Gordon: Privacy Budget Management for W3C's Privacy-Preserving Attribution API

Abstract

Privacy-preserving advertising APIs like Privacy-Preserving Attribution (PPA) promise to enhance web privacy while enabling effective ad measurement. PPA replaces cross-site tracking with encrypted reports governed by differential privacy (DP), but current designs lack a principled approach to privacy budget management—creating uncertainty around critical design decisions. We present *Gordon*, a privacy budget manager for PPA that clarifies per-site budget semantics and introduces a global budgeting system grounded in resource isolation principles. Gordon enforces utility-preserving limits via quota budgets and improves global filter utilization through a novel batched scheduling algorithm. Together, these mechanisms establish a robust foundation for enforcing privacy protections in adversarial environments. We implement Gordon in Rust and Firefox and evaluate it on real-world ad data, demonstrating its resilience and effectiveness.

1 Introduction

Privacy-preserving advertising APIs, now under development and standardization in major browsers via the W3C, offer a rare opportunity to enhance online privacy while sustaining the web's primary funding model. Historically, browsers have lacked structured support for ad-related tasks like *conversion attribution measurement*, which requires linking ads viewed on content sites to purchases made on seller sites– a cross-origin function that is fundamentally at odds with the same-origin principle that underpins browser designs. This lack of support for the advertising workload has fueled widespread cross-site tracking through third-party cookies, fingerprinting, and other workarounds. The goal of the new APIs is to provide a well-structured, privacy-preserving alternative that aligns with browser principles while meeting advertising needs. However, these APIs remain in their early stages, with significant technical challenges still unresolved – creating a big opportunity for the academic community to contribute.

Such collaborations have already had impact, underscoring that the space is ripe for foundational contributions. The *Cookie Monster* paper, presented at SOSP last year [**cm-paper**], introduced the first formal framework based on individual differential privacy (individual DP) to systematically analyze and optimize these APIs—a framework later adopted by Google in privacy analysis of its ARA API [**ara-paper**]. That same framework now underpins Privacy-Preserving Attribution (PPA), the W3C's unified draft API undergoing standardization since October 2024 [**ppa**]. Today, the API and privacy framework landscape is largely consolidated around PPA and Cookie Monster, which anchor ongoing design and standardization efforts within *PATWG*, a W3C working group that includes representatives from all major browsers [**patwg**]. We are active participants in PATWG, tackling technical challenges from a scientific perspective to help advance the practicality of these APIs under strong privacy guarantees.

In this paper, we address a key open challenge: *privacy budget management in PPA*. PPA replaces cross-site tracking with a system where content sites register ads with the browser, seller sites request encrypted reports, and reports are only accessible DP aggregation via secure multi-party computation or a trusted execution environment. Before sending an encrypted report, the browser deducts privacy loss from a *per-site privacy budget*, limiting how much new information a site can infer about a user. While PPA, through Cookie Monster's algorithm, optimizes privacy loss accounting within each per-site budget using individual DP, it does not address how to manage these granular budgets to balance privacy with utility in an inherently adversarial advertising ecosystem.

The absence of a principled approach to privacy budget management has led to unresolved questions within PATWG, creating uncertainty in key design decisions. For instance, should some sites get budget while others do not—and if so, based on what criteria?[1] Should there be a cap on the number of sites allocated budget, and if so, how can we prevent a denial-of-service attack where one entity exhausts it?[2] Should API invocations be rate-limited to prevent privacy or DoS attacks? To date, there is no consensus, largely due to the lack of a solid foundation to drive solutions.

We describe *Gordon*, a *privacy budget manager for PPA* that addresses semantic gaps in per-site privacy loss accounting and challenges arising from the introduction of a coarse-grained global budget to protect user privacy against adversaries controlling multiple sites. To clarify the semantics of PPA's ambiguous per-site budgeting—often affected by the shifting roles of third parties in the advertising ecosystem—we propose changes to its interface, protocol, and terms of use, some of which have already been accepted by PATWG.

For the global budget, the challenge is configuring and managing it to support benign workloads while resisting malicious attempts to deplete it. Our insight is to treat the global privacy budget as a *shared resource*—analogous to traditional computing resources but governed by privacy constraints—and to apply classic resource isolation techniques, such as quotas and fair-share scheduling [**drf**; LPT+21], to this new domain. First, beyond per-site and global budgets, Gordon introduces

[1] Live discussion in W3C's PAT community group, April 2024.

[2] https://github.com/w3c/ppa/issues/69, January 2025.

quota budgets that regulate global filter consumption, ensuring graceful utility degradation for compliant sites under attack. It does so by forcing adversaries to operate within the bounds of expected workloads—bounds they can currently evade to wreak havoc on PPA's global budget. Second, recognizing that quotas can underutilize the global filter under benign conditions, we explore *batched operation*—collecting and scheduling report requests periodically—to improve utilization while maintaining DoS protection. We use a fair-share-inspired algorithm, illustrating how OS scheduling research can address foundational gaps in emerging privacy solutions. Together, these mechanisms establish a principled and practical foundation for PPA, offering browsers a robust basis for enforceable defenses.

We implement Gordon in two components: (1) `pdslib`, a generic on-device individual DP library that subsumes Cookie Monster and extends it with Gordon's budget management, and (2) integration into Mozilla Firefox's Private Attribution, a minimal PPA implementation. Upon release, these prototypes will serve as reference implementations for PPA, a service the PATWG has acknowledged as valuable.

We evaluate Gordon on a real-world dataset from the Criteo ad-tech company. Our results show that ... TODO(Pierre).

2 PPA Overview and Gaps

2.1 PPA architecture

Fig. 1(a) illustrates the architecture of *Privacy-Preserving Attribution (PPA)*, W3C's browser-based API that enables *conversion attribution measurement* while preserving user privacy. Traditionally, browsers enforce a *same-origin policy*, while conversion attribution–the process of determining whether users who see an ad later make a purchase–is inherently *cross-origin*. It requires linking ad impressions shown on content sites (e.g., *news.ex*, *blog.ex*) to conversions occurring on advertiser sites (e.g., *shoes.ex*). In the absence of a structured API for this, advertisers rely on workarounds like third-party cookies, fingerprinting, and backend data exchanges—bypassing browser policies to accommodate workloads misaligned with current API structures.

PPA addresses this gap by enabling *cross-origin ad measurement* while preserving *single-origin privacy*, using differential privacy (DP) and secure aggregation via secure multiparty computation (MPC) or a trusted execution environment (TEE). This design bounds cross-origin information leakage, allowing the API to support effective ad measurement while upholding the intention of the browser's same-origin policy.

PPA defines four principals. **Impression sites** (*news.ex*, *blog.ex*) are content sites where ads are displayed. These sites register ad impressions with the browser using the function `saveImpression()`. **Conversion sites**, a.k.a. **advertiser sites** (*shoes.ex*), are sites where purchases or other conversions occur. When a user does a conversion, these sites invoke `measureConversion()` to link the event to any relevant prior ad impressions. **Intermediary sites** (*r1.ex*, *r2.ex*) are adtechs, typically embedded as frames in impression and conversion sites, that facilitate ad delivery and measurement. Unlike traditional tracking-based adtechs, they don't collect cross-site data directly but receive encrypted reports via a third function, `getReport()`, which they then submit for secure aggregation. **Aggregation services** (e.g., *divviup.org*) are trusted MPC/TEE services that aggregate encrypted reports, applying DP to produce aggregated conversion metrics while ensuring no single entity can reconstruct individual user data.

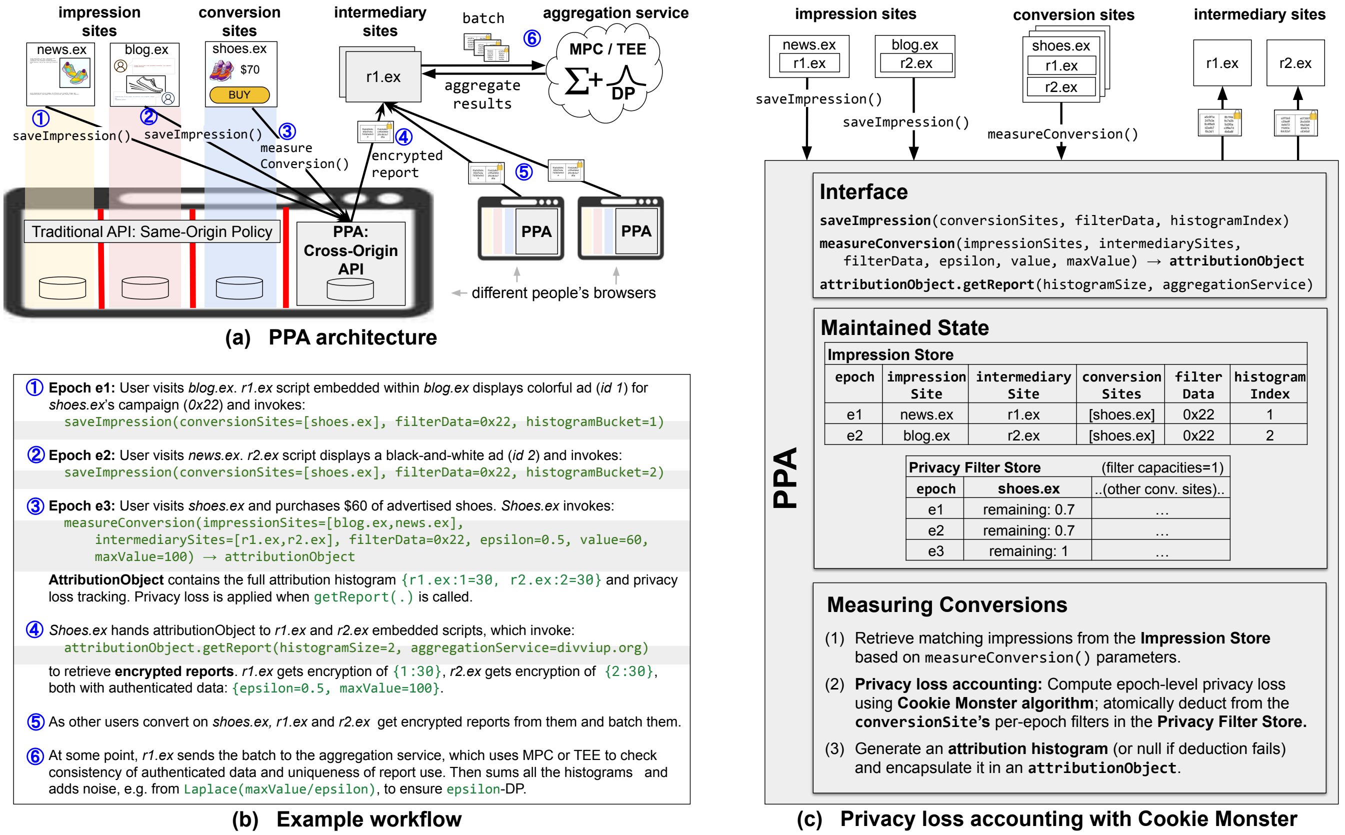
2.2 Example workflow

Fig. 1(b) shows an example workflow for PPA, consisting of six steps (the same steps are also marked in the Fig. 1(a) architecture). The example entails an advertiser, *shoes.ex*, that launches an ad campaign to promote a new product. To compare the effectiveness of two ad creatives–a colorful ad highlighting the shoe's design and a black-and-white ad emphasizing materials and comfort–*shoes.ex* partners with two placement adtechs, *r1.ex* and *r2.ex*. Each adtech places the ads on content sites, *e.g., r1.ex* on *blog.ex* and *r2.ex* on *news.ex*. In addition to placing ads, these adtechs provide a *measurement service* that allows *shoes.ex* to compare the performance of its creatives within their respective networks.

① When a user visits *blog.ex*, *r1.ex* displays the colorful ad and registers the impression by calling `saveImpression()` with the parameters shown in the figure. ② Later, the user visits *news.ex*, where *r2.ex* displays the black-and-white ad and registers it by also calling `saveImpression()`. These impressions are stored *locally in the browser* within an Impression Store, along with important metadata, shown in Fig. 1(c).

③ Subsequently, if the user visits *shoes.ex* and purchases the shoes for $60, the site invokes `measureConversion()` with the parameters shown in Fig. 1(b). This function searches the Impression Store in the browser for *relevant impressions*, matching the `impressionSite` and `conversionSite` metadata of the impressions to the parameters of `measureConversion()`. It then generates an *attributionObject*, which encapsulates the attribution histogram and manages privacy loss accounting. Assuming that PPA applies *uniform attribution*, it will assign the $60 conversion value equally between the two registered impressions, assigning $30 to each and resulting in the following attribution histogram: `{1:30, 2:30}`.

④ The attributionObject is *lazy*, i.e., no privacy loss occurs until it is used to request a report. To support DP queries, *shoes.ex* hands over the attributionObject to the *r1.ex* and *r2.ex* contexts within the browser, which invoke `attributionObject.getReport()`, specifying the aggregation service they intend to use (from a list of such services trusted by the browser). The browser processes these invocations by: (1) filtering the attribution histogram so that each intermediary only sees its own contributions (*r1.ex* gets `{1:30}`, *r2.ex* gets `{2:30}`); (2) encrypting the report and secret-sharing it (if MPC is used), while attaching some critical parameters as authenticated data, such as `epsilon` and `maxValue`; and



(a) PPA architecture

① **Epoch e1:** User visits *blog.ex*. *r1.ex* script embedded within *blog.ex* displays colorful ad (*id 1*) for *shoes.ex*'s campaign (*0x22*) and invokes:

```
saveImpression(conversionSites=[shoes.ex], filterData=0x22, histogramBucket=1)
```

② **Epoch e2:** User visits *news.ex*. *r2.ex* script displays a black-and-white ad (*id 2*) and invokes:

```
saveImpression(conversionSites=[shoes.ex], filterData=0x22, histogramBucket=2)
```

③ **Epoch e3:** User visits *shoes.ex* and purchases $60 of advertised shoes. *Shoes.ex* invokes:

```
measureConversion(impressionSites=[blog.ex,news.ex],
    intermediarySites=[r1.ex,r2.ex], filterData=0x22, epsilon=0.5, value=60,
    maxValue=100) → attributionObject
```

AttributionObject contains the full attribution histogram `{r1.ex:1=30, r2.ex:2=30}` and privacy loss tracking. Privacy loss is applied when `getReport(.)` is called.

④ *Shoes.ex* hands attributionObject to *r1.ex* and *r2.ex* embedded scripts, which invoke:

```
attributionObject.getReport(histogramSize=2, aggregationService=divviup.org)
```

to retrieve **encrypted reports**. *r1.ex* gets encryption of `{1:30}`, *r2.ex* gets encryption of `{2:30}`, both with authenticated data: `{epsilon=0.5, maxValue=100}`.

⑤ As other users convert on *shoes.ex, r1.ex* and *r2.ex* get encrypted reports from them and batch them.

⑥ At some point, *r1.ex* sends the batch to the aggregation service, which uses MPC or TEE to check consistency of authenticated data and uniqueness of report use. Then sums all the histograms and adds noise, e.g. from `Laplace(maxValue/epsilon)`, to ensure `epsilon`-DP.

(b) Example workflow

Interface

```
saveImpression(conversionSites, filterData, histogramIndex)
measureConversion(impressionSites, intermediarySites,
   filterData, epsilon, value, maxValue) → attributionObject
attributionObject.getReport(histogramSize, aggregationService)
```

Maintained State

Impression Store

epoch	impression Site	intermediary Site	conversion Sites	filter Data	histogram Index
e1	news.ex	r1.ex	[shoes.ex]	0x22	1
e2	blog.ex	r2.ex	[shoes.ex]	0x22	2

Privacy Filter Store (filter capacities=1)

epoch	shoes.ex	..(other conv. sites)..
e1	remaining: 0.7	…
e2	remaining: 0.7	…
e3	remaining: 1	…

Measuring Conversions

(1) Retrieve matching impressions from the **Impression Store** based on `measureConversion()` parameters.

(2) **Privacy loss accounting:** Compute epoch-level privacy loss using **Cookie Monster algorithm**; atomically deduct from the **`conversionSite's`** per-epoch filters in the **Privacy Filter Store.**

(3) Generate an **attribution histogram** (or null if deduction fails) and encapsulate it in an **`attributionObject`**.

(c) Privacy loss accounting with Cookie Monster

Fig. 1. PPA overview.

(3) performing privacy loss accounting before sending the encrypted reports over to the intermediaries.

⑤ As more users purchase *shoes.ex*'s advertised product, additional encrypted reports are generated, each containing zero, one, or two attributed ads. ⑥ The intermediaries batch these reports and submit them to an aggregation service, which performs the final step: (1) validating the reports, ensuring all parameters in authenticated data match and that no report is reused; (2) summing the attribution values; and (3) applying DP, adding noise (such as from a Laplace distribution with scale `maxValue/epsilon`) to protect individual users. The resulting *noised, aggregated conversion metrics* are then provided to *r1.ex* and *r2.ex*, which relay the ad-effectiveness comparison back to *shoes.ex*, helping it discern which of the colorful vs. black-and-white ads leads to higher revenue.

2.3 Privacy loss accounting with Cookie Monster

PPA enforces privacy using the individual differential privacy (individual DP) framework from the Cookie Monster paper [**cm-paper**], which tracks each user's privacy loss separately and optimizes for on-device attribution. This is a key departure from traditional DP, which maintains a single global guarantee across users. individual DP allows PPA to bound privacy loss more efficiently—based only on the actual contribution of a device to a query.

Within each browser, PPA enforces individual DP at the *epoch* level, dividing the impression stream into time intervals (e.g., a week), each with its own privacy budgets. Each device maintains an *Impression Store* to log impressions per epoch and a *Privacy Filter Store* to track per-epoch budgets. A *privacy filter* acts as the epoch's budget manager: it deducts privacy loss only if sufficient budget remains and only when data from that epoch contributes to a query; if depleted, it blocks further use of that epoch's data. Importantly, PPA maintains separate epoch-level privacy filters *per site*, a design choice that we show raises budget management questions.

Fig. 1(c) shows these internal components and illustrates how privacy loss is computed and enforced. When a conversion occurs (`measureConversion()`), the browser uses the Cookie Monster algorithm to: (1) retrieve all relevant impressions from the Impression Store, grouped by epoch; (2) compute *individual privacy loss per epoch*, using `value / maxValue * epsilon` if an epoch has at least one relevant impression, or zero otherwise; and (3) attempt atomic deduction of this loss across all contributing epochs from the *conversion site's filters*, returning a null attribution if deduction fails, and the real one otherwise.

For example, in Fig. 1(b), epochs $e1$ and $e2$ each incur an individual privacy loss of `value / maxValue * epsilon = 0.3`, while traditional DP would charge the full `epsilon = 0.5` loss. Even better, epoch $e3$, which contains no relevant impressions, incurs *zero* individual privacy loss. Although this process is nominally part of `measureConversion()`, in practice it is deferred until `getReport()`: if no report is requested, no privacy loss is incurred. Fig. 1(c) shows the resulting filter state after *r1.ex* and *r2.ex* request their reports: assuming an initial filter capacity of 1, the conversion site *shoes.ex* retains 0.7 budget in $e1$ and $e2$, and the full 1 in $e3$. In contrast, standard DP would leave only 0.5 in each epoch.

