On the Differential Privacy and Interactivity of Privacy Sandbox Reports

Badih Ghazi*
Google
Mountain View, California, USA
badihghazi@gmail.com

Charlie Harrison*
Google
Texas, USA
csharrison@google.com

Arpana Hosabettu*
Google
Mountain View, California, USA
arpanah@google.com

Pritish Kamath*
Google
Mountain View, California, USA
pritishk@google.com

Alexander Knop*
Google
New York City, New York, USA
alexanderknop@google.com

Ravi Kumar*
Google
Mountain View, California, USA
ravi.k53@gmail.com

Ethan Leeman*
Google
Cambridge, Massachusetts, USA
ethanleeman@google.com

Pasin Manurangsi*
Google
Bangkok, Thailand
pasin@google.com

Mariana Raykova*
Google
New York City, New York, USA
marianar@google.com

Vikas Sahu*
Google
Mountain View, California, USA
vikassahu@google.com

Phillipp Schoppmann*
Google
New York City, New York, USA
schoppmann@google.com

Abstract

The Privacy Sandbox initiative from Google includes APIs for enabling privacy-preserving advertising functionalities as part of the effort around limiting third-party cookies. In particular, the Private Aggregation API (PAA) and the Attribution Reporting API (ARA) can be used for ad measurement while providing different guardrails for safeguarding user privacy, including a framework for satisfying differential privacy (DP). In this work, we provide an abstract model for analyzing the privacy of these APIs and show that they satisfy a formal DP guarantee under certain assumptions. Our analysis handles the case where both the queries and database can change interactively based on previous responses from the API.

Keywords

Ads, Privacy Sandbox, Aggregation Service, Differential Privacy, Individual Differential Privacy, Key Discovery, Requerying

1 Introduction

Third-party cookies have been a cornerstone of online advertising for more than two decades. They are small text files that are stored on a user's computer by websites other than the one they are currently visiting, allowing websites to track users across the internet and gather data about their browsing habits, thus enabling advertisers and publisher to measure performance of ad campaigns. However, in recent years, growing privacy concerns have led major web browsers to take action: both Apple's Safari [48] and Mozilla's Firefox [10] deprecated third-party cookies in 2019 and 2021, respectively; Google has made an informed choice proposal to explicitly ask users to choose whether they want to disable or enable third-party cookies [38]. This marks a significant change in the online advertising landscape, increasing the need for new solutions that prioritize user privacy.

In addition to the informed choice proposal, Google has led the Privacy Sandbox [31] initiative, which includes a set of privacy-preserving technologies aimed at replacing third-party cookie-based ad measurement. Two key components of this initiative are the Private Aggregation API (PAA) [32] and the Attribution Reporting API (ARA) [29], which seek to provide advertisers with *reports* that yield insights into the performance of ad campaigns while protecting user privacy. This is already rolled out and activated on roughly 3.5% and 22.1% of all page loads in Chrome respectively as of February 1, 2025 [40, 41]. These APIs offer various privacy guardrails, including a framework [30] for satisfying differential privacy (DP) [15, 16].

Recall that an advertising (ad, for short) campaign is a collection of impressions, each of those indicating a user interaction: either a user viewing an ad or clicking on one. The websites on which the ads are displayed, and possibly clicked, are referred to as *publishers*. The goal of an ad campaign is to drive useful actions on the advertiser website (e.g., purchases); these events are usually referred to as *conversions*. Therefore, goal of measurement for advertiser is to learn about campaign performance to answer questions like how many users did this ad campaign reach or how many converted as a result of this ad.

*Alphabetical author order.

Proceedings on Privacy Enhancing Technologies 2025(3), 382–397

https://doi.org/10.56553/popets-2025-0104

A main component of ARA and PAA are the *summary reports* [28], which enable the estimation of aggregate ad metrics such as the average number of conversions that can be attributed to a certain ad campaign [27], or the reach of the ad campaign [3, 37]. These reports are usually sliced by some parameters called *keys* (e.g., the ad-tech could slice by geography or a device type). Summary reports have been proposed in two flavors depending on whether the set of keys over which the aggregates are sliced is pre-specified or not; the latter is referred to as *key discovery* [2] and arises in practical settings where the set of attributes (pertaining to the ad, publisher, advertiser and/or user) is very large. In addition, it was proposed to extend summary reports to enable *requerying* [50], which would allow analysts to aggregate their measurement reports on different overlapping slices in an interactive manner (i.e., where the output of previous queries can influence the choice of the subsequent queries issued by the analyst).

ARA also offers *event-level reports* [26], which can provide for each ad impression, a discretized and noisy estimate of the list of conversions and conversion values attributed to this impression. This type of data enables the training of ad conversion models, which are machine learning models that predict the expected number of conversions (or the conversion value) that would be driven by an ad impression. The output of these ML models is typically used as input when determining the bid price in the online auctions that power automated bidding across the Web.

Despite being deployed in the web browsers of hundreds of millions of users and supporting critical ad functionalities, the privacy guarantee of neither ARA nor PAA has been formalized rigorously. This gap has been raised in recent work [42, 51].

Our Contributions. In this work, we address this gap and obtain several results formalizing the DP properties of ARA and PAA. Specifically:

- We prove that summary reports in ARA and PAA as well as event-level reports in ARA satisfy a formal DP guarantee even in a highly interactive setting (which captures common practical use cases) where the queries and the underlying database can change arbitrarily depending on previous reports.
- Our privacy proof for ARA and PAA summary reports applies to the aforementioned proposed extensions that would support key discovery and requerying.

Note, however, that both ARA and PAA protect only the *cross-website* (a.k.a. "third-party") information while the single-website (a.k.a. "first-party") information is assumed to be known to the ad-tech[1]. For example, if a user is logged-in on the publisher website, then the publisher knows the ads shown to the user and if a user is logged-in on the advertiser website, then the advertiser knows the user's conversions. However, if the two websites do not share the same login credentials, the ad-tech will be unable to attribute the conversion to the ad shown, without third party cookies. The ARA allows the ad-tech to access this attribution information, but in a privacy-preserving manner.

[1]In this work, we use the term *ad-tech* to refer to an analyst that performs analysis on ads and their attributed conversions, thereby helping publishers and advertisers with the placement and measurement of digital ads.

1.1 Related Work

Differential Privacy. Over the last two decades, DP [15, 16] has become a widely popular notion of privacy in data analytics and modeling, due to its compelling mathematical properties and the strong guarantees that it provides. It has been used in several practical deployments in government agencies (e.g., [7, 22]) and the industry (e.g., [21]); we refer the reader to [14] for a list covering many deployments. As DP is rolled out in ad measurement to replace third-party cookies, it is likely to become one of the largest, if not the largest, real-world deployment of DP, in terms of the number of daily queries and affected users.

Private Ad Measurement. In addition to the Privacy Sandbox on Google Chrome and Android, several other APIs have been proposed by various browsers, ad-tech companies, and researchers. These include Private Click Measurement (PCM) on Safari [49], SKAdNetwork on iOS [5], Interoperable Private Attribution (IPA) that was developed by Meta and Mozilla [43], Masked LARK from Microsoft [23], and Cookie Monster which was recently introduced in [42]. All of these APIs, except PCM, use DP to ensure privacy. We note that our proof, inspired by Cookie Monster [42] is using individual differential privacy accounting. Interestingly, while CookieMonster analyzes a similar setting, their notion of adjacency assumes that only impressions, or only conversions are known to the adversary. In other words, CookieMonster setting assumes the ability to hide impressions and/or conversions from the adversary, which may not be compatible with proposed APIs from most browsers, in particular the ARA and the PAA.

The recent work of [12] studied the interplay between attribution and DP budgeting for an abstract conversion measurement system. It showed that depending on the attribution logic (e.g., first-touch, last-touch, uniform), the privacy unit (e.g., per impression, per user $\times$ publisher, per user $\times$ advertiser), and whether contribution capping is performed before or after attribution, the sensitivity of the output aggregate can increase with the number of publishers and advertisers rendering the noise too high for accurate measurements.

There has also been recent work on optimizing the utility of the Privacy Sandbox ARA summary reports. For hierarchical query workloads, [11] gave algorithms for denoising summary reports, ensuring consistency, and optimizing the contribution budget across different levels of the hierarchy. On the other hand, [1] presented methods for optimizing the allocation of the contribution budget for general (not necessarily hierarchical) workloads.

Finally, there has been recent work on DP ad (click and conversion) modeling (e.g., [8, 13, 44]), some of it based on the Privacy Sandbox APIs.

Organization. Section 2 provides a simplified description of ARA and PAA; we focus primarily on details relevant for our analysis. Section 3 provides formal notations, covers basic background on DP and defines the notion of an interactive mechanism and adversary that is relevant for our modeling of ARA and PAA. In Section 4, we model summary reports of ARA and PAA and provide the formal DP guarantee for the same. In Section 5 we model event-level reports of ARA and provide a formal DP guarantee.

2 Privacy Sandbox Measurement APIs

We next describe the details of the ARA and PAA APIs and their role in collecting measurements about ads and their *attributed conversions*. (See Table 2 for a glossary of the involved terminology.)

The *Attribution Reporting API (ARA)* enables ad-techs to measure ad-conversions in a privacy-preserving manner (without third-party cookies). In particular, ARA supports two types of reports:

- *summary reports*, that allow collecting aggregated statistics of ad campaigns and their "attributed" conversions, and
- *event-level reports*, that associate a particular ad with very limited (and noisy) data on the conversion side, which are sent with a larger time delay.

The *Private Aggregation API (PAA)* is another API that also supports *summary reports* for collecting aggregated statistics in a privacy-preserving manner. While it is a general API (not necessarily about ads), a typical use-case is in estimating *reach* (the number of users who were exposed to an ad) and *frequency* (the number of users that were exposed to an ad k times, for each k).

We first provide a high-level overview of the *summary reports* in ARA and PAA (in Section 2.1), and of the *event-level reports* in ARA (in Section 2.2).

2.1 Summary Reports

Summary reports in both ARA and PAA rely on two main components: (i) the *client*, which runs in the browser, and (ii) the *aggregation service*, which runs in a trusted execution environment (TEE) [18]. Each client performs local operations based on browser activity and sends aggregatable reports to ad-techs. (The exact mechanism that generates these reports is different for ARA and PAA, and we explain this shortly.) Formally, an *aggregatable report* is a tuple (r, k, v, m) containing a *key* $k \in \mathcal{K} := \{0,1\}^{128}$, a *value* $v \in \{0, 1, \ldots, \Lambda_1\}$ (where $\Lambda_1 := 2^{16}$), a random identifier $r \in \mathcal{I} := \{0,1\}^{128}$ unique to each report (the Privacy Sandbox implementation uses AEAD [47] to ensure that the identifier is tamper-proof), and some metadata m that can depend on "trigger information" as explained later. Here, k and v are encrypted, where the secret key for decrypting is held by the aggregation service thereby ensuring that an ad-tech cannot see them and m is visible to the ad-tech in the clear, who can use it to batch the reports for aggregation. For simplicity, henceforth, we drop m from aggregate reports as it does not affect the privacy analysis.

2.1.1 Aggregation Service. Ad-techs can send a subset of the reports, $S = \{(r_1, k_1, v_1), \ldots, (r_n, k_n, v_n)\}$, they hold (filtered based on the available metadata) to the aggregation service, with the goal of learning, for any $k \in \mathcal{K}$, the aggregated value $w_k = \sum_{i:k_i=k} v_i$. The aggregation service aggregates the reports to generate a noisy version of this aggregated value. It supports two modes: one with *key discovery* and other without.

- Without key discovery, the ad-tech needs to provide a subset $L \subseteq \mathcal{K}$ of keys they are interested in, and they only get corresponding aggregated values w_ℓ for $\ell \in L$ after addition of noise sampled from the discrete Laplace distribution, denoted as $\mathsf{DLap}(a)$, which is supported over all integers with probability mass at x proportional to $e^{-a|x|}$.
- With key discovery, the aggregation service adds noise to each w_k sampled from the *truncated* discrete Laplace distribution, $\mathsf{DLap}_\tau(a)$, supported over integers in $[-\tau, \tau]$ with probability mass at x proportional to $e^{-a|x|}$. But the noisy values are released only for keys k where these noisy values are greater than τ. The subset L of keys is thus "discovered" from the reports themselves.

Algorithm 1 AggregationService (with or without Key Discovery)

[Parts specific to the service without key discovery are in blue.]
[Parts specific to the service with key discovery are in green.]

