Flame: Differentially Private Federated Learning in the Shuffle Model

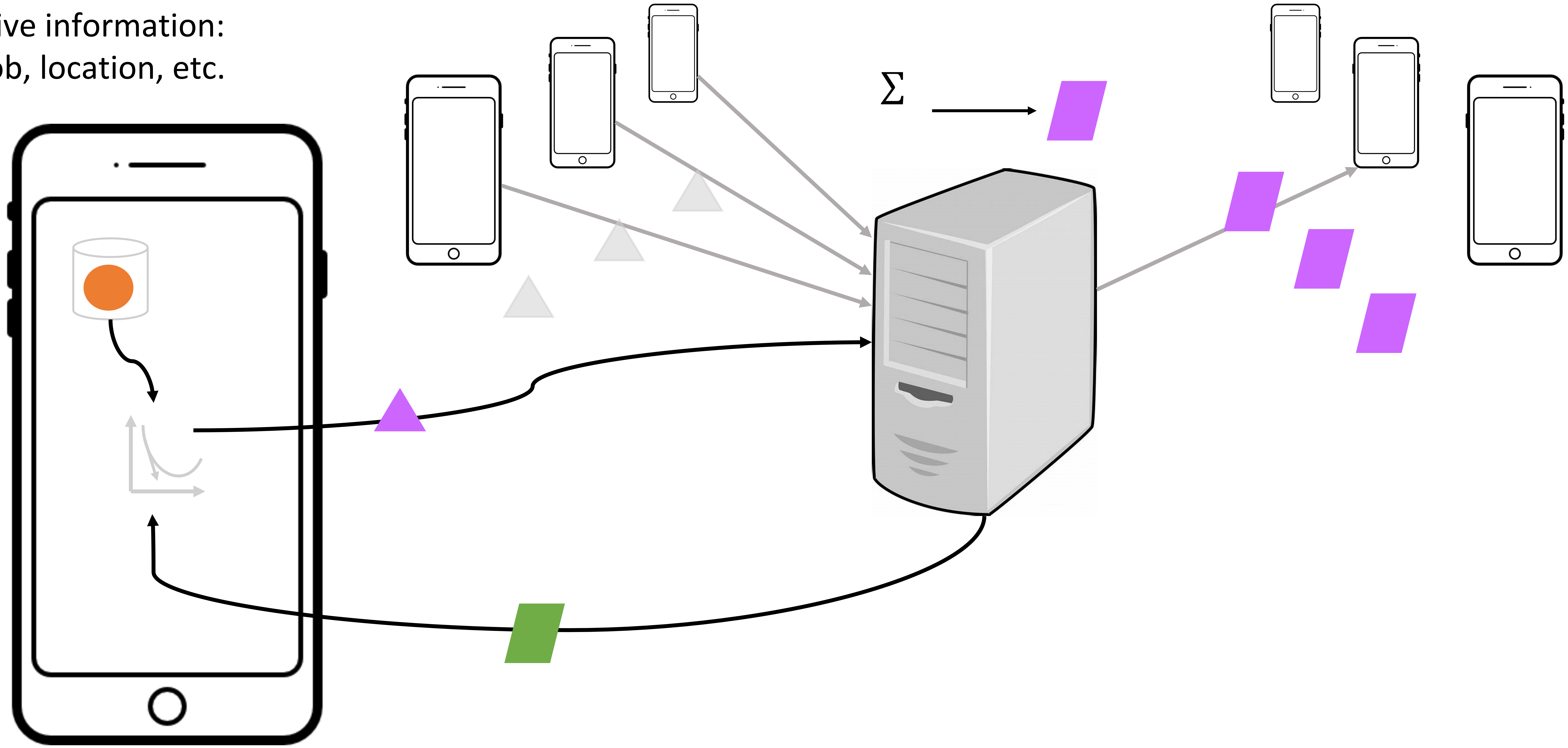
Ruixuan Liu[*]
Yang Cao[†]
Hong Chen[*]
Ruoyang Guo[*]
Masatoshi Yoshikawa[†]

Renmin University of China[*]
Kyoto University[†]

Motivation

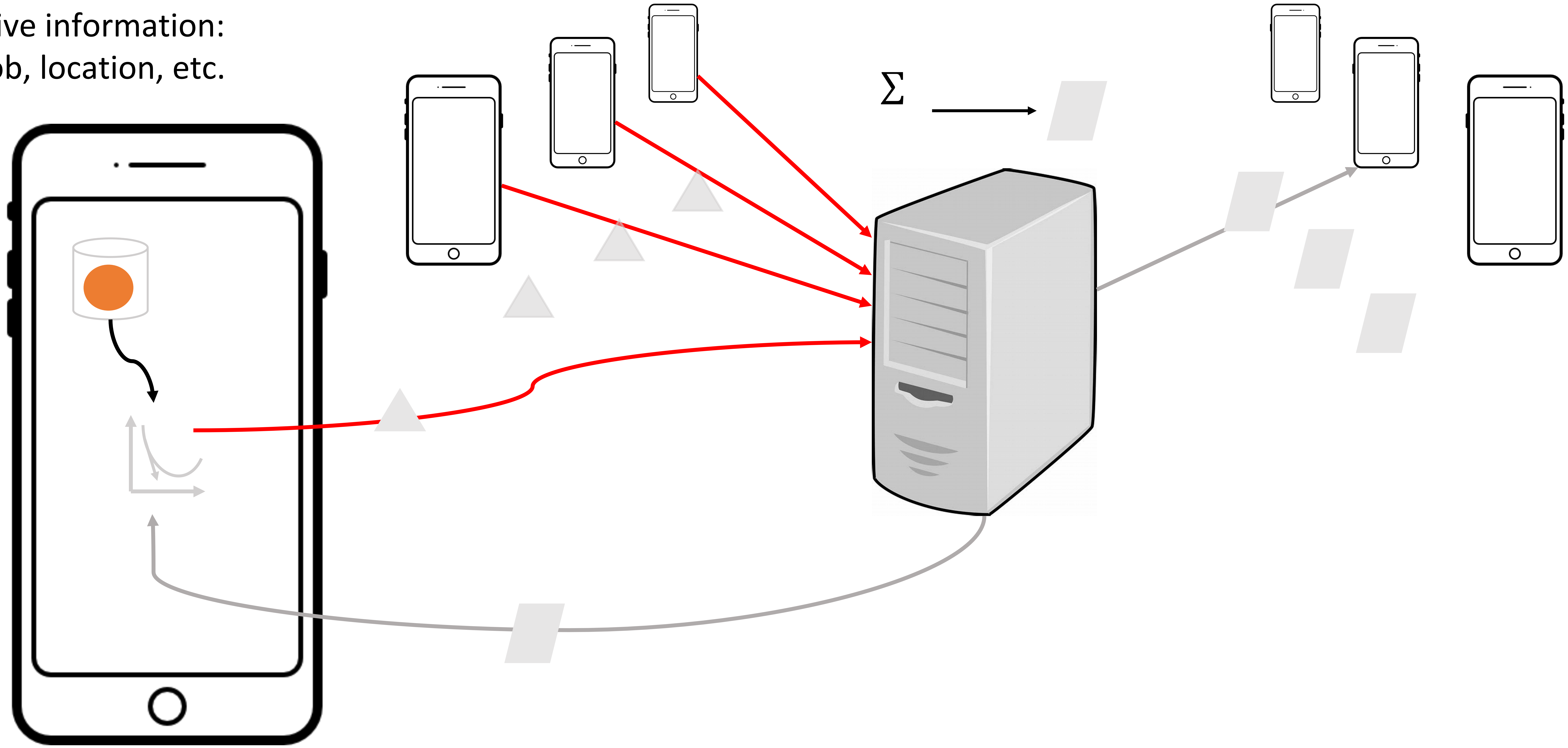
Privacy in Federated Learning

Sensitive information:
age, job, location, etc.



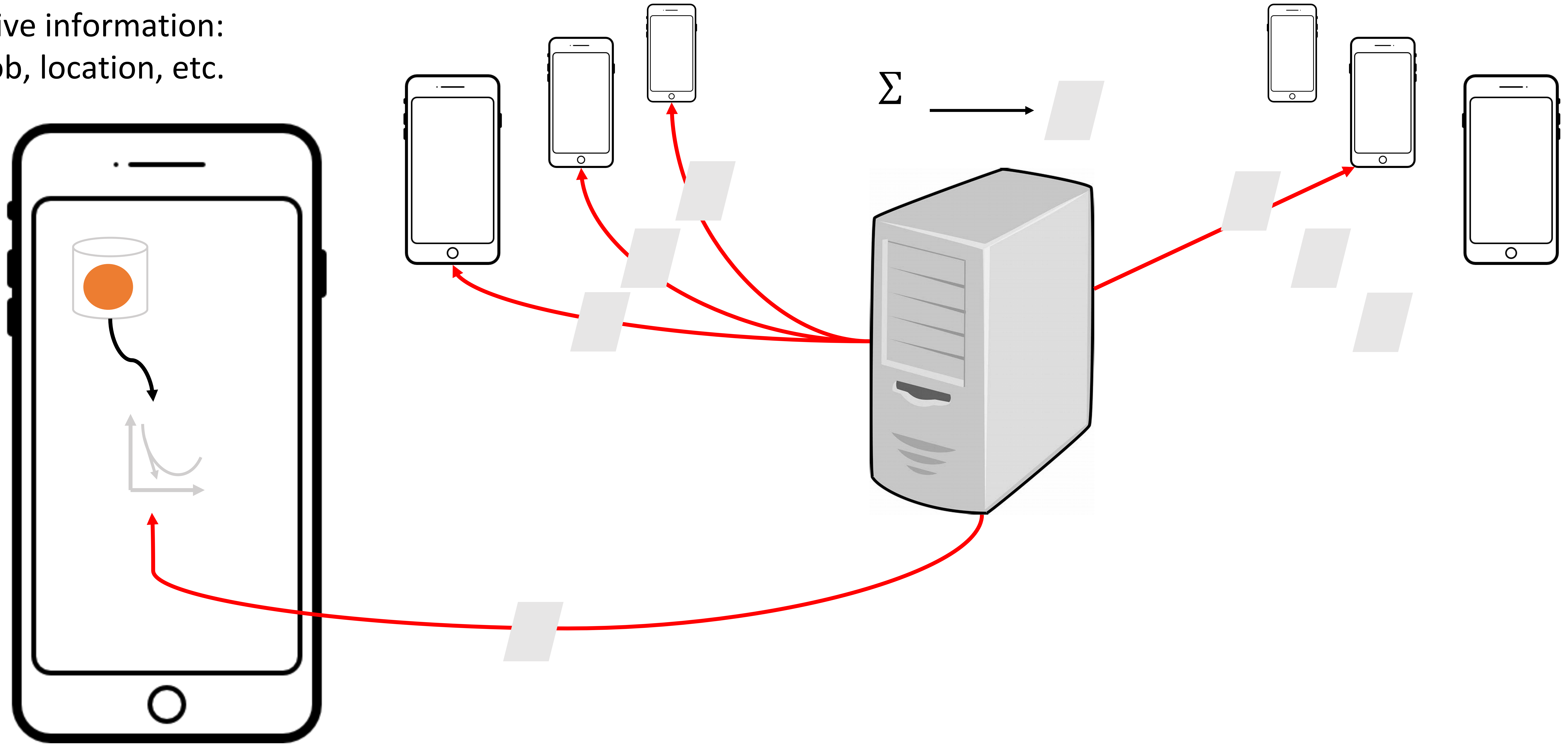
Privacy in Federated Learning

Sensitive information:
age, job, location, etc.



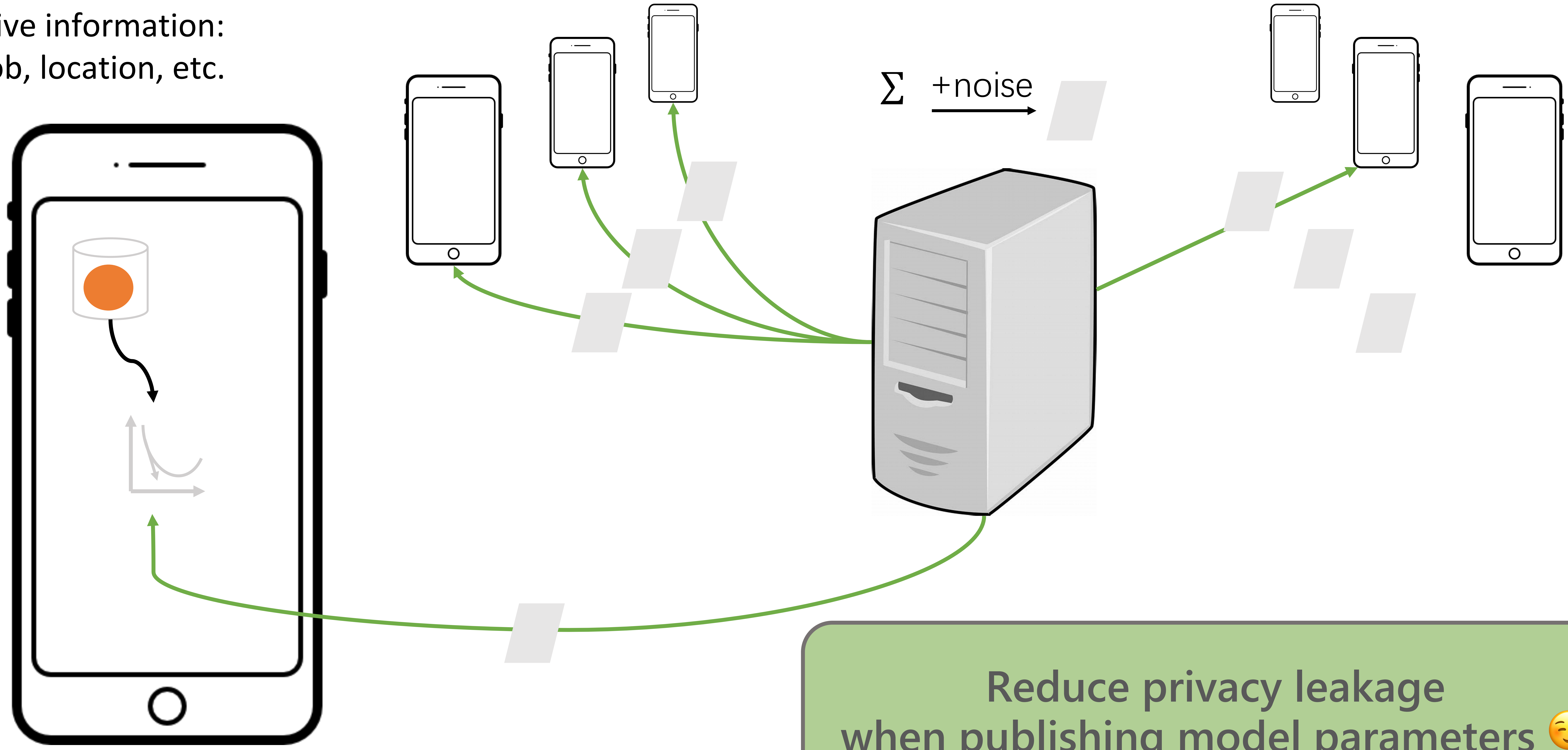
Privacy in Federated Learning

Sensitive information:
age, job, location, etc.



Differential Privacy for Federated Learning

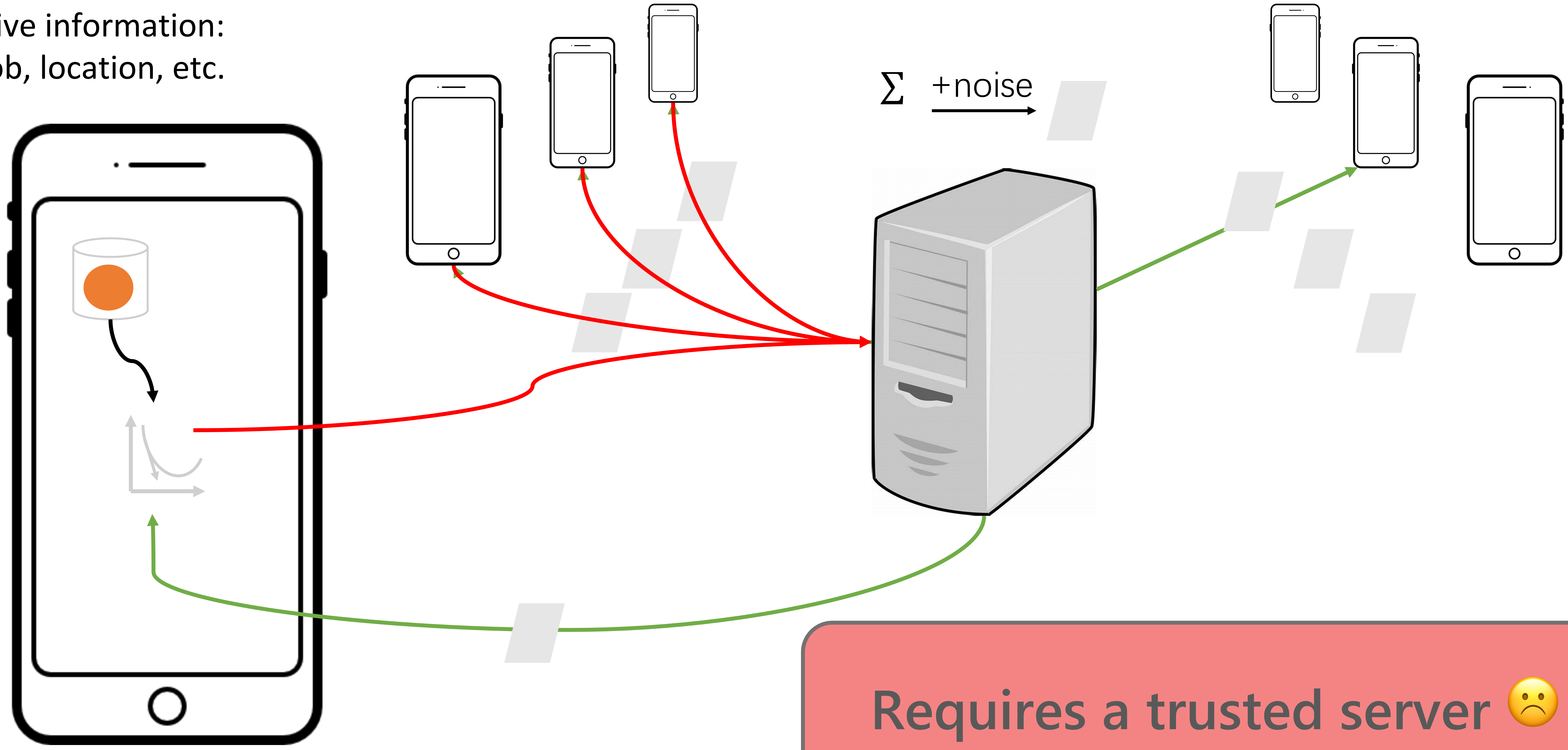
Sensitive information:
age, job, location, etc.



Reduce privacy leakage
when publishing model parameters ☺️

Differential Privacy for Federated Learning

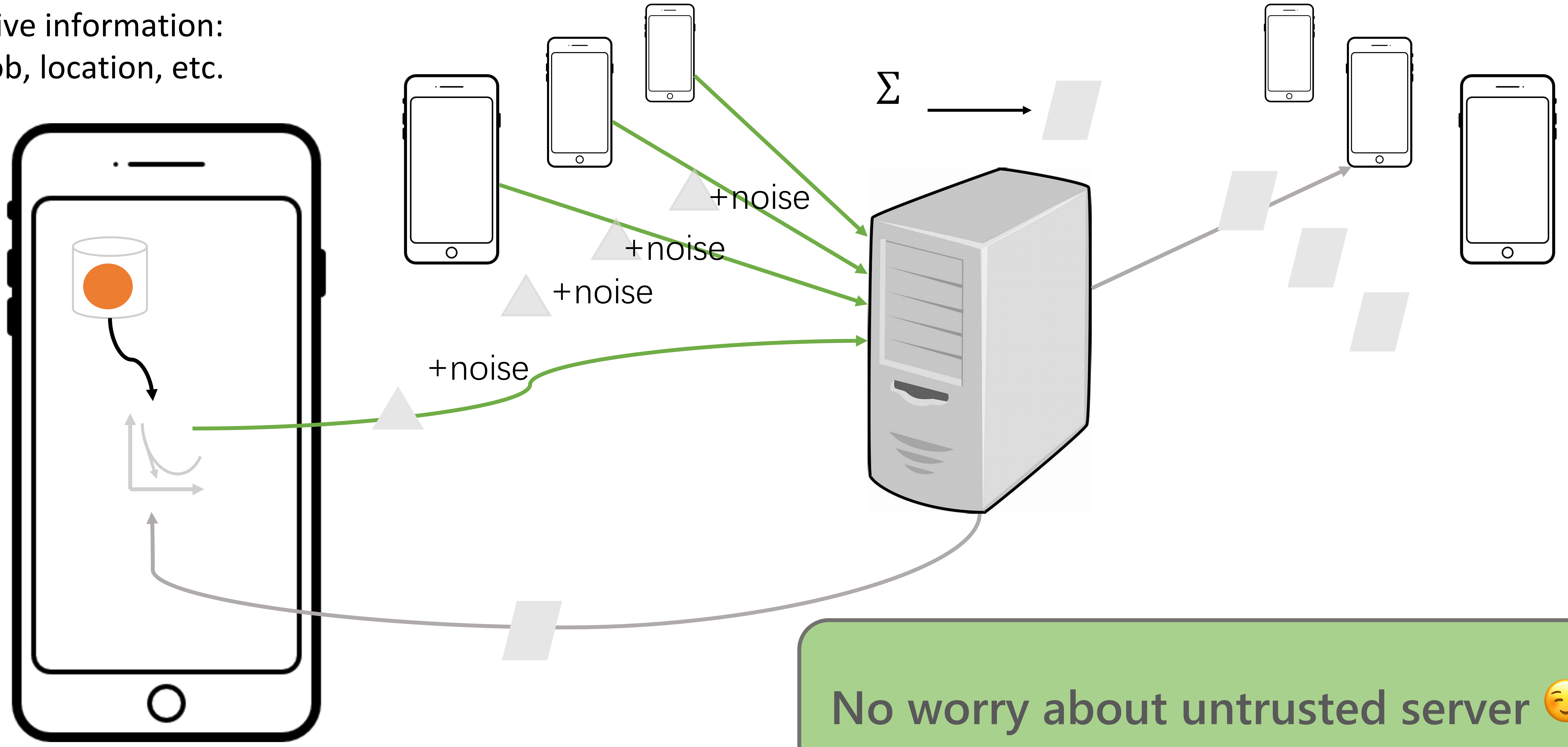
Sensitive information:
age, job, location, etc.



Requires a trusted server ☹

Local Differential Privacy for Federated Learning

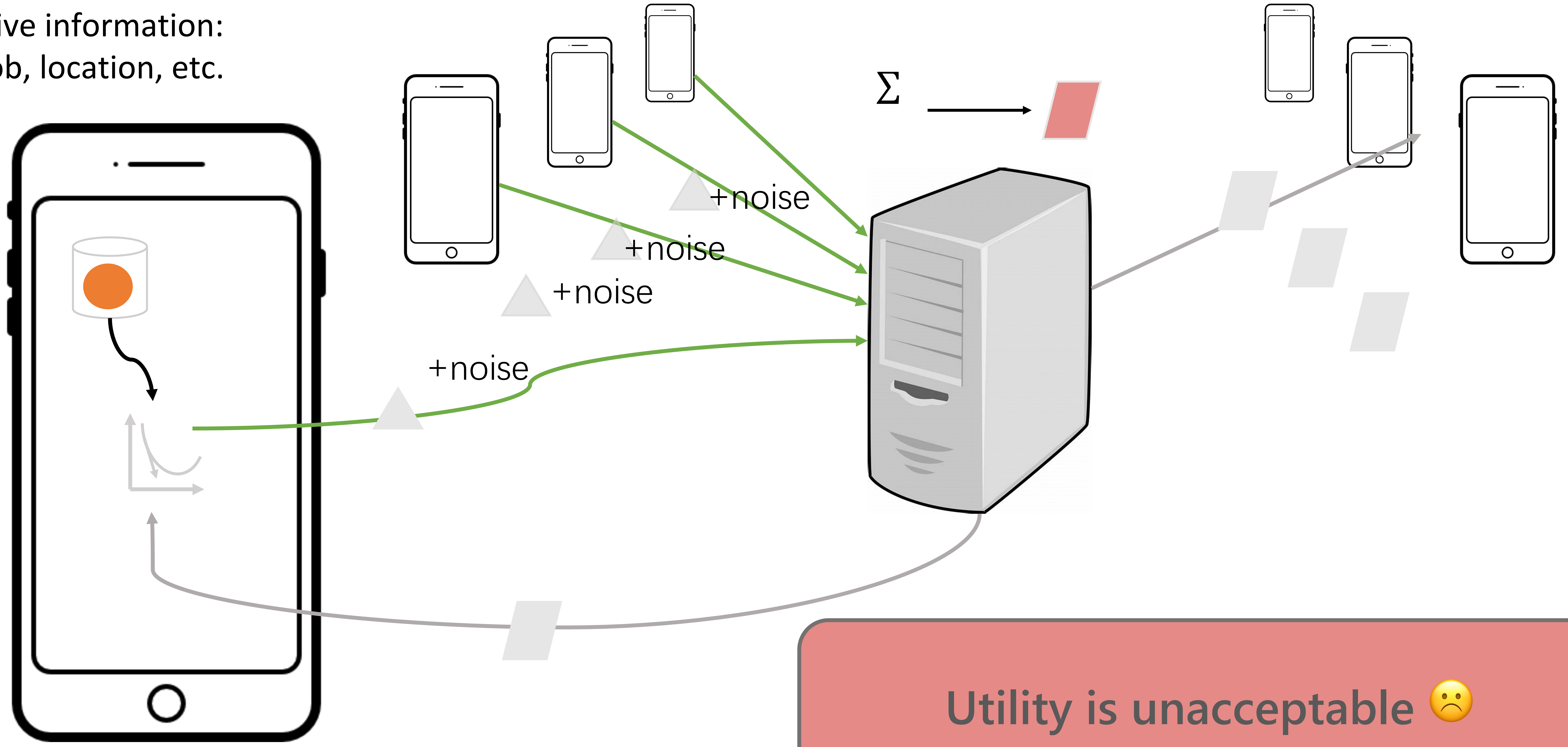
Sensitive information:
age, job, location, etc.