This example underscores the central advantage of individual DP accounting, which has fueled PATWG's confidence in PPA's practicality. We articulate this advantage using a novel *stock-and-flow* analogy, which captures essential dynamics of PPA's behavior under individual DP —and serves as a recurring theme in this paper.

2.4 Stock-and-flow pattern

A key informal argument for PPA's practicality, voiced in PATWG discussions, is that under *intended use*, individual DP accounting is expected to limit privacy consumption by tying it to *user actions on both impression and conversion sites*. Indeed, under individual DP, non-zero privacy loss arises only when both an impression (signifying user visit to a impression site) and a conversion (signifying visit to a conversion site) are present. This creates a *stock-and-flow pattern* of intended use, in which *privacy stock* is created on impression sites as impressions are saved, and *privacy flow* is triggered on conversion sites when reports are requested over those impressions—*both gated by user actions*. PATWG discussions generally endorse the view that users who engage more with impression and conversion sites should, naturally, incur more privacy loss—up to a limit, which we discuss next.

2.5 Global privacy filter

PPA acknowledges that relying solely on per-site filters risks exposing users to adversaries capable of coordinating API activity across multiple sites. Such behavior amplifies information gain from attribution, proportionally to the number of sites involved. Since per-site filters impose no bound on this, PPA proposes "safety limits"—per-epoch global filters that span site boundaries—originally suggested by [**pam**]. While the spec gives no detail on how to manage these filters—a gap this paper addresses (see next section)—our input has shaped the spec's guiding principles: (1) global filter capacities must be much larger than per-site budgets, by necessity; and (2) these filters should "remain inactive during normal browsing and [trigger] only under high-intensity use or attack" [**ppa**].

2.6 Foundational gaps

We identify two key gaps in PPA related to managing its two filter types–per-site and global–which we address in Gordon.

Gap 1: Unclear semantics of per-site filters. PPA adopts Cookie Monster's accounting model, which tracks privacy loss *per querier*, but is ambiguous about who counts as a querier in real-world deployments. For **single-advertiser queries**, PPA maps the querier to the conversion site—e.g., an intermediary requests a report on behalf of a specific advertiser like *shoes.ex*, and privacy loss is charged to that advertiser's budget. Yet intermediaries also receive these reports and may reuse them for their own purposes, raising the question of whether they too should be considered queriers. The ambiguity grows with PPA's planned support for **cross-advertiser queries**, where intermediaries aim to optimize across multiple advertisers (e.g., training models to choose the best ad for a given context). Since intermediaries directly benefit from such queries, PATWG plans to charge privacy loss against their own budgets. This blurs the boundary between client-serving and self-serving queries, complicating the semantics of per-site accounting and increasing the risk of report misuse. In PATWG discussions, the global filter is often cited as a fallback, offering clearer semantics even when per-site guarantees break down—but this introduces its own configuration and management challenges, which we explore next. This paper proposes changes to the PPA API, protocol, and terms of use to clarify per-site semantics and the assumptions under which they remain sound (§4.1).

Gap 2: Lack of mechanisms to manage the global filter. A critical yet under-specified part of PPA is the configuration and management of the global filter, a shared resource across all parties requesting reports from a browser. This raises two key challenges: (1) how to set its capacity to support benign workloads and (2) how to prevent malicious actors from depleting it—either to boost their own utility or to deny service to others (e.g., competitors). While per-site budgets cap consumption per domain, they offer weak protection, as domain names are cheap and easily acquired. PATWG-discussed mitigations range from requiring sites to register with a trusted authority to browser-side heuristics for identifying illegitimate use of the API. But site registration faces resistance from some industry participants for undermining the API's open nature while heuristics rely on notions of "legitimacy" that are hard to define, especially for a nascent API with no deployment history and potentially valuable, unforeseen use cases. For instance, should the number of invocations be limited? Over what period and to what value? Should access to device-side budgets be restricted? On what grounds? While discussion in PATWG continues, we argue that the group lacks a foundation—a minimal set of principled mechanisms with well-defined properties under clear assumptions—to guide browsers toward targeted, defense-in-depth strategies that are both protective and not over-constraining for the API. This paper contributes such a foundation, from the vantage point of PPA's internal privacy budget management and for two settings: the current PPA design (§4.3) and an extended version that affords more efficient mechanisms (§4.4).

3 Gordon Overview

We address PPA's gaps by (1) clarifying the two distinct threat models that per-site and global guarantees address (§3.1) and

(2) introducing Gordon to both restore the semantics of per-site filters and manage the global filter to support legitimate use while limiting abuse (§3.3). §3.2 introduces an example.

3.1 Threat model

PPA and Gordon operate under a similar threat model. Users trust their browser, device, and the browser-supported aggregation services. They extend limited trust to first-party sites they visit intentionally—i.e., through *explicit actions* like direct navigations or clicks—granting them access to first-party data and cookies. Embedded intermediaries are not trusted at all, and no site—first-party or otherwise—is trusted with raw cross-site information.

As API designers, we must address two threat levels. The first is **intended use**, which assumes well-intentioned actors. Our goal here is to make compliance easy through careful API design and well-defined semantics. In the security literature, such actors are termed *honest-but-curious*: they follow the protocol but aim to extract as much information as permitted. Because some rules cannot be enforced by protocol alone, the API must include terms of use to close this gap. We define honest-but-curious adversaries as those who respect both the protocol and its terms of use.

PPA's *per-site filters* are meant to provide strong privacy guarantees against individual honest-but-curious sites. However, ambiguities in the current API and the lack of formal terms of use leave these guarantees semantically underspecified. Gordon addresses these gaps directly.

The second level involves **adversarial use**, where actors subvert the protocol and terms of use to extract excessive user information or maximize query utility through unauthorized budget consumption. Per-site budgets offer some protection when queriers operate independently or with limited coordination, leveraging DP's compositionality. But they fail under *large-scale Sybil attacks*, where an adversary registers many fake domains to bypass per-site caps. For example, a malicious conversion site X may use automatic redirection to cycle through Sybils, each triggering a single-advertiser report that maxes out its respective filter—multiplying the user's privacy loss by the number of Sybils.

PPA's *global filter* is designed to mitigate large-scale Sybil attacks by enforcing a coarser-grained budget. However, it introduces a new vulnerability: *denial-of-service (DoS) depletion attacks*. A malicious actor can deliberately exhaust the global budget, blocking legitimate queries—either to boost their own utility or harm competitors. These attacks can mirror the Sybil strategies used against per-site filters. §4.3 gives example attacks to which PPA is currently vulnerable.

Gordon embeds resilience directly into privacy budget management to defend against DoS depletion. While this layer alone does not provide complete end-to-end protection, it establishes a strong foundation and clarifies what browsers must enforce to achieve it. Building on the stock-and-flow model from §2.4, Gordon assumes that under intended use, privacy consumption is driven by explicit user actions—such as navigations or clicks—on distinct content and conversion first-party sites. As long as benign usage adheres to this pattern, Gordon fulfills PPA's guiding principles for the global filter (§2.5): supporting normal workloads under benign conditions and degrading gracefully under attack.

For this graceful degradation to hold in practice, two assumptions must be enforced: (1) browsers can reliably distinguish intentional user actions from automatic navigations, and (2) malicious actors cannot easily induce large numbers of users to intentionally visit many distinct attacker-controlled domains. If these assumptions fail, Gordon still upholds its privacy guarantees, but its DoS resilience will diminish.

3.2 Running example

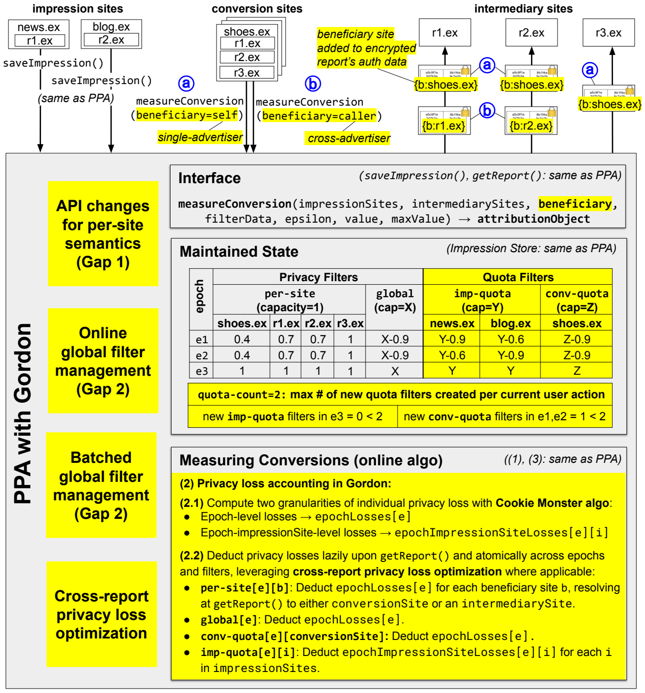
We update the *shoes.ex* example to support *cross-advertiser queries*, a feature PPA plans to add soon. Our Gordon design anticipates this shift, which significantly impacts privacy budget management. To reflect this, we modify the example: *shoes.ex* contracts with *r1.ex* and *r2.ex* for ad placement and evaluation as before, but now *r1.ex* and *r2.ex* also optimize placements across advertisers and content sites. They will each therefore be interested in obtaining two encrypted reports for each conversion: one for single-advertiser measurement on behalf of *shoes.ex* and one for cross-advertiser optimization on their own behalf. Additionally, we introduce *r3.ex*, which focuses solely on single-advertiser measurements and specializes in cross-intermediary reporting, providing a complete view of *shoes.ex*'s ad performance across the two placement intermediaries *r1.ex* and *r2.ex*. *r3.ex* will require only one encrypted report for the single-advertiser measurement on *shoes.ex*'s behalf.

3.3 Gordon architecture

Fig. 2 shows Gordon's architecture, with proposed changes to PPA highlighted in yellow (relative to Fig. 1(c)). Gordon modifies all three layers of PPA: the interface, the privacy filter architecture, and how privacy loss is accounted for during conversion measurement and report requests. These changes span four major conceptual shifts (yellow boxes on the left).

API changes for per-site semantics (Gap 1): We introduce changes in the API, protocol, and terms of use to resolve ambiguity in budget attribution. We add a *beneficiary site* parameter to the API, authenticate it toward the aggregation service, and restrict report usage—both at the aggregator and via terms of use—to ensure reports serve only the intended site's DP queries. These changes prevent intermediaries from re-purposing reports funded by conversion sites, restoring semantic clarity to per-site accounting for parties who comply with the protocol and its terms. This resolves PPA's Gap 1 (§2.6). Though not targeted at global filter defense, these changes outlaw Sybil and DoS attacks against it.

Online global filter management (Gap 2): Without further altering the API or protocol, we rework PPA's internal state



Interface *(saveImpression(), getReport(): same as PPA)*

```
measureConversion(impressionSites, intermediarySites, beneficiary,
    filterData, epsilon, value, maxValue) → attributionObject
```

Maintained State *(Impression Store: same as PPA)*

epoch	Privacy Filters					Quota Filters		
	per-site (capacity=1)				global (cap=X)	imp-quota (cap=Y)		conv-quota (cap=Z)
	shoes.ex	r1.ex	r2.ex	r3.ex		news.ex	blog.ex	shoes.ex
e1	0.4	0.7	0.7	1	X-0.9	Y-0.9	Y-0.6	Z-0.9
e2	0.4	0.7	0.7	1	X-0.9	Y-0.6	Y-0.9	Z-0.9
e3	1	1	1	1	X	Y	Y	Z

quota-count=2: max # of new quota filters created per current user action

new **imp-quota** filters in e3 = 0 < 2 | new **conv-quota** filters in e1,e2 = 1 < 2

Measuring Conversions (online algo) *((1), (3): same as PPA)*

(2) Privacy loss accounting in Gordon:

(2.1) Compute two granularities of individual privacy loss with **Cookie Monster algo**:
- Epoch-level losses → `epochLosses[e]`
- Epoch-impressionSite-level losses → `epochImpressionSiteLosses[e][i]`

(2.2) Deduct privacy losses lazily upon `getReport()` and atomically across epochs and filters, leveraging **cross-report privacy loss optimization** where applicable:
- **`per-site[e][b]`**: Deduct `epochLosses[e]` for each beneficiary site `b`, resolving at `getReport()` to either `conversionSite` or an `intermediarySite`.
- **`global[e]`**: Deduct `epochLosses[e]`.
- **`conv-quota[e][conversionSite]:`** Deduct `epochLosses[e]`.
- **`imp-quota[e][i]`**: Deduct `epochImpressionSiteLosses[e][i]` for each `i` in `impressionSites`.

Fig. 2. Gordon architecture. Changes vs. PPA (Fig. 1(c)) in yellow. Roxana: Need to replace X, Y, Z with concrete values based on evaluation.

and privacy loss accounting to defend the global filter from DoS depletion by malicious sites. We introduce *quota filters*, which regulate consumption to either isolate compliant sites from attack or ensure their utility degrades gracefully. Unlike per-site and global filters, which enforce privacy semantics, quota filters are solely for utility preservation—a novel use in the DP literature. We bound three quantities: (1) *how much* each impression site can contribute to global budget consumption via impressions registered for an adversary; (2) *how much* each conversion site can trigger consumption through report requests from an embedded adversary; and (3) *how many* unique impression and conversion site domains can interact with the API following a single user action. The first two are DP filters that consume budget in sync with the global filter and are tuned to support benign workloads. The third is a lightweight counter that anchors quota instantiation to a user-driven trigger. Together, these address PPA's Gap 2 under its current operation.

Batched global filter management (Gap 2): While effective, quotas may underutilize the global filter under benign conditions. This is analogous to the classic tradeoff in scheduling, where static partitioning of resources across participants, while being fair, can lead to underutilization. In PPA's current online model, where each report request must be resolved immediately, quotas are the main tool for ensuring isolation. But relaxing them (e.g., redistributing unused quota) risks reintroducing attack vectors. To address this, we explore *batched operation*, where report requests are collected and scheduled periodically in fixed *scheduling intervals* (e.g., daily). In this setting, a scheduler can schedule all remaining unscheduled requests fairly, improving utilization while maintaining comparable DoS protection. This also addresses Gap 2, but under a modified operation.

Cross-report privacy loss optimization: We formalize a new budget optimization across Gordon's filters. Building on insights first surfaced by PATWG participants, we observe that multiple reports tied to the same conversion—e.g., issued by different intermediaries—may not incur additional privacy loss against a shared filter if their impression sets are disjoint, since they reveal no more than the original attribution histogram to a common observer. We prove this property formally and leverage it in Gordon to optimize accounting across multiple privacy and quota filters.

4 Detailed Design

We detail Gordon's four core components, grounding each in our running example and the right side of Fig. 2.

4.1 API changes for per-site semantic (Gap 1)

We begin by detailing our proposed changes to clarify the semantics of PPA's per-site filters, which are intended to guarantee privacy against honest-but-curious sites. Currently, they do not—due to ambiguity in how privacy loss is charged to one site's budget while reports are delivered to another. In our *shoes.ex* example, intermediaries *r1.ex* and *r2.ex* each request a report on a conversion event, nominally on behalf of *shoes.ex*. PPA treats this as a single-advertiser query and deducts privacy loss from *shoes.ex* 's budget, since the reports are meant to help it measure ad performance. However, both *r1.ex* and *r2.ex* receive the reports and may also use them for their own analytics. Now suppose *r1.ex* and *r2.ex* want to run cross-advertiser queries—e.g., training a model to choose between showing an ad from *shoes.ex*, *hats.ex*, or *bags.ex*, based on content-site context. In that setting, PPA plans to charge privacy loss to the intermediaries themselves, recognizing them as the primary beneficiaries. But when the same entities receive reports in both contexts, the line between client-serving and self-serving queries becomes blurred. Without clear guardrails, even well-meaning intermediaries might be tempted to reuse reports charged to *shoes.ex* 's budget to improve their own cross-advertiser DP query.Poor incentives also arise for conversion sites themselves, who may shard into domains like *shoes.ex*, *shoes-cart.ex*, and *shoes-purchase.ex* to stretch their privacy budgets for more accurate measurements. Without clear constraints on report usage, per-site privacy loss accounting becomes semantically incoherent even for honest-but-curious actors.

Proposed changes. Our approach in Gordon is to explicitly declare the *beneficiary site*—the site on whose behalf the DP query will be run. This requires modifications to the API, the protocol with the aggregation service, and the API's terms of use. For the **API**, we introduce a `beneficiary` parameter in `measureConversion()`, which resolves to the conversion site for single-advertiser measurements and to the requesting intermediary site for cross-advertiser optimizations. This resolution happens upon the `getReport()` invocation. The browser deducts privacy loss from the epoch-level filters of the beneficiary site, creating these filters if necessary. We place no restrictions on per-site filter creation, as for these filters we operate under the honest-but-curious model.

For the **protocol**, we include the query beneficiary site in the report's authenticated data and require the trusted aggregator to verify its consistency across all reports in a batch. If any two reports diverge on the beneficiary, the query will be rejected. This prevents intermediaries from re-purposing reports obtained for different conversion site clients—funded by the clients' distinct privacy budgets—for their own DP cross-advertiser queries.

For the API's **terms of use**, we mandate that: *DP-query results tied to a specific* `beneficiarySite` *shall not be used to improve queries for other beneficiaries*. Thus, not only can reports obtained for a given `beneficiarySite` solely be used in DP queries on its behalf, but also results obtained from them will only be shared with that site, regardless of who executes the queries. The preceding terms of use are more powerful than first meets the eye. In particular, it prohibits combining reports and DP results that a company might obtain by assuming multiple identities (e.g., *shoes.ex*, *shoes-cart.ex*, *shoes-purchase.ex*). It also makes Sybil behavior as described in §3.1 unlawful, hence we can expect honest-but-curious sites to not engage in it.

These measures remove ambiguity and make privacy loss accounting against a single site's budget semantically meaningful under well-defined assumptions clearly stated in the terms of use. PPA has already incorporated the protocol change following our recommendation and we plan to propose the other two changes shortly.

Example. In Fig. 2, *shoes.ex* invokes `measureConversion()` twice for a single \$60 purchase: ⓐ once for its own measurement (`beneficiary = self`) and ⓑ once for intermediary optimization (`beneficiary = caller`), creating two separate `attributionObjects`. For the first `attributionObject` ⓐ, any intermediary (*r1.ex*, *r2.ex*, *r3.ex*) can call `getReport()`, which resolves the beneficiary to *shoes.ex* and deducts from its per-epoch filters. Thanks to Cookie Monster's per-query accounting and Gordon's cross-report optimizations (§4.2), *shoes.ex* retains 0.4 budget in epochs $e1$ and $e2$, and a full 1 in $e3$. For the second `attributionObject` ⓑ, *r1.ex* and *r2.ex* call `getReport()` on their own behalf, which resolves the beneficiary to *r1.ex* and *r2.ex*, respectively, triggering deduction from their own budgets and leaving 0.7 in $e1$ and $e2$ for each. Since *r3.ex* does not issue optimization queries, it receives no such object and preserves its budget. Each report returned by Gordon includes the authenticated beneficiary, shown in yellow over the ciphertext: `b:shoes.ex` marks single-advertiser reports requested by all intermediaries; `b:r1.ex` and `b:r2.ex` label cross-advertiser reports for *r1.ex* and *r2.ex*, respectively.

