

istics of client devices become influential and dominant to the global model production, as the central server deals with all the client devices that generate non-IID data.

Solution: A hierarchical aggregator adds edge servers between the central server and client devices. The combination of server-edge-client architecture can improve both computation and communication efficiency of the federated model training process. Edge servers collect local models from the nearest client devices, a subset of all the client devices. After every k_1 round of local training on each client, each edge server aggregates its clients' models. After every k_2 edge model aggregations, the cloud server aggregates all the edge servers' models, which means the communication with the central server happens every $k_1 k_2$ local updates [28].

Consequences:

Benefits:

- *Communication efficiency.* The hierarchical aggregators speed up the global model aggregation and improve communication efficiency.
- *Scalability.* Adding an edge layer helps to scale the system by improving the system's ability to handling more client devices.
- *Data heterogeneity and non-IID reduction.* The partial aggregation in a hierarchical manner aggregates local models that have similar data heterogeneity and non-IIDness before the global aggregation on the central server. This greatly reduces the effect of data heterogeneity and non-IIDness on global models.
- *Computation and storage efficiency.* The edge devices are rich with computation and storage resources to perform partial model aggregation. Furthermore, edge devices are nearer to the client devices which increase the model aggregation and computation efficiency.

Drawbacks:

- *System reliability.* The failure of edge devices may cause the disconnection of all the client devices under those edge servers and affect the model training process, model performance, and system reliability.
- *System security.* Edge servers could become security breach points as they have lower security setups than the central server and the client devices. Hence, they are more prone to network security threats or becoming possible points-of-failure of the system.

Related patterns: *Client Registry, Client Cluster, Model Co-versioning Registry*

Known uses:

- *HierFAVG* is an algorithm that allows multiple edge servers to perform partial model aggregation incrementally from the collected updates from the client devices.

- *Hierarchical Federated Learning (HFL)* enables hierarchical model aggregation in large scale networks of client devices where communication latency is prohibitively large due to limited bandwidth. The *HFL* seeks a consensus on the model and uses edge servers to aggregate model updates from client devices that are geographically near.
- *Federated Learning + Hierarchical Clustering (FL+HC)* is the addition of a hierarchical clustering algorithm to the federated learning system. The cluster is formed according to the data distributions similarity based on the following distance metrics: (1) Manhattan, (2) Euclidean, (3) Cosine distance metrics.
- *Astraea* is a federated learning framework that tackles non-IID characteristics of federated clients. The framework introduces a mediator to the central server and the client devices to balance the skewed client data distributions. The mediator performs the z-score-based data augmentation and downsampling to relieve the global imbalanced of training data.

3.4.4. Pattern 14: Secure Aggregator

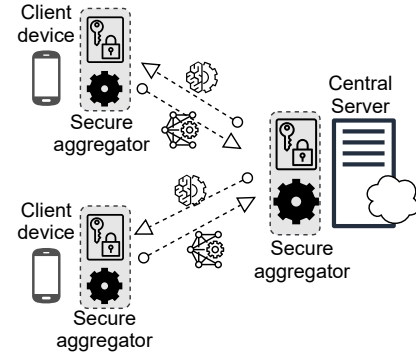


Figure 17: Secure Aggregator.

Summary: A security aggregator manages the model exchange and aggregation security protocols to protect model security. Fig. 17 illustrates the security aggregator on each components with the security protocols embedded in them.

Context: The central server sends global models to any existing or unknown device every round with no data privacy and security protocols that protect the communication from unauthorised access. Furthermore, model parameters contain pieces of private user information that can be inferred by the data-hungry machine learning algorithms.

Problem: There are no security measures to tackle the honest-but-curious and active adversary security threats which exist in federated learning systems.

Forces: The problem requires to balance the following forces:

- *Client device security.* Client device security issues exist when dishonest and malicious client devices join the training process and poison the overall model performance by disrupting the training process or providing false updates to the central server.