Candidate interoperability target for PPM

Tim Geoghegan
Internet Security Research Group (ISRG)
timg@letsencrypt.org

PPM @ IETF 113, March 25 2022

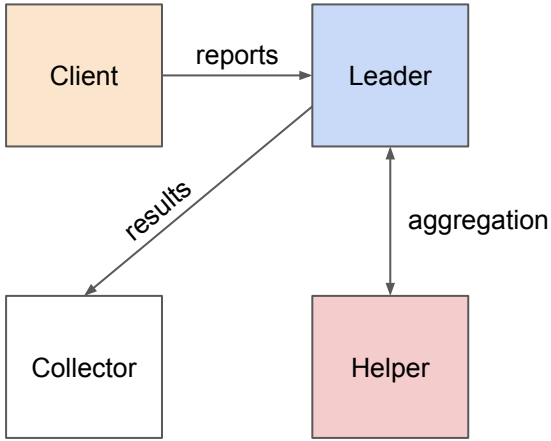
Status of draft-gpew-priv-ppm-01

Privacy Preserving Measurement Recap

- draft-gpew-priv-ppm specifies PPM, a protocol framework for privately computing aggregate functions
  - Based on Prio [CGB17], but since generalized to work with any type of Verifiable Distributed Aggregation Function (VDAF)
  - Requires multiple non-colluding servers (aggregators) for meaningful privacy
- Targets a variety of motivating use cases
  - Simple statistical aggregates (mean, median, sum, histograms)
  - Relationships between multiple values (correlation)
  - Heavy hitters (common strings)

Protocol Overview

- PPM is three "sub-protocols" executed simultaneously
  - *Upload Flow* – Client pushes report (encrypted input shares) to the Leader
  - *Aggregate Flow* – Leader and Helper(s) interact to verify and aggregate reports and compute aggregate shares
  - *Collect Flow* – Collector pulls encrypted aggregate shares from the Leader
- Each protocol is built on well-established HTTP semantics and protocol design (e.g., RFC8555)



PPM Status

- draft-gpew-priv-ppm-01 is close to being fully implementable and realizes the WG's primary deliverables:

> … The Privacy Preserving Measurement (PPM) work will standardize protocols for deployment of these techniques on the Internet. This will include mechanisms for:
>
> - Client submission of individual measurements, potentially along with proofs of validity ✅ (*Upload Flow*)
> - Verification of validity proofs by the server(s), if sent by client ✅ (*Aggregate Flow*)
> - Computation of aggregate values by the server(s) and reporting of results to the entity taking the measurement ✅ (*Collect Flow*)
>
> The WG will deliver one or more protocols which can accommodate multiple PPM algorithms. ✅ (*VDAF*)

Question: *is this draft a good starting point for the WG?*

The interop target proposal

Beyond draft-01

- Cloudflare and ISRG teams are working to get independent implementations of PPM interoperating
- We have been converging on an "interop target"
  - https://github.com/abetterinternet/ppm-specification/pull/179
- Goals:
  - Run PPM with Prio3 VDAFs
  - Hammer out edge cases and error handling
  - Discover which parts of draft-01 are difficult to implement or operate
  - Gain experience and gather data to inform discussions in the WG

Where are we today?

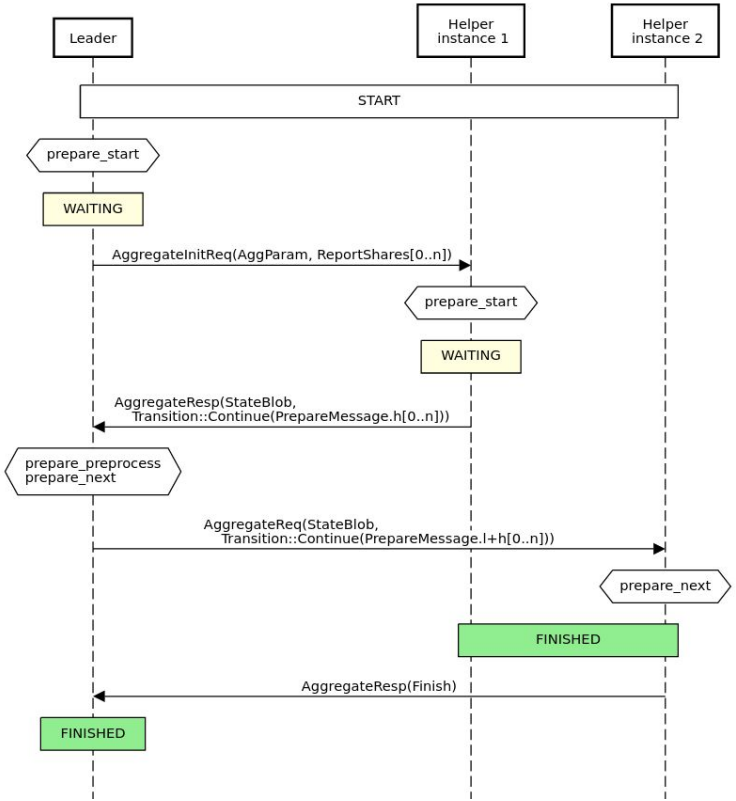
- VDAF specification of Prio3 is mature enough to implement, with test vectors in libprio-rs
  - https://datatracker.ietf.org/doc/draft-patton-cfrg-vdaf/
  - https://github.com/divviup/libprio-rs
- ISRG has a toy, incomplete implementation of PPM protocol draft-01
  - Protocol can be implemented end-to-end, though edge cases abound
  - Not tested against any other implementations, no persistence
  - https://github.com/divviup/ppm-prototype
- Cloudflare and ISRG teams working on deployable implementations
  - ISRG's Janus: https://github.com/divviup/janus
  - Cloudflare code coming soon!

