communication-efficient and differentially-private distributed SGD." NeurIPS 2018.
[2] Peter Kairouz, et al. "The distributed discrete gaussian mechanism for federated learning with secure aggregation." ICML 2021.
[3] Naman Argawal, et al. "The skellam mechanism for differentially private federated learning." NeurIPS 2021.
[4] Albert Cheu, et al. "Distributed Differential Privacy via Shuffling." EuroCrypt 2019.

# Compare with Prior Works

## Mean estimation with SecAgg and DP

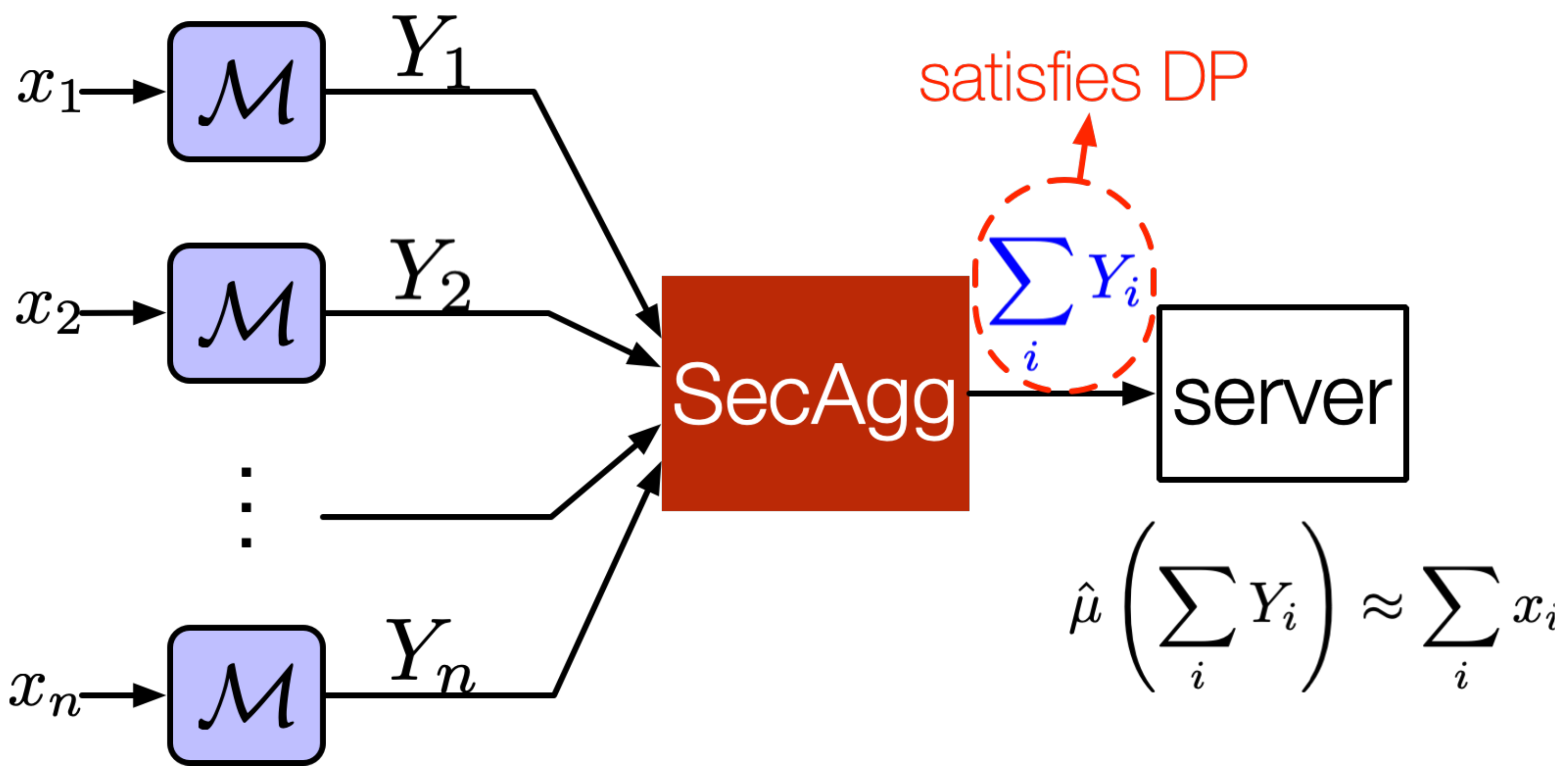



| | communication | MSE | bias |
|---|---|---|---|
| PBM | $O\left(d\log\left(n\cdot\left\lceil\frac{\varepsilon^2}{d}\right\rceil\right)\right)$ | $O_\delta\left(\frac{c^2 d}{n^2\varepsilon^2}\right)$ | no |
| Skellam | $O\left(d\log\left(n\cdot\left\lceil\frac{d}{\varepsilon^2}\right\rceil\right)\right)$ | $O_\delta\left(\frac{c^2 d}{n^2\varepsilon^2}\right)$ | yes |
| DDG | $O\left(d\log\left(n\cdot\left\lceil\frac{d}{\varepsilon^2}\right\rceil\right)\right)$ | $O_\delta\left(\frac{c^2 d}{n^2\varepsilon^2}\right)$ | yes |
| binomial | $O\left(d\log\left(n\cdot\left\lceil\frac{d}{\varepsilon^2}\right\rceil\right)\right)$ | $O_\delta\left(\frac{c^2 d\log d}{n^2\varepsilon^2}\right)$ | yes |


[1] Suresh Ananda Theertha, et al. "cpSGD: Communication-efficient and differentially-private distributed SGD." NeurIPS 2018.
[2] Peter Kairouz, et al. "The distributed discrete gaussian mechanism for federated learning with secure aggregation." ICML 2021.
[3] Naman Argawal, et al. "The skellam mechanism for differentially private federated learning." NeurIPS 2021.
[4] Albert Cheu, et al. "Distributed Differential Privacy via Shuffling." EuroCrypt 2019.

