Asynchronous Federated Learning at Scale

Software Engineering for Machine Learning Applications (SEMLA) International Symposium 2022

Mike Rabbat
Research Scientist & Manager
Meta AI - FAIR

Meta AI

Collaborators and Colleagues

Dzmitry Huba

Mani Malek

Kshitiz Malik

Jesik Min

Ilya Mironov

John Nguyen

Anthony Shoumikhin

Harish Srinivas

Pavel Ustinov

Kaikai Wang

Carole-Jean Wu

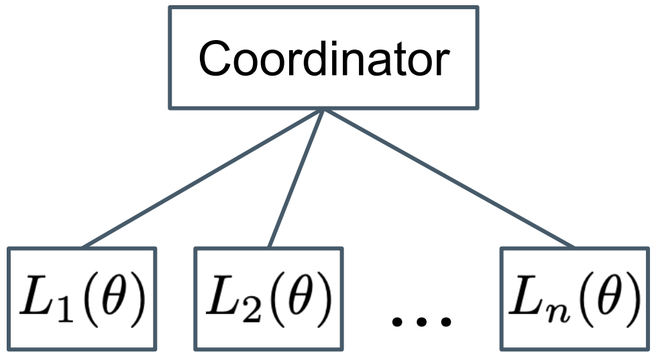
Ashkan Yousefpour

Hongyuan Zhan

Ruiyu Zhu

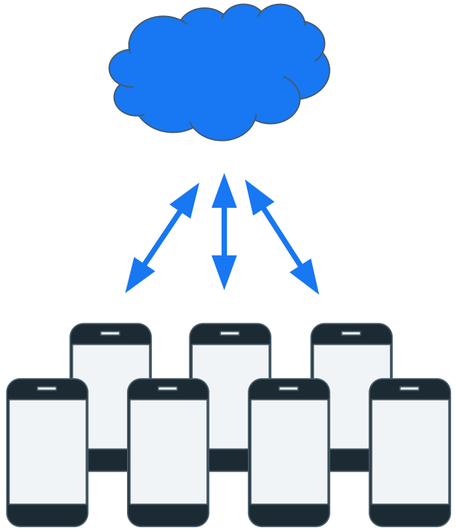
Federated Optimization

- Not just another parallel/distributed optimization problem!



- The primary difference: Privacy

- All data is not accessible centrally
  - Cannot permute data across workers to give iid subsets
  - Cannot communicate updates (gradients) transparently



Privacy and gradient inversion attacks

Model updates leak unintended information about clients' training data.

Original image Recovered

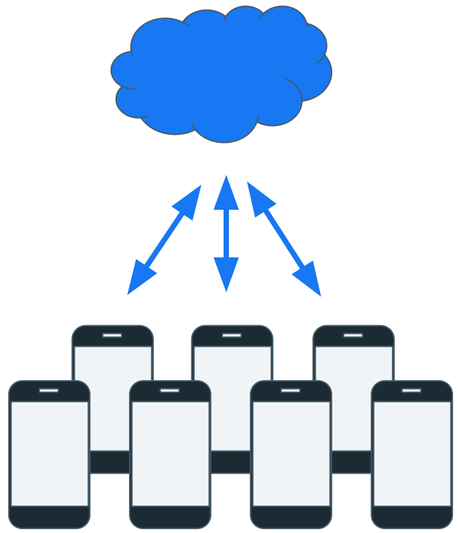
J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller. "Inverting gradients–how easy is it to break privacy in federated learning?" NeurIPS 2020

Solution: Use privacy-enhancing technologies:

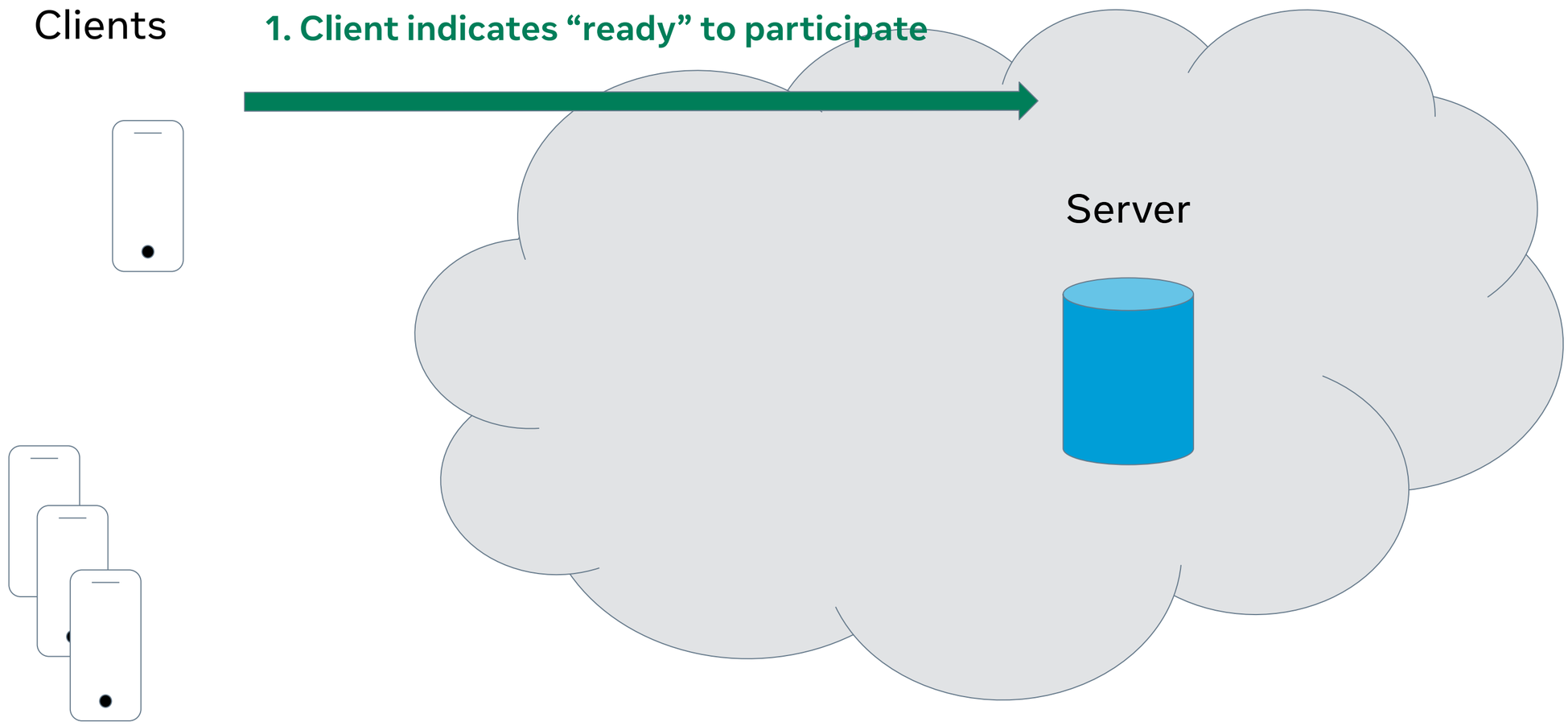
1. **Secure Aggregation** of model updates from multiple clients
2. Train using **Differential Privacy**

