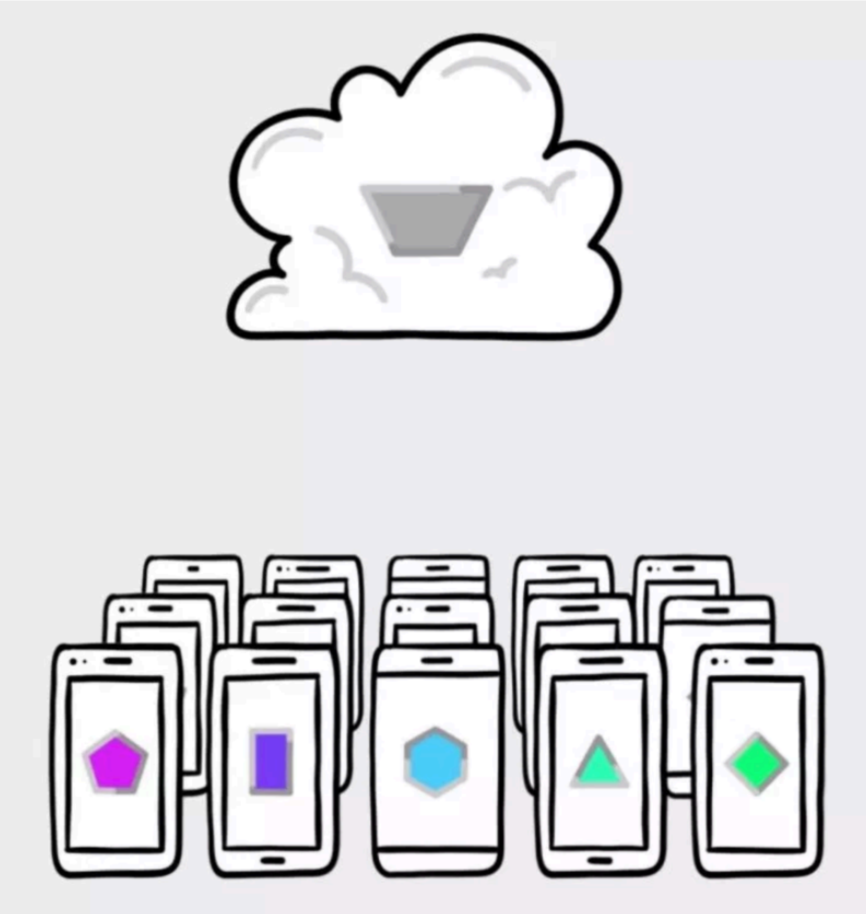
# Federated Learning

Shusen Wang

# Motivating Examples

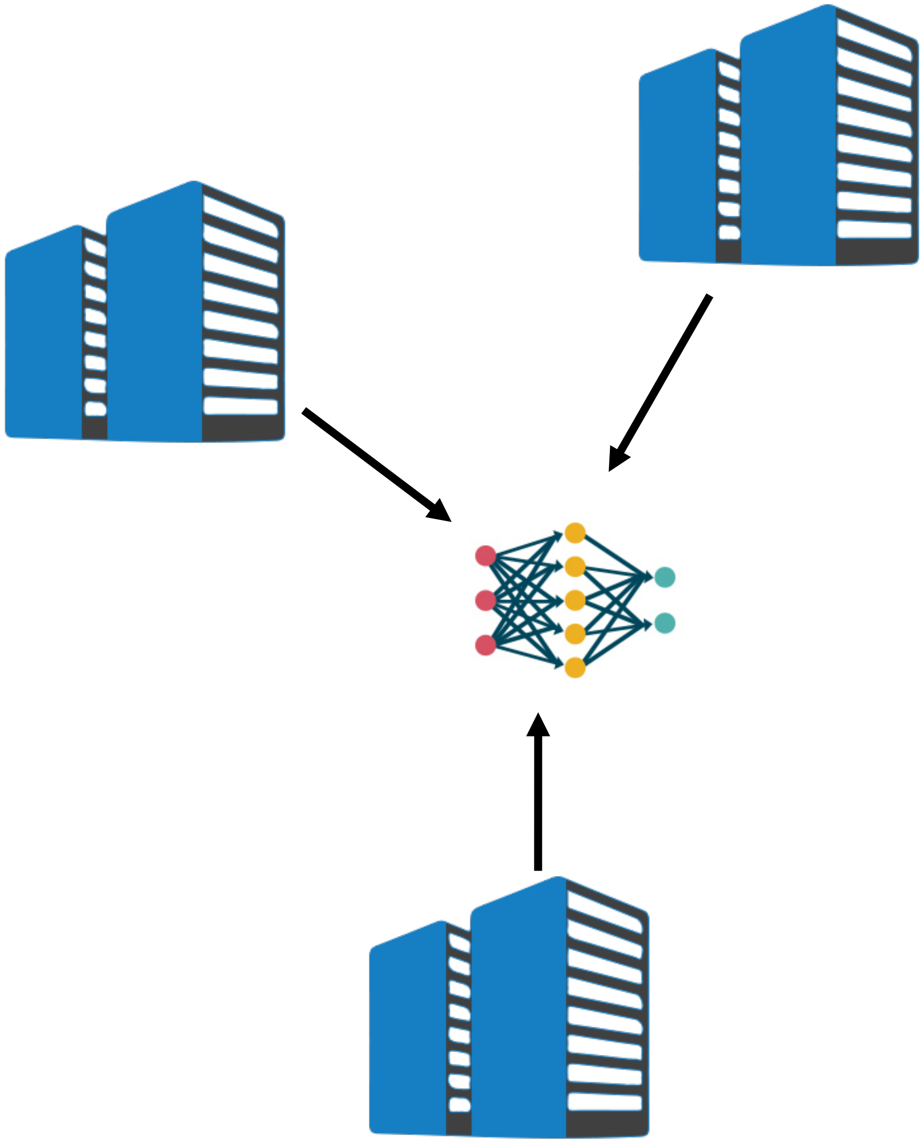


**Problem**: Google wants to train a model using users' mobile data.

**Possible solution**: Centralized learning
- Collect users' data.
- Train a model on the cluster.

**Challenge**: Users may refuse to upload their data, especially sensitive data, to Google's server.

# Motivating Examples



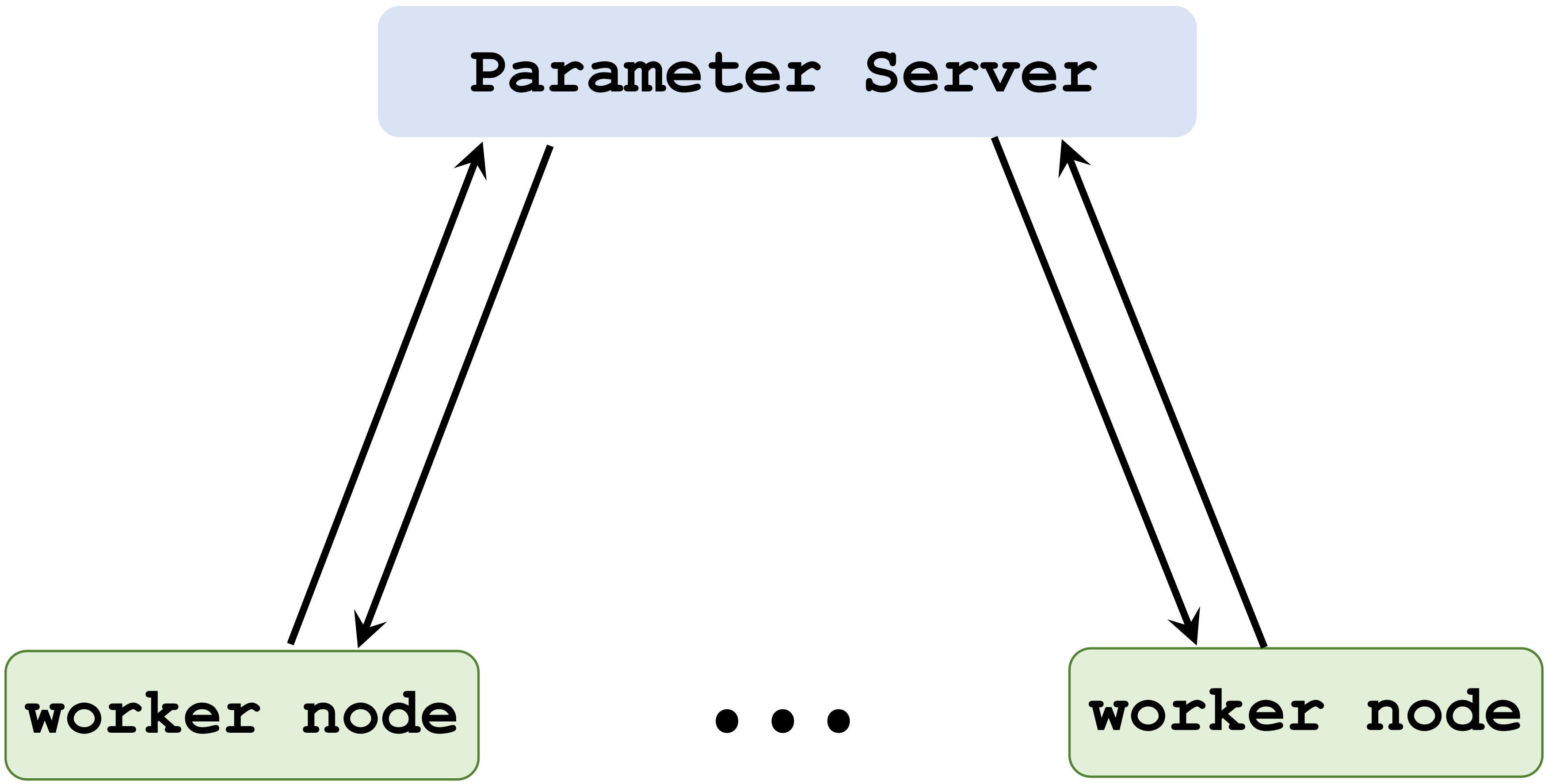
**Problem**: Hospitals want to jointly train a model using medical data.

