Privacy Preserving Measurement
Internet-Draft
Intended status: Standards Track
Expires: 22 August 2026

M. Thomson
Mozilla

18 February 2026

Distributed Aggregation Protocol (DAP) Extensions for the Attribution API

draft-thomson-ppm-dap-attribution-01

Abstract

This defines extensions to the DAP protocol that support the Attribution API. These extensions provide support for differentially-private aggregation and the operating modes that the Attribution API depends on.

About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at https://martinthomson.github.io/dap-attribution/draft-thomson-ppm-dap-attribution.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-thomson-ppm-dap-attribution/.

Discussion of this document takes place on the Privacy Preserving Measurement Working Group mailing list (mailto:ppm@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/ppm/. Subscribe at https://www.ietf.org/mailman/listinfo/ppm/.

Source for this draft and an issue tracker can be found at https://github.com/martinthomson/dap-attribution.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 22 August 2026.

Copyright Notice

Copyright (c) 2026 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction

The Attribution API [ATTR] is a web platform feature that provides sites that use advertising with the ability to measure the effectiveness of that advertising. Measurements that the API produce are aggregated using the Distributed Aggregation Protocol (DAP) [DAP].

This document defines extensions to DAP that support the use of DAP by the Attribution API. These extensions fall into two categories:

* Support for the differential privacy model used in the Attribution API.

* Support for the operational modes that better fit the deployment model that the Attribution API uses.

These extensions are not narrowly defined such that they can only be used by the Attribution API. Other applications might be able to use them, but any effort to make them fully generic stops short of making the extensions more complex that is required for Attribution.

For example, privacy budget extensions are defined to use the epsilon definition from (竜, 0)-differential privacy or (竜, 隆)-differential privacy with a fixed 隆 value. This is simpler than a design that might allow for multiple budget metrics.

1.1. Differential Privacy in Attribution

The Attribution API provides information to websites based on user activity on other websites, which is ordinarily prohibited for privacy reasons [WEB-PRIV].

To protect privacy, the Attribution API uses differential privacy [DP] in a combination of the central and individual models [ATTR-DP]. The privacy architecture of the Attribution API allocates responsibility for managing sensitivity and privacy budgets to browser instances (DAP Clients) and for the addition of noise to an aggregation service (DAP Aggregators, collectively).

To this end, reports that are submitted for aggregation need to be bound to the privacy budget that was expended at a Client when the report was generated. This ensures that Aggregators can apply noise with sufficient amplitude to maintain the intended differential privacy guarantee.

An extension that reports the amount of privacy budget that was consumed in the generation of a report is described in Section 3.

1.2. Attribution Operating Mode Extensions

The Attribution API is expected to operate in a mode that differs somewhat from the way that DAP is architected. Several extensions are defined to support this operating mode.

In DAP, a task is a long-running context that Clients continuously contribute to. Reports are directly uploaded by Clients to the Leader as they are generated. The DAP batch mode determines how reports are grouped for aggregation. A new collection job is initiated by the Collector when an aggregate is needed, though this might fail if the requirements for the task -- the batch mode and minimum batch size, primary -- are not met.

A simple representation of the DAP architecture is illustrated in Figure 1.

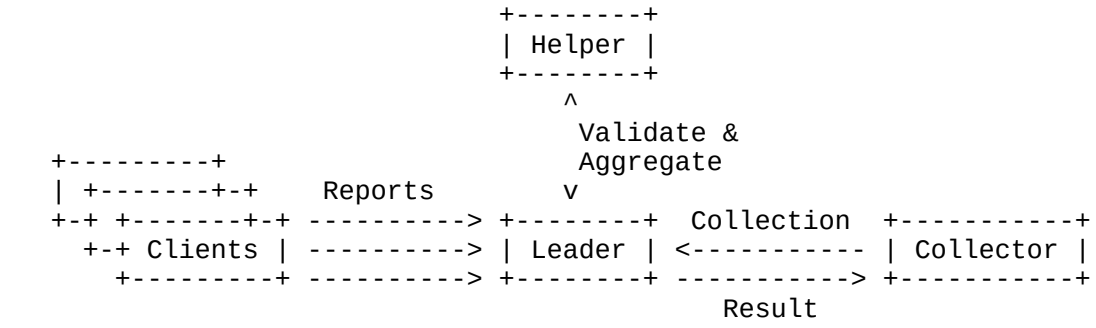

Figure 1: Simplified DAP Architecture

In the Attribution API, reports are delivered to the website that requests them. The site is expected to gather a bundle of reports and submit them for aggregation when they have a sufficiently large set.

