BGP Attributes and Policy Control

ISP/IXP Workshops

Agenda

- BGP Attributes
- BGP Path Selection
- Applying Policy

BGP Attributes

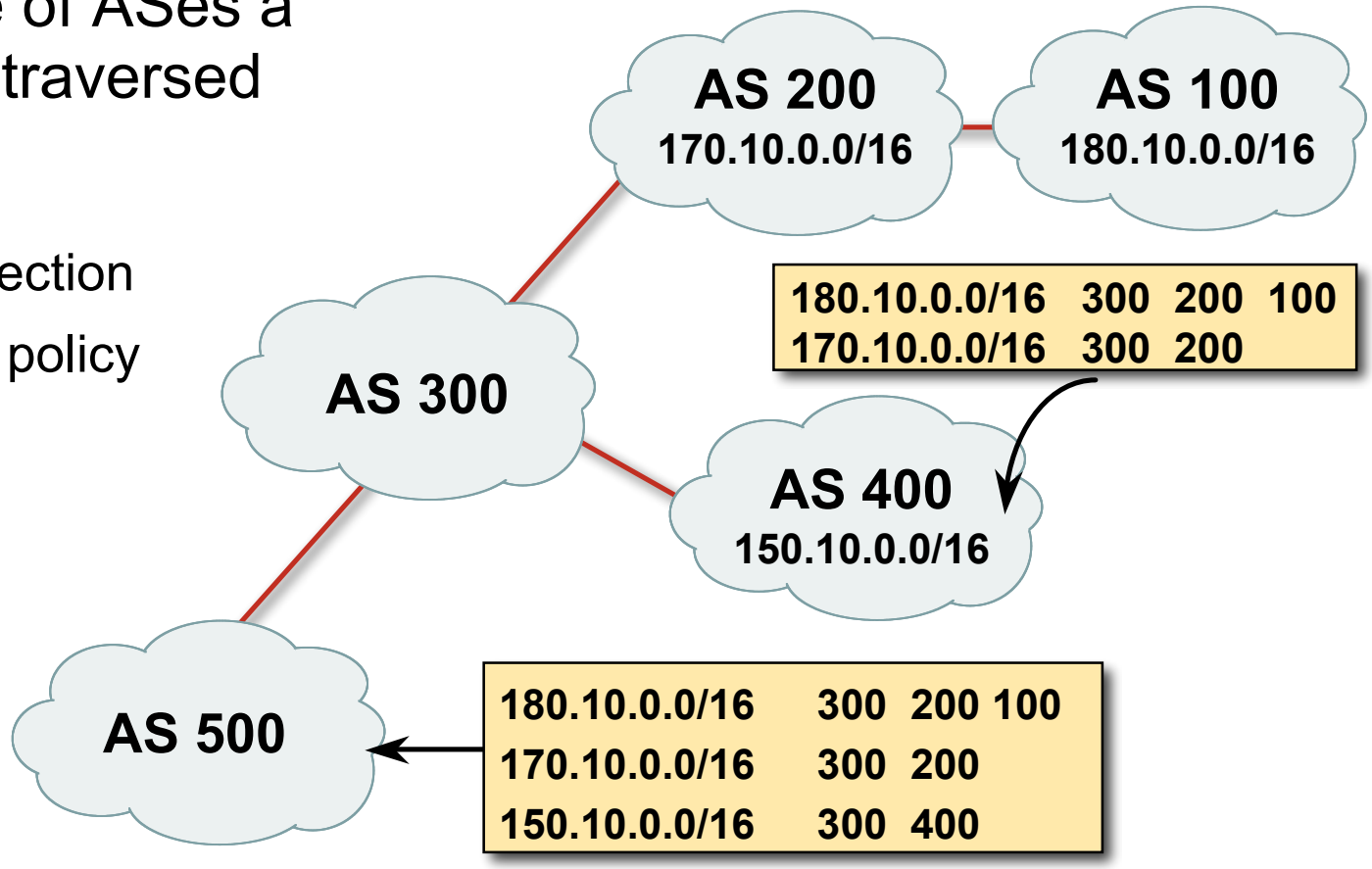
The “tools” available for the job

What Is an Attribute?

- Describes the characteristics of prefix
- Transitive or non-transitive
- Some are mandatory

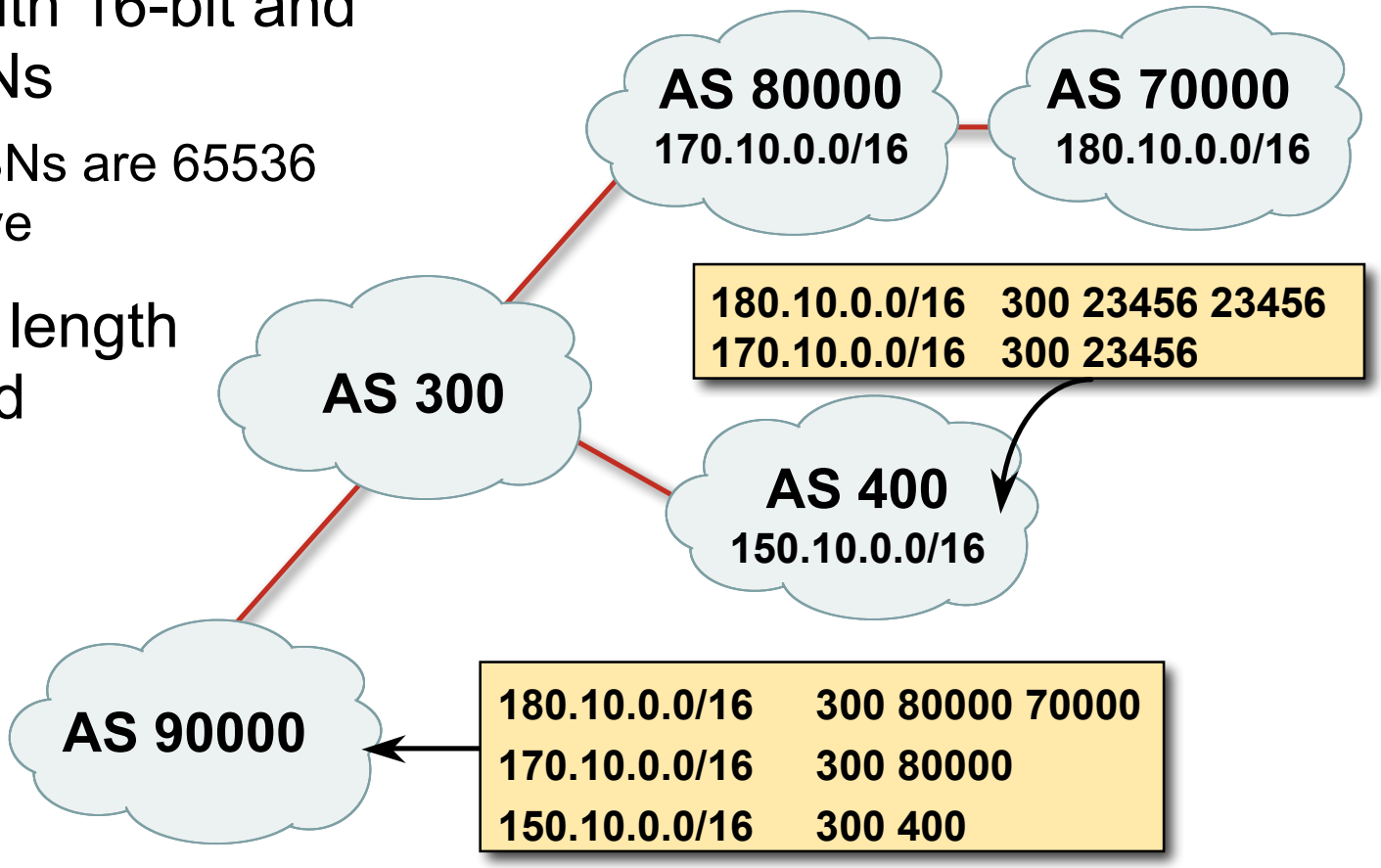
AS-Path

- Sequence of ASes a route has traversed
- Used for:
  - Loop detection
  - Applying policy