**Possible solution**: Centralized learning
- Aggregate the data.
- Train a model on the server.

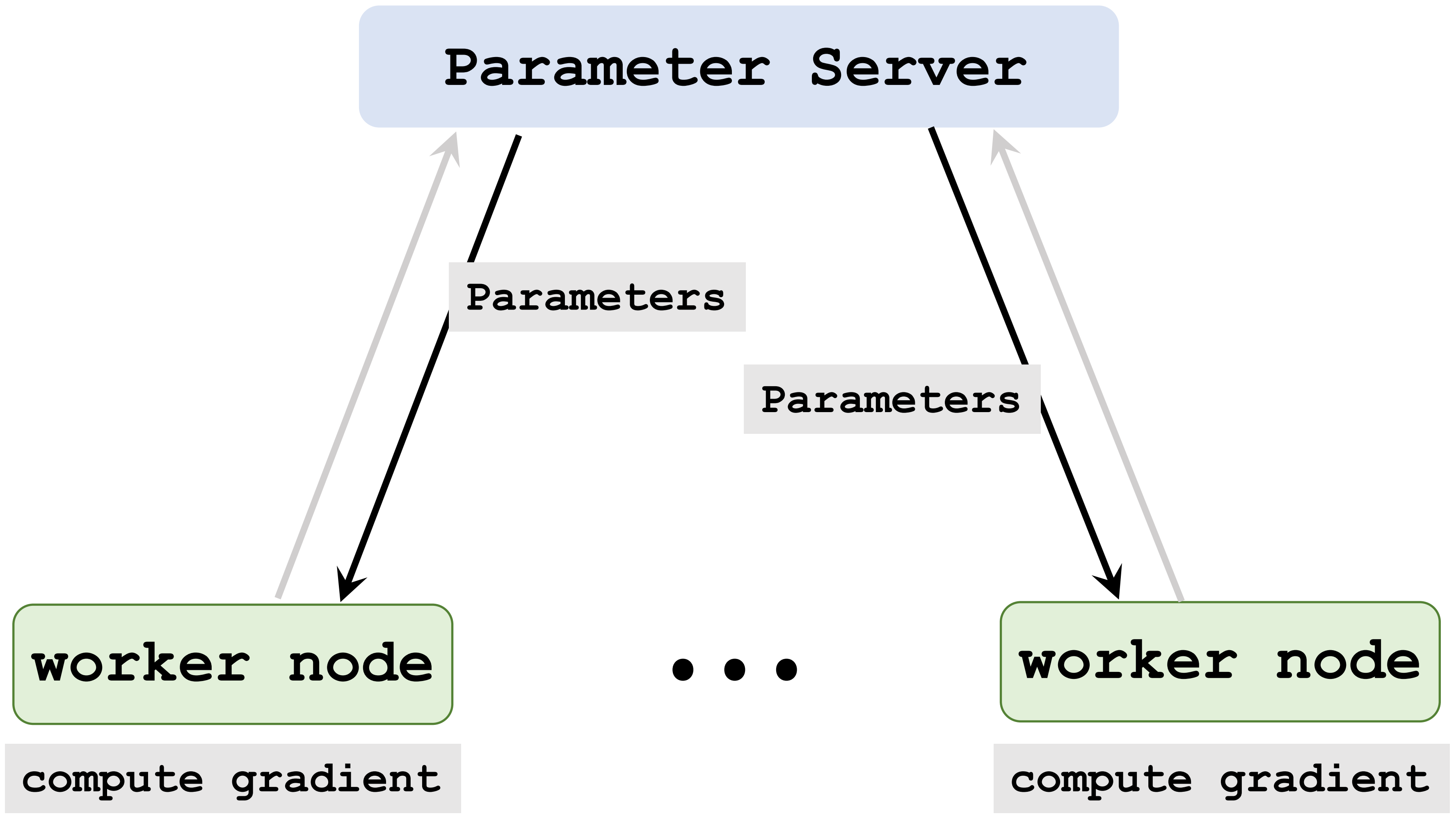
**Challenge**: Laws or policies may forbid giving patients' data to others.

