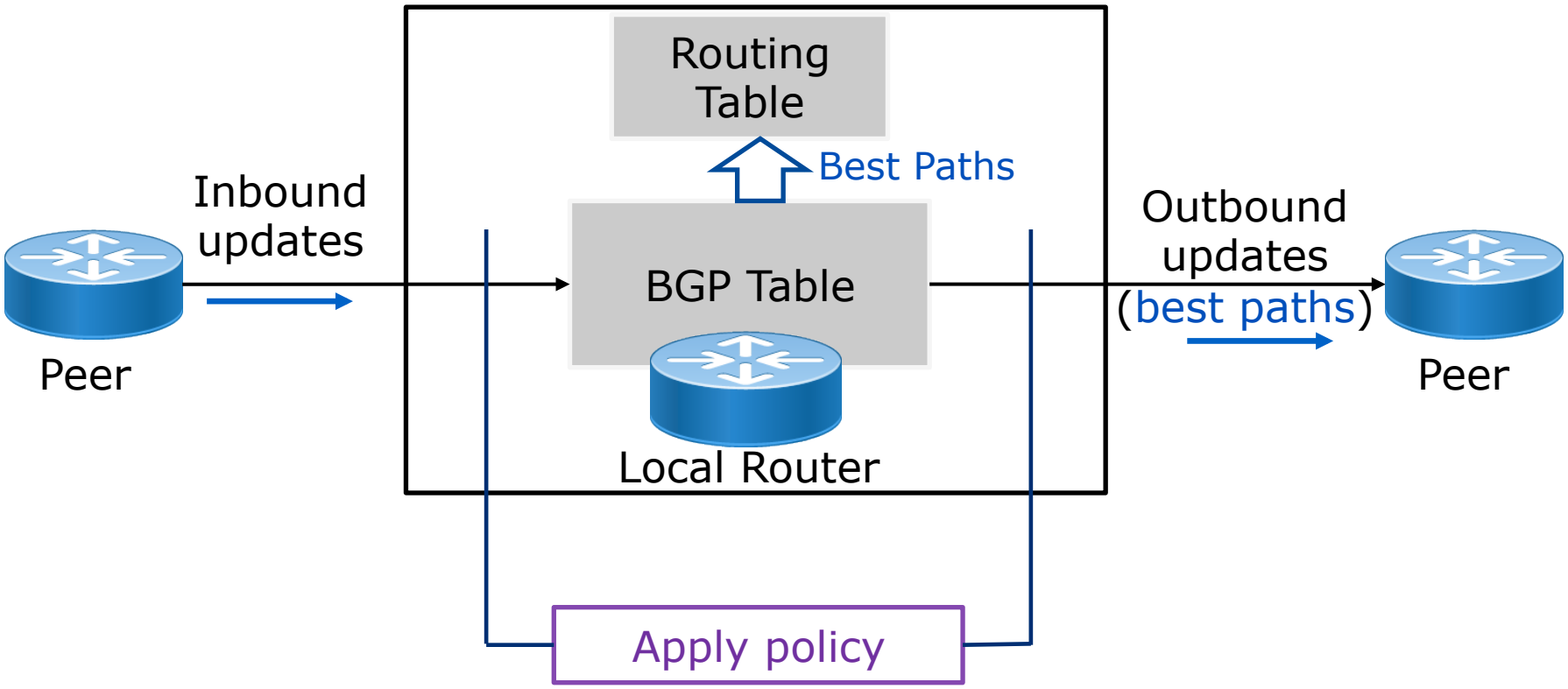
BGP Policy Control

APNIC

BGP policy control



Policy Control – Selecting prefixes

- Match via a prefix list filter
- Match via an AS path filter
- Match via a community tag filter
- Or some combination of the above

Policy Control – Setting attributes

- Change the AS path, via prepending
- Set a Local Preference value
- Set a MED value
- Add or remove community tags

Prefix lists

- Typically allow you to select a prefix or a range of prefixes
- Might also select an aggregate and some sub-aggregates
- Could be built using a tool from data held in an Internet Routing Registry
  - IRRToolSet (rtconfig)
  - BGPQ3

Cisco IOS prefix list filters

```
ip prefix-list name/num [seq#] permit | deny
prefix/length [ge value][le value]
```

- Example 1:

```
ip prefix-list TEST permit 0.0.0.0/0 ge 8 le 24
```

  - Allows any prefix with prefix length between 8 and 24
  - Implicit DENY at the end!

