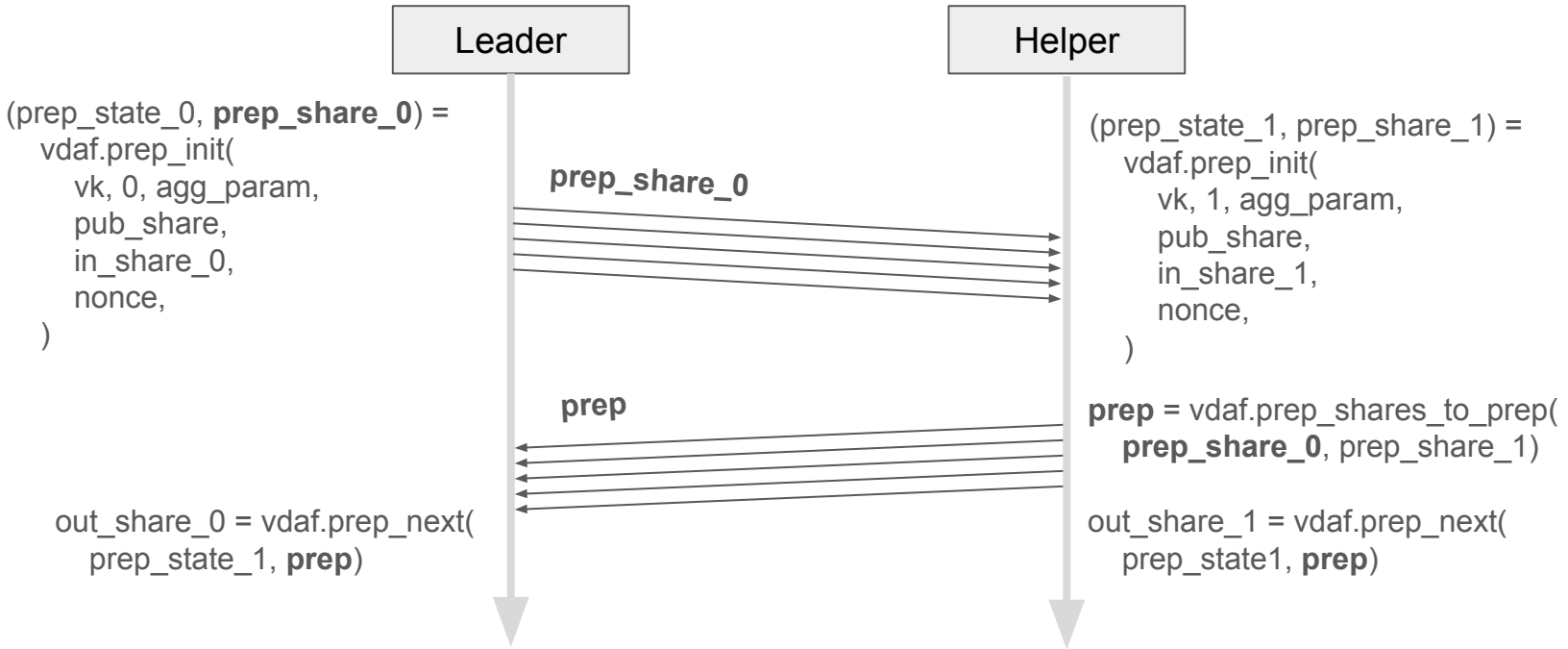
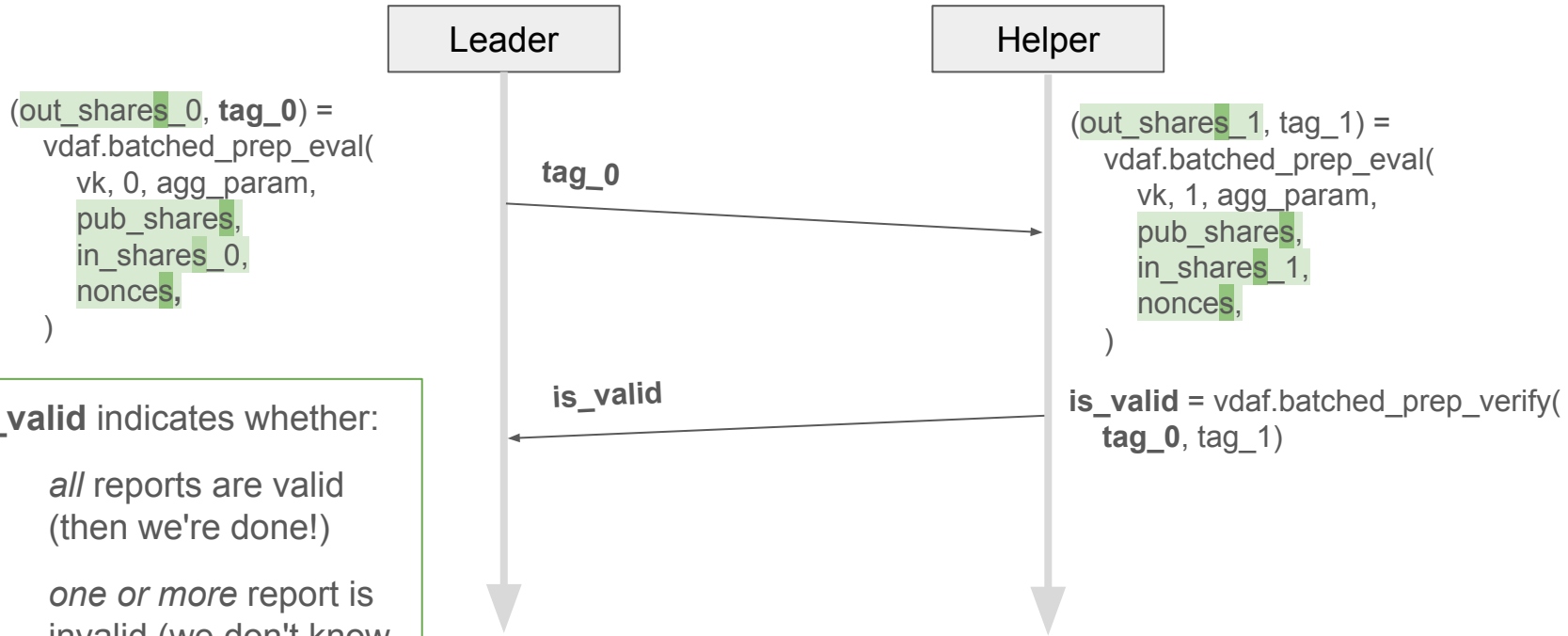
# Batched VDAF preparation

