PrivApprox

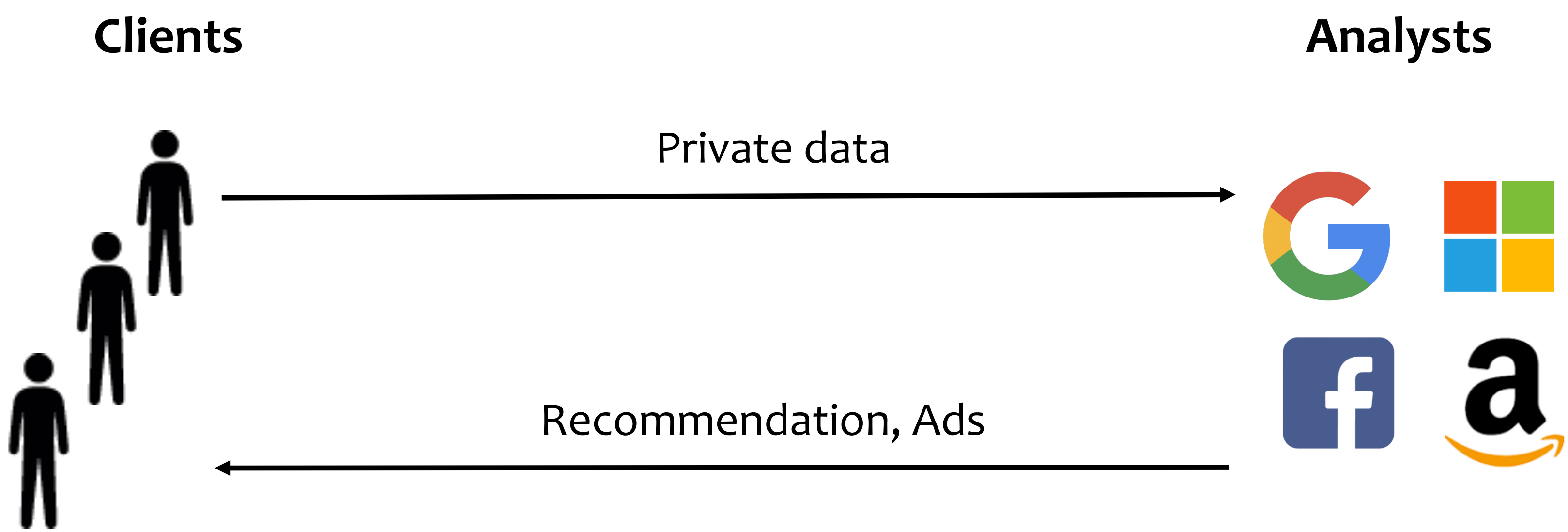
Privacy-Preserving Stream Analytics

https://privapprox.github.io

Do Le Quoc, Martin Beck,
Pramod Bhatotia, Ruichuan Chen, Christof Fetzer, Thorsten Strufe

July 2017

Motivation

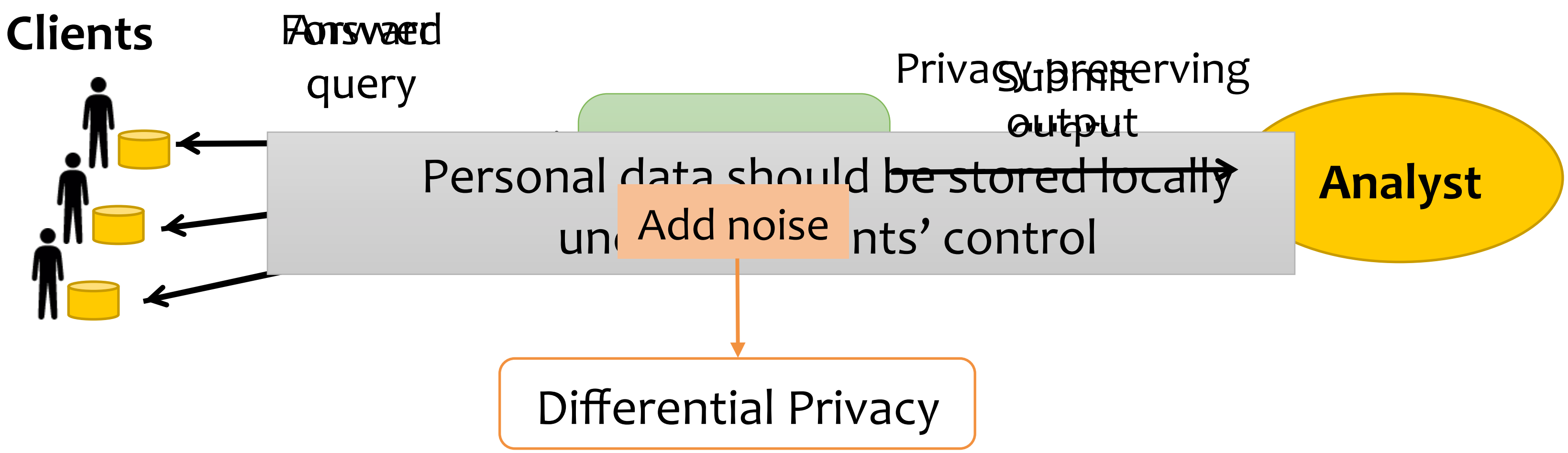


Strong privacy guarantee

High utility analytics in real-time

How to preserve users' privacy while supporting high-utility data analytics for low-latency stream processing?

