JANOG40 Meeting in Fukushima

Segment Routing チュートリアル

シスコシステムズ合同会社
鎌田 徹平 / 竹田 直哉
2017 年 7 月 18 日

Disclaimer

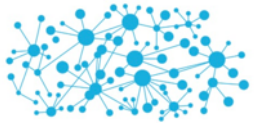
- 本資料の内容は 2017 年 7 月 18 日時点の最新情報です
- 将来にわたり情報・仕様が更新される可能性があります

アジェンダ

- Segment Routing 概要
- SR-MPLS
  - 基本動作：コントロールプレーンとデータプレーン
  - 高速迂回：TI-LFA FRR
  - トラフィックエンジニアリング：SR-TE
- SRv6
  - SRv6 データプレーン：SR-MPLS との違い
  - SRv6 ネットワークプログラマビリティ
- ユースケース
- まとめ

Segment Routing 概要

Simple



Scalable

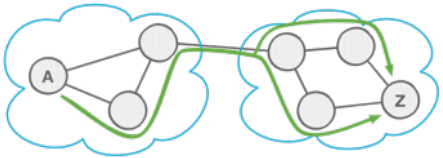
Seamless deployment

SR

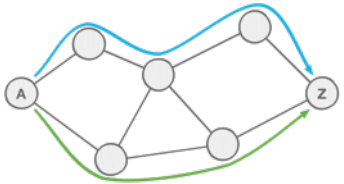
Segment Routing

www.segment-routing.net

Network Programming



Unified Forwarding Plane



Traffic Engineering

Segment Routing とは？

ネットワークを Segment で表現し、シンプルで柔軟な制御を実現するルーティング

IGP (OSPF/IS-IS) のみで実現	→	コントロールプレーンのシンプル化
ステートレス	→	機器負荷・運用負荷の軽減
ソースルーティング	→	送信元における柔軟なパス選択
プログラマビリティ	→	自動化との親和性、サービスの表現
高速迂回	→	動的に最適パスを考慮して設定される FRR
シームレスマイグレーション	→	LDP, RSVP からのマイグレーションが容易
データプレーン非依存	→	MPLS または IPv6

Segment Routing 標準化動向

- IETF 標準化 : SPRING working group
- プロトコル拡張 : 複数のグループで進展
  - IS-IS
  - OSPF
  - PCE
  - IDR
  - 6MAN
- 幅広いベンダーとカスタマーが支持

→ オープンなプロトコル

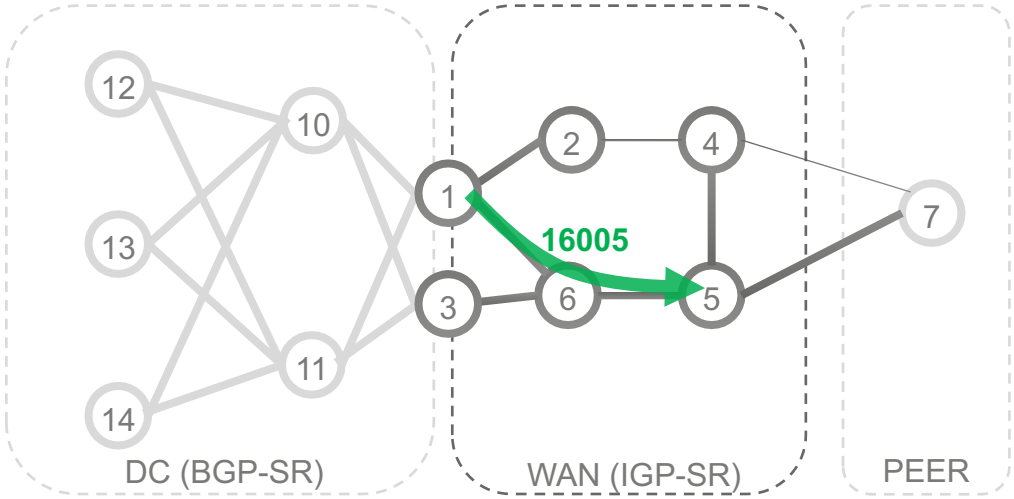
Sample IETF Documents
Problem Statement and Requirements (RFC 7855)
Segment Routing Architecture (draft-ietf-spring-segment-routing)
IPv6 SPRING Use Cases (draft-ietf-spring-ipv6-use-cases)
Segment Routing with MPLS data plane (draft-ietf-spring-segment-routing-mpls)
Topology Independent Fast Reroute using Segment Routing (draft-francois-rtgwg-segment-routing-ti-lfa)
IS-IS Extensions for Segment Routing (draft-ietf-isis-segment-routing-extensions)
OSPF Extensions for Segment Routing (draft-ietf-ospf-segment-routing-extensions)
PCEP Extensions for Segment Routing (draft-ietf-pce-segment-routing)

40近い IETF ドラフトが進展中

Segment の種類： Prefix Segment

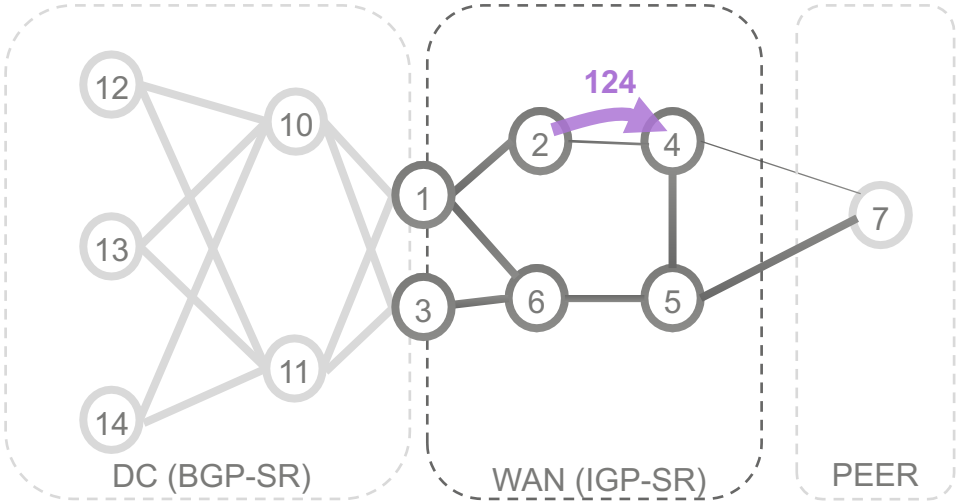
- OSPF または IS-IS によって広報される
- 特定の IGP prefix までの最短パスを表す（ノードを表現可能＝Node Segment）
- 16000 + Index

  （IGP ドメイン内でユニークな値）



Segment の種類: Adjacency Segment

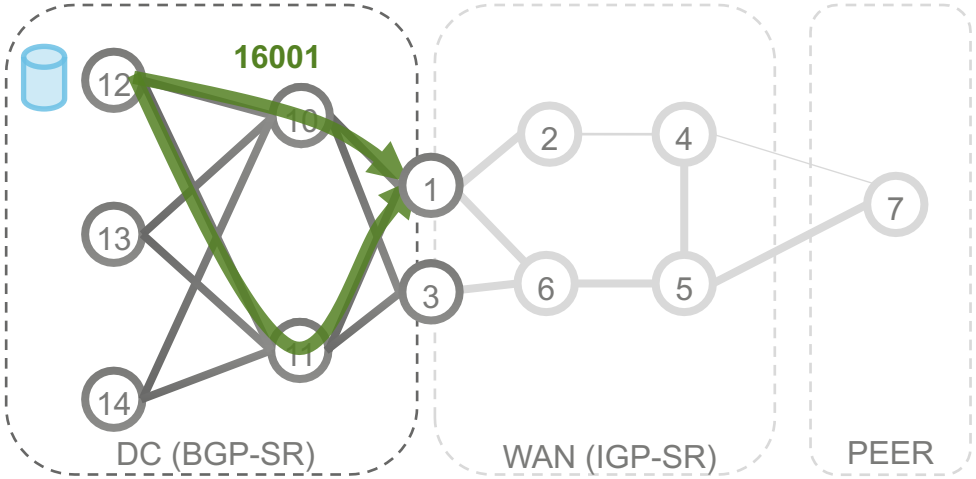
- OSPF または IS-IS によって広報される
- 2 つのノード間の特定リンクを表現
- 1XY
  - X is the “from”
  - Y is the “to”



Segment の種類: BGP Prefix Segment

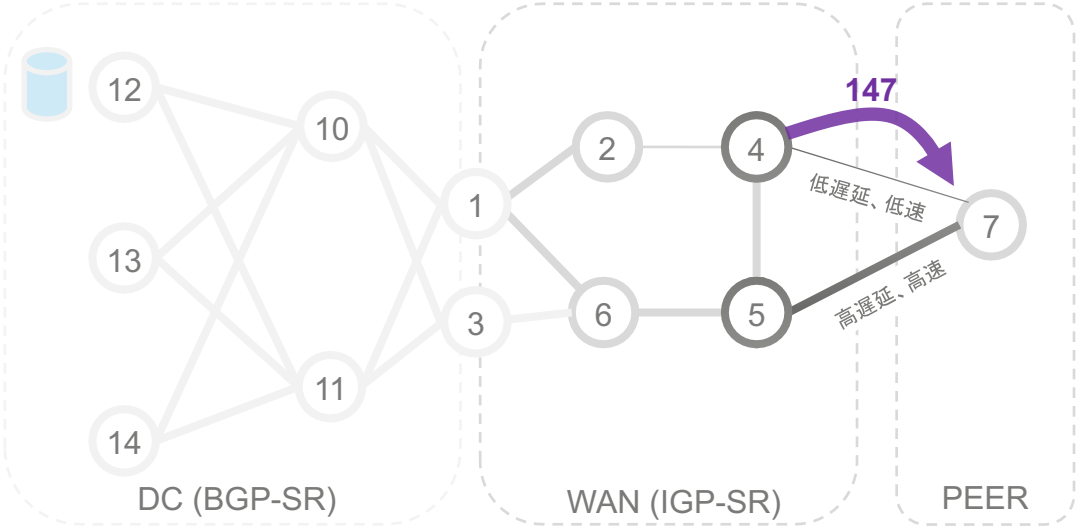
- BGP によって広報される
- 特定の BGP prefix までの最短パスを表す
- 16000 + Index

  （IGP ドメイン内でユニークな値）



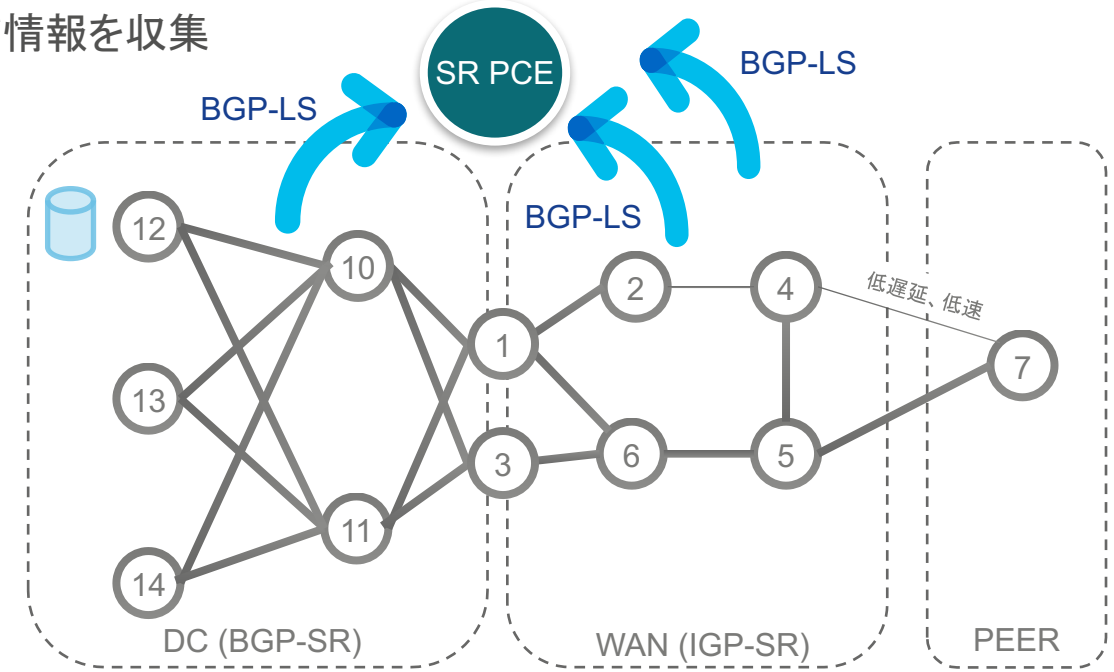
Segment の種類: BGP Peering Segment

- BGP-LS によって広報される
- 特定の BGP peer に転送する
- 1XY
  - X is the “from”
  - Y is the “to”



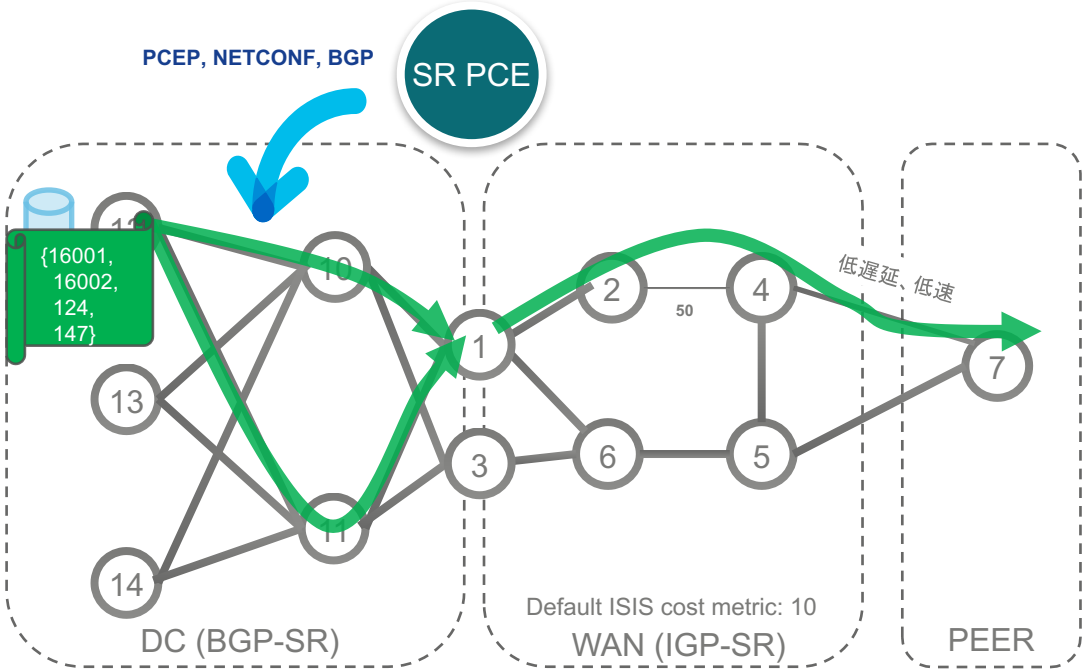
WAN コントローラとの親和性

- SR PCE が BGP-LS によって情報を収集
  - IGP segments
  - BGP segments
  - Topology



Segment リストによる制御

- SR PCE で計算
- 緑色のパスは下記のようにエンコードされる
  - 16001
  - 16002
  - 124
  - 147
- SR PCE は End-to-End で計算されたアプリケーション毎に制御されたパスをフロー単位で設定する



Segment Routing のデータプレーン

Segment Routing のデータプレーンには MPLS と IPv6 の 2 つがある

MPLS: Segment リストは **ラベルスタック** で表現される

IPv6: Segment リストは **IPv6 拡張ヘッダ** で表現される

SR-MPLS Tutorial

シスコシステムズ合同会社

竹田 直哉
ntakeda@cisco.com

SR-MPLS

コントロールプレーンとデータプレーン

SR-MPLS のコントロール/データプレーン

コントロールプレーン

- Segment によってラベルパスを表現
- プロトコルは IGP (OSPF, IS-IS) のみ
- ラベル配布プロトコル (RSVP, LDP) は不要

データプレーン

- 従来の LFIB 構造を利用 → 最小限のインパクト

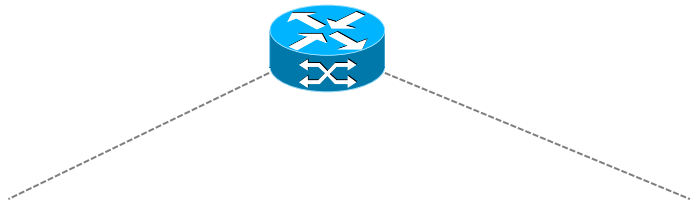
SID (Segment ID)

Prefix SID

- SID は **インデックス** としてエンコードされる
- インデックスは SRGB* からのオフセットを表す
- グローバルユニーク
- **Node SID** としても使われる

Adjacency SID

- SID は **絶対値** としてエンコードされる (インデックス値ではない)
- 各 Adjacency に対して自動的に値が割り当てられる
- ローカルユニーク



SRGB = [16000 - 23999]. Advertised as base = 16,000, range = 7,999
Prefix SID = 16041. Advertised as Prefix SID Index = 41
Adjacency SID = 24000. Advertised as Adjacency SID = 24000

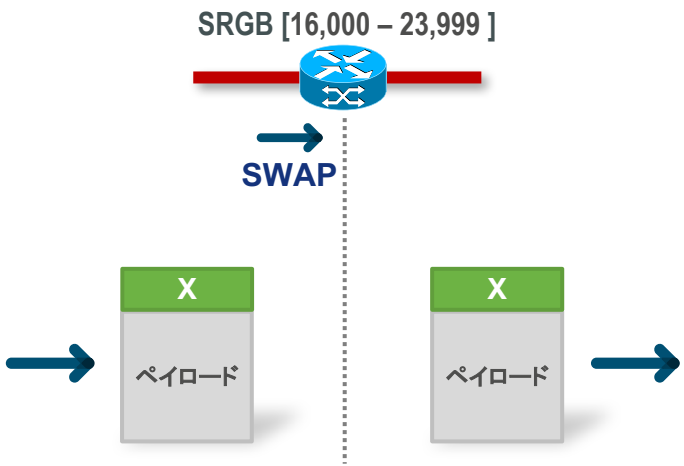
(*) SRGB: Segment Routing Global Block
Global Segment 向けのローカルラベル用に予約された 32-bit の SID スペース

SR OSPF コントロールプレーンの動作

- Loopback インタフェースのホストルートに **Prefix-SID** を割り当て
- **Adjacency SID** を各 Adjacency 毎に配布
- MPLS PHP と explicit-null ラベルをシグナリング