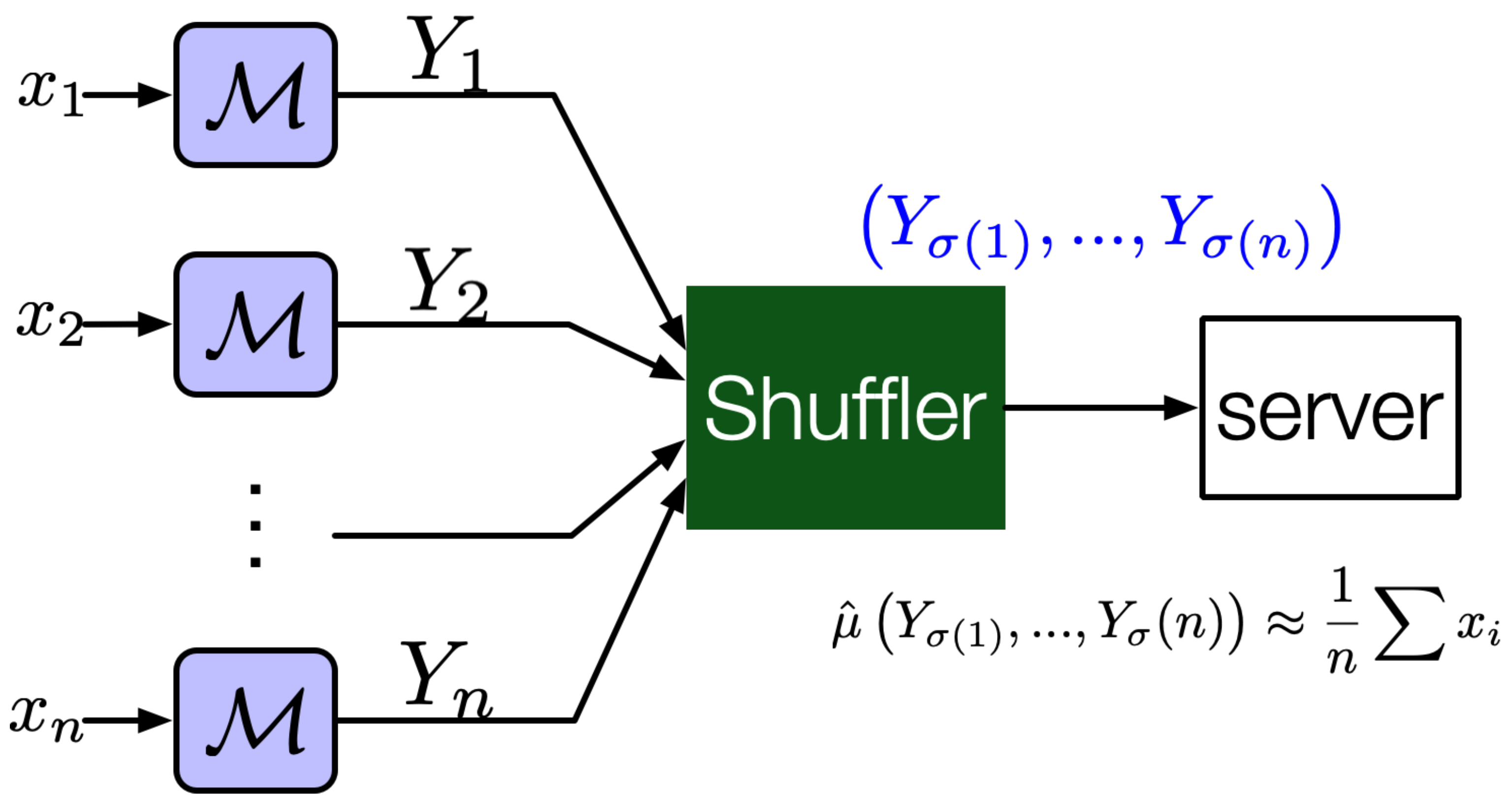
# Compare with Prior Works

**Mean estimation with SecAgg and DP**




| | communication | MSE | bias |
|---|---|---|---|
| PBM | $O\left(d\log\left(n\cdot\left\lceil\frac{\varepsilon^2}{d}\right\rceil\right)\right)$ | $O_\delta\left(\frac{c^2 d}{n^2\varepsilon^2}\right)$ | no |
| Skellam | $O\left(d\log\left(n\cdot\left\lceil\frac{d}{\varepsilon^2}\right\rceil\right)\right)$ | $O_\delta\left(\frac{c^2 d}{n^2\varepsilon^2}\right)$ | yes |
