ALTO New Transport

draft-ietf-alto-new-transport-03
draft-schott-alto-new-transport-pull-00 (Question)
draft-schott-alto-new-transport-push-00

Presenter: Roland Schott

November 11, 2022

IETF 115

Outline

- Recap of discussions and WG decisions
- Major changes from IETF 114
- Discussions and remaining issues to finalize

Review References

- Four excellent, early-HTTP-expert/AD reviews
  - [MT] Martin Thompson (July 11)
    - https://mailarchive.ietf.org/arch/msg/alto/sa1Pv7jmTfBF3TbGuJr_PffIjXg/
  - [SD] Spencer Dawkins (July 15)
    - https://datatracker.ietf.org/doc/review-ietf-alto-new-transport-01-artart-early-dawkins-2022-07-15/
  - [MN] Mark Nottingham (July 17)
    - https://mailarchive.ietf.org/arch/msg/alto/D84S0qLbgtpL0-jf93gNPS3NUJE/
  - [MD]
    - Comments by AD Martin Duke at IETF 114

Recap: IETF 114 Reviews/Discussions: Finalizing Op Mode(s)

- Four potential operational modes to transfer updates to a resource from the ALTO server to the ALTO client:
  - Client pull
  - Client long pull
    - Blocking in HTTP/1.x
    - Allow request on next seq number
  - Server push
    - PUSH_PROMISE (HTTP/2-3)
  - Server put
    - ALTO server as HTTP client

```
{
  [
    {"seq":        101,
     "media-type": "application/alto-costmap+json",
     "tag":        "a10ce8b059740b0b2e3f8eb1d4785acd42231bfe" },
    {"seq":        102,
     "media-type": "application/merge-patch+json",
     "tag":        "cdf0222x59740b0b2e3f8eb1d4785acd42231bfe" },
    {"seq":        103,
     "media-type": "application/merge-patch+json",
     "tag":        "8eb1d4785acd42231bfecdf0222x59740b0b2e3f",
     "link":       "/tqs/2718281828459/snapshot/2e3f"}

  ],
```

Recap: How Much to Specify Ordering Control: Transport-Aware of App Semantics [MN, MT, SD reviews]

Design 1:

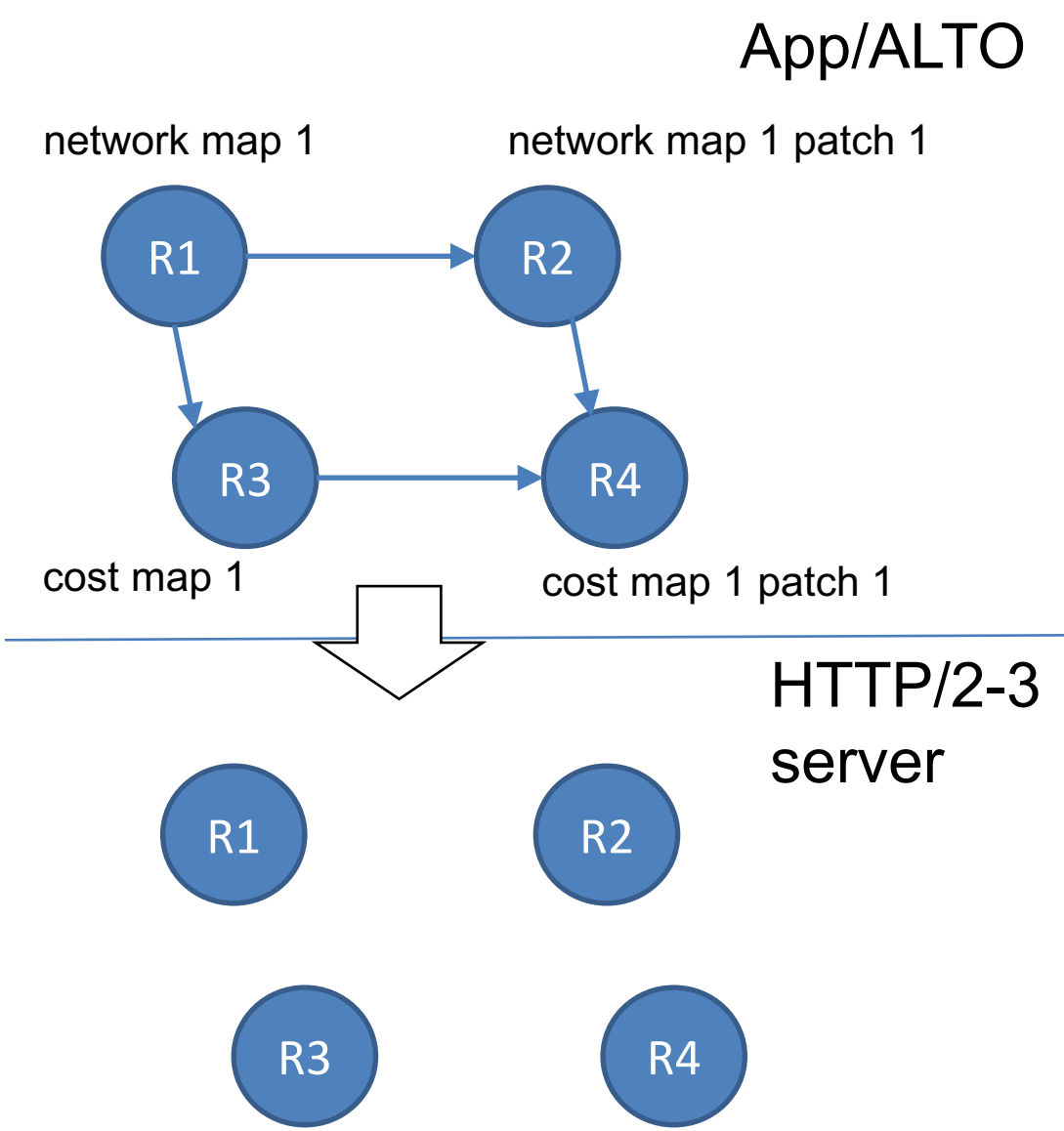
- ALTO specifies only: mapping each Ri to an independent HTTP/2-3 stream; HTTP does scheduling.
- Issue: HTTP could schedule R4, then R3, then R2 and then R1 in transmitting order

