Multihoming Techniques

bdNOG8

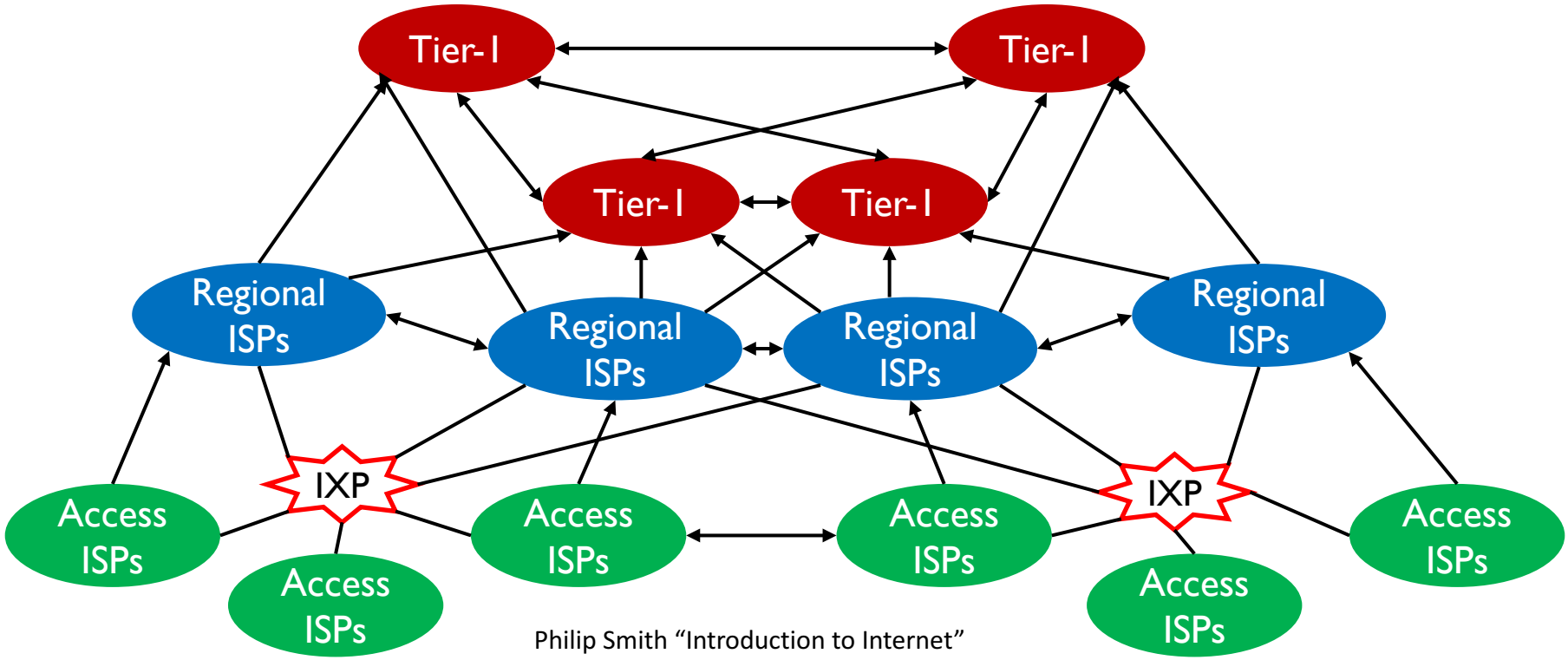
May 4 – 8, 2018

Jashore, Bangladesh.

APNIC

ISP Hierarchy

- Default free zone
  - Internet Routers that have explicit routes to every network on the Internet
    - Regional /Access Providers think there could be some missing routes (default routes exists!)



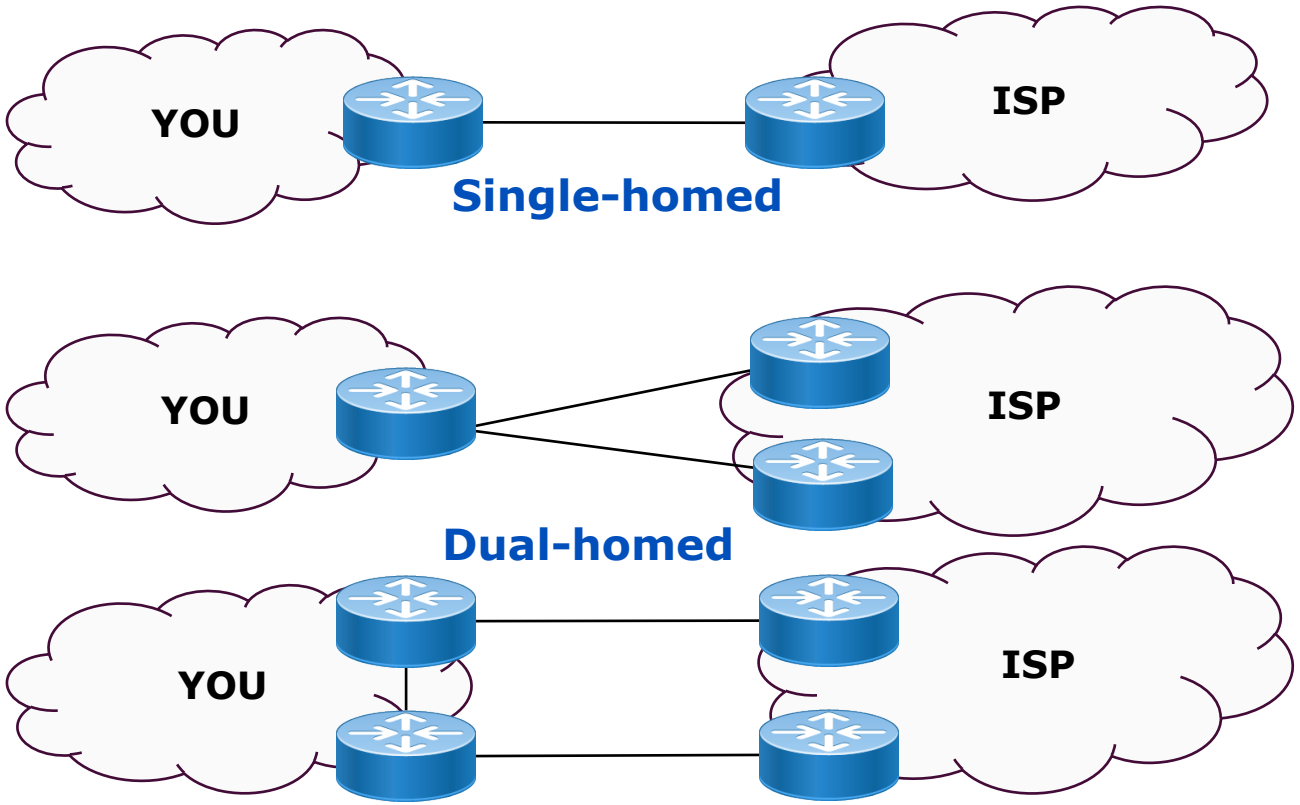
Philip Smith "Introduction to Internet"