Params: Contribution budget $\Lambda_1 \in \mathbb{Z}_{>0}$,
privacy parameters $\varepsilon_* \in \mathbb{R}_{\geq 0}$ and $\delta_* \in [0,1]$,
State: Privacy budget trackers $B_\varepsilon : \mathcal{I} \to \mathbb{R}_{\geq 0}$, $B_\delta : \mathcal{I} \to [0,1]$.
Inputs: Privacy parameters $\varepsilon > 0$ and $\delta \in [0,1]$,
Aggregatable reports $(r_1, k_1, v_1), \ldots, (r_n, k_n, v_n)$,
Subset $L = \{\ell_1, \ldots, \ell_m\} \subseteq \mathcal{K}$ of keys.
Output: Summary report $(\ell_1, w_1), \ldots, (\ell_m, w_m)$.
if $\exists i \in [n]$ such that $B_\varepsilon(r_i) + \varepsilon > \varepsilon_*$ or $B_\delta(r_i) + \delta > \delta_*$ **then**
  Abort *[Privacy budget violated for some report.]*
else
  $B_\varepsilon(r_i) \leftarrow B_\varepsilon(r_i) + \varepsilon$ for all $i \in [n]$
  $B_\delta(r_i) \leftarrow B_\delta(r_i) + \delta$ for all $i \in [n]$
Let $L \subseteq K$ be the set of distinct keys among $k_1, \ldots, k_n$
$\tau \leftarrow \Lambda_1 \cdot (1 + \log(\Lambda_0/\delta)/\varepsilon)$
for $\ell \in L$ **do**
  $w_\ell \leftarrow z_\ell + \sum_{j\,:\,k_j=\ell} v_j$ for $z_\ell \sim \mathsf{DLap}_\tau(\varepsilon/\Lambda_1)$
return $\{(\ell, w_\ell) : \ell \in L \text{ and } w_\ell > \tau\}$

In addition, the aggregation service uses a *privacy budget service* to enforce that the privacy budget is respected for each aggregatable report.[2] This entails maintaining $B_\varepsilon : \mathcal{I} \to \mathbb{R}_{\geq 0}$ that tracks, for each report, the sum of ε values with which the said report has participated in aggregation requests; the privacy budget service enforces that this sum never exceeds a fixed value ε_*.[3] Additionally, if using the key discovery mode, an additional state of $B_\delta : \mathcal{I} \to [0,1]$ is maintained that tracks the number of times a report participated in an aggregation request; and it is enforced that this never exceeds a fixed value ℓ_*.[4] For simplicity we present the aggregation service (with or without key discovery) and the privacy budget service together in Algorithm 1. All pseudocode we provide in this paper are for explaining the underlying functionality, and not meant to reflect actual implementation of these APIs.

Remark 2.1. *We note that the currently supported implementation of the aggregation service is weaker than our description in that the key discovery mode is not supported and $\ell_* = 1$ is enforced for all reports (that is, each report can participate in at most one aggregation request). However, our results hold even under these proposed extensions of key discovery [2] and requerying [50].*

[2] In this paper we assume that the aggregation service is tracking budget "per report", but in reality it uses a coarse id called *shared report id* that consists of the API version, the website that generated the report, and the reporting time in seconds [34] to track a single budget for all reports assigned to the shared report id.
[3] Currently this value is equal to 64 [39].
[4] Currently this value is equal to 1 for both ARA and and PAA; this corresponds to no "re-querying" [50].

Next, we describe the clients for ARA and PAA that generate aggregatable reports based on cross-site information and send them to the ad-tech.

2.1.2 ARA Client. We explain the workings of ARA-SR-Client (the client that supports ARA summary reports in the browser) in context of an example visualized in Figure 1. Suppose a publisher's page (`some-blog.com`) displays ads as part of a "Thanksgiving" campaign about sneakers, sandals, and flip-flops that can be purchased on an advertiser's page (`shoes-website.com`). The ad-tech would like to measure the amount of money spent on shoes on the advertiser's page that could be attributed to these ads, and in particular, with an attribution rule that "the purchase must happen within three days after an ad was shown".

To use ARA-SR-Client, the ad-tech annotates both the publisher and advertiser pages with additional JavaScript or HTML that points to a URL with a specific HTTP header (see [29] for details). ARA-SR-Client can get invoked in two ways, either when the ad is shown on the publisher's page, or when a conversion happens on the advertiser's page as described below.

- On the *publisher page*, the ad-tech registers a so-called "*attribution source*" with the ARA-SR-Client in the browser (e.g., corresponding to a view/click ad event). In our example the source corresponds to an ad shown on `some-blog.com`. This entails specifying a tuple (`srcId`, `dest`, `expDate`, `srcFilt`, `srcKey`), where
  - `srcId` is an identifier associated to the source event (in our example it is a random id generated when an ad is shown on `some-blog.com`),
  - `dest` is the advertiser domain where the conversion could happen (`shoes-website.com` in our example),
  - `expDate` is an expiration date for attributing conversions to the source (three days in our example),
  - `srcFilt` is a set of filter keys, each being a bounded bit string, (treated as strings "sneakers", "sandals", and "flip-flops" in our example for simplicity),
  - `srcKey` $\in \mathcal{K}$ is a key associated with the source (in our example we take it to be the string "`Thanksgiving:`" for illustration).

  Each source registration invokes the "Source registration" part of Algorithm 2, which updates a set $\mathcal{S}$ of registered sources.
- On the *advertiser page*, the ad-tech registers a so-called "*trigger*" corresponding to a qualifying user activity (such as a purchase on `shoes-website.com` in our example) by specifying a tuple (`trigId`, `trigFilt`, `trigKey`, `trigValue`), where
  - `trigId` is an identifier associated with the trigger (in our example it is a random id generated when the purchase happened on `shoes-website.com`),
  - `trigFilt` is a set of filter keys similar to `srcFilt` (in our example it could be either "sneakers", or "sandals", or "flip-flops"),
  - `trigKey` $\in \mathcal{K}$ is a key associated to the trigger (in our example we take it to be the string "`shoes-value`" for illustration).
  - `trigValue` $\in \{0, 1, \ldots, \Lambda_1\}$ is a value associated to the trigger (in our example it is the price of the shoes purchased).

  Each trigger registration invokes the "Trigger registration" part of Algorithm 2.

At any point of time, ARA-SR-Client uses $L_1 : \mathcal{S} \to \mathbb{Z}_{\geq 0}$ to track the sum of all values contributed, and $L_0 : \mathcal{S} \to \mathbb{Z}_{\geq 0}$ to denote the number of non-zero values contributed per registered source, with $L_0(s)$ and $L_1(s)$ initialized to 0 at the time of registration of source $s \in \mathcal{S}$.

When a user visits the advertiser's page and a trigger is registered, ARA-SR-Client (Algorithm 2) matches the trigger to the most recently registered *active* source (namely, those with `expDate` ahead of the current time) and generates an aggregatable report; aggregatable reports for ARA contain `trigId` as part of its metadata (earlier referred to as m in (r, k, v, m)). If no source is matched, a null report, denoted $(r, \perp, \perp)$ in Algorithm 2 is sent. Such reports are ignored by the aggregation service (Algorithm 1).

Remark 2.2. *In reality the metadata m and the time the report is sent can allow the ad-tech to associate the received report with the particular device. Moreover, if the trigger does not get attributed to any source, and* `trigId` *is not set, the corresponding null report is not sent (see [35]). This could potentially allow the ad-tech to know if a trigger was attributed to a source or not. These are handled by the ARA client in practice with some heuristics such as sending reports without a trigger id with some delay, as well as sending some fake null reports with small probability. However, these heuristics would not allow us to prove a formal DP guarantee, so we do not consider this case.*

Remark 2.3. *ARA-SR-Client (Algorithm 2) is using the so-called "last-touch" attribution, namely, that in presence of several potentially matching sources, the most recently registered one is chosen. While we choose to only consider last-touch attribution for simplicity, our results hold for any attribution method (as long as any trigger is fully attributed to only one source). Indeed, ARA uses a more involved attribution strategy where impressions can be assigned priority (at source registration) and the trigger is attributed to the last source with the highest priority.*

Here is how our example would play out, as visualized in Figure 1. A publisher's page (`some-blog.com`) displays an ad, on Nov 18th about shoes that can be purchased on an advertiser's page (`shoes-website.com`). When an ad is displayed at `some-blog.com` the source gets registered with a random `srcId`, with `dest` equal to "`shoes-website.com`", an expiry date of Nov 21st, that is three days into the future from the time the ad was shown, a set of `srcFilt = {"sneakers", "sandals", "flip-flops"}` that restricts which purchases can be attributed to this source and, a source key `srcKey` = "`Thanksgiving:`" to tie this ad to a certain campaign.

When the user subsequently purchases a sneaker from the advertiser's page, the advertiser can choose to register a trigger with a random `trigId`, a set of `trigFilt = {"sneakers"}` that restricts the potential sources that this conversion can be attributed to, a trigger key `trigKey` = "`shoes-value`" to record the interpretation of the value, and finally `trigValue` = 70, which is the price of the sneaker.

The ARA-SR-Client (Algorithm 2) then attributes the trigger to the source and generates an aggregatable report (r, k, v) with the key being k = "`Thanksgiving:shoes-value`" (where for ease of illustration we use string concatenation instead of bitwise-OR), value $v = 70$, and r being a random id.

Recall that the ad-tech knows details of the ad impression that was displayed on the publisher website, as well as details of the conversion that happened on the advertiser's website; these are

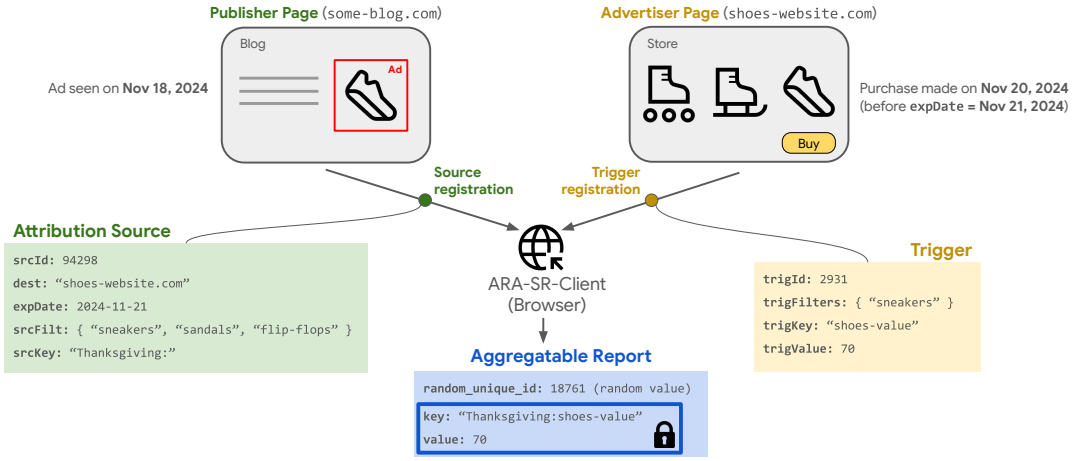

Figure 1: Illustrative example of an aggregatable report generated by ARA-SR-Client.

Algorithm 2 ARA-SR-Client

Params: Contribution and sparsity budgets $\Lambda_1, \Lambda_0 \in \mathbb{Z}_{>0}$.
State:
- Set of registered sources $\mathcal{S}$,
- Sparsity Budget Tracker $L_0 : \mathcal{S} \to \mathbb{Z}_{\geq 0}$,
- Contribution Budget Tracker $L_1 : \mathcal{S} \to \mathbb{Z}_{\geq 0}$.

Source registration:
On input $s = (\texttt{srcId}, \texttt{dest}, \texttt{expDate}, \texttt{srcFilt}, \texttt{srcKey})$
$\mathcal{S} \leftarrow \mathcal{S} \cup \{s\}$ *[Add s to set of registered sources.]*
$L_0(s) = 0$ and $L_1(s) = 0$

Trigger registration:
On input $(\texttt{dest}, \texttt{trigId}, \texttt{trigFilt}, \texttt{trigKey}, \texttt{trigValue})$.
$r \leftarrow$ a random report id in $\mathcal{I}$
if $\texttt{trigValue} > 0$ **then**
  for active $s \in \mathcal{S}$ (in reverse chronological order) **do**
    if $\texttt{dest} = s.\texttt{dest}$ and $\texttt{trigFilt} \cap s.\texttt{srcFilt} \neq \emptyset$ **then**
      if $L_0(s) + 1 \leq \Lambda_0$ and $L_1(s) + \texttt{trigValue} \leq \Lambda_1$ **then**
        $v \leftarrow \texttt{trigValue}$
        $k \leftarrow$ bit-wise OR of $\texttt{srcKey}$ and $\texttt{trigKey}$
        $L_0(s) \leftarrow L_0(s) + 1$
        $L_1(s) \leftarrow L_1(s) + \texttt{trigValue}$
        Send report (r, k, v) and halt
Send null report $(r, \perp, \perp)$

referred to as "first-party" information. However, without third-party cookies, the ad-tech cannot link the two events as happening on the same browser and thus cannot attribute the conversion to the impression. The ARA helps ad-techs access this "third-party" information linking the conversion to the impression. But it does so in a privacy-preserving manner by enforcing that the total number of generated reports associated to any source is at most Λ_0, and that the total value of such generated reports is at most Λ_1. We provide the formal privacy guarantees implied by these properties of ARA summary reports as Theorem 4.4 in Section 4.

2.1.3 PAA Client & Shared Storage. The PAA client is typically used in tandem with the Shared Storage API [36]. Shared Storage is a key-value database stored on the browser that can be accessed by the code using PAA across any websites visited on a browser, which the ad-tech can annotate with their code.

Recall that ARA-SR-Client tracks the contribution and sparsity budgets for each "source". However, unlike ARA, the PAA does not have a concept of registering a source. PAA instead enforces contribution and sparsity budgets separately for each ad-tech and each "time-window". Formally, let $\mathcal{U}$ denote the set of all devices, and let $\mathcal{T}$ denote the partition of all time into contiguous time windows, each of a fixed length.[5] We use Ξ to denote the shared storage, where Ξ_u denotes the part of storage accessible to the device $u \in \mathcal{U}$. At any point of time, PAA-SR-Client uses $L_1 : \mathcal{U} \times \mathcal{T} \to \mathbb{Z}_{\geq 0}$ to track the sum of all values contributed and $L_0 : \mathcal{U} \times \mathcal{T} \to \mathbb{Z}_{\geq 0}$ to track the number of non-zero values contributed per (u, t), with $L_0(u, t)$ and $L_1(u, t)$ initialized to 0 at the start of time window t for any $u \in \mathcal{U}$.

We explain the working on PAA-SR-Client in context of an example visualized in Figure 2. Assume an ad-tech has four ad campaigns (for shoes, pants, jackets, and shirts) running in parallel and they want to estimate the reach of each campaign (the number of people exposed to the ads from campaigns) across different publisher websites. Suppose the ad-tech expects that most of the people would only see at most two ads within a single time window $t \in \mathcal{T}$.