Federated Optimization

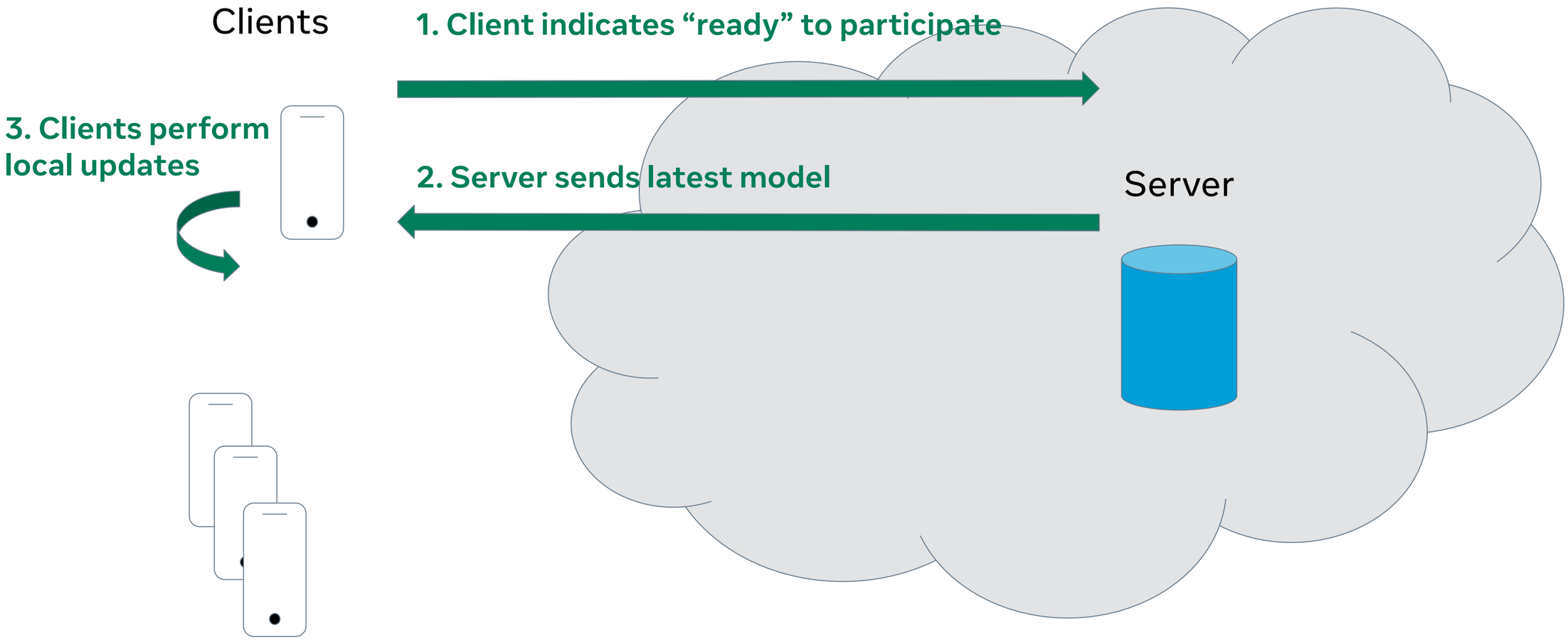
- The key difference: Privacy
  - Use Privacy-Enhancing Technologies
- All data is not accessible centrally
  - Cannot permute data across workers to give iid subsets
  - Cannot communicate updates (gradients) transparently
- Additional challenges:
  - Data imbalance
  - Data heterogeneity
  - Device availability
  - Device capabilities
  - Low-bandwidth communications
  - System heterogeneity



