draft-ietf-alto-multi-cost-02.txt

Updates since IETF94

S. Randriamasy

W. Roome

N. Schwan

WG feedback on 01

- Check spelling consistency
  - E.g. client vs Client
  - Clarifications on "testable-cost-type-names" and or-constraints
  - Co-existence of "constraints: true" and "testable-cost-type-names"
- Add IPv6 examples
- draft-ietf-alto-multi-cost-02.txt posted on June 13th
- Feedback since last version
- More feedback since WGLC
- Thank you all for reading and commenting

Updates in v02 - digest

- draft-ietf-alto-multi-cost-02.txt posted on June 13th
- Updates on
  - Section 3.6 Extended constraints tests:
    - Created subsections 3.6.1 to 3.6.5 to clarify new extensions
  - Updated design to ensure compatibility between "testable-cost-type-names“ and legacy ALTO Clients
  - Section 4.1.2 Accept Input Parameters
    - Clarified and updated specification according to new design
- Started Version 03
  - To further clarify constraints expression
  - Check spelling and wording consistency
  - Address new feedback

Sub-sections of3.6 Extended Constraint Tests

- § 3.6.1 Extended constraint predicates
  - Explains new constraint format
- § 3.6.2 Extended logical combination of predicates
  - Explains the combination or OR and AND constraints
- § 3.6.3 Testable Cost Types in constraints
  - Explains how client lists cost types on which it expresses constraints
    - Can be different than the requested cost-type
- § 3.6.4 Testable Cost Type Names in IRD capabilities
  - useful when a server is unable or unwilling to implement constraint tests on all cost types
- § 3.6.5 Legacy client issues
  - Explains why "testable-cost-type-names“ and "cost-constraints“ are mutually exclusive
    - See next slide

Section 4.1.1 Capabilities

- Note: Legacy ALTO (i.e. RFC7285) compatible design principle
  - A legacy ALTO Client must be able to send legacy requests to a Multi-Cost aware ALTO Server and get legacy responses as specified in RFC7285
- Updates on "testable-cost-type-names" design
  - "testable-cost-type-names" and "cost-constraints" are now mutually exclusive to prevent legacy ALTO clients from issuing constraint tests on untestable cost types.
    - If cost-constraints = true then constraints allowed on all cost-types
    - Else, constraints allowed for Multi-Cost clients on ”testable-cost-type-names”

Extension of § 4.1.2 Accept Input Parameters

- Text on "constraints" and "or-constraints" input members in section 4.1.2
  - updated according to new "testable-cost-type-names“ design
- "or-constraint" member has been corrected to
  - [JSONString or-constraints<0..*><0..*>;],
- NOTE: that this member will be corrected to
  - [JSONString or-constraints<1..*><1..*>;] in the next draft version
    - To avoid empty AND arrays neutralizing OR-arrays
      - Will be explained in v03

Section 5.6 Endpoint Cost Service example

- Added IPv6 example addresses

```
"endpoints" : {
  "srcs": [ "ipv4:192.0.2.2",
  "ipv6:2001:db8::1:0 ],
  "dsts": [
     "ipv4:192.0.2.89",
     "ipv4:198.51.100.34",
     "ipv4:203.0.113.45",
     "ipv6:2001:db8::10"
   ]
}
```

Next steps

- Finalize v03 wrt WG feedback
- E.g. recent WG feedback
  - "multi-cost-types" field in “meta” member of Multi-Cost ALTO responses
  - Given legacy ALTO (i.e. RFC7285) compatible design principle
    - A legacy ALTO client will always send legacy requests to a MC-ALTO Server and see "cost-types" in responses meta
    - Only MC ALTO Client will see "multi-cost-types"

