Dyncast in CFN
(dynamic anycast in compute first networking)

draft-geng-rtgw-cfn-dyncaast-ps-usecase
draft-li-rtgw-cfn-dyncaast-architecture

liyizhou@huawei.com
Typical Multi-edge Computing Usage Scenario

- Service providers are exploring the edge computing (AR, VR, connected Car)
  - Shorter, faster - compare with cloud
  - Energy (battery) saving, dataset size & governance consideration – compare with host side
- large number of edge sites in a city
- limited and varying computing resource for each site
Problem: How to optimally route service demands based on computing and network metrics to the best edge?

<table>
<thead>
<tr>
<th>Service</th>
<th>Computational complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object detection, feature extraction, template matching</td>
<td>Media</td>
</tr>
<tr>
<td>Object recognition, model training</td>
<td>High</td>
</tr>
</tbody>
</table>
Current Practices, considerations and gaps – efficiency and latency

- Use geographical location, pick closest
  - Edges are not so far apart. Locations do not matter most.
- Health check in an infrequent base (>1s), switch when fail-over
  - Limited computing resources on edge, change rapidly (<1s)
- Random or round robin pick, network cost is not a concern or updated infrequently just to keepalive
  - Edges are not deployed in equal cost way, network status is considered at a later stage not at the same time
- Centralized determination, good for content retrieval.
  - Not as good as for computation which has more dynamic nature and larger number
- Early binding: clients query first and then steer traffic.
  - Edge computing flow can be short. Early binding has high overhead.
- Caching at the client.
  - Stale info could be used, hard to guarantee to route optimally for each service demand
- Others:
  - Network based solution uses least network cost, computing load is hardly considered
  - Traditional anycast bases on single request/reply packet, no flow affinity
Routing the packets to the best edge with joint consideration of computing and network load. Transparent to the clients.

Client uses anycast address to access green service.

Data plane:
Data flow to the selected MEC Site 2

Control plane:
Service info notification and update to CFN node
Info distribution among CFN nodes

CFN: Compute first networking. A distributed computing environment across a network to perform flexible load management and performance optimizations.
Service and Service Instance in CFN-Dyncast

- **Services:**
  - Blue
  - Red
  - Green

- A service offers one specific function no matter where it is deployed.
- One Service can have several instances running on different edges.
- Service instance is a running environment (e.g., a node) that makes the functionality of a service available.
- All service instances running the same service are identified by the same **Service Identifier (SID)**.
Features to be supported

• Anycast based service addressing methodology
• Flow affinity
• Computing Aware Routing
Anycast based service addressing methodology

- SID: Service ID - an anycast address - unique ID to identify a service
- BID: Binding ID - an unicast address - accessible to a particular service instance

• Same service, i.e. same service ID (SID)
• Two service instances at different places with binding ID BID22 & BID 32
Flow affinity – select the best edge and stick to it

A new flow: selects the most appropriate CFN node and service instance.
Flow affinity: the subsequent packets of an existing flow are always delivered to the same service instance

Example of flow binding table

<table>
<thead>
<tr>
<th>Flow Identifier</th>
<th>CFN egress</th>
<th>timeout</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>src_IP</td>
<td>dst_IP</td>
</tr>
<tr>
<td>X</td>
<td>SID2</td>
<td>-</td>
</tr>
<tr>
<td>Y</td>
<td>SID2</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Computing Aware Routing

Service info notification and update to CFN node

Info distribution
Summary

• Three features to be supported
  – Anycast based service addressing methodology
  – Flow affinity
  – Computing Aware Routing

• Potential work required
  – Represent computing metrics in defined service/service instance context
  – Distribute the metrics, format and how dynamic/frequent the updates should be
  – Use the metrics in route determination
  – Definition of requirements for any new data plane extensions and procedures.
Your help is welcome

- We will have a virtual side meeting on cfn-dyncaast
  - Understand the problem space, gaps and challenges
  - Review the dyncaast architecture
  - Discuss the potential work and where to fit them in IETF

- Time: Wed (Nov 18), 75min, 5 min after plenary ends
  - UTC 10:45 - 12:00
  - CET (UTC+1) 11:45 - 13:00
  - Bangkok Time (UTC+7) 17:45 - 19:00
  - CST (UTC+8) 18:45 - 20:00
  - PST (UTC-8) 02:45 - 04:00

- Webex:
  - Webex Meeting number (access code): 175 335 6387
  - Password:7wrDVwRt7B4
  - Password if joining from a phone: 79738978
  - Webex: https://fipe-meeting.my.webex.com/fipe-meeting.my/j.php?MTID=m2e7d90ec32145ba12f4a6b7e8baf3bcd

- Information also available on side meeting wiki and github:
  https://trac.ietf.org/trac/ietf/meeting/wiki/109sidemeetings
  https://github.com/cfn-dyncaast/ietf109