e_alto: Extensions for ALTO Aggregation

Lyle Bertz

draft-bertz-alto-aggrimpl-00

e_alto (Erlang ALTO Server) Experience

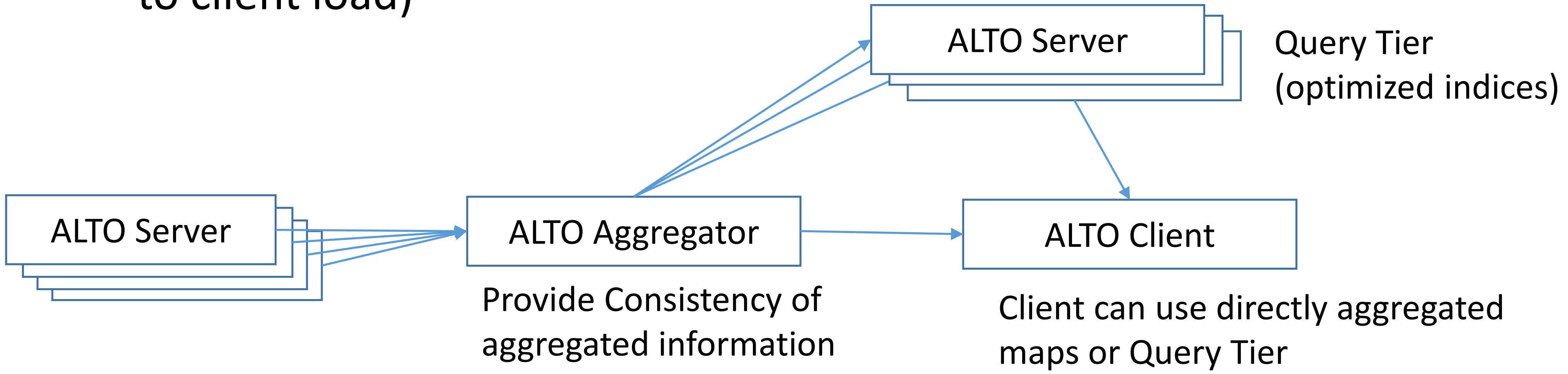
- http://github.com/lylebe/e_alto
- Status: Alpha (needs interop-93 tests + auto-gen attributes, e.g. pid)
- Supports all services including constraints
- Would have been easier if we used JSON Schema v4 ☹
  - v4 Schema extensions maintain openness, e.g. patternproperties & addtionalproperties
  - No JSON v4 Schema Validator in Erlang
- Works on finished maps in storage (local files)
- Based upon JSON files which started thoughts about aggregation in ALTO
  - It can act as an aggregator today (if there are no map overlaps)
  - Roadmap is addressing aggregation aggressively based upon the findings and designs specified here

ALTO and Aggregation

- ALTO data cannot come from only one source
    - SDN Controllers (e.g. ALTO in ODL)
    - NFV components – NFVO
    - VNFs
- ALTO Aggregation Options
    - Client Aggregation
        - Lots of work
        - What happened to ALTO being simple for the client?
    - 3rd Party Aggregation via another ALTO Server
        - Works on behalf of the client
        - Maintains Client simplicity
        - Consistent with charter item "Protocol extensions for reducing the volume of on-the-wire data exchange required to align the ALTO server and clients." as it offloads sources and clients from some load.
        - We already see where multiple ALTO Servers may be required – ALTO Cross Domain I-D

Aggregation Objectives

- Allow data origins to only care about their data
- Allow client KISS
  - Aggregate data on behalf of clients.
  - Horizontally scale (multiple aggregators) to optimize by query types, metrics, or query load (multiple aggregators that index the same data in order to scale to client load)


ALTO Server
Query Tier
(optimized indices)
ALTO Server
ALTO Aggregator
ALTO Client
Provide Consistency of aggregated information
Client can use directly aggregated maps or Query Tier

Aggregation of Multiple Maps

e_alto can provide multiple maps (from multiple sources) behind a single URI but problems were found:

- ISSUE 1: Endpoint Cost Service became a nightmare as e_alto can support searching fine (epcs) and coarse grain (costmap) structures
  - Added new constraint 'finegrain' to indicate EPCS search can only return finegrain values. Required a type specifier to be noted when the epcs result was loaded. Will add to 'meta' field.
    - Presence of 'finegrain' implies all entries are not coarse, i.e. come from a costmap (although this definition may not be strong enough)
- ISSUE 2: Quickly lost track of origins (even just file origins)
  - (Planned Solution) When querying for property origin for an endpoint the ALTO server returns
    - 'self' – ALTO Server is the origin of this data
    - URI of the IRD of the server it received its data from
    - This is a last known hop (origin) attribute – Server/Client uses recursion to find the origin server

Demand based Measurements

- e_alto track misses and hits on ecs and epcs queries which inspired an idea (and roadmap item)
  - Tracking query misses one can track what the clients are asking that is not currently available
  - If an aggregator then queries an origin with only misses it can query information sources that *could* provide the data
  - The sources *could* then look at a miss and realize this is an important question (if asked often enough)
    - This could spur ALTO Servers to use internal interfaces to spur queries
    - Maybe a new protocol can be used to negotiate measurement initiation between ALTO servers then update (via ALTO SSE) to the Aggregators

Reconciliation of Updates – Future Work

Quickly realized this is a JSON issue (not ALTO specific)

1. Use SSE, GET and Vector-Clocks for Strong Eventual Consistency for overlapping data
    1. Use JSON Merge Patch (RFC 7396) in general
    2. Use JSON Patch (RFC 6902) for small updates OR complex array updates
    3. Develop an assertion (protection) framework based on JSON Pointer (RFC 6901) – this will be the bulk of the research to complete
2. Provide vector clocks for full recovery under partial failure
    1. Ability to query and If-Modified-Since and get the deltas (SSE events) using GET
    2. Tradeoff between a full GET and parse and only grabbing the last few updates

Summary

e_alto is not beta yet but findings are promising

- Aggregation is possible
- Added ‘finegrain’ constraint for epcs to allow for a picky epcs query
    - Requires tracking this at an epcs response level
    - Should be added to meta when the entire map is finegrain
    - N/A for Costmaps
- Add origin property for Endpoints & PIDs to help find sources of information
- Use query misses (counting of them) to determine what is important to measure
    - Possible new measurement initiation negotiation between ALTO sources
    - Would provide data Clients are asking for
- Aggregator data update reconciliation
    - ALTO has good mechanisms in place (GET from ALTO standard and SSE)
    - Add Strong Eventual Consistency Concepts, e.g. Vector-Clocks, and Path protection to avoid conflicts

    All of this is in the I-D