Exchanging Routes

- Pay someone to advertise your networks
  - TRANSIT
  - Make sure they have good onward peering/transit!
- Interconnect with as other ASes to exchange locally originated routes and traffic
  - PEERING
  - Private Peering
    - Between two ASes
  - Public Peering
    - at an IXP (domestic/global)

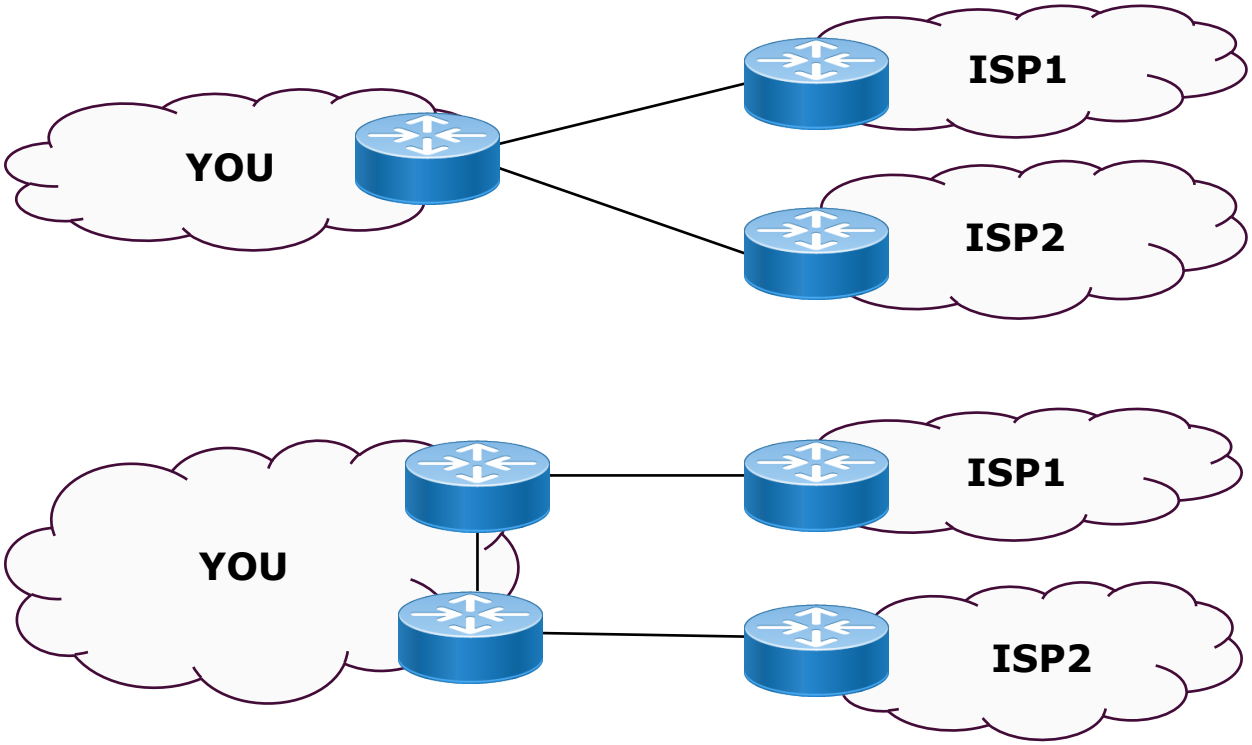
Achieving Redundancy

