RTGエリアホットトピック CAN bof

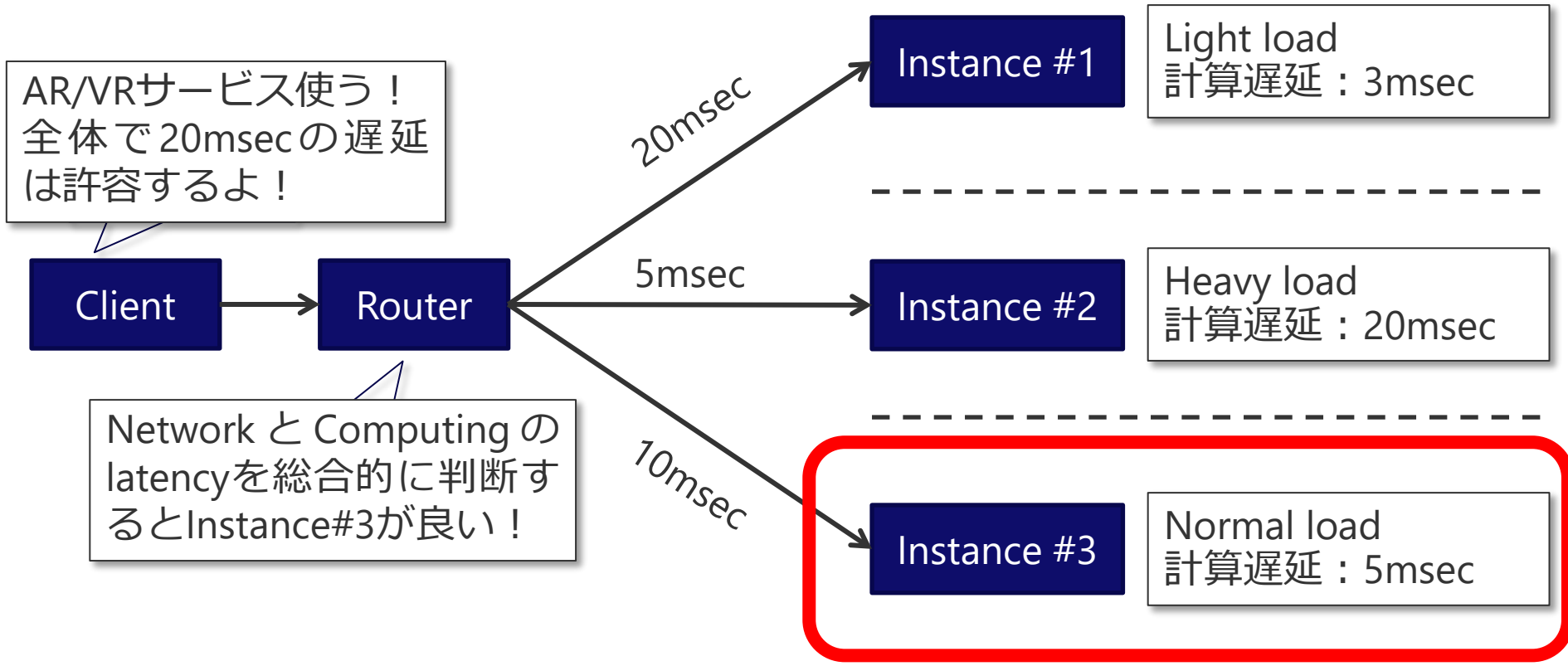
KDDI総合研究所
宮坂 拓也

はじめに：発表概要

本発表は・・・

- IETF113において開催されたComputing-Aware Networking (CAN) BoFについて紹介します
  - その名の通り、サービスインスタンスの計算資源の状況も加味したRoutingを実現することを目指すもの