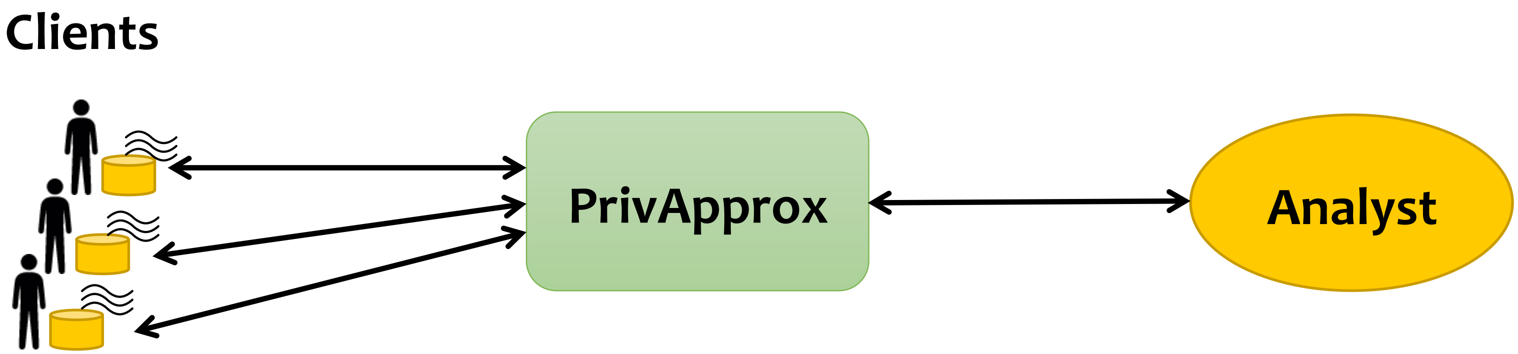
State-of-the-art systems



Limitations:

- Deal with only "single-shot" batch queries ☹
- Require synchronization between system components ☹
- Require a trusted aggregator ☹

PrivApprox



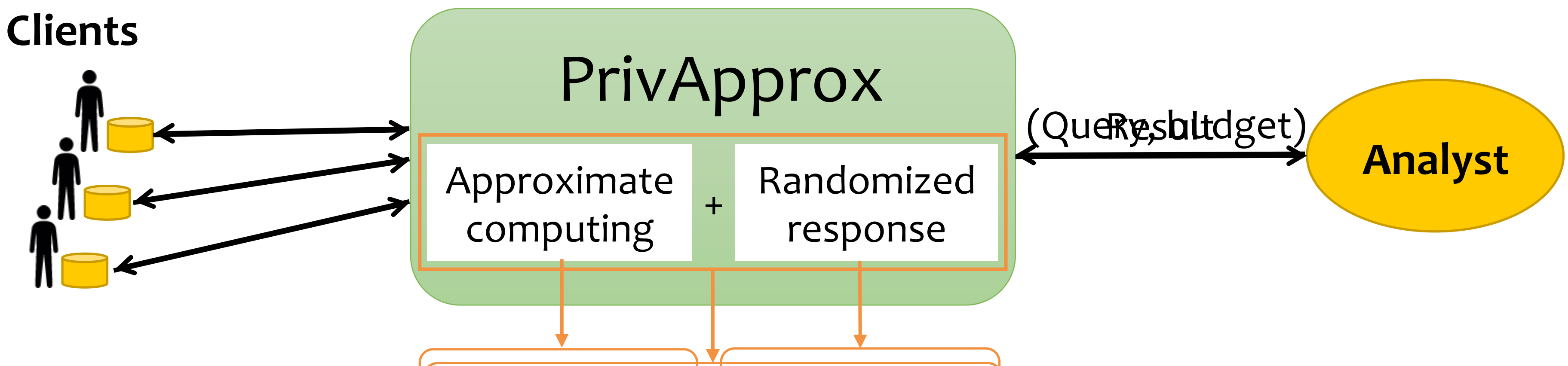
PrivApprox:

- Supports **stream processing** with **low latency** ☺
- Enables a truly **synchronization-free** distributed architecture ☺
- Requires lower trust in aggregator ☺