- More than one path to the same ISP
  - Dual-homed



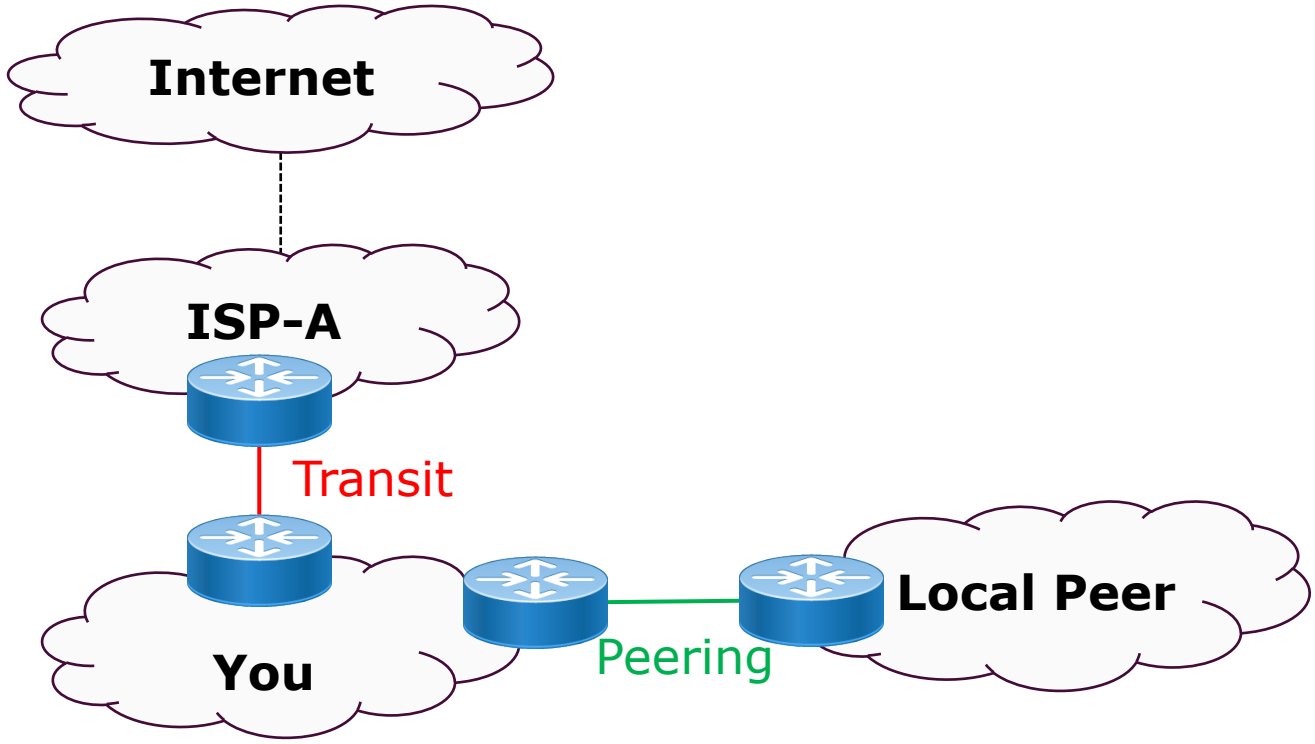
Achieving Redundancy – Multihoming

- More than one upstream ISP
  - Multi-homed



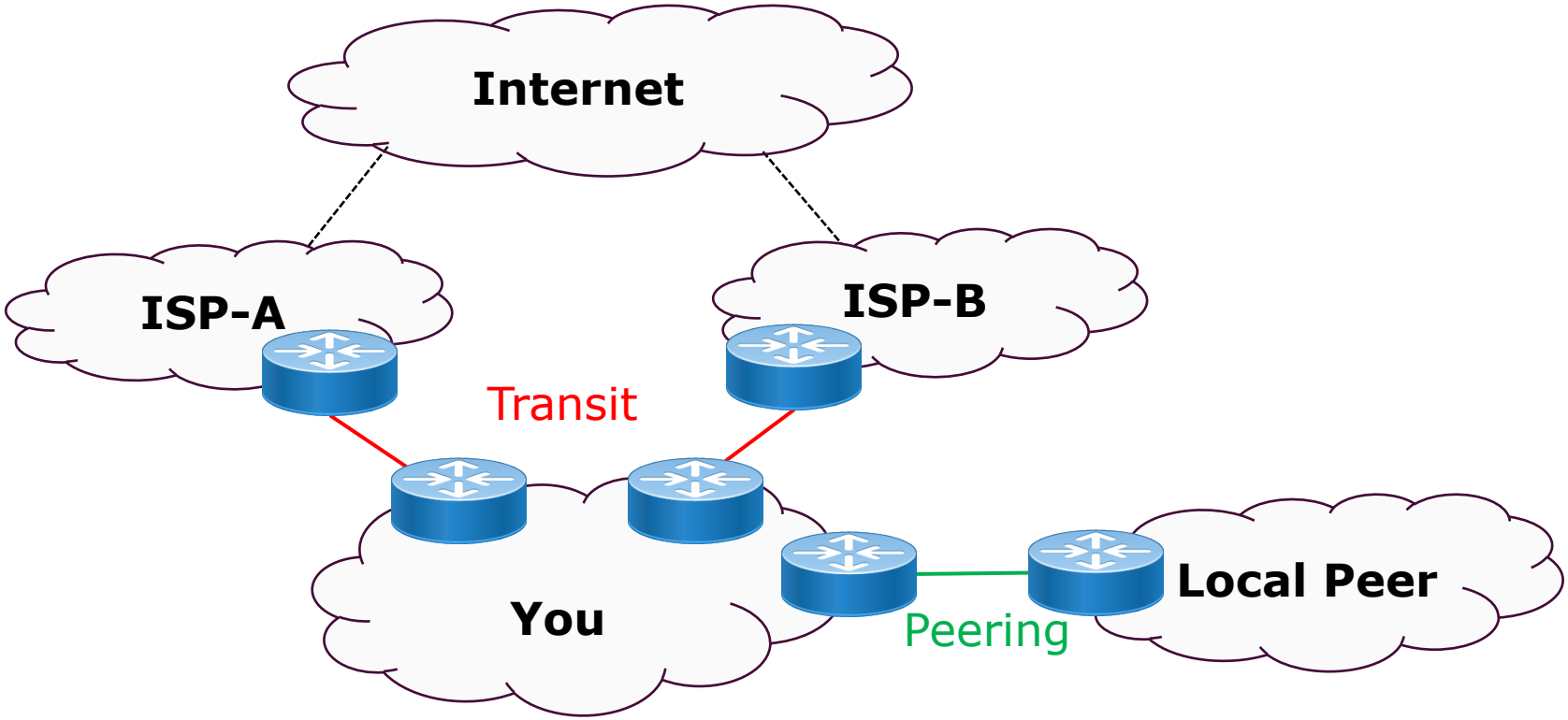
Multihoming

- One upstream and local peering



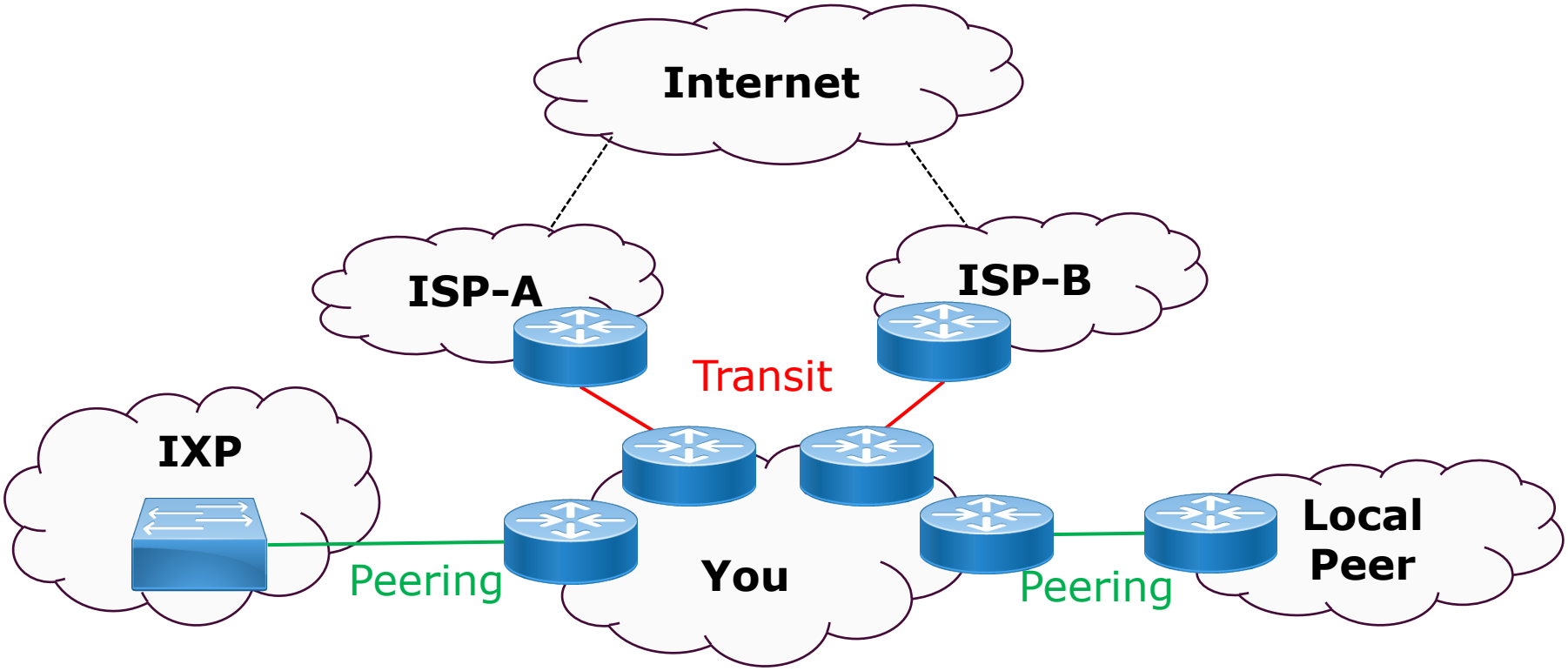
Multihoming

- More than one upstream ISP and local peering



Multihoming