IETF 120 – PPM – Christopher Patton

# Prio3 and Poplar1 do preparation ***per-report***



# New VDAFs support ***batched*** preparation




**is_valid** indicates whether:

- *all* reports are valid (then we're done!)
- *one or more* report is invalid (we don't know which ones)

# Whisper (Rathee et al., IEEE S&P 2024)

- Same functionality as Prio3
- Transforms an FLP into a proof system ***silently verifiable*** (hence "Whisper"): last step of preparation is linear, allowing it to be aggregated across reports (into the **tag**)
  - Idea: Client sends to each Aggregator the **verifier share** generated by its co-Aggregator: Aggregators arrange to verify that the Client sent the right value (similar to how joint randomness is checked in Prio3)
- Bandwidth cost is the same, but the verifier share is uploaded by Client rather than the Leader
  - $O(N^{1/2})$ bits, where $N$ is the input length
  - Issue #130: bandwidth is often cheaper for Clients than for Aggregators

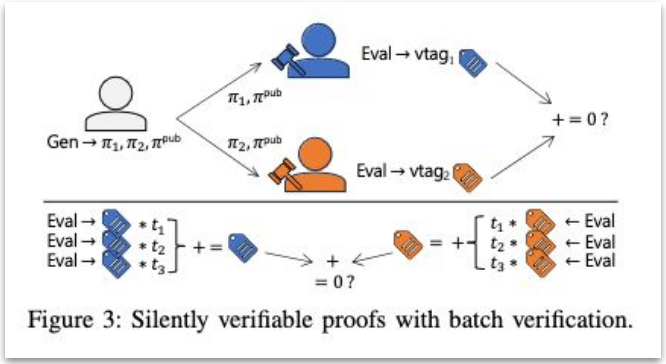
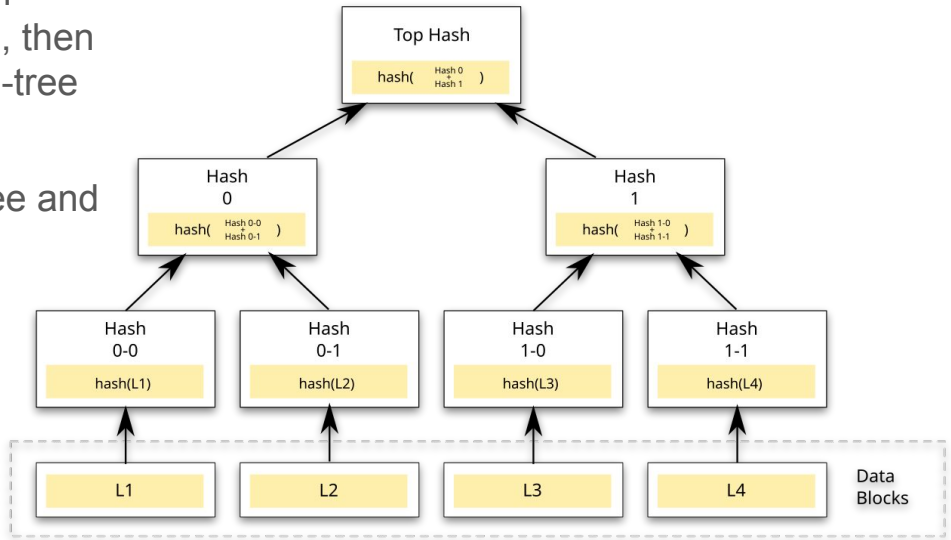