１スライドで理解するCANが目指す世界観



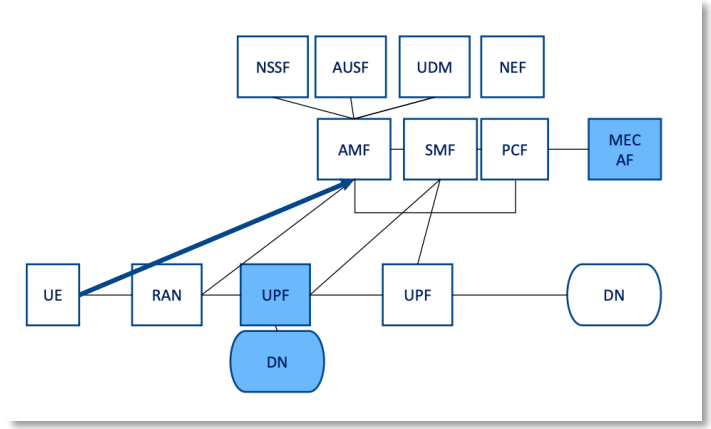
他の標準化団体の動向：ETSI MEC

ETSI MEC

- MECアーキテクチャ、APIの確立
- 5G MECアーキテクチャの検討
  - 5Gとの連携：Traffic Influence

EdgeがNetworkに導入されることで
Networkに新しい課題が出てくる？

- 例：最適なEdgeの選択



Traffic Influence

https://datatracker.ietf.org/meeting/113/materials/slides-113-can-mec-cnc-01

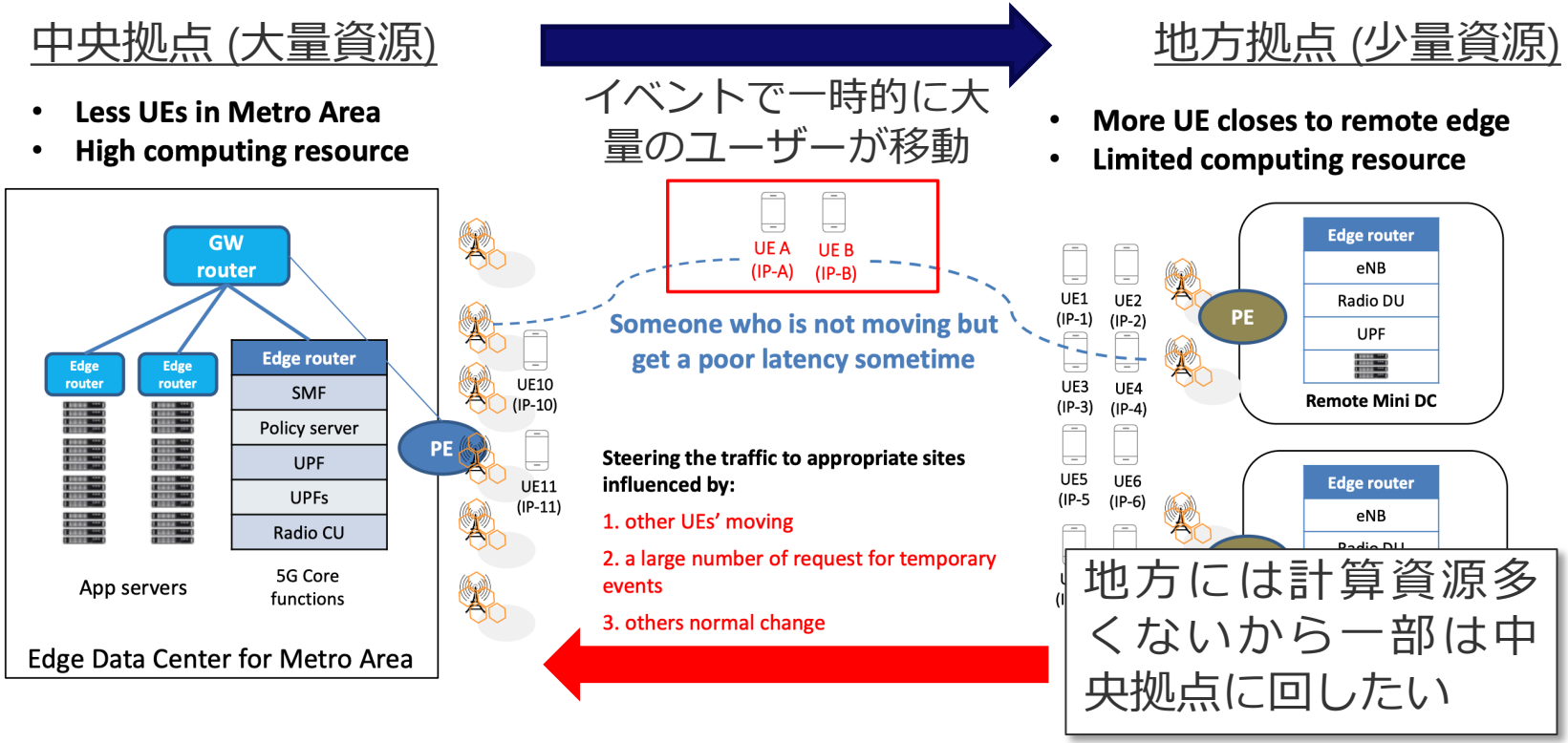
他の標準化団体の動向：ITU-T CNC

ITU-T SG13 CNC

- Computing and Network Convergence
- ITU-Tにおいても、ComputingとNetworkの融合(?)による新しいアーキテクチャに関する検討をしているとのこと
- Active item
  - Requirement of CNC
  - QoS requirement and framework of CNC
  - Management requirement and framework of CNC

https://datatracker.ietf.org/meeting/113/materials/slides-113-can-mec-cnc-01

CAN use-case



https://datatracker.ietf.org/meeting/113/materials/slides-113-can-use-cases-04.pdf

CAN use-case

Network と Computing それぞれのLatencyを複合的に判断

Upper bound latency for motion-to-photon(MTP): includes frame rendering and requires less than **20 ms** to **avoid motion sickness,** consisted of:

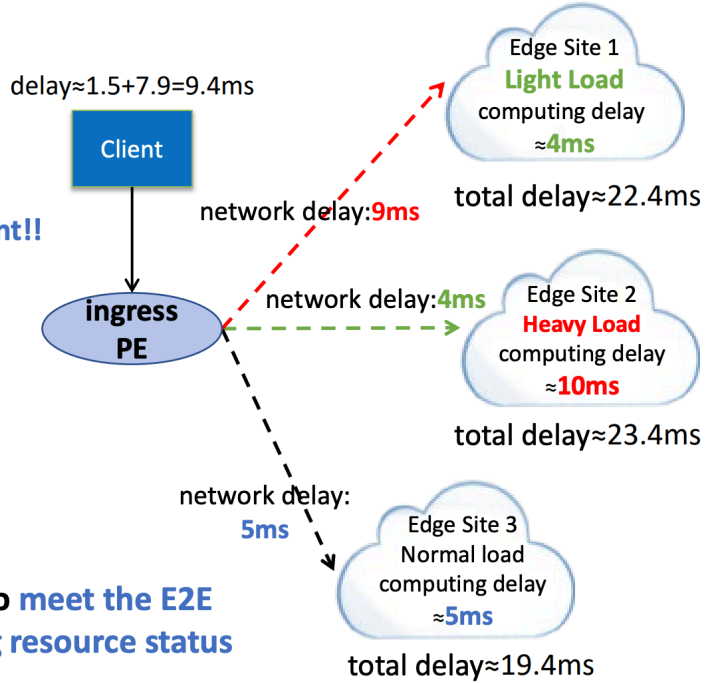
1. sensor sampling delay: <1.5ms (client)
2. display refresh delay: ≈7.9 ms(client)
3. frame rendering computing delay with **GPU≈ 5.5ms** (server)
4. network delay(budget) =20-1.5-7.9-5.5 = **5.1ms**(network)

Budgets for computing delay and network delay are almost equivalent!!

- choose edge site 1 according to load only, total delay≈22.4ms
- choose edge site 2 according to network only, total delay≈23.4ms
- choose edge site 3 according to both, **total delay≈19.4ms**

It can't meet the total delay requirements or find the best choice according to either the network or computing resource status:

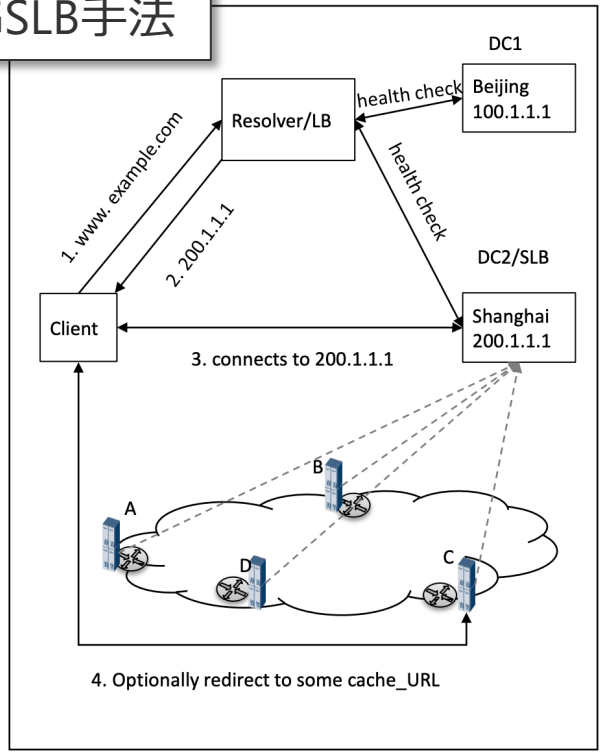
Require to dynamically steer traffic to the appropriate edge to meet the E2E delay requirements considering both network and computing resource status



https://datatracker.ietf.org/meeting/113/materials/slides-113-can-use-cases-04.pdf

よくある手法とのGap: GSLB

DNS/GSLB手法



- Early binding: clients resolve IP address first and then steer traffic.
  - Use the DNS entry cached at client, stale info may be used.
  - Often, resolver and LB are separate entities which incurs even more signaling overhead by needing to first resolve 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!

振り分けが遅い, パフォーマンス出ない

- Health check: on an infrequent base, switch when fail-over
  - Limited computing resources on edge will change rapidly, while more frequent health check is prohibitive in cost

死活監視が頻繁でない

- Load balance over DNS: usually focused on edge server load first, then utilizing lowest latency routing to the selected server's IP address
  - Lacks the combined consideration for load & latency's for a better E2E guarantee
  - Problem of how to obtain necessary metrics for decision

