Papaya: Federated Analytics Stack

NSDI 2025: Harish Srinivas, Graham Cormode, Mehrdad Honarkhah, Samuel Lurye, Jonathan Hehir, Lunwen He, George Hong, Ahmed Magdy, Dzmitry Huba, Kaikai Wang, Shen Guo, Shoubhik Bhattacharya

@Meta

Federated Analytics enables multiple entities (clients) to collaborate in solving a data analytics problem, under the coordination of a central server or service provider.

Why Federated Analytics?

Data is generated at the edge device.

- Mobile, biosensors, wearables, industrial IoT, smart glasses, cameras, ...

Can data remain at the edge?

Move Compute → Data. Strong desire to gather of metrics and patterns while respecting user privacy.

- Keep data on device and only gather minimal derived data
- Only reveal aggregated summary information to downstream users
- Allow additional privacy via noise addition and random sampling

Challenges in practical systems

Accessible to analysts

Limited expressivity, long iteration time, and deep expertise in federated paradigms

Scalability

Varying population regimes, large number of queries, resource constrained clients and unpredictable load on server.

Complex privacy management

Multiple actors with varying trust levels, diversity of differential privacy models, and evolving data analysis needs.

Differences from Federated Learning

Federated analytics sits alongside federated learning, but has a different focus, requiring different approaches

	Federated Analytics	**Federated Learning**
Primary objective	Varied metrics and patterns	Train ML models
Cohort size	Millions to billions	Thousands
Message size	Up to Kilobytes	Megabytes upwards
Client requirements	Weak (simple statistics)	Moderate (model training)
Number of iteration rounds	One or few	Thousands

Trusted Execution Environments (TEEs)

- Our implementation uses of hardware security in the form of Trusted Execution Environments (TEEs)
- These ensure confidentiality and integrity of the aggregation of client messages
- The TEE uses attestation to prove what code was executed on the client's data
- Code is made available for 3rd party audit, and clients can opt out based on their privacy preferences

Design Overview

Cross device setting

Papaya FA Components

- **Client Runtime**: must be lightweight and performant.
  The runtime comprises a local encrypted store, scheduler, and execution engine.

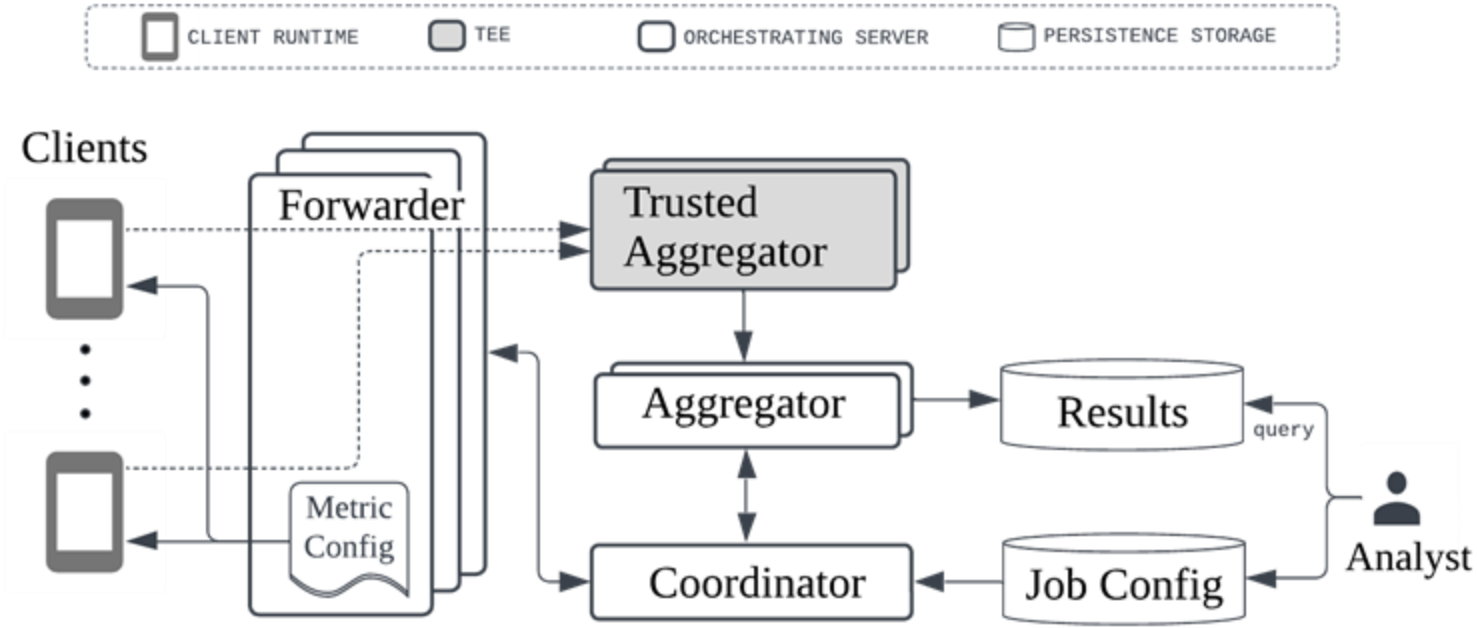
- **Trusted Secure Aggregator (TSA)**: handles a single federated query, running in a TEE
  It applies a Secure Sum and Threshold to aggregate a histogram of the client reports.