| DDG | $O\left(d\log\left(n\cdot\left\lceil\frac{d}{\varepsilon^2}\right\rceil\right)\right)$ | $O_\delta\left(\frac{c^2 d}{n^2\varepsilon^2}\right)$ | yes |
| binomial | $O\left(d\log\left(n\cdot\left\lceil\frac{d}{\varepsilon^2}\right\rceil\right)\right)$ | $O_\delta\left(\frac{c^2 d\log d}{n^2\varepsilon^2}\right)$ | yes |

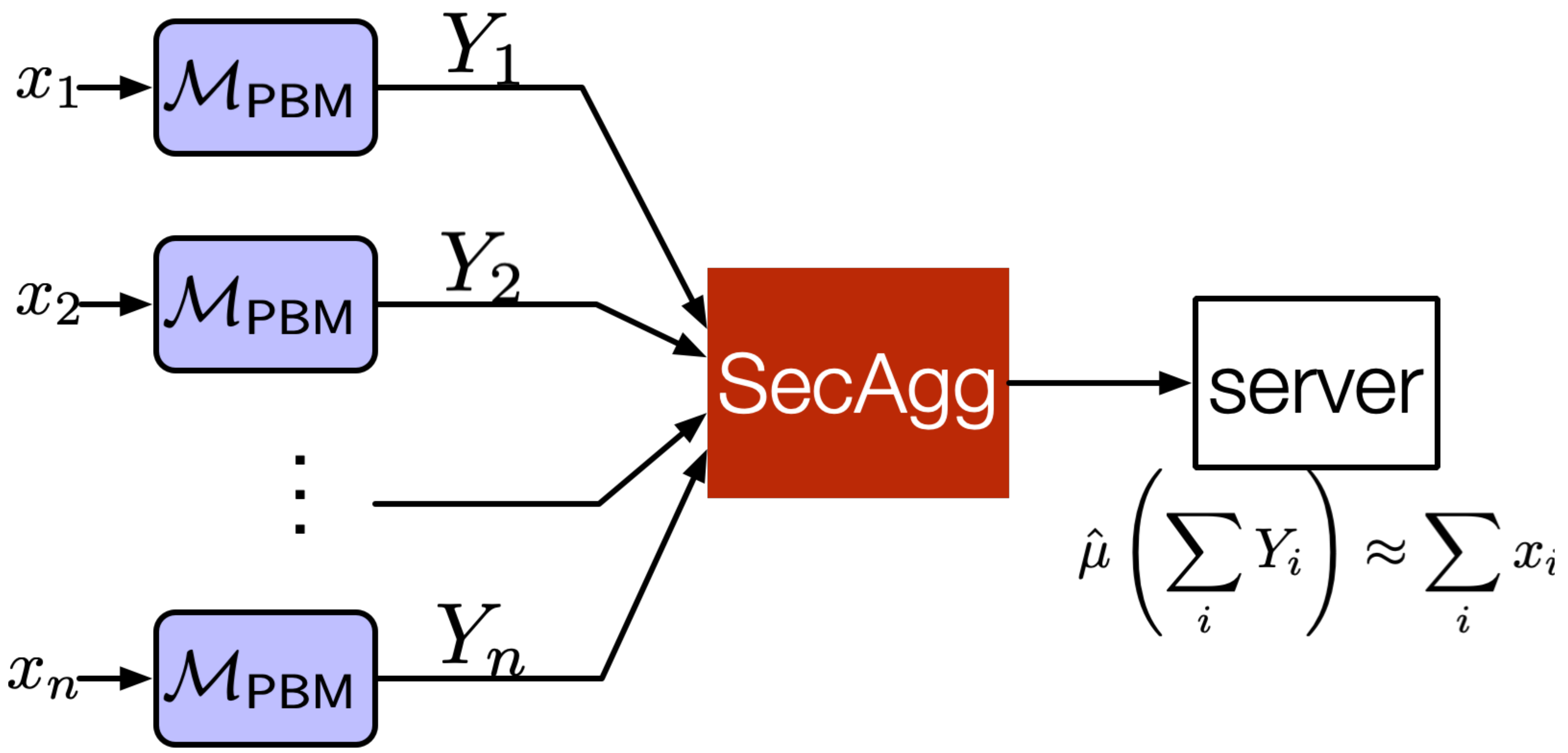
**Mean estimation with secure shuffling and DP**