- More than one upstream ISP with local and public peering

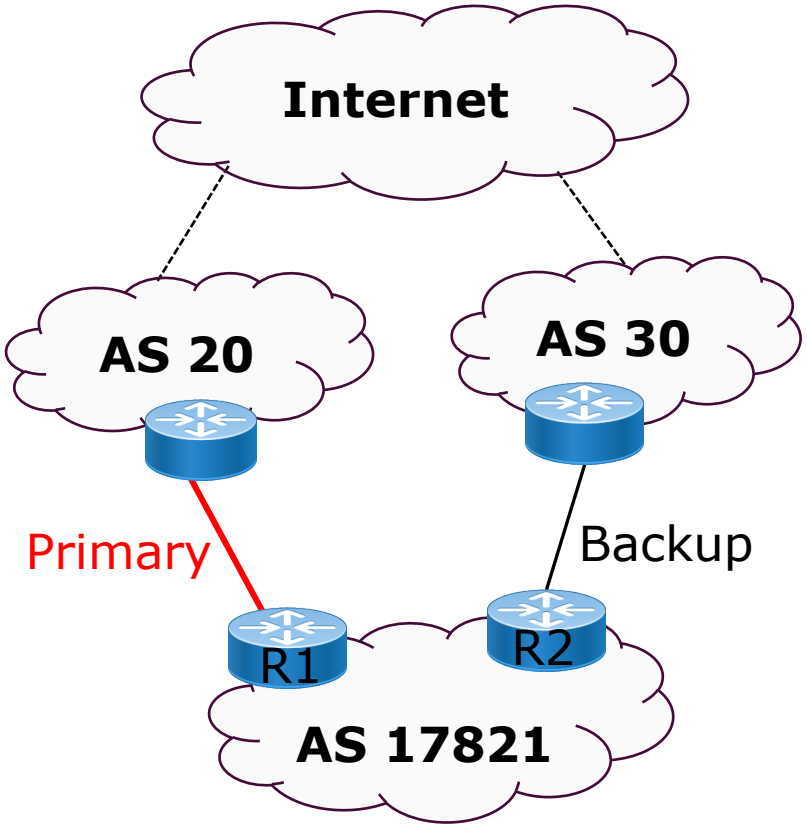


Recap: Path control Attributes

- Inbound Traffic:
  - AS-PATH, MED, Community
- Outbound Traffic:
  - Local Preference

Two Upstream – One backup

- Both incoming and outgoing traffic via R1
- R2 path to be used only if the path via R1 fails
  - AS-PATH to control inbound traffic
  - LOCAL-PREF for outbound



