BGP Scaling Techniques

ISP/IXP Workshops

BGP Scaling Techniques

- Original BGP specification and implementation was fine for the Internet of the early 1990s
  - But didn't scale
- Issues as the Internet grew included:
  - Scaling the iBGP mesh beyond a few peers?
  - Implement new policy without causing flaps and route churning?
  - Keep the network stable, scalable, as well as simple?

BGP Scaling Techniques

- Current Best Practice Scaling Techniques
  - Route Refresh
  - Peer-groups
  - Route Reflectors (and Confederations)
- Deprecated Scaling Techniques
  - Soft Reconfiguration
  - Route Flap Damping

Dynamic Reconfiguration

Non-destructive policy changes

Route Refresh

- Policy Changes:

  Hard BGP peer reset required after every policy change because the router does not store prefixes that are rejected by policy

- Hard BGP peer reset:

  Tears down BGP peering

  Consumes CPU

  Severely disrupts connectivity for all networks

- Solution:

  Route Refresh

Route Refresh Capability

- Facilitates non-disruptive policy changes
- No configuration is needed
  - Automatically negotiated at peer establishment
- No additional memory is used
- Requires peering routers to support "route refresh capability" – RFC2918
- clear ip bgp x.x.x.x [soft] in tells peer to resend full BGP announcement
- clear ip bgp x.x.x.x [soft] out resends full BGP announcement to peer

Dynamic Reconfiguration

- Use Route Refresh capability
  - Supported on virtually all routers
  - find out from "show ip bgp neighbor"
  - Non-disruptive, "Good For the Internet"

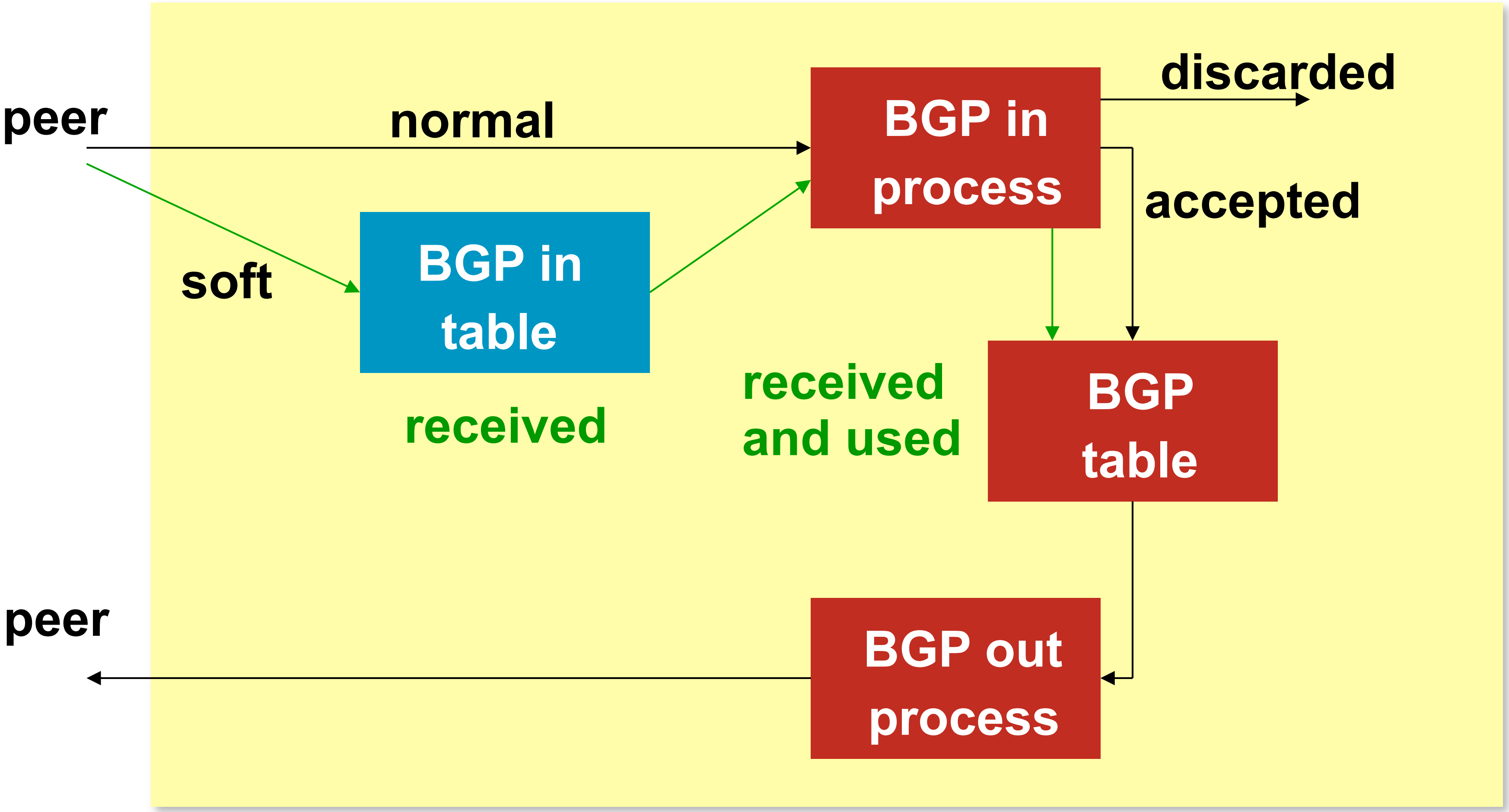
- Only hard-reset a BGP peering as a last resort

Consider the impact to be equivalent to a router reboot

Soft Reconfiguration

- Now deprecated — but:
- Router normally stores prefixes which have been received from peer after policy application
  - Enabling soft-reconfiguration means router also stores prefixes/ attributes received prior to any policy application
  - Uses more memory to keep prefixes whose attributes have been changed or have not been accepted
- Only useful now when operator requires to know which prefixes have been sent to a router prior to the application of any inbound policy

Soft Reconfiguration



Configuring Soft Reconfiguration

```
router bgp 100
 neighbor 1.1.1.1 remote-as 101
 neighbor 1.1.1.1 route-map infilter in
 neighbor 1.1.1.1 soft-reconfiguration inbound
! Outbound does not need to be configured !
```

- Then when we change the policy, we issue an exec command

  clear ip bgp 1.1.1.1 soft [in | out]

- Note:

  When “soft reconfiguration” is enabled, there is no access to the route refresh capability

  clear ip bgp 1.1.1.1 [in | out] will also do a soft refresh

Peer Groups

Peer Groups

- Problem – how to scale iBGP
  - Large iBGP mesh slow to build
  - iBGP neighbours receive the same update
  - Router CPU wasted on repeat calculations
- Solution – peer-groups
  - Group peers with the same outbound policy
  - Updates are generated once per group

Peer Groups – Advantages

- Makes configuration easier
- Makes configuration less prone to error
- Makes configuration more readable
- Lower router CPU load
- iBGP mesh builds more quickly
- Members can have different inbound policy
- Can be used for eBGP neighbours too!

Configuring a Peer Group

```
router bgp 100
 neighbor ibgp-peer peer-group
 neighbor ibgp-peer remote-as 100
 neighbor ibgp-peer update-source loopback 0
 neighbor ibgp-peer send-community
 neighbor ibgp-peer route-map outfilter out
 neighbor 1.1.1.1 peer-group ibgp-peer
 neighbor 2.2.2.2 peer-group ibgp-peer
 neighbor 2.2.2.2 route-map  infilter in
 neighbor 3.3.3.3 peer-group ibgp-peer
```

! note how 2.2.2.2 has different inbound filter from peer-group !