Design 2:

- ALTO specifies that server submits to the HTTP transport in DAG order: submit Ri only when what Ri depends on are finished: R1; R2/R3, R4
- Issue: Sliding window is large and transport can fit R1/R2/R3/R4 into a single window

Design 3:

- ALTO indicates the dependencies to HTTP
- Issue: HTTP client can indicate a parent in req header, but this is signaling from client to server; what we need are (1) app signaling to HTTP server, (2) multiple dependencies



Recap: How/Whether to Specify Settings

- IETF 114 version allows client to specify two (HTTP/2) control knobs on server behaviors
  - 0x02 SETTINGS_ENABLE_PUSH (a BCP14 “MUST”)
  - 0x03 SETTINGS_MAX_CONCURRENT_STREAMS (a BCP14 “must”)
- HTTP/3 changed these settings (RFC9114) [SD review]
  - SETTINGS_ENABLE_PUSH (0x02):This is removed in favor of the MAX_PUSH_ID frame, which provides a more granular control over server push. Specifying a setting with the identifier 0x02 is HTTP/3 error.
  - SETTINGS_MAX_CONCURRENT_STREAMS (0x03):QUIC controls the largest open stream ID as part of its flow-control logic. Specifying it is HTTP/3 error
- Suggestion: (1) remove them in the spec and discuss them in operations; (2) specify generic requirements statement

Recap: Other Issues

- Style guide recommends using HTTP/1.1 to specify examples
- Introduce media type detail [IANA spec]

Outline

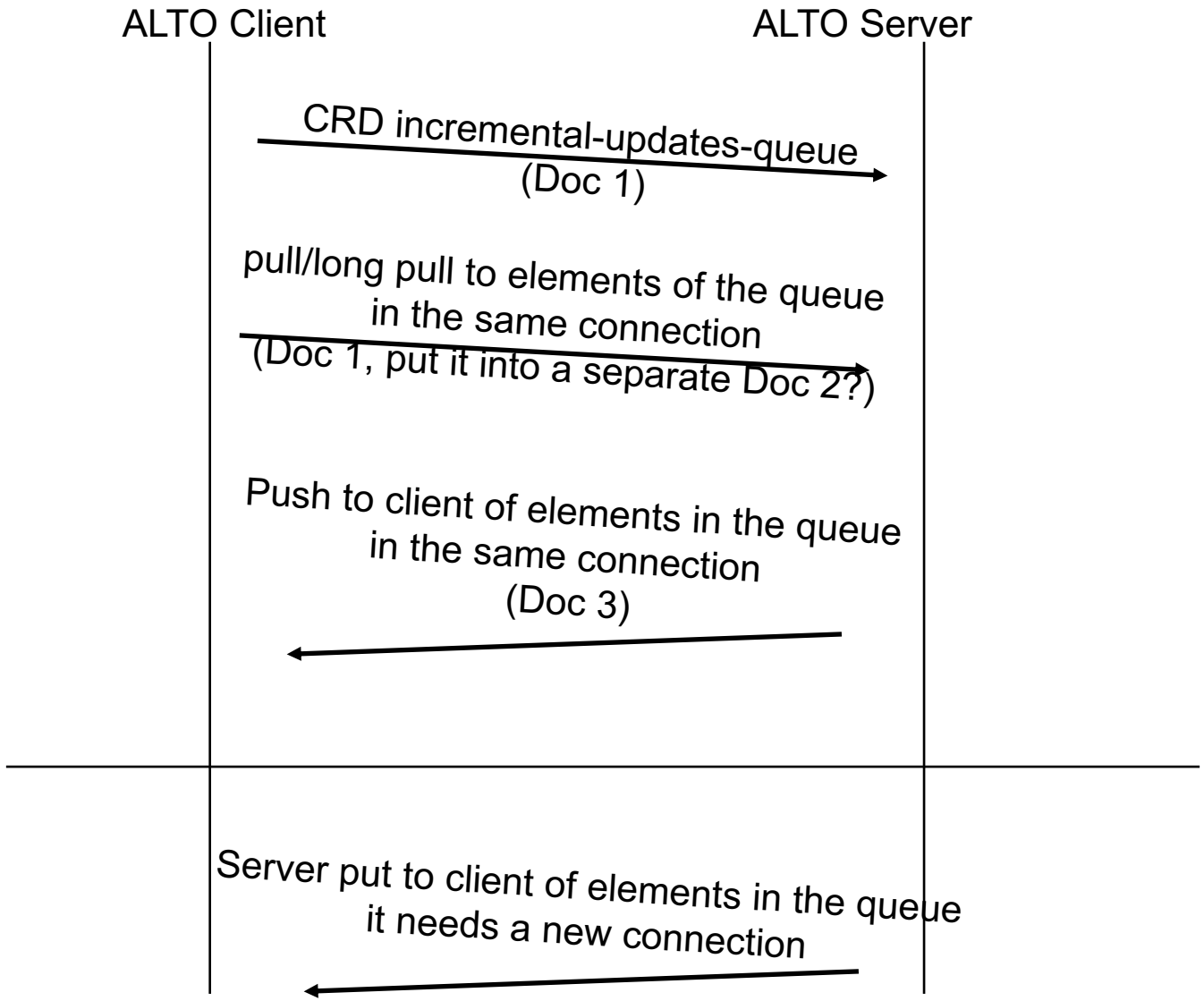
- Recap of discussions and WG decisions
- Major changes from IETF 114
- Discussions and remaining issues to finalize