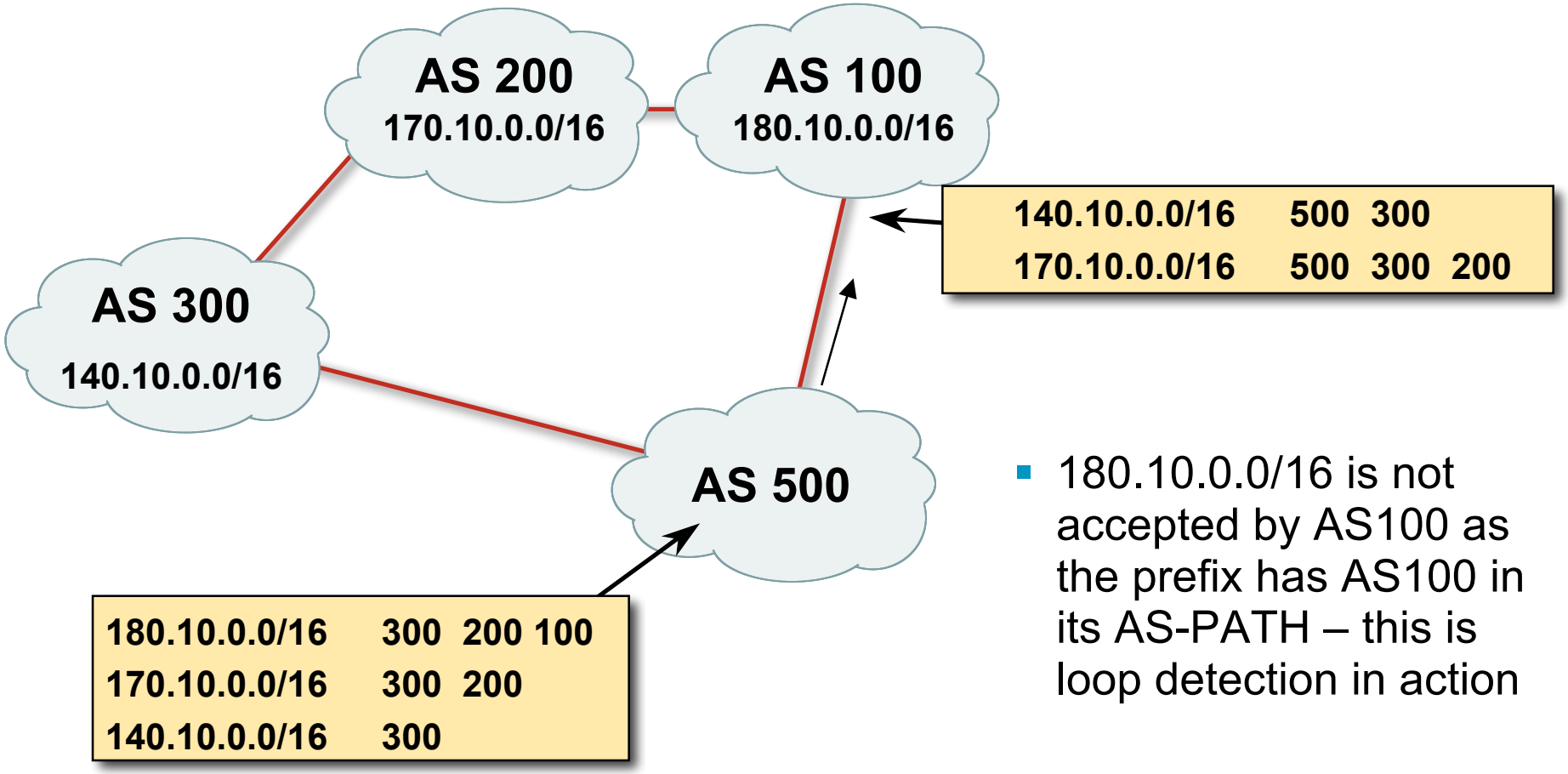


AS-Path (with 16 and 32-bit ASNs)

- Internet with 16-bit and 32-bit ASNs
  - 32-bit ASNs are 65536 and above
- AS-PATH length maintained