- **Untrusted Orchestrating Server (UO)**: to handle the execution of each query.
  It sends queries to clients and receives results from TSA for publication.

Papaya FA Workflow

- **Query creation**: data analysts define their queries and register with the system
- **Client computation**: Client downloads query spec and transforms relevant data from local store
- **Aggregation**: performed by the TSA working with the clients
- **Post processing**: after the TSA has released aggregate private results back to UO.
- **Result publishing**: UO uploads results of the query to a database for consumption.

Overall design

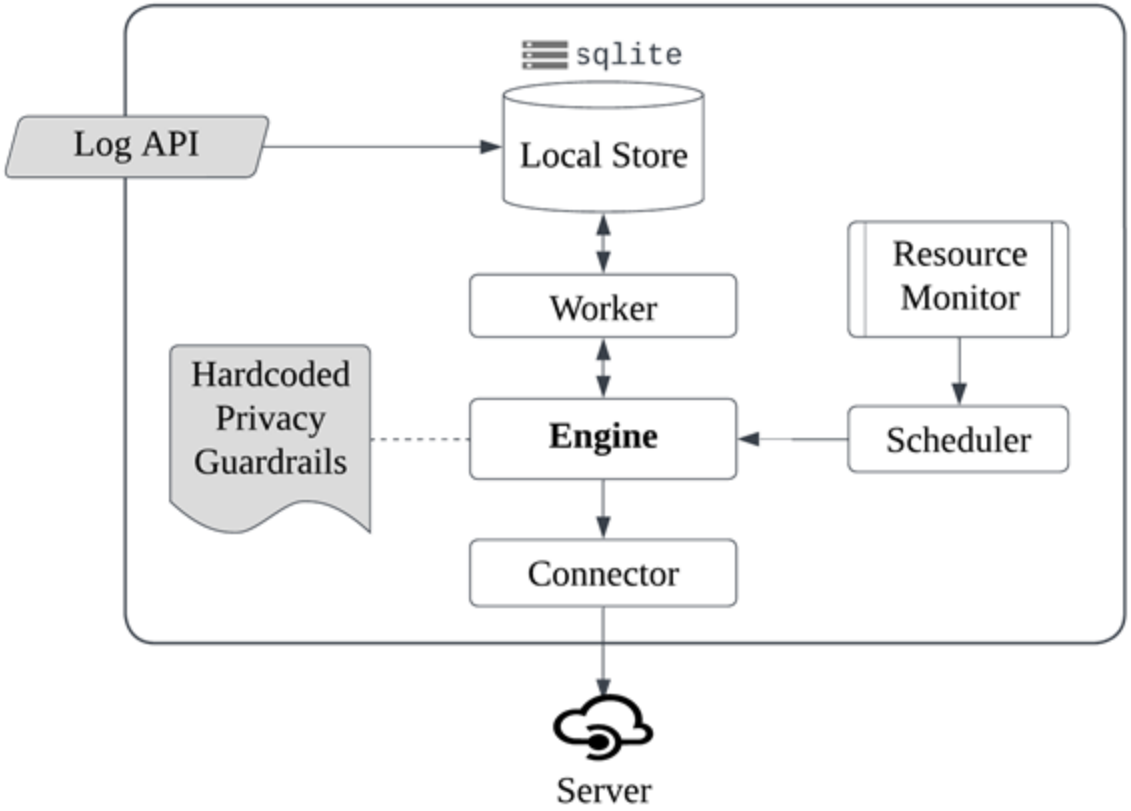


The Federated Query

1. The query of interest
2. What is the privacy requirement for this use case?
3. Where to publish the result?

```
query:
  onDeviceQuery: "SELECT ...", // SQL to run on device
  dimensionCols: ["city", "day"] // grouping columns
  metricCols: // aggregations (e.g., count, mean, ...)
    mean: ["timeSpent"]
privacy:
  centralDP:
    epsilon: ...
  kAnonThreshold: ...
output: ... // where to persist the anonymized result
```

Client Runtime

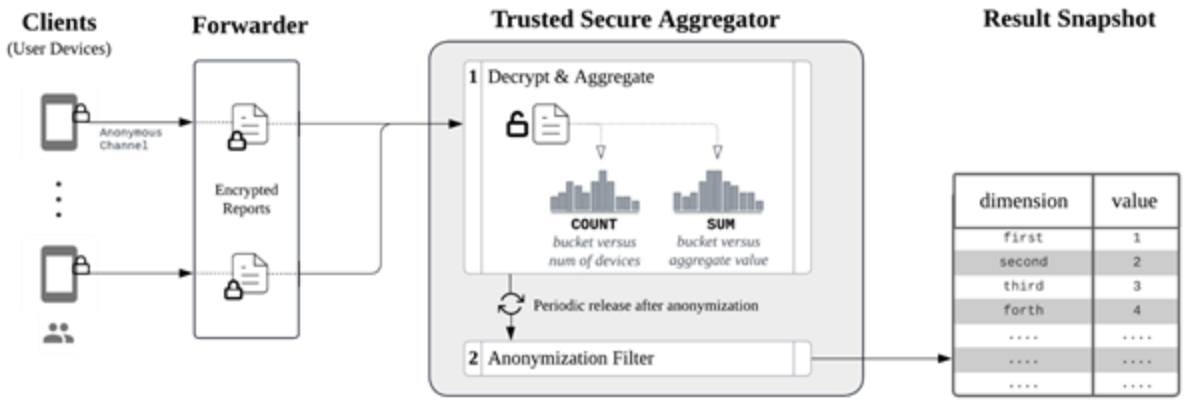


TSA: Secure sum and threshold

Single primitive that sums sparse client histograms

Privacy in three ways:

- aggregation across clients,
- noise addition, and
- removal of small values



Privacy and Security

Secure data handling

We achieve “privacy in depth” by adopting multiple protections at different stages:

- **Data at rest** is on client’s devices. Encrypted and subject to scope and lifetime restrictions
- **Processing** of a user’s data is on their device or in an environment where their device can verify the data handling.
- **Clients control over computations**. Determine what to participate in, based on query config.
- **Validation before sharing.** Client devices through attestation ensure data will be handled correctly before sharing.

Privacy of outputs

Any data seen by orchestrator is “anonymized” according to guardrails set by the client.