Two Upstream – One backup

- Always announce the aggregate on both!
- R1 (main link) config:

```
router bgp 17821
network 100.100.0.0 mask 255.255.224.0
neighbor 20.20.20.1 remote-as 20
neighbor 20.20.20.1 prefix-list AGGR out
neighbor 20.20.20.1 prefix-list DEF in
!
ip prefix-list AGGR permit 100.100.0.0/19
ip prefix-list DEF permit 0.0.0.0/0
!
ip route 100.100.0.0 255.255.224.0 null0
```

Advertise aggregate in BGP

Prefix-list applied to outbound routes

Prefix-list applied to inbound routes

Define the prefix-lists

Aggregate should exist in the routing table (pull-up route)

Two Upstream – One backup

- R2 (backup) config:

```
router bgp 17821
network 100.100.0.0 mask 255.255.224.0
neighbor 30.30.30.1 remote-as 30
neighbor 30.30.30.1 prefix-list AGGR out
neighbor 30.30.30.1 route-map BACKUP-OUT out
neighbor 30.30.30.1 prefix-list DEF in
neighbor 30.30.30.1 route-map BACKUP-IN in
!
ip prefix-list AGGR permit 121.10.0.0/19
ip prefix-list DEF permit 0.0.0.0/0
!
ip route 100.100.0.0 255.255.224.0 null0
!
route-map BACKUP-OUT permit 10
 set as-path prepend 17821 17821 17821
!
route-map BACKUP-IN permit 10
 set local-preference 80
```

Advertise aggregate in BGP

Route-map applied to outbound routes

Route-map applied to inbound routes

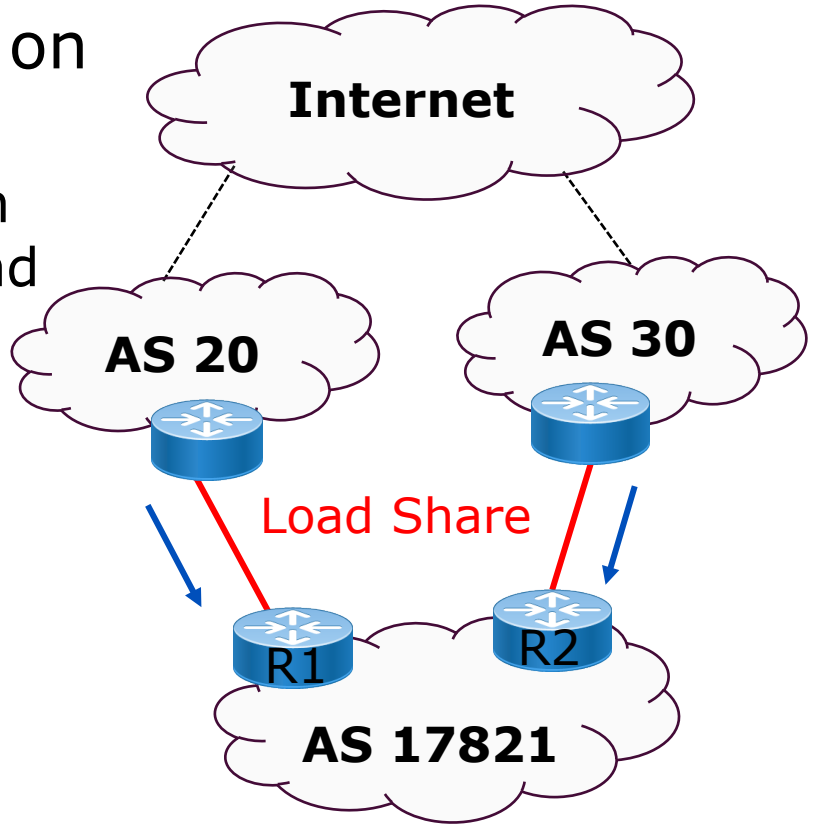
Define the prefix-lists

BACKUP-OUT prepends the AS-PATH for all outbound BGP updates

BACKUP-IN sets lower local-pref for all inbound BGP updates

Two Upstream – Load Sharing (Inbound Traffic)

- Always announce aggregate on both!
  - Announce one sub-aggregate on first, and the other on the second link.
- Requires good address planning
  - Customers need to be assigned from both address blocks



Two Upstream – Load Sharing (Inbound Traffic)

- R1 config:

```
router bgp 17821
network 100.100.0.0 mask 255.255.224.0
network 100.100.0.0 mask 255.255.240.0
neighbor 20.20.20.1 remote-as 20
neighbor 20.20.20.1 prefix-list SUB-A out

!
ip prefix-list SUB-A permit 100.100.0.0/19
ip prefix-list SUB-A permit 100.100.0.0/20

!
ip route 100.100.0.0 255.255.224.0 null0
ip route 100.100.0.0 255.255.240.0 null0
```

Advertise both aggregate and first sub-prefix in BGP

Advertise sub-aggregate along with the aggregate

Sub-aggregate should exist in the routing table (pull-up route)

Two Upstream – Load Sharing (Inbound Traffic)

- R2 config:

```
router bgp 17821
network 100.100.0.0 mask 255.255.224.0
network 100.100.16.0 mask 255.255.240.0
neighbor 30.30.30.1 remote-as 30
neighbor 30.30.30.1 prefix-list SUB-B out

!
ip prefix-list SUB-B permit 100.100.0.0/19
ip prefix-list SUB-B permit 100.100.16.0/20

!
ip route 100.100.0.0 255.255.224.0 null0
ip route 100.100.16.0 255.255.240.0 null0
```

Advertise both aggregate and second sub-prefix in BGP

Advertise sub-aggregate along with the aggregate

Sub-aggregate should exist in the routing table (pull-up route)

Load Sharing – Outbound (Full)

- What about outbound traffic load balancing?
- Case I: Full Internet routes (more memory/CPU)
  - Accept default route from one (AS20)
  - Full routes from the other (AS30)
    - Higher local-pref prefixes originated by AS30 and its immediate neighbors (one AS hop away) – traffic goes via AS30
    - Lower local-pref all other routes (lower than 100) – traffic to these goes via AS20