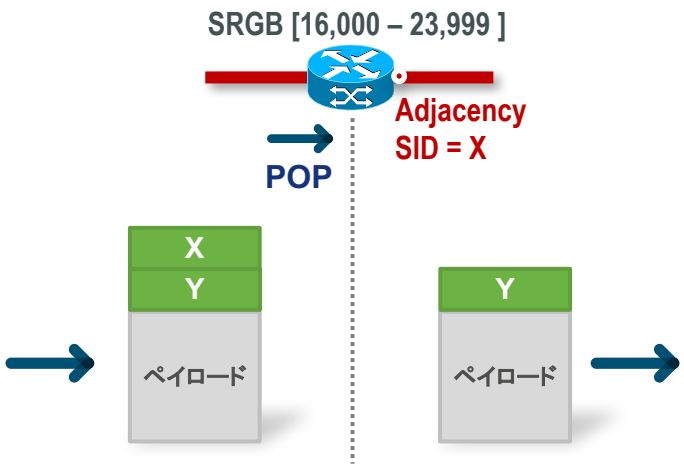
MPLS データプレーンの動作

Prefix SID



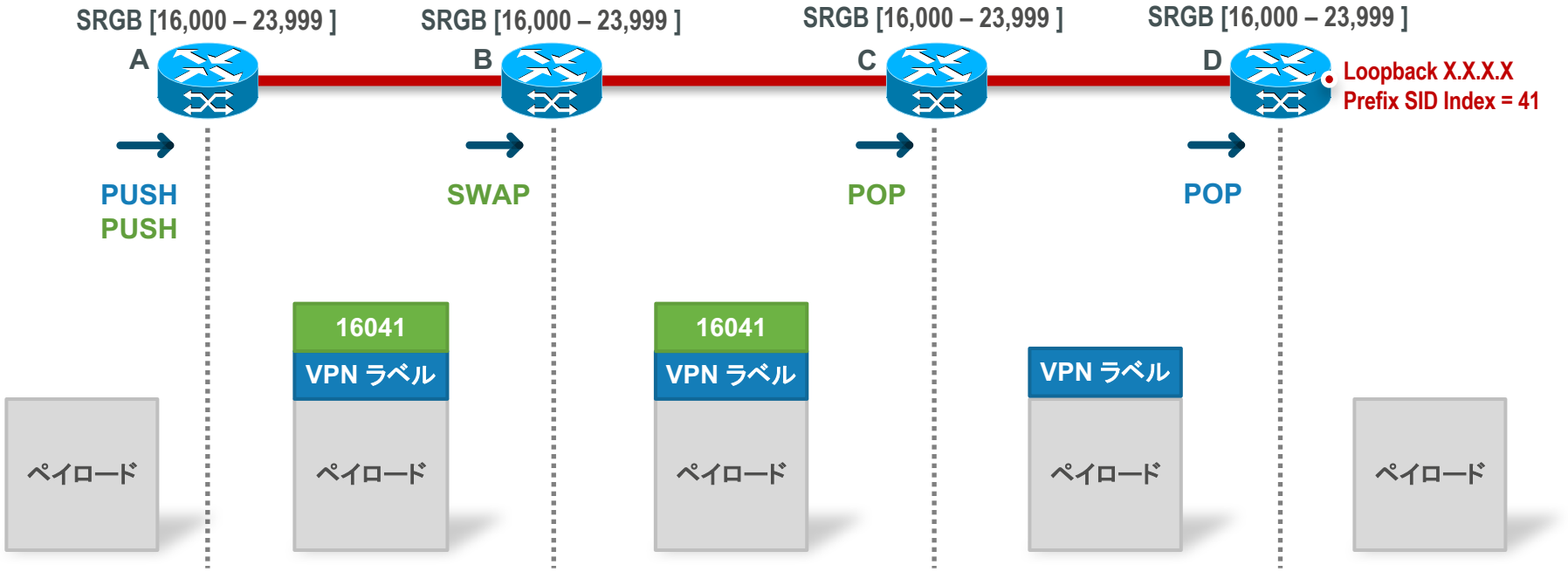
- パケット転送は IGP の最短パス (ECMP) に従う
- インプットラベルに対して SWAP 処理を実行
- 同じ SRGB であれば、同じトップラベル
- Egress LSR から PHP をシグナリングされれば実行

Adjacency SID

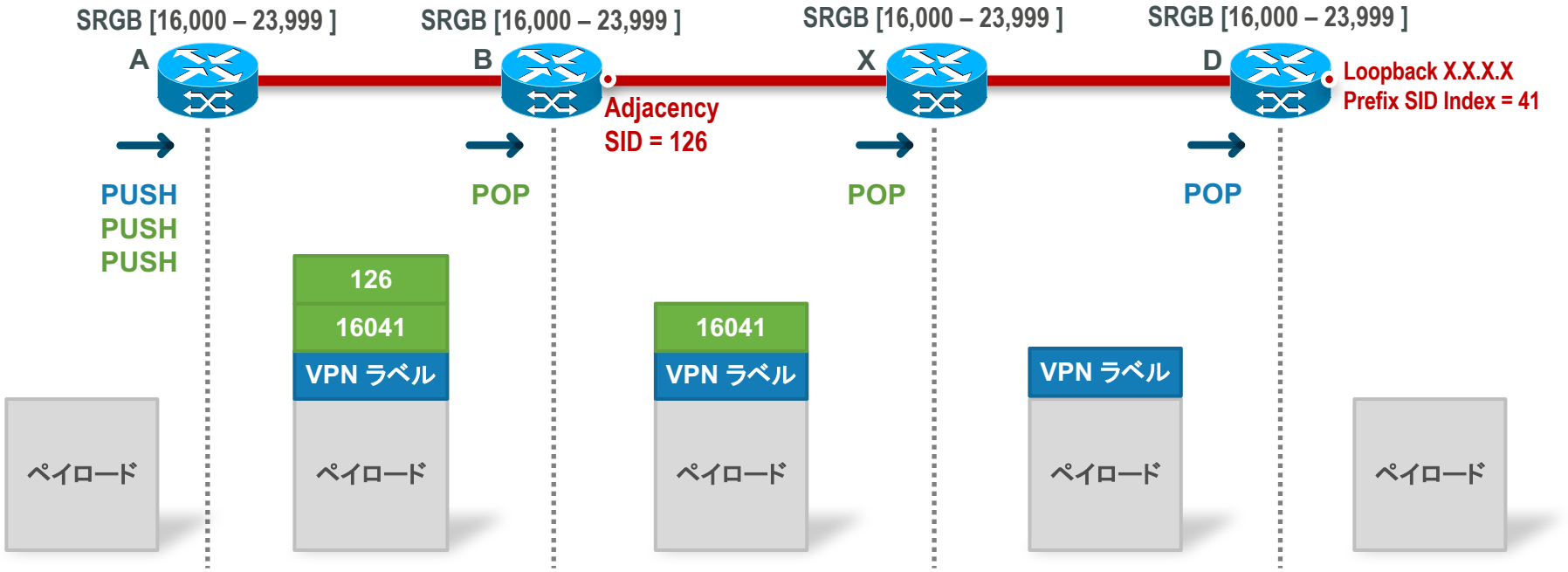


- パケット転送は IGP adjacency に従う
- 入力ラベルに対して POP 処理を実行
- トップラベルは異なる可能性が高い
- Penultimate hop ルータで最後の Adjacency SID を常に POP

MPLS データプレーンの動作
Prefix SID

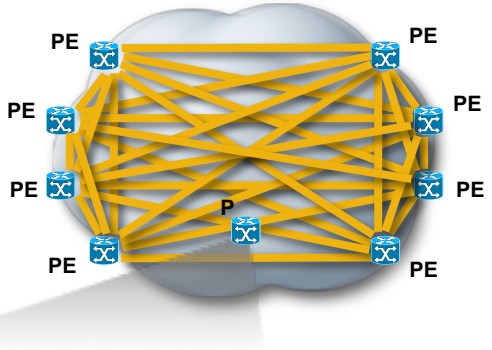


MPLS データプレーンの動作
Adjacency SID



MPLS データプレーンの動作
LFIB

- IGP により LFIB にエントリが作成される
- 他のラベル配布プロトコル (LDP, RSVP, BGP) も引き続き LFIB にエントリを追加可能
- パスの数に関係なく、フォワーディングテーブルのエントリ数は一定 (Nodes + Adjacencies)

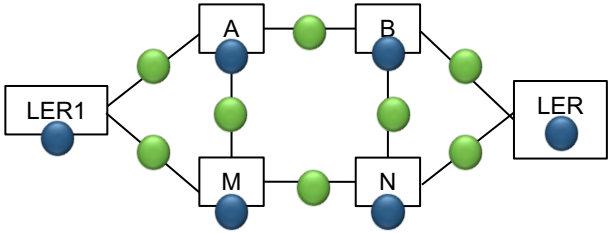


	In Label	Out Label	Out Interface
Network Node Segment Ids	L1	L1	Intf1
	L2	L2	Intf1
	…	…	…
	L8	L8	Intf4
	L9	L9	Intf2
Node Adjacency Segment Ids	L10	Pop	Intf2
	…	…	…
	Ln	Pop	Intf5

Forwarding table remains constant

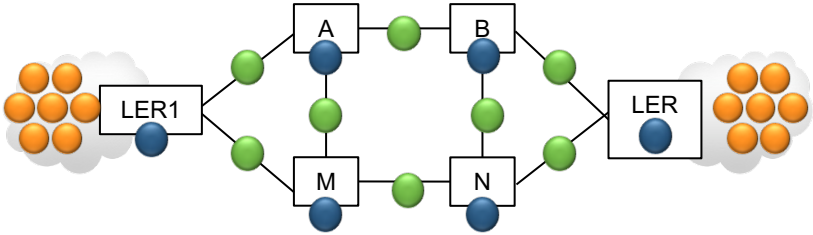
Segment Routing と従来の MPLS の比較

Segment Routing



- 最小限のステート保持（Node/Adj）
- ノードに対してSPF
- コントロールプレーンは IGP のみ

LDP/RSVP



- FEC/LSP ベースのステート保持
- FEC/LSP に対してSPF/CSPF
- Hop-by-Hop のシグナリング

OSPF 設定例 (IOS XR)

```
router ospf DEFAULT
 router-id 172.16.255.1
 segment-routing mpls
 segment-routing forwarding mpls
 area 0
  interface Loopback0
   passive
   prefix-sid index 4
  !
  interface GigabitEthernet0/0/0/0
   network point-to-point
  !
 !
!
```

全 Area で SR-MPLS を有効にする

全インタフェース上で SR-MPLS フォワーディングを有効にする

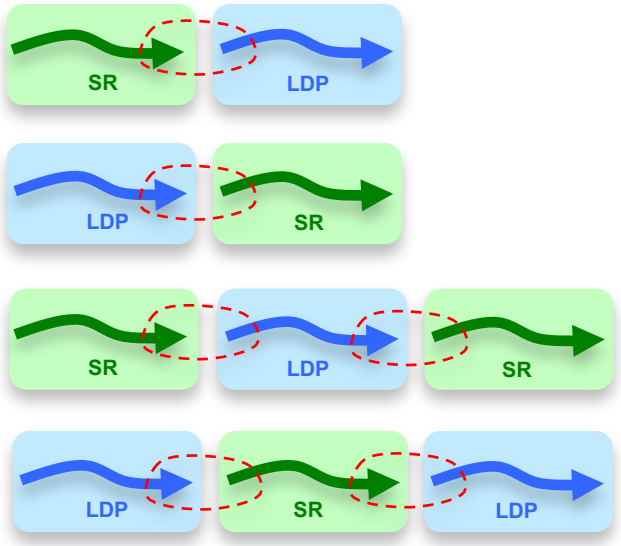
Prefix-SID index を指定する

設定は非常にシンプル!!

従来 MPLS 環境からのマイグレーション

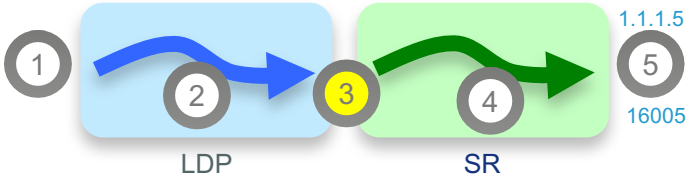
SR では様々なマイグレーションシナリオが考慮されている

1. LDP-to-SR インターワーキング
2. SR-to-LDP インターワーキング
3. LDP/SR 共存

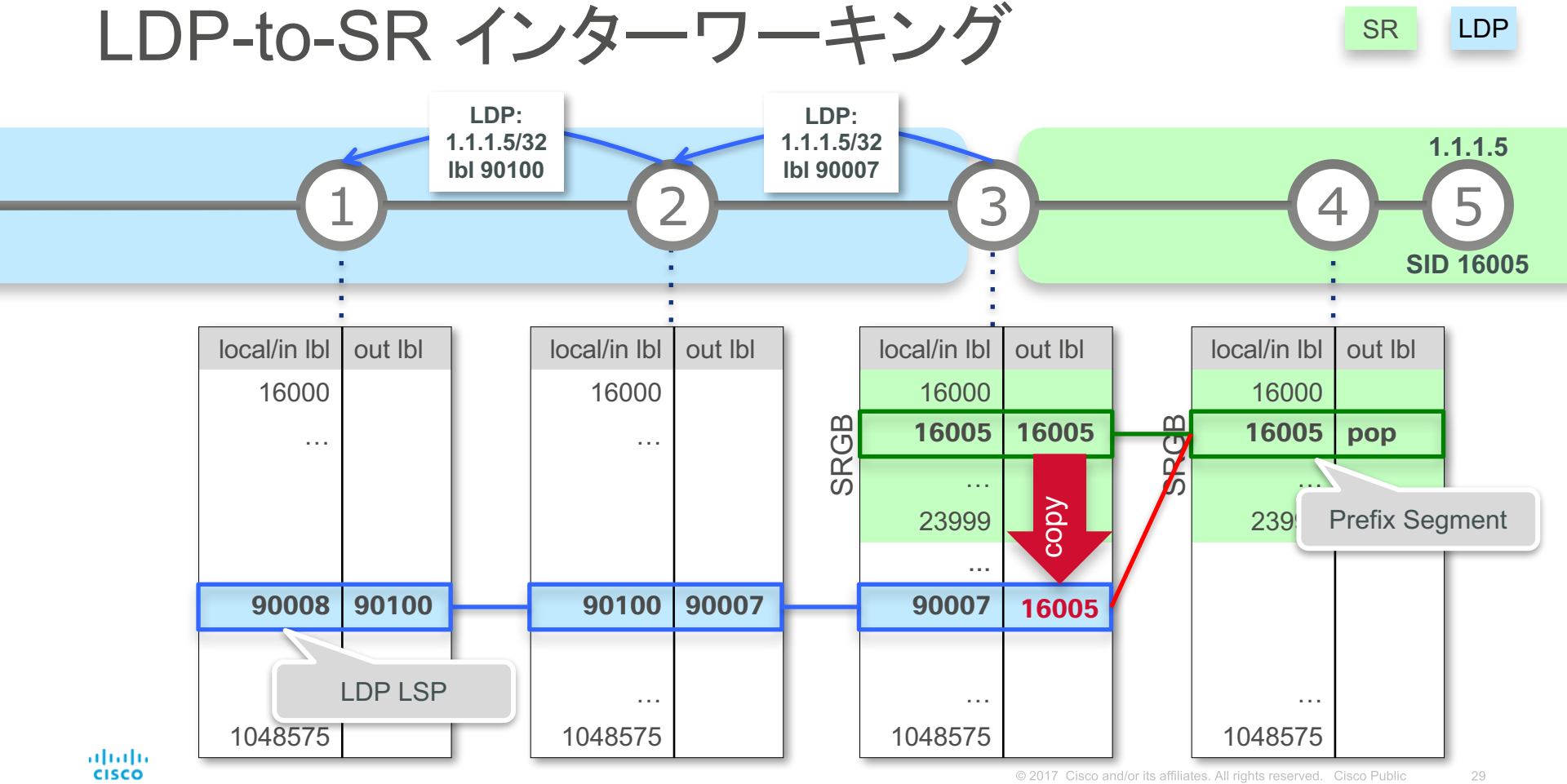


LDP-to-SR インターワーキング

- LDP enable なノードから LDP enable ではないノードへの通信
- LDP/SR の境界になるノードは LDP-to-SR のエントリをインストールする
- 下記の図では Node 3 が下記のような LDP-to-SR エントリをインストールする
  - Incoming ラベル: 1.1.1.5/32 に対して LDP が割り当てた**ローカルラベル**
  - Outgoing ラベル: 1.1.1.5/32 に対して SR が割り当てた **Prefix SID**
  - Outgoing インタフェース: to Node 4
- このエントリは設定しなくても自動でインストールされる



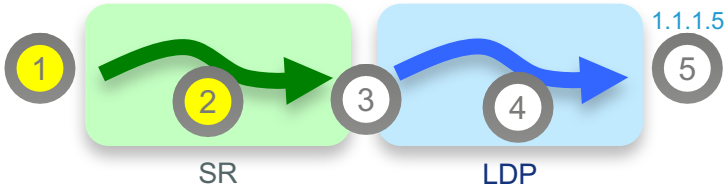
LDP-to-SR インターワーキング



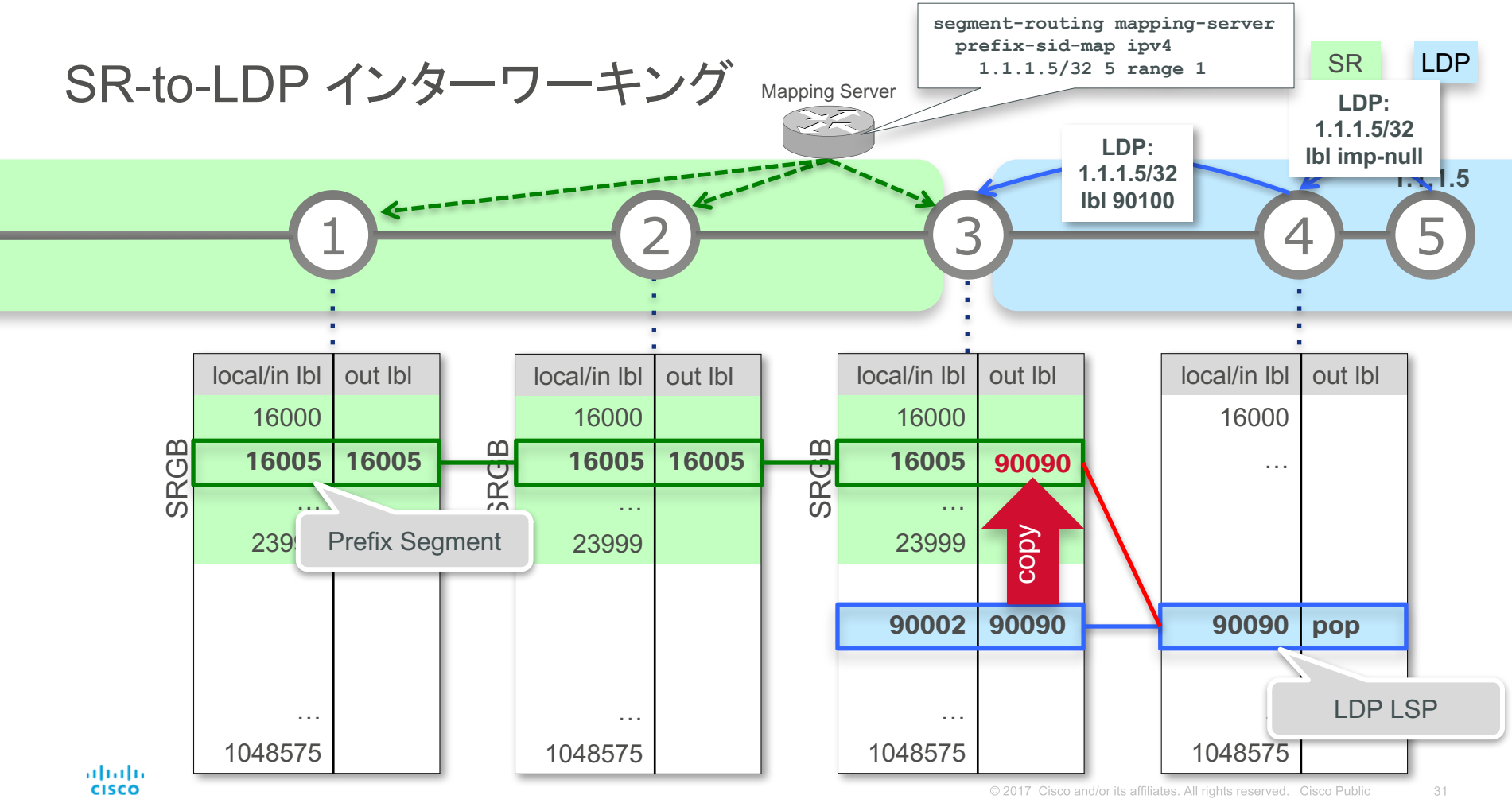
SR-to-LDP インターワーキング

- 宛先が SR ノードではないため、SR ノードは宛先の Prefix-SID を知ることができない
- **Mapping Server (MS)*** 機能を用いて non-SR ノードに代わって Prefix-SID を広報する必要がある
- SR ノードは Mapping Server から広報された Prefix-SID をインストールする
- これによって non-SR な宛先へ SR 転送が接続可能となる

** Mapping Server は IOS XR における機能名。データパス上のノードにおいて実装するか、BGP のルートリフレクタのようにデータパス上に設置しないことも可能*



SR-to-LDP インターワーキング



LDP/SR 共存

- Internet-Draft* によると

  “A local policy on a router MUST allow to prefer the SR-provided IP2MPLS entry.”