Our implementation supports noise addition to achieve differential privacy under various models:

- **Central DP** at the enclave: the TSA adds noise before releasing results to UO
- **Local DP**: clients can add their own privacy noise before sending to TSA, e.g., randomized response
- **Distributed DP noise**: each client adds small noise so that the aggregate is protected
- **Sampling+Threshold DP**: privacy due to random client sampling and suppression of small values
- **Periodic data release**: only a few releases during execution, protecting against differencing attacks

Challenges Revisited

Accessible to analysts

Flexible computation model with familiar SQL and adhoc query support.

Scale

One-shot algorithms, batched computation on devices and predictable load on server.

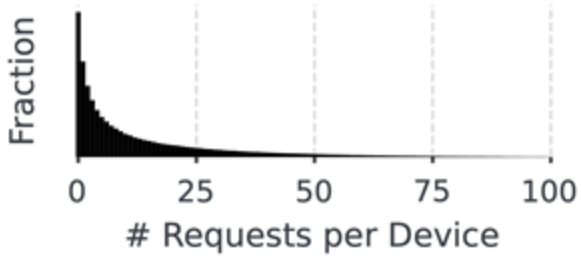
Complex privacy management

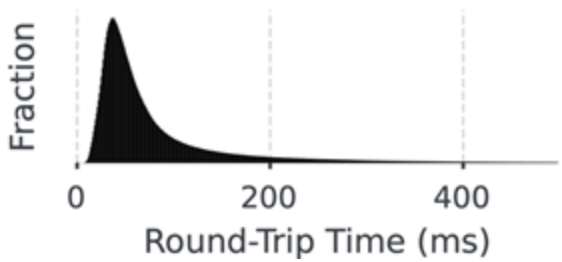
Each client takes responsibility to ensure that their data is handled properly during its processing. Outputs meet the expected privacy standard.

Experimental study

Carefully chose metrics that are represent data and system heterogeneity in production.

- Round trip times (RTT) per request and request volume
- Each device measures round trip times (RTTs) and stores it locally
- Client population of ~100M Android devices





Experimental study

The queries answered by the system

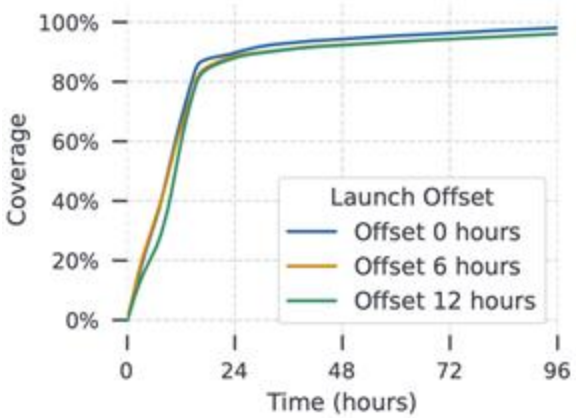
- Federated histogram of the RTT distribution
- Federated histogram of the request volume (number of requests per client)

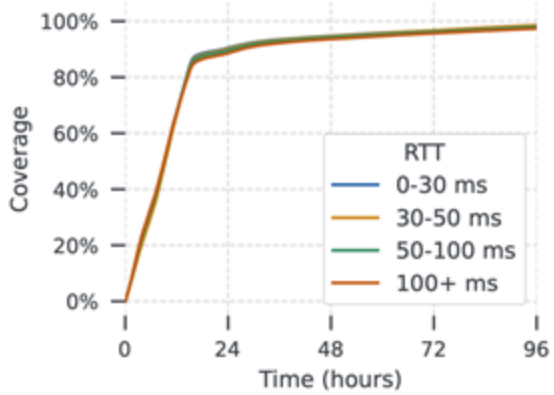
Evaluation metrics

- How long does it take for the system to iterate over all devices and data?
- How accurate is the answer?
- What is the impact by adding differentially private guarantees in this setup?