Figure 3: Silently verifiable proofs with batch verification.

# PLASMA (Mouris et al., PETS 2024)

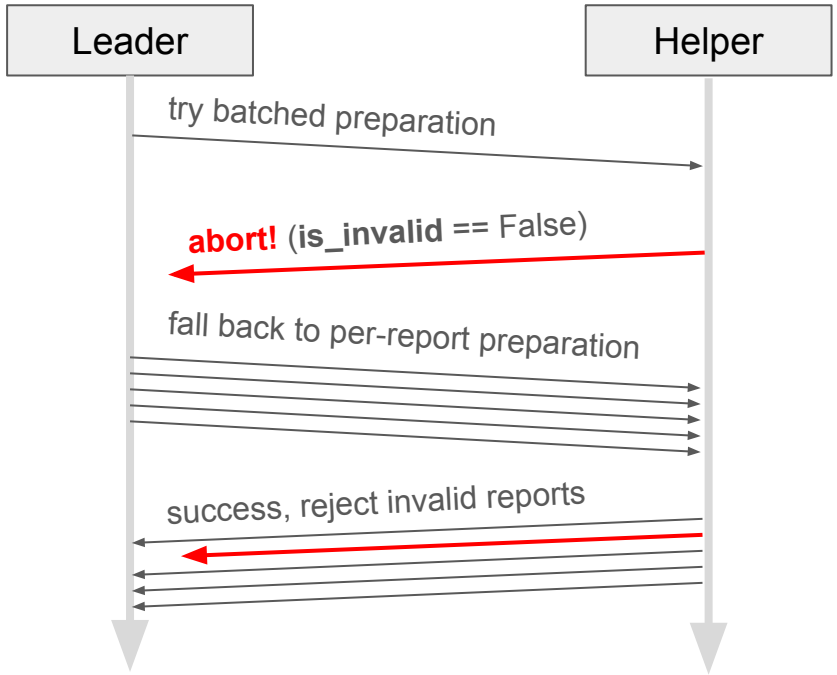
- Same functionality as Poplar1
- Transforms IDPF into ***verifiable*** IDPF: each Aggregator produces a **proof string** for each report. If they compute the same proof string, then they are guaranteed to have traversed a sub-tree with at most one non-zero path.
    - Idea: Put proof strings into a Merkle tree and set the **tag** to the root of the tree.




https://en.wikipedia.org/wiki/Merkle_tree

# PR #565: Support batched preparation DAP

- Allow retrying after batched verification failure (**is_valid** == False). Either:
  - retry with smaller reports/job; or
  - fall back to per-report preparation
- Whisper **does not save bandwidth cost for DAP as-is**, even when all reports are valid
  - Due to Leader-Upload architecture: Leader proxies report share to Helper
  - In a Split-Upload architecture (ruled out at IETF 113) the Client uploads report share to Helper directly



# Is batched preparation a desirable property of VDAFs?

- Huge savings in Aggregator⇔Aggregator bandwidth:
  - Under ideal conditions (all reports are valid)
  - With certain architectures (in particular, not for DAP)
    - Revisit Split-Upload in future drafts?
- Denial-of-Service attack: clients can cheaply force fallback to per-report preparation by submitting invalid reports
  - Mitigations:
    - Generate reports in a trusted execution environment? May be possible for Android/iOS but probably harder in the browser.
    - Client rate limiting, a la draft-priebe-ppm-dap-reportauth?