Privacy Preserving Measurement

Abhishek Tiwari

Citation: *A. Tiwari*, "Privacy Preserving Measurement", Abhishek Tiwari, 2024. doi:10.59350/fnpfz-3v466

Published on: October 01, 2024

In 1982, Andrew Yao proposed the Millionaire Problem which discusses how two millionaires can learn who is richest one without disclosing their actual wealth. They solve this problem by comparing their wealth using secure two party computation to ensure that they learn only the richest one and nothing else is revealed. The problem was later generalised for secure multiparty computation by Goldreich et al in 1987.

Privacy Preserving Measurement (PPM) can be considered as an extension of Millionaire problem. PPM refers to a set of techniques and protocols that allow useful statistics and insights to be derived from data without exposing individual-level information. When combined with formal privacy guarantees like differential privacy, PPM enables organizations to perform analytics and measurement in a privacy-preserving manner.

In this post, we'll explore the key concepts behind PPM, how it works in practice, and how differential privacy enhances its privacy protections. We'll also look at some real-world applications and the future potential of this technology.

Need

Traditional approaches to data analytics typically involve collecting raw user data in a central location for processing. This creates a variety of privacy and security risks. Sensitive personal information could be exposed in a data breach. Insiders with access to the data could misuse it. The centralized data set becomes an attractive target for hackers. Users have limited visibility into how their data is being used. It's difficult to prevent the data from being repurposed for unintended uses.

PPM aims to enable useful measurement and analytics while avoiding these pitfalls. The core idea is to structure data collection and processing in a way that useful aggregate insights can be extracted without needing access to individual-level data. PPM allows aggregate statistics to be computed without exposing raw user data, preventing the linking of data points to specific individuals, giving users more control over how their data is used, reducing the security risks associated with centralised data collection, and enabling privacy-preserving collaboration between multiple parties.

Key Concepts

A few core concepts and techniques form the foundation of PPM model.

Secure Multi-Party Computation (MPC): This refers to cryptographic protocols that allow multiple parties to jointly compute a function over their private inputs, without revealing those inputs to each other. In the context of PPM, MPC allows multiple data holders to collaborate on computing aggregate statistics without sharing raw data. MPC should ensure the following two properties : (i) input

privacy, i.e., parties' inputs should remain private and only the output of the function is learned; and (ii) correctness, i.e., even if some parties maliciously act, the correct output is obtained.

