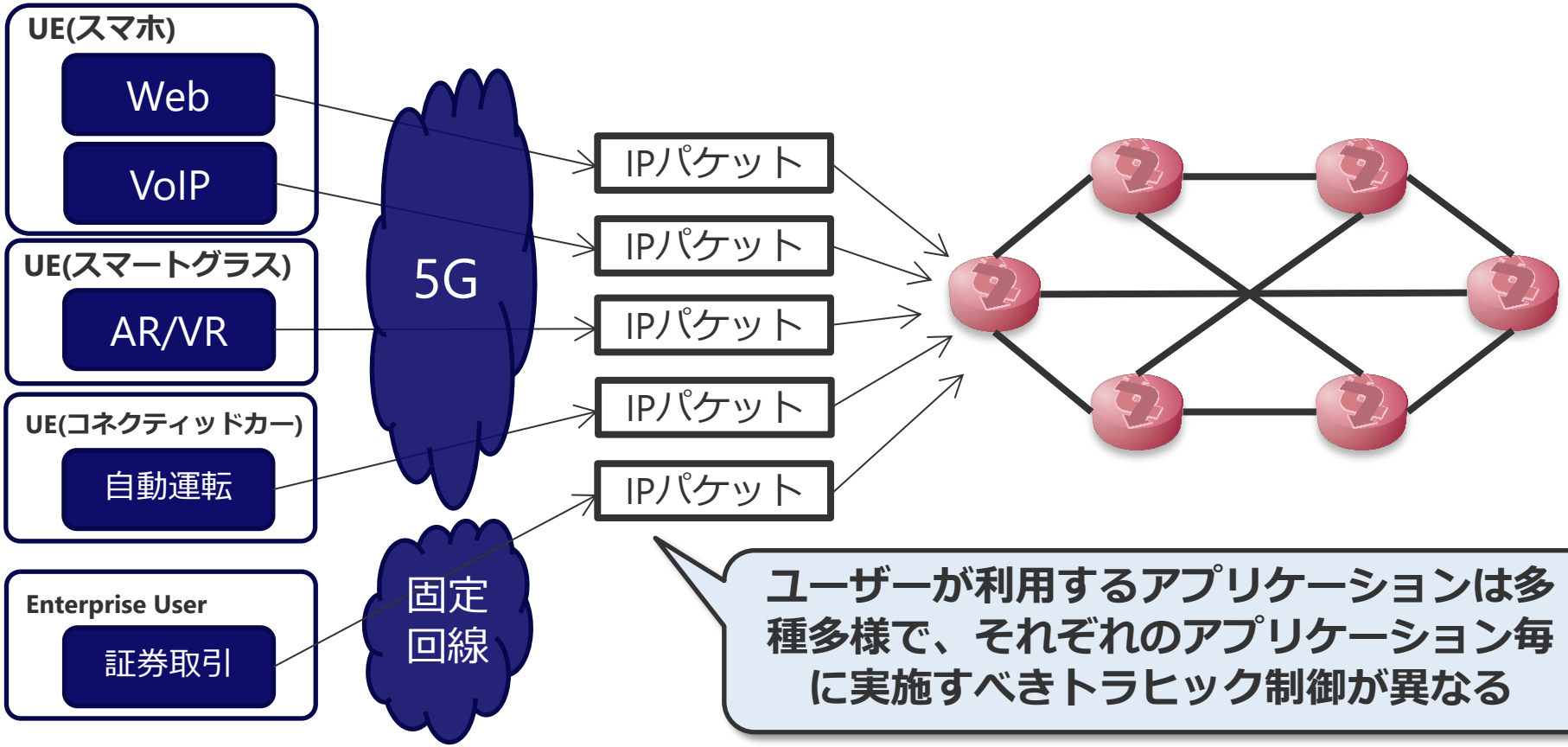
Application-aware Networking: アプリケーションの通信品質要求を満たすネットワーク実現に向けて

JANOG47 2021/01/28 15:45~16:30 (45分)

