PRIVACY-PRESERVING MEASUREMENT TECHNIQUES: INFORMAL COMPARISON

Sofía Celi
Brave Software, Inc

Preface

- This is definitely not a complete overview: initial work
- Aims to answer the question: *if I want to execute measurements with privacy, which scheme should I use?*
  - There is an array of options
  - There is an array of security/privacy notions and attackers they provide/protect against
  - Unclear expectations on efficiency and monetary costs

See some further notes: https://sofiaceli.com/thoughts/ppm-tech-01.pdf

Main notion

Aggregate measurements are a way by which **systems** (servers, cloud servers: the data collectors) can **receive data from a population** (a number of users) and compute **useful aggregate statistics** over them.

- Centralized leakage of private user data
- PPM techniques: provide a level of both security and of privacy

https://datatracker.ietf.org/wg/ppm/about/

Wishful thinking

Dalenius [Dal77] stated a desire for something like “semantic security”:

> *access to a statistical database should not enable anyone to learn anything about a user that could not be learned without access*.

- Achieved to a degree by the different techniques

[Dal77] T. Dalenius. Towards a methodology for statistical disclosure control

TECHNIQUES AND SCHEMES

Differential privacy

Add some randomness, or noise, at some points in time: to the data collected, to the output of the aggregate statistic (or function), or to the mechanism itself.

(ϵ -differential privacy).

> A randomized function K gives ϵ-differential privacy if for all data sets $D1$ and $D2$ differing on at most one element, and all $S \subseteq \mathrm{Range}(K)$: $\Pr[K(D1) \in S] \leq \exp(\epsilon) \times \Pr[K(D2) \in S]$.

Meaning: The output of the function is similar on both data sets if you change or remove the one element -> cannot differentiate

Differential privacy

- RAPPOR (2014 – 2019) [EPK14]:
    - memoization and randomization
    - privacy is not degraded if the survey is repeated with the same set of users
    - inefficient due the local randomness added

[EPK14] Ulfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. RAPPOR: Randomized aggregatable privacy-preserving ordinal response.

- PROCHLO [BEM+17]:
    - Encode, Shuffle, Analyze (ESA) architecture
    - The local randomness is augmented by a private channel that randomly permutes a set of user-supplied data
    - Requires trusted architecture -> honest execution

[BEM+17] Andrea Bittau, Ulfar Erlingsson, Petros Maniatis, Ilya Mironov, Ananth Raghunathan, David Lie, Mitch Rudominer, Ushasree Kode, Julien Tinnes, and Bernhard Seefeld. Prochlo.

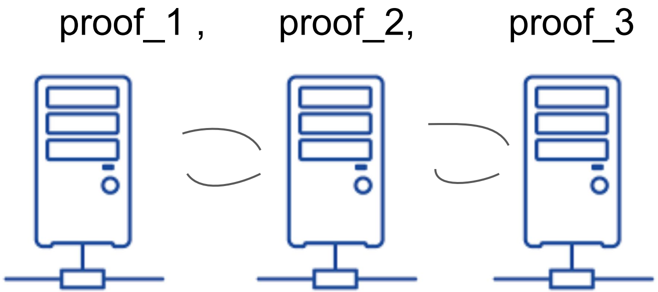
Prio-based [CGB17]

- Private aggregation that aims to provide privacy, robustness, and scalability
- Works with a small number of servers (and large amount of clients), and, as long as one of them is honest, the system leaks nearly nothing about users data, except for what the aggregate statistic itself reveals
- Variations (mainly improving client-to-server communication):
    - Prio+ [AGJ+21]
    - Prio2 [AG21]
    - Prio3 [BBC+19, GPRW22]

[CGB17] Henry Corrigan-Gibbs and Dan Boneh. Prio: Private, robust, and scalable computation of aggregate statistics
[AGJ+21] Surya Addanki, Kevin Garbe, Eli Jaffe, Rafail Ostrovsky, and Antigoni Polychroniadou. Prio+: Privacy preserving aggregate statistics via boolean shares.
[AG21] Apple and Google. Exposure notification privacy-preserving analytics (enpa)
[BBC+19] Dan Boneh, Elette Boyle, Henry Corrigan-Gibbs, Niv Gilboa, and Yuval Ishai. Zero knowledge proofs on secret-shared data via fully linear PCPs
[GPRW22] Tim Geoghegan, Christopher Patton, Eric Rescorla, and Christopher Wood. Privacy preserving measurement.