- `segment-routing sr-prefer` コマンドで切替可能**
  - デフォルト → LDP が優先される
  - 設定有り → SR が優先される

* https://tools.ietf.org/html/draft-ietf-spring-segment-routing-ldp-interop
** IOS XR の場合

コントロール/データプレーン まとめ

- ✔ コントロールプレーンプロトコルは IGP のみ
- ✔ SID (Segment ID) によってラベルパスを表現
- ✔ Prefix SID (Node SID) と Adjacency SID
- ✔ State in the packet
- ✔ データプレーンは従来の LFIB を利用
- ✔ 既存 MPLS 環境からのシームレスマイグレーション

高速迂回：TI-LFA FRR

TI-LFA FRR とは

LFA (Loop Free Alternate) FRR

あらかじめ自動的にバックアップパスを計算し、障害時に高速迂回を実現する技術

Remote LFA FRR

ループが発生しないノードまで T-LDP でトンネルすることで、バックアップパスを準備する

トポロジによっては、ループフリーなバックアップパスを準備できない

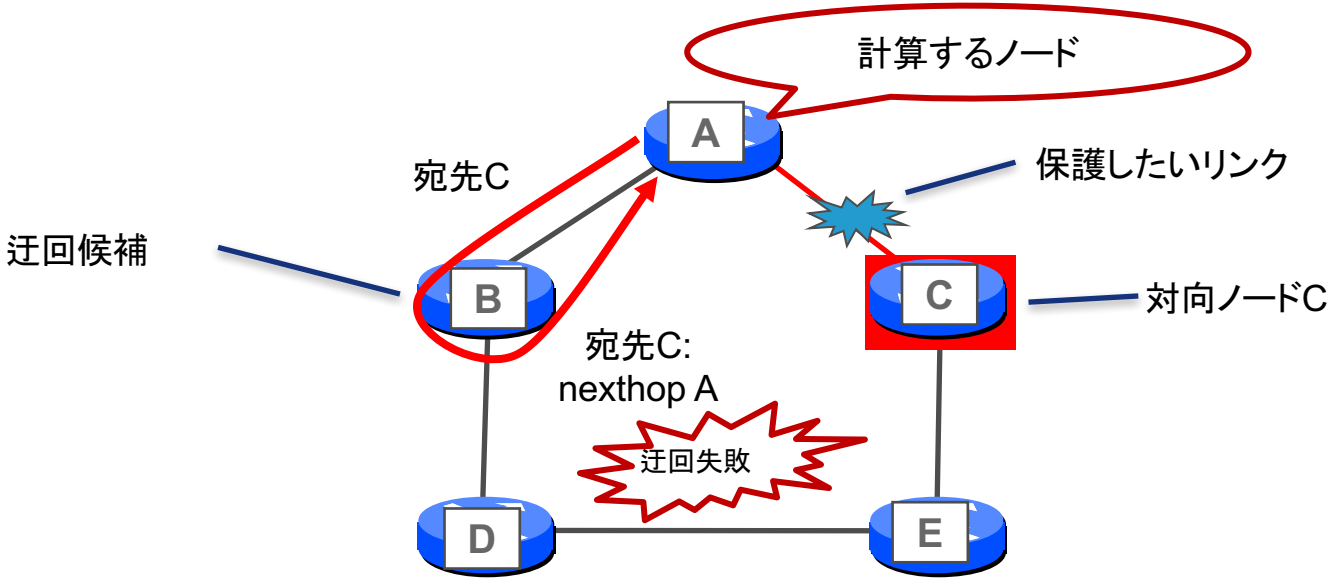
TI-LFA (Topology Independent) FRR

あらゆるトポロジにおいて、バックアップパスを用意することが可能な方式

- **100%-coverage 50-msec link and node protection**
- **輻輳**や**非最適ルーティング**の排除
- **シンプル**で理解しやすいオペレーション
- **自動計算**によって全障害シナリオをカバー
- 迂回のための**別プロトコル (T-LDP) 不要**
- ネットワークにステートを持たせる必要が無い（PLR のみが迂回パスの情報を持つ）

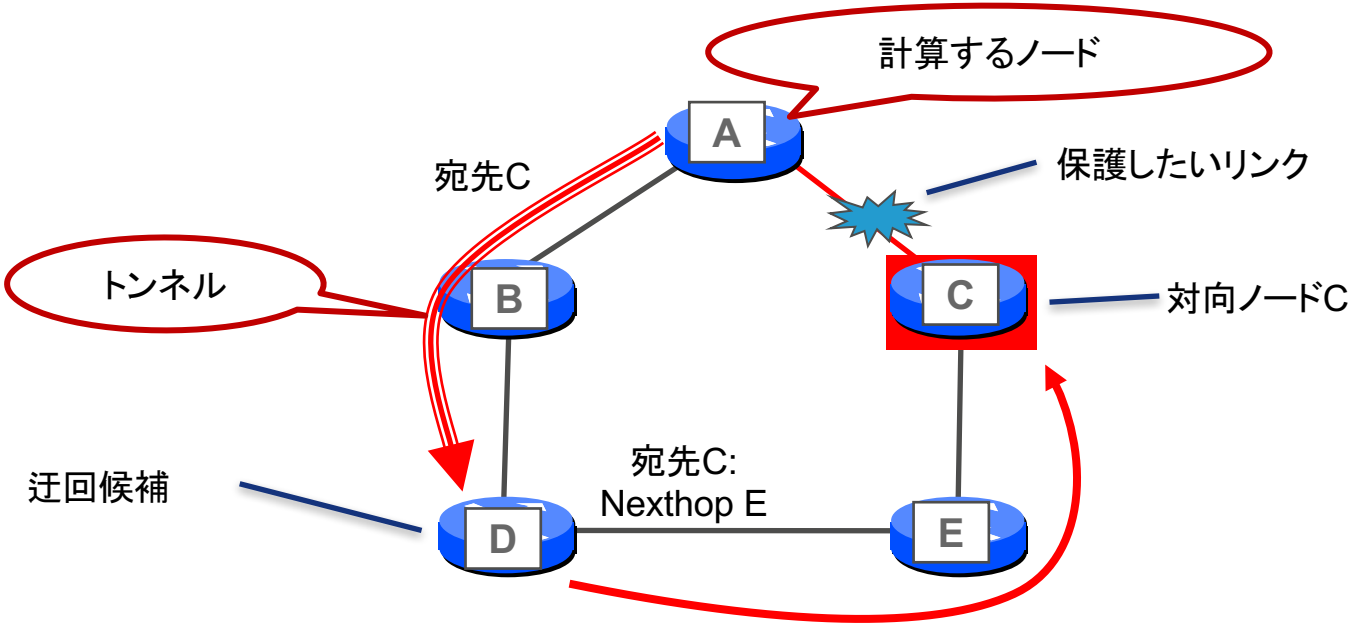
Remote LFA FRR

- Per-Link LFA, Per-Prefix LFA となるネイバーが見当たらないケース
- リングトポロジーでは良くある例

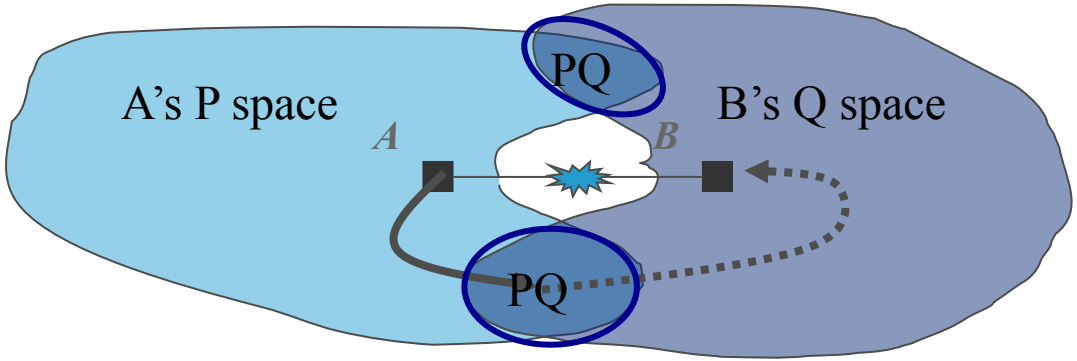


Remote LFA FRR

- LFA となるリモートノードまでトンネルすれば良い
  - どうやってトンネル出口を見つけるのか?
  - どうやってトンネルするのか?



Remote LFA FRR – PQノード

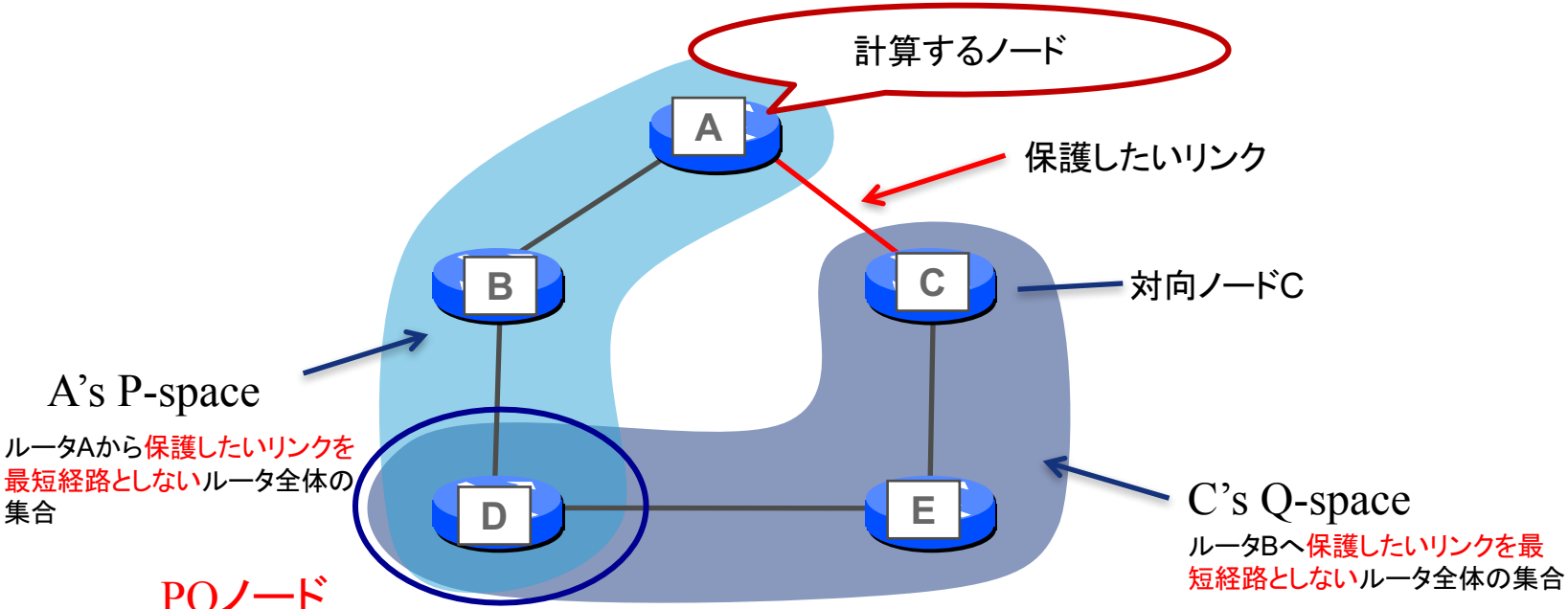


- P-space はルータAからリンクABを最短経路としないルータ全体の集合
  (ECMP含む、詳細は次ページ)
- Q-space はルータBへリンクABを最短経路としないルータ全体の集合
  (ECMP含まない、詳細は次ページ)
- P-space と Q-space の両方に属するルータをPQノードと呼ぶ
  - ルータAがパケットをPQノードへトンネルすれば、
    パケットはリンクABを経由せずにBへ到達する

ひどく非対称なメトリックがなければカバー率は100%に近い

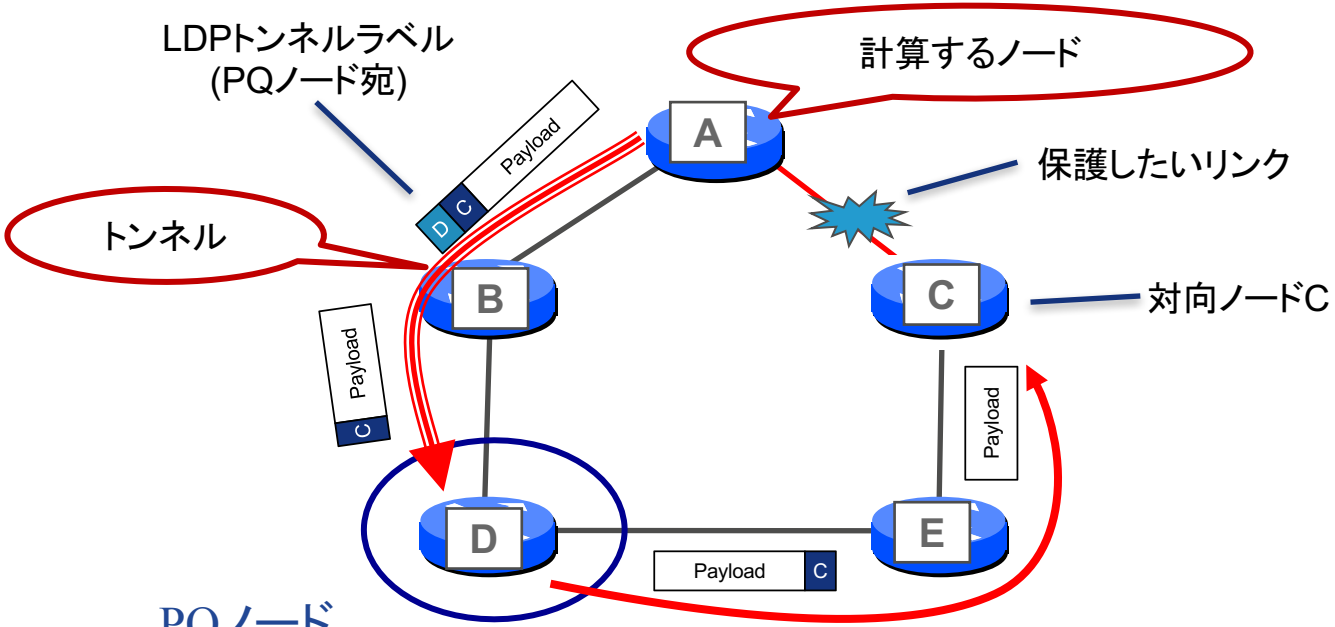
Remote LFA FRR 計算: PQノード

- P-space と Q-space の両方に属するルータを PQノードと呼ぶ
- Remote LFA用トンネルの出口となりうる(1台とは限らない)



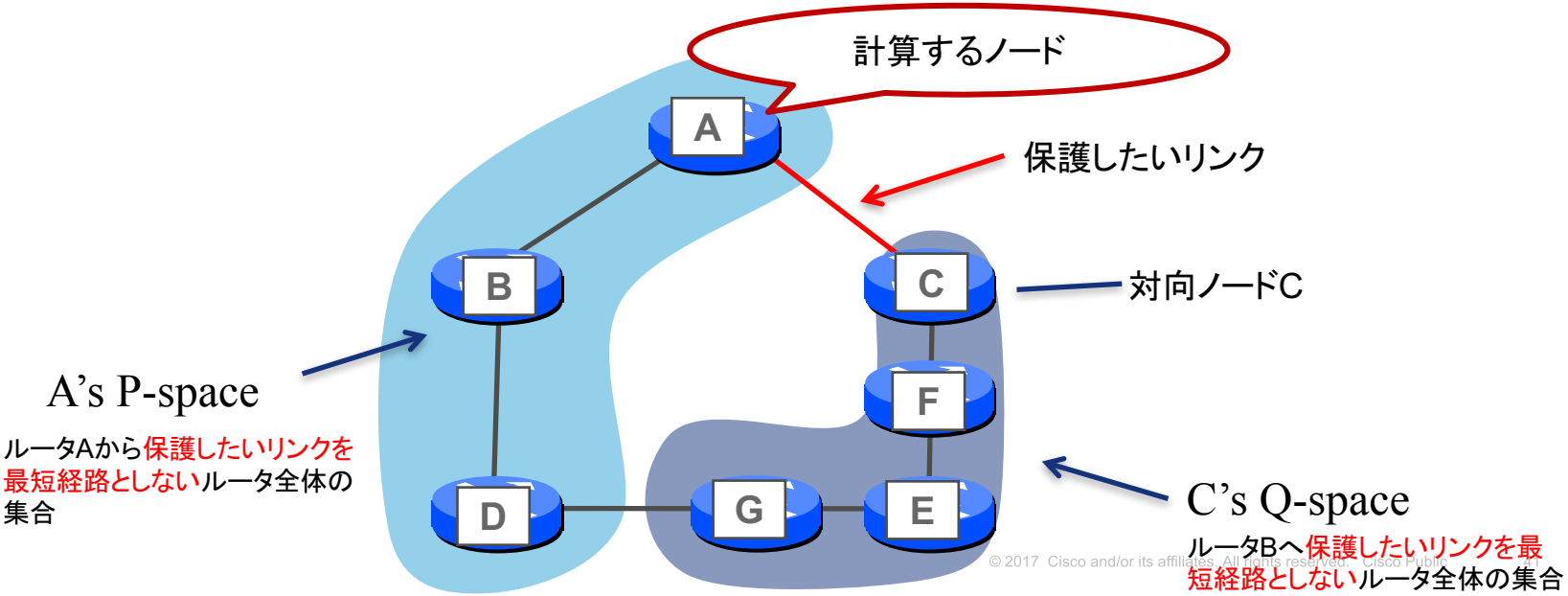
Remote LFA FRR: パケット転送

- PQノードへパケットをMPLS LSPでトンネル
- LDPを使用



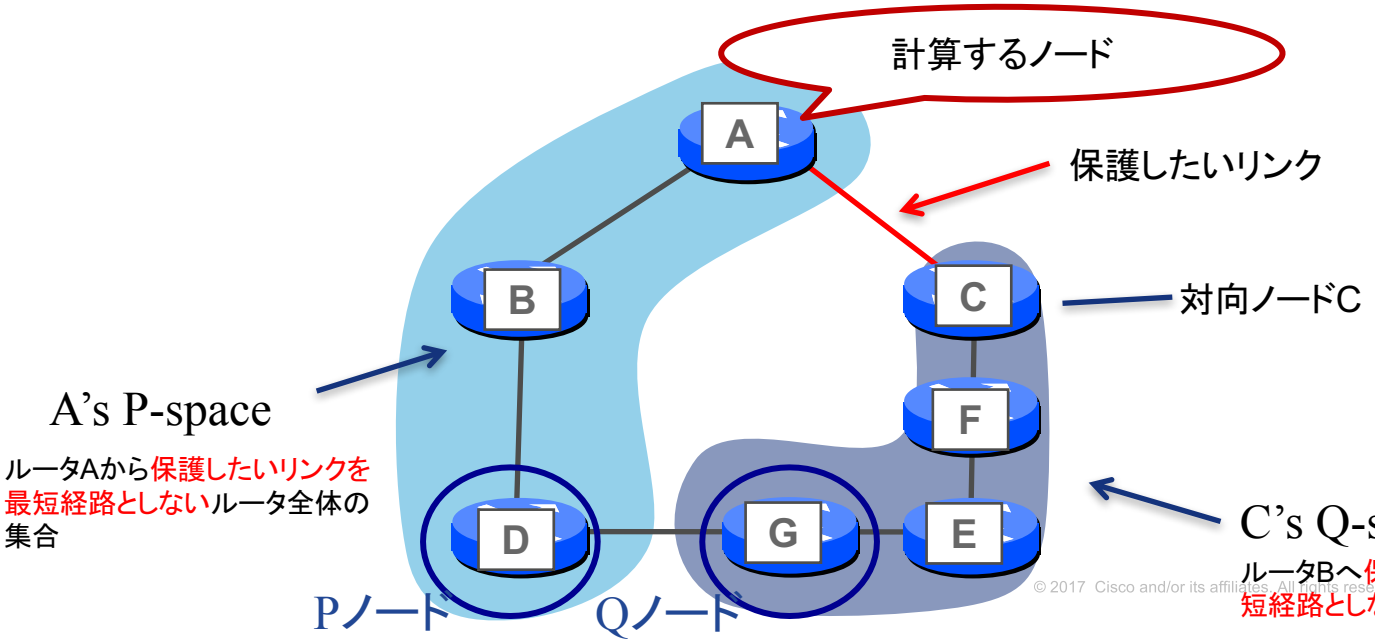
Remote LFA FRR 計算ができないケース

- P-space と Q-spaceの両方に属するPQノードが存在しない場合Remote LFAは動かない
- これに対応する手法としてSegment Routing Topology Independent LFA(TI LFA)を紹介させて頂きます