Configuring a Peer Group

```
router bgp 100
 neighbor external-peer peer-group
 neighbor external-peer send-community
 neighbor external-peer route-map set-metric out
 neighbor 160.89.1.2 remote-as 200
 neighbor 160.89.1.2 peer-group external-peer
 neighbor 160.89.1.4 remote-as 300
 neighbor 160.89.1.4 peer-group external-peer
 neighbor 160.89.1.6 remote-as 400
 neighbor 160.89.1.6 peer-group external-peer
 neighbor 160.89.1.6 filter-list infilter in
```

Peer Groups

- Always configure peer-groups for iBGP
  - Even if there are only a few iBGP peers
  - Easier to scale network in the future
- Consider using peer-groups for eBGP
  - Especially useful for multiple BGP customers using same AS (RFC2270)
  - Also useful at Exchange Points where ISP policy is generally the same to each peer
- Peer-groups are essentially obsoleted
  - But are still widely considered best practice
  - Replaced by update-groups (internal coding – not configurable)
  - Enhanced by peer-templates (allowing more complex constructs)

BGP Peer Templates

- Used to group common configurations
  - Uses peer-group style of syntax
  - Much more flexible than peer-groups
- Hierarchical policy configuration mechanism
  - A peer-template may be used to provide policy configurations to an individual neighbor, a peer-group or another peer-template
  - The more specific user takes precedence if policy overlaps
    - individual neighbor → peer-group → peer-template

BGP Peer Templates

- First appeared in 12.0(24)S and 12.2(25)S
  - Integrated in 12.3T, now in 12.4
- Two types of templates
- Session Template
  - Can inherit from one session-template
  - Used to configure parameters which are independent of the AFI (address-family-identifier)
  - e.g. remote-as, ebgp-multihop, passwords, etc
- Peer/policy Template
  - Can inherit from multiple peer/policy templates
  - Used to configure AFI dependant parameters
  - Filters, next-hop-self, route-reflector-client, etc

Session Template

```
router bgp 100
 !
 template peer-session all-sessions
  version 4
  timers 10 30
 exit-peer-session
 !
 template peer-session iBGP-session
  remote-as 100
  password 7
   022F021B12091A61484B0A0B1C07064B180C23
   38642C26272B1D
  description iBGP peer
  update-source Loopback0
  inherit peer-session all-sessions
 exit-peer-session
!
 template peer-session eBGP-session
  description eBGP peer
  ebgp-multihop 2
  inherit peer-session all-sessions
 exit-peer-session
!
```

```
 !
 no synchronization
 bgp log-neighbor-changes
 neighbor 1.1.1.1 inherit peer-session iBGP-session
 neighbor 1.1.1.2 inherit peer-session iBGP-session
 neighbor 1.1.1.3 inherit peer-session iBGP-session
 neighbor 10.1.1.1 remote-as 1442
 neighbor 10.1.1.1 inherit peer-session eBGP-session
 neighbor 10.1.1.2 remote-as 6445
 neighbor 10.1.1.2 inherit peer-session eBGP-session
 no auto-summary
!
```

- 1.1.1.1 → 1.1.1.3 are configured with commands from all-sessions and iBGP-session
- 10.1.1.1 → 10.1.1.2 are configured with commands from all-sessions and eBGP-session

Policy Template

```
router bgp 100
 template peer-policy all-peers
  prefix-list deny-martians in
  prefix-list deny-martians out
 exit-peer-policy
 !
 template peer-policy external-policy
  remove-private-as
  maximum-prefix 1000
  inherit peer-policy all-peers 10
 exit-peer-policy
 !
 template peer-policy full-routes-customer
  route-map full-routes out
  inherit peer-policy external-policy 10
 exit-peer-policy
 !
 template peer-policy partial-routes-
  customer
  route-map partial-routes out
  inherit peer-policy external-policy 10
 exit-peer-policy
 !
 template peer-policy internal-policy
  send-community
  inherit peer-policy all-peers 10
 exit-peer-policy
 !
 template peer-policy RRC
  route-reflector-client
  inherit peer-policy internal-policy 10
 exit-peer-policy

 neighbor 1.1.1.1 inherit peer-policy internal-policy
 neighbor 1.1.1.2 inherit peer-policy RRC
 neighbor 1.1.1.3 inherit peer-policy RRC
 neighbor 10.1.1.1 inherit peer-policy full-routes-customer
 neighbor 10.1.1.2 inherit peer-policy partial-routes-customer
```

Policy Template

```
!
template peer-policy foo
 filter-list 100 out
 prefix-list foo-filter out
 inherit peer-policy all-peers 10
exit-peer-policy
!
template peer-policy bar
 prefix-list bar-filter out
exit-peer-policy
!
template peer-policy seq_example
 inherit peer-policy bar 20
 inherit peer-policy foo 10
exit-peer-policy
!
neighbor 10.1.1.3 remote-as 200
neighbor 10.1.1.3 inherit peer-policy seq_example
```

```
Router#show ip bgp neighbors 10.1.1.3 policy
 Neighbor: 10.1.1.3, Address-Family: IPv4
   Unicast
 Inherited polices:
  prefix-list deny-martians in
  prefix-list bar-filter out
  filter-list 100 out
Router#
```

- A policy template can inherit from multiple templates
- Seq # determines priority if overlapping policies
  - Higher seq # has priority

BGP Update Groups

- First appeared in 12.0(24)S and 12.2(25)S

  Integrated in 12.3T, now in 12.4

- The Problem: peer-groups help BGP scale but customers do not always use peer-groups, especially with eBGP peers
- The Solution: treat peers with a common outbound policy as if they are in a peer-group

BGP Update Groups

- Peers with a common outbound policy are placed into an update-group
- Reduce CPU cycles
  - BGP builds updates for one member of the update-group
  - Updates are then replicated to the other members of the update-group
- Same benefit of configuring peer-groups but without the configuration hassle
- Peer-groups may still be used
  - Reduces config size
  - No longer makes a difference in convergence/scalability