# Distributed Learning vs. Federated Learning

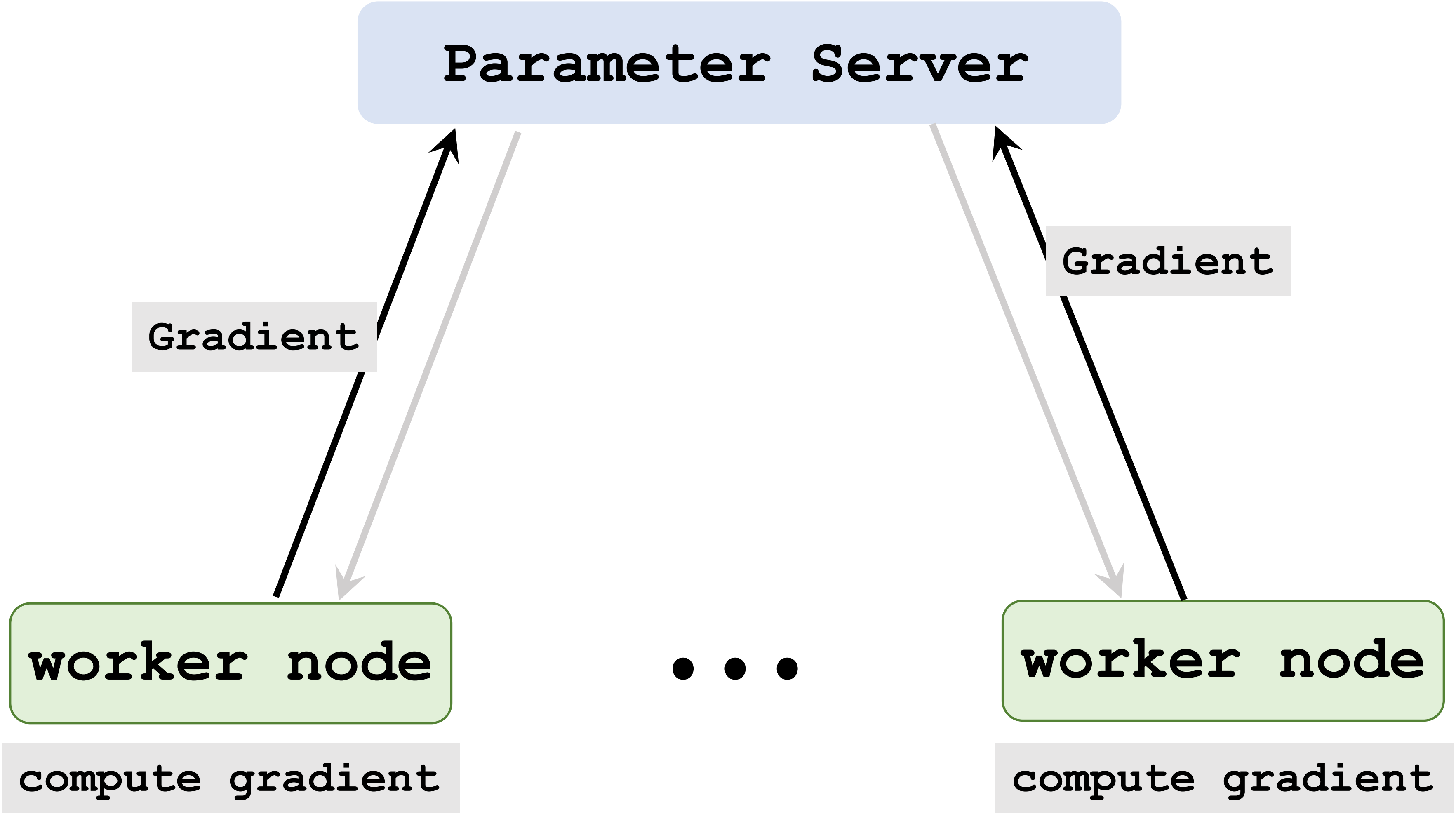
# Distributed Learning



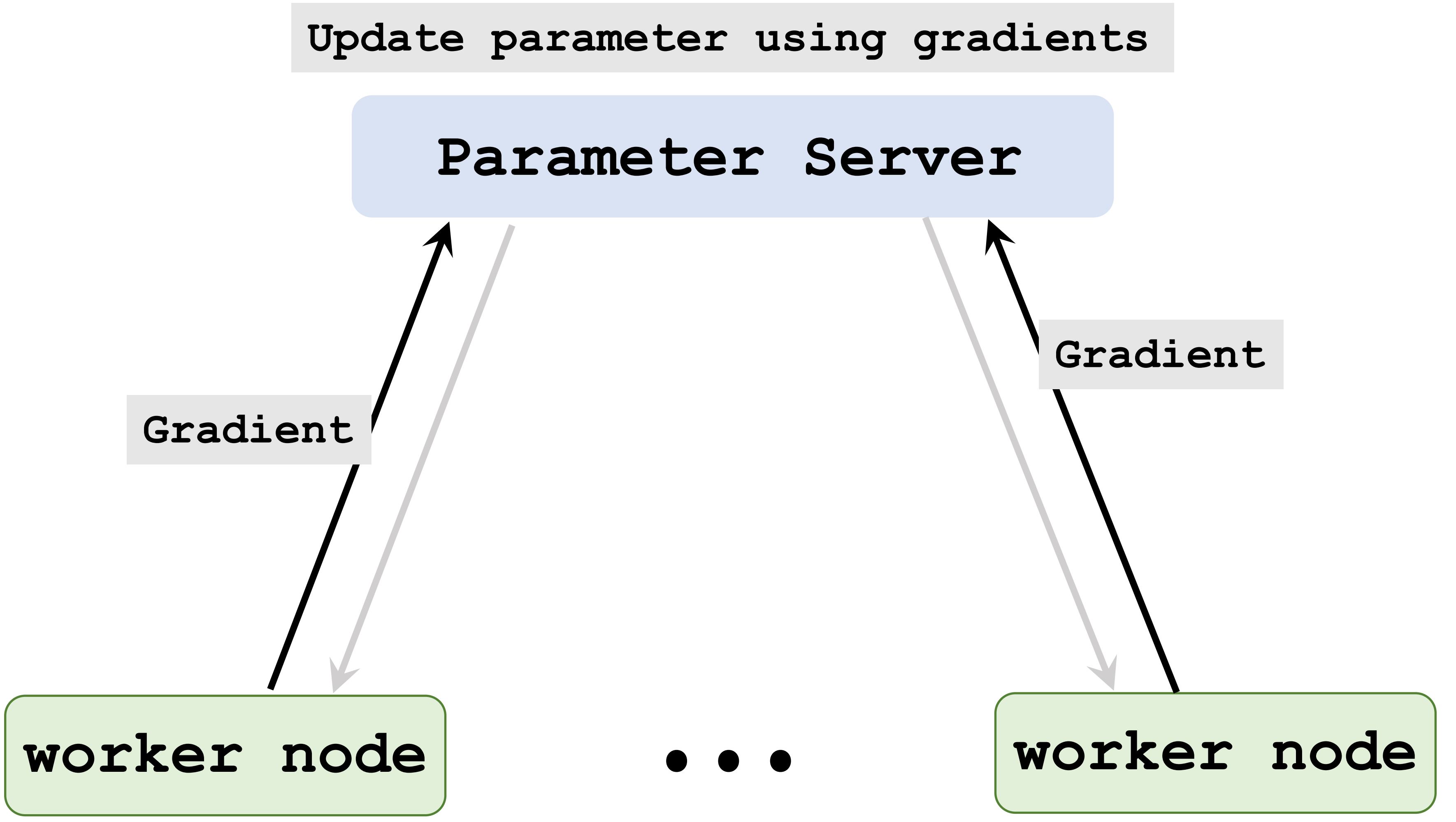
# Distributed Learning



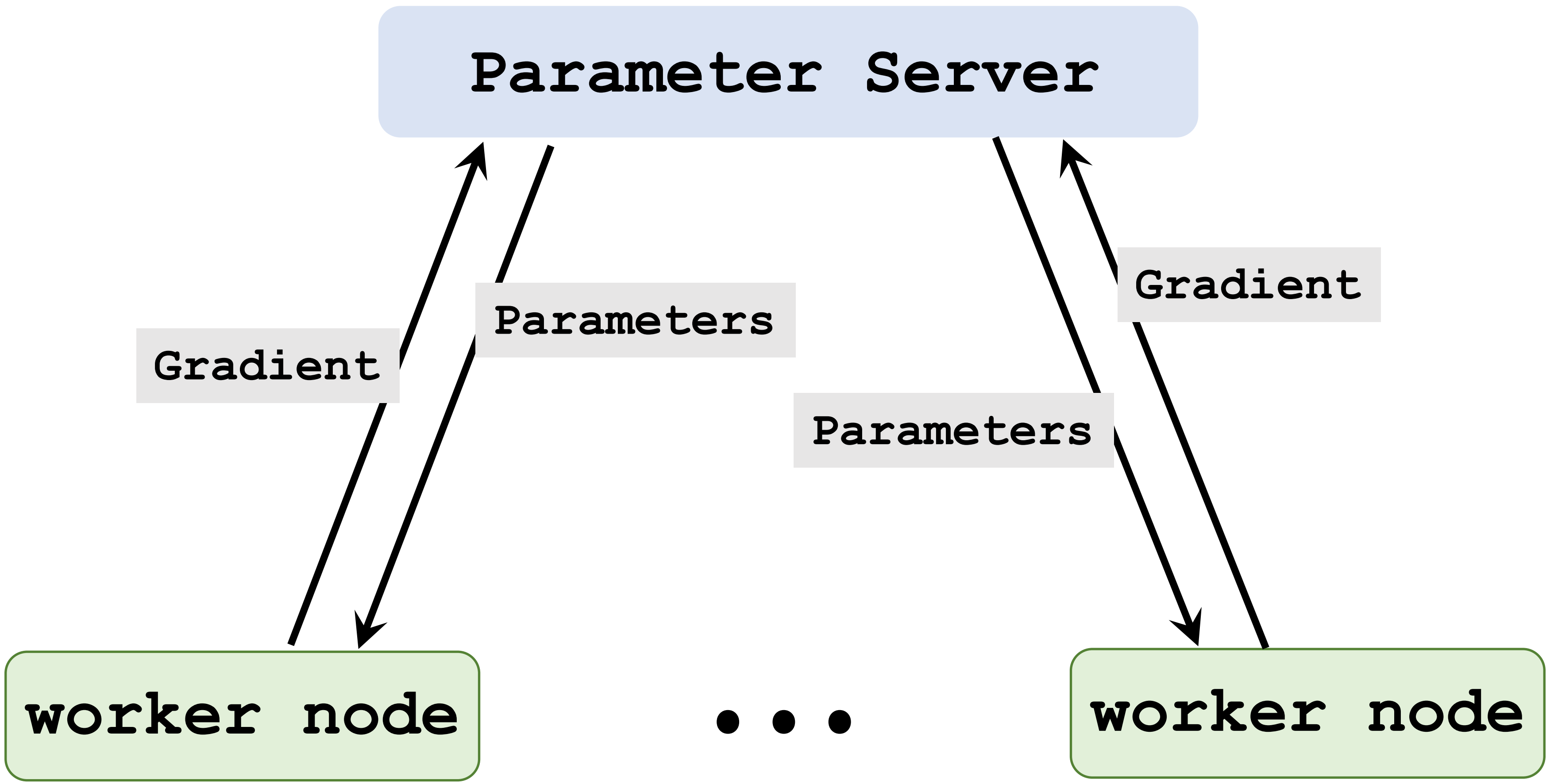
# Distributed Learning



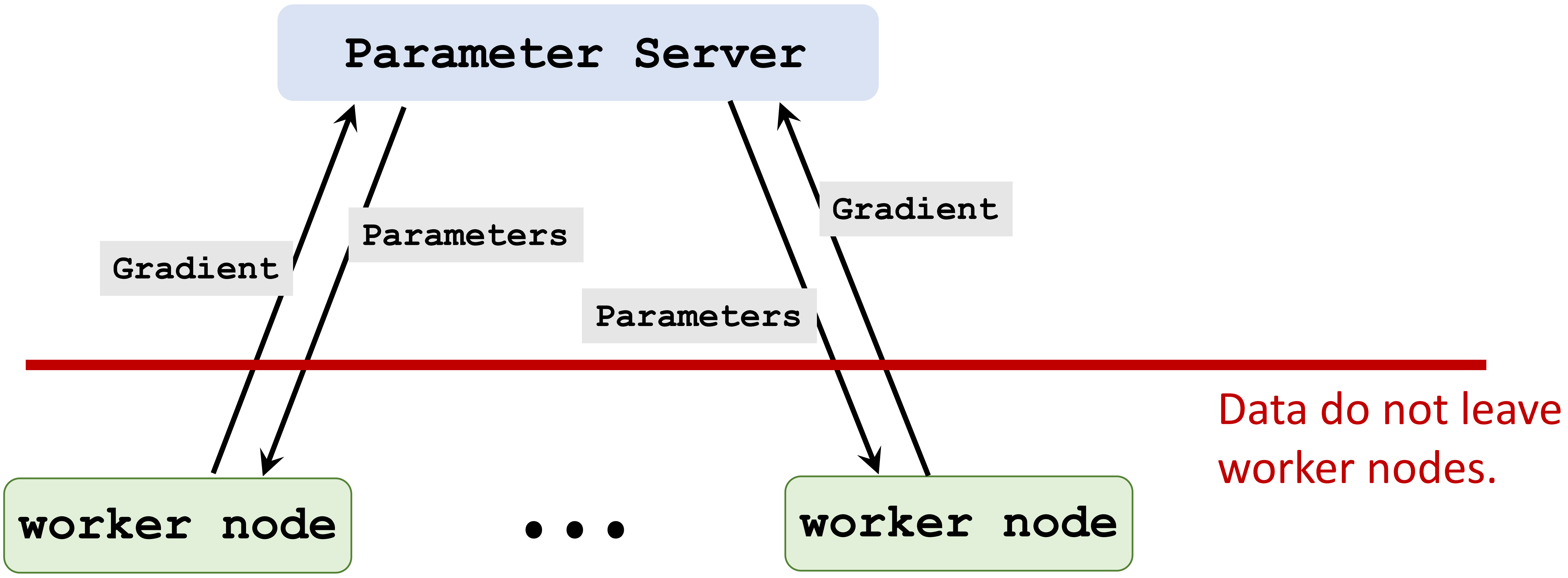
# Distributed Learning



# Distributed Learning



# Distributed Learning



# What is federated learning?

Federated learning [1, 2] is a kind of distributed learning.

How does federated learning differ from traditional distributed learning?