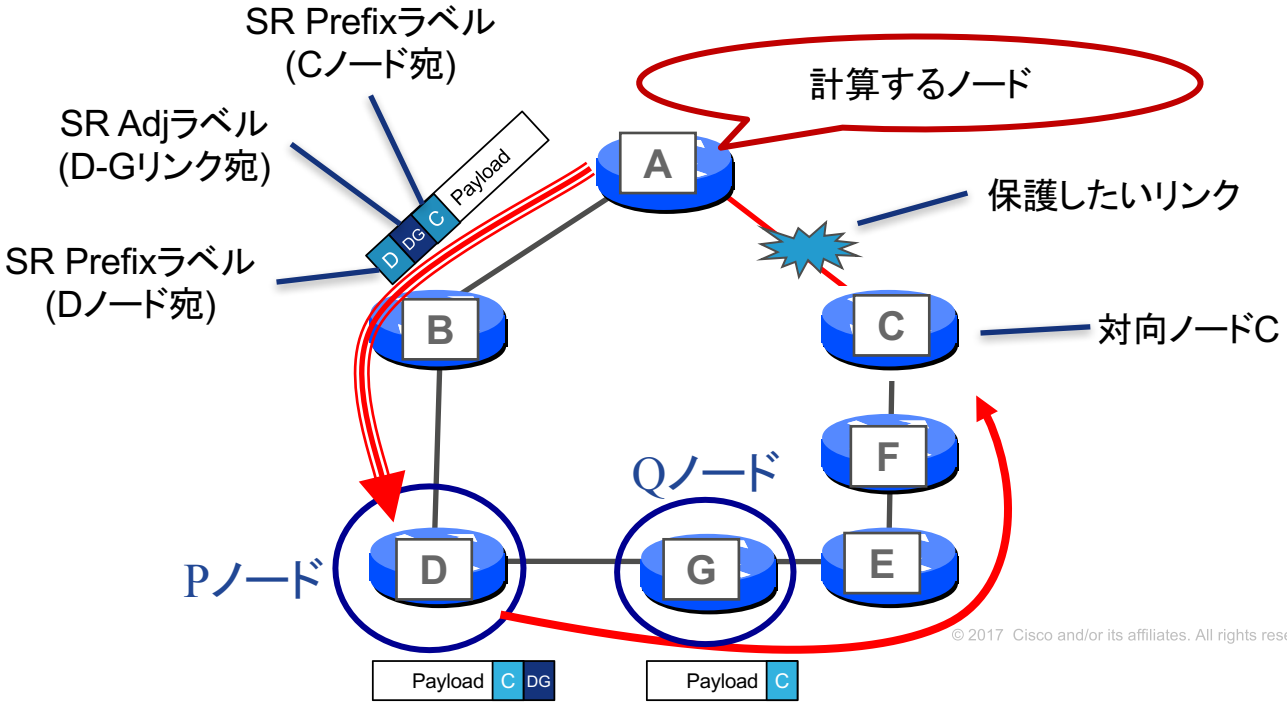
TI-LFA FRR

- P-space と Q-spaceの計算を行いPノードと最も近いQノードを計算する
- 下記の例ではDがPノード、GがQノードとなる



TI-LFA FRR: パケット転送

- AはPノードのPrefix-SIDとD-G間のAdj-SIDをPushしてパケットをBackup Pathに転送
- A-D-Gを経由するので、ループすることはない



TI-LFA FRR 設定例

```
router ospf 1
 router-id 1.1.1.1
 segment-routing mpls
 segment-routing forwarding mpls
 fast-reroute per-prefix
 fast-reroute per-prefix ti-lfa enable
 area 0
  interface Loopback0
   prefix-sid index 1
  !
  interface GigabitEthernet0/0/0/3
  !
  interface GigabitEthernet0/0/0/7
  !
 !
!
```

TI-LFA FRR を有効にする

TI-LFA FRR: show ospf route

```
RP/0/0/CPU0:Router#show ospf 1 routes 5.5.5.5/32 backup-path

Topology Table for ospf 1 with ID 1.1.1.1

Codes: O - Intra area, O IA - Inter area
       O E1 - External type 1, O E2 - External type 2
       O N1 - NSSA external type 1, O N2 - NSSA external type 2

O    5.5.5.5/32, metric 2
       10.1.0.2, from 5.5.5.5, via GigabitEthernet0/0/0/7, path-id 1
          Backup path: TI-LFA, P node: 4.4.4.4, Label: 16004, Q node: 2.2.2.2, Label: 24000
              10.0.3.2, from 5.5.5.5, via GigabitEthernet0/0/0/3, protected bitmap 0000000000000001
              Attribues: Metric: 105, SRLG Disjoint
```

Double-segment LFA
(P and Q)

TI-LFA FRR: show route

```
RP/0/0/CPU0:Router#show route 5.5.5.5/32 detail

Routing entry for 5.5.5.5/32
  Known via "ospf 1", distance 110, metric 2, type intra area
  Installed Nov 24 07:22:18.605 for 00:47:25
  Routing Descriptor Blocks
    10.0.3.2, from 5.5.5.5, via GigabitEthernet0/0/0/3, Backup (remote)
      Remote LFA is 4.4.4.4, 2.2.2.2
      Route metric is 0
      Labels: 0x3e84 0x5dc0 0x3e85 (16004 24000 16005)
      Tunnel ID: None
      Extended communities count: 0
      Path id:66              Path ref count:1
      NHID:0x1(Ref:6)
      OSPF area: 0
    10.1.0.2, from 5.5.5.5, via GigabitEthernet0/0/0/7, Protected
      Route metric is 2
      Label: 0x3 (3)
      Tunnel ID: None
      Extended communities count: 0
      Path id:1       Path ref count:0
      NHID:0x2(Ref:7)
      Backup path id:66
      OSPF area: 0
```

Prefix-SID to P

Prefix-SID to destination

Adjacency-SID from P to Q

Double-segment LFA IP backup path

Primary path

TI-LFA FRR まとめ

- あらゆるトポロジーで 50msec FRR
- IGP で自動化 (No RSVP)
- コンバージェンス後のパス最適化
- Midpoint でのバックアップステート無し
- 詳細なオペレータリポートが公開されています

  S. Litkowski, B. Decraene, Orange

  https://www.slideshare.net/BrunoDecraene/mpls-wcc-2014-segment-routing-tilfa-fast-reroute

トラフィックエンジニアリング : SR-TE

SR-TE の必要性

Segment Routing

- シンプルでスケーラブルなルーティング
- 自動的に構成される高速迂回機能

Simplify/Automate

SR-TE

- 柔軟なパス選択
- プログラマビリティ
- サービスチェイニング

Programmable

SR-TE と RSVP-TEの比較

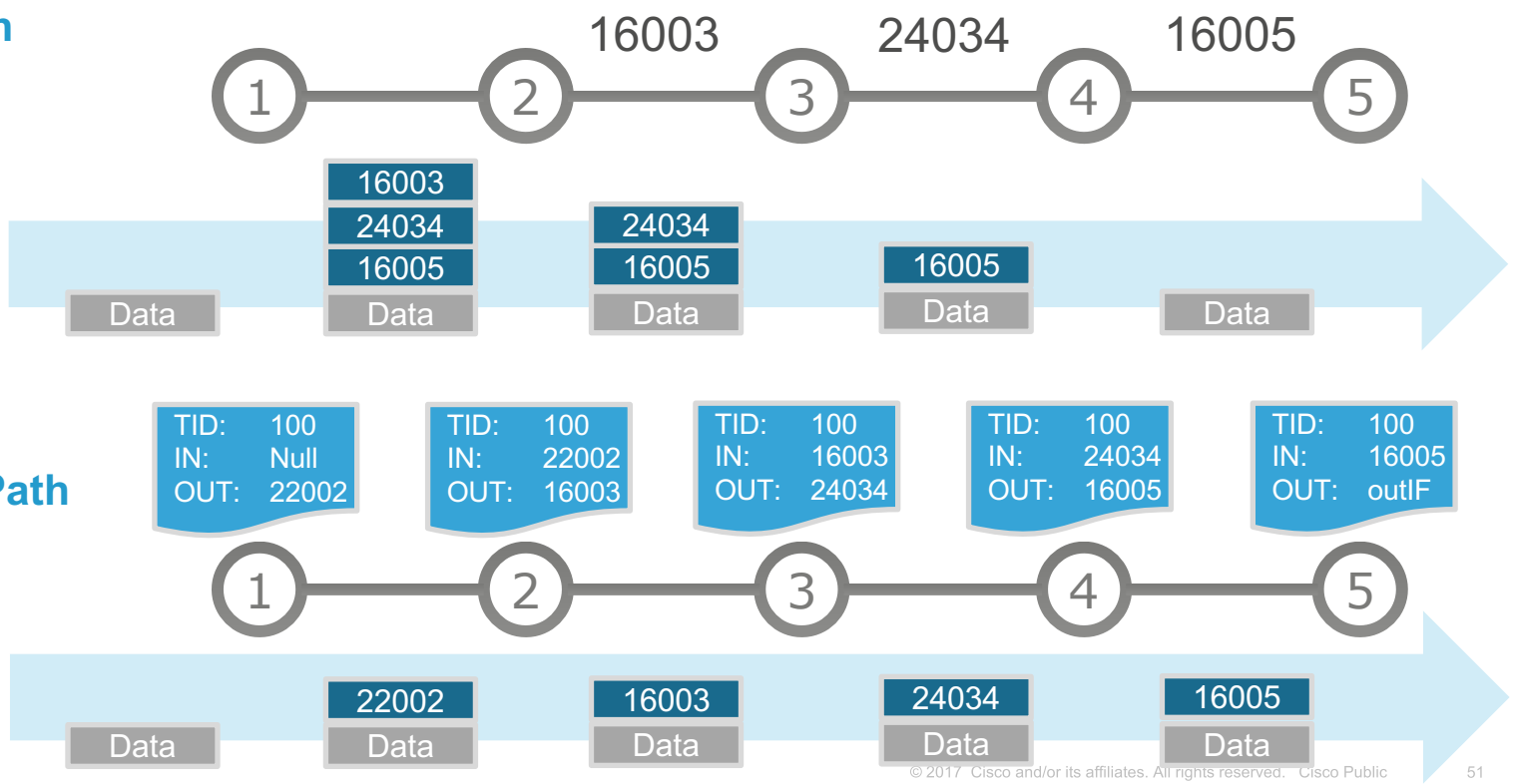
	Segment Routing	従来の MPLS
コントロールプレーン プロトコル	IGP *	IGP + LDP (+RSVP)
LSP ステート	Headend のみがステートを保持 **	全ノード (Headend/Midpoint/Tailend) がステートを保持
ECMP-aware TE	ECMP 有り or 無しで LSP を設定可能 明示的に ECMP-aware のノードセグメントのリストを構築可能	パス毎に明示的にトンネルを列挙する必要あり
FRR	TI-LFA FRR 自動的な リンク/ノード/SRLG プロテクション 明示的に設定されたパスプロテクション	RSVP-TE FRR 明示的に設定されたリンク/ノード/パスプロテクション

- SR-TE では ECMP-aware の最短パスフォワーディング環境において、柔軟で効率的なトラヒックエンジニアリングが可能
- ヘッドエンドに Segment リストと宛先等のフローを結びつけるポリシーを指示するのみで容易にルーティング制御ができる

** 各プロトコル (OSPF or IS-IS) に SR 拡張が必要*

*** Headend に Segment リストと宛先等のフローとを結びつけるポリシーを指示するのみで容易にルーティング制御可能*

SR-TE と RSVP-TE の比較

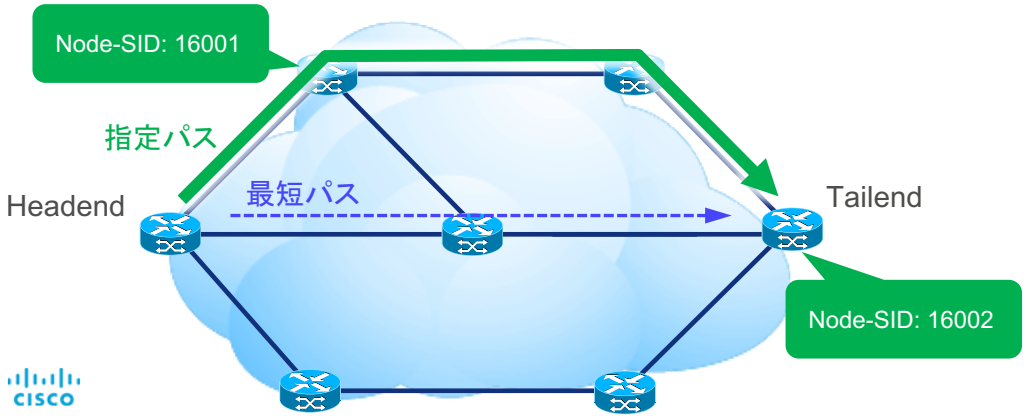


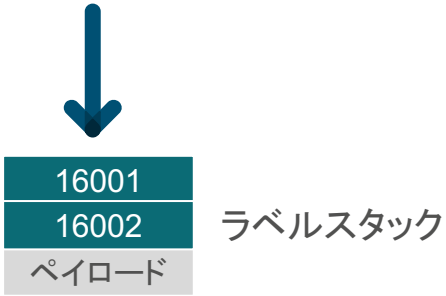
Explicit Path-Option
SID を指定した場合の動作例

ヘッドエンドは SID を指定

```
explicit-path name path-3
  index 1 next-label 16001
  index 2 next-label 16002

interface tunnel-te 100
  ipv4 unnumbered loopback0
  destination 5.5.5.5
  path-option 1 explicit name path-3 segment-routing
```

Node-SID (Prefix-SID) または Adjacency-SID
の組合せで SID リストを指定可能



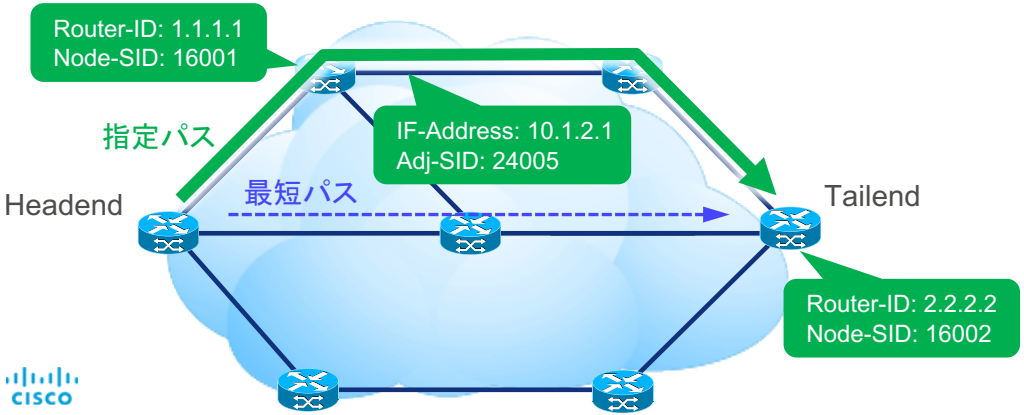


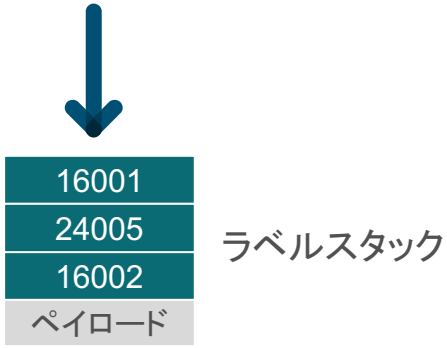
Explicit Path-Option
IP アドレスを指定した場合の動作例

ヘッドエンドは指定パスの検証を行ってラベルスタックを算出

```
explicit-path name path-1
  index 1 next-address ipv4 unicast 1.1.1.1
  index 2 next-address ipv4 unicast 10.1.2.1
  index 3 next-address ipv4 unicast 2.2.2.2

interface tunnel-te 100
  path-option 1 explicit name path-1 segment-routing
```

ヘッドエンドは IP ホップを SID リストに解決
1.1.1.1 → 16001
10.1.2.1 → 24005
2.2.2.2 → 16002





Explicit Path-Option
Inter-Area の動作例

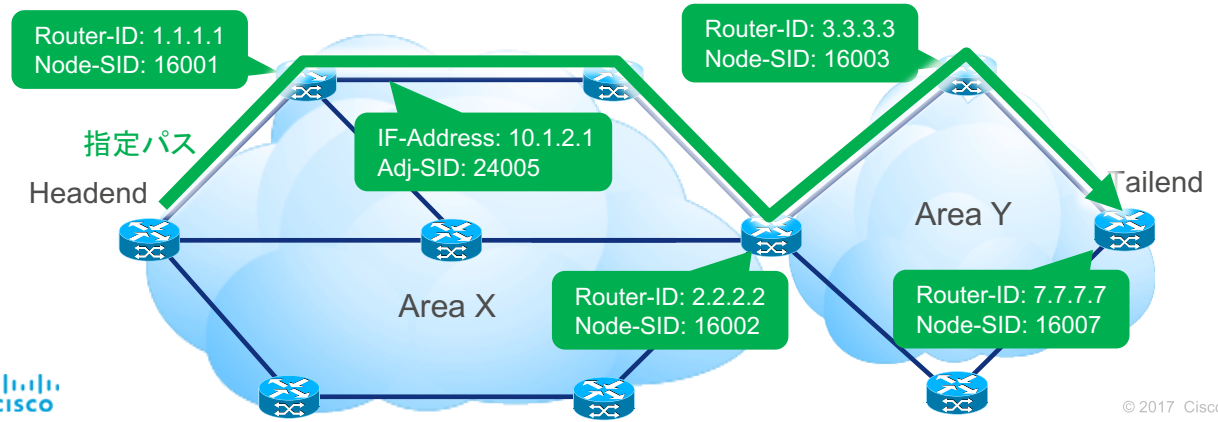
エリア間で Prefix-SID 交換が無い場合

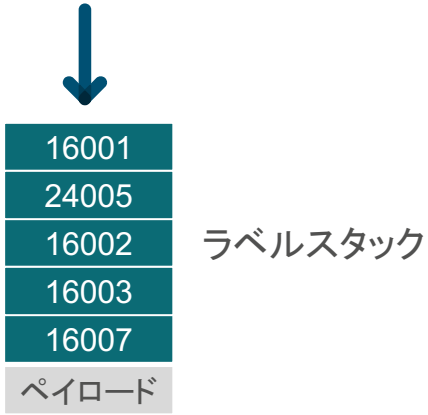
```
explicit-path name path-4
 index 1 next-address ipv4 unicast 1.1.1.1
 index 2 next-address ipv4 unicast 10.1.2.1
 index 3 next-address ipv4 unicast 2.2.2.2
 index 4 next-label 16003
 index 5 next-label 16007

interface tunnel-te 100
  path-option 1 explicit name path-4 segment-routing
```