No worry about untrusted server ☺️

Local Differential Privacy for Federated Learning

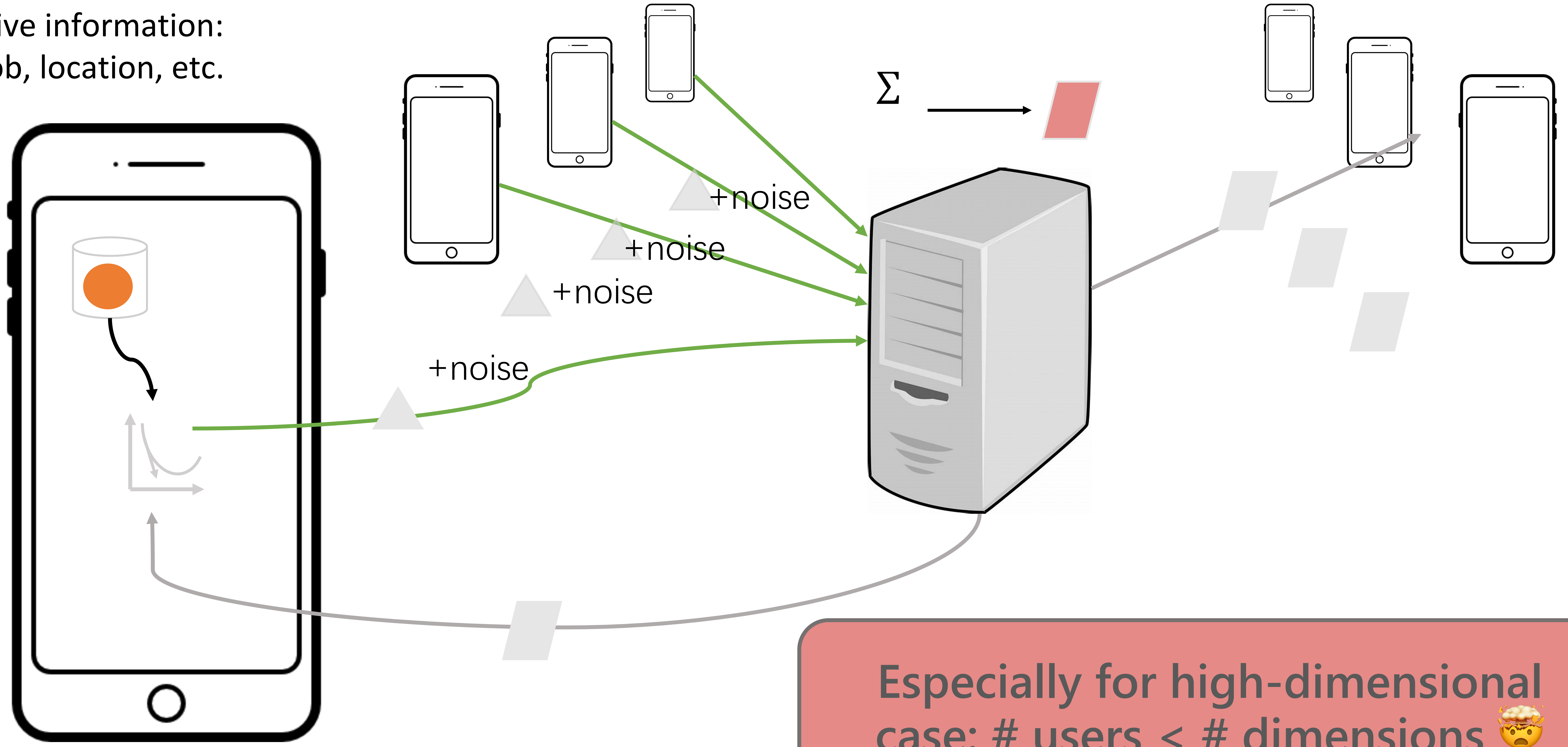
Sensitive information:
age, job, location, etc.



Utility is unacceptable ☹

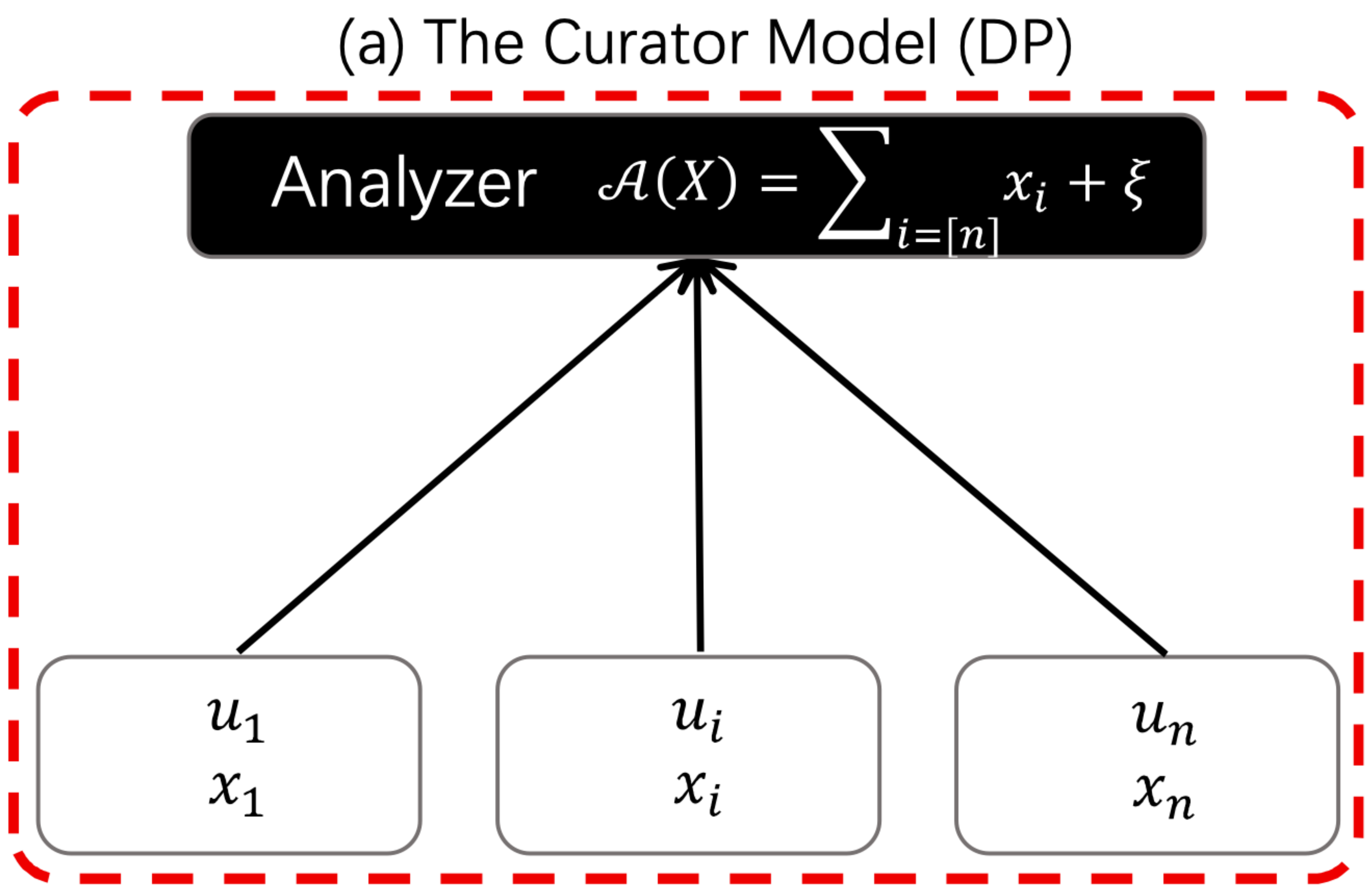
Local Differential Privacy for Federated Learning

Sensitive information:
age, job, location, etc.

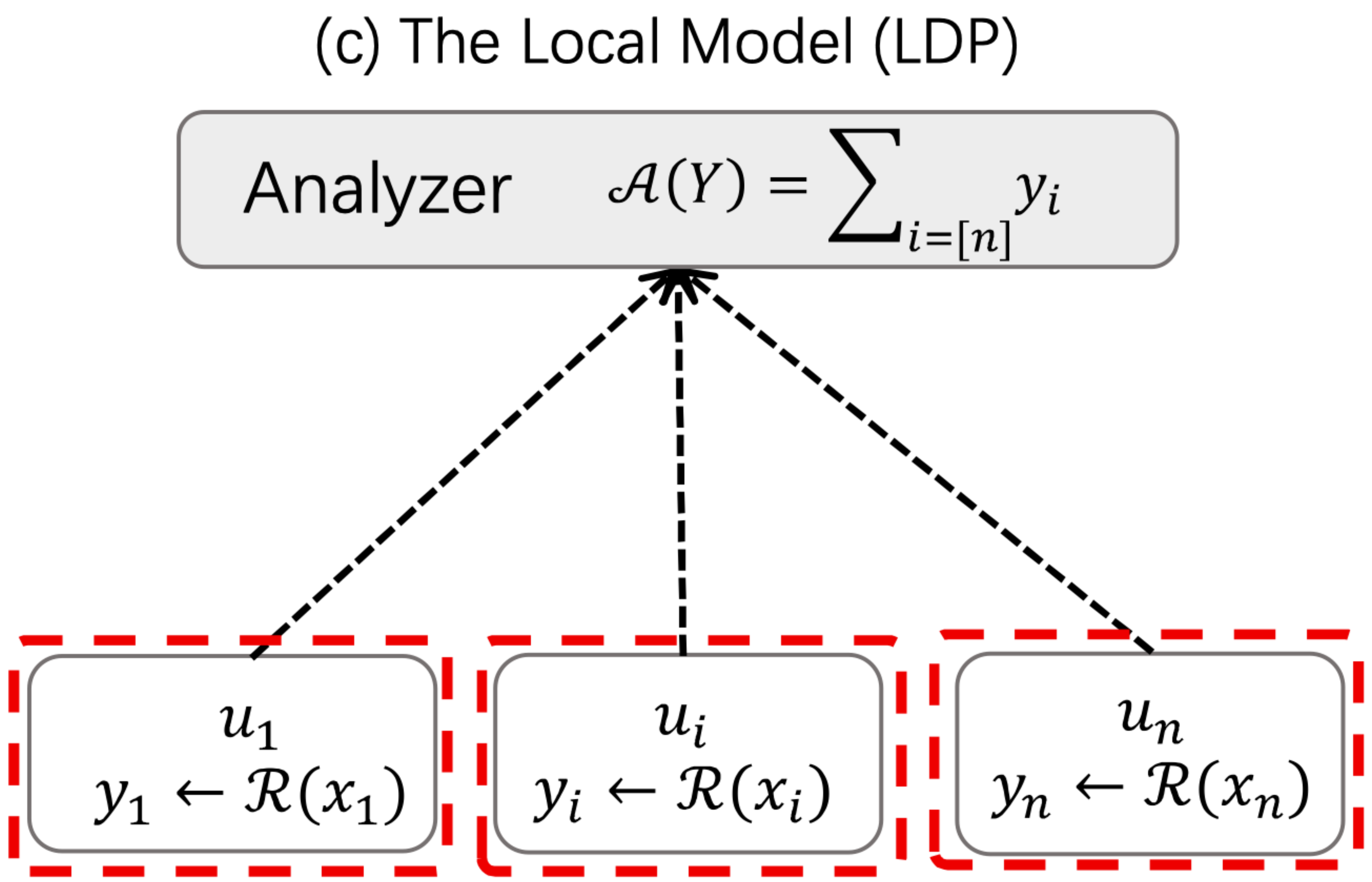


Especially for high-dimensional case: # users < # dimensions 🤯

Dilemma of Privacy-Utility Trade-off



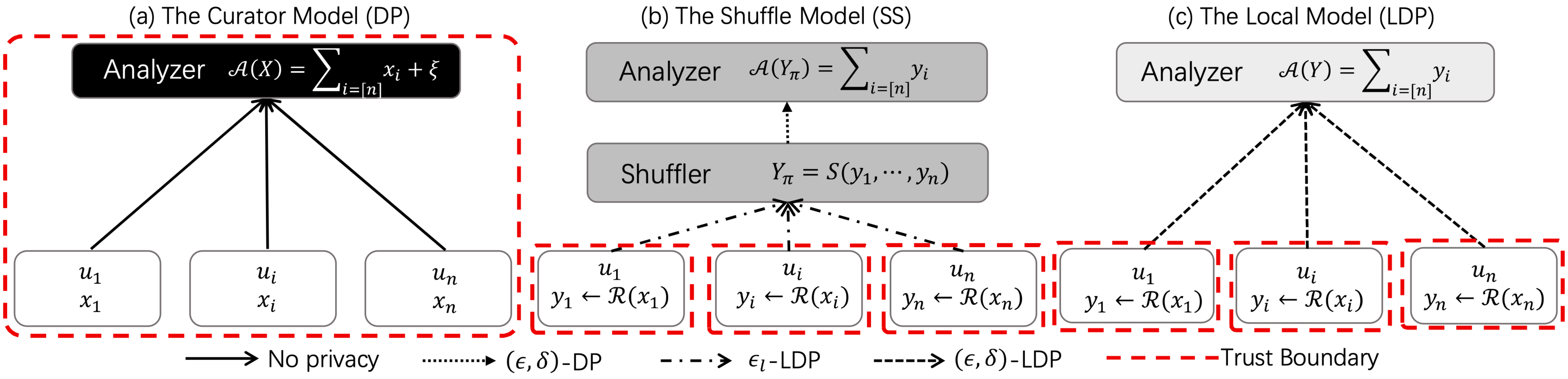
Better Utility



Better Privacy

Backgrounds

Better Trade-off in the Shuffle Model



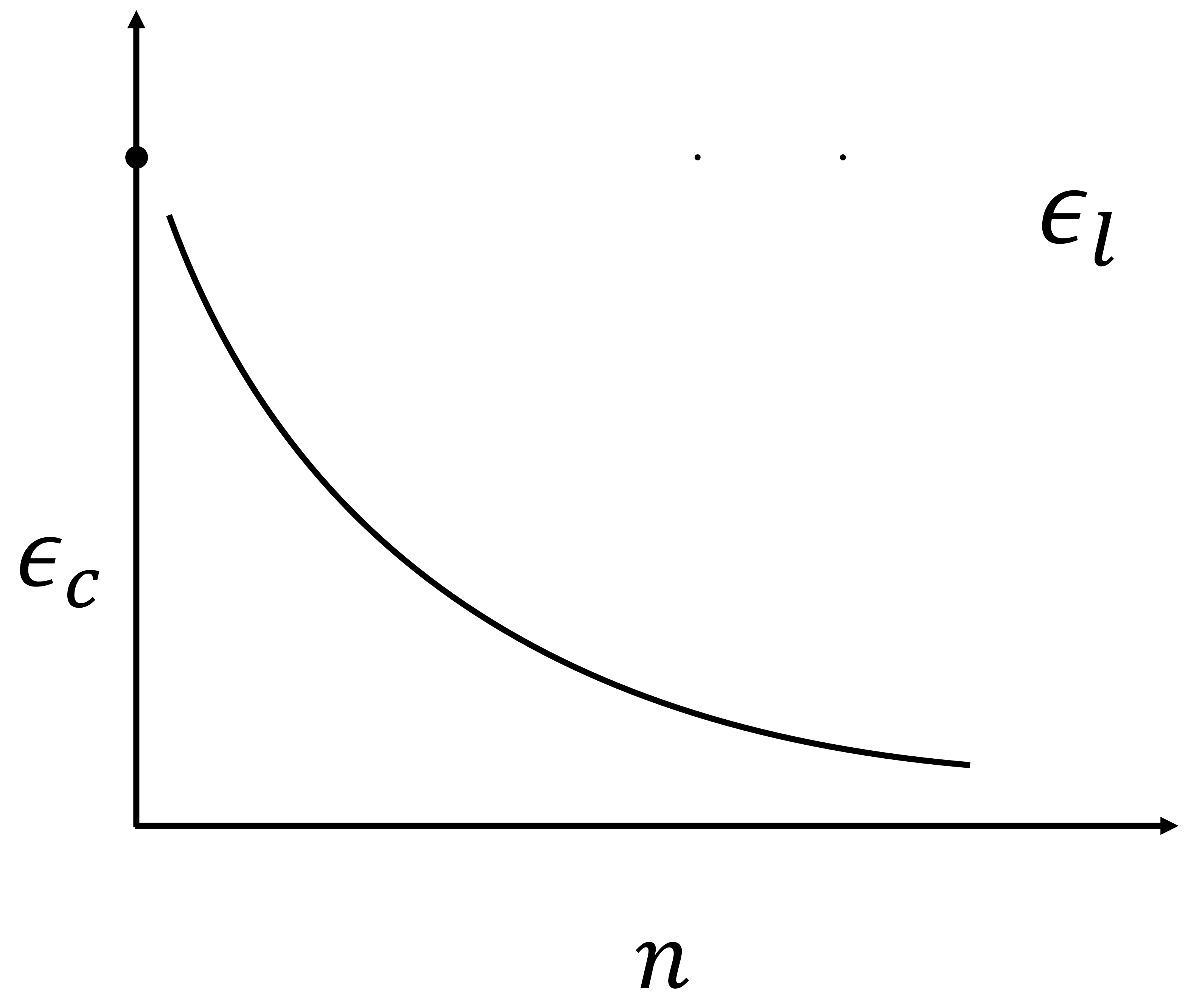
Better Privacy than DP

Better Utility than LDP

Better Trade-off in the Shuffle Model

Privacy amplification effect from shuffling [SODA'19][CRYPTO'19]

- Given a local privacy budget ϵ_l, the central privacy is amplified $\epsilon_c < \epsilon_l$



[SODA'2019]: Erlingsson L , Feldman V , Mironov I , et al. Amplification by Shuffling: From Local to Central Differential Privacy via Anonymity[M]// Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms. 2019.
[CRYPTO'2019]: Balle B., Bell J., Gascón A., Nissim K. (2019) The Privacy Blanket of the Shuffle Model. In: Boldyreva A., Micciancio D. (eds) Advances in Cryptology – CRYPTO 2019. CRYPTO 2019. Lecture Notes in Computer Science, vol 11693. Springer, Cham. https://doi.org/10.1007/978-3-030-26951-7_22

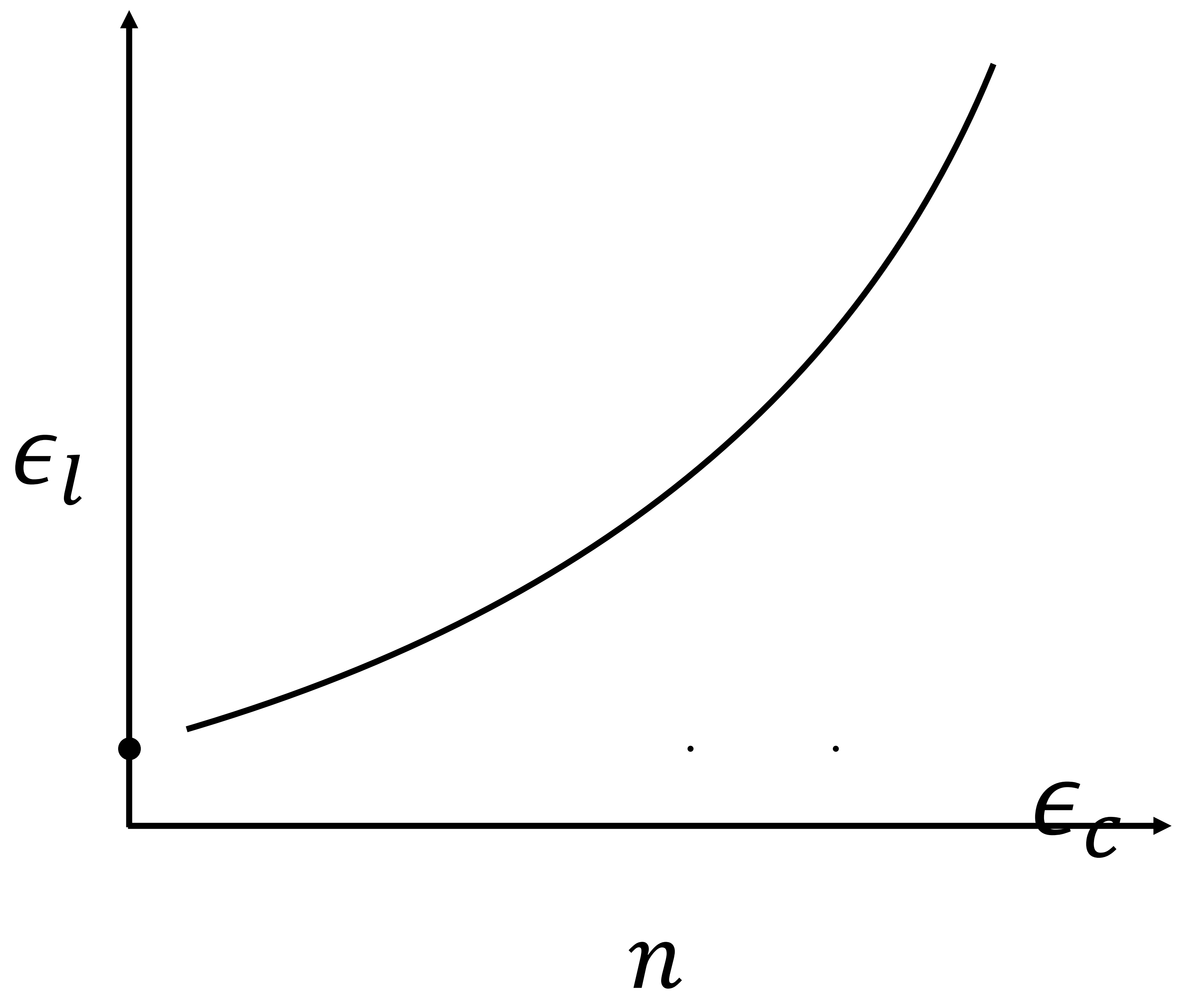
Better Trade-off in the Shuffle Model

Less noise due to the privacy amplification effect

- Demo task: n users, each holds a private value $x_i \in [0,1]$. Estimate $\sum_{i=1}^{n} x_i$

DP model	Local DP	Shuffle DP	Curator DP
Noise	$\Theta(n^{1/2})$	$\Theta(n^{1/6})$	$\Theta(1)$

[CRYPTO'19]



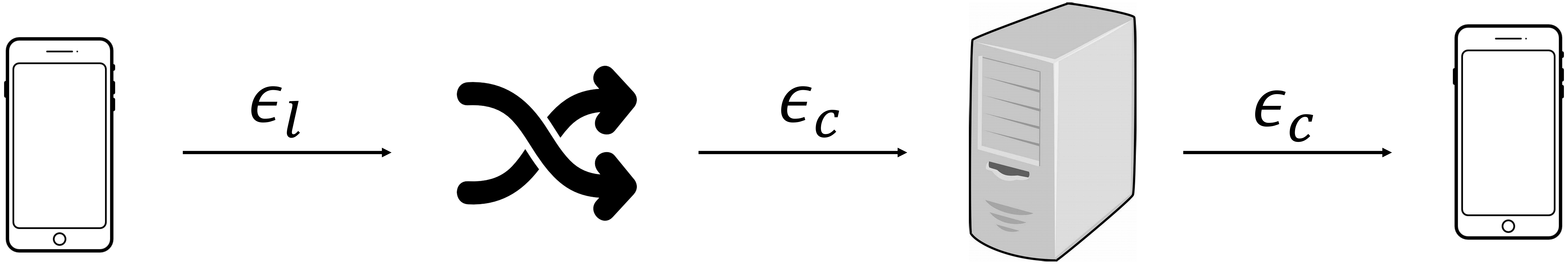
- Under a given central privacy budget ϵ_c, less local noises are required