遅延と負荷の統合的考慮ができていない

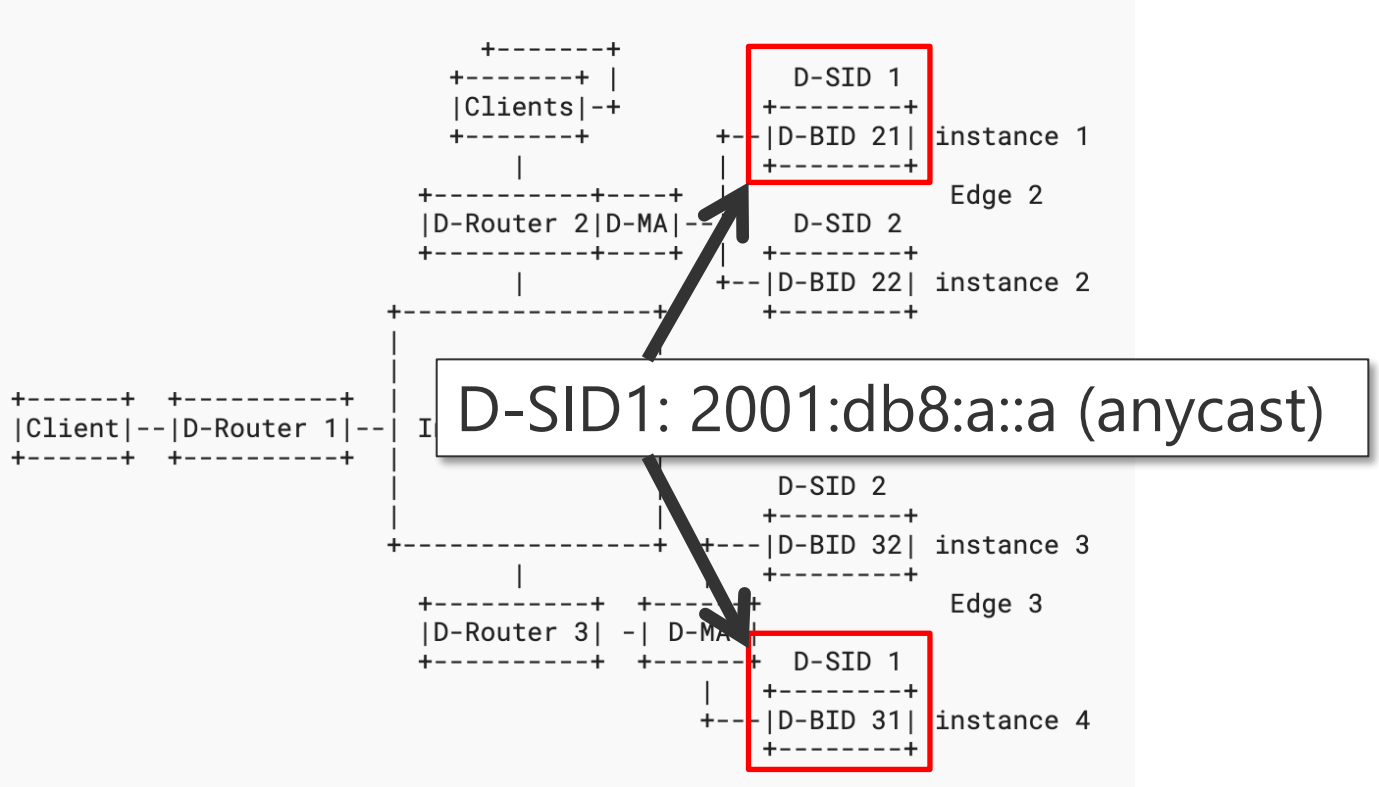
https://datatracker.ietf.org/meeting/113/materials/slides-113-can-gap-analysis-requirements-04

CAN solution: Dyncast

Dynamic anycast (Dyncast)

- Metric
  - 各Service Instanceの負荷情報をNetwork内で広報
- IP addressing
  - Dyncast Service Identifier (D-SID)
    - サービスを識別するAnycast address
  - Dyncast Binding Identifier (D-BID)
    - サービスインスタンスを識別するUnicast address
- Node
  - Dyncast Router (D-Router)
    - Dyncast機能をサポートするルーター
  - Dyncast Metric Agent (D-MA)
    - Metricを収集・分配するAgent、Routingの意思決定は行わない。