ヘッドエンドエリア

非ヘッドエンドエリア

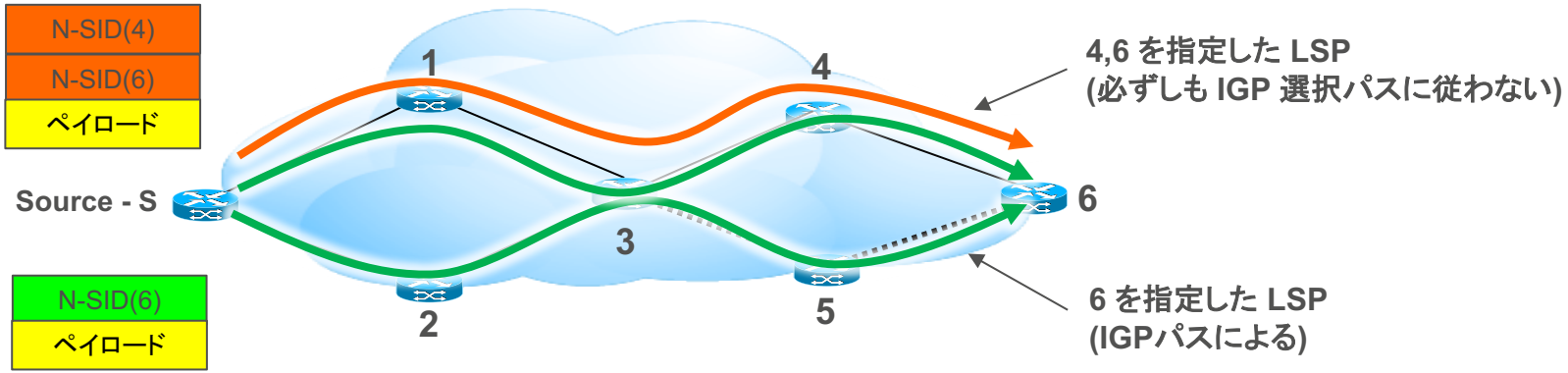
ヘッドエンドがリモートエリアの SID 情報を
持たない場合、ホップを SID で指定する





SR-TE ECMP-aware ロードバランシング

- SR-TE は ECMP ロードバランシングにネイティブで対応
- SR-TE LSP が ECMP を持つ 1 つ以上の Prefix-SID を通る場合、ヘッドエンドまたは LSP が通る Midpoint から各 Prefix-SID の ECMP 上でロードバランス



SR-TE 設定例

```
router ospf 1
 router-id 100.0.0.1
 segment-routing mpls
 segment-routing forwarding mpls
 fast-reroute per-prefix
 fast-reroute per-prefix ti-lfa enable
 !
 area 0
  mpls traffic-eng
  interface Loopback0
   passive enable
   prefix-sid index 2
  !
!
  interface GigabitEthernet0/0/0/0
   network point-to-point
  !
  interface GigabitEthernet0/0/0/1
   network point-to-point
  !
 area 1
  interface GigabitEthernet0/0/0/2
   network point-to-point
```

全 Area で SR-MPLS を有効にする

全インタフェース上で SR-MPLS フォワーディングを有効にする

TI-LFAを有効にする

Area 0 で MPLS-TE を有効にする

Prefix-SID index を指定する

show mpls traffic-eng segment-routing

```
RP/0/RSP0/CPU0:Router#show mpls traffic-eng segment-routing 8.8.8.8
IGP[0]:: OSPF 1 area 0
Nodes:
 IGP Id: 8.8.8.8, MPLS TE Id: 8.8.8.8
   SRGB Info: Start 16000, Size 8000
   Link[0]: Intf Addr: 78.0.0.8, Nbr Intf Addr: 78.0.0.7, Type: Point-to-Point
       Nbr IGP Id: 7.7.7.7, Nbr MPLS TE Id: 7.7.7.7
       Label: 24009, flags: V, L  Label: 24008, flags: B, V, L
   Link[1]: Intf Addr: 86.1.0.8, Nbr Intf Addr: 86.1.0.6, Type: Point-to-Point
       Nbr IGP Id: 6.6.6.6, Nbr MPLS TE Id: 6.6.6.6
       Label: 24001, flags: V, L  Label: 24000, flags: B, V, L
   Link[2]: Intf Addr: 86.2.0.8, Nbr Intf Addr: 86.2.0.6, Type: Point-to-Point
       Nbr IGP Id: 6.6.6.6, Nbr MPLS TE Id: 6.6.6.6
       Label: 24003, flags: V, L  Label: 24002, flags: B, V, L
   Link[3]: Intf Addr: 118.1.2.8, Nbr Intf Addr: 118.1.2.11, Type: Point-to-Point
       Nbr IGP Id: 11.11.11.11, Nbr MPLS TE Id: 11.11.11.11
       Label: 24005, flags: V, L  Label: 24004, flags: B, V, L
   Link[4]: Intf Addr: 118.1.3.8, Nbr Intf Addr: 118.1.3.11, Type: Point-to-Point
       Nbr IGP Id: 11.11.11.11, Nbr MPLS TE Id: 11.11.11.11
       Label: 24007, flags: V, L  Label: 24006, flags: B, V, L

Prefixes:
 8.8.8.8/32, SID index: 108, flags: N
   Adv. router(s)
   --------------
   8.8.8.8

   Paths
   -----
   Path Id Role          Outgoing Interface Next Hop        Outgoing Label
   ------- ------------- ------------------ --------------- --------------
   1       Primary       Gi0/0/0/7          78.0.0.8        i-nul
           Backup (65)   Gi0/0/0/6          76.1.0.6        16108
```

show mpls traffic-eng tunnels

```
RP/0/RSP0/CPU0:Router#show mpls traffic-eng tunnels 1
Name: tunnel-te1  Destination: 12.12.12.12  Ifhandle:0xa20
  Signalled-Name: RTR_t1
  Status:
    Admin:    up Oper:   up   Path:  valid   Signalling: connected

    path option 10, (Segment-Routing) type explicit path1 (Basis for Setup)
      Protected-by PO index: none
    G-PID: 0x0800 (derived from egress interface properties)
    Bandwidth Requested: 0 kbps  CT0
    Creation Time: Wed Sep 30 17:46:35 2015 (01:13:04 ago)
  Config Parameters:
    Bandwidth:        0 kbps (CT0) Priority:  7  7 Affinity: 0x0/0xffff
    Metric Type: TE (default)
    Path Selection:
      Tiebreaker: Min-fill (default)
      Protection: any (default)
  ...
  History:
    Tunnel has been up for: 01:13:04 (since Wed Sep 30 17:46:35 UTC 2015)
  Segment-Routing Path Info (OSPF 1 area 0)
    Segment0[Node]: 8.8.8.8, Label: 16108
    Segment1[Node]: 11.11.11.11, Label: 16111
    Segment2[Node]: 12.12.12.12, Label: 16112
Displayed 1 (of 1) heads, 0 (of 0) midpoints, 0 (of 0) tails
Displayed 1 up, 0 down, 0 recovering, 0 recovered heads
```

show ospf routes backup-path & show mpls forwarding

```
RP/0/RSP0/CPU0:Router#show ospf routes 8.8.8.8/32 backup-path

Topology Table for ospf 1 with ID 7.7.7.7

Codes: O - Intra area, O IA - Inter area
       O E1 - External type 1, O E2 - External type 2
       O N1 - NSSA external type 1, O N2 - NSSA external type 2

O    8.8.8.8/32, metric 2
       78.0.0.8, from 8.8.8.8, via GigabitEthernet0/0/0/7, path-id 1
           Backup path:
              76.1.0.6, from 8.8.8.8, via GigabitEthernet0/0/0/6, protected bitmap 0000000000000001
              Attribues: Metric: 3, SRLG Disjoint

RP/0/RSP0/CPU0:Router#show mpls forwarding tunnels 1
Tunnel        Outgoing    Outgoing     Next Hop        Bytes
Name          Label       Interface                    Switched
------------- ----------- ------------ --------------- ------------
tt1      (SR) 16111       Gi0/0/0/7    78.0.0.8        0
              16108       Gi0/0/0/6    76.1.0.6        0            (!)
```

SR-TE まとめ

- RSVP-TE のような専用のラベル配信プロトコルは不要
- ECMP-aware のパス制御
- LSP ステートはヘッドエンドのみ
- インタードメイン環境における End-to-End パス指定
- ヘッドエンドのみで制御可能のため、外部コントローラからのプログラミング、BGP等のルーティングポリシーとの連携が容易に実現可能

SR-MPLS まとめ

SR-MPLS まとめ

- ネットワークの**シンプル化**：制御プレーン簡素化、ステート削減
- ネットワークの**自動化**：プログラマビリティ
- スケーラビリティと柔軟性
- **100% のトポロジカバー率**で最適パスを考慮した **FRR**
- **シームレスマイグレーション**：既存ラベルプロトコル (LDP, RSVP)

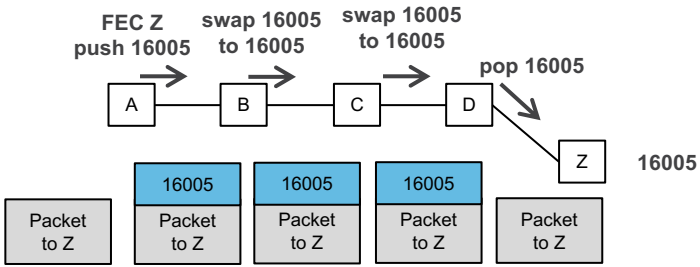
SRv6 Tutorial
SR Usecase

シスコシステムズ合同会社

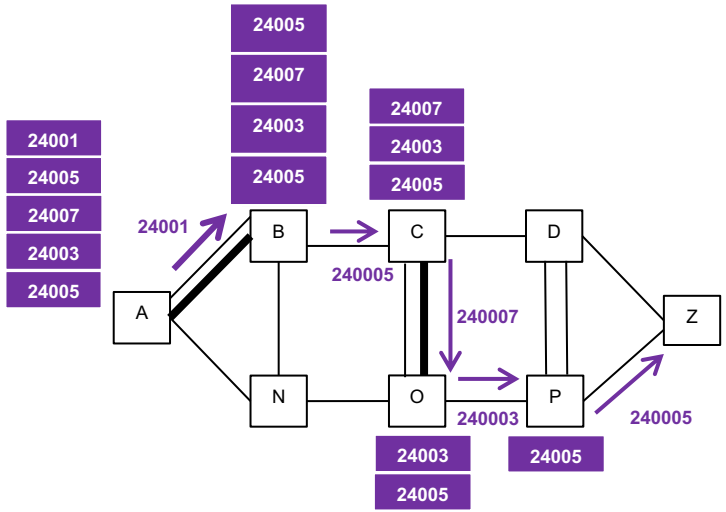
鎌田 徹平
tkamata@cisco.com

Segment Routing MPLS (SR-MPLS) Recap

- ネットワークをSegmentで表現する新しい技術
- SegmentにはNodeとAdjacency2つの要素がある
- LDP/RSVPを使わず、直接IGPによりこれらのIDをアドバタイズする
- ネットワークからLDP/RSVPのステートを排除する事が出来る
- 現在はIETF SPRING(*Source Packet Routing in Networking)* WG にてアーキテクチャーを定義



Node SID



Adjacency SID

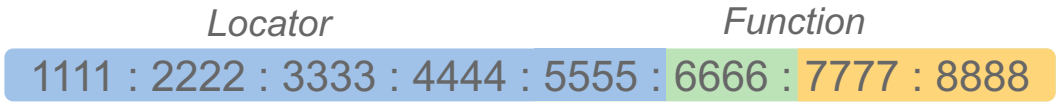
Segment Routing Data-Planes

- セグメントルーティングデータプレーン
  - **SR-MPLS**: segment routing applied to MPLS data-plane
  - **SR-IPv6**: segment routing applied to IPv6

- SR-IPv6 は MPLSネットワークでない環境上や、現在MPLSを適用していないネットワークエリアへ適用できる (例: データセンター)

- SRの後方互換性
  - SR nodes fully interoperate with non-SR nodes
  - No need to have a full network upgrade

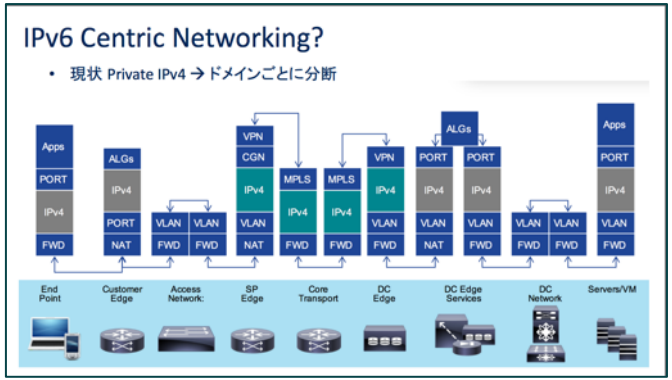
Segment format



- SRv6 SID は128ビットのIPv6アドレス表記
  - **Locator**: セグメントをペアレントノードへ**Route**するためのビット
  - **Function**: ペアレントノードにおいて取られる**Action**を示すビット
    > **Argument** [optional]: 最後のビットはFunctionで参照される引数
- ビット長は可変
  - SIDのフォーマットはペアレントノードがローカルに規定
- SIDはペアレントノードにおいて明示的に有効にしないといけない
  - ローカルアドレスではデフォルトでSIDとして**有効ではない**
  - SIDはインタフェースに関連付けられている必要はない

ペアレントノード
そのSIDの保有ノード、オリジネーター。

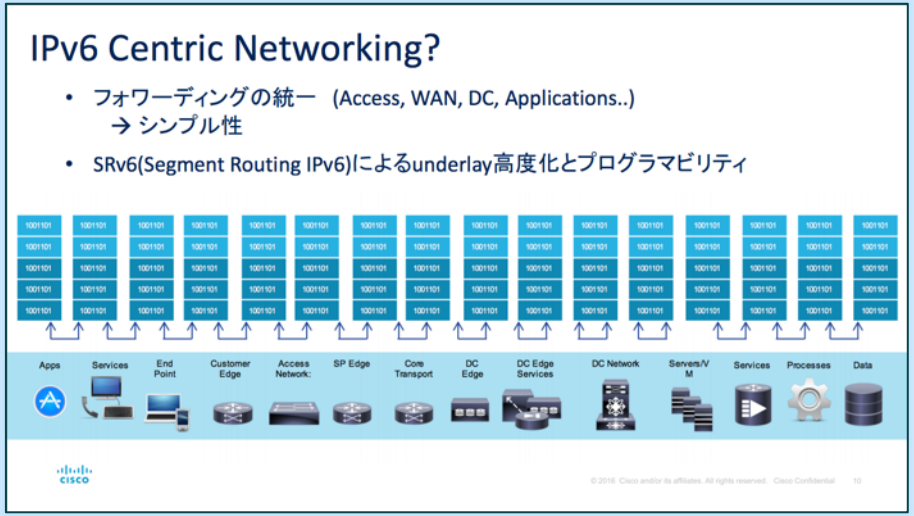
SRv6の何がいいのか？
（ラベルとの違い）



1) **IPv6によりドメインごとに分断されないネットワークが実現できる。SIDが128bitもあるので、色んな機能が定義できる可能性がある。**
   - 識別子のためにvlan idなどを使う必要はもう無い
   - MobilityやContent Networkingなどにも応用できる
   - End-to-endでの（Application, DC, Core, Access, CPE, UE..）、共通転送メカニズムになりうる
   - VPNやMobilityなどのためにTunnel必要ない
   - Label Shim-Layerを排除できる
   - ネットワーク内のステートを最小化できる

2) **SIDをRouting情報として広報出来る。SRv6に対応していないルーターでも転送できる。**
   - IPv6が届けば、どこからでもOverlay、Chainingなどが出来る。
   - Strategic NodeだけがSRv6に対応していればいい。

3) **Linux、VPPで実装されている。**



SRv6 Draft 3点セット

Index

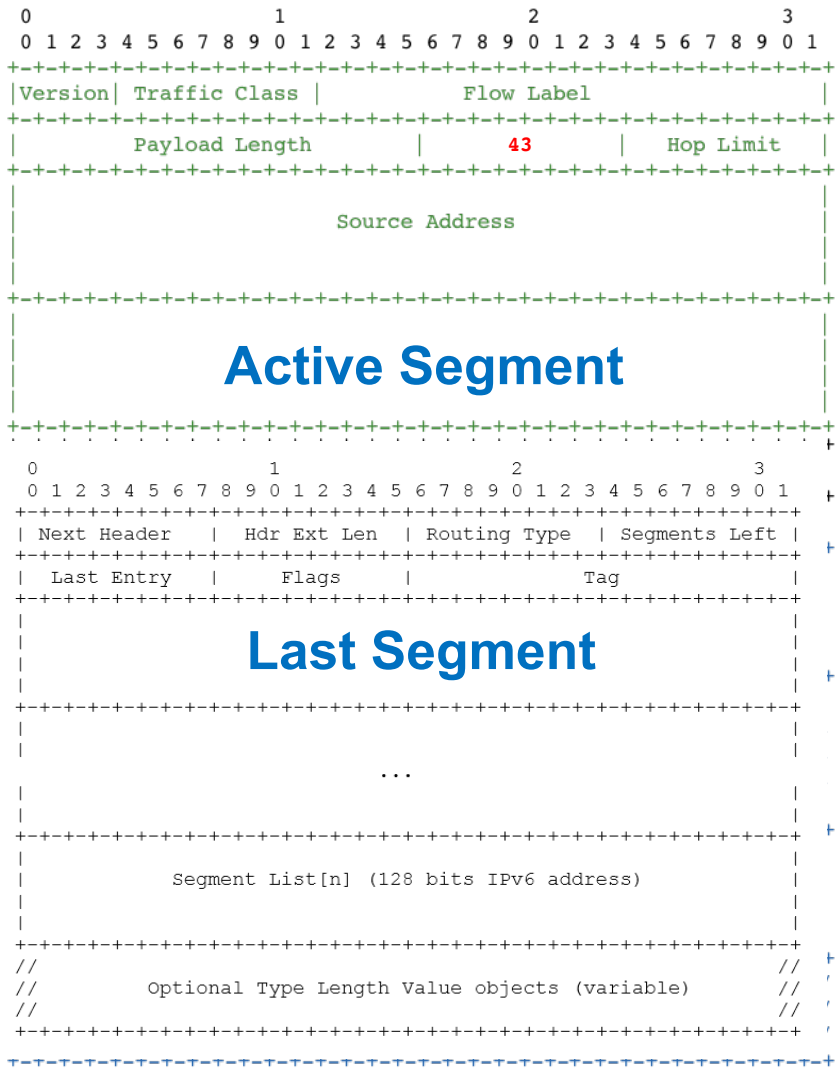
- **draft-ietf-6man-segment-routing-header**
  SRv6のヘッダーと、その基本的な処理方法
- **draft-filsfils-spring-srv6-network-programming**
  SRv6の「Network Programming Concept」の定義。（SRv6のより詳細な動作）
- **draft-dawra-idr-srv6-vpn**
  SRv6を使ったIPv4/IPv6 VPN、EVPN用のBGP拡張

SR Header

draft-ietf-6man-segment-routing-header

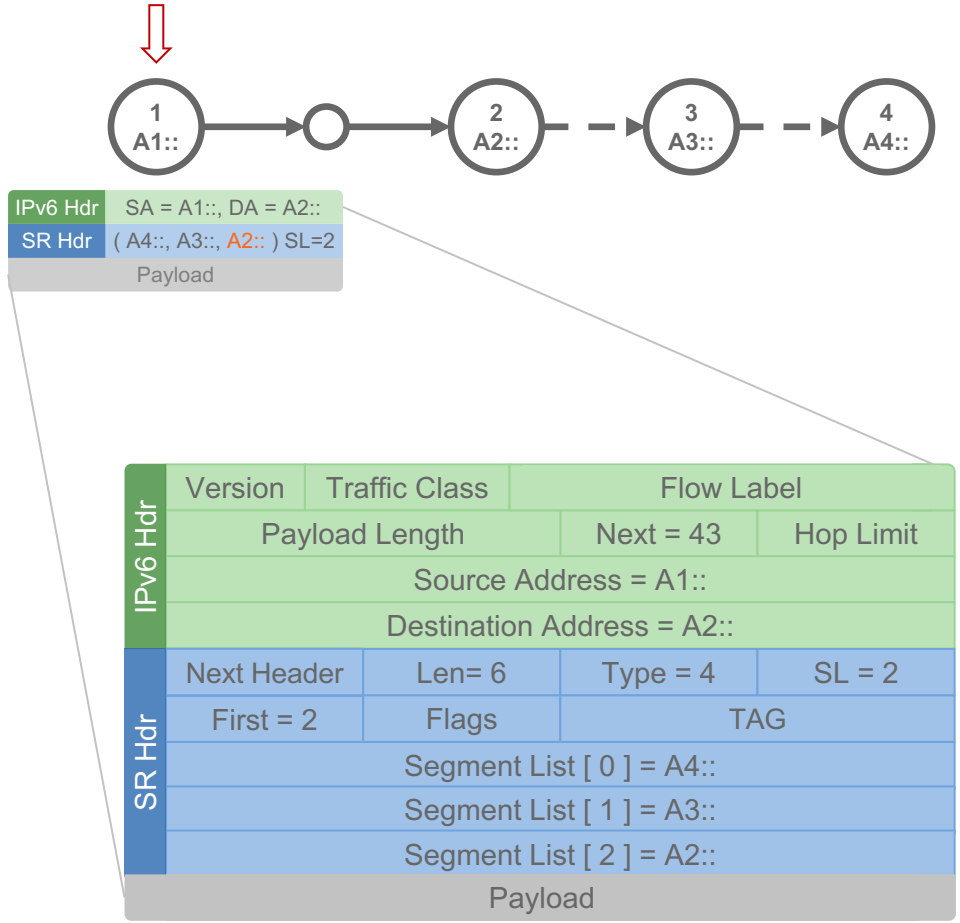
SRH

- SRH は以下を含む
  - the list of segments
  - Segments left (SL)
  - Flags
  - TLV
- アクティブセグメントは IPv6 DA に格納される
- ネクストセグメント (次の宛先) はSegment List の index “SL(Segment-Left)-1”に格納されているセグメント
- 最後のセグメントは index 0
  - Segment List は逆順で格納される