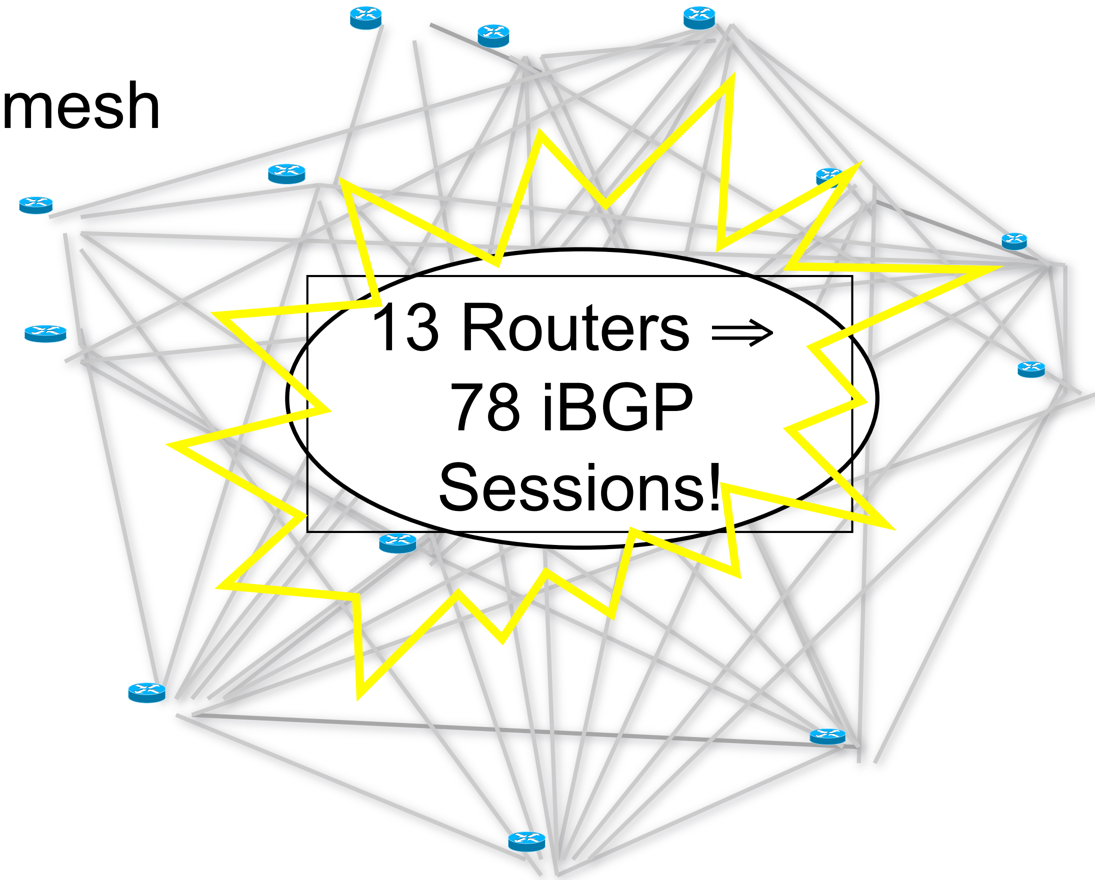
Route Reflectors

Scaling the iBGP mesh

Scaling iBGP mesh

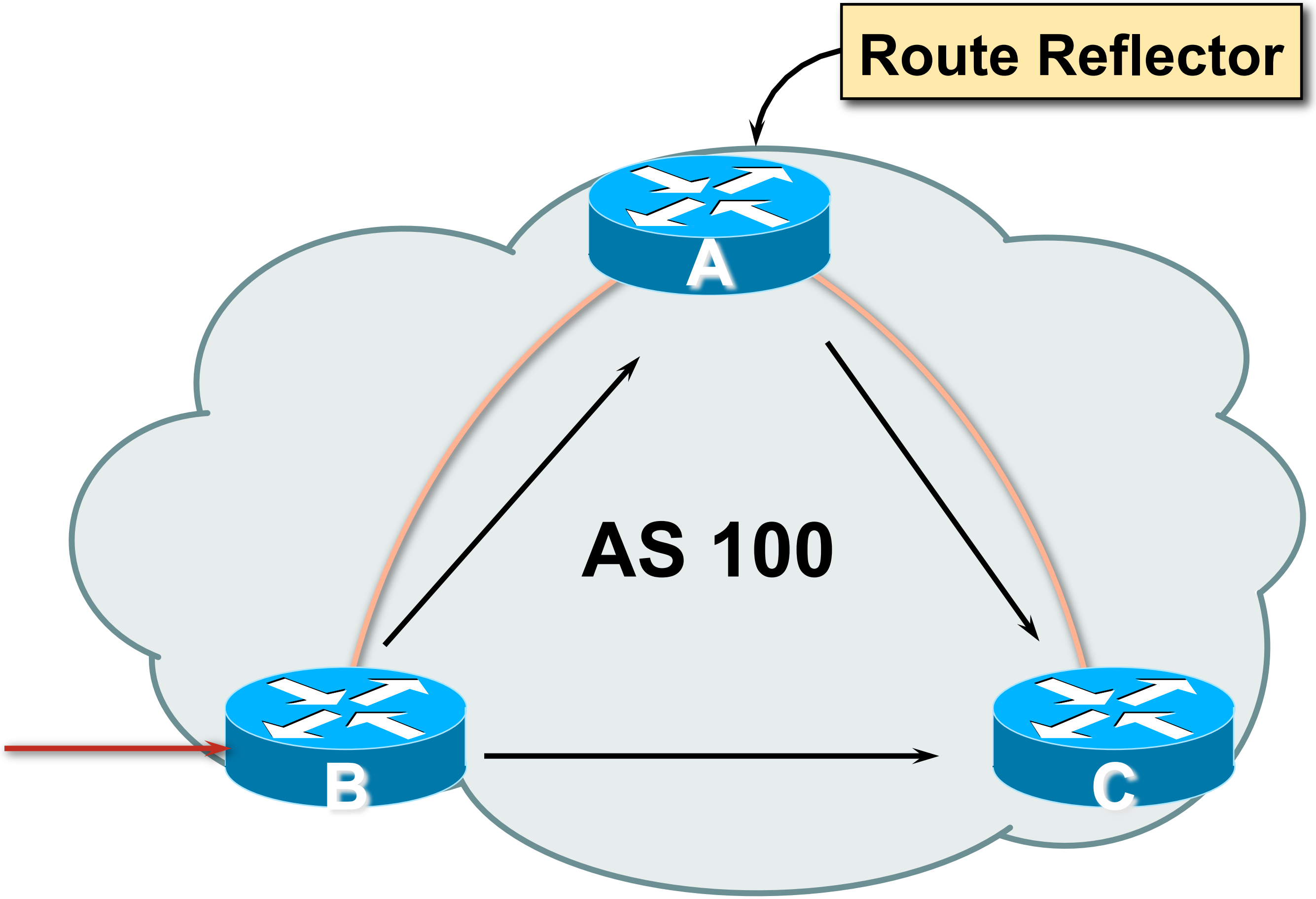
- Avoid ½n(n-1) iBGP mesh

n=1000 ⇒ nearly
half a million
ibgp sessions!



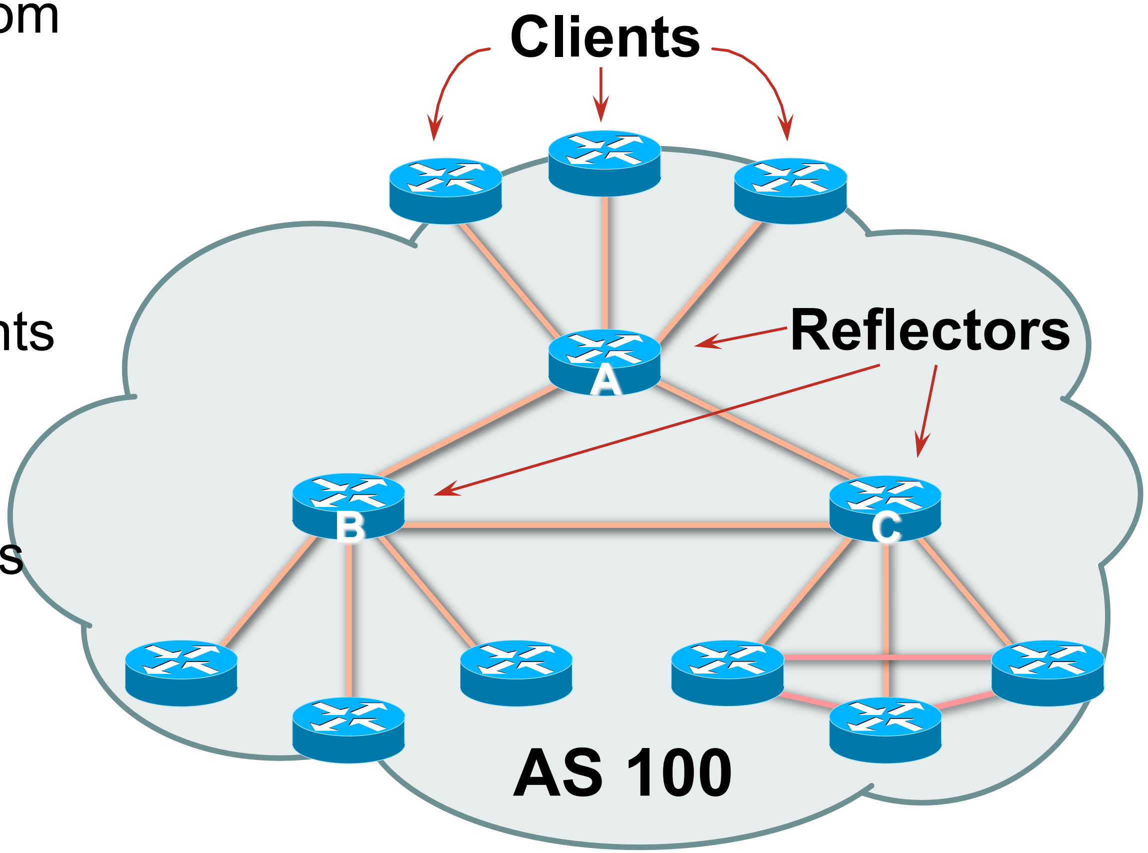
- Two solutions
  - Route reflector – simpler to deploy and run
  - Confederation – more complex, has corner case advantages

Route Reflector: Principle



Route Reflector

- Reflector receives path from clients and non-clients
- Selects best path
- If best path is from client, reflect to other clients and non-clients
- If best path is from non-client, reflect to clients only
- Non-meshed clients
- Described in RFC4456



Route Reflector Topology

- Divide the backbone into multiple clusters
- At least one route reflector and few clients per cluster
- Route reflectors are fully meshed
- Clients in a cluster could be fully meshed
- Single IGP to carry next hop and local routes

Route Reflectors: Loop Avoidance

- Originator_ID attribute

  Carries the RID of the originator of the route in the local AS (created by the RR)

- Cluster_list attribute

  The local cluster-id is added when the update is sent by the RR

  Cluster-id is router-id (address of loopback)

  Do NOT use **`bgp cluster-id x.x.x.x`**

Route Reflectors: Redundancy

- Multiple RRs can be configured in the same cluster – not advised!