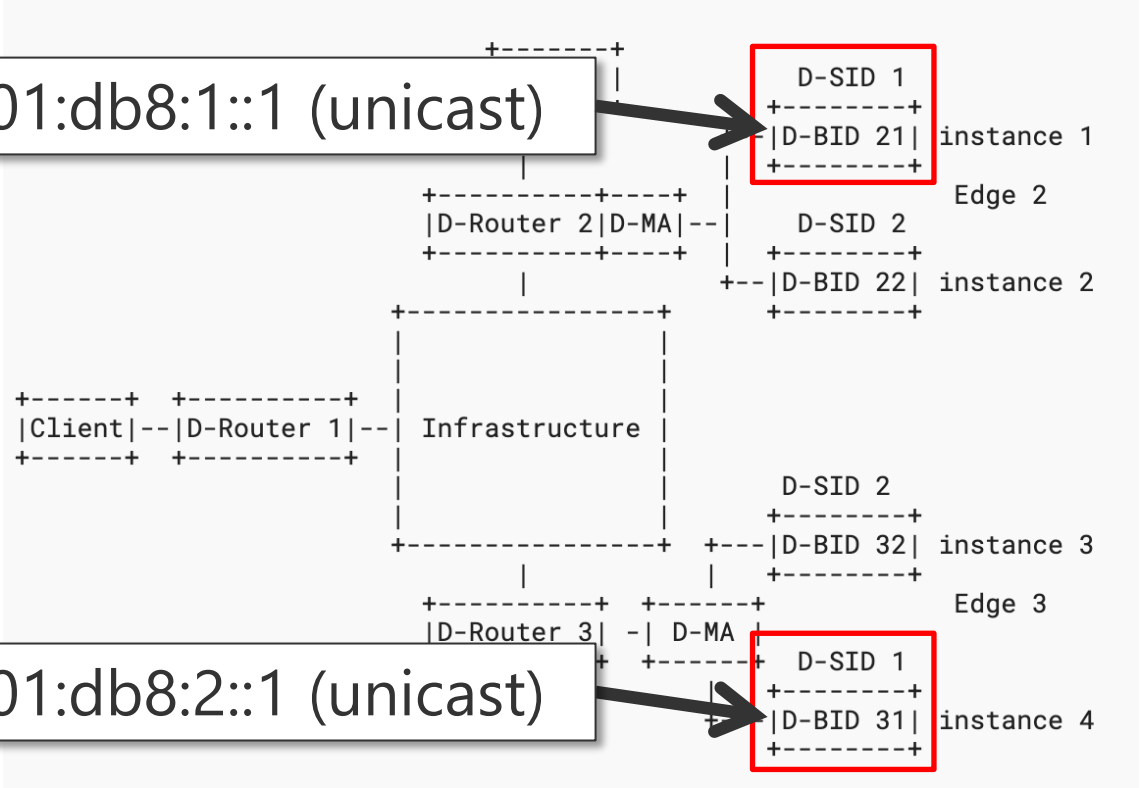
D-SID example



https://www.ietf.org/archive/id/draft-li-dyncast-architecture-03.html

D-BID example

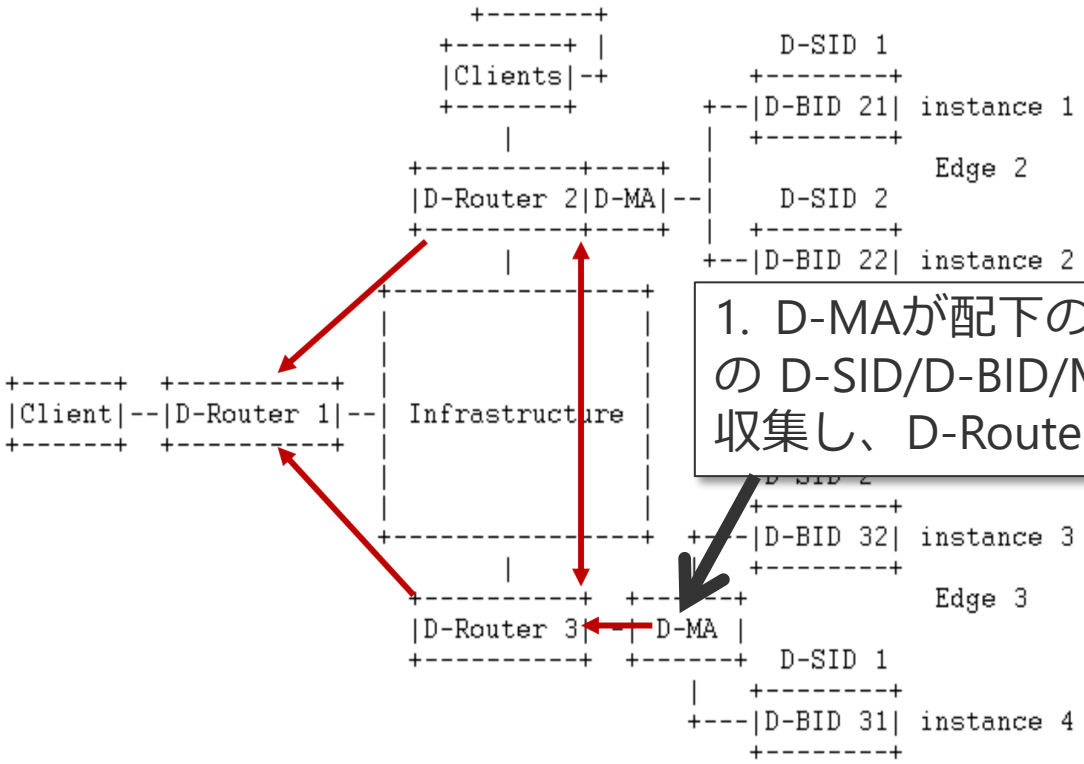
D-SID1: 2001:db8:1::1 (unicast)