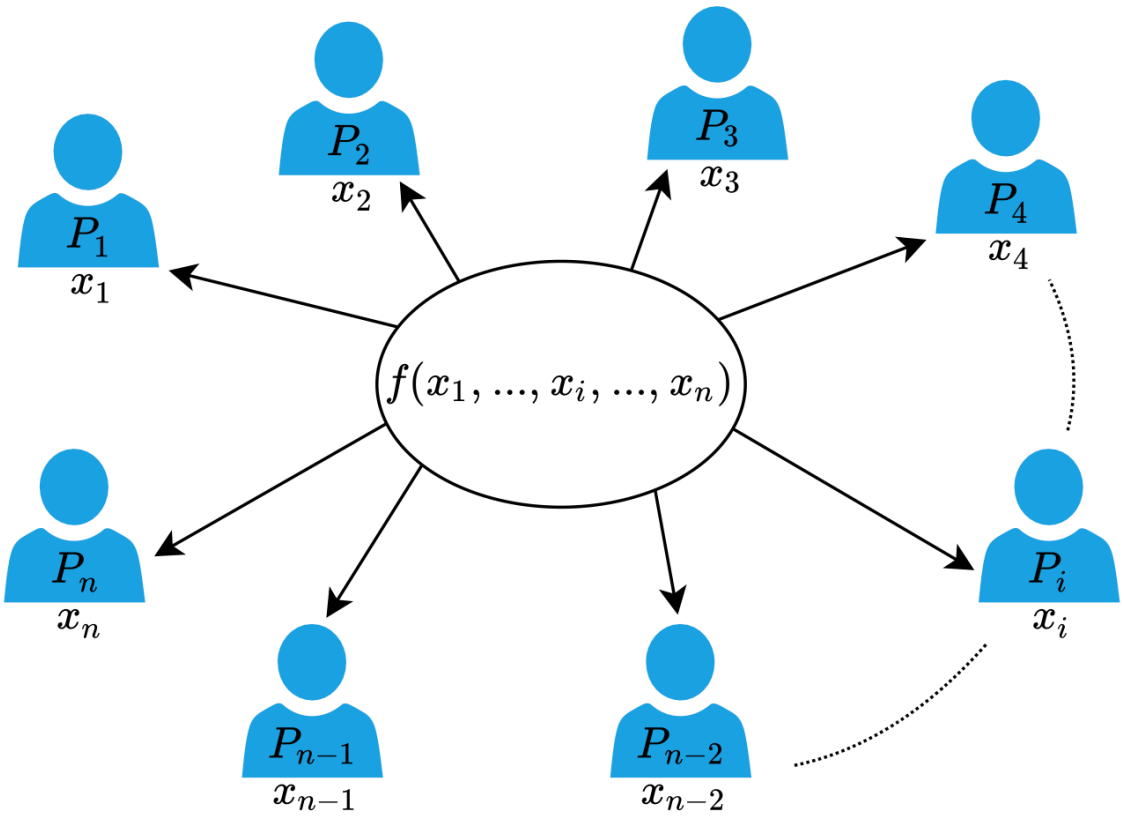

Figure 1: Let P1, . . . , Pn be n parties and each of them having input x1, . . . , xn, respectively. These parties want to jointly compute function F over all inputs {x1,...,xn} and learn the output without revealing their input. Image credits Beyza Bozdemir

Homomorphic Encryption: This is a form of encryption that allows computations to be performed on encrypted data without decrypting it first. It enables a party to perform analytics on encrypted user data without accessing the underlying plaintext.

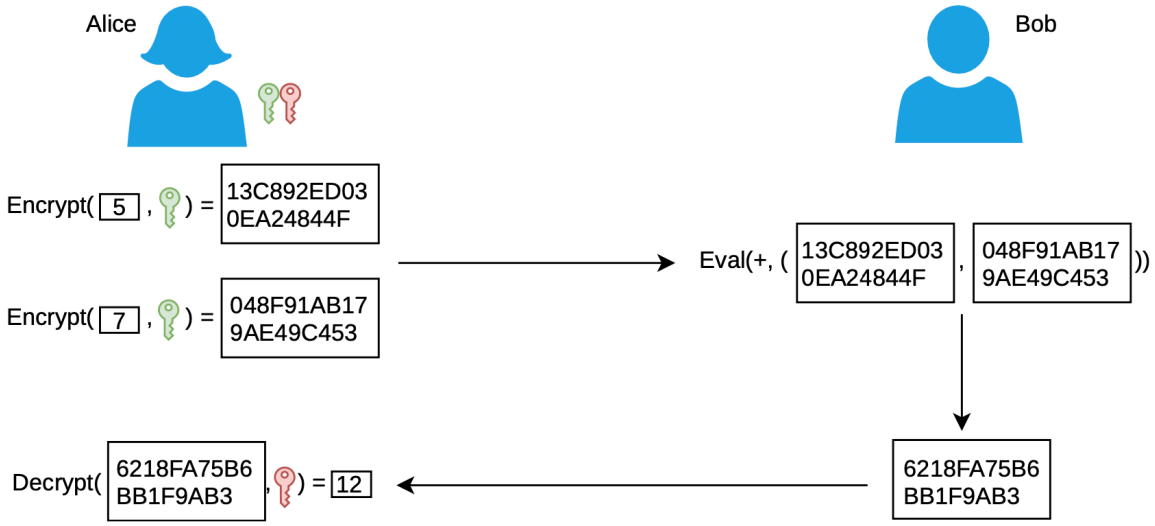

Figure 2: Alice can delegate some of her computations over sensitive data to an untrusted third party, Bob. Alice encrypts her data with her public key (in green) and sends it to Bob. When receiving the data, Bob can evaluate a function over Alice's encrypted inputs and obtain the encrypted result. Bob sends the result back to Alice who is the only party able to decrypt it with her private (or secret) key (in red). Image credits Beyza Bozdemir.

Secure Aggregation: This refers to techniques for combining inputs from multiple parties in an encrypted or cryptographically secure way, such that only the final aggregate result is revealed.

Distributed Noise Addition: Methods for having multiple parties collaboratively add statistical noise to results in a distributed fashion, enabling differential privacy without a trusted central party.

Cryptographic Commitments: Techniques that allow a party to commit to a piece of data or computation result without revealing it, and later prove that they computed it correctly.