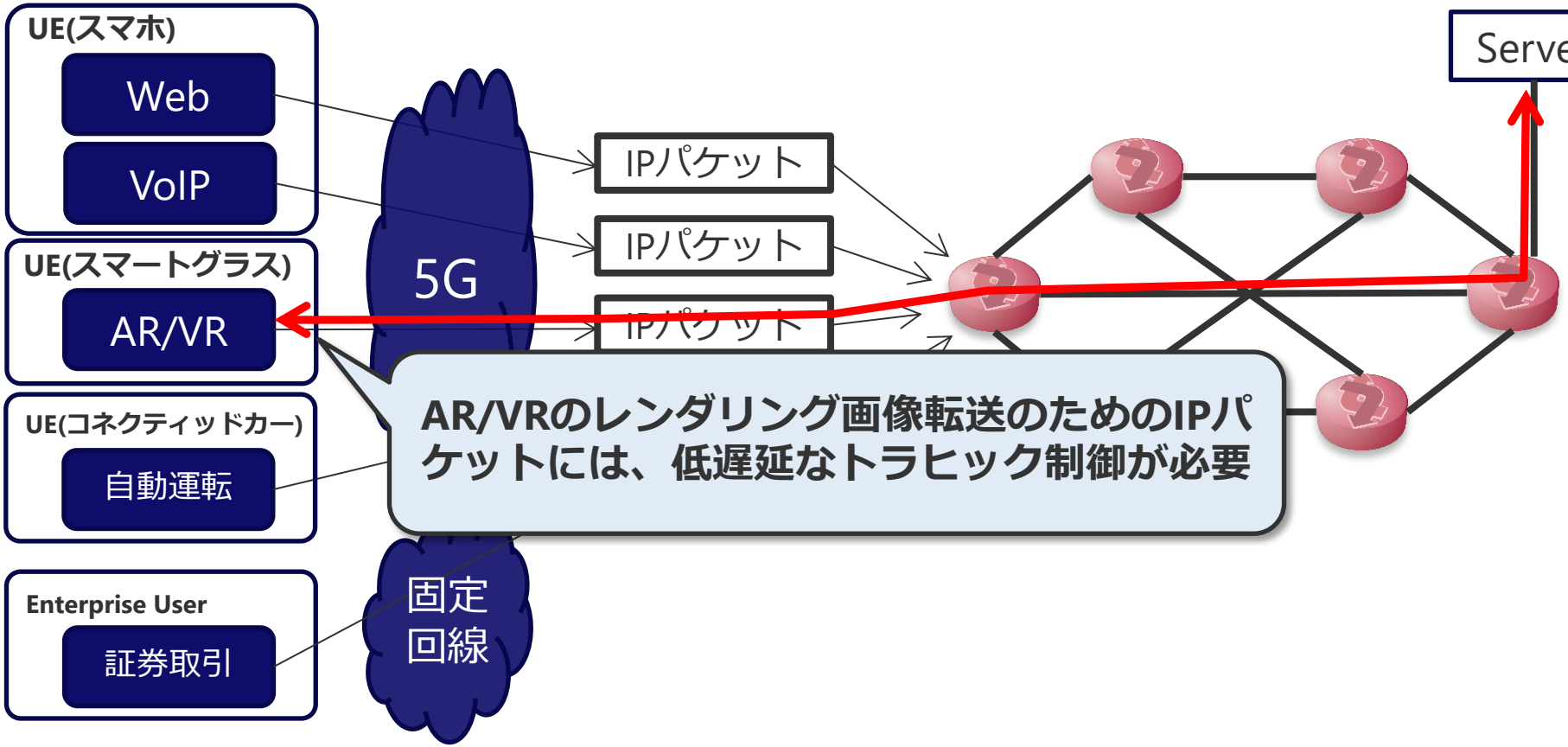
KDDI総合研究所 宮坂拓也

KDDI Research

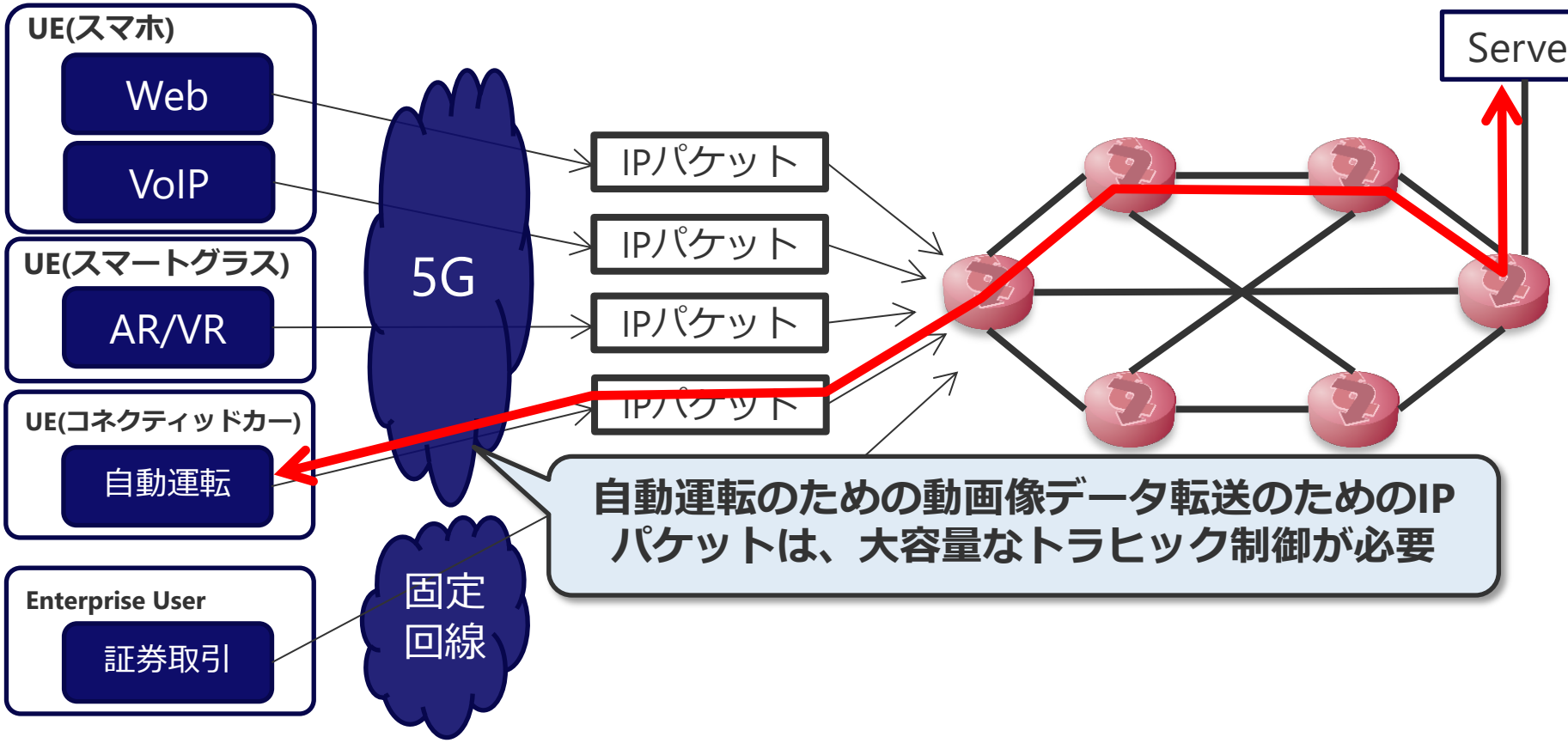
アプリケーションとネットワークオペレーター



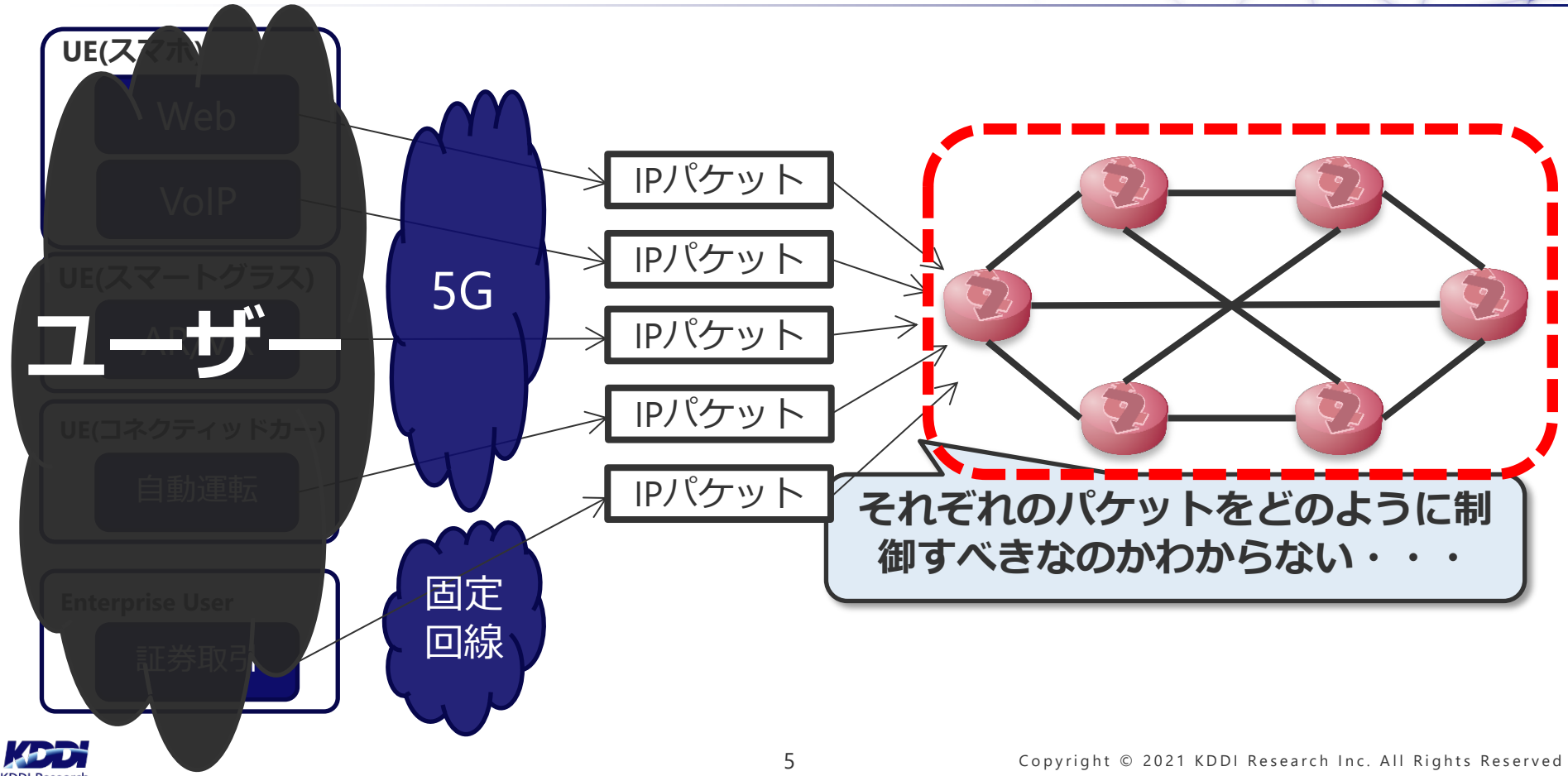
アプリケーションとネットワークオペレーター



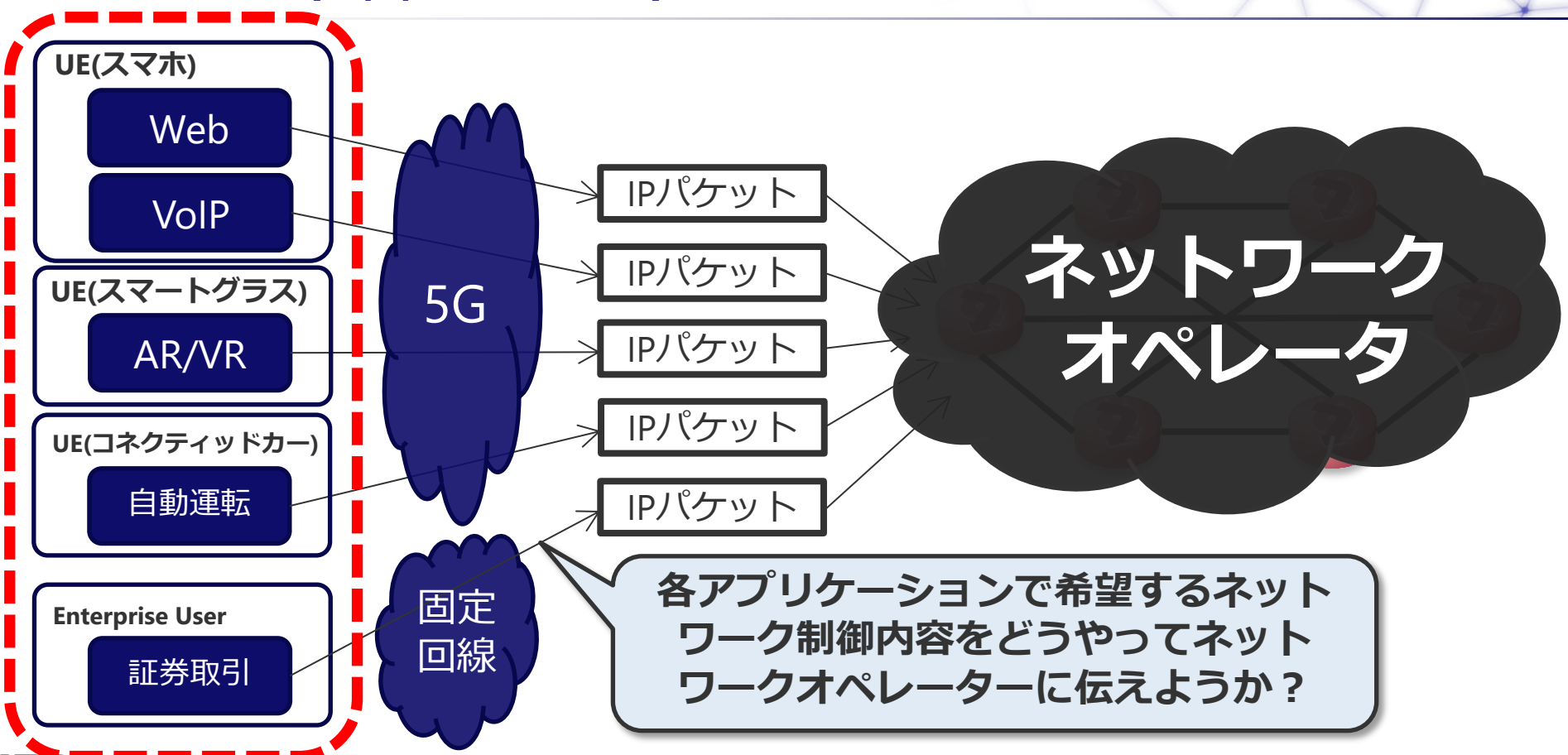
アプリケーションとネットワークオペレーター



ネットワークオペレーターの悩み



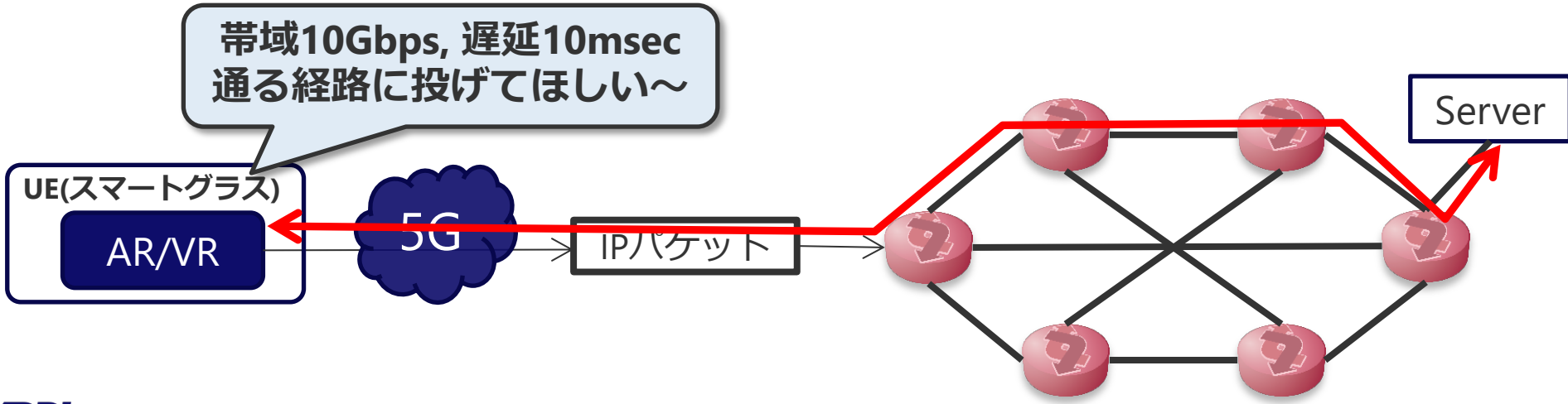
ユーザー(application)の悩み



本発表におけるApplication-aware networkingの定義

Application-aware Networking

- 本資料では、**アプリケーションの要求を満たすように、L3ネットワーク内をルーティングする技術一般**のこととする



本日の話の流れ

- 歴史を振り返る
- 標準化領域における活動紹介
- 研究領域における活動紹介
- KDDI総合研究所での取り組み紹介

歴史を振り返る

歴史を振り返る (1/3)

IntServ(RFC1633, 1984)とDiffServ(RFC2475, 1998)

- **IntServ**
  - RSVPを用いて、流れるフロー毎にQoSを実施(リソース確保等)
  - フロー毎のステートをISP内のネットワークに持つことにより、**スケーラビリティ**が問題になり、一般的には利用されなかった
- **DiffServ**
  - フロー毎ではなく、クラス単位でのQoS
  - RSVPのようなControl-planeのプロトコルはなく、IPヘッダー内のDSCPフィールドを用いる
  - 結局、こちらが広く普及

[参考] https://www.nic.ad.jp/ja/materials/iw/1998/notes/C12.pdf

歴史を振り返る (1/3)

IntServ(RFC1633, 1984)とDiffServ(RFC2475, 1998)

- **IntServ**
  - RSVPを用いて、流れるフロー毎にQoSを実施(リソース確保等)
  - フロー毎のステートをISP内のネットワークに持つことにより、**スケーラビリティ**が問題になり、一般的には利用されなかった
- **DiffServ**
  - フロー毎では
  - RSVPのようなControl-planeのプロトコルはなく、IPヘッダー内のDSCPフィールドを用いる
  - 結局、こちらが広く普及

スケーラビリティの問題により、Application-aware networkingのようなフロー毎の制御は困難という判断

[参考] https://www.nic.ad.jp/ja/materials/iw/1998/notes/C12.pdf

歴史を振り返る (2/3)

SDN, OpenFlowの隆興

- 2000年代後半頃より、**Software Defined Network**(SDN)に関する各種活動が盛んになり、**OpenFlow**の標準化がOpen Networking Foundation(ONF)にて進められた