Prio-based [CGB17]

f-privacy: an adversary, who controls any number of clients and all but one server, learns nothing about the honest clients’ values, except what they can learn from the aggregation function itself.



Servers gossip and consider that proof valid or not

Secret shared non-interactive proofs (SNIPs):

Pick three random shares that sum to 1:
15 + (-12) + (-2)
proof_1 , proof_2, proof_3

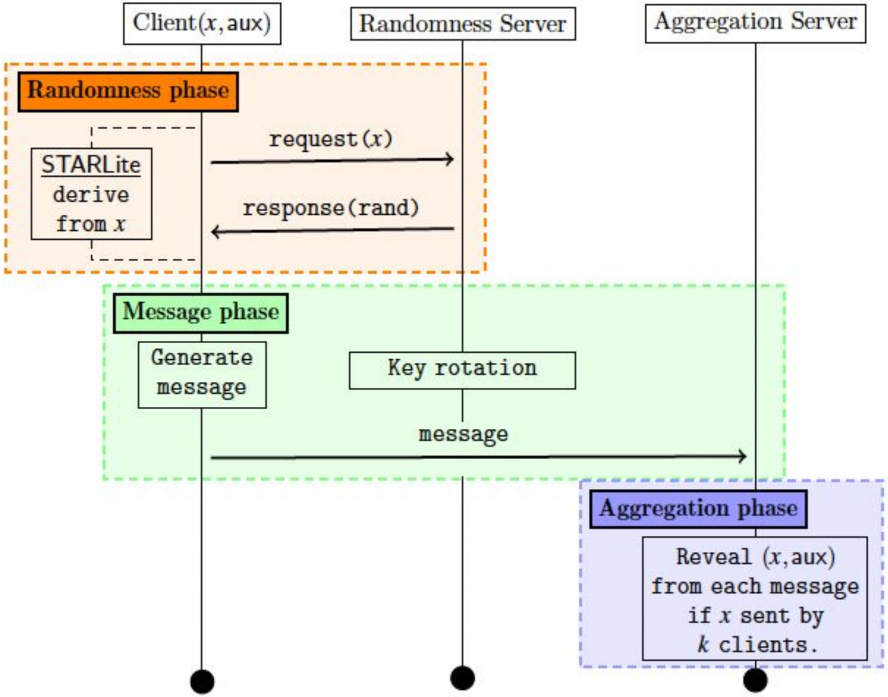
Server(s) learn the sum of the values but not the private values themselves

Servers assure that this shares sum to 0 or 1, without learning ‘x’.
Expressed as an arithmetic circuit:
If *Valid*(x) holds, without leaking ‘x’: Valid(x) = x ∈ {0, 1}

STAR [DSQ+21]

- Learn only data sent by *k-clients (k-heavy-hitters)*
  - The server only learns any data from a client if there are at least *k − 1* other clients submitting
  - Prevents the data collector from learning uniquely identifying (or uniquely co-occurring patterns of) data from a unique client
- Each client constructs a ciphertext of their data, using an encryption key derived from randomness
- Client sends: the ciphertext, a *k-out-of-N* secret share of the randomness, and a tag informing the server which shares to combine.
- The aggregation server: organizes the shares into subsets depending on the tags, and recovers the encryption keys from those subsets of size ≥ K.