By leveraging these cryptograph primitives, PPM protocols can be constructed to perform useful measurement tasks in a privacy-preserving manner.

Distributed Aggregation Protocol

One of the most promising standardised approaches for PPM is the Distributed Aggregation Protocol (DAP) being developed by the IETF Privacy Preserving Measurement working group. DAP provides a framework for privacy-preserving data collection and aggregation across a large number of client devices.

DAP Roles

DAP defines several key roles that work together to enable privacy-preserving measurement. The primary roles in DAP are:

1. Clients: These are typically end-user devices (like smartphones or browsers) that generate the original measurement data. Clients encrypt their data and split it into shares before sending it to the Aggregators.
2. Aggregators: DAP requires at least two non-colluding Aggregator servers. These servers receive encrypted shares from Clients and perform secure multi-party computation to aggregate the data without seeing individual values. DAP defines two specific Aggregator roles:
    - Leader: This Aggregator coordinates the protocol execution, interacts with Clients and the Collector, and orchestrates the aggregation process.
    - Helper: This Aggregator assists the Leader in the computation but has more limited responsibilities.
3. Collector: This is the entity that initiates measurement tasks and receives the final aggregated, privacy-preserving results. The Collector defines queries but does not have access to individual Client data.

Each of these roles plays a crucial part in ensuring that useful aggregate data can be collected and analyzed while strongly protecting individual privacy. The separation of roles and the requirement for multiple non-colluding parties are key to DAP's privacy guarantees.

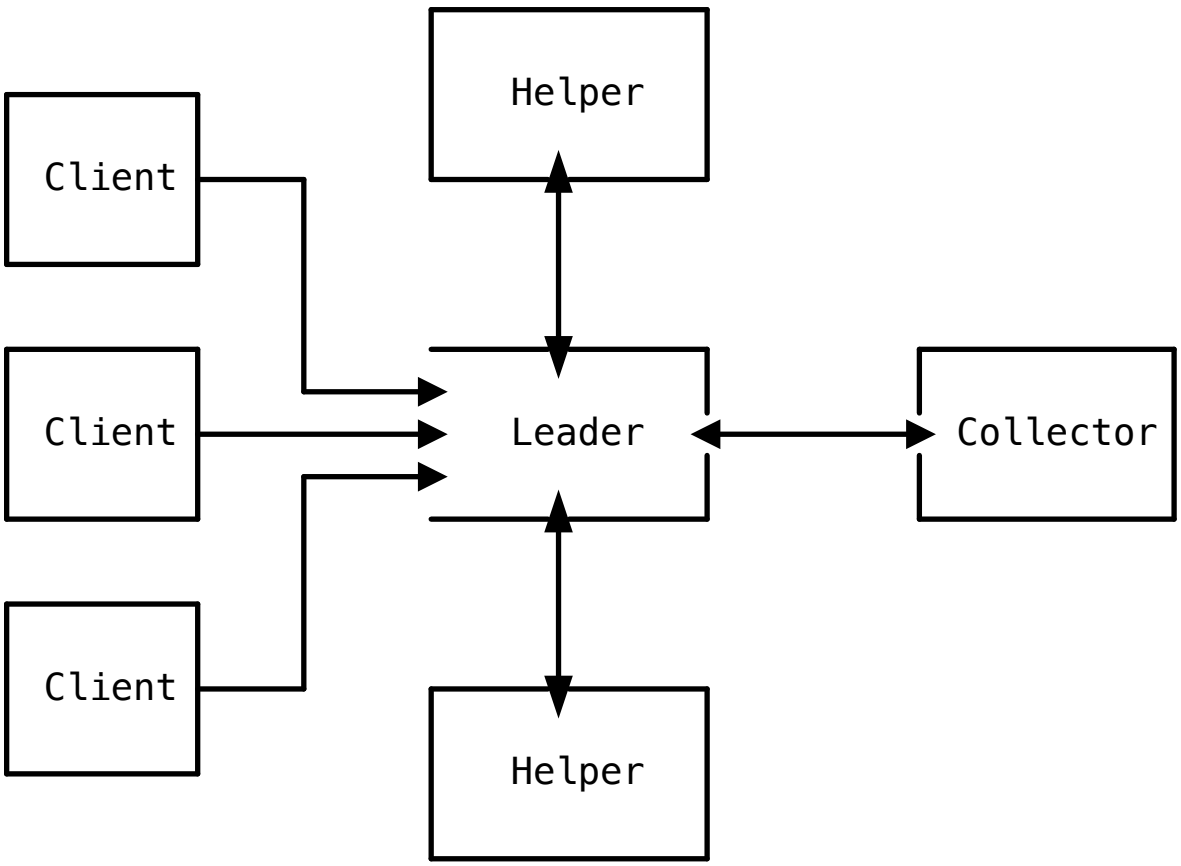

Figure 3: DAP Roles - Clients generate the original measurement data, Aggregators participate in multi-party aggregation, and Collector receives the final aggregated, privacy-preserving results.

