

Multihoming: Outbound Traffic Engineering

ISP Workshops



These materials are licensed under the Creative Commons Attribution-NonCommercial 4.0 International license (<http://creativecommons.org/licenses/by-nc/4.0/>)

Last updated 28th September 2020

Acknowledgements

- This material originated from the Cisco ISP/IXP Workshop Programme developed by Philip Smith & Barry Greene
- Use of these materials is encouraged as long as the source is fully acknowledged and this notice remains in place
- Bug fixes and improvements are welcomed
 - Please email *workshop (at) bgp4all.com*

Philip Smith

Service Provider Multihoming

- Previous examples dealt with loadsharing inbound traffic
 - Of primary concern at Internet edge
 - What about outbound traffic?
- Transit Providers strive to balance traffic flows in both directions
 - Balance link utilisation
 - Try and keep most traffic flows symmetric
 - Some edge networks try and do this too
- The original “Traffic Engineering”

Service Provider Multihoming

- Balancing outbound traffic requires inbound routing information
 - Common solution is “full routing table”
 - Rarely necessary
 - Why use the “routing mallet” to try solve loadsharing problems?
 - “Keep It Simple” is often easier (and \$\$\$ cheaper) than carrying N-copies of the full routing table

Service Provider Multihoming

MYTHS!!

Common MYTHS

1. **You need the full routing table to multihome**
 - People who sell router memory would like you to believe this
 - Only true if you are a transit provider
 - Full routing table can be a significant hindrance to multihoming
2. **You need a BIG router to multihome**
 - Router size is related to data rates, not running BGP
 - In reality, to multihome, your router needs to:
 - Have two interfaces,
 - Be able to talk BGP to at least two peers,
 - Be able to handle BGP attributes,
 - Handle at least one prefix
3. **BGP is complex**
 - In the wrong hands, yes it can be! Keep it Simple!

Service Provider Multihoming: Some Strategies

- Take the prefixes you need to aid traffic engineering
 - Look at NetFlow data for popular sites
- Prefixes originated by your immediate neighbours and their neighbours will do more to aid load balancing than prefixes from ASNs many hops away
 - Concentrate on local destinations
- Use default routing as much as possible
 - Or use the full routing table with care

Service Provider Multihoming

□ Examples

- One upstream, one local peer
- One upstream, local exchange point
- Two upstreams, one local peer
- Three upstreams, unequal link bandwidths

□ Require BGP and a public ASN

□ Examples assume that the local network has their own /19 IPv4 address block

Service Provider Multihoming

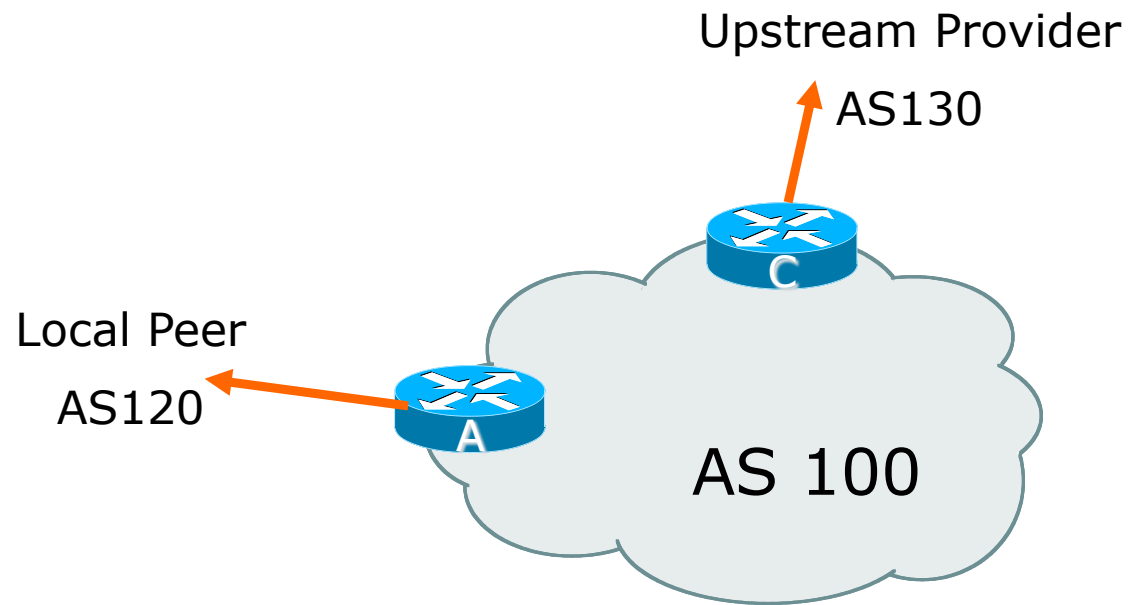


One upstream, one local peer

One Upstream, One Local Peer

- Very common situation in many regions of the Internet
- Connect to upstream transit provider to see the “Internet”
- Connect to the local competition so that local traffic stays local
 - Saves spending valuable \$ on upstream transit costs for local traffic

One Upstream, One Local Peer



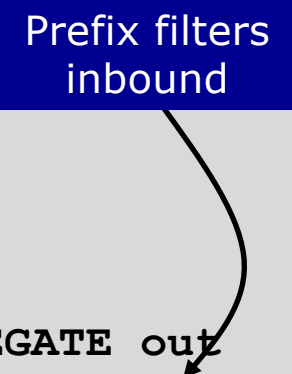
One Upstream, One Local Peer

- Announce /19 aggregate on each link
- Accept default route only from upstream
 - Either 0.0.0.0/0 or a network which can be used as default
- Accept all routes the local peer originates