4.2 Cross-report privacy loss optimization

We illustrate Gordon's *cross-report* privacy loss optimization using our running example, deferring a general treatment to Appendix A.2. This optimization is orthogonal to Cookie Monster's *per-report* individual-DP-based strategies (§2.3), and instead leverage structure *across* reports, often requested by different intermediaries for the same conversion.

In Fig. 2, *r1.ex*, *r2.ex*, and *r3.ex* request single-advertiser reports from `attributionObject` ⓐ, all on behalf of client *shoes.ex*; separately, *r1.ex* and *r2.ex* request cross-advertiser reports from ⓑ for their own purposes. All five reports operate on the same attribution histogram, assigning \$30 to each of two impressions (epochs `e1`, `e2`). Cookie Monster computes a base epoch-level privacy loss of 0.3 per report (§2.3). Naïvely, one would expect a cumulative deduction of 0.9 from *shoes.ex*'s filters (three reports) and 1.5 from the global filter (five reports). Yet the Privacy Filters table shows only deductions of 0.6 and 0.9, respectively.

The discrepancy arises because some reports *shard* the histogram into non-overlapping pieces—enabling parallel-composition-like optimizations. *r1.ex* and *r2.ex*'s single-advertiser reports from ⓐ each include a disjoint portion: `{1:30}` and `{2:30}`, respectively. Since both are funded by the same per-site filter (of *shoes.ex*), their combined release leaks no more than a single full histogram toward *shoes.ex*, incurring only 0.3 privacy loss. They likewise count as one deduction against the shared global filter. In contrast, *r3.ex*'s report includes the full histogram (to give *shoes.ex* a complete view across intermediaries; see §3.2), overlapping with both *r1.ex* and *r2.ex* and adding another 0.3 of loss to both *shoes.ex*'s filter and the global filter. A similar optimization applies to cross-advertiser reports from ⓑ. These are funded from separate filters (those of *r1.ex* and *r2.ex*), so each incurs 0.3 loss. But against the global filter, they again count as one, bringing the total global filter deduction to 0.9 instead of the unoptimized 1.5.

Appendix A.2 formalizes the optimization, whose logic we encapsulate in the `attributionObject`. This object dynamically optimizes budget deduction across the per-site, global, and quota filters on each `getReport()` call, on the basis of prior invocations and deductions.

4.3 Online global filter management (Gap 2)

With per-site filters clarified and optimized, we can now hope for *strong privacy guarantees against individual honest-but-curious sites*, assuming tight configuration of these filters (e.g., capacity $\epsilon_{per-site} = 1$). Our terms of use also prohibit collusion, as DP results across identities must not be combined. Still, non-compliant behavior remains possible, making the global filter essential as a safeguard against worst-case privacy loss—i.e., an adversary accessing results from all sites. Reports are returned only if they can be funded by both per-site and global filters. Appendix **??** formalizes this multi-granularity filter accounting, which we have not seen articulated in prior work, despite its practical significance.

The key challenge lies in configuring and managing the global filter to support benign workloads while resisting depletion attacks. We begin by outlining specific attacks and limitations of existing defenses, motivating our defense.

DoS depletion attacks. An adversary X may aim to exhaust the global budget—either to increase their own utility or disrupt others'. The latter is especially potent in PPA via single-advertiser queries and becomes more dangerous if cross-advertiser support is added. To bypass per-site limits, X can register $s = \epsilon_{\text{global}}/\epsilon_{\text{per-site}}$ Sybil domains and distribute queries across them.

Attack 1: Cross-advertiser reports. X builds a site embedding the s Sybil domains as intermediaries. When user u visits, the site: (1) registers r impressions with X as the conversion site and a different Sybil as intermediary; and (2) has each intermediary request a cross-advertiser report, exhausting its per-site budget. This drains global budget for u. If many users visit X even once in an epoch, X can disrupt measurements by other sites for that epoch. If users continue arriving across epochs, X can sustain disruption—mounting a persistent attack with a single popular site and just one visit per user per epoch. PPA isn't currently vulnerable, lacking cross-advertiser support. But a similar attack works via single-advertiser reports:

Attack 2: Single-advertiser reports. Now the Sybils serve as impression and conversion domains. When u visits X, X auto-redirects s times, switching domains to register impressions and trigger single-advertiser reports in lock step; each Sybil can register impressions for a different Sybil as conversion site. As before, this depletes the global budget. While aggressive redirection might be heuristically flagged, redirection is too common for browsers to block outright.

Attack 3: Single-advertiser reports, subtler version. X (1) registers s impressions with Sybil conversion sites and (2) redirects once to load a new Sybil domain that requests a report. Reports use maximum attribution windows to draw budget across past epochs via impressions previously registered by X. If many users visit X's site roughly s times over each epoch's *data lifetime* (typically months), X can drive sustained global budget depletion—again with a single site but requiring more than one visit per user.

Limitations of existing defenses. Certain behaviors in these attacks clearly exceed reasonable use. For example: (1) PPA is intended for cross-site measurement, so identical values for `impressionSite` and `conversionSite` (Attack 1) should be disallowed. (2) A single user action should not drive both registration of an impression and trigger a conversion measurement of it—even under different domains (Attack 1). (3) Excessive redirection (Attack 2) should be detectable. (4) Allowing an epoch's entire global budget to be exhausted in seconds is fundamentally flawed. These heuristics imply minimal browser defenses, but more principled protections are needed to guard against subtler abuse like Attack 3.

In PATWG discussions, several mitigations have been proposed—restricting which sites receive per-site filters (e.g., via mandatory registration), rate-limiting API calls, or capping the number of sites granted filters per epoch. While useful as part of browsers' defense-in-depth strategies, such measures risk over-constraining a nascent, evolving workload. Mandatory registration could limit API access, undermining the open-web ethos. Hard limits on per-site impression counts are tricky: some sites may show many ads, others few. The same with conversions, which can represent many things, from rare purchases to frequent XXX TODO(Ben): give an example of XXX. Capping the number of intermediaries per conversion might constrain advertisers' ability to work with diverse partners. And if the API is repurposed beyond advertising—for instance, to measure engagement or reach—workload characteristics may shift further. Fixed constraints that seem reasonable today could stifle innovation or penalize legitimate new use.

Our approach: Enforce stock-and-flow. We seek approaches grounded in *minimal workload assumptions*, offering a flexible springboard for implementing effective protection in browsers without over-constraining the API. As discussed in §2.3, intended API use follows a *stock-and-flow pattern*, where privacy loss is driven by explicit user actions—such as navigations or clicks—across distinct content and conversion domains. Attacks above break this pattern by relying on automatic flows and collapsing domain roles.

We restore this pattern through *quotas* that govern privacy consumption: (1) impression-site quotas cap stock creation per epoch, (2) conversion-site quotas cap triggered flow per epoch, and (3) a count-based limit bounds the number of new sites that can create the preceding quotas in response to a single user action. Unlike indirect metrics (e.g., such as API call counts per unit of time, number of intermediaries, or domains with per-site filters), our first two quotas operate directly on the core protected resource: the global filter. Each is grounded in a notion of *share*, calibrated to expected workloads. Finally, the third quota enforces the stock-and-flow pattern's behavioral anchor—explicit user action. Together, these quotas force adversaries to operate within the contours of expected workloads, significantly curbing their ability to drain the global budget from a single site with limited user interaction.

Gordon quota system. Fig. 2 highlights (yellow background) the internal state maintained by Gordon to manage PPA's global privacy filters. We use two types of quotas: (1) *quota filters* (`imp-quota`, `conv-quota`) implemented as *DP filters* not for privacy accounting but to regulate global filter consumption; and (2) a standard *count-based quota* that limits the number of new quota filters that can be instantiated per user action (e.g., a click or navigation). Algorithms XXX-ZZZ in appendix formalize the system's behavior.

The impression-site quota filter, `imp-quota`, is scoped per impression site and per epoch. It bounds the portion of global privacy loss attributable to flows leveraging stock created by impressions from that site. When a site `i` first invokes `saveImpression()` in epoch `e`, Gordon creates `imp-quota[e][i]` with a preconfigured capacity representing its share of the global filter. While any site `i` can receive a `imp-quota`, it is only consumed if a subsequent report matches an impression from `i` in that epoch—that is, if the site's stock is used.

The conversion-site quota filter, `conv-quota`, is defined symmetrically: scoped per conversion site and per epoch, it bounds the global privacy loss attributable to flows initiated by conversions on that site. `conv-quota[e][c]` is created when conversion site `c`, on or after epoch `e`, first calls `measureConversion()` in a way that could incur non-zero individual privacy loss in epoch `e`. It is consumed upon a `getResult()`– that is, when a privacy flow occurs.

Fig. 2 sketches Gordon's privacy loss accounting algorithm (box "Measuring Conversions;" full algorithm in appendix, Algorithms XXX-ZZZ). First, individual privacy losses are computed per epoch using the Cookie Monster algorithm. Then, on each `getReport()` call, we attempt to deduct these losses across all relevant filters and epochs in an atomic process that succeeds and alters any filter's state only if all checks pass. For each epoch `e`, the relevant filters are: `per-site` of the beneficiary, `global`, `conv-quota` of the conversion site, and `imp-quota` for each impression site with non-zero loss (as computed by Cookie Monster). To efficiently handle impression-site quotas, we scope loss computation to the (epoch, impression site) level and charge the resulting loss to the corresponding `imp-quota` filter. We also apply the cross-report optimizations from §4.2 to remove redundant charges. While the combined capacity and budget usage of all instantiated impression-site (or conversion-site) quotas may at any time exceed that of the global filter, our algorithm's checks against both the quotas and the global filter ensure that the global privacy guarantee is never breached.

Quota filters limit how much each first-party site can contribute to global privacy consumption. In a world without automatic redirects—where each domain change stems directly from a user action—this would suffice to reestablish PPA's intended user-driven stock-and-flow behavior. But since automatic redirects are pervasive on the web, we relax that assumption: following a single explicit user action, we permit a bounded number of unique first-party domains that trigger creation of new quota filters in an epoch. This bound, `quota-count`, is configurable, and while not necessarily one, we expect it to be small (e.g., 2 or 3). Finally, we recommend that browsers disallow a single domain from registering both an impression and a conversion upon the same user action.

"Normal" workload parameters:

M: max # of impression sites in an epoch contributing to non-zero loss in epoch.

N: max # of conversion sites that request non-zero loss from an epoch.

n: max # of conversion sites that request non-zero loss from a single (epoch, impression site) pair.

r: max budget consumed by an intermediary's cross-advertiser queries on a single conversion site, as a fraction of the intermediary's $\epsilon_{\text{per-site}}$.

Filter	Capacity configuration
Per-site filter	$\epsilon_{\text{per-site}}$: configuration parameter
Global filter	$\epsilon_{\text{global}} = \max(N, n \cdot M)(1+r)\epsilon_{\text{per-site}}$
Impression-site quota	$\epsilon_{\text{imp-quota}} = n(1+r)\epsilon_{\text{per-site}}$
Conversion-site quota	$\epsilon_{\text{conv-quota}} = (1+r)\epsilon_{\text{per-site}}$

Tab. 1. Gordon filter configurations.

Configuration to "normal" workload. A key question is how to configure privacy and quota filters to avoid impeding "normal" workloads in no-attack scenarios. Our approach has three steps. First, we define four browser-adjustable parameters that outline the scale of the intended workload, denoted N, M, n, r and defined at the top of Table 1. Second, given these parameters and $\epsilon_{\text{per-site}}$, we specify constraints that configurations of ϵ_{global}, $\epsilon_{\text{imp-quota}}$, and $\epsilon_{\text{conv-quota}}$ must satisfy to support this workload: $\epsilon_{\text{conv-quota}} \geq (1+r)\epsilon_{\text{per-site}}$ (to support $\epsilon_{\text{per-site}}$ for single-advertiser queries plus $r\epsilon_{\text{per-site}}$ for cross-advertiser queries); $\epsilon_{\text{imp-quota}} \geq n\epsilon_{\text{conv-quota}}$ (to allow an impression site to be queried by n conversion sites using their full $\epsilon_{\text{conv-quota}}$); $\epsilon_{\text{global}} \geq N\epsilon_{\text{conv-quota}}$ and $\epsilon_{\text{global}} \geq M\epsilon_{\text{imp-quota}}$ (to support N conversion and M impression sites at full quota). Third, we solve these constraints to derive configuration formulas. The bottom of Table 1 shows intuitive formulas from manual resolution. We use these formulas to establish Gordon 's analytical resilience below and evaluate it experimentally in §6.

Resilience to DoS depletion. We prove the following property of Gordon's resilience to DoS depletion attacks:

Theorem 1 (Resilience to DoS depletion)**.** *Consider an adversary who manages to create M^{adv} and N^{adv} `imp-quota` and `conv-quota` filters, respectively. The maximum budget ϵ^{adv}_{global} that the adversary can consume from the global filter on a device d is such that:*

$$\epsilon^{adv}_{global} \leq \min(M^{adv}\epsilon_{imp\text{-}quota},\ N^{adv}\epsilon_{conv\text{-}quota}).$$

Proof. Appendix A.1.5. □

This rules out *Attack 1* from earlier, where all impressions and conversions are registered under the same top-level domain, forcing reports through a single `imp-quota` and `conv-quota`. This yields $M^{\text{adv}} = N^{\text{adv}} = 1$, capping the attacker's global-filter consumption at $\epsilon^{\text{adv}}_{\text{global}} \leq \min(\epsilon_{\text{imp-quota}}, \epsilon_{\text{conv-quota}})$, far from depleting it.

Our `quota-count` bound further blocks *Attack 2*, which, following a single user visit, uses automatic redirection to hop among domains and register impressions and conversions. The `quota-count` —expected to be small—limits the number of quota filters that can be created after one user action. This ensures the adversary can consume only modestly more than in Attack 1 (roughly half `quota-count` times more if impression-/conversion-site quota capacities are similar).

In more general cases, such as *Attack 3*, an adversary who manages to receive U_{adv} interactions from device d at epoch e is limited to the creation of M^{adv} and N^{adv} `imp-quota` and `conv-quota` quotas, respectively, such that $M^{\text{adv}} + N^{\text{adv}} \leq \texttt{quota-count} \times U^{\text{adv}}$. Here, our quotas ensure *graceful degradation* for the benign workload as a function of U^{adv}. We prove the following bound for such an adversary's impact:

Theorem 2 (Graceful degradation)**.** *Consider a adversary collecting U^{adv} user actions on sites under their control for device d. Under the configuration of Table 1, the budget ϵ^{adv}_{global} that this adversary can consume from the global filter is upper-bounded by:*

$$\epsilon^{adv}_{global} \leq (1+r)\epsilon_{per\text{-}site} \times \frac{n}{1+n}\left(\texttt{quota-count} \times U^{adv}\right).$$

Proof. Thm. 1 implies the most efficient way to allocate the $\texttt{quota-count} \times U^{\text{adv}} = M^{\text{adv}} + N^{\text{adv}}$ filter creations available to the attacker is such that $M^{\text{adv}}\epsilon_{\text{imp-quota}} = N^{\text{adv}}\epsilon_{\text{conv-quota}}$, or $M^{\text{adv}} n(1+r)\epsilon_{\text{per-site}} = N^{\text{adv}}(1+r)\epsilon_{\text{per-site}}$. This yields $M^{\text{adv}} = \frac{1}{n+1}\texttt{quota-count} \times U^{\text{adv}}$ and $N^{\text{adv}} = \frac{n}{n+1}$ $\texttt{quota-count} \times U^{\text{adv}}$. Applying Thm. 1 concludes the proof. □

4.4 Batched global filter management (Gap 2)

As in classic scheduling, static resource partitioning—used in our quota system—can underutilize the global filter in benign cases. Consider a device that visits only two impression sites: *news.ex*, which serves ads for many conversion sites, and *blog.ex*, which serves just one. Due to the per-site filter, *blog.ex*'s lone conversion site can consume only a small fraction of the global budget ($\epsilon_{\text{per-site}}$). Meanwhile, conversions drawing on *news.ex*'s impressions are limited to its quota ($\epsilon_{\text{imp-quota}}$), even if they serve many conversions across different sites. If these are the only visited impression sites, much of the global budget remains unused—despite no privacy harm in letting *news.ex* consume more. This underutilization would especially hurt major impression sites that may host high user traffic, so it is critical to address.

In PPA's current online operation—where every report request must be answered immediately—quotas are necessary to enforce isolation. We observe that if PPA allowed *batched operation*, where requests are collected over time, the scheduler can sort the pending requests by some metric that ensures isolation, while still trying to fully utilize the remaining budget.

Roxana: We need a precise algorithm in the appendix. The below is not 100% clear what different things mean, e.g., what's a fair share, what the quotas are in the release setting, when the fair share is calculated and based on what, etc.

Scheduling interval and response time. We divide each epoch's data lifetime into T *scheduling intervals* (e.g., one week each), each with three phases: initialization, online, and batch. At the start of initialization, quotas reset and a portion of the global filter's budget is *released* ($\frac{\epsilon_{\text{global}}}{T}$). This released budget, combined with any leftover from the prior interval, is used to complete as many pending requests as possible.

We extend PPA's API to let requests specify a *response time*—the interval after which the response is returned. For privacy, responses are delivered only at the declared time, regardless of when (or whether) the request is scheduled. If it cannot be scheduled in time, an encrypted null report is returned. Shorter response times yield faster responses but reduce the chance of success due to tighter budget availability. Requests that span multiple scheduling intervals have higher chances of allocation. We next elaborate the three phases of the scheduling intervals, starting with the online phase.

Online phase. In the online phase of each interval, requests consume budget by the order of arrival. In this phase, only requests that have sufficient budget left in their quotas, as well as in the privacy filters, get scheduled. Since they are limited by their quotas, they have to request less or equal to their *fair share*, TODO(Asaf): pls define fair share (the quotas are not "fair shares" in the global filter as I initially thought). Pierre: In the code (and pseudocode in appendix) I don't need to define fair share. Maybe this means we can define a "fair" request as a request that gets allocated in the online phase? Whether a request gets allocated or not depends on the previous requests that came before. But in particular, to be allocated a request needs to pass the impression-site quota for each requested impression site i, *i.e.,* $\epsilon_i^{<t} + \epsilon_i \leq \epsilon_{\text{imp-quota}}$, where ϵ_i is the budgeted requested for site i and $\epsilon_i^{<t}$ is the budget allocated to site i on previous requests. otherwise they are requesting more budget than their quotas allow. If requests arrive and are not scheduled (either because they have no more quota or filter budget left, or because they requested more than their fair share), they are added to a *queue* and may get another chance at being scheduled in the batch phase.