How DAP works

The protocol begins with Clients generating measurements based on some predefined task. Instead of sending these measurements directly, each Client uses a cryptographic technique to split its measurement into multiple "input shares." These shares are encrypted in such a way that no single party can reconstruct the original measurement from a share alone. The encrypted input shares are then sent to the Leader Aggregator.

Upon receiving the encrypted input shares, the Leader Aggregator doesn't attempt to decrypt them directly. Instead, it distributes the shares between itself and the Helper Aggregator. This distribution of shares is a critical aspect of DAP's privacy guarantees, as it ensures that no single Aggregator has access to a complete measurement.

The heart of DAP's functionality lies in its use of Verifiable Distributed Aggregation Functions (VDAFs). VDAFs are cryptographic protocols that allow the Aggregators to jointly process the input shares without ever seeing the underlying data in the clear. VDAFs serve two crucial purposes in DAP. First, they enable the Aggregators to verify that each measurement is valid according to predefined criteria, without actually seeing the measurement itself. This is essential for ensuring the integrity of the aggregate result without compromising individual privacy. Second, VDAFs allow the Aggregators to transform the input shares into "output shares" that can be aggregated to produce the final result.

VDAFs work by having each Aggregator perform part of a computation on its share of the data. These partial computations are then combined in a way that produces the correct result as if the computation had been performed on the complete, unencrypted data. Importantly, this process reveals nothing about individual measurements to either Aggregator. The VDAF used in a particular DAP deployment is chosen based on the specific type of measurement and aggregation required.

The aggregation process in DAP is organized around the concept of "batches." A batch is a set of reports (i.e., measurements) that are aggregated together. The protocol supports different types of queries that determine how these batches are formed. For example, time-interval queries group reports based on when they were generated, while fixed-size queries aim to create batches with a specific number of reports. The use of batches, combined with minimum batch size requirements, provides an additional layer of privacy protection by ensuring that individual measurements are always aggregated with a sufficient number of other measurements.

When the Collector wishes to obtain an aggregate result, it initiates the collection process by sending a query to the Leader Aggregator. This query specifies which batch of measurements should be aggregated and any necessary parameters for the aggregation. The Leader and Helper Aggregators then work together to compute "aggregate shares" based on the output shares they hold for the specified batch. These aggregate shares are encrypted and sent to the Collector.

Finally, the Collector receives the encrypted aggregate shares from both Aggregators. By combining these shares, the Collector is able to compute the final aggregate result. Importantly, this process reveals only the aggregate statistic to the Collector, without exposing any information about individual measurements.

DAP incorporates several additional features to enhance its security and flexibility. It supports extensions that allow for client authentication or the inclusion of additional metadata with measurements. The protocol also includes mechanisms for detecting and preventing various attacks, such as replay attacks or attempts to submit invalid measurements.

This approach has several key benefits. Raw user data is never visible to the aggregators or data collector. No single party has access to individual reports. The system is scalable to millions of clients. DAP supports various aggregation functions beyond simple sums, including histograms and complex statistics, by adjusting the client-side encoding and server-side aggregation logic.