Our Solution

FLAME = **F**ederated **L**e**a**rning in the Shuffle **M**od**e**l

Trust Model of FLAME

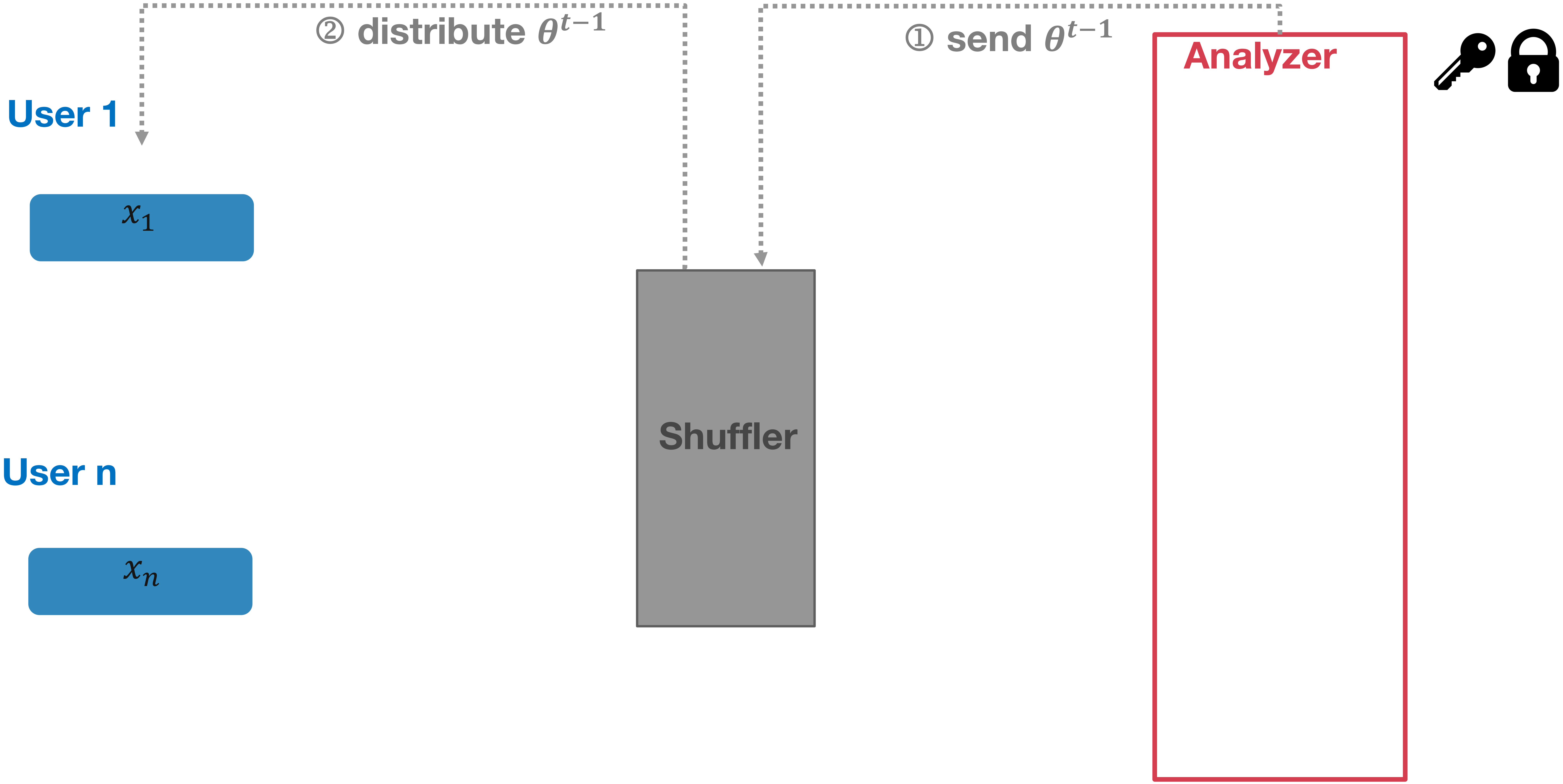
Separate trust on different parties



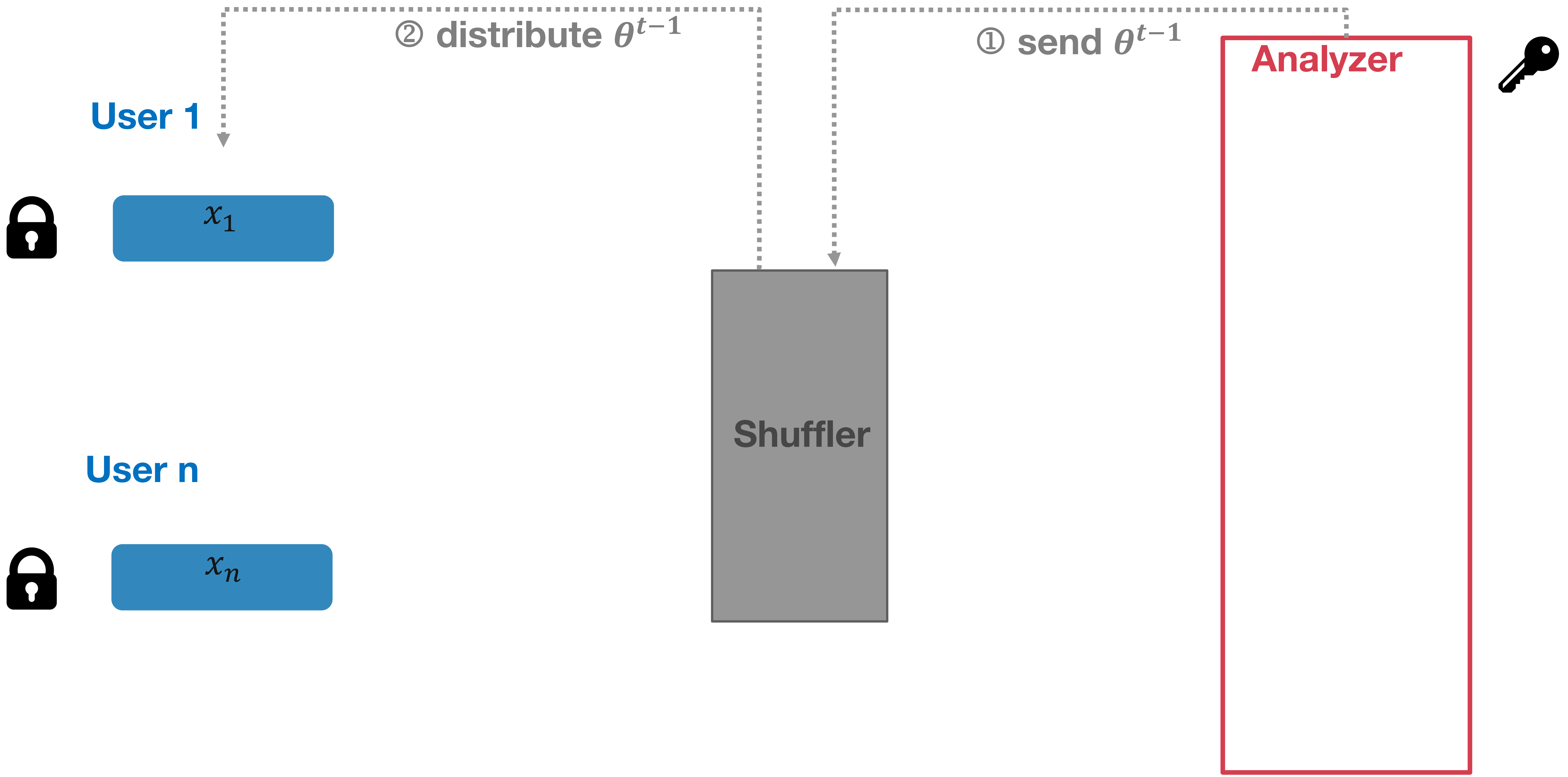
Parties	Shuffler S			Analyzer A			Observer $\mathcal{O}$
Sensitive info	gradient vector		ID	gradient vector		ID	query model
	index	value		index	value		
DP-FL	N/A			$\surd$		$\surd$	(ϵ_c, δ_c)-DP
LDP-FL				(ϵ_c, δ_c)-LDP		$\surd$	
FLAME	$\times$	$\times$	$\surd$	(ϵ_c, δ_c)-DP		$\times$	

$\surd$: trusted, $\times$: untrusted.