- Example 2:

```
ipv6 prefix-list TEST-v6 permit 2001:6400::/32 le 48
```

  - Permit the prefix 2400:6400::/32 up to /48
  - Implicit DENY at the end!

Cisco IOS prefix list filters

- Example 3:

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

  – Deny default route

- Example 4:

```
ipv6 prefix-list TEST-v6 deny ::/0
```

  – Deny IPv6 default routes

AS path filters

- Regular expressions are used to select routes based on components in their AS path, or the whole path

```
.     Matches any one character
*     Matches any sequence of pattern before *
+     match at least one preceding expression
^     beginning with
$     ending with
_     matches start, end, space, comma, braces
()    to contain expression
[]    to contain number ranges
```

- Typically used when prefix lists are too long
- Use anchors to match against start or end of the path

Cisco IOS AS path access list

- Example regular expressions:

  ^$ locally originated routes

  _2914$ originated by AS 2914

  _4637_174_ passing through 4637 and 174

  ^(_1273)+$ originated by 1273, multiple occurrence

  ^701(_[0-9]+)*_(5539|8495|8763)$

  Received from 701 and originated from either 5539, 8495 or 8763 while passing through any ASes

- Example:

  ip as-path access-list 10 permit ^(_1221)+(_2764)*$

Community access lists

- If you define and use communities these can be used to select prefixes
- Set a community when you import a prefix into BGP
- Set a community when you learn a prefix from a peer
- Select prefixes based on the presence of a community
- Be careful to not allow peers to set your communities if you use them for filtering
  - e.g. tagging their prefixes as your “customer” routes
- Communities have global scope unless you explicitly delete them on export

Collecting policy together

- While prefix lists and AS path access lists can be applied directly it is more common to use policy statements to collect the various components together
- Cisco IOS – route-map
- Junos – policy-statement
- Huawei – route-policy

Cisco IOS Route-Map

route-map *name* [**permit | deny**] [*sequence*]

- Default is permit
  - Implicit DENY at the end!

```
route-map TEST permit 10
 match A B C
 match D
 set X
 set Y
route-map TEST permit 20
 match E
 set Z
route-map TEST permit 30
```

```
If {(A or B or C)
and D} match
Then {set X and Y}
```

```
Else
If E matches
Then set Z
```

```
Else (for everything else)
Do/set nothing
```

Match (conditions) & Set (actions)

Command	Description
`match community`	BGP community tag
`match as-path`	AS-path access list
`match ip address`	Access list or prefix-list

Command	Description
`set as-path <prepend>`	Modify AS-path
`set community`	Apply BGP community tag
`set metric`	Modify MED
`set local-preference`	Modify local preference

Route Map

```
router bgp 17821
 neighbor 30.30.30.1 remote-as 30
 neighbor 30.30.30.1 route-map AS-OUT out
 neighbor 30.30.30.1 route-map LP-IN in
!
route-map AS-OUT permit 10
 set as-path prepend 17821 17821 17821
!
route-map LP-IN permit 10
 match as-path 1
 set local-preference 150
!
route-map LP-IN permit 20
!
ip as-path access-list 1 permit _30$
```