To use the PAA and Shared Storage APIs, the ad-tech annotates any webpage it has access to with a JavaScript that can read and write to Shared Storage and can register an ad *event*; the registration requires providing a method π that maps the current state of the shared storage, to the next state of the storage, a key $k \in \mathcal{K}$, and a corresponding value $v \in \{0, 1, \ldots, \Lambda_1\}$ (in our example the key is the campaign name and the value is $\Lambda_1/2$ if the ad from that campaign was seen for the first time and 0 otherwise; this is looked up from shared storage). When an event is registered, PAA-SR-Client (Algorithm 3) checks that the new addition would not violate the contribution and sparsity budget for the ad-tech at the current time

[5] Each time window is 10 minutes long in the current implementation of PAA [24].

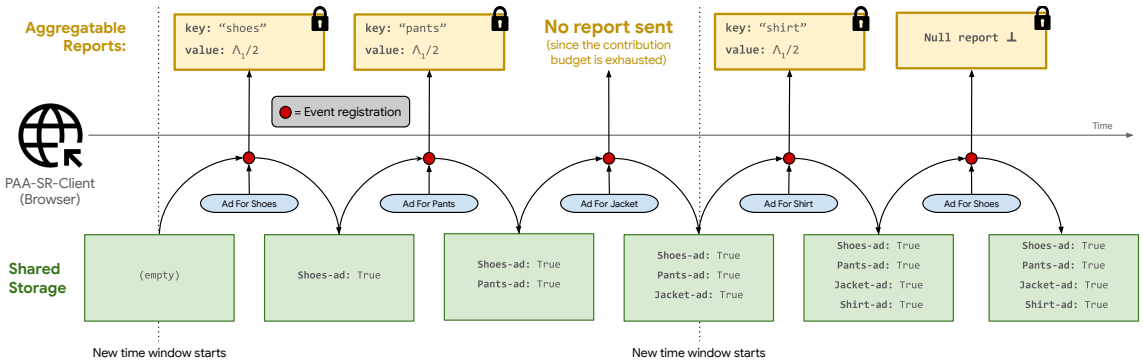

Figure 2: Illustrative example of an aggregatable report generated by PAA-SR-Client.

Algorithm 3 PAA-SR-Client

Params: Contribution and sparsity budgets $\Lambda_1, \Lambda_0 \in \mathbb{Z}_{>0}$.
State:

- Shared storage $\Xi \in \mathfrak{X}^{\mathcal{U}}$ (we use $\mathfrak{X}$ to denote the set of possible states of the shared storage, one for each device $u \in \mathcal{U}$),
- Sparsity Budget Tracker $L_0 : \mathcal{U} \times \mathcal{T} \to \mathbb{Z}_{\geq 0}$,
- Contribution Budget Tracker $L_1 : \mathcal{U} \times \mathcal{T} \to \mathbb{Z}_{\geq 0}$.

Event Registration
On following inputs:

- The device u registering the event,
- Time window t when event is invoked, and
- Function $\pi : \mathfrak{X} \to \mathfrak{X} \times \mathcal{K} \times \{0, 1, \ldots, \Lambda_1\}$ that on input being the current state of shared storage Ξ, returns its next state in addition to a key value pair (k, v).

$\Xi_u, k, v \leftarrow \pi(\Xi_u)$ *[Note: Ad-tech does not learn what is inside Ξ_a. Moreover, even the returned (k, v) is not visible to ad-tech.]*
$r \leftarrow$ a random report id in $\mathcal{I}$
if $v > 0$ **then**
  if $L_0(a, t) + 1 \leq \Lambda_0$ and $L_1(a, t) + v \leq \Lambda_1$ **then**
    $L_0(u, t) \leftarrow L_0(u, t) + 1$
    $L_1(u, t) \leftarrow L_1(u, t) + v$
    Send aggregatable report (r, k, v) and halt
Send null report $(r, \perp, \perp)$

window, and if that is indeed the case, it generates an aggregatable report (r, k, v) for a random ID $r \in \mathcal{I}$.

To run through our example, the user sees first the ads for shoes and pants and the reports are sent; next the user sees an ad for a jacket, but the contribution budget is exhausted so no report is sent. Finally, in the next time window, the user sees an ad for a shirt and the report is sent since the budget is refreshed. However, when the user sees an ad for shoes again, a null report is sent because the contribution is 0, since shared storage indicates that a shoes ad was seen previously.

Similar to case of ARA, recall that the ad-tech knows details of any specific event in PAA, which is considered "first-party" information. However, without third-party cookies, the ad-tech cannot link information across events happening on the same browser across different websites; this would be "third-party" information. The PAA helps ad-techs access this "third-party" information via the use of the Shared Storage API. But it does so in a privacy-preserving manner by enforcing that the total number of generated reports for any ad-tech in any time window is at most Λ_0, and that the total value of such generated reports is at most Λ_1. The access to Shared Storage is also "sandboxed" in a manner that the information within can only be used for purposes of generating the aggregatable report. We provide the formal privacy guarantees implied by these properties of PAA summary reports as Theorem 4.6 in Section 4.

2.2 Event-level Reports

In addition to summary reports, ARA also supports *event-level* reports [33]. Before describing the event-level reports formally, we consider an example visualized in Figure 3. Again, suppose a publisher's webpage (`some-blog.com`) displays an ad for footwear sold on an advertiser's site (`shoes-website.com`).

To use ARA-Event-Client, the ad-tech annotates both the publisher and advertiser pages in a manner that is similar to case of summary reports (see Section 2.1.2). ARA-Event-Client can get invoked in three ways, (i) when the ad is shown on publisher's page, (ii) when a conversion happens on the advertiser's page, and (iii) on the passing of a "reporting window", as explained below (this happens automatically without any action from the ad-tech). For simplicity, we first describe the "noiseless" version of ARA-Event-Client in Algorithm 4.

- On the *publisher page*, the ad-tech registers an *attribution source* with ARA-Event-Client in the browser. This entails specifying a tuple (`srcId`, `dest`, `expDate`, `srcFilt`, `maxRep`, `trigSpec`), where
  - ▹ `srcId`, `dest`, `expDate`, `srcFilt` are the same as in the case of the ARA summary reports,
  - ▹ `maxRep` is the maximum number of event-level reports that can be attributed to this source (in our example it is 3),
  - ▹ `trigSpec`,[6] short for "trigger specificiation", is a list of at most 32 elements (in our case we have two specifications, one for "sneakers" and one for "sandals") each consisting of

[6] In real implementation a list of possible values of trigger data is passed separately and there is a default specification, but we ignore this here for simplicity.

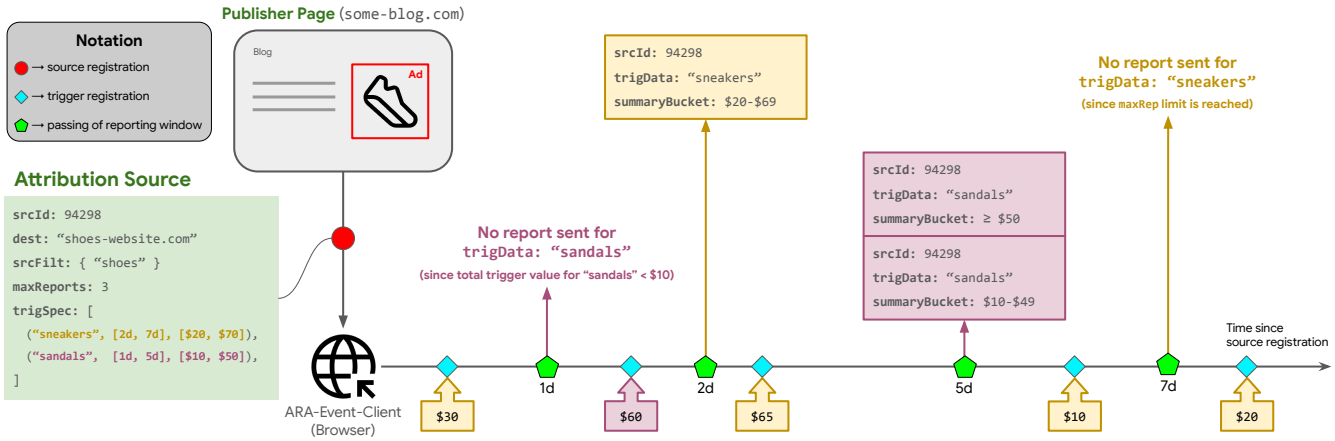

Figure 3: Illustrative example of event-level reports generated by ARA-Event-Client.

- `trigData` ∈ TD, where TD is the set of possible values that `trigData` can take; while this is restricted to be a 5-bit integer, we treat `trigData` as a string in our example for ease of visualization.
- at most five *reporting windows* that specify when the new reports should be sent; each reporting window is time (in seconds) from the source registration time.
- *summary buckets*, an increasing list of integer thresholds such that a report is sent when the threshold is met (subject to `maxRep` limit).

Each source registration invokes the "Source registration" part of Algorithm 4; visualized by the red dot in Figure 3. This adds the source to the list of active registered sources $\mathcal{S}$ and initializes the current values and last reported bucket values to zero.

- On the *advertiser page*, the ad-tech registers a *trigger* corresponding to a qualifying user activity (such as a purchase on `shoes-website.com` in our example) by specifying a tuple containing (`dest`, `trigId`, `trigFilt`, `trigData`, `trigValue`), where
  - `dest`, `trigId`, `trigFilt` are the same as in the case of ARA summary reports,
  - `trigData` is data associated with the trigger, which must match one from `trigSpec` (if it does not match any in the specification it gets ignored), and
  - `trigValue` is an integer value associated with the trigger (in our example the amount spent on the shoes).

  Each trigger registration invokes the "Trigger registration" part of Algorithm 4; visualized by the cyan diamond in Figure 3. Here ARA-Event-Client searches for an active source that this trigger could be attributed to, and upon finding one, adds the value of the current trigger to the value associated to the source for this `trigData`.

Finally, for each registered source s, whenever the amount of time passed equals the reporting window for any `trigData` in s.`trigSpec`, a report is sent to the ad-tech using Algorithm 4; in our example, 1 day after the source is registered, the first reporting window for `trigData` = "`sandals`" is passed and ARA-Event-Client get invoked. All such invocations associated to passing of reporting windows is visualized as a green pentagon in Figure 3.

Algorithm 4 ARA-Event-Client (Noiseless version).

State:
- Sequence of active registered sources $\mathcal{S}$,
- Current value $V : \mathcal{S} \times \text{TD} \to \mathbb{Z}_{\geq 0}$,
- Last reported bucket $U : \mathcal{S} \times \text{TD} \to \mathbb{Z}_{\geq 0}$,
- Number of reports sent $N : \mathcal{S} \to \mathbb{Z}_{\geq 0}$.

Source registration:
On input s = (`srcId`, `dest`, `expDate`, `srcFilt`, `maxRep`, `trigSpec`)
$\mathcal{S} \leftarrow \mathcal{S} \cup \{s\}$ *[Add s to set of active sources.]*
$N(s) \leftarrow 0$
for `trigData` ∈ TD **do**
  $V(s, \texttt{trigData}) \leftarrow 0$ and $U(s, \texttt{trigData}) \leftarrow 0$

Trigger registration:
On input (`dest`, `trigId`, `trigFilt`, `trigData`, `trigValue`)
if `trigValue` > 0 **then**
  for $s \in \mathcal{S}$ (in reverse chronological order) **do**
    if `dest` = s.`dest` and `trigFilt` ∩ s.`srcFilt` ≠ ∅ **then**
      if `trigData` listed in s.`trigSpec` **then**
        $V(s, \texttt{trigData}) \leftarrow V(s, \texttt{trigData})$ + `trigValue`
        [Add to value associated to this source and `trigData`.]
        break

At passing of time window for (s, `trigData`) ∈ $\mathcal{S}$ × TD:
Let $b_1, \ldots, b_k \in \mathbb{Z}_{>0}$ be the list of associated summary buckets to `trigData` in `trigSpec` of source s, in increasing order.
for $i = 1, \ldots, k$ **do**
  if $U(s, \texttt{trigData}) < b_i \leq V(s, \texttt{trigData})$ **then**
    $U(s, \texttt{trigData}) = b_i$
    if $N(s) <$ s.`maxRep` **then**
      $N(s) \leftarrow N(s) + 1$
      Send report (s.`srcId`, `trigData`, b_i) **to ad-tech**

We now walk through the behavior of ARA-Event-Client in the example visualized in Figure 3; we present the "noiseless" behavior of the algorithm before describing the noisy version.

- **Day 0:** An ad is shown on publisher's page and the source s gets registered.

- **Day 1:** The user buys \$30 sneakers from advertiser's page. This is attributed to source s and $V(s, \texttt{sneakers})$ is set to 30.
  At the end of day 1, the reporting window for `sandals` is passed, but no report is generated because $V(s, \texttt{sandals})$ is still 0.
- **Day 2:** The user buys \$60 sandals. This is attributed to source s, and $V(s, \texttt{sandals})$ is set to 60.
  At the end of day 2, the reporting window for `sneakers` is passed, and a report is generated indicating a bucket value of \$20-\$69. At this point $U(s, \texttt{sneakers})$ is 20.
- **Day 4:** The user buys \$65 sneakers. This is attributed to source s and $V(s, \texttt{sneakers})$ gets incremented to become 95.
- **Day 5:** At the end of day 5, the second reporting window for `sandals` is passed, and two reports are generated for sandals as $V(s, \texttt{sandals}) = 60$, which exceeds both summary bucket values of \$10 and \$50.
- **Day 6:** The user buys \$10 sneakers. This is attributed to source s and $V(s, \texttt{sneakers})$ is incremented to become 105.
- **Day 7:** At the end of day 7, the second reporting window for `sneakers` is passed, but no reports are sent, as the maximum number of reports 3 for this source has been reached. Furthermore, the source now gets marked as inactive and no further purchases get attributed to this source.