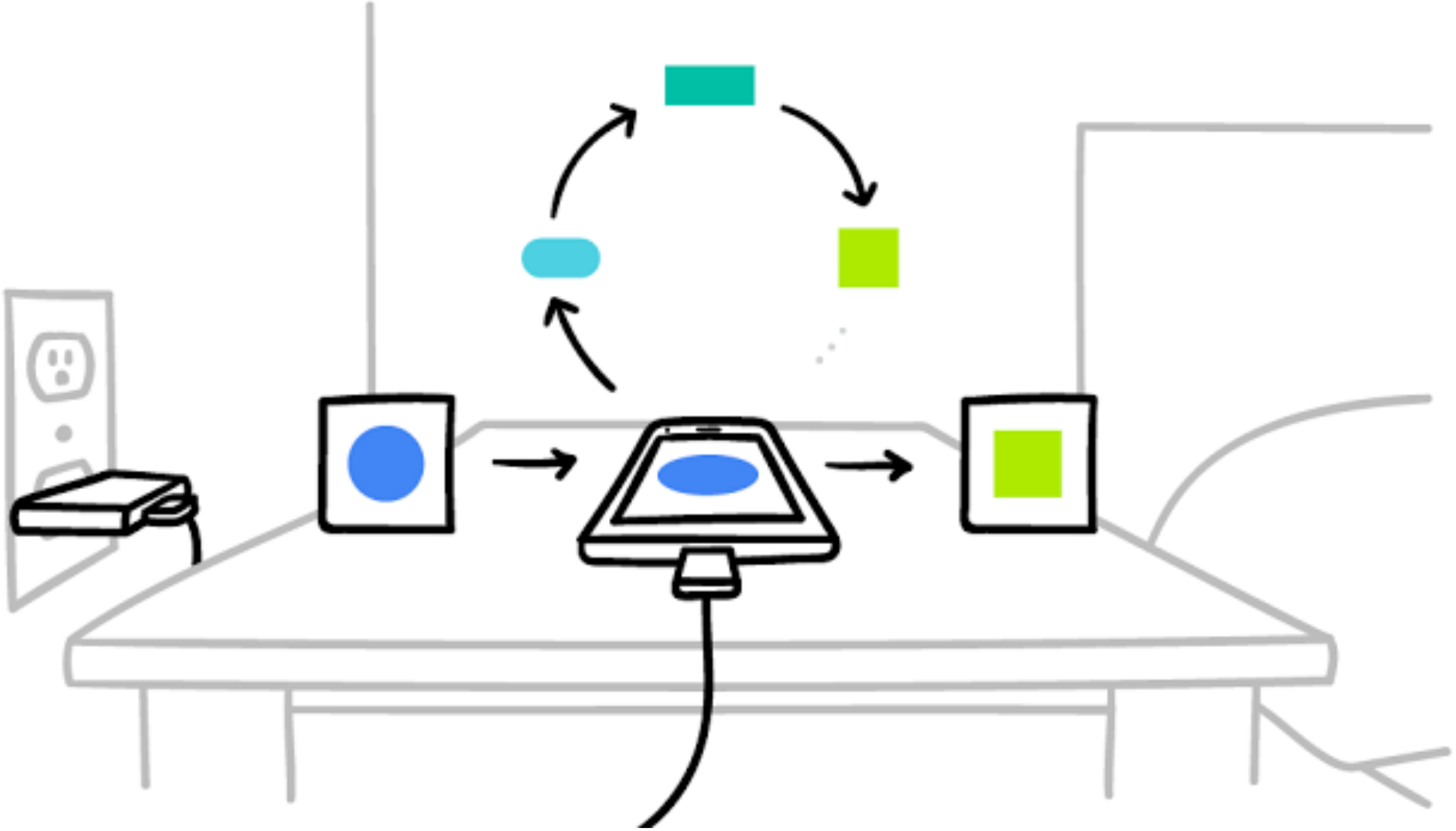
1. Users have control over their device and data.
2. Worker nodes are unstable.
3. Communication cost is higher than computation cost.
4. Data stored on worker nodes are not IID.
5. The amount of data is severely imbalanced.

**References**


1. McMahan and others: Communication-Efficient Learning of Deep Networks from Decentralized Data. In *AISTATS*, 2017.
2. Konevcny, McMahan, and Ramage: Federated optimization: distributed optimization beyond the datacenter. *arXiv:1511.03575*, 2015

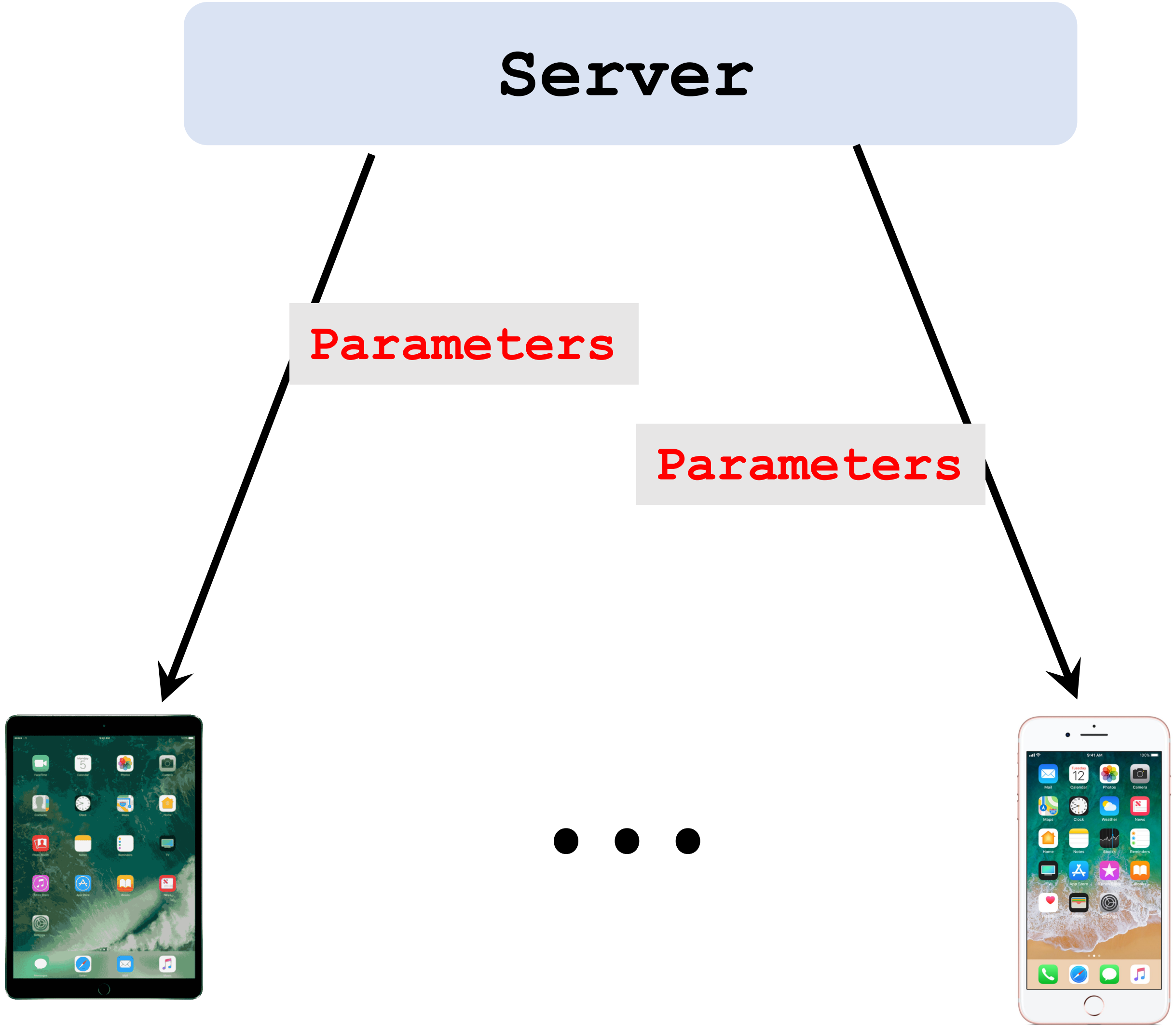
# Research Direction 1: Communication-Efficiency