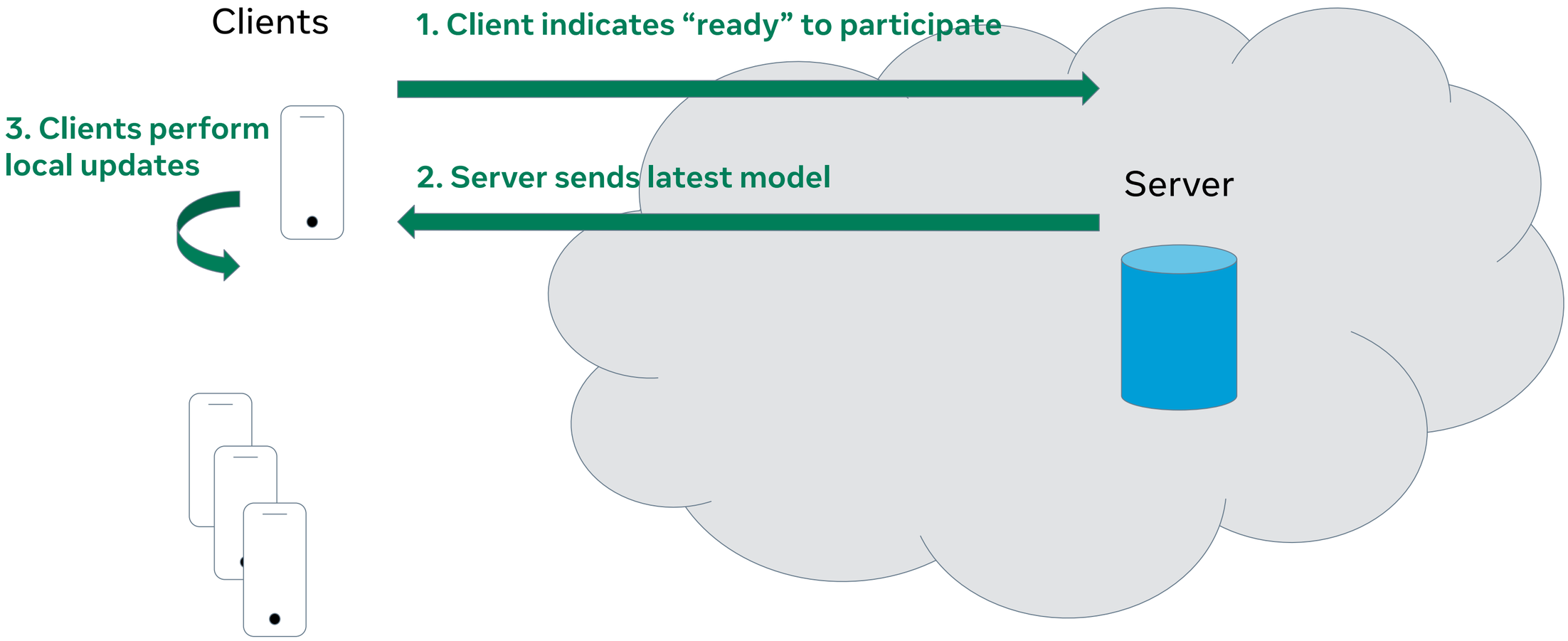
Synchronous Federated Training



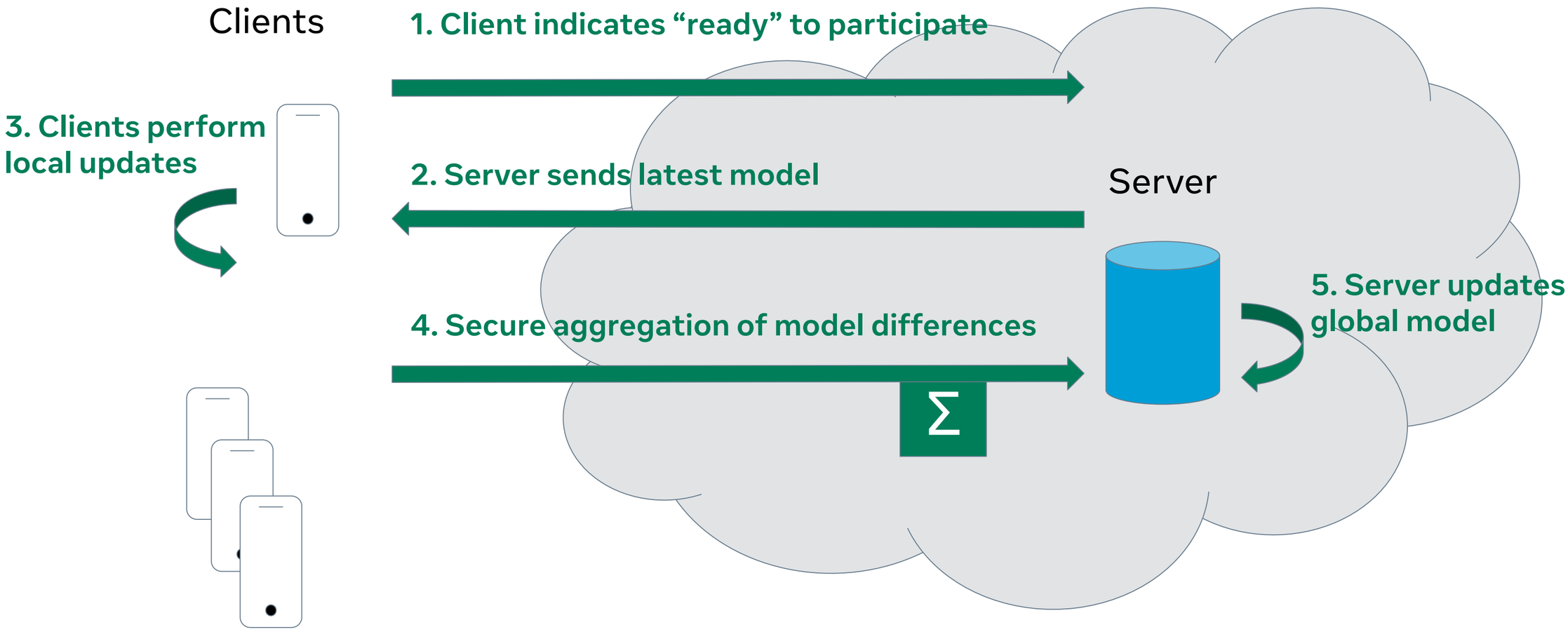
Synchronous FL



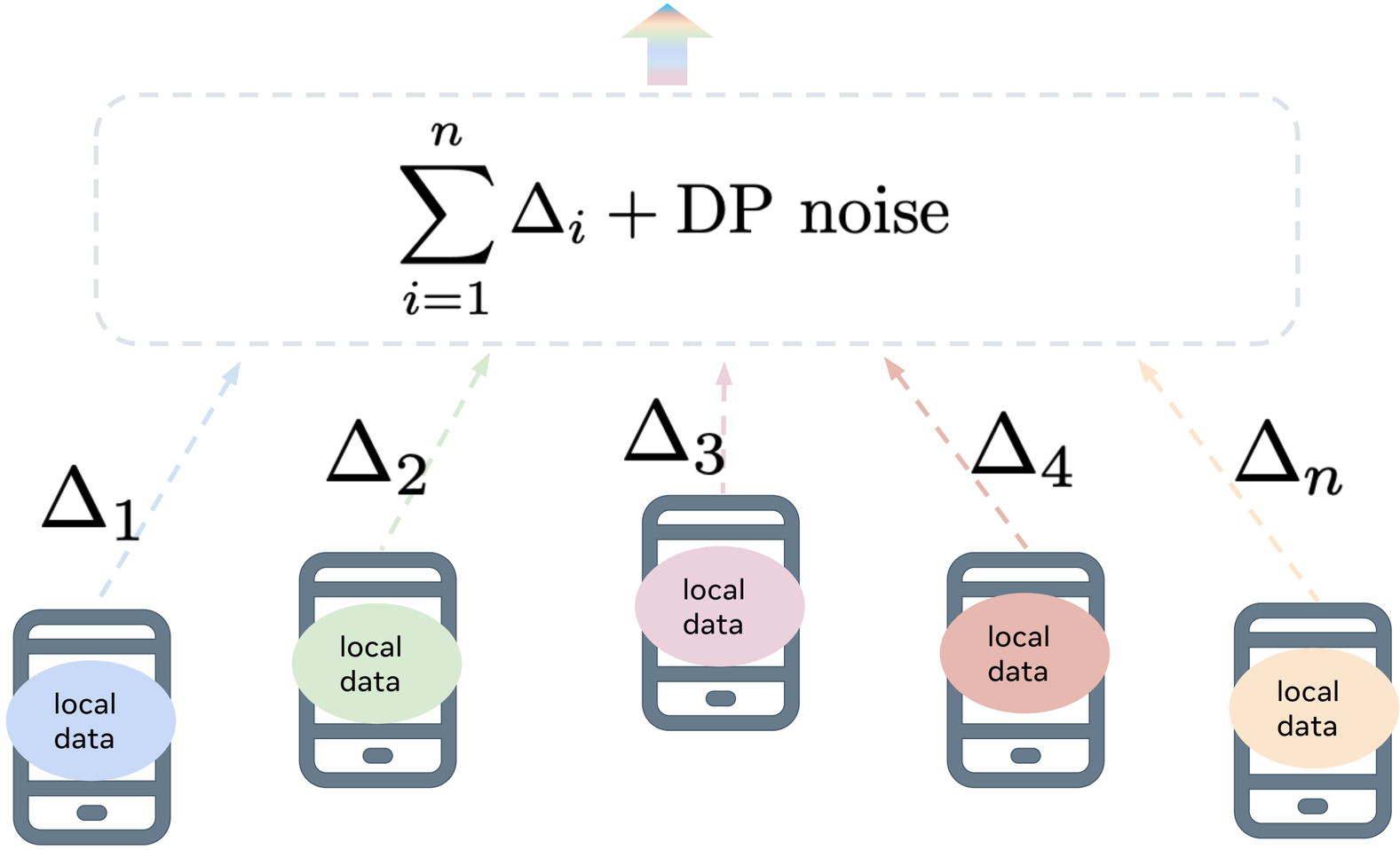
Synchronous FL



Synchronous FL

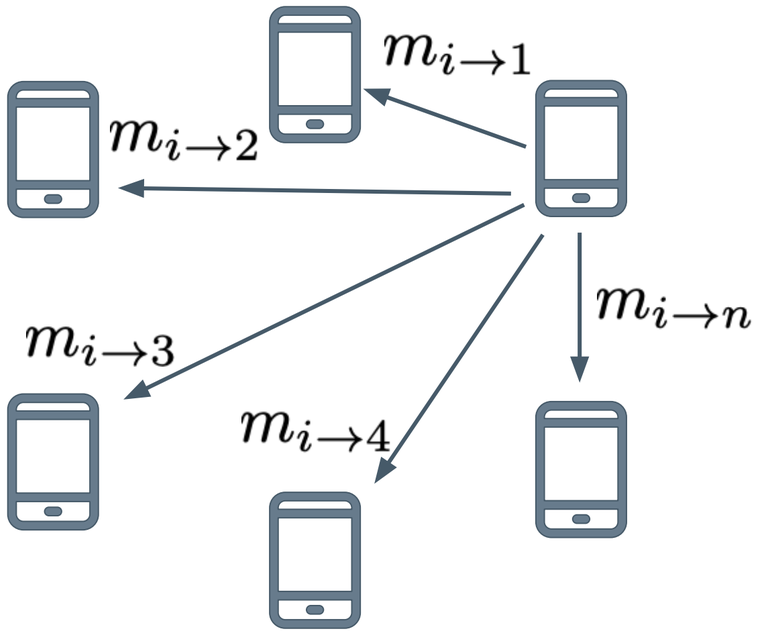


Aggregation Layer: Secure Multi-Party Computation



"Practical Secure Aggregation for Federated Learning on User-Held Data", ACM CCS 2017
Bonawitz, Ivanov, Kreuter, Marcedone, McMahan, Patel, Ramage, Segal, Seth