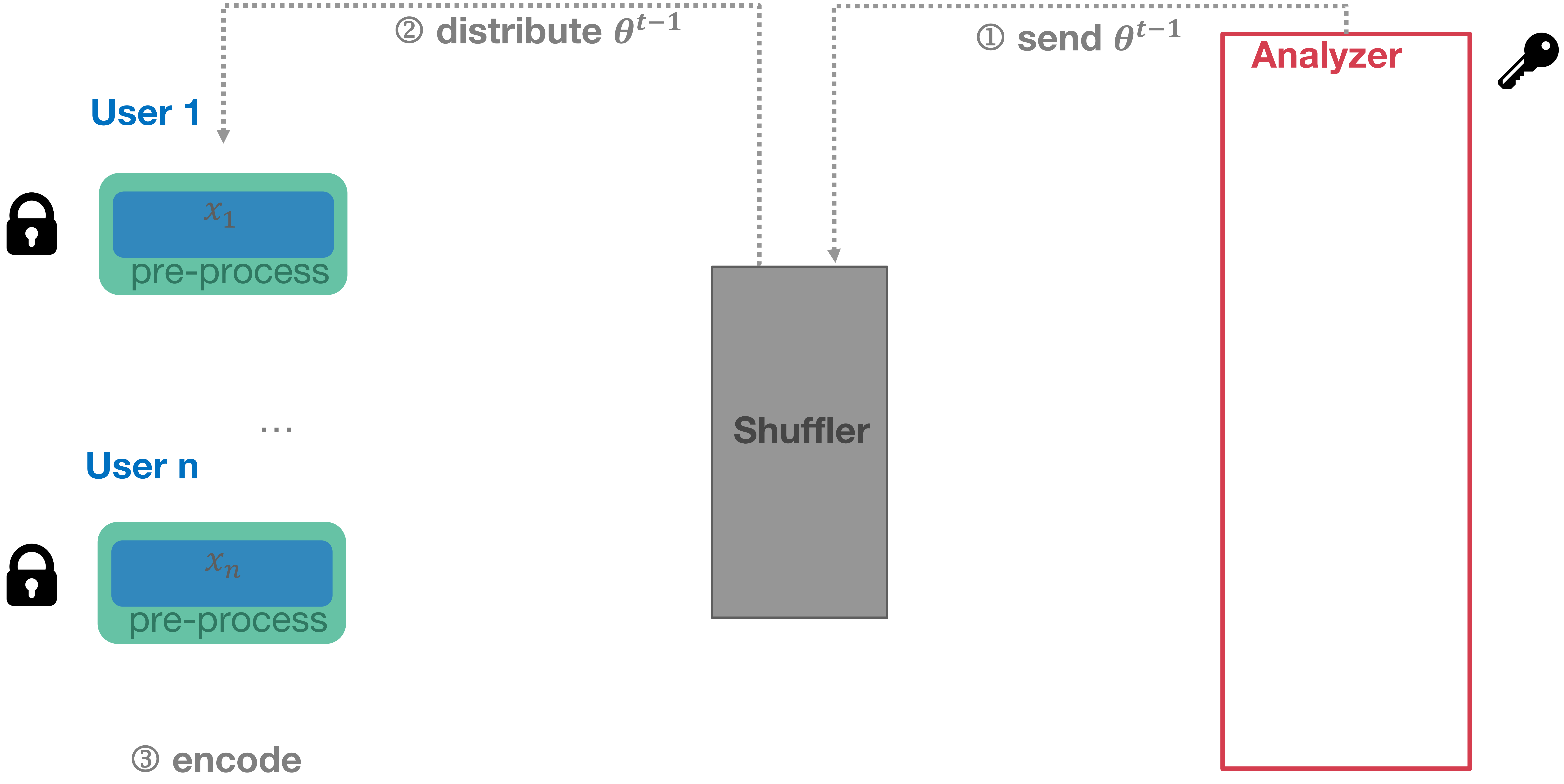
Framework Overview



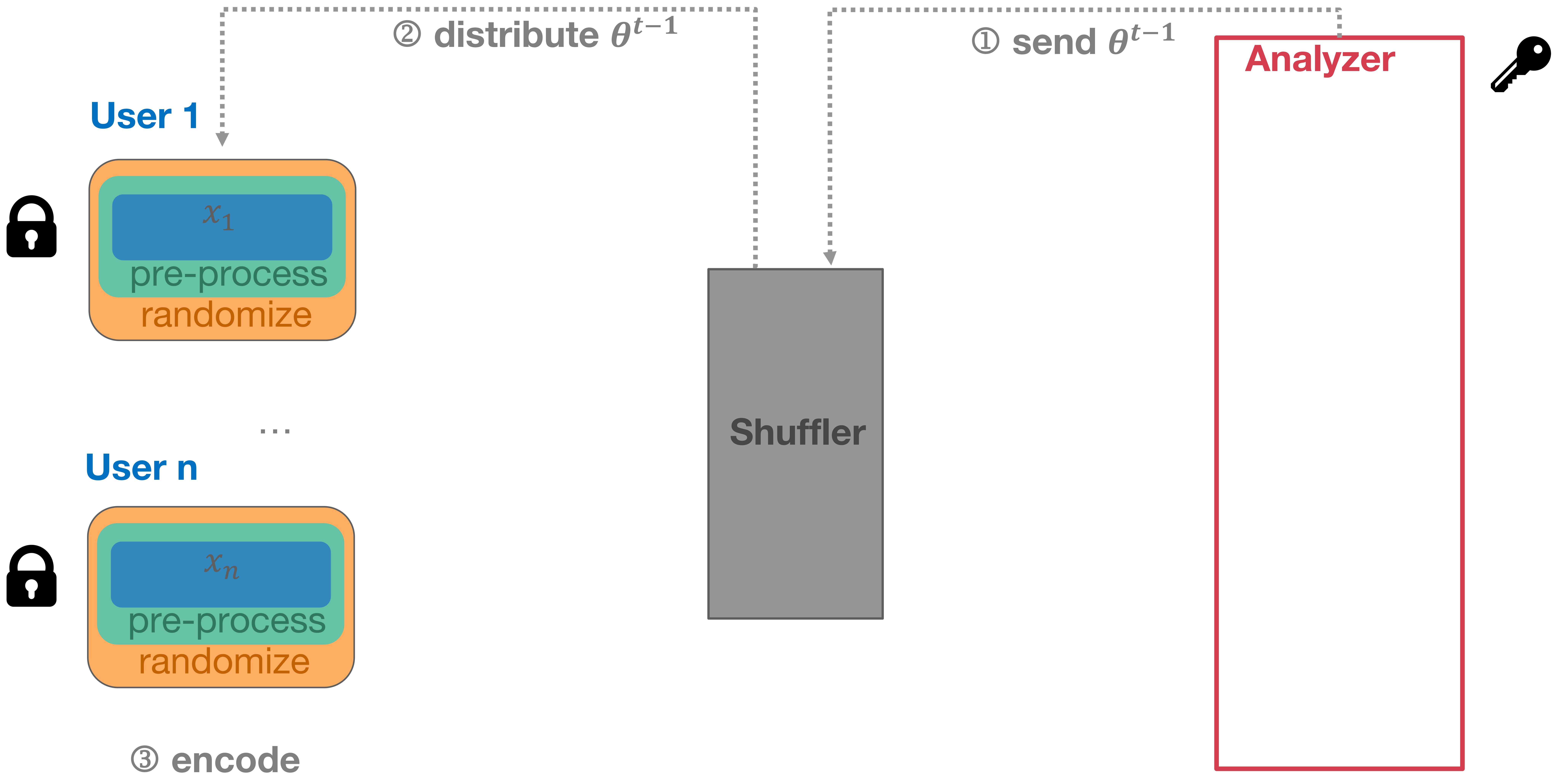
Framework Overview



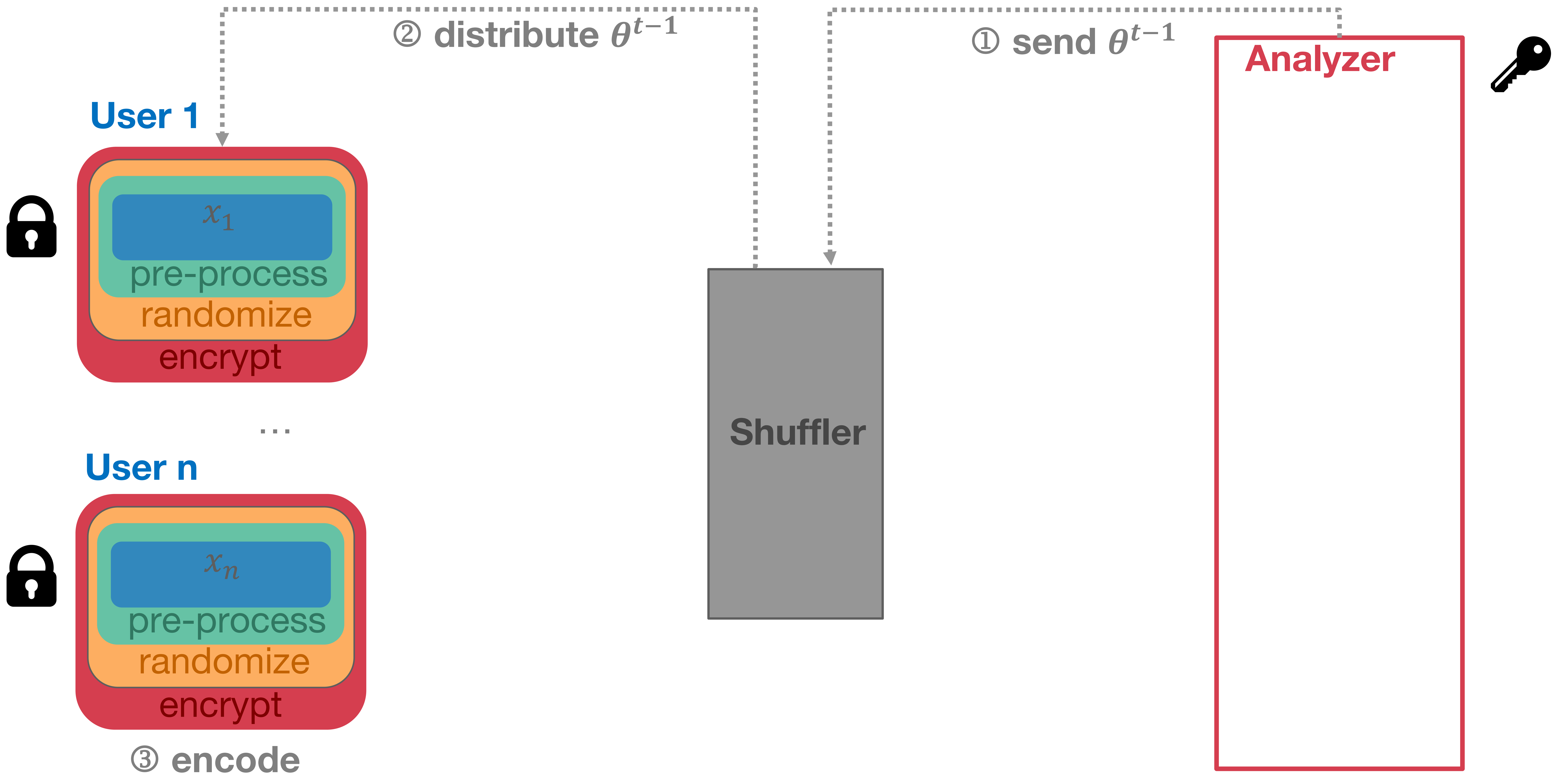
Framework Overview



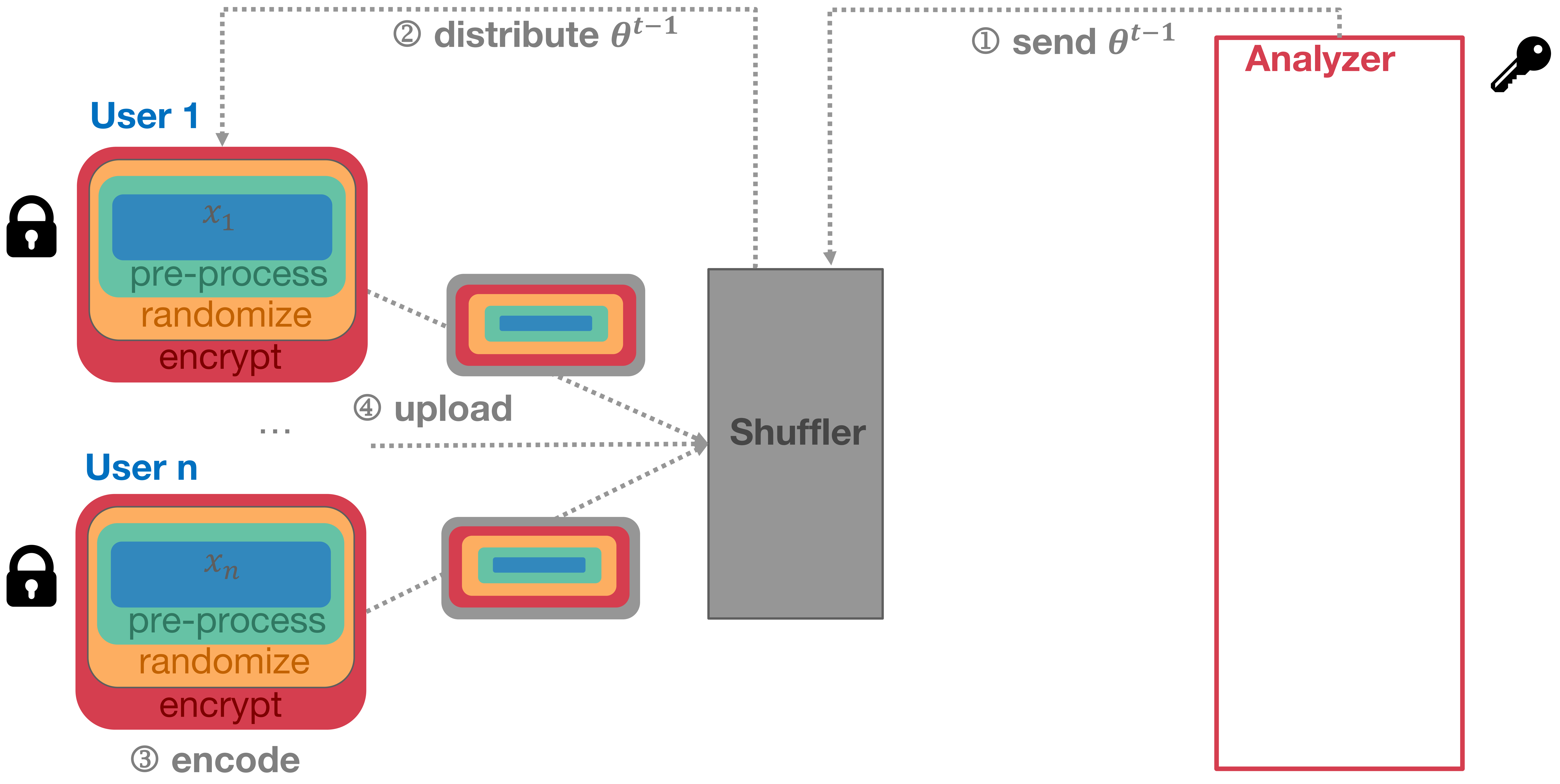
Framework Overview



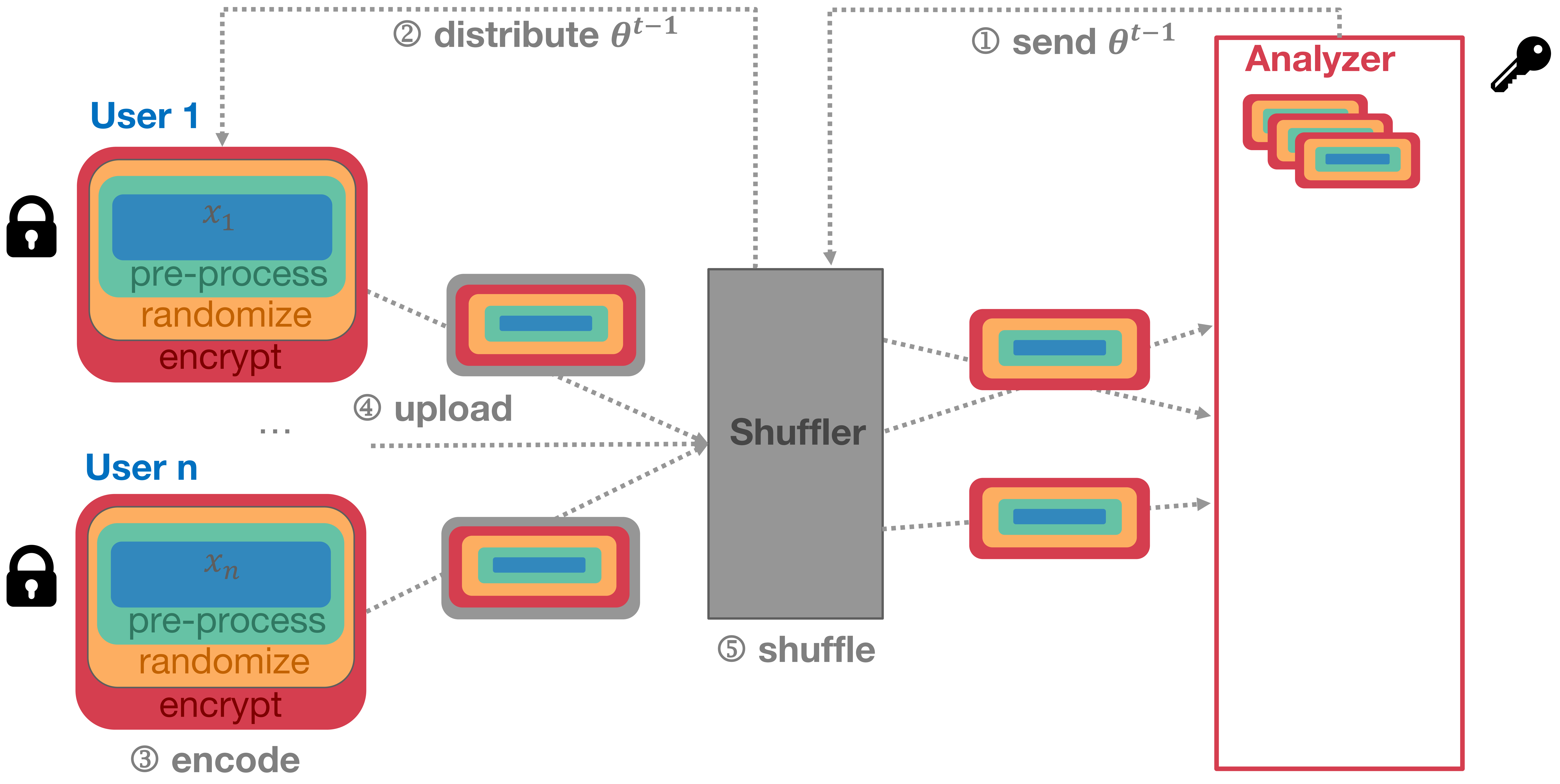
Framework Overview



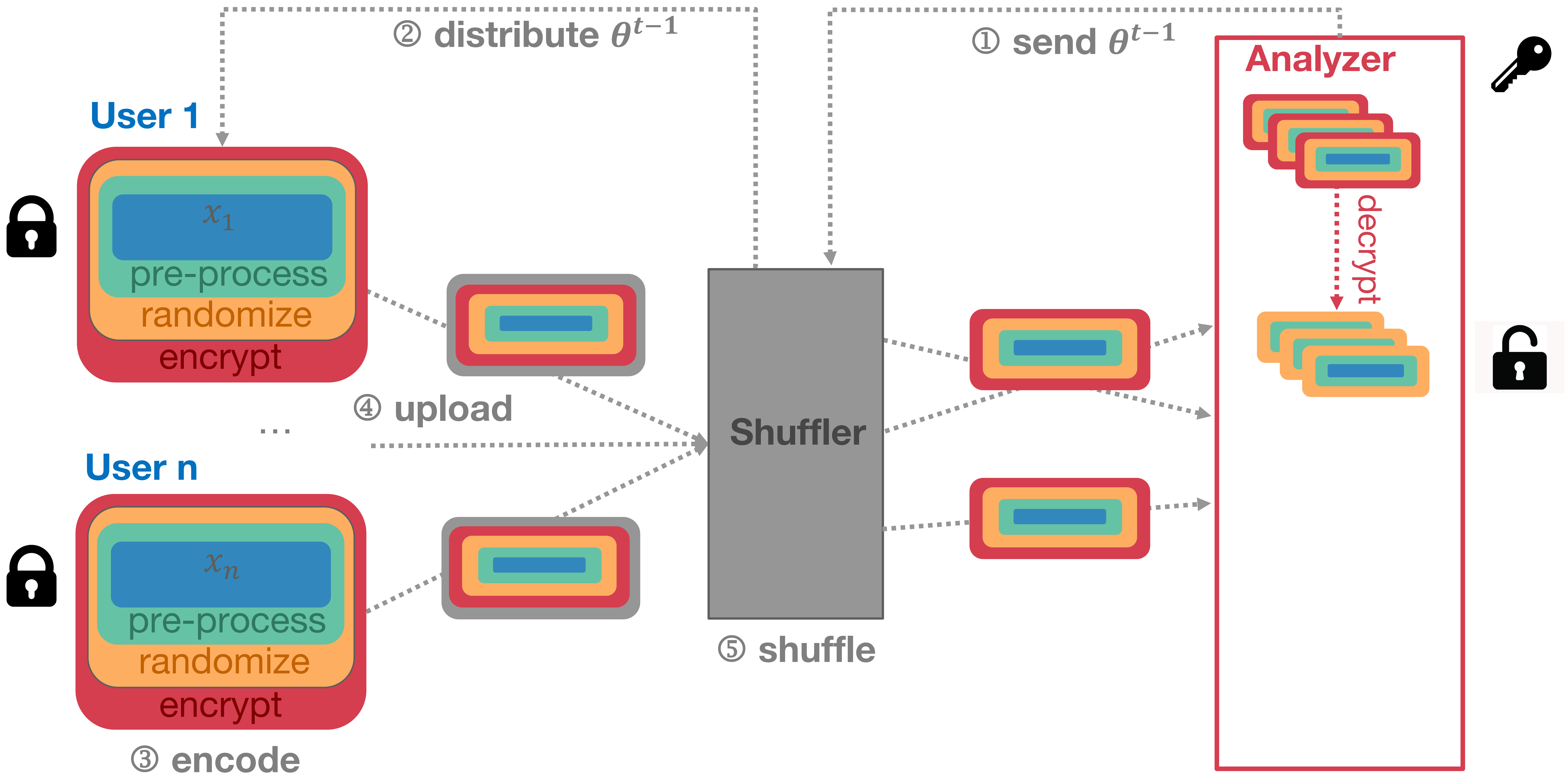
Framework Overview



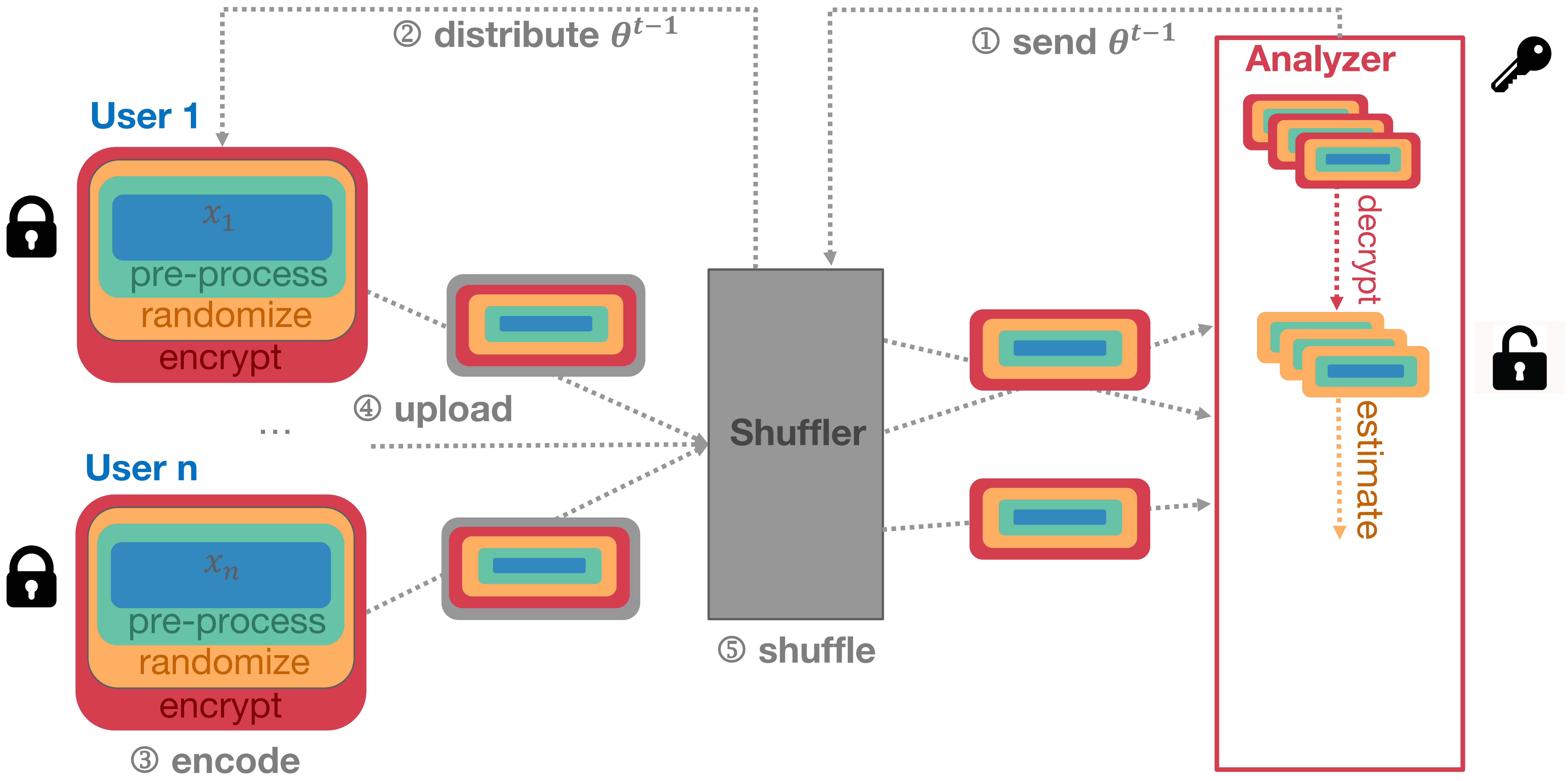
Framework Overview



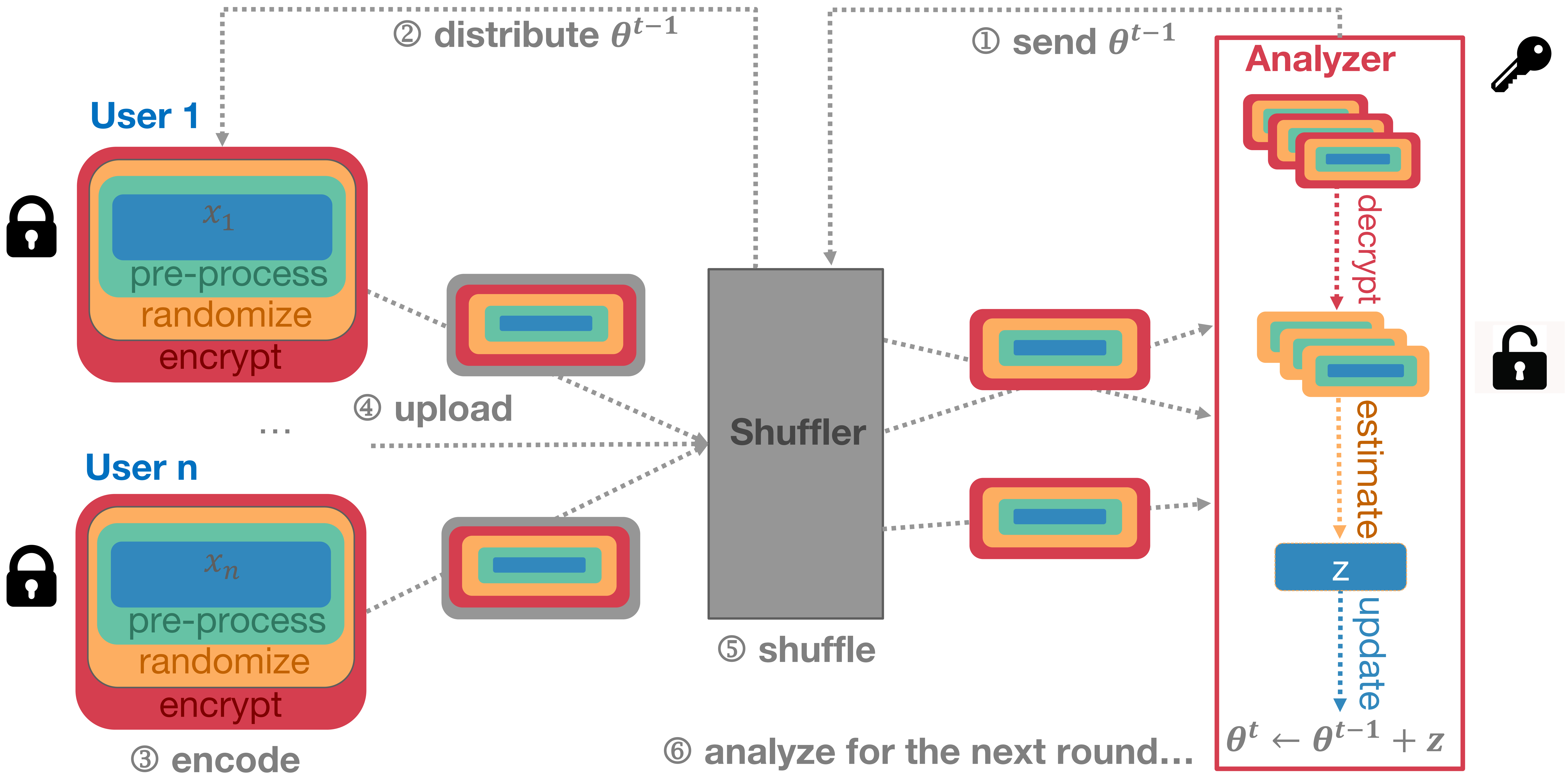
Framework Overview



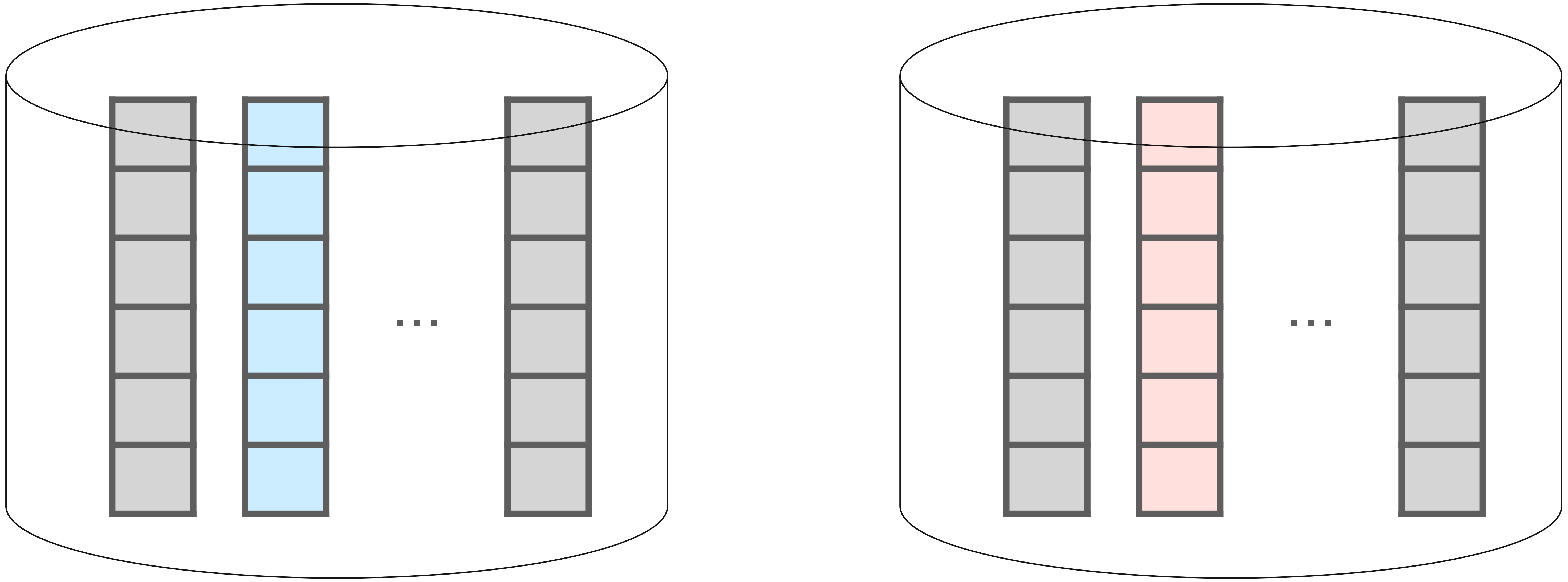
Framework Overview



Framework Overview



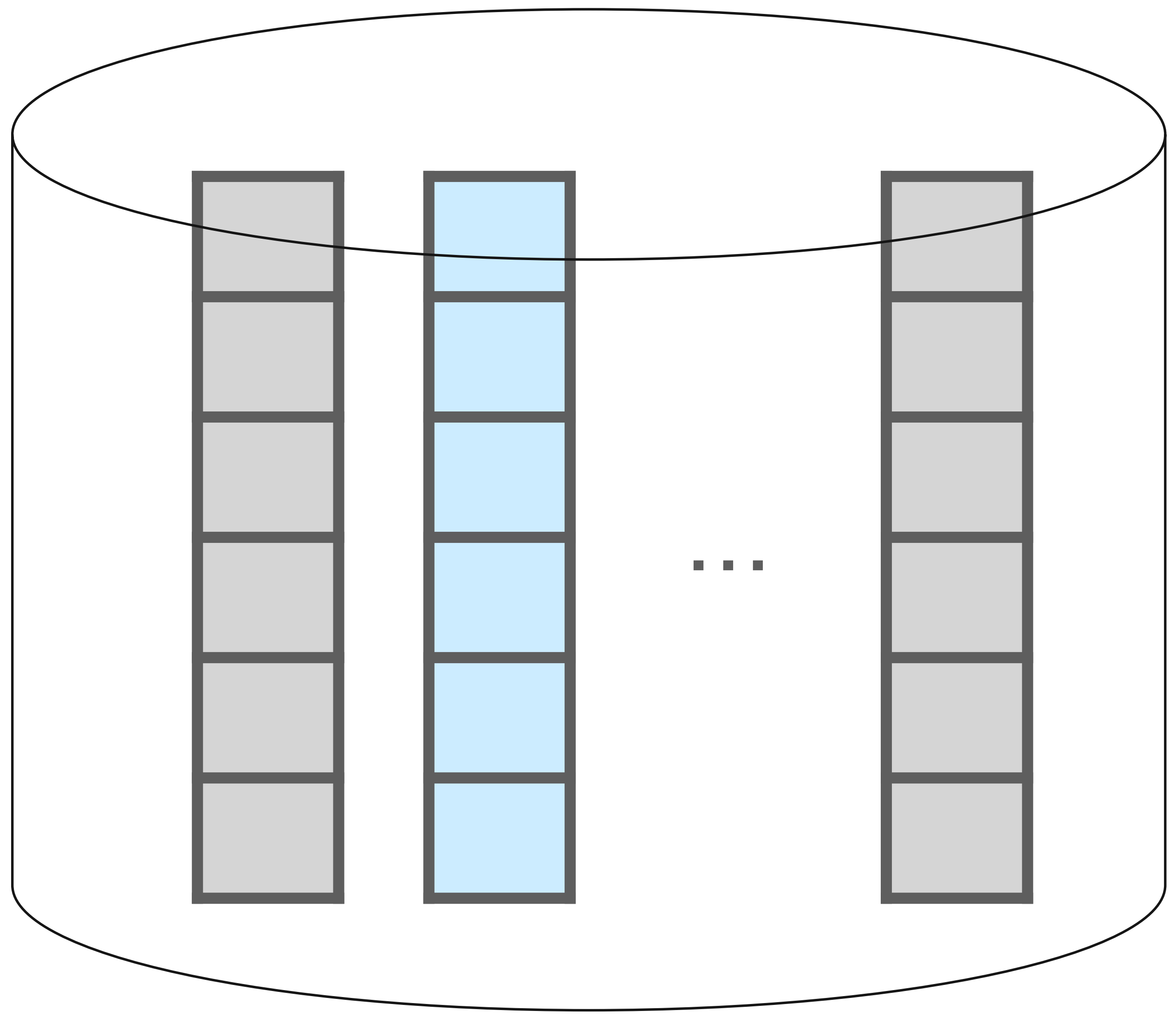
Privacy Definition



Neighboring datasets:
Any two datasets that differ by replacing one user's update

$$\Pr[M(X) \in S] \leq e^{\epsilon} \Pr[M(X') \in S] + \delta$$

A baseline solution: SS-Simple



for each user $i \in [n]$ **do**

$x_i \leftarrow \text{LocalUpdate}(\theta^{t-1})$

$\triangleright$ `Encoding` $\mathcal{E}$ `by each user`

$\bar{x}_i \leftarrow \text{Clip}(x_i, -C, C)$

$\tilde{x}_i \leftarrow (\bar{x}_i + C)/(2C)$

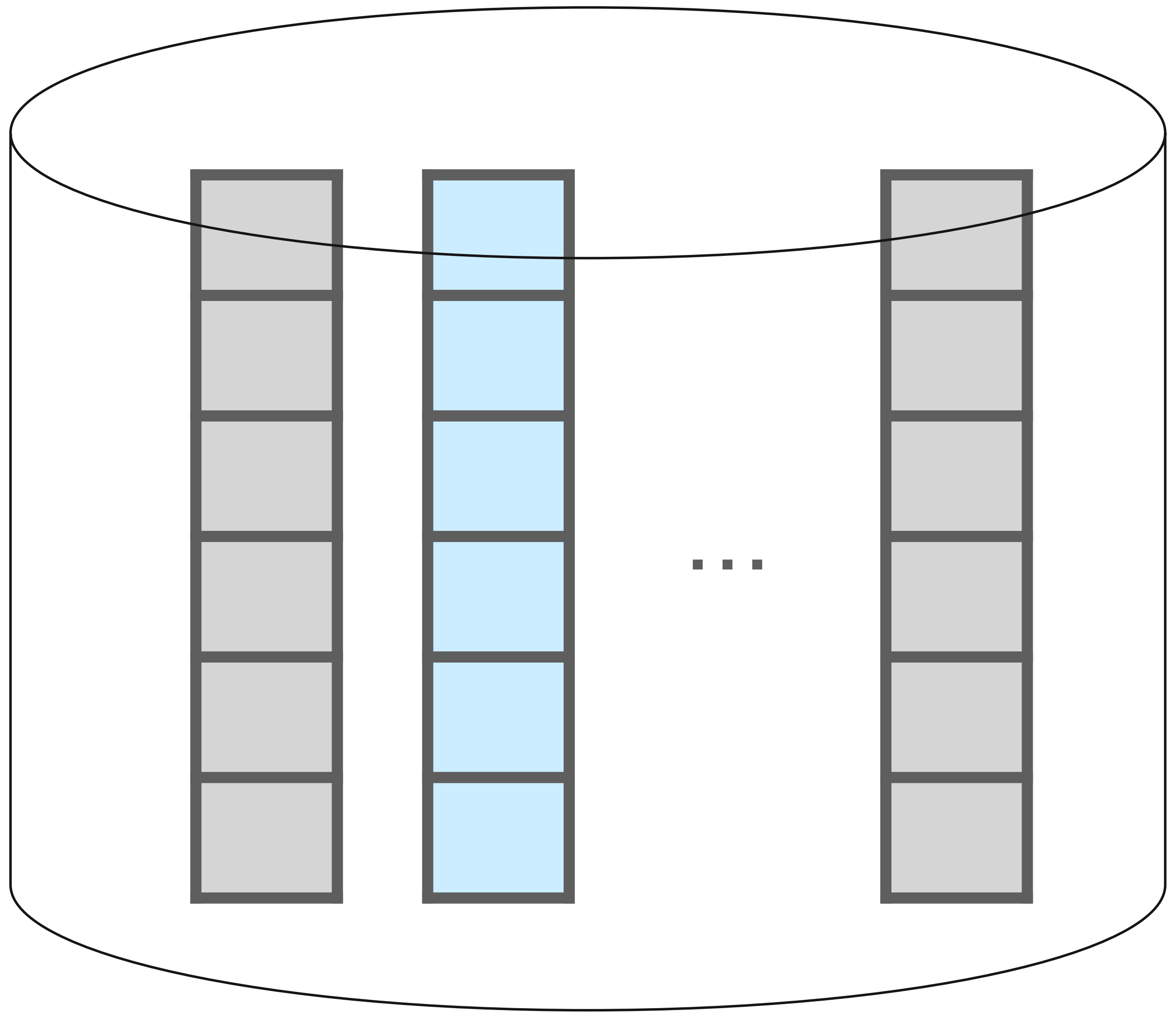
$\langle idx_i, y_i \rangle \leftarrow \mathsf{Randomize}(\tilde{x}_i, \epsilon_l)$

$c_i \leftarrow \text{Enc}_{pk_a}(y_i)$

user i sends $m_i = \langle idx_i, c_i \rangle$ to Shuffler

end for

A baseline solution: SS-Simple



for each user $i \in [n]$ **do**

$x_i \leftarrow \text{LocalUpdate}(\theta^{t-1})$

$\triangleright$ `Encoding` $\mathcal{E}$ `by each user`

$\bar{x}_i \leftarrow \text{Clip}(x_i, -C, C)$

$\tilde{x}_i \leftarrow (\bar{x}_i + C)/(2C)$

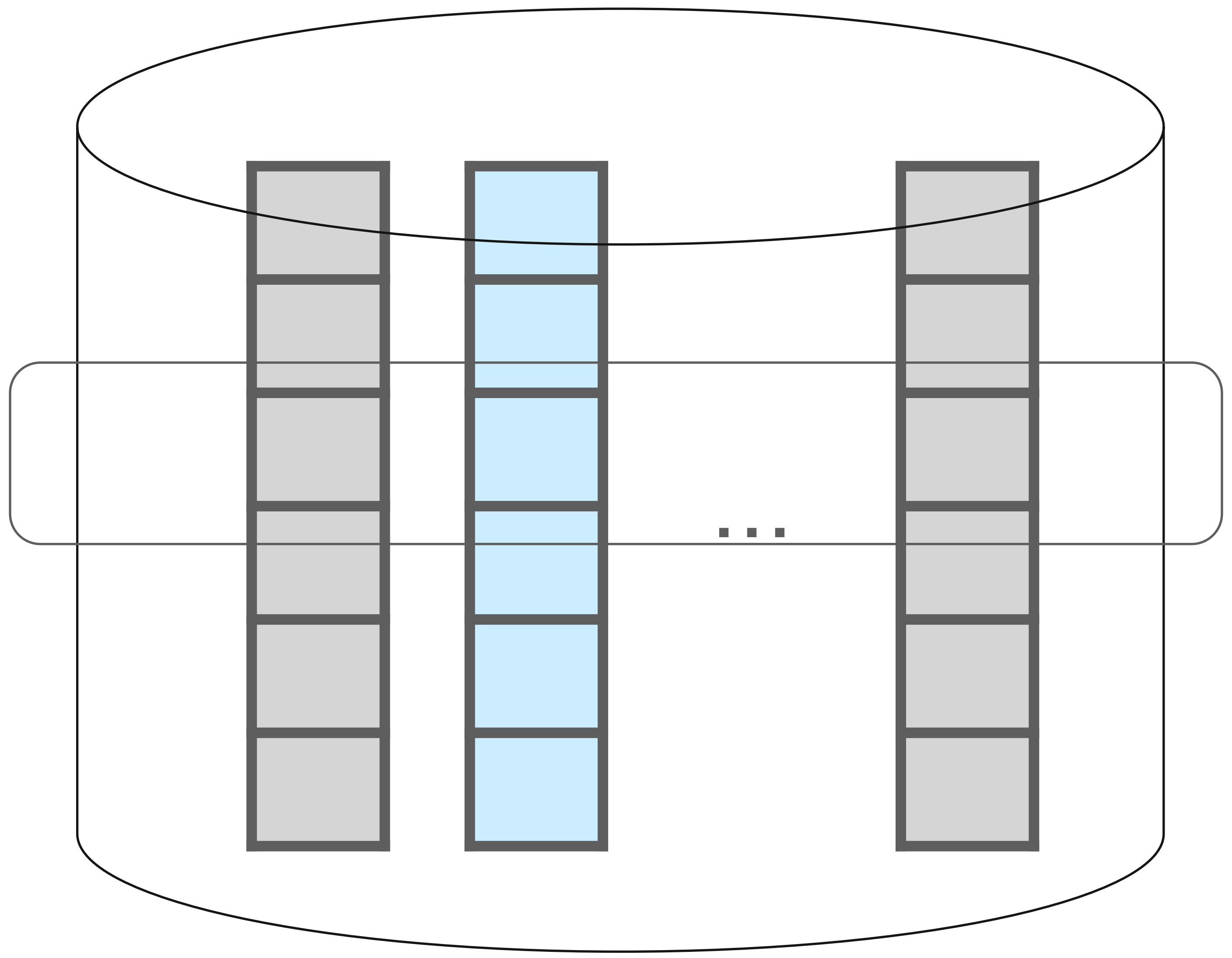
$\langle idx_i, y_i \rangle \leftarrow \text{Randomize}(\tilde{x}_i, \epsilon_l)$