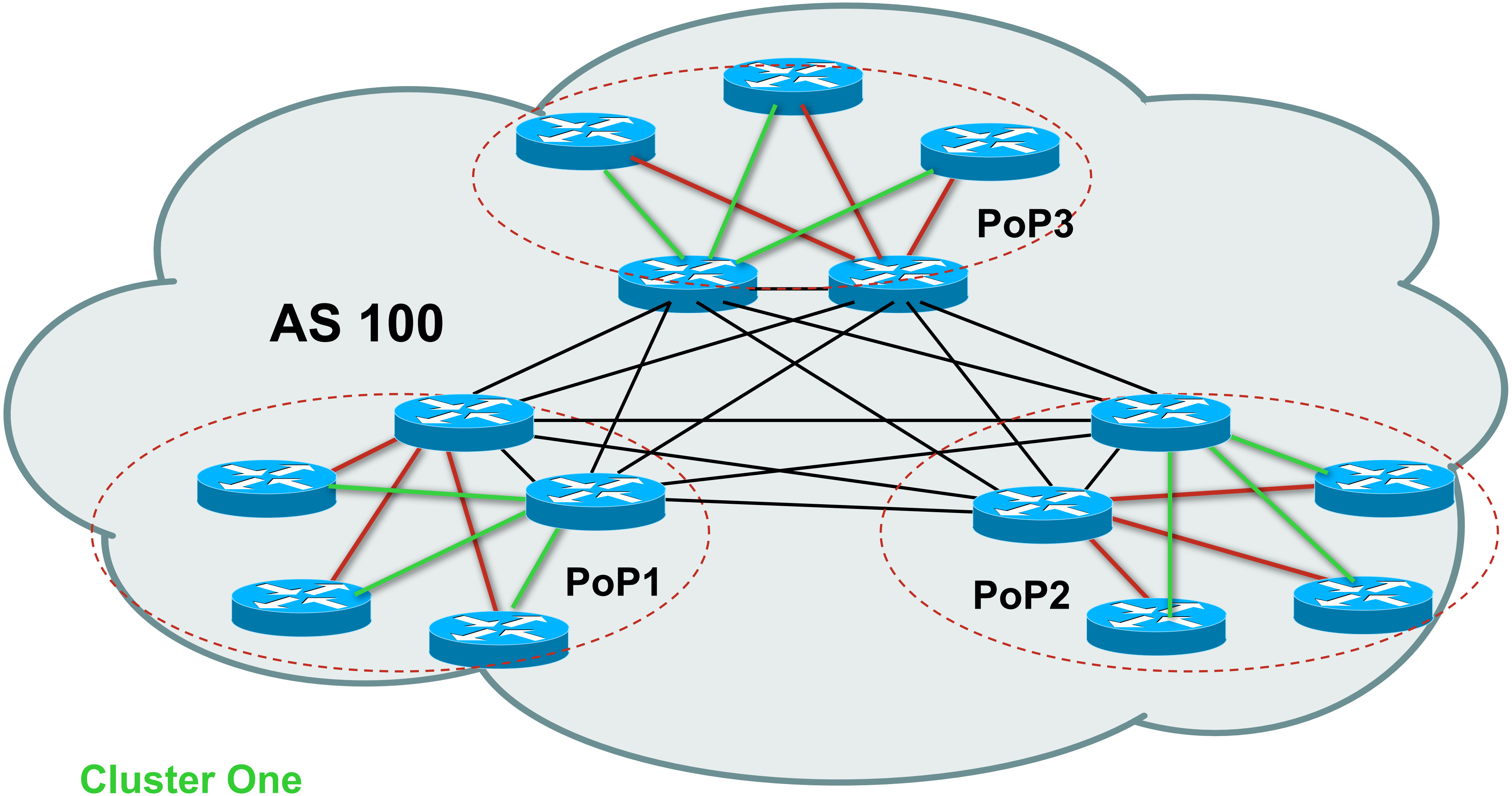
  All RRs in the cluster must have the same cluster-id (otherwise it is a different cluster)

- A router may be a client of RRs in different clusters

  Common today in ISP networks to overlay two clusters – redundancy achieved that way

  → Each client has two RRs = redundancy

Route Reflectors: Redundancy



Cluster One

Cluster Two

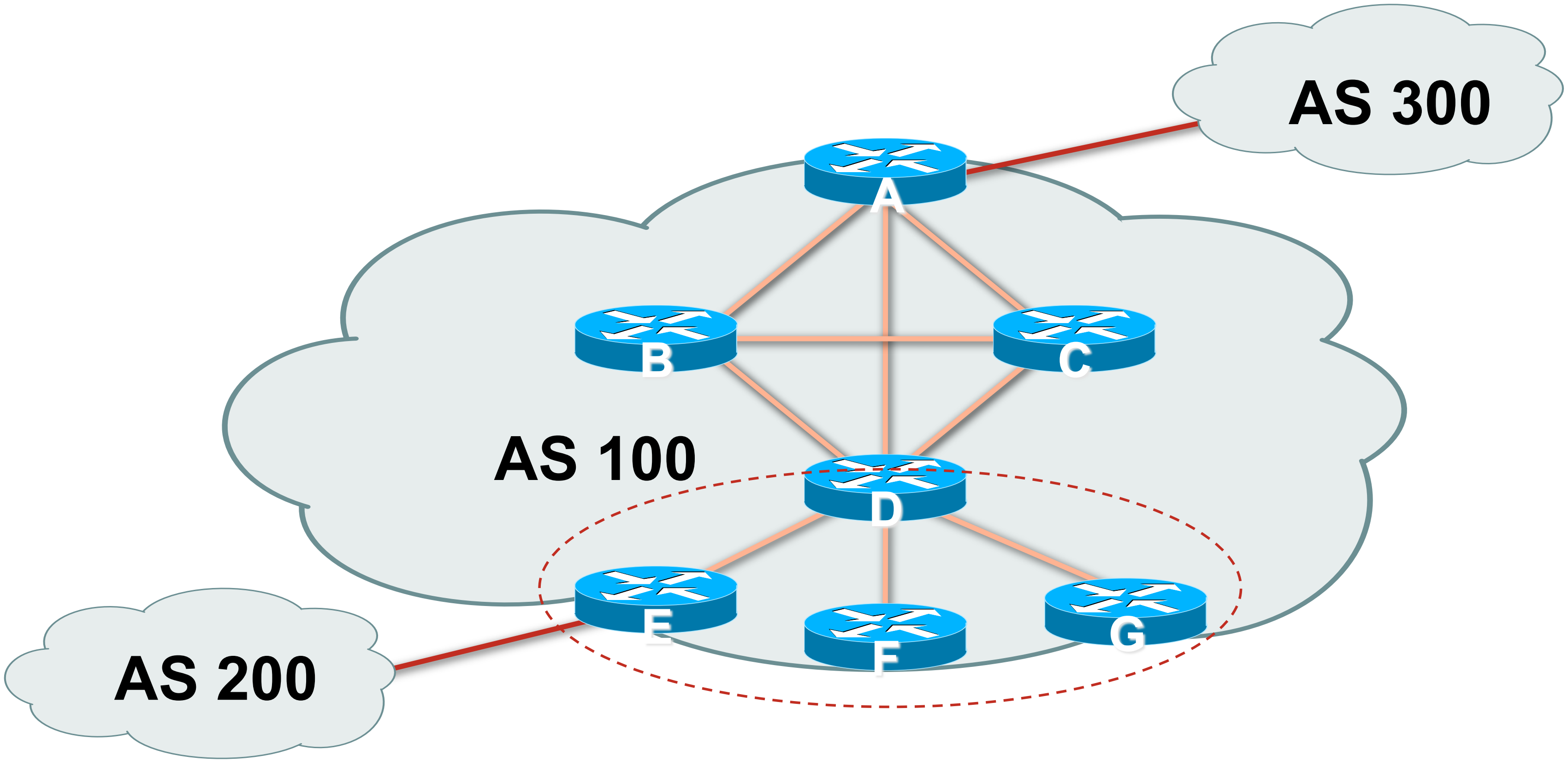
Route Reflector: Benefits

- Solves iBGP mesh problem
- Packet forwarding is not affected
- Normal BGP speakers co-exist
- Multiple reflectors for redundancy
- Easy migration
- Multiple levels of route reflectors

Route Reflectors: Migration

- Where to place the route reflectors?
  - Follow the physical topology!
  - This will guarantee that the packet forwarding won't be affected
- Configure one RR at a time
  - Eliminate redundant iBGP sessions
  - Place one RR per cluster

Route Reflectors: Migration



- Migrate small parts of the network, one part at a time.