One Upstream, One Local Peer

□ Router A Configuration

```
router bgp 100
  address-family ipv4
    network 100.64.0.0 mask 255.255.224.0
    neighbor 100.66.10.2 remote-as 120
    neighbor 100.66.10.2 prefix-list AGGREGATE out
    neighbor 100.66.10.2 prefix-list AS120-prefixes in
    neighbor 100.66.10.2 activate
  !
  ip prefix-list AS120-prefixes permit 122.5.16.0/19
  ip prefix-list AS120-prefixes permit 121.240.0.0/20
  !
  ip prefix-list AGGREGATE permit 100.64.0.0/19
  !
  ip route 100.64.0.0 255.255.224.0 null0 250
```

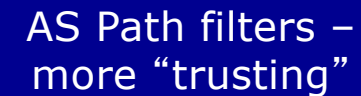


One Upstream, One Local Peer

□ Router A – Alternative Configuration

```
router bgp 100
  address-family ipv4
    network 100.64.0.0 mask 255.255.224.0
    neighbor 100.66.10.2 remote-as 120
    neighbor 100.66.10.2 prefix-list AGGREGATE out
    neighbor 100.66.10.2 filter-list 10 in
    neighbor 100.66.10.2 activate
  !
  ip as-path access-list 10 permit ^(120_)+$
  !
  ip prefix-list AGGREGATE permit 100.64.0.0/19
  !
  ip route 100.64.0.0 255.255.224.0 null0
```

AS Path filters –
more “trusting”



One Upstream, One Local Peer

□ Router C Configuration

```
router bgp 100
  address-family ipv4
    network 100.64.0.0 mask 255.255.224.0
    neighbor 100.66.10.1 remote-as 130
    neighbor 100.66.10.1 prefix-list DEFAULT in
    neighbor 100.66.10.1 prefix-list AGGREGATE out
    neighbor 100.66.10.1 activate
  !
  ip prefix-list AGGREGATE permit 100.64.0.0/19
  ip prefix-list DEFAULT permit 0.0.0.0/0
  !
  ip route 100.64.0.0 255.255.224.0 null0
```

One Upstream, One Local Peer

- ❑ Two configurations possible for Router A
 - Filter-lists assume peer knows what they are doing
 - Prefix-list higher maintenance, but safer
 - Some network operators use **both**
- ❑ Local traffic goes to and from local peer, everything else goes to upstream provider

Aside:

Configuration Recommendations

- Private Peers
 - The peering Network Operators exchange prefixes they originate
 - Sometimes they exchange prefixes from neighbouring ASes too
- Be aware that the private peer EBGP router should carry only the prefixes you want the private peer to receive
 - Otherwise they could point a default route to you and unintentionally transit your backbone

Service Provider Multihoming

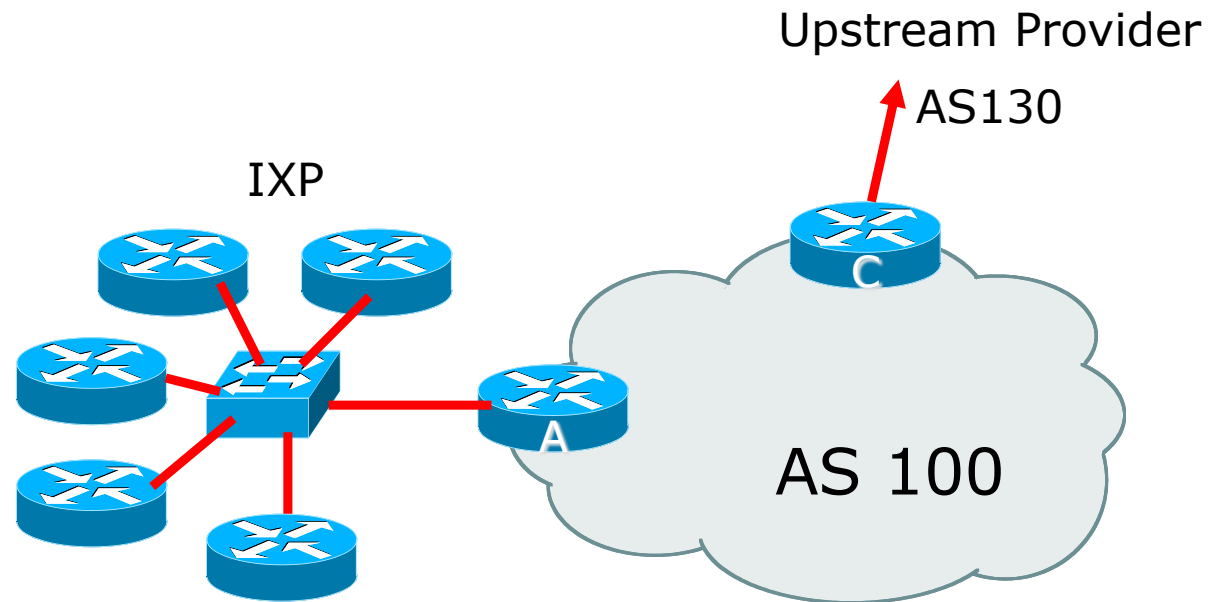


One upstream, Local Exchange Point

One Upstream, Local Exchange Point

- Very common situation in many regions of the Internet
- Connect to upstream transit provider to see the “Internet”
- Connect to the local Internet Exchange Point so that local traffic stays local
 - Saves spending valuable \$ on upstream transit costs for local traffic
- This example is a scaled up version of the previous one