- Compare to [4]: RR with secure shuffling
  - both introduce local binomial noise
  - under different secure models (SecAgg v.s secure shuffler)
  - we provide a Rènyi DP with numerically tight constants
  - extend to multi-dimensional mean estimation for FL

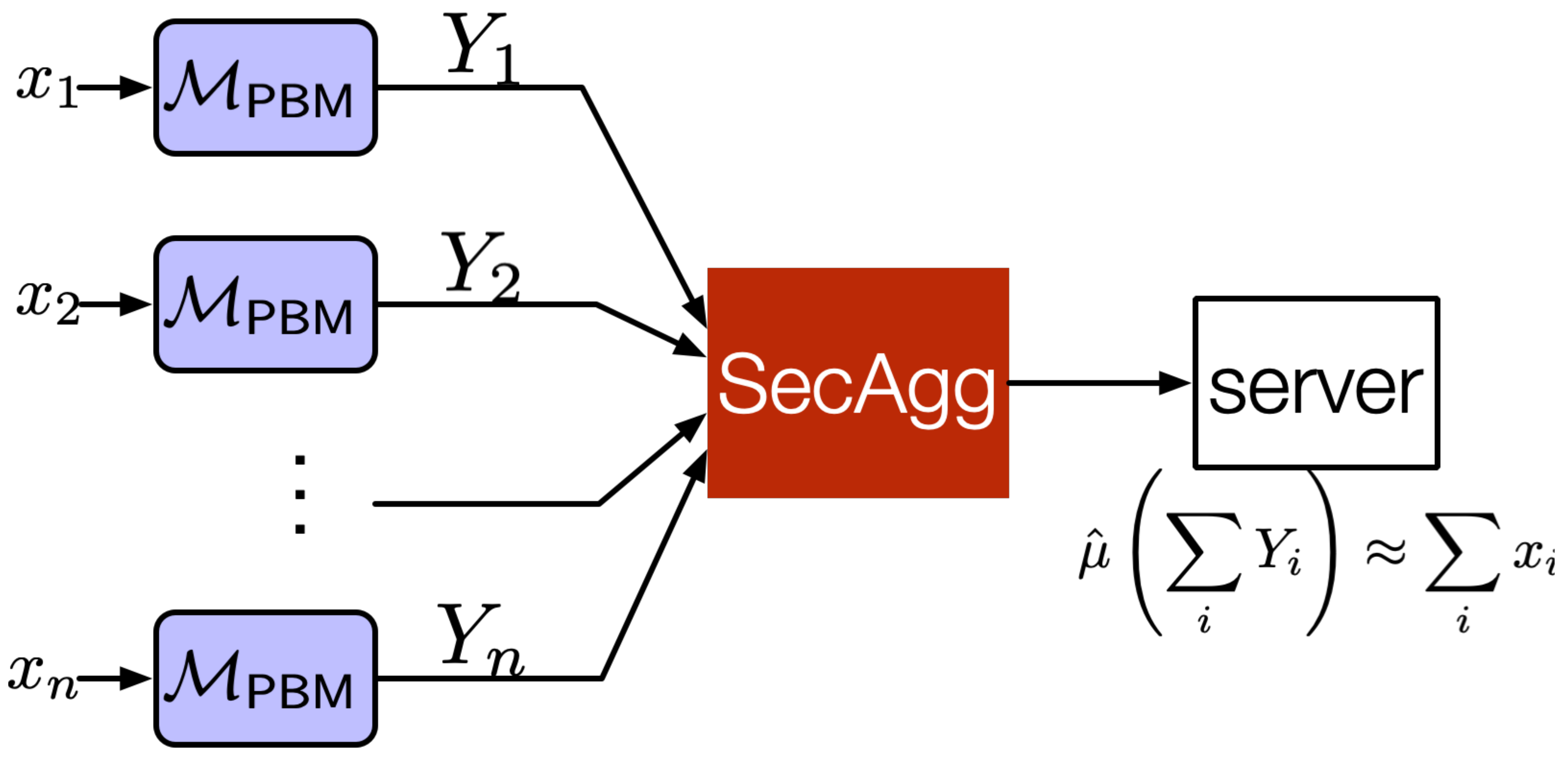

[1] Suresh Ananda Theertha, et al. "cpSGD: Communication-efficient and differentially-private distributed SGD." NeurIPS 2018.
[2] Peter Kairouz, et al. "The distributed discrete gaussian mechanism for federated learning with secure aggregation." ICML 2021.
[3] Naman Argawal, et al. "The skellam mechanism for differentially private federated learning." NeurIPS 2021.
[4] Albert Cheu, et al. "Distributed Differential Privacy via Shuffling." EuroCrypt 2019.