$c_i \leftarrow \text{Enc}_{pk_a}(y_i)$

user i sends $m_i = \langle idx_i, c_i \rangle$ to Shuffler

end for

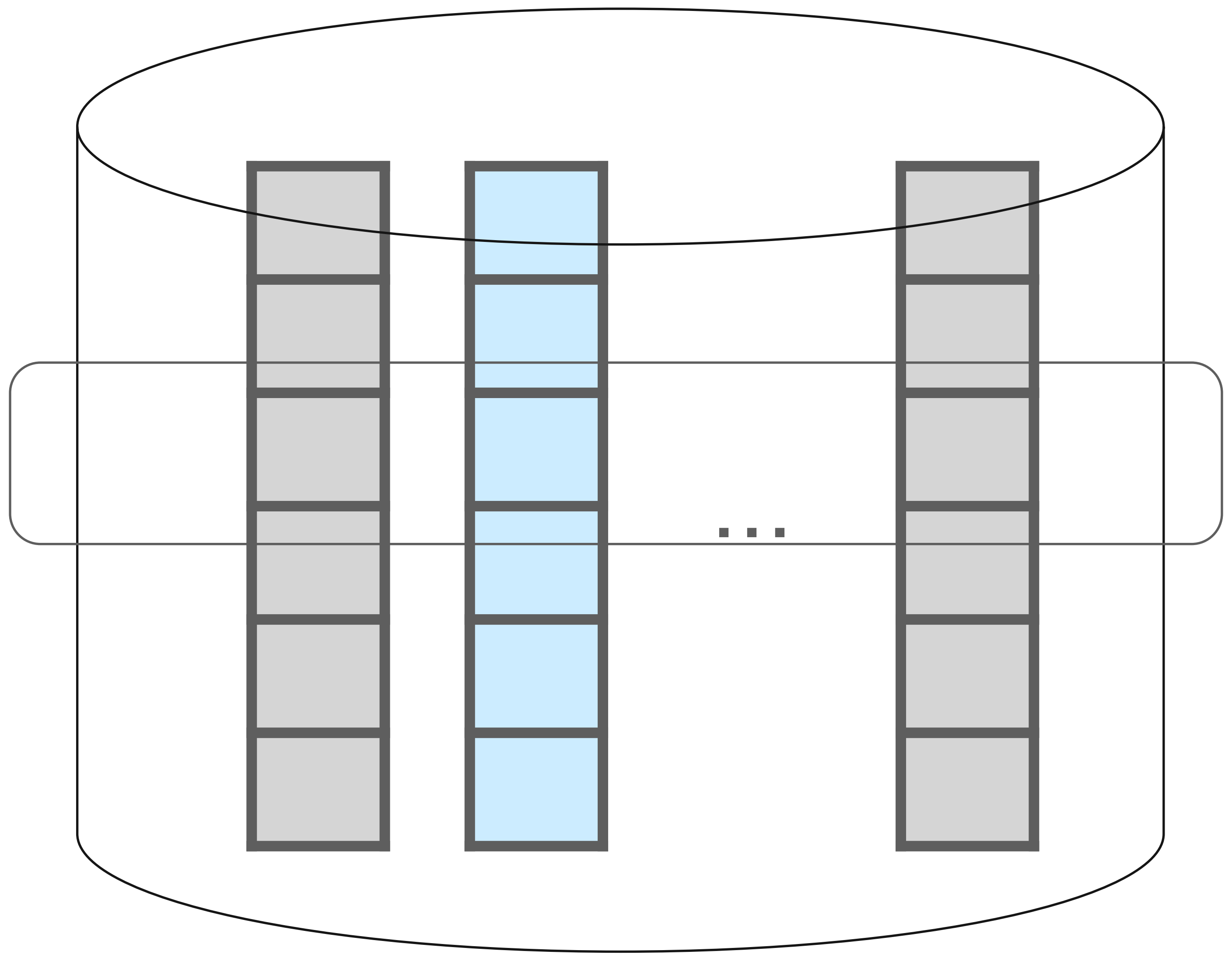
A baseline solution: SS-Simple



- Demo task: n users, each holds a private value $x_i \in [0,1]$. Estimate $\sum_{i=1}^{n} x_i$

for each user $i \in [n]$ **do**
$x_i \leftarrow \text{LocalUpdate}(\theta^{t-1})$
▷ `Encoding` $\mathcal{E}$ `by each user`
$\bar{x}_i \leftarrow \text{Clip}(x_i, -C, C)$
$\tilde{x}_i \leftarrow (\bar{x}_i + C)/(2C)$
$\langle idx_i, y_i \rangle \leftarrow$ Randomize$(\tilde{x}_i, \epsilon_l)$
$c_i \leftarrow \text{Enc}_{pk_a}(y_i)$
user i sends $m_i = \langle idx_i, c_i \rangle$ to Shuffler
end for

A baseline solution: SS-Simple

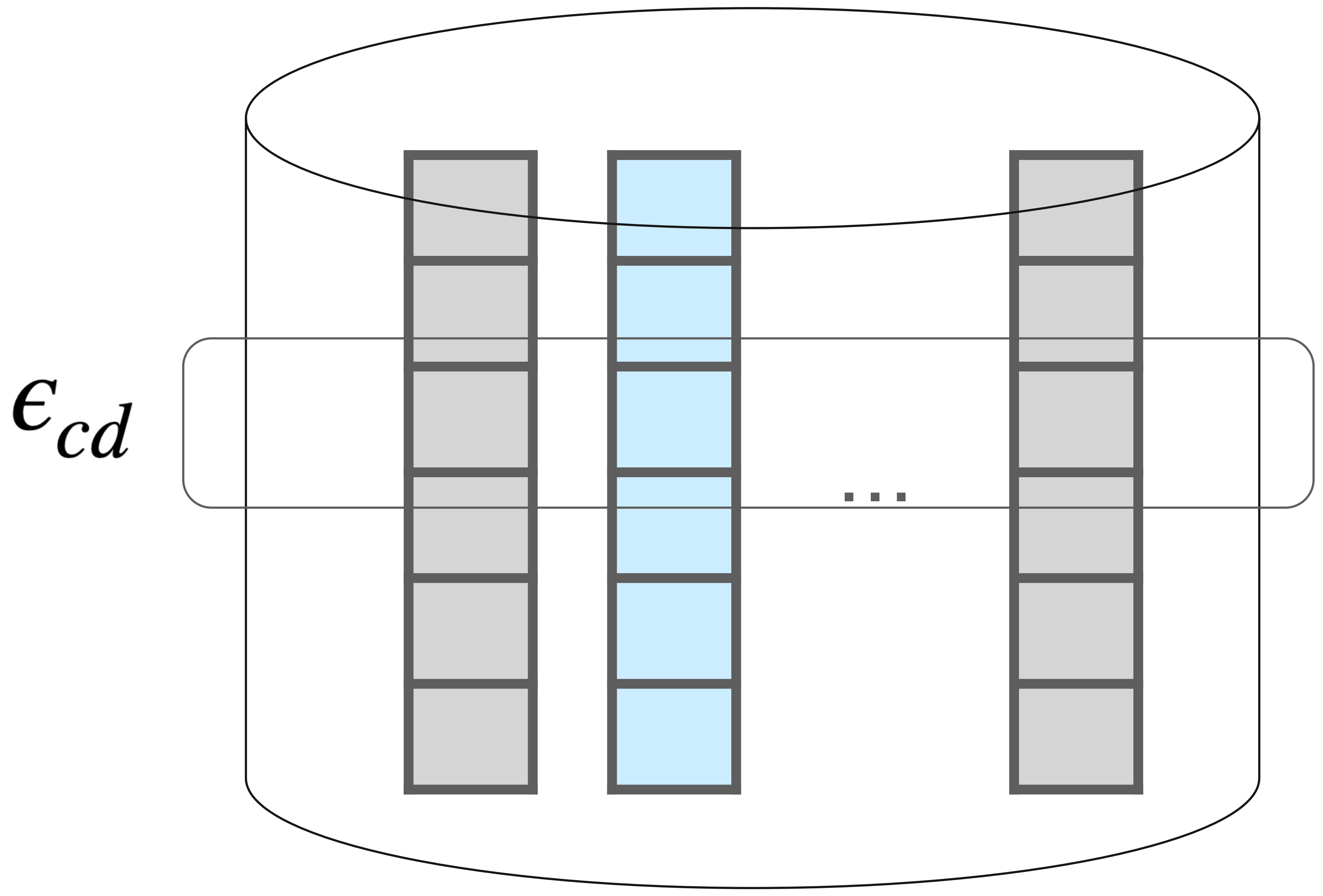


$\epsilon_l \rightarrow_d \epsilon_{ld} = \epsilon_l / d$

- Demo task: n users, each holds a private value $x_i \in [0,1]$. Estimate $\sum_{i=1}^{n} x_i$

for each user $i \in [n]$ **do**
  $x_i \leftarrow \text{LocalUpdate}(\theta^{t-1})$
  ▷ `Encoding` $\mathcal{E}$ `by each user`
  $\bar{x}_i \leftarrow \text{Clip}(x_i, -C, C)$
  $\tilde{x}_i \leftarrow (\bar{x}_i + C)/(2C)$
  $\langle idx_i, y_i \rangle \leftarrow$ Randomize$(\tilde{x}_i, \epsilon_l)$
  $c_i \leftarrow \text{Enc}_{pk_a}(y_i)$
  user i sends $m_i = \langle idx_i, c_i \rangle$ to Shuffler
end for

A baseline solution: SS-Simple



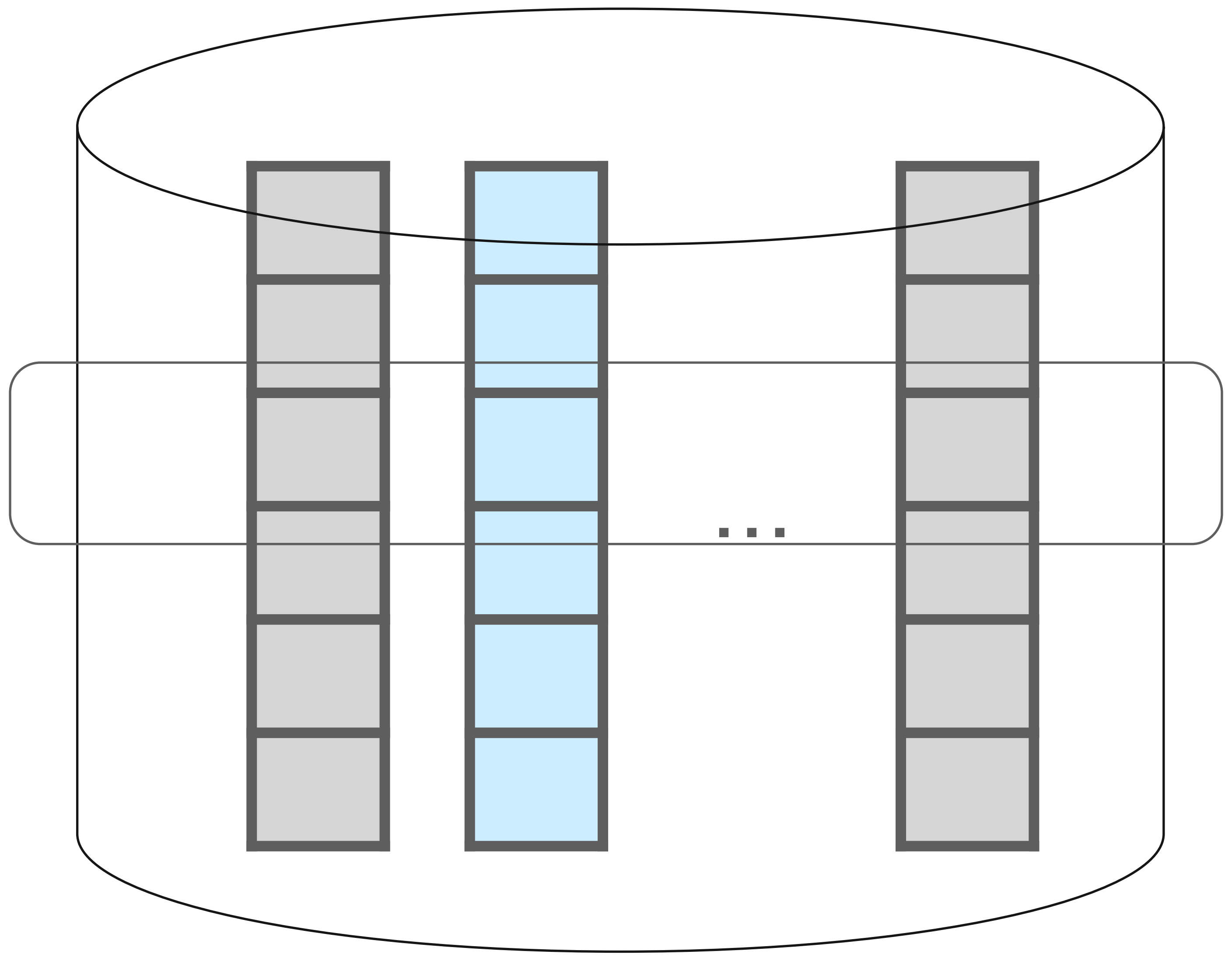
$\epsilon_l \rightarrow_d \epsilon_{ld} = \epsilon_l / d \rightarrow_s \epsilon_{cd}$

- Demo task: n users, each holds a private value $x_i \in [0,1]$. Estimate $\sum_{i=1}^{n} x_i$

for each user $i \in [n]$ **do**
  $x_i \leftarrow \text{LocalUpdate}(\theta^{t-1})$
  $\triangleright$ `Encoding` $\mathcal{E}$ `by each user`
  $\bar{x}_i \leftarrow \text{Clip}(x_i, -C, C)$
  $\tilde{x}_i \leftarrow (\bar{x}_i + C)/(2C)$
  $\langle idx_i, y_i \rangle \leftarrow$ Randomize$(\tilde{x}_i, \epsilon_l)$
  $c_i \leftarrow \text{Enc}_{pk_a}(y_i)$
  user i sends $m_i = \langle idx_i, c_i \rangle$ to Shuffler
end for

A baseline solution: SS-Simple

ϵ_c

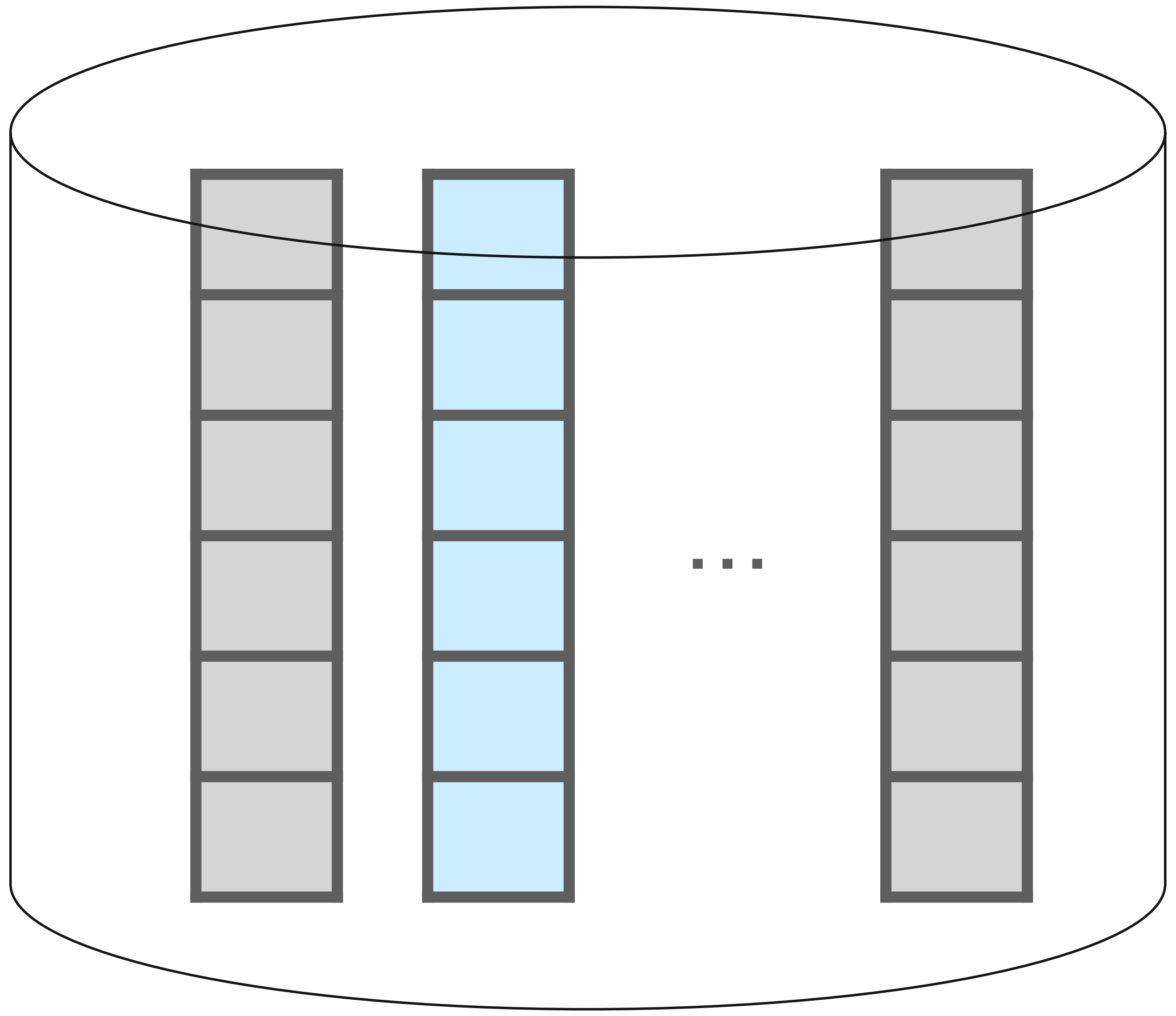


$$\epsilon_l \rightarrow_d \epsilon_{ld} = \epsilon_l / d \rightarrow_s \epsilon_{cd} \rightarrow_c \epsilon_c$$

- Demo task: n users, each holds a private value $x_i \in [0,1]$. Estimate $\sum_{i=1}^{n} x_i$

for each user $i \in [n]$ **do**
  $x_i \leftarrow \text{LocalUpdate}(\theta^{t-1})$
  $\triangleright$ `Encoding` $\mathcal{E}$ `by each user`
  $\bar{x}_i \leftarrow \text{Clip}(x_i, -C, C)$
  $\tilde{x}_i \leftarrow (\bar{x}_i + C)/(2C)$
  $\langle idx_i, y_i \rangle \leftarrow \text{Randomize}(\tilde{x}_i, \epsilon_l)$
  $c_i \leftarrow \text{Enc}_{pk_a}(y_i)$
  user i sends $m_i = \langle idx_i, c_i \rangle$ to Shuffler
end for

A baseline solution: SS-Simple

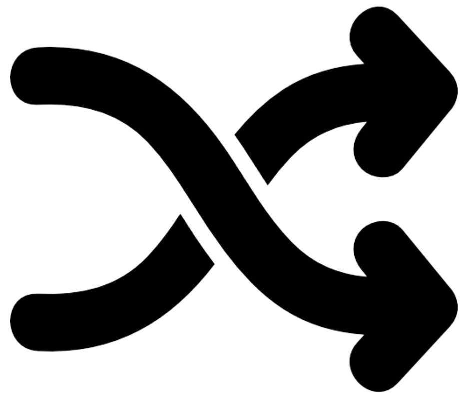


idx$_i \leftarrow \{1, \cdots, d\}$

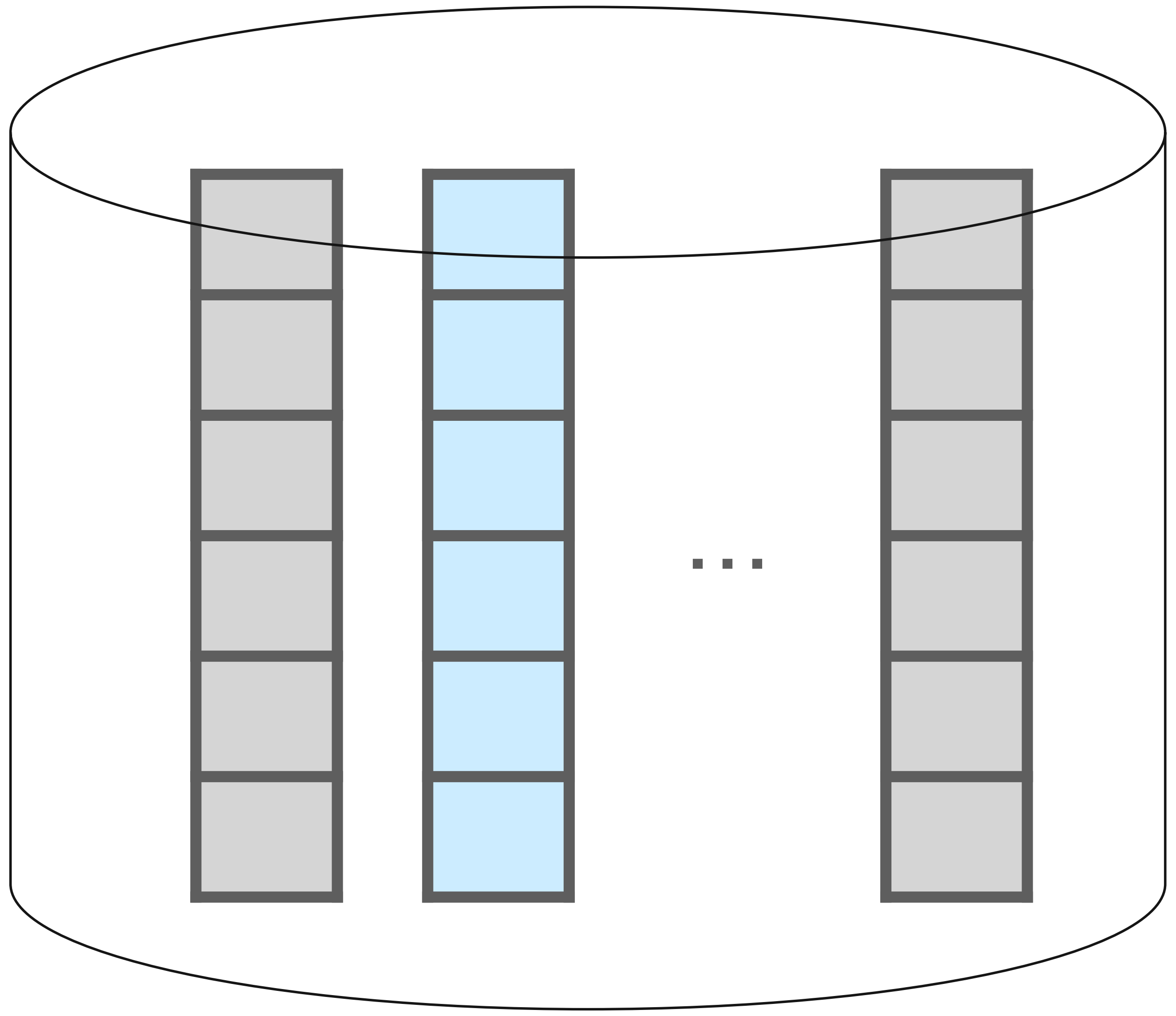
$y_i \leftarrow \{R_{\epsilon_{ld}}(x_{i,1}), \cdots, R_{\epsilon_{ld}}(x_{i,d})\}$

for each user $i \in [n]$ **do**
  $x_i \leftarrow \text{LocalUpdate}(\theta^{t-1})$
  ▷ Encoding $\mathcal{E}$ by each user
  $\bar{x}_i \leftarrow \text{Clip}(x_i, -C, C)$
  $\tilde{x}_i \leftarrow (\bar{x}_i + C)/(2C)$
  $\langle idx_i, y_i \rangle \leftarrow$ Randomize$(\tilde{x}_i, \epsilon_l)$
  $c_i \leftarrow \text{Enc}_{pk_a}(y_i)$
  user i sends $m_i = \langle idx_i, c_i \rangle$ to Shuffler
end for

A baseline solution: SS-Simple



- Learns nothing from the plaintext of index (full index list is not sensitive)
- Learns nothing from the encrypted values (does not have the key to decrypt)