One Upstream, Local Exchange Point



One Upstream, Local Exchange Point

- Announce /19 aggregate to every neighbouring AS
- Accept default route only from upstream
 - Either 0.0.0.0/0 or a network which can be used as default
- Accept all routes originated by IXP peers

One Upstream, Local Exchange Point

□ Router A Configuration

```
interface fastethernet 0/0
  description Exchange Point LAN
  ip address 100.67.10.1 mask 255.255.255.224
!
router bgp 100
  address-family ipv4
    neighbor IXP-PEERS peer-group
    neighbor IXP-PEERS prefix-list AGGREGATE out
    neighbor IXP-PEERS remove-private-AS
    neighbor IXP-PEERS send-community
    neighbor IXP-PEERS route-map SET-LOCAL-PREF in
!
```

...next slide

One Upstream, Local Exchange Point

```
neighbor 100.67.10.2 remote-as 200
neighbor 100.67.10.2 peer-group IXP-PEERS
neighbor 100.67.10.2 prefix-list PEER200 in
neighbor 100.67.10.2 activate
neighbor 100.67.10.3 remote-as 201
neighbor 100.67.10.3 peer-group IXP-PEERS
neighbor 100.67.10.3 prefix-list PEER201 in
neighbor 100.67.10.3 activate
neighbor 100.67.10.4 remote-as 202
neighbor 100.67.10.4 peer-group IXP-PEERS
neighbor 100.67.10.4 prefix-list PEER202 in
neighbor 100.67.10.4 activate
neighbor 100.67.10.5 remote-as 203
neighbor 100.67.10.5 peer-group IXP-PEERS
neighbor 100.67.10.5 prefix-list PEER203 in
neighbor 100.67.10.5 activate
```

...next slide

One Upstream, Local Exchange Point

```
!  
ip prefix-list AGGREGATE permit 100.64.0.0/19  
ip prefix-list PEER200 permit 100.65.0.0/19  
ip prefix-list PEER201 permit 100.66.0.0/19  
ip prefix-list PEER202 permit 100.67.0.0/19  
ip prefix-list PEER203 permit 100.68.128.0/19  
!  
route-map SET-LOCAL-PREF permit 10  
  description Set local preference on all routes to 150  
  set local-preference 150  
!
```

One Upstream, Local Exchange Point

- ❑ Note that Router A does not generate the aggregate for AS100
 - If Router A becomes disconnected from backbone, then the aggregate is no longer announced to the IX
 - BGP failover works as expected
- ❑ Note the inbound route-map which sets the local preference higher than the default
 - This is a visual reminder that BGP Best Path for local traffic will be across the IXP
 - (And allows for future case where operator may need to take BGP routes from their upstream(s))

One Upstream, Local Exchange Point

□ Router C Configuration

```
router bgp 100
  address-family ipv4
    network 100.64.0.0 mask 255.255.224.0
    neighbor 100.66.10.1 remote-as 130
    neighbor 100.66.10.1 prefix-list DEFAULT in
    neighbor 100.66.10.1 prefix-list AGGREGATE out
    neighbor 100.66.10.1 activate
  !
  ip prefix-list AGGREGATE permit 100.64.0.0/19
  ip prefix-list DEFAULT permit 0.0.0.0/0
  !
  ip route 100.64.0.0 255.255.224.0 null0
```

One Upstream, Local Exchange Point

- Note Router A configuration
 - Prefix-list higher maintenance, but safer
 - No generation of AS100 aggregate
- IXP traffic goes to and from local IXP, everything else goes to upstream

Aside:

IXP Configuration Recommendations

- IXP peers
 - The peering Network Operators at the IXP exchange prefixes they originate
 - Sometimes they exchange prefixes from neighbouring ASes too
- Be aware that the IXP border router should carry only the prefixes you want the IXP peers to receive and the destinations you want them to be able to reach
 - Otherwise they could point a default route to you and unintentionally transit your backbone
- If IXP router is at IX, and distant from your backbone
 - Don't originate your address block at your IXP router