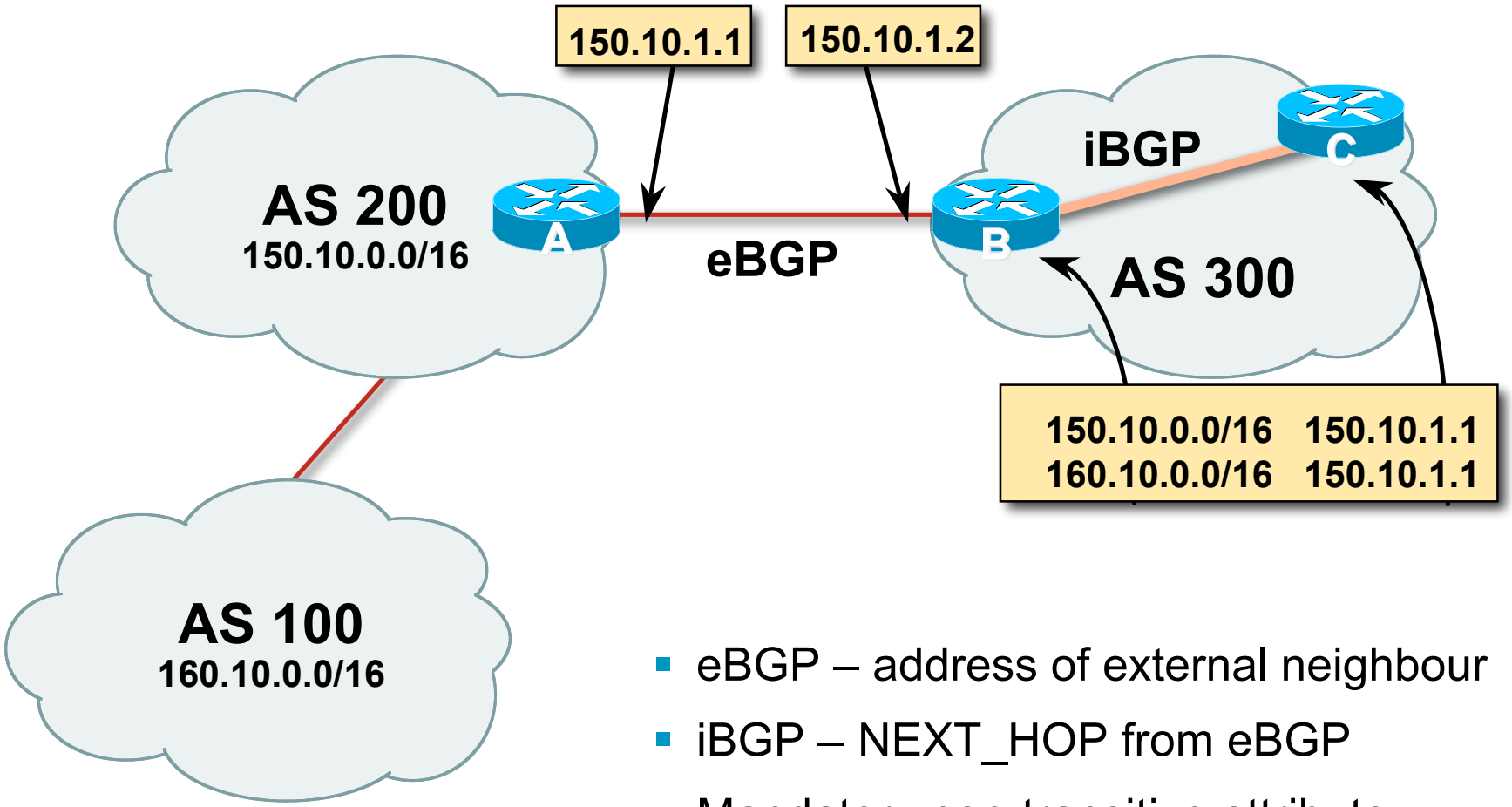


AS-Path loop detection



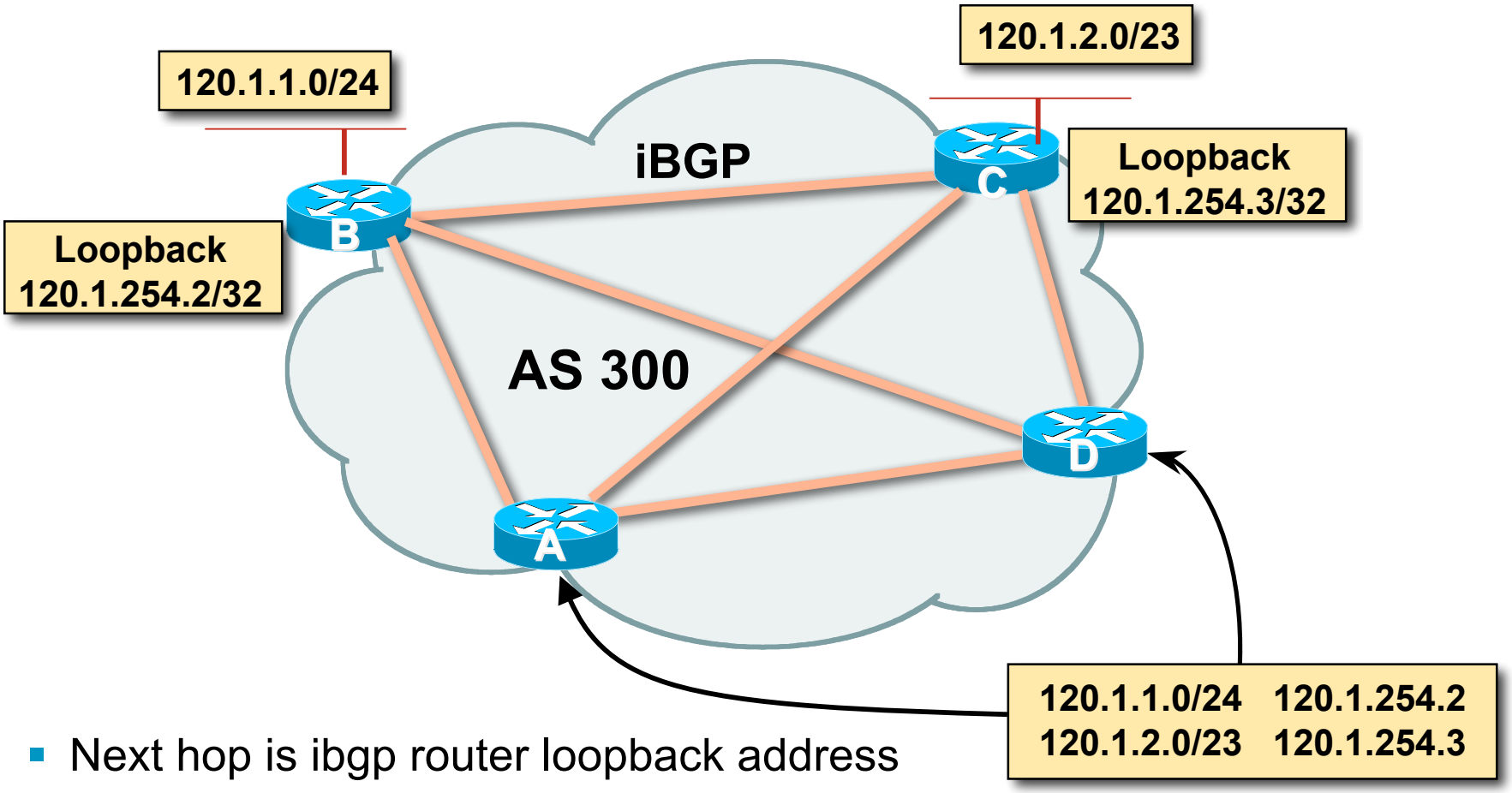
- 180.10.0.0/16 is not accepted by AS100 as the prefix has AS100 in its AS-PATH – this is loop detection in action

Next Hop



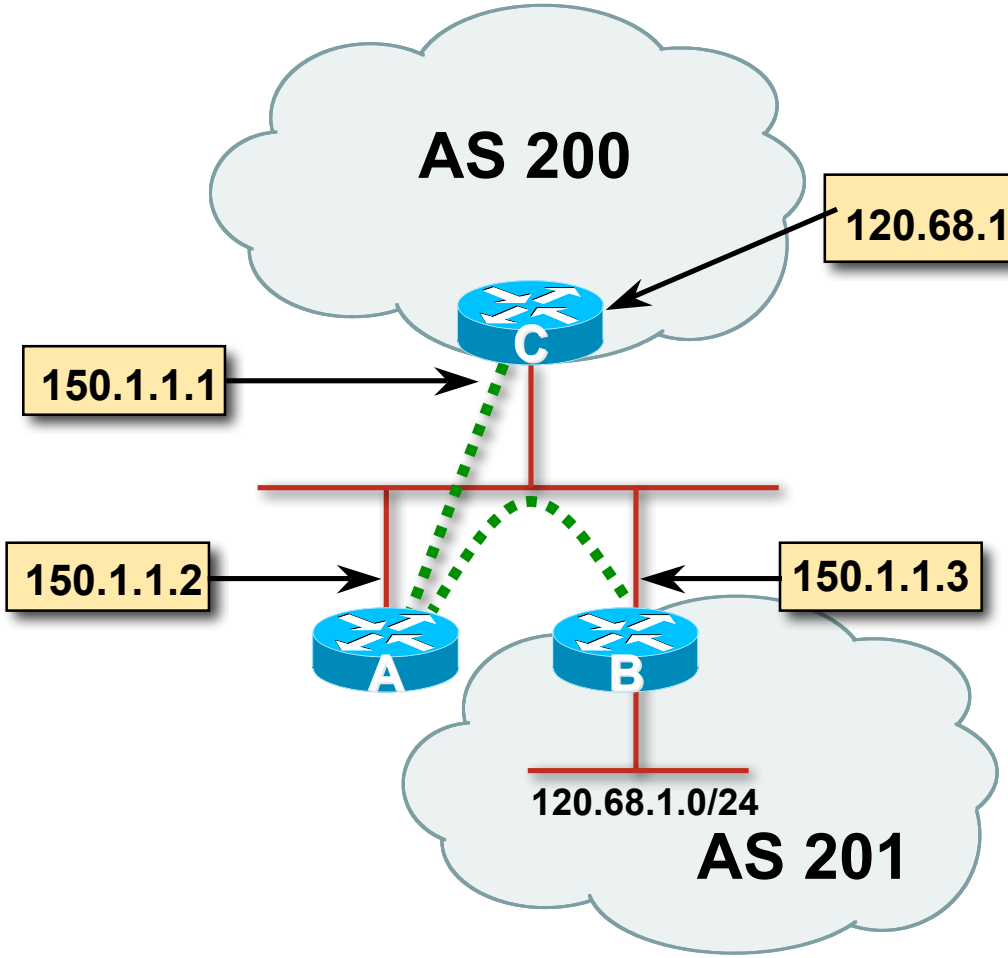
- eBGP – address of external neighbour
- iBGP – NEXT_HOP from eBGP
- Mandatory non-transitive attribute

iBGP Next Hop



- Next hop is ibgp router loopback address
- Recursive route look-up

Third Party Next Hop



- eBGP between Router A and Router C
- eBGP between RouterA and RouterB
- 120.68.1/24 prefix has next hop address of 150.1.1.3 – this is passed on to RouterC instead of 150.1.1.2
- More efficient
- No extra config needed

Next Hop Best Practice

- IOS default is for external next-hop to be propagated unchanged to iBGP peers
  - This means that IGP has to carry external next-hops
  - Forgetting means external network is invisible
  - With many eBGP peers, it is unnecessary extra load on IGP
- ISP Best Practice is to change external next-hop to be that of the local router

```
neighbor x.x.x.x next-hop-self
```

Next Hop (Summary)

- IGP should carry route to next hops
- Recursive route look-up
- Unlinks BGP from actual physical topology
- Use “next-hop-self” for external next hops
- Allows IGP to make intelligent forwarding decision

Origin

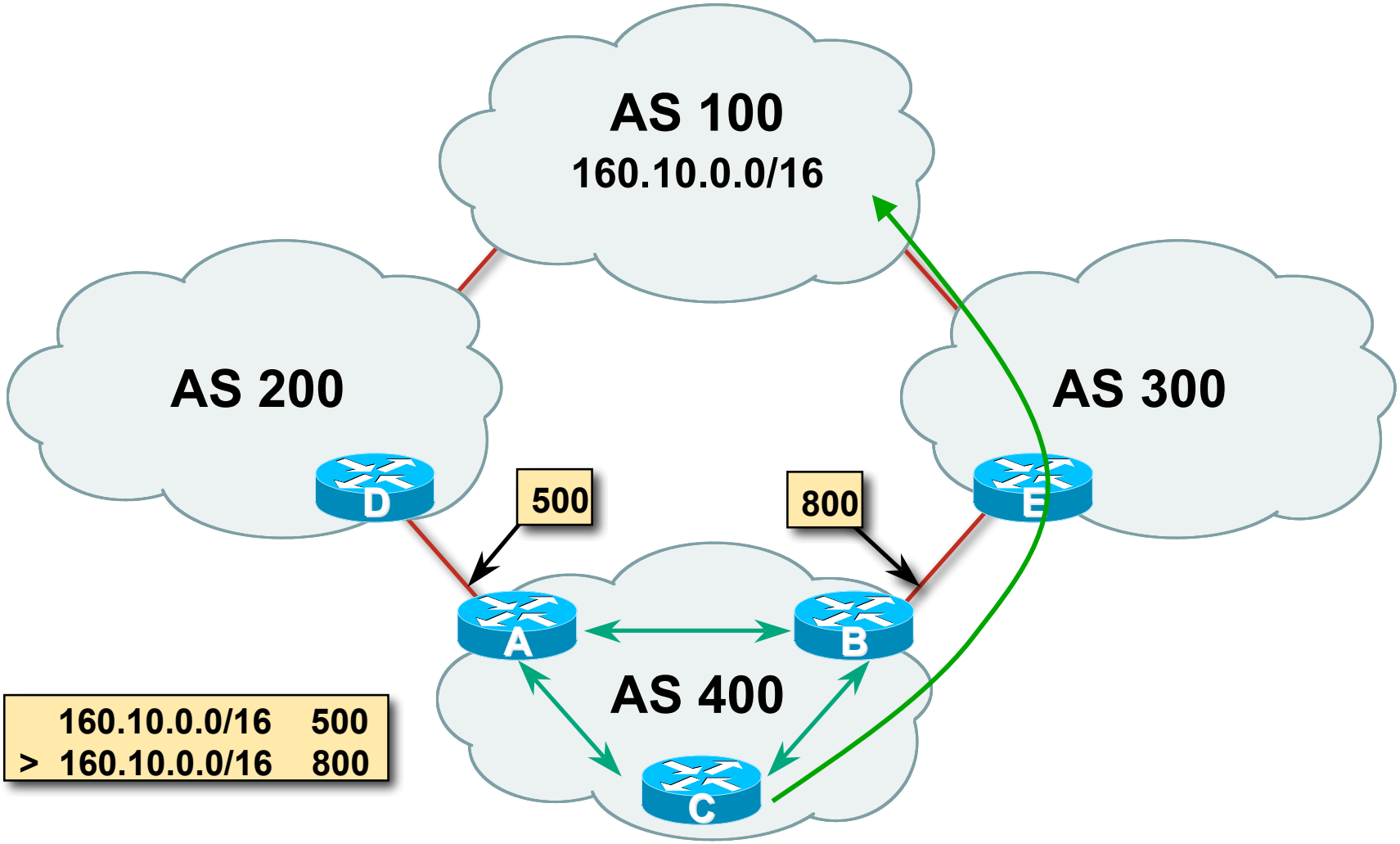
- Conveys the origin of the prefix
- Historical attribute
  - Used in transition from EGP to BGP
- Transitive and Mandatory Attribute
- Influences best path selection
- Three values: IGP, EGP, incomplete
  - IGP – generated by BGP network statement
  - EGP – generated by EGP
  - incomplete – redistributed from another routing protocol