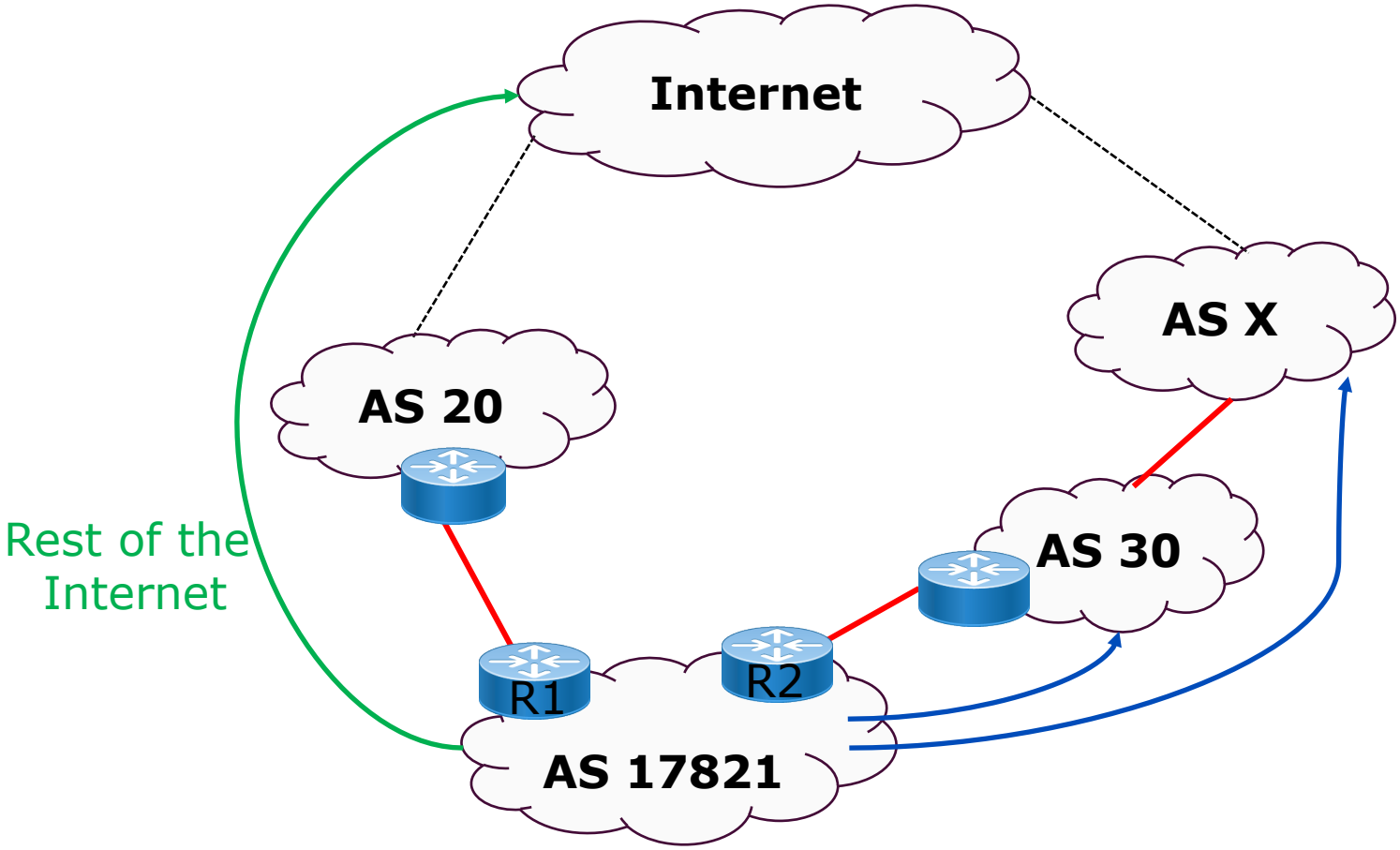
Load Sharing – Outbound (Partial)

- Partial Routes – less HW resources!

- Case II: Partial Internet routes
  - Accept default from AS20
  - Default and full from AS30 (well-connected than AS20)
  - filter to only accept prefixes originated by AS30 and its neighbor ASes (AS-Path ACLs)
    - Higher pref those routes
    - Low pref the default route
    - so that traffic to these goes via AS20

  - Traffic to rest of Internet via AS 20

Load Sharing – Outbound (Partial)



Load Sharing – Outbound (Partial)

- R1 configuration:

```
router bgp 17821
neighbor 20.20.20.1 remote-as 20
neighbor 20.20.20.1 prefix-list DEF in
!
ip prefix-list DEF permit 0.0.0.0/0
!
```

Load Sharing – Outbound (Partial)

- R2 config:

```
router bgp 17821
neighbor 30.30.30.1 remote-as 30
neighbor 30.30.30.1 filter-list 30 in
neighbor 20.20.20.1 prefix-list ALL in
neighbor 30.30.30.1 route-map DEF-LOW in
!
ip prefix-list DEF permit 0.0.0.0/0
!
ip prefix-list ALL deny <bogons-rfc1918>
ip prefix-list ALL permit 0.0.0.0/0 le 32
!
ip as-path access-list 30 permit ^(30_)+$
ip as-path access-list 30 permit ^(30_)+_[0-9]+$
!
route-map DEF-LOW permit 10
 match ip address prefix-list DEF
 set local-preference 90
route-map DEF-LOW permit 20
```

Filter inbound routes with AS-PATH ACL using filter-list

Accept full internet feed except bogon routes and RFC 1918 routes

Purely for redundancy (if path via AS 20 fails)

Accept routes local to and received from AS30 (AS-path prepend included)

Received from AS30 but AS-PATH length of two (its neighbor ASes)

