CAN Gap Analysis and Requirements

113 IETF CAN BOF

Gap Analysis

draft-liu-dyncast-ps-usecases-03

P. Liu, China Mobile
P. Eardley, British Telecom
D. Trossen, Huawei
M. Boucadair, Orange
LM. Contreras, Telefonica
C.Li,Huawei

Existing solutions

Here we list some ‘existing solutions’ based on the assumption of supporting the network and computing joint optimization (futher assumption of appropriate extension if needed).

DNS：

- 'early binding' to explicitly bind from the service identification to the network address
- ‘geographical location’ to pick the closest computing resource
- ‘health check’ to realize load balance

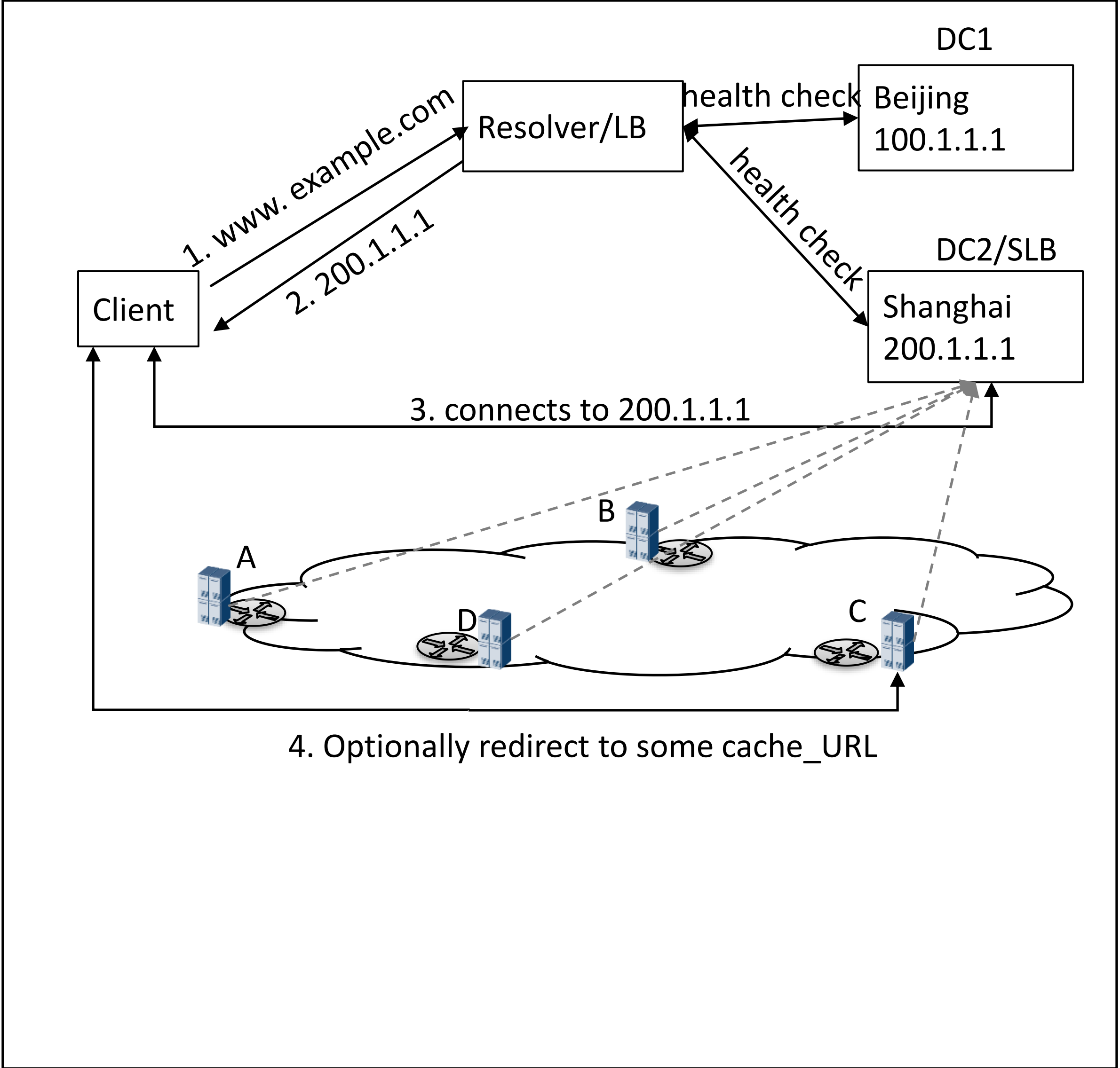
Load balancer or application based ways：