Outline

- ~~Motivation~~
- Overview
- Design
- Evaluation

System overview

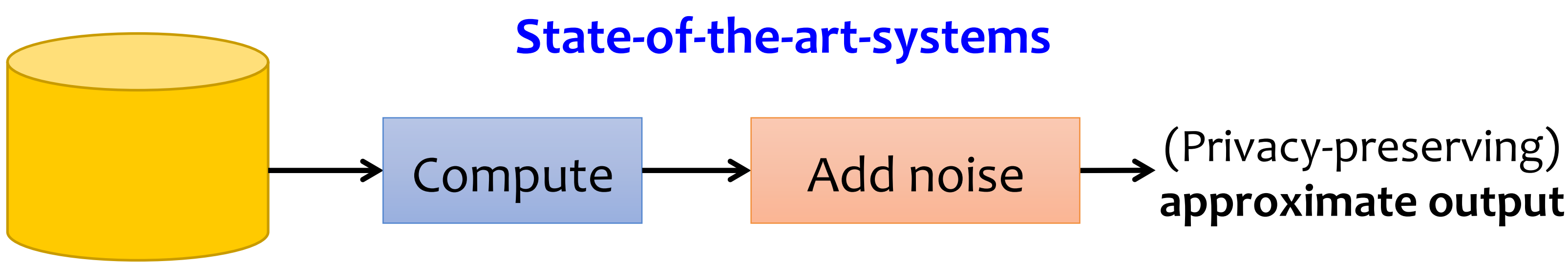


Execution Low latency Zero-knowledge Privacy Privacy

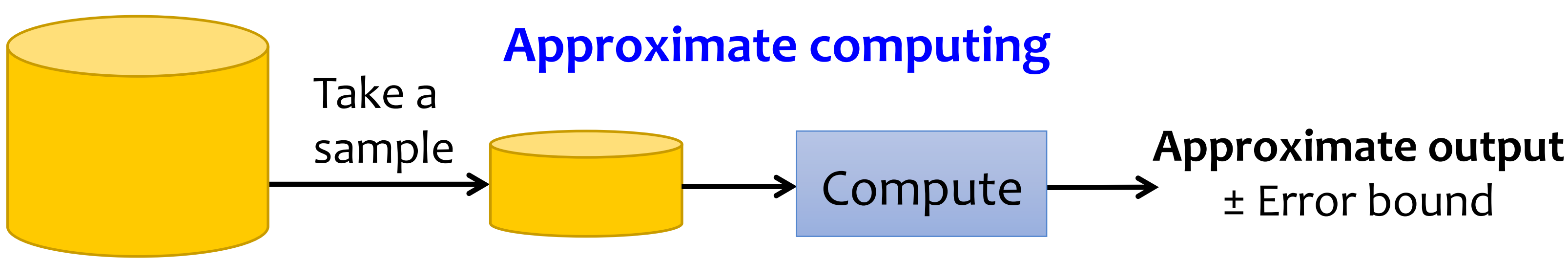
- **Latency/throughput** guarantees
- Desired **computing resources** for query processing
- Desired accuracy

Zero-knowledge Privacy **>** Differential Privacy

#1: Approximate computing

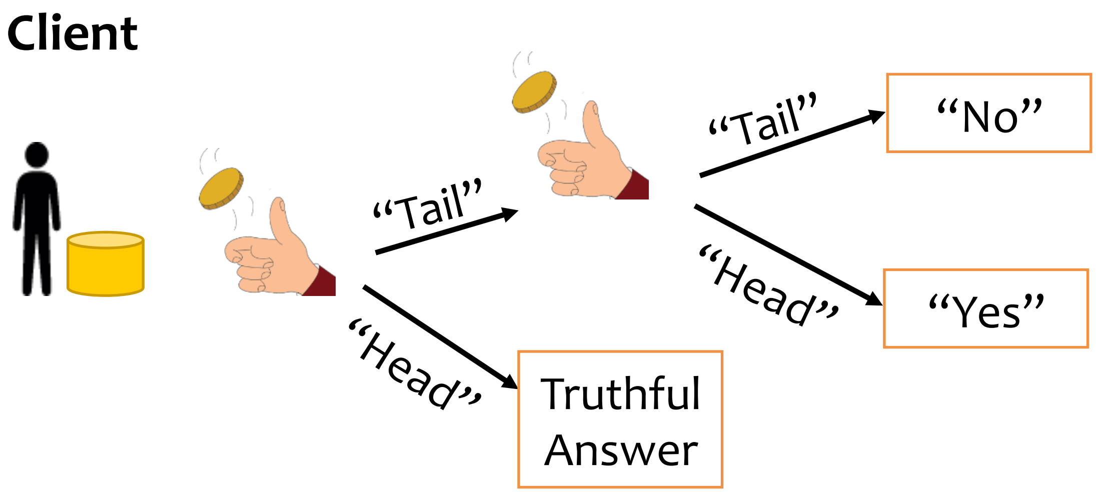


Idea: To achieve low latency, compute over a sub-set of data items instead of the entire data-set



#2: Randomized response

Idea: To preserve privacy, clients may not need to provide truthful answers every time



Provides **plausible deniability** for clients responding to sensitive queries; achieves **differential privacy** (RAPPOR [CCS'14])