- コントローラーによる一斉制御、フロー毎に最適な制御を実現
- 採用例：Google B4

SDNの隆興により、ネットワークの集中制御への可能性、フロー毎のネットワーク制御への機運が高まってくる？

歴史を振り返る (3/3)

Segment Routing

- 2017年頃より標準化が進められたSegment Routing(SR)では、ネットワーク内の経由ノードで維持するStateを最小限にし、送信ノードの意図するRoutingを実現可能 (もちろん送信ノードではStateを持つ必要はある)
  - IntServのスケーラビリティの問題がある程度解消する可能性が出てきた
- SR over MPLS(SR-MPLS): MPLSデータプレーンでのSR
- SR over IPv6(SRv6): IPv6データプレーン(Routing header)でのSR

IntServ導入の問題となった、ネットワーク内のState問題がSRで解消されるかもしれない？という機運の高まりも

歴史ふりかえりまとめ

☹ 2000年くらいまでは、IntServは標準化したものの、実ネットワークには、ほぼ(?)利用されなかった

☺ SDN, OpenFlowの隆興により、コントローラーによる中央制御・フロー毎の制御という機運が高まる

☺ Segment Routingの標準化により、中継ルーターの観点でスケーラブルなネットワーク制御の実現可能性が高まる

以上の時代背景とともに、Application-aware Networkingに関する研究・標準化活動が近年盛んになっているとも読み取れる

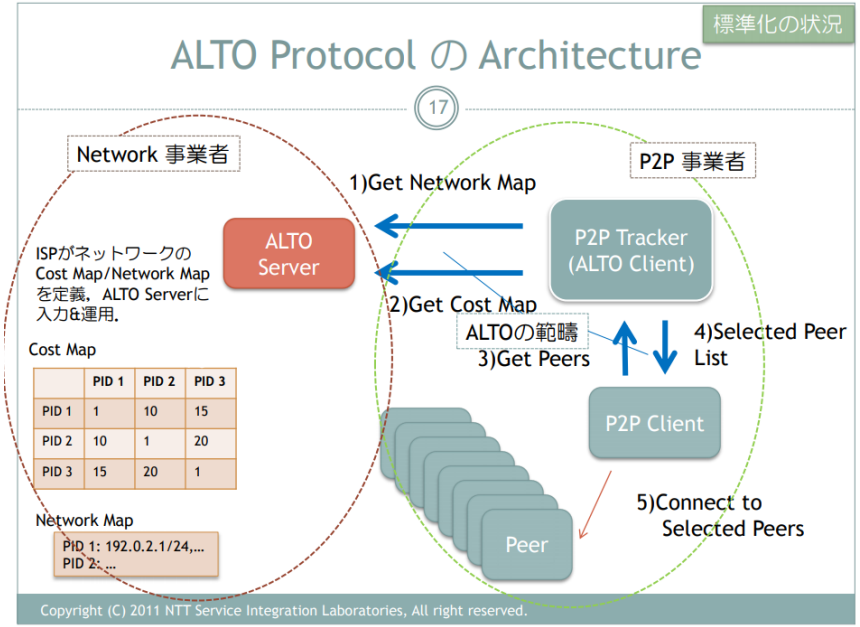
標準化領域における活動紹介

KDDI Research

IETF Application-Layer Traffic Optimization (ALTO)

IETF ALTO

- 2008年頃より**IETF ALTO WG**にてプロトコル標準化が進められている
- (1) ISPがアプリケーション側に対して抽象化したネットワーク情報を通知し、(2) アプリケーションがそれをもとにして最適な制御を行うモデル
- 当初はP2P通信への適用を前提に構築されたプロトコルであるが、現在ではCDNなどへの適用も考えられている



[参考] http://www.fmmc.or.jp/P2P/pub/sympo/pdf2011/S1-2-2.pdf

[参考] https://datatracker.ietf.org/wg/alto/about/

IRTF Path Aware Networking (PAN) RG

IRTF PAN RG

- 2017年に設立したIRTFのResearch Group (RG)
- End nodeに対して通信パスの品質情報を通知し、その情報をもとにEnd nodeが適切な制御を実施することが目的
  - ALTOでもいいのでは・・？
- 現状では、「PANの定義」についてもまだ不明確な状態で、研究成果の共有や、過去のPAN関連の取り組みについて整理している状況

過去の取り組みが普及しなかった理由を整理...