Service Provider Multihoming

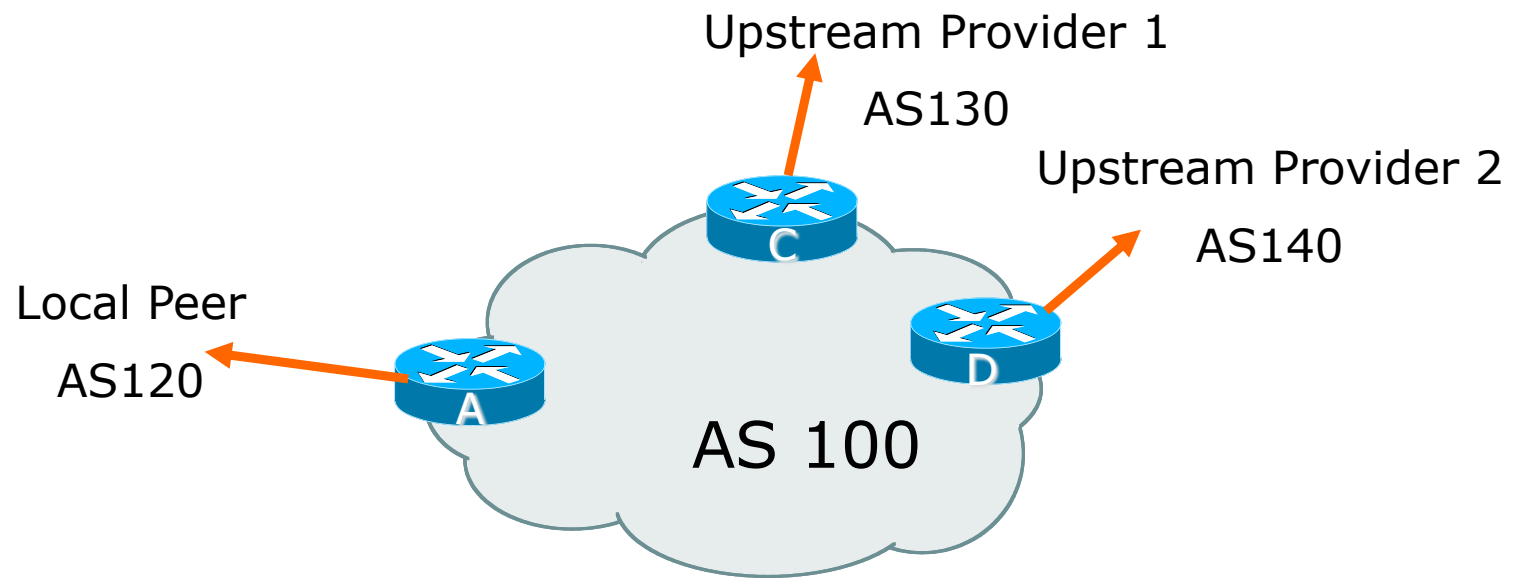


Two upstreams, one local peer

Two Upstreams, One Local Peer

- Connect to both upstream transit providers to see the “Internet”
 - Provides external redundancy and diversity – the reason to multihome
- Connect to the local peer so that local traffic stays local
 - Saves spending valuable \$ on upstream transit costs for local traffic

Two Upstreams, One Local Peer



Two Upstreams, One Local Peer

- Announce /19 aggregate on each link
- Accept default route only from upstreams
 - Either 0.0.0.0/0 or a network which can be used as default
- Accept all routes originated by local peer
- Note separation of Router C and D
 - Single edge router means no redundancy
- Router A
 - Same routing configuration as in example with one upstream and one local peer

Two Upstreams, One Local Peer

□ Router C Configuration

```
router bgp 100
  address-family ipv4
    network 100.64.0.0 mask 255.255.224.0
    neighbor 100.66.10.1 remote-as 130
    neighbor 100.66.10.1 prefix-list DEFAULT in
    neighbor 100.66.10.1 prefix-list AGGREGATE out
    neighbor 100.66.10.1 activate
  !
  ip prefix-list AGGREGATE permit 100.64.0.0/19
  ip prefix-list DEFAULT permit 0.0.0.0/0
  !
  ip route 100.64.0.0 255.255.224.0 null0
```


Two Upstreams, One Local Peer

□ Router D Configuration

```
router bgp 100
  address-family ipv4
    network 100.64.0.0 mask 255.255.224.0
    neighbor 100.66.10.5 remote-as 140
    neighbor 100.66.10.5 prefix-list DEFAULT in
    neighbor 100.66.10.5 prefix-list AGGREGATE out
    neighbor 100.66.10.5 activate
  !
  ip prefix-list AGGREGATE permit 100.64.0.0/19
  ip prefix-list DEFAULT permit 0.0.0.0/0
  !
  ip route 100.64.0.0 255.255.224.0 null0
```