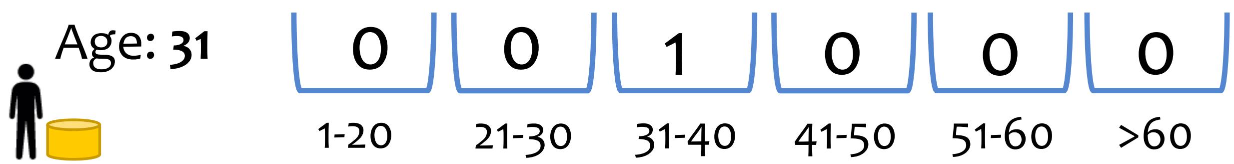
Outline

- ~~Motivation~~
- ~~Overview~~
- Design
- Evaluation

Query model

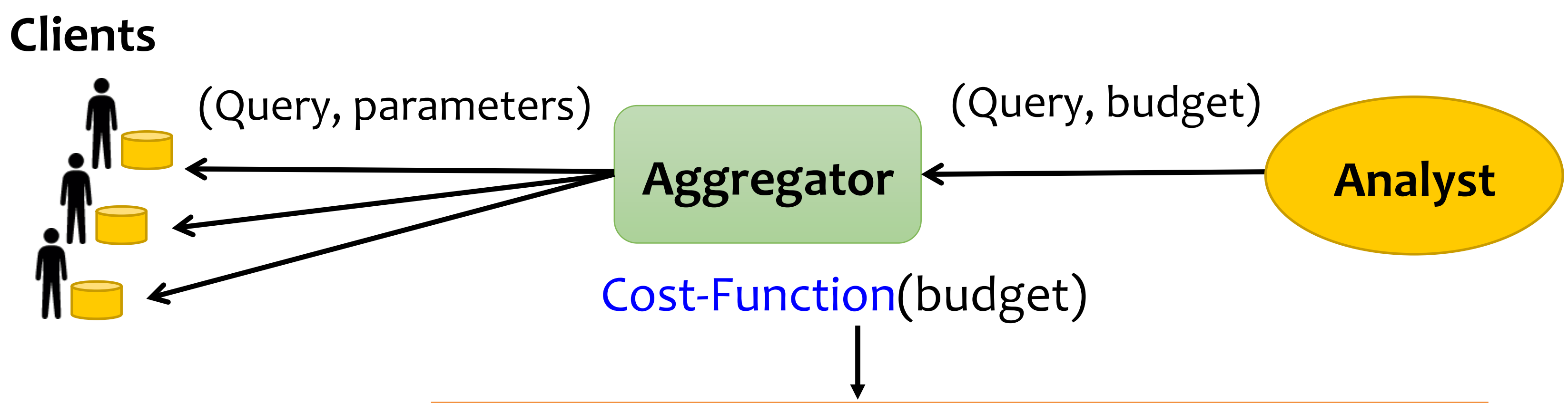
Divide answer's value range into **buckets,**
enforce a **binary answer** in each bucket

Query: SELECT age FROM clients WHERE city = 'Santa Clara'



Client cannot arbitrarily manipulate answers

Workflow: Submit query

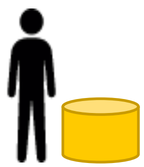


System parameters:

- Sampling parameter
- Randomized response parameters

Workflow: Answer query

Client



Step #1 — Sampling

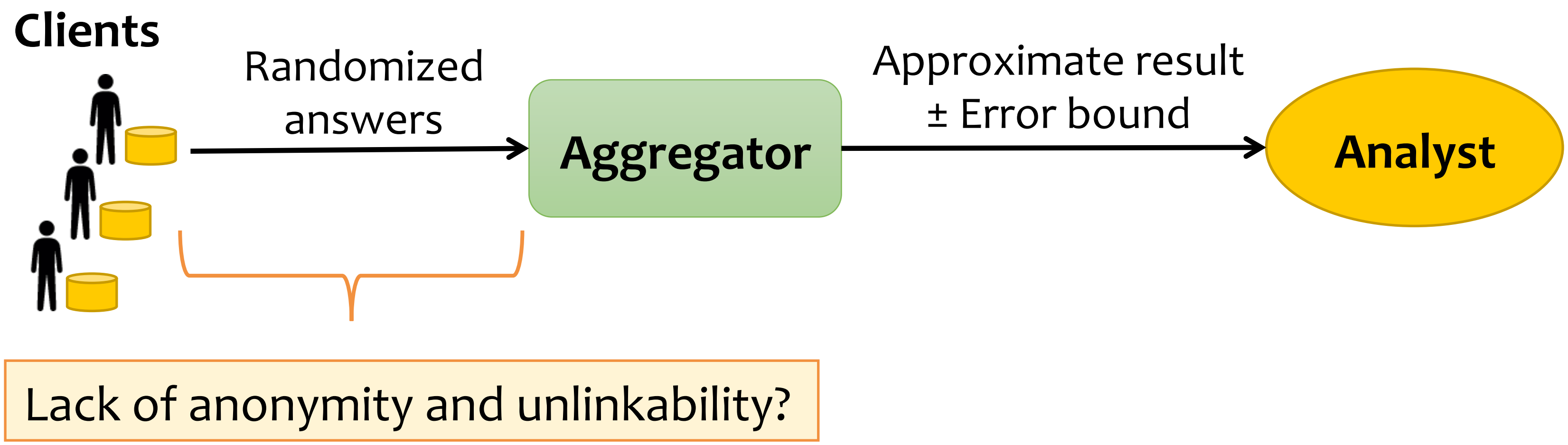
Step #2 — Randomized Response

Step #3 — Send randomized answer

Zero-knowledge privacy

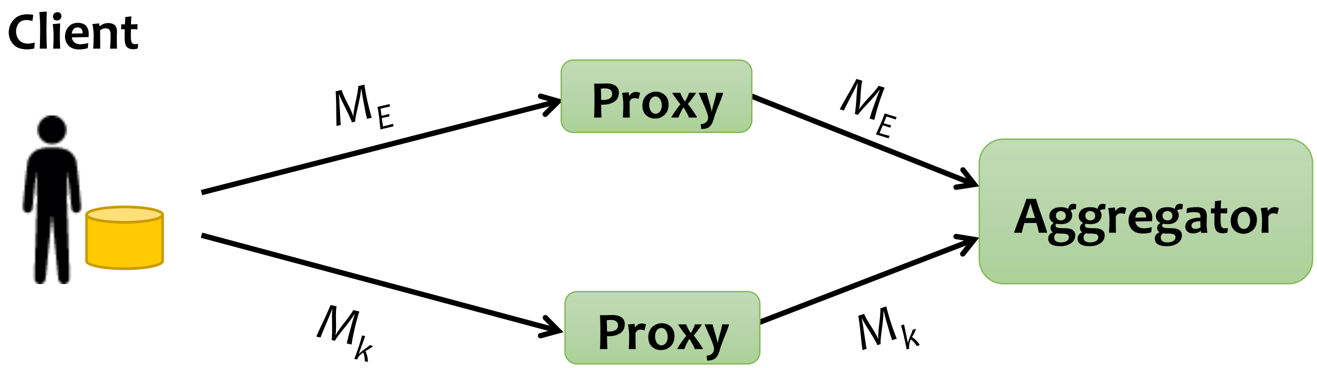
See the paper for details!