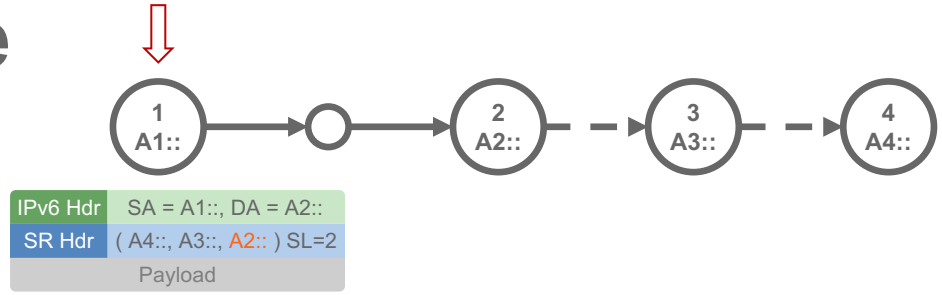


Source Node

- SR-capableなソースノード
- SR Header (SRH) は以下より構築
  - Segment list in reversed order of the path
    > Segment List [0] is the LAST segment
    > Segment List [$n-1$] is the FIRST segment
  - Segments Left is set to $n-1$
  - First Segment is set to $n-1$
- IP DA は first segment がセットされる
- IP AD にしたがってパケット送信
  - 通常の IPv6 フォワーディング



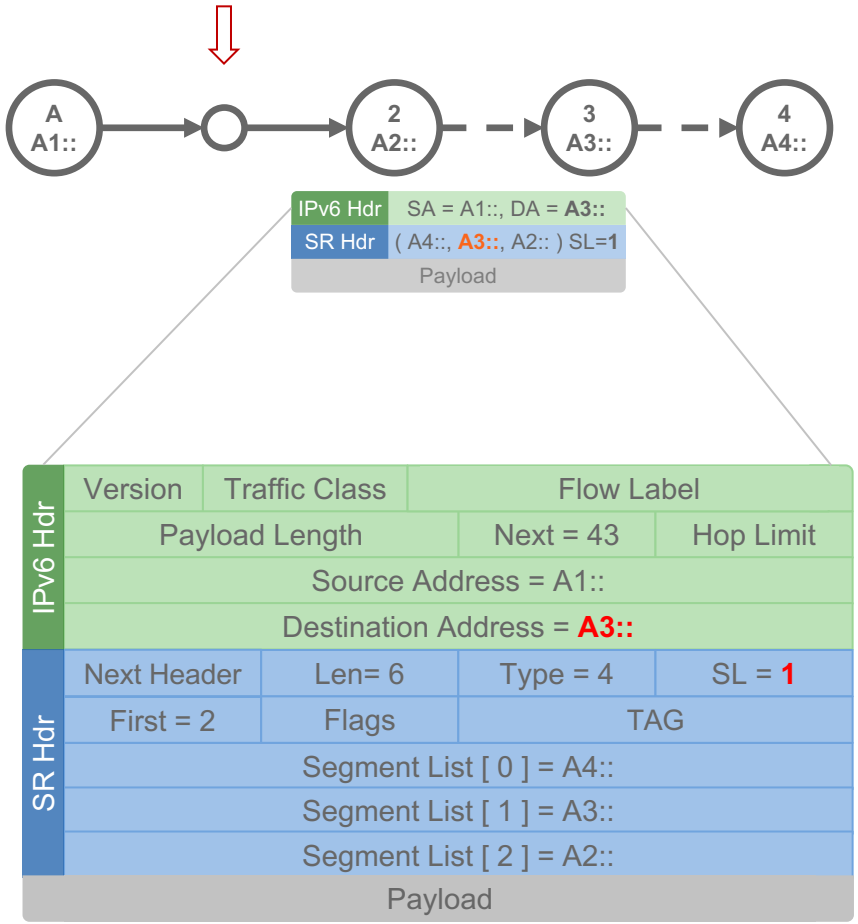
Non-SR Transit Node



- 単純な IPv6 フォワーディング
- 単純なIPv6 DA ベース
- SRH のinspectionやupdateは行わない

SR Segment Endpoints

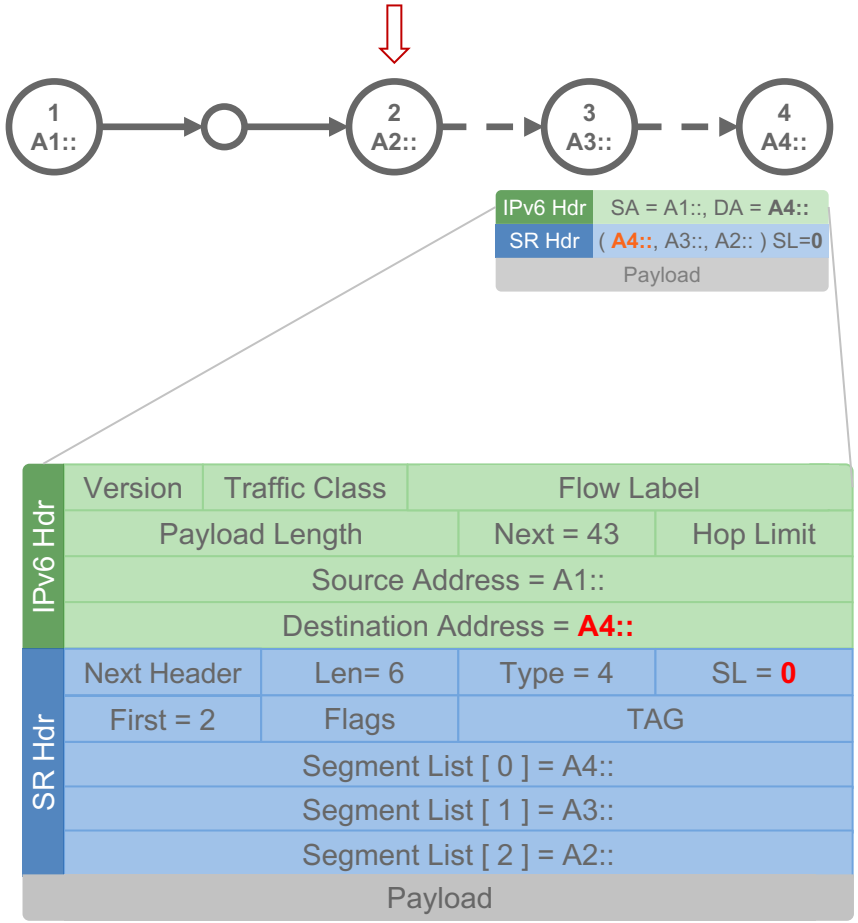
- SR エンドポイント: SR-capableでIP DA に自アドレスを持つノード
- SR エンドポイントはSRHを確認し以下を実行:
  - IF Segments Left > 0, THEN
    - > Decrement Segments Left (-1)
    - > Update DA with Segment List [Segments Left]
    - > Forward according to the new IP DA



SR Segment Endpoints

- SR エンドポイント: SR-capableでIP DA に自アドレスを持つノード
- SR エンドポイントはSRHを確認し以下を実行:
  - IF Segments Left > 0, THEN
    - > Decrement Segments Left (-1)
    - > Update DA with Segment List [Segments Left]
    - > Forward according to the new IP DA
  - ELSE (Segments Left = 0)
    - > Remove the IP and SR header
    - > Process the payload:
      - Inner IP: Lookup DA and forward
      - TCP / UDP: Send to socket
      - …

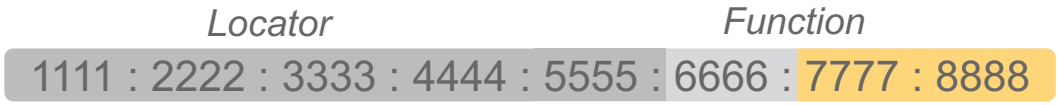
Standard IPv6 processing
The final destination does not have to be SR-capable.



SRv6 Network Programming

draft-filsfils-spring-srv6-network-programming

Segment format

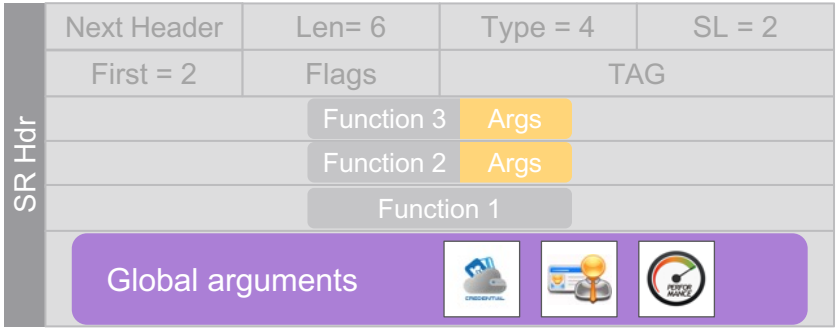


- SRv6 SID は128ビットのIPv6アドレス表記
  - **Locator**: セグメントをペアレントノードへ**Route**するためのビット
  - **Function**: ペアレントノードにおいて取られる**Action**を示すビット
    > **Argument** [optional]: 最後のビットはFunctionで参照される引数
- ビット長は可変
  - SIDのフォーマットはペアレントノードがローカルに規定
- SIDはペアレントノードにおいて明示的に有効にしないといけない
  - ローカルアドレスではデフォルトでSIDとして**有効ではない**
  - SIDはインタフェースに関連付けられている必要はない

ペアレントノード
そのSIDの保有ノード、オリジネーター。

SID Function – Anything

- SID Functionはペアレントモードにおいてローカルに定義される
  - つまり、なんでも定義できる
- SRヘッダーが”Network Program”の情報を含んでいる

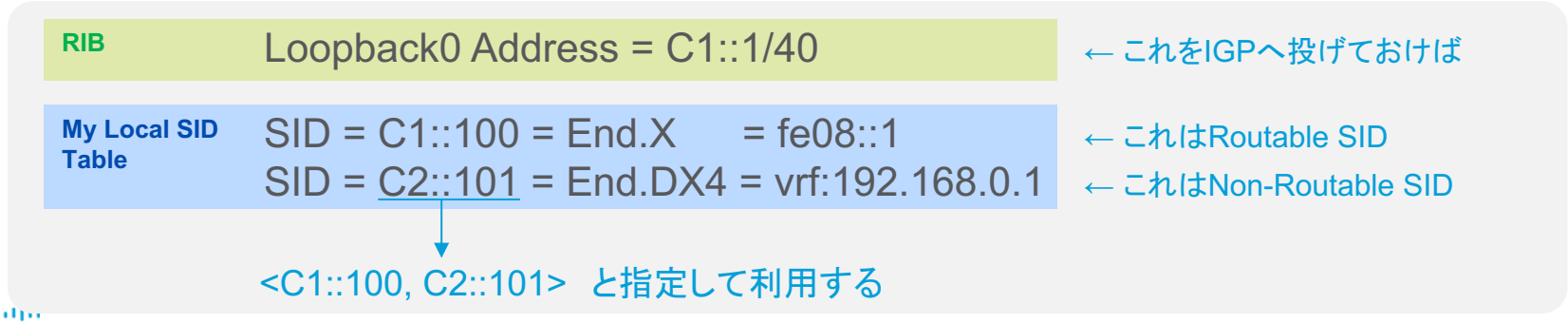


The concept of "SRv6 network programming" refers to the capability for an application to encode any complex program as a set of individual functions distributed through the network. Some functions relate to underlay SLA others to overlay/tenant, others to complex applications residing in VM and containers.

“SRv6 Network Programming”のコンセプトは、アプリケーションが複雑なプログラムを「ネットワーク上に分散する個々の機能の組み合わせ」としてエンコードすることを可能にする。

補足1

- IPv6 SID = IPv6 Addressではない。(IFに紐付かない)
- ただし、IPv6 SIDのLocator部分はIPv6 PrefixとしてIGPに広報することが出来る。(しなくてもいい)
- つまり、SIDをどこからでもRoutableにすることが出来る。(IPv6 forwardingをしている場合)
- RoutableでないSIDは、RoutableなSIDとListすることで利用可能になる。
  (ラベルで言うPrefix-SIDとAdjacency SIDみたいに)
- RoutableであればAdjacency SIDを直接指定することも可能。



Well Know Function

- 一般的な機能はDraft内でWell Know Functionとして定義されている。
- ただし、

> The list is not exhaustive. In practice, any function can be attached to a local SID: e.g. a node N can bind a SID to a local VM or container which can apply any complex function on the packet.

リストは網羅的ではありません。実際には、任意の機能をローカルSIDに取り付けることができます。例えば、ノードNは、SIDをローカルVMまたはコンテナにバインドすることができ、コンテナはパケット上に複雑な機能を適用することができる

End = IPv6 DAが自分

Well-Known “End” Functions

Function	場所	動作概要	機能
End	Core	DestinationとSRHを書き換えて、Next-hopをRIBから探して送る	Prefix-SID
End.X	Core	DestinationとSRHを書き換えて、決められたNext-hopへ送る	Adjacency-SID
End.T	Core	DestinationとSRHを書き換えて、Next-hopを「指定されたRIB」から探して送る	Multi-table Operation
End.DX2	Edge	SRHを外して、決められた送信IF（VLAN）へ送る (NH=59)	L2VPN
End.DX6	Edge	SRHを外して、決められたIPv6 Next-hopへ送る(NH=41)	VPNv6 Per-CE Label
End.DX4	Edge	SRHを外して、決められたIPv4 Next-hopへ送る(NH=4)	VPNv4 Per-CE Label
End.DT6	Edge	SRHを外して、IPv6 Next-hopを「指定されたRIB」から探して送る(NH=41)	VPNv6 Per-VRF Label
End.DT4	Edge	SRHを外して、IPv4 Next-hopを「指定されたRIB」から探して送る(NH=4)	VPNv4 Per-VRF Label
End.B6	Edge	SRHは触らず、新しいSID List（SRH）を挿入して、その先頭へ送る	Binding SID
End.B6.Encaps	Edge	SRHを書き換えて、新しいSID List（Outer Header）でEncapして、その先頭へ送る	Binding SID (Encap)
End.BM	Edge	DestinationとSRHを書き換えて、Labelを付与して、その先頭へ送る	SRv6/SR-MPLS Binding
End.S	Core	一番最後（or 複数）のSIDでTable検索し、Next-hopを探して送る	ICN
End.AS	Core	Outer Headerを外して、決められた送信IFへ送る。決められた受信IFに入ってきたPacketにOuter Headerを付与し、その先頭へ送る	Service-Chaining (Proxy)
End.AM	Core	DestinationとSRHを書き換えて、決められた送信IFへ送る。決められた受信IFに入ってきたPacketにSRHを付与し、その先頭へ送る	Service-Chaining (マスカレード)

Transit behavior

Transit = IPv6 DAが自分じゃない。

Function	場所	動作概要
T	Core	ただのIPv6 Routing
T.Insert	Core	新しいSRHを挿入して、その先頭に送る
T.Encaps	Core	新しいIPv6 Header（SRHつき）を追加して、その先頭に送る（L3）
T.Encaps.L2	Core	新しいIPv6 Header（SRHつき）を追加して、その先頭に送る（L2）

T.InsertはRFC2460の規定に注意。

https://tools.ietf.org/html/draft-ietf-6man-rfc2460bis-08#page-28

> The insertion of Extension Headers by any node other than the source of the packet causes serious problems. Two examples include breaking the integrity checks provided by the Authentication Header Integrity [RFC4302], and breaking Path MTU Discovery which can result in ICMP error messages being sent to the source of the packet that did not insert the header, rather than the node that inserted the header.
>
> One approach to avoid these problems is to encapsulate the packet using another IPv6 header and including the additional extension header after the first IPv6 header, for example, as defined in [RFC2473]

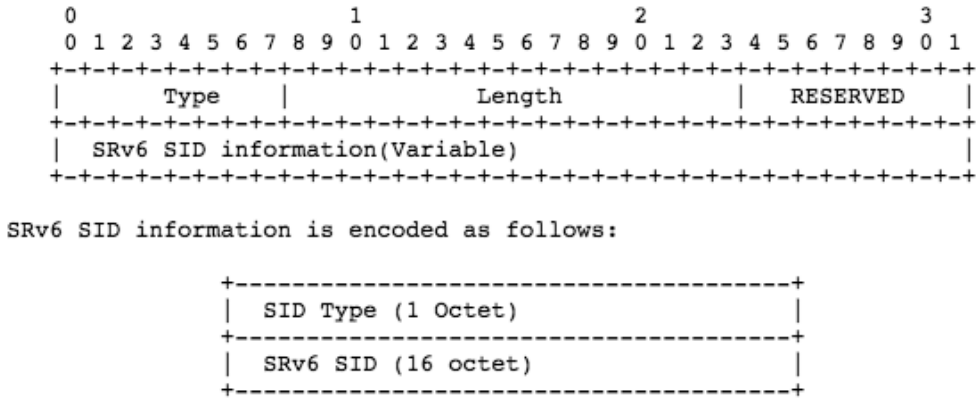
BGP SRv6-VPN
draft-dawra-idr-srv6-vpn

SRv6-VPN

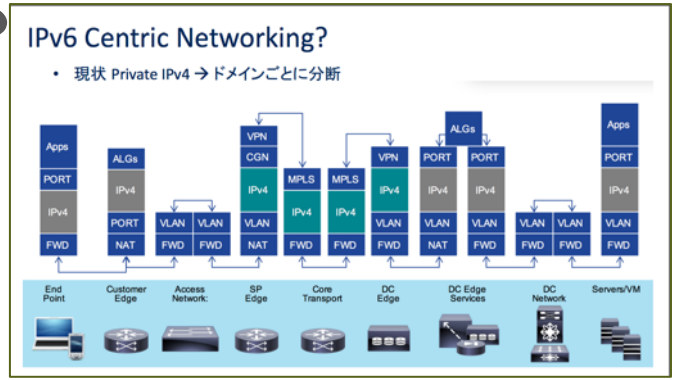
Function	場所	動作概要	機能
End	Core	DestinationとSRHを書き換えて、Next-hopをRIBから探して送る	Prefix-SID
End.X	Core	DestinationとSRHを書き換えて、決められたNext-hopへ送る	Adjacency-SID
End.T	Core	DestinationとSRHを書き換えて、Next-hopを「指定されたRIB」から探して送る	Multi-table Operation
End.DX2	Edge	SRHを外して、決められた送信IF（VLAN）へ送る (NH=59)	L2VPN
End.DX6	Edge	SRHを外して、決められたIPv6 Next-hopへ送る(NH=41)	VPNv6 Per-CE Label
End.DX4	Edge	SRHを外して、決められたIPv4 Next-hopへ送る(NH=4)	VPNv4 Per-CE Label
End.DT6	Edge	SRHを外して、IPv6 Next-hopを「指定されたRIB」から探して送る(NH=41)	VPNv6 Per-VRF Label
End.DT4	Edge	SRHを外して、IPv4 Next-hopを「指定されたRIB」から探して送る(NH=4)	VPNv4 Per-VRF Label
End.B6	Edge	SRHは触らず、新しいSID List（SRH）を挿入して、その先頭へ送る	Binding SID
End.B6.Encaps	Edge	SRHを書き換えて、新しいSID List（Outer Header）でEncapして、その先頭へ送る	Binding SID (Encap)
End.BM	Edge	DestinationとSRHを書き換えて、Labelを付与して、その先頭へ送る	SRv6/SR-MPLS Binding
End.S	Core	一番最後（or 複数）のSIDでTable検索し、Next-hopを探して送る	ICN
End.AS	Core	Outer Headerを外して、決められた送信IFへ送る。決められた受信IFに入ってきたPacketにOuter Headerを付与し、その先頭へ送る	Service-Chaining (Proxy)
End.AM	Core	DestinationとSRHを書き換えて、決められた送信IFへ送る。決められた受信IFに入ってきたPacketにSRHを付与し、その先頭へ送る	Service-Chaining (マスカレード)

