T4
ルーティング チュートリアル
AS間経路制御 – BGP –

小島 慎太郎
インターネットマルチフィード(株)
koji@mfeed.ad.jp

AS間経路制御

AS間の経路を“制御”しようということ

http://www.flickr.com/photos/7687126@N06/2313399001/

完全に制御する
ことはできない

http://www.flickr.com/photos/gnislew/1453056073

AS間経路制御

- 顧客 / peering partner / transit 提供者にも彼らなりのpolicy がある

- 自分たちで制御できるものを

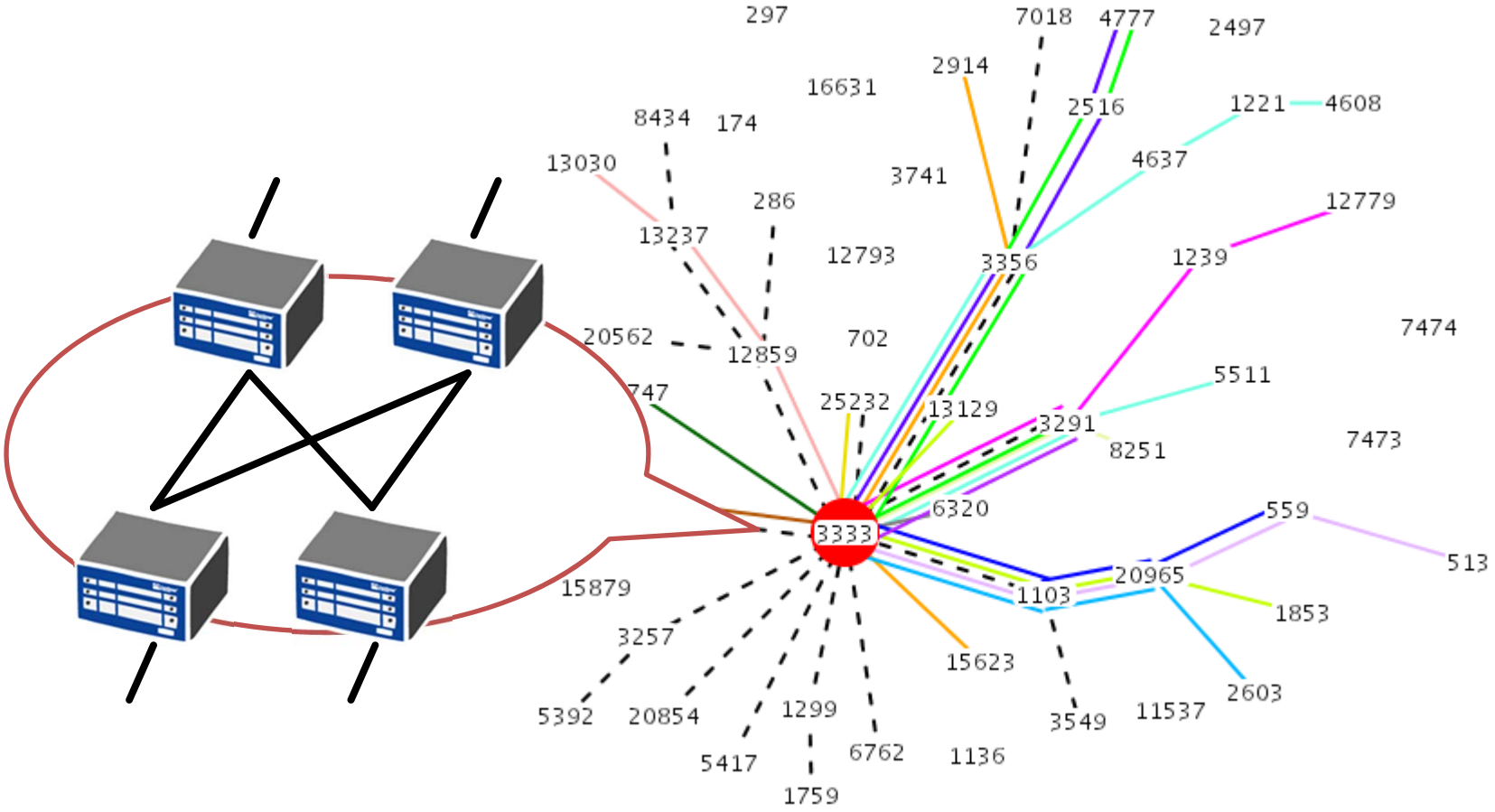
“最大限”

制御する方法をご紹介します

AS間の経路を制御するために

こうではなく

AS間の経路を制御するために



こんな感じ (が目標です)

http://bgplay.routeviews.org/

AS間経路制御 – BGP –

- BGP 設計
  - addressing
  - eBGP policy / 設計
    - 経路広告
    - 経路受信
  - iBGP policy / 設計
- BGP 運用
  - traffic engineering
  - その他
- 道具箱に 道具を詰める
- community の一員になる

BGP 設計

BGP 設計

1. network policy を決める
    1. 可用性 (どのレベルの障害に耐えうるか? POP / router 筐体 / module / 回線 / 電源)
    2. 品質 (packet loss 率 < 0.? % / jitter < 0.x? ms / Latency @日本国内 < ??ms)
    1. 運用性
    2. 拡張性
    3. cost (10G ポートあたり < ?百万円)

2. network policy を満たす実装のひとつとして，network を設計する
    1. POP 配置 / DC 選定
    2. cable path
    3. 収容設計
    4. 機器選定

BGP 設計

3. 設計したnetwork に ”どうpacket を流すか?” routing policy を決める (通常service ごとにpolicy が変わる)
   1. どのような情報を流すか?
   2. 顧客にどのようなnetwork service を提供するか?

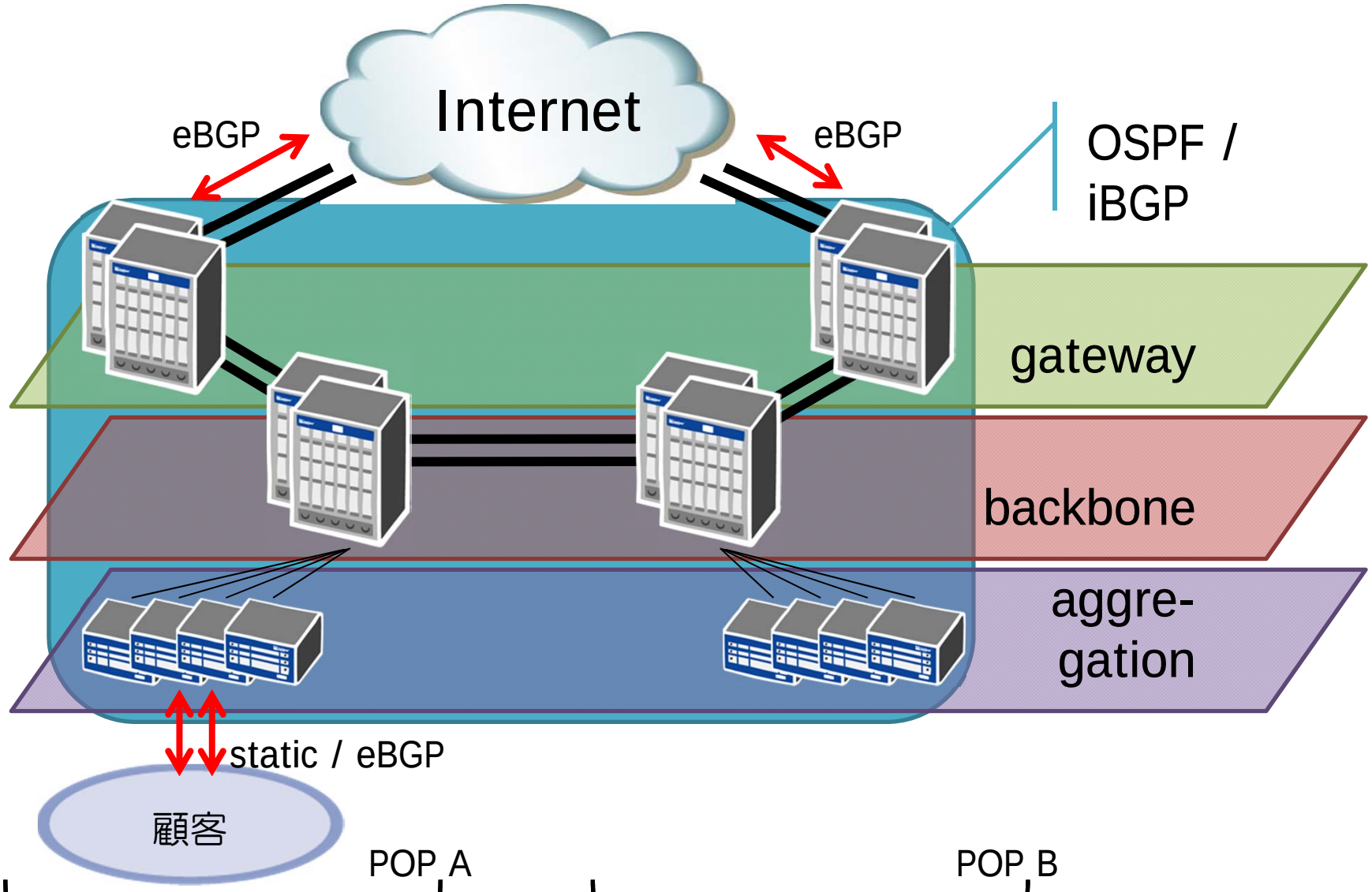
   好ましいpolicy が作れない場合，network を見直す必要があるかも

4. routing policy を満たすよう，routing (IGP/EGP) を設計する
   1. protocol は?
   2. Route Reflector (RR) or Confederation
   3. どの情報をどこに流す?

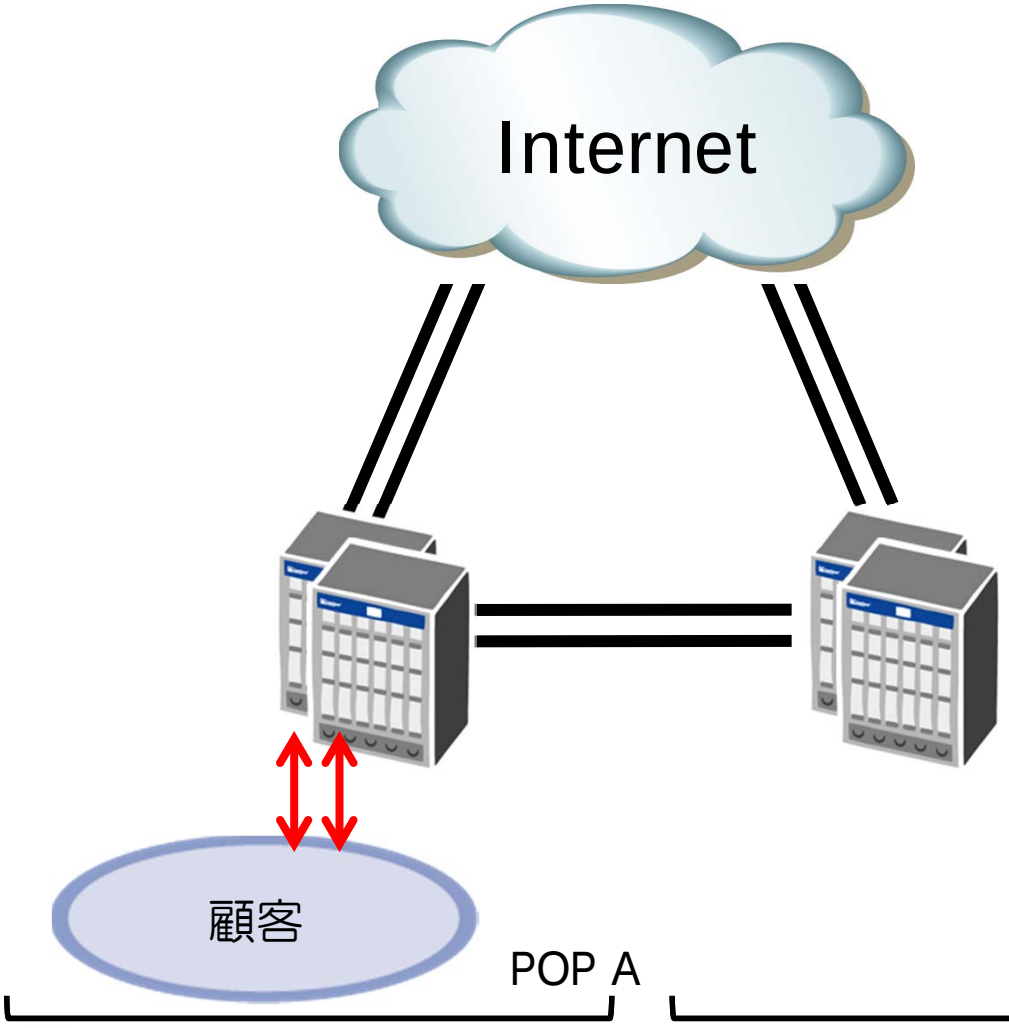
   うまく設計できない場合，routing policy を見直す必要があるかも

具体的な BGP の話に移る前に

教科書的な中規模Network



たぶん理想的にはこう



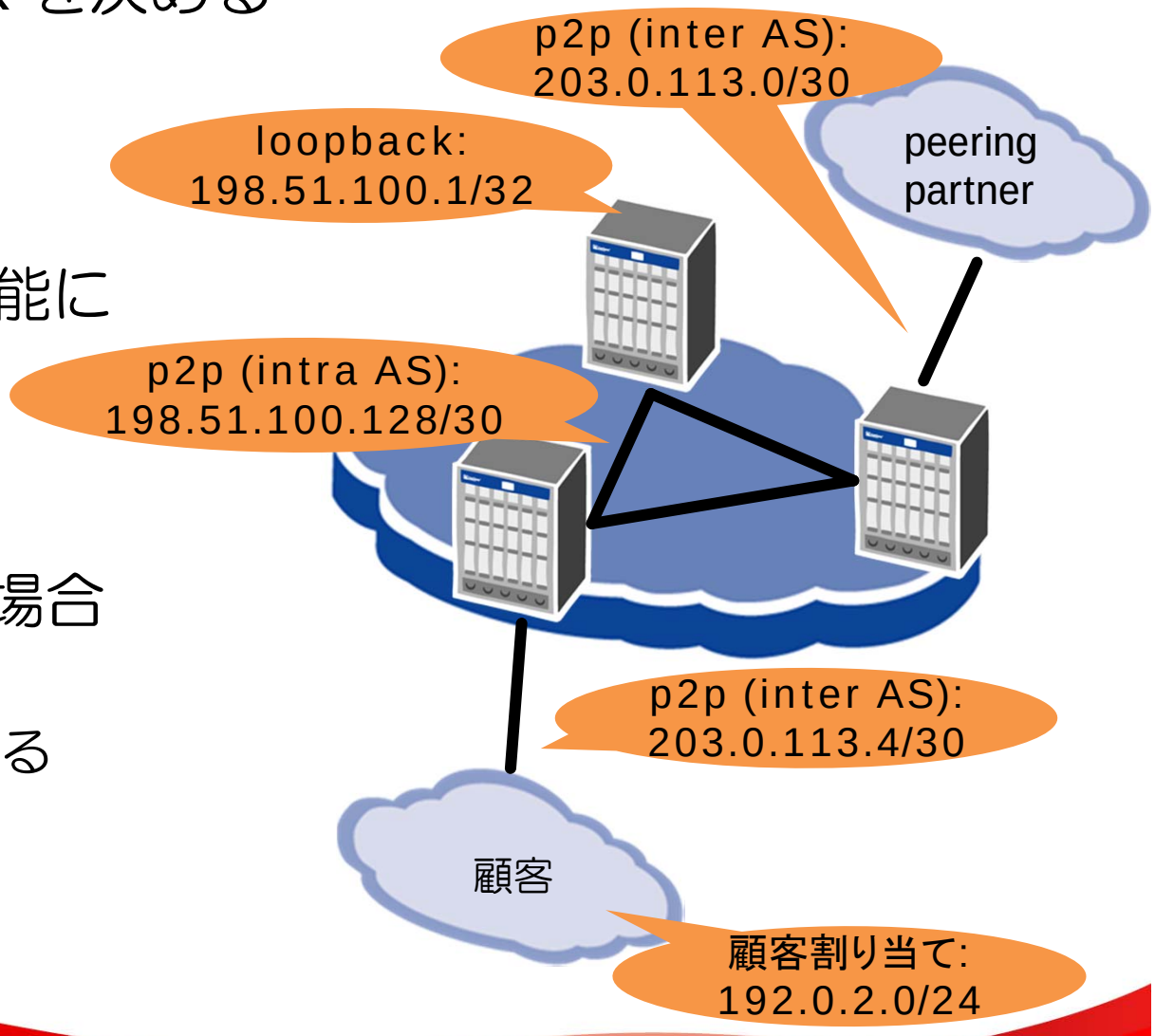
- router の集約による統計多重効果
- Virtual Chassis などによる論理台数の削減
- シンプルな運用

BGP 設計

Addressing

BGP 設計 / Addressing

- 目的に応じてblock を決める
  - loopback, p2p
  - 顧客割り当て
- できる限り集約可能に
  - security
  - シンプルさ
- 大規模network の場合
  - 拠点 / 地域ごとに address blockを分ける




シンプルに
わかりやすく
http://www.flickr.com/photos/techsavvyed/5926978939/

BGP 設計 / アドレッシング

- 例　　(最低限これだけ分けていれば十分)

type	用途	prefix 長	routing (外部へ…)
infra 用	loopback	ipv4 / 32, ipv6 / 128	広告しない (*2)
infra / AS 境界用	p2p	ipv4 / 30, ipv6 / 126 (*1)	広告しない (*2)
顧客用	割り当て Server SW	ipv4 / 28~20, ipv6 / 48 ipv4 / 24, ipv6 / 64	BGP 顧客のみ広告する (*3)

(*1) /31, /127 も使えるかも (実装依存)

[RFC3021] Using 31-Bit Prefixes on IPv4 Point-to-Point Links

[RFC6164] Using 127-Bit IPv6 Prefixes on Inter-Router Links

(*2) supernet として広告する.

security のため広告しない，というアイデアもあるが

外部からの到達性確認のため広告することを推奨.

(*3) static 顧客，Server 用SW network はsupernet のみ広告.

BGP 設計 / Addressing

- loopback address へのssh はcpu の手前でpacket filter を通す
  - network edge は多すぎて設定し忘れる

```
interfaces {
    lo0 {
        unit 0 {
            family inet {
                filter {
                    input ssh_filter;
                }
            }
        }
    }
}

firewall {
    filter ssh_filter { ... }
}
```

Juniper の例

```
policy-map ssh_filter
  class ...
  police ...

control-plane
 service-policy input ssh_filter
```

! Cisco 6500 の例

  - loopback address が増えすぎると，ACL のメンテナンスが困難になる → 用途別にできる限り集約

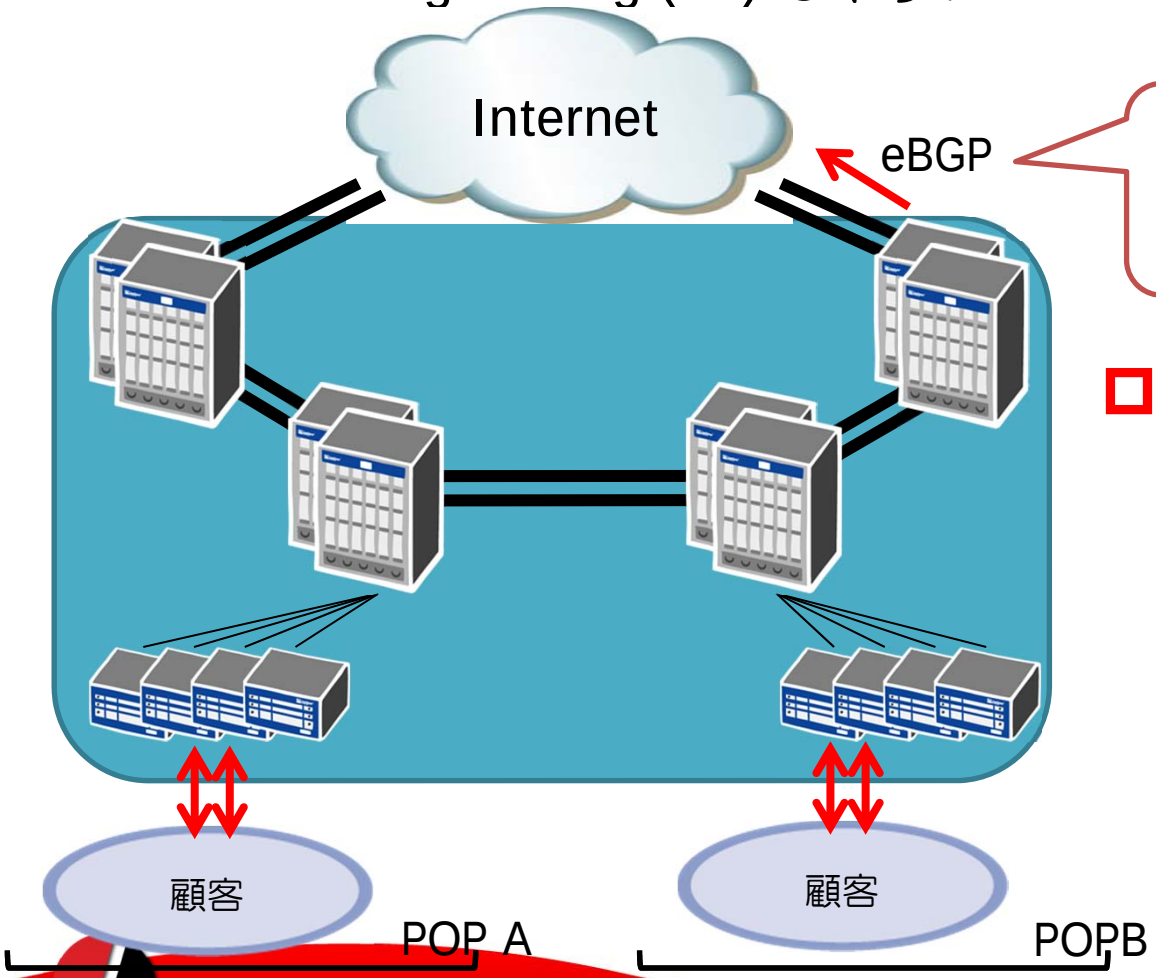
BGP 設計 / Addressing

- p2p block もなるべく集約可能に
  - ~~経路が増えるとrouter への負荷が．．．~~
  - シンプルな設計は，理解しやすい
    - address を見て，どのへんのloopback, p2p address か推測できるだけで運用スピードが上がる

```
 1  210.173.162.117  0.466 ms  0.408 ms  0.358 ms
 2  210.173.162.110  0.480 ms  0.438 ms  0.448 ms
 3  202.232.13.73  0.501 ms  0.500 ms  0.498 ms
 4  58.138.98.217  1.265 ms  0.969 ms  1.106 ms
 5  58.138.82.233  1.057 ms 58.138.80.245  1.219 ms 58.138.80.241  25.255 ms
 6  206.132.169.121  102.497 ms 216.98.96.62  99.842 ms 206.132.169.109  101.315 ms
 7  206.132.169.82  114.063 ms 206.132.169.2  110.394 ms  100.695 ms
 8  198.172.90.101  116.817 ms  105.842 ms 128.241.219.17  106.500 ms
 9  129.250.4.9  115.313 ms 129.250.5.54  96.205 ms  107.022 ms
10  129.250.4.24  152.412 ms 129.250.2.169  148.279 ms  142.641 ms
11  129.250.2.154  142.619 ms 129.250.2.174  139.038 ms 129.250.3.67  145.834 ms
12  129.250.17.38  148.640 ms  147.068 ms 129.250.17.42  154.900 ms
```

BGP 設計 / Addressing

- 顧客割り当てblock もなるべく集約可能に
  - traffic engineering (TE) しやすい



- POPごとにprefix を分けていないとコントロールできない
- 実際はなかなか難しい. . .