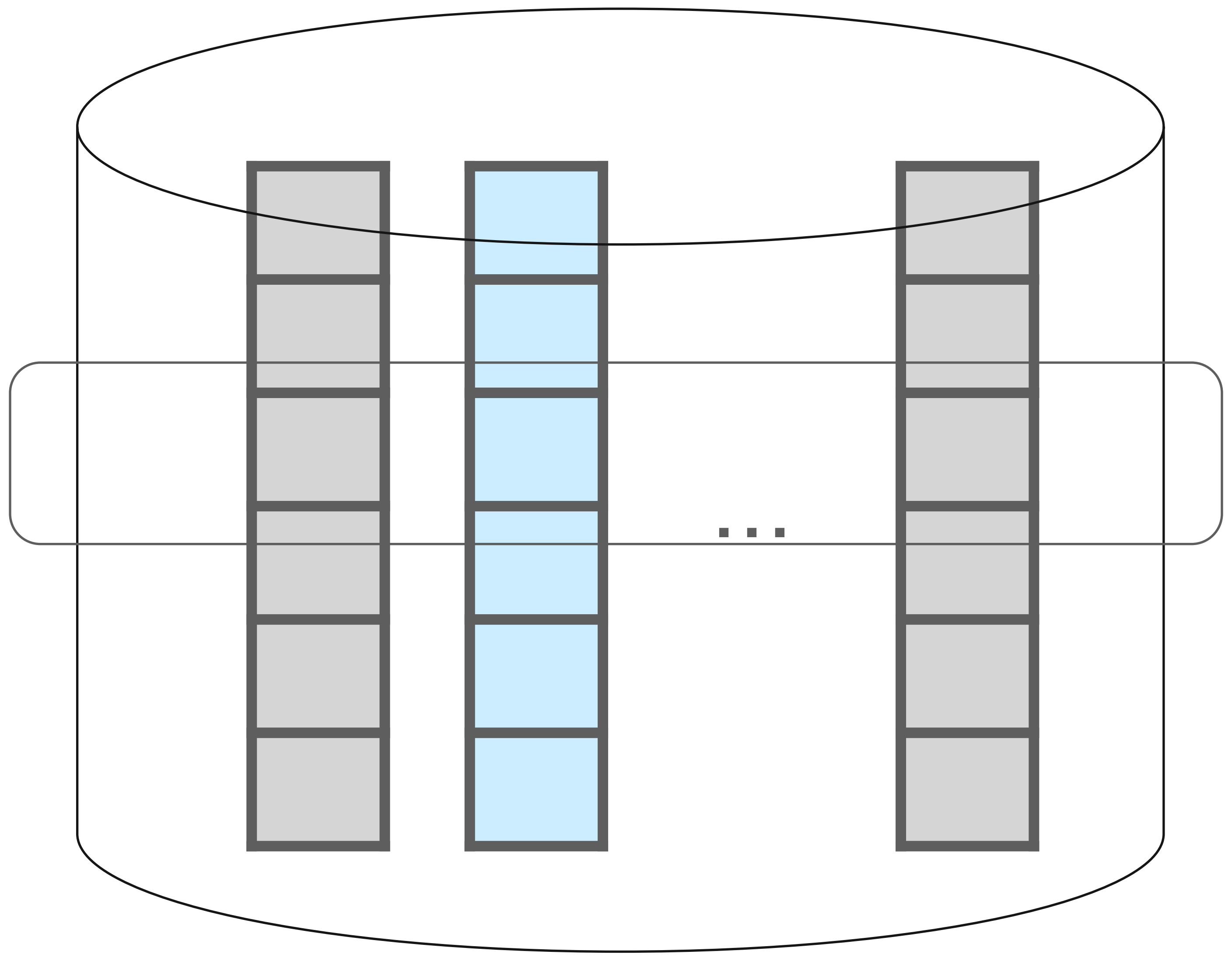


$\text{idx}_i \leftarrow \{1, \cdots, d\}$

$y_i \leftarrow \{R_{\epsilon_{ld}}(x_{i,1}), \cdots, R_{\epsilon_{ld}}(x_{i,d})\}$

for each user $i \in [n]$ **do**
  $x_i \leftarrow \text{LocalUpdate}(\theta^{t-1})$
  ▷ `Encoding` $\mathcal{E}$ `by each user`
  $\bar{x}_i \leftarrow \text{Clip}(x_i, -C, C)$
  $\tilde{x}_i \leftarrow (\bar{x}_i + C)/(2C)$
  $\langle idx_i, y_i \rangle \leftarrow \text{Randomize}(\tilde{x}_i, \epsilon_l)$
  $c_i \leftarrow \text{Enc}_{pk_a}(y_i)$
  user i sends $m_i = \langle idx_i, c_i \rangle$ to Shuffler
end for

Problem of SS-Simple



- Demo task: n users, each holds a private value $x_i \in [0,1]$. Estimate $\sum_{i=1}^{n} x_i$

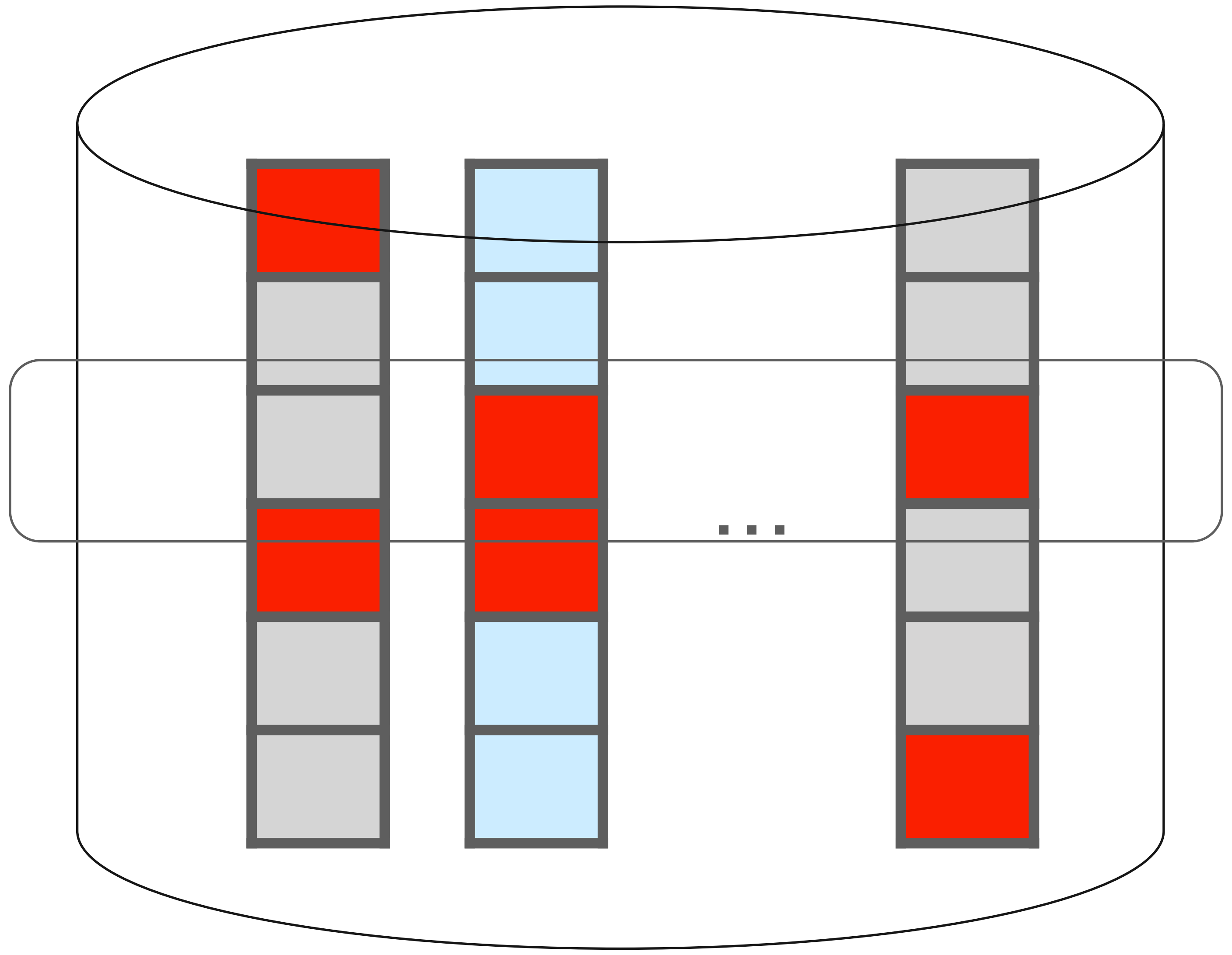
$$\epsilon_l \rightarrow_d \epsilon_{ld} = \boxed{\epsilon_l/d}$$

- Problem

  small budget (large noise) for each value

Problem of SS-Simple

- Demo task: n users, each holds a private value $x_i \in [0,1]$. Estimate $\sum_{i=1}^{n} x_i$



$$\epsilon_l \rightarrow_d \epsilon_{ld} = \boxed{\epsilon_l / d}$$

- Problem

  small budget (large noise) for each value

- A typical way for perturbing multi-dimensional vector

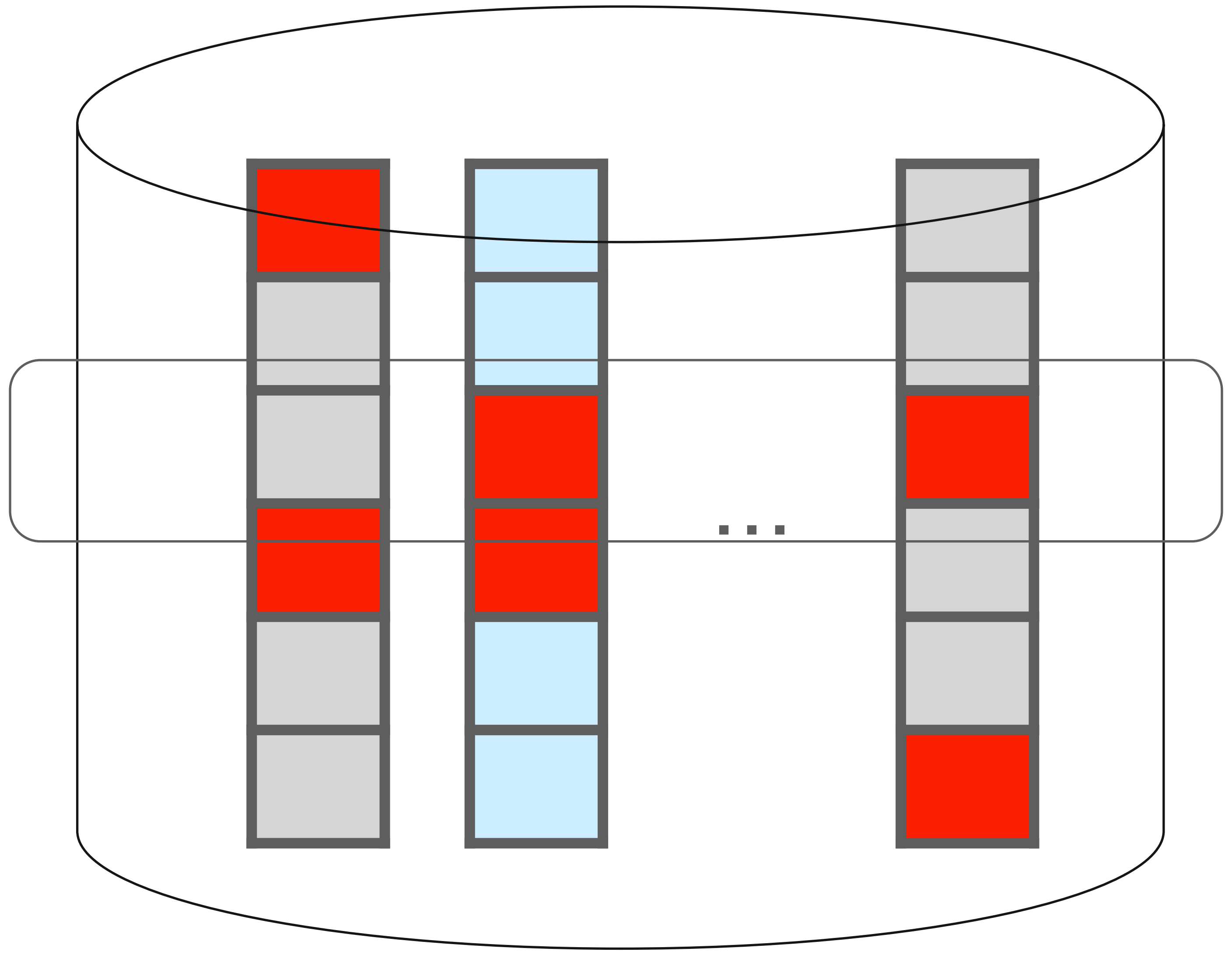
  sample and perturb a fraction of dimensions

$$O(d) \rightarrow O(\sqrt{d})$$

[ICDE'2019]

Double amplification solution: SS-Double

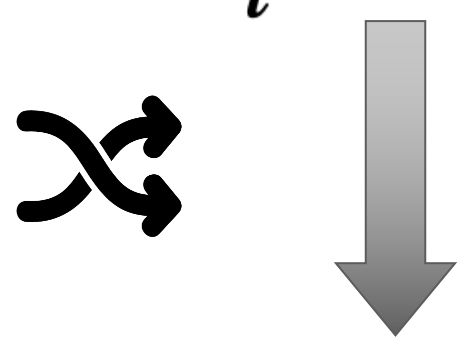
$$\epsilon_l \rightarrow \epsilon_c = \epsilon_l$$



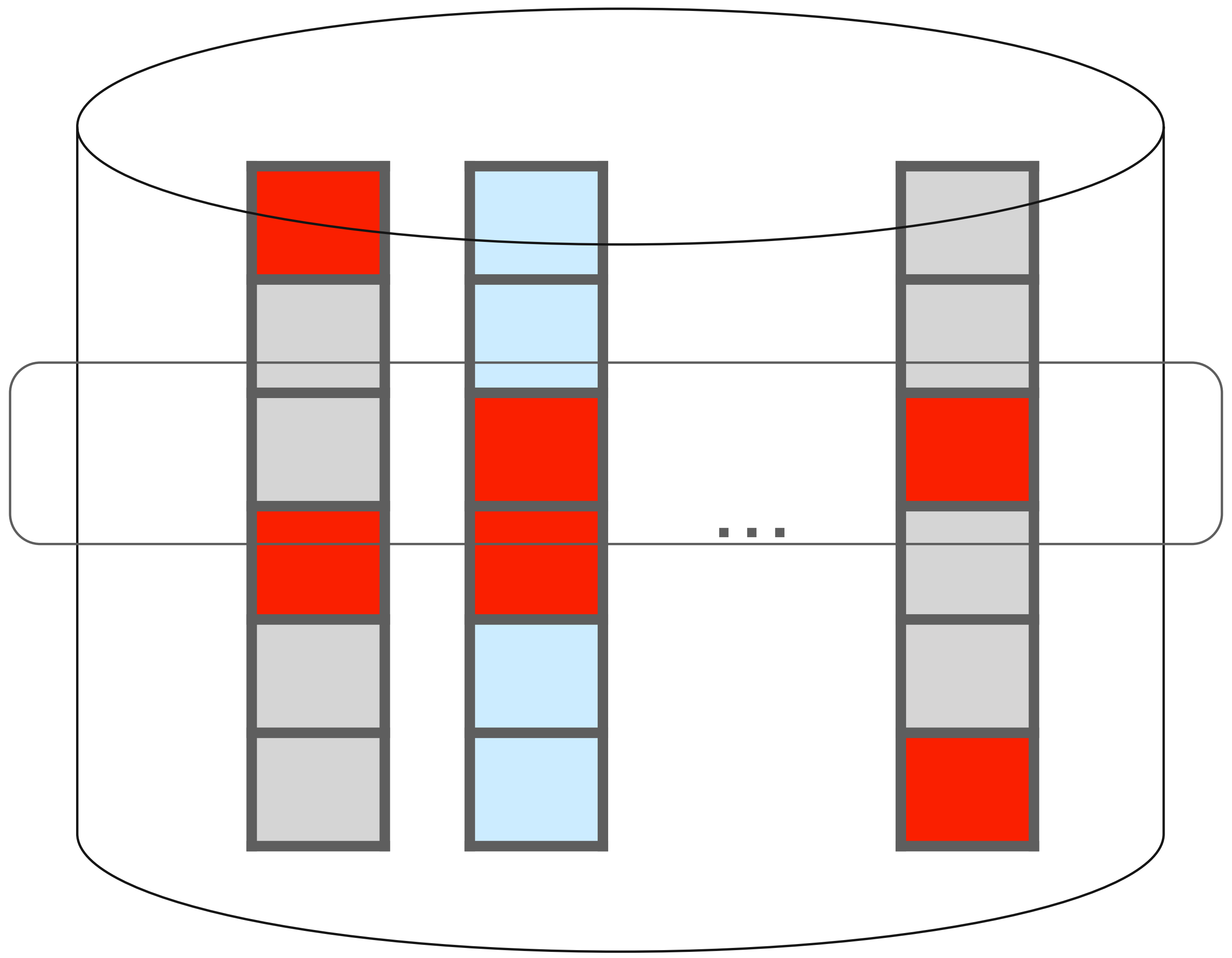
DP naturally holds for LDP

Double amplification solution: SS-Double

$$\epsilon_l \to \epsilon_c = \epsilon_l$$



$$\epsilon_l \to_d \epsilon_{ld} = \epsilon_l/d \to_s \epsilon_{cd} \to_c \epsilon_c$$

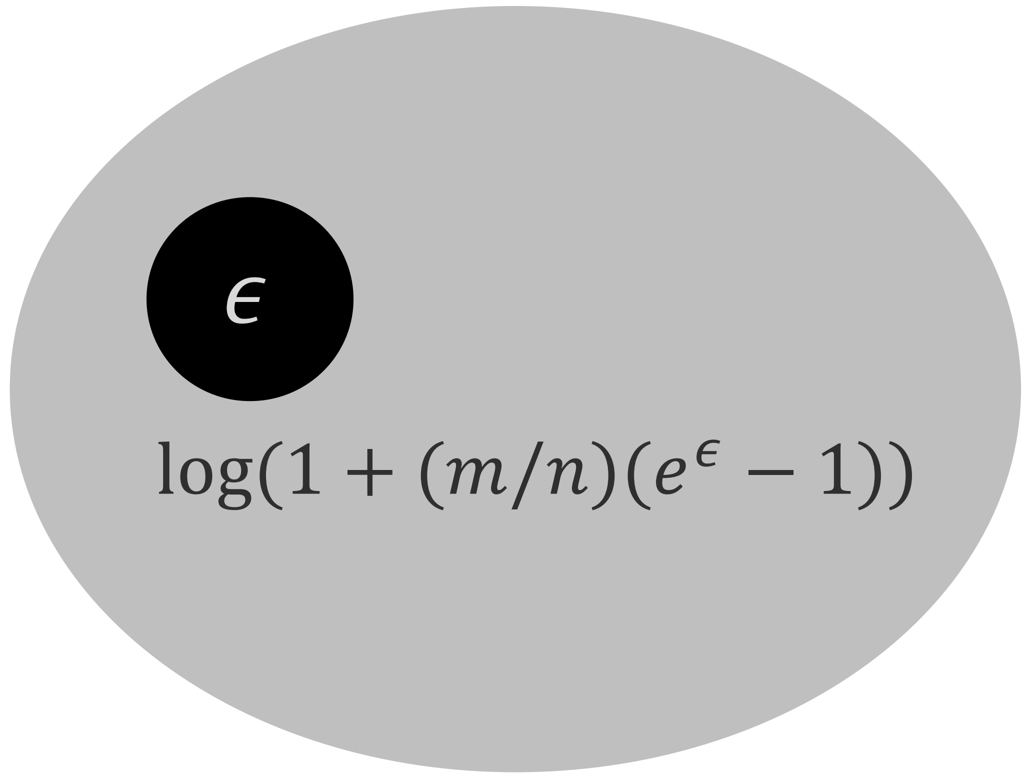


Privacy amplification by shuffling

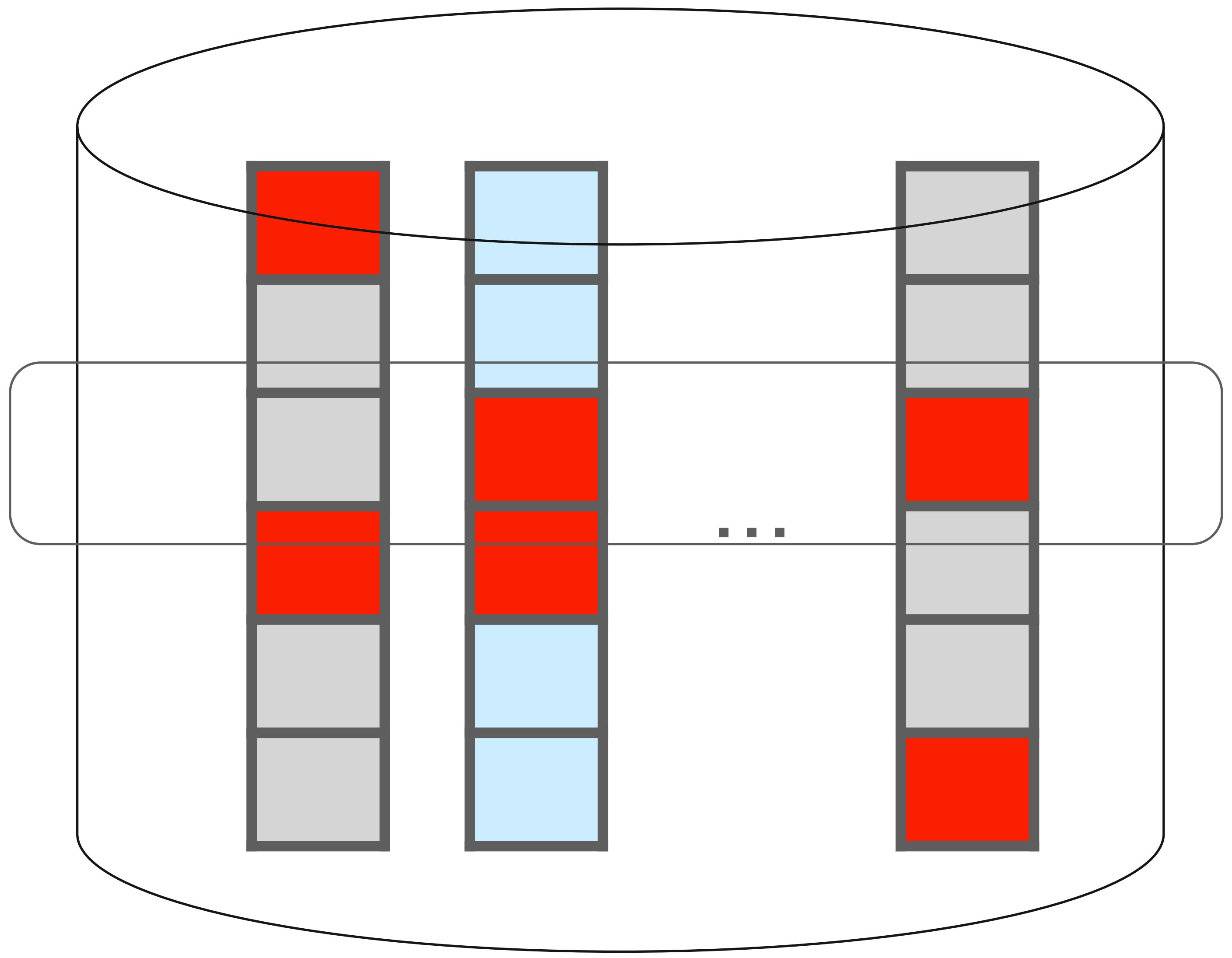
Double amplification solution: SS-Double

$$\epsilon_l \rightarrow \epsilon_c = \epsilon_l$$

$$\epsilon_l \rightarrow_d \epsilon_{ld} = \epsilon_l/d \rightarrow_s \epsilon_{cd} \rightarrow_c \epsilon_c$$

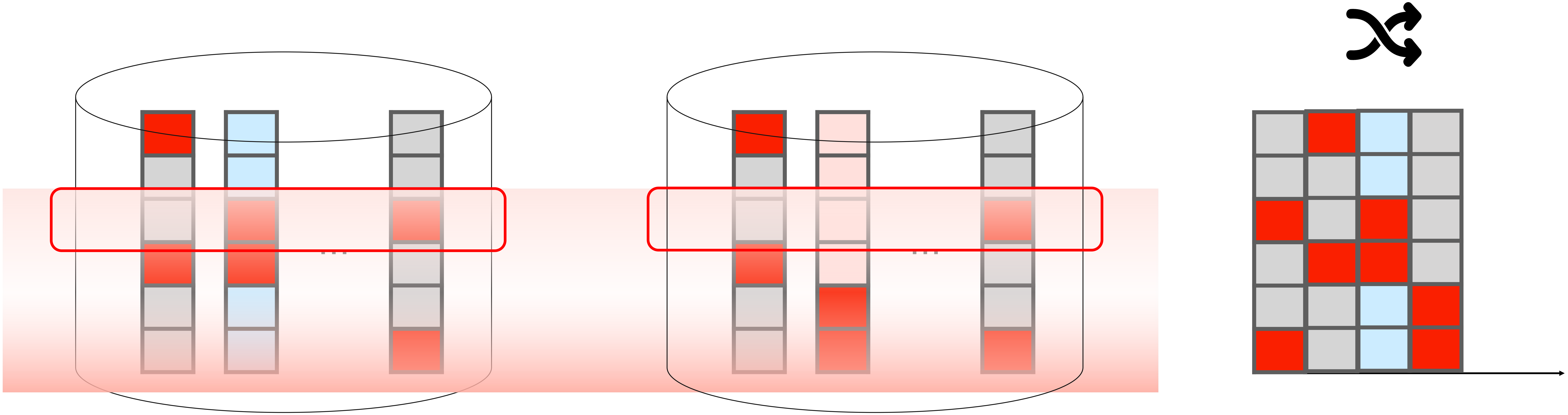


$$\epsilon_l \rightarrow_d \epsilon_{lk} = \epsilon_l/k \rightarrow_s \epsilon_{ck} \rightarrow_{smp} \epsilon_{cd} \rightarrow_c \epsilon_c$$



Double privacy amplification

Dummy padding for SS-Double



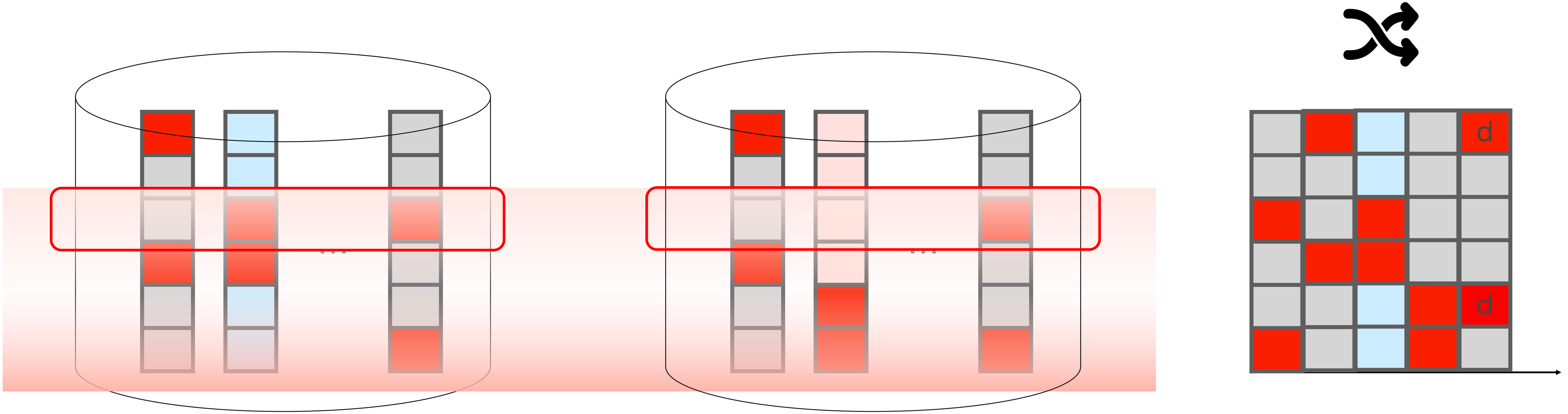
Challenge

- proof of privacy amplification by shuffling relies on bounded-size neighboring datasets
- subsampling may lead to two neighboring sub-datasets with distinct size

Solution

- Let the shuffler pad each dimension to the same size with dummy values

Dummy padding for SS-Double



Challenge

- proof of privacy amplification by shuffling relies on bounded-size neighboring datasets
- subsampling may lead to two neighboring sub-datasets with distinct size

Solution

- Let the shuffler pad each dimension to the same size with dummy values

From the Privacy blanket!