Bonawitz et al. ‘17 (simplified)



$$\Delta_1 + \sum_j m_{j\to 1}$$

$$\Delta_i + \sum_j m_{j\to i}$$

$$\Delta_n + \sum_j m_{j\to n}$$

$$\sum_i \Delta_i$$

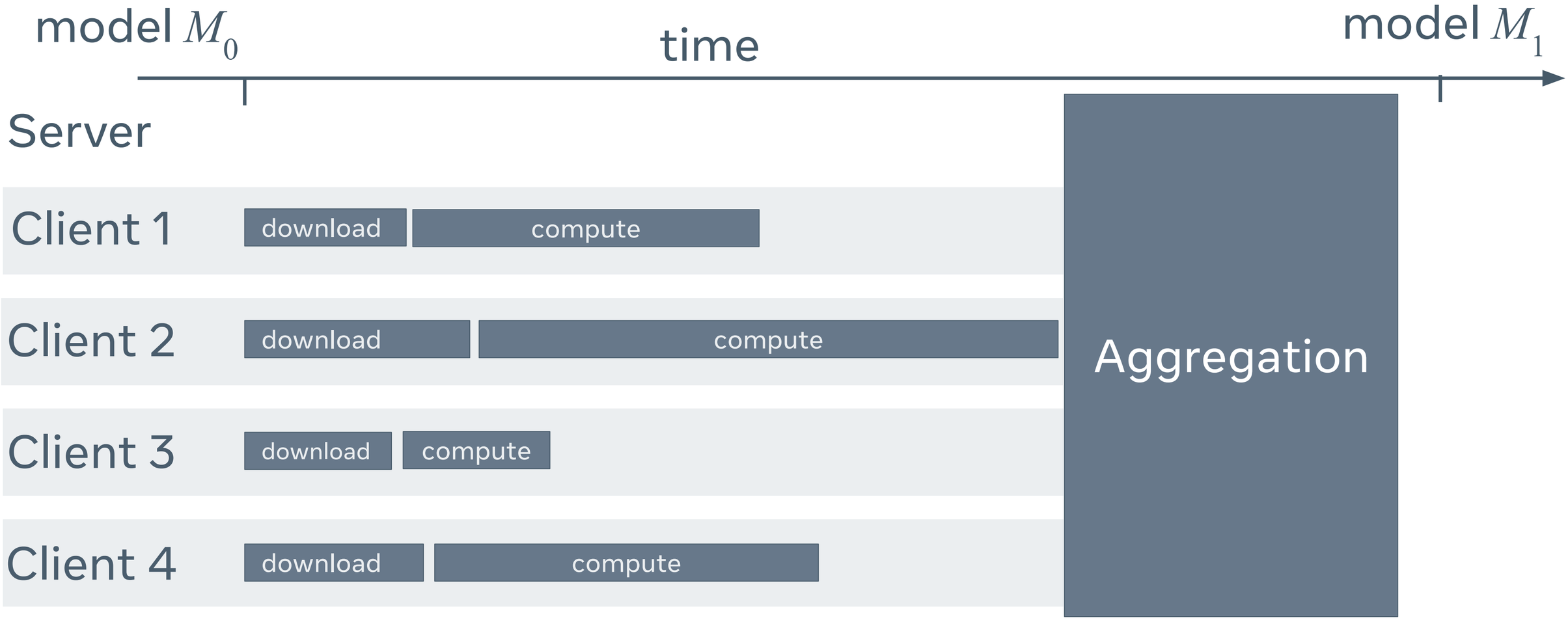
if $m_{i\to j} = -m_{j\to i}$

“Practical Secure Aggregation for Federated Learning on User-Held Data”, ACM CCS 2017
Bonawitz, Ivanov, Kreuter, Marcedone, McMahan, Patel, Ramage, Segal, Seth