BGPの
おさらい

IGP との関係

- BGP で解決できるのは，”Internet 上のある目的地に達するために 自AS 内のどの出口から出ればいいか?” のみ
- “その出口にはどうやったら到達できるか?” はIGP 頼み

- IGP の主な用途は，BGP のprotocol nexthop を解決すること
  - むやみにBGP 経路をIGP に注入しないほうがいい

A Border Gateway Protocol 4 (BGP-4)

- RFC1654 (July 1994)
- RFC1771 (March 1995)
- RFC4271 (January 2006)
- もちろん たくさん拡張されている
  - RFC1997 BGP Communities Attribute
  - RFC2385 Protection of BGP Sessions via the TCP MD5 Signature
  - RFC3065 AS Confederations for BGP
  - RFC4451 BGP MED Considerations
  - RFC4456 BGP Route Reflection
  - RFC4360 BGP Extended Communities Attribute
  - RFC5004 Avoid BGP Best Path Transitions from One External to Another
  - RFC5668 4-Octet AS Specific BGP Extended Community
  - ...

BGP の経路選択

6.
MED はremote AS が
同じ場合のみ評価される

1. (最も高い WEIGHT を持つパスが優先されます. - 一部メーカーのみ)
2. **最も高い LOCAL_PREF を持つパスが優先されます.**
3. network または aggregate BGP サブコマンドによって, あるいは IGP からの再配布を通じて, ローカルで発信されたパスが優先されます.
4. **最短の AS_PATH を持つパスが優先されます.**
5. 最小のオリジン タイプを持つパスが優先されます.
6. **最小の Multi-Exit Discriminator (MED) を持つパスが優先されます.**
7. iBGP パスよりも eBGP パスの方が優先されます.
8. **BGP ネクストホップへの最小の IGP メトリックを持つパスが優先されます.**
9. 両方のパスが外部のときは, 先に受信したパス (最も古いパス) が優先されます.
10. 最小のルータ ID を持つ BGP ルータから送られたルートが優先されます.
11. 発信元 ID またはルータ ID が複数のパスで同じ場合は, 最小のクラスタ リスト長を持つパスが優先されます.
12. 最小の隣接ルータ アドレスから送られたパスが優先されます.

http://www.cisco.com/cisco/web/support/JP/100/1008/1008556_25-j.html

BGP 設計

eBGP Policy

/ 設計

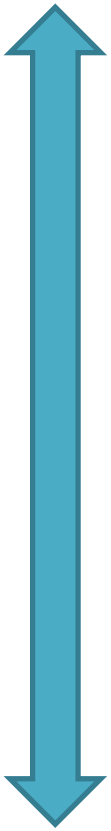
eBGP Policy を考えるポイント

- どんな経路を
  - どんなeBGP session からもらうか? (もらわないか?)
    - その際の優先度は?
  - どんなeBGP session へ広告するか? (広告しないか?)
    - その際の優先度は?
- 経路の各path attribute は誰のものか?

  - full route を持てないrouter はある?
    - ✓default route を利用: よく考えないと危険
      簡単にrouting loop が起きる
  - BGP ではなく，別のprotocol (OSPF など) に redistribute しないといけない，とかある?

経路の種別

- 顧客の経路
  - BGP 顧客
  - static 顧客 (実際はPA に集約)
- 自ASの経路
  - PA/PI 経路
- peering partner の経路
- transit provider の経路
- 不要な経路



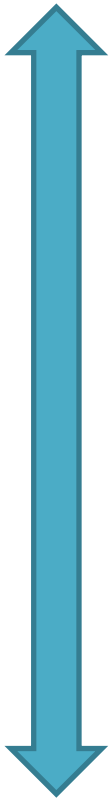
優先度: 高

優先度: 低

ビジネス上の観点から，基本的には上記の順に優先する (優先 = なるべくその経路に従って packet を流したい / 流してほしい)

eBGP セッションの種別

- 顧客
- peer
  - paid peer (収益を得ている)
  - private peer
  - public peer (IX)
  - paid peer (費用を払っている)
- transit



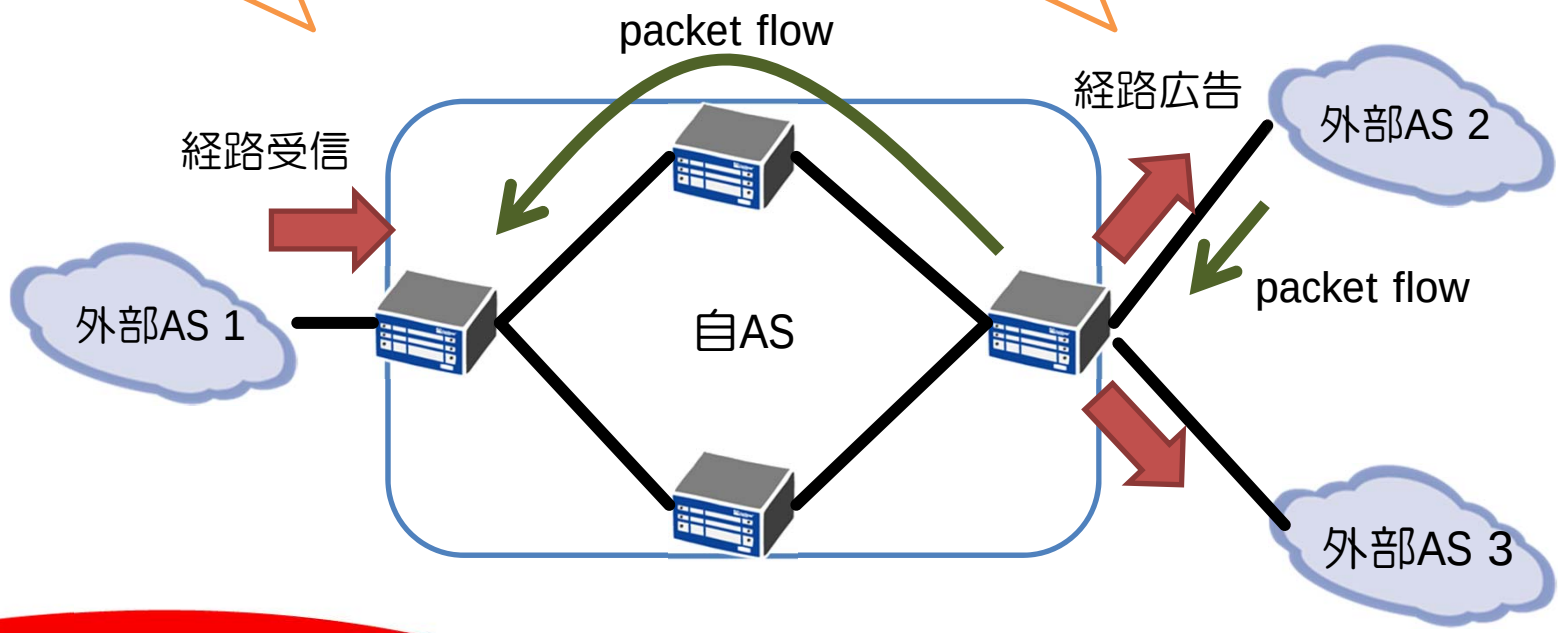
同様に，基本的には上記の順に優先する (優先 = なるべくその接続/回線にpacket を流したい)

eBGP Policy の基本

■受信 / 広告Policy が何に影響するか?

受信Policy
自AS 網内でどのように
routing させるか

広告Policy
自AS 網外でどのように
routing させるか



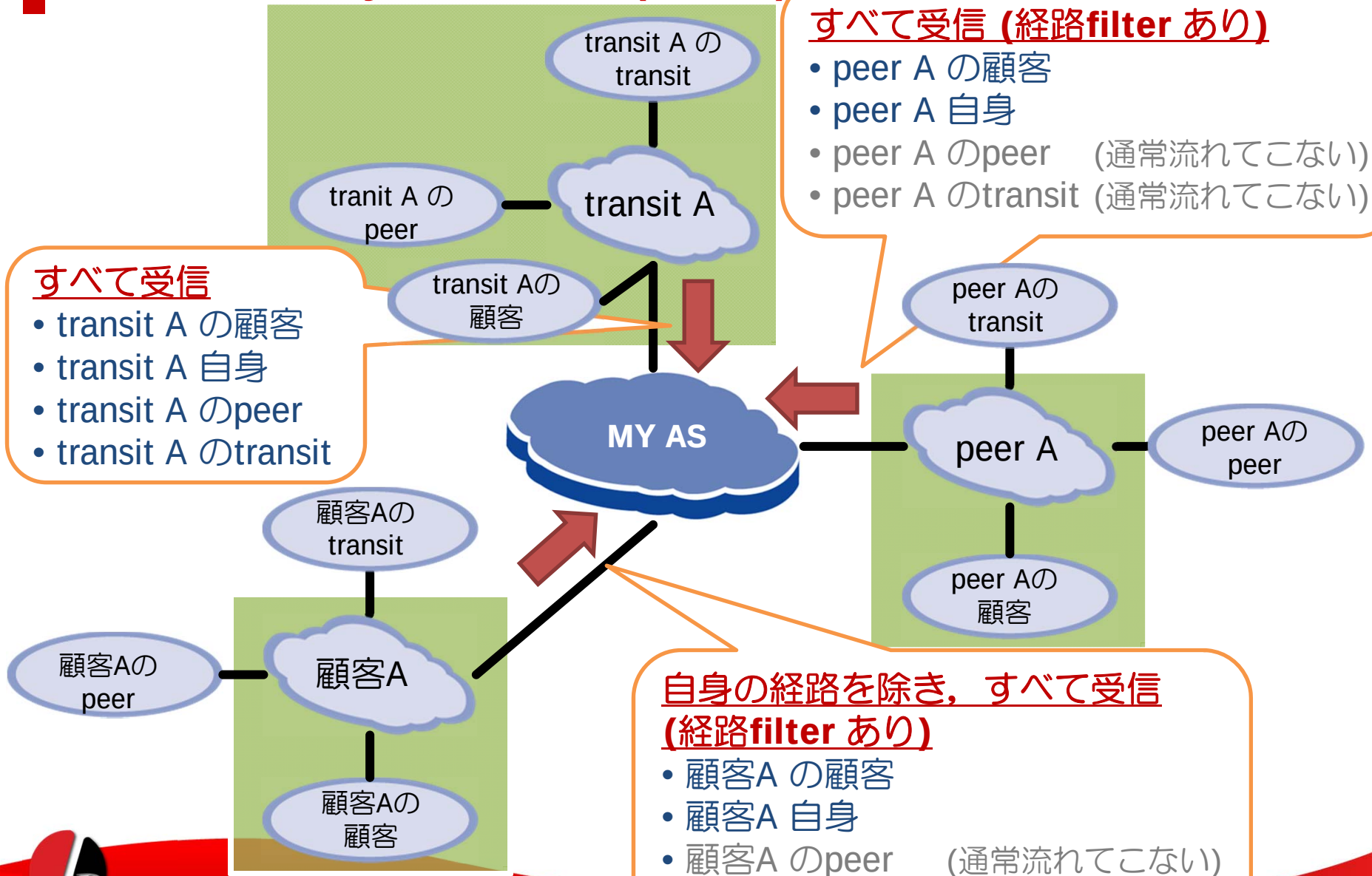
BGP 設計

eBGP Policy

経路受信

eBGP Policy の基本 (受信)

bogon filter はすべてに必要



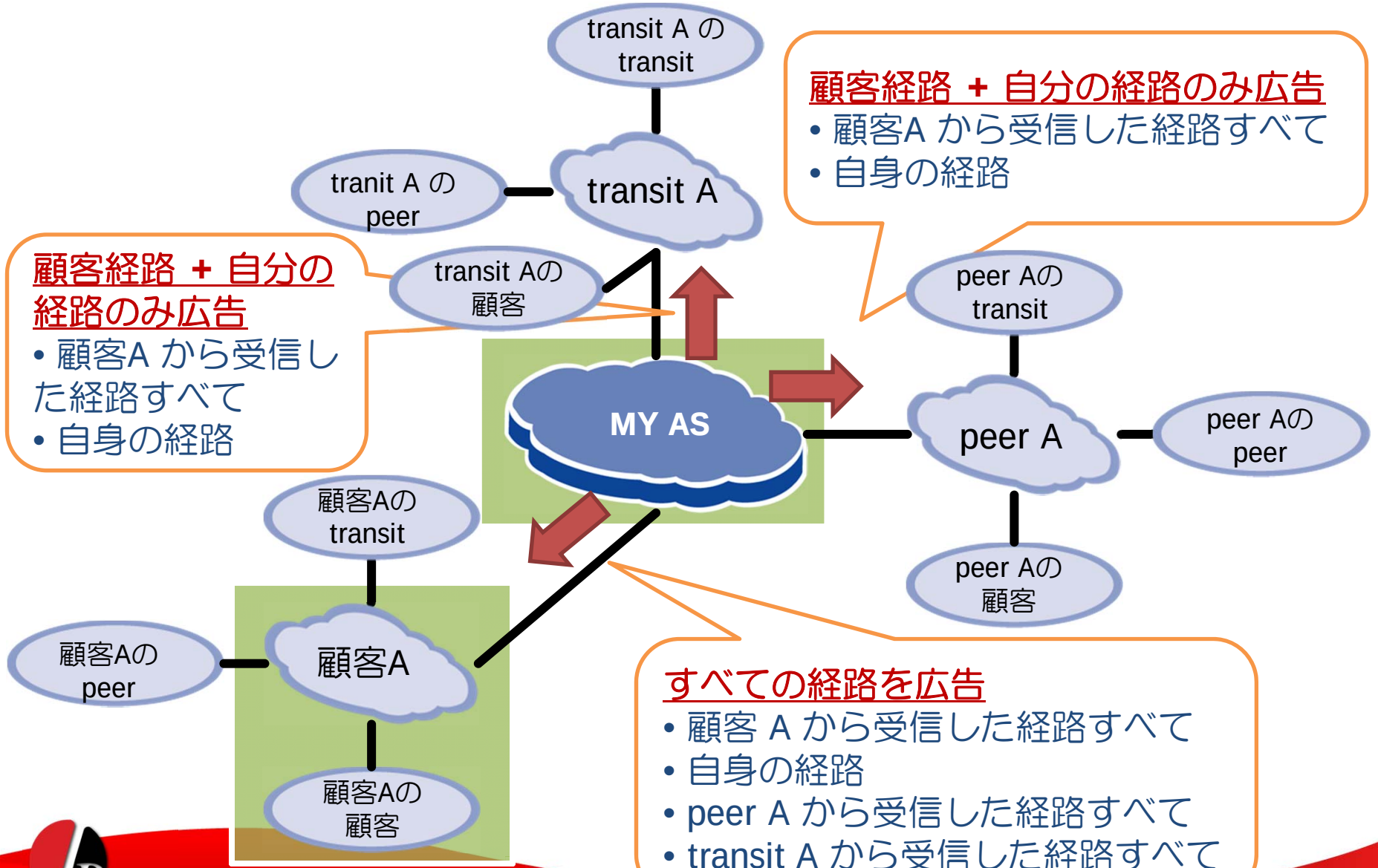
すべて受信 (経路filter あり)
- peer A の顧客
- peer A 自身
- peer A のpeer (通常流れてこない)
- peer A のtransit (通常流れてこない)

すべて受信
- transit A の顧客
- transit A 自身
- transit A のpeer
- transit A のtransit

自身の経路を除き，すべて受信 (経路filter あり)
- 顧客A の顧客
- 顧客A 自身
- 顧客A のpeer (通常流れてこない)
- 顧客A のtransit (通常流れてこない)

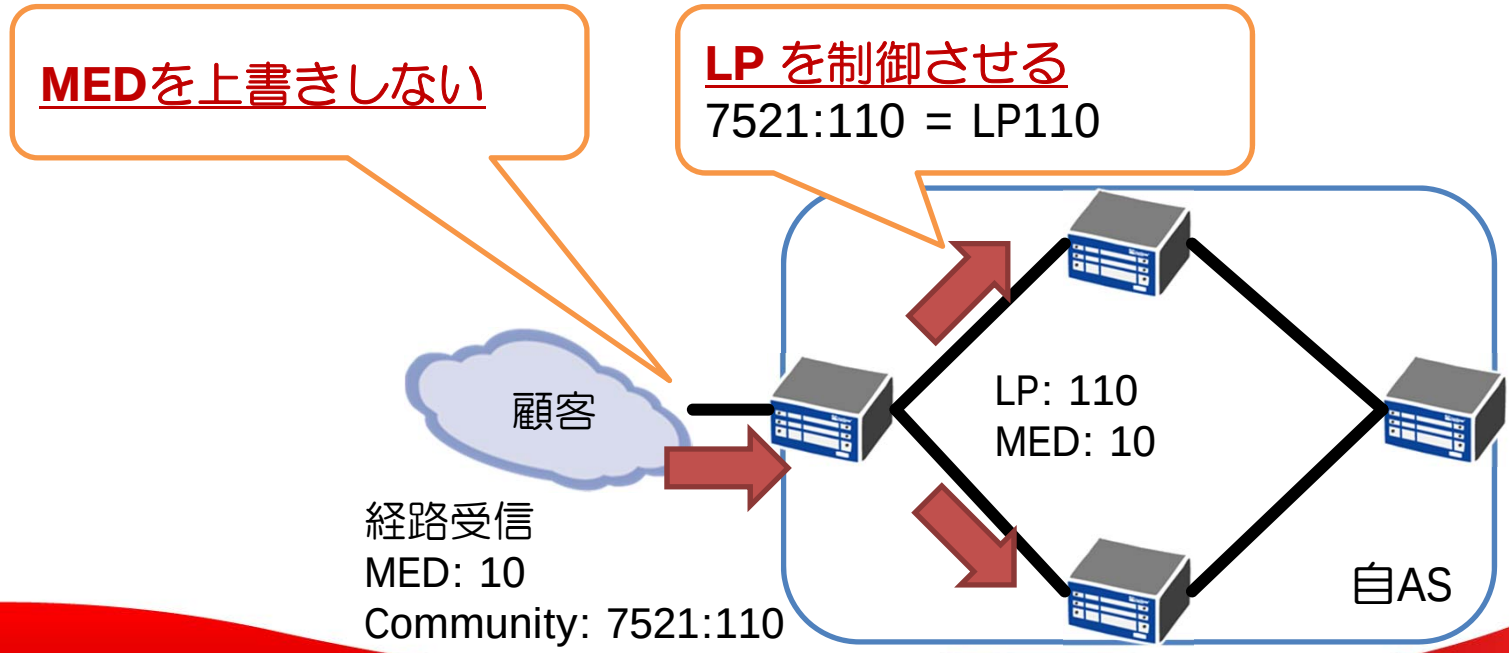
eBGP Policy の基本 (広告)

bogon filter はすべてに必要



経路の各path attribute は誰のもの?