Workflow: Answer query



#3: Anonymity and unlinkability

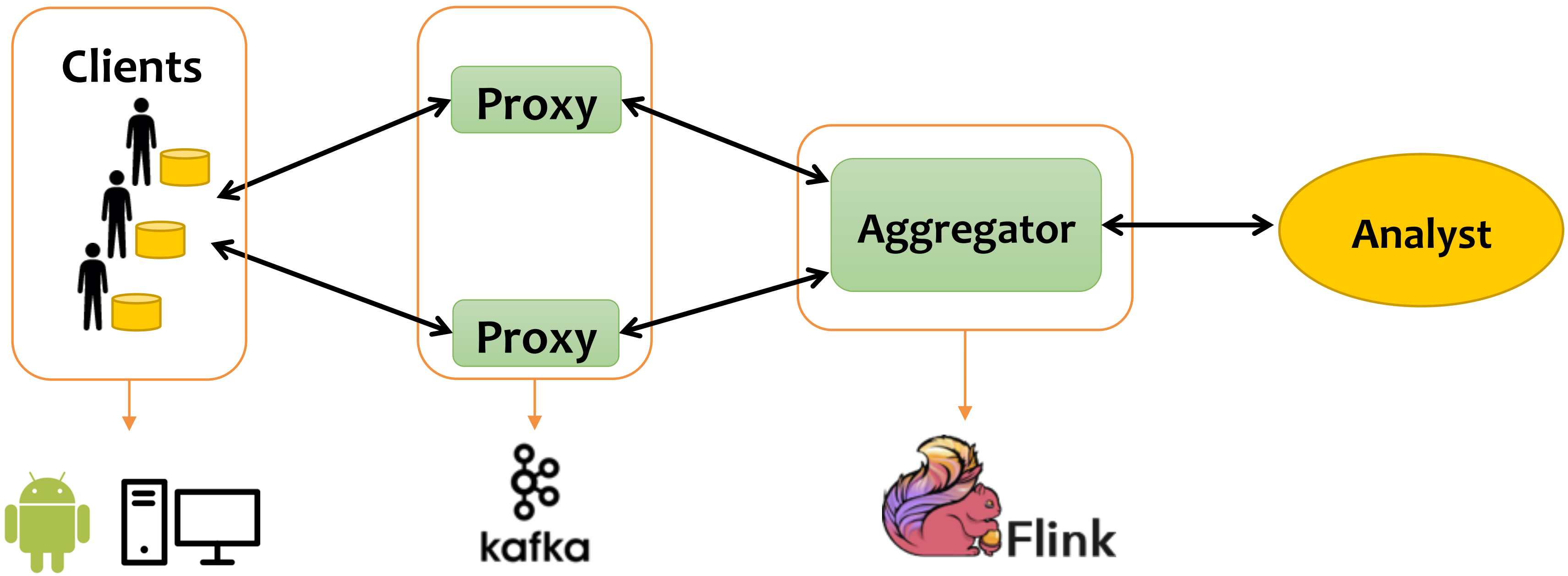
Idea: XOR-based Encryption



Encrypt answer *M*:
GenerateKey -> M_k
M XOR M_k -> M_E

Decrypt answer M_E:
M_E XOR M_k -> M

Implementation



Outline

- ~~Motivation~~
- ~~Overview~~
- ~~Design~~
- Evaluation

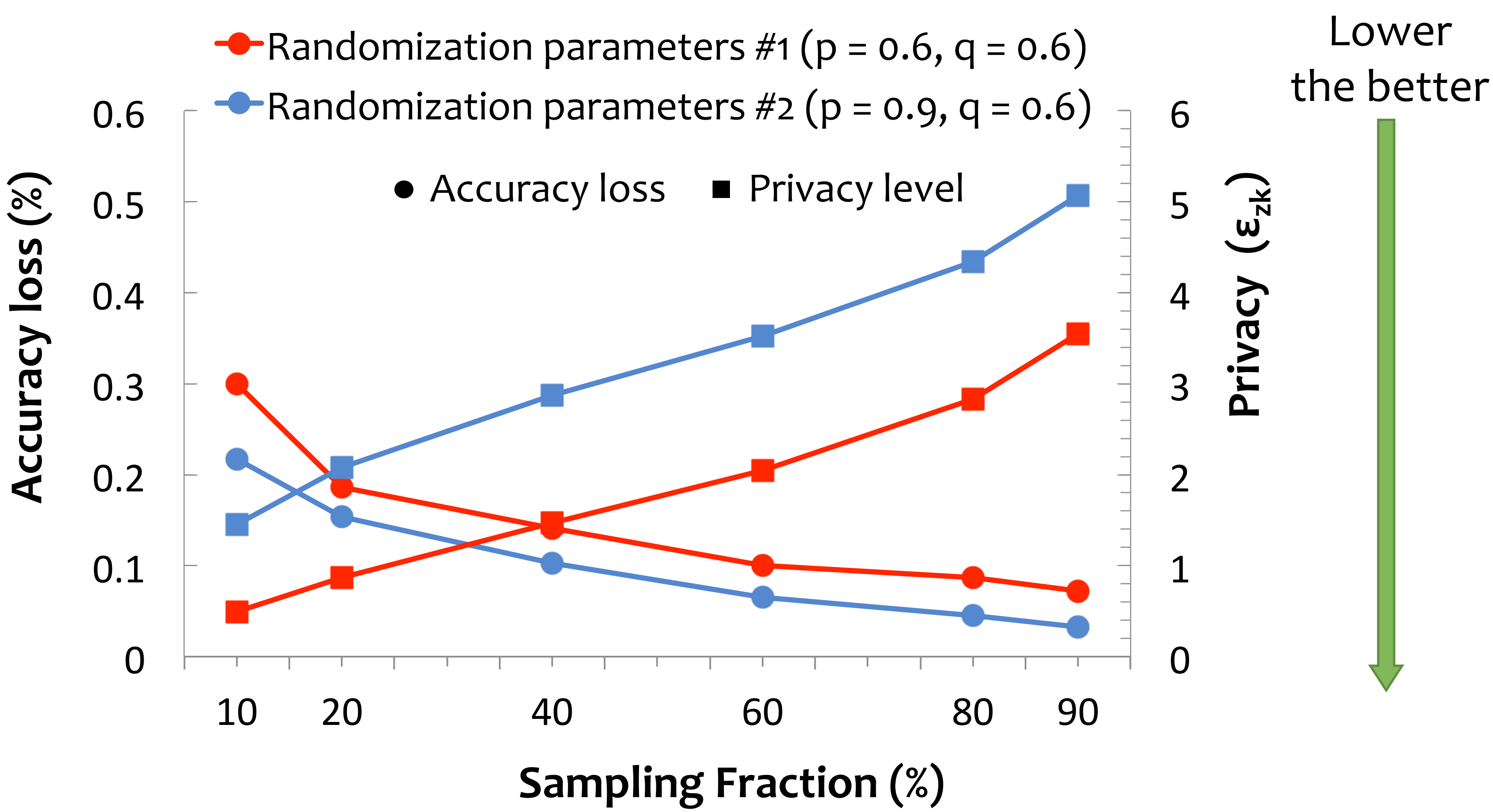
Experimental setup

- Evaluation questions
  - Utility vs privacy
  - Throughput & latency
  - Network overhead

See the paper for more results!