Reality Check

1. ML deals in floating-point numbers; MPC assumes integers
2. Clients are mobile devices: can only communicate through the server
3. Clients are flaky: they can (and will) drop out mid-way through the protocol
4. Computation is iterative, **low latency is key**

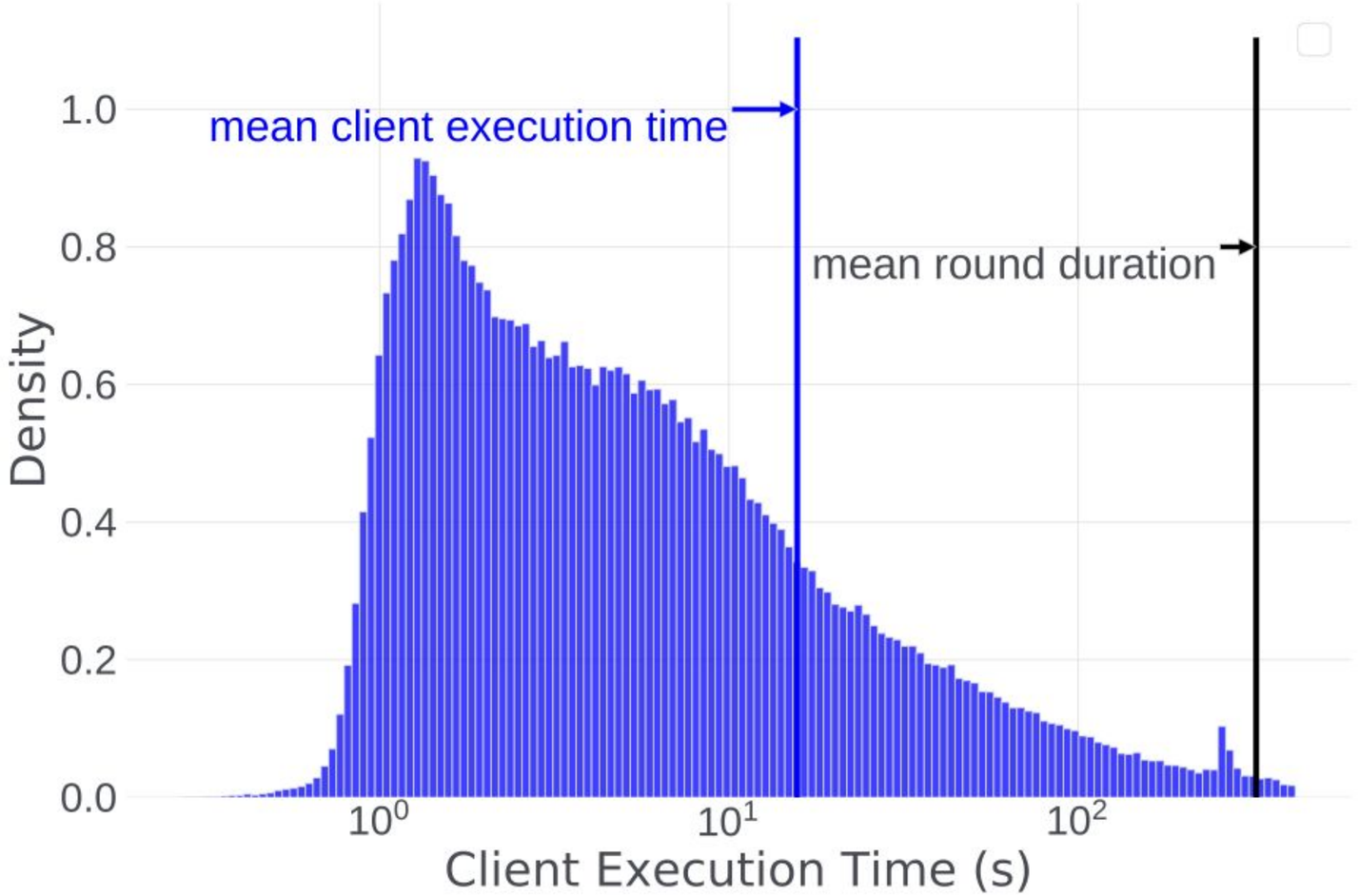
Synchronous FL



Reality Check

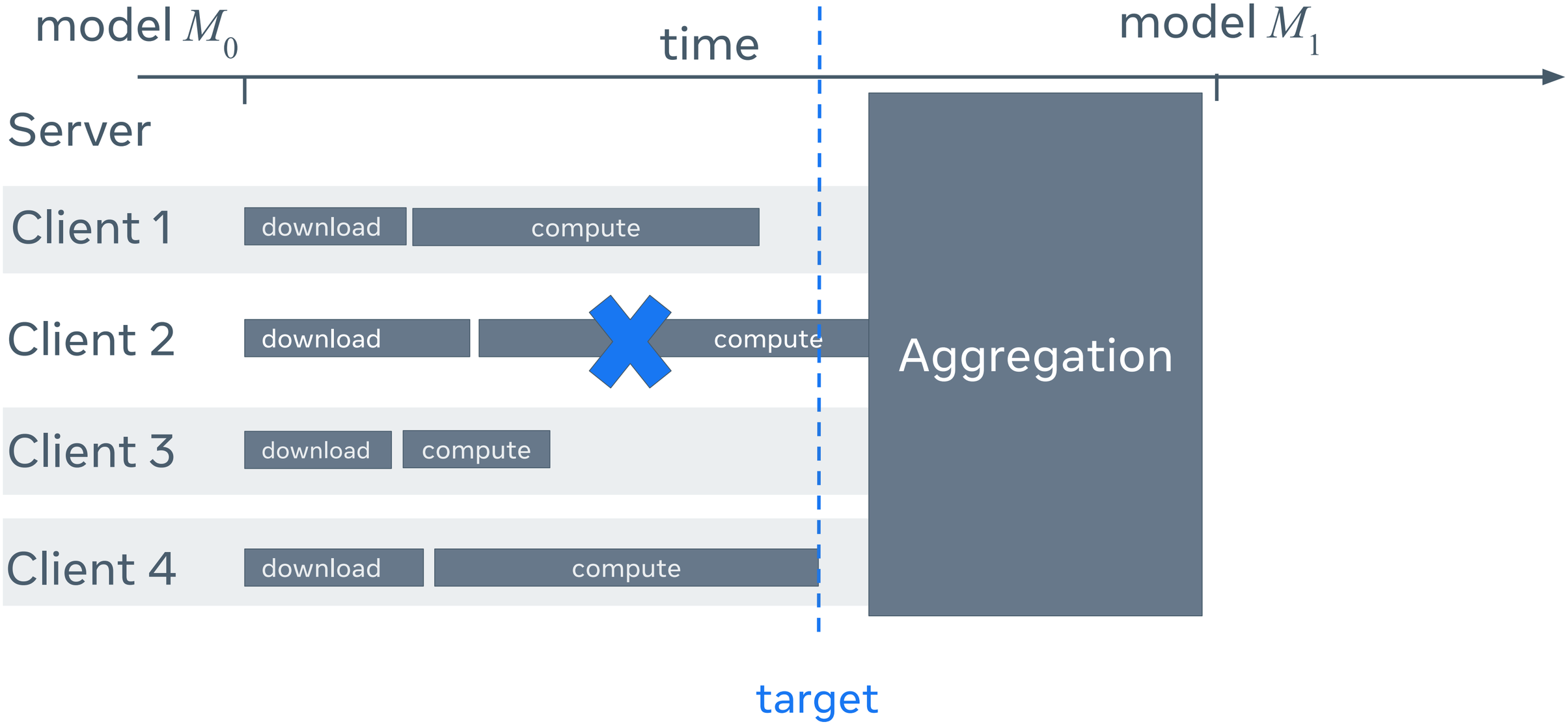
1. ML deals in floating-point numbers; MPC assumes integers
2. Clients are mobile devices: can only communicate through the server
3. Clients are flaky: they can (and will) drop out mid-way through the protocol
4. Computation is iterative, low latency is key
5. **Clients are heterogeneous**