- 以下のような機能を提供しているISP も
  http://onesc.net/communities/
  - 顧客経路のMED を上書きしない
  - 自AS 内でのLocal Preference (LP) を制御する
    BGP Community を顧客向けに提供する



eBGP Policy 設計例 (受信)

- 基本的に，受け取った経路は使う
  - 例えば peer から”peer のpeer 経路” を受信したとして，(相手の設定ミスの可能性大だが) 拒否する理由は特にない
  - 使いたくない理由があれば別 (優先度を下げる or 拒否する)
    - 輻輳しそう，など

優先度	経路種別	LP	MED
1	顧客	150~300 (*)	上書きしない
2	自AS	200	なし
3	peer	200	上書き (十分大きな値)
4	transit	120	上書き

(*) 200 をまたいで数段階 (default: 300)

- 幅に余裕をもって数値を設定
- LP のdefault 値が100 の実装が多いので，安全を期すならLP の値は>100

eBGP Policy 設計例 (受信) – 解説 –

顧客の明確な意図がない限り最優先.
常にRIB に保持

経路を指定して他の顧客やpeer に迂回させることができる

peer の200 と比較して，必ず次のAS_PATH で優先度が決定

顧客のTE選択肢を増やす

優先度	経路種別	LP	MED
1	顧客	150~300 (*)	上書きしない
2	自AS	200	なし
3	peer	200	上書き (十分大きな値)
4	transit	120	上書き

AS_PATH が同じなら
IGP ベースの
closest exit を
強制 + always-
compare-med
やpeer & 顧客
なAS 対策で
MED を高めに.
2^32 – 1 でも
いいかも

private / public peer で2段階あるISPもある

一応>100

AS_PATH が同じなら
IGP ベースのclosest exit
を強制

(*) 200 をまたいで数段階 (default: 300)

eBGP Policy 設計例 (受信) – 経路Filter –

- 顧客経路の優先度が非常に高いため，細かくチェックする必要がある
  - IRR ベース が理想的
    - 頻繁にメンテナンスできるのであれば手動でも問題ないかもしれない
  - filter 方法の候補
    - exact match なprefix filter
    - exact match なprefix filter & origin AS filter
  - bogon prefix, 自AS のprefix は経路filter で除外
  - BGP community は透過的に扱う

eBGP Policy 設計例 (受信) – 経路Filter –

- peer 経路も手放しでは危ない
  - 経路hijack の可能性
  - 細かいfilter を設定してしまうとメンテナンスされなくなるかもしれない
    - ”AS_PATHをメールで伝え合う” 仕組みが動いていた
    - やはり限界があるので，顧客経路同様IRR ベースがよい
- 一方，他の国では以下の組み合わせが多い
  - maximum-prefix (prefix-limit) filter
    - 実績ベース (昨日から20% 増えたらアウト，など)
    - peering db ベース
  - prefix length によるfilter
    - ipv4: le /24
    - ipv6: le /48 など

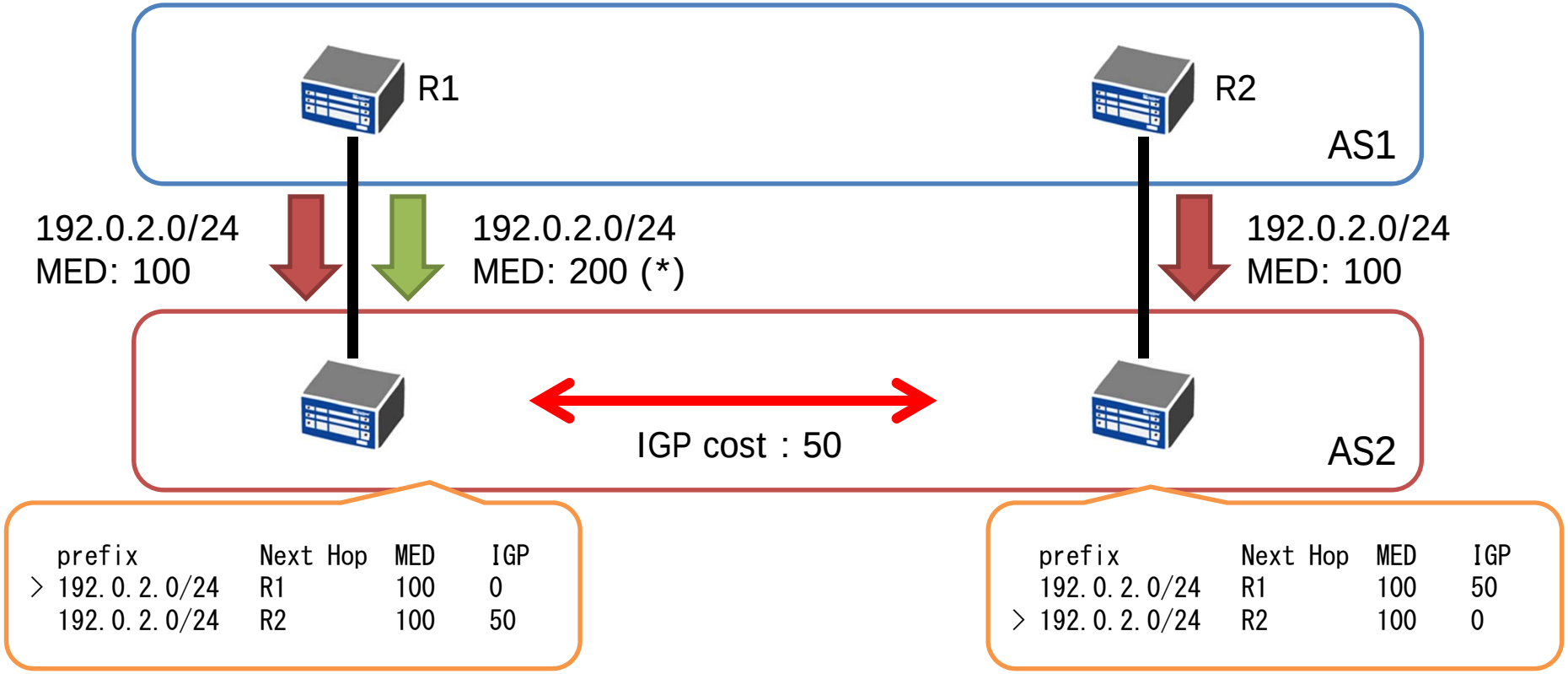
Maximum-Prefix (Prefix-Limit) Filter

- 注意: transit に設定すると危険な場合がある
  - network 上のevent により急にprefix 数が増えると，transit が全断するリスクがある
    - internet から遮断されることになる

経路Filter に関する参考情報

- JANOG Comment JC1000~1003 に大変丁寧にまとめられている
  - http://www.janog.gr.jp/doc/janog-comment/

eBGP Policy 設計例 (受信) – closest exit –



(*) MED が異なる経路でもclosest exit したい場合はMED を上書きする必要がある

eBGP Policy 設計例 (受信) – closest exit –

- IGP を用いる代わりに，MED を加算することでも一応 実現できる

Juniper:	metric add 500
Cisco:	set metric +500

× router 間でMED 加算して回る必要がある

△ こちらもMED の上書きが必要

(“壁” になっている 500 という数値が十分かどうかはpeer partner の広告policy 次第)

- 本来のMED の用途とは異なる

経路制御のオプション (受信)

- どの経路を優先するかについて
  - transit / peer 経路
    - LP の微調整
    - AS_PATH prepend
    - MED 微調整
    - passive IGP
    - 経路を止める
  - 顧客経路
    - 顧客からの依頼に基づく場合を除き，操作しない

経路制御のオプション (受信)

- LP / MED などの数値はなるべく”弱くなる” 方に制御する
  - ヘンにtraffic を吸い込むのを防ぐ
  - MED:　増やす
  - LP:　減らす　(多くの実装でdefault: 100)
- AS_PATH prepend: prepend された経路がRIB に残ってしまう可能性が出るので，なるべく避ける
  - transit を売っている場合など，全体として経路が遠く見えてしまうのは良くない
- passive IGP は経路ごとの制御ができない

経路制御のオプション (受信)

- LP / MED / AS_PATH 操作
  - なるべくメンテナンス頻度を下げる
    - peer AS
    - nexthop
    - origin AS
    - AS_PATH
    - prefix

    の順で操作

    (例えばprefix は時間が経つと変化する可能性が高い)

経路制御のオプション (受信)

- irregular なことは目立つように / 消しやすく

```
protocols {
    bgp {
        neighbor x.x.x.x {
            import [ regular irregular finalize ];
        }
    }
}

policy-options {
    policy-statement regular {
        from { ... }
        then {
            # 通常のpolicy
            next policy;
        }
    }

    #
    # remove me soon !!
    #
    policy-statement irregular {
        from { ... }
        then {
            # irregular なpolicy
            next policy;
        }
    }
```

Juniperの例

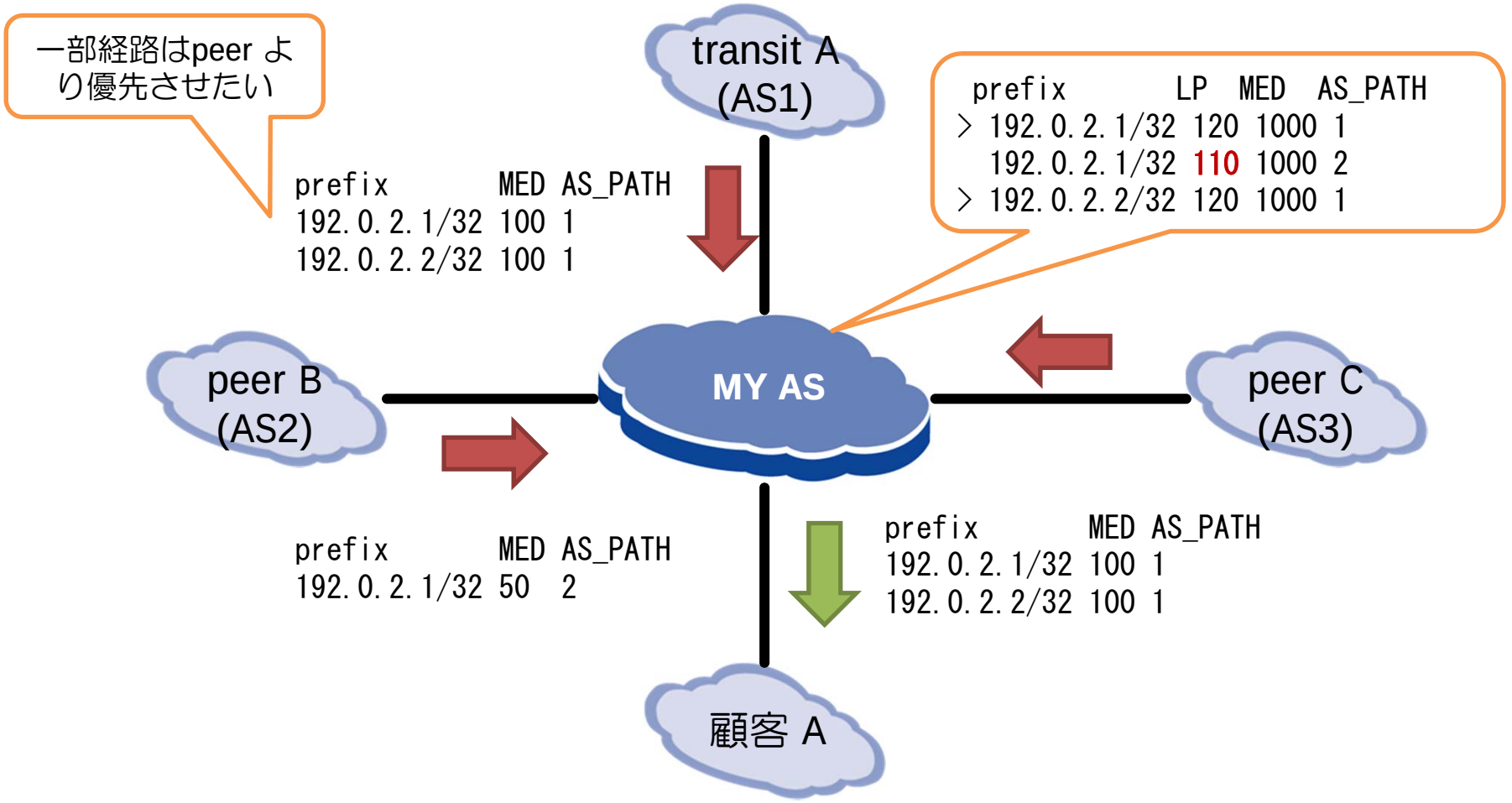
```
    policy-statement finalize {
        then accept;
    }
}
```

Juniperの例 (続き)

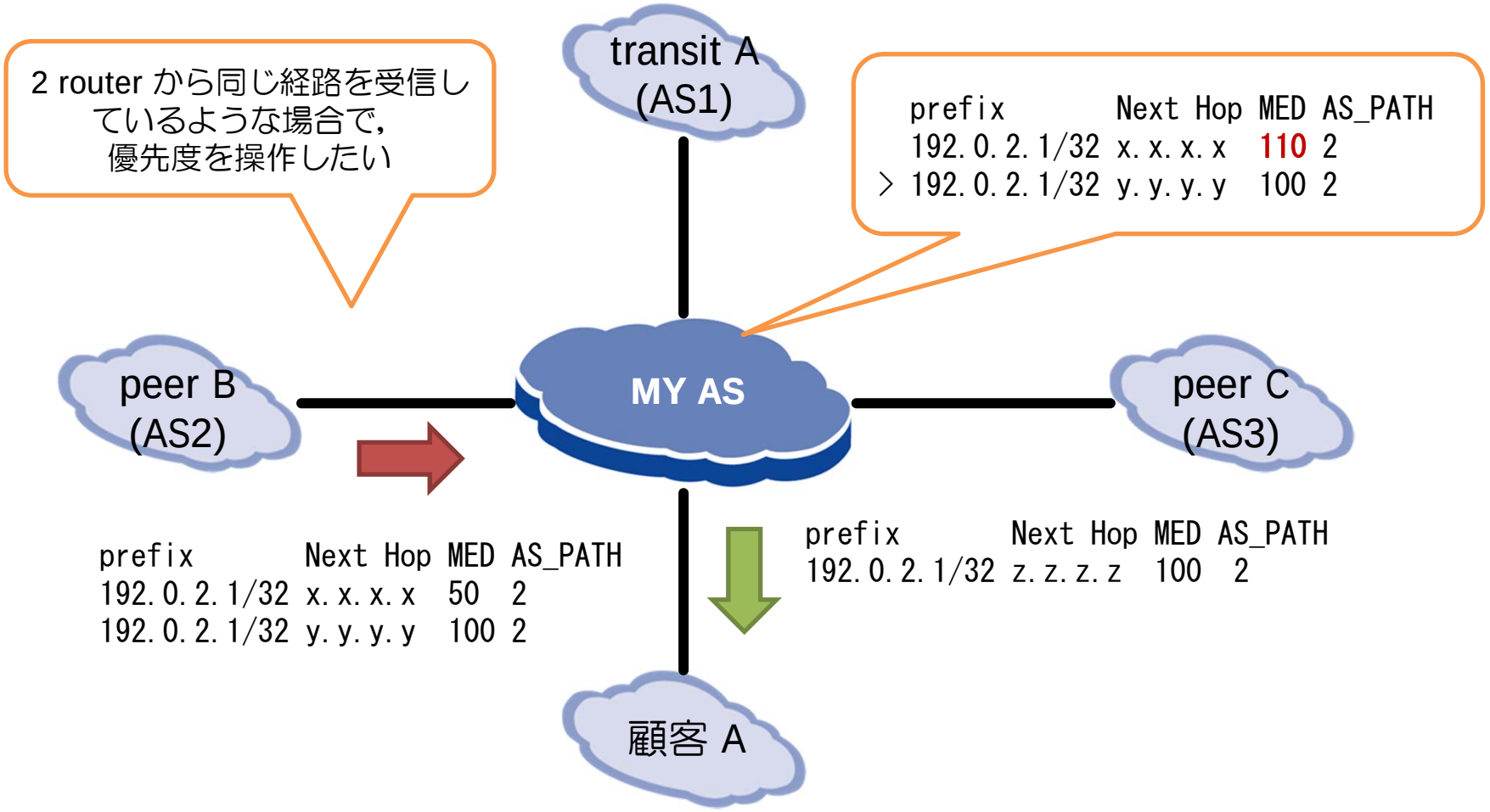
```
no route-map route-filter
route-map route-filter permit 10
  ! 通常のpolicy
route-map route-filter permit 20
  ! 通常のpolicy
route-map route-filter permit 30
  ! 通常のpolicy

!
! remove me soon !!
!
route-map route-filter permit 25
  ! irregular なpolicy
```

! Cisco の例

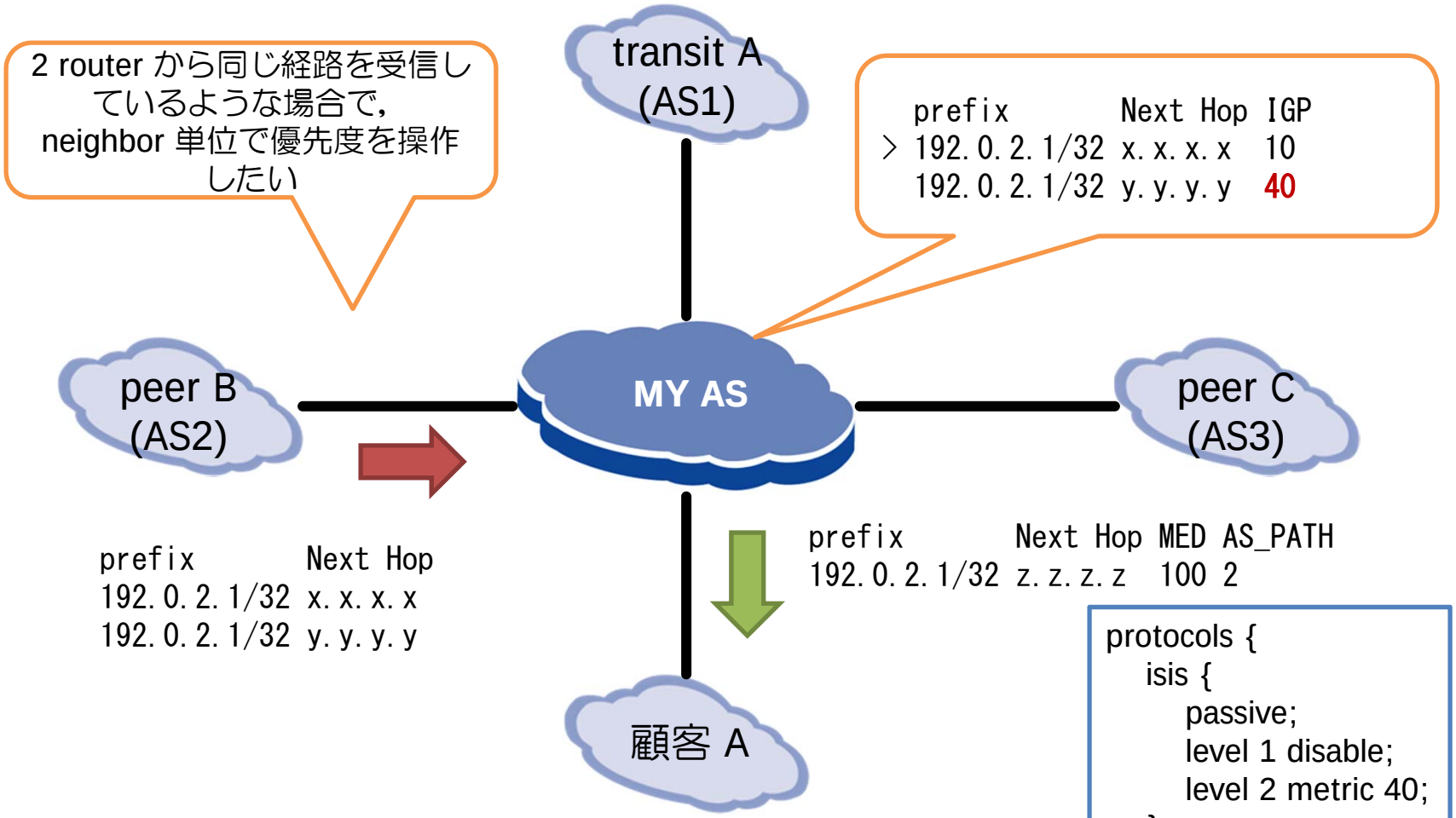
経路制御のオプション (受信) – LP 制御 –



経路制御のオプション (受信)
– MED 制御 –



経路制御のオプション (受信)
– IGP 制御 –



BGP 設計

eBGP Policy

経路広告

eBGP Policy 設計例 (広告)