# Trade computation for communication

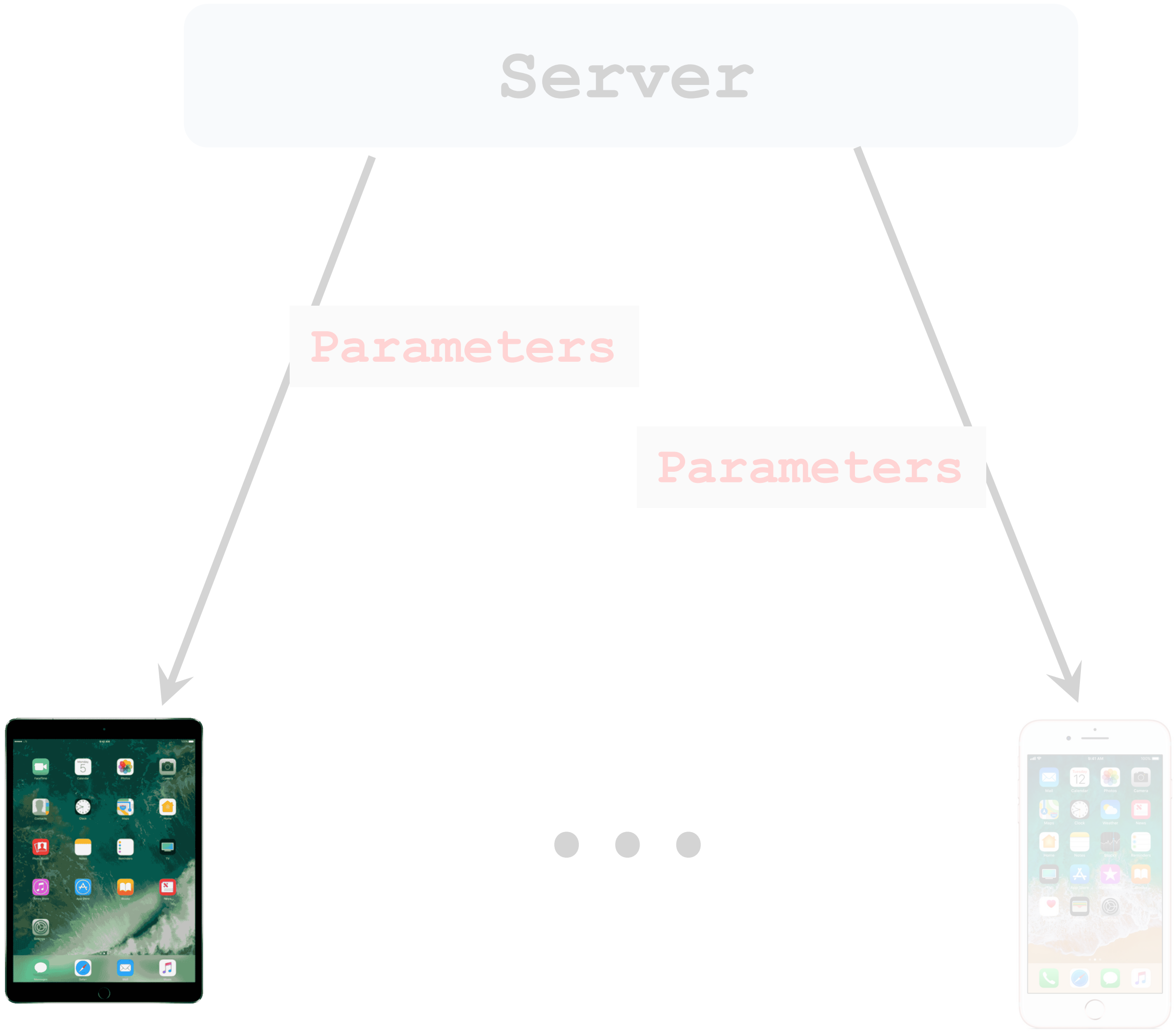


The image is From Google Research Blog

# Let us recall parallel gradient descent



# Let us recall parallel gradient descent



# Let us recall parallel gradient descent

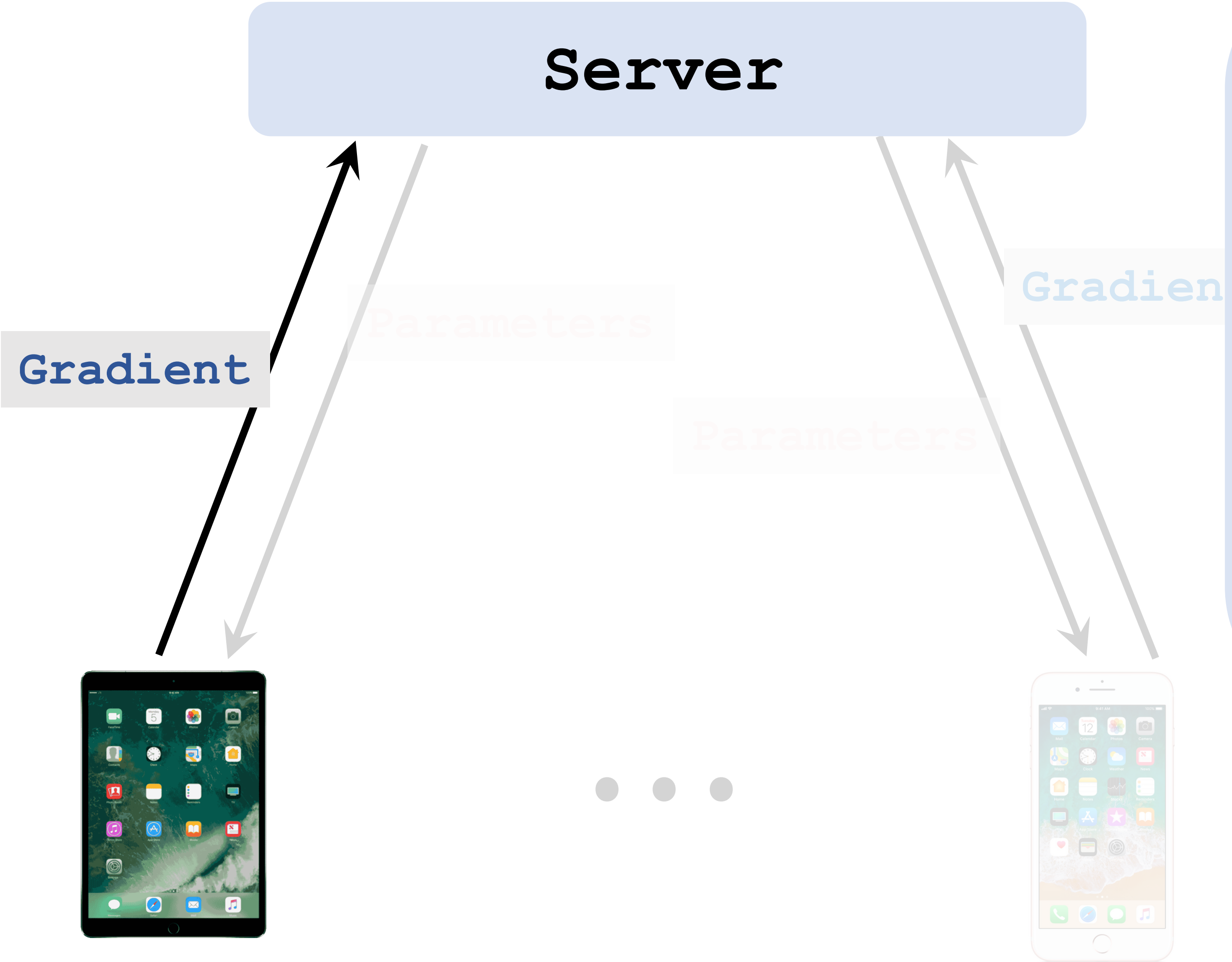
Server

Param

**The $i$-th worker performs:**

1. Receiving model parameters $\mathbf{w}$ from the server.
2. Using $\mathbf{w}$ and its local data to compute gradient $\mathbf{g}_i$.
3. Sending $\mathbf{g}_i$ to the server.

# Let us recall parallel gradient descent

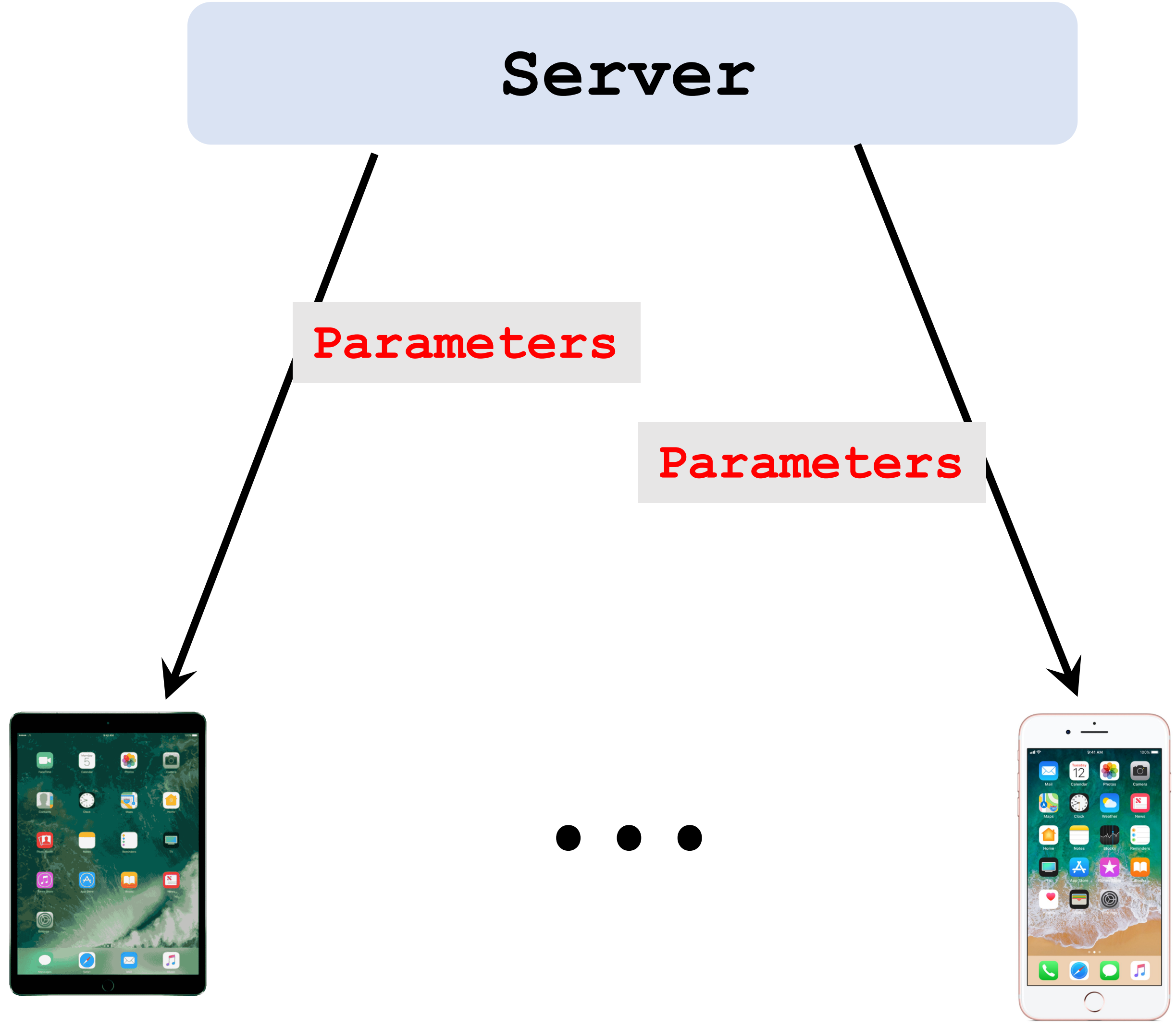



**The server performs:**

1. Receiving gradients $\mathbf{g}_1, \cdots, \mathbf{g}_m$ from all the $m$ workers.
2. Computing $\mathbf{g} = \mathbf{g}_1 + \cdots + \mathbf{g}_m$.
3. Updating model parameters:

$$\mathbf{w} \leftarrow \mathbf{w} - \alpha \cdot \mathbf{g}.$$

# Federated Averaging Algorithm

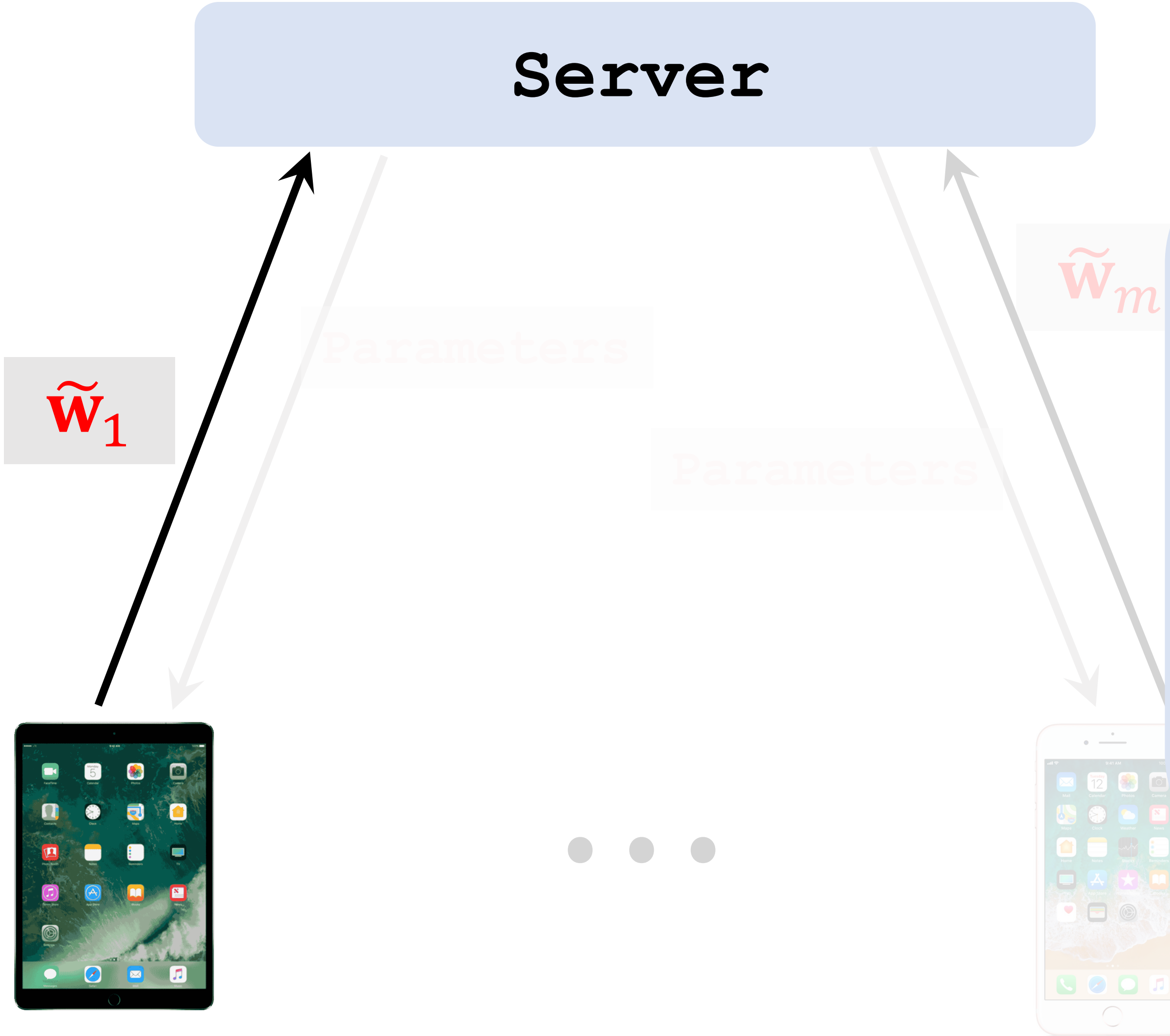


# Federated Averaging Algorithm

**The $i$-th worker performs:**

1. Receiving model parameters $\mathbf{w}$ from the server.
2. Repeating the followings:
   a) Using $\mathbf{w}$ and its local data to compute gradient $\mathbf{g}$.
   b) Local update: $\mathbf{w} \leftarrow \mathbf{w} - \alpha \cdot \mathbf{g}$.
3. Sending $\tilde{\mathbf{w}}_i = \mathbf{w}$ to the server.

# Federated Averaging Algorithm



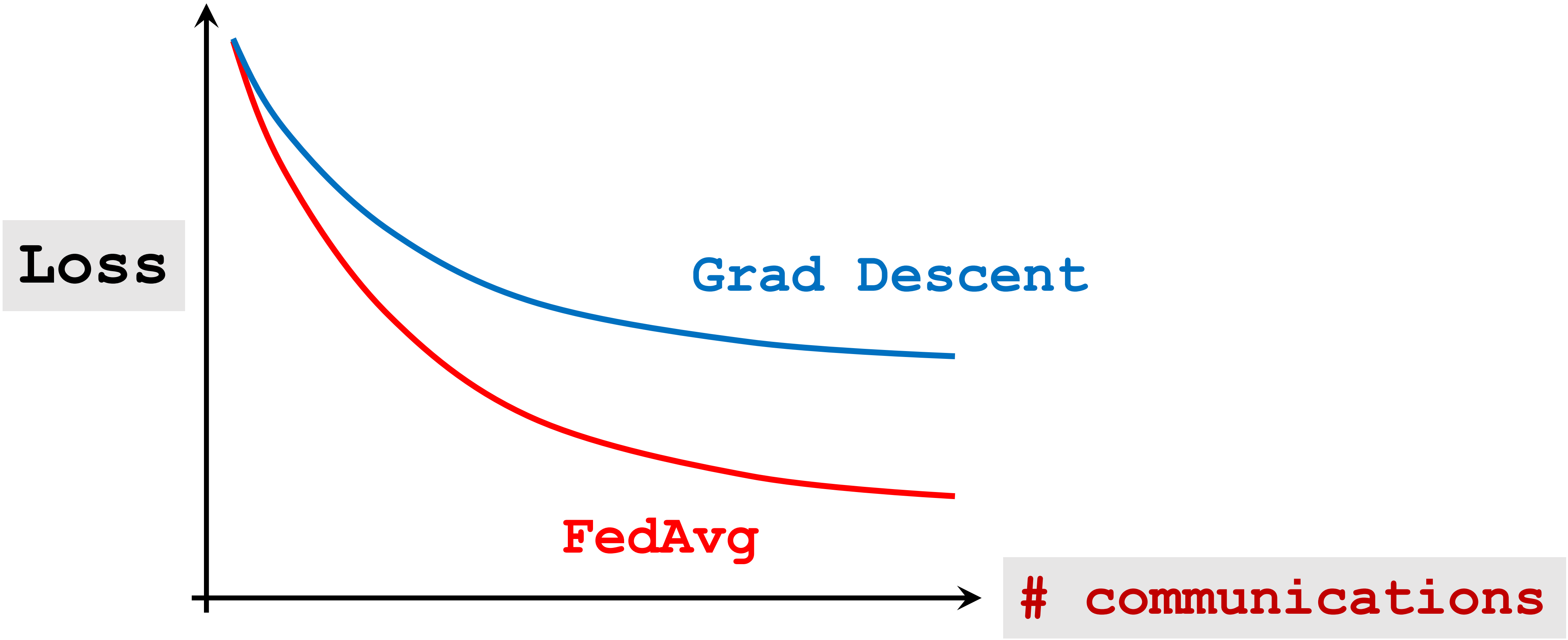

**The server performs:**

1. Receiving $\tilde{\mathbf{w}}_1, \cdots, \tilde{\mathbf{w}}_m$ from all the $m$ workers.
2. Updating model parameters:

$$\mathbf{w} \leftarrow \frac{1}{m}\left(\tilde{\mathbf{w}}_1 + \cdots + \tilde{\mathbf{w}}_m\right).$$

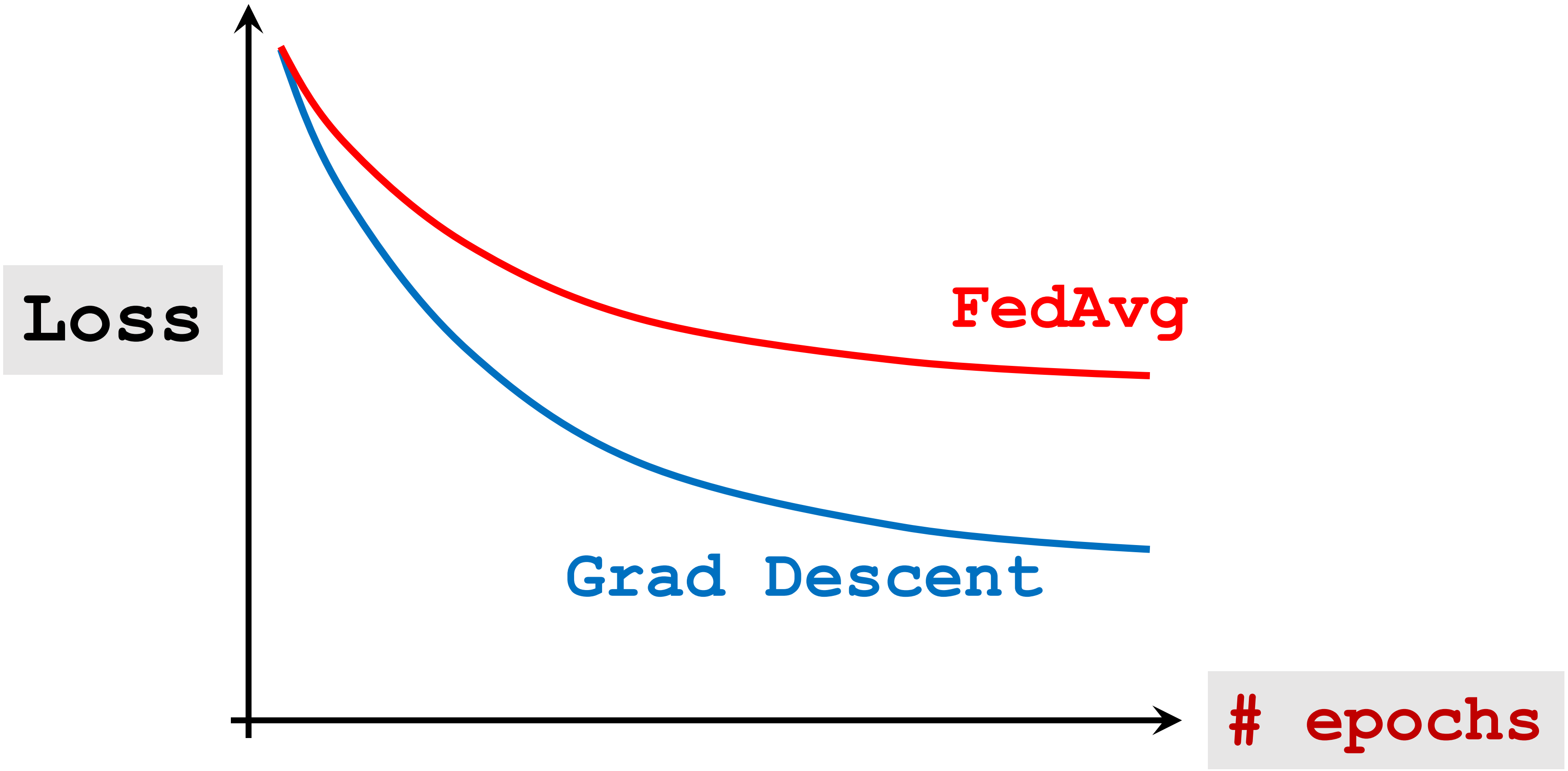
# Computation vs. Communication

Measured by # communications, Federated Averaging is faster.