5. Contributions 12
5.1. Stream Transport (ST, ST2, ST2+) 12
5.1.1. Reasons for Non-deployment 13
5.1.2. Lessons Learned. 13
5.2. Integrated Services (IntServ) 13
5.2.1. Reasons for Non-deployment 14
5.2.2. Lessons Learned. 15
5.3. Quick-Start TCP 15
5.3.1. Reasons for Non-deployment 16
5.3.2. Lessons Learned 17
5.4. ICMP Source Quench 18
5.4.1. Reasons for Non-deployment 18
5.4.2. Lessons Learned 18
5.5. Triggers for Transport (TRIGTRAN) 19
5.5.1. Reasons for Non-deployment 20
5.5.2. Lessons Learned. 21
5.6. Shim6 21
5.6.1. Reasons for Non-deployment 22
5.6.2. Lessons Learned 23
5.6.3. Addendum on MultiPath TCP 23
5.7. Next Steps in Signaling (NSIS) 23
5.7.1. Reasons for Non-deployment 24
5.7.2. Lessons Learned 25
5.8. IPv6 Flow Label 26
5.8.1. Reasons for Non-deployment 27
5.8.2. Lessons Learned 28

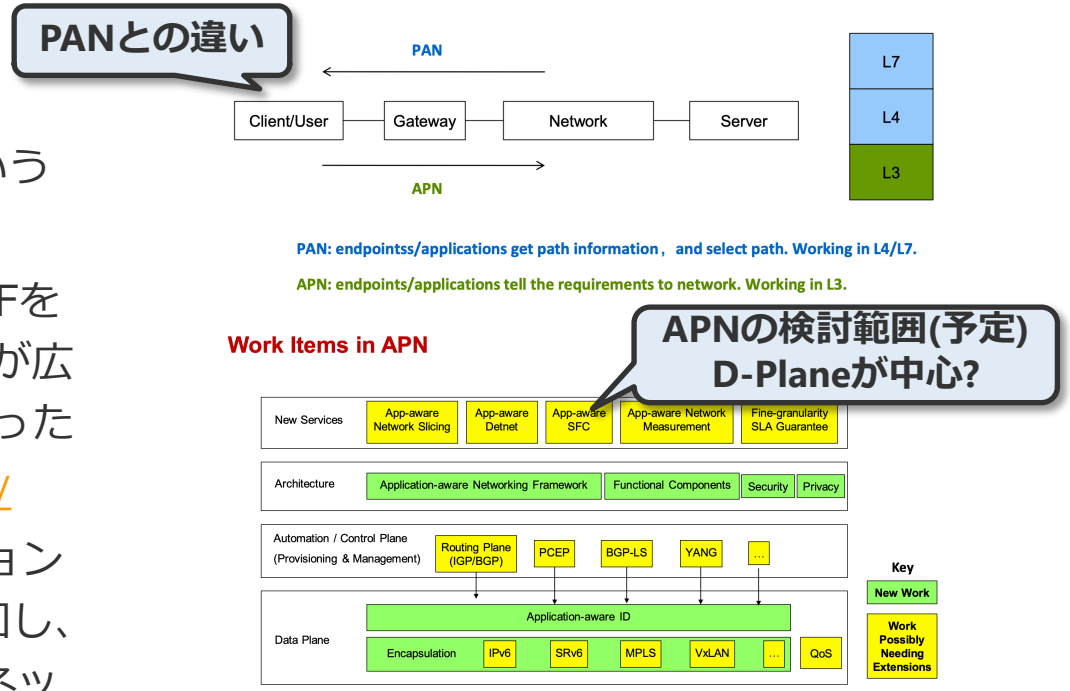
[参考] https://datatracker.ietf.org/doc/draft-irtf-panrg-what-not-to-do/

[参考] https://datatracker.ietf.org/rg/panrg/about/

IETF Application-aware Networking (APN)

IETF APN

- IETF105,108にてside-meetingという形で実施
  - 2020年11月開催のIETF109でBoFを申請したが、ユースケースの範囲が広すぎるとのことで、実施されなかった
    - https://ietf.org/blog/ietf109-bofs/
- 活動の目的としては、アプリケーションからネットワークへ、SLAなどを通知し、それに基づいたルーティング制御をネットワーク側が実施する機構を作ること



[参考] https://github.com/APN-Community/IETF108-Side-Meeting-APN

[参考] https://www.ietf.org/proceedings/interim-2020-panrg-01/slides/slides-interim-2020-panrg-01-sessa-application-aware-networking-and-path-awareness-00

New IP?

New IPにも同様なロードマップ

User-defined Customized Request for Networks

Packet Level Network Programming

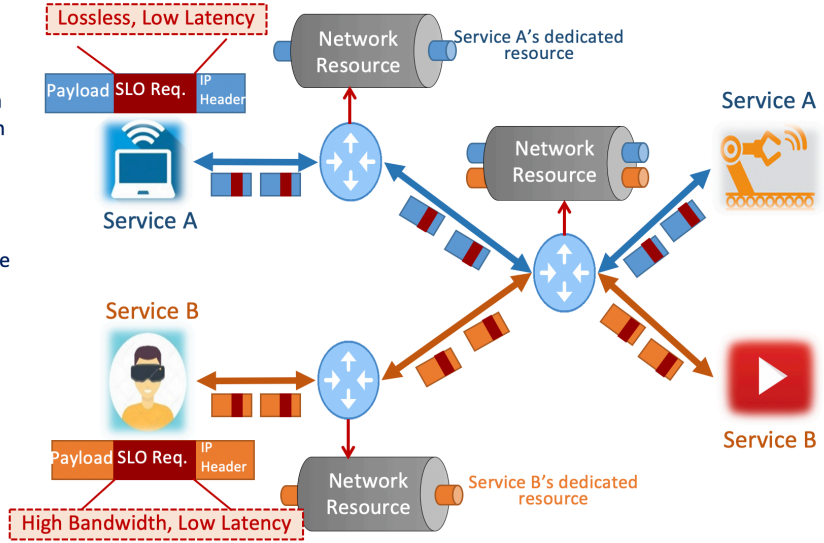
- Packet carries instructions, metadata and executing permission to program network behaviors

In-Band Signaling

- combine control plane and data plane capabilities together.

User/Application Granularity In-Band OAM

- enables fine-grained and accurate performance monitoring to adjust SLOs



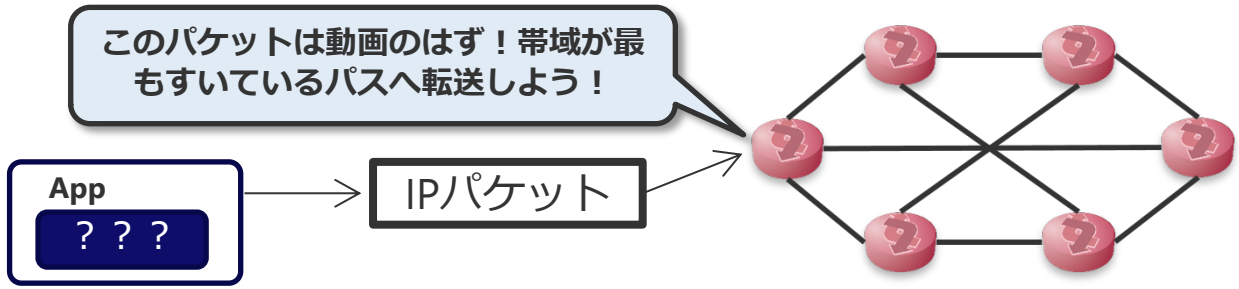
HUAWEI

[参考] https://datatracker.ietf.org/liaison/1653/

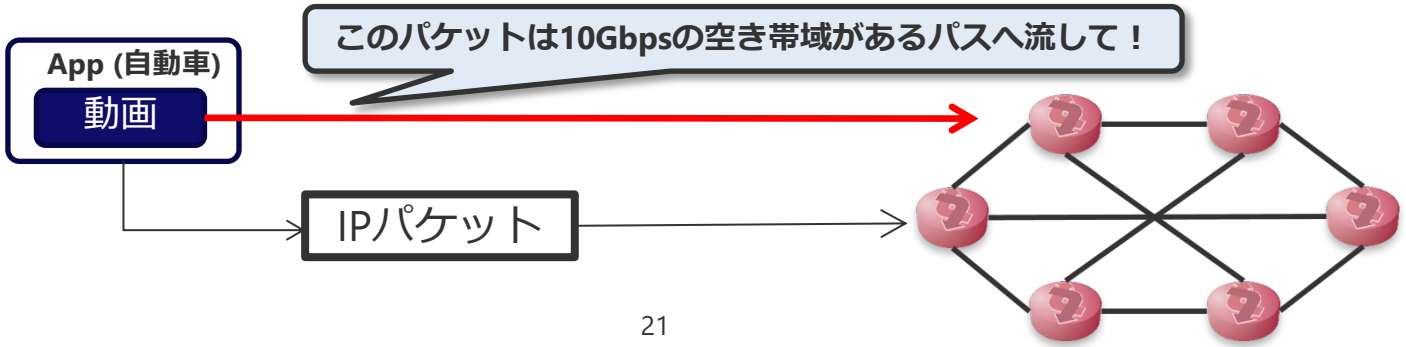
研究領域における活動紹介

2つのタイプが存在する

1. Network主導型



2. Application主導型



Network主導型

1. Network側でアプリケーション種別を推定
   - Network側で、受信IPパケットからアプリケーション種別を推定し、その種別に適切なトラヒック制御を行う手法

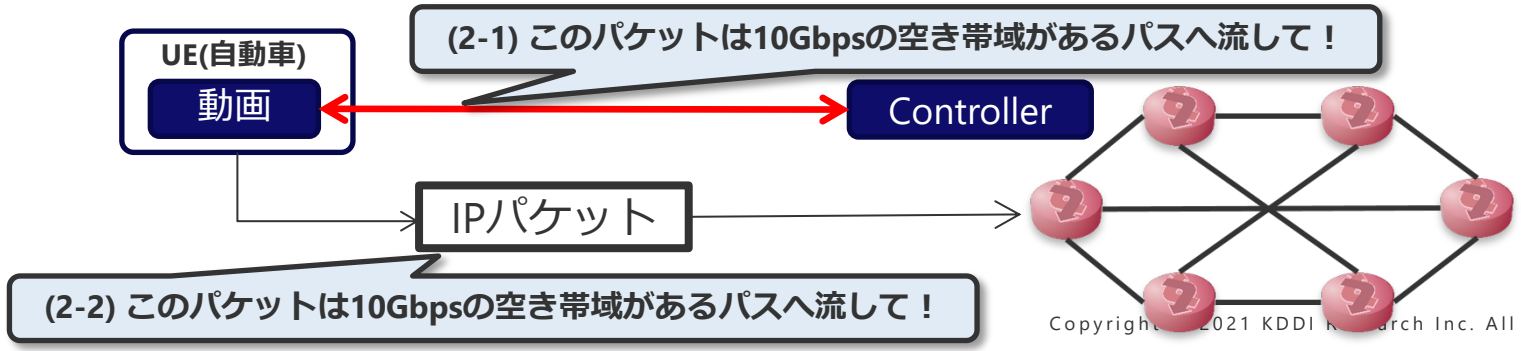
手法群	潜在的な課題
(1-1) Deep packet inspection(**DPI**)による推定	TLSなどでパケットが暗号化されているとアプリケーション種別の推定精度が落ちる、または推定不可能となる
(1-2) 機械学習による推定	100%の精度でアプリケーション種別推定が難しい