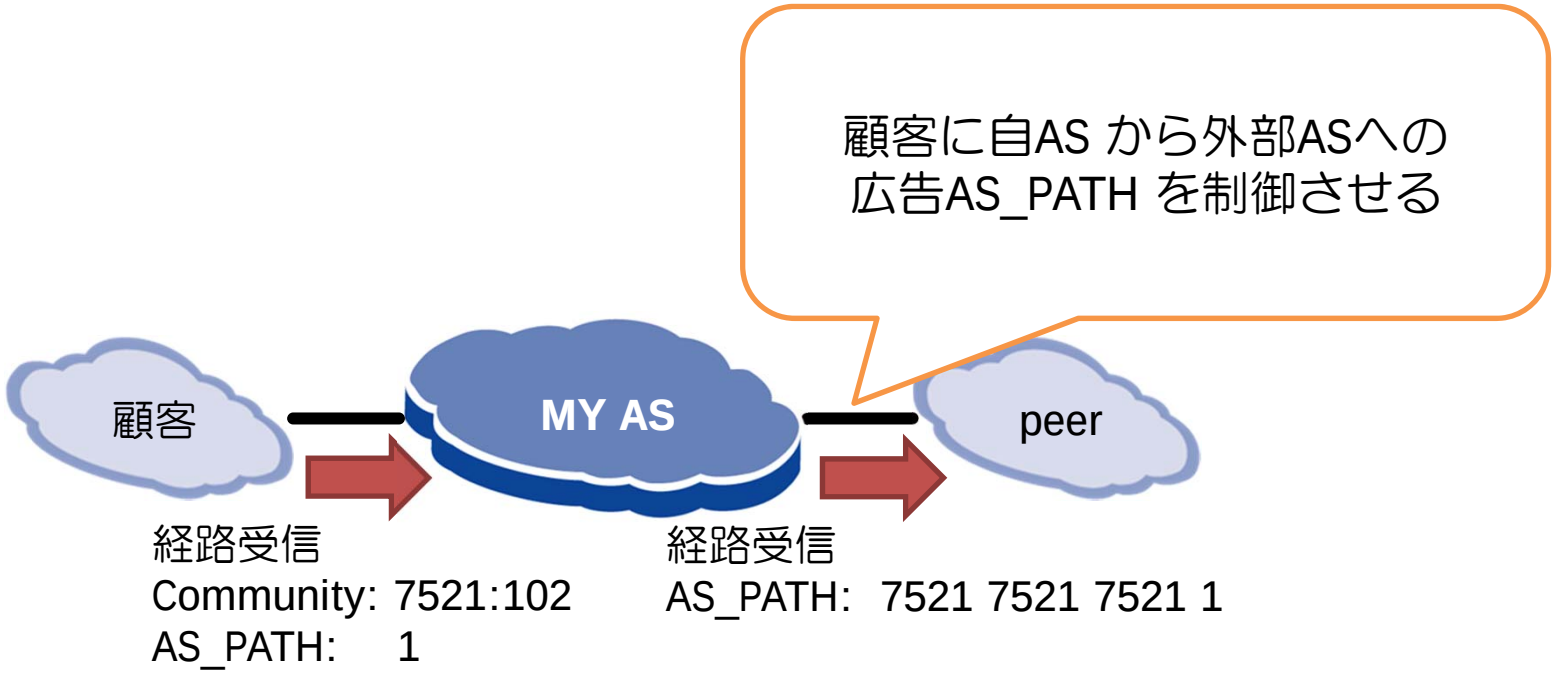
- 自AS が持つすべての経路(full route) を顧客に
- 顧客からのすべての経路+自AS の経路を
  - transit に
  - peer に

  広告
- 経路の種別による”マーク” をBGP Community として付与すると顧客は便利に使える

  http://www.us.ntt.net/support/policy/routing.cfm
  - 外部AS から，どの地域で受け取ったか?
  - 外部AS とは，transit か? peer か? 顧客か?

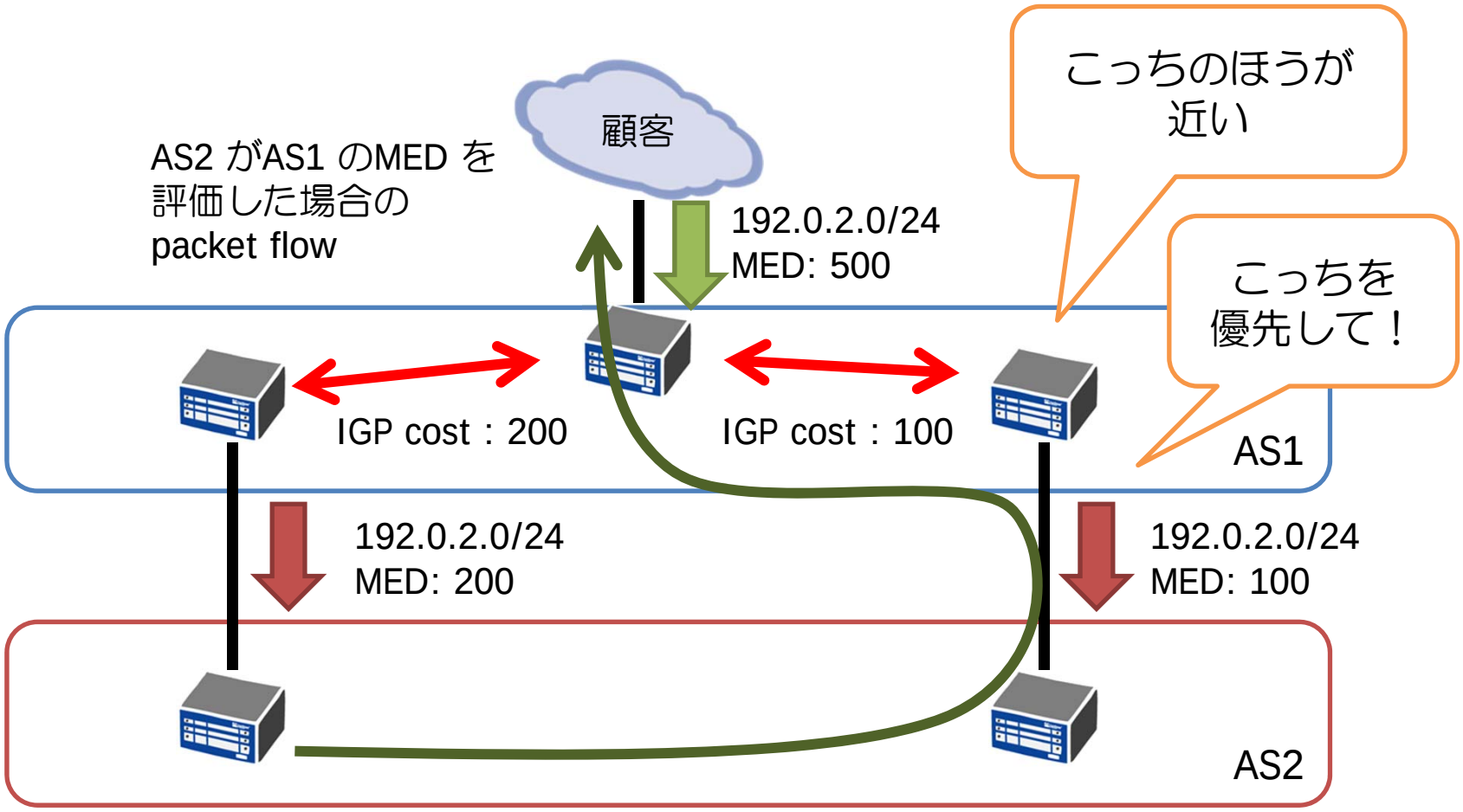
eBGP Policy 設計例 (広告)

- prepend community (例)



eBGP Policy 設計例 (広告)

- MED: 優先度をつけて広告する



eBGP Policy 設計例 (広告)

- metric-out igp が便利 (IGP cost をMED として利用)

```
protocols {
    bgp {
        group ebgp {
            metric-out igp;
            neighbor x.x.x.x { ... }
        }
    }
}
```

Juniper の例

```
router bgp xxxx
  ...
  neighbor y.y.y.y route-map ebgp
route-map ebgp
  ...
  set metric-type internal
```

! Cisco の例

- 注意: 外部AS が常にMED を評価してくれるとは限らない
  - ダメもとでも広告しておく価値あり
  - 評価してもらいたいなら，交渉

eBGP Policy 設計例 (広告) – 経路filter –

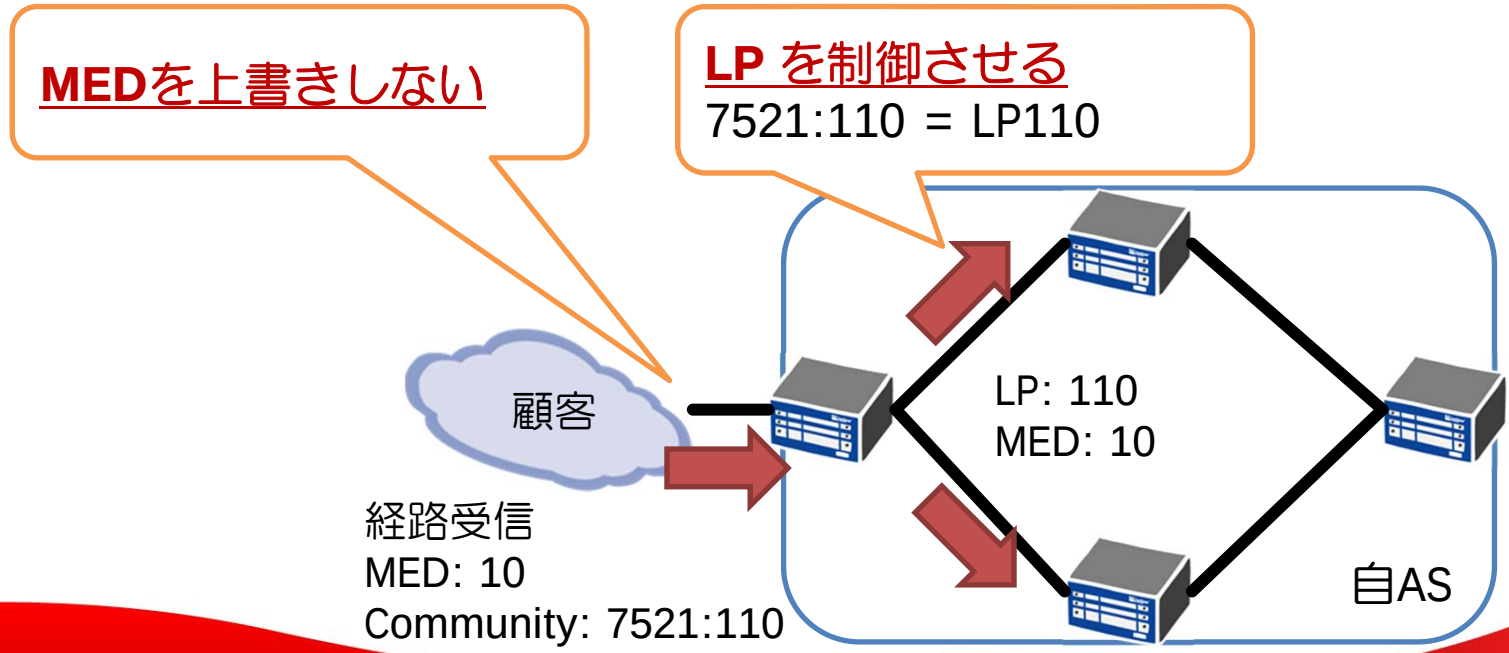
- 内部の細かい経路は広告しない
  - connected (direct) 経路はBGP にredistribute しないなど
    - するときにはno-export community を付けておく
- bogon その他，internet に流すべきでない経路が含まれないかを再確認
- remove-private しておく (private AS は削って広告)

経路制御のオプション (広告)

- 誰に対して制御するかについて
  - transit / peer
    - AS_PATH prepend
    - MED 微調整
    - BGP community (一部transit のみ)
      - ✓transit AS 内のLP を制御
      - ✓transit AS → 他AS 経路広告を制御 (広告しない / prepend)
    - 経路広告を止める
  - 顧客
    - 顧客からの依頼に基づく場合を除き，操作しない

経路の各path attribute は誰のもの 再掲

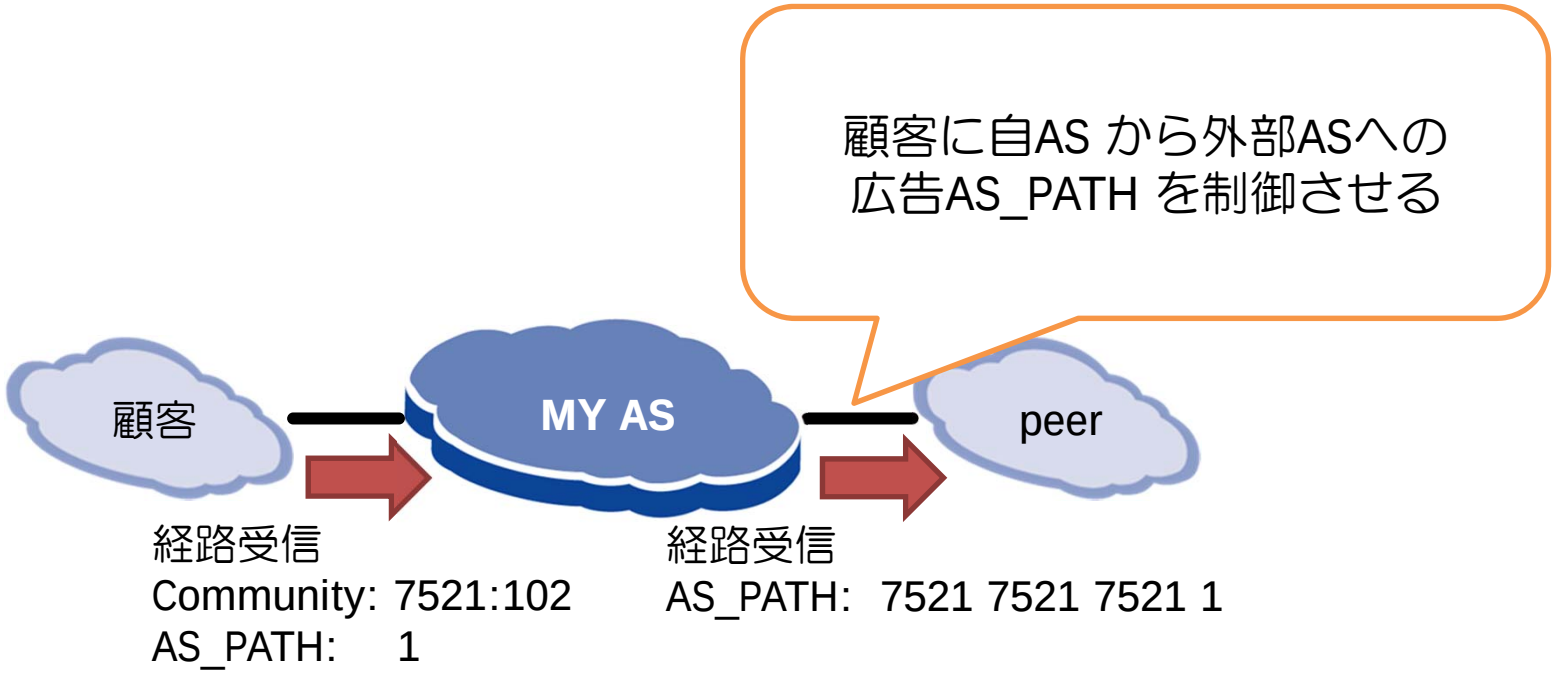
- 以下のような機能を提供しているISP も
  http://onesc.net/communities/
  - 顧客経路のMED を上書きしない
  - 自AS 内でのLocal Preference (LP) を制御する BGP Community を顧客向けに提供する



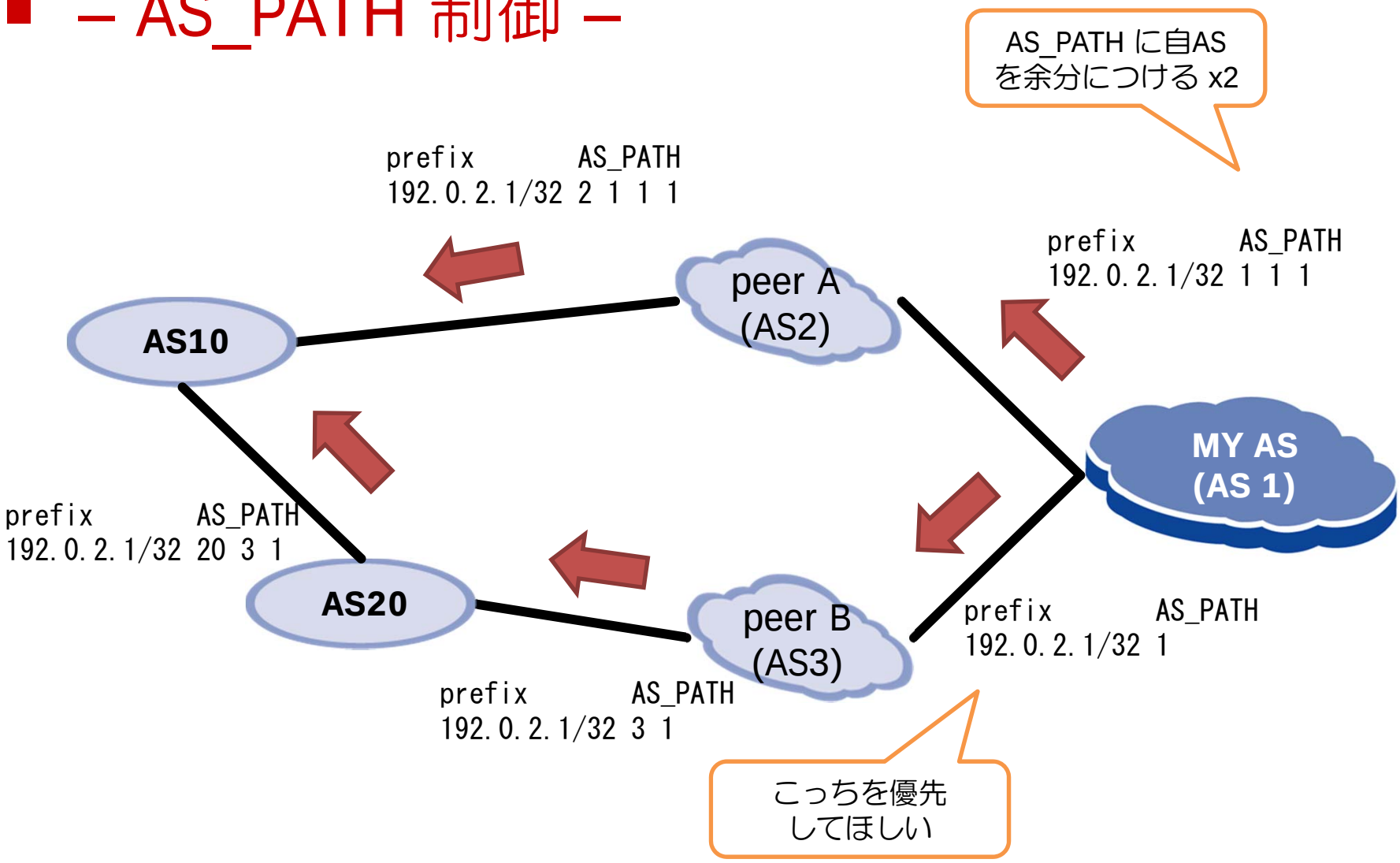
eBGP Policy 設計例 (広告)

再掲

- prepend community (例)



経路制御のオプション (広告)
– AS_PATH 制御 –



経路制御のオプション (広告)

- 広告経路の制御が効かない場合も多々ある
  - 顧客 / peering partner / transit 提供者にも彼らなりの policy があるので，やむを得ない
  - 接続しているtransit / peer のrouting 設計を理解することがすごく重要
    - MED は効くか?
    - AS_PATH prepend 可能か?
    - best path はどのattribute で決まっているか?
    - 制御系BGP community はあるか?
    - そのような設計になっているのはなぜか?