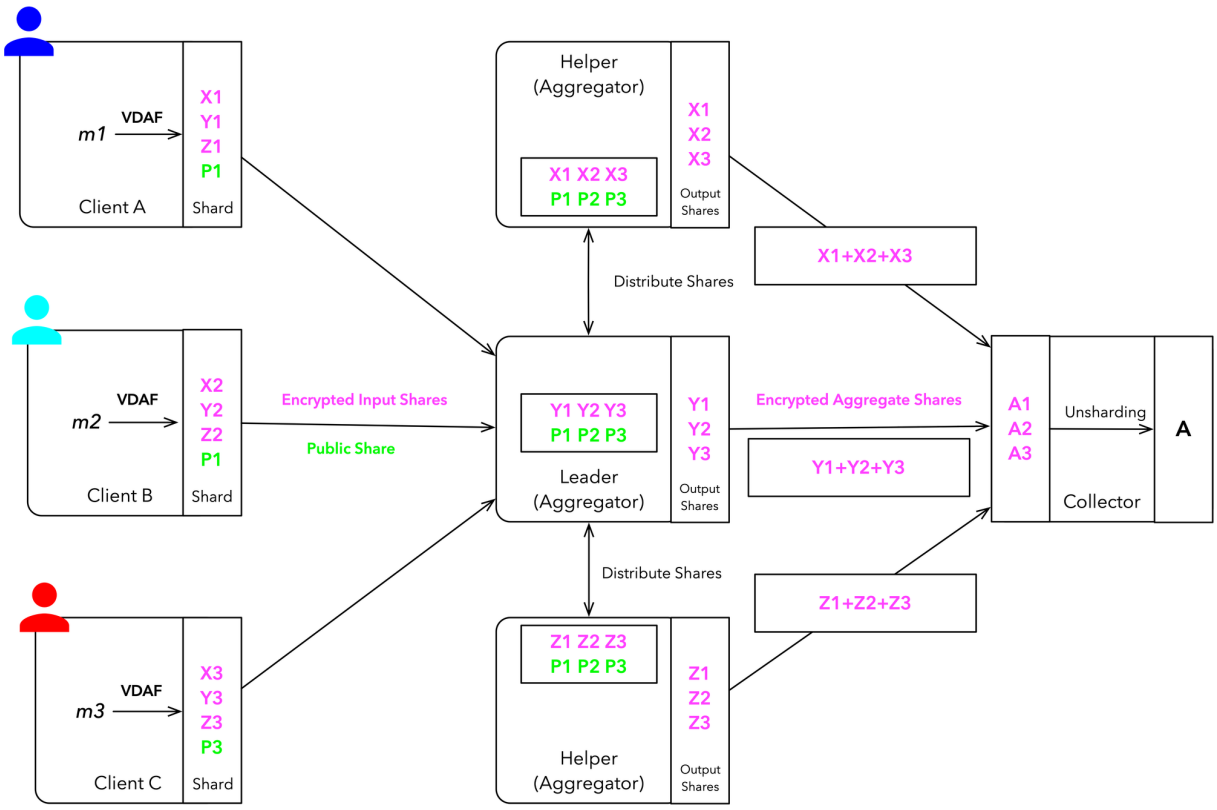

Figure 4: Each Client generates input shares from its measurement and distributes them among the Aggregators. (2) Each Aggregator converts each input share into an output share compatible with the aggregation function. This computation involves the aggregation parameter. Each Aggregator combines a sequence of output shares into its aggregate share and sends the aggregate share to the Collector. The Collector combines the aggregate shares into the aggregate result.

Prio

DAP has its roots in the Prio system, which was introduced in 2017 by Henry Corrigan-Gibbs and Dan Boneh from Stanford University. Prio, short for "Privacy-Preserving Aggregation of Randomized Inputs and Outputs," was designed as a privacy-preserving system for computing aggregate statistics over user data. It introduced the concept of secret-sharing client data among multiple servers and using zero-knowledge proofs to validate inputs without revealing individual data points. DAP builds upon and extends the ideas presented in Prio, incorporating improvements and optimizations developed in subsequent research. For instance, DAP utilizes VDAFs, which include an enhanced version of Prio called Prio3. This evolution from Prio to DAP represents a significant advancement in privacy-preserving measurement techniques, moving from academic research to a more comprehensive, standardized protocol suitable for wide-scale deployment.

Distributed Differential Privacy (DP)

While not a core part of the protocol, DAP is designed to be compatible with differential privacy techniques, which can provide additional protection against inference attacks on the aggregate results. The DAP protocol supports a distributed DP approach where the aggregator servers collaboratively add noise to results. This fits well with DAP's multi-party computation model. Unlike Local DP or Central DP, in Distributed DP multiple non-colluding parties collaborate to add noise in a distributed fashion. This balances privacy and utility without requiring a fully trusted party. Since each Aggregator is adding noise independently, privacy can be guaranteed even if all but one of the Aggregators is malicious.

Non-Collusion Assumption

DAP's privacy guarantees rely on at least one aggregator being honest. As long as at least one of them executes the protocol honestly no input is ever seen in the clear by any aggregator. As long as the Leader and Helper don't collude, individual measurements remain private.