Aggregator

- Conveys the IP address of the router or BGP speaker generating the aggregate route
- Optional & transitive attribute
- Useful for debugging purposes
- Does not influence best path selection
- Creating aggregate using “aggregate-address” sets the aggregator attribute:

```
router bgp 100
 aggregate-address 100.1.0.0 255.255.0.0
```

Local Preference



Local Preference

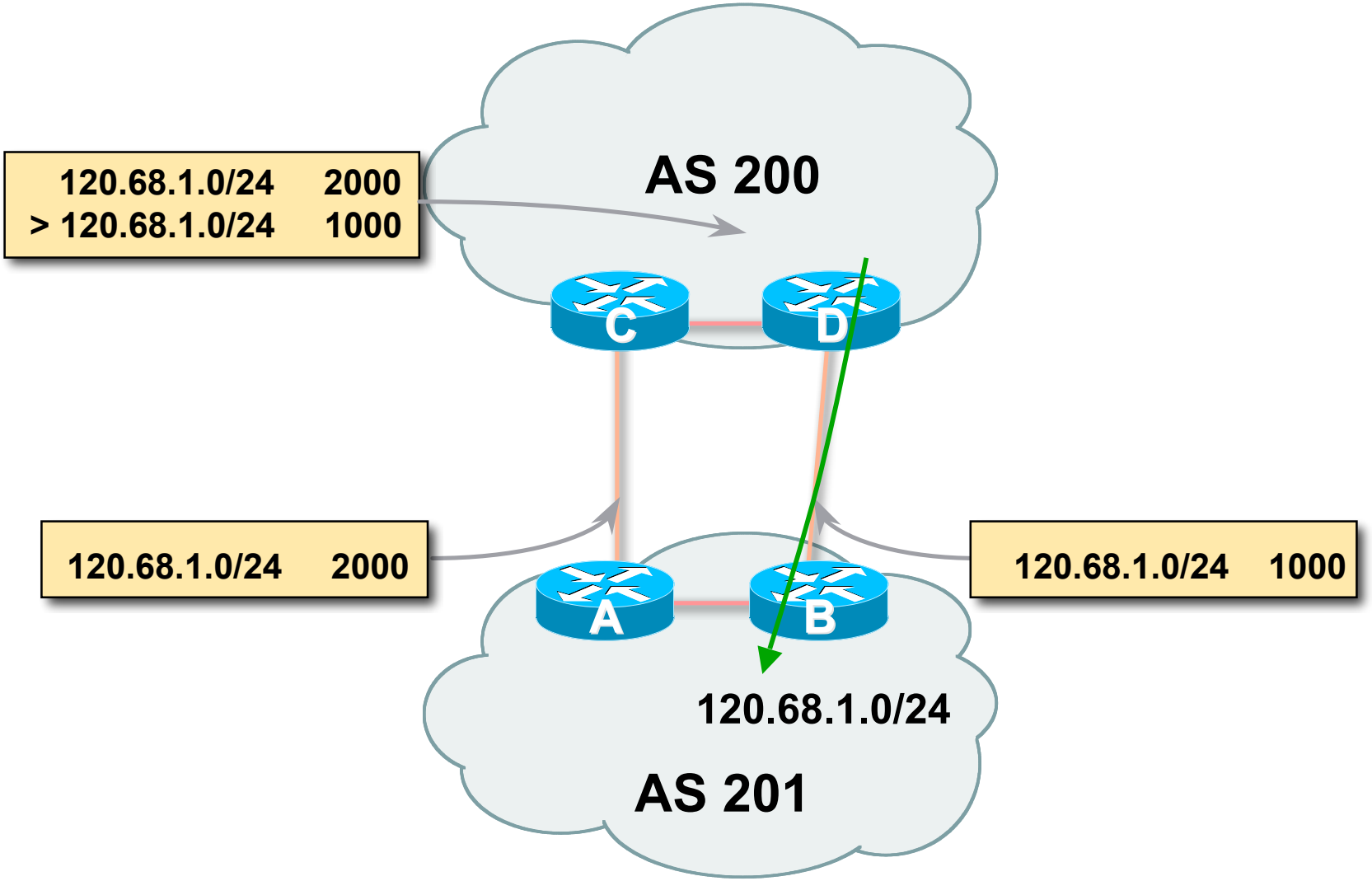
- Non-transitive and optional attribute
- Local to an AS only
  - Default local preference is 100 (IOS)
- Used to influence BGP path selection
  - determines best path for *outbound* traffic
- Path with highest local preference wins

Local Preference

- Configuration of Router B:

```
router bgp 400
 neighbor 120.5.1.1 remote-as 300
 neighbor 120.5.1.1 route-map local-pref in
!
route-map local-pref permit 10
 match ip address prefix-list MATCH
 set local-preference 800
!
ip prefix-list MATCH permit 160.10.0.0/16
```

Multi-Exit Discriminator (MED)



Multi-Exit Discriminator

- Inter-AS – non-transitive & optional attribute
- Used to convey the relative preference of entry points

  determines best path for inbound traffic
- Comparable if paths are from same AS

  `bgp always-compare-med` allows comparisons of MEDs from different ASes
- Path with lowest MED wins
- Absence of MED attribute implies MED value of zero (RFC4271)

MED & IGP Metric

- IGP metric can be conveyed as MED

  `set metric-type internal` in route-map

  enables BGP to advertise a MED which corresponds to the IGP metric values

  changes are monitored (and re-advertised if needed) every 600s

  `bgp dynamic-med-interval <secs>`

Multi-Exit Discriminator

- Configuration of Router B:

```
router bgp 400
 neighbor 120.5.1.1 remote-as 200
 neighbor 120.5.1.1 route-map set-med out
!
route-map set-med permit 10
 match ip address prefix-list MATCH
 set metric 1000
!
ip prefix-list MATCH permit 120.68.1.0/24
```