    - 直接答えてもらえればラッキー
    - 推測の蓄積でも十分役立つ

経路制御のオプション (広告)

以下のオプションは
高い確率で効果が期待できる

- BGP Community

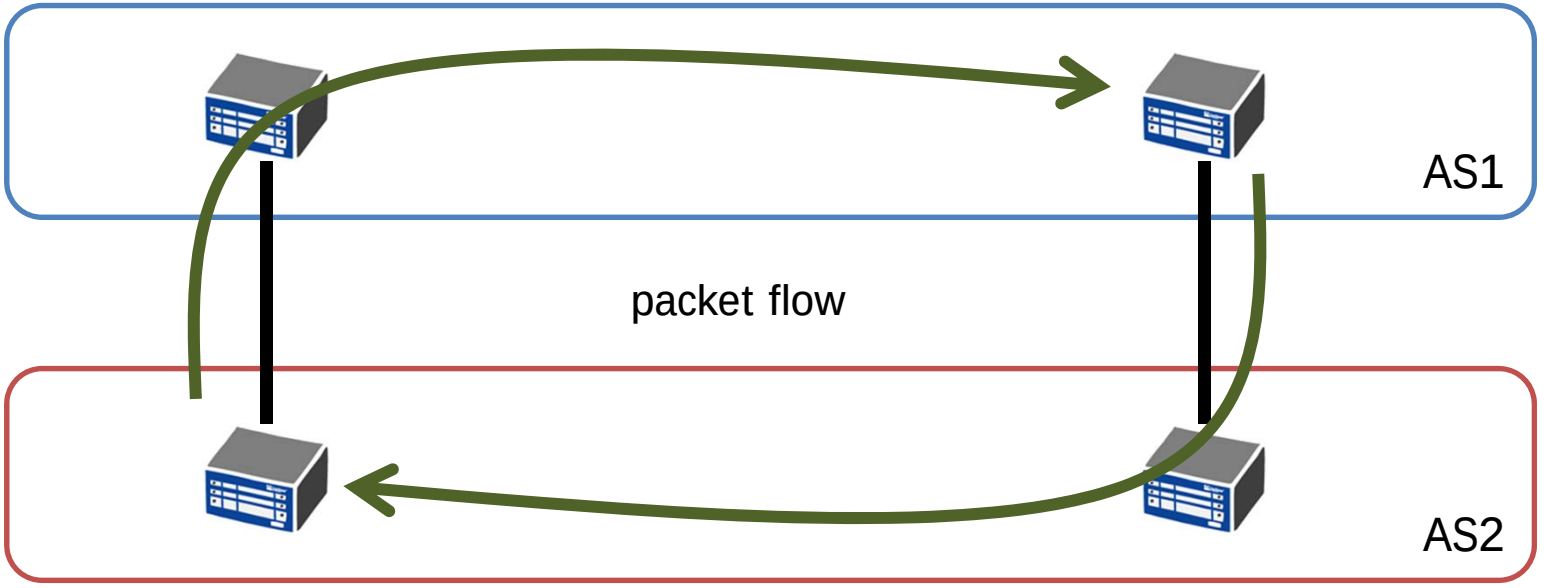
- 経路広告を止める

奥の手!

  - 多くの場合，冗長性を損なう
  - more specific な経路にする (/20 = 2x /21)
    - 管理が煩雑になる


シンプルに
わかりやすく
http://www.flickr.com/photos/techsavvyed/5926978939/

それぞれのpolicy で設計するので



- よく非対称なpacket flow になりがち
  - ping の往復のpath が違うことを意識しないといけない
  - trouble shoot しにくい

お互い誠意を持って運用しよう

http://www.flickr.com/photos/waldec/2470541755/

beer and peer !

http://www.flickr.com/photos/mckln/3449304905

BGP 設計

iBGP Policy

/ 設計

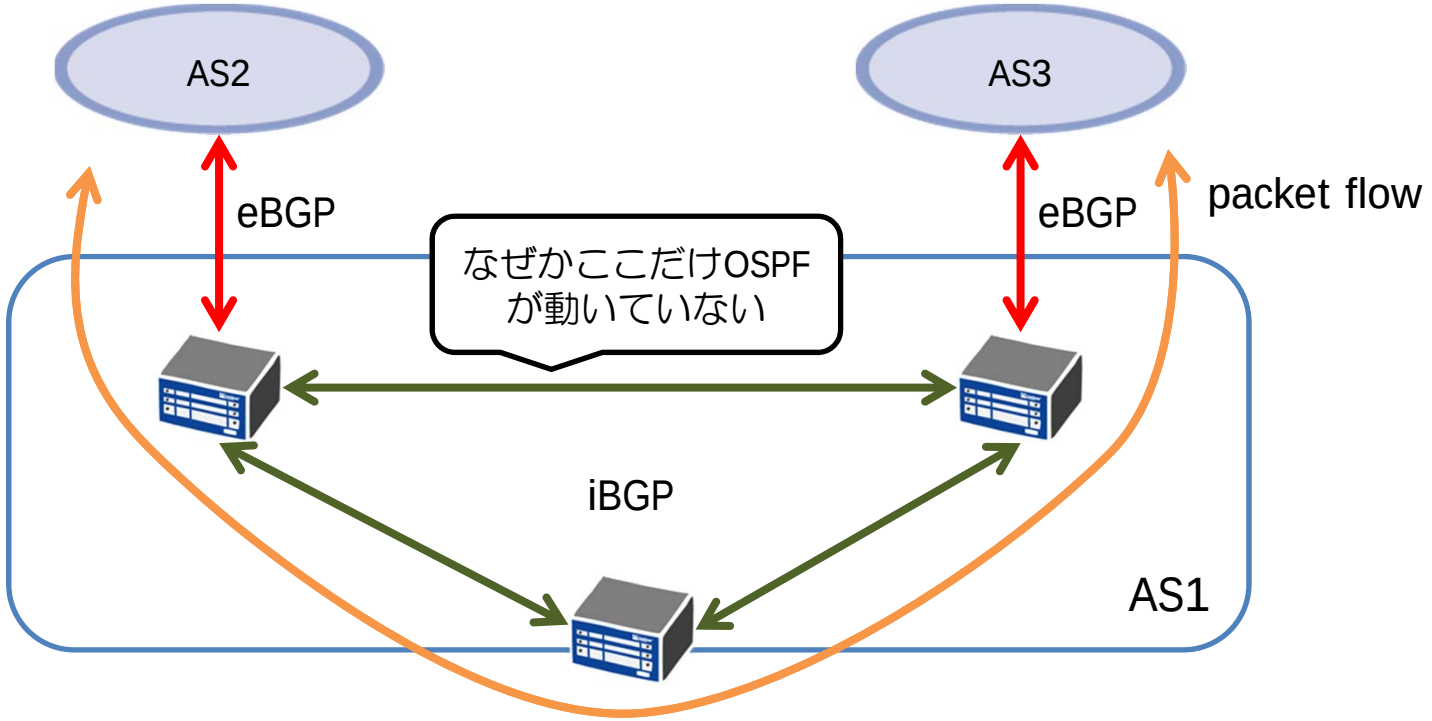
iBGP Policy を考えるポイント

- network の中でどこまでBGP を動作させるか?
  - なるべくDFZ (Default Free Zone) が好ましい
- iBGP full-mesh のスケールは?

  - BGP を動作させないrouter やdefault route が存在する場合はrouting loop に注意
  - iBGP full-mesh がスケールしなくなったら
    - Route Reflector or BGP Confederation

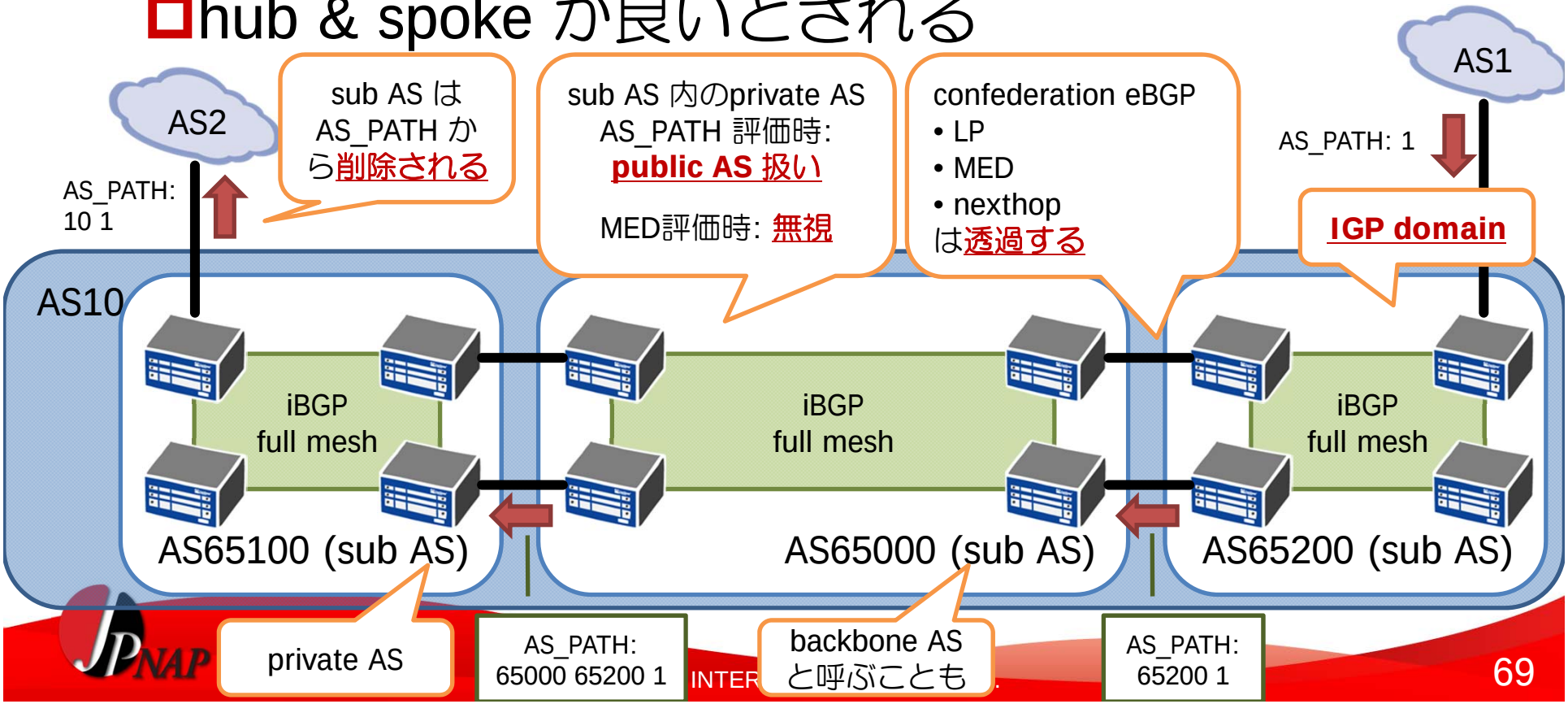
論理Topology 設計

- 物理 / 論理topology を揃えておく
  - network をシンプルに保つため



BGP Confederation

- Route Reflector (RR) 同様，BGP スケーラビリティを向上させる技術
  - 複数のsub AS に分割し，sub AS 間をeBGP 接続
  - hub & spoke が良いとされる



BGP Confederation

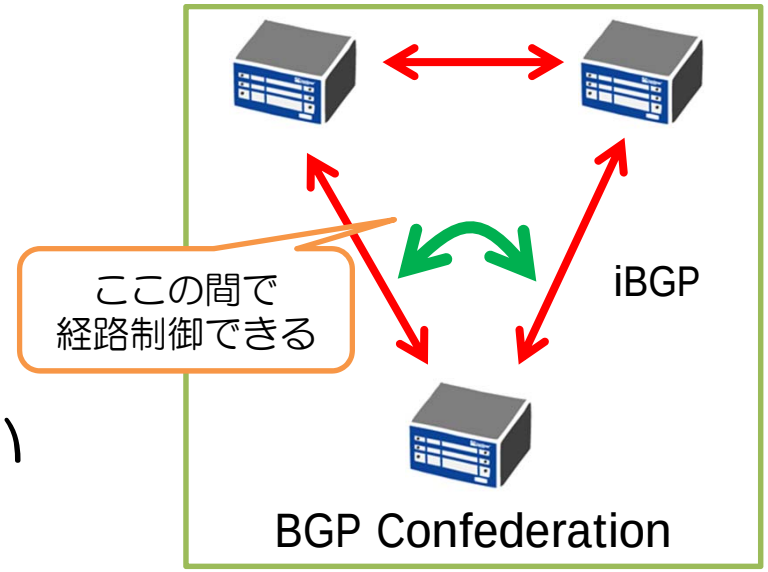
- 利点
  - IGP domain を分割できる
    - 大規模になると不安定になりがちなIGP への対策
  - sub AS 単位でBGP policy を分けられる
    - サービス別/国別/エリア別などで運営母体を分けるときにぴったりハマる
    - M&A などにより統合したnetwork を，1AS に移行するステップとして
- 欠点
  - sub AS の規模が大きくなるとiBGP session 数 / それに伴う経路数が負荷になる
    - (best ではない) 経路が多い → route convergence time が長い

sub AS が大きくなってきたら，RR 導入もOK

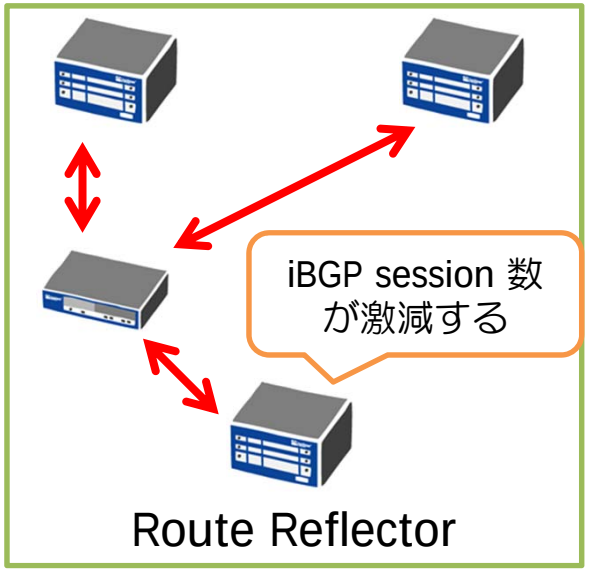
(BGP Confederation とRR の併用)

BGP Confederation vs. RR

- BGP Confederation
  - BGP policy を分けたい
  - IGP domain を分けたい
  - iBGP による細かい制御をしたい



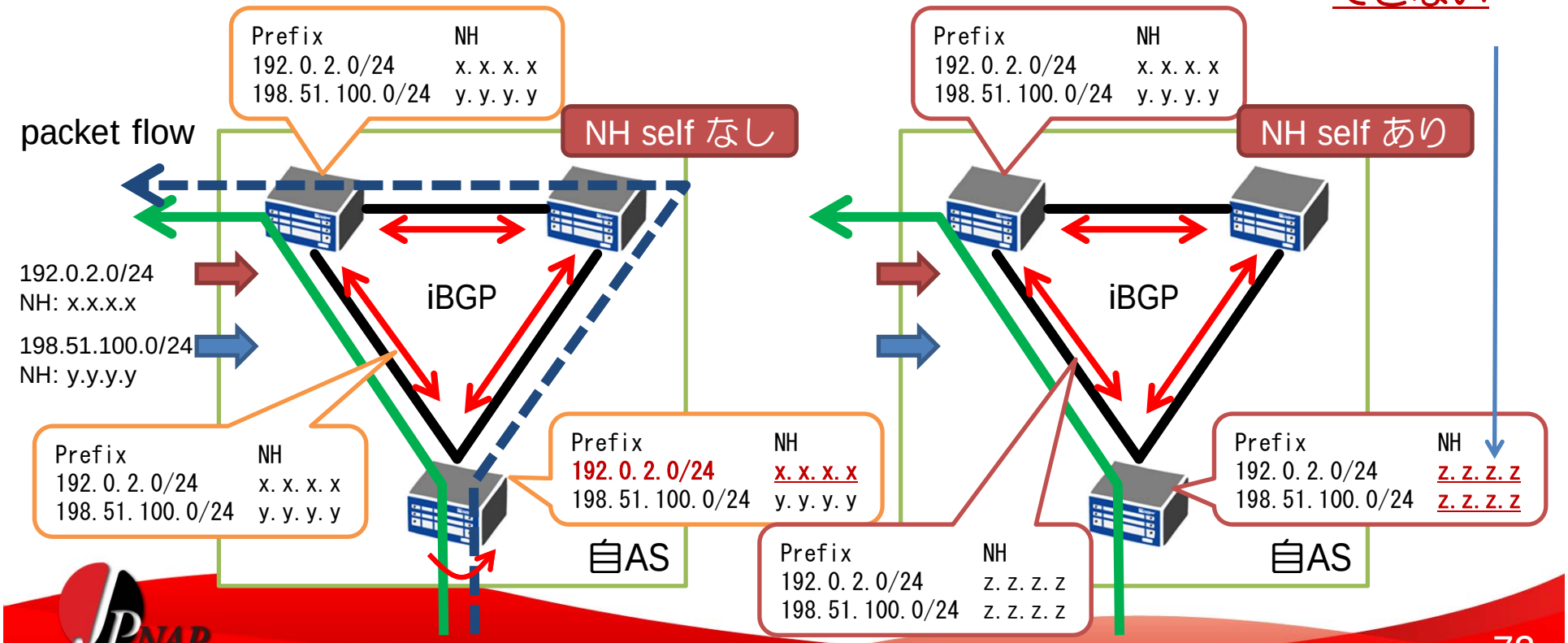
- RR
  - BGP policy を分けたくない
  - IGP domain を分けたくない
  - iBGP による細かい制御をしたくない
    - iBGP session を減らしたい



next-hop self

意外と重要!

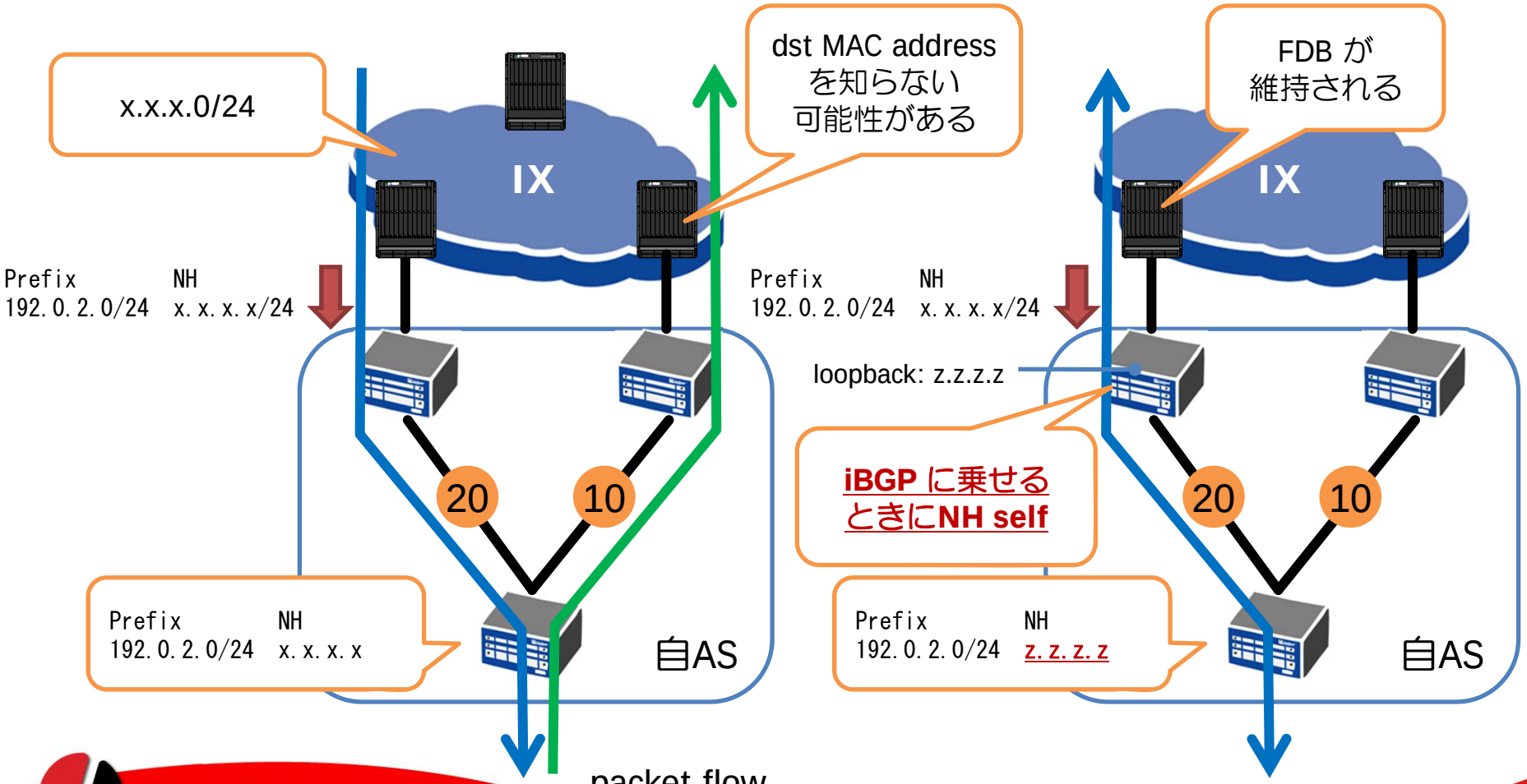
- 自AS 内のtraffic を細かく制御したい場合はnext-hop self しないほうがいい
  - 経路制御のオプションを1つ失う
- したほうがいい場合もある (後述)