Thank you

Back-up follows

Multi-Cost ALTO in a nutshell

- Returns ***array*** of costs instead of *scalar* cost
- Defines 'OR‘ constraints,
  - Supports **decision trade-offs** such as:
  - *"give me costs among {those PIDs/Endpoints} with either moderate ‘routingcost’ or ‘hopcount’ equal to 0*
    - For example: 'hopcount' = 0 **OR** routingcost in [5, 10]"
- Applicable service information resources:
  - Filtered Cost Map (FCM),
    - **For full Multi-Cost Map: use empty SRC & DEST**
  - Endpoint Cost Service (ECS)
- Does not introduce new media types
- Backwards compatible with legacy ALTO Clients

Example § 5.1: Filtered multi-cost map resource in IRD

```
"filtered-multicost-map" : {
  "uri" :
  "http://alto.example.com/multi/costmap/filtered",
  "media-types" : ["application/alto-costmap+json" ],
  "accepts" : ["application/alto-costmapfilter+json" ],
  "uses" : [ "my-default-network-map" ],
  "capabilities" : {
     "cost-constraints" : true,
     "max-cost-types" : 2,
     cost-type-names" : [ "num-routingcost",
                          "num-hopcount"],
}
```

Indicates that this service is MC compatible

ALTO Server allows constraints on ALL cost-types it provides

Multi-Cost ALTO clients « see » also fields in slanted blue

Legacy ALTO clients « see » only fields in black and ignore others they do not know

Example § 5.1: filtered-cost-map-extended resource in IRD

```
"filtered-cost-map-extended" : {
  "uri" : "http://alto.example.com/multi/costmap/filtered",
   "media-types" : ["application/alto-costmap+json" ],
   "accepts" : ["application/alto-costmapfilter+json" ],
   "uses" : [ "my-default-network-map" ],
   "capabilities" : {
      "max-cost-types" : 3,
      "cost-type-names" : [ "num-routingcost"
                            "num-hopcount",
                            "num-bwscore"],
      "testable-cost-type-names" : [ "num-routingcost",
                                     "num-hopcount"]
    }
},
```

Base ALTO clients DO NOT see cost-constraints allowed and thus do not express constraints

Multi-Cost ALTO clients express cost-constraints on ***testable-cost-type-names***

Example § 5.4: full MC Map - with testable cost types

```
POST multi/costmap/filtered HTTP/1.1
Host: alto.example.com
Content-Type: application/alto-costmapfilter+json
Accept: application/alto-costmap+json,application/alto-error+json
{
     "multi-cost-types" : [
       {"cost-mode": "numerical", "cost-metric": "routingcost"},
    ],
     "testable-cost-types" : [
       {"cost-mode": "numerical", "cost-metric": "routingcost"},
       {"cost-mode": "numerical", "cost-metric": "hopcount"}
    ],
     "or-constraints": [
           ["[0] le 10", "[1] le 2"],
           ["[0] le 3",  "[1] le 6"]
     ],
    "pids" : {
       "srcs" : [ ],
       "dsts" : [ ]
    }
  }
```

Motivation – use cases

- Use multiple selection metrics for endpoints and e2e paths
  - To jointly meet application needs while keeping network awareness
    - E.g. by ***jointly*** getting 'routingcost' meeting NP interests and 'bandwidth score' meeting app interests
- Save time and bandwidth on ALTO requests
  - 1 Multi-Cost transaction on N metrics rather than N on 1 metric
  - 1 Multi-Cost Map is smaller than N Cost Maps
- Consistency of metric values
  - Different cost-types may change at different paces
  - For multi-variate optimization
- Enrich filtering constraints to represent compromises, e.g.
  - *select paths with moderate 'routingcost' OR null 'hopcount'*

Multi-Cost transactions

- Multi-Cost Requests and responses convey an *Array* of costs
  - Array may contain any Cost Mode combination
    - Requested Cost-types array

      `["num-routingcost", "ord-hopcount", "string-status"]`
    - Taking values:

      `[23, 6, "medium"]`
  - **RULE:** cost values for each Source/Destination pair MUST be provided in the same order as in the array of Multi-Cost Types

Design

- Multi-Cost filtering constraints
  - Combine AND and OR operators
  - Are applied to cost-types present in value request
    - NOTE: [draft-lee-alto-app-net-info-exchange] proposes to use constraints on metrics not present in value request