Utility boosting solution: SS-Topk

Insight

- The random subsampling treats all dimensions equally and thus may discard “important” dimensions
- Top-k sparsification [EMNLP’2017] is an efficient and general technique to boost the learning performance

Challenge

- Selecting Top-k is data-dependent
- Explicitly revealing Top-k index to the shuffler has privacy risks

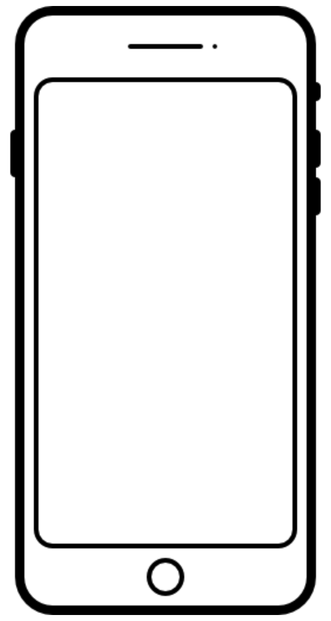
Goal

- Define and control the information leakage from Top-k index while maintaining the utility as far as possible

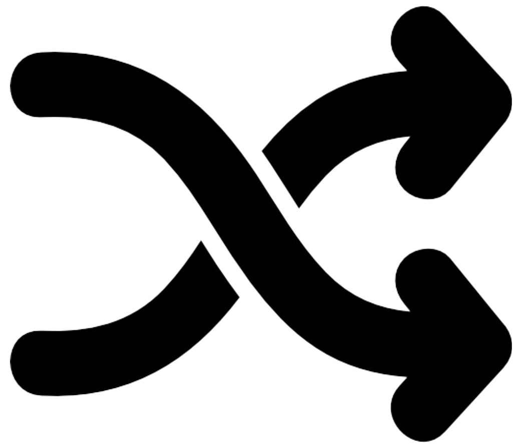
Utility boosting solution: SS-Topk

Index-privacy

Definition 3 *A mechanism* $\mathcal{K}_{\nu}^{\beta}$ *provides* ν*-index privacy for a d-dimensional vector, if and only if for any* $j \in [d], \nu \geq 1$, *we have:* $\Pr[\mathbb{I}_j = 1|\mathcal{K}_{\nu}^{\beta}(j)] \leq \nu \cdot \boxed{\Pr[\mathbb{I}_j = 1]}$ *and* $\Pr[\mathbb{I}_j = 0|\mathcal{K}_{\nu}^{\beta}(j)] \geq \frac{\Pr[\mathbb{I}_j=0]}{\nu}$.



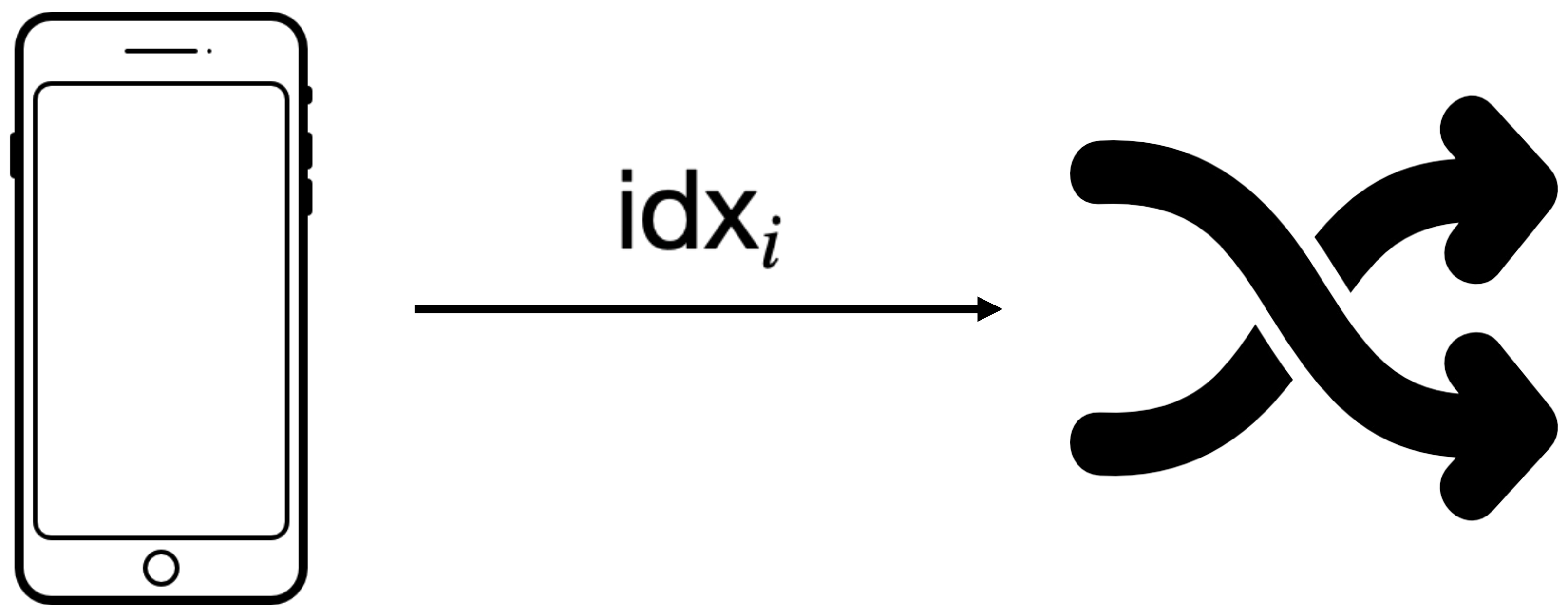
?



Utility boosting solution: SS-Topk

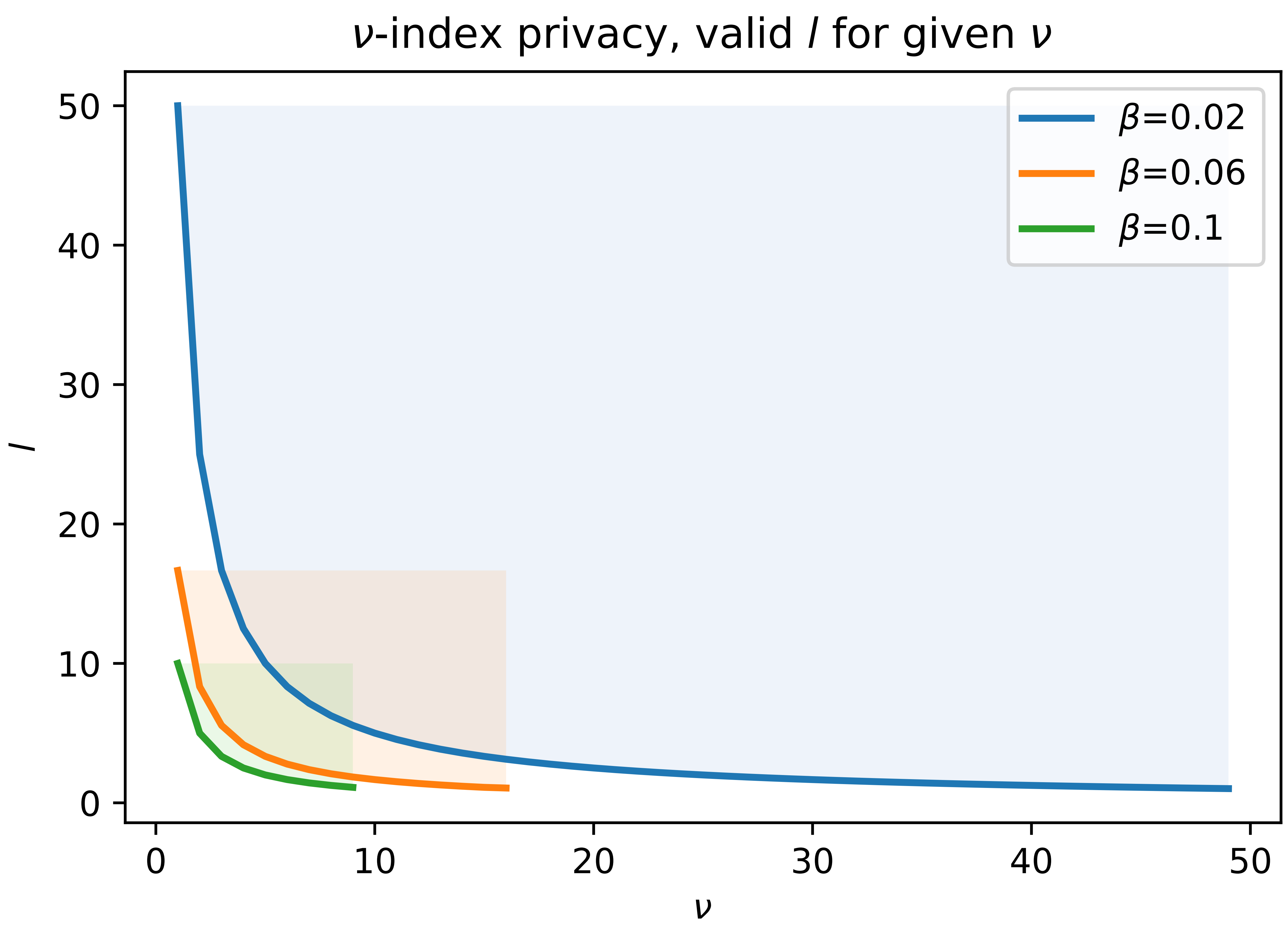
Index-privacy

Definition 3 *A mechanism $\mathcal{K}_\nu^\beta$ provides ν-index privacy for a d-dimensional vector, if and only if for any $j \in [d], \nu \geq 1$, we have:* $\boxed{\Pr[\mathbb{I}_j = 1 | \mathcal{K}_\nu^\beta(j)]} \leq \nu \cdot \Pr[\mathbb{I}_j = 1]$ *and* $\Pr[\mathbb{I}_j = 0 | \mathcal{K}_\nu^\beta(j)] \geq \frac{\Pr[\mathbb{I}_j = 0]}{\nu}$.



Utility boosting solution: SS-Topk

Index-privacy

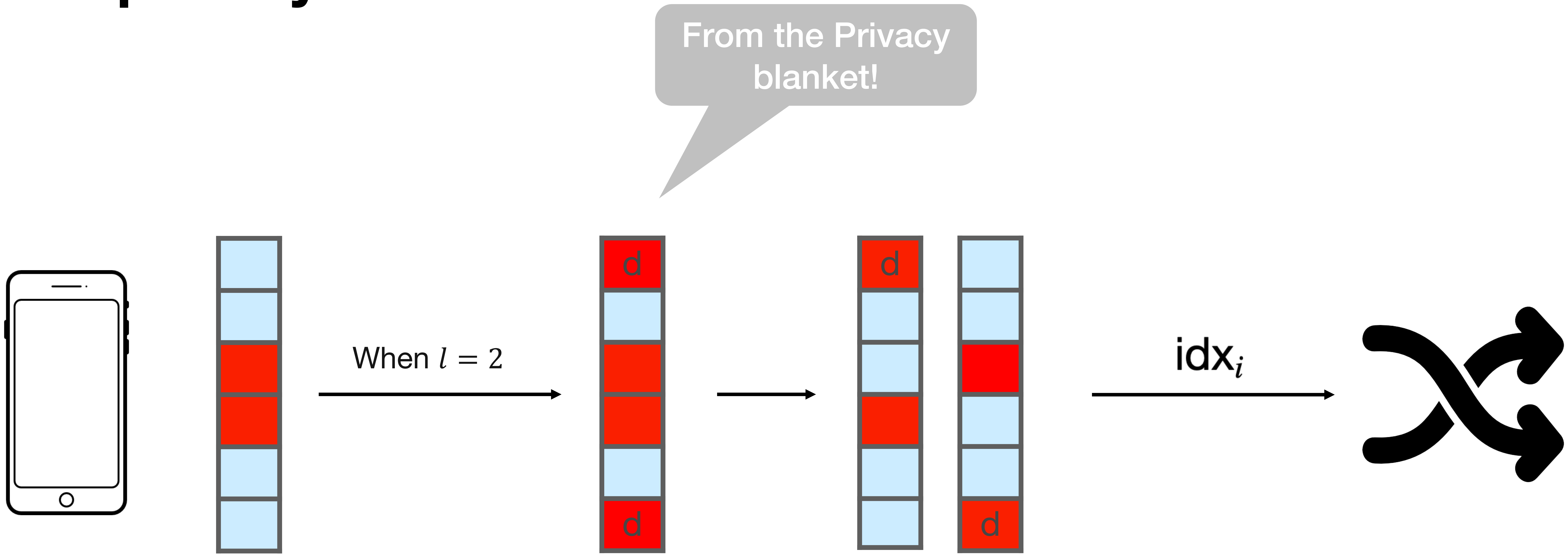


Proposition 2 *The range of ν-index privacy is $1 \leq \nu \leq \frac{1}{\beta}$, where the strongest index privacy $\nu = 1$ is achieved when $l = \lceil \frac{1}{\beta} \rceil$ and no index privacy is achieved when $l = 1$.*

Theorem 5 *Given a protocol with $\mathcal{K}_{\nu}^{\beta}$, n_p, the strongest index privacy it allows for each user is $\nu = \max\{1, \frac{1}{\lfloor \frac{n_p}{n\beta} \rfloor \cdot \beta}\}$.*

Utility boosting solution: SS-Topk

Index-privacy

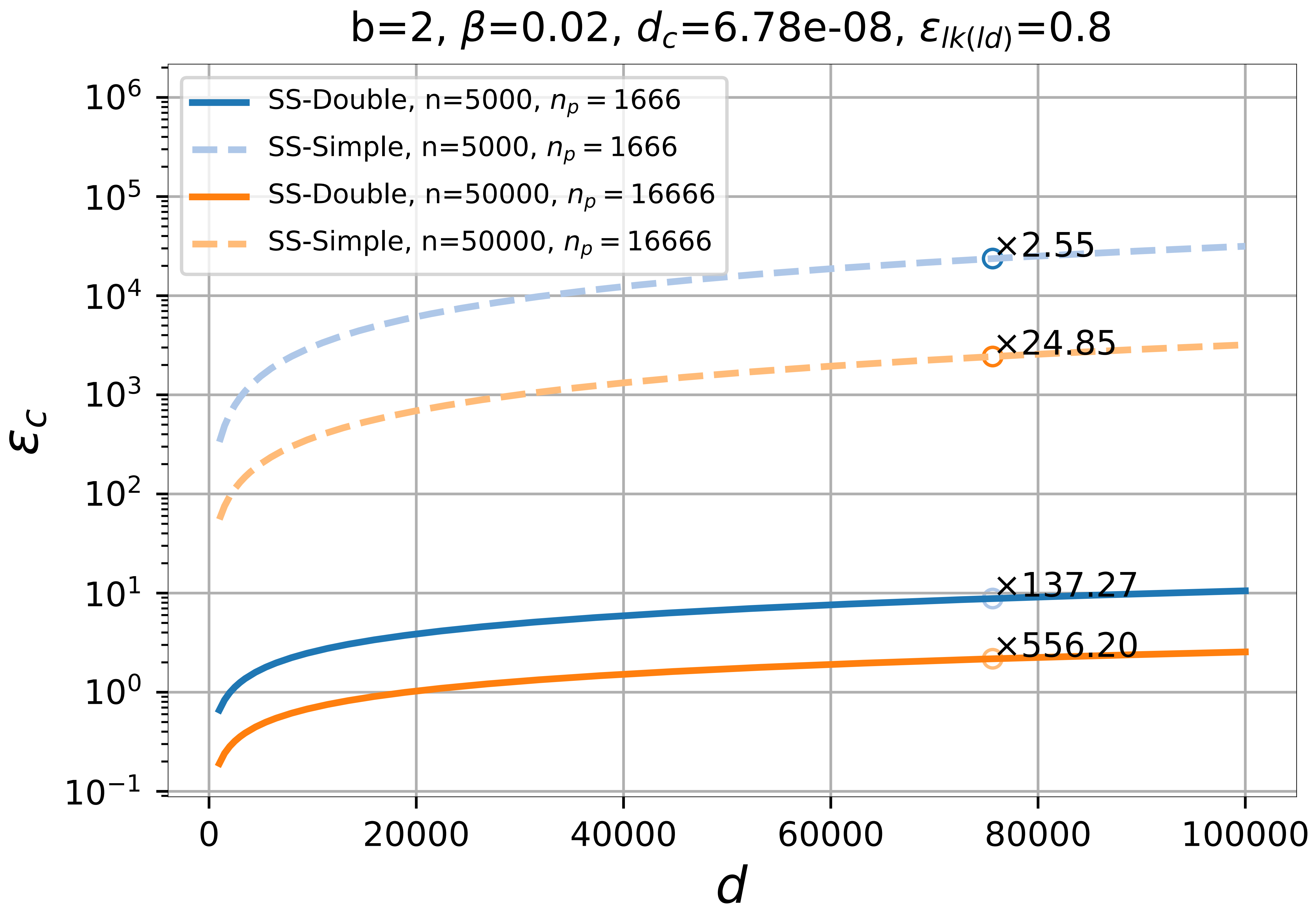


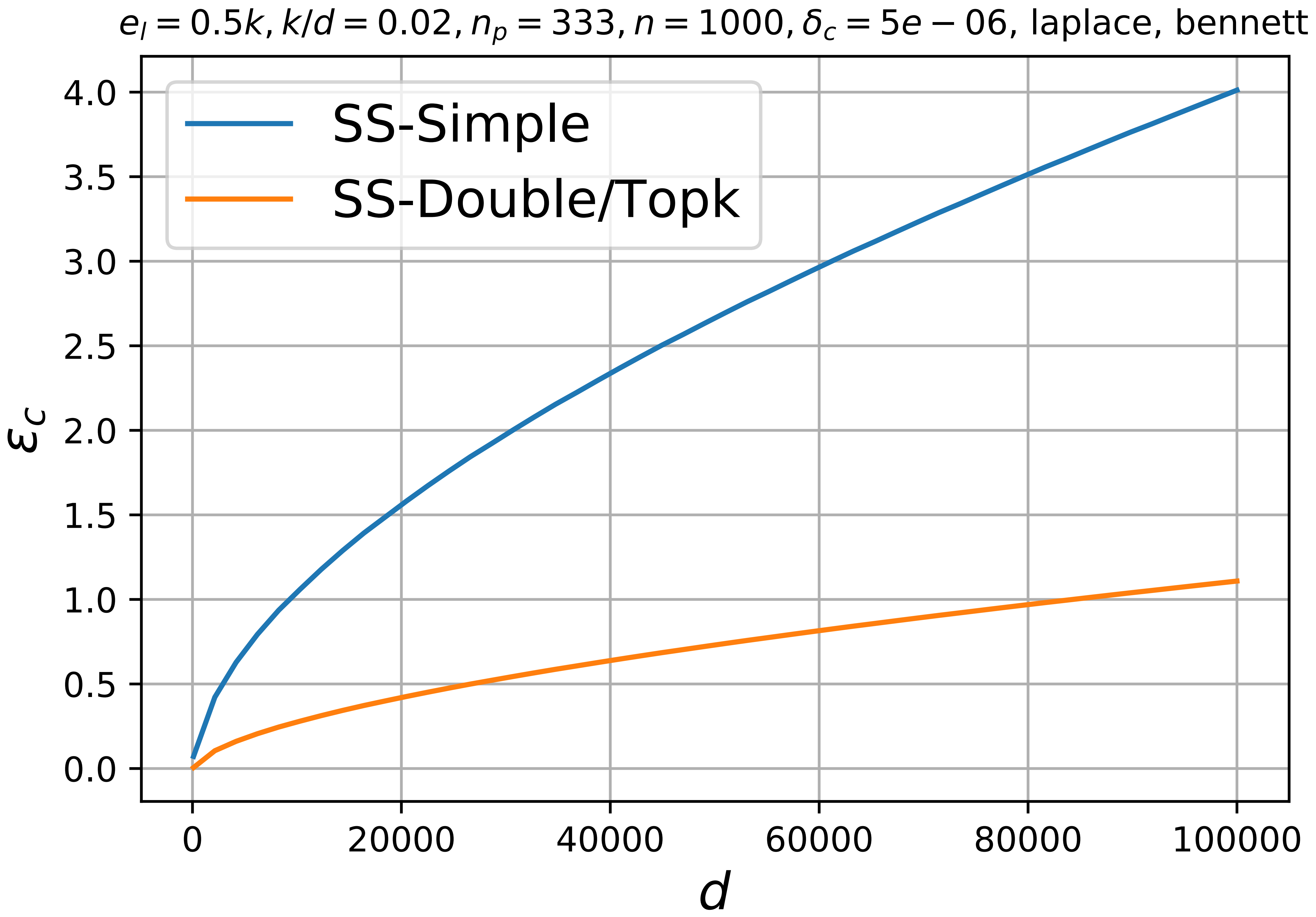
Each Top-k index is hidden in l indexes

Evaluations

Evaluations

Double privacy amplification effect

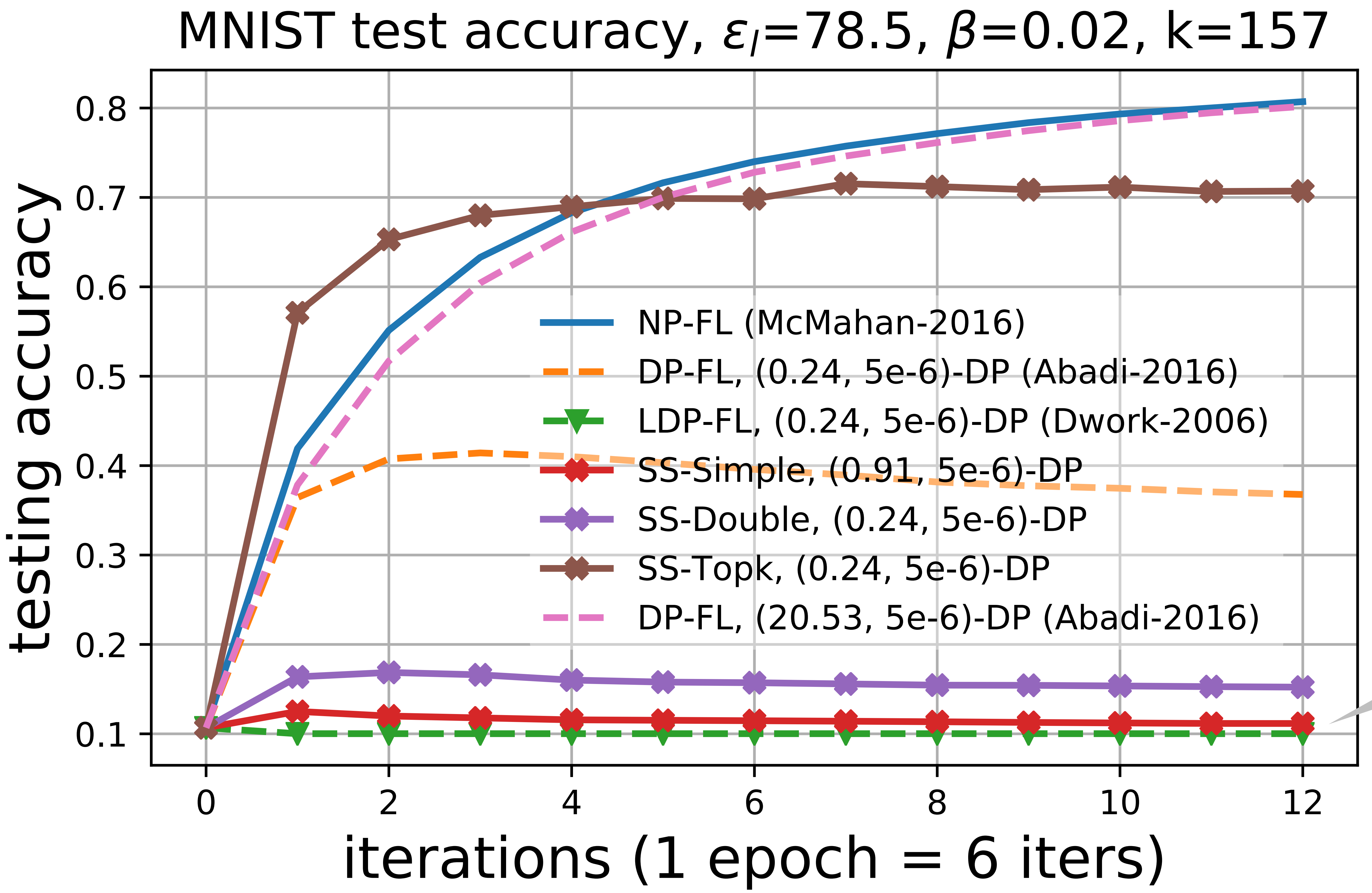




- The magnification ratio ϵ_l/ϵ_c is enlarged by dozens of times with double amplification
- The improvement is more significant for a larger d