next-hop self したほうがいい場合

■IX 経由でもらった経路をiBGP に流すとき

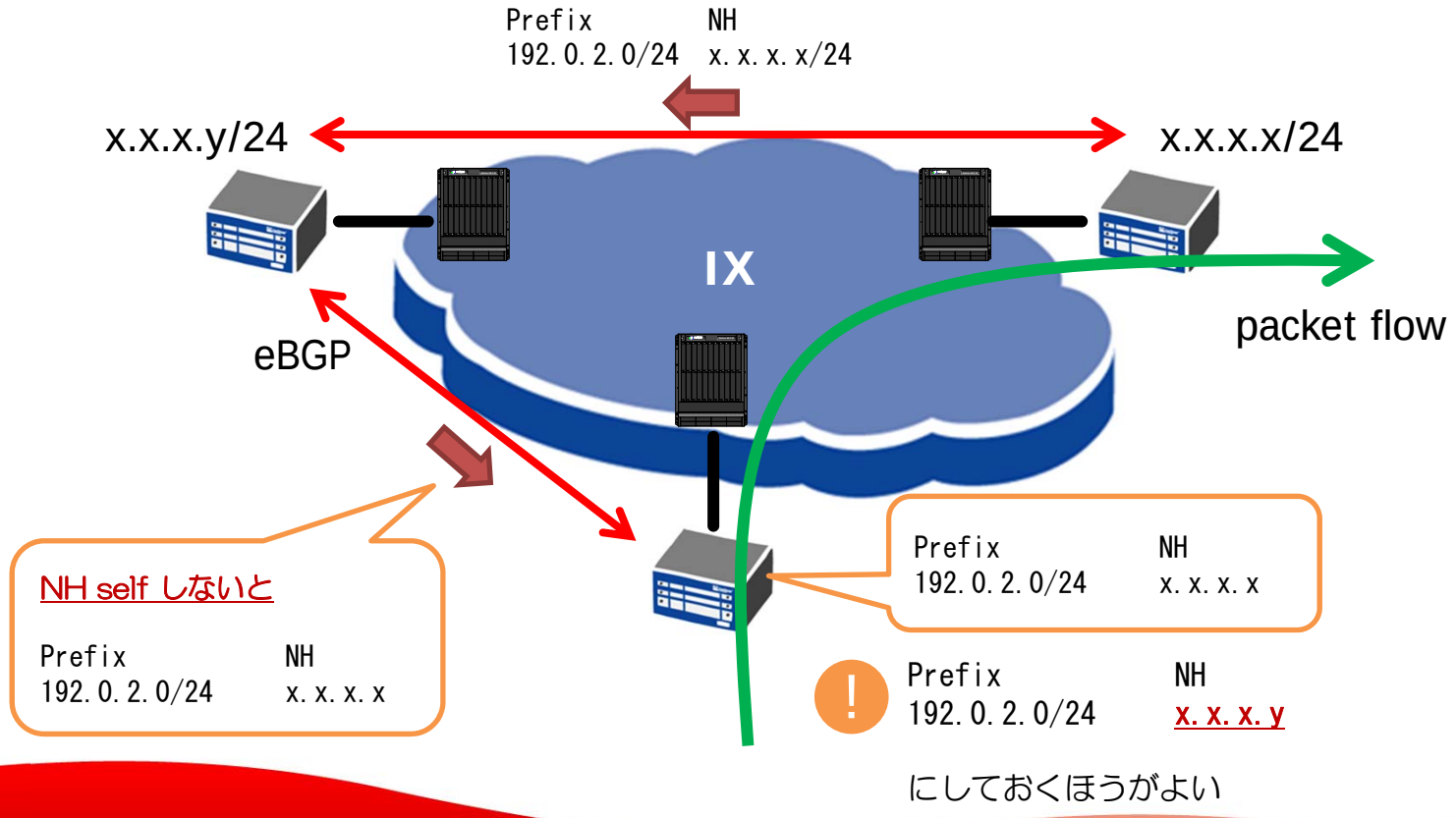
http://www.janog.gr.jp/doc/janog-comment/jc1005.txt



next-hop self したほうがいい場合

- IX 経由でもらった経路を 同じIX 上の別の eBGP の流すとき

http://www.janog.gr.jp/doc/janog-comment/jc1005.txt



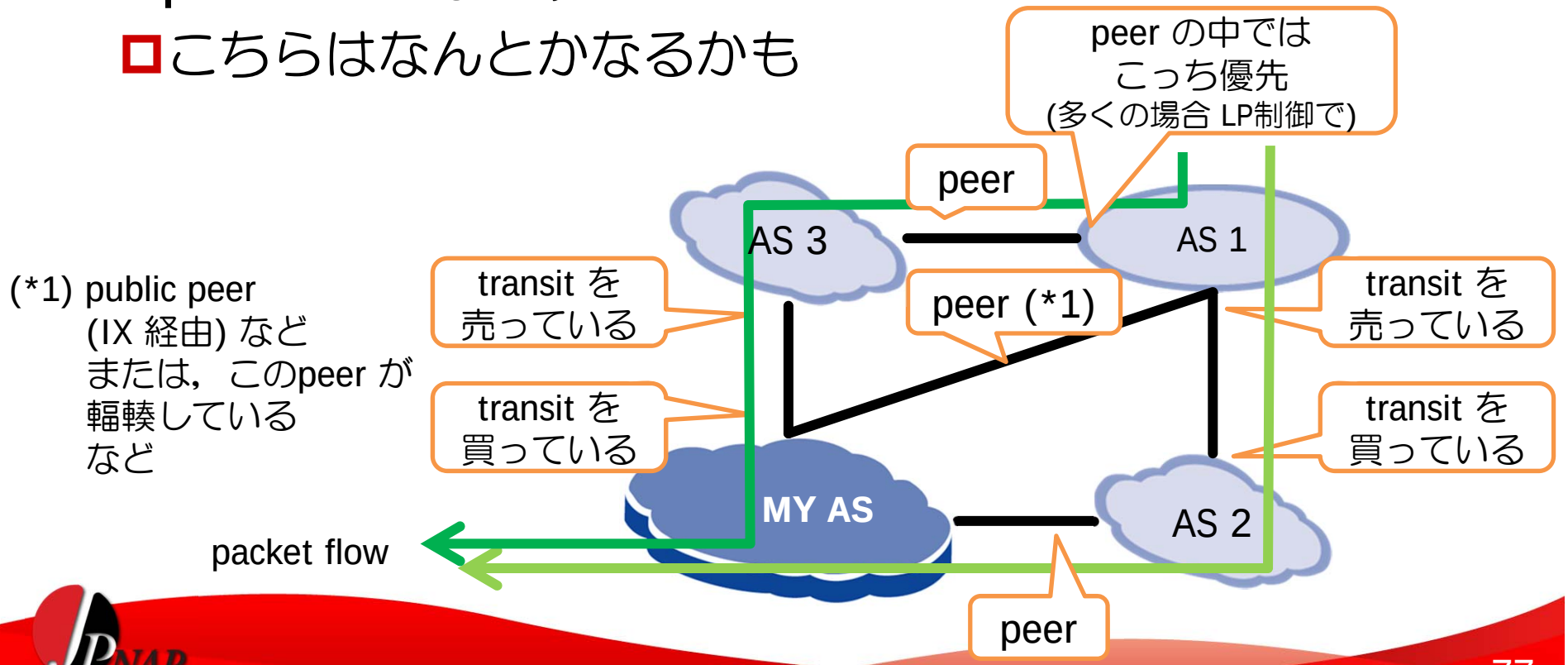
PA/PI Address のOrigination

- 安定的にInternet に存在し続ける必要がある
  - なくなると，自AS への到達性が失われる可能性がある
  - network 上で最も安定している数台のrouter で
  - Route Reflector を使っている場合は，そこで originate するのがよさそう
    - originate するrouter は，既にあるcritical なrouter (RR) で兼用する

BGP 運用
traffic engineering

問題: 直接peer しているのに traffic が流れてこない

1. AS_PATH 的には近いのに，別のpeer からtraffic が入ってくる
   - 図のようなケースで AS1 内のrouting を変えることは困難
2. peer ではなく，transit からtraffic が入ってくる
   - こちらはなんとかなるかも

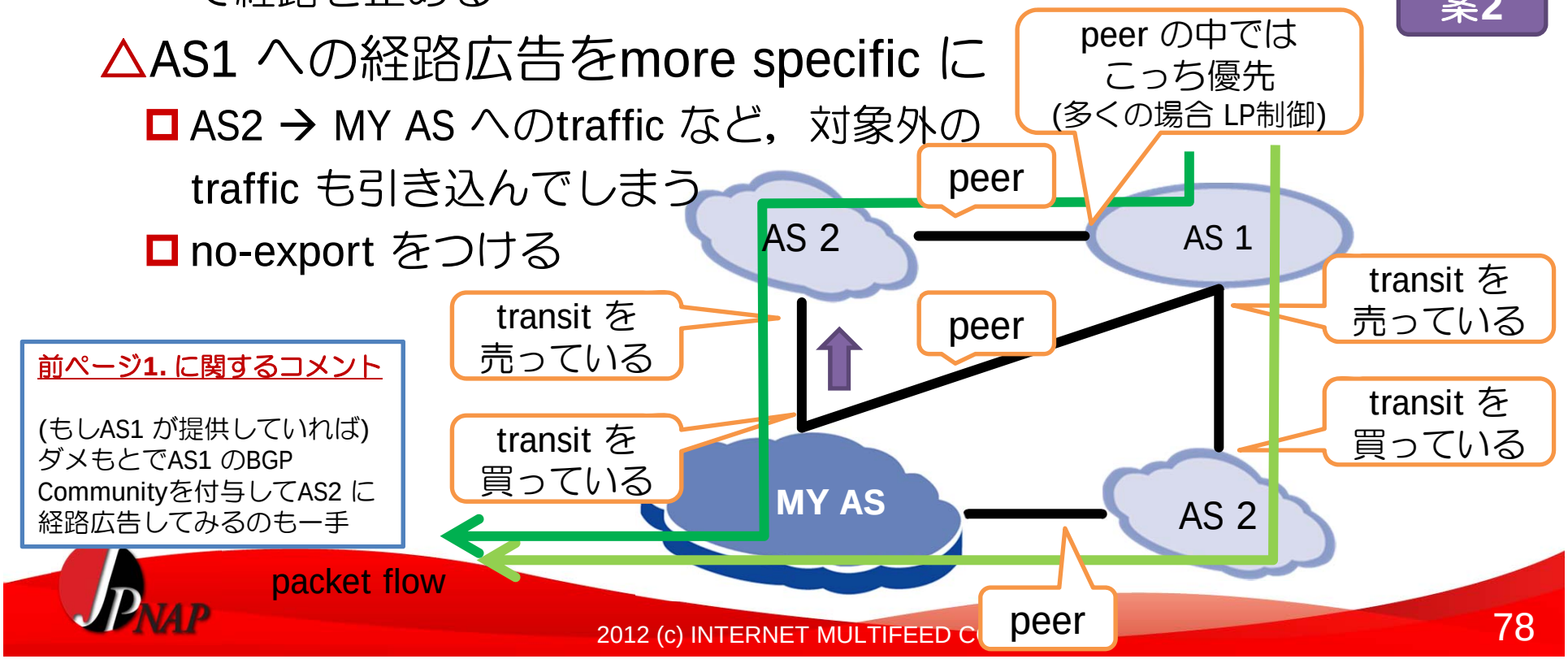


TE 案: 直接peer しているのに traffic が流れてこない

- AS1 にメールする (または AS2 にメールする) 案1
- AS2 への経路広告を操作する → transit 全体に影響するので，基本的には良くない
  - (もし提供していれば) AS2 のBGP Community を使い，AS1 に対して経路を止める

△AS1 への経路広告をmore specific に
  - AS2 → MY AS へのtraffic など，対象外の traffic も引き込んでしまう
  - no-export をつける

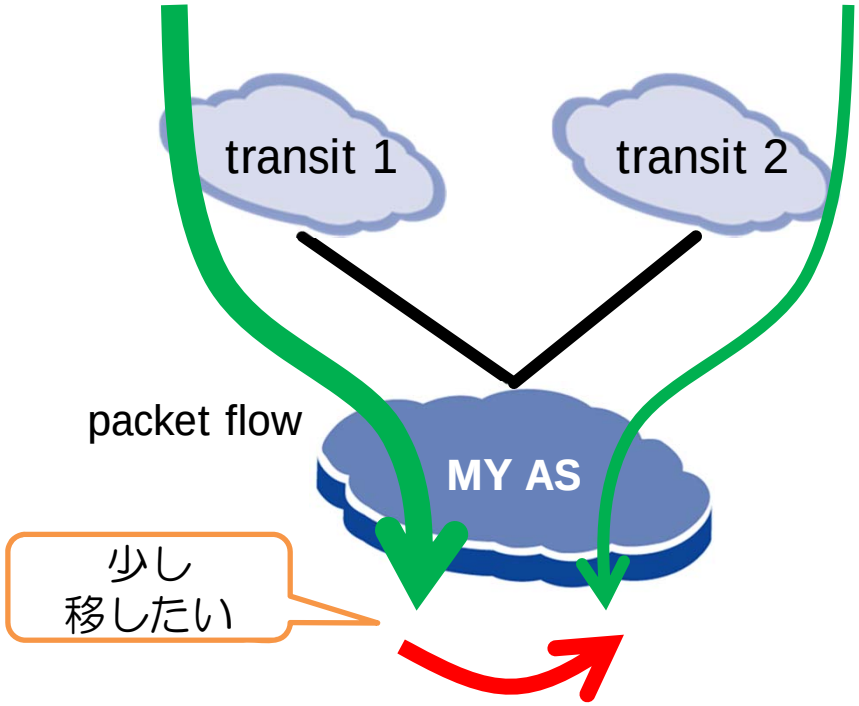
案2



前ページ1. に関するコメント

(もしAS1 が提供していれば)
ダメもとでAS1 のBGP Communityを付与してAS2 に経路広告してみるのも一手

問題: transit 間でtraffic を動かしたい



TE 案: transit 間でtraffic を動かしたい

案1

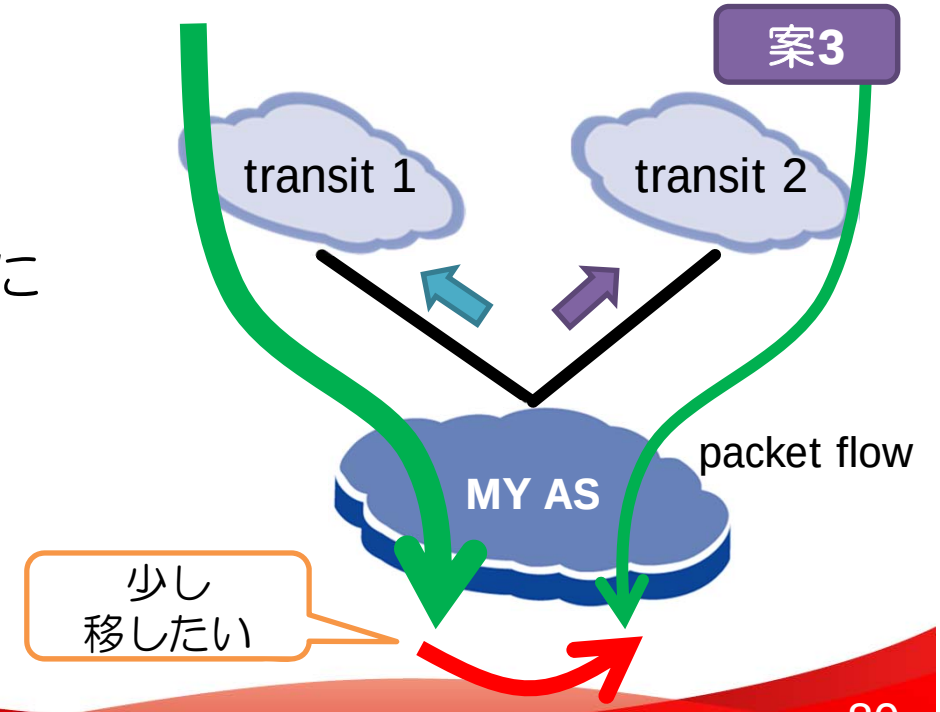
- (特定AS ホルダーからのtraffic が大きい場合) 直接peer する
- transit 1 への経路広告時にAS_PATH prepend

案2

  - 経験的にいくつもprepend しても効果は薄い
    - 経験的には限界は+3程度．+3 prepend して効果がなければ，増やしてもたぶん同じ
- (もし提供していれば) transit 1 のBGP community を使い，transit 1 内でのLP を下げる

△丁度いいvolume の経路について，
transit 2 への広告をmore specificに
△transit 1 への経路広告を止める

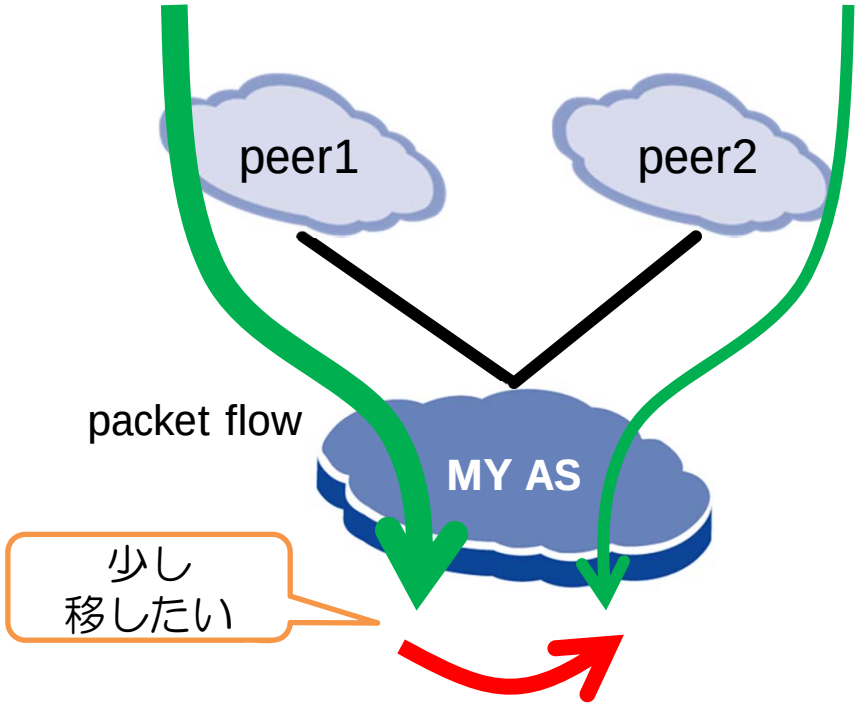
案3



補足: transit 間でtraffic を動かしたい

- transit コストを削減するため，ISP は日々制御している
  - 同じ量のtraffic を流すためのコストを最小化する
  - 例えば，まず案1 → NG なら案2
- transit の選択は，コスト/品質以外にも決定要素がある
  - ビジネス上のバーター
  - 資本関係

問題: peer 間でtraffic を動かしたい



TE 案: peer 間でtraffic を動かしたい

- peer 1 への経路広告時にAS_PATH prepend
  - 経験的にいくつもprepend しても効果は薄い
    - 経験的には限界は+3程度．+3 prepend して効果がなければ，増やしてもたぶん同じ

案1

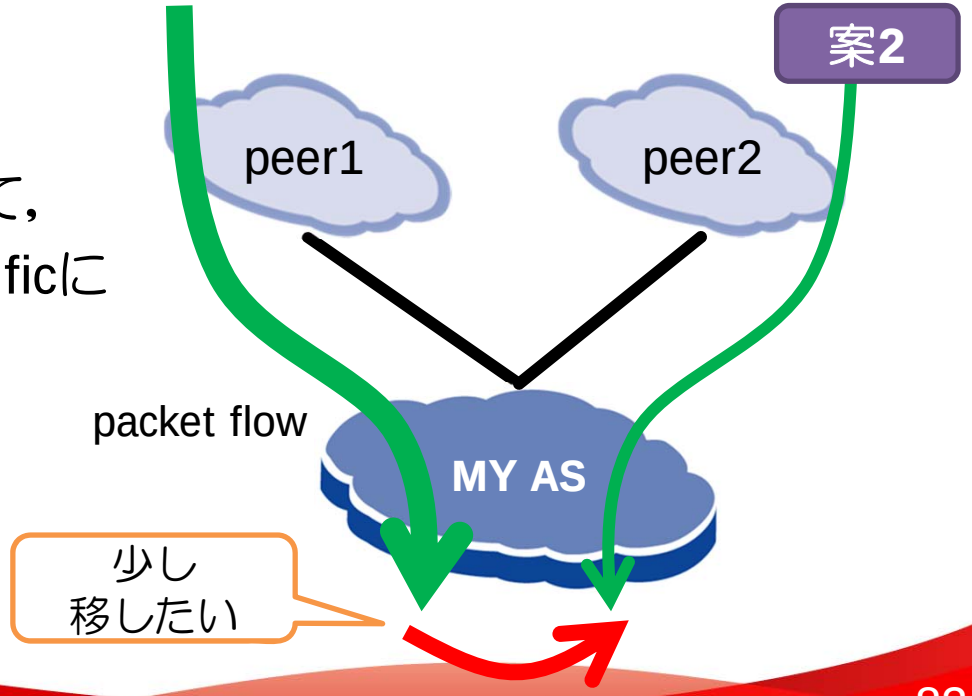
- (もしpeer1 と複数peer しているなら) traffic をどけたい peer 1 とのeBGP session でのみ特定の経路広告を止める

案2

△丁度いいvolume の経路について，
transit 2 への広告をmore specificに

△transit 1 への経路広告を止める

peer 1, peer 2 のASN が違うので，MED 操作が効かない !!