- external components designed for computing domain to discovery the instance and use some load balancing algorithms
- may also be based on DNS system and require app level query

Message broker:

- collect the computing resource status by an agent

Deficiency in existing solutions-DNS/DNS+LB



- Early binding: clients resolve IP address first and then steer traffic.
  - Early binding has high overhead since resolution and data plane traffic, i.e. at least 4 messages, will all go via the client access link
  - Usually DNS responses cached at client, i.e., stale info could be used.
- Often, resolver and LB are separate entities
  - Incurs even more signaling overhead by needing to first resolve (to CDN LB) and then redirect to LB for final decision
- Resolution is L7 or app-level decision making, i.e. DB lookup
  - Originally intended for control, NOT data plane speed!
- Centralized determination good for long affinities
  - Not as good for computation which has more dynamic nature and smaller *affinities*, i.e. service transaction lengths
- Traditional anycast based on single request/reply, no flow affinity
- Health check on an infrequent base (>1s), switch when fail-over
  - Limited computing resources on edge, change rapidly (<1s)
  - Any more frequent health check is prohibitive in cost
- LB usually focused on edge server load with network status not considered at same time but at a later stage
  - Usually captured in the routing policy to specific edge server

Deficiency in existing solutions- Load balancer at egress router in network

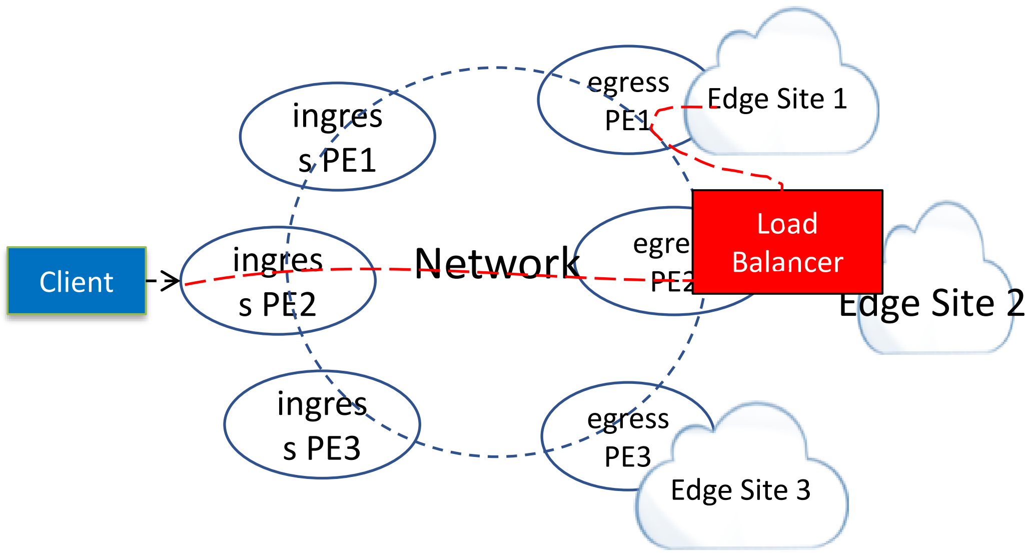
Single LB at the egress for all server instances:

Pros:

- easy deployment

Cons:

- Single point of failure at the LB
- The network path from the LB to server instances at other sites might not be optimal, e.g., the red dotted path



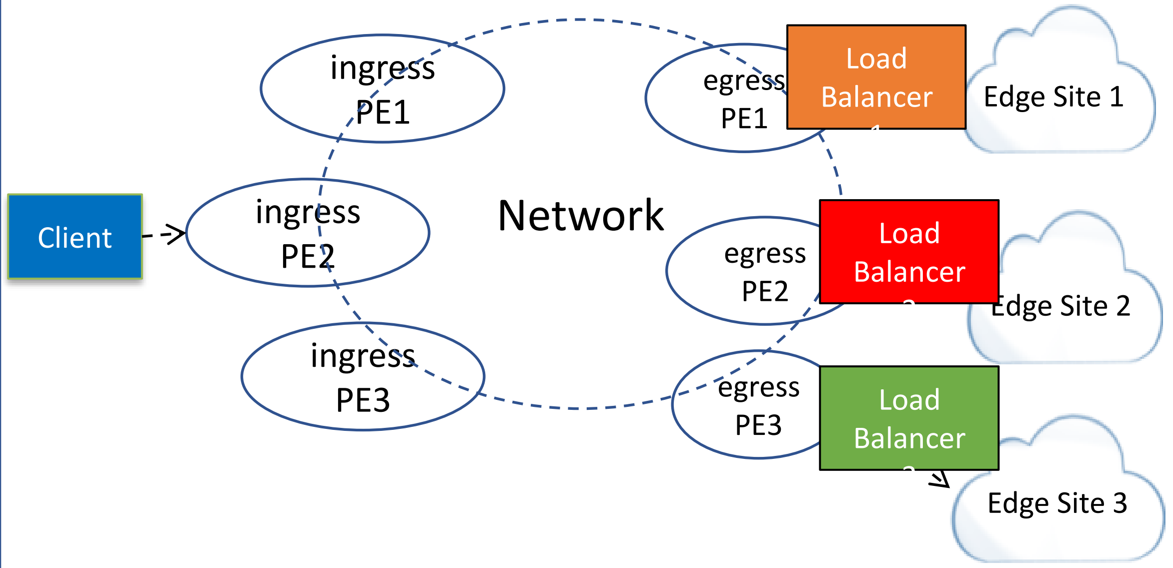
Multiple LBs at different sites:

Pros:

- Avoid suboptimal path from one LB to server instances at different sites.

Cons:

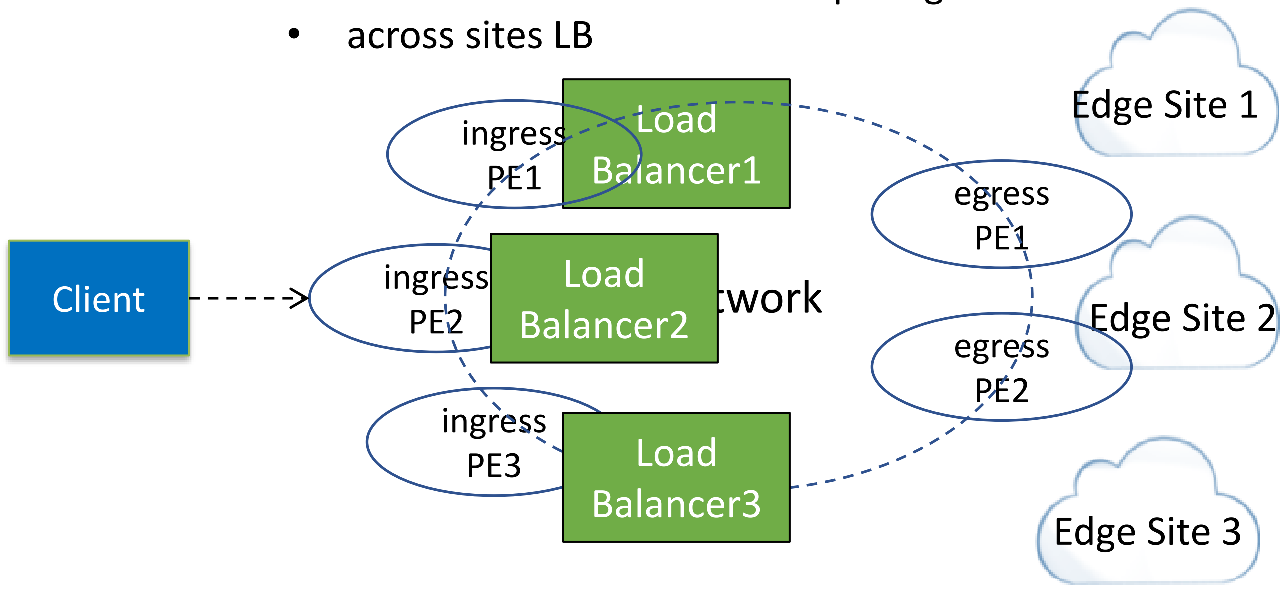
- The LB that has the shortest path to the ingress might not have sufficient compute resources. Whereas the server instances behind other LBs are underutilized.