Configuring a Route Reflector

- Router D configuration:

```
router bgp 100
 ...
 neighbor 1.2.3.4 remote-as 100
 neighbor 1.2.3.4 route-reflector-client
 neighbor 1.2.3.5 remote-as 100
 neighbor 1.2.3.5 route-reflector-client
 neighbor 1.2.3.6 remote-as 100
 neighbor 1.2.3.6 route-reflector-client
 ...
```

BGP Scaling Techniques

- These 3 techniques should be core requirements on all ISP networks
  - Route Refresh (or Soft Reconfiguration)
  - Peer groups
  - Route Reflectors

BGP Confederations

Confederations

- Divide the AS into sub-AS
  - eBGP between sub-AS, but some iBGP information is kept
    - Preserve NEXT_HOP across the sub-AS (IGP carries this information)
    - Preserve LOCAL_PREF and MED
- Usually a single IGP
- Described in RFC5065

Confederations

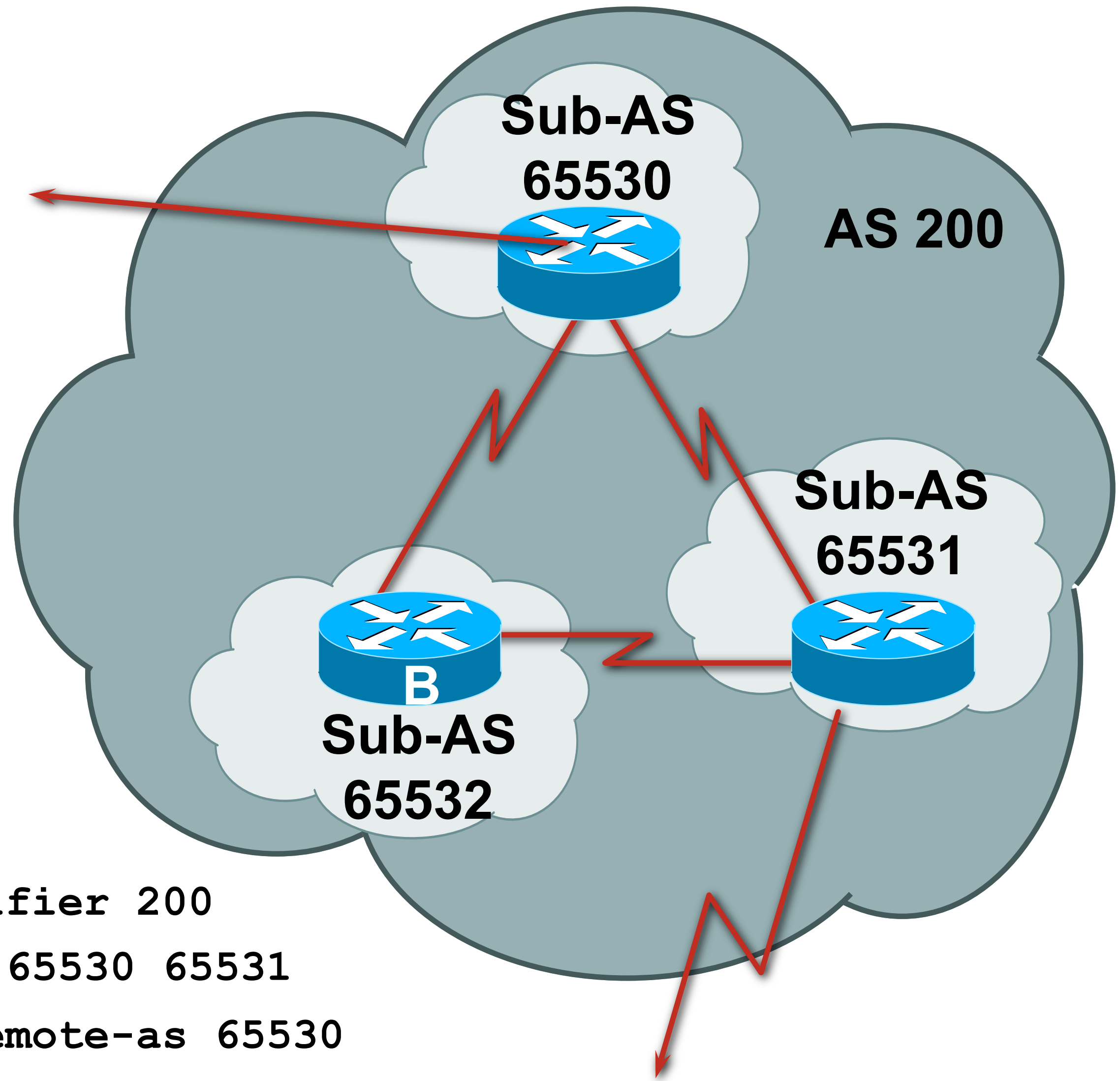
- Visible to outside world as single AS – "Confederation Identifier"

  Each sub-AS uses a number from the private space (64512-65534)

- iBGP speakers in sub-AS are fully meshed

  The total number of neighbors is reduced by limiting the full mesh requirement to only the peers in the sub-AS

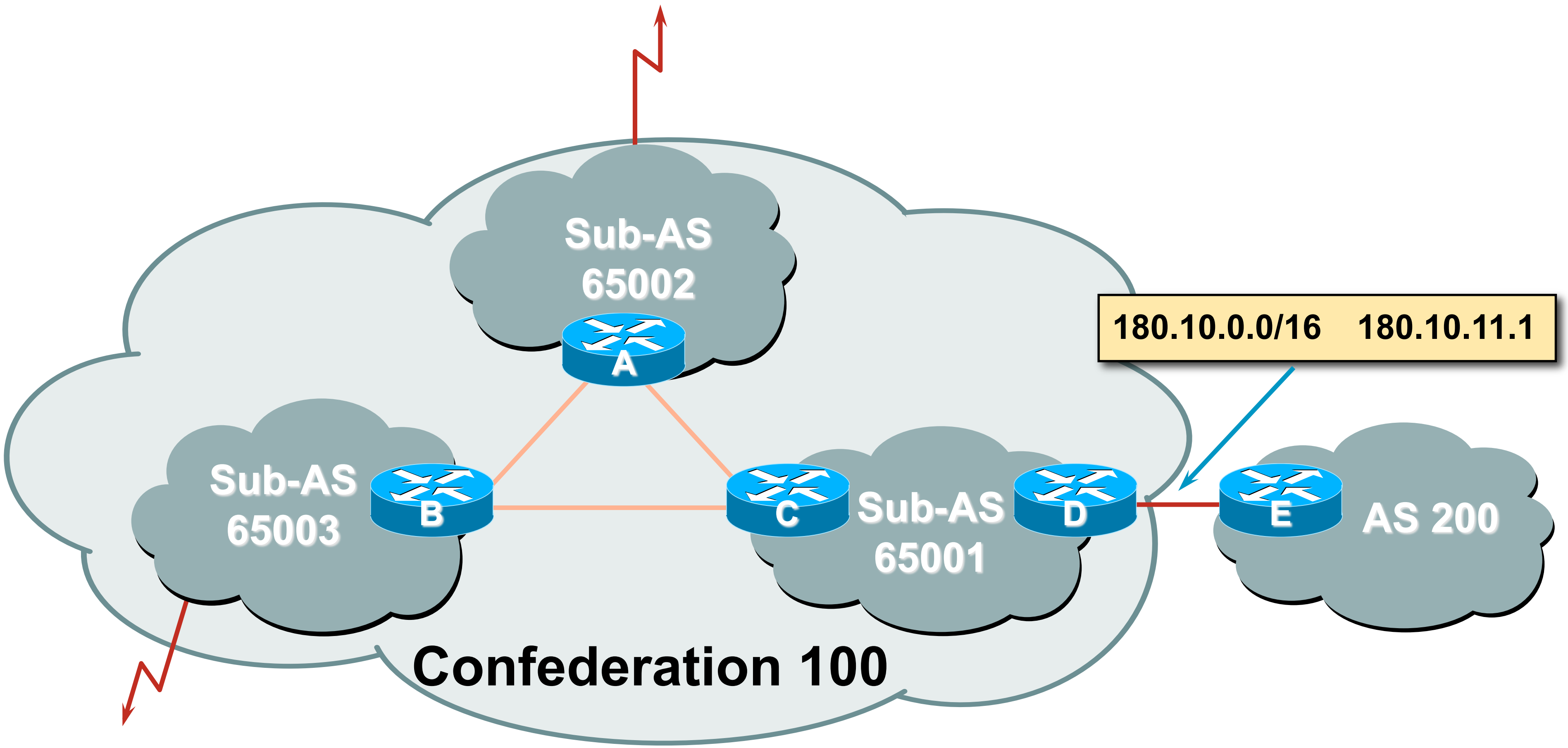
Confederations



- Configuration (rtr B):

```
router bgp 65532
 bgp confederation identifier 200
 bgp confederation peers 65530 65531
 neighbor 141.153.12.1 remote-as 65530
 neighbor 141.153.17.2 remote-as 65531
```