Weight

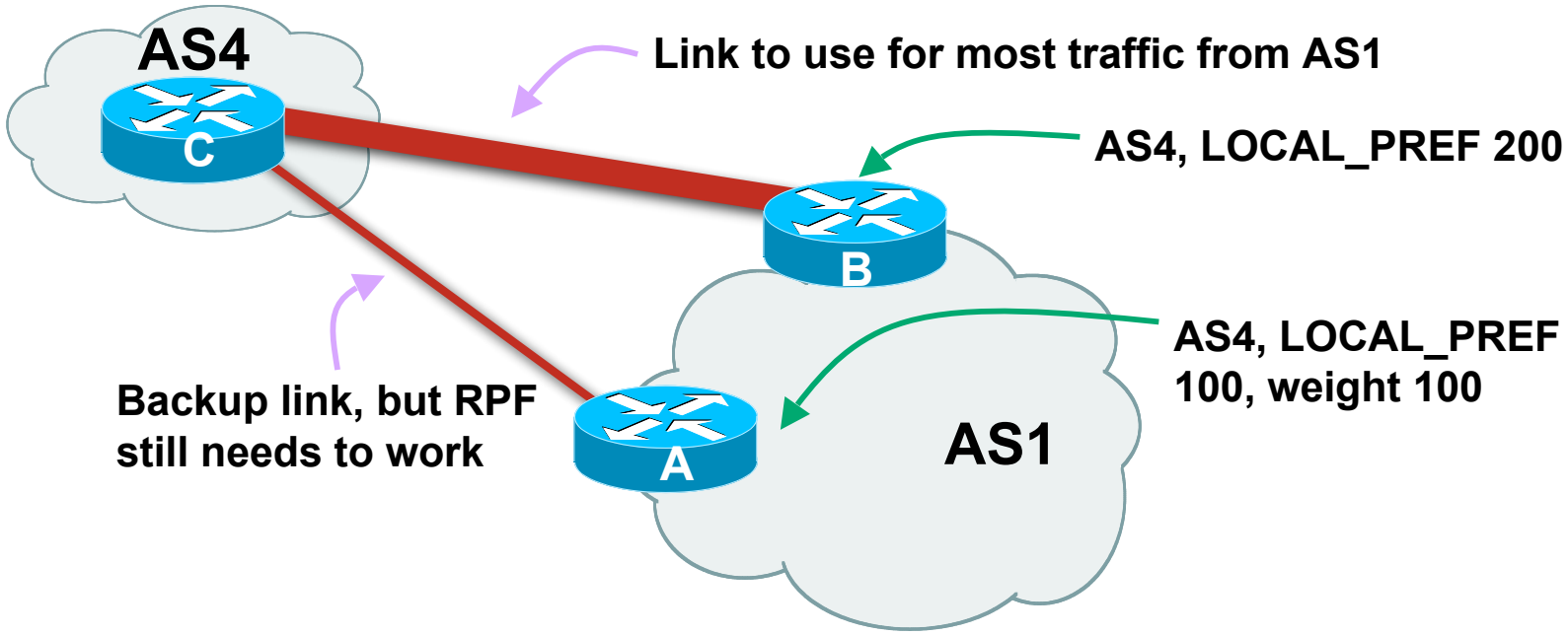
- Not really an attribute – local to router
- Highest weight wins
- Applied to all routes from a neighbour

```
neighbor 120.5.7.1 weight 100
```

- Weight assigned to routes based on filter

```
neighbor 120.5.7.3 filter-list 3 weight 50
```

Weight – Used to help Deploy RPF



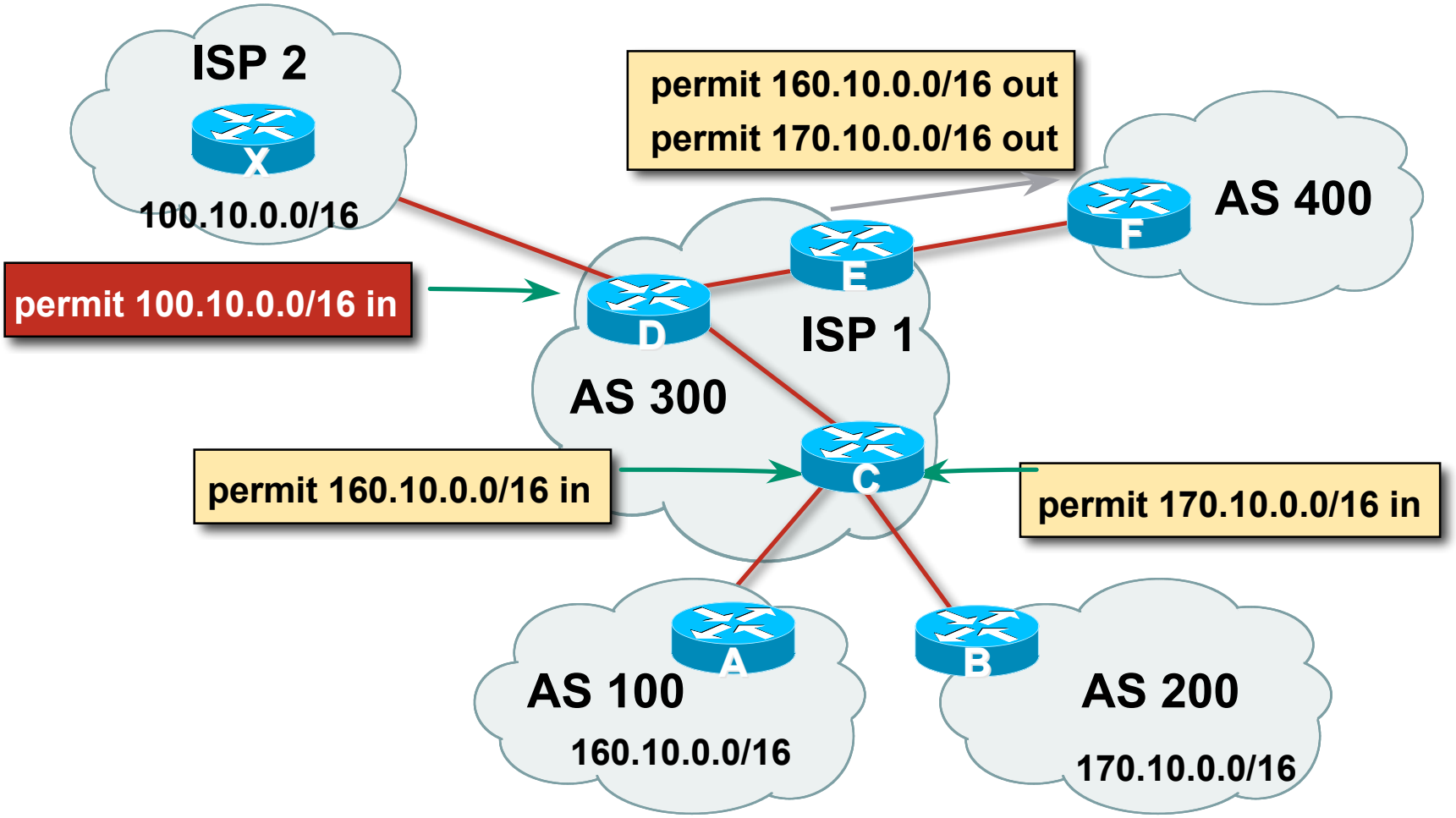
- Best path to AS4 from AS1 is always via B due to local-pref
- But packets arriving at A from AS4 over the direct C to A link will pass the RPF check as that path has a priority due to the weight being set

  If weight was not set, best path back to AS4 would be via B, and the RPF check would fail