Noisy Version of ARA-Event-Client. Note that for each source, the set O of possible combinations of reports that could be sent is finite. In our example, reports are sent for "`sandals`" only at the end of Day 1 and Day 5. The number of reports sent on each of the two days is one among the 6 possible options shown in Table 1. Similarly, the number of reports sent at the end of Day 2 and Day 7 is also among 6 possible options. While these lead to 36 possible configurations of reports sent, 9 of them correspond to sending of 4 reports, which is larger than the maximum limit of 3 reports that could be sent. Thus, in total, there are 27 possible valid configurations of reports that could get sent. The private version of event-level reports works as follows: with probability $\frac{e^\varepsilon - 1}{e^\varepsilon + |O| - 1}$ we proceed like in the noiseless version in Algorithm 4, otherwise we sample an element of O uniformly at random, and generate responses accordingly. This choice can be made at the time of source registration, even before any triggers are observed. We provide the formal privacy guarantees implied by these properties of ARA event-level reports as Theorem 5.1 in Section 5.

Day 1	Day 5
0	0
0	1
0	2
1	0
1	1
2	0

Table 1: Possible number of reports sent on Day 1 and Day 5 for "`sandals`".

3 Differential Privacy

We use the notion of differential privacy (DP), which typically considers *mechanisms* $\mathcal{M} : \mathcal{D} \to \Delta(\mathcal{R})$ that map input datasets $D \in \mathcal{D}$ to probability distributions over a set $\mathcal{R}$ of responses. Central to the notion of DP, is the definition of "adjacency" of databases. Loosely speaking two databases $D, D' \in \mathcal{D}$ are said to be adjacent if they "differ in one record". We defer the definition of a database and the notion of adjacencies relevant for the analysis in each application to Section 4. But for any notion of database and adjacency, differential privacy (DP) quantifies the ability (or lack thereof) of an adversary to distinguish two "adjacent" datasets D and D' by observing a sample from $\mathcal{M}(D)$ or $\mathcal{M}(D')$.

Definition 3.1 ((ε, δ)-Indistinguishability). Two distributions P, Q are said to be (ε, δ)-*indistinguishable*, denoted $P \approx_{\varepsilon,\delta} Q$ if for all events W, it holds that

$$P(W) \le e^\varepsilon Q(W) + \delta \text{ and } Q(W) \le e^\varepsilon P(W) + \delta.$$

Definition 3.2 (Differential Privacy). [15, 16] A mechanism $\mathcal{M} : \mathcal{D} \to \Delta(\mathcal{R})$ satisfies (ε, δ)-*DP* if for all adjacent datasets $D, D' \in \mathcal{D}$, it holds that $\mathcal{M}(D) \approx_{\varepsilon,\delta} \mathcal{M}(D')$. The special case of $(\varepsilon, 0)$-DP is denoted as ε-DP for short.

We will often refer to parameter ε or both ε and δ in the definition of (ε, δ)-DP as the *privacy budget*.

However, as we discuss shortly, modeling the setting of ARA and PAA as mechanisms operating on a "static" database does not capture the interactive nature of these systems. Hence we consider a stronger notion of "interactive mechanisms" and "interactive adversaries" defined in Section 3.1.

Since the aggregation service adds noise from the (truncated) discrete Laplace distribution, we rely on the following fact to prove DP properties of summary reports (for both ARA and PAA). For any integer $d \in \mathbb{Z}_{>0}$, let $\mathsf{DLap}_\tau(a)^{\otimes d}$ be the distribution over $\mathbb{Z}^d$ where each coordinate is drawn independently from $\mathsf{DLap}_\tau(a)$.

Fact 3.3 ([20]). *For all $\varepsilon > 0$ and integer $\Delta > 0$, and vectors $u, v \in \mathbb{Z}^d$ such that $\|u - v\|_1 = \Delta$, the distributions P and Q drawn as $u + \xi$ and $v + \xi$ for $\xi \sim \mathsf{DLap}(\varepsilon/\Delta)^{\otimes d}$ satisfy $P \approx_{\varepsilon,0} Q$.*

Furthermore, if $u - v$ has at most s non-zero coordinates, then for all $\delta > 0$ and $\tau \ge \Delta \cdot (1 + \log(s/\delta)/\varepsilon)$, the distributions P and Q drawn as $u + \xi$ and $v + \xi$ for $\xi \sim \mathsf{DLap}_\tau(\varepsilon/\Delta)^{\otimes d}$ satisfy $P \approx_{\varepsilon,\delta} Q$.

While the privacy guarantee of the discrete Laplace mechanism was studied in [20], we include a proof for the case of truncated discrete Laplace noise in Section A for completeness.

3.1 Interactive Mechanisms

As noted earlier, DP is typically defined for "one-shot" mechanisms that map databases to probability distributions over a response set. However, there are two key reasons why the generation process of ARA/PAA summary reports is not a "one-shot" mechanism:

(1) There is adaptivity in the choice of contribution values, which can be changed based on previously observed summary reports. For example, in context of ARA, the ad-tech can change the scale and interpretation of `trigValue` when making contributions based on previously observed summary reports.
(2) The ad-tech can have some influence on the new events that get added to the database themselves. For example, in the context of ARA, the advertiser might decide based on summary reports obtained on day 1 to change the product price on day 2, which could influence the number of subsequent conversions.

To argue the DP properties of summary and event-level reports, it is helpful to model their generation process as an interaction between a mechanism and an adversary defined below. Abstractly speaking, let $\mathcal{D}$ denote the set of all "databases", and let $\mathcal{Q}$ and $\mathcal{R}$ denote a set of "queries" and "responses" as used below.

Algorithm 5 Interactive Transcript IT $(\mathcal{M} : \mathcal{A})$.

Inputs: ▷ Interactive mechanism $\mathcal{M}$ with initial state S_0,
▷ Interactive adversary $\mathcal{A}$
$\Pi \leftarrow ()$ and $t \leftarrow 1$ *[Empty transcript.]*
while $\mathcal{A}(\Pi) \neq$ ✋ **do**
$(D_t, q_t) \leftarrow \mathcal{A}(\Pi)$ *[Adversary creates database & query.]*
$(S_t, r_t) \sim \mathcal{M}(S_{t-1}, D_t, q_t)$ *[Mechanism samples response.]*
$\Pi \leftarrow \Pi \circ r_t$ and $t \leftarrow t + 1$
return Π

Definition 3.4 (Interactive Mechanism). An *interactive mechanism* with state set $\mathcal{S}$ is represented by an initial state $S_0 \in \mathcal{S}$ and a function $\mathcal{M} : \mathcal{S} \times \mathcal{D} \times \mathcal{Q} \to \mathcal{S} \times \Delta(\mathcal{R})$, that maps the current state $S \in \mathcal{S}$, a database $D \in \mathcal{D}$ and a query $q \in \mathcal{Q}$ to the next state $S' \in \mathcal{S}$ and a distribution over responses $r \in \mathcal{R}$.

We will often abuse notation to denote the output of $\mathcal{M}(S, D, q)$ as (S', r) where S' is the next state and r is drawn from the distribution over $\mathcal{R}$ returned by $\mathcal{M}(S, D, q)$.

Definition 3.5 (Interactive Adversary). An *interactive adversary* $\mathcal{A} : \mathcal{R}^* \to (\mathcal{D} \times \mathcal{Q}) \cup \{$✋$\}$ maps the history of responses $(r_1, r_2, \ldots) \in \mathbb{R}^*$, to the next database $D \in \mathcal{D}$ and query $q \in \mathcal{Q}$, or "halt" (✋).

The interaction between $\mathcal{M}$ with initial state $S_0 \in \mathcal{S}$, and an interactive adversary $\mathcal{A}$ is described in Algorithm 5 and results in the probability distribution IT $(\mathcal{M} : \mathcal{A})$ of *transcripts* Π, which is a sequence of responses $(r_1, r_2, \ldots) \in \mathcal{R}^*$.

In order to define what it means for an interactive mechanism to satisfy DP, we need to define the notion of "adjacency" for databases. For now, let us abstractly say that database is a set of records $(x, y) \in \mathcal{X} \times \mathcal{Y}$. The set $\mathcal{X}$ is assumed to be known to the adversary and we will refer to $x \in \mathcal{X}$ as a "privacy unit". Let $\mathcal{Y}$ be an arbitrary set for the sake of our notation here; we instantiate it appropriately as relevant later. For any database D, let D^{-x} denote the database that loosely speaking "removes records in D corresponding to $x \in \mathcal{X}$ or replaces them with a certain generic one". We leave this notion to be abstract for now, and we instantiate the specific notion of adjacency when describing the formal guarantees of ARA and PAA. For any interactive mechanism $\mathcal{M}$ and any $x \in \mathcal{X}$, let $\mathcal{M}^{-x}$ denote the mechanism that replaces the dataset D_t by D_t^{-x} at each step: that is, $(S_t, r_t) \sim \mathcal{M}(S_{t-1}, D_t, q_t)$ is replaced by $(S_t, r_t) \sim \mathcal{M}(S_{t-1}, D_t^{-x}, q_t)$ in Algorithm 5. We define DP for interactive mechanisms as follows.

Definition 3.6 (DP for Interactive Mechanisms). An interactive mechanism $\mathcal{M}$ satisfies (ε, δ)-DP if for all interactive adversaries $\mathcal{A}$ and all $x \in \mathcal{X}$, it holds that IT $(\mathcal{M} : \mathcal{A}) \approx_{\varepsilon,\delta}$ IT $(\mathcal{M}^{-x} : \mathcal{A})$.

Remark 3.7. *A single round version of our definition coincides with the standard definition of DP for the adjacency notion where D and D^{-x} are adjacent. For multiple rounds, our definition is strictly stronger than the standard definition of DP for interactive mechanisms, where the adversary always returns the same database at each step; we refer to such adversaries as "stable".*

4 Analysis of Summary Reports

4.1 DP guarantees from IDP

For any interactive mechanism $\mathcal{M}$ and a sequence of databases and queries $((D_1, q_1), (D_2, q_2), \ldots)$, let $\mathcal{F}_t(\cdot)$ denote the distribution over response r_t as returned by $\mathcal{M}(S_{t-1}, \cdot, q_t)$; note that the sequence $(S_0, S_1, \ldots, S_{t-1})$ of mechanism states is deterministic given the sequence of databases and queries. We say that the sequence $((\varepsilon_1, \delta_1), (\varepsilon_2, \delta_2), \ldots)$ is a *privacy rollout* of the mechanism $\mathcal{M}$ for $x \in \mathcal{X}$ on the sequence $((D_1, q_1), (D_2, q_2), \ldots)$, if $\mathcal{F}_t(D_t) \approx_{\varepsilon_t,\delta_t} \mathcal{F}_t(D_t^{-x})$ holds for all t.

Definition 4.1 (Individual DP). An interactive mechanism $\mathcal{M}$ satisfies $(\varepsilon_*, \delta_*)$-*IDP* if for all $x \in \mathcal{X}$ and all sequences $((D_1, q_1), (D_2, q_2), \ldots)$ of databases and queries, if $((\varepsilon_1, \delta_1), (\varepsilon_2, \delta_2), \ldots)$ is a privacy rollout of $\mathcal{M}$ for x on the said input sequence, then $\sum_t \varepsilon_t \leq \varepsilon_*$ and $\sum_t \delta_t \leq \delta_*$.

Our main technical result is that IDP implies DP. This can be viewed as the approximate-DP variant of [19, Theorem 4.5], which proves a qualitatively similar statement for Rényi DP although our result is for the more general case of interactive adversaries, wherein even the database can change based on previous responses. We note however that our proof technique is quite general, and can be applied in the Rényi DP setting to extend the result of [19] to the case of interactive adversaries as well.

Theorem 4.2. *If an interactive mechanism satisfies $(\varepsilon_*, \delta_*)$-IDP, then it satisfies $(\varepsilon_*, \delta_*)$-DP.*

To prove the above, we apply the tool of DP filters [25] in the setting of IDP [17, 19].

We defer the full proof to Section 4.4, and first describe how this implies the DP guarantees of the ARA and PAA summary reports by appropriately instantiating the notion of a database and adjacency as well as the notion of queries and responses and showing that the interactive mechanism that generates the corresponding summary reports satisfies an IDP guarantee and hence by the above result, also satisfies a DP guarantee.

4.2 Privacy of ARA Summary Reports

To prove the DP properties of ARA, we formalize an end-to-end mechanism $\mathcal{M}_{\text{SR}}$ (Algorithm 6) that simulates the joint behavior of ARA-SR-Client (Algorithm 2) and the aggregation service (Algorithm 1) ultimately generating the summary reports.

Databases and Adjacency. We model a database $D \in \mathcal{D}_{\text{ARA}}$ as consisting of records $(x, y) \in \mathcal{X}_\perp \times \mathcal{Y}$ where $\mathcal{X}_\perp = \mathcal{X} \cup \{x_\perp\}$ is the set $\mathcal{X}$ of all possible "sources" registered across all devices in addition to a "dummy source" that we denote as $x_\perp$ and $\mathcal{Y}$ is the set of all possible "triggers" registered across all devices; note that triggers are in one-to-one correspondence with the report ID r part of the generated aggregatable report (r, k, v), and thus for simplicitly, we interchangeably use $y \in \mathcal{Y}$ to denote the report ID. For any database $D \in \mathcal{D}$ and $x \in \mathcal{X}$, let D^{-x} be the dataset obtained by moving all aggregatable reports associated to x to instead be associated with $x_\perp$, that is, replace (x, y) by $(x_\perp, y)$. We note that adjacent databases in our notion have the same set of x's and y's, and thus, DP is not protecting against knowledge of these, but only the knowledge of which y's are attributed to which x's.