# Computation vs. Communication

Measured by # epochs, Federated Averaging is slower.



# Convergence of FedAvg

- The original paper [1] does not have theory.
- Paper [2] proved FedAvg converges for IID data.
- Paper [3] is the first to prove FedAvg (with SGD) converges for non-IID data.
- Paper [4] proved FedAvg (with GD) converges for non-IID data.

**References**


1. McMahan and others: Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017.
2. Stich: Local SGD converges fast and communicates little. In *ICLR*, 2018.
3. Li and others: On the convergence of FedAvg on non-IID data. *arXiv*, 2019.
4. Khaled and others: First analysis of local GD on heterogeneous data. *arXiv*, 2019.

# Communication-Efficient Algorithms

Communication-efficient algorithms for distributed learning, e.g.,

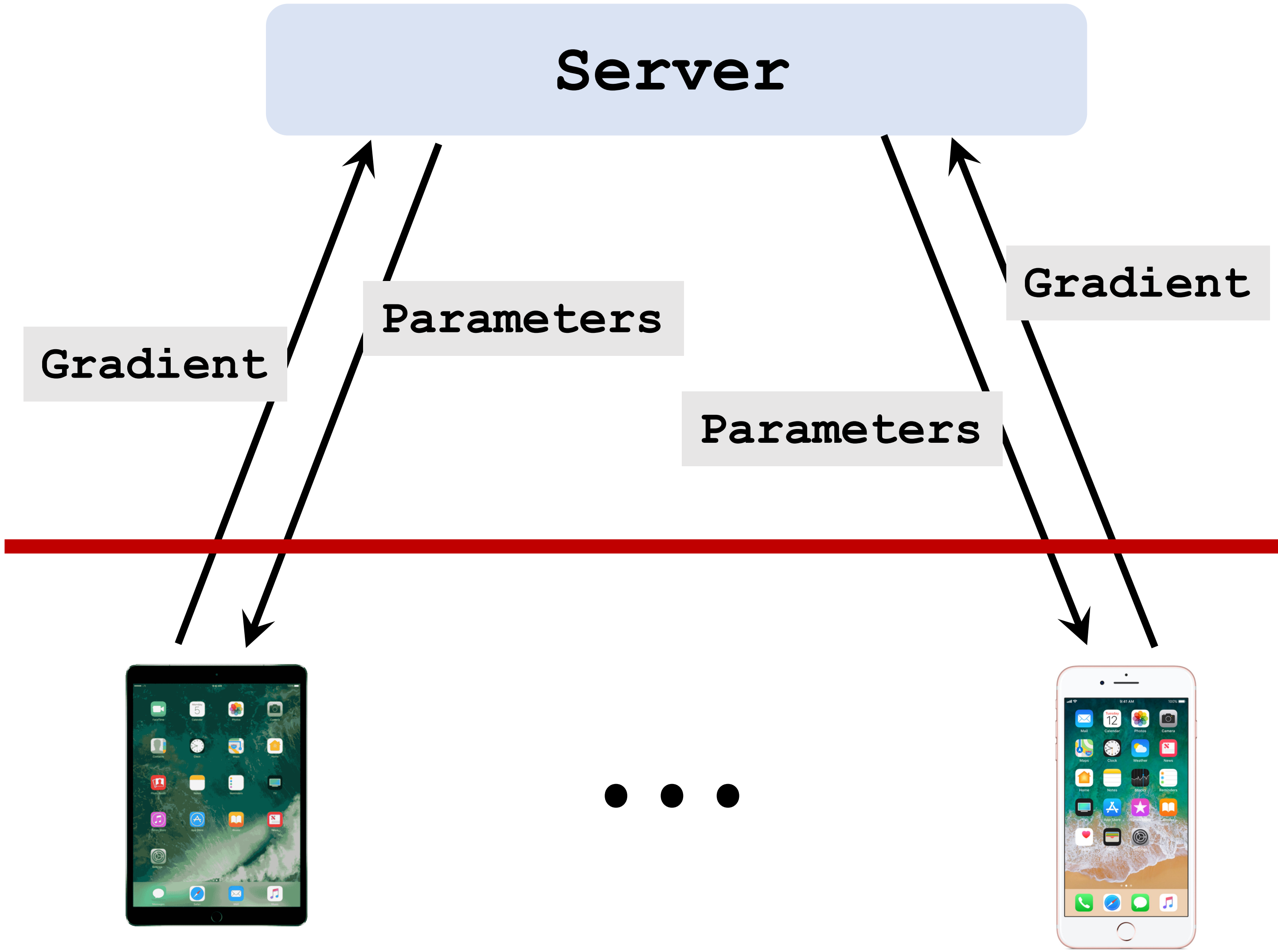
- Approximate Newton's algorithms [1, 2, 3].
- Primal-dual algorithms [4].
- One-shot averaging [5].

**Reference:**


1. Shamir, Srebro, & Zhang: Communication efficient distributed optimization using an approximate Newton-type method. In *ICML*, 2014.
2. Wang and others: GIANT: Globally improved approximate newton method for distributed optimization. In *NIPS*, 2018.
3. Mahajan and others: An efficient distributed learning algorithm based on effective local functional approximations. *Journal of Machine Learning Research*, 2019.
4. Smith and others: CoCoA: A general framework for communication-efficient distributed optimization. *Journal of Machine Learning Research*, 2018.
5. Zhang, Duchi, & Wainwright: Communication-efficient algorithms for statistical optimization. *Journal of Machine Learning Research*, 2013.

# Research Direction 2: Privacy

# Is federated learning (FL) safe?




Data do not leave worker nodes.

# Is federated learning (FL) safe?

Gradient carries information in the training data.

- Least squares regression:

$$\min_{\mathbf{w}} \sum_{i=1}^{n} l(\mathbf{w}, \mathbf{x}_i, y_i), \quad \text{where } \; l(\mathbf{w}, \mathbf{x}_i, y_i) = \frac{1}{2}\left(\mathbf{x}_i^T \mathbf{w} - y_i\right)^2.$$

- Stochastic gradient:

$$\mathbf{g}_i = \frac{\partial\, l(\mathbf{w}, \mathbf{x}_i, y_i)}{\partial\, \mathbf{w}} = \left(\mathbf{x}_i^T \mathbf{w} - y_i\right) \mathbf{x}_i.$$

# Is federated learning (FL) safe?

- If an ML model is useful, it must reveal information about the data on which it was trained [1].
- Training data can be reversely inferred from the model [2].
- In FL, gradients and model parameters leak users' data [1, 3].


**References**

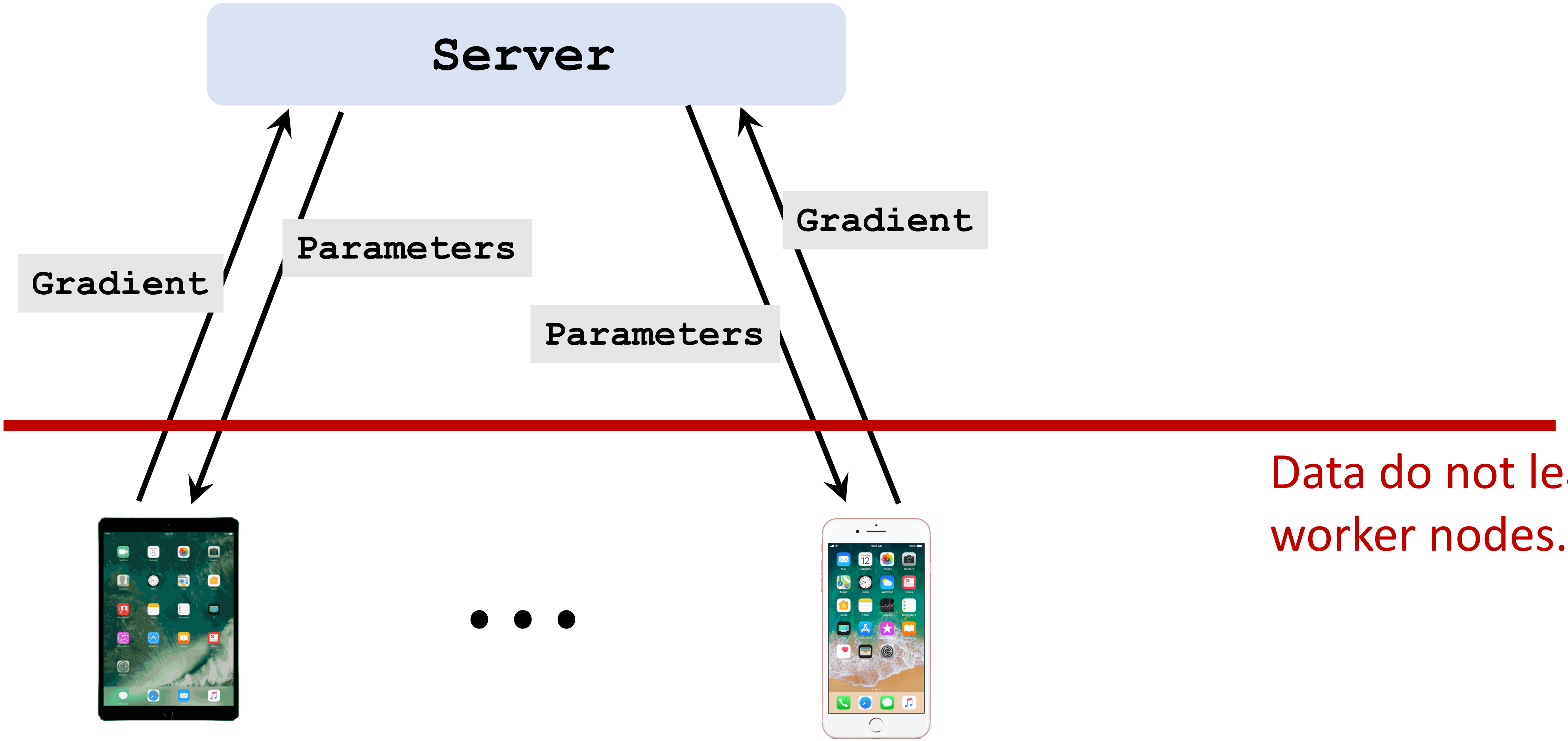
1. Melis et al. Exploiting Unintended Feature Leakage in Collaborative Learning. In *IEEE Symposium on Security & Privacy*, 2019.
2. Fredrikson et al. Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures. In *CCS*, 2015.
3. Hitaj et al. Deep models under the GAN: information leakage from collaborative deep learning. In *ACM SIGSAC Conference on Computer and Communications Security*, 2017.