Confederations: Next Hop



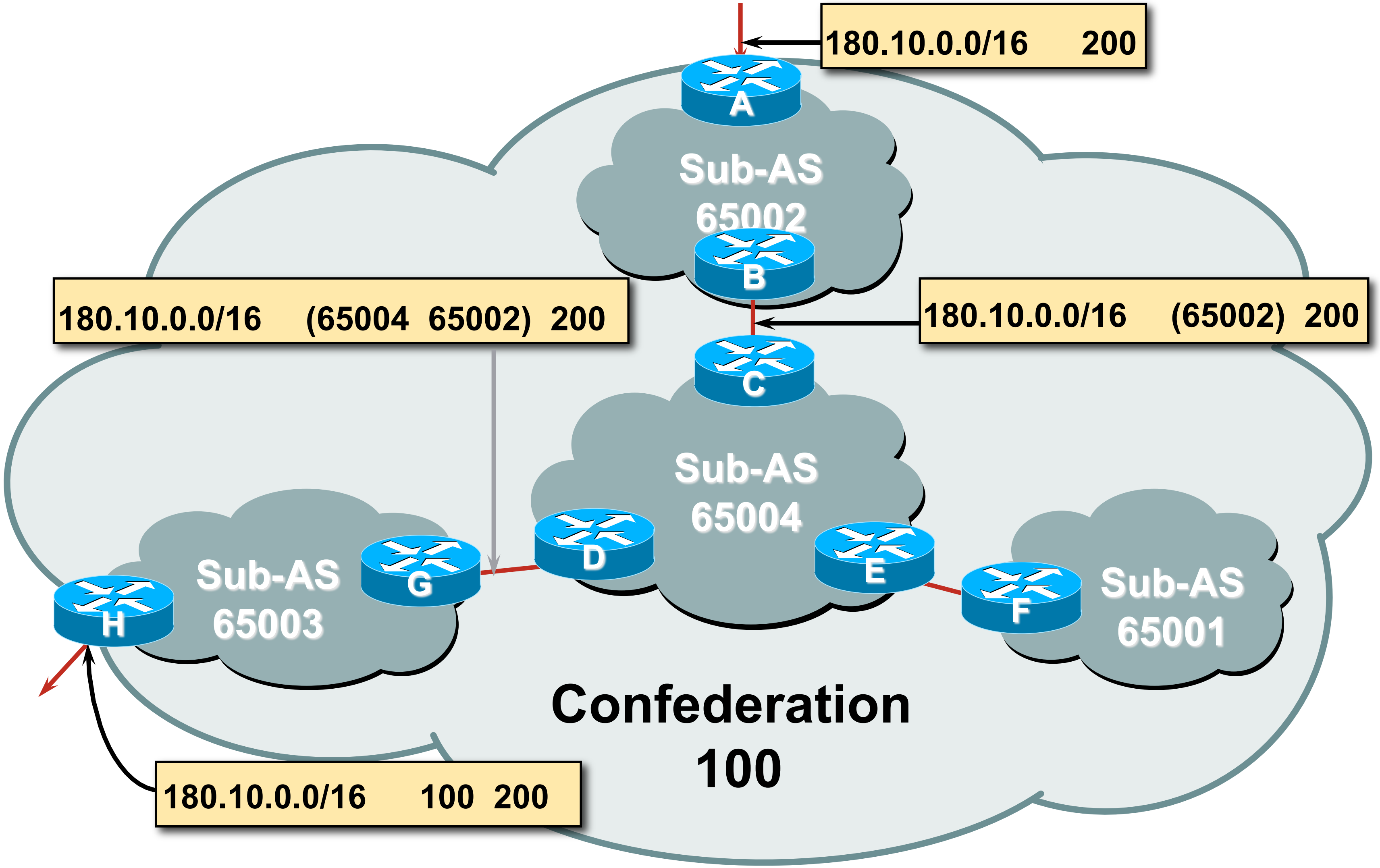
Confederation: Principle

- Local preference and MED influence path selection
- Preserve local preference and MED across sub-AS boundary
- Sub-AS eBGP path administrative distance

Confederations: Loop Avoidance

- Sub-AS traversed are carried as part of AS-path
- AS-sequence and AS path length
- Confederation boundary
- AS-sequence should be skipped during MED comparison

Confederations: AS-Sequence



Route Propagation Decisions

- Same as with "normal" BGP:
  - From peer in same sub-AS → only to external peers
  - From external peers → to all neighbors
- "External peers" refers to
  - Peers outside the confederation
  - Peers in a different sub-AS
    - Preserve LOCAL_PREF, MED and NEXT_HOP

Confederations (cont.)

- Example (cont.):

```
BGP table version is 78, local router ID is 141.153.17.1
Status codes: s suppressed, d damped, h history, * valid, > best, i
- internal
Origin codes: i - IGP, e - EGP, ? - incomplete
Network         Next Hop      Metric LocPrf Weight Path
*> 10.0.0.0     141.153.14.3   0     100       0     (65531) 1 i
*> 141.153.0.0 141.153.30.2   0     100       0     (65530) i
*> 144.10.0.0  141.153.12.1   0     100       0     (65530) i
*> 199.10.10.0 141.153.29.2   0     100       0     (65530) 1 i
```

More points about confederations

- Can ease “absorbing” other ISPs into you ISP – e.g., if one ISP buys another (can use local-as feature to do a similar thing)
- You can use route-reflectors with confederation sub-AS to reduce the sub-AS iBGP mesh

Confederations: Benefits

- Solves iBGP mesh problem
- Packet forwarding not affected
- Can be used with route reflectors
- Policies could be applied to route traffic between sub-AS's

Confederations: Caveats

- Minimal number of sub-AS
- Sub-AS hierarchy
- Minimal inter-connectivity between sub-AS's
- Path diversity
- Difficult migration

  BGP reconfigured into sub-AS

  must be applied across the network

RRs or Confederations

	Internet Connectivity	Multi-Level Hierarchy	Policy Control	Scalability	Migration Complexity
Confederations	Anywhere in the Network	Yes	Yes	Medium	Medium to High
Route Reflectors	Anywhere in the Network	Yes	Yes	Very High	Very Low

Most new service provider networks now deploy Route Reflectors from Day One

Deploying 32-bit ASNs

How to support customers usingthe extended ASN range

32-bit ASNs

- Standards documents

  Description of 32-bit ASNs

  www.rfc-editor.org/rfc/rfc4893.txt

  Textual representation

  www.rfc-editor.org/rfc/rfc5396.txt

- AS 23456 is reserved as interface between 16-bit and 32-bit ASN world

32-bit ASNs – terminology

- 16-bit ASNs

  Refers to the range 0 to 65535

- 32-bit ASNs

  Refers to the range 65536 to 4294967295

  (or the extended range)

- 32-bit ASN pool

  Refers to the range 0 to 4294967295

Getting a 32-bit ASN

Sample RIR policy

- From 1st January 2007
  - 32-bit ASNs were available on request
- From 1st January 2009
  - 32-bit ASNs were assigned by default
  - 16-bit ASNs were only available on request
- From 1st January 2010
  - No distinction – ASNs assigned from the 32-bit pool

Representation

- Representation of 0-4294967295 ASN range
  - Most operators favor traditional format (asplain)
  - A few prefer dot notation (X.Y):
    - asdot for 65536-4294967295, e.g 2.4
    - asdot+ for 0-4294967295, e.g 0.64513
  - **But regular expressions will have to be completely rewritten for asdot and asdot+ !!!**
- For example:
  - ^[0-9]+$ matches any ASN (16-bit and asplain)
  - This and equivalents extensively used in BGP multihoming configurations for traffic engineering
- Equivalent regexp for asdot is: ^([0-9]+)|([0-9]+\.[0-9]+)$
- Equivalent regexp for asdot+ is: ^[0-9]+\.[0-9]+$

Changes

- 32-bit ASNs are backward compatible with 16-bit ASNs
- **There is no flag day**
- You do NOT need to:
  - Throw out your old routers
  - Replace your 16-bit ASN with a 32-bit ASN
- You do need to be aware that:
  - Your customers will come with 32-bit ASNs
  - ASN 23456 is not a bogon!
  - You will need a router supporting 32-bit ASNs to use a 32-bit ASN locally
- If you have a proper BGP implementation, 32-bit ASNs will be transported silently across your network

How does it work?