An important feature of this design is that the website (as a DAP Collector) chooses which reports to aggregate. This gives the site an opportunity to review the circumstances in which reports were generated and filter reports according to their needs.

A secondary reason for Clients to deliver reports to the website, rather than submit them directly to the Leader, is that this removes a real-time dependency on the Leader. A Leader can therefore be less available than the sites that depend on its services.

A simple representation of the architecture used by the Attribution API is illustrated in Figure 2.

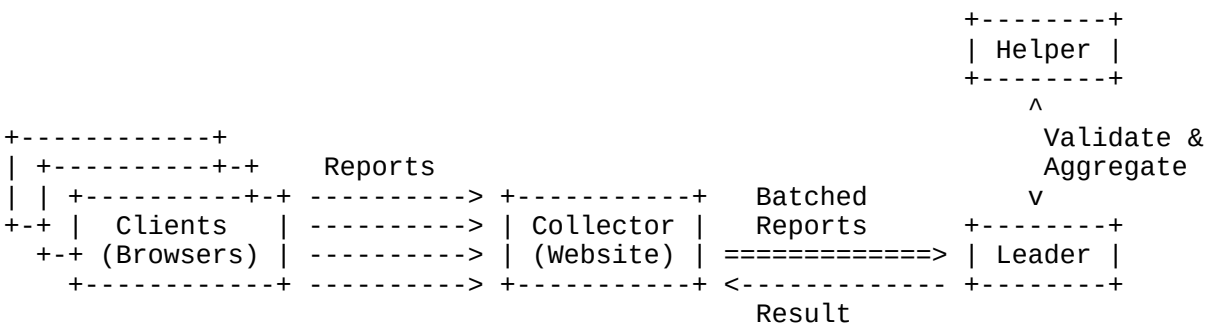

Figure 2: Attribution API Architecture

To support this mode of operation, a new batch mode for DAP is defined called "collector-selected"; see Section 4. For other batch modes, the Leader has sufficient information to select reports for inclusion in a collection job. In comparison, this batch mode requires that reports carry an annotation set by the Collector at the time that each report is uploaded.

1.3. Other Extensions

Three other extensions are defined in this document.

The minimum privacy budget collection job extension (Section 5) added to collection job initialization places guardrails around what reports can be included in a collection job. Reports that lack a privacy budget extension or those with a value below the indicated threshold must be rejected by Aggregators.

For the Attribution API, setting a minimum privacy budget is roughly equivalent to capping the noise that is added to an aggregate. Without this extension, noise could be determined from the privacy budget values bound to each report. This ensures that reports that do not match expectations can be dropped efficiently, rather than having Aggregators add unexpectedly large amounts of noise.

The collector identity task extension (Section 6) binds the identity of the Collector to tasks. This extension fixes the entity can request collection of reports.

In the Attribution API, the Collector is either a "conversion site", or an "internediary site"; see [ATTR]. The conversion site is the top-level site where reports are generated. An intermediary site is any entity that operates independently from the top-level site, and it includes the providers of resources (such as images or other content) and framed content (that is, HTML iframes).

The budget source task extension (Section 7) binds the identity of the context that provides privacy budget to tasks. This extension ensures that reports that draw from different privacy budgets cannot be aggregated together.

For the Attribution API, each "conversion site" receives their own source of privacy budget. Binding tasks to their identity ensures that the privacy guarantees associated with that privacy budget hold when Aggregators are responsible for adding noise. That is, reports that draw from different budgets cannot be aggregated together with a single quantity of noise.

2. Conventions and Definitions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

This document relies on the definitions in DAP [DAP] for protocol roles and functions. Some reference is made to the terms and concepts in the Attribution API [ATTR], though familiarity with these should not be necessary to understand how the extensions interact with DAP.

3. Differential Privacy Budget Report Extension

The privacy budget report extension (see Section 4.4.3 of [DAP]) codepoint 0xTBD, encodes the amount of privacy budget that might have been expended by a Client as a result of producing a report.

The value of the codepoint is an integer encoding of the number of micro-epsilons of budget that are expended. That is, each unit is a one-millionth of an epsilon (ε) as used in (ε , δ)-differential privacy.

The micro-epsilon value is encoded as a 32-bit integer in network byte order. This permits expenditure of up to ε =4294.967295 to be encoded.