Aggregate sub-protocol

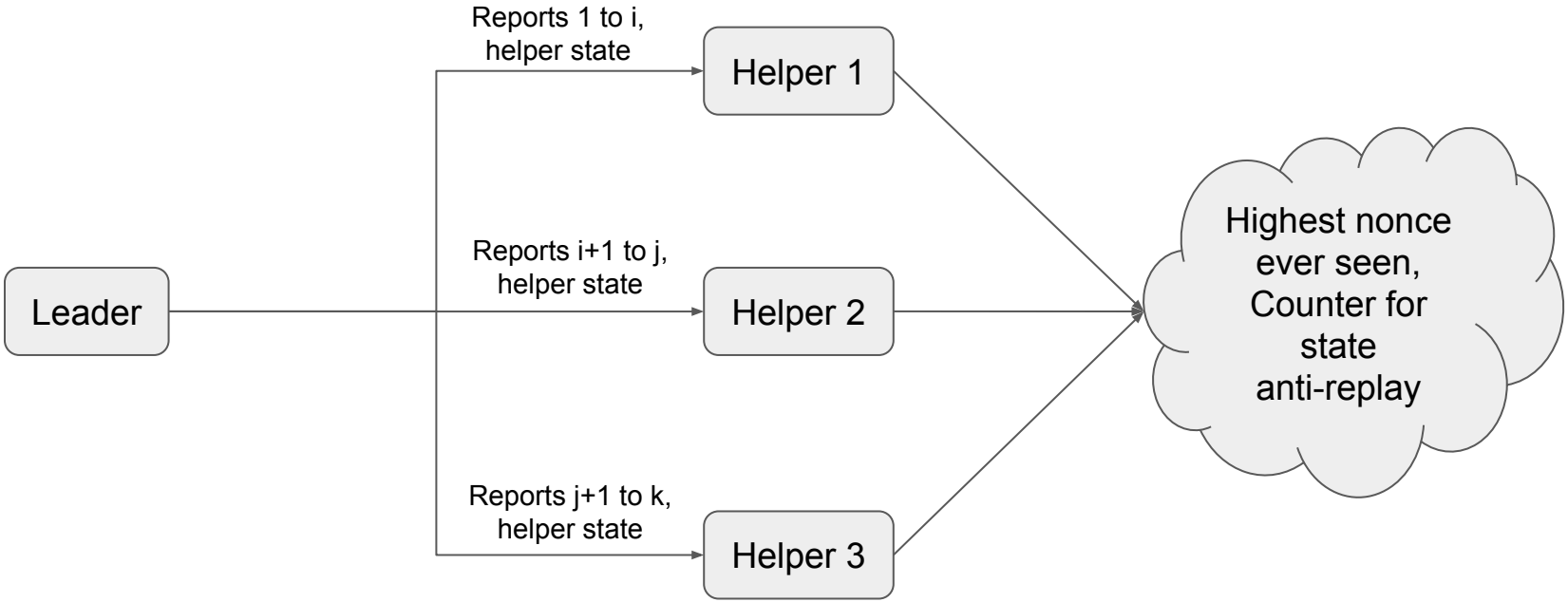
- Aggregate flow is the meat of the complexity and consists of executing a VDAF
  - *Preparation* means turning input shares into output shares that can be summed into aggregates
- Aggregation in draft-gpew-priv-ppm-01 lacks sufficient detail to be implemented
- State machine and messages have changed to catch up with VDAF messages in draft-patton-cfrg-vdaf-01