Application主導型

2. Applicationが自ら通知する手法

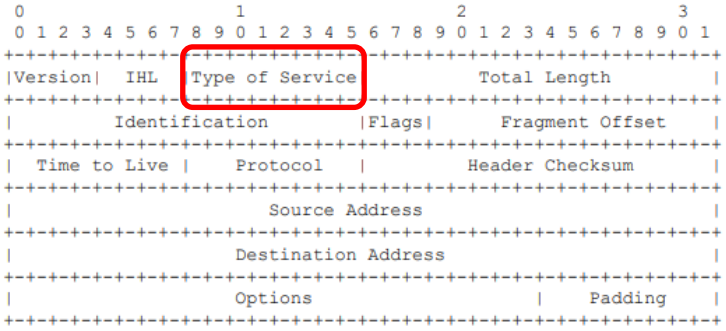
- ApplicationがNetwork側にトラフィック制御の要求事項を通知し、Networkがそれに合わせたルーティングを実施する

手法群	潜在的な課題
(2-1) Out-bandに制御要求を通知する手法 (例: API)	API実行のための追加オーバーヘッドが発生 アプリケーションへ制御要求実装が必要
(2-2) In-bandに制御要求を通知する手法 (例: パケットヘッダ)	既存ヘッダ(IPv4/IPv6)では十分なビット長がなく、複雑なトラヒック制御要求を通知できない



十分なビット長がない？

IPv4

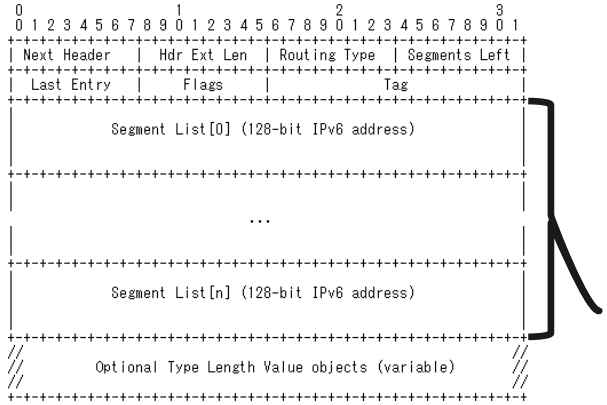


IPv6

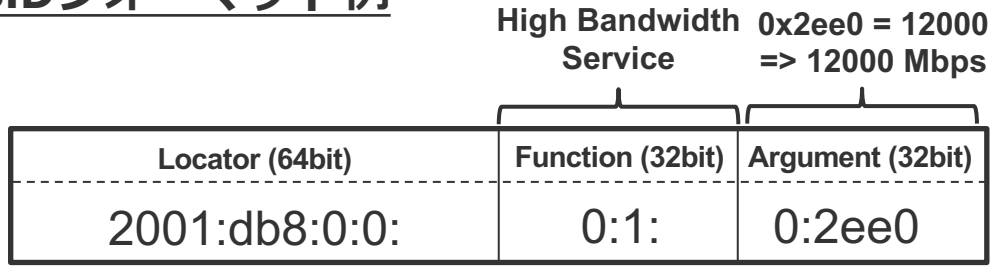
IPv4/IPv6のDSCP(6bit)、IPv6 Flow Label(20bit)を用いた先行研究例があるが、ビット長が小さく、複雑なトラヒック制御要求が困難

SRv6: Segment Routing over IPv6

IPv6 SRH (RFC 8754)



SIDフォーマット例



IPv6アドレスの形で制御の指示ができる

- IETFのspring WG, 6man WGにおいて積極的に標準化が進められているルーティング技術
  - IPv6の拡張ヘッダであるRouting Headerの新しいTypeである、Segment Routing Header (SRH)(RFC8754)を定義
  - 送信者は**128bitのIPv6アドレス**の形で、実施してほしい制御内容(SID)を指定
    - アプリケーションが複雑なトラヒック制御要求をするのに十分なビット長では？
    - Locator/Function/Argumentによりトラヒック制御を指定

KDDI総合研究所での取り組み紹介
NetworkAPI

IEICE IN研究会(依頼講演)：https://www.ieice.org/ken/paper/20200521G1xW/
ACM Workshop on Network Application Integration/CoDesign: https://dl.acm.org/doi/10.1145/3405672.3405805