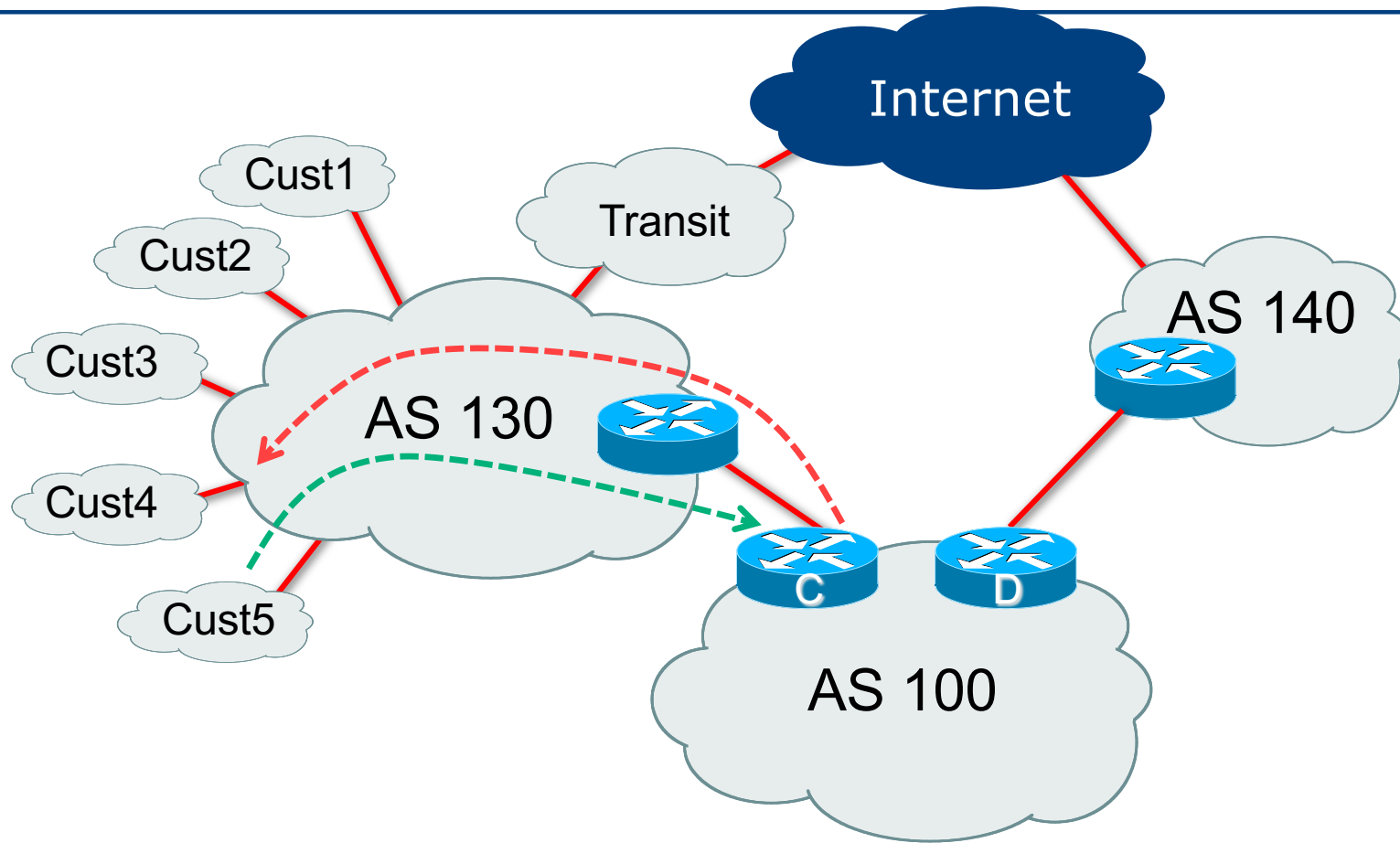
Two Upstreams, One Local Peer

- This is the simple configuration for Router C and D
- Traffic out to the two upstreams will take nearest exit
 - Inexpensive routers required
 - This is not useful in practice especially for international links
 - Loadsharing needs to be better

Two Upstreams, One Local Peer

- Better configuration options:
 - Accept full routing from both upstreams
 - **Expensive & unnecessary!**
 - Accept default from one upstream and some routes from the other upstream
 - **The way to go!**

Loadsharing with different ASes



Two Upstreams, One Local Peer

Full Routes

□ Router C Configuration

```
router bgp 100
  address-family ipv4
    network 100.64.0.0 mask 255.255.224.0
    neighbor 100.66.10.1 remote-as 130
    neighbor 100.66.10.1 prefix-list RFC6890-deny in
    neighbor 100.66.10.1 prefix-list AGGREGATE out
    neighbor 100.66.10.1 route-map AS130-loadshare in
    neighbor 100.66.10.1 activate
```

!

```
ip prefix-list AGGREGATE permit 100.64.0.0/19
```

!

! See <http://tools.ietf.org/html/rfc6890>

...next slide

Allow all prefixes
apart from
RFC6890 blocks

Two Upstreams, One Local Peer

Full Routes

```
ip route 100.64.0.0 255.255.224.0 null0
!
ip as-path access-list 10 permit ^(130_)+$
ip as-path access-list 10 permit ^(130_)+_[0-9]+$
!
route-map AS130-loadshare permit 10
  match ip as-path 10
  set local-preference 120
!
route-map AS130-loadshare permit 20
  set local-preference 80
!
```

Two Upstreams, One Local Peer

Full Routes

□ Router D Configuration

```
router bgp 100
  address-family ipv4
    network 100.64.0.0 mask 255.255.224.0
    neighbor 100.66.10.5 remote-as 140
    neighbor 100.66.10.5 prefix-list RFC6890-deny in
    neighbor 100.66.10.5 prefix-list AGGREGATE out
    neighbor 100.66.10.5 activate
  !
ip prefix-list AGGREGATE permit 100.64.0.0/19
!
```

! See <http://tools.ietf.org/html/rfc6890>

Allow all prefixes
apart from
RFC6890 blocks

Two Upstreams, One Local Peer

Full Routes

- Router C configuration:
 - Accept full routes from AS130
 - Tag prefixes originated by AS130 and AS130's neighbouring ASes with local preference 120
 - Traffic to those ASes will go over AS130 link
 - Remaining prefixes tagged with local preference of 80
 - Traffic to other all other ASes will go over the link to AS140
- Router D configuration same as Router C without the route-map

Two Upstreams, One Local Peer

Full Routes

- Full routes from upstreams
 - Summary of routes received:

ASN	Full Routes		Partial Routes	
AS140	825000	@lp 100		
AS130	30000	@lp 120		
	795000	@lp 80		
Total	1650000			

Two Upstreams, One Local Peer

Full Routes

- Full routes from upstreams
 - Expensive – needs lots of memory and CPU
 - Need to play preference games
 - Previous example is only an example – real life will need improved fine-tuning!
 - Previous example doesn't consider inbound traffic – see earlier in presentation for examples

Two Upstreams, One Local Peer

Partial Routes: Strategy

- Ask one upstream for a default route
 - Easy to originate default towards a BGP neighbour
- Ask other upstream for a full routing table
 - Then filter this routing table based on neighbouring ASN
 - E.g. want traffic to their neighbours to go over the link to that ASN
 - Most of what upstream sends is thrown away
 - Easier than asking the upstream to set up custom BGP filters for you

Two Upstreams, One Local Peer

Partial Routes

Router C Configuration

```
router bgp 100
  address-family ipv4
    network 100.64.0.0 mask 255.255.224.0
    neighbor 100.66.10.1 remote-as 130
    neighbor 100.66.10.1 prefix-list RFC6890-deny in
    neighbor 100.66.10.1 prefix-list AGGREGATE out
    neighbor 100.66.10.1 filter-list 10 in
    neighbor 100.66.10.1 route-map TAG-DEFAULT-low in
    neighbor 100.66.10.1 activate
```

!