Queries, Responses and Mechanism States. The query set $\mathcal{Q}$ for $\mathcal{M}_{\text{SR}}$ consists of tuples $q = (\varepsilon, \delta, Y, f)$ where $(\varepsilon, \delta) \in \mathbb{R}_{\geq 0} \times [0, 1]$ are privacy parameters for the query, $Y \subseteq \mathcal{Y}$ is a subset of triggers whose corresponding reports need to be aggregated and $f : \mathcal{X} \times \mathcal{Y} \to \mathcal{K} \times \mathbb{N}$ is a function mapping a pair of a source and a trigger to the candidate (key, value) for the aggregatable report generated by them. The response set $\mathcal{R}$ of $\mathcal{M}_{\text{SR}}$ is the set of summary reports, namely $(\mathcal{K} \times \mathbb{Z})^*$. Finally, the state set $\mathcal{S}$ of $\mathcal{M}_{\text{SR}}$ is given by the tuple $(\{(L_x, s_x)\}_{x \in \mathcal{X}}, \{(\varepsilon_y, \delta_y)\}_{y \in \mathcal{Y}}, R)$. Here, L_x (resp., s_x) is the sum of all (resp., number of non-zero) contributions attributed to $x \in \mathcal{X}$, ϵ_y, δ_y are privacy budgets consumed for each report (equivalently trigger) $y \in \mathcal{Y}$, and R is the set of all aggregatable reports generated so far.

Relating $\mathcal{M}_{\text{SR}}$ to ARA Client and Aggregation Service. For $\mathcal{M}_{\text{SR}}$ to simulate the end-to-end generation of summary reports by ARA, we instantiate the database D at each step to be the set of *new* impressions and trigger pairs registered since the last query to $\mathcal{M}_{\text{SR}}$. In phase #1, $\mathcal{M}_{\text{SR}}$ updates the set R in its state to have all the aggregatable reports generated so far, applying the contribution and sparsity bounding similar to ARA-SR-Client (Algorithm 2). In phase #2, $\mathcal{M}_{\text{SR}}$ tracks and enforces that the privacy budget used per report specified in Y is under limits. If not, it aborts. Else it updates these privacy budgets per report and in phase #3, returns the noisy summation per key (with or without key discovery as specified). Phases #2 and #3 simulate the aggregation service (Algorithm 1).

Theorem 4.3. *$\mathcal{M}_{\text{SR}}$ satisfies $(\varepsilon_*, \delta_*)$-IDP with respect to aforementioned set of databases and neighbouring relation for ARA.*

Before proving Theorem 4.3, we note that putting this together with Theorem 4.2, immediately implies the following corollary.

Corollary 4.4. *$\mathcal{M}_{\text{SR}}$ satisfies $(\varepsilon_*, \delta_*)$-DP with respect to aforementioned set of databases and neighbouring relation for ARA.*

Proof of Theorem 4.3. Let $(D_1, (\varepsilon_1, \delta_1, Y_1, f_1)), \ldots,$ be any sequence of databases and queries that is provided to $\mathcal{M}_{\text{SR}}$; assume the state at step t is $(\{(L_{t,x}, s_{t,x})\}_{x \in \mathcal{X}}, \{(\varepsilon_{t,y}, \delta_{t,y})\}_{y \in \mathcal{Y}}, R_t)$. We assume without loss of generality that there is no round where $\mathcal{M}_{\text{SR}}$ returns a response of **abort**. This is because whether $\mathcal{M}_{\text{SR}}$ outputs **abort** or not is only a function of the y values in the database, and is the same for D and D^{-x}.

Consider any $x \in \mathcal{X}$. Denote by $R_{x,t} \subseteq R_t$ the set of all aggregatable reports generated for x. Let $Y_{x,t}$ denote the set of all $y \in Y_t$ such that $(y, k_y, v_y) \in R_{x,t}$ for some (k_y, v_y) and let $Y_x := \bigcup_t Y_{x,t}$. Note that $\sum_{y \in Y_x} v_y \leq \Lambda_1$ and $|Y_x| \leq \Lambda_0$ for all t.

Thus, from the privacy guarantee of the discrete Laplace mechanism (Theorem 3.3) we have that $\mathcal{M}_{\text{SR}}(S_{t-1}, D_t, (\varepsilon_t, \delta_t, Y_t, q_t)) \approx_{\varepsilon_{x,t}, \delta_{x,t}} \mathcal{M}_{\text{SR}}(S_{t-1}, D_t^{-x}, (\varepsilon_t, \delta_t, Y_t, q_t))$ where

$$\varepsilon_{x,t} := \frac{\varepsilon_t}{\Lambda_1} \cdot \sum_{y \in Y_{x,t}} v_y \qquad \text{and} \qquad \delta_{x,t} := \frac{\delta_t}{\Lambda_0} \cdot |Y_{x,t}|,$$

and each step t. Thus, we get that $\mathcal{M}_{\text{SR}}$ satisfies $(\varepsilon_*, \delta_*)$-IDP, since for any $x \in \mathcal{X}$, it holds that

$$\sum_t \varepsilon_{x,t} = \sum_t \left(\frac{\varepsilon_t}{\Lambda_1} \cdot \sum_{y \in Y_{x,t}} v_y \right) \leq \sum_{y \in Y_x} \frac{v_y}{\Lambda_1} \cdot \sum_{t \,:\, Y_{x,t} \ni y} \varepsilon_t \leq \varepsilon_*,$$

Algorithm 6 Interactive mechanism $\mathcal{M}_{\text{SR}} : \mathcal{S} \times \mathcal{D} \times \mathcal{Q} \to \mathcal{S} \times \Delta(\mathcal{R})$.

Params: ▸ Contribution budget Λ_1, Sparsity budget Λ_0,
▸ Global privacy parameters $(\varepsilon_*, \delta_*)$.
State: ▸ $\{(L_x, s_x)\}_{x \in \mathcal{X}}$, $\{(\varepsilon_y, \delta_y)\}_{y \in \mathcal{Y}}$, and $R \subseteq \mathcal{Y} \times \mathcal{K} \times \mathbb{N}$.
Inputs: ▸ Database $D \in \mathcal{D}_{\text{ARA}}$,
▸ Privacy parameters $\varepsilon > 0$ and $\delta \in [0, 1]$,
▸ The list of triggers $Y \subseteq \mathcal{Y}$ whose corresponding reports are to be aggregated, and
▸ The function $f : \mathcal{X} \times \mathcal{Y} \to \mathcal{K} \times \mathbb{N}$

# 1. Contribution Bounding Per $x \in \mathcal{X}$ (cf. ARA-SR-Client)
for $(x, y) \in D$ **do**
  $(k, v) \leftarrow f(x, y)$
  if $L_x + v \leq \Lambda_1$ and $s_x + 1 \leq \Lambda_0$ **then**
    $L_x \leftarrow L_x + v$
    $s_x \leftarrow s_x + 1$
    $R \leftarrow R \cup \{(y, k, v)\}$

# 2. Budget Bounding Per $y \in \mathcal{Y}$ (cf. Aggregation Service)
if $\exists y \in \mathcal{Y}$ such that $\varepsilon_y + \varepsilon > \varepsilon_*$ or $\delta_y + \delta > \delta_*$ **then**
  return $((\{(L_x, s_x)\}_{x \in \mathcal{X}}, \{(\varepsilon_y, \delta_y)\}_{y \in \mathcal{Y}}, R)$, **abort**)
else
  $\varepsilon_y \leftarrow \varepsilon_y + \varepsilon$ and $\delta_y \leftarrow \delta_y + \delta$ for all $y \in Y$

3. Noisy summation (cf. Aggregation Service)
$\tau \leftarrow \begin{cases} \infty & \text{if } \delta = 0, \\ \Lambda_1 \cdot (1 + \log(\Lambda_0/\delta)/\varepsilon) & \text{if } \delta > 0. \end{cases}$
$S \leftarrow \emptyset$
for $k \in \mathcal{K}$ **do**
  $c_k \leftarrow \xi + \sum_{(y, k', v) \in R \,:\, y \in Y,\ k' = k} v$ for $\xi \sim \text{DLap}_\tau(\varepsilon/\Lambda_1)$
  if $\tau = \infty$ or $c_k > \tau$ **then**
    $S \leftarrow S \cup \{(k, c_k)\}$

return $\left((\{(L_x, s_x)\}_{x \in \mathcal{X}}, \{(\varepsilon_y, \delta_y)\}_{y \in \mathcal{Y}}, R), S\right)$

where the last inequality follows because $\sum_{t : Y_{x,t} \ni y} \varepsilon_t \leq \varepsilon_*$ for all $y \in \mathcal{Y}$ and $\sum_{y \in Y_x} v_y \leq \Lambda_1$ for all $x \in \mathcal{X}$. Similarly,

$$\sum_t \delta_{x,t} = \sum_t \left(\frac{\delta_t}{\Lambda_0} \cdot |Y_{x,t}| \right) \leq \sum_{y \in Y_x} \frac{1}{\Lambda_0} \sum_{t \,:\, Y_{x,t} \ni y} \delta_t \leq \delta_*,$$

where the last inequality follows because $\sum_{t : Y_{x,t} \ni y} \delta_t \leq \delta_*$ for all $y \in \mathcal{Y}$ and $|Y_x| \leq \Lambda_0$ for all $x \in \mathcal{X}$. □

4.3 Privacy of PAA Summary Reports

The DP properties of PAA can be proved with the same formalism of the interactive mechanism $\mathcal{M}_{\text{SR}}$ (Algorithm 6), with just a reinterpretation of databases and the adjacency notion.

In the context of PAA summary reports, we model a database $D \in \mathcal{D}_{\text{PAA}}$ as consisting of records $(x, y) \in \mathcal{X} \times \mathcal{Y}_\perp$ where $\mathcal{X}$ is the set of all pairs consisting of (device, t) for $t \in \mathcal{T}$ is the time window and $\mathcal{Y}_\perp$ is the set of all possible states of the shared storage $\mathcal{Y}$ and a dummy shared storage $y_\perp$ (denoting an "empty" shared storage). For any database $D \in \mathcal{D}$ and $x \in \mathcal{X}$, let D^{-x} be the dataset obtained by replacing all shared storage states associated to x by $y_\perp$. The notion of queries, responses and states remain the same as in the case of ARA.

The following theorem has essentially the same proof as Theorem 4.3, so we do not repeat it.

Theorem 4.5. *$\mathcal{M}_{\text{SR}}$ satisfies $(\varepsilon_*, \delta_*)$-IDP with respect to aforementioned set of databases and neighbouring relation for PAA.*

Hence, a corollary similar to Theorem 4.4 holds.

Corollary 4.6. *$\mathcal{M}_{\text{SR}}$ satisfies $(\varepsilon_*, \delta_*)$-DP with respect to aforementioned set of databases and neighbouring relation for ARA.*

Note however, that it might be desirable to have privacy guarantees per device, but the above only provides a privacy guarantee per (device, time window t). One could try to obtain a device-level DP guarantee by using the "group privacy" property of DP.

Fact 4.7 (Group Privacy [45]). *If distributions $P_0, P_1, \ldots, P_k$ are such that $P_i \approx_{\varepsilon,\delta} P_{i+1}$ for all i, then the distributions $P_0 \approx_{\varepsilon',\delta'} P_k$ for $\varepsilon' = k\varepsilon$ and $\delta' = \delta\frac{e^{k\varepsilon}-1}{e^{\varepsilon}-1}$.*

However, it is tricky to apply this property because if the shared storage is disabled or cleared in a certain time window t, it can affect the future states of shared storage and thereby also affect the aggregatable reports generated by events on that device.

One can however formulate a weaker notion of device-level DP guarantee for PAA, by considering a variant of DP with gradual expiration that is considered in [4].

Definition 4.8 (DP for Interactive Mechanisms with Gradual Expiration). An interactive mechanism $\mathcal{M}$ satisfies (ε, δ)-*DP with gradual expiration* if for all interactive adversaries $\mathcal{A}$, all $e \in \mathbb{Z}$, all devices $u \in \mathcal{U}$ and all $t_1, t_2 \in \mathcal{T}$ with $t_1 < t_2$, the distributions $\text{IT}\left(\mathcal{M}^{-(u,>t_1)} : \mathcal{A}\right) \approx_{\varepsilon',\delta'} \text{IT}\left(\mathcal{M}^{-(u,>t_2)} : \mathcal{A}\right)$ where $\varepsilon' = \varepsilon(t_2 - t_1)$, $\delta' = \delta\frac{e^{\varepsilon'}-1}{e^{\varepsilon}-1}$, and $\mathcal{M}^{-(a,>t_1)}$ denotes the mechanism that replaces the dataset D_t by $D_t^{-(u,>t_1)}$ (obtained by replacing all shared storage states associated to (u, t) for $t > t_1$ by $y_\perp$) at each step.

It is immediate to see that if $\mathcal{M}$ satisfies (ε, δ)-DP, then it satisfies (ε, δ)-DP with gradual expiration via Theorem 4.7. Thus, we get that in the context of PAA, the mechanism $\mathcal{M}_{\text{SR}}$ satisfies $(\varepsilon_*, \delta_*)$-DP with gradual expiration.

4.4 Proof of Theorem 4.2

As mentioned before, to prove Theorem 4.2, we apply the tool of DP filters [25] in the setting of IDP [17, 19], that we discuss in Sections 4.4.1 and 4.4.2 below respectively.