Stragglers



“Papaya: Practical, Private, and Scalable Federated Learning”, https://arxiv.org/abs/2111.04877
Huba, Nguyen, Malik, Zhu, R, Yousefpour, Wu, Zhan, Ustinov, Srinivas, Wang, Shoumikhin, Min, Malek

Dealing with Stragglers



Reality Check

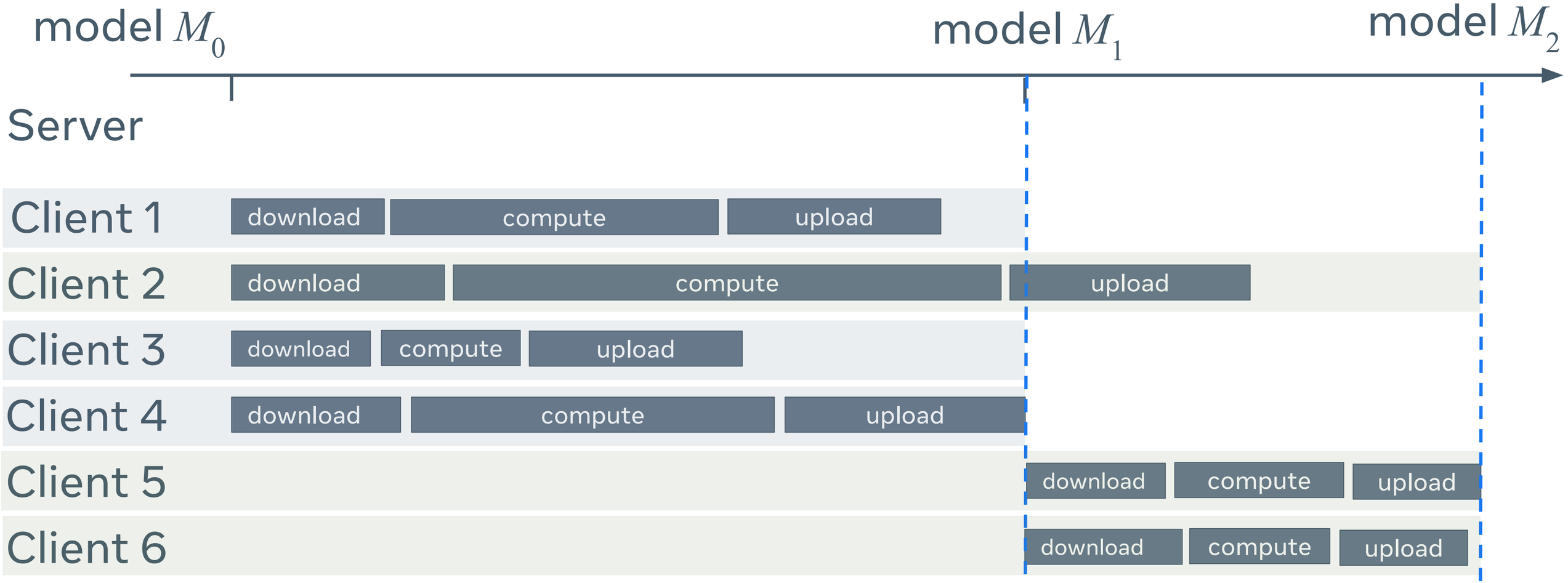
1. ML deals in floating-point numbers; MPC assumes integers
2. Clients are mobile devices: can only communicate through the server
3. Clients are flaky: they can (and will) drop out mid-way through the protocol
4. Computation is iterative, low latency is key
5. Clients are heterogeneous
6. **Clients are different in wonderful, interesting ways**

Client diversity is real

	Time to Train	Average Perplexity	Perplexity of top 25%	Perplexity of top 1%
Basic FL	130 hours	68	67	47
FL with timeout	19 hours	73	73	73

“Papaya: Practical, Private, and Scalable Federated Learning”, https://arxiv.org/abs/2111.04877
Huba, Nguyen, Malik, Zhu, Rabbat, Yousefpour, Wu, Zhan, Ustinov, Srinivas, Wang, Shoumikhin, Min, Malek

Asynchronous FL: FedBuff (Nguyen et al. 21)



Dutta et al., “Slow and stale gradients can win the race,” AISTATS 2018
Nguyen et al. “Federated Learning with Buffered Asynchronous Aggregation”, AISTATS 2022

FedBuff - Buffered Asynchronous Aggregation

- Clients potentially started computing from different models
  - Late updates are still valuable but should be weighted accordingly

- Aggregate responses from clients in a secure buffer, only release to the server once enough responses have been received

- How to implement a secure buffer?

Nguyen et al. “Federated Learning with Buffered Asynchronous Aggregation”, AISTATS 2022

Trusted Execution Environments (TEEs)

Secure area within the main processor which guarantees confidentiality and integrity against the host OS

What happens in the TEE stays in the TEE*

*Under certain strong assumptions

Several commercial implementations
(Intel SGX, ARM TrustZone, AMD SEV, ...)