# Summary



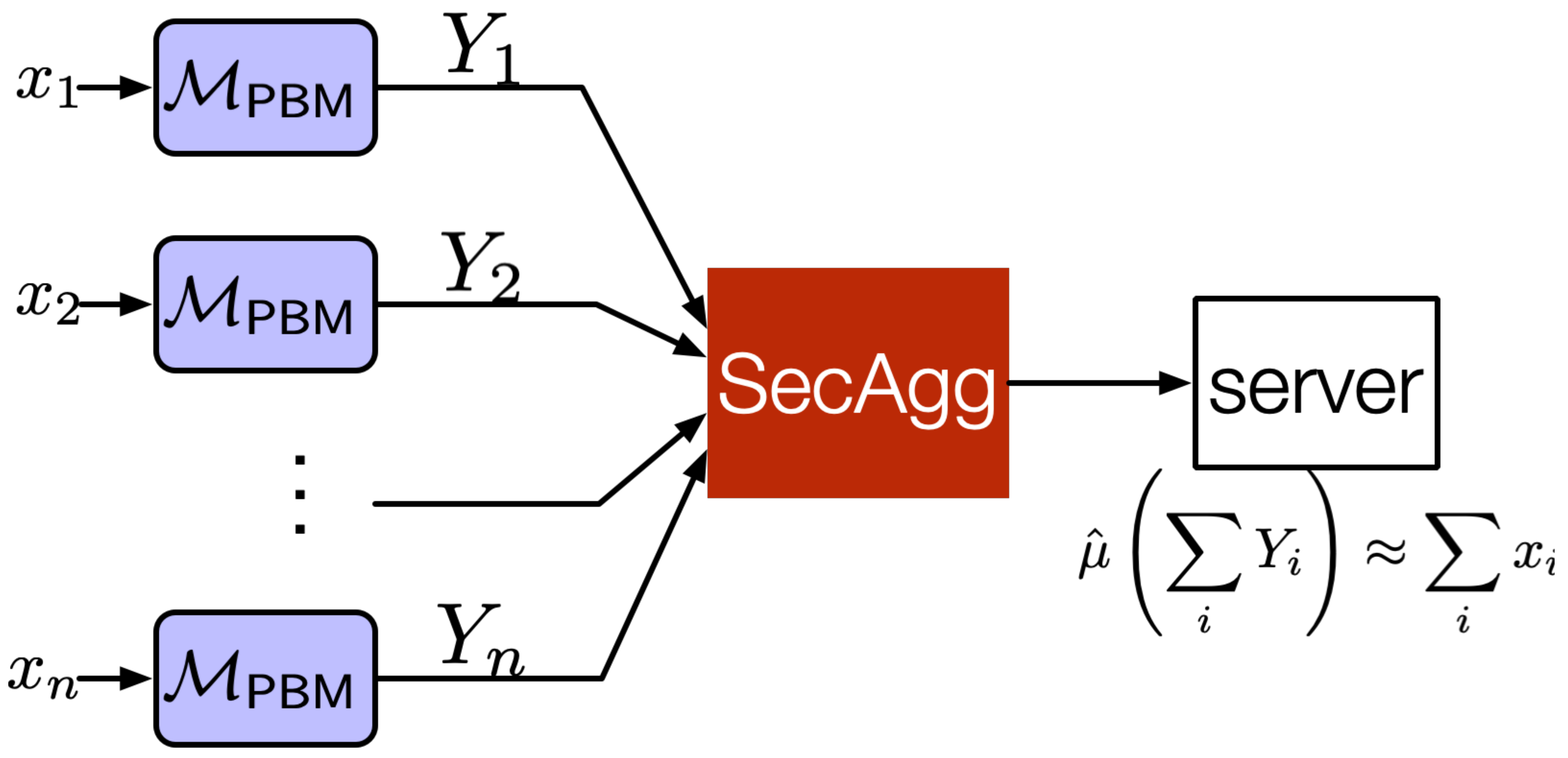
$x_1$
$\mathcal{M}_{\text{PBM}}$
$Y_1$
$x_2$
$\mathcal{M}_{\text{PBM}}$
$Y_2$
$x_n$
$\mathcal{M}_{\text{PBM}}$
$Y_n$
SecAgg
server
$\hat{\mu}\left(\sum_i Y_i\right) \approx \sum_i x_i$