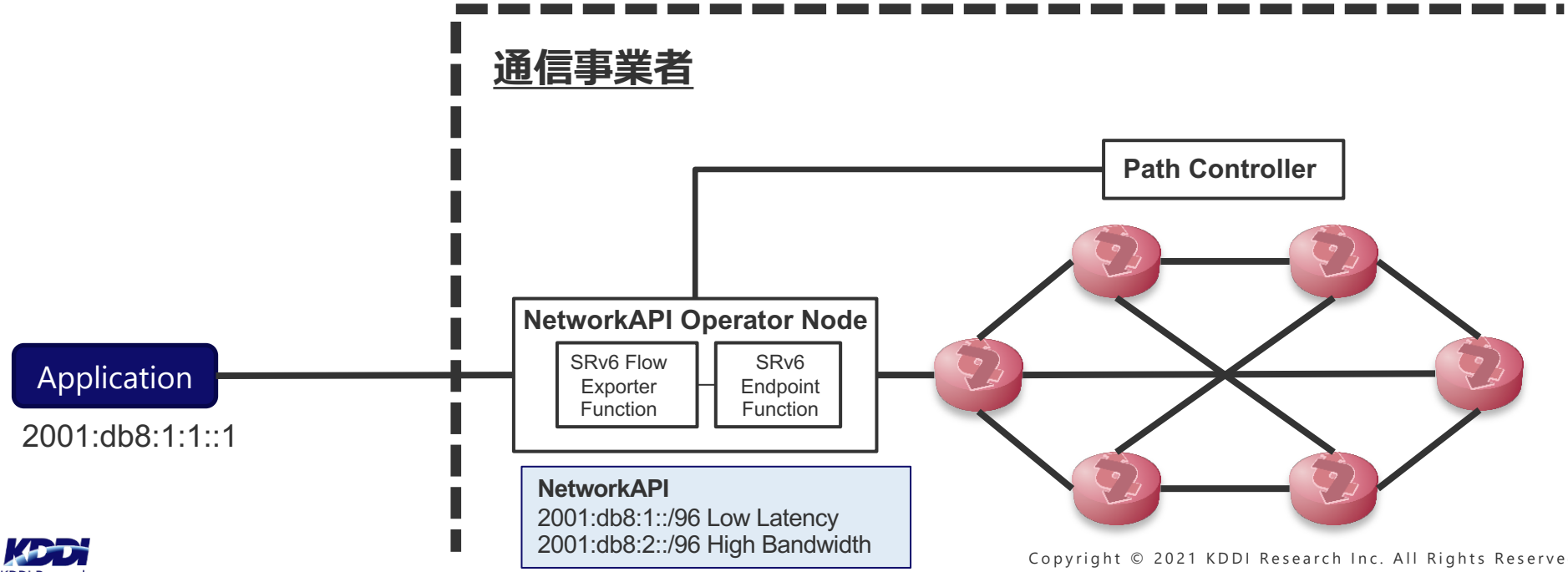
KDDI Research

NetworkAPI

基本コンセプト

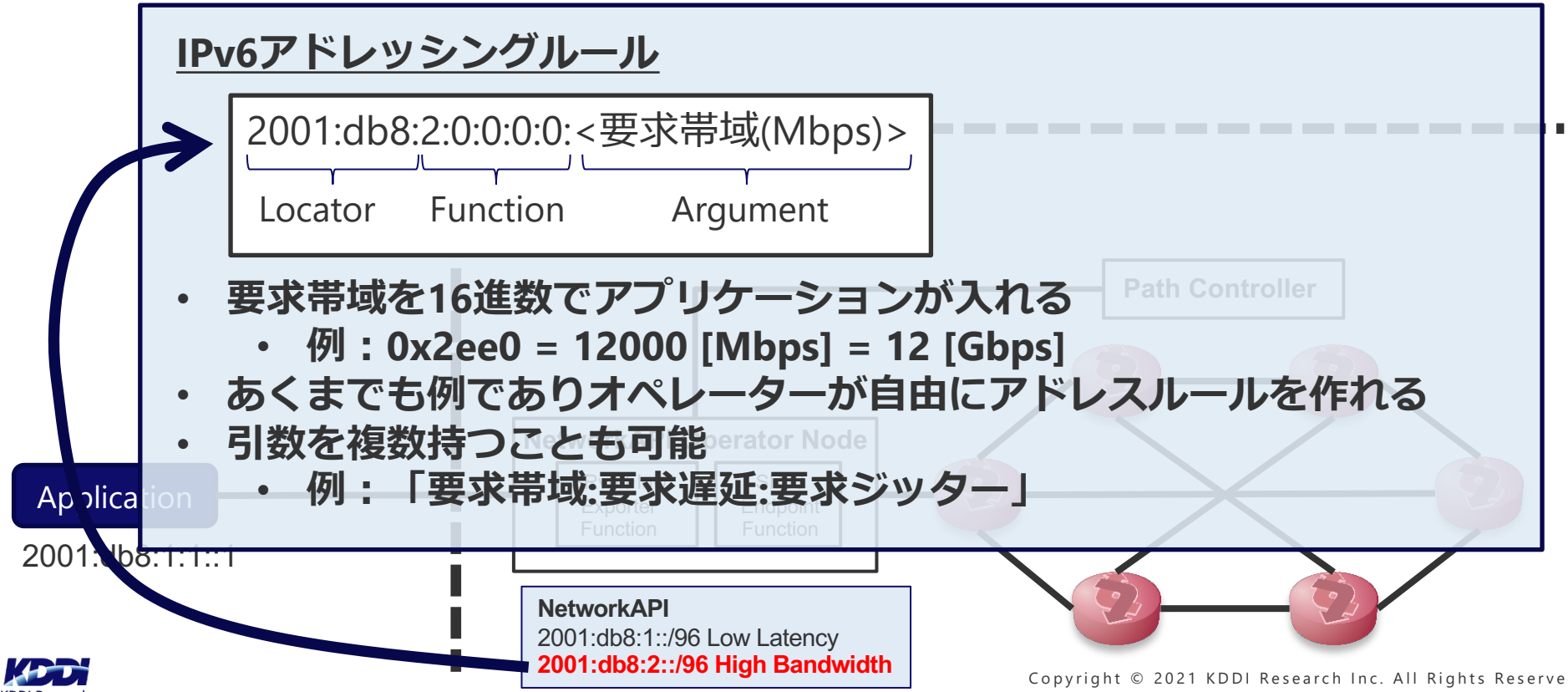
- **SRv6**を用いて、IPv6アドレスの形式で通信事業者が**トラヒック制御サービスをユーザーへ提供**する
  - IP anycastにより、トラヒック制御を示す**IPv6アドレスはアプリケーションのロケーションに依存せず一意**な値となる