Batch phase. At the end of the scheduling interval, Gordon tries to utilize all the unused global budget that has been released so far, by scheduling from the remaining requests in the queue that have not already generated responses. To this end, Gordon tries to schedule the requests in the queue one by one, making sure they have sufficient per-site and global filter budgets, but ignoring the quotas. There are different possible algorithms for sorting the requests in the queue, with different possible objective functions. For example, we could try to maximize per-site or global budget consumption, or conversely optimize for fairness across the different impression sites. By default, we choose the following algorithm, which optimizes for fairness. Gordon schedules outstanding requests one by one from the queue. Each time after it schedules a request from the queue Gordon sorts the requests in the queue based on the impression site that received the least budget so far, and among the requests that belong to the same impression site it sorts the requests based on their requested privacy budget (from small to large). It continues greedily scheduling requests until it reaches a point where either there are no more requests to schedule, or all the remaining requests cannot be scheduled, since they do not have sufficient remaining budget in the filters.

Initialization phase. All the requests that have not been allocated budget at the end of the batch phase and still do not need to be returned to the querier site, are forwarded to the queue of the next scheduling interval. As mentioned before, at start of the new scheduling interval, we reset the quotas and $\frac{\epsilon_{\text{global}}}{T}$ of the global filter's budget is released. Before starting the online phase of the new scheduling interval, Gordon first tries to schedule any outstanding requests from the queue that requested less or equal to their fair share, using the newly-reset quotas and newly-released global budget. After this initial step, the online phase of the next scheduling interval begins.

Analysis. We prove three properties of the algorithm: ability to utilize the global filter, resilience to depletion, and the incentive it creates for requests to wait as long as they can.

Theorem 3 (Pareto efficiency)**.** *The batched global filter scheduling algorithm is Pareto efficient with regards to the global filter budget.*

Proof. Appendix **??**. □

This property suggests that the batched algorithm should achieve high global-filter utilization. However, this guarantee applies only to the global-filter scheduling algorithm, not to Gordon as a whole: per-site filters—which are not managed by the scheduler—may still block some requests, limiting overall utilization. In practice, as shown in §6.5, Gordon with batching does improve utilization.

Theorem 4 (Resilience to DoS depletion)**.** *XXX*

Proof. Appendix **??**. □

Theorem 5 (Response time tradeoff)**.** *The longer the requested response time, the higher the probability requests will get allocated privacy budget.*

Proof. See Appendix B, by induction. □

4.5 Recommendations for PATWG

Gordon provides browsers with foundational building blocks for defending against DoS depletion attacks on PPA's global filter—though not an end-to-end solution. Operating within the budget management layer, our techniques offer built-in resilience independent of specific web attack vectors. However, they rest on assumptions—namely, that attackers cannot easily induce many users to visit many attacker-controlled domains—which browsers must enforce to achieve full protection. Our threat model (§3.1) leaves enforcement out of scope, but Gordon establishes a foundation to drive end-to-end solutions, which has so far been lacking in PATWG, hindering its progress. We conclude this section with a set of **Don'ts** and **Do's**, some addressing directions raised in PATWG.

Don'ts: (1) *Don't rate-limit API invocations*: This is not directly useful and risks stifling benign use cases. In Gordon, sites may register arbitrary impressions and conversions, and intermediaries may request any number of reports. The true limit is on how many distinct *domains* can act after a single user action. (2) *Don't limit the number of per-site filters*: These are meant to track privacy loss from honest-but-curious sites. They are not suitable levers for defending the global filter from malicious actors trying to deplete it. (3) *Don't require intermediary registration*: With proper budget management—by Gordon and by first parties managing their own quotas—intermediaries do not impact privacy or resilience guarantees. While Gordon leaves to future work the ability for first parties to control how intermediaries consume their quota, we believe this can be done rigorously, further reducing the need for intermediary registration with a PPA authority.

Do's: Focus on *detecting and disabling patterns of site sharding across domains*. For example, sites may attempt to shard themselves—e.g., routing each user interaction through a distinct domain—to inflate their quota access and deplete the global filter. While some level of sharding is inevitable (e.g., legitimate third-party integrations like shopping carts), aggressive self-sharding for DoS purposes should be explicitly prohibited. First, PPA should ban such behavior in its terms of use, which large, legitimate sites are likely to respect. Second, browsers should develop heuristics to detect noncompliant patterns and block offending sites from using the API. One possible signal is when a landing site frequently links to dynamically changing domains that invoke the API before returning users to the same main site. Third, Gordon's sorting algorithm could be extended to penalize suspicious-but-not-yet-blocked behavior. These are examples of concrete, actionable directions that PATWG can now pursue based on the resilience foundation provided by Gordon.

5 Prototype

Roxana: WIP. Do not read.

We implement Gordon in two components: (1) `pdslib`, a generic on-device individual DP library and (2) its integration into Mozilla Firefox's Private Attribution, a minimal PPA implementation. **`pdslib`:** TODO(Mark).

Firefox integration: TODO(Giorgio).

6 Evaluation

Pierre: WIP. Do not read.

Evaluation questions:

Q1: What parameters define "normal" operation in the Criteo workload?

Q2: In the online setting, how do query error rates vary with different quota capacities?

Q3: Can quotas preserve low error rates for benign queries under DoS attacks?

Q4: Do quotas lead to under-utilization, and can batching mitigate this?

6.1 Methodology

Datasets. Pierre: Criteo PrivateAd, resampled. Multiple advertisers, multiple publishers, one intermediary (Criteo). Training on the first 10 days, evaluating on the last 20 days.

Evaluation scenario. Pierre: Each advertiser runs single-advertiser measurements. Histogram queries.

Benign workload process. Pierre: Generate conversions. Keep the 73 conversion sites that have more than 100 reports per day on average. Define the histogram buckets. Calibrate the batch size.

Attack workload process. Pierre: ...

Metric. Pierre: RMSRE, histogram generalization from ARA paper.

Roxana: Move a very short version of the below to the first section that studies error attribution. We further analyze how much each type of filter contributes to the overall error in a query, by introducing an "error cause" metric defined as follows. For each report, we compute how many and which filters were out-of-budget while computing the report. If any per-site filter filter is out-of-budget, we label the whole report as potentially biased and attribute the error to the per-site filter filter. We look at global filter filters next, then conversion-site quota filters and finally impression-site quota filters. Next, for each query we compute the number of reports in each category, and divide it by the number of reports in the query. This gives us the fraction of reports in the query affected by the per-site filter (resp. global filter, impression-site quota, conversion-site quota) filter. Finally, we average the fractions over all the queries in the workload.

Pierre: Expand on other sources of error, DP error and RMSRE calibration. Bias variance.

Baselines. We compare Gordon against two baselines. The first baseline is **Cookie Monster**, which only maintains per-site filter filters. The second baseline, which we call **PPA**, is Cookie Monster augmented by a global filter filter, which corresponds to the current state of the PPA draft specification with the global filter filter as a safety limit.

Defaults. $\epsilon_{\text{per-site}} = 1$.

6.2 "Normal" workload parameters in Criteo (Q1)

In §4.3 we established sufficient conditions under which filters do not harm normal operation, thanks to parameters r, N, M and n. In our Criteo workload we have $r = 0$ since we only consider measurement queries, and can thus take $\epsilon_{\text{conv-quota}} = \epsilon_{\text{per-site}}$. However, determining the values of N, M and n is not immediate since it would require executing a benign workload to count how many non-zero reports each device-epoch sent to or from various parties. Pierre: TODO: do that, it's tighter and easier to explain in the text. Keeping rough statistics as a placeholder, but they are overestimations (*e.g.,* count of "authorized" conversion sites for each impression site, instead of actual conversions that happen). Instead, we can derive upper bounds $\tilde{N} \geq N, \tilde{M} \geq M, \tilde{n} \geq n$ by computing simple statistics from a dataset of impressions and conversions, as detailed next. For $\tilde{N}$, we compute a percentile of the distribution of number of unique conversion sites in each device-epoch, and multiply by the maximum attribution window length (2 in Criteo). This gives a crude upper bound on the maximum number of times most devices are queried, which is itself an upper bound on the maximum number of times most devices are queried *with non-zero loss*. For $\tilde{M}$, we take a percentile of the number of unique impressions sites across device-epochs. For $\tilde{n}$, we take a percentile of the number of unique conversion sites across impression sites and device-epochs, and multiply by the maximum attribution window length.

Percentile	$\tilde{N}$	$\tilde{M}$	$\tilde{n}$	ϵ_{global}	$\epsilon_{\text{imp-quota}}$
50	2	1	2	2	2
90	4	2	4	8	4
95	4	2	4	8	4
99	6	3	6	18	6
100	12	7	14	98	14

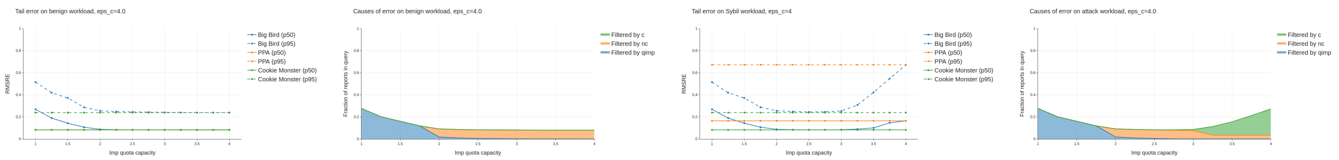
Tab. 2. Percentile values for $\tilde{N}, \tilde{M}$, and $\tilde{n}$.

Finally, we take $\epsilon_{\text{conv-quota}} = \epsilon_{\text{per-site}}$, $\epsilon_{\text{imp-quota}} = \tilde{n}\epsilon_{\text{per-site}}$ and $\epsilon_{\text{global}} = \max(\tilde{N}, \tilde{n} \cdot \tilde{M})\epsilon_{\text{per-site}}$. For instance, taking values at the 95th percentile gives $\epsilon_{\text{global}} = 8$, $\epsilon_{\text{imp-quota}} = 4$, $\epsilon_{\text{conv-quota}} = 1$.

6.3 Query errors under normal workload (Q2)

We fix $\epsilon_{\text{global}} = 4$, a value that is large enough to accommodate all the queries without triggering the global filter filter. Pierre: TODO: rerun experiments with $\epsilon_{\text{global}} = 8$ or whatever we get from Q1. We vary $\epsilon_{\text{imp-quota}}$ and measure its impact on query error. Fig. 3a reports both median and 95th percentile RMSRE. Since Cookie Monster and PPA lack a impression-site quota filter, their errors remain constant across $\epsilon_{\text{imp-quota}}$ values. Moreover, Cookie Monster and PPA exhibit identical error, as we set ϵ_{global} high enough to deactivate PPA's global filter filter—effectively reducing it to Cookie Monster. In contrast, Gordon 's error increases at low $\epsilon_{\text{imp-quota}}$, as the impression-site quota filter blocks some reports. For $\epsilon_{\text{imp-quota}} \geq 2$, the filter no longer affects benign-query error, suggesting that reasonably-configured quotas can preserve utility.

Fig. 3b attributes the error to each filter type, for Gordon. Roxana: I had initially misunderstood the y axis. Explain this graph and the process of attributing error here, not in methodology, as people lose track. Mention just extremely briefly that errors in queries can come from multiple sources in Gordon: DP itself, but also filters and quotas. Remind them how quotas/filters OOB add bias into the query results (the may not know this as we never reminded them of bias etc.! Don't say too much, just find a very simple, intuitive way



(a) Error, normal workload **(b)** Error causes, normal workload **(c)** Error under attack **(d)** Error causes under attack

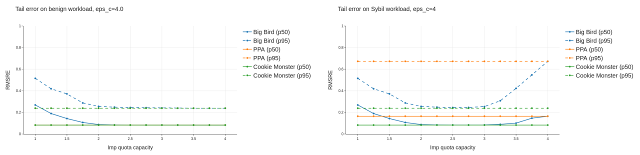
Fig. 3. Online quota system evaluation. (a), (b): Query error and its root causes in a benign case. (c), (d): Benign-query error and root causes under attack. Roxana: Changes to graphs: (1) Move the legends inside the graphs (there’s space on top given the [0,1] y axis (which you should keep!). You can order the legend entries 2x2 to fit atop each graph. (2) Enlarge all fonts and line widths and point sizes significantly so they become visible when areas are squeezed. (3) Remove graph titles. (4) Turn everything into black-and-white – no color differentiations pls! You can differentiate based on point types (consistent across the colors hence system types). For area graphs, you can use different textures in place of colors. (5) Make the PPA points much larger than the Cookie Monster points so you can see them from behind CM. (6) Rename Big Bird to Gordon. (7) Label Cookie Monster as “Cookie Monster $\epsilon_{\text{global}} = \infty$”, PPA as “PPA $\epsilon_{\text{global}} = 4$, Gordon as “Gordon $\epsilon_{\text{global}} = 4$”. (8) x axis label: “Impression site quota capacity ($\epsilon_{\text{imp-quota}}$). x axis label: RMSRE becomes “RMSRE (error, lower is better)”; “Fraction of reports in query” becomes “% of reports in query impacted by a filter.” For (b) and (d), legend should say: “per-site filter, “global filter,” “quota filter”. (9) Please transform y axis for (b) and (d) to 0-100%, to not confuse it with the other 0-1 RMSRE axes. Pierre: Also probably combine all the 4 graphs as a single pdf on the Python side, to share captions etc.

of explaining it in one short sentence). Say that the graph is showing attribution of errors that come from the filters or quotas. Pierre: Add Cookie Monster and PPA for reference as numbers in the text? They only have per-site filter reports, the same as Gordon. For $\epsilon_{\text{imp-quota}} = 1$, almost a third of reports in each query are filtered out by the impression-site quota filter. This explains the high error seen at low values of $\epsilon_{\text{imp-quota}}$ in Fig. 3a. For $\epsilon_{\text{imp-quota}} = 2$ and higher, almost all the biased reports come from per-site filter filters. These filters also exist in Cookie Monster and PPA, which explains why Gordon’s error converges to Cookie Monster’s error in Fig. 3a.

6.4 Query errors under DoS attack (Q3)

We augment the benign scenario with a DoS depletion attack akin to Attacks 2 and 3 (since our dataset does not permit cross-advertiser queries for more than one intermediary site). First, we create a malicious impression site X, by duplicating the impression site X^* that had the most impressions in Criteo. This ensures that a non-negligible fraction of the devices visit X at some point, thus enabling the attack to have visible effect on query accuracy. Next, we create new malicious conversion sites $Y_1^1, \ldots, Y_1^s, \ldots, Y_k^1, \ldots, Y_k^s$ by duplicating s times each of the $k = 10$ conversion sites $Y_1^*, \ldots, Y_k^*$ with most conversions attributed to X^* in Criteo. Finally, we create impressions for X by copying all the impressions X^* has for $Y_1^*, \ldots, Y_s^*$, and we create conversions for Y_i^j by copying all the conversions that Y_i^* has for X. $s = 1$ corresponds to Attack 3, since conversions happen over time, following the same pattern as the benign conversions $Y_1^*, \ldots, Y_k^*$ have for X^*. A larger value for s corresponds to Attack 2, since we have a chain of s conversion sites $Y_i^1, \ldots, Y_i^s$ (using automatic redirection or tricking the user into clicking) instead of a single conversion site Y_i^*.

Fig. 3c shows that impression-site quota limits the impact X has on the c-filter, for an attack with $s = 10$. We do not include X’s queries in the error. For a well-configured impression-site quota filter, honest queriers are perfectly isolated from X, and are still able to run their own queries without being perturbed by the impression-site quota.



(a) Error, normal workload **(b)** Error under attack

Fig. 4. Batched system evaluation. Roxana: BOGUS FIGURES, DO NOT LOOK.

Fig. 3d further drills down into the causes for high error at both ends of the graph in Fig. 3c. For low $\epsilon_{\text{imp-quota}}$, the error comes from the impression-site quota filter, as in Fig. 3b. The attack has no direct impact on error, but impression-site quota is hurting even honest queries. For high $\epsilon_{\text{imp-quota}}$, the error comes from the global filter filter. This is because the impression-site quota filter is too loose and lets X deplete the global filter filter, thereby denying reports for honest queriers.

6.5 Batched system evaluation (Q4)

7 Related Work

8 Conclusions

References

[LPT+21] Tao Luo, Mingen Pan, Pierre Tholoniat, Asaf Cidon, Roxana Geambasu, and Mathias Lécuyer. "Privacy Budget Scheduling". In: *15th USENIX Symposium on Operating Systems Design and Implementation (OSDI 21)*. USENIX Association, July 2021, pp. 55–74. ISBN: 978-1-939133-22-9. URL: https://www.usenix.org/conference/osdi21/presentation/luo.

[TKM+24] Pierre Tholoniat, Kelly Kostopoulou, Peter McNeely, Prabhpreet Singh Sodhi, Anirudh Varanasi, Benjamin Case, Asaf Cidon, Roxana Geambasu, and Mathias Lécuyer. "Cookie Monster: Efficient On-Device Budgeting for Differentially-Private Ad-Measurement Systems". In: *Proceedings of the ACM SIGOPS 30th Symposium on Operating Systems Principles*. SOSP '24. New York, NY, USA: Association for Computing Machinery, Nov. 15, 2024, pp. 693–708. ISBN: 9798400712517. DOI: 10.1145/3694715.3695965.

A Formal analysis

A.1 Model with Single-Beneficiary Reports

We start from the same model as Cookie Monster, and then refine it by adding constraints on the data and the queries. §A.1 makes the following assumptions, that we relax in §A.2.

- The set of public events for a beneficiary site b is the set of all conversions where b appears as beneficiary site. In other words, we hardcode $P = C_b$. This prevents publisher reports or certain adtech optimization queries, that were technically supported by Cookie Monster. [1]
- Each report has a single beneficiary site: either the conversion site (measurement report) or another site (optimization report). In this formalism, a conversion site can register the same conversion data multiple times with different beneficiary sites if it wants to execute both measurement and optimizations.

This section is organized as follows:

- §A.1.1-A.1.3: Preliminaries and notations. Alg. 1-2 are how on-device algorithms of Gordon are set-up and run.
- §A.1.4: Main privacy guarantee theorem, which is based on the isolation and utility guarantees proved in §A.1.5 and the privacy loss calculations on the epoch level and on the epoch-site level (proved in §A.1.6), respectively.
- §A.1.5: Proofs for the isolation and utility guarantees formulated in §4.3 called "Resilience to DoS Depletion". This section relies on the 2-PC guarantee of on-device algorithms implemented in Alg. 3.
- §A.1.6: Proofs that justify the privacy budget losses on the epoch level (for per-site filter, global filter, and `conv-quota` filters) and on the epoch-site level (for `imp -quota` filters) which are used by Alg. 2 to compute ϵ_x^t and $\epsilon_x^{\mathrm{i},t}$ at time step t, respectively. The epoch level and epoch-site level privacy loss computations are specified in Alg. 4 and Alg. 5, respectively.

A.1.1 Sites and roles.

We use a different terminology compared to Cookie Monster, to better align with PPA. Moreover, we make sites appear explicitly in the data and query model. We take a set of sites $\mathcal{S}$ (*e.g.,* domain name). The same site can appear under different roles:

- impression site: site where an impression occurs (publisher in Cookie Monster)
- conversion site: site where a conversion occurs (advertiser in Cookie Monster)
- beneficiary site: site that receives the results of a DP query (querier in Cookie Monster)

A.1.2 Data model.