Allow all prefixes
apart from
RFC6890 blocks

AS filter-list filters
prefixes based on
origin ASN

Two Upstreams, One Local Peer

Partial Routes

```
ip prefix-list AGGREGATE permit 100.64.0.0/19
ip prefix-list DEFAULT permit 0.0.0.0/0
!
ip route 100.64.0.0 255.255.224.0 null0
!
ip as-path access-list 10 permit ^(130_)+$
ip as-path access-list 10 permit ^(130_)+_[0-9]+$
!
route-map TAG-DEFAULT-low permit 10
  description Default route gets local pref 80
  match ip address prefix-list DEFAULT
  set local-preference 80
!
route-map TAG-DEFAULT-low permit 20
  description All other routes are untouched
!
```

Two Upstreams, One Local Peer

Partial Routes

□ Router D Configuration

```
router bgp 100
  address-family ipv4
    network 100.64.0.0 mask 255.255.224.0
    neighbor 100.66.10.5 remote-as 140
    neighbor 100.66.10.5 prefix-list DEFAULT in
    neighbor 100.66.10.5 prefix-list AGGREGATE out
    neighbor 100.66.10.5 activate
  !
  ip prefix-list AGGREGATE permit 100.64.0.0/19
  ip prefix-list DEFAULT permit 0.0.0.0/0
  !
  ip route 100.64.0.0 255.255.224.0 null0
```

Two Upstreams, One Local Peer

Partial Routes

- Router C configuration:
 - Accept full routes from AS130
 - (or get them to send less)
 - Filter ASNs so only AS130 and AS130's neighbouring ASes are accepted
 - Allow default, and set it to local preference 80
 - Traffic to those ASes will go over AS130 link
 - Traffic to other all other ASes will go over the link to AS140
 - If AS140 link fails, backup via AS130 – and vice-versa

Two Upstreams, One Local Peer

Partial Routes

- Partial routes from upstreams
 - Summary of routes received:

ASN	Full Routes		Partial Routes	
AS140	825000	@lp 100	1	@lp 100
AS130	30000	@lp 120	30000	@lp 100
	795000	@lp 80	1	@lp 80
Total	1650000		30002	

Distributing Default route with IGP

□ Router C IGP Configuration

```
router ospf 100
  default-information originate metric 30
!
```

□ Router D IGP Configuration

```
router ospf 100
  default-information originate metric 10
!
```

- Primary path is via Router D, with backup via Router C
 - Preferred over carrying default route in IBGP
- See the “BGP Case Studies” presentation for more details

Two Upstreams, One Local Peer

Partial Routes

- Partial routes from upstreams
 - Not expensive – only carry the routes necessary for loadsharing
 - Need to filter on AS paths
 - Previous example is only an example – real life will need improved fine-tuning!
 - Previous example doesn't consider inbound traffic – see earlier in presentation for examples

Aside:

Configuration Recommendation

- When distributing internal default by IBGP or OSPF/ISIS
 - Make sure that routers connecting to private peers or to IXPs do **NOT** carry the default route
 - Otherwise they could point a default route to you and unintentionally transit your backbone
 - Simple fix for Private Peer/IXP routers:

```
ip route 0.0.0.0 0.0.0.0 null0
ipv6 route ::/0 null0
```

Service Provider Multihoming



Three upstreams, unequal bandwidths

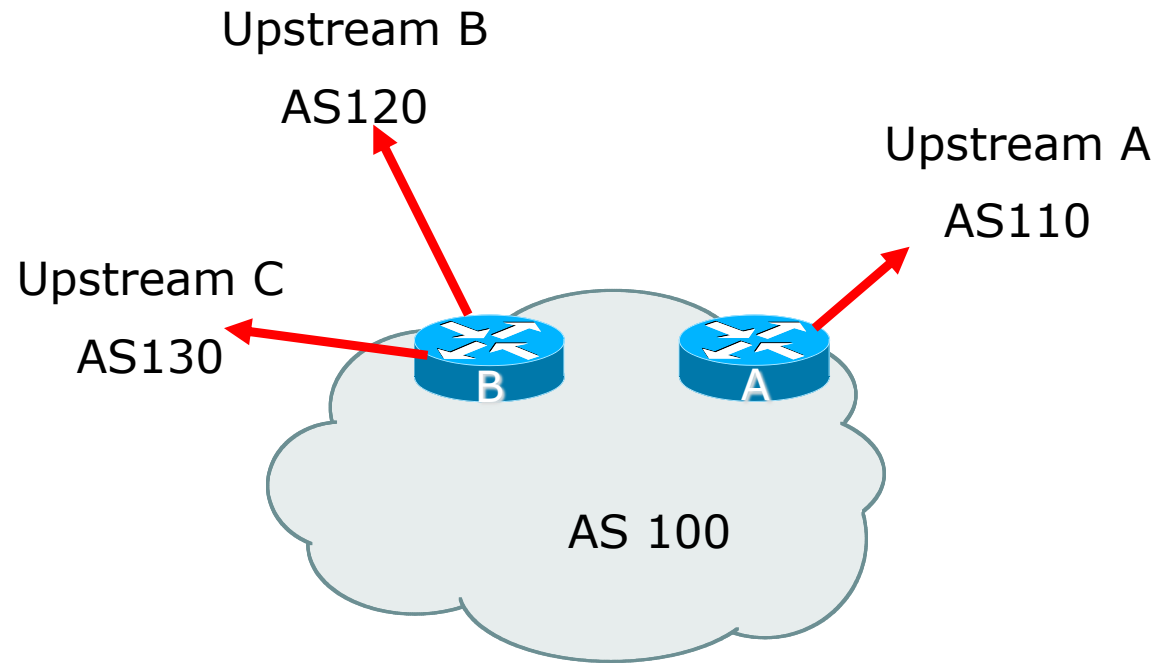
Three upstreams, unequal bandwidths

- This example based on real life complex 3-upstream configuration
- Autonomous System has three upstreams
 - 2.5Gbps to Upstream A
 - 1Gbps to Upstream B
 - 622Mbps to Upstream C
- What is the strategy here?
 - One option is full table from each
 - 3x 825k prefixes \Rightarrow 2475k paths
 - Other option is partial table and defaults from each
 - How??