Evaluations

Utilities

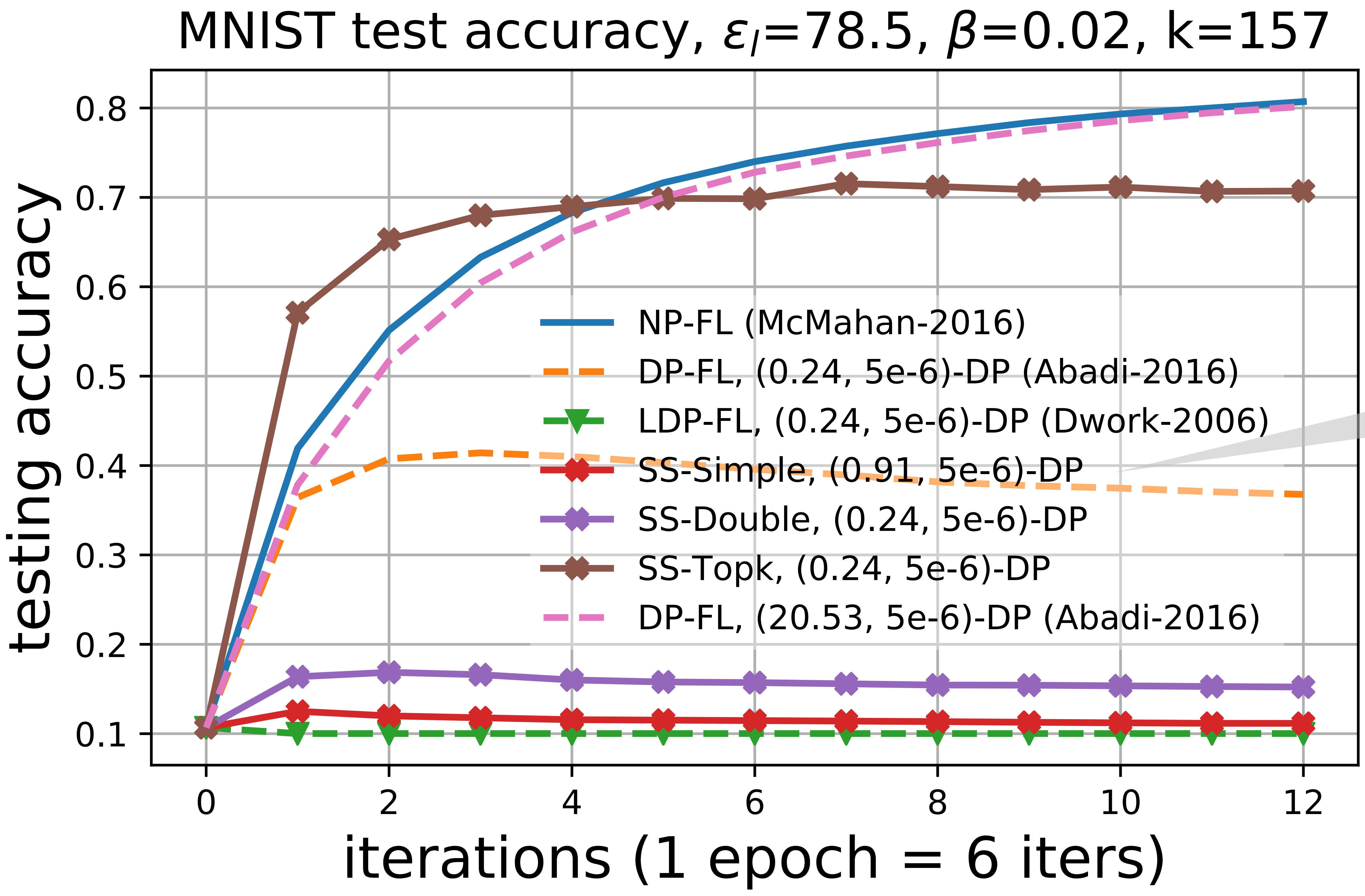


The performance of LDP-FL is no greater than random guessing in the high-dimensional case with d=7850, n=1000

- SS-Topk > DP-FL > SS-Double > SS-Simple > LDP-FL

Evaluations

Utilities

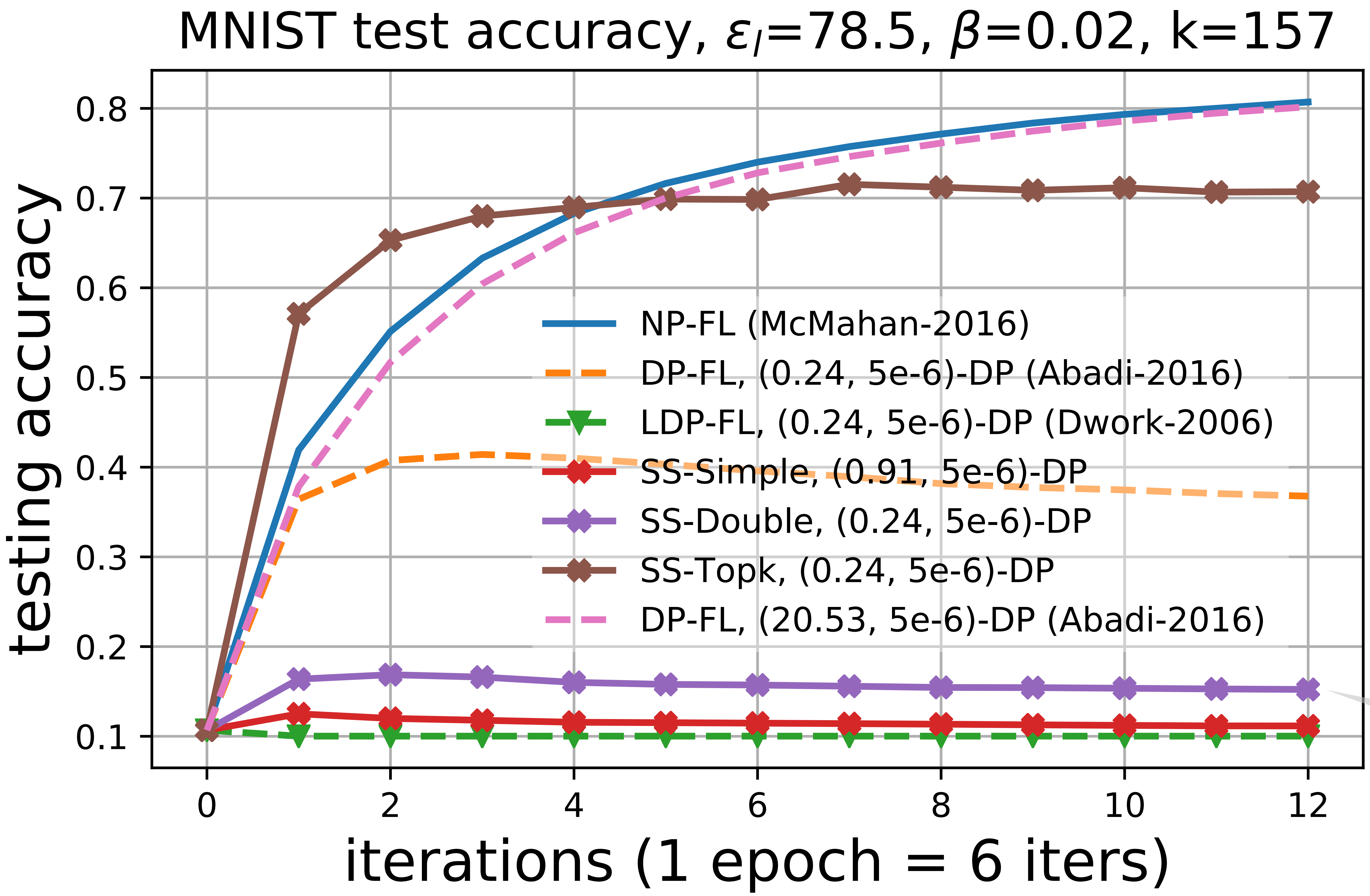


- The central privacy is enhanced by Double amplification from 0.91 to 0.24

- SS-Topk > DP-FL > SS-Double > SS-Simple > LDP-FL

Evaluations

Utilities

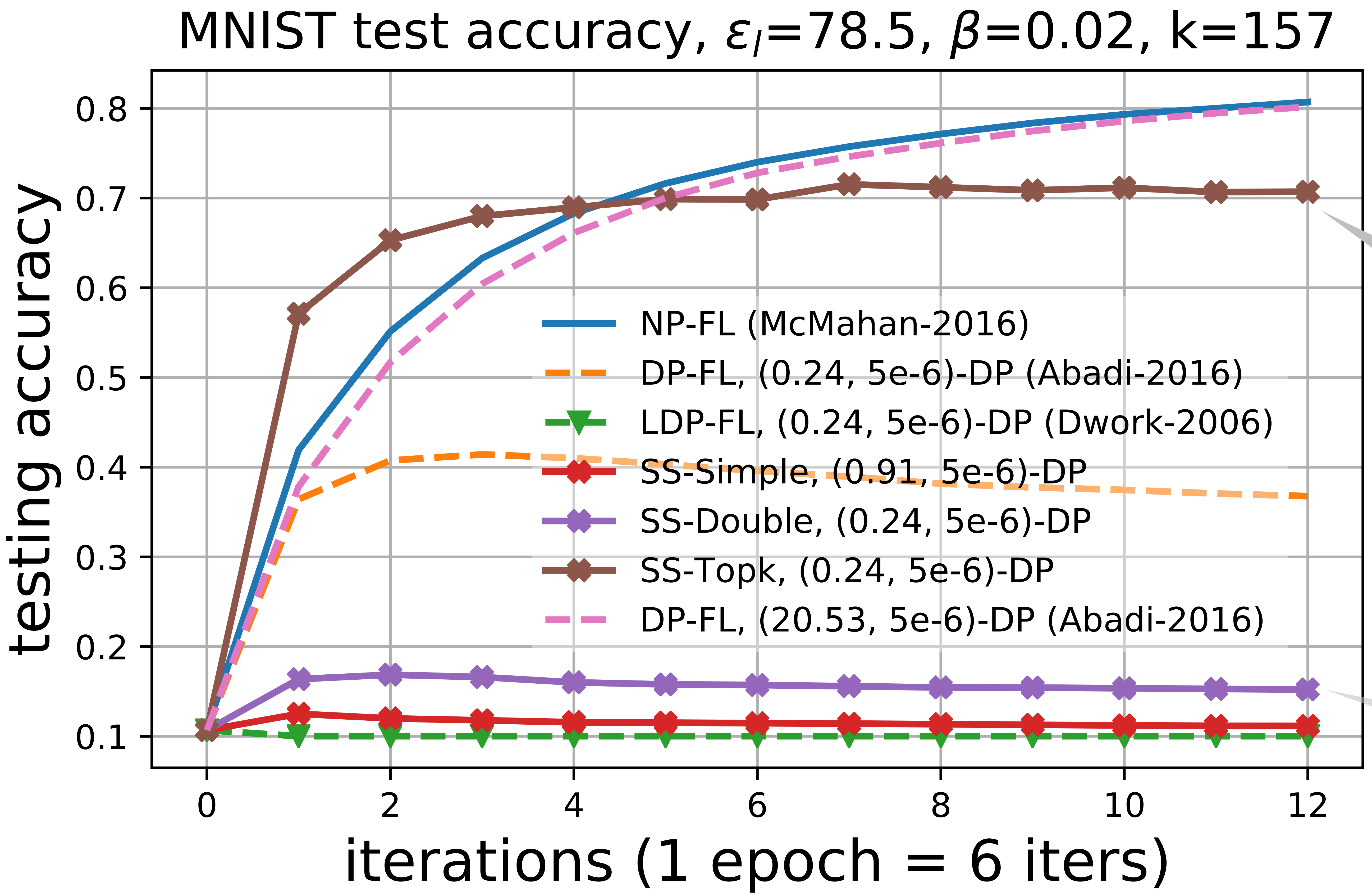


- The random subsampling of SS-Double reduces injected error in the averaged vector

- SS-Topk > DP-FL > SS-Double > SS-Simple > LDP-FL

Evaluations

Utilities



[NeurIPS'2019]

"gradient compression successfully defends the attack with the pruned gradient is more than 20%"

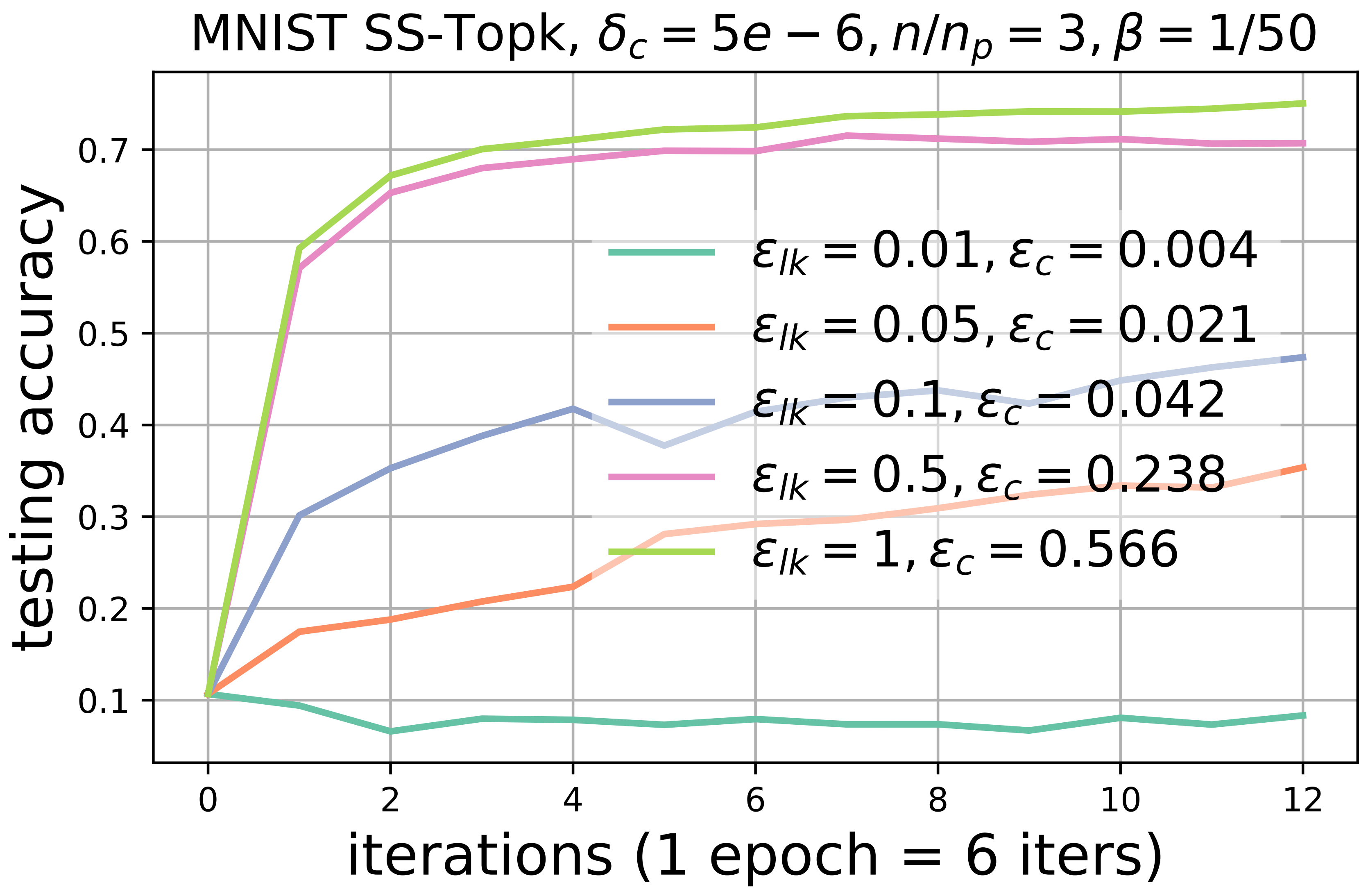
- With the same padding size, Top-k strategy in SS-Topk boosts the utility significantly
- The index privacy level against the shuffler is $\nu = 3.125, l = 16$

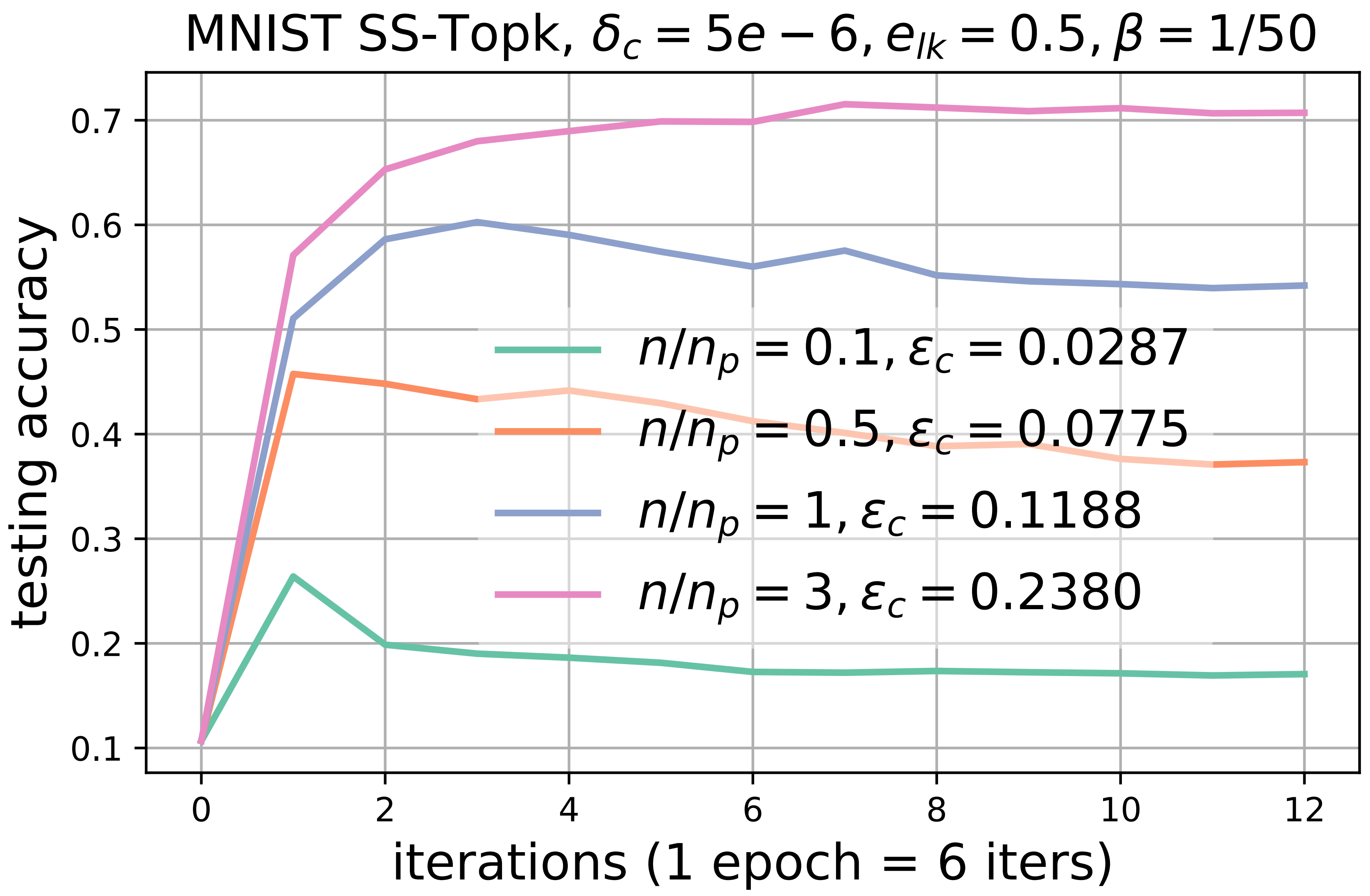
- The random subsampling of SSDouble reduces injected error in the averaged vector

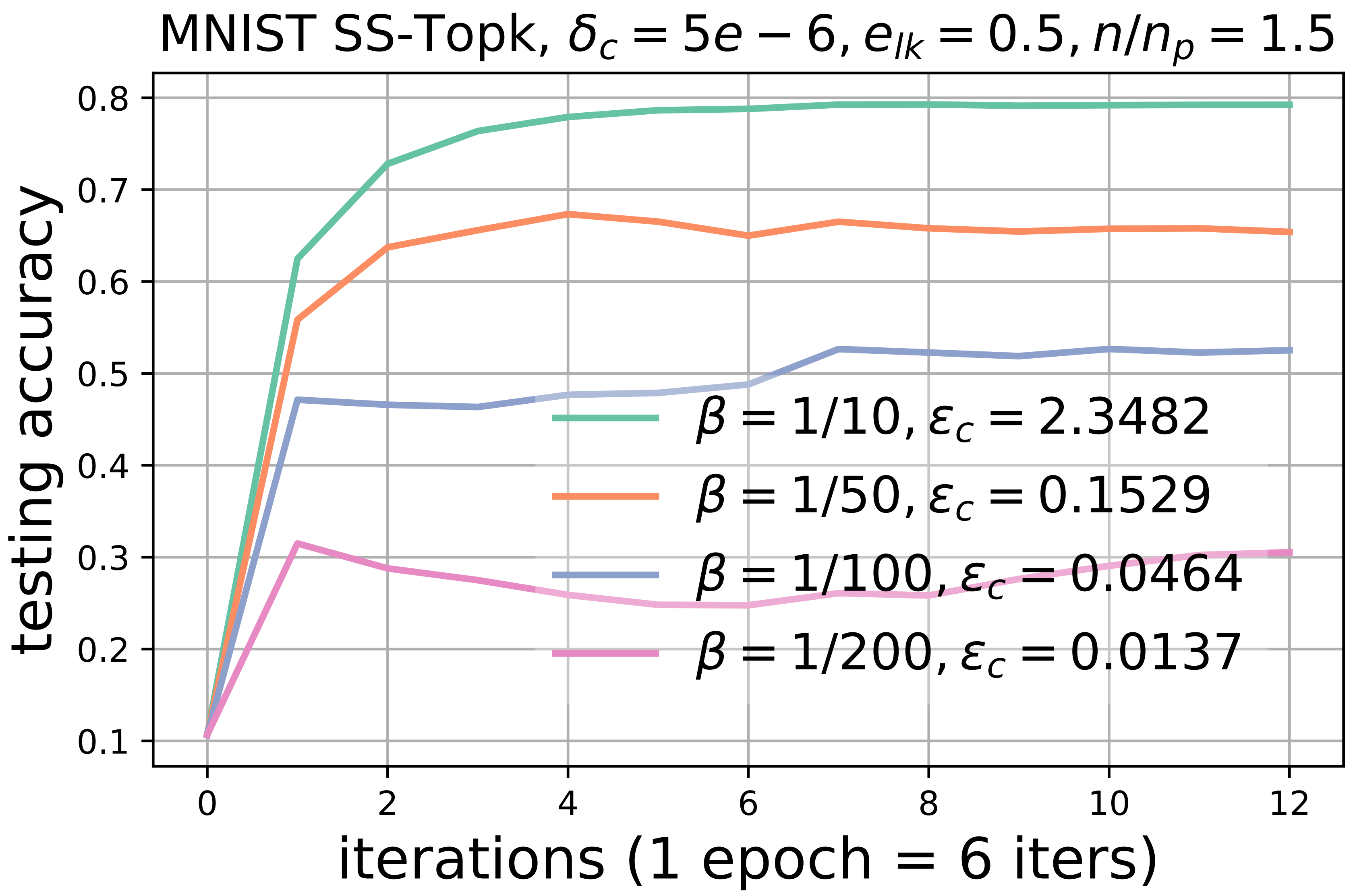
- SS-Topk > DP-FL > SS-Double > SS-Simple > LDP-FL

Evaluations

Variant parameters







- A larger local privacy budget for each dimension leads to higher testing accuracy
- Higher ratio of n/n_p indicates less noise is injected
- Larger sampling ratio implies better utility

Takeaways

- Multi-fold privacy amplification effect is a promising way to bound privacy in practice for better utility
- Separating trust on different parties largely reduces the privacy leakage while maintaining utility
- How far a privacy attack can go under a certain index-privacy level without revealing corresponding values is an open question

Thanks