NetworkAPI Framework



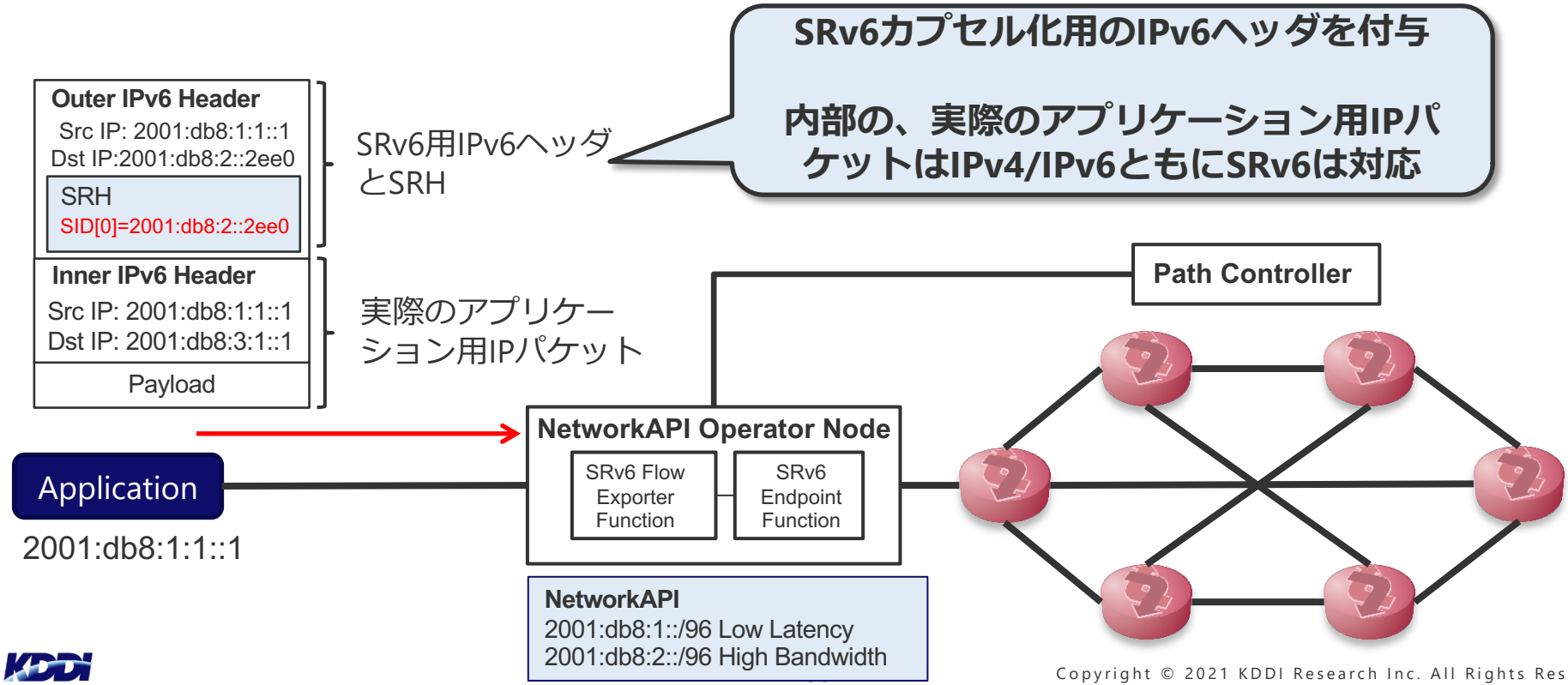
KDDI Research

NetworkAPI Framework



NetworkAPI Framework

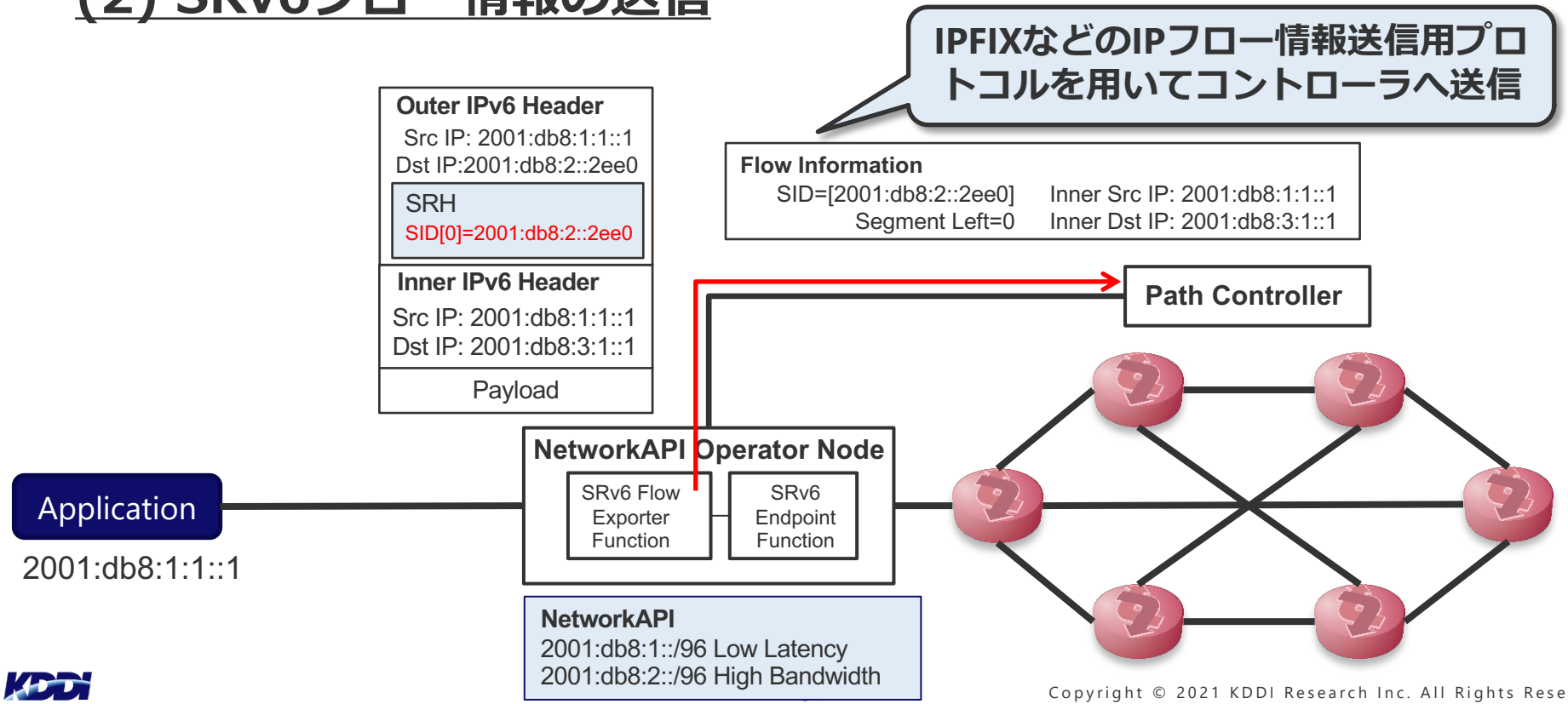
(1) SRv6によるカプセル化



KDDI Research

NetworkAPI Framework

(2) SRv6フロー情報の送信



KDDI Research

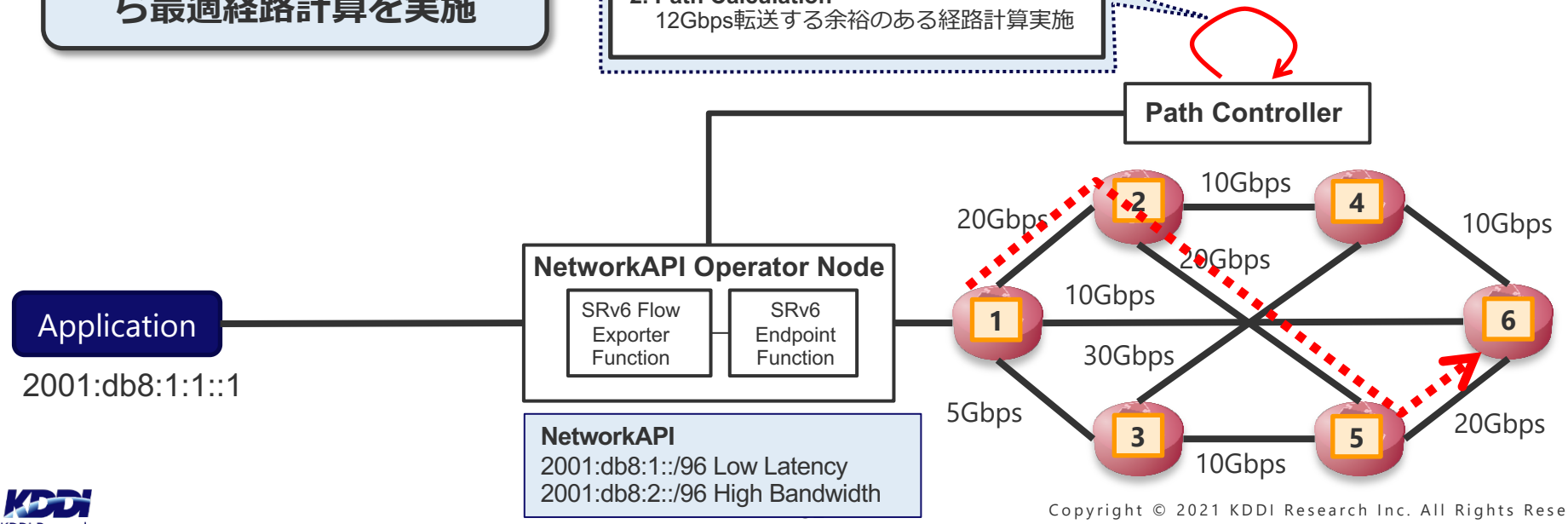
NetworkAPI Framework

(3) 経路計算

Flow Information
SID=[2001:db8:2::2ee0]　Inner Src IP: 2001:db8:1:1::1
Segment Left=0　Inner Dst IP: 2001:db8:3:1::1

現在のネットワーク状況から最適経路計算を実施

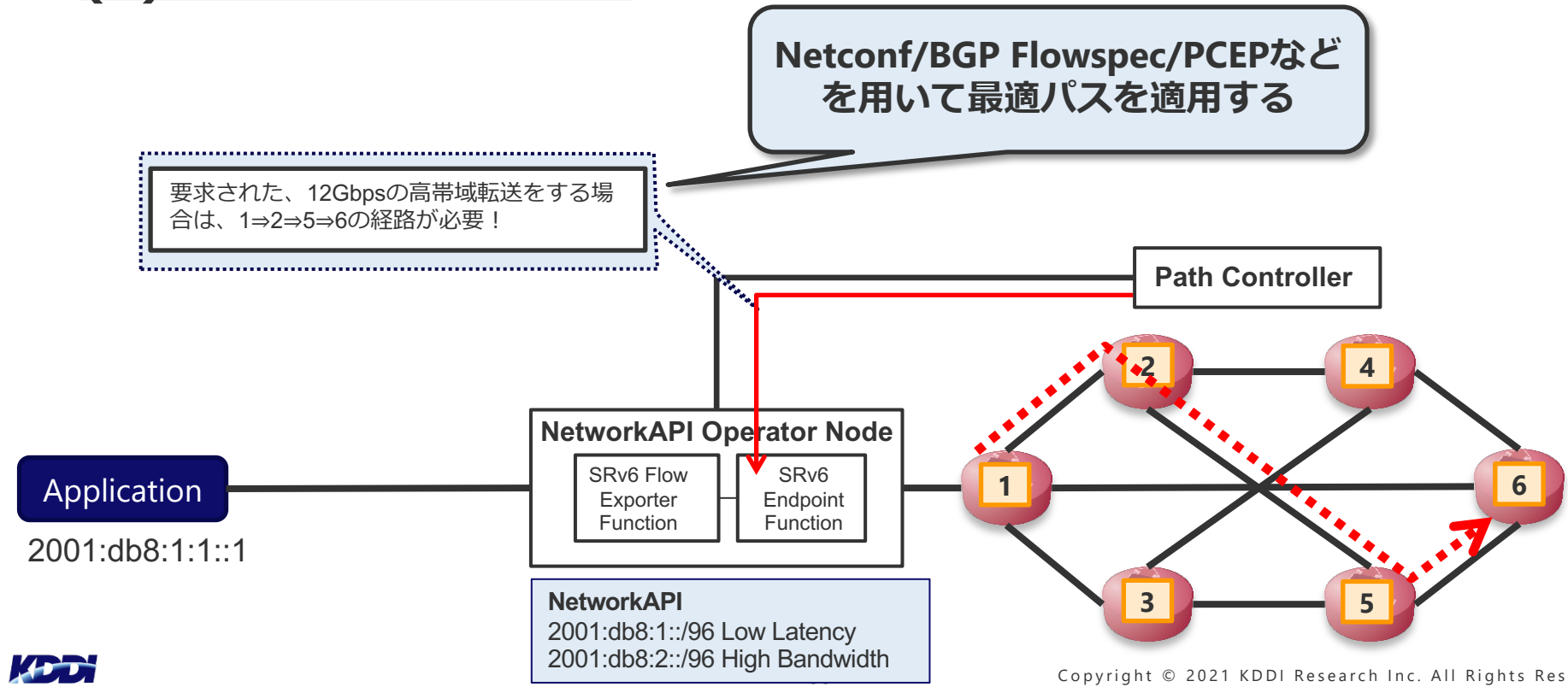
1. **SID Lookup**
   2001:db8:2::/96は高帯域転送サービス！
   引数の0x2ee0=12000[Mbps]＝12Gbps！
2. **Path Calculation**
   12Gbps転送する余裕のある経路計算実施