Farewell to the helper state blob

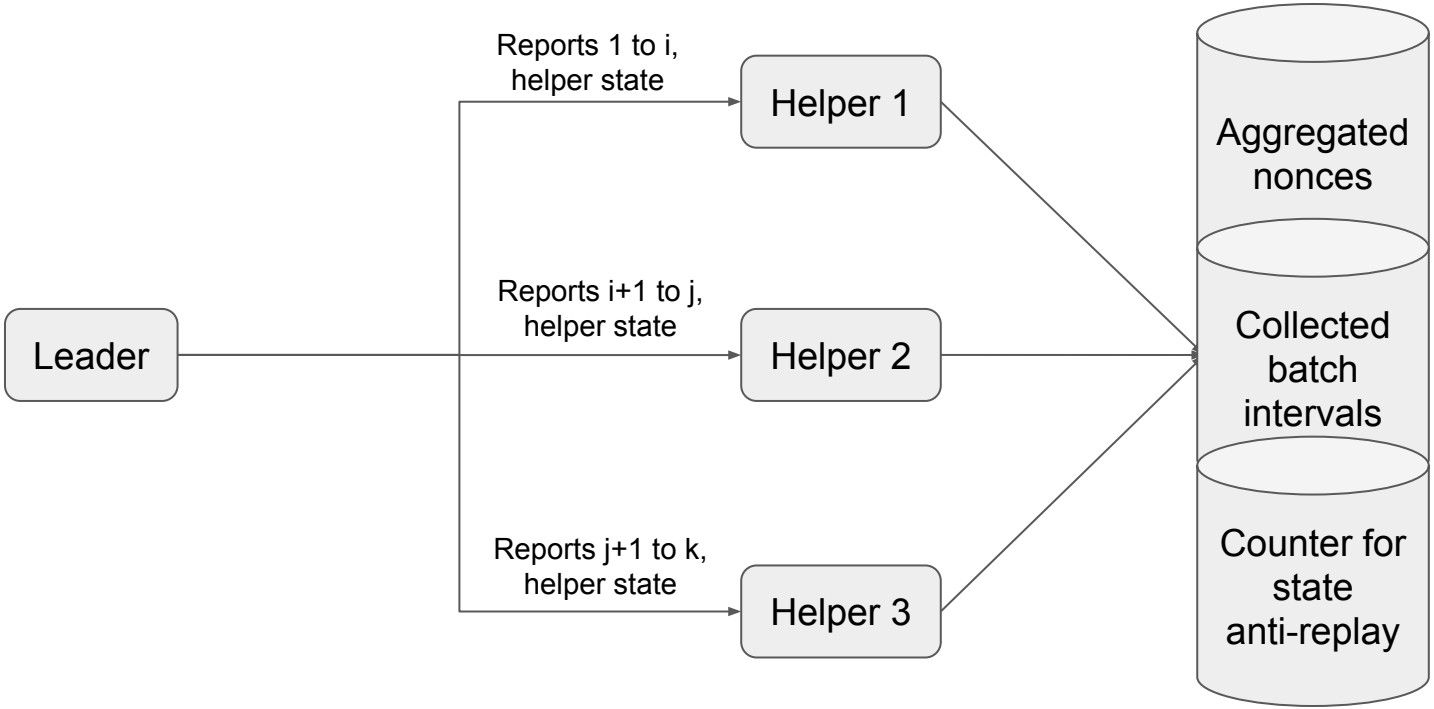
- Helper state blob was stored by leader
  - Goal was to eliminate storage requirements for the helper
- But the state blob requires:
  - Extra bandwidth
  - Encryption
  - Anti-replay (helper has to store a counter)
- And makes parallelizing aggregation difficult
  - What if leader sends the helper the same state blob twice? Which state modification wins?