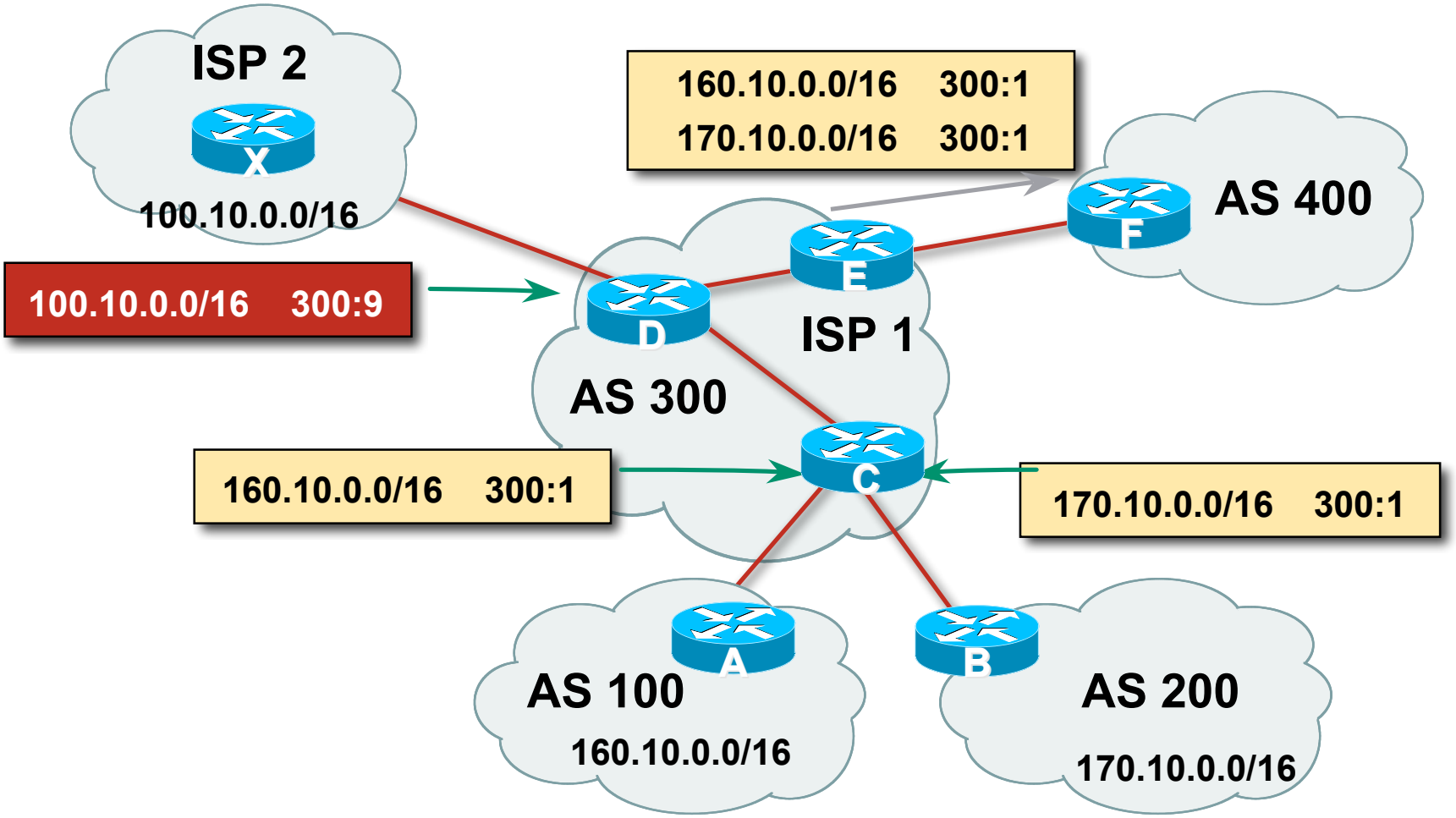
Community

- Communities are described in RFC1997
  - Transitive and Optional Attribute
- 32 bit integer
  - Represented as two 16 bit integers (RFC1998)
  - Common format is <local-ASN>:xx
  - 0:0 to 0:65535 and 65535:0 to 65535:65535 are reserved
- Used to group destinations
  - Each destination could be member of multiple communities
- Very useful in applying policies within and between ASes

Community Example (before)



Community Example (after)



Well-Known Communities

- Several well known communities

  www.iana.org/assignments/bgp-well-known-communities

- no-export 65535:65281

  do not advertise to any eBGP peers

- no-advertise 65535:65282

  do not advertise to any BGP peer

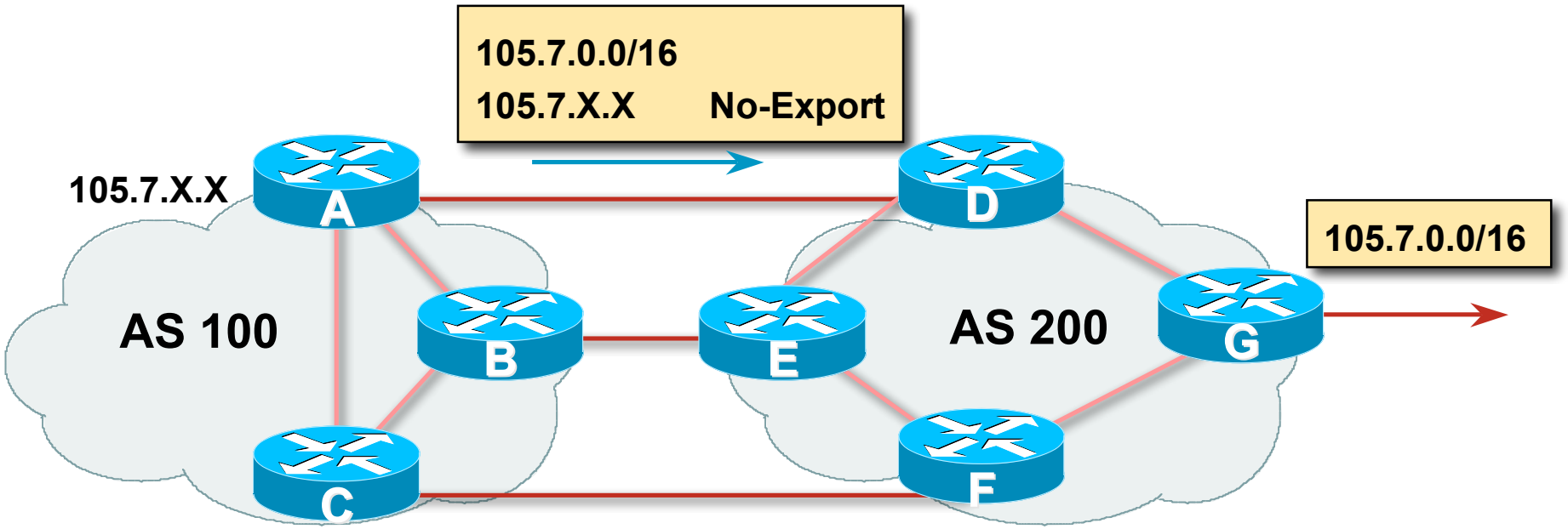
- no-export-subconfed 65535:65283

  do not advertise outside local AS (only used with confederations)

- no-peer 65535:65284

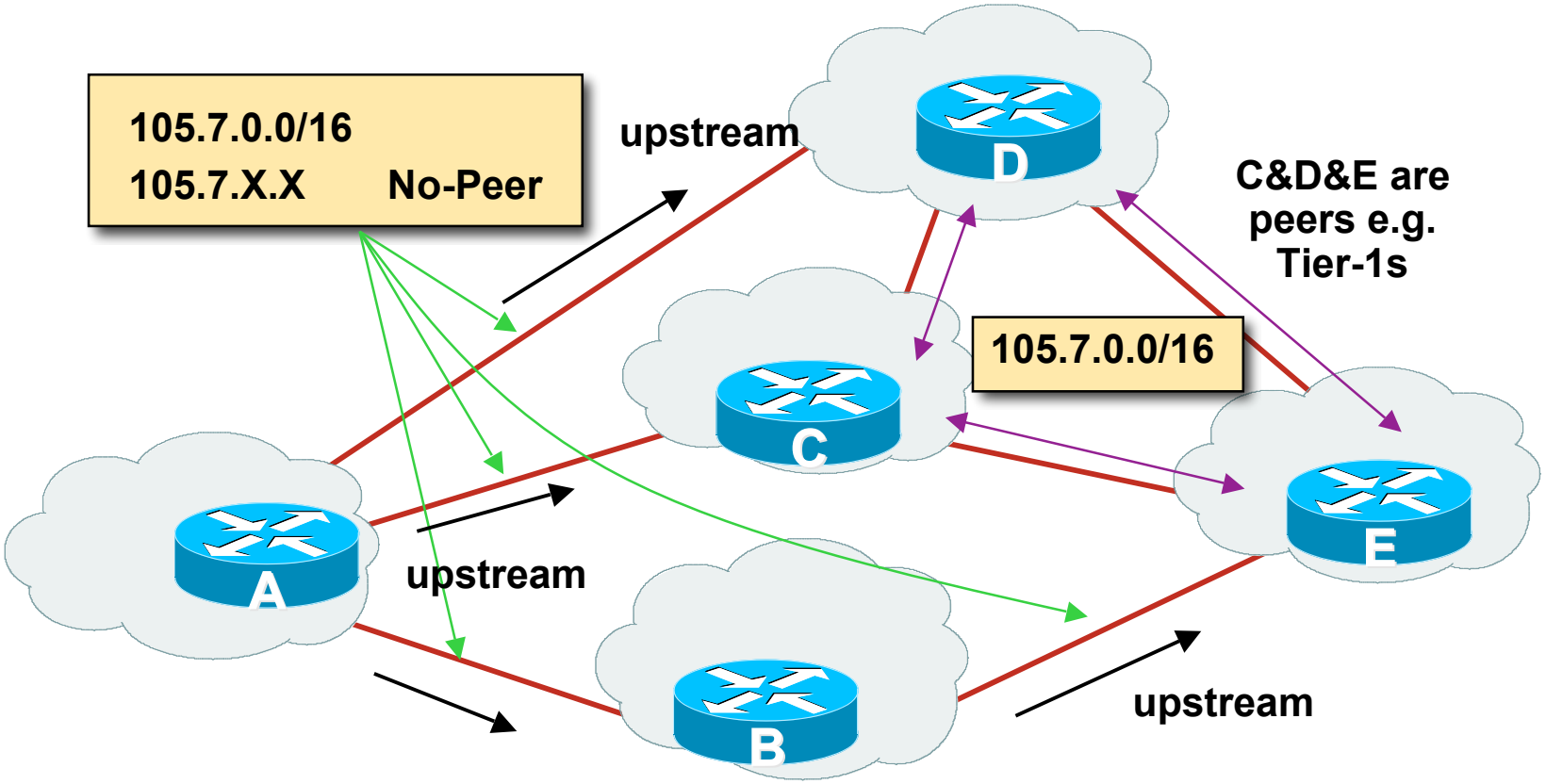
  do not advertise to bi-lateral peers (RFC3765)

No-Export Community



- AS100 announces aggregate and subprefixes
  - Intention is to improve loadsharing by leaking subprefixes
- Subprefixes marked with no-export community
- Router G in AS200 does not announce prefixes with no-export community set

No-Peer Community



- Sub-prefixes marked with no-peer community are not sent to bi-lateral peers

  They are only sent to upstream providers

Summary
Attributes in Action

```
Router1>sh ip bgp
BGP table version is 28, local router ID is 100.1.15.224
Status codes: s suppressed, d damped, h history,
              * valid, > best,i - internal, r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 100.1.0.0/20     0.0.0.0                  0         32768 i
*>i100.1.16.0/20    100.1.31.224             0    100      0 i
*>i100.1.32.0/19    100.1.63.224             0    100      0 i
...
```

BGP Path Selection Algorithm

Why is this the best path?