KDDI Research

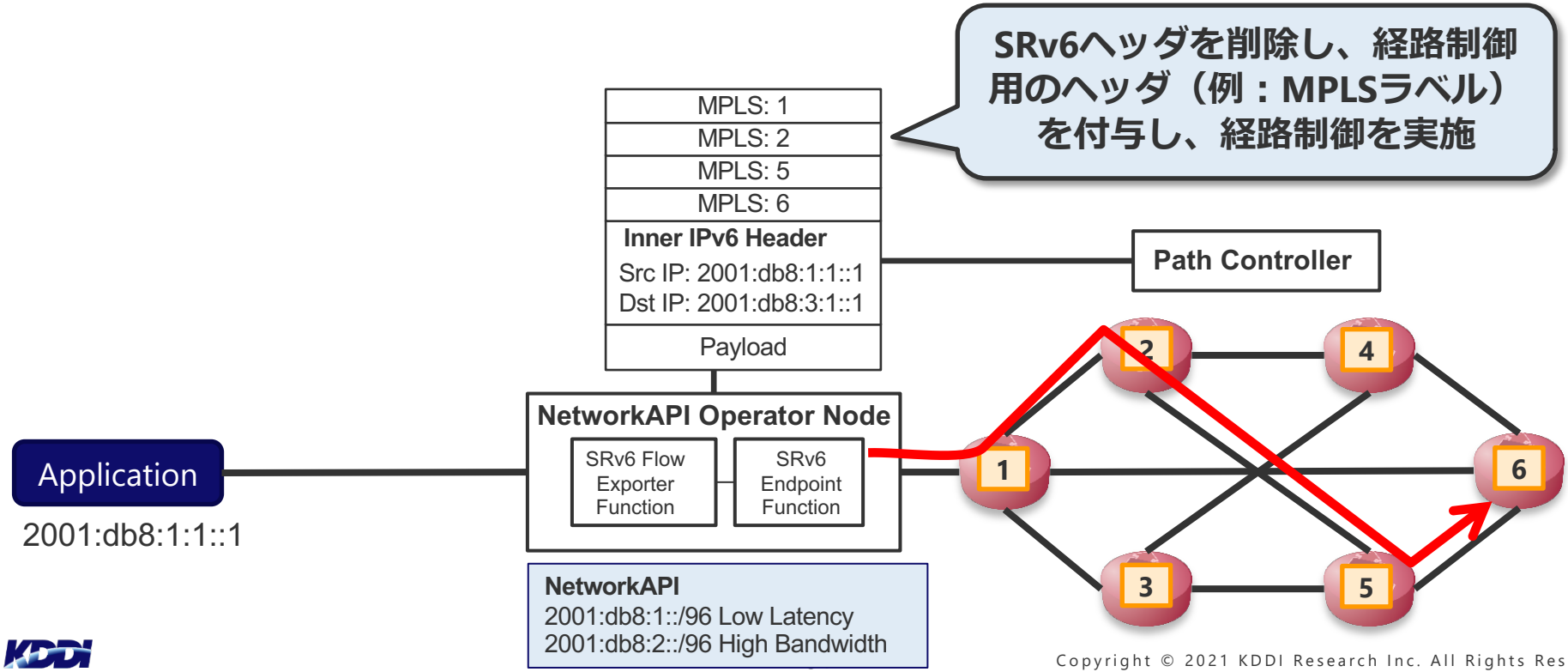
NetworkAPI Framework

(4) 経路計算結果の適用



NetworkAPI Framework

(5) SRv6パケット処理と経路制御実施

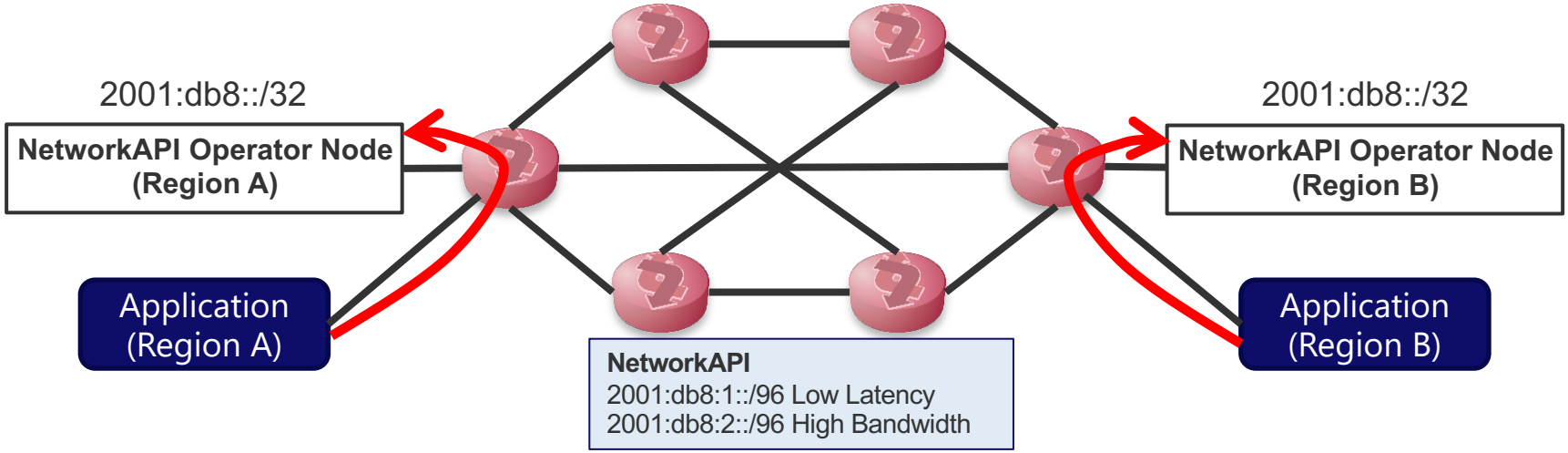


KDDI Research

IP anycast in NetworkAPI Framework

IP anycastにより、SIDのLocatorに関するPrefixを各地域で広報することで、アプリケーションは自身のロケーションに関わらず、一意なIPv6アドレスを利用可能

- これがないと、アプリケーションは適時通信事業者に対して現在のIPv6アドレスが正しいか確認が必要になってしまう

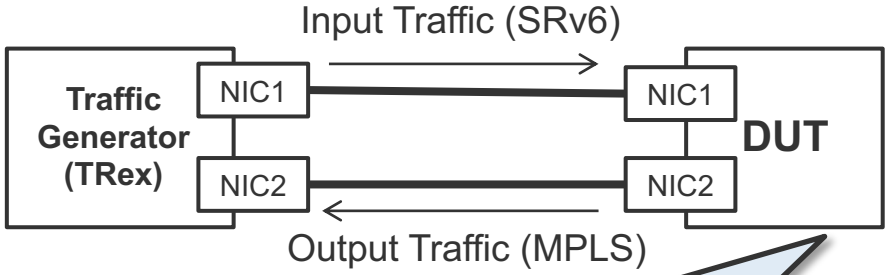


プロトタイプ実装と評価環境

プロトタイプ実装

- Linux kernel上にて実装
  - SRv6関連パケット処理機構はLinux kernelで標準に搭載されているものを利用
    - SRv6 End.DT6ファンクションを利用
  - SRv6フロー情報送信はLinux kernel moduleとして開発されているOSSのipt-netflowをベースに作成
    - プロトコルとしてIPFIXを利用
    - https://github.com/aabc/ipt-netflow

評価環境



機器情報

- Traffic Generator
  - CentOS7.6
  - TRex
- DUT
  - Ubuntu18.04 (Linux kernel 5.0.0)

プロトタイプ実装のパケット転送性能評価

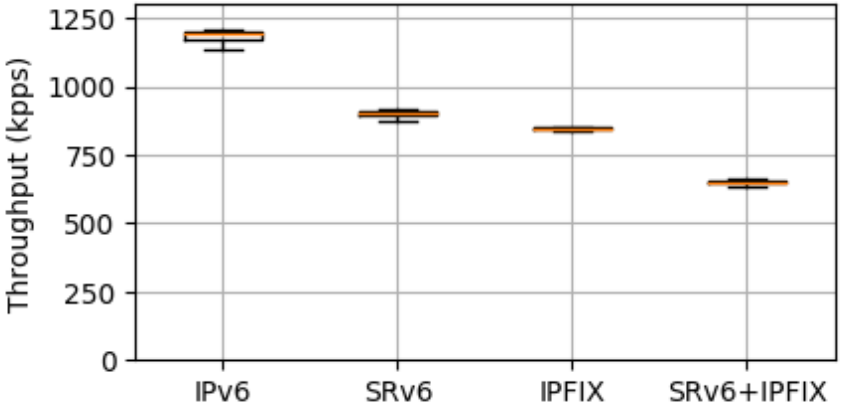
パケット転送性能評価

- 比較手法

IPv6	**性能比較のベース**。DUTはSRv6/IPFIX処理をせずに、IPv6パケット転送のみ実施
SRv6	DUTはSRv6処理(SRv6ヘッダのカプセル化解除＋MPLSラベル付与)のみ実施
IPFIX	DUTはIPFIX処理(SRv6フロー情報の送信)のみ実施
SRv6+IPFIX	**提案手法**。DUTはSRv6/IPFIX処理ともに実施

- 送信IPv6(SRv6)フロー情報
  - パケットサイズ：128byte
  - 10万フロー
- 各手法100回スループット測定を実施

評価結果

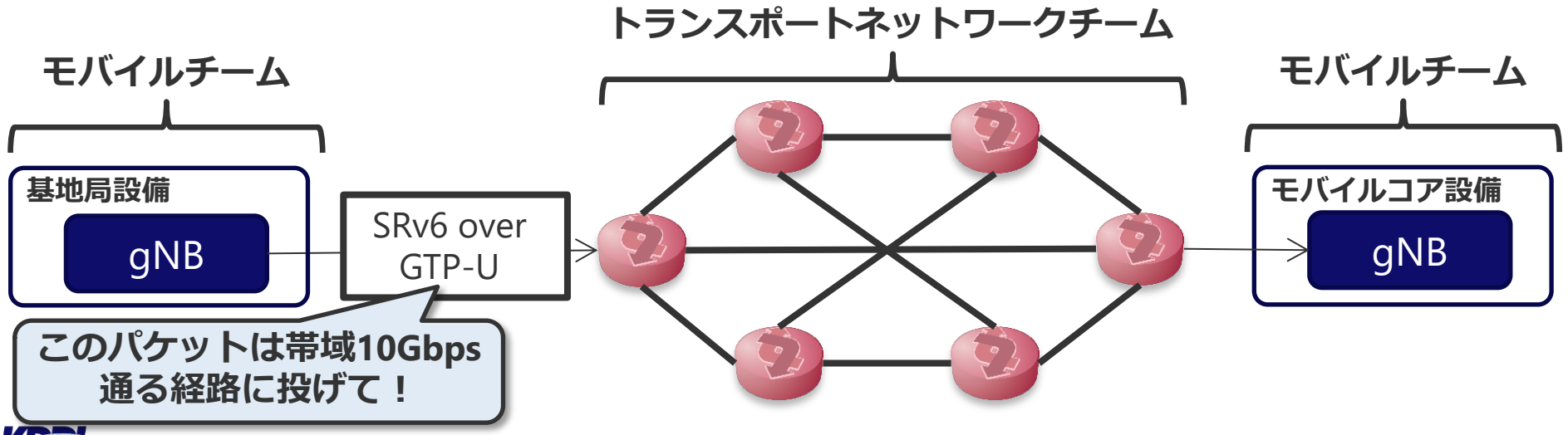


	IPv6	SRv6	IPFIX	SRv6 +IPFIX
平均スループット[kpps]	1183.44	899.573	844.508	**647.487**
IPv6との比較	1.000	0.7601	0.7136	**0.5471**

社内向け提供でも便利そう？

社内で「ネットワーク担当チーム」「サービス・アプリケーション担当チーム」が分かれている場合にも使える、むしろ顧客に提供する前に最初はそういうところから？

- 今までは、人手のコミュニケーションをベースにネットワーク側の設定変更等をしていたが、とても大変…☹



まとめ

まとめ

本資料では、**Application-aware Networking**について(1)過去と現在を整理し、(2)研究例の紹介を実施しました

最後に、特に聞いてみたいこと…

- To ネットワークオペレーターの皆さま
  - 実際にここまで細かな制御など要求されたりします？
- To コンテンツプロバイダーの皆さま
  - ネットワーク側へどんな要望がありますか？
  - 開発にあたり「ここはこうしたい」等ありますか？
- もちろん他のご意見も大歓迎です！