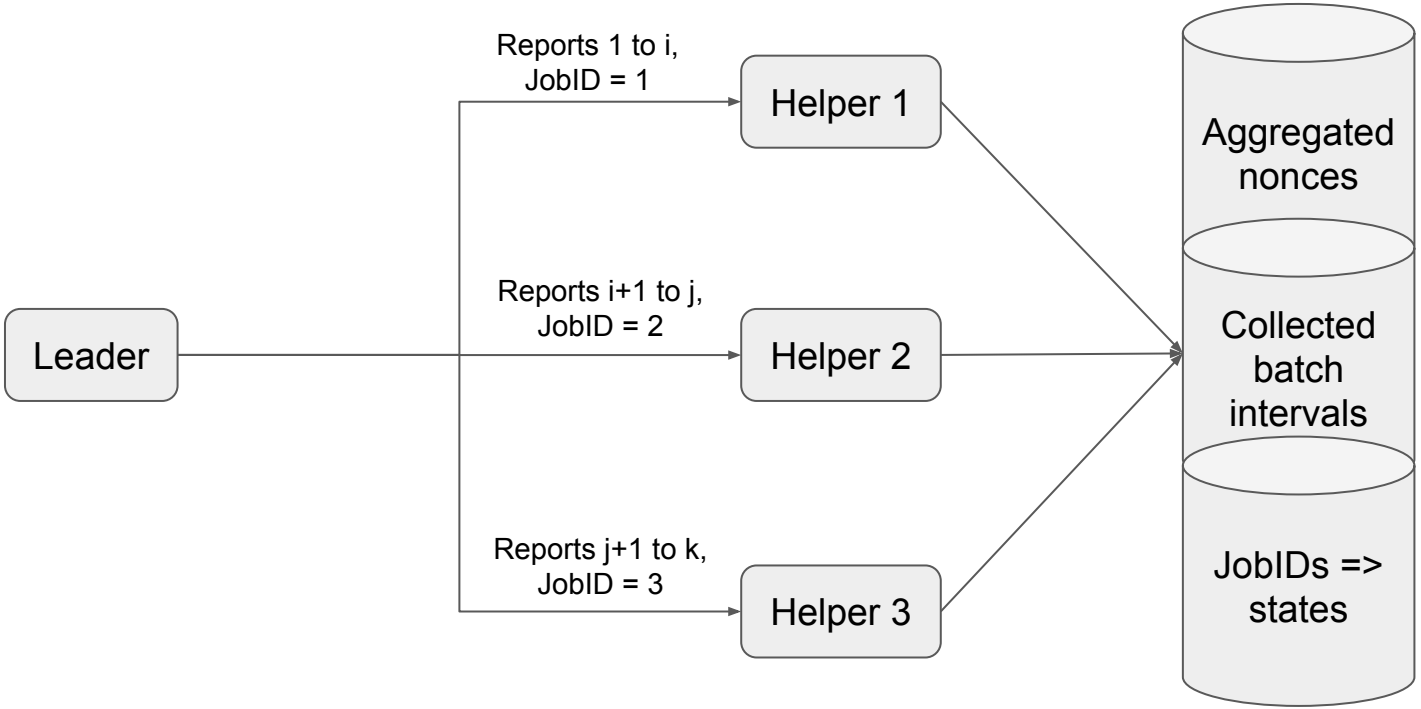
Anti-replay in draft-00



Anti-replay in draft-01



Anti-replay and aggregation job IDs in interop target



Aggregator mutual authentication

- Messages exchanged between aggregators must be authenticated
- Interop target includes an HMAC-SHA256 tag in aggregate protocol messages
- Aggregators share an HMAC secret
- TLS server auth isn't scoped right
- Akin to negotiation and rotation of shared, secret verification parameters

```
struct {
      TaskID task_id;
      AggregationJobID job_id;
      AggregateReqType msg_type;
      select (msg_type) {
      agg_init_req:  AggregateInitReq;
      agg_cont_req:  AggregateContinueReq;
      }
      opaque tag[32];
} AggregateReq;
```

Survey of PPM request authentication

Interaction	Design requirement	Specified mechanism
Client ⇒ leader (report uploads)	1. Confidentiality 2. Server authentication 3. Optional client authentication	1. HPKE encryption to each aggregator 2. HPKE config fetched over TLS 3. Out-of-scope
Leader ⇔ Helper (aggregation protocol)	Mutual authentication	HMAC-SHA256 with shared secret
Collector ⇒ Leader (collect request)	Nothing (yet)	Nothing (out-of-scope?)
Aggregator ⇒ collector (aggregate shares)	1. Confidentiality 2. No authentication (yet)	1. HPKE encryption of aggregate share 2. Nothing (out-of-scope?)

Unbalanced contributions spoil aggregations

- For values $v_1 \ldots v_n$, each sharded into shares for aggregators a and b,

$$v_a + v_b \equiv v$$

$$\sum_{i=1}^{n} v_{i,a} + \sum_{i=1}^{n} v_{i,b} \equiv \sum_{i=1}^{n} v \equiv a$$

- Everything is *mod p*
- What if >=1 of the shares is dropped in one aggregator?
- a is $\in [0..p]$ and $v_{i,a}$ and $v_{i,b}$ are random values $\in [0..p]$

$$\sum_{i=1}^{n} (v_{i,a} + v_{i,b}) - v_{k,b} \equiv random\ garbage$$

Aggregation share checksumming

Focus on detecting and measuring mismatched aggregations

```
checksum = XOR(
    SHA-256(nonce_1),
    SHA-256(nonce_2),
    …,
    SHA-256(nonce_n),
)
```

```
struct {
  TaskID task_id;
  Interval batch_interval;
  uint64 report_count;
  opaque checksum[32];
  opaque tag[32];
} AggregateShareReq;
```

Open questions

- Should PPM specify message authentication?
- How are shared, secret parameters negotiated and rotated?
- Lifecycle of reports and preparation states
  - How do aggregators know when it's safe to delete data?
  - Do we need a commit phase during preparation protocol?