SRv6-VPN SID TLV

- BGP Prefix-SID AttributeにTLVを付与してVPN用のSIDを広報する。
- 対象としているNLRI=VPNv4, VPNv6, EVPN（後ほど追加）



まとめ：SRv6の何がいいのか？（ラベルとの違い）

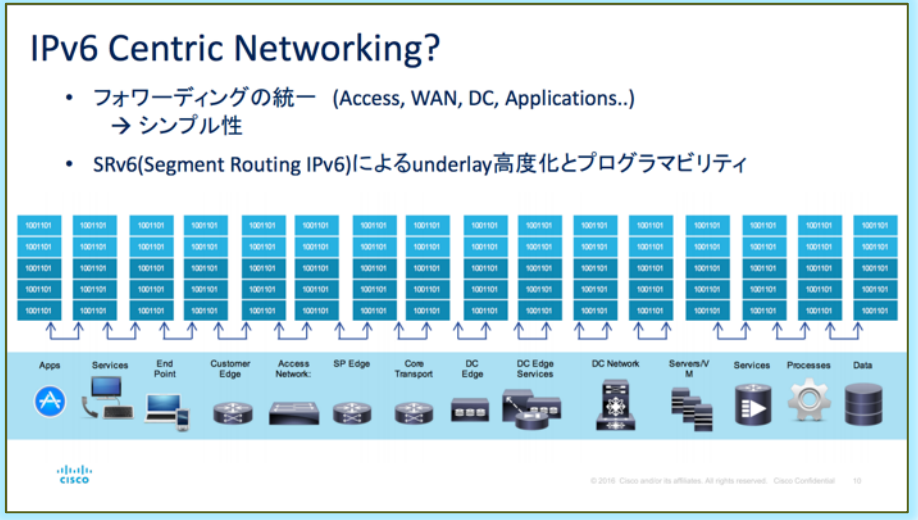


1) **IPv6によりドメインごとに分断されないネットワークが実現できる。SIDが128bitもあるので、色んな機能が定義できる可能性がある。**
   - 識別子のためにvlan idなどを使う必要はもう無い
   - MobilityやContent Networkingなどにも応用できる
   - End-to-endでの（Application, DC, Core, Access, CPE, UE..）、共通転送メカニズムになりうる
   - VPNやMobilityなどのためにTunnel必要ない
   - Label Shim-Layerを排除できる
   - ネットワーク内のステートを最小化できる

2) **SIDをRouting情報として広報出来る。SRv6に対応していないルーターでも転送できる。**
   - IPv6が届けば、どこからでもOverlay、Chainingなどが出来る。
   - Strategic NodeだけがSRv6に対応していればいい。

3) **Linux、VPPで実装されている。**

End-to-EndのNetwork Programmingが、NetworkのCost最小で実現できる可能性がある。



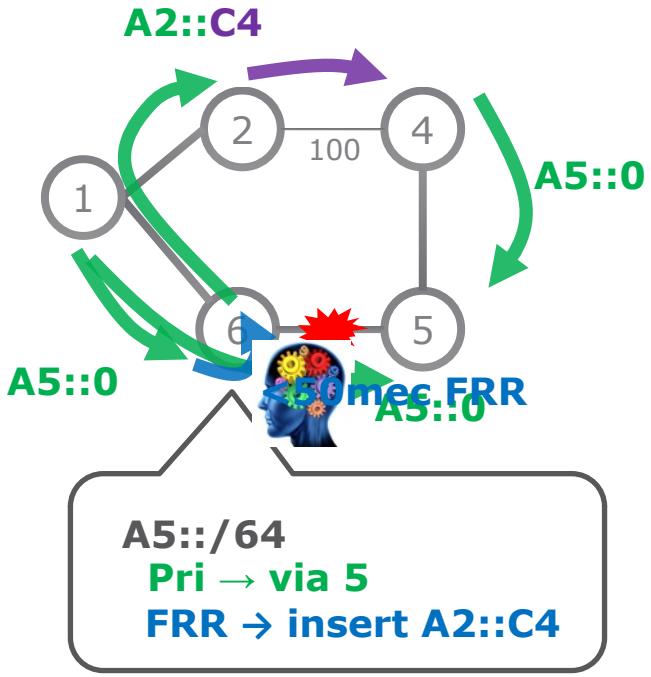
Segment Routing Use case

SR の一般的なユースケース

- TILFA
- 自動 ECMP ロードバランス
- トラフィックマトリックスの自動化
- ブラックホール検知
- アプリケーション単位のトラフィックエンジニアリング
- ネットワークの状態に応じたトラフィックエンジニアリング
- サービスチェイニング
- eBGP ピアリングリンク状態監視
- MicroLoop Avoidance

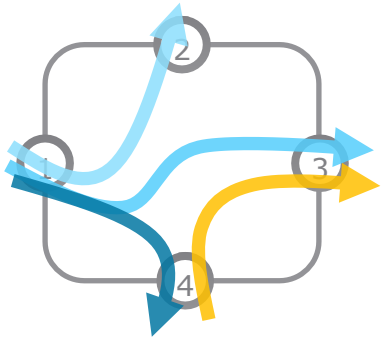
TILFA for SRv6

- MPLS同様にlocal link, node/SRLGに対して50msecのProtection
- Simpleに動作して、簡単
  - IGPメトリックに従って各ノードが自律的に計算
  - どんなトポロジでも100%カバー
  - 切り替え後の経路を確認可能 (backup = post convergence)
- Backup pathが最適経路
  - post-convergence pathのLink利用率を計算可能
  - Re-optimization後に経路が切り替わることもない
- Incremental deployment
- Distributed and Automated Intelligence



トラフィックマトリックス収集の自動化

- トラフィックマトリックスの目的
  - キャパシティープランニング
  - セントラライズされたトラフィックエンジニアリング
  - IP/Optical の最適化
- Prefix SIDは網内でユニークに設定可能
  (ユニークじゃなくてもいい)
  - TEを使用しなくても対地のトラフィックマトリックスを収集可能
- SR を使用したトラフィックマトリックス収集の自動化



	1	2	3	4
1				
2				
3				
4				

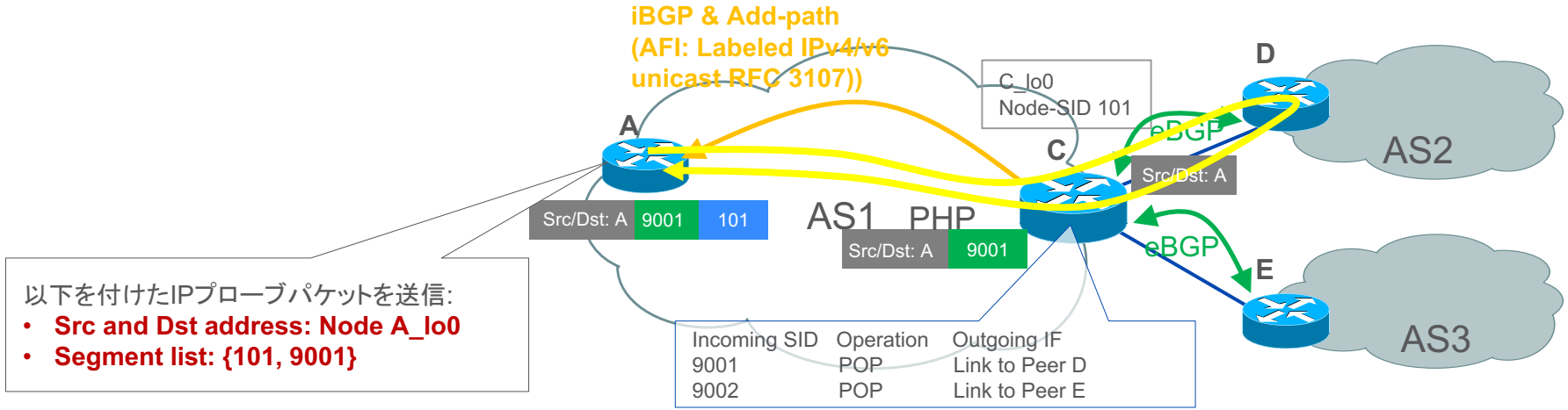
Show Prefix-SID Counter History Database

```
show traffic-collector ipv4 counters ( prefix [<prefix>] | label <label> ) [detail] [private]
```

```
RP/0/RSP0/CPU0:R2#show traffic-collector ipv4 counters prefix 1.1.1.10/32 detail
Prefix: 1.1.1.10/32  Label: 16010 State: Active
Base:
    Average over the last 5 collection intervals:
        Packet rate: 9496937 pps, Byte rate: 9363979882 Bps

    History of counters:
        23:01 - 23:02: Packets 9379529, Bytes: 9248215594
        23:00 - 23:01: Packets 9687124, Bytes: 9551504264
        22:59 - 23:00: Packets 9539200, Bytes: 9405651200
        22:58 - 22:59: Packets 9845278, Bytes: 9707444108
        22:57 - 22:58: Packets 9033554, Bytes: 8907084244
```

PEからPrefix毎にCounter取得
History表示も可能

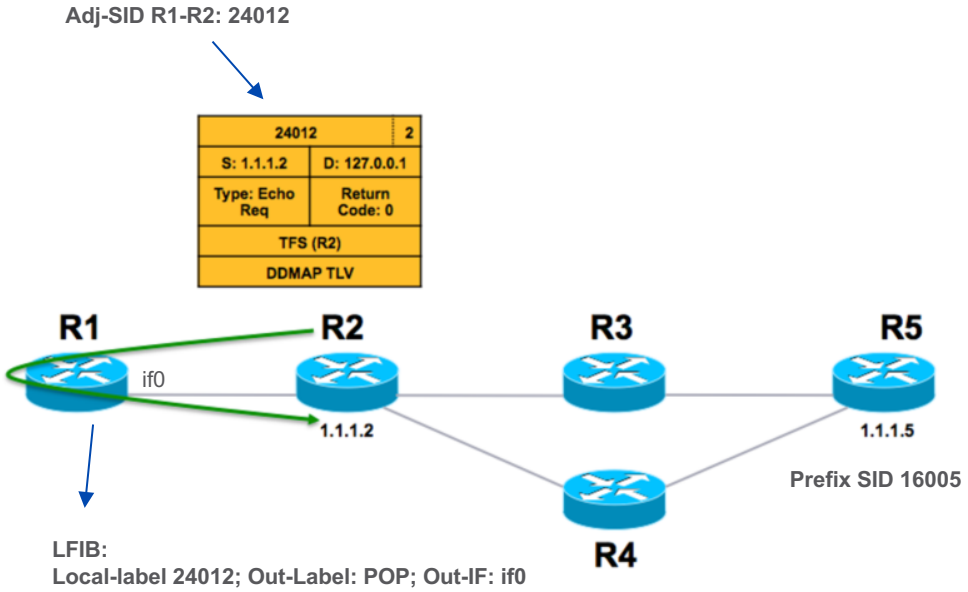
Monitoring a remote peering link



- ノードAは(BFD Echoモードのように)IPプローブを送信することによりCからDへのピアリングリンクのデータプレーンの正常性を監視可能
  - Node-SID 101によりプローブをC及び外部のAdj-SID 9001に向ける
  - PeerAdj-SID 9001によりプローブをCからDへのピアリングリンクに向ける
  - SRヘッダがCで削除されDは単純なIPパケットをAの宛先アドレスで受信
  - DはIPフォワーディングを通してプローブをAに返送

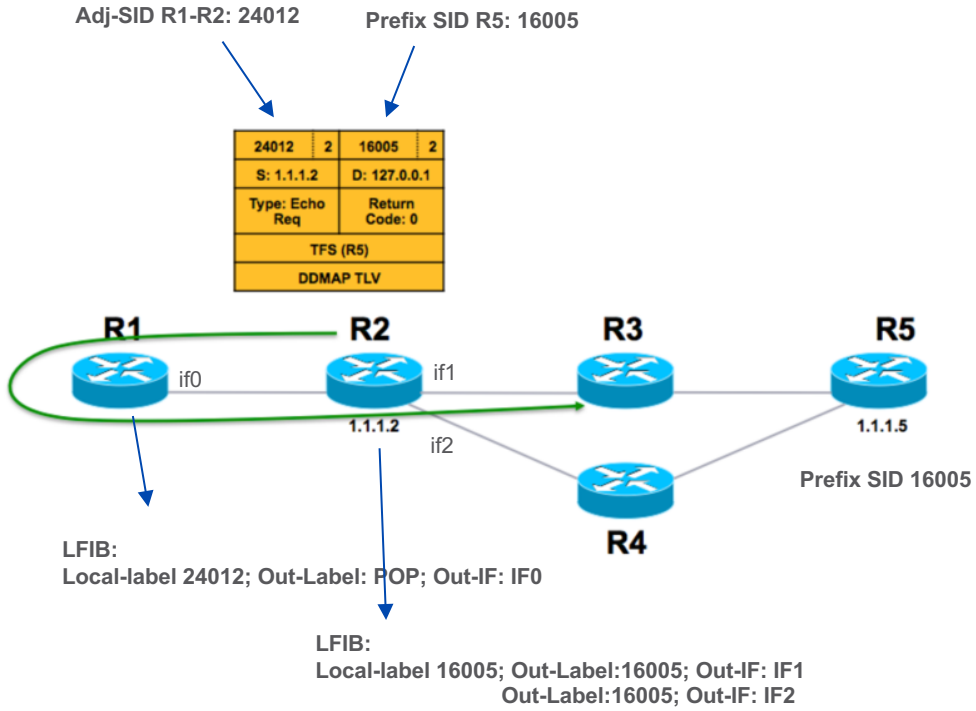
Definitive Black hole Detection

- 2step プロセス
- STEP1: 全ての上流のNeighborに対してMPLSのpathの疎通を確認 (R1からR2向けのAdj-SIDをつけてR2からR1向けにPingをだす)



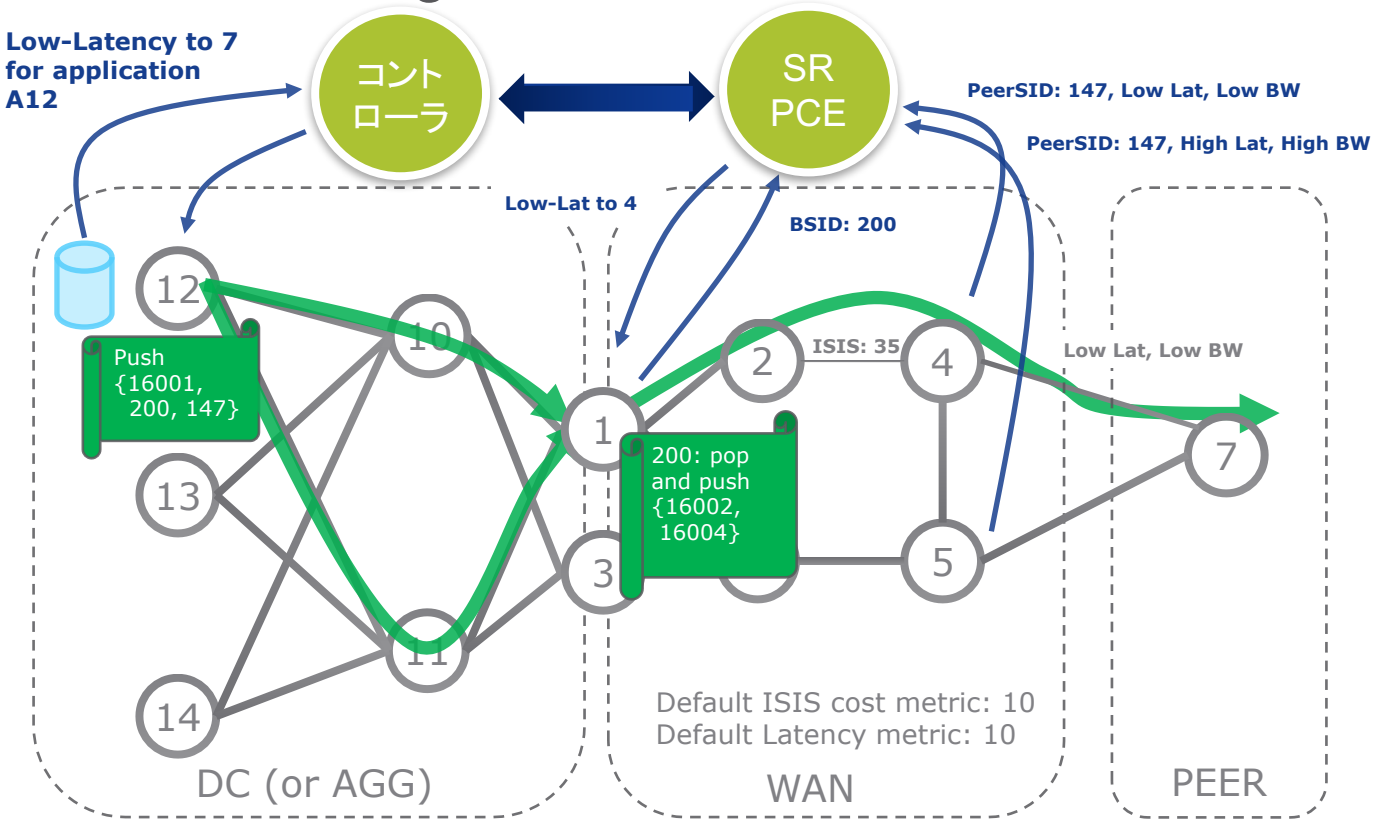
Definitive Black hole Detection

- STEP2: 各Destination Prefixに対するSIDを2つ目のSIDとしてつけることでLFIBとRIBのconsistencyチェックを行うことができます



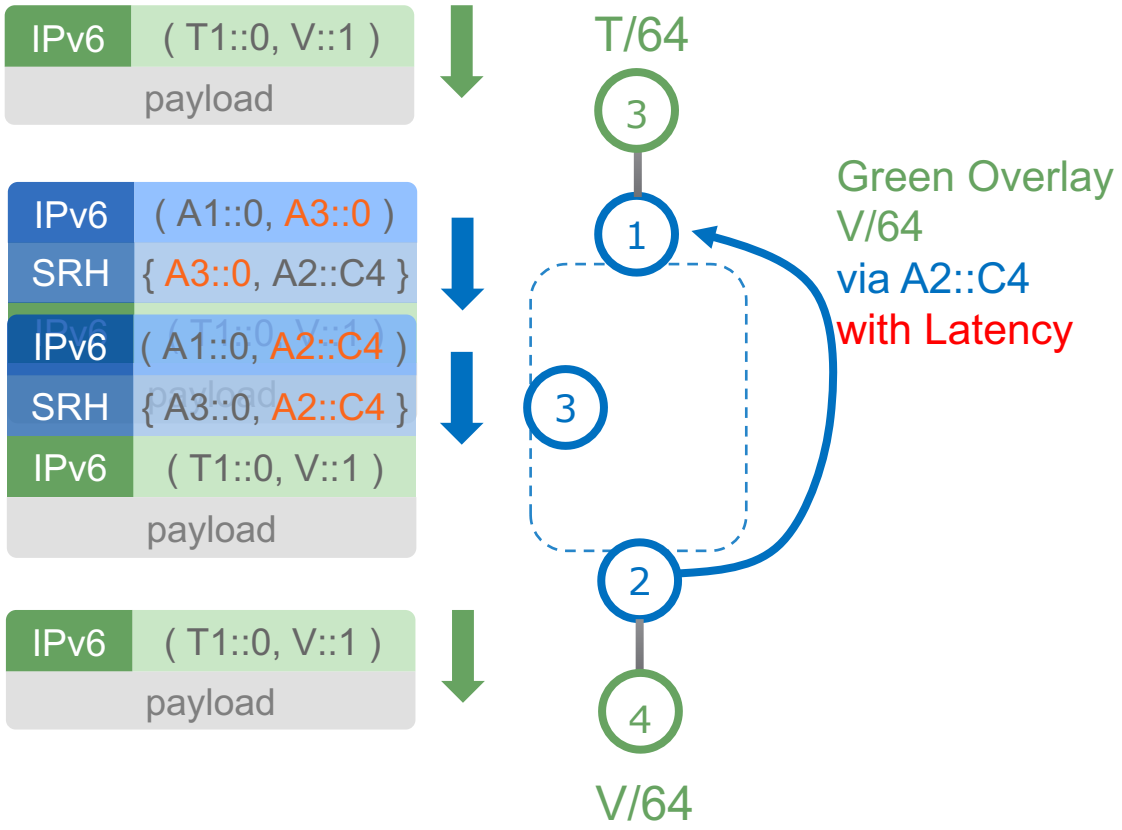
アプリケーション毎に制御されたルーティング
Application Engineered Routing