Strategy

- Two external routers (gives router redundancy)
 - Do **NOT** need three routers for this
- Connect biggest bandwidth to one router
 - Most of inbound and outbound traffic will go here
- Connect the other two links to the second router
 - Provides maximum backup capacity if primary link fails
- Use the biggest link as default
 - Most of the inbound and outbound traffic will go here
- Do the traffic engineering on the two smaller links
 - Focus on regional traffic needs

Diagram



- ❑ Router A has 2.5Gbps link to Upstream A
- ❑ Router B has 1Gbps and 622Mbps links to Upstreams B&C

Outbound load-balancing strategy

- Available BGP feeds from Transit providers:
 - Full table
 - Customer prefixes and default
 - Default Route
- These are the common options on Internet today
 - Very rare for any provider to offer anything different
 - Very rare for any provider to customise BGP feed for a customer

Outbound load-balancing strategy

- Accept only a default route from the provider with the largest connectivity, Upstream A
 - Because most of the traffic is going to use this link
- If Upstream A won't provide a default:
 - Still run BGP with them, but discard all prefixes
 - Point static default route to the upstream link
 - Distribute the default in the IGP
- Request the full table from Upstream B & C
 - Most of this will be thrown away
 - (“Default plus customers” is not enough)

Outbound load-balancing strategy

- How to decide what to keep and what to discard from Upstreams B & C?
 - Most traffic will use Upstream A link — so we need to find a good/useful subset
- Discard prefixes transiting the global transit providers
 - Global transit providers generally appear in most non-local or regional AS-PATHs
- Discard prefixes with Upstream A's ASN in the path
 - Makes more sense for traffic to those destinations to go via the link to Upstream A

Outbound load-balancing strategy

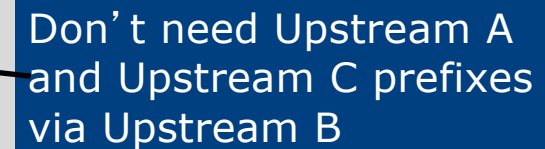
- Global Transit (Tier-1) Providers at the time of this exercise included:

209	CenturyLink	(Qwest)
701	VerizonBusiness	(UUNET)
1239	Softbank	(Sprint)
2914	NTT America	(NTT/Verio)
3549	Level 3	(GlobalCrossing)
3356	Level 3	
3561	CenturyLink	(Savvis, ex C&W)
7018	AT&T	

Upstream B peering Inbound AS-PATH filter

```
ip as-path access-list 1 deny _209_  
ip as-path access-list 1 deny _701_  
ip as-path access-list 1 deny _1239_  
ip as-path access-list 1 deny _3356_  
ip as-path access-list 1 deny _3549_  
ip as-path access-list 1 deny _3561_  
ip as-path access-list 1 deny _2914_  
ip as-path access-list 1 deny _7018_  
!  
ip as-path access-list 1 deny _ISPA_  
ip as-path access-list 1 deny _ISPC_  
!  
ip as-path access-list 1 permit _ISPB$  
ip as-path access-list 1 permit _ISPB_[0-9]+$  
ip as-path access-list 1 permit _ISPB_[0-9]+_[0-9]+$  
ip as-path access-list 1 permit _ISPB_[0-9]+_[0-9]+_[0-9]+$  
ip as-path access-list 1 deny .*
```

Don't need Upstream A
and Upstream C prefixes
via Upstream B



Outbound load-balancing strategy: Upstream B peering configuration

- Part 1: Dropping Global Transit Provider prefixes
 - This can be fine-tuned if traffic volume is not sufficient
 - (More prefixes in = more traffic out)
- Part 2: Dropping prefixes transiting Upstream A & C network
- Part 3: Permitting prefixes from Upstream B, their BGP neighbours, and their neighbours, and their neighbours
 - More AS_PATH permit clauses, the more prefixes allowed in, the more egress traffic
 - Too many prefixes in will mean more outbound traffic than the link to Upstream B can handle

Outbound load-balancing strategy

- Similar AS-PATH filter can be built for the Upstream C BGP peering
- If the same prefixes are heard from both Upstream B and C, then establish proximity of their origin AS to Upstream B or C
 - e.g. Upstream B might be in Japan, with the neighbouring ASN in Europe, yet Upstream C might be in Europe
 - Transit to the ASN via Upstream C makes more sense in this case

Inbound load-balancing strategy

- The largest outbound link should announce just the aggregate
- The other links should announce:
 - The aggregate with AS-PATH prepend
 - Subprefixes of the aggregate, chosen according to traffic volumes to those subprefixes, and according to the services on those subprefixes
- Example:
 - Link to Upstream B could be used just for Broadband customers — so number all such customers out of one contiguous subprefix
 - Link to Upstream C could be used just for commercial leased line customers — so number all such customers out of one contiguous subprefix