Low-pref default route

Using Communities

- Community attribute provides greater flexibility for traffic shaping than prefix-list
  - Simplifies BGP configuration
  - Greater policy control
- Not sent by default to BGP peers
  - explicitly send (`neighbor x.x.x.x send-community`)
- Can carry policy information
  - Example:
    - `ASN:80 (set local-pref 80)`
    - `ASN:1  (set as-path prepend ASN)`
    - `ASN:888 (set ip next-hop 192.0.2.1 – Cymru bogons)`

COMMUNITY recap

- Used to group prefixes (incoming/outgoing) and apply policies to the communities
  - A prefix can belong to more than one community
- Is (was?) a 32-bit integer
  - Represented as two 16-bit integers [`ASN:number`]
    - Works well for 2-byte ASN
- With 4-byte ASNs
  - Common to see [`private-ASN:number`]
  - RFC 8092 (BGP Large Communities): 96-bit integer
    - [32-bit ASN:32-bit:32-bit]

Setting Communities

```
router bgp 17821
neighbor 20.20.20.1 remote-as 20
neighbor 20.20.20.1 send-community
!
address-family ipv4 unicast
 network 100.100.0.0 mask 255.255.224.0 route-map SET-COMM-AGG
 network 100.100.0.0 mask 255.255.248.0 route-map SET-COMM-3G
 network 100.100.8.0 mask 255.255.248.0 route-map SET-COMM-BB
 network 100.100.16.0 mask 255.255.248.0 route-map SET-COMM-ENT
 network 100.100.24.0 mask 255.255.248.0 route-map SET-COMM-CORP
!
ip route 100.100.0.0 255.255.224.0 null0
ip route 100.100.0.0 255.255.248.0 null0 254
ip route 100.100.8.0 255.255.248.0 null0 254
ip route 100.100.16.0 255.255.248.0 null0 254
ip route 100.100.24.0 255.255.248.0 null0 254
!
```

Setting Communities

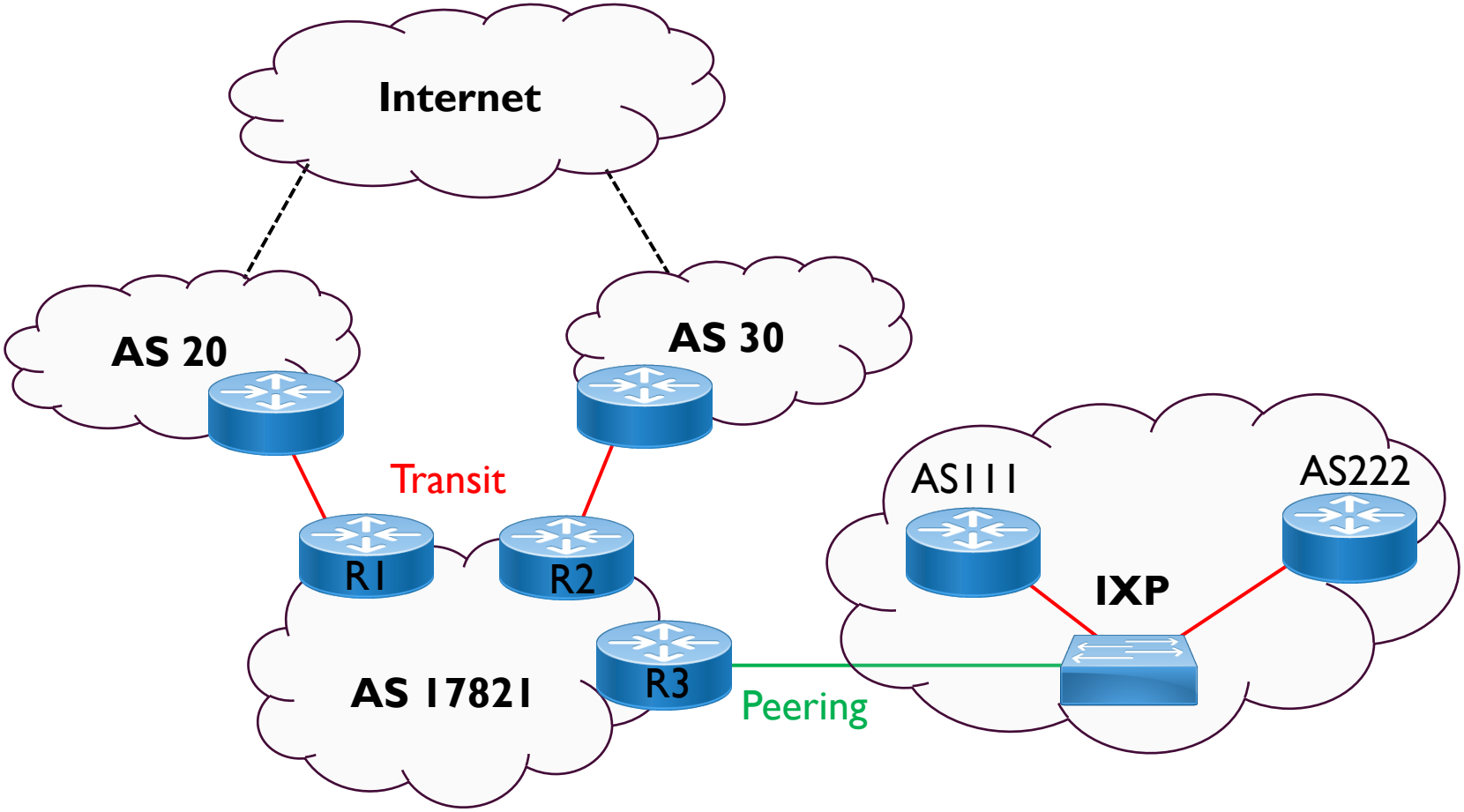
```
!
route-map SET-COMM-AGG permit 10
 set community 100:1000
!
route-map SET-COMM-3G permit 10
 set community 100:1101
!
route-map SET-COMM-BB permit 10
 set community 100:1102
!
route-map SET-COMM-ENT permit 10
 set community 100:1103
!
route-map SET-COMM-CORP permit 10
 set community 100:1104
!
```