- If local router and remote router supports configuration of 32-bit ASNs

    BGP peering is configured as normal using the 32-bit ASN

- If local router and remote router does not support configuration of 32-bit ASNs

    BGP peering can only use a 16-bit ASN

- If local router only supports 16-bit ASN and remote router/network has a 32-bit ASN

    Compatibility mode is initiated…

How does it work?

- 4-byte AS support is advertised within BGP capability negotiation
  - Speakers who support 4-byte AS are known as NEW speakers
  - Those who do not are known as OLD speakers
- New Reserved AS#
  - AS_TRANS = AS #23456
  - 2-byte placeholder for a 4-byte AS number
  - Used for backward compatibility with OLD speakers
- Two new attributes, both are “optional transitive”
  - AS4_AGGREGATOR
  - AS4_PATH

4-byte AS – Formatting Updates

From the perspective of a NEW speaker…

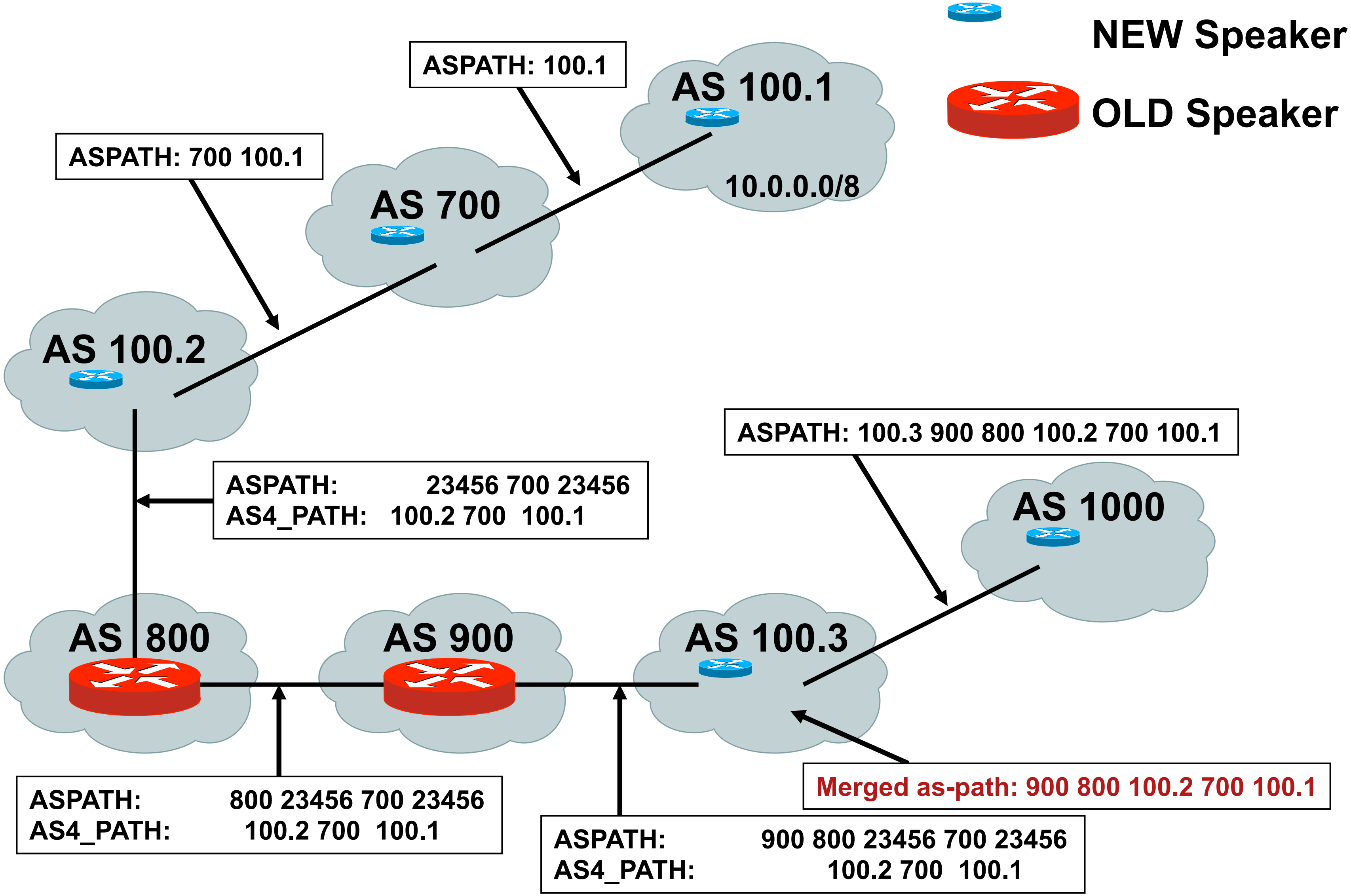
- When Formatting UPDATEs to another NEW speaker
  - Encode each AS number in 4-bytes
  - AS_PATH and AGGREGATOR are the relevant fields for BESTPATH
  - We should not see AS4_PATH and AS4_AGGREGATOR
- When Formatting UPDATEs to an OLD speaker
  - If the AGGREGATOR/ASPATH does not contain a 4-byte AS we are fine
  - If it does, substitute AS_TRANS (AS #23456) for each 4-byte AS
  - AS4_AGGREGATOR or AS4_PATH will contain a 4-byte encoded copy of the attribute if needed
  - OLD speaker will blindly pass along AS4_AGGREGATOR and AS4_PATH attributes

4-byte AS – Receiving Updates

From the perspective of a NEW speaker…

- When Receiving UPDATEs from a NEW speaker
  - Decode each AS number as 4-bytes
  - AS_PATH and AGGREGATOR are encoded as 4-bytes ASN
- When Receiving UPDATEs from an OLD speaker
  - AS4_AGGREGATOR will override AGGREGATOR
  - AS4_PATH and ASPATH must be merged to form the correct as-path
- Merging AS4_PATH and ASPATH
  - AS_PATH – `275 250 225` *`23456 23456`* `200` *`23456`* `175`
  - AS4_PATH – *`100.1 100.2`* `200` *`100.3`* `175`
  - Merged as-path – `275 250 225` *`100.1 100.2`* `200` *`100.3`* `175`

4-byte AS – ASPATH & AS4_PATH in a mixed environment



Question: What about loops?

- You might receive an as path as:

  `275 250 225 `*`23456 23456`*` 200 `*`23456`*` 175`

- Will an OLD speaker reject this because 23456 is in the AS path multiple times? NO!

  The OLD speaker checks for its AS in the AS path.

  Because a OLD speaker will never be AS number 23456, there will be no loop.

  AS 23456 is a 2-byte placeholder for a 4-byte AS number and is not used by customers.

Question: Peering with 4-byte AS

- How do OLD speakers peer with NEW speakers?

  For every NEW speaker that has a 4-byte AS number, the OLD speaker will peer using “remote-as 23456” in the bgp configuration

- If an OLD speaker is peering with two NEW speakers, how will the OLD speaker know which neighbor to send a subnet’s traffic to if both have remote-as 23456

  The OLD speaker knows the next-hop IP address for a given subnet and will send it to the peer with that next-hop address.

If 32-bit ASN not supported:

- Inability to distinguish between peer ASes using 32-bit ASNs
  - They will all be represented by AS23456
  - Could be problematic for transit provider's policy
- Inability to distinguish prefix's origin AS
  - How to tell whether origin is real or fake?
  - The real and fake both represented by AS23456
- Incorrect NetFlow summaries:
  - Prefixes from 32-bit ASNs will all be summarised under AS23456
  - Traffic statistics need to be measured per prefix and aggregated
  - Makes it hard to determine peerability of a neighbouring network

Implementations (Jan 2010)

- Cisco IOS-XR 3.4 onwards
- Cisco IOS-XE 2.3 onwards
- Cisco IOS 12.0(32)S12, 12.4(24)T, 12.2SRE, 12.2(33)SXI1 onwards
- Cisco NX-OS 4.0(1) onwards
- Quagga 0.99.10 (patches for 0.99.6)
- OpenBGPd 4.2 (patches for 3.9 & 4.0)
- Juniper JunOSe 4.1.0 & JunOS 9.1 onwards
- Redback SEOS
- Force10 FTOS7.7.1 onwards

http://as4.cluepon.net/index.php/Software_Support for a complete list

Route Flap Damping

Network Stability for the 1990s

Network Instability for the 21st Century!