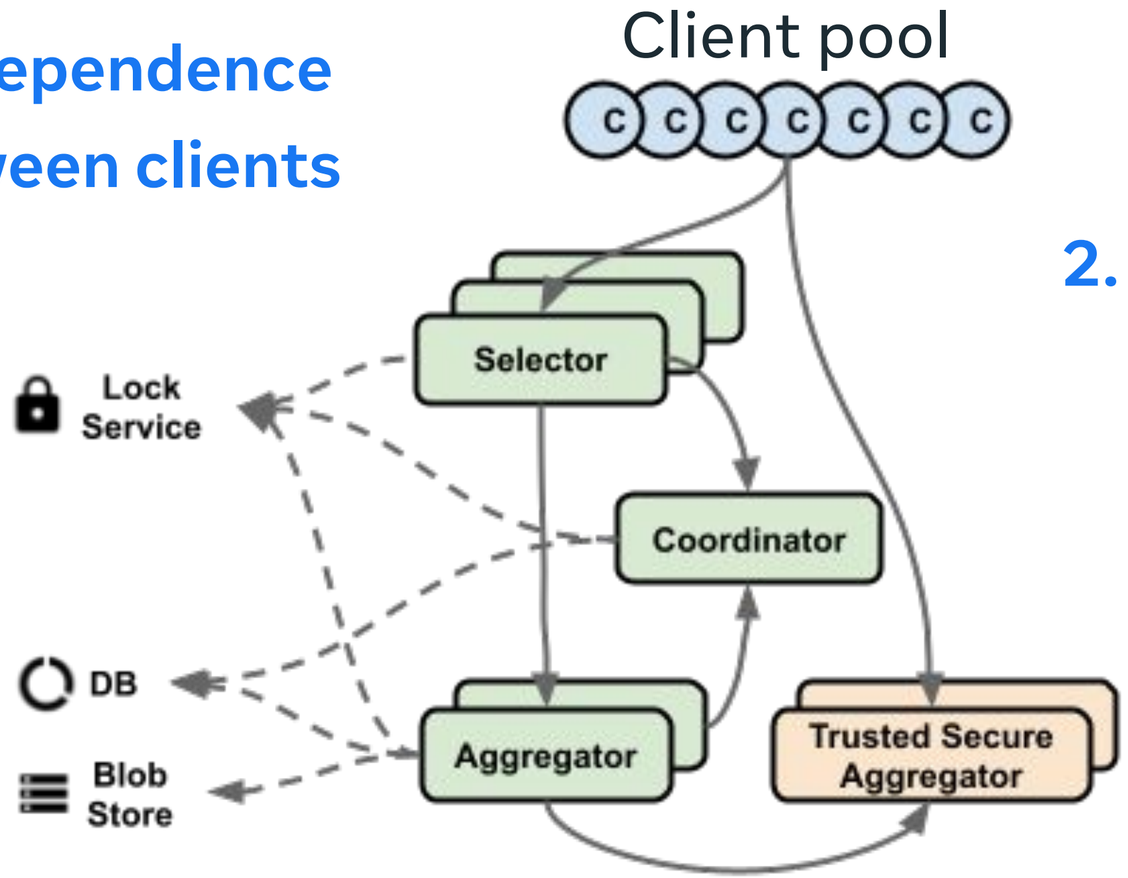
FedBuff

	Time to train	Average perplexity	Perplexity of top 25%	Perplexity of top 1%
Basic FL	130 hours	68	67	47
FL with cutoffs	19 hours	73	73	73
FedBuff	**18 hours**	**57**	**56**	**39**

“Papaya: Practical, Private, and Scalable Federated Learning”, https://arxiv.org/abs/2111.04877
Huba, Nguyen, Malik, Zhu, Rabbat, Yousefpour, Wu, Zhan, Ustinov, Srinivas, Wang, Shoumikhin, Min, Malek

System Design Highlights

1. **No dependence between clients**



2. **Fast client replacement**

3. **Fast model aggregation**

“Papaya: Practical, Private, and Scalable Federated Learning”, https://arxiv.org/abs/2111.04877
Huba, Nguyen, Malik, Zhu, Rabbat, Yousefpour, Wu, Zhan, Ustinov, Srinivas, Wang, Shoumikhin, Min, Malek

Summary

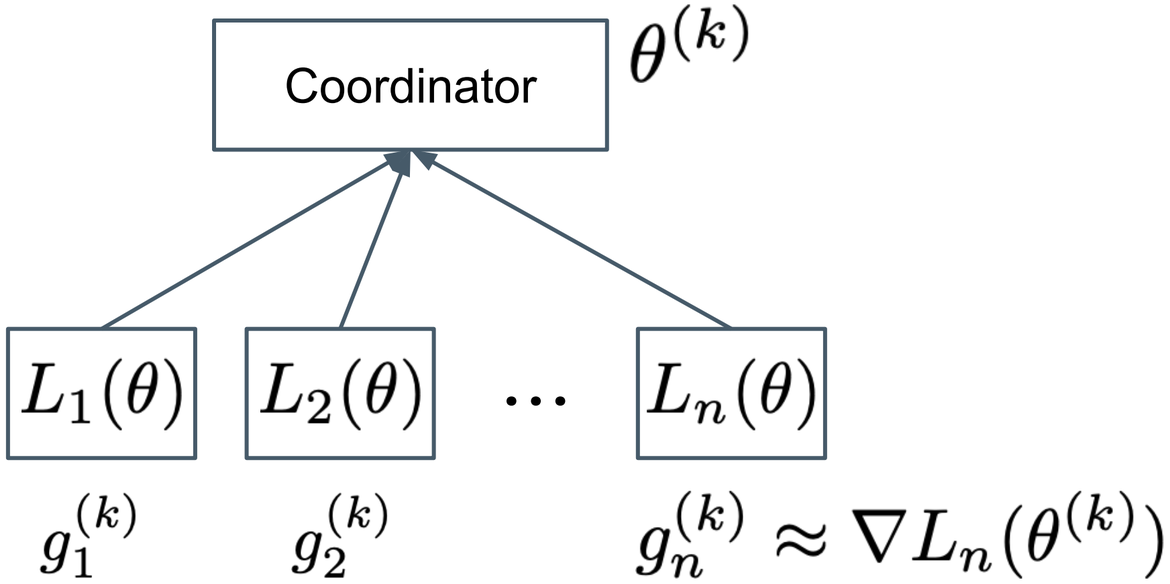
- Heterogeneity gives rise to very real challenges in FL
- How we deal with it has implications
  - Timeouts for slow users
  - Excluding older, less capable devices (Maeng et al., 2022)
- Asynchronous secure aggregation
  - TEE Secure buffer implementation
  - Alternative MPC protocol of So, Ali, Güler & Avestimehr (2021)

FedBuff - https://arxiv.org/abs/2106.06639
Papaya - https://arxiv.org/abs/2111.04877
FLSim - https://github.com/facebookresearch/FLSim