Collection speed

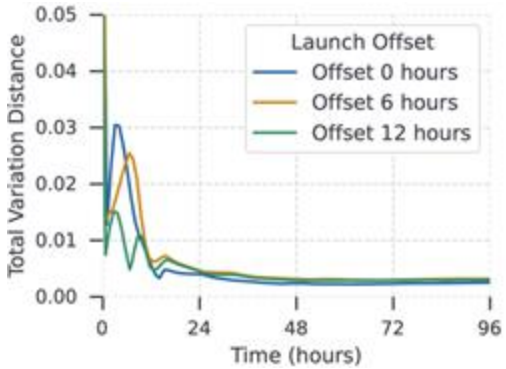
- Rapid coverage from ~85% of devices that respond in the first window, then long tail of others
- Not much variation due to time of day (offset) or bias in coverage.

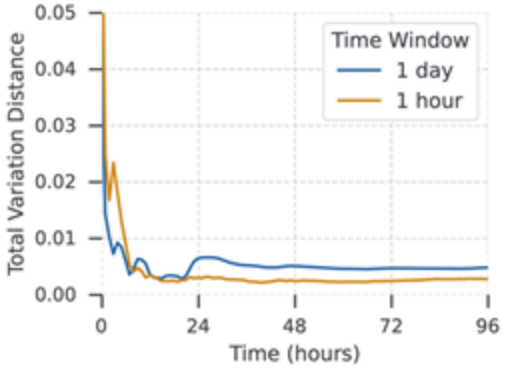




Accuracy analysis

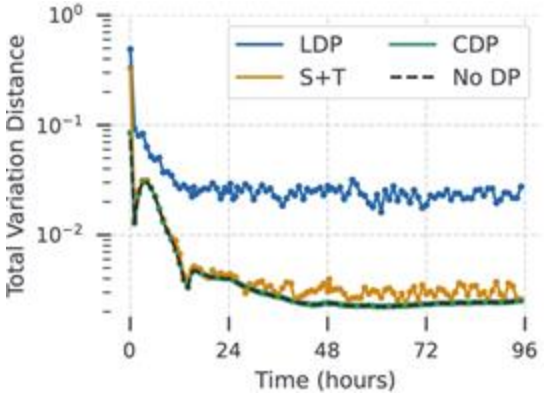
- Accuracy measured via Total Variation Distance from ground truth: lower is better
- High accuracy is achieved after a few hours of data collection
- Faster data collection is possible by narrowing the clients' check-in window (currently ~15 hours)

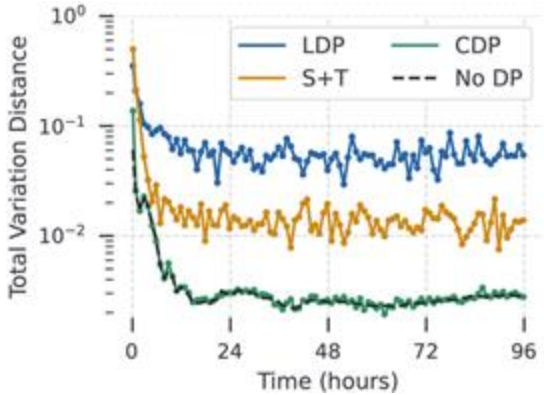




Impact of Privacy Noise

- Comparing impact of no privacy noise (No DP), TSA noise (CDP), client noise (LDP), and sampling (S+T)
- Central noise addition (CDP) has negligible impact on accuracy
- DP via sampling (S+T) has less impact than client noise (LDP) but more than CDP





Concluding remarks

Our deployment demonstrates that it is possible to achieve effective federated data collection at scale. We identify several directions for further development:

- Robustness to stronger threat models e.g., malicious clients attempting to poison results
- Stronger bounds on the DP privacy “budget”, accounting for more queries over time
- More primitive aggregates to help support a wider range of queries

Thank You!