Grouping Communities

- We can group communities together using community-list:

```
!
ip community-list 20 permit 100:1000
ip community-list 21 permit 100:1101
ip community-list 22 permit 100:1102
ip community-list 23 permit 100:1103
ip community-list 24 permit 100:1104
!
```

Two Upstream and IXP – Communities



Two Upstream and IXP

- R3 (IXP) configuration:
  - both incoming and outgoing traffic, IXP should be the preferred path!

```
router bgp 17821
 neighbor IX-PEERS peer-group
 neighbor 12.12.12.111 remote-as 111
 neighbor 12.12.12.111 peer-group IX-PEERS
 neighbor 12.12.12.222 remote-as 222
 neighbor 12.12.12.222 peer-group IX-PEERS
!
 address-family ipv4
  neighbor IX-PEERS send-community
  neighbor IX-PEERS remove-private-as
  neighbor IX-PEERS route-map IX-IN in
  neighbor IX-PEERS route-map IX-OUT out
```

Define peer-groups for all IX peers

Add neighbors to the peer group

Define common policies applied to all neighbors on the peer-group
- Send communities
- Remove private ASNs

Apply inbound and outbound routing policies

Two Upstream and IXP

- R3 (IXP) configuration (contd..):

```
!
ip community-list 20 permit 100:1000
ip community-list 21 permit 100:1101
ip community-list 22 permit 100:1102
ip community-list 23 permit 100:1103
ip community-list 24 permit 100:1104
!
route-map IX-IN permit 10
 set local-preference 250
 set community 100:1212 add !(IX ASN)
!
route-map IX-OUT permit 10
 match community 20 21 22 23 24
 set metric 10
!
```

Define the communities

High local-pref for routes received from IX peers (outbound traffic via IX)

Define a community for all routes learned via IXP

Send all our prefixes (aggregates and sub-aggregates)

Set lower MED for all routes sent to IX peers (inbound traffic via IX)

Two Upstream and IXP

- For Transit/Upstream:
  - Tier-1 ISPs (or ISPs who are run properly) use communities to group their regional prefixes
  - Filter based on those to shape outbound traffic to Internet!
    - Ex: receive US routes from one ISP, and Europe routes from the other
  - Example:
    - NTT US – 2914:3000
    - NTT Europe – 2914:3200
    - NTT Asia – 2914:3400
    - NTT South America – 2914:3600

Two Upstream and IXP

- For Inbound traffic:
  - We can use our sub-prefixes to balance incoming traffic
  - Ex: Advertise half of our routes to one, and the other half to the other
    - keep playing until we reach symmetry!
  - But remember to announce the aggregate to both (REDUNDANCY!)

Two Upstream and IXP

- R1 configuration:
  - Let us assume NTT (AS2914) as transit here

```
router bgp 17821
 neighbor 29.29.29.1 remote-as 2914
 neighbor 29.29.29.1 description eBGP with NTT
!
 address-family ipv4
  neighbor 29.29.29.1 send-community
  neighbor 29.29.29.1 route-map NTT-IN in
  neighbor 29.29.29.1 route-map NTT-OUT out
!
! We want Asia, US and SA routes
ip community-list 1 permit 2914:3000 !US
ip community-list 1 permit 2914:3400 !AS
ip community-list 1 permit 2914:3600 !SA
ip community-list 2 permit 2914:3200 !EU
```

- Send communities
- Apply inbound and outbound routing policies

Define communities for NTT global routes

- In this example, we will source US and Asia routes from NTT

Two Upstream and IXP

- R1 configuration (contd..):

```
!
route-map NTT-IN permit 10
 match community 1
 set local-preference 210
route-map NTT-IN permit 20
 match community 2
 set local-preference 50
route-map NTT-IN permit 40
!
route-map NTT-OUT permit 10
 match community 20
 match community 21
 match community 22
!
```

Route-map to influence outbound traffic
- Set higher local-pref for US, Asia, and SA routes (outbound traffic)
- Still lower than IX!

Lower local-pref for EU routes (will prefer the second ISP, but available if that link fails)

Route-map to influence inbound traffic
- Send our aggregate (in case ISP2 fails)
- And half of our sub-prefixes

Two Upstream and IXP

- R2 configuration:
  - Let us assume Zayo (AS6461) as transit here

```
router bgp 17821
 neighbor 64.64.64.1 remote-as 6461
 neighbor 64.64.64.1 description eBGP with Zayo
!
 address-family ipv4
  neighbor 64.64.64.1 send-community
  neighbor 64.64.64.1 route-map ZAYO-IN in
  neighbor 64.64.64.1 route-map ZAYO-OUT out
!
! Zayo Europe routes
ip community-list 3 permit 6461:5996
ip community-list 3 permit 6461:5998
ip community-list 3 permit 6461:5999
! Zayo Global routes
ip community-list 4 permit 6461:5997
```

- Send communities
- Apply inbound and outbound routing policies

Define communities for Zayo global routes

- In this example, we will source EU routes from Zayo

Two Upstream and IXP

- R2 configuration (contd..):

```
!
route-map ZAYO-IN permit 10
 match community 3
 set local-preference 210
route-map ZAYO-IN permit 20
 match community 4
 set local-preference 50
route-map ZAYO-IN permit 40
!
route-map ZAYO-OUT permit 10
 match community 20
 match community 23
 match community 24
!
```

Route-map to influence outbound traffic
- Set higher local-pref for EU routes (outbound traffic)
- Still lower than IX!

Lower local-pref for global routes (NTT is preferred, but will work if that link fails)

Route-map to influence inbound traffic
- Send our aggregate (in case ISP1 fails), and
- other second-half of our sub-prefixes

References

- Philip Smith “Advanced Multihoming”


Questions