3.1. Privacy Budgeting

A privacy budget ensures that the total information release can be bounded while providing more flexibility to the recipients of the noisy results. Recipients are able to adjust how budget is used to control how noise is distributed across multiple information releases.

The amount of noise added to aggregates is based on the expended budget. In general, spending more privacy budget means that less noise is needed to maintain the same level of privacy; conversely, spending less budget means more noise.

A budget might be specified in terms of a metric (like the epsilon parameter in (ε, δ)-differential privacy) that is expended with each information release. As noted, this extension uses the ε metric.

For example, for an overall budget of ε=10 might be split four ways: (0.5, 1.5, 2, 6). Noise might then be added, drawing from a distribution with a width inversely proportional to the budget spent; that is, a distribution with a standard deviation proportional to 2, 2/3, 1/2, and 1/6 respectively.

3.2. Applicability to Differential Privacy Models

The extensions in this document (Section 3 and Section 5) only support ε-differential privacy or (ε, δ)-differential privacy with a fixed delta (δ). Systems that use other budgeting methods require the use of a different extension.

A delta (δ) parameter is not directly bound to reports. This parameter is rarely used in privacy budgeting. Instead, a maximum value for δ might be fixed as part of the configuration in a specific deployment. Setting a value for δ is necessary when selecting a differential privacy mechanism. In setting a value for δ, a deployment needs to consider the total report volume and the total number of tasks that each client might contribute to.

> Note: Where the delta (δ) value is non-zero, and a client might generate many reports, clients might also need to limit the number of reports to prevent the overall delta value from growing large.

3.3. Use in the Attribution API

When used in the Attribution API [ATTR], the privacy budget report extension does not always encode the exact amount of privacy budget that was expended. The individual DP model used in Attribution (see [ATTR-DP]), allows a Client to expend less of its budget than this value for several reasons.

This introduces some complexity, because any reduction in the budget spent is based on private information. Consequently, any reduction in budget expenditure needs to be kept secret. In that setting, the extension reports the requested expenditure, which is the maximum value that might have been spent.

This extension gives users of the Attribution API the ability to manage privacy budget expenditure on a per-report basis.

By binding the amount of budget spent to each report, the Client can transfer responsibility for applying noise to Aggregators. The addition of noise in a single place can ensure a better trade-off between the amount of added noise and privacy parameters.

The Attribution API differs from some other uses of DAP. which instead involve Clients adding noise to reports at the time they are generated. To maintain the usefulness of aggregates, the amount of noise added by each Client is kept low. To maintain strong privacy, Clients partly rely on DAP mixing their contributions with the contributions of other Clients, following the shuffle DP approach [SHUFFLE]. Such uses depend on attaining a certain minimum batch size in order to meet their differential privacy targets.

Applying noise during aggregation reduces the importance of the minimum batch size parameter in task configuration. However, it adds to the work that Clients need to trust Aggregators to perform.

To maintain consistency with the DAP threat model for privacy (see Section 8 of [DAP]), this can mean either performing noise addition in an MPC protocol or having Aggregators each independently apply the requisite amount of noise.

A number of MPC protocols for adding noise exist, but these are often not efficient in the two-party setting relative to the VDAF protocols typically used. On the other hand, the redundant addition of noise by Aggregators results in more noise when both Aggregators are honest. Even so, adding two amounts of noise often results in less overall noise than other approaches.

4. Collector-Selected Batch Mode

The collector-selected batch mode (codepoint 0xTBD) give the Collector control over which reports are included in collection job.

In this batch mode the Leader is not able to determine the associated batch for a report based on the contents of each report alone. This differs from the existing leader-selected (Section 5.2 of [DAP]) or time interval (Section 5.1 of [DAP]) batch modes.

To that end, uploads of reports use a different URL template from the usual location for report uploads; see Section 4.1.

This arrangement prevents the Leader from accepting reports until a collection job is created. A Leader MAY accept hold requests to upload reports for a short period prior to the creation of a collection job. Accepting reports would be on the expectation that a request to create the indicated collection job is imminent or still being processed, so it could reduce latency.

However, the Leader MUST limit the reports it accepts prior to accepting the corresponding collection job. Without a limit, the Leader gives malicious actors the unrestricted capability to exhaust its resources.

A Leader that does not accept reports MUST reject the request, and can respond with an indication to try later. This response could use a 404 status code and the Retry-After response field; see Sections 15.5.5 and 10.2.3 of [HTTP].