BGP Path Selection Algorithm for IOS Part One

- Do not consider path if no route to next hop
- Do not consider iBGP path if not synchronised (Cisco IOS)
- Highest weight (local to router)
- Highest local preference (global within AS)
- Prefer locally originated route
- Shortest AS path

BGP Path Selection Algorithm for IOS Part Two

- Lowest origin code

  IGP < EGP < incomplete

- Lowest Multi-Exit Discriminator (MED)

  If bgp deterministic-med, order the paths before comparing

  If bgp always-compare-med, then compare for all paths

  otherwise MED only considered if paths are from the same AS (default)

BGP Path Selection Algorithm for IOS Part Three

- Prefer eBGP path over iBGP path
- Path with lowest IGP metric to next-hop
- For eBGP paths:

  If multipath is enabled, install N parallel paths in forwarding table

  If router-id is the same, go to next step

  If router-id is not the same, select the oldest path

BGP Path Selection Algorithm for IOS Part Four

- Lowest router-id (originator-id for reflected routes)
- Shortest cluster-list

  Client must be aware of Route Reflector attributes!
- Lowest neighbour address

Applying Policy with BGP

How to use the “tools”

Applying Policy with BGP

- Policy-based on AS path, community or the prefix
- Rejecting/accepting selected routes
- Set attributes to influence path selection
- Tools:
  - Prefix-list (filters prefixes)
  - Filter-list (filters ASes)
  - Route-maps and communities

Policy Control – Prefix List

- Per neighbour prefix filter
  - incremental configuration
- Inbound or Outbound
- Based upon network numbers (using familiar IPv4 address/mask format)
- Using access-lists for filtering prefixes was deprecated long ago
  - **Strongly discouraged!**

Prefix-list Command Syntax

- Syntax:

  `[no] ip prefix-list` ***`list-name`*** **`[seq`** ***`seq-value`*****`]`**
  `permit|deny` ***`network/len`*** **`[ge`** ***`ge-value`*****`]`** **`[le`** ***`le-value`*****`]`**

  `network/len`: The prefix and its length

  `ge` ***`ge-value`***: "greater than or equal to"

  `le` ***`le-value`***: "less than or equal to"

- Both "ge" and "le" are optional

  Used to specify the range of the prefix length to be matched for prefixes that are more specific than *`network/len`*

- Sequence number is also optional

  `no ip prefix-list sequence-number` to disable display of sequence numbers

Prefix Lists – Examples

- Deny default route

```
ip prefix-list EG deny 0.0.0.0/0
```

- Permit the prefix 35.0.0.0/8

```
ip prefix-list EG permit 35.0.0.0/8
```

- Deny the prefix 172.16.0.0/12

```
ip prefix-list EG deny 172.16.0.0/12
```

- In 192/8 allow up to /24

```
ip prefix-list EG permit 192.0.0.0/8 le 24
```

  This allows all prefix sizes in the 192.0.0.0/8 address block, apart from /25, /26, /27, /28, /29, /30, /31 and /32.

Prefix Lists – Examples

- In 192/8 deny /25 and above

```
ip prefix-list EG deny 192.0.0.0/8 ge 25
```

  This denies all prefix sizes /25, /26, /27, /28, /29, /30, /31 and /32 in the address block 192.0.0.0/8.

  It has the same effect as the previous example

- In 193/8 permit prefixes between /12 and /20

```
ip prefix-list EG permit 193.0.0.0/8 ge 12 le 20
```

  This denies all prefix sizes /8, /9, /10, /11, /21, /22, … and higher in the address block 193.0.0.0/8.

- Permit all prefixes

```
ip prefix-list EG permit 0.0.0.0/0 le 32
```

  0.0.0.0 matches all possible addresses, “0 le 32” matches all possible prefix lengths

Policy Control – Prefix List

- Example Configuration

```
router bgp 100
 network 105.7.0.0 mask 255.255.0.0
 neighbor 102.10.1.1 remote-as 110
 neighbor 102.10.1.1 prefix-list AS110-IN in
 neighbor 102.10.1.1 prefix-list AS110-OUT out
!
ip prefix-list AS110-IN deny 218.10.0.0/16
ip prefix-list AS110-IN permit 0.0.0.0/0 le 32
ip prefix-list AS110-OUT permit 105.7.0.0/16
ip prefix-list AS110-OUT deny 0.0.0.0/0 le 32
```

Policy Control – Filter List

- Filter routes based on AS path

  Inbound or Outbound

- Example Configuration:

```
router bgp 100
 network 105.7.0.0 mask 255.255.0.0
 neighbor 102.10.1.1 filter-list 5 out
 neighbor 102.10.1.1 filter-list 6 in
!
ip as-path access-list 5 permit ^200$
ip as-path access-list 6 permit ^150$
```

Policy Control – Regular Expressions

- Like Unix regular expressions

.	Match one character
*	Match any number of preceding expression
+	Match at least one of preceding expression
^	Beginning of line
$	End of line
\	Escape a regular expression character
_	Beginning, end, white-space, brace
|	Or
()	brackets to contain expression
[]	brackets to contain number ranges

Policy Control – Regular Expressions

- Simple Examples

.*	match anything
.+	match at least one character
^$	match routes local to this AS
_1800$	originated by AS1800
^1800_	received from AS1800
_1800_	via AS1800
_790_1800_	via AS1800 and AS790
_(1800_)+	multiple AS1800 in sequence (used to match AS-PATH prepends)
_\(65530\)_	via AS65530 (confederations)

Policy Control – Regular Expressions

- Not so simple Examples

^[0-9]+$	Match AS_PATH length of one
^[0-9]+_[0-9]+$	Match AS_PATH length of two
^[0-9]*_[0-9]+$	Match AS_PATH length of one or two
^[0-9]*_[0-9]*$	Match AS_PATH length of one or two (will also match zero)
^[0-9]+_[0-9]+_[0-9]+$	Match AS_PATH length of three
_(701|1800)_	Match anything which has gone through AS701 or AS1800
_1849(_.+_)12163$	Match anything of origin AS12163 and passed through AS1849

Policy Control – Route Maps

- A route-map is like a “programme” for IOS
- Has “line” numbers, like programmes
- Each line is a separate condition/action
- Concept is basically:

  if *match* then do *expression* and exit

  else

  if *match* then do *expression* and exit

  else etc
- Route-map “continue” lets ISPs apply multiple conditions and actions in one route-map

Route Maps – Caveats

- Lines can have multiple set statements
- Lines can have multiple match statements
- Line with only a match statement
  - Only prefixes matching go through, the rest are dropped
- Line with only a set statement
  - All prefixes are matched and set
  - Any following lines are ignored
- Line with a match/set statement and no following lines
  - Only prefixes matching are set, the rest are dropped

Route Maps – Caveats

- Example

  Omitting the third line below means that prefixes not matching **list-one** or **list-two** are dropped

```
route-map sample permit 10
 match ip address prefix-list list-one
 set local-preference 120
!
route-map sample permit 20
 match ip address prefix-list list-two
 set local-preference 80
!
route-map sample permit 30      ! Don't forget this
```

Route Maps – Matching prefixes

- Example Configuration

```
router bgp 100
 neighbor 1.1.1.1 route-map infilter in
!
route-map infilter permit 10
 match ip address prefix-list HIGH-PREF
 set local-preference 120
!
route-map infilter permit 20
 match ip address prefix-list LOW-PREF
 set local-preference 80
!
ip prefix-list HIGH-PREF permit 10.0.0.0/8
ip prefix-list LOW-PREF permit 20.0.0.0/8
```

Route Maps – AS-PATH filtering

- Example Configuration

```
router bgp 100
 neighbor 102.10.1.2 remote-as 200
 neighbor 102.10.1.2 route-map filter-on-as-path in
!
route-map filter-on-as-path permit 10
 match as-path 1
 set local-preference 80
!
route-map filter-on-as-path permit 20
 match as-path 2
 set local-preference 200
!
ip as-path access-list 1 permit _150$
ip as-path access-list 2 permit _210_
```

Route Maps – AS-PATH prepends

- Example configuration of AS-PATH prepend

```
router bgp 300
 network 105.7.0.0 mask 255.255.0.0
 neighbor 2.2.2.2 remote-as 100
 neighbor 2.2.2.2 route-map SETPATH out
!
route-map SETPATH permit 10
 set as-path prepend 300 300
```