D-SID1: 2001:db8:2::1 (unicast)

https://www.ietf.org/archive/id/draft-li-dyncast-architecture-03.html

Dyncast metric distribution



1. D-MAが配下のInstanceの D-SID/D-BID/Metric を収集し、D-Routerへ送信

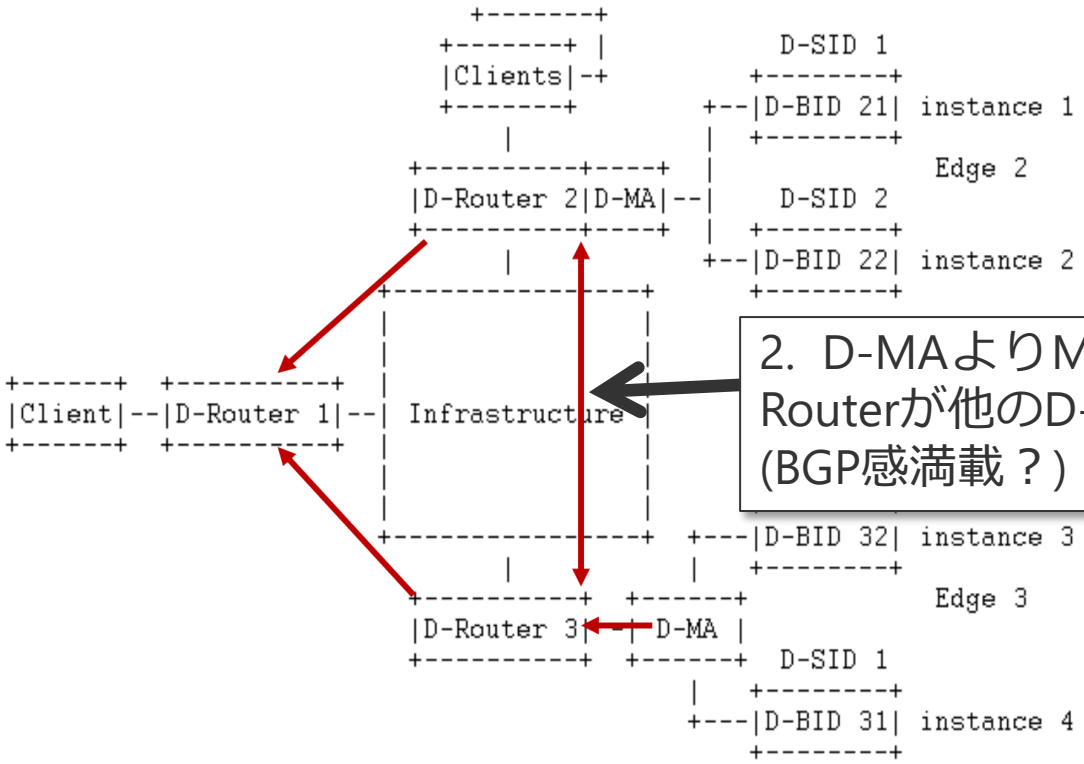
Service/Metrics Information

- (D-SID 1, D-BID 21, <metrics1>)
- (D-SID 2, D-BID 22, <metrics2>)

Service/Metrics Information

- (D-SID 1, D-BID 31, <metrics3>)
- (D-SID 2, D-BID 32, <metrics4>)

