Computing-Aware Networking(CAN)

draft-liu-dyncast-ps-usecases-04
draft-liu-dyncast-gap-reqs-00
draft-li-dyncast-architecture-04
draft-liu-can-computing-resource-modeling-00

P. Liu, China Mobile

Motivations – Why CAN?

Computing-Aware Networking (CAN) aims at computing and network resource optimization by steering traffic to appropriate computing resources, considering not only networking metric but also computing resource metric and service affiliation.

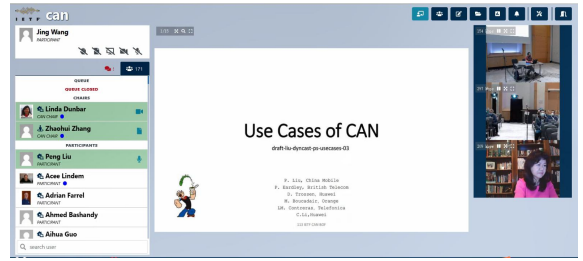
CAN is to solve the problem of " the 'closest' is not the 'best'", providing the best user experience of low latency, high reliability and stable services experience when moving across different areas, based on the Increasing development of integrated computing and networking infrastructure, including CDN and edge computing.

CAN BoF at IETF 113

A non-wg forming BoF with178 people participated in the discussion

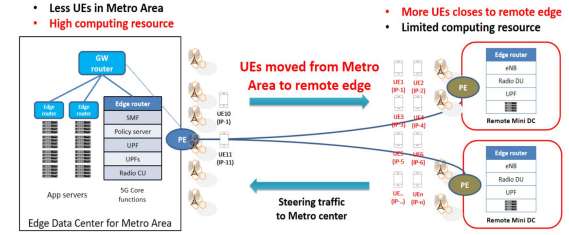
Use cases:

- The deployment of edge computing and user mobility show the necessity of “steering traffic among different edge sites”.
- Majority of modern applications have multiple instances instantiated at many locations and have short flows
- Many apps, such as AR/VR typically requiring 20ms E2E delay, need "considering more factors when steering traffic ”



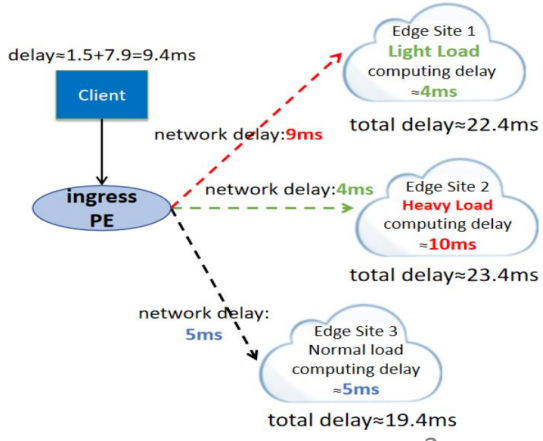
Gap analysis:

- DNS was not designed for dynamic scenarios with fast changes of serving service instance
  - •DNS caching and learning computing status are additional problems
- Load balancer should learn about the network status, while as an off-path approach, it also adds latency and bottleneck



Potential Solutions:

- **Dynamic anycast:** edge routers get the computing metrics and select the service instance
- **On-path load balancer:** centralized/decentralized LBs coordinates, also learns the network status



Conclusion:

- The BoF was successful in agreeing that the **use cases are important**.
- **Study existing solutions**, e.g. ALTO. Please continue pay attention to reusing the existing work.
- Load Balancer/D-router or Dyncast have own characteristics and could **be further investigated**.
- The next hop of this work (WG to land in) is **still not clear**, please continue working.

Issues Related to Upper Layer Protocols

36 issues raised from the BoF discussion; Lots of discussions on dyncast@ietf.org and rtgwg@ietf.org

Summary of key issues

- **What is the relation between CAN traffic steering and service deployment, discovery, and the upper layer protocol? #19 #21 #23**

  The whole process includes service deployment, service discovery and service traffic steering. CAN assumes the services have been deployed and discovered in multi-edge sites, aiming at the traffic steering.

  One the one hand, CAN should coordinate with L4 and L7 for service deployment and discovery. On the other hand, the computing metrics collection in CAN will also benefit service deployment and discovery.

- **What is the relation between CAN and ALTO? #5**

  ALTO solves the problem of service instance selection as an off-path solution, which can be seen as an alternative way of addressing the problem space of CAN at the Application Layer. So in that respect, even targeting a common problem, both provide different approaches, then imposing different needs but also taking different assumptions on how applications and networks interact.

  Off-path systems, e.g., ALTO, replies for applications/services before traffic delivery might not be optimal or valid after the handover. So, more details are needed of ALTO including some extension to support multi-deployment, quick interaction, integrate more performance metric information.

More issues could be found at https://github.com/CAN-IETF/CAN-BoF-ietf113/issues

Key Drafts

New versions of *problem statement*, *use cases*, *gap analysis* and *requirements* were posted, following the BoF discussion and addressing some of the issues:

- **draft-liu-dyncast-ps-usecases-04**
  - Updated the problem statement and use cases (https://datatracker.ietf.org/doc/draft-liu-dyncast-ps-usecases/)
- **draft-liu-dyncast-gap-reqs-00**
  - Replacing previous version by adding the gap analysis and updating requirements (https://datatracker.ietf.org/doc/draft-liu-dyncast-gap-reqs/)
- **draft-li-dyncast-architecture-04**
  - Addressed some issues in the BoF (https://datatracker.ietf.org/doc/draft-li-dyncast-architecture/)

New draft of computing resource modeling,

- **draft-liu-can-computing-resource-modeling-00**
  - describes possible modeling of compute resource and usage in CAN (https://datatracker.ietf.org/doc/draft-liu-can-computing-resource-modeling/)

Next Steps

- **Refine the drafts.**
- **Welcome more comments and suggestions from tsvwg.**

Thank you!