A database D is a set of device-epoch records where each record $x = (d, e, F) \in \mathcal{X} = \mathcal{D} \times \mathcal{E} \times \mathcal{P}(\mathcal{S} \times \mathcal{I} \cup \mathcal{S} \times \mathcal{S} \times C)$ contains a device d, an epoch e and a set of impression and conversion events F. Each event $f \in F$ contains the site (impression site i or conversion site c) where the event occurred: $f = (i, \mathrm{imp}) \in \mathcal{S} \times \mathcal{I}$ or $f = (c, b, \mathrm{conv}) \in \mathcal{S} \times \mathcal{S} \times C$. Additionally, conversions contain the beneficiary site b that will receive the conversion report. [2]

Definition 1 (Filter ($\mathcal{F}_x$))**.** *For each device-epoch record $x = (d, e, F)$, we maintain several types of privacy filters:*

- $\mathcal{F}_x^{nc[b]}$*: Non-collusion filter for beneficiary site b, with capacity* $\epsilon_{nc[b]}$
- $\mathcal{F}_x^{c}$*: Collusion filter with capacity* ϵ_c*.*
- $\mathcal{F}_x^{q\text{-}conv[c]}$*: Conversion site quota filter for site c, with capacity* $\epsilon_{conv\text{-}quota}$
- $\mathcal{F}_x^{q\text{-}imp[i]}$*: Impression site quota filter for site i, with capacity* $\epsilon_{imp\text{-}quota}$

Each filter maintains a state of consumed budget and provides operations:

- canConsume(ϵ)*: Returns TRUE if the filter can accommodate additional privacy loss ϵ. In particular, letting $\mathbb{I}_{pass}$ be how it's defined in lemma 1,*
  - *For a per-site filter, global filter, or* `conv-quota` *filter, return TRUE iff it satisfies,*

    $$\epsilon_x^t \leq \epsilon_{\mathrm{initial}} - \sum_{k \in [t-1]} \epsilon_x^k \cdot \mathbb{I}_{\mathrm{pass}}[k], \tag{1}$$

    where $\epsilon_{\mathrm{initial}}$ is the respective initialized privacy budget of the global filter and `conv-quota` *filters in function 8, and ϵ_x^t is the privacy loss at step k from the corresponding filter (epoch-level budget consumption in Alg. 4).*
  - *For a* `imp-quota` *filter, return TRUE iff it satisfies*

    $$\epsilon_x^{\mathrm{i},t}[i] \leq \epsilon_{imp\text{-}quota} - \sum_{k \in [t-1]} \epsilon_x^{\mathrm{i},k}[i] \cdot \mathbb{I}_{pass}[k], \tag{2}$$

    where $\epsilon_x^{\mathrm{i},t}[i]$ is the site-level privacy loss at step k from the `imp-quota` *filter (site-level budget consumption in Alg. 5).*
- tryConsume(ϵ)*: Deducts privacy loss ϵ from the filter's remaining capacity iff* canConsume *on all relevant filters return TRUE, and the amount consumed is by basic compositions of pure DP filters.*

A.1.3 Query model.

Definition 2 (Attribution function, adapted from Cookie Monster)**.** *Fix a set of relevant impression sites $\mathbf{i}_A \subset \mathcal{S}$ and a set of* impressions [3] *relevant to the query $F_A \subset \mathbf{i}_A \times \mathcal{I}$. Fix $k, m \in \mathbb{N}^*$ where k is a number of epochs. An* attribution function *is a function $A : \mathcal{P}(\mathcal{I})^k \to \mathbb{R}^m$ that takes k event sets $F_1, \ldots, F_k$ from k epochs and outputs an m-dimensional vector $A(F_1, \ldots, F_k)$, such that only* relevant events *contribute to A. That is, for all $(F_1, \ldots, F_k) \in \mathcal{P}(\mathcal{I})^k$, we have:*

$$A(F_1, \ldots, F_k) = A(F_1 \cap F_A, \ldots, F_k \cap F_A). \tag{3}$$

Definition 3 (Report identifier and attribution report, same as Cookie Monster)**.** *Fix a domain of report identifiers $\mathbb{Z}$. Consider a mapping $d(\cdot)$ from report identifiers R to devices $\mathcal{D}$ that gives the device d_r that generated a report r.* [4]

Given an attribution function A, a set of epochs E and a report identifier $r \in \mathbb{Z}$, the attribution report *$\rho_{r,A,E}$, or ρ_r for short, is a function over the whole database D defined by:*

$$\rho_r : D \in \mathbb{D} \mapsto A(D^E_{d_r}). \tag{4}$$

Definition 4 (Query, same as Cookie Monster)**.** *Consider a set of report identifiers $R \subset \mathbb{Z}$, and a set of attribution reports $(\rho_r)_{r\in R}$ each with output in $\mathbb{R}^m$.*[5] *The* query *for $(\rho_r)_{r\in R}$ is the function $Q : \mathbb{D} \to \mathbb{R}^m$ is defined as $Q(D) := \sum_{r\in R} \rho_r(D)$ for $D \in \mathbb{D}$.*

Algorithm 1 Gordon Notations and Setup

```
Input
  Database $D$ // Fixed for simplicity here, could be adaptive.
  Stream of adaptively chosen queries
function $\mathcal{M}(D)$
  $(S_b)_{b\in\mathcal{S}} = (\emptyset)_{b\in\mathcal{S}}$
  for $(d, e, F) \in D$ do
    for $f \in F : f = (c, b, \text{conv})$ do
      Generate report identifier $r \xleftarrow{\$} U(\mathbb{Z})$
      // Save mapping from r to the device that generated it
      $d_r \leftarrow d$
      $S_b \leftarrow S_b \cup \{(r, f)\}$
  // Each beneficiary receives its public events and corresponding report identifiers
  for $b \in \mathcal{S}$ do
    output $S_b$ to $b$
  // Beneficiaries ask queries interactively
  for $t \in [t_{\max}]$ do
    receive $Q_t^{b_t}$ from beneficiary site $b_t$.
    output AnswerQuery($Q_t^{b_t}$) to $b_t$
// Collect, aggregate and noise reports to answer Q
function AnswerQuery(Q)
  Get report identifiers $R$ and noise parameter $\sigma$ from $Q$
  for $r \in R$ do
    Read $Q$ to get conversion site $c$, beneficiary site $b$, impression sites $\mathbf{i}$, target epochs $E$, attribution function $A$ for report $r$.
    $\rho_r \leftarrow$ GenerateReport$(d, c, b, \mathbf{i}, E, A, \sigma)$
  Sample $X \sim \mathcal{L}(\sigma)$
  return $\sum_{r\in R} \rho_r + X$
```

Algorithm 2 Gordon On-Device Algorithm

```
Input
  Per-site budget $\epsilon_{\text{per-site}}$
  Normal number of conversion sites $N$
  Normal number of impression sites $M$
  Fraction of budget for optimization queries $r$
  EpochBudget (Alg. 4) and EpochImpSiteBudget (Alg. 5) subroutines
// Query at time t
function Capacities($\epsilon_{\text{per-site}}, N, M, r$)[6]
  $\epsilon_{\text{global}} \leftarrow N(1 + r)\epsilon_{\text{per-site}}$ Pierre: Update to add $n$
  $\epsilon_{\text{conv-quota}} \leftarrow \epsilon_{\text{global}}/N$
  $\epsilon_{\text{imp-quota}} \leftarrow \epsilon_{\text{global}}/M$
  return $\epsilon_{\text{global}}, \epsilon_{\text{imp-quota}}, \epsilon_{\text{conv-quota}}$
// Generate report and update on-device budget
function GenerateReport($d, c, b, \mathbf{i}, E, A, \sigma$)
  for $e \in E$ do
    $x \leftarrow (d, e, D^e_d)$
    if $\mathcal{F}_x$ is not defined then
      $\epsilon_{\text{global}}, \epsilon_{\text{imp-quota}}, \epsilon_{\text{conv-quota}} \leftarrow$ Capacities$(\epsilon_{\text{per-site}}, N, M, r)$
      $\mathcal{F}_x \leftarrow$ InitializeFilters$(\epsilon_{\text{global}}, \epsilon_{\text{per-site}}, \epsilon_{\text{imp-quota}}, \epsilon_{\text{conv-quota}})$
    $F_e \leftarrow D^e_d$
    // Compute individual device-epoch-level privacy losses, same as
    $\epsilon^t_x \leftarrow$ EpochBudget$(x, d, E, A, \mathcal{L}, \sigma)$
    $\epsilon^{\mathbf{i},t}_{\mathbf{x}} \leftarrow \{\}$ // Initialize empty map
    for $i \in \mathbf{i}$ do
      // Compute individual device-epoch-site-level privacy losses, Alg. 5
      $\epsilon^i_x \leftarrow$ EpochImpSiteBudget$(x, i, d, E, A, \mathcal{L}, \sigma)$
      $\epsilon^{\mathbf{i},t}_{\mathbf{x}}[i] \leftarrow \epsilon^i_x$
    // Atomic filter check and update, Alg. 3
    if AtomicFilterCheckAndConsume$(\mathcal{F}_x, b, c, \mathbf{i}, \epsilon^t_x, \epsilon^{\mathbf{i},t}_{\mathbf{x}})$ = FALSE then
      $F_e \leftarrow \emptyset$ // Empty the report if any filter check fails
  $\rho \leftarrow A((F_e)_{e\in E})$ // Clipped attribution report
  return $\rho$
```

A.1.4 Privacy guarantees.

Mechanisms. Alg. 1 defines two types of interactive mechanisms. First, for each beneficiary site b we can denote by $\mathcal{M}^b$ the interactive mechanism that only interacts with b. Second, $\mathcal{M}$ is the interactive mechanism that interacts with all the beneficiary sites concurrently.

Levels of accounting. EpochImpSiteBudget computes privacy loss at a different granularity than EpochBudget, using Def. 5.

Definition 5 (Device-epoch-site neighborhood relation)**.** *Consider a device $d \in \mathcal{D}$, an epoch $e \in \mathcal{E}$, an impression*

Algorithm 3 2-Phase Commit Subroutine

```
Input:
  $\epsilon_x^t$: epoch-level privacy loss for a particular query
  $\epsilon_x^{\mathbf{i},t}$: epoch-site-level privacy loss for a particular query
  canConsume: function as is defined in Def. 1
  tryConsume: function as is defined in Def. 1
Output:
  Boolean function if all filters have enough budget for the privacy loss $\epsilon_x^t$ or not.
function AtomicFilterCheckAndConsume($\mathcal{F}_x, b, c, \mathbf{i}, \epsilon_x^t, \epsilon_x^{\mathbf{i},t}$)
  // Phase 1: Prepare - check if all filters can consume
  if $\mathcal{F}_x^{\text{per-site filter}[b]}$.canConsume($\epsilon_x^t$) = FALSE then
    return FALSE
  if $\mathcal{F}_x^{\text{global filter}}$.canConsume($\epsilon_x^t$) = FALSE then
    return FALSE
  if $\mathcal{F}_x^{\text{conversion-site quota}[c]}$.canConsume($\epsilon_x^t$) = FALSE then
    return FALSE
  for $i \in \mathbf{i}$ do
    if $\mathcal{F}_x^{\text{impression-site quota}[i]}$.canConsume($\epsilon_x^{\mathbf{i},t}[i]$) = FALSE then
      return FALSE
  // Phase 2: Commit - consume from all filters
  // Privacy filters under no collusion
  $\mathcal{F}_x^{\text{per-site filter}[b]}$.tryConsume($\epsilon_x^t$)
  // Privacy filter under collusion
  $\mathcal{F}_x^{\text{global filter}}$.tryConsume($\epsilon_x^t$)
  // Quota filter for conversion site [7]
  $\mathcal{F}_x^{\text{conversion-site quota}[c]}$.tryConsume($\epsilon_x^t$)
  // Privacy filters for impression sites
  for $i \in \mathbf{i}$ do
    // Invididual device-epoch-impression site loss.[8]
    $\mathcal{F}_x^{\text{impression-site quota}[i]}$.tryConsume $\epsilon_x^{\mathbf{i},t}[i]$)
  return TRUE
```

site $i \in \mathcal{S}$ and a set of impression events happening on i: $F_i \in \{i\} \times \mathcal{I}$. [9]

We say $D \sim_{d,e,i,F_i} D'$ if there exists D_0 and $F \in \mathcal{S} \setminus \{i\} \times \mathcal{I}$ such that:

$$\{D, D'\} = \{D_0 + (d, e, F), D_0 + (d, e, F \cup F_i)\} \tag{5}$$

Theorem 6. *Consider $x \in \mathcal{X}$ on device d.*

- *For each beneficiary site $b \in \mathcal{S}$, $\mathcal{M}^b$ satisfies individual device-epoch $\epsilon_{per\text{-}site}(x)$-DP for x under public information C_b.*
- *$\mathcal{M}$ satisfies individual device-epoch $\epsilon_{global}(x)$-DP for x under public information C.* [10]

Proof. Pierre: TODO, might need more formal privacy games.

Pierre: TODO: prove that the atomic filter doesn't break the privacy proofs, because we add a "canConsume" API to the filter to do the 2PC. [11] □

Algorithm 4 Compute epoch-level privacy budget

```
Input:
  $x$: device-epoch record $(d, e, F)$
  $d$: device identifier
  $E$: set of epochs
  $A$: attribution function
  $\mathcal{L}$: parametrized noise distribution
  $\sigma$: noise parameter
Output:
  Returns the epoch-level individual privacy loss for device-epoch $x$
function EpochBudget($x, d, E, A, \mathcal{L}, \sigma$)
  // Extract components from device-epoch record
  $(d', e, F) \leftarrow x$
  $E \leftarrow \{f \in F \| \|A(f)\|_1 > 0\}$
  if $E_{\text{relevant}} = \emptyset$ then
    // Case 1: No relevant events in this epoch
    $\Delta \leftarrow 0$
  if $|E| = 1$ then
    // Case 2: Single epoch query, then individual sensitivity equals the L1 individual sensitivity of the one event
    $\Delta \leftarrow \max_{D \in \mathbb{D}}(\|A(D) - A(D + x)\|_1)$
  else
    // Case 3: Multiple epochs query, then individual sensitivity equals report's global sensitivity
    $\Delta \leftarrow \max_{D \in \mathbb{D}, x' \in \chi}(\|A(D) - A(D + x')\|_1)$
  return $\Delta$ / $\sigma$
```

A.1.5 Proofs for Resilience to DoS Depletion (§4.3).

Definition 6. *We define the following notations for privacy loss accounting:*

- *$\epsilon_c^{\leq t}$: the cumulative privacy loss charged to the c-filter up to step t before 2PC is triggered at step t.*
- *$D^{\leq e}$: The subset of database D containing records with epochs up to and including e.*

Lemma 1 (2-phase commit filter guarantees)**.** *For query k, let:*

$$\mathbb{I}_{pass}(k) \triangleq \begin{cases} 1 & \textit{if } \text{AtomicFilterCheckAndConsume } \textit{returns} \\ & \quad \mathsf{TRUE} \textit{ for query } k \\ 0 & \textit{otherwise} \end{cases} \tag{6}$$

The AtomicFilterCheckAndConsume *function in Algorithm 3 guarantees the following properties:*

For any query k processed by AtomicFilterCheckAndConsume*, if $\mathbb{I}_{pass}(k) = 1$, then*

1. ***Epoch-level Consistency Property:** the nc-filter, c-filter, and q-conv-filter all consume exactly the same amount of budget ϵ_x^t for that query.*

Algorithm 5 Compute epoch-site level privacy budget

```
Input:
    $x$ : device-epoch record $(d, e, F)$
    $i$ : impression site
    $d$ : device
    $E$ : set of epochs, from which we can extract set of impression sites in these epochs relevant to $A$
    $A$ : attribution function
    $\mathcal{L}$ : parameterized noise distribution
    $\sigma$ : noise scale
Output:
    Returns the epoch-site-level individual privacy loss for impression site $i$ at device-epoch $x$
    function EpochImpSiteBudget($x, i, d, E, A, \mathcal{L}, \sigma$)
        Extract epoch $e$ from $x = (d, e, F)$
        $\mathbf{i}_x \leftarrow \{i \in S \mid x = (d, e, F) \ \wedge \ (i, \text{imp}) \in F\}$
        // Relevant epoch-site level events for current epoch
        $E_{\text{relevant}} \leftarrow \{f \in F | f = (i, \text{imp}) \ \wedge \ \|A(f)\|_1 > 0\}$
        // Compute epoch-site-level privacy losses
        if $E_{\text{relevant}} = \emptyset$ then
            // Case 2 of equations (58) or (59): No relevant epoch-site events in epoch
            $\Delta \leftarrow 0$
        // Check if we have single or multiple epoch-sites in E
        if $|E| = 1$ AND $|\{\text{site } i \text{ relevant to } A \text{ in epoch } e \in E\}| = 1$ then
            // Case 1 of equation (58): Single epoch, and only one impression site with relevant events to A in this epoch
            $\Delta \leftarrow \max_{D \in \mathbb{D}}(\|A(D) - A(D + \mathbf{i}_x)\|_1)$
            Pierre: In practice we use an upper bound on the individual sensitivity, not the actual max. Same for global sensitivity.
        else
            // Case 1 of equation (59): Request either touches multiple epochs or multiples epoch-sites with relevant events
            $\Delta \leftarrow \max_{D \in \mathbb{D}, x' \in \chi}(\|A(D) - A(D + \mathbf{i}_{x'})\|_1)$

        return $\Delta$ / $\sigma$
```

2. ***Epoch-site-level Consistency Property:*** *the q-imp filter consumes exactly $\epsilon_x^i[i]$, which represents the device-epoch-impressionsite-level individual privacy loss.*

Proof. We can prove both properties at the same time. Fix an arbitrary query k, for which $\mathbb{I}_{\text{pass}}(k) = 1$. From Algorithm 3, we observe that AtomicFilterCheckAndConsume returns TRUE if and only if: (1) all canConsume checks in Phase 1 pass, and (2) All tryConsume operations in Phase 2 are executed. For a query from conversion site c with beneficiary site b and impression sites $\mathbf{i}$, the function calls:

- $\mathcal{F}_x^{\text{per-site filter}[b]}.\text{tryConsume}(\epsilon_x^t)$
- $\mathcal{F}_x^{\text{global filter}}.\text{tryConsume}(\epsilon_x^t)$
- $\mathcal{F}_x^{\text{conversion-site quota}[c]}.\text{tryConsume}(\epsilon_x^t)$
- For each $i \in \mathbf{i}$: $\mathcal{F}_x^{\text{impression-site quota}[i]}.\text{tryConsume}(\epsilon_{\mathbf{x}}^{\mathbf{i},t}[i])$

Note that ϵ_x^t is computed once at EpochBudget in Line 6 of Algorithm 2 and represents the device-epoch-level individual privacy loss. Similarly, each $\epsilon_{\mathbf{x}}^{\mathbf{i},t}[i]$ is computed once via EpochImpSiteBudget and represents the device-epoch-impressionSite-level individual privacy loss. Therefore, when $\mathbb{I}_{\text{pass}}(k) = 1$, the nc-filter, c-filter, and q-conv filter all consume exactly the same amount ϵ_x^t, while each q-imp filter consumes its specific amount $\epsilon_{\mathbf{x}}^{\mathbf{i},t}[i]$, which is proportional to its sensitivity at the impression site level. □

Consider an execution of Alg. 1. A report at time step k concerns with *one* conversion site c_k, and some subset of impression sites $\mathbf{i}_k \subseteq S$. The union of attackers, in particular, can control an arbitrary subset of conversion sites $\mathbf{bad_c} \subseteq S$, which may or may not contain c_k at a given k. We denote by N^{adv} the size of $|\mathbf{bad_c}|$ over the entire lifetime. Similarly, the adversary can control an arbitrary subset of impression sites $\mathbf{bad_i} \subseteq S$, which may or may not intersect with $\mathbf{i}_k$. We denote by N^{adv} the size of $|\mathbf{bad_i}|$ over the entire lifetime. We let $\mathbf{bad} = \mathbf{bad_c} \cup \mathbf{bad_i}$ and $\mathbf{good} = S \backslash \mathbf{bad}$.