Dyncast metric distribution



2. D-MAよりMetricを受信したD-Routerが他のD-Routerへ送信する(BGP感満載？)

Service/Metrics Information

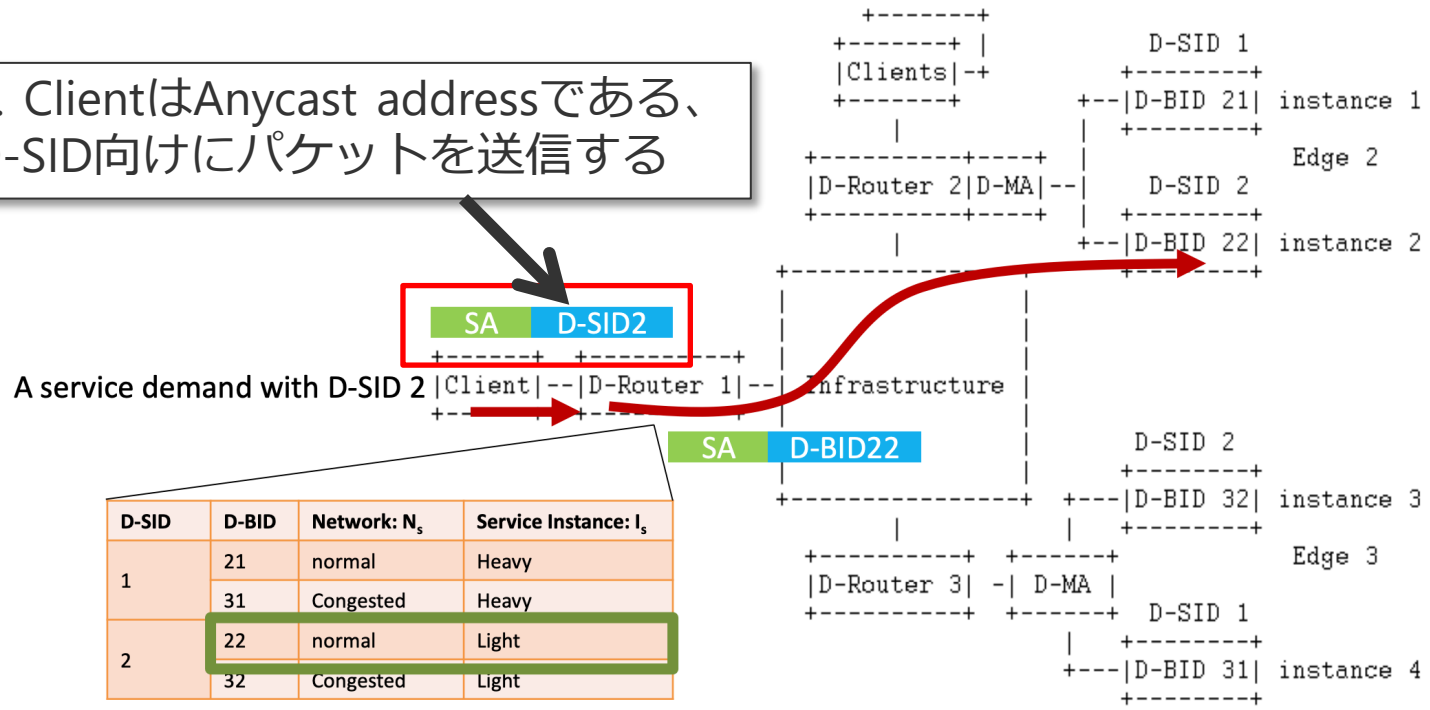
- (D-SID 1, D-BID 21, <metrics1>)
- (D-SID 2, D-BID 22, <metrics2>)

Service/Metrics Information

- (D-SID 1, D-BID 31, <metrics3>)
- (D-SID 2, D-BID 32, <metrics4>)

Dyncast forwarding

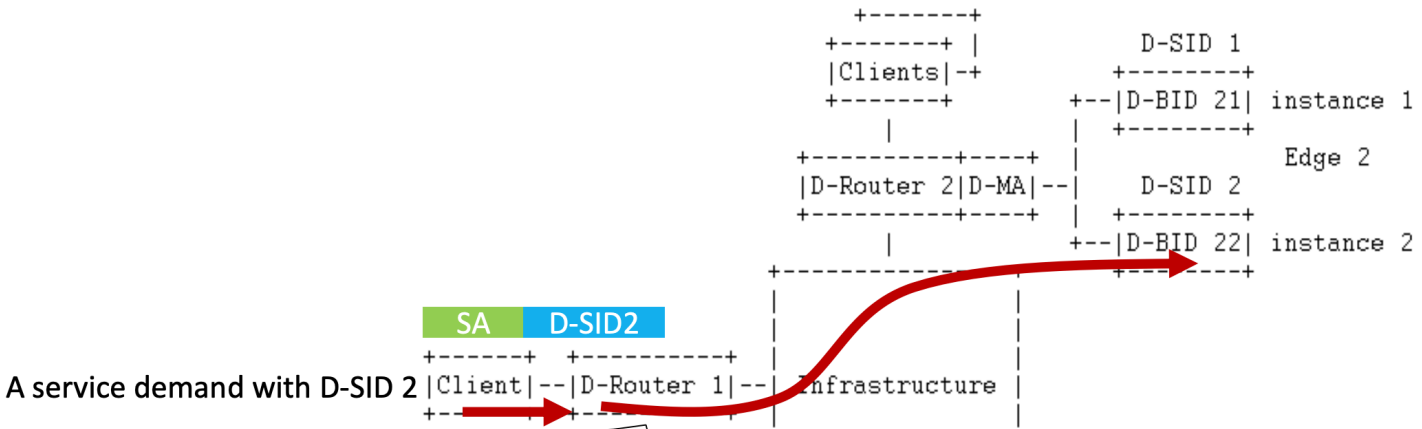
1. ClientはAnycast addressである、D-SID向けにパケットを送信する



D-SID	D-BID	Network: N_s	Service Instance: I_s
1	21	normal	Heavy
	31	Congested	Heavy
2	22	normal	Light
	32	Congested	Light

- N_s: Network Metrics (congestion, latency….)
- I_s: Instance Service Metrics (load, resources available, …)