# How is privacy disclosed?


**References**

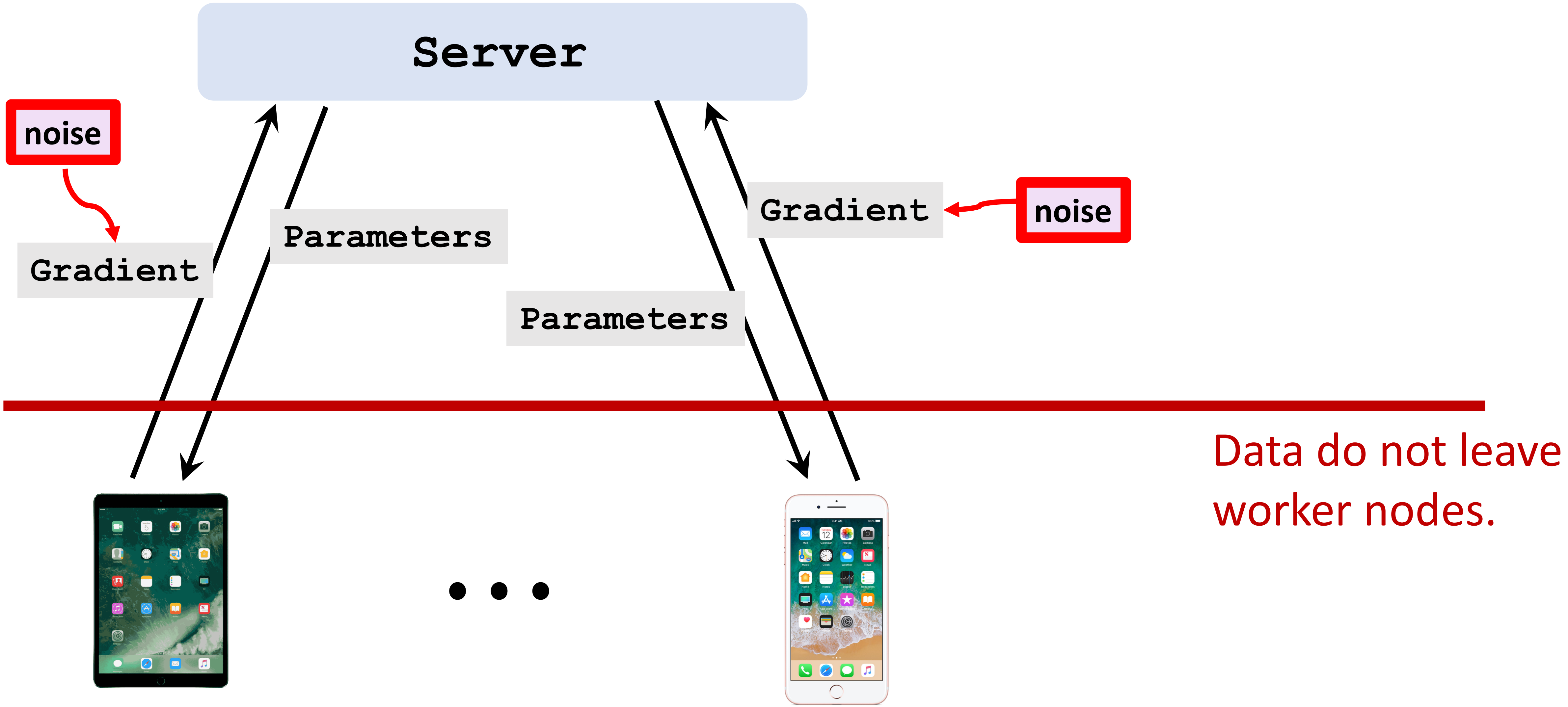
1. Melis et al. Exploiting Unintended Feature Leakage in Collaborative Learning. In *IEEE Symposium on Security & Privacy*, 2019.

# Can the attacks be defended?



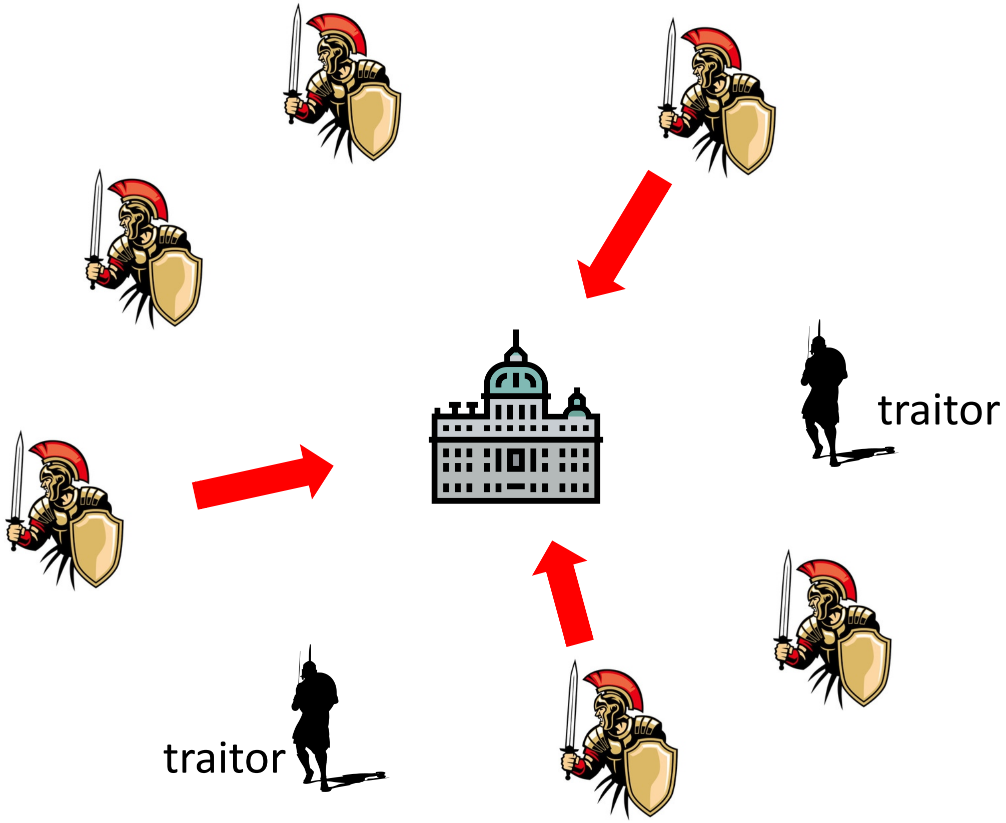

Data do not leave worker nodes.

# Can the attacks be defended?



# Research Direction 3: Adversarial Robustness

# Byzantine Generals Problem




**Reference**

- Lamport, Shostak, & Pease: The Byzantine Generals Problem. *ACM Transactions on Programming Languages and Systems*, 1982.

# Attacks on Federated Learning

- Attack 1: Data poisoning attack [1].
- Attack 2: Model poisoning attack [2].

**References**

1. Shafah and others: Poison frogs! targeted clean-label poisoning attacks on neural networks. In *NIPS*, 2018.
2. Bhagoji and others: Analyzing federated learning through an adversarial lens. In *ICML*, 2019.

# Attacks on Federated Learning

- Attack 1: Data poisoning attack [1].
- Attack 2: Model poisoning attack [2].

- Defense 1: Server check validation accuracy.
- Defense 2: Server check gradient statistics.
- Defense 3: Byzantine-tolerant aggregation [3, 4, 5].

**References**


1. Shafah and others: Poison frogs! targeted clean-label poisoning attacks on neural networks. In *NIPS*, 2018.
2. Bhagoji and others: Analyzing federated learning through an adversarial lens. In *ICML*, 2019.
3. Blanchard, Guerraoui, & Stainer: Machine learning with adversaries: Byzantine tolerant gradient descent. In *NIPS*, 2017.
4. Chen, Su, & Xu: Distributed statistical machine learning in adversarial settings: Byzantine gradient descent. *In Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 2017.
5. Yin and others: Byzantine-robust distributed learning: Towards optimal statistical rates. In *ICML*, 2018.

# Summary

# What is federated learning (FL)?

- FL is a kind of distributed learning.
- Objective: jointly learn a model without sharing data.
- FL has unique challenges, e.g.,
  - non-IID data,
  - slow communication.

# Research Directions

- Direction 1: Communication-efficient algorithms.
- Direction 2: Defense against privacy leakage.
- Direction 3: Robustness to Byzantine faults.

# Thank you!