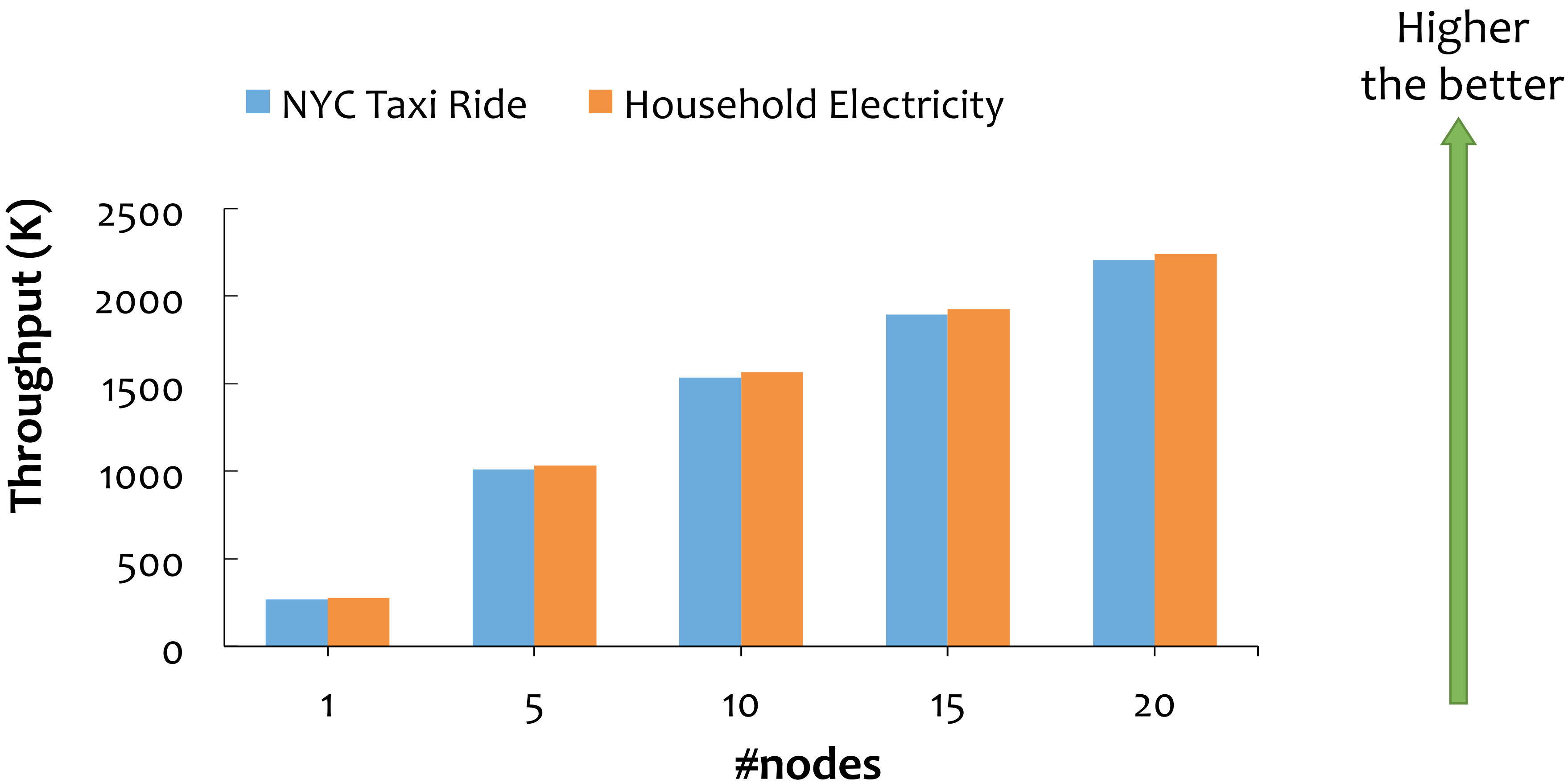
- Testbed
  - Cluster: 44 nodes
  - Dataset: NYC Taxi ride records, household electricity usage