4.4.1 Privacy Filters. To discuss privacy filters, consider databases $U \subseteq \mathcal{X}$ that contain only privacy units. We use U^{-x} to denote the dataset $U \setminus \{x\}$.[7]

For $\Theta = \mathbb{R}_{\geq 0} \times [0, 1]$ being the parameter space underlying (ε, δ)-DP, let $\phi : \Theta^* \to \{👍, 👎\}$ be a function, called *filter*, that maps a sequence of parameters to 👍 or 👎. For any $\mathcal{R}$, and $\mathcal{P}(\mathcal{X})$ denoting the powerset of $\mathcal{X}$, we consider the following notion of a *universal interactive mechanism* $\mathcal{M}_\phi : \mathcal{S} \times \mathcal{P}(\mathcal{X}) \times \mathcal{Q}_\mathcal{U} \to \mathcal{S} \times \Delta((\mathcal{R} \cup \{\perp\}))$, parameterized by ϕ, that performs on-the-fly privacy budgeting defined as follows:

- $\mathcal{Q}_\mathcal{U} := \{q : \mathcal{P}(\mathcal{X}) \to \Delta(\mathcal{R})\}$ consists of "universal queries" that can be arbitrary "one-shot" mechanisms; an example of such a mechanism is $q(U) = \sum_{x \in U} v_x + \text{DLap}(a)$, where v_x and a are chosen as part of the query,
- $\mathcal{S} = \Theta^*$ consists of sequences of privacy parameters.

On query q such that q satisfies θ-DP,[8] $\mathcal{M}_\phi$ operates as follows:

$$\mathcal{M}_\phi((\theta_1, \ldots, \theta_t), D, q) := \begin{cases} ((\theta_1, \ldots, \theta_t, \theta), q(D)) & \text{if } \phi(\theta_1, \ldots, \theta_t, \theta) = 👍 \\ ((\theta_1, \ldots, \theta_t, \mathbf{0}), \perp) & \text{if } \phi(\theta_1, \ldots, \theta_t, \theta) = 👎 \end{cases}$$

That is, if ϕ applied on the current state (sequence of θ_i's so far) concatenated with the current θ returns 👍, then θ is concatenated to the current state, and the one-shot mechanism q is applied on U. But if not, then $\mathbf{0}$ is concatenated to the state and $\perp$ is returned.

The interactive mechanism $\mathcal{M}_\phi$ interacts with an adversary $\mathcal{A}$ in the same way as in Algorithm 5 to produce $\text{IT}\left(\mathcal{M}_\phi : \mathcal{A}\right)$.

For $(\varepsilon, \delta) \in \Theta$, we define the filter $\phi_{\varepsilon,\delta} : \Theta^* \to \{👍, 👎\}$ as:

$$\phi_{\varepsilon,\delta}((\varepsilon_1, \delta_1), \ldots, (\varepsilon_n, \delta_n)) := \begin{cases} 👍 & \text{if } \sum_{i=1}^{n} \varepsilon_i \leq \varepsilon \;\&\; \sum_{i=1}^{n} \delta_i \leq \delta \\ 👎 & \text{otherwise.} \end{cases}$$

It is known that $\mathcal{M}_\phi$ with the above filter satisfies DP guarantees.

Lemma 4.9 ([25]). *For all $(\varepsilon, \delta) \in \Theta$, $\mathcal{M}_{\phi_{\varepsilon,\delta}}$ satisfies (ε, δ)-DP against stable adversaries.*[9]

4.4.2 Individual Differential Privacy. We now consider a universal interactive mechanism with a privacy filter but using the notion of IDP [17, 19].

Definition 4.10. For $p : \mathcal{X} \to \Theta$, a mechanism $\mathcal{F} : \mathcal{P}(\mathcal{X}) \to \Delta(\mathcal{R})$ satisfies p-*IDP* if for all $U \in \mathcal{P}(\mathcal{X})$ and all $x \in \mathcal{X}$, it holds that $\mathcal{F}(U) \approx_{p(x)} \mathcal{F}(U^{-x})$.

As before, for $\phi : \Theta^* \to \{👍, 👎\}$, we consider a universal interactive mechanism $\mathcal{M}_\phi^{\text{ind}} : \mathcal{S} \times \mathcal{P}(\mathcal{X}) \times \mathcal{Q}_\mathcal{U} \to \mathcal{S} \times \Delta(\mathcal{R})$, that performs on-the-fly privacy budgeting, where $\mathcal{S} = (\Theta^\mathcal{X})^*$ consists of sequences of privacy parameters, one for each unit $x \in \mathcal{X}$. On query q such that q satisfies p-IDP, $\mathcal{M}_\phi^{\text{ind}}$ operates as follows. On current state $(p_1, \ldots, p_t)$, it constructs a "masked database" $\overline{U} := U \cap \{x : \phi(p_1(x), \ldots, p_t(x), p(x)) = 👍\}$ and the consumed individual privacy $p_{t+1} : \mathcal{X} \to \Theta$ as:

$$p_{t+1}(x) := \begin{cases} p(x) & \text{if } \phi(p_1(x), \ldots, p_t(x), p(x)) = 👍 \\ \mathbf{0} & \text{if } \phi(p_1(x), \ldots, p_t(x), p(x)) = 👎. \end{cases}$$

And the mechanism returns,

$$\mathcal{M}_\phi((p_1, \ldots, p_t), U, q) := ((p_1, \ldots, p_{t+1}), q(\overline{U})).$$

Lemma 4.11. *For all $(\varepsilon, \delta) \in \Theta$, $\mathcal{M}_{\phi_{\varepsilon,\delta}}^{\text{ind}}$ satisfies (ε, δ)-DP against stable adversaries.*

[7] The notation U^{-x} is useful only when $x \in U$. Otherwise $U^{-x} = U$, which can lead to vacuous statements. Since these are also correct we do not enforce that $x \in U$.

[8] q may satisfy θ-DP for a number of different $\theta \in \Theta$, and while any such θ could be used, it is important to specify a particular choice of θ to make $\mathcal{M}_\phi$ well-defined. The definition of $\mathcal{Q}$ could be modified to additionally provide the specific θ along with q, but we avoid doing so for simplicity.

[9] [25] also provides an improved "advanced composition"-like filter for (ε, δ)-DP, that was subsequently improved in [46]. However, since we only use this version, we do not state the advanced version.

Proof. Fix some database $U \subseteq \mathcal{X}$. Consider a stable adversary $\mathcal{A} : \mathcal{R}^* \to (\mathcal{P}(\mathcal{X}) \times \mathcal{Q}_\mathcal{U}) \cup \{\text{✋}\}$ that returns the same database U on each step. We want to show that for any $x \in \mathcal{X}$ it holds that $\mathrm{IT}\left(\mathcal{M}^{\mathrm{ind}}_{\phi_\theta} : \mathcal{A}\right) \approx_{\varepsilon,\delta} \mathrm{IT}\left(\left(\mathcal{M}^{\mathrm{ind}}_{\phi_\theta}\right)^{-x} : \mathcal{A}\right)$. This follows by "zooming in on unit x". Namely, consider $\mathcal{X}' = \{x\}$, and let $\mathcal{Q}'_\mathcal{U} := \{q : \mathcal{X}' \to \Delta(\mathcal{R})\}$ be the set of universal queries on $\mathcal{X}'$. Construct an adversary $\mathcal{A}' : \mathcal{R}^* \to (\mathcal{P}(\mathcal{X}') \times \mathcal{Q}'_\mathcal{U}) \cup \{\text{✋}\}$ as follows: $\mathcal{A}'(\Pi)$ first computes $(D, q) \leftarrow \mathcal{A}(\Pi)$, and return (D', q') where $D' = D \cap \mathcal{X}'$ and $q'(U') := q(\tilde{U}^{-x} \cup U')$ for $U' \subseteq \mathcal{X}'$ and $\tilde{U}^{-x} \subseteq U^{-x}$ that have the privacy budget to participate. Thus, when $U \ni x$, we have $\mathrm{IT}\left(\mathcal{M}^{\mathrm{ind}}_{\phi_\theta} : \mathcal{A}\right) \equiv \mathrm{IT}\left(\mathcal{M}_{\phi_\theta} : \mathcal{A}'\right) \approx_{\varepsilon,\delta} \mathrm{IT}\left(\left(\mathcal{M}_{\phi_\theta}\right)^{-x} : \mathcal{A}'\right) \equiv \mathrm{IT}\left(\left(\mathcal{M}^{\mathrm{ind}}_{\phi_\theta}\right)^{-x} : \mathcal{A}\right)$, by Theorem 4.9. □

4.4.3 *Putting it Together: Proof of Theorem 4.2.*

Proof of Theorem 4.2. Let $\mathcal{M}$ be an interactive mechanism satisfying $(\varepsilon_*, \delta_*)$-IDP guarantees. To prove the statement it suffices to show that for any adversary $\mathcal{A}$ that interacts with $\mathcal{M}$, there is an adversary $\mathcal{A}'$ for $\mathcal{M}_{\phi_{\varepsilon_*,\delta_*}}$ such that

$$\mathrm{IT}\left(\mathcal{M}^{\mathrm{ind}}_{\phi_{\varepsilon_*,\delta_*}} : \mathcal{A}'\right) \equiv \mathrm{IT}\left(M : \mathcal{A}\right), \text{ and}$$
$$\mathrm{IT}\left(\left(\mathcal{M}^{\mathrm{ind}}_{\phi_{\varepsilon_*,\delta_*}}\right)^{-x} : \mathcal{A}'\right) \equiv \mathrm{IT}\left(M^{-x} : \mathcal{A}\right).$$

Conditioned on a sequence $(r_1, \ldots, r_t)$ of responses, let the corresponding states of $\mathcal{M}$ when interacting with $\mathcal{A}$ be $S_0, S_1, \ldots, S_t$; note S_i is deterministic given database D_i and query q_i, which are in turn deterministic given $r_1, \ldots, r_{i-1}$.

We define $\mathcal{A}'$ that on input $(r_1, \ldots, r_t)$, computes $(D_{t+1}, q_{t+1}) \leftarrow \mathcal{A}(r_1, \ldots, r_t)$ and returns (D_{t+1}, q'_{t+1}) where $q'_{t+1}(\tilde{D})$ is distribution over responses r returned by $\mathcal{M}(S_t, \tilde{D}, q_{t+1})$. It is easy to see that $\mathcal{A}'$ satisfy the condition since $\mathcal{M}_{\phi_{\varepsilon_*,\delta_*}}$ will always output $q'(D_{t+1})$ given $\mathcal{M}$ is $(\varepsilon_*, \delta_*)$-IDP, that is, the filter never masks any element of the database; hence from Theorem 4.11, we conclude that $\mathrm{IT}\left(M : \mathcal{A}\right) \approx_{\varepsilon_*,\delta_*} \mathrm{IT}\left(M^{-x} : \mathcal{A}\right)$. □

5 Analysis of Event-Level Reports

The database and the privacy unit in event-level reports is the same as described in Section 4.2.

The event-level API is based on what we refer to as *interactive randomized response (IRR)*; see Algorithm 8 for details. In this setting, we have finite set $O_1, \ldots, O_i, \ldots$ of outputs at each time step and a finite set $O \subseteq O_1 \times \cdots \times O_i \times \cdots$ of valid combinations of these outputs. The algorithm decides (randomly) at the very beginning of the run whether it is going to report truthfully (i.e., $s^* = \perp$) or whether it is going to report some other output (i.e., $s^* \in O$). In the latter case, the algorithm simply reports based on s^* regardless of the input. On the other hand, in the former case, the algorithm evaluates the query it receives and outputs truthfully in each step.

It is possible to see that event-level reports can be captured by the interactive mechanism $\mathcal{M}_{\mathrm{ER}}$. Indeed, let us choose $O^{(x)}$ so that $O^{(x)}_i$ is the set of all combinations of possible reports that could be sent at ith second. Then q_x on ith iteration is the function that creates the reports that would be sent on ith second. (Note that we allow q_x to depend on all the events, not just the one created during this second, this makes our privacy guarantee stronger than actually necessary.)

Algorithm 7 Interactive Transcript $\mathrm{IT}(\mathcal{M} : \mathbb{A})$.

Inputs: ▷ Interactive mechanism $\mathcal{M}$ with initial state S_0,
▷ A distribution of interactive adversaries $\mathbb{A}$.
Sample $\mathcal{A} \sim \mathbb{A}$.
return $\mathrm{IT}(\mathcal{M} : \mathcal{A})$.

Algorithm 8 InteractiveRandomizedResponse $\mathcal{I}_{\varepsilon,O}$.

Params: ▷ Privacy parameter $\varepsilon \in \mathbb{R}$,
▷ Output set $O \subseteq O_1 \times \cdots \times O_i \times \cdots$.
Inputs: ▷ State S encoding $i \in \mathbb{N}$ and $s^* \in O \cup \{\perp\}$,
▷ Set of events $Y \subseteq \mathcal{Y}$ of events,
▷ Query $q : \mathcal{P}(\mathcal{Y}) \to O_i$.
if $i = 1$ **then**
Set s^* to $\perp$ with probability $\frac{e^\varepsilon - 1}{e^\varepsilon + |O| - 1}$, otherwise set it to a random sample from O
if $s^* = \perp$ **then**
return $(s^*, q(Y))$
else
return (s^*, s^*_i)

The main result of this section is that this is indeed ε-DP.

Theorem 5.1. *$\mathcal{M}_{\mathrm{ER}}$ satisfies ε-DP.*

To prove this theorem, one may notice that while our definition of DP for interactive mechanisms assumed that the adversary is deterministic it is not strictly necessary. This follows from a simple *joint-convexity* property of DP.