Meta AI

Backup Slides

Local Update Methods in One Slide



Recall Parallel Gradient Method: Workers compute a (stochastic) gradient and send it back to the coordinator

---

$$\theta_i^{(k,0)} = \theta^{(k)}$$

$$\theta_i^{(k,s+1)} = \theta_i^{(k,s)} - \alpha_l g_i^{(k,s)}, \quad s = 1, \ldots, S-1$$

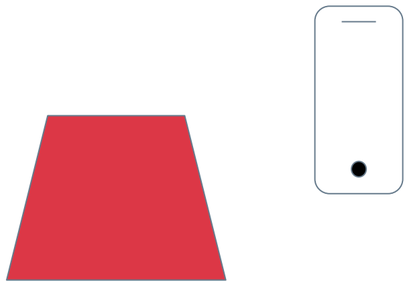
$$\Delta_i^{(k)} = \theta_i^{(k,S)} - \theta^{(k)}$$ "Model update"

Local update methods (FedAvg/LocalSGD):
Take several steps locally, send update to the coordinator

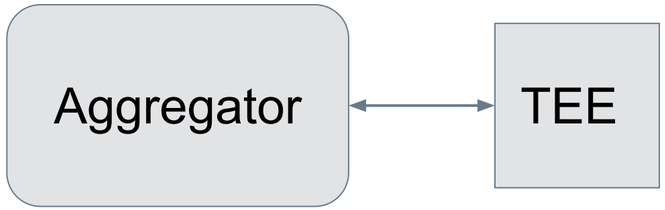
Convergence analysis in

- S Stich (2018)
- Wang & Joshi (2018)
- Adaptive generalization in Reddi et al. (2020)

Asynchronous Secure Aggregation

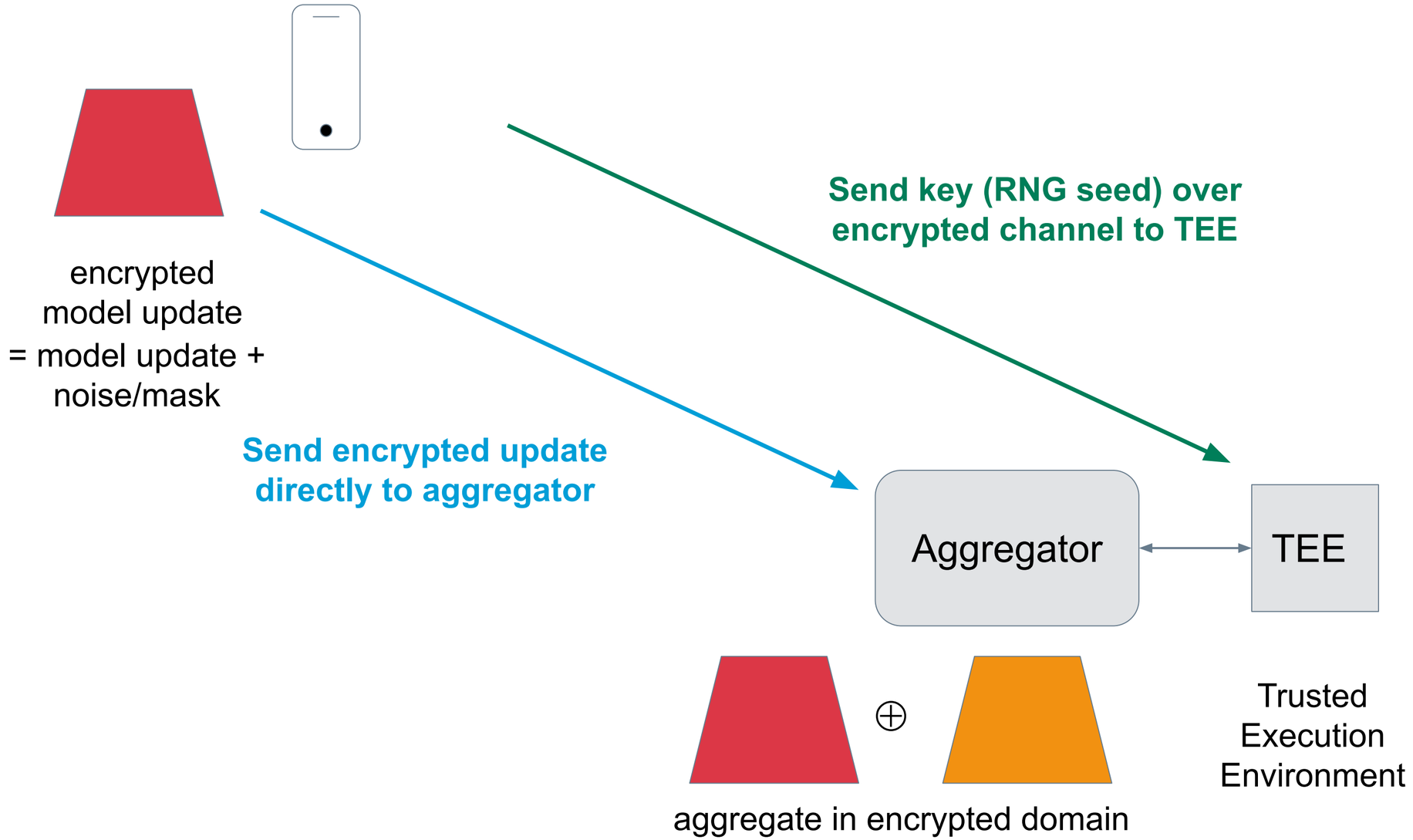


encrypted
model update
= model update +
noise/mask

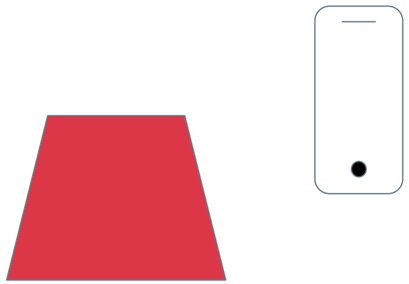


Trusted
Execution
Environment

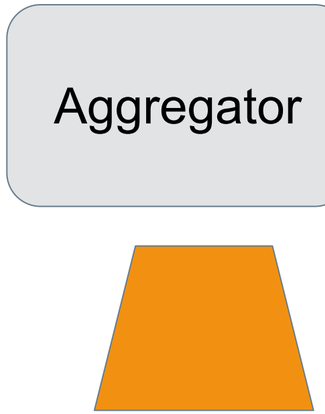
Asynchronous Secure Aggregation



Asynchronous Secure Aggregation



encrypted
model update
= model update +
noise/mask



Unencrypt aggregated update with mask from TEE

Trusted
Execution
Environment