# Summary




- **Unbiased** mean estimation scheme

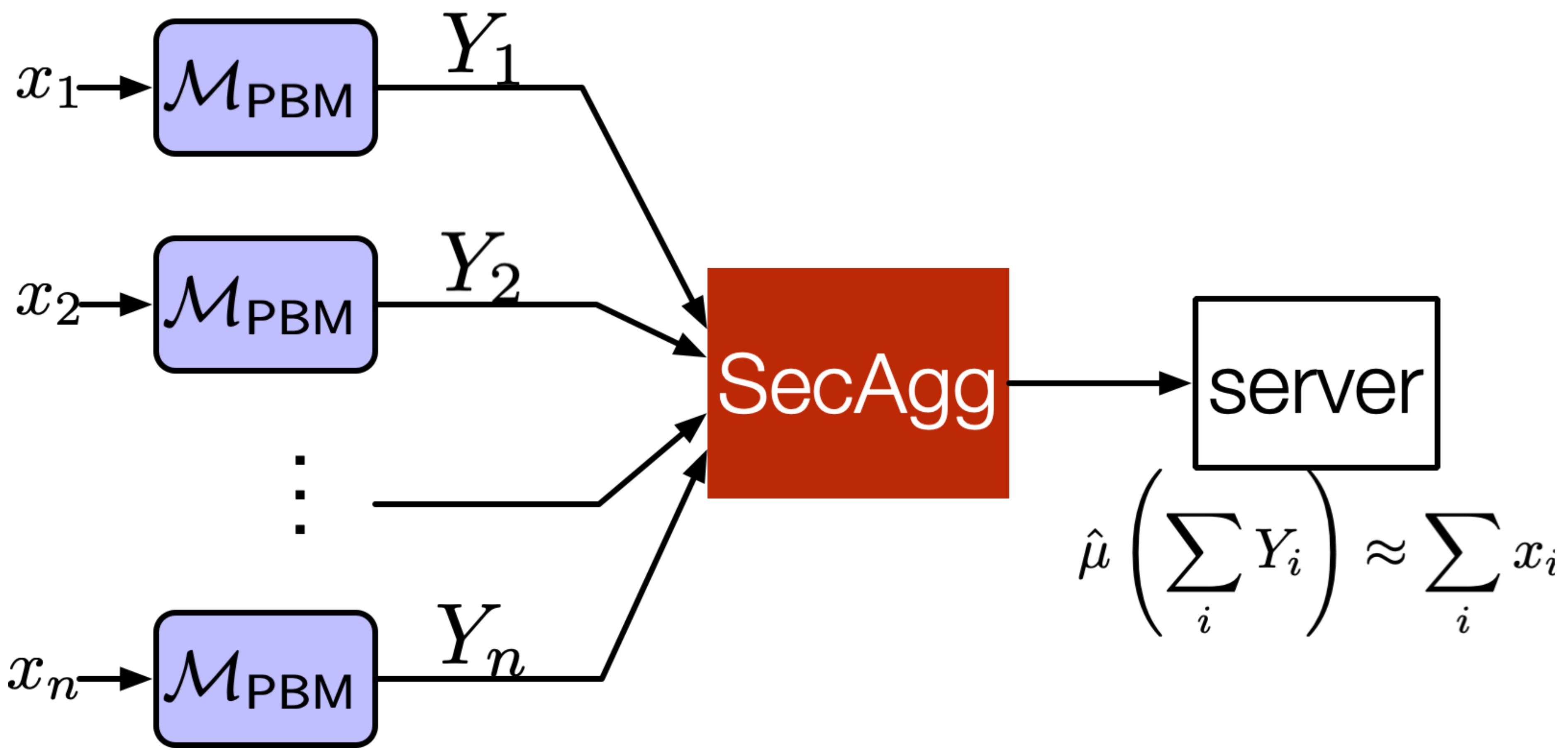
# Summary




| | communication | MSE | bias |
|---|---|---|---|
| PBM | $O\left(d\log\left(n\cdot\left\lceil\frac{\varepsilon^2}{d}\right\rceil\right)\right)$ | $O_\delta\left(\frac{c^2 d}{n^2\varepsilon^2}\right)$ | no |
| Skellam | $O\left(d\log\left(n\cdot\left\lceil\frac{d}{\varepsilon^2}\right\rceil\right)\right)$ | $O_\delta\left(\frac{c^2 d}{n^2\varepsilon^2}\right)$ | yes |
| DDG | $O\left(d\log\left(n\cdot\left\lceil\frac{d}{\varepsilon^2}\right\rceil\right)\right)$ | $O_\delta\left(\frac{c^2 d}{n^2\varepsilon^2}\right)$ | yes |
| binomial | $O\left(d\log\left(n\cdot\left\lceil\frac{d}{\varepsilon^2}\right\rceil\right)\right)$ | $O_\delta\left(\frac{c^2 d\log d}{n^2\varepsilon^2}\right)$ | yes |

- **Unbiased** mean estimation scheme
- Communication **decreases** with $\varepsilon$