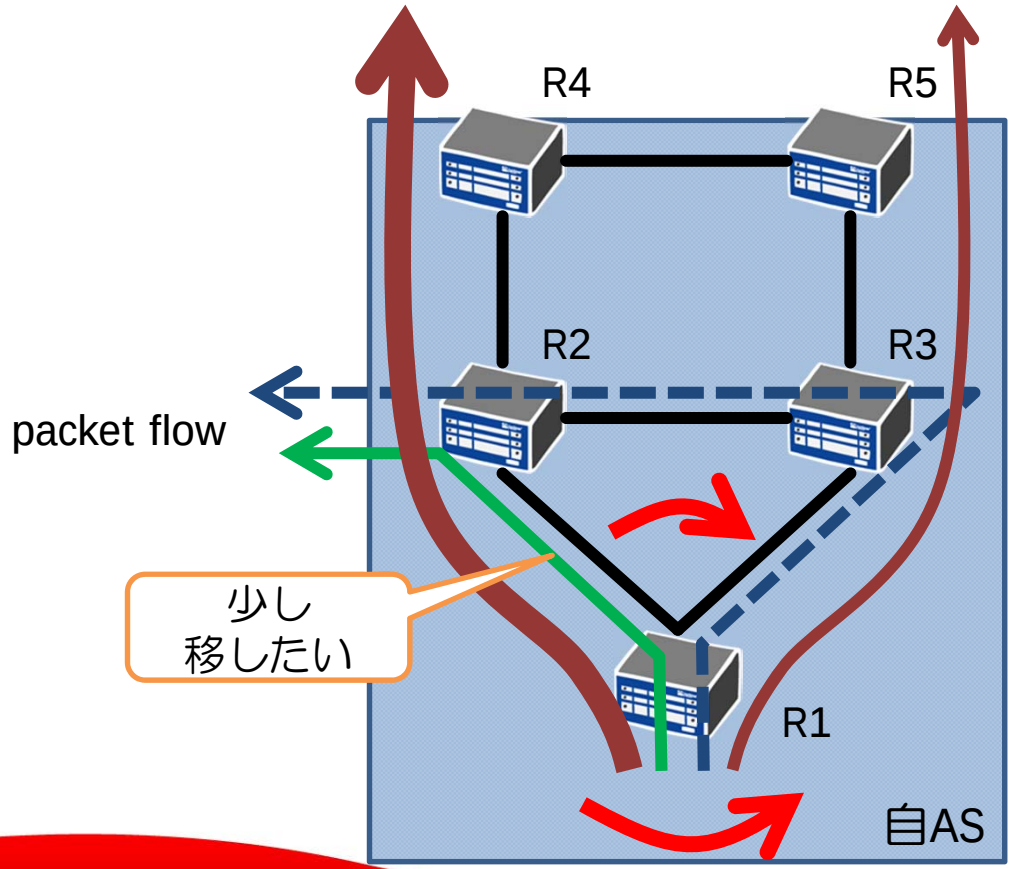


補足: peer 間でtraffic を動かしたい

- peer 間でTE したい理由
  - 品質が悪いから
    - 時間帯により輻輳する
    - latency が大きい
  - paid peer だから
  - なぜかは分からないが顧客に依頼されたから

問題: 自AS 網内のtraffic balance を変えたい

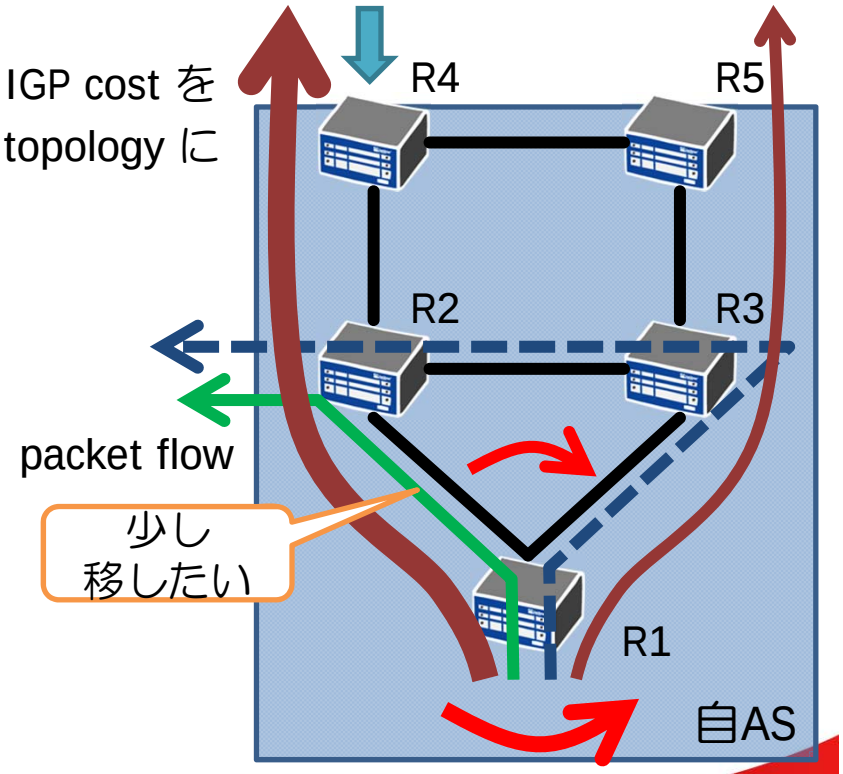
- R1-R2 link とR1-R3 link のbalance がよくないので，少しtraffic を移したい



TE 案: 自AS 網内のtraffic balance を変えたい

- まず R4から出るtraffic の一部をR5 に移せないか考える
  - R4, R5 両方でpeer していて，顧客ではないAS があれば，R4 での経路受信時に特定経路について
    - × LP を下げる (制御が強すぎる – AS_PATH が効かなくなる)
    - △ AS_PATH prepend (制御がまだ少々強い．自AS 内全域に影響を与える *1)
    - ◆ MED を追加する (LP 操作は制御が強い)
    - ◆ (MED を制御に使えない場合 *2) passive IGP cost を付ける (そのBGP session 全体に影響し，topology によっては劇的に変化するので注意)

案1



(*1) 一方，MED は一般的にAS_PATH より頻繁に変更されることが多く，このような操作に使いやすい．また，別の箇所で追加したMED を減らす(キャンセル) することも可能．

AS_PATH はtransitive であり (MED はnon-transitive) ，万が一prepend した経路がbest になると，外部には弱い経路が伝わってしまう．

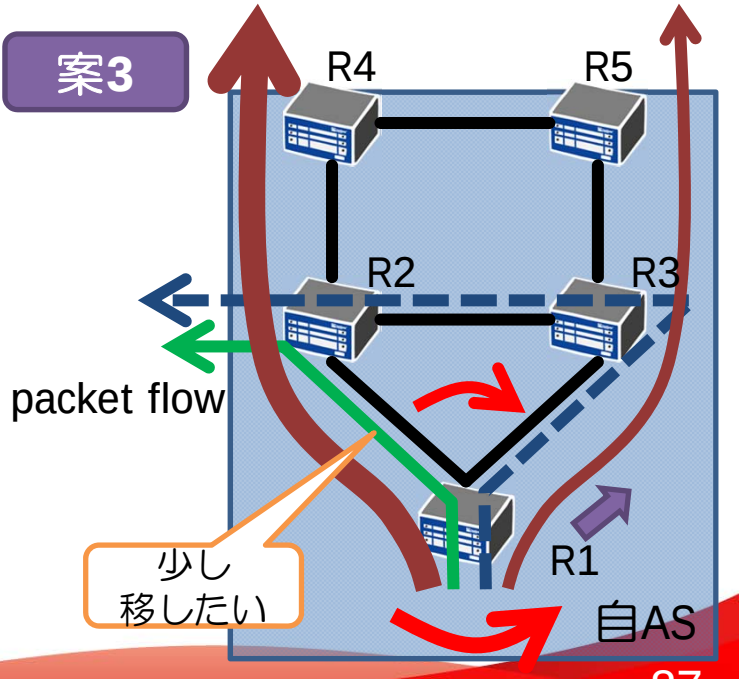
(*2) peer 経路のMED を2^32-1 で上書きしている場合など

TE 案: 自AS 網内のtraffic balance を変えたい

- ~~まずR4から出るtraffic の一部をR5 に移せないか考える~~
  - ちょうどいい移動対象traffic がなかったら
- R1→R2→R4 traffic をR1→R3→R5→R5 にできないか考える
  - R2-R4 のIGP cost を上げる (影響範囲が大きいので注意)
- R1→R2 traffic をR1→R3→R2 にできないか考える
  - (R2 が持っているeBGP session のうち，session 単位の traffic 合計で丁度いいものがあれば) next-hop になっているconnected prefix (ipv4 なら/30 など) へのstatic 経路を R1でR3 向けに設定する.

案2

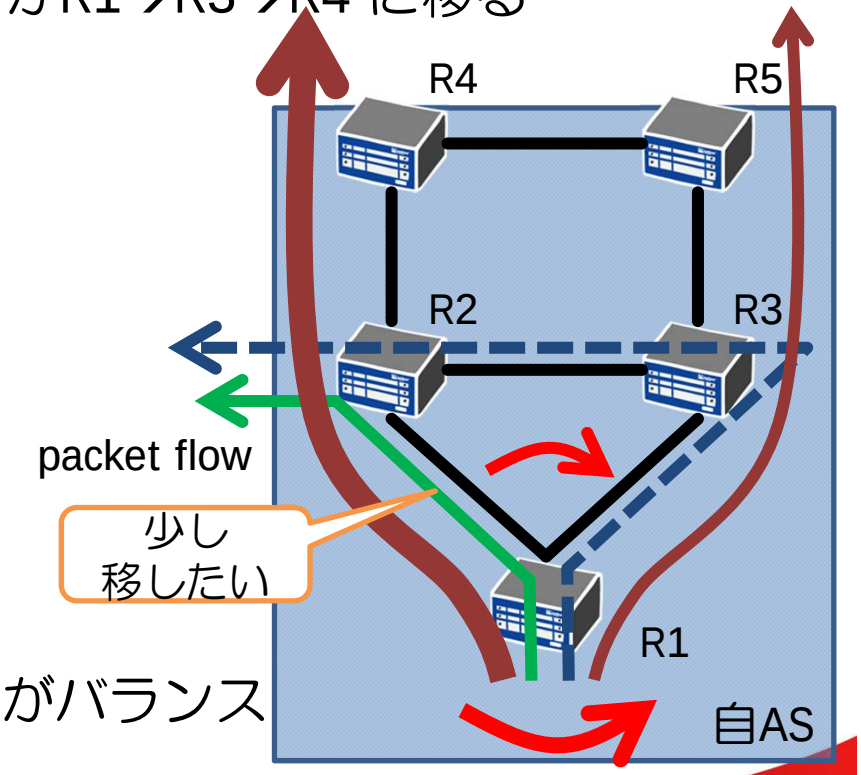
案3



(iBGP でnext-hop self していない場合に限る．また構成によってはrouting loop が起きる可能性があるので注意)

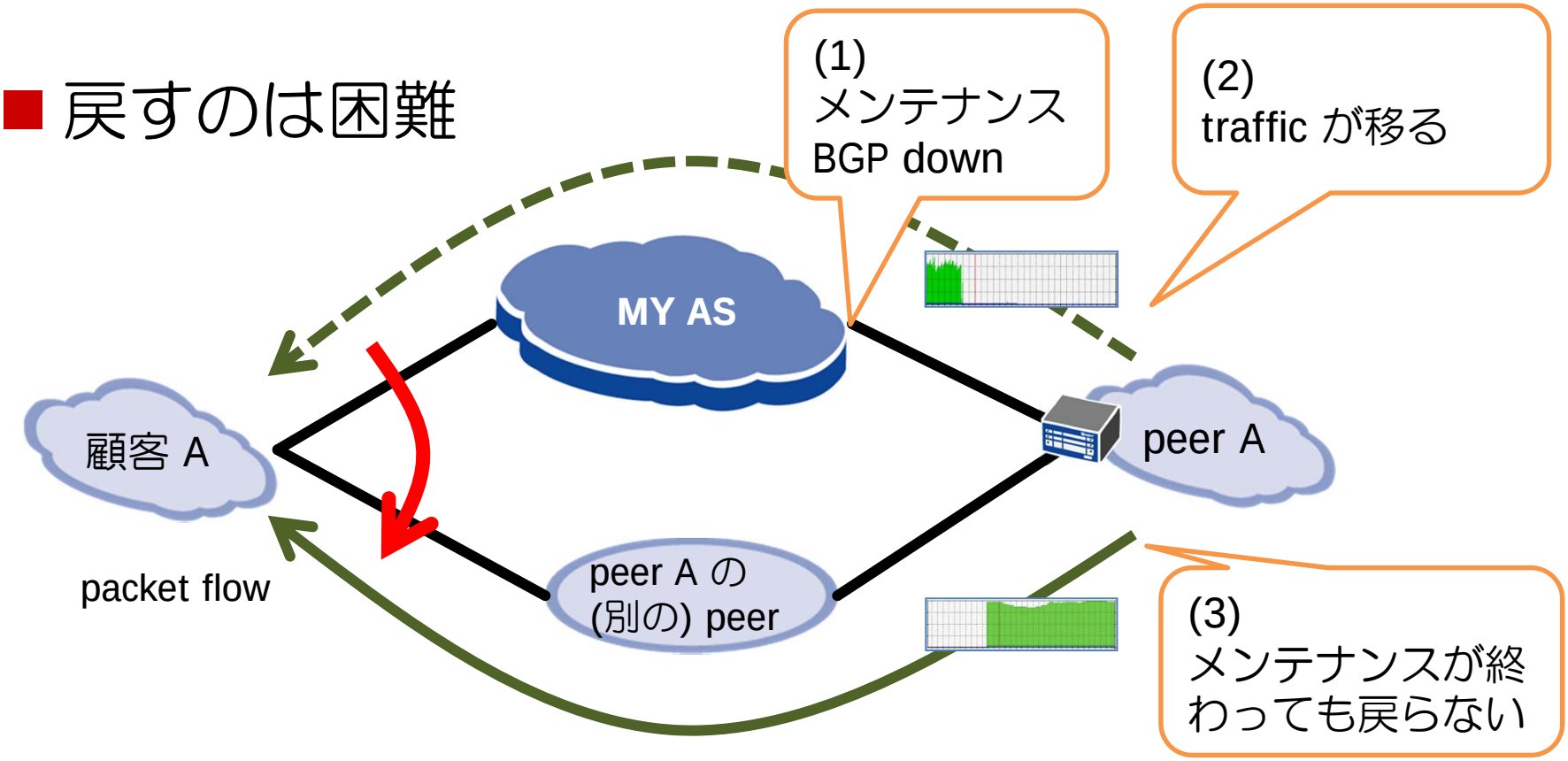
TE 案: 自AS 網内のtraffic balance を変えたい

- 物理的な変更を伴うアイデア
  - R3-R4 (R2-R5 も) にlink を追加
    - R1-R3-R4 / R1-R2-R4 でECMP を効かす
      - ✓R1→R2→R4 traffic の半分がR1→R3→R4 に移る
  - 収容を変える (R4 → R5)
- その他
  - R4, R5 が同じIX に接続しているなら
    - iBGP のnext-hop self をやめる
      - ✓R1→R2→R4, R1→R3→R5がバランス



peer 間 などでよくある，traffic 移動

- 戻すのは困難



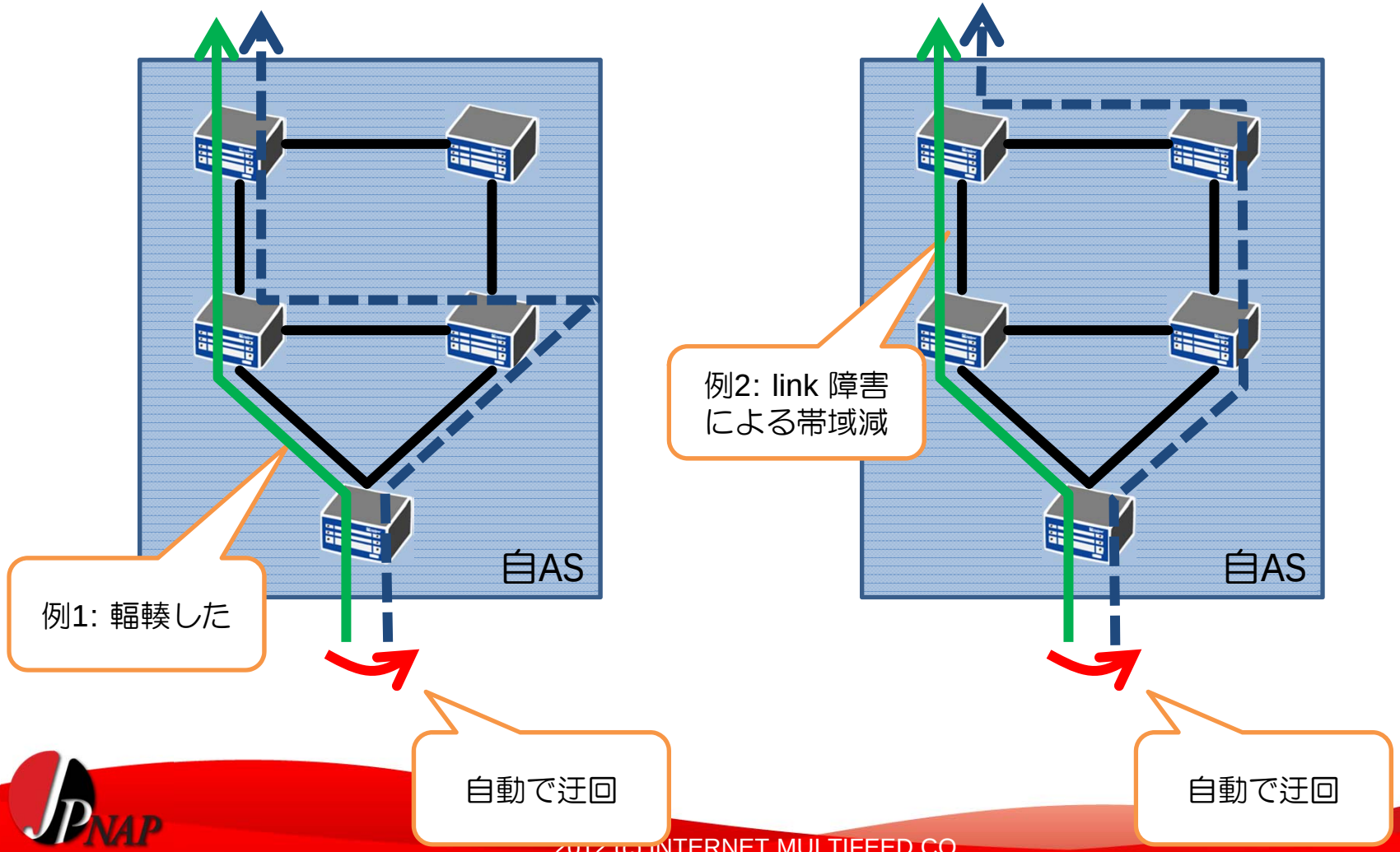
- eBGP 間のbest path selection はrouter ID ではなく “経路の生存期間” で決定する場合が多い

MPLS-TE

- 自AS 網内のtraffic balance を自動で制御する技術
  - 空き帯域を自動で探し，そこへrouting
  - QoS も可能
- MPLS (Multi-Protocol Label Switching) + RSVP (ReSerVation Protocol)
- IP Packet にlabel をつけてカプセル化
  - 仮想的なトンネル (LSP: Label Switching Path) をつくる