- アプリケーションフロー毎の制御
- エンド-エンド
  - DC, WAN, AGG, PEER
- シグナリングなし
- Midpoint ステートなし
- 境界で再クラシファイなし

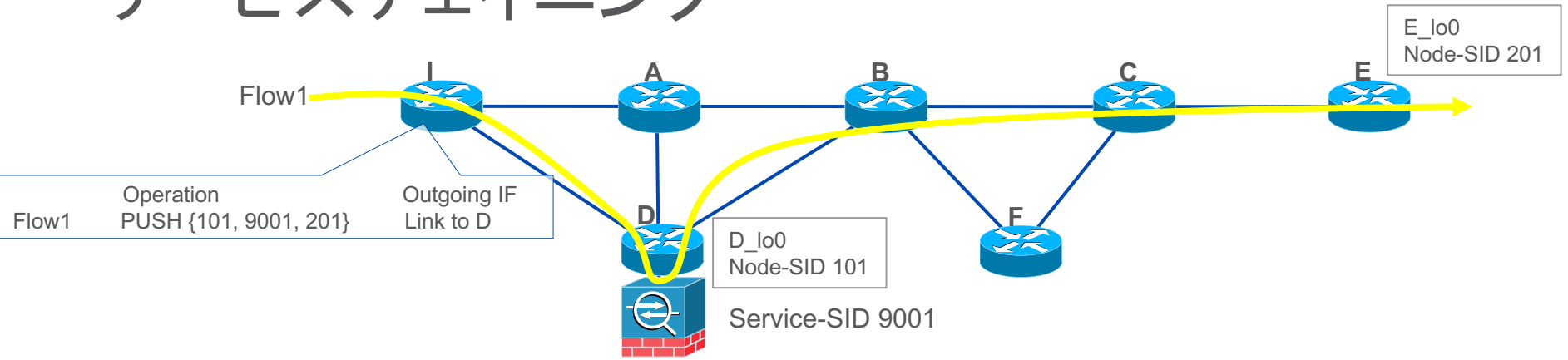


SRv6 Overlay with Underlay Control

- SRv6は不要なoverlay protocolを排除するだけではない
- SRv6では通常のOverlay protocol(BGP等)で解決できない問題を解決できる(遅延を考慮したパス設定など)



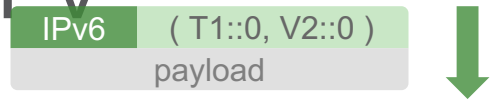
サービスチェイニング

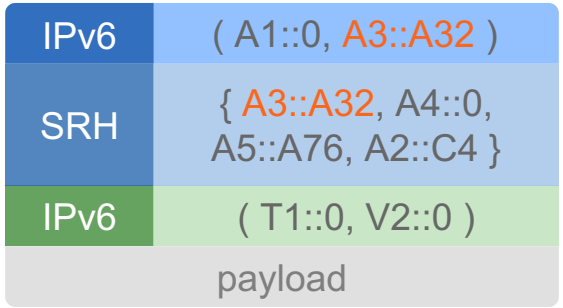


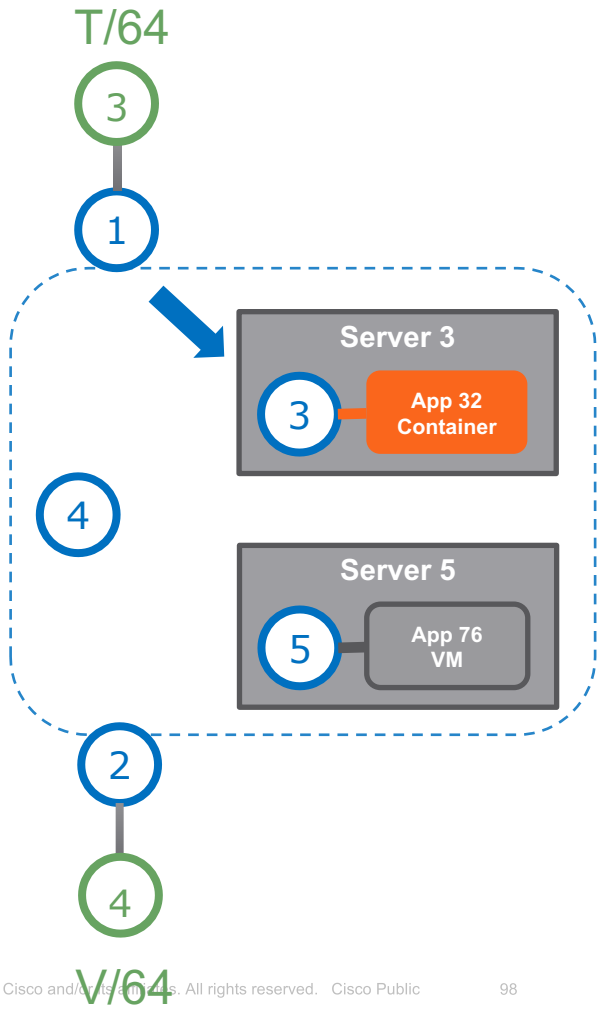
- SRを使ってDPI、アカウンティング等のようなサービスノードをパケットが通るように向けることが可能
- IからEへの或るフローはDに接続されている特定の処理(DPI、FW等)が必要な場合もある
- Dのローカル範囲であるサービスセグメントを使ってアプリケーションをサポート可能
- ポリシーによりフローのパケットにセグメントリスト{101, 9001, 201}を付けてフォワードするようにIに指示

SRv6 Integrated NFV Service chaining

- SID:A3::A32の意味
  - コンテナ内のアプリを32とする
  - Location@SIDがA3::/64
- Stateless
  - NSHではper-chainのstateをFabric内に作る必要がある
  - SRには不要
- アプリはSRサポート/未サポートどちらでもOK



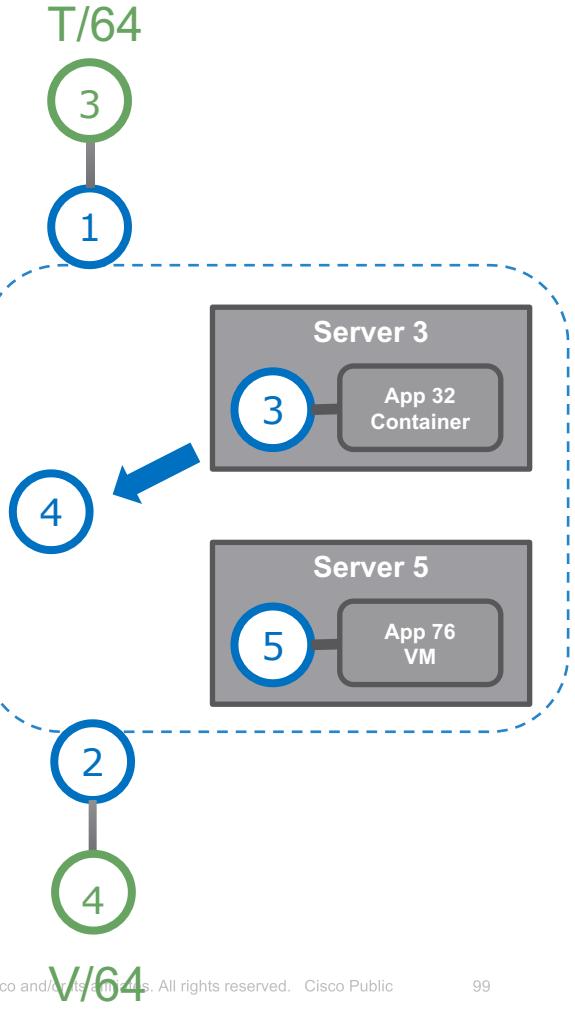




SRv6 Integrated NFV

- UnderlayのSLAを考慮してRouterへパケットを転送

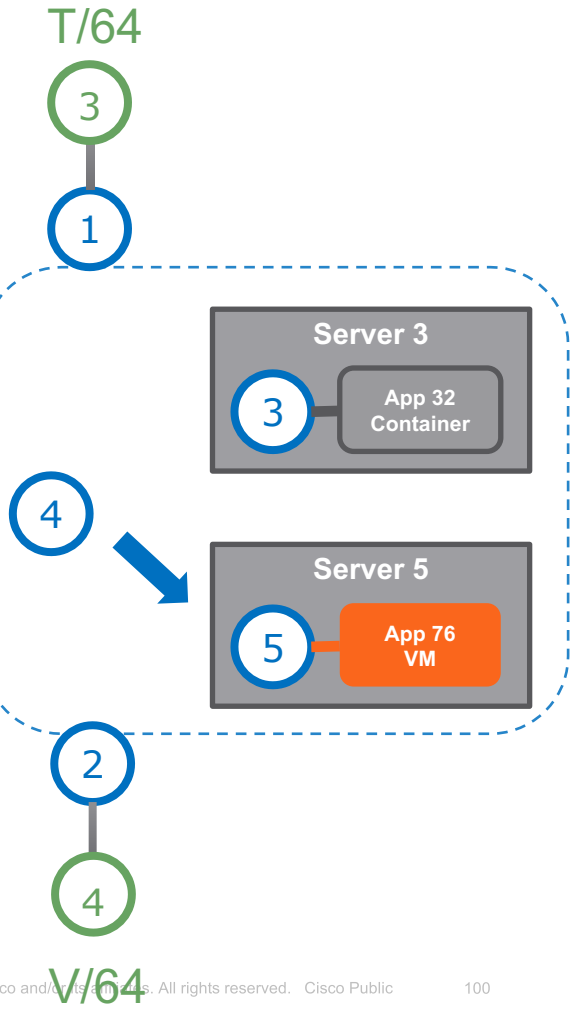
IPv6	(A1::0, A4::0)
SRH	{ A3::A32, A4::0, A5::A76, A2::C4 }
IPv6	(T1::0, V2::0)
payload	



SRv6 Integrated NFV

- SID:A3::A32の意味
  - アプリを76とする
  - Location@SIDがA5::/64
- Stateless
  - NSHではper-chainのstateをFabric内に作る必要がある
  - SRには不要
- アプリはSRサポート/未サポートどちらでもOK

IPv6	(A1::0, A5::A76)
SRH	{ A3::A32, A4::0, A5::A76, A2::C4 }
IPv6	(T1::0, V2::0)
payload	

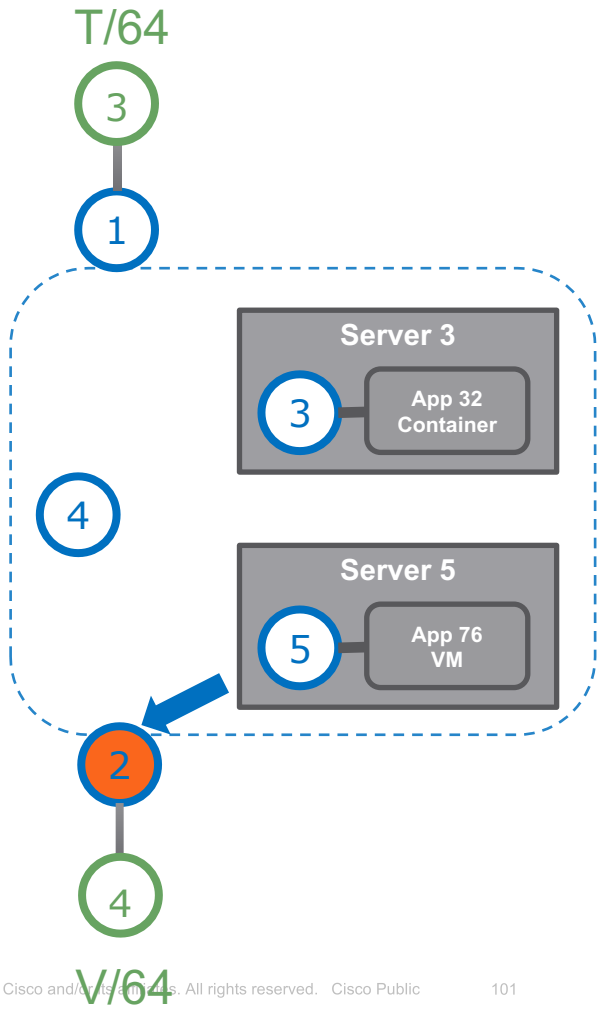


SRv6 Integrated NFV

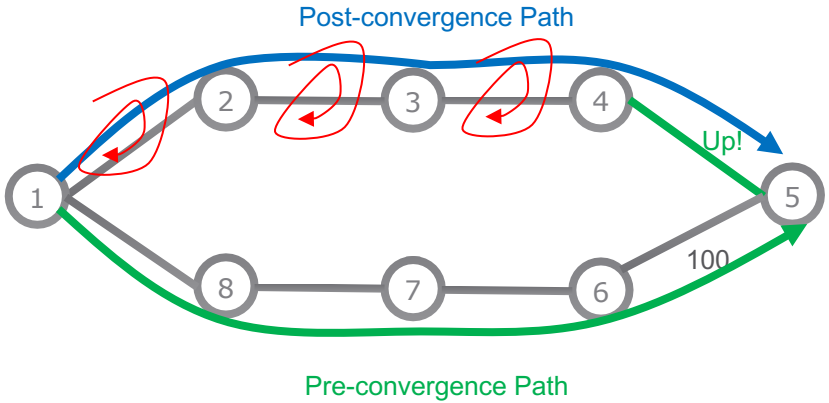
- SRv6 Edge Routerにサービスチェイニング後のパケットを転送

IPv6	(A1::0, A2::C4)
SRH	{ A3::A32, A4::0, A5::A76, A2::C4 }
IPv6	(T1::0, V2::0)
payload	

IPv6	(T1::0, V2::0)
payload	



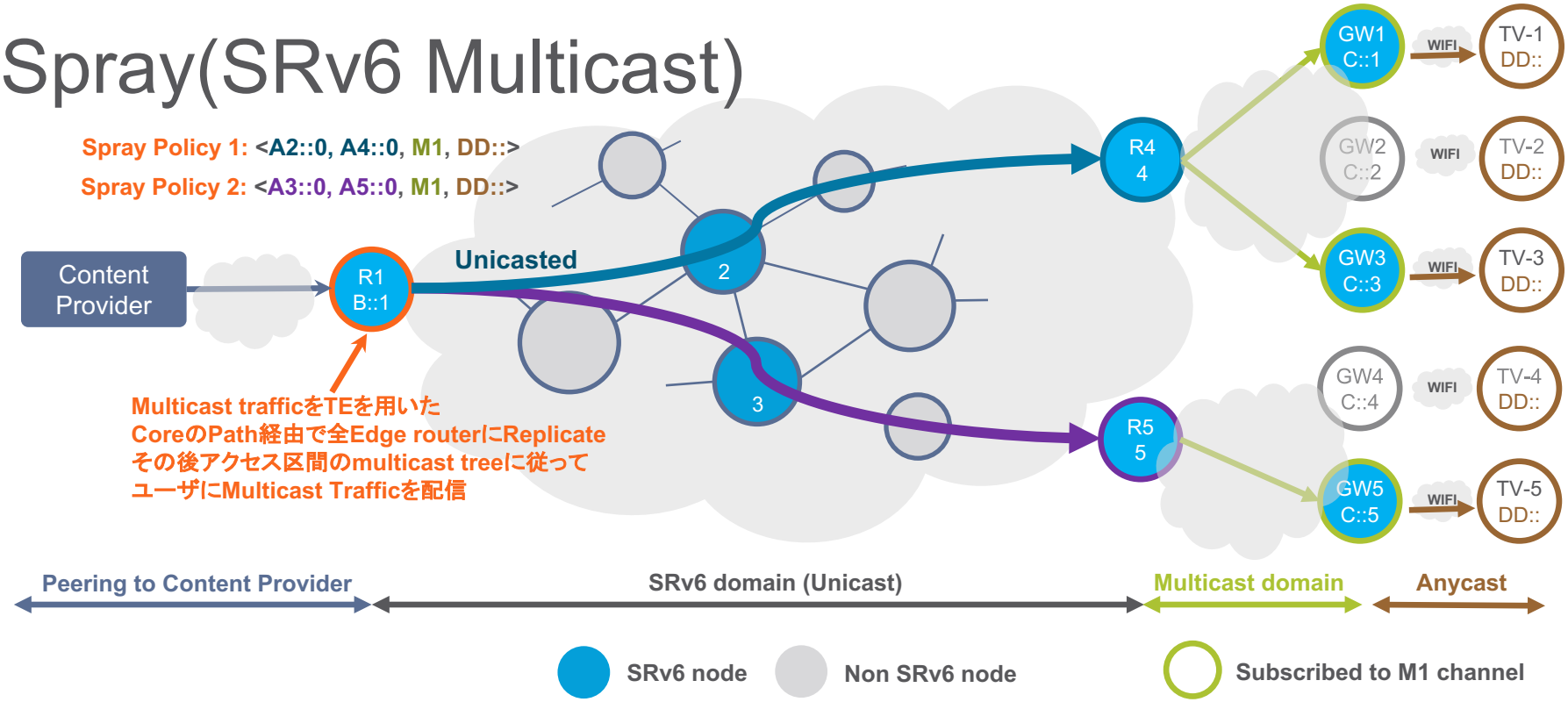
MicroLoop Avoidance
Upon link up convergence



- IP hop-by-hop routing may induce uloop at any topology transition
  - Link up/down, metric up/down

Microloopは避けることが出来ない永遠の課題

Spray(SRv6 Multicast)



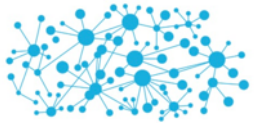
CoreではMulticastを用いることなく、Flexibleにコンテンツを流すことができる

Summary

Segment Routing Summary

- ネットワークの**シンプル化**：制御プレーン簡素化、ステート削減
- スケーラビリティと柔軟性
- **100% のトポロジカバー率**で最適ルーティングパス考慮した **FRR**
- データプレーンアグノスティック (**MPLS, IPv6**)
- **シームレスマイグレーション**：既存ラベルプロトコル (LDP, RSVP)
- ApplicationがNetworkをコントロール(**自動化**)
  - Stateは全てSourceで保持、infrastructureではステートは持たない
  - DC内のサーバからMetro/WANを通してEnd-to-endでpolicy-awareなnetwork architectureを実現 (既存のMPLS/IPv6のNetwork越しでも運用可能)
- 今後 5～10 年先を見据えた IP Architecture です!!

Simple



Scalable

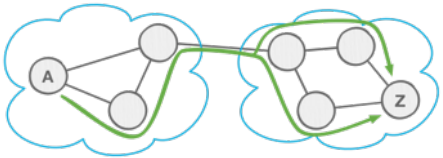
Seamless deployment

SR

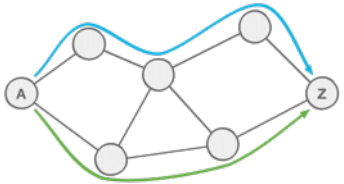
Segment Routing

www.segment-routing.net

Network Programming



Unified Forwarding Plane



Traffic Engineering


THERE'S NEVER BEEN A
BETTER TIME
to change the world
cisco