Fact 5.2 (Joint Convexity (see e.g. Lemma B.1 in [9])). *Given two families of distributions $\{P_i\}_i$ and $\{Q_i\}_i$, if $P_i \approx_{\varepsilon,\delta} Q_i$ for all i, then for all mixture distributions $P = \sum_i \alpha_i P_i$ and $Q = \sum_i \alpha_i Q_i$, it holds that $P \approx_{\varepsilon,\delta} Q$.*

We extend the notion of a transcript to support distributions of adversaries (Algorithm 7). For any distribution $\mathbb{A}$ over interactive adversaries, by applying the above fact for $P_\mathcal{A} = \mathrm{IT}(\mathcal{M} : \mathcal{A})$ and $Q_\mathcal{A} = \mathrm{IT}(\mathcal{M}^{-x} : \mathcal{A})$ for each $\mathcal{A}$ in the support of $\mathbb{A}$, we get the following corollary.

Corollary 5.3. *For all mechanisms $\mathcal{M}$, if for all $x \in \mathcal{X}$ and all interactive adversaries it holds that $\mathrm{IT}(\mathcal{M} : \mathcal{A}) \approx_{\varepsilon,\delta} \mathrm{IT}(\mathcal{M}^{-x} : \mathcal{A})$, then $\mathrm{IT}(\mathcal{M} : \mathbb{A}) \approx_{\varepsilon,\delta} \mathrm{IT}(\mathcal{M}^{-x} : \mathbb{A})$ holds for all distributions $\mathbb{A}$ over interactive adversaries.*

Proof of Theorem 5.1. The proof consists of two parts: first, we show that $\mathcal{M}_{\mathrm{ER}}$ is private when $\mathcal{X}$ has only one element x; next, we prove that general privacy guarantee follows from this.

Assume that $\mathcal{X} = \{x\}$. Note that state of the mechanism does not change over the course of execution. Let us denote the random variable for this state as s^*. It is easy to see that for any $\tilde{o} \in O$,

$$\Pr[\mathrm{IT}(\mathcal{M}_{\mathrm{ER}} : \mathcal{A}) = \tilde{o} \mid s^* = \tilde{o}] = \Pr[\mathrm{IT}\left(\mathcal{M}^{-x}_{\mathrm{ER}} : \mathcal{A}\right) = \tilde{o} \mid s^* = \tilde{o}] = 1.$$

Algorithm 9 Interactive mechanism $\mathcal{M}_{\mathrm{ER}} : \mathcal{S}\times\mathcal{D}\times\mathcal{Q} \to \mathcal{S}\times\Delta(\mathcal{R})$.

Params: ▷ Privacy parameter $\varepsilon > 0$,
▷ Output sets $\{O^{(x)}\}_{x\in\mathcal{X}}$.
Inputs: ▷ State S encoding $i \in \mathbb{N}$ and $\{s_x^* \in O^{(x)} \cup \{\perp\}\}_{x\in\mathcal{X}}$,
▷ Database $D \in \mathcal{D}$,
▷ Queries $\{q_x : \mathcal{Y}^* \to O_i^{(x)}\}_{x\in\mathcal{X}}$.
for $x \in \mathcal{X}$ **do**
$(s'_x, r_x) \leftarrow \mathcal{I}_{\varepsilon,O^{(x)}}(i, s_x^*, N_D(x), q_x)$
return $((i, \{s'_x\}_{x\in\mathcal{X}}), \{r_x\}_{x\in\mathcal{X}})$

Therefore,

$$\Pr[\mathrm{IT}(\mathcal{M}_{\mathrm{ER}} : \mathcal{A}) = \tilde{o}] \geq \Pr[s^* = \tilde{o}] = \frac{1}{e^\varepsilon + |O^{(x)}| - 1} \text{ and}$$

$$\Pr[\mathrm{IT}(\mathcal{M}_{\mathrm{ER}}^{-x} : \mathcal{A}) = \tilde{o}] \geq \Pr[s^* = \tilde{o}] = \frac{1}{e^\varepsilon + |O^{(x)}| - 1}.$$

Furthermore, for any $\tilde{o} \in O$,

$$\begin{aligned}
&\Pr[\mathrm{IT}(\mathcal{M}_{\mathrm{ER}} : \mathcal{A}) = \tilde{o}] \\
&= \Pr[\mathrm{IT}(\mathcal{M}_{\mathrm{ER}} : \mathcal{A}) = \tilde{o} \wedge s^* = \perp] + \Pr[\mathrm{IT}(\mathcal{M}_{\mathrm{ER}} : \mathcal{A}) = \tilde{o} \wedge s^* \neq \perp] \\
&\leq \Pr[s^* = \perp] + \Pr[s^* = \tilde{o}] \\
&= \frac{e^\varepsilon - 1}{e^\varepsilon + |O^{(x)}| - 1} + \frac{1}{e^\varepsilon + |O^{(x)}| - 1} = \frac{e^\varepsilon}{e^\varepsilon + |O^{(x)}| - 1},
\end{aligned}$$

which implies that $\mathrm{IT}(\mathcal{M}_{\mathrm{ER}} : \mathcal{A}) \approx_{\varepsilon,0} \mathrm{IT}(\mathcal{M}_{\mathrm{ER}}^{-x} : \mathcal{A})$.

First, we denote the mechanism $\mathcal{M}_{\mathrm{ER}}$ operating on a dataset $\mathcal{X} = \{x\}$ as $\mathcal{M}_{\mathrm{ER}}^{(x)}$. Let us now assume that $\mathcal{X} > 1$. We claim that for each $x \in \mathcal{X}$, any adversary $\mathcal{A}$, and any transcript Π, there is a distribution $\mathbb{A}'$ over adversaries for $\mathcal{M}_{\mathrm{ER}}^{(x)}$ and a transcript Π' for $\mathcal{M}_{\mathrm{ER}}^{(x)}$ such that

$$\Pr[\mathrm{IT}(\mathcal{M}_{\mathrm{ER}} : \mathcal{A}) = \Pi] = \Pr[\mathrm{IT}\left(\mathcal{M}_{\mathrm{ER}}^{(x)} : \mathbb{A}'\right) = \Pi'],$$

$$\Pr[\mathrm{IT}(\mathcal{M}_{\mathrm{ER}}^{-x} : \mathcal{A}) = \Pi] = \Pr[\mathrm{IT}\left(\left(\mathcal{M}_{\mathrm{ER}}^{(x)}\right)^{-x} : \mathbb{A}'\right) = \Pi'].$$

Note that this implies that $\mathcal{M}_{\mathrm{ER}}$ is ε-DP.

Let us now construct $\mathbb{A}'$. First, we define an adversary $\mathcal{A}'_{\{s^*_{x'}\}_{x'\neq x}}$ that runs $\mathcal{A}$ using actual response for x and simulated responses for $x' \neq x$ using $s^*_{x'}$ (note that if the state is fixed, the mechanism is deterministic). It is clear that a distribution $\mathbb{A}'$ that samples first $\{s^*_{x'}\}_{x'\neq x}$ and returns the adversary $\mathcal{A}'_{\{s^*_{x'}\}_{x'\neq x}}$ satisfies the desired condition. □

6 Conclusion

In this work, we modeled the summary reports in the Attribution Reporting API (ARA) and the Private Aggregation API (PAA), as well as the event-level reports in ARA. We established formal DP guarantees for these mechanisms, even against the stringent notion of interactive adversaries that can influence the database in subsequent rounds based on responses in previous rounds.

Acknowledgments

We thank the anonymous reviewers, for their feedback, which significantly improved the clarity of this paper. We thank Roxana Geambasu, Pierre Tholoniat for discussions about [42]. We are also grateful to Jolyn Yao and Christina Ilvento for their invaluable contributions, without which this paper might not have been possible.

References

[1] Hidayet Aksu, Badih Ghazi, Pritish Kamath, Ravi Kumar, Pasin Manurangsi, Adam Sealfon, and Avinash V. Varadarajan. 2024. Summary Reports Optimization in the Privacy Sandbox Attribution Reporting API. *PoPETS* 2024, 4 (2024), 605–621.

[2] Hidayet Aksu and Charlie Harrison. 2023. *Summary reports with key discovery.* Google. https://github.com/WICG/attribution-reporting-api/blob/main/aggregate_key_discovery.md

[3] Hidayet Aksu, Alexander Knop, and Pasin Manurangsi. 2024. *Reach Implementation Best Practices in the Privacy Sandbox Shared Storage + Private Aggregation APIs.* Google. https://github.com/patcg-individual-drafts/private-aggregation-api/blob/main/reach_whitepaper.md

[4] Joel Daniel Andersson, Monika Henzinger, Rasmus Pagh, Teresa Anna Steiner, and Jalaj Upadhyay. 2024. Continual Counting with Gradual Privacy Expiration. In *NeurIPS*.

[5] Apple. 2024. *SKAdNetwork.* Apple. https://developer.apple.com/documentation/storekit/skadnetwork/

[6] Mark Bun and Thomas Steinke. 2016. Concentrated Differential Privacy: Simplifications, Extensions, and Lower Bounds. In *Theory of Cryptography (TCC) (Lecture Notes in Computer Science, Vol. 9985)*, Martin Hirt and Adam D. Smith (Eds.). 635–658. https://doi.org/10.1007/978-3-662-53641-4_24

[7] United States Census Bureau. 2023. *Shared Storage overview.* United States Census Bureau. https://www.census.gov/programs-surveys/decennial-census/decade/2020/planning-management/process/disclosure-avoidance.html

[8] Lynn Chua, Qiliang Cui, Badih Ghazi, Charlie Harrison, Pritish Kamath, Walid Krichene, Ravi Kumar, Pasin Manurangsi, Nicolas Mayoraz, Hema Venkata Krishna Giri Narra, Steffen Rendle, Amer Sinha, Avinash V. Varadarajan, and Chiyuan Zhang. 2024. Training Differentially Private Ad Prediction Models With Semi-Sensitive Features. In *AdKDD*.

[9] Lynn Chua, Badih Ghazi, Pritish Kamath, Ravi Kumar, Pasin Manurangsi, Amer Sinha, and Chiyuan Zhang. 2024. How Private are DP-SGD Implementations?. In *International Conference on Machine Learning, ICML*. OpenReview.net. https://openreview.net/forum?id=xWI0MKwJSS

[10] Luke Crouch and Maxx Crawford. 2022. *Over a decade of anti-tracking work at Mozilla.* Mozilla. https://blog.mozilla.org/en/privacy-security/mozilla-anti-tracking-milestones-timeline/

[11] Matthew Dawson, Badih Ghazi, Pritish Kamath, Kapil Kumar, Ravi Kumar, Bo Luan, Pasin Manurangsi, Nishanth Mundru, Harikesh Nair, Adam Sealfon, and Shengyu Zhu. 2023. Optimizing Hierarchical Queries for the Attribution Reporting API. In *AdKDD*.

[12] John Delaney, Badih Ghazi, Charlie Harrison, Christina Ilvento, Ravi Kumar, Pasin Manurangsi, Martin Pál, Karthik Prabhakar, and Mariana Raykova. 2024. Differentially Private Ad Conversion Measurement. *PoPETS* 2024, 2 (2024), 124–140.

[13] Carson Denison, Badih Ghazi, Pritish Kamath, Ravi Kumar, Pasin Manurangsi, Krishna Giri Narra, Amer Sinha, Avinash V. Varadarajan, and Chiyuan Zhang. 2023. Private Ad Modeling with DP-SGD. In *AdKDD*.

[14] Damien Desfontaines. 2025. *A list of real-world uses of differential privacy.* https://desfontain.es/blog/real-world-differential-privacy.html

[15] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. 2006. Our Data, Ourselves: Privacy Via Distributed Noise Generation. In *EUROCRYPT*. 486–503.

[16] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. 2016. Calibrating Noise to Sensitivity in Private Data Analysis. *J. Priv. Confidentiality* 7, 3 (2016), 17–51.

[17] Hamid Ebadi, David Sands, and Gerardo Schneider. 2015. Differential Privacy: Now it's Getting Personal. In *POPL*. 69–81.

[18] Renan Feldman. 2024. *Public Cloud TEE Requirements.* Google. https://github.com/privacysandbox/protected-auction-services-docs/blob/main/public_cloud_tees.md

[19] Vitaly Feldman and Tijana Zrnic. 2021. Individual Privacy Accounting via a Rényi Filter. In *NeurIPS*. 28080–28091.

[20] Arpita Ghosh, Tim Roughgarden, and Mukund Sundararajan. 2012. Universally Utility-maximizing Privacy Mechanisms. *SIAM J. Comput.* 41, 6 (2012), 1673–1693.

[21] Google. 2022. *See how your community moved differently due to COVID-19.* https://www.google.com/covid19/mobility/

[22] Shlomi Hod and Ran Canetti. 2025. Differentially Private Release of Israel's National Registry of Live Births. In *S & P*. 101–101.

[23] Joseph J. Pfeiffer III, Denis Charles, Davis Gilton, Young Hun Jung, Mehul Parsana, and Erik Anderson. 2021. Masked LARk: Masked Learning, Aggregation and Reporting worKflow. arXiv:2110.14794

[24] Privacy Sandbox. 2024. *Private Aggregation API fundamentals.* Google. https://developers.google.com/privacy-sandbox/private-advertising/private-aggregation/fundamentals#contribution_budget

[25] Ryan M. Rogers, Salil P. Vadhan, Aaron Roth, and Jonathan R. Ullman. 2016. Privacy Odometers and Filters: Pay-as-you-Go Composition. In *NIPS*. 1921–1929.