Route Flap Damping

- For many years, Route Flap Damping was a strongly recommended practice
- Now it is strongly discouraged as it causes far greater network instability than it cures
- But first, the theory…

Route Flap Damping

- Route flap
  - Going up and down of path or change in attribute
    - BGP WITHDRAW followed by UPDATE = 1 flap
    - eBGP neighbour going down/up is NOT a flap
  - Ripples through the entire Internet
  - Wastes CPU
- Damping aims to reduce scope of route flap propagation

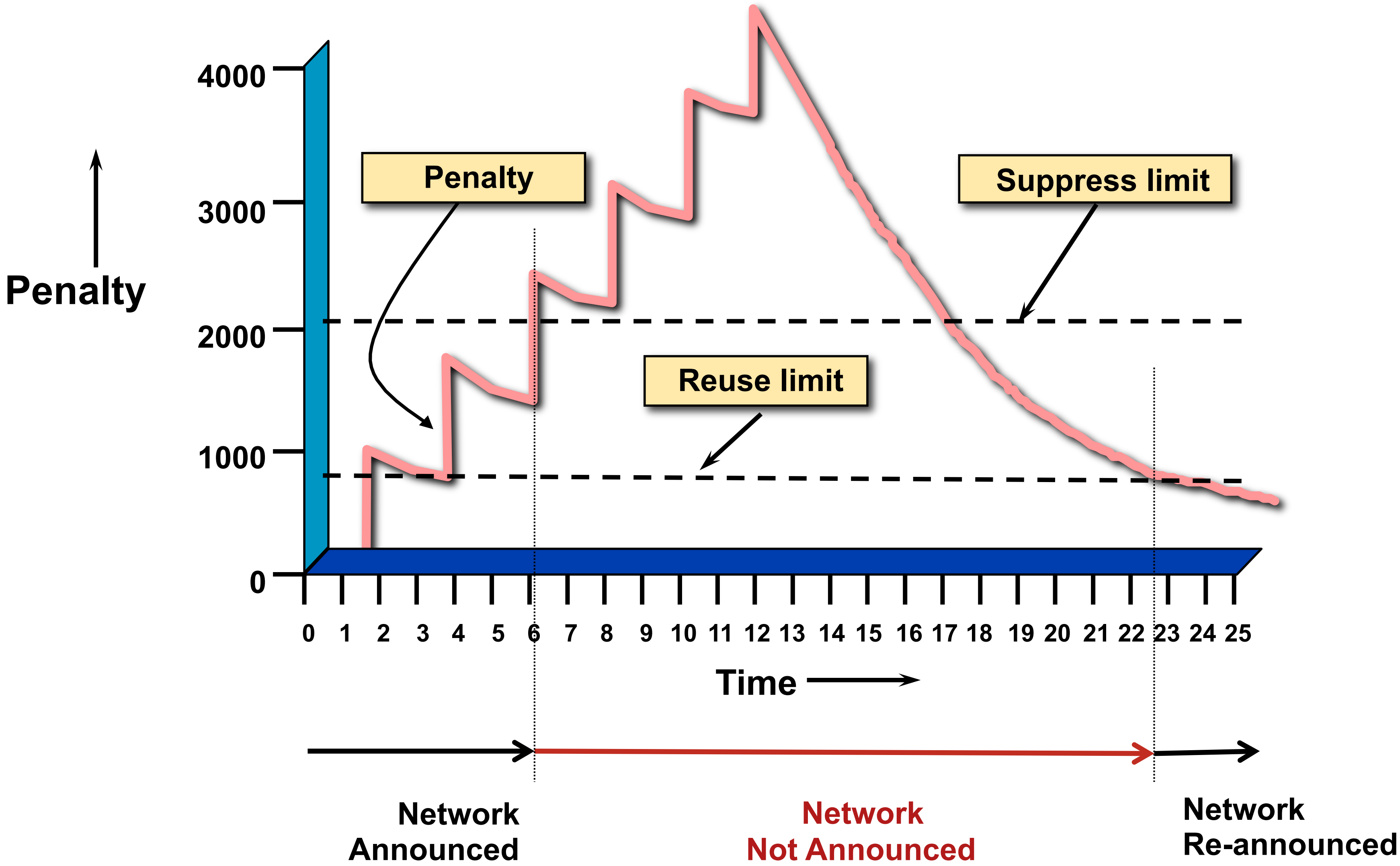
Route Flap Damping (continued)

- Requirements
  - Fast convergence for normal route changes
  - History predicts future behaviour
  - Suppress oscillating routes
  - Advertise stable routes
- Implementation described in RFC 2439

Operation

- Add penalty (1000) for each flap
  - Change in attribute gets penalty of 500
- Exponentially decay penalty
  - half life determines decay rate
- Penalty above suppress-limit
  - do not advertise route to BGP peers
- Penalty decayed below reuse-limit
  - re-advertise route to BGP peers
  - penalty reset to zero when it is half of reuse-limit

Operation



Penalty
4000
3000
2000
1000
0
Penalty
Suppress limit
Reuse limit
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
Time
Network
Announced
Network
Not Announced
Network
Re-announced

Operation

- Only applied to inbound announcements from eBGP peers
- Alternate paths still usable
- Controlled by:

  Half-life (default 15 minutes)

  reuse-limit (default 750)

  suppress-limit (default 2000)

  maximum suppress time (default 60 minutes)

Configuration

- Fixed damping

```
router bgp 100
bgp dampening [<half-life> <reuse-value> <suppress-penalty> <maximum suppress time>]
```

- Selective and variable damping

```
bgp dampening [route-map <name>]
route-map <name> permit 10
 match ip address prefix-list FLAP-LIST
 set dampening [<half-life> <reuse-value> <suppress-penalty> <maximum suppress time>]
ip prefix-list FLAP-LIST permit 192.0.2.0/24 le 32
```

Operation

- Care required when setting parameters
- Penalty must be less than reuse-limit at the maximum suppress time
- Maximum suppress time and half life must allow penalty to be larger than suppress limit

Configuration

- Examples – ✘

  bgp dampening 15 500 2500 30

  reuse-limit of 500 means maximum possible penalty is 2000 – no prefixes suppressed as penalty cannot exceed suppress-limit

- Examples – ✓

  bgp dampening 15 750 3000 45

  reuse-limit of 750 means maximum possible penalty is 6000 – suppress limit is easily reached

Maths!

- Maximum value of penalty is

$$\text{max-penalty} = \text{reuse-limit} \times 2^{\left(\frac{\text{max-suppress-time}}{\text{half-life}}\right)}$$

- Always make sure that suppress-limit is LESS than max-penalty otherwise there will be no route damping

Route Flap Damping History

- First implementations on the Internet by 1995
- Vendor defaults too severe

  RIPE Routing Working Group recommendations in ripe-178, ripe-210, and ripe-229

  http://www.ripe.net/ripe/docs

  But many ISPs simply switched on the vendors' default values without thinking

Serious Problems:

- "Route Flap Damping Exacerbates Internet Routing Convergence“

  Zhuoqing Morley Mao, Ramesh Govindan, George Varghese & Randy H. Katz, August 2002

- “What is the sound of one route flapping?”

  Tim Griffin, June 2002

- Various work on routing convergence by Craig Labovitz and Abha Ahuja a few years ago

- “Happy Packets”

  Closely related work by Randy Bush et al

Problem 1:

- One path flaps:

  BGP speakers pick next best path, announce to all peers, flap counter incremented

  Those peers see change in best path, flap counter incremented

  After a few hops, peers see multiple changes simply caused by a single flap → prefix is suppressed

Problem 2:

- Different BGP implementations have different transit time for prefixes
    - Some hold onto prefix for some time before advertising
    - Others advertise immediately
- Race to the finish line causes appearance of flapping, caused by a simple announcement or path change → prefix is suppressed

Solution:

- Do NOT use Route Flap Damping whatever you do!
- RFD will unnecessarily impair access to:
  - Your network and
  - The Internet
- More information contained in RIPE Routing Working Group recommendations:
  - www.ripe.net/ripe/docs/ripe-378.[pdf,html,txt]

BGP Scaling Techniques

ISP/IXP Workshops