MPLS-TE

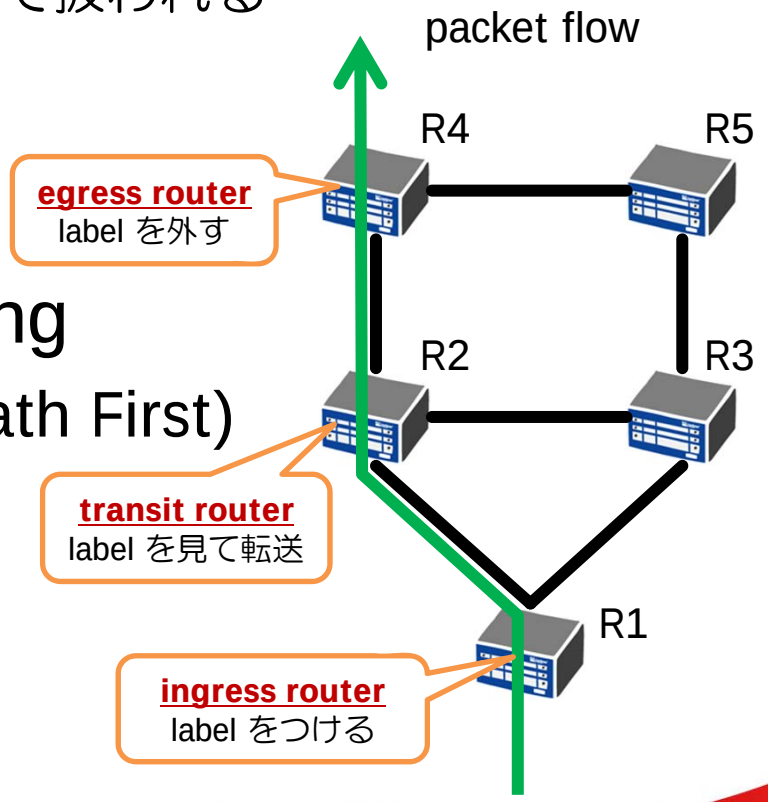
- traffic のend-point は変わらず，reroute する



MPLS + RSVP のしくみ

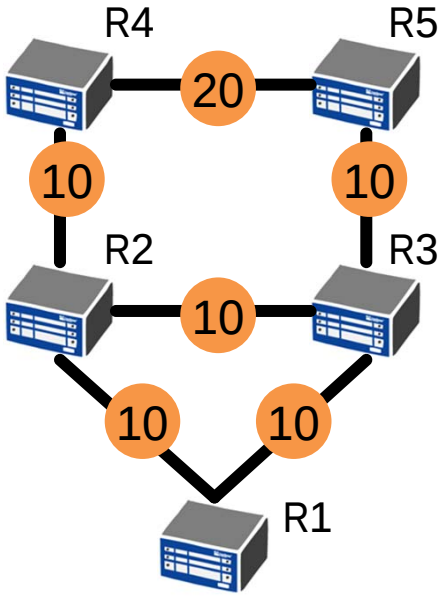
- LSP はLSR (Label Switching Router: MPLS を設定するrouter) 群で 2x full mesh 分 張る
  - LSP は片方向通信のみ
    - 通信方向が異なると 別のLSPとして扱われる
    - R1→R2→R4
    - R4→R2→R1

    は別物
- LSPを張る前にRSVP signaling
  - CSPF (Constrainted Shortest Path First)
    - Link State 型のIGP に依存する
      - ✓ OSPF
      - ✓ ISIS



RSVP のしくみ

- LSP を張る前に，LSP 毎にあらかじめ設定された帯域が確保できるかをingress / egress router 間でsignal を送り合う
  - IGP cost の小さい順に調べる
  - 各LSR は 接続されているRSVP link の空き帯域を計算
  - R1-R4 LSP の例
    1. R1-R2-R4 で張れるか? → NG
    2. R1-R3-R4 は? → NG
    3. R1-R3-R5-R4 は?
- LDP を使ったり(帯域計算なし)，LSP を特定Path に固定したりもできる



MPLS-TE

- network event により帯域が縮退
  1. RSVP signaling
  2. LSP が空き帯域に移る
- 自動で空き帯域を探索してくれるので，手動によるTE 不要
- LSP が落ちても即packet loss ではない
  - IGP にfallback する (BGP のprotocol nexthopの解決のためにLSP + IGP を使う．優先度が LSP > IGP)
- fast reroute
  - LSP が落ちたときの convergence を早くするために，あらかじめbackup LSP を張っておく

```
koji@test-router> show route 192.0.2.0/24

inet.0: 57 destinations, 57 routes (57 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.0.2.0/24       *[BGP/170] 5d 07:12:01, localpref 100, from 192.0.2.1
                      AS path: I
                    > to 198.51.100.1 via ae1.0, label-switched-path r1-r5-00


koji@test-router> show route 192.0.2.1

inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.0.2.1/32       *[RSVP/7] 5d 06:47:49, metric 16
                    > to 198.51.100.1 via ae1.0, label-switched-path r1-r5-00
                    [IS-IS/18] 5d 06:47:48, metric 16
                    > to 198.51.100.1 via ae1.0, label-switched-path r1-r5-00
```

MPLS-TE

- LSP に優先度を付けてpreemptive に動作させる → QoS
  - closed network ではよく使われている
- 10年以上前の枯れた技術
- Layer 2.5

MPLS-TE

- 利点
  - 手動TE からの開放
  - 帯域設計がラク
    - 最悪reroute してくれる
- 欠点
  - overlay model
  - label overhead
  - LSP size を設定するため，日々LSP 毎のtraffic 量をカウントする必要がある (MIB など)
  - 設定が煩雑
  - 理解しづらい

BGP 運用

その他

peering 判断の基本

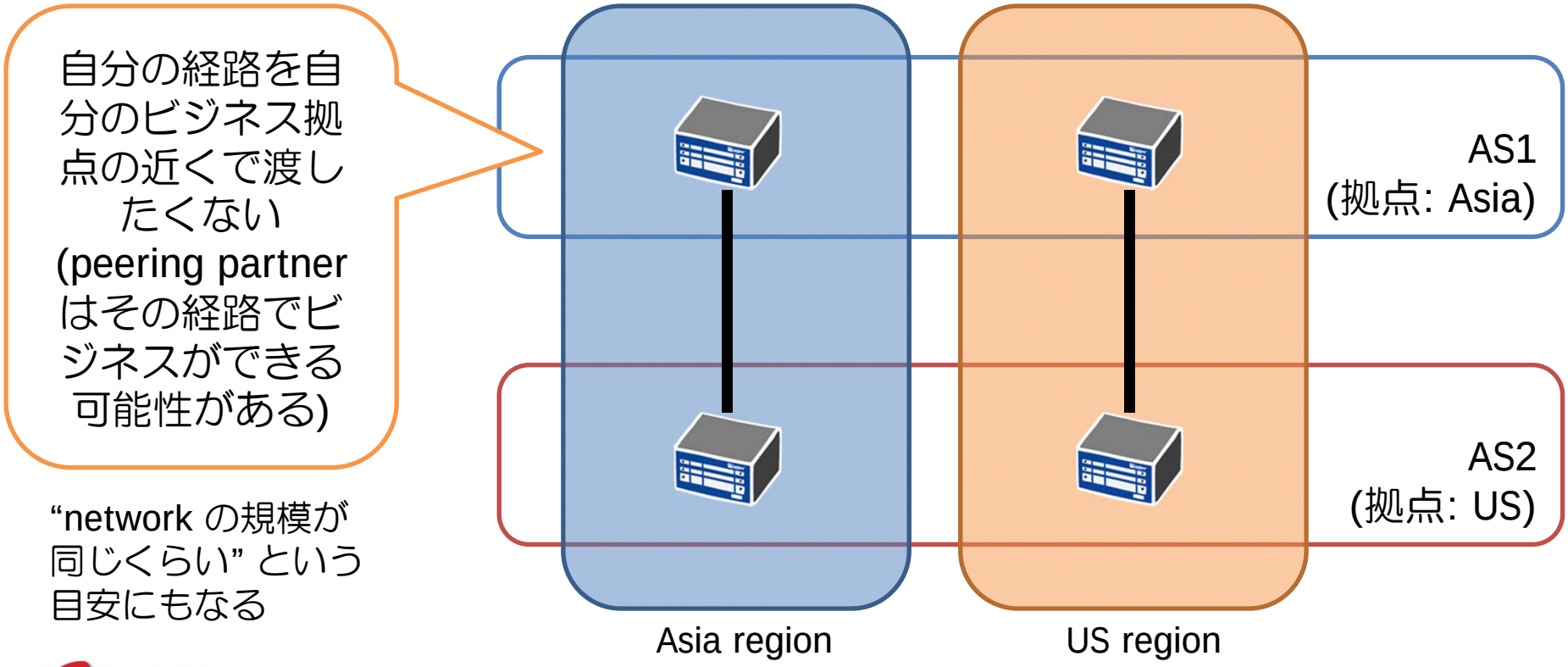
- transit コストを抑えることが主な目的
  - peer をするAS ホルダー同士が，お互いの力関係にもよるが
    - peer することでコストメリットがある
    - お互いのビジネスを侵害しない

    など，両者のpeering policy を満たして初めてpeering を行う

  - “コストメリットがある” ことの目安として”traffic が???Mbps 出ること” という条件がある場合や，(続く)

peering 判断の基本

- “お互いのビジネスを侵害しない” 条件として複数リージョン / 複数POP での接続を条件とするISPもいる



“network の規模が同じくらい” という目安にもなる

peering 判断の基本

- AS ホルダー同士の力関係により
  - paid peer
  - “ある地域でtransit を買えば，別の地域でpeer できる” というバーター

のような形態もある

- www.peeringdb.com にある程度まとめられている．”General Policy” が “Open” なAS ホルダーは peering に応じてくれる可能性大
  - web or mysql で一覧が取得できる

```
$ mysql -r -hpeeringdb.com -upeeringdb -ppeeringdb Peering
mysql> SELECT asn, name, aka, policy_locations, policy_ratio FROM
peerParticipants WHERE policy_general IN ('Open') ORDER BY asn;
```

peering 判断の基本

- ICP にとってISP とのpeering により ”ビジネスが侵害” される可能性は低いため，単純に”コストメリットがあるか?” が peering 判断の大きな評価軸になる場合が多い

  - 一方， ICP はICP とpeering するメリットはあまりない

transit provider のfilter を制御する

- どこかのInternet Routing Registry (IRR) に登録する必要がある
  - 日本でよく使われているもの
    - JPIRR (jpirr.nic.ad.jp)
      - ✓JPNIC 会員のみ
    - RADB (whois.radb.net)
      - ✓有償 ($500/y)
    - NTTCOM (rr.ntt.net)
      - ✓ntt.net ユーザーのみ

		このIRR の情報を保持するか?		
		JPIRR	RADB	NTTCOM
このIRRが	JPIRR	○	○	×
	RADB	○	○	○
	NTTCOM	○	○	○

お互いにmirror しているため，どこか1箇所に登録すればよい

transit provider のfilter を制御する

- いくつかのobject を登録する
  - Maintainer
  - Aut-num
  - Route
  - AS-Set

http://www.nic.ad.jp/doc/jpnic-01077.html

- 登録したAS-Set object をtransit provider に伝える
- www.peeringdb.com にも登録しておく

経路奉行

- JPIRR に登録するといいことがある
  - BGP route hijacking を検知 / 通知してくれる
    - http://www.nic.ad.jp/ja/ip/irr/jpirr_exp.html
  - ISAlarm, BGPMON (*) みたいなもの

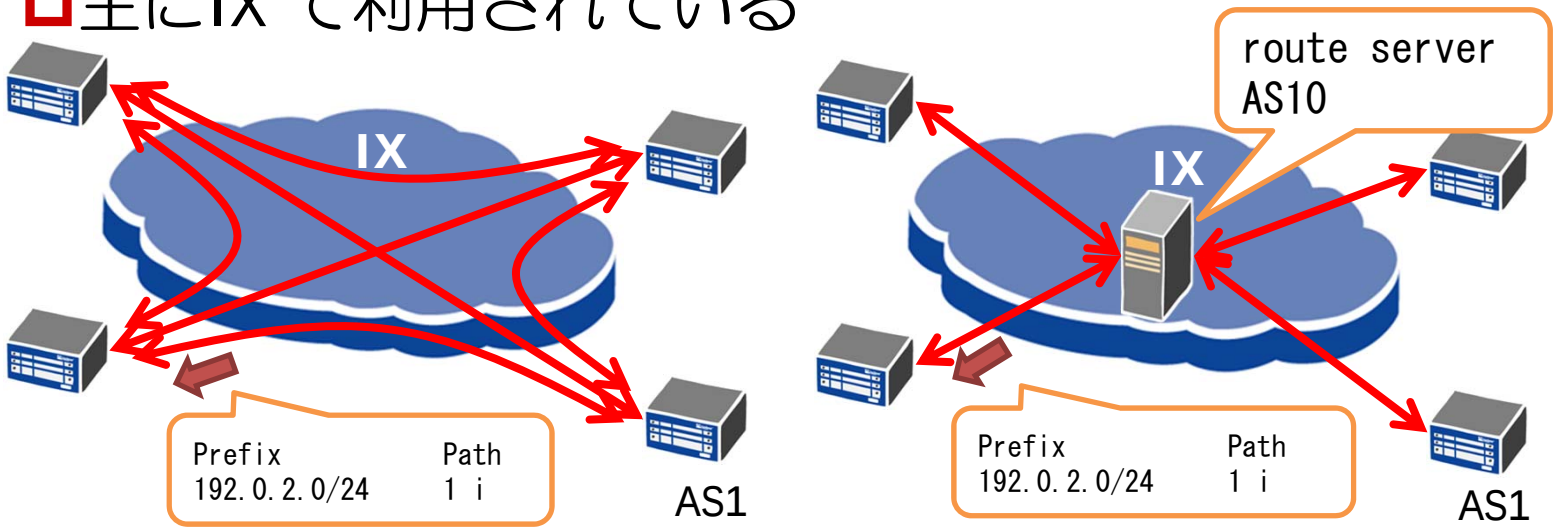
```
route:          202.12.30.0/24
descr:          JPNICNET
                Japan Network Information Center
                Kokusai Kogyo Kanda Bldg. 6F
                2-3-4 Uchi-Kanda
                Chiyoda-ku, Tokyo 101-0047
                JAPAN
                X-Keiro: okadams@nic.ad.jp            <--追加記述
                X-Keiro: okadams-noc@nic.ad.jp  <--複数あて先に通知する場合記述
origin:         AS2515
admin-c:        SN3603JP
tech-c:         YK11438JP
tech-c:         MO5920JP
notify:         system@nic.ad.jp
mnt-by:         MAINT-AS2515
changed:        apnic-ftp@nic.ad.jp 20080116
source:         JPIRR
```

詳しくはこのあとの“肌で感じる！インターネットルーティングセキュリティ”で !!

(*) ISAlarm: http://www.ripe.net/is/alarms
BGPMON: http://www.bgpmon.net/

route server

- eBGP session をscaleさせる技術
  - 主にIX で利用されている



- 通常のpeering の手法 (bilateral) とroute server 経由のpeering の手法 (multilateral) で，RIB の中身がほぼ同じ
  - remote AS (この例ではAS10) はAS_PATH に載らない
- **eBGP session 数を減らす**ことで運用をラクに
  - open policy のAS ホルダー向き

道具箱に
道具を詰める

xflow

- TE のため
  - どのAS, prefix が何bps 持っているか?
  - あるlink に乗っているtraffic のうち，??% を移すためにどのAS, prefix を制御すればいいか?

  などを調べたい
- netflow v5 / v9
- sflow v5
- OSS collector (nfdump, flowtools)で十分
- sampling rate: 1/8192 ~ 1/10000

web services

internet の構造を知るためのヒント

as-relationships:

http://www.caida.org/data/active/as-relationships/

- どのAS がどのAS からtransitを買っている(ように見える)か．．．という情報が掲載されている
- リサーチの成果物なので事実との不一致もある

bgplay

http://bgplay.routeviews.org/

経路を解析することでAS 間の接続性が確認できる

- routeviews: route server にも直接telnet 可能

  http://www.routeviews.org/

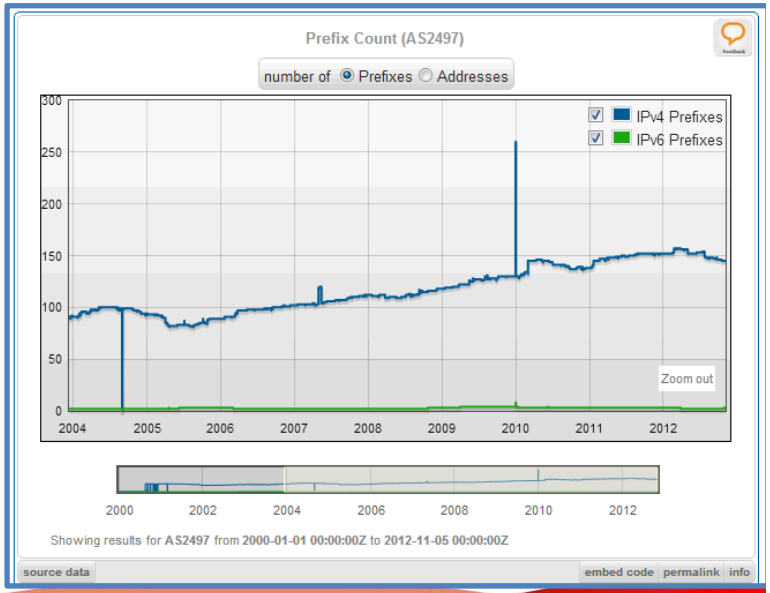
web services

- internet の構造を知るためのヒント
  - rips stat

    https://stat.ripe.net

    さまざまなinternet resource の状態が閲覧できる.
    すごく便利. (iphone app もある)

  - looking glass
    - http://www.bgp4.as/looking-glasses
    - http://www.traceroute.org
    - http://www.lookinglass.org/
    - http://www.bgp4.net/



rancid

- http://www.shrubbery.net/rancid/

- network device へのlogin を簡略化できる
- 指定のcommand 群の実行を，複数device に対して実行できる
- expect でprogram 可能
- config の自動backup / VCS への投入

ruby / python

- CLI をwrap してprogram
- SNMP get

- ruby はweb と親和性が高い
- python はnetwork 系library が豊富
- “自動でrouter に設定を入れる” など

mrtdump / bgpdump

- bgp update / withdraw をdump, 解析
- bgp table 自体をexport することも可能

- Quagga などOSS route server が一台手元にあると便利
  - VM でもOK
  - bird, openbgpd も

exabgp

- BGP session を張り，任意のBGP update を作れるsimulation 用tool
  - 商用のBGP test tool と比べても，使い方によっては遜色ない
    - performance の面で弱い可能性はある
  - 最近よくメンテナンスされていて使いやすくなっている

経路filter 関係

- peval (IRR Toolset)
  - そろそろメンテナンスが厳しいらしい
  - 代替品
    - Net::IRR
    - bgpq3
    - IRRPowerTools
    - Md4.7
    - p2BGPTool

    - capirca というのもある

自動化支援 (router メーカー製)

- event driven でコマンド実行
  - Juniper: Junoscript
    - xml ベース
  - Cisco: Embedded Event Manager (EEM)
    - tcl ベース
- Cisco: One Platform KIT (One PK)
  - packet processing までできるらしい

Community の一員になる

photo by Akinori Maemura

JANOG

- network 運用にまつわる情報の共有 / 議論
- 問題の共有
  - オペレーターcommunity で解決

  - 例:
    1. あるIP address block がGeoIP サービスに誤認識された
    2. そのせいでオンラインサービスに接続できなかった

各種 IX Meeting

- ユーザー会
  - JPNAP
  - JPIX
- Japan Peering Forum
  - Equinix

各種 Network Meeting

- Apricot (Asia)
- NANOG (North America)
- RIPE (Europe)

まとめ


シンプルに
わかりやすく
http://www.flickr.com/photos/techsavvyed/5926978939/

ありがとうございました

小島 慎太郎
インターネットマルチフィード (株)
koji@mfeed.ad.jp
http://about.me/codeout

Appendix

xflow export 𝒟sample config

- Juniper M120

```
forwarding-options {
    sampling {
        input {
            family inet {
                rate 10000;
            }
        }
        output {
            cflowd 10.0.0.1 {
                port 2055;
            }
        }
    }
}

firewall {
    filter sampling_filter {
        term accept_all {
            then {
                sample;
                accept;
            }
        }
    }
}
```

```
interfaces {
    ge-0/0/0 {
        unit 0 {
            family inet {
                filter {
                    input sampling_filter;
                    output sampling_filter;
                }
            }
        }
    }
}

groups {
    sampling_interface {
        interfaces {
            <"[fgx]"e*> {
                unit <*> {
                    family inet {
                        filter {
                            input sampling_filter;
                            output sampling_filter;
                        }
                    }
                }
            }
        }
    }
}
interfaces {
    apply-groups sampling_interface;
}
```

xflow export 𝒟sample config

- Juniper M120

```
forwarding-options {
    sampling {
        input {
            family inet {
                rate 10000;
            }
        }
        output {
            cflowd 10.0.0.1 {
                port 2055;
            }
        }
    }
}

firewall {
    filter sampling_filter {
        term accept_all {
            then {
                sample;
                accept;
            }
        }
    }
}
```

```
interfaces {
    ge-0/0/0 {
        unit 0 {
            family inet {
                filter {
                    input sampling_filter;
                    output sampling_filter;
                }
            }
        }
    }
}

groups {
    sampling_interface {
        interfaces {
            <"[fgx]"e*> {
                unit <*> {
                    family inet {
                        filter {
                            input sampling_filter;
                            output sampling_filter;
                        }
                    }
                }
            }
        }
    }
}
interfaces {
    apply-groups sampling_interface;
}
```

xflow export のsample config

Cisco

7600 (PFC/DFC でのsample)

```
mls flow ip interface-full
mls netflow
mls nde sender version 5
mls sampling packet-based 1024 16000

ip flow-export version 5
ip flow-export destination 172.16.162.99 2055

interface GigabitEthernet2/22
 ip flow ingress
 mls netflow sampling
```

12000

```
flow-sampler-map sampler
  mode random one-out-of 10000
 !
 ip flow-export version 5
 ip flow-export destination 10.0.0.1 2055

 interface FastEthernet0/0
  ip route-cache flow
  flow-sampler sampler
```

xflow export のsample config

- Brocade BI8000

```
sflow enable
 sflow sample 8192
 sflow destination 10.0.0.1 6343

 interface ethernet 1/1
  sflow forwarding
```