The ALTO Path Vector Extension

draft-ietf-alto-path-vector-08

K. Gao[3] Y. Lee[1] S. Randriamasy[2] Y. R. Yang[5] J. Zhang[4,5]

[1]Huawei [2]Nokia Bell Labs [3]Sichuan University [4]Tongji University [5]Yale University

July 25, 2019 @ IETF 105

Updates

In -07

- Finalize the specification
  - Revert the cost type design to -05
  - Add a new IRD capability and request field to negotiate properties
  - Add a new property to support use cases other than coflow scheduling
- Re-categorize some references (based on the IESG statement)

In -08

- Better clarify the necessity of the PV draft
  - Motivated by new usage scenarios from both application requirements (high-speed data transfers) and network innovations (in-network computation and storage)
  - Provide correlations of network paths, in addition to preferences of network paths (which motivates the base protocol)
- Add two more use cases
- Clarify some design decisions including concepts (ANE, Part Resource ID), and procedures (property negotiation, incremental updates)

Cost Type

In -05:

- cost mode: array, cost metric: ane-path

In -06:

- cost mode: path-vector, cost metric: maxresbw

Since -07: (same as -05)

- cost mode: array, cost metric: ane-path

This cost type better conform to the ALTO cost type design principle: mode - interpretation, metric - semantics.

Property Negotiation

In -05:

- Property query is handled by the Unified Property Map extension

In -06:

- Property is encoded as the cost metric

Since -07:

- Available properties are announced in an IRD entry capability
- Selected properties are submitted in a query
- It mimics the negotiation process of cost types

Property Negotiation Example

Left: IRD entry, Right: PV request

```
    "uri": "http://alto.example.com/proxy-props",
    "media-type": "application/alto-propmap+json",
    "accpets": "application/alto-propmapparams+json",
    "capabilities": {
      "mappings": {
        "http-proxy": [ "price" ]
      }
    }
  },
  "endpoint-cost-pv": {
    "uri": "http://alto.exmaple.com/endpointcost/pv",
    "media-type": "multipart/related;
                   type=application/alto-endpointcost+json",
    "accepts": "application/alto-endpointcostparams+json",
    "capabilities": {
      "cost-type-names": [ "path-vector" ],
      "ane-properties": [ "maxresbw", "persistent-entities" ]
    },
    "uses": [ "http-proxy-props" ]
  },
  "update-pv": {
```

```
Content-Length: [TBD]
Content-Type: application/alto-endpointcostparams+json

{
  "cost-type": {
    "cost-mode": "array",
    "cost-metric": "ane-path"
  },
  "endpoints": {
    "srcs": [ "ipv4:192.0.2.2" ],
    "dsts": [ "ipv4:192.0.2.89",
              "ipv4:203.0.113.45",
              "ipv6:2001:db8::10" ]
  },
  "ane-properties": [ "maxresbw", "persistent-entities" ]
}

HTTP/1.1 200 OK
Content-Length: [TBD]
Content-Type: multipart/related; boundary=example-2;
              type=application/alto-endpointcost+json
```

Persistent Entity Property

Introduced in -07:

- An array of entity identifiers that are persistent in the scope of an ALTO server
- Not present is same as an empty list
- Motivations:
  - Allow clients to query further information related to entities discovered by PV
  - Enable applications such as in-network cache planning, etc.
  - In contrast to ANE, which is dynamically generated and specific to the query, persistent entities are persistent and can be used to query related properties in another unified property map.

Persistent Entity Property Example

Domain used in the example: http-proxy (not formally registered)

- Exported by another unified property map:

```
"http-proxy-props": {
  ...
  "capabilities": {
    "mappings": { "http-proxy": [ "price" ] }
  }
},
```

- Used by a PV resource:

```
"endpoint-cost-pv": {
  ...
  "capabilities": {
    "cost-type-names": [ "path-vector" ],
    "ane-properties": [ "maxresbw", "persistent-entities" ]
  },
  "uses": [ "http-proxy-props" ]
},
```

Persistent Entity Property Example (Cont.)