- Use your own AS number when prepending

  Otherwise BGP loop detection may cause disconnects

Route Maps – Matching Communities

- Example Configuration

```
router bgp 100
 neighbor 102.10.1.2 remote-as 200
 neighbor 102.10.1.2 route-map filter-on-community in
!
route-map filter-on-community permit 10
 match community 1
 set local-preference 50
!
route-map filter-on-community permit 20
 match community 2 exact-match
 set local-preference 200
!
ip community-list 1 permit 150:3 200:5
ip community-list 2 permit 88:6
```

Route Maps – Setting Communities

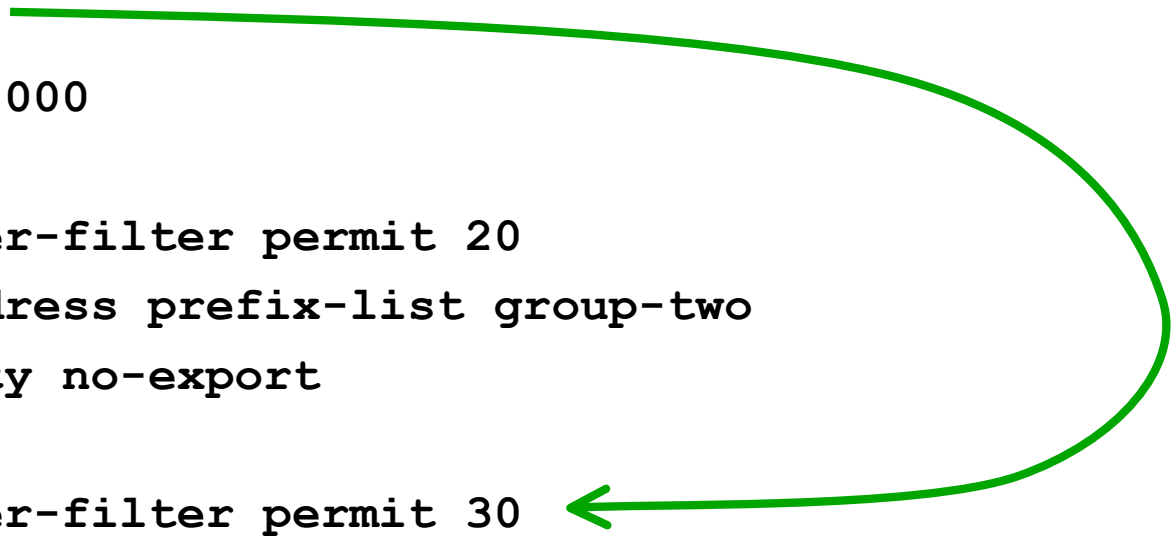
- Example Configuration

```
router bgp 100
 network 105.7.0.0 mask 255.255.0.0
 neighbor 102.10.1.1 remote-as 200
 neighbor 102.10.1.1 send-community
 neighbor 102.10.1.1 route-map set-community out
!
route-map set-community permit 10
 match ip address prefix-list NO-ANNOUNCE
 set community no-export
!
route-map set-community permit 20
 match ip address prefix-list AGGREGATE
!
ip prefix-list NO-ANNOUNCE permit 105.7.0.0/16 ge 17
ip prefix-list AGGREGATE permit 105.7.0.0/16
```

Route Map Continue

- Handling multiple conditions and actions in one route-map (for BGP neighbour relationships only)

```
route-map peer-filter permit 10
 match ip address prefix-list group-one
 continue 30
 set metric 2000
!
route-map peer-filter permit 20
 match ip address prefix-list group-two
 set community no-export
!
route-map peer-filter permit 30
 match ip address prefix-list group-three
 set as-path prepend 100 100
!
```



Managing Policy Changes

- New policies only apply to the updates going through the router AFTER the policy has been introduced or changed
- To facilitate policy changes on the entire BGP table the router handles the BGP peerings need to be "refreshed"

  This is done by clearing the BGP session either in or out, for example:

  ```
  clear ip bgp <neighbour-addr> in|out
  ```

- Do NOT forget in or out — doing so results in a hard reset of the BGP session

Managing Policy Changes

- Ability to clear the BGP sessions of groups of neighbours configured according to several criteria
- **`clear ip bgp <addr> [in|out]`**

  `<addr>` may be any of the following

x.x.x.x	IP address of a peer
`*`	all peers
ASN	all peers in an AS
`external`	all external peers
`peer-group` <*name*>	all peers in a peer-group

BGP Attributes and Policy Control

ISP/IXP Workshops

Supplementary Materials

Policy Control – Route Maps

- Route Map MATCH Articles
  - as-path
  - clns address
  - clns next-hop
  - clns route-source
  - community
  - interface
  - ip address
  - ip next-hop
  - ip route-source
  - length
  - metric
  - nlri
  - route-type
  - tag

Policy Control – Route Maps

- Route map SET Articles
  - as-path
  - automatic-tag
  - cln
  - comm-list
  - community
  - dampening
  - default interface
  - interface
  - ip default next-hop
  - ip next-hop

Policy Control – Route Maps

- Route map SET Articles
  - ip precedence
  - ip qos-group
  - ip tos
  - level
  - local preference
  - metric
  - metric-type
  - next-hop
  - nlri multicast
  - nlri unicast
  - origin
  - tag
  - traffic-index
  - weight

Aggregation Policies

- Suppress Map

  Used to suppress selected more-specific prefixes (e.g. defined through a route-map) in the absence of the **summary-only** keyword.

- Unsuppress Map

  Used to unsuppress selected more-specific prefixes per BGP peering when the **summary-only** keyword is in use.

Aggregation Policies – Suppress Map

- Example

```
router bgp 100
 network 102.10.10.0
 network 102.10.11.0
 network 102.10.12.0
 network 102.10.33.0
 network 102.10.34.0
 aggregate-address 102.10.0.0 255.255.0.0 suppress-map block-net
 neighbor 102.5.7.2 remote-as 200
!
route-map block-net permit 10
 match ip address prefix-list SUPPRESS
!
ip prefix-list SUPPRESS permit 102.10.8.0/21 le 32
ip prefix-list SUPPRESS deny 0.0.0.0/0 le 32
!
```

Aggregation Policies – Suppress Map

- show ip bgp on the local router

```
router1#sh ip bgp
BGP table version is 11, local router ID is 102.5.7.1
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
*> 102.10.0.0/16  0.0.0.0                         32768 i
s> 102.10.10.0    0.0.0.0                0          32768 i
s> 102.10.11.0    0.0.0.0                0          32768 i
s> 102.10.12.0    0.0.0.0                0          32768 i
*> 102.10.33.0    0.0.0.0                0          32768 i
*> 102.10.34.0    0.0.0.0                0          32768 i
```

Aggregation Policies – Suppress Map

- show ip bgp on the remote router

```
router2#sh ip bgp
BGP table version is 90, local router ID is 102.5.7.2
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
*> 102.10.0.0/16    102.5.7.1                        0  100 i
*> 102.10.33.0      102.5.7.1          0             0  100 i
*> 102.10.34.0      102.5.7.1          0             0  100 i
```

Aggregation Policies – Unsuppress Map

- Example

```
router bgp 100
 network 102.10.10.0
 network 102.10.11.0
 network 102.10.12.0
 network 102.10.33.0
 network 102.10.34.0
 aggregate-address 102.10.0.0 255.255.0.0 summary-only
 neighbor 102.5.7.2 remote-as 200
 neighbor 102.5.7.2 unsuppress-map leak-net
!
route-map leak-net permit 10
 match ip address prefix-list LEAK
!
ip prefix-list LEAK permit 102.10.8.0/21 le 32
ip prefix-list LEAK deny 0.0.0.0/0 le 32
```

Aggregation Policies – Unsuppress Map

- show ip bgp on the local router

```
router1#sh ip bgp
BGP table version is 11, local router ID is 102.5.7.1
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
*> 102.10.0.0/16  0.0.0.0                         32768 i
s> 102.10.10.0    0.0.0.0                0          32768 i
s> 102.10.11.0    0.0.0.0                0          32768 i
s> 102.10.12.0    0.0.0.0                0          32768 i
s> 102.10.33.0    0.0.0.0                0          32768 i
s> 102.10.34.0    0.0.0.0                0          32768 i
```

Aggregation Policies – Unsuppress Map

- show ip bgp on the remote router

```
router2#sh ip bgp
BGP table version is 90, local router ID is 102.5.7.2
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
*> 102.10.0.0/16    102.5.7.1                        0  100 i
*> 102.10.10.0      102.5.7.1         0              0  100 i
*> 102.10.11.0      102.5.7.1         0              0  100 i
*> 102.10.12.0      102.5.7.1         0              0  100 i
```

Aggregation Policies – Aggregate Address

- Summary-only used
  - all subprefixes suppressed
  - unsuppress-map to selectively leak subprefixes
  - bgp per neighbour configuration
- Absence of summary-only
  - no subprefixes suppressed
  - suppress-map to selectively suppress subprefixes
  - bgp global configuration