Theorem 1 (global filter isolation properties). *Consider an adversary who manages to create M^{adv} and N^{adv} `imp-quota` and `conv-quota` filters, respectively. The maximum budget ϵ_{global}^{adv} that the adversary can consume from the global filter on a device d is such that:*

$$\epsilon_{global}^{adv} \leq \min(M^{adv}\epsilon_{imp\text{-}quota},\ N^{adv}\epsilon_{conv\text{-}quota}).$$

To show that an adversary can consume $\epsilon_{\text{global}}^{\text{adv}}$ from the global filter as claimed in theorem 1, we show that the adversary consumes at most $\epsilon_{\text{global}}^{\text{adv}}$ privacy budget from global filter at any time t where it controls at most `imp-quota` and `conv-quota` filters, respectively.

In particular, at step t in Line 16, suppose that beneficiary site b requests a report $\rho_{r,E,A}$ with noise σ through conversion site c for impression sites $\mathbf{i}$. Consider a device-epoch x, with individual budget ϵ_x^t computed at Line 6 in Alg. 2. Denote by $N_{\text{adv}}^{\leq t}$ the number of conversion sites in $\mathbf{bad_c}$ with respect to x that were queried with non-zero budget by step t. Denote by $M_{\text{adv}}^{\leq t}$ the number of impression sites in $\mathbf{bad_i}$ with respect to x that were queried with non-zero budget by step t.

Lemma 2. *At the end of time t, given that the adversary has created at most M^{adv} and N^{adv} `imp-quota` and `conv-quota` filters, respectively, we have the following upper bounds on the global filter filter budget that the adversary can consume:*

- *The adversary consumes at most $M_{\text{adv}}^{\leq t}\epsilon_{imp\text{-}quota} \leq M^{\text{adv}}\epsilon_{imp\text{-}quota}$ budget from the global filter.*
- *The adversary consumes at most $N_{\text{adv}}^{\leq t}\epsilon_{conv\text{-}quota} \leq N^{\text{adv}}\epsilon_{conv\text{-}quota}$ budget from the global filter.*

We first use this lemma to prove theorem 1.

Proof (theorem 1). M^{adv} and N^{adv} respectively gives an upper bound on the global filter, so the smaller one of the two is the tighter upper bound on how much the adversary consumes from the global filter. That means the $\epsilon^{\text{adv}}_{\text{global}}$ amount is precisely

$$\epsilon^{\text{adv}}_{\text{global}} \le \min(M^{\text{adv}}\epsilon_{\text{imp-quota}},\ N^{\text{adv}}\epsilon_{\text{conv-quota}}).$$

□

Next, we prove the two upper bounds in lemma 2 for $\epsilon^{\text{adv}}_{\text{global}}$ based on M^{adv} and N^{adv}, respectively.

Proof (first upper bound by N^{adv}). The total privacy loss in the global filter incurred by the attackers for the c-filter by step t, not inclusive, is:

$$\epsilon^{\mathbf{bad}}_{\text{used}} = \epsilon^{\le t-1}_{c,\mathbf{bad}} = \sum_{k=[t-1]:c_k \in \mathbf{bad_c}} \epsilon^k_c \cdot \mathbb{I}_{pass}(k). \tag{7}$$

By basic composition under a pure DP filter and the definition of individual privacy loss, the total privacy loss equals:

$$= \sum_{c \in \mathbf{bad_c}} \sum_{k<t:c_k=c} \Delta_x \rho^k_{b_k} \cdot \mathbb{I}_{\text{pass}}(k), \tag{8}$$

where $\Delta_x \rho^k_{b_k}$ represents how much the attribution report generated for query k from beneficiary b_k can change with respect to x.

By the consistency property of lemma 1, Alg. 3 ensures that privacy loss is only incurred on k where $\mathbb{I}_{\text{pass}}(k) = 1$. So, for each conversion site c, the filter q-conv$[c]$ precisely tracks the privacy losses incurred, so

$$\epsilon_{\text{conv-quota}}[c]^{\le t-1} = \sum_{k<t:c_k=c} \Delta_x \rho^k_{b_k} \cdot \mathbb{I}_{\text{pass}}(k). \tag{9}$$

Substitute this equality into equation (8), we get:

$$\epsilon^{\le t-1}_{c,\mathbf{bad}} = \sum_{c \in \mathbf{bad_c}} \epsilon_{\text{conv-quota}}[c]^{\le t-1}. \tag{10}$$

This sum can be restricted to conversion sites with non-zero privacy loss, i.e.:

$$= \sum_{c \in \mathbf{bad_c}:\epsilon_{\text{q-conv}}[c]^{\le t-1}>0} \epsilon_{\text{conv-quota}}[c]^{\le t-1} \tag{11}$$

$$\le \sum_{c \in \mathbf{bad_c}:\epsilon_{\text{q-conv}}[c]^{\le t-1}>0} \epsilon_{\text{conv-quota}}, \tag{12}$$

where $\epsilon_{\text{q-conv}}$ is the capacity of each q-conv filter. It follows that the number of conversion sites with non-zero privacy loss is precisely $N^{\le t}_{\text{adv}}$, so:

$$\le \|\{c \in \mathbf{bad_c} : \epsilon_{\text{conv-quota}}[c]^{\le t-1} > 0\}\| \cdot \epsilon_{\text{conv-quota}} \tag{13}$$

$$= N^{\le t-1}_{\text{adv}} \cdot \epsilon_{\text{conv-quota}}. \tag{14}$$

Now, during the 2-PC for time t, we have the following cases:

- Suppose ϵ^t_x is a reasonable value, in the sense that it's bounded by the capacity $\epsilon_{\text{conv-quota}}$. Then,

$$\epsilon^{\le t}_{c,\mathbf{bad}} = \epsilon^{\mathbf{bad}}_{\text{used}} + \epsilon^t_x \le N^{\le t}_{\text{adv}} \cdot \epsilon_{\text{conv-quota}}. \tag{15}$$

- Otherwise, ϵ^t_x is unreasonable, in which case ϵ^t_x exceeds the capacity $\epsilon_{\text{conv-quota}}$. In this case,

$$\epsilon_{\text{conv-quota}}[c_t]^{\le t-1} + \epsilon^t_x \ge \epsilon_{\text{conv-quota}}, \tag{16}$$

causing $\mathcal{F}^{\text{q-conv[c]}}_x$.canConsume$(\epsilon^t_x)$ to return FALSE by definition, so no budget is spent at all. In such a case,

$$\epsilon^{\le t}_{c,\mathbf{bad}} = \epsilon^{\mathbf{bad}}_{\text{used}} + 0 = \epsilon^{\mathbf{bad}}_{\text{used}} \le N^{\le t-1}_{\text{adv}} \cdot \epsilon_{\text{conv-quota}}, \tag{17}$$

by equation (14).

Since the adversary has created at most N^{adv} by the end of time t, it must be the case that $N^{\le t-1}_{\text{adv}} \le N^{\le t}_{\text{adv}} \le N^{\text{adv}}$. This means that, in either case, the attackers can consume at most $N^{\le t}_{\text{adv}}\epsilon_{\text{conv-quota}} \le N^{\text{adv}}\epsilon_{\text{conv-quota}}$ of the global filter budget by the end of time t, as desired □

Proof (second upper bound by M^{adv}). By basic composition under a pure DP filter, we know ϵ_c is by definition the sum of c-filter consumption at each time up to the end of time $t-1$:

$$\epsilon^{\le t-1}_{c,\mathbf{bad}} = \sum_{k\in[t-1]:c_k \in \mathbf{bad_c}} \epsilon^k_x \cdot \mathbb{I}_{\text{pass}}(k) \tag{18}$$

$$= \sum_{k\in[t-1]:c_k \in \mathbf{bad_c}} \Delta_x \rho^k_{b_k} \cdot \mathbb{I}_{\text{pass}}(k), \tag{19}$$

where $\Delta_x \rho^k_{b_k}$ is how much the attribution report generated for query k from beneficiary b_k can change with respect to x, which we substitute next. Let $\vec{x} = (x_{i_1}, \ldots, x_{i_m})$ be the vector of $|\mathbf{i}_k| = m$ device-epoch-sites, where $x_i = (d, e, F_i)$ has all its events on site i. Then, we let the corresponding neighboring dataset of $D \in \mathbb{D}$ be $D + \vec{x}$, so that:

$$\rho(D + \vec{x}) - \rho(D) = \sum_{j=1}^{m} \rho(D + x_{i_j}) - \rho(D + x_{i_{j-1}}), \tag{20}$$

where $x_{i_0} = 0$. Thus, we substitute by the following decomposition of a query's sensitivity across the impression sites relevant to that query, by how it is assigned in Alg. 2 Line 6:

$$\Delta_x \rho^k_{b_k} = \max_{D,D' \in \mathcal{D}'=D+x} \|\rho^k_{b_k}(D') - \rho^k_{b_k}(D)\|_1 \tag{21}$$

$$= \max_{D \in \mathcal{D}} \|\rho^k_{b_k}(D+x) - \rho^k_{b_k}(D)\|_1 \tag{22}$$

$$\leq \max_{D \in \mathcal{D}} \sum_{j \in [m] : i_j \in \mathbf{bad_i}} \|\rho^k_{b_k}(D + x_1 + \ldots + x_i) - \tag{23}$$

$$\rho^k_{b_k}(D + x_1 + \ldots + x_{i-1})\|_1 \tag{24}$$

$$\leq \sum_{j \in [m] : i_j \in \mathbf{bad_i}} \max_{\hat{D} \in \mathcal{D}} \|\rho^k_{b_k}(\hat{D} + x_{i_j}) - \rho^k_{b_k}(\hat{D})\|_1 \tag{25}$$

$$= \sum_{j \in [m] : i_j \in \mathbf{bad_i}} \Delta^{i_j}_x \rho^k_{b_k} = \sum_{i \in \mathbf{bad_i}} \Delta^i_x \rho^k_{b_k} \tag{26}$$

$$= \sum_{i \in \mathbf{bad_i}} \epsilon^{\mathbf{i},k}_{\mathbf{x}}[i], \tag{27}$$

Plugging in this, we get

$$\epsilon^{\leq t-1}_{c,\mathbf{bad}} \leq \sum_{k \in [t-1] : c_k \in \mathbf{bad_c}} \sum_{i \in \mathbf{bad_i}} \epsilon^{\mathbf{i},k}_{\mathbf{x}}[i] \cdot \mathbb{I}_{\text{pass}}(k) \tag{28}$$

$$= \sum_{i \in \mathbf{bad_i}} \sum_{k \in [t-1] : c_k \in \mathbf{bad_c}, i \in \mathbf{i}_k} \epsilon^{\mathbf{i},k}_{\mathbf{x}}[i] \cdot \mathbb{I}_{\text{pass}}(k), \tag{29}$$

$$\leq \sum_{i \in \mathbf{bad_i}} \sum_{k \in [t-1] : i \in \mathbf{i}_k} \epsilon^{\mathbf{i},k}_{\mathbf{x}}[i] \cdot \mathbb{I}_{\text{pass}}(k), \tag{30}$$

by changing order of summation, and the last inequality by relaxing the "$c_k \in \mathbf{bad_c}$" condition. But note that

$$\epsilon^{\leq t-1}_{\text{imp-quota}}[i] = \sum_{k \in [t-1] : i \in \mathbf{i}_k} \epsilon^{\mathbf{i},k}_{\mathbf{x}}[i] \cdot \mathbb{I}_{\text{pass}}(k), \tag{31}$$

because, by epoch-site-level consistency property in lemma 1, we know that only relevant site i at time k, where every filter has enough budget to pass the 2-PC check, will have epoch-site level privacy losses incurred. Substituting this equality into equation (30), we get:

$$\epsilon^{\leq t-1}_{c,\mathbf{bad}} \leq \sum_{i \in \mathbf{bad_i}} \epsilon^{\leq t-1}_{\text{imp-quota}}[i] \tag{32}$$

$$= \sum_{i \in \mathbf{bad_i} : \epsilon^{\leq t-1}_{\text{imp-quota}}[i] > 0} \epsilon^{\leq t-1}_{\text{imp-quota}}[i] \tag{33}$$

$$\leq \sum_{i \in \mathbf{bad_i} : \epsilon^{\leq t-1}_{\text{imp-quota}}[i] > 0} \epsilon_{\text{imp-quota}} \tag{34}$$

$$= \left|\left\{ i \in \mathbf{bad_i} : \epsilon^{\leq t-1}_{\text{imp-quota}}[i] > 0 \right\}\right| \cdot \epsilon_{\text{imp-quota}}, \tag{35}$$

because only non-zero privacy losses that were incurred contribute meaningfully to the composition. Finally, we note that $\left|\left\{ i \in \mathbf{bad_i} : \epsilon^{\leq t-1}_{\text{imp-quota}}[i] > 0 \right\}\right| \leq M^{\leq t-1}$ by definition and:

$$\epsilon^{\leq t-1}_{c,\mathbf{bad}} \leq M^{\leq t-1}_{\text{adv}} \cdot \epsilon_{\text{imp-quota}}. \tag{36}$$

Following this result, similar to the proof for part 1:

- Suppose $\epsilon^t_x \leq \epsilon_{\text{imp-quota}}$,
  $$\epsilon^{\leq t}_{c,\mathbf{bad}} \leq M^{\leq t}_{\text{adv}} \cdot \epsilon_{\text{imp-quota}}. \tag{37}$$
- Else, $\epsilon^t_x > \epsilon_{\text{imp-quota}}$, then $\epsilon_{\text{imp-quota}}$ will be exceeded, causing canConsume to return FALSE, so,
  $$\epsilon^{\leq t}_{c,\mathbf{bad}} = \epsilon^{\leq t-1}_{c,\mathbf{bad}} + 0 = \epsilon^{\leq t-1}_{c,\mathbf{bad}} \leq M^{\leq t-1}_{\text{adv}} \cdot \epsilon_{\text{imp-quota}}, \tag{38}$$
  by equation (36).

Since by the end of time t, the adversary has created at most M^{adv} `imp-quota` filters, we know $M^{\leq t-1}_{\text{adv}} \leq M^{\leq t}_{\text{adv}} \leq M^{\text{adv}}$, which means that in both cases we have:

$$\epsilon^{\leq t}_{c,\mathbf{bad}} \leq M^{\leq t}_{\text{adv}} \epsilon_{\text{imp-quota}} \leq M^{\text{adv}} \epsilon_{\text{imp-quota}}, \tag{39}$$

as desired. □

Corollary 7. *Suppose in any t time range before and after which all filters receive refreshed privacy budgets, the union of attackers is expected to control at most $N^{\leq t}_{\text{adv}}$ conversion sites and $M^{\leq t}_{\text{adv}}$ impression sites with high probability, and the workload of benign users is expected to be $N^{\leq t}_{x,\mathbf{good}}$ conversion sites and $M^{\leq t}_{x,\mathbf{good}}$ impression sites.*

Then, setting $N = N^{\leq t}_{\text{adv}} + N^{\leq t}_{x,\mathbf{good}}$ and $M = M^{\leq t}_{\text{adv}} + M^{\leq t}_{x,\mathbf{good}}$, and setting $\epsilon_{\textit{conv-quota}}$ to ϵ_c/N and setting $\epsilon_{\textit{imp-quota}}$ to ϵ_c/M guarantee benign users to have sufficient privacy global filter budgets for their workloads with high probability.

Proof. Same analysis as theorem 1 by plugging in the new total values of conversion sites and impression sites that receive queries with non-zero sensitivities, respectively. □

Analysis in the proof for the second upper bound of lemma 2 can be further optimized when we restrict the class of attributions. We first define a class of histogram attribution based on definition 8:

Definition 7 (Single-touch attribution function)**.** *An attribution function A is considered single-touch if we attribute to the entire conversion to exactly one impression event $f \in (F_1 \cap F_A) \cup \cdots \cup (F_k \cap F_A)$, where the selection criterion of that one event may be, for example, last touch. As such, we have*

$$a_{\mathbf{F}}(f) = \begin{cases} a_{\mathbf{F}}(f^*) & \text{, if } f = f^* \\ 0 & \text{, otherwise} \end{cases} \tag{40}$$

$$\implies A(\mathbf{F}) = a_{\mathbf{F}}(f^*) \cdot H(f^*), \tag{41}$$

where $H(f^*)$ is the one hot-encoding defined same way as in definition 8.