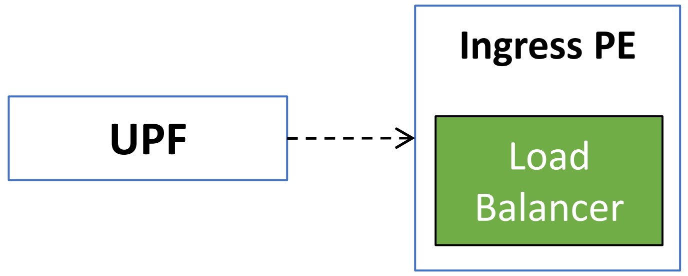
Deficiency in existing solutions- Load balancer at ingress router in network

- Need LBs at the ingress to be informed of the network conditions constantly,
  - either by the existing routing protocols (IGP/BGP) or inventing a new one which is costly. So that path conditions can be incorporated into selecting the optimal site.
  - potential choice for CAN, according to dyncast arch draft
- NOTE: In 3GPP architecture, LB may be located at each UPF

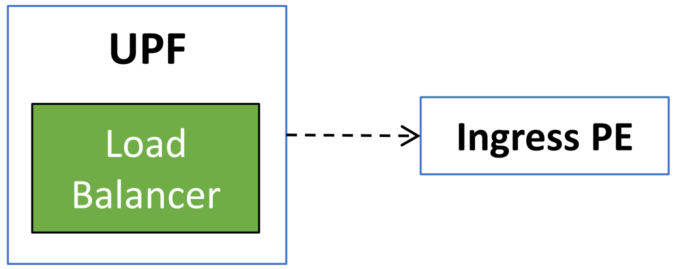
- have both network and computing statues
- across sites LB



Potential way for IETF



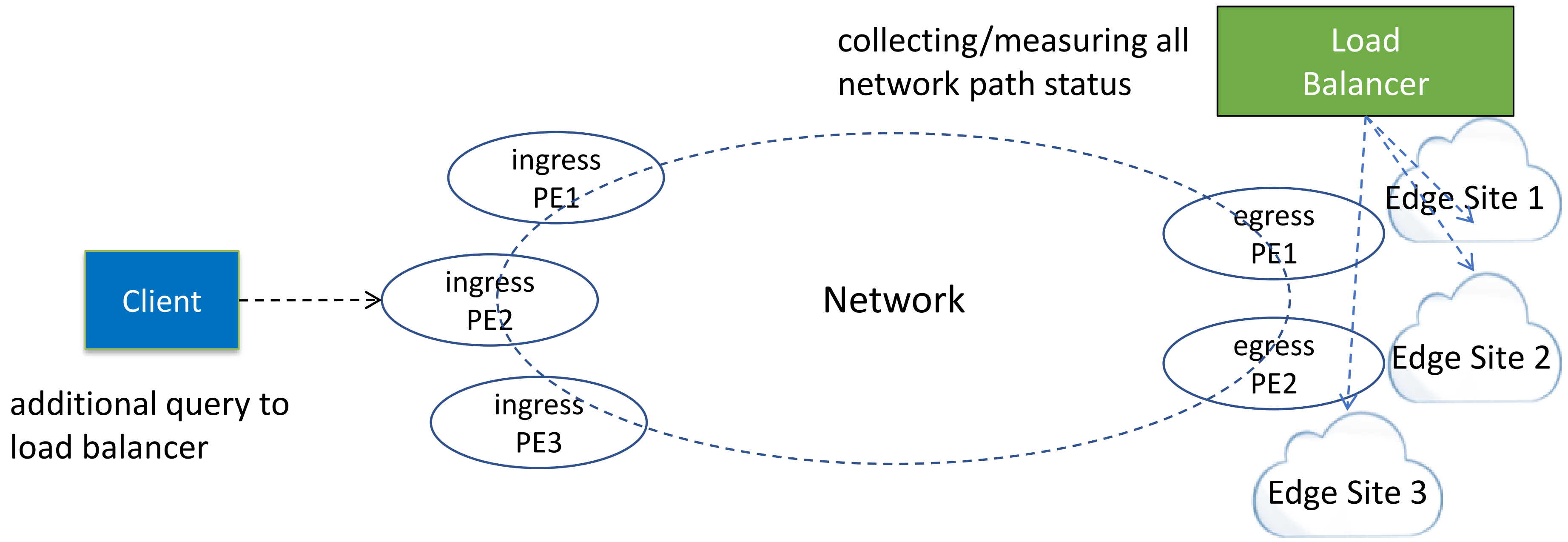
Potential way for 3GPP architecture



Deficiency in existing solutions- Load balancer at external locations

- Similar messaging overhead as DNS+LB, i.e. indirection with additional signaling to query the appropriate service instance, causing additional latencies, e.g., at least 4 messages go via client access link
- inefficiencies of health check since collecting/measuring all network path status info may lead to out of sync problems

-> this does not seem to gain anything over existing DNS+LB solutions?!



Deficiency in existing solutions- Architectural questions for load balancer

- Is ’LB’ positions as an app-level element but for in-path address re-writing, which sounds architecturally wrong?
- If it does service proxying (not address re-writing), is that what we would want?
- Or is LB an L3-level realization of d-router (see dyncast arch draft) with data plane table look up but ’wrapped’ into some deployment language, such as VNF deployment in 3GPP as colocation to UPF?