[26] Privacy Sandbox. 2021. *Attribution Reporting for Web overview: Event-level reports.* Google. https://developers.google.com/privacy-sandbox/private-advertising/attribution-reporting#event-level_reports

[27] Privacy Sandbox. 2021. *Attribution Reporting for Web overview: Summary reports.* Google. https://developers.google.com/privacy-sandbox/private-advertising/attribution-reporting#summary_reports

[28] Privacy Sandbox. 2021. *Attribution Reporting: generating summary reports.* Google. https://developers.google.com/privacy-sandbox/private-advertising/attribution-reporting/summary-reports

[29] Privacy Sandbox. 2022. *Attribution Reporting for Web overview.* Google. https://developers.google.com/privacy-sandbox/relevance/attribution-reporting

[30] Privacy Sandbox. 2022. *Differential Privacy.* Google. https://github.com/WICG/attribution-reporting-api/blob/main/AGGREGATE.md#differential-privacy

[31] Privacy Sandbox. 2022. *Maximize ad relevance without third-party cookies.* Google. https://privacysandbox.com/news/maximize-ad-relevance-after-third-party-cookies/

[32] Privacy Sandbox. 2022. *Private Aggregation API overview.* https://developers.google.com/privacy-sandbox/relevance/private-aggregation

[33] Privacy Sandbox. 2024. *Attribution Reporting with event-level reports.* Google. https://github.com/WICG/attribution-reporting-api/blob/main/EVENT.md

[34] Privacy Sandbox. 2024. *Intelligent Tracking Prevention 2.3.* Google. https://github.com/privacysandbox/aggregation-service/blob/main/docs/batching-strategies.md/#aggregatable-report-accounting

[35] Privacy Sandbox. 2024. *A list of real-world uses of differential privacy.* https://desfontain.es/blog/real-world-differential-privacy.html

[36] Privacy Sandbox. 2024. *Shared Storage overview.* Google. https://developers.google.com/privacy-sandbox/private-advertising/shared-storage

[37] Privacy Sandbox. 2024. *Unique reach measurement.* Google. https://developers.google.com/privacy-sandbox/private-advertising/private-aggregation/unique-reach

[38] Privacy Sandbox. 2024. *Update on the plan for phase-out of third-party cookies on Chrome.* Google. https://privacysandbox.com/news/update-on-the-plan-for-phase-out-of-third-party-cookies-on-chrome/

[39] Privacy Sandbox. 2025. *Aggregation Service for the Attribution Reporting API.* Google. https://github.com/WICG/attribution-reporting-api/blob/main/AGGREGATION_SERVICE_TEE.md#initial-experiment-plans

[40] Chrome Platform Status. 2024. *Percentage of page loads over time for Conversion-APIAll.* Google. https://chromestatus.com/metrics/feature/timeline/popularity/3365

[41] Chrome Platform Status. 2024. *Percentage of page loads over time for PrivateAggregationApiAll.* Google. https://chromestatus.com/metrics/feature/timeline/popularity/4333

[42] Pierre Tholoniat, Kelly Kostopoulou, Peter McNeely, Prabhpreet Singh Sodhi, Anirudh Varanasi, Benjamin Case, Asaf Cidon, Roxana Geambasu, and Mathias Lécuyer. 2024. Cookie Monster: Efficient On-Device Budgeting for Differentially-Private Ad-Measurement Systems. In *SOSP*. 693–708.

[43] Martin Thomson. 2022. *Privacy Preserving Attribution for Advertising.* Mozilla. https://blog.mozilla.org/en/mozilla/privacy-preserving-attribution-for-advertising/

[44] Matilde Tullii, Solenne Gaucher, Hugo Richard, Eustache Diemert, Vianney Perchet, Alain Rakotomamonjy, Clément Calauzènes, and Maxime Vono. 2024. Position Paper: Open Research Challenges for Private Advertising Systems under Local Differential Privacy. In *WISE*. 107–122.

[45] Salil P. Vadhan. 2017. The Complexity of Differential Privacy. In *Tutorials on the Foundations of Cryptography*, Yehuda Lindell (Ed.). Springer, 347–450.

[46] Justin Whitehouse, Aaditya Ramdas, Ryan Rogers, and Steven Wu. 2023. Fully-Adaptive Composition in Differential Privacy. In *ICML*. 36990–37007.

[47] Wikipedia. 2024. *Authenticated encryption with associated data.* Wikipedia. https://en.wikipedia.org/wiki/Authenticated_encryption#Authenticated_encryption_with_associated_data

[48] John Wilander. 2019. *Intelligent Tracking Prevention 2.3.* Apple. https://webkit.org/blog/9521/intelligent-tracking-prevention-2-3/

[49] John Wilander. 2021. *Introducing Private Click Measurement, PCM.* Apple. https://webkit.org/blog/11529/introducing-private-click-measurement-pcm/

[50] Albert Wu. 2024. *Support for Requerying in Aggregation Service: Feedback Requested.* Privacy Sandbox. https://github.com/privacysandbox/aggregation-service/issues/71

[51] Yingtai Xiao, Jian Du, Shikun Zhang, Qiang Yan, Danfeng Zhang, and Daniel Kifer. 2024. Click Without Compromise: Online Advertising Measurement via Per User Differential Privacy. arXiv:2406.02463

A Analysis of (Truncated) Discrete Laplace Mechanism

We provide a proof of Theorem 3.3 for completeness. The first component of this proof is the following tail-bound for discrete Laplace distributions.

Lemma A.1. *Let τ be an integer such that $\tau \geq \log(1/\delta)/a + \Delta$. Then $\Pr_{X\sim \mathsf{DLap}_\tau(a)}[X > \tau - \Delta] \leq \delta$.*

Proof.

$$\begin{aligned}\Pr_{X\sim \mathsf{DLap}_\tau(a)}[X > \tau] &= \frac{\sum_{x=\tau-\Delta+1}^{\tau} e^{-ax}}{\sum_{x=-\tau}^{\tau} e^{-a|x|}} \\ &\leq \frac{\sum_{x=\tau-\Delta+1}^{\tau} e^{-ax}}{\sum_{x=0}^{\tau} e^{-ax}} \\ &\leq \frac{\sum_{x=\tau-\Delta+1}^{\infty} e^{-ax}}{\sum_{x=0}^{\infty} e^{-ax}} \\ &\leq e^{-a(\tau-\Delta)} \leq \delta.\end{aligned}$$

where the last inequality uses that $\tau \geq \log(1/\delta)/a + \Delta$. □

Proof of Theorem 3.3. First, we observe that by shifting the vectors u and v, we can assume without loss of generality that $u = 0$.

First, we consider the case of (untruncated) discrete Laplace noise. Let P and Q be the distributions of $u + \zeta$ and $v + \zeta$ respectively for $\zeta \sim \mathsf{DLap}(a)^{\otimes d}$. In other words, for any $w \in \mathbb{Z}^d$, it holds that

$$P(w) = \frac{1}{Z}e^{-a\|u-w\|_1} \qquad \text{and} \qquad Q(w) = \frac{1}{Z}e^{-a\|v-w\|_1}$$

where $Z = (\frac{e^a+1}{e^a-1})^d$. Thus, we have

$$\frac{P(w)}{Q(w)} = e^{-a(\|u-w\|_1 - \|v-w\|_1)}$$

and it is thus easy to see that

$$e^{-a\Delta} \leq e^{-a\|u-v\|_1} \leq \frac{P(w)}{Q(w)} \leq e^{a\|u-v\|_1} \leq e^{a\Delta}$$

and thus, $P \approx_{\varepsilon,0} Q$ when $a = \varepsilon/\Delta$.

Next, moving to the case of truncated discrete Laplace noise, let P and Q be the distributions of $u + \zeta$ and $v + \zeta$ respectively for $\zeta \sim \mathsf{DLap}_\tau(a)^{\otimes d}$. In particular, we have

$$P(w) = \begin{cases} \frac{1}{Z}e^{-a\|u-w\|_1} & \text{if } \|u-w\|_\infty \leq \tau \\ 0 & \text{if } \|u-w\|_\infty > \tau \end{cases}$$

and similarly for Q, where $Z = \sum_{x=-\tau}^{\tau} e^{-a|x|}$. Let $S := \{w : \|u - w\|_\infty \leq \tau \text{ and } \|v - w\|_\infty \leq \tau\}$. Similar to the case of untruncated case above, it follows that for all $w \in S$, it holds that $e^{-a\Delta} \leq P(w)/Q(w) \leq e^{a\Delta}$. Thus, for $a = \varepsilon/\Delta$, it holds for all $E \subseteq \mathbb{Z}^d$ that

$$\begin{aligned}P(E) &= P(E \cap S) + P(E \setminus S) \\ &\leq e^{\varepsilon}Q(E \cap S) + P(E \setminus S) \\ &\leq e^{\varepsilon}Q(E) + P(\mathbb{Z}^d \setminus S)\end{aligned}$$

Thus, $P \approx_{\varepsilon,\delta} Q$ where $\delta := P(\mathbb{Z}^d \setminus S)$. To complete the proof, we need to show that when $\tau \geq \Delta(1 + \log(s/\delta)/\varepsilon)$, it holds that $P(\mathbb{Z}^d \setminus S) \leq \delta$; recall that u and v differ on s coordinates. We have

$$\begin{aligned}P(\mathbb{Z}^d \setminus S) &= 1 - P(S) \\ &= 1 - \prod_{i=1}^{d} \Pr_{w_i\sim u_i+\mathsf{DLap}_\tau(a)}[|w_i - v_i| \leq \tau] \\ &\leq s \cdot \Pr_{X\sim \mathsf{DLap}_\tau(a)}[X > \tau - \Delta] \qquad (1)\end{aligned}$$

where, we use that when $u_i = v_i$, we have

$$\Pr_{w_i\sim u_i+\mathsf{DLap}_\tau(a)}[|w_i - v_i| \leq \tau] = 1$$

and when $u_i \neq v_i$, we have

$$\begin{aligned}\Pr_{w_i\sim u_i+\mathsf{DLap}_\tau(a)}[|w_i - v_i| \leq \tau] &= \Pr_{X\sim \mathsf{DLap}_\tau(a)}[|X + u_i - v_i| \leq \tau] \\ &\geq 1 - \Pr_{X\sim \mathsf{DLap}_\tau(a)}[X > \tau - \Delta].\end{aligned}$$

Note that Theorem A.1 implies that

$$\Pr_{X\sim \mathsf{DLap}_\tau(a)}[X > \tau - \Delta] \leq \delta/s.$$

since $\tau \geq \Delta(1 + \log(s/\delta)/\varepsilon)$. Combining with Equation (1), we get that $P(\mathbb{Z}^d \setminus S) \leq \delta$, thereby completing the proof. □

B Extension to other notions of DP

While we primarily studied (ε, δ)-DP notion in this paper, our definitions and techniques readily extend to any other notion of DP that admits privacy filters. In particular, we could consider the following notion of Approximate zero Concentrated DP.

Definition B.1 ([6]). Two distributions P, Q are said to be (ρ, δ)-*AzCDP-indistinguishable*[10] if there exist events W and W' such that

$$P(W) \geq 1 - \delta, \quad Q(W') \geq 1 - \delta,$$
$$\mathrm{R}_\alpha\left(P|_W \parallel Q|_{W'}\right) \leq \rho\alpha, \quad \text{and} \quad \mathrm{R}_\alpha\left(Q|_{W'} \parallel P|_W\right) \leq \rho\alpha,$$

where for any $\alpha > 1$, $\mathrm{R}_\alpha(U \parallel V) := \frac{1}{\alpha-1}\log\left(\int U(x)^\alpha V(x)^{1-\alpha}\,dx\right)$ denotes the α-Rényi divergence between U and V.

All the proof techniques we applied for (ε, δ)-DP also extend to hold for (ρ, δ)-AzCDP. In particular, we have to rely on the privacy filter $\phi_{\rho,\delta}$ for AzCDP that is defined similarly.

$$\phi_{\rho,\delta}((\rho_1,\delta_1),\ldots,(\rho_n,\delta_n)) := \begin{cases} 👍 & \text{if } \sum_{i=1}^{n}\rho_i \leq \rho \;\&\; \sum_{i=1}^{n}\delta_i \leq \delta \\ 👎 & \text{otherwise.}\end{cases}$$

Lemma B.2 ([46]). *For all $\rho \geq 0$ and $\delta \in [0, 1]$, the universal interactive mechanism $\mathcal{M}_{\phi_{\rho,\delta}}$ satisfies (ρ, δ)-AzCDP.*

AzCDP is useful in performing privacy accounting of the Gaussian mechanism, where using standard (ε, δ)-DP notion results in sub-optimal privacy guarantees under composition.

[10] (ρ, δ)-AzCDP is referred to as δ-*approximate* ρ-*zCDP* in [6].

Keyword	Meaning
ad	Advertisement shown on a publisher website.
ad-tech	The entity that helps advertisers & publishers with placement and measurement of digital ads.
advertiser	The entity that is paying for the advertisement, e.g. an online shoes shop.
aggregatable report	An encrypted report that is sent to ad-tech every time a trigger is registered.
ARA	Attribution Reporting API, that supports generation of *summary reports* and *event level reports* for attributed conversions.
attribution	A conversion is attributed to an impression if the ads system beleives that this conversion happened due to this impression.
conversion	An action on the advertiser website; for example, it could be a purchase.
impression	An event where a user is exposed to some marketing information; for example, an ad is shown to the user.
key discovery	The functionality that allows ad-techs to get a summary report without passing a list of keys of interest.
PAA	Private Aggregation API, that supports generation of *summary reports*, corresponding to cross-website events.
publisher	The entity that hosts the website that displays an advertisement, e.g. a news website.
requerying	The functionality that allows ad-techs to process the same report multiple times using aggregation service.
shared storage	The API that allows persisting a cross-web key-storage with read access being restricted to preserve privacy.
source	The event on publisher website registered by the ad-tech with the browser; in typical use-cases, it corresponds to an impression. The `srcKey` gets used in the generation of the aggregatable report for any trigger that get attributed to this source.
summary report	The report obtained as a result of aggregating aggregatable reports and adding noise to the result.
trigger	The event on advertiser website registered by the ad-tech that makes the ARA Client generate an aggregatable report; in typical use-cases it corresponds to conversions.

Table 2: Glossary of commonly used terminology regarding the Privacy Sandbox.