Adoption

PPM techniques are seeing increasing adoption across various domains, with major tech companies leading the way in implementing these privacy-enhancing technologies.

Cloudflare's Daphne Project

Cloudflare has been at the forefront of PPM development with their Daphne project, an open-source implementation of a Distributed Aggregation Protocol (DAP) aggregator server. Daphne is designed to work as part of a two-party system, currently implementing the DAP Helper role.

Mozilla Firefox Telemetry

Mozilla has been exploring PPM techniques for collecting browser telemetry data. One potential application is privately aggregating and reporting client-side connection errors to an origin. This use case for DAP allows Mozilla to collect aggregate statistics about network errors (e.g., number of tcp.timed_out errors for a particular domain) without revealing sensitive information about individual users' browsing habits.

Google and Apple's COVID-19 Exposure Notifications

During heights of COVID-19, Google and Apple collaborated on the Exposure Notifications System (ENS) to enable automated COVID-19 exposure alerts while maintaining strong privacy guarantees. They further enhanced this system with Exposure Notification Privacy-preserving Analytics (ENPA), which allows public health authorities (PHA) to collect aggregate metrics without compromising individual privacy. ENPA uses multi-party computation to distribute the aggregation process across multiple servers, applies local differential privacy on user devices before data submission, and follow a rigorous cryptographic protocol ensuring that even Apple and Google cannot see individual contributions.

ENPA demonstrates how PPM can be applied to sensitive health data, enabling crucial public health insights while respecting individual privacy.

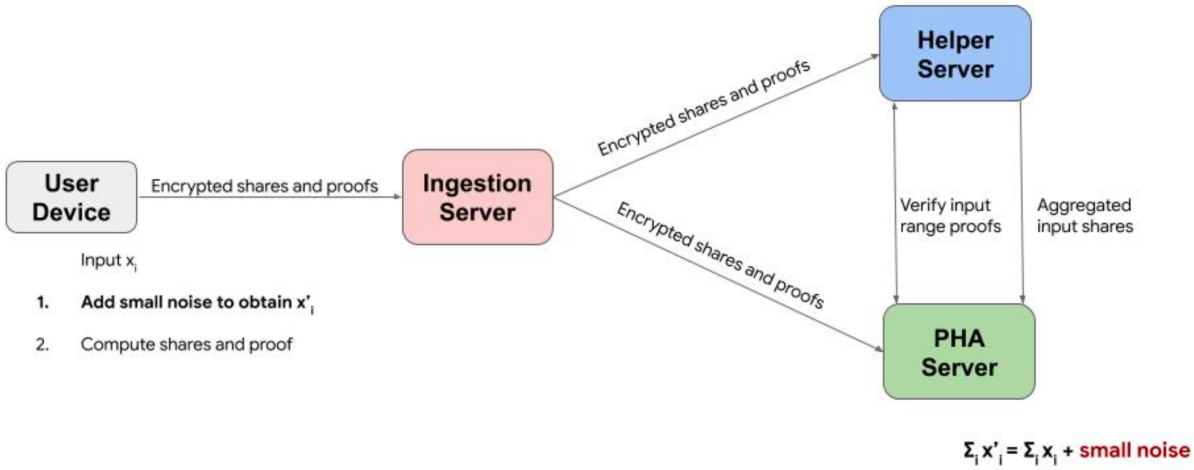

Figure 5: Each client adds local noise to its input value, then splits it into two cryptographic shares and computes a distributed zero- knowledge range proof for the value that is shared. The client encrypts one of the shares and the correspond- ing proof part under the public key of the PHA, and the other share and proof part under the public key of the Helper server. The client sends the resulting ciphertexts to the ingestion server, accompanied with an attestation (i.e., a device-specific cryptographic proof) of being a legitimate device.

Mozilla Firefox: Private Ad Attribution

In addition to telemetry data collection, Mozilla has been exploring privacy-preserving methods for ad attribution. They've been working on a Private Attribution API that aims to provide advertisers with conversion data without compromising user privacy. This system builds upon the concepts of Privacy

Preserving Measurement (PPM) and relies on differential privacy to provide a privacy guarantee that holds even if sites are malicious and attempt to abuse the API.