Corollary 8. [12] *If the attribution function A at every time k up to including t is single-touch, then, the second upper bound is improved to $\epsilon^{\text{adv}}_{\text{global}} \leq \epsilon_{\textit{imp-quota}}$.*

Proof. In the single-touch case, Line 22 would be a max over the multiple impression sites touched on, not a sum, so, instead of using triangular inequality, we have:

$$\Delta_x \rho_{b_k}^k = \max_{D \in \mathcal{D}} \|\rho_{b_k}^k(D+x) - \rho_{b_k}^k(D)\|_1 \tag{42}$$

$$= \max_{i \in \mathbf{bad_i}} \max_{D \in \mathcal{D}} \|\rho_{b_k}^k(D+x) - \rho_{b_k}^k(D)\|_1 \tag{43}$$

$$= \max_{i \in \mathbf{bad_i}} \epsilon_{\mathbf{x}}^{\mathbf{i},k}[i] \tag{44}$$

$$\Longrightarrow \epsilon_{c,\mathbf{bad}}^{\le t-1} \le \max_{i \in \mathbf{bad_i}} \epsilon_{\text{imp-quota}}^{\le t-1}[i] \le \epsilon_{\text{imp-quota}}. \tag{45}$$

This is without regard to t, so by the end of t, if the adversary has created at most M^{adv} many impression sites, $\epsilon_{c,\mathbf{bad}}^{\le t} \le \epsilon_{\text{imp-quota}}$. □

Theorem 9. *Consider the same set-up as theorem 1. Further denote by $\epsilon_{nc}(q)$ the remaining budget for beneficiary b in the nc-filter. Denote by $\epsilon_{conv\text{-}quota}(c)$ the remaining budget for conversion site c in the q-conv filter. Denote by $\epsilon_{imp\text{-}quota}(c)$ the remaining budget for impression site i in the q-imp filter.*

If the following conditions all hold, then $\mathbb{I}_{pass}(t) =$ TRUE, i.e. the AtomicFilterCheckAndConsume *returns* TRUE*:*

1. *The adversary has created at most M^{adv} and N^{adv} `imp-quota` and `conv-quota` filters, respectively by the end of time t*
2. $\epsilon_x^t \le \epsilon_{nc}(b)$
3. $\epsilon_x^t \le \epsilon_{conv\text{-}quota}(c)$
4. $\epsilon_{\mathbf{x}}^{\mathbf{i},t}[i] \le \epsilon_{imp\text{-}quota}(i)$ *for $i \in \mathbf{i}$*

Proof. In theorem 1, we have proved the upper bound for the adversarial consumption, $\epsilon_{\text{global}}^{\text{adv}}$ from the global filter given the first condition, so by the set-up of §4.3, the benign users will have enough global filter budget left for their queries and:

$$\mathcal{F}_x^c.\,\text{canConsume}(\epsilon_x^t) = \mathsf{TRUE}. \tag{46}$$

In Alg. 3 Line **??**, we define canConsume to return TRUE iff enough budget is left to consume the specified budget in the corresponding filter (by basic pure DP composition). So:

- $\epsilon_x^t \le \epsilon_{\text{nc}}(b) = \epsilon_{\text{nc}} - \sum_{k=1}^{t-1} \epsilon_x^k(b) \cdot \mathbb{I}_{\text{pass}}[k]$ means that, by definition:

$$\mathcal{F}_x^{\text{nc[b]}}.\,\text{canConsume}(\epsilon_x^t) = \mathsf{TRUE}. \tag{47}$$

- $\epsilon_x^t \le \epsilon_{\text{conv-quota}}(c) = \epsilon_{\text{conv-quota}} - \sum_{k=1}^{t-1} \epsilon_x^k(c) \cdot \mathbb{I}_{\text{pass}}[k]$ means that, by definition:

$$\mathcal{F}_x^{\text{q-conv[c]}}.\,\text{canConsume}(\epsilon_x^t) = \mathsf{TRUE}. \tag{48}$$

- For any $i \in \mathbf{i}$, $\epsilon_{\mathbf{x}}^{\mathbf{i},t}[i] \le \epsilon_{\text{imp-quota}}(i) = \epsilon_{\text{imp-quota}} - \sum_{k=1}^{t-1} \epsilon_{\mathbf{x}}^{\mathbf{i},k}(i) \cdot \mathbb{I}_{\text{pass}}[k]$ means that, by definition:

$$\mathcal{F}_x^{\text{q-imp[i]}}.\,\text{canConsume}(\epsilon_x^t) = \mathsf{TRUE}. \tag{49}$$

Thus, when all conditions are satisfied, all of equations (46)-(49) hold, which means all filter checks should pass and AtomicFilterCheckAndConsume should return TRUE. □

A.1.6 Sensitivity analysis and privacy budget.

For epoch-level budget consumption, we defer readers to §C of [TKM+24]. For the rest of this section, we refine to epoch-site level budget consumption results that justify Alg. 5.

Theorem 10 (Global sensitivity of reports per epoch-site)**.** *Fix a report identifier r, a device d, a set of epochs $E_r = \{e_1^{(r)}, \dots, e_k^{(r)}\}$, and a set of sites $I_r^{(e)} = \{i_1^{(e)}, \dots, i_{m_e}^{(e)}\}$ for each epoch $e \in E_r$. Let $A(\cdot)$ be the attribution function of interest, so that the corresponding report is $\rho : D \mapsto A(D_d^{E_r, \{I_r^{(e)}\}_{e \in E_r}})$. We have $\Delta(\rho)$*

$$= \max_{\substack{i \in [k], e := e_i^{(r)}, j \in [m_e], e' = e_k^{(r)} \\ \mathcal{F} := (F_{1,1}, \dots, F_{k, m_{e'}}) \subseteq (\mathcal{S} \times \mathcal{S} \times \mathcal{C} \cup \mathcal{S} \times \mathcal{I})^k}} \|A(\mathcal{F} : F_{i,j} := \emptyset) - A(\mathcal{F})\|_1. \tag{50}$$

Proof. Fix a report ρ. Let k be $|E_r|$. Then, for $i \in [k]$, we enumerate through the epochs in E_r, and call the i-th epoch e_i. By definition of global sensitivity:

$$\Delta(\rho) = \max_{D, D' \in \mathbb{D} : \exists x \in \mathcal{X}, D' = D + x} \|\rho(D) - \rho(D)\|_1, \tag{51}$$

from which we expand what ρ function does:

$$\Delta(\rho) = \max_{\substack{D, D' \in \mathbb{D}: \\ \exists x \in \mathcal{X}, D' = D + x}} \|A(D_d^{E_r, \{I_r^{(e)}\}_{e \in E_r}}) - A((D')_d^{E_r, \{I_r^{(e)}\}_{e \in E_r}})\|_1 \tag{52}$$

$$= \max_{\substack{D, D' \in \mathbb{D}: \\ \exists x = (d, e, F) \in \mathcal{X}: \\ e \in E_r, D' = D + x, \\ \forall (i, imp) \in F : i \in I_r^{(e)}}} \|A(D_d^{E_r, \{I_r^{(e)}\}_{e \in E_r}}) - A((D')_d^{E_r, \{I_r^{(e)}\}_{e \in E_r}})\|_1, \tag{53}$$

because for $(d', e', F) \in \mathcal{X}$ with $d' \neq d$ or $e' \notin E_r$, or any $(i, imp) \in F$ where $i \notin I_r^{(e)}$, they would not be considered as in the computation of $A(\cdot)$ and $A(D_d^{E_r, \{I_r^{(e)}\}_{e \in E_r}}) = A((D')_d^{E_r, \{I_r^{(e)}\}_{e \in E_r}})$.

Next, we show that the following sets are equal:

- $\{(D_d^{E_r, \{I_r^{(e)}\}_{e \in E_r}}, (D')_d^{E_r, \{I_r^{(e)}\}_{e \in E_r}}) : D, D' \in \mathbb{D} : \exists x = (d, e, F) \in \mathcal{X} : e \in E_r, D' = D + x, \forall (i, imp) \in F : i \in I_r^{(e)}\}$.
- $\{(\mathcal{F} : F_{i,j} = \emptyset, \mathcal{F}) : i \in [k], e := e_i^{(r)}, j \in [m_e], e' = e_k^{(r)}, \mathcal{F} := (F_{1,1}, \dots, F_{k, m_{e'}}) \subseteq (\mathcal{S} \times \mathcal{S} \times \mathcal{C} \cup \mathcal{S} \times \mathcal{I})^k\}$

We show that for all instances on the one side, there exists a corresponding instance on the other. In one direction, take any tuple from the first set, where $(D')_d^{E_r, \{I_r^{(e)}\}_{e \in E_r}} = D_d^{E_r, \{I_r^{(e)}\}_{e \in E_r}} + x$, for some $x = (d, e, F)$. Firstly, the x satisfies $d = d_r$, $e \in E_r$, and $\forall f = (i, imp) \in F, i \in I_r^{(e)}$. Therefore, we construct $\mathcal{F}$ as follows:

$F_{i,j} = \{f \in F : f = (j, imp)$ or $(c, q, conv)$ is relevant to site $j\}$.

Note that every $F_{i,j}$ is set independently from x, except for the one that contains the events for site j_0 such that F contains either (j_0, imp) or $(c, q, conv)$ that is associated with j_0, in epoch i_0 such that $e_{i_0}^{(r)} = e \in E_r$. Since $x \notin D$, we know that F_{i_0,j_0} is set to $\emptyset$ on the side corresponding to D, and it is set to contain all events related to j_0 in epoch i_0 on the side corresponding to D'. That is, the corresponding instance in the second set would be $(\mathcal{F} : F_{i_0,j_0} = \emptyset, \mathcal{F})$.

Conversely, let $(\mathcal{F} : F_{i,j} = \emptyset, \mathcal{F})$ be from the second set. Then, we know the set of events corresponding to site j in epoch i is empty for the first element. So, we let $x := (d, e, F) \in \mathcal{X}$, where d is the same devices as was fixed, $e = e_i^{(r)}$, and F contains events related to site j in epoch i. Then, we let $(D, D') \in \mathbb{D} \times \mathbb{D}$, where $D' = D + x$ and everything in D and D' other than x corresponds to the rest of the $F_{i,j}$ that are the same in both $\mathcal{F}$ and $\mathcal{F} : F_{i,j} = \emptyset$. As such, $(D_d^{E_r,\{I_r^{(e)}\}_{e\in E_r}}, (D')_d^{E_r,\{I_r^{(e)}\}_{e\in E_r}})$ is the corresponding element in the first set.

Finally, we substitute the first set in equation line (53) by the second set accordingly, and the resulting max should be equal as the two sets are equivalent, and the resulting form will be the one claimed. □

Lemma 3. *Suppose the attribution function $A(\cdot)$ has an m-dimensional outputs, and each dimension of any $\mathcal{F} \subseteq (\mathcal{S} \times \mathcal{S} \times \mathcal{I} \cup \mathcal{S} \times \mathcal{C})^k$ satisfy $\forall i \in [m]$, $A(\mathcal{F})_i \in [0, A^{\max}]$, then we have $\Delta(\rho) \leq mA^{\max}$.*

Proof. Given the form in theorem 10, we know $\Delta(\rho)$ equals the max over $\|A(\mathcal{F} : F_{i,j} = \emptyset) - A(\mathcal{F})\|_1 = \sum_{i=1}^{m} |A(\mathcal{F} : F_{i,j} = \emptyset)_i - A(\mathcal{F})_i|$. But, we know for each dimension, $A(\mathcal{F}_i) \in [0, A^{\max}]$, so the absolute value of the difference in each dimension is $\in [0, A^{\max}]$. Thus, the sum over m dimensions is $\|A(\mathcal{F} : F_{i,j} = \emptyset) - A(\mathcal{F})\|_1 \leq mA^{\max}$. □

Theorem 11 (Global sensitivity of queries per epoch-site)**.** *Let R be the set of report identifiers relevant to a query Q. Then, we write $(\rho_r, d_r, E_r)_{r\in R}$ as the reports, devices, and epoch windows corresponding to each $r \in R$. Then, we have*

$$\Delta(Q) \leq \max_{(d,e,F)\in\mathcal{X}} \sum_{\substack{r\in R: d=d_r, e\in E_r,\\ \exists i\in I_r^{(e)}: F \text{ contains events related to } i}} \Delta(\rho_r).$$

Proof. By definition of global sensitivity,

$$\Delta(Q) = \max_{D,D'\in\mathbb{D}: \exists x\in\mathcal{X}, D'=D+x} \|Q(D) - Q(D')\|_1 \tag{54}$$

$$= \max_{x\in\mathcal{X}} \max_{D,D'\in\mathbb{D}: D'=D+x} \|Q(D) - Q(D')\|_1. \tag{55}$$

Let $x = (d, e, F) \in \mathcal{X}$, where F contains events associated with site j. For $r \in R$ such that $d \neq d_r$ or $e \notin E_r$, or F contains events related to site $i \notin I_r^{(e)}$, we have $\rho(D) = \rho(D')$. So, by applying triangle-inequality in the mean time:

$$\|Q(D) - Q(D')\|_1 \leq \sum_{r\in R: d=d_r, e\in E_r, i\in I_r^{(e)}} \|\rho_r(D) - \rho_r(D')\|_1 \tag{56}$$

$$\leq \sum_{r\in R: d=d_r, e\in E_r, i\in I_r^{(e)}} \Delta(\rho_r), \tag{57}$$

where the second inequality is by definition of $\Delta(\rho_r)$. Note that this bound is independent from the choice of D and D', so we take the max over (d, e, i) and get

$$\Delta(Q) \leq \max_{d,e,i} \sum_{r\in R: d=d_r, e\in E_r, i\in I_r^{(e)}} \Delta(\rho_r).$$

But, the choice of i can be used to set F accordingly, so, equivalently, this suffices to prove the theorem. □

Lemma 4. *If each device-epoch-site participates in at most one report, then $\Delta(Q) = \max_{r\in R} \Delta(\rho_r)$.*

Proof. Since each device-epoch-ste participates in at most one report, $\sum_{r\in R: d=d_r, e\in E_r, i\in I_r^{(e)}} \Delta(\rho_r) = \Delta(\rho_r)$. So,

$$\Delta(Q) \leq \max_{(d,e,F)\in\mathcal{X}} \Delta(\rho_r).$$

But then, $\Delta(Q)$ must match the sensitivity of one of the r's in this case, so the inequality is tight: $\Delta(Q) = \max_{r\in R} \Delta(\rho_r)$. Particularly, $\forall r$, $\exists D \sim D'$, such that $\|\rho_r(D') - \rho_r(D)\|_1 = \Delta(\rho_r)$. □

Theorem 12 (Individual sensitivity of reports per epoch-site)**.** *Fix a report identifier r, a device d_r, a set of epochs $E_r = \{e_1^{(r)}, \dots, e_k^{(r)}\}$, a set of sites $I_r^{(e)} = \{i_1^{(e)}, \dots, i_{m_e}^{(e)}\}$ for each epoch $e \in E_r$, an attribution function A with relevant events F_A, and the corresponding report $\rho : D \mapsto A(D_{d_r}^{E_r,\{I_r^{(e)}\}_{e\in E_r}})$. Fix a device-epoch record $x = (d, e, F) \in \mathcal{X}$, where $F \subseteq \mathcal{S} \times \mathcal{S} \times \mathcal{C} \cup \mathcal{S} \times \mathcal{I}$, so that $x_i = (d, e, F_i)$ is the projection where F_i contains only events related to site i.*

If the report requests a single epoch $E_r = \{e_r\}$ as well as a single site in the one epoch, $I_r^{(e_r)} = \{i_r\}$, then we have:

$$\Delta_{x_i}(\rho) = \begin{cases} \|A(F_i) - A(\emptyset)\|_1 & \text{, if } d = d_r, e = e_r \text{ and } i = i_r \\ 0 & \text{, otherwise.} \end{cases} \tag{58}$$

In particular, it should be intuitive to see why $\mathcal{F}$ is a single F_i in the first case, because we only have one epoch which contains one site, so there should be only one set of events corresponding to the one site in the one epoch.

Otherwise, either $|E_r| \geq 2$ or $|I_r^{(e)}| \geq 2$ for some $e \in E_r$, or both, and so we have:

$$\Delta_{x_i}(\rho) \leq \begin{cases} \Delta(\rho) & \text{, if } d = d_r,\ e \in E_r,\ i \in I_r^{(e)} \text{ and } F_i \cap F_A \neq \emptyset \\ 0 & \text{, otherwise.} \end{cases} \tag{59}$$

Proof. Fix a report ρ and $x_i = (d, e, F_i) \in \mathcal{X}$. Consider any $D, D' \in \mathbb{D}$ such that $D' = D + x_i$. We have $\rho(D) = A(D_{d_r}^{E_r,\{I_r^{(e)}\}_{e\in E_r}})$ and $\rho(D') = A((D')_{d_r}^{E_r,\{I_r^{(e)}\}_{e\in E_r}})$.

- First, if $d \neq d_r$, $e \notin E_r$, or $i \notin I_r^{(e)}$, then $(D')_{d_r}^{E_r,\{I_r^{(e)}\}_{e\in E_r}} = D_{d_r}^{E_r,\{I_r^{(e)}\}_{e\in E_r}}$. Hence, $\|\rho(D) - \rho(D')\|_1 = 0$ for all such D, D', which implies $\Delta_{x_i}(\rho) = 0$.
- Next, suppose that the report requests a single epoch $E_r = \{e_r\}$ with a single site $I_r^{(e_r)} = \{i_r\}$:
  - If $d = d_r$, $e = e_r$, and $i = i_r$, then since $D + x_i = D'$, we must have $(d_r, e_r, F_i) \notin D$, and thus $D_{d_r}^{e_r,i_r} = \emptyset$. On the other hand, $(D')_{d_r}^{e_r,i_r} = F_i$ (restricted to events relevant to site i_r). Thus, $\|\rho(D)-\rho(D')\|_1 = \|A(F_i)-A(\emptyset)\|_1$.
  - If $d \neq d_r$, $e \neq e_r$, or $i \neq i_r$, then (d, e, F_i) doesn't change the outcome and $(D')_{e_r}^{i_r} = D_{e_r}^{i_r}$. Hence, $\|\rho(D)-\rho(D')\|_1 = 0$.
- Now, suppose that the report requests either an arbitrary range of epochs E_r each of whom has at least one site, or a single epoch that has multiple sites $I_r^{(e_r)}$:
  - If $d \neq d_r$, $e \notin E_r$, or $i \notin I_r^{(e)}$, then $A((D')_{d_r}^{E_r,\{I_r^{(e)}\}_{e\in E_r}}) = A(D_{d_r}^{E_r,\{I_r^{(e)}\}_{e\in E_r}})$, i.e., $\|\rho(D') - \rho(D)\|_1 = 0$.
  - If we have $d = d_r$, $e = e_j^{(r)} \in E_r$, and $i \in I_r^{(e)}$, but F_i is simply not related to the attribution request, i.e. $F_i \cap F_A = \emptyset$. Then, by definition of F_A, we have $A((D')_{d_r}^{E_r,\{I_r^{(e)}\}_{e\in E_r}}) = A(D_{d_r}^{E_r,\{I_r^{(e)}\}_{e\in E_r}})$, i.e., $\|\rho(D) - \rho(D')\|_1 = 0$.
  - Otherwise, it must be the case that $d = d_r$, $e = e_j^{(r)} \in E_r$, $i \in I_r^{(e)}$ and $F_i \cap F_A \neq \emptyset$ and there are events in the intersection that is related to some site i in epoch e, so we have:

$$\|\rho(D) - \rho(D')\|_1 = \|A(\mathcal{F} : F_{j,i} = \emptyset) - A(\mathcal{F})\|_1, \quad (60)$$

  where j is the index of epoch e in E_r, and $F_{j,i}$ represents the relevant events for site i in epoch $e_j^{(r)}$.

The first two cases are independent over choices of $D \sim D'$, so taking the max over such choices still gives $\Delta_{x_i}(\rho) = 0$. Unfortunately, the third identity does depend on the choice of $D \sim D'$, and taking the max only gives the general definition of global sensitivity, in the worst case. Particularly,

$$\Delta_{x_i}(\rho) = \max_{\mathcal{F}=\{F_{j,i}\}:F_{j,i}\subseteq\mathcal{S}\times\mathcal{S}\times\mathcal{C}\cup\mathcal{S}\times\mathcal{I}} \|\rho(D) - \rho(D')\|_1 \quad (61)$$

$$= \max_{\mathcal{F}=\{F_{j,i}\}:F_{j,i}\subseteq\mathcal{S}\times\mathcal{S}\times\mathcal{C}\cup\mathcal{S}\times\mathcal{I}} \|A(\mathcal{F} : F_{j,i} = \emptyset) - A(\mathcal{F})\|_1 \quad (62)$$

$$\leq \Delta(\rho), \quad (63)$$

where the last inequality is tight due to the definition of global sensitivity and theorem 10, for the worst case choice of x_i in general.

□

Theorem 13 (Individual sensitivity of queries per device-epoch-site)**.** *Fix a query Q with corresponding report identifiers R and reports $(\rho_r)_{r\in R}$. Fix a device-epoch-site record $x_i = (d, e, F_i) \in \mathcal{X}$, where F_i contains only events related to site i. We have:*

$$\Delta_{x_i}(Q) \leq \sum_{r\in R} \Delta_{x_i}(\rho_r) \quad (64)$$

In particular, if x_i participates in at most one report ρ_r, then $\Delta_{x_i}(Q) = \Delta_{x_i}(\rho_r)$.