Overall deficiency in existing solutions

- **Dynamicity of Relations:** Existing solutions exhibit limitations in providing dynamic 'instance affinity'
  - E.g., DNS is not designed for this level of dynamicity (i.e., minute level originally, client needs to flushing the local DNS cache, frequent resolving may lead to overload of DNS)
- **Efficiency:** Existing solutions may introduce additional latencies and inefficiencies (e.g., additional path stretch & more messages) in packet transmission
- **Complexity and Accuracy:** Existing solutions require careful planning for the placement of necessary control plane functions in relation to the resulting data plane traffic, which is difficult and may lead to the inaccuracy of the scheduling.
- **Metric exposure and use:** Existing solutions lack the necessary information to make the right decision on the selection of the suitable service instance due to the limited semantic or due to information not being exposed
- **Security**: Existing solutions may expose control as well as data plane to the possibility of a distributed Denial-of-Service attack on the resolution system as well as service instance.
- **Infra changes**: Existing solutions require changes to service and/or network infrastructure, with no solution limiting the necessary changes to the very ingress point of the network

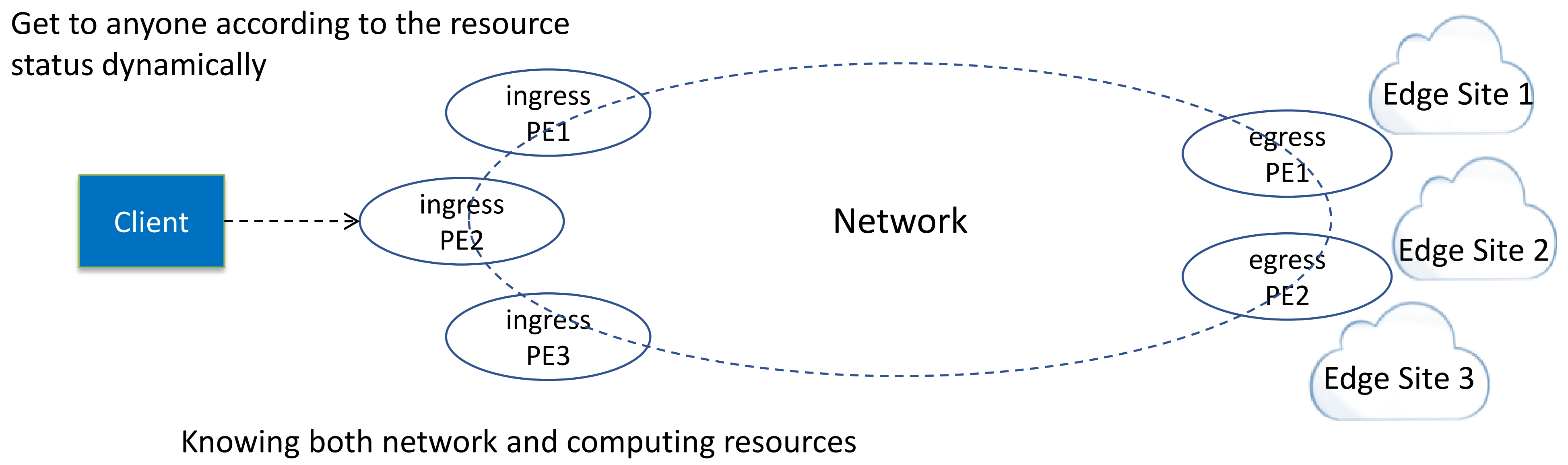
Requirements

draft-liu-dyncast-reqs-02

P. Liu, China Mobile
T. Jiang, China Mobile
P. Eardley, British Telecom
D. Trossen, Huawei
C.Li,Huawei

Main goals

- **Anycast: considering to access the location of multi-computing resource**
- **Dynamiclly: considering to select the appropriate computing resource dynamiclly**
- **Multi-metric: considering both the network and computing resource statues**


Get to anyone according to the resource status dynamically
Client
ingress PE1
ingress PE2
ingress PE3
Network
egress PE1
egress PE2
Edge Site 1
Edge Site 2
Edge Site 3
Knowing both network and computing resources

Potential requirements

- **Support joint scheduling and optimization of network and computing**
- **Support considering and using both network and computing metrics**
- **Support the session continuity and service continuity**
- **Support management of computing and network**
- **Support the interface between network and computing components**

Thank you!