Accuracy vs privacy



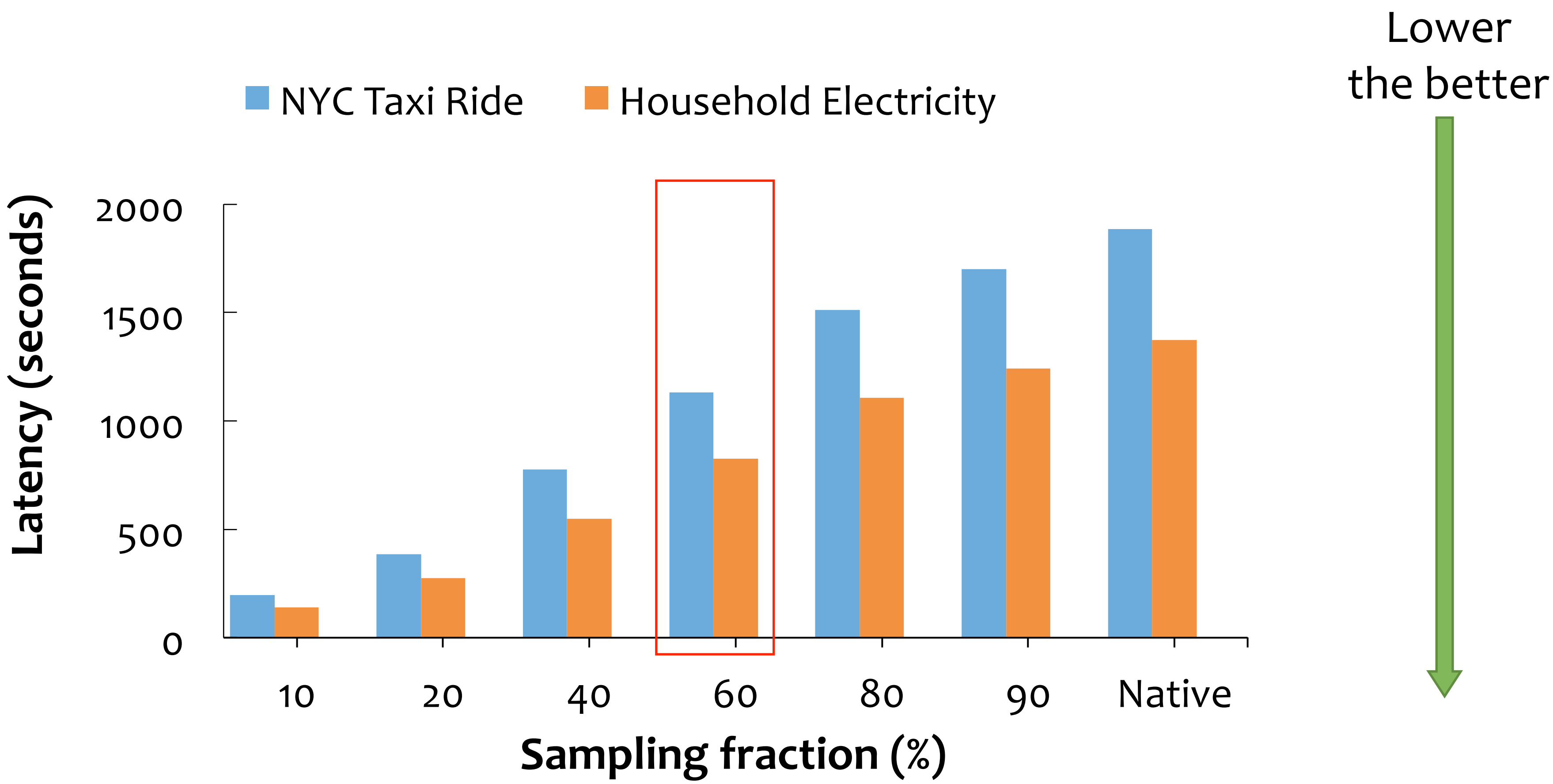
Trade-off between utility and privacy

Throughput



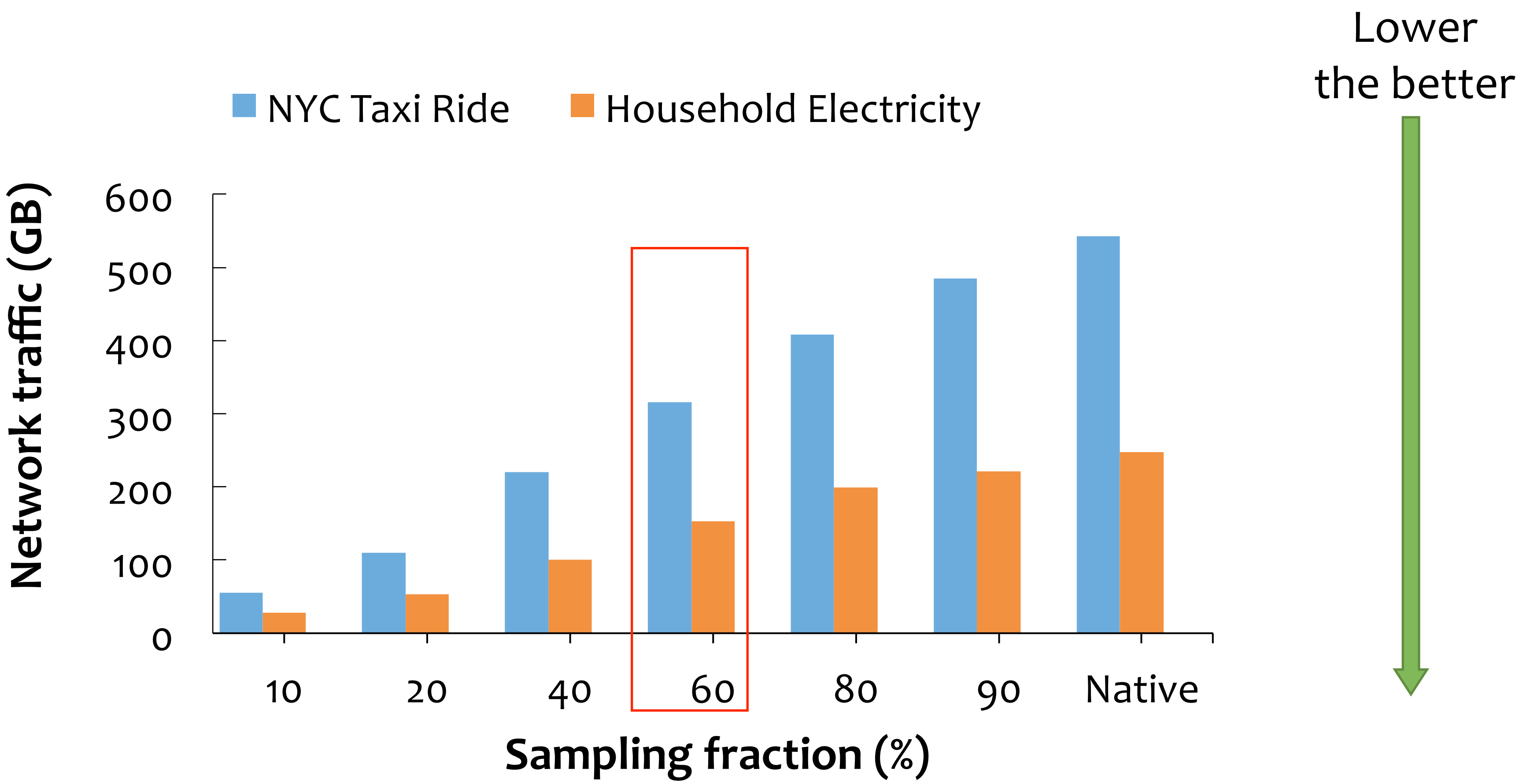
~8X speedup when going from one node to 20 nodes

Latency



~1.66X lower than the native execution with sampling fraction of 60%

Network overhead



~1.6X lower than the native execution with sampling fraction of 60%

Conclusion

PrivApprox: a privacy-preserving stream analytics system over distributed datasets

Privacy	Zero-knowledge privacy
Practical	Adaptive execution based on query budget
Efficient	Randomized response & sampling techniques

Thank you!

https://privapprox.github.io