Mozilla's Private Attribution API involves multi-party computation which includes clients (user devices), aggregators, and a collector (the entity receiving the final aggregate data). User devices split their measurement data into secret shares, which are then encrypted and sent to different aggregators. To achieve differential privacy, noise is added to the aggregate results in a distributed manner by the aggregators. The system includes mechanisms to manage privacy budgets across different queries and advertisers.

Cloudflare: Making Network Error Logging Private with DAP and DP

Cloudflare has been exploring the use of the Distributed Aggregation Protocol (DAP) in combination with Differential Privacy (DP) to make Network Error Logging (NEL) private. This use case demonstrates how PPM techniques can be applied to improve existing web technologies.

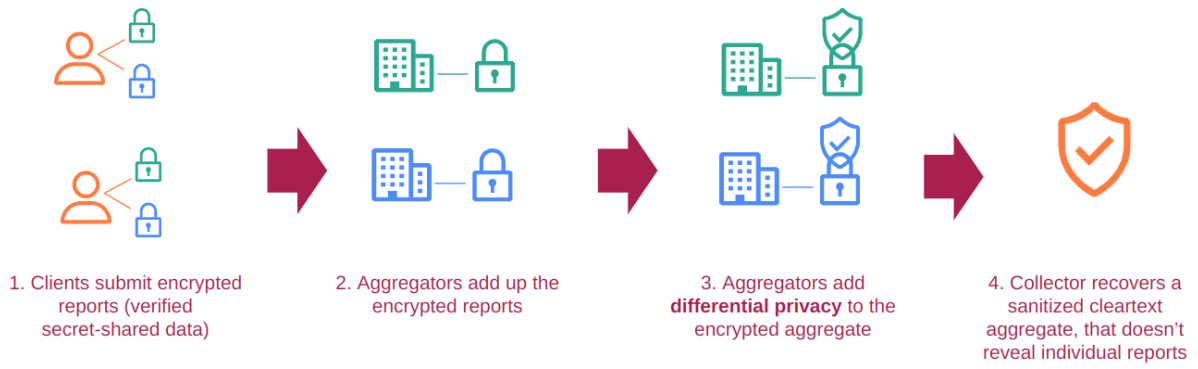

Figure 6: Aggregator Randomization. Image credits Cloudflare.

The system works as follows:

1. Client-side error logging: When a client encounters a network error, instead of sending a detailed error report directly to the origin server, it prepares a privacy-preserving report.
2. Report encryption: The client encrypts the report using the DAP protocol, splitting it into shares for multiple aggregators.
3. Secure aggregation: The encrypted reports are sent to Cloudflare's Daphne servers, which act as aggregators. These servers perform secure multi-party computation to aggregate the reports without seeing individual values.

4. Differential privacy: Noise is added to the aggregate results to achieve differential privacy. Cloudflare implements this using a distributed noise addition technique, where multiple non-colluding parties collaboratively add noise.
5. Histogram generation: The system produces differentially private histograms of error types, allowing website owners to understand common network issues without compromising individual user privacy.

Key considerations in this implementation include:

1. Privacy budget management: Cloudflare carefully manages the privacy budget (ε) across different types of queries and time periods.
2. Batch size optimization: The system is designed to balance privacy and utility by adjusting batch sizes for aggregation.
3. Error type encoding: Network errors are encoded into a fixed set of categories to enable efficient histogram generation.
4. Timestamp precision: Report timestamps are rounded to reduce the risk of timing-based attacks while still providing useful temporal data.

This approach allows Cloudflare to provide valuable insights to website owners about network errors affecting their users, while strongly protecting individual user privacy. It showcases how existing web infrastructure can be enhanced with state-of-the-art privacy-preserving techniques.

By implementing DAP with differential privacy for NEL, Cloudflare demonstrates a practical application of PPM that could serve as a model for privacy-enhancing other types of web measurements and analytics.

Conclusion

The development of DAP has been driven by collaboration between researchers, technology companies, and standardization bodies, aiming to create a robust and flexible system that can be applied to various real-world privacy-sensitive data collection scenarios.

While PPM has made good progress, there are still challenges to overcome: performance and scalability for complex computations, balancing privacy and utility for different applications, standardising protocols and APIs for interoperability, and educating users and organisations on the benefits and limitations.

Research is ongoing in areas like more efficient secure multi-party computation protocols, advanced composition techniques for differential privacy, PPM-friendly machine learning and AI techniques, and verifiable computation to ensure protocol compliance.