DAP updates and open issues

IETF 120 – PPM

Changes since IETF 118

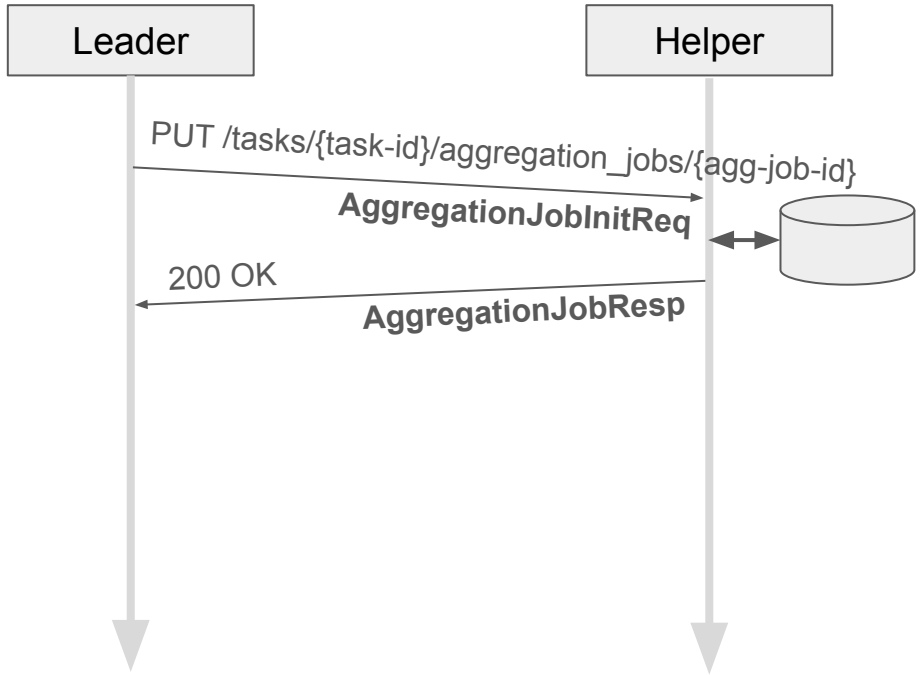
- draft-ietf-ppm-dap-09: `fixed_size` queries: Make max batch size optional
- draft-ietf-ppm-dap-10: Incorporate feedback from early HTTPDIR review (Mark Nottingham)
- draft-ietf-ppm-dap-11: Result of consensus call following discussion at the interim: remove support for multi-collection. DAP no longer supports heavy hitters via Poplar1.

PR #563: Remove `max_batch_size` task parameter

- Context: The `fixed_size` query type is so named because it has both a minimum and maximum batch size. The maximum batch size turned out to be slightly annoying to implement, as well as not particularly useful. In fact, we made it optional in draft-ietf-ppm-dap-09.
- Proposal: Remove the maximum batch size and rename the query type to `leader_selected`.
  - Question (semantics): does "query type" refer to:
    - the restrictions on how reports may be mapped to batches; or
    - the conditions that must be met when a batch is collected?

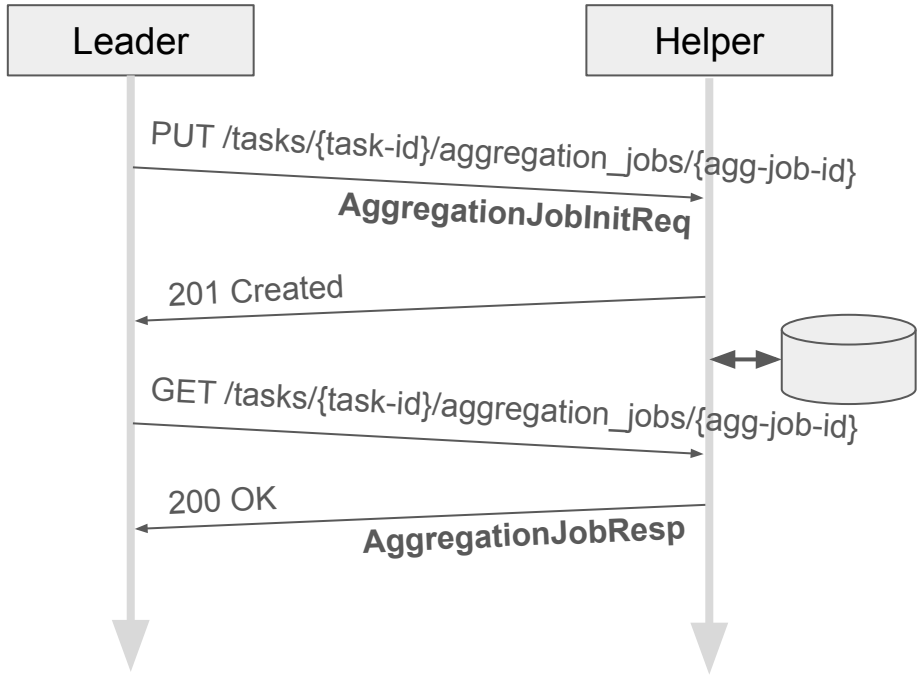
PR #564: Make aggregation asynchronous

- Context: Aggregation is "synchronous": each HTTP request blocks until Helper completes VDAF preparation and commits states changes (record report IDs for anti-replay, update aggregate share).
    - Deployment experience: a small number of requests take 10+ seconds to resolve or need to be aborted and retried.



PR #564: Make aggregation asynchronous

- Context: Aggregation is "synchronous": each HTTP request blocks until Helper completes VDAF preparation and commits states changes (record report IDs for anti-replay, update aggregate share).
  - Deployment experience: a small number of requests take 10+ seconds to resolve or need to be aborted and retried.
- Proposal: Allow Helper to process job asynchronously
  - Helper responds to PUT with 201 Created
  - Leader sends GET to poll for result



PR #566: Document deviations from RFC 8446

Context: DAP implicitly overloads RFC 8446 (TLS 1.3) presentation language. The following is for fields with fixed constants (section 3.7). We use it for specifying how a message is formatted ("the Helper replies with the following message")

```
struct {
  PrepareStepState prepare_step_state = 2; /* reject */
  ReportId report_id;
  ReportShareError report_share_error;
} PrepareStep;
```

Proposal: Make this syntax explicit with new `struct variant` notation.

```
struct variant {
  PrepareStepState prepare_step_state = 2; /* reject */
  ReportId report_id;
  ReportShareError report_share_error;
} PrepareStep;
```

PR #567: Drop `task_id` param from HPKE config endpoint

- Context: DAP Aggregators may have per-task HPKE configurations to reduce risk of key compromise. To support this, Clients indicate the task ID when requesting the configs.
  - Complicates applications that need anonymity (e.g., DAP over OHTTP): Aggregators knows which tasks a Client participates in.
  - This feature is not used in any known implementation.
- Proposal: Remove the optional `task_id` parameter from the `/hpke_config` endpoint, forcing an Aggregator endpoint to use the same set of HPKE configs for all tasks.

PR #568: Content type versioning

- Context: the content type for an HTTP request (eg., `"application/dap-aggregation-job-init-req"`) is meant to convey how the request body is parsed. However:
  - "DAP v2" will likely be wire-incompatible with the current protocol
  - Drafts of the current protocol may be wire-incompatible
- Proposal: Clarify that major revisions to DAP will use new media types (e.g., `"dap"` → `"dap2"`). Also, note that media types have an optional parameter that can be used to convey the draft number (may be useful for debugging).