- Returned in the PV response

```
...
{
  "meta": {
    "dependent-vtags": [
      { "resource-id": "endpoint-cost-pv.ecs", "tag": ... },
      { "resource-id": "http-proxy-props", "tag": ... }
    ]
  },
  "property-map": {
    "ane:NET001": {
      "persistent-entities": [ "http-proxy:192.0.2.1" ]
    },
    "ane:L002": { "maxresbw": 48000000 },
    "ane:L003": { "maxresbw": 35000000 }
  }
}
```

Restructured Sections

In -07:

- Introduction: requirements on the path vector design
- Overview: how the extension addresses the requirements
- Motivation: the co-flow use case

Since -08:

- Introduction: the importance of path vector extension in two aspects
  - It can be used to support new usage scenarios
  - It provides fundamentally different information: correlations of network paths (while the base protocol provides preferences of network paths)
- Overview: a top-down exploration of the design space and design decision justifications
  - Why encode the information in a single message
  - Why introduce abstract network element
  - Why the specification extensions are essential
- Motivation (renamed to use cases): 3 use cases covering different usage scenarios
  - only the correlations of network paths (shared risk resource group)
  - correlations of network paths and bandwidth information (co-flow scheduling)
  - correlations of network paths and in-network resources (in-network cache planning)

Synchronized with the SSE draft -16

In -07:

- Still assume SSE uses resource-id to demultiplex updates

Since -08:

- Synchronized with SSE draft -16, which uses client-id to demultiplex updates
- Define Part Resource ID to demultiplex update streams of the (endpoint) cost map part and the property map part

Part Resource ID

- Two usages:
  - In the PV response, specify the resource dependency
  - In the SSE update stream, demultiplex updates for the two resources returned by PV
- Add a WARNING that the resource-id and client-id for each part MAY violate the length constraint of ResourceId and ClientId. Recommend clients and servers to use identifiers of less than 31 characters when using PV

Part Resource ID Example

- Specified in the part header

```
--example-1
Resource-Id: ecsmap
...
--example-1
Resource-Id: propmap
```

- Used in vtag and dependent-vtags to specify dependency

```
Resource-Id: ecsmap
...
    "vtag": { "resource-id": "endpoint-cost-pv.ecsmap", "tag": ... },
...
Resource-Id: propmap
...
    "dependent-vtags": [ { "resource-id": "endpoint-cost-pv..ecsmap", "tag": ...
```

Part Resource ID Example (Cont.)

- Used in SSE update streams to demultiplex updates

```
event: application/merge-patch+json, ecspvsub1.ecsmap
data: <Merge patch for endpoint-cost-map-update>

event: application/merge-patch+json, ecspvsub1.propmap
data: <Merge patch for property-map-update>
```

Cost Calendar Compatibility

In -07:

- The integration with Cost Calendar is left as a future requirement

Since -08:

- The Cost Calendar extension can be used directly with the path vector extension
- One requirement: the same ANE in different time intervals with different properties MUST be treated as different ANEs
  - It can cover time-varying routing and time-varying properties simultaneously

Cost Calendar Example

- Request:

```
{
  "cost-type": { "cost-mode": "array", "cost-metric": "ane-path" },
  "endpoints": {
    "srcs": [ "ipv4:192.0.2.2" ],
    "dsts": [ "ipv4:192.0.2.89", "ipv4:203.0.113.45" ]
  },
  "ane-properties": [ "maxresbw" ],
  "time-interval-size": 3600,
  "number-of-intervals": 2
}
```

- Response (PV part)

```
{ ...
  "ipv4:192.0.2.2": {
    "ipv4:192.0.2.89":   [ ["ane:L001", "ane:L003"], ["ane:L004", "ane:L003"] ],
    "ipv4:203.0.113.45": [ ["ane:L002", "ane:L003"], ["ane:L005", "ane:L003"] ]
  }
}
```

Cost Calendar Example (Cont.)

- Response (property map part)

```
{
  ...
    "ane:L001": { "maxresbw": 1000000000 },
    "ane:L002": { "maxresbw": 1000000000 },
    "ane:L003": { "maxresbw": 1000000000 },
    "ane:L004": { "maxresbw":  500000000 },
    "ane:L005": { "maxresbw": 1500000000 }
  ...
}
```

L004 and L005 can either be different ANEs or the same ANEs as L001 and L002 but with different property values.

Conclusion

- Current status
  - The main specifications are stable
  - The design decisions are better clarified and justified
- Great thanks to the coauthors and the reviewers for the feedback and guidance
- Next steps:
  - WGLC? (Agreed in IETF 104 to issue WGLC after feedback is collected)

Q & A

Join the Discussion at `alto@ietf.org`!

Questions and Comments are Welcome!