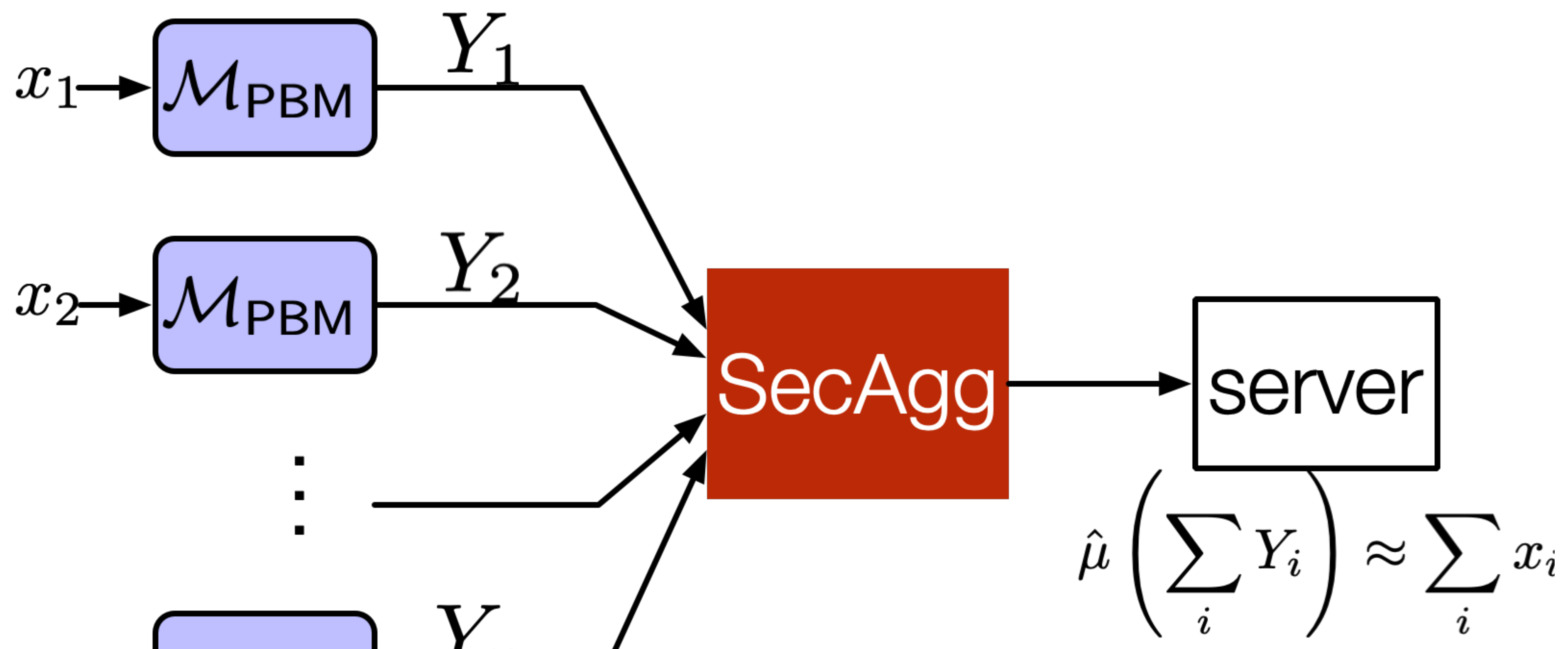
# Summary




| | communication | MSE | bias |
|---|---|---|---|
| PBM | $O\left(d \log\left(n \cdot \left\lceil \frac{\varepsilon^2}{d} \right\rceil\right)\right)$ | $O_\delta\left(\frac{c^2 d}{n^2 \varepsilon^2}\right)$ | no |
| Skellam | $O\left(d \log\left(n \cdot \left\lceil \frac{d}{\varepsilon^2} \right\rceil\right)\right)$ | $O_\delta\left(\frac{c^2 d}{n^2 \varepsilon^2}\right)$ | yes |
| DDG | $O\left(d \log\left(n \cdot \left\lceil \frac{d}{\varepsilon^2} \right\rceil\right)\right)$ | $O_\delta\left(\frac{c^2 d}{n^2 \varepsilon^2}\right)$ | yes |
| binomial | $O\left(d \log\left(n \cdot \left\lceil \frac{d}{\varepsilon^2} \right\rceil\right)\right)$ | $O_\delta\left(\frac{c^2 d \log d}{n^2 \varepsilon^2}\right)$ | yes |

- **Unbiased** mean estimation scheme
- Communication **decreases** with $\varepsilon$
- Order-optimal privacy-utility trade-off

# Summary




| | communication | MSE | bias |
|---|---|---|---|
| PBM | $O\left(d\log\left(n\cdot\left\lceil\frac{\varepsilon^2}{d}\right\rceil\right)\right)$ | $O_\delta\left(\frac{c^2d}{n^2\varepsilon^2}\right)$ | no |
| Skellam | $O\left(d\log\left(n\cdot\left\lceil\frac{d}{\varepsilon^2}\right\rceil\right)\right)$ | $O_\delta\left(\frac{c^2d}{n^2\varepsilon^2}\right)$ | yes |
| DDG | $O\left(d\log\left(n\cdot\left\lceil\frac{d}{\varepsilon^2}\right\rceil\right)\right)$ | $O_\delta\left(\frac{c^2d}{n^2\varepsilon^2}\right)$ | yes |
| binomial | $O\left(d\log\left(n\cdot\left\lceil\frac{d}{\varepsilon^2}\right\rceil\right)\right)$ | $O_\delta\left(\frac{c^2d\log d}{n^2\varepsilon^2}\right)$ | yes |

- **Unbiased** mean estimation scheme
- Communication **decreases** with $\varepsilon$
- Order-optimal privacy-utility trade-off
- Allows for numerically computing the **exact** privacy loss