Proof. Take $D, D' \in \mathcal{D}$ such that $D' = D + x_i$. By the triangle inequality:

$$\Delta_{x_i}(Q) = \max_{D'=D+x_i\in\mathcal{D}} \|Q(D) - Q(D')\|_1 \quad (65)$$

$$= \max_{D'=D+x_i\in\mathcal{D}} \left\| \sum_{r\in R} \rho_r(D) - \rho_r(D') \right\|_1 \quad (66)$$

$$\leq \sum_{r\in R} \max_{D'=D+x_i\in\mathcal{D}} \|\rho_r(D) - \rho_r(D')\|_1 \quad (67)$$

$$\leq \sum_{r\in R} \Delta_{x_i}(\rho_r), \quad (68)$$

where the last inequality is by the definition of individual sensitivity.

When x_i participates in at most one report ρ_{r_0}, we have $\Delta_{x_i}(\rho_r) = 0$ for all $r \neq r_0$. Therefore, $\Delta_{x_i}(Q) \leq \Delta_{x_i}(\rho_{r_0})$. This inequality is tight because there exists a pair (D^*, D'^*) with $D'^* = D^* + x_i$ such that $\|\rho_{r_0}(D'^*) - \rho_{r_0}(D^*)\|_1 = \Delta_{x_i}(\rho_{r_0})$, and for this pair, $\|Q(D^*) - Q(D'^*)\|_1 = \Delta_{x_i}(\rho_{r_0})$. □

A.2 Cross-report privacy loss optimization

In practice, a single conversion event can trigger multiple reports, each sent to a different beneficiary site. These reports are typically processed in isolation, with each one independently subtracting budget from the c-filter. However, when the reports have certain structurel, such as forming a single large histogram partitioned across beneficiary sites, it can be more efficient to analyze them jointly. For example, each beneficiary site might receive a report containing only the impressions that occurred on their site for a given conversion event. In this section, we explore how c-filter privacy deduction can be optimized by jointly processing such partitioned reports, reducing redundant budget consumption.

{Mathias: we need intuition in english for this def (for me and the generic reader both :)): what are the F_k in the sense of what they represent/why F needs to be decomposed this way? can they overlap (right now yes)? what do the conditions mean?} Alison: I rewrote this to make it clearer as to what the goals are. Let me know if I should add more details

Definition 8 (Histogram attribution function)**.** *Let A be an attribution function $A : \mathcal{P}(\mathcal{I})^k \to \mathbb{R}^m$, for a fixed k and m. A is a histogram attribution function, if for any vector of a set of impressions* **F**, *the following conditions all hold:*

1. *There exists* $a_{\mathbf{F}} : \mathcal{I} \to \mathbb{R}_+$ *a function that attributes a value to each impression* $f \in \mathcal{I}$ *depending on all the other impressions* **F**.
2. *There exists a one-hot encoding function* H *that maps each event* f *to one of* m *buckets. That is,* $H : \mathcal{I} \to \{0,1\}^m$ *such that* $\forall f \in \mathcal{I}, ||H(f)||_1 = 1$.
3. $A(\mathbf{F}) = \sum_{f\in\mathbf{F}} a_{\mathbf{F}}(f) \cdot H(f)$

Definition 9 (Query partitions for beneficiary sites)**.** *Let* A *be a histogram attribution function* $A : \mathcal{P}(\mathcal{I})^k \to \mathbb{R}^m$, *for a fixed* k *and* m *and with an associated one-hot encoding function* $H : \mathbb{R}^m \to \{0,1\}^m$. *Fix* $n \in \mathbb{N}_+$. *Let* **F** *be a fixed vector of event sets. We define the following:*

- H_j *is an* encoding partition *of* H *for some* $j \in [n]$ *if* $H_j : \mathcal{I} \to \{0,1\}^m$ *and for some subset of impressions* $S_j \subseteq \mathcal{I}$ *we have that*
$$H_j(f) = \begin{cases} H(f) & \text{if } f \in S_j \\ 0 & f \in \mathcal{I} \setminus S_j \end{cases}$$
- $\Psi : \{0,1\}^m \times \cdots \times \{0,1\}^m \to \{0,1\}^m$ *is a* concatenation map *for a set of encoding partitions* H_j *of* H *if*
$$\Psi(H_1(f), H_2(f), ..., H_n(f)) = H(f) \tag{69}$$
for all $f \in \mathcal{I}$.
- *A set of encoding partitions* $H_1, H_2, ..., H_n$ *is a* valid partition set *of encoding* H *if there exists a concatenation isomorphism* Ψ *such that* $\Psi(H_1, H_2, ..., H_n) = H$ *and for all* $f \in \mathcal{I}$ *we have that* $||H_j(f)||_1 = 1$ *for exactly one* $j \in [n]$.
- A_j *is the* attribution partition *of* A *for some* $j \in [n]$ *where*
$$A_j(\mathbf{F}) := \sum_{f\in\mathbf{F}} a_{\mathbf{F}}(f) \cdot H_j(f) \tag{70}$$
- ρ^j *is the* report partition *for* $j \in [n]$ *for a fixed report identifier* $r \in \mathbb{Z}$ *where*
$$\rho^j_r(D) = A_j(D^E_d) \tag{71}$$

Let $H_1, H_2, ..., H_n$ *be a valid encoding partition of encoding* H. *Let* $B = \{b_1, \ldots, b_n\}$ *be a set of beneficiary sites, where each* b_j *has a corresponding encoding partition* H_j. *Consider a set of report identifiers* R. *We define* Q_j *as the* query partition *that results by using the attribution partition* A_j *and the corresponding report partitions* ρ^j. *We have:*

$$Q_j(D) := \sum_{r\in R} \rho^j_r(D) \tag{72}$$

Proposition 1 (Recovering report partitions by postprocessing a concatenated report)**.** *Let* $H_1, H_2, ..., H_n$ *be a valid encoding partition of encoding* H, *with concatenation map* Ψ. *By definition, each* H_j *has a corresponding subset of impressions* $S_j \subseteq \mathcal{I}$. *Since* Ψ *is an*

Proposition 2 (Individual sensitivity of a report partition)**.** *Fix a report identifier* r, *a device* d_r, *a set of epochs* E_r, *a beneficiary* $j \in B$, *an attribution partition function* A_j *with relevant events* F^j_A *and the corresponding report partition* $\rho^j : D \to A_j(D^{E_r}_{d_r})$. *We define*

$$A^{\max} := \max_{F\in\mathcal{P}(\mathcal{I}\cup C)^k} \sum_{i=1}^{k} \sum_{f\in F_i\cap F_A} a_F(f) \tag{73}$$

For a fixed device-epoch record $x = (d, e, F) \in \mathcal{X}$, *we have that the individual sensitivity of* ρ_j *is*

$$\Delta_x(\rho^j) \le \begin{cases} 0 & \text{if } d \ne d_r, e \notin E_r \text{ or } F \cap F^j_A = \emptyset \\ \|A_j(F)\|_1 & \text{if } d = d_r \text{ and } E_r = \{e\} \\ 2A^{\max} & \text{if } d = d_r, e \in E_r \text{ and } F \cap F^j_A \ne \emptyset \end{cases}$$

Proof. Follows directly from Theorem 4 of [TKM+24]. □

Proposition 3 (Individual sensitivity of a concatenated report)**.** *Fix a report identifier* r, *a device* d, *a set of epochs* E_r, *a histogram attribution function* A *and a set of beneficiaries* $B = \{b_1, b_2, ..., b_n\}$. *Suppose each beneficiary has a corresponding report partition (i.e.* ρ_j *is the report partition for beneficiary* b_j*) that results from histogram attribution function* A. *For a fixed device-epoch record* $x = (d, e, F) \in \mathcal{X}$, *the individual sensitivity of the concatenated report* ρ *is*

$$\Delta_x(\rho) \le \begin{cases} 0 & \text{if } d \ne d_r, e \notin E_r \text{ or } F \cap F_A = \emptyset \\ \|A(F)\|_1 & \text{if } d = d_r \text{ and } E_r = \{e\} \\ 2A^{\max} & \text{if } d = d_r, e \in E_r \text{ and } F \cap F_A \ne \emptyset \end{cases}$$

Alison: I will add these details, but it should be the same as proposition 1

Proposition 4 (Sensitivity of query partitions Q_j)**.** *Fix a relevant event set* F_A *and a corresponding partition* F^j_A. *Let* Q_j *be the query partition of beneficiary* j *that considers only events in* F^j_A. *Fix a device-epoch record* $x = (d, e, F) \in \mathcal{X}$, *where* F. *Then if* x *participates in a single report* r', *we have that*

$$\Delta_x(Q_j) = \Delta_x(\rho^j_r) \tag{74}$$

Proof. Take $D, D' \in \mathcal{D}$ such that $D' = D + x$. We have that

$$\begin{aligned} \Delta_x(Q_j) &= \max_{D'=D+x_i\in\mathcal{D}} \|Q_j(D) - Q_j(D')\|_1 && (75) \\ &= \max_{D'=D+x\in\mathcal{D}} \left\| \sum_{r\in R} \rho^j_r(D) - \rho^j_r(D') \right\|_1 && (76) \\ &= \max_{D'=D+x\in\mathcal{D}} \left\| \rho^j_{r'}(D) - \rho^j_{r'}(D') \right\|_1 && (77) \\ &= \Delta_x(\rho^j_{r'}) && (78) \end{aligned}$$

by the definition of individual sensitivity. □

{Mathias: I can't quite get what Proposition 2 and Theorem 9 say. I was expecting something like $\Delta_x(Q) \le \max_{i\in[n]} \Delta_x(\rho^i_r)$ or something, to basically say "if we can decompose to queries over the event set, we only pay once, and we pay the max sensitivity if it's not equal" or something like this.}

Proposition 5 (Multi-beneficiary optimization for query Q)**.** *Let F_A be a fixed relevant event set, such that F_A is partitioned into n disjoint subsets and $F_A = F_A^1 \sqcup F_A^2 \sqcup ... \sqcup F_A^n$. Let $B = \{b_1, \ldots, b_n\}$ be a set of beneficiary sites, where each b_j has a set of relevant events F_A^j. Consider a set of report identifiers R. Let each beneficiary site b_j also have a corresponding query partition Q_j that results by considering only F_A^j and the corresponding report partitions ρ_r^j for $r \in R$ and attribution partition A_j. Let $Q = Q_1, Q_2, ..., Q_n$ be the query that results from processing all of the beneficiaries queries Q_j at once. Fix a device-epoch record $x = (d, e, F) \in \mathcal{X}$. We have that the sensitivity of Q is such that*

$$\Delta_x(Q) \leq \sum_{r \in R} \Delta_x(\rho_r) \tag{79}$$

Proof. We first notice that for a fixed r we have that $\sum_{j=1}^{n} \rho_r^j(D) = \rho_r(D)$. This follows directly from our definition of report partition.

$$\begin{aligned}
\sum_{j=1}^{n} \rho_r^j(D) &= \sum_{j=1}^{n} A_j(D_d^E) && (80)\\
&= \sum_{j=1}^{n} \sum_{i=1}^{k} \sum_{f \in F_i \cap F_A^j} a_{\mathbf{F}}(f) \cdot H_j(f) && (81)\\
&= \sum_{i=1}^{k} \sum_{f \in (F_i \cap F_A^1) \cup ... \cup (F_i \cap F_A^n)} a_{\mathbf{F}}(f) \cdot H(f) && (82)\\
&= \sum_{i=1}^{k} \sum_{f \in (F_i \cap F_A)} a_{\mathbf{F}}(f) \cdot H(f) && (83)\\
&= \rho_r(D) && (84)
\end{aligned}$$

Now, take $D, D' \in \mathcal{D}$, such that $D' = D + x$. We have that

$$\begin{aligned}
\Delta_x(Q) &= \max_{D'=D+x \in \mathcal{D}} \|Q(D) - Q(D')\|_1 && (85)\\
&= \max_{D'=D+x \in \mathcal{D}} \|\sum_{j=1}^{n} Q_j(D) - Q_j(D')\|_1 && (86)\\
&= \max_{D'=D+x \in \mathcal{D}} \|\sum_{j=1}^{n} \sum_{r \in R} \rho_r^j(D) - \rho_r^j(D')\|_1 && (87)\\
&= \max_{D'=D+x \in \mathcal{D}} \|\sum_{r \in R} \sum_{j=1}^{n} \rho_r^j(D) - \sum_{j=1}^{n} \rho_r^j(D')\|_1 && (88)\\
&= \max_{D'=D+x \in \mathcal{D}} \|\sum_{r \in R} \rho_r(D) - \rho_r(D')\|_1 && (89)\\
&\leq \sum_{r \in R} \max_{D'=D+x \in \mathcal{D}} \|\rho_r(D) - \rho_r(D')\|_1 && (90)\\
&\leq \sum_{r \in R} \Delta_x(\rho_r) && (91)
\end{aligned}$$

□

Theorem 14 (Cross-report privacy loss optimization)**.**

Example We now provide more details on how these optimizations by giving more details to the example from Section 4.2. provided in provide an example to illustrate the c-filter budget optimizations of processing Q as a whole rather than processing each individual beneficiary site's Q_j.

- Without Multi-Beneficiary Optimization
  - *individual sensitivity used for nc-filter deduction for beneficiary site j*: $\sum_{r \in R} \Delta_x(\rho_r^j)$
  - *individual sensitivity used for c-filter deduction*: $\sum_{j=1}^{n} \sum_{r \in R} \Delta_x(\rho_r)$
- With Multi-Beneficiary Optimization
  - *individual sensitivity used for nc-filter deduction for beneficiary site j*: $\sum_{r \in R} \Delta_x(\rho_r)$
  - *individual sensitivity used for c-filter deduction*: $\sum_{r \in R} \Delta_x(\rho_r)$

Since we have that $\sum_{r \in R} \Delta(\rho_r) \leq \sum_{j=1}^{n} \sum_{r \in R} \Delta(\rho_r^j)$, processing Q in its entirety spends less privacy budget.

B Batched Algorithm

Alg. 0 describes the batched algorithm for a single epoch.

Algorithm 6 Batched algorithm

```
Input:
$\epsilon_{\text{per-site}}, N, M, r, n$: same parameters as Alg. 2.
$T$: number of scheduling intervals in the epoch's data lifetime.

function Main
    $\epsilon_{\text{global}}, \epsilon_{\text{imp-quota}}, \epsilon_{\text{conv-quota}} \leftarrow \text{Capacities}(\epsilon_{\text{per-site}}, N, M, r, n)$
    $\mathcal{F} \leftarrow \text{InitializeFilters}(\epsilon_{\text{global}}, \epsilon_{\text{per-site}}, \epsilon_{\text{imp-quota}}, \epsilon_{\text{conv-quota}})$
    $\mathcal{F}^{\text{global filter}}.capacity \leftarrow 0$ // Initially no global budget available
    $\mathcal{R}_{batch} \leftarrow \emptyset$ // Requests for the batch phase
    for $t \in [T]$ do
        $\mathcal{R}_{new} \leftarrow \text{ReceiveNewRequests}()$
        $\mathcal{A}, \mathcal{R}_{batch} \leftarrow \text{ScheduleBatch}(\mathcal{R}_{new}, \mathcal{R}_{batch})$
        $\text{SendReportsForRelease}(\mathcal{A})$

function ScheduleBatch($\mathcal{R}_{new}, \mathcal{R}_{batch}$)
    // 1. Initialization phase
    $\mathcal{F}^{\text{global filter}}.capacity \leftarrow \mathcal{F}^{\text{global filter}}.capacity + \epsilon_{\text{global}}/T$
    $\mathcal{F}^{\text{impression-site quota}} \leftarrow \text{InitializeFilter}(\epsilon_{\text{imp-quota}}/T)$
    $\mathcal{A}_{init}, \mathcal{U}_{init} \leftarrow \text{TryAllocate}(\mathcal{R}_{batch})$
    $\mathcal{A} \leftarrow \mathcal{A}_{init}$
    // 2. Online phase
    $a_{online}, u_{online} \leftarrow \text{TryAllocate}(\mathcal{R}_{new})$
    $\mathcal{A} \leftarrow \mathcal{A} \cup \mathcal{A}_{online}$
    // 3. Batch phase
    $\mathcal{F}^{\text{impression-site quota}}.capacity \leftarrow \infty$
    $batch \leftarrow \mathcal{U}_{init} \cup \mathcal{U}_{online}$
    do
        $sorted \leftarrow \text{SortBatch}(batch)$
        $(a, u) \leftarrow \text{TryAllocateOne}(sorted)$
        $batch \leftarrow u$
        $\mathcal{A} \leftarrow \mathcal{A} \cup a$
    while $\mathcal{A} \neq \emptyset$
    return $\mathcal{A}, batch$
```

We define $\mathcal{A}, \mathcal{U} \leftarrow \text{TryAllocate}(\mathcal{R})$ as follows. TryAllocate takes a set of report requests $\mathcal{R}$, and executes Alg. 2's GenerateReport function for each request. It then returns two sets: $\mathcal{A}$ the reports for requests that were allocated successfully (*i.e.,* the filters returned TRUE at line Line 6), and $\mathcal{U}$ the remaining unallocated requests. TryAllocateOne behaves like TryAllocate, except that it stops after the first successfully allocated request.

SortBatch attaches a weight (b_r, ϵ_r) to each request r in a batch, given access to filters $\mathcal{F}$, and then sorts by smallest weight first (in lexicographic order). The weights are defined as follows. ϵ_r is the global epsilon requested by r (either available as a request parameter, or computed as $\epsilon_r = \Delta\rho_r/\sigma$ for a Laplace noise scale σ). b_r is the smallest budget consumed by any impression site $i \in \mathbf{i}_r$ requested by r:

$$b_r := \min_{i \in \mathbf{i}_r} \mathcal{F}^{\text{impression-site quota}}[i].consumed \qquad (92)$$

Finally, SendReportsForRelease prepares the reports from allocated requests to be sent at the right time, depending on the duration specified by each request.

Proof for response time tradeoff.

Proof. We prove by induction that the longer the response time for a request, the higher the probability the request will get allocated privacy budget. The base case ($0 \leq n < l$, where l is the length of a single schedule interval, and where without loss of generality the request is submitted at the beginning of a scheduling interval) is a response time that results in a response within the same scheduling interval as the request. In this case, the request is only granted the opportunity to be scheduled during the online phase. If the response time is one scheduling interval longer ($l \leq n < 2l$) than the base case, the request has the opportunity to be scheduled either: in the online phase of the scheduling interval it was requested in, in the batch phase of that scheduling interval, and in the initialization phase of the next scheduling interval, thereby increasing its probability of getting scheduled. The same property holds true for any response time n that falls within the k next intervals ($k \cdot l \leq n < k \cdot l + 1$), where if we increase the response time by one interval ($k \cdot l + 1 \leq n < k \cdot l + 2$), the request would have two additional opportunities (at the batch phase of scheduling interval k and the initialization phase of scheduling interval $k + 1$) to get scheduled, thereby increasing its probability of getting allocated budget. □