> | Where a collection job already exists, the high entropy collection job ID in the URL could make it unnecessary to require authentication of upload requests for this batch mode; see [CAP-URL]. This is not the case if reports are accepted without confirming the existence of the identified collection job.

A Leader MUST reject attempts to upload reports to the regular report upload resource (as defined in Section 4.5.2 of [DAP]) when the collector-selected batch mode is configured for a task.

4.1. Report Upload URL Template

Reports in the collector-selected batch mode are uploaded to a URL that follows the template:

{leader}/tasks/{task-id}/reports/{collection-job-id}

The inclusion of the collection job ID (see Section 4.7 of [DAP]) differs from other batch modes where the Collector does not need to provide any additional information to a Leader.

4.2. Batch Mode Parameters

This batch mode uses the same parameters as the leader-selected batch mode (Section 5.2 of [DAP]). That is, the payload Query.config is empty. Both the PartialBatchSelector.config and the BatchSelector.config contains a batch ID that is assigned by the Leader.

5. Minimum Privacy Budget Collection Job Extension

The minimum privacy budget collection job extension (see Section 4.6.2 of [DAP]), codepoint 0xTBD, allows a Collector to inform Aggregators of a minimum value for the privacy budget report extension (see Section 3) that it expects to be included in the collection job.

The format of this extension is a 32-bit network-endian encoding of an integer in units of micro-epsilon. This is identical to the format of the privacy budget report extension (Section 3).

This extension is defined primarily as a safeguard. In the absence of this extension, Aggregators could determine the amount of privacy budget that was expended by every report and generate noise based on the minimum value across all reports.

That approach is potentially error prone. If a Collector accidentally includes a report with a much lower budget, the Aggregate it receives would have more noise added than expected. The collection job extension effectively sets a cap on the noise that might be added.

Aggregators can use this extension in one of two ways:

* The value in the collection job extension directly determines the magnitude of the noise that is added to the aggregate.

* The value is only used to filter reports and the minimum value of the privacy budget report extension across all accepted reports determines the magnitude of added noise.

Either approach maintains differential privacy guarantees. The latter can result in adding less noise in the case that the Collector provides a low value, but is more complex to implement. There is also no way provided to indicate to the Collector that less noise was added than they might have planned.

6. Collector Identity Task Extension

The collector identity task extension (see Section 4.2.2 of [DAP]), codepoint 0xTBD, binds the task -- and all reports submitted to that task -- to a single Collector.

This extension does not specify how to encode the identity of the Collector. Different uses of DAP can choose an encoding that best suits the situation.

The Attribution API has its own understanding of how to encode the identity of the Collector. The value is a UTF-8-encoded string of the registrable domain from the intermediary site tuple (if there is an intermediary site) or the conversion site tuple (where there is no intermediary site).

6.1. Collector Identity and HPKE Configuration

Regardless of how the Collector is identified, if an identity is included, the Leader and Helper MUST have a process for validating the HPKE configuration they use to encrypt aggregate shares for the Collector.

That process MUST provide confirmation that the identified entity authorizes the HPKE configuration. This does not depend on active proof of possession for the corresponding private key [SIGMA] but rather an affirmation that the public key is approved by the identified entity.

One option for Collector identification is to use a URL, following the pattern used for the Leader and Helper; see Section 4.2 of [DAP]. If the URL uses an authenticated protocol, such as HTTP with the "https" scheme [RFC9110], retrieving an HPKE configuration from that URL (or similarly authenticated resources that are referenced from the response) provides the necessary authorization for the included key.

The Attribution API does not define a process for authorizing a Collector HPKE configuration based on the encoded Collector identity.

7. Budget Source Task Extension

The budget source task extension (see Section 4.2.2 of [DAP]), codepoint 0xTBD, binds a task -- and all reports submitted to that task -- to a single source of privacy budget.

This extension does not specify how to encode the identity of this entity. Different uses of DAP can choose an encoding that best suits the needs of the differentially private usage.

The Attribution API has its own understanding of how to encode the identity of the budget source. The value is a UTF-8-encoded string of the registrable domain from the conversion site tuple.

8. Security Considerations

Security factors specific to each extension are covered in the respective sections.

Use of DAP is subject to the security considerations of DAP (Section 8 of [DAP]) and the VDAF that is in use (Section 9 of [VDAF].

9. IANA Considerations

This document registers a new batch mode in the "DAP Batch Mode Identifiers" registry established in Section 9.2.1 of [DAP].