[DSQ+21] Alex Davidson, Peter Snyder, E. B. Quirk, Joseph Genereux, and Benjamin Livshits. STAR: distributed secret sharing for private threshold aggregation reporting



From: [DSQ+21] Alex Davidson, Peter Snyder, E. B. Quirk, Joseph Genereux, and Benjamin Livshits. STAR: distributed secret sharing for private threshold aggregation reporting, Shared with permission

POPLAR [BBCG+21]

- Very similar to Prio
- Allows for finding the most popular strings among a collection of clients, as well as counting the number of clients that hold a given string
- Requires two non-colluding data-collection servers that *n* clients communicate with
- Preserves client privacy as long as one of the two servers is honest

[BBCG+21] Dan Boneh, Elette Boyle, Henry Corrigan-Gibbs, Niv Gilboa, and Yuval Ishai. Lightweight techniques for private heavy hitters

Scheme	**Type of data**	**Privacy Notion**	**Robustness Notion**	**Trust**
Prio	Only numeric	f-*privacy* (or $\tilde{f}$-*privacy*)	Preserved only in the face of adversarial clients	At least, one server should be honest
Prio+	Only numeric (validation is boolean)	f-*privacy* (or $\tilde{f}$-*privacy*)	Preserved only in the face of adversarial clients	At least, one server should be honest
Prio2	Only numeric	f-*privacy* (or $\tilde{f}$-*privacy*)	Should inherit from Prio (needs review)	At least, one server should be honest
Prio3	Only numeric	f-*privacy* (or $\tilde{f}$-*privacy*)	Should inherit from Prio (needs review)	At least, one server should be honest
STAR	All types	Threshold k-anonimity	In all cases up to a leakage parameter	All parties can be untrusted
POPLAR	All types	f-*privacy* (or $\tilde{f}$-*privacy*)	Preserved in the face of adversarial clients, and up to a leakage parameter for one malicious server.	At least, one server should be honest

Figure 1: Comparison of properties of different schemes.

Scheme	Leakage	Expressive Functionality	Efficiency	Monetary Cost
Prio	Depending on the aggregation function (on the majority, no leakage)	Can be used with multiple aggregation functions	Slow client-to-server communication	
Prio+	Depending on the aggregation function (on the majority, no leakage)	Can be used with multiple aggregation functions (but it is costly to translate boolean circuits to arithmetic)	Improved client-to-server communication	
Prio2	Depending on the aggregation function (on the majority, no leakage)	Can be used with multiple aggregation functions	Improved client-to-server communication	
Prio3	Depending on the aggregation function (on the majority, no leakage)	Can be used with multiple aggregation functions	Improved client-to-server communication	
STAR	The server learns which clients share the same measurement	Limited use of different aggregation functions	Fast	Cheap
POPLAR	Leakage of all heavy-hitting prefixes	Limited use of different aggregation functions	Slow client-to-server communication	Costly

Figure 2: Comparison of costs, functionality and leakage of different schemes.

User needs

In the design of these schemes, the **voice of the end-user is notably absent**

- Do users understand that their data is collected in a privacy-preserving manner?
- Can users consent to sharing or remove themselves from a system that uses *x* scheme?
- Do they understand the notion of privacy that is given by an *x* scheme?
- Do they know the used scheme and the limitations of it?

User needs

- Findings of [CKR21] suggest that users care about data disclosure and the privacy of it; but, giving them a “random” definition of privacy does not make them more willing to share their data
- Schemes must emphasize user agency:
  - explicit about the exact properties they guarantee
  - any change to either the scheme/property needs user notification, consent and opt-out
  - explicit about what the data will be used for
- Should be an ingrained consideration of the schemes rather than an application-specific or architectural option
- Users might care about ‘individual privacy’ but also about ‘group privacy’

[CKR21] Rachel Cummings, Gabriel Kaptchuk, and Elissa M. Redmiles. “I need a better description”: An investigation into user expectations for differential privacy

THANK YOU!

@claucece

See: https://sofiaceli.com/thoughts/ppm-tech-01.pdf