Dyncast forwarding

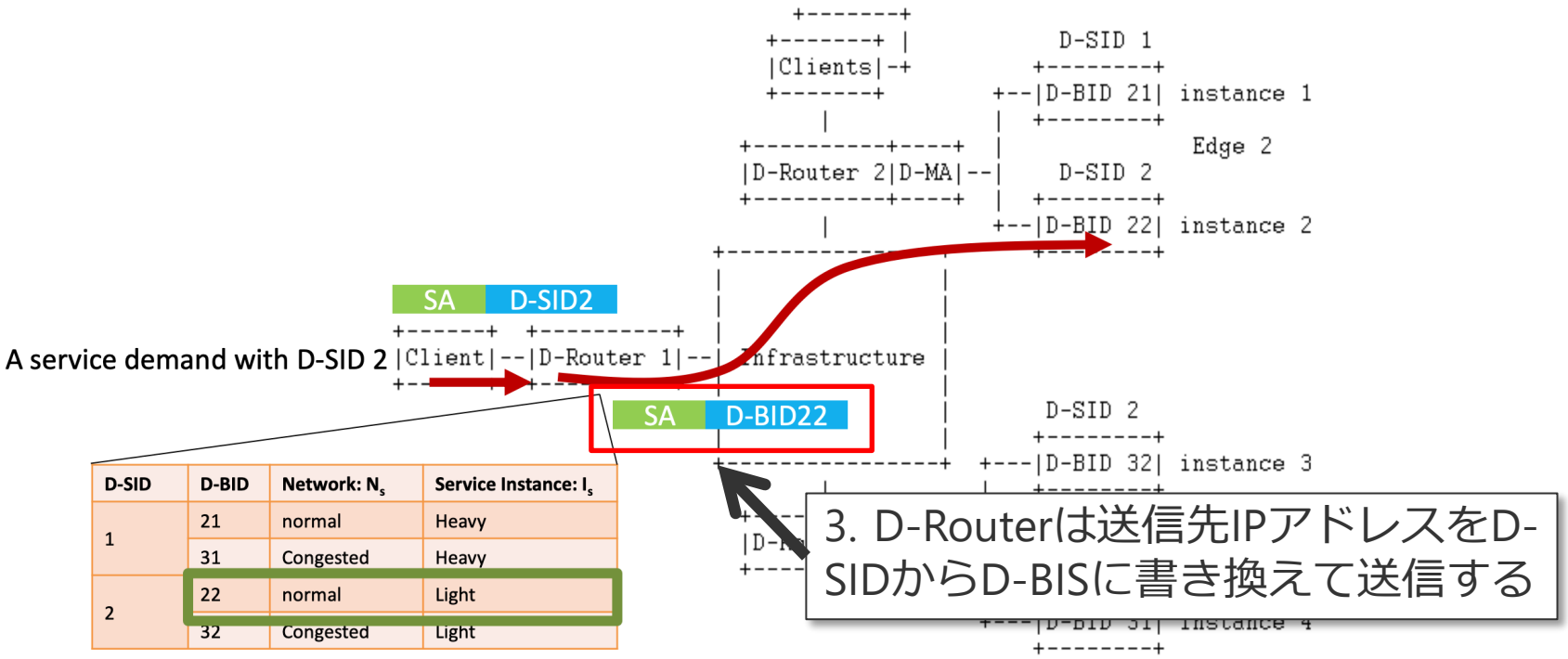


D-SID	D-BID	Network: N_s	Service Instance: I_s
1	21	normal	Heavy
	31	Congested	Heavy
2	22	normal	Light
	32	Congested	Light

2. D-Routerで受信したD-SID情報と、他のD-Router/D-MAから受信したMetric情報から最適なService Instanceを選択する
(この場合、D-BID=22が選択)

- N_s: Network Metrics (congestion, latency….)
- I_s: Instance Service Metrics (load, resources available, …)

Dyncast forwarding



D-SID	D-BID	Network: N_s	Service Instance: I_s
1	21	normal	Heavy
	31	Congested	Heavy
2	22	normal	Light
	32	Congested	Light

- N_s: Network Metrics (congestion, latency….)
- I_s: Instance Service Metrics (load, resources available, …)

Flow tableの管理

各D-Routerでは、既存のIP flow(5-tuple)が現在どのD-BIDに割り当てているか管理する必要がある

- ClientとService instance間でセッションを接続している時にD-Routerが他のService instanceのD-BIDに転送したらセッションが切れるため
- どのように管理するかなどは現状でOpen issue

Flow Identifier					BID	timeout
src_IP	dst_IP	src_port	dst_port	proto		
X	SID2	-	8888	tcp	BID22	xxx
Y	SID2	-	8888	tcp	BID32	xxx

(最終スライド) BoF結果

Area directorからのコメント

- 良い議論、エネルギーを感じる、BoFは成功した
- ユースケースは重要であるというコンセンサスは感じた
- ALTOなどの既存技術を確認してほしい
- Architectureについてunderlayで本当に実施するか更に議論すべき
- Load Balancerを標準化(LB間のMetricとMessage等)するという意見もチャット等で見れた
- **The next hop of this work (WG to land in) is still not clear, please continue working.**

https://datatracker.ietf.org/meeting/113/materials/minutes-113-can-00