Value	Name	Reference
TBD	collector_selected	Section 4

Table 1: DAP Match Mode

This document registers task extensions in the "DAP Task Extension Identifiers" registry established in Section 9.2.2 of [DAP].

New task extensions are tabulated in Table 2.

Value	Name	Reference
TBD	collector_identity	Section 6
TBD	budget_source	Section 7

Table 2: Task Configuration Extensions

This document registers a DAP report extension in the "DAP Report Extension Identifiers" registry established in Section 9.2.3 of [DAP].

The new report extension registration is tabulated in Table 3.

Value	Name	Reference
TBD	privacy_budget	Section 3

Table 3: DAP Extensions

This document registers a collection job extension in the "DAP Collection Job Extension Identifiers" registry established in Section 9.2.4 of [DAP].

The new collection job extension is tabulated in Table 4.

Value	Name	Reference
TBD	min_dp_budget	Section 5

Table 4: Collection Job Extensions

10. References

10.1. Normative References

[DAP] Geoghegan, T., Patton, C., Pitman, B., Rescorla, E., and C. A. Wood, "Distributed Aggregation Protocol for Privacy Preserving Measurement", Work in Progress, Internet-Draft, draft-ietf-ppm-dap-17, 30 January 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-ppm-dap-17>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/rfc/rfc8174>.

[VDAF] Barnes, R., Cook, D., Patton, C., and P. Schoppmann, "Verifiable Distributed Aggregation Functions", Work in Progress, Internet-Draft, draft-irtf-cfrg-vdaf-18, 30 January 2026, <https://datatracker.ietf.org/doc/html/draft-irtf-cfrg-vdaf-18>.

10.2. Informative References

[ATTR] Paseltiner, A., Leiserson, A., Case, B., Savage, B., Harrison, C., and M. Thomson, "Attribution API", 14 January 2025, <https://w3c.github.io/attribution>.

[ATTR-DP] Tholoniat, P., Caulfield, A., Cavicchioli, G., Chen, M., Goutzoulias, N., Case, B., Cidon, A., Geambasu, R., Lcuyer, M., and M. Thomson, "Beyond Per-Querier Budgets: Rigorous and Resilient Global Privacy Enforcement for the W3C Attribution API", 14 October 2025, <https://arxiv.org/abs/2506.05290>.

[CAP-URL] Tennison, J., "Good Practices for Capability URLs", 18 February 2014, <https://www.w3.org/TR/capability-urls/>.

[DP] Dwork, C. and A. Roth, "The Algorithmic Foundations of Differential Privacy", Emerald, Foundations and Trends in Theoretical Computer Science vol. 9, no. 3-4, pp. 211-487, DOI 10.1561/0400000042, August 2014, <https://doi.org/10.1561/0400000042>.

[HTTP] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "HTTP Semantics", STD 97, RFC 9110, DOI 10.17487/RFC9110, June 2022, <https://www.rfc-editor.org/rfc/rfc9110>.

[RFC9110] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "HTTP Semantics", STD 97, RFC 9110, DOI 10.17487/RFC9110, June 2022, <https://www.rfc-editor.org/rfc/rfc9110>.

[SHUFFLE] Erlingsson, ., Feldman, V., Mironov, I., Raghunathan, A., Song, S., Talwar, K., and A. Thakurta, "Encode, Shuffle, Analyze Privacy Revisited: Formalizations and Empirical Evaluation", 10 January 2020, <https://arxiv.org/abs/2001.03618>.

[SIGMA] Krawczyk, H., "SIGMA: The ‘SIGn-and-MAc’ Approach to Authenticated Diffie-Hellman and Its Use in the IKE Protocols", Springer Berlin Heidelberg, Lecture Notes in Computer Science pp. 400-425, DOI 10.1007/978-3-540-45146-4_24, ISBN ["9783540406747", "9783540451464"], 2003, <https://doi.org/10.1007/978-3-540-45146-4_24>.

[SITE] WHATWG, "HTML - Living Standard", 26 January 2021, <https://html.spec.whatwg.org/#site>.

[WEB-PRIV] Berjon, R. and J. Yasskin, "Privacy Principles", 24 November 2025, <https://w3ctag.github.io/privacy-principles/>.

Acknowledgments

Roxana Geambesu noted that a binding to site identity (Section 6) was an important component of a robust differential privacy system design for the Attribution API. David Cook provided useful feedback about the design and document. Chris Patton provided helpful input on how to integrate with the DAP architecture.

Author’s Address

Martin Thomson
Mozilla
Email: mt@lowentropy.net