Router A: EBGP Configuration Example

```
router bgp 100
  address-family ipv4
    network 100.64.0.0 mask 255.255.224.0
    neighbor 100.66.10.1 remote 110
    neighbor 100.66.10.1 prefix-list DEFAULT in
    neighbor 100.66.10.1 prefix-list AGGREGATE out
    neighbor 100.66.10.1 activate
  !
ip prefix-list DEFAULT permit 0.0.0.0/0
ip prefix-list AGGREGATE permit 100.64.0.0/19
!
```


Router B: EBGP Configuration Example

```
router bgp 100
  address-family ipv4
    network 100.64.0.0 mask 255.255.224.0
    neighbor 100.66.1.1 remote 120
    neighbor 100.66.1.1 filter-list 1 in
    neighbor 100.66.1.1 prefix-list ISP-B out
    neighbor 100.66.1.1 route-map to-ISP-B out
    neighbor 100.66.1.1 activate
    neighbor 100.67.2.1 remote 130
    neighbor 100.67.2.1 filter-list 2 in
    neighbor 100.67.2.1 prefix-list ISP-C out
    neighbor 100.67.2.1 route-map to-ISP-C out
    neighbor 100.67.2.1 activate
  !
ip prefix-list AGGREGATE permit 100.64.0.0/19
!
```

...next slide

Router B: EBGP Configuration Example

```
ip prefix-list ISP-B permit 100.64.0.0/19
ip prefix-list ISP-B permit 100.64.0.0/21 ← /21 to ISP B
!                                     "adsl customers"
ip prefix-list ISP-C permit 100.64.0.0/19
ip prefix-list ISP-C permit 100.64.28.0/22 ← /22 to ISP C
!                                     "biz customers"
route-map to-ISP-B permit 10
  match ip address prefix-list AGGREGATE
  set as-path prepend 100 ← e.g. Single prepend
!                                     on ISP B link
route-map to-ISP-B permit 20
!
route-map to-ISP-C permit 10
  match ip address prefix-list AGGREGATE
  set as-path prepend 100 100 ← e.g. Dual prepend
!                                     on ISP C link
route-map to-ISP-C permit 20
```

What about outbound backup?

- We have:
 - Default route from Upstream A by EBGP
 - Mostly discarded full table from Upstreams B&C
- Strategy:
 - Originate default route by OSPF on Router A (with metric 10) — link to Upstream A
 - Originate default route by OSPF on Router B (with metric 30) — links to Upstreams B & C
 - Plus on Router B:
 - Static default route to Upstream B with distance 240
 - Static default route to Upstream C with distance 245
 - When link goes down, static route is withdrawn

Outbound backup: steady state

- Steady state (all links up and active):
 - Default route is to Router A — OSPF metric 10
 - (Because default learned by EBGP \Rightarrow default is in RIB \Rightarrow OSPF will originate default)
 - Backup default is to Router B — OSPF metric 20
 - EBGP prefixes learned from upstreams distributed by IBGP throughout backbone
 - (Default can be filtered in IBGP to avoid “RIB failure error”)

Outbound backup: failure examples

- Link to Upstream A down, to Upstreams B&C up:
 - Default route is to Router B — OSPF metric 20
 - (EBGP default gone from RIB, so OSPF on Router A withdraws the default)
- Above is true if link to B or C is down as well
- Link to Upstreams B & C down, link to Upstream A is up:
 - Default route is to Router A — OSPF metric 10
 - (static defaults on Router B removed from RIB, so OSPF on Router B withdraws the default)
- See the “BGP Case Studies” for a more detailed example

Other considerations

- ❑ Default route should not be propagated to devices terminating non-transit peers and customers
- ❑ Rarely any need to carry default in IBGP
 - Best to filter out default in IBGP mesh peerings
 - Or tag default route with **no-advertise** community when learned on EBGP peerings
- ❑ Still carry other EBGP prefixes across IBGP mesh
 - Otherwise routers will follow default route rules resulting in suboptimal traffic flow
 - Not a big issue because not carrying full table

Router A: IBGP Configuration Example

- Filtering default route out of IBGP sessions

```
router bgp 100
  address-family ipv4
    network 100.64.0.0 mask 255.255.224.0
    neighbor IBGP peer-group
    neighbor IBGP remote-as 100
    neighbor IBGP prefix-list IBGP-FILTER out
    neighbor 100.64.0.2 peer-group IBGP
    neighbor 100.64.0.2 activate
    neighbor 100.64.0.3 peer-group IBGP
    neighbor 100.64.0.3 activate
  !
  ip prefix-list IBGP-FILTER deny 0.0.0.0/0
  ip prefix-list IBGP-FILTER permit 0.0.0.0/0 le 32
  !
```

Router A: EBGP Configuration Example

- Preferred! Tag default route with **no-advertise** community

```
router bgp 100
  address-family ipv4
    network 100.64.0.0 mask 255.255.224.0
    neighbor 100.66.10.1 remote 110
    neighbor 100.66.10.1 route-map AS110-in in
    neighbor 100.66.10.1 prefix-list AGGREGATE out
    neighbor 100.66.10.1 activate
  !
  ip prefix-list DEFAULT permit 0.0.0.0/0
  ip prefix-list AGGREGATE permit 100.64.0.0/19
  !
  route-map AS110-in permit 10
    match ip address prefix-list DEFAULT
    set community no-advertise
  !
```


Three upstreams, unequal bandwidths:

Summary

- Example based on many deployed working multihoming/loadbalancing topologies
- Many variations possible — this one is:
 - Easy to tune
 - Light on border router resources
 - Light on backbone router infrastructure
 - Sparse BGP table \Rightarrow faster convergence

Multihoming: Outbound Traffic Engineering



ISP Workshops