Major Structure Changes from IETF 114

- Split the single document into multiple documents
  - Principle: decompose HTTP version independent components (work across HTTP/1.x-2-3); if dependent on a version, make the component a specific document

- Main change: 3 documents
  - Doc 1: Specify common model supporting incremental updates
  - Doc 2 (question): Specify client pull, client long pull (realistic only HTTP/2-3)
    - Still part of Doc 1, need to move to a sep doc if WG agrees
  - Doc 3: Server push (HTTP/2-3)
  - leave server put as future work (because additional complexity such as NAT)

High-Level Dependency of Documents



Base Document (Doc 1): Transport Data Structure

- HTTP version independent and operational model independent specification
- Foundation:
  - Transport of an information resource organized as a sequence of incremental updates (operational log in distributed computing), called an information updates queue
  - The ALTO server is the master of the updates queue
  - CRD operations:
    - The information update queue must first be created by an ALTO client at an ALTO server
    - The ALTO client can read/delete the queue status at the ALTO server
    - The close of the connection from the ALTO client to the ALTO server results in the deletion of the queue (no persistency)
  - Sequence:
    - Only the ALTO server can write to it; client can issue commands sequential or in parallel

```
{
[
  {"seq":        101,
   "media-type": "application/alto-costmap+json",
   "tag":        "a10ce8b059740b0b2e3f8eb1d4785acd42231bfe" },
  {"seq":        102,
   "media-type": "application/merge-patch+json",
   "tag":        "cdf0222x59740b0b2e3f8eb1d4785acd42231bfe" },
  {"seq":        103,
   "media-type": "application/merge-patch+json",
   "tag":        "8eb1d4785acd42231bfecdf0222x59740b0b2e3f",
   "link":       "/tqs/2718281828459/snapshot/2e3f"}

],
```

  - The structure of the queue is an array of elements, where each element has the basic fields:
    - Sequence number
      - Must be incremental, no gap, up to 64 bits; if reach limit, wrap to 0
    - Media-type
    - Tag

Client Pull Document (Doc 2): Client Read Updates

- Client -> server: Very simple design, only GET method on <updates-queue-uri>/<seq>
  - Allow caching and content distribution to large scale, for future extension
- Server -> client
  - Long poll support: fetch the next seq
  - Transfer scheduling (at the server/client)
    - Pull design allows client to issue concurrent pull requests, but the results can have dependency => specify the scheduling will lead to version specification; we specify them as only operational considerations
- Transfer processing (at the client) requirements
  - Specify (SHOULD include) tag to enforce correctness
- Specify that other HTTP transport control (e.g., concurrency control) should be honored, but is transparent to ALTO transport (discuss in WG)

Server Push Document (Doc 3): Server Push Updates to Clients

- Client -> server:
  - Indicate acceptance by setting up state (join receiver-set)
    - Start: put self into <updates-queue>/rs
    - Stop: delete self from the receiver set
    - Different from early doc: join when creating the updates queue, but this causes coupling of Doc 1 and Doc 3.
- Server -> client
  - Specify as PUSH_PROMISE in HTTP/2-3
  - Transfer scheduling
    - Specify only operational considerations on transfer scheduling of dependent updates as performance optimization
    - Specify that other HTTP transport control (e.g., concurrency control) should be honored, but is transparent to ALTO transport

Server Put Document (Potential Doc 4): Defer

- Can be an excellent tool for server-to-server communications
- Leave Server PUT (i.e., replicated operational log) as future work

Next Step

- The latest documents in WG git
- Will upload the latest documents to data tracker by the weekend
- Seek WG reviews