Route Map

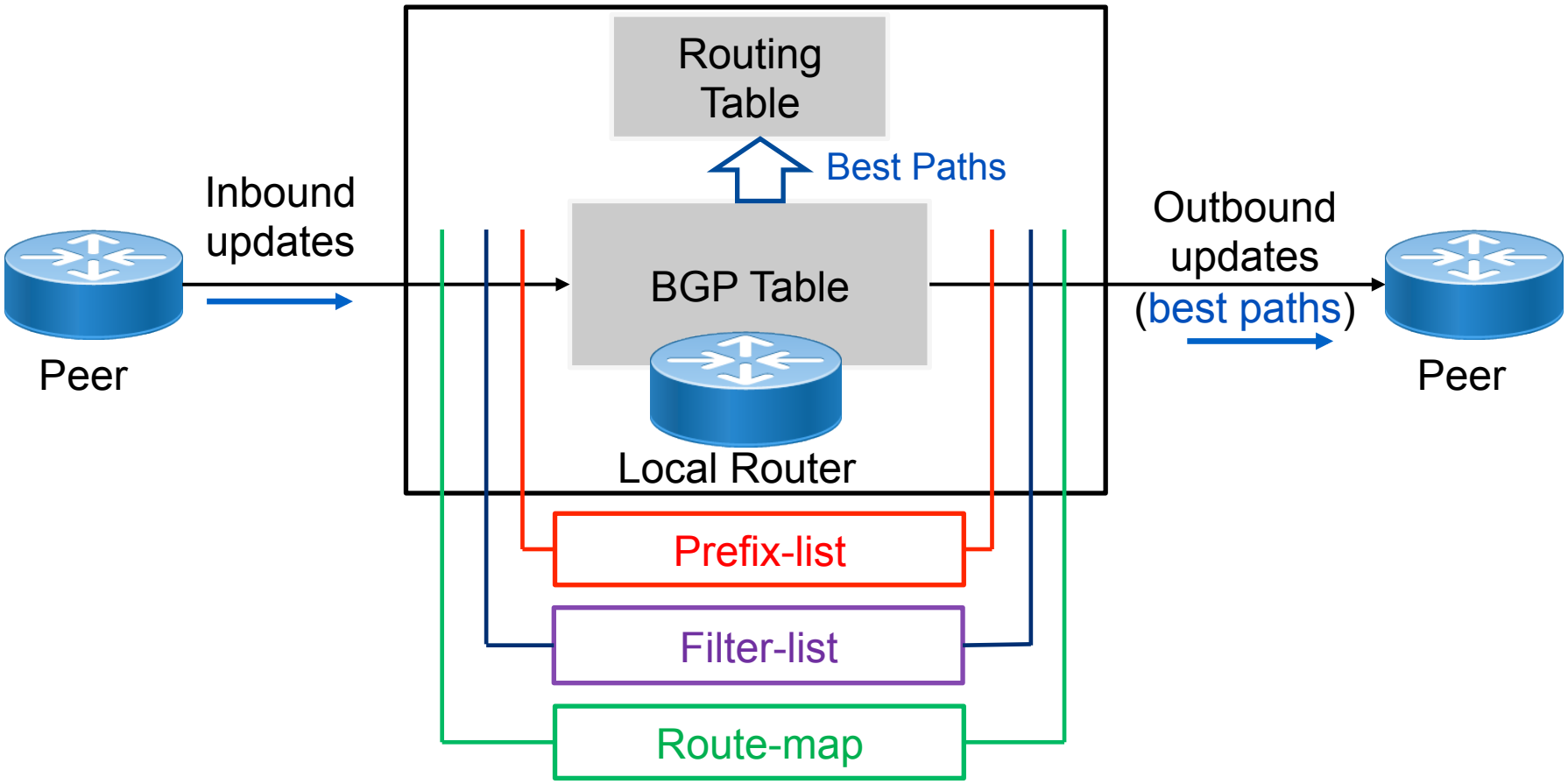
- Setting and Matching communities:

```
router bgp 17821
 network 100.100.0.0 mask 255.255.224.0 route-map SET-AGG
 neighbor 20.20.20.1 remote-as 20
 neighbor 20.20.20.1 send-community
 neighbor 20.20.20.1 route-map TR-IN in
!
route-map SET-AGG permit 10
 set community 100:1000
!
route-map TR-IN permit 10
 match community 5
 set local-preference 150
!
route-map TR-IN permit 20
!
ip community-list 5 permit 20:3000
ip community-list 5 permit 20:4000
```

Applying Policy Filters

- Incoming/Outgoing updates are filtered through policies
  - BGP table does not contain routes rejected by policies
- Whenever there is a BGP policy change, we need to
  - Trigger an update to force in/outbound routes through the new filters (else only the ones already in BGP table)
  - either through a Hard Reset or a Soft Reset (Route Refresh)
- If the filter is applied to:
  - Outbound routes: need to resend its BGP table through the filter
  - Inbound routes: need its neighbors to resend their BGP tables

Cisco IOS order of processing



Hard Reset

- Hard reset of a BGP session
  - Tears down the TCP connection
  - Re-establish the TCP session
  - Resend the BGP table to neighbors affected by the reset
  - Relearn all routes from neighbors

```
clear ip bgp *
clear ip bgp <peer-address>
clear bgp ipv6 unicast *
clear bgp ipv6 unicast <peer-address>
```

  - Disrupts network connectivity
    - Same as a router reboot!

Route Refresh Capability

- RFC 2819, RFC 7313
- Negotiated capability between two BGP speakers
- Does not tear down the BGP session
- Either
  - Requests the peer to resend all its route entries in Adj-RIB-Out
  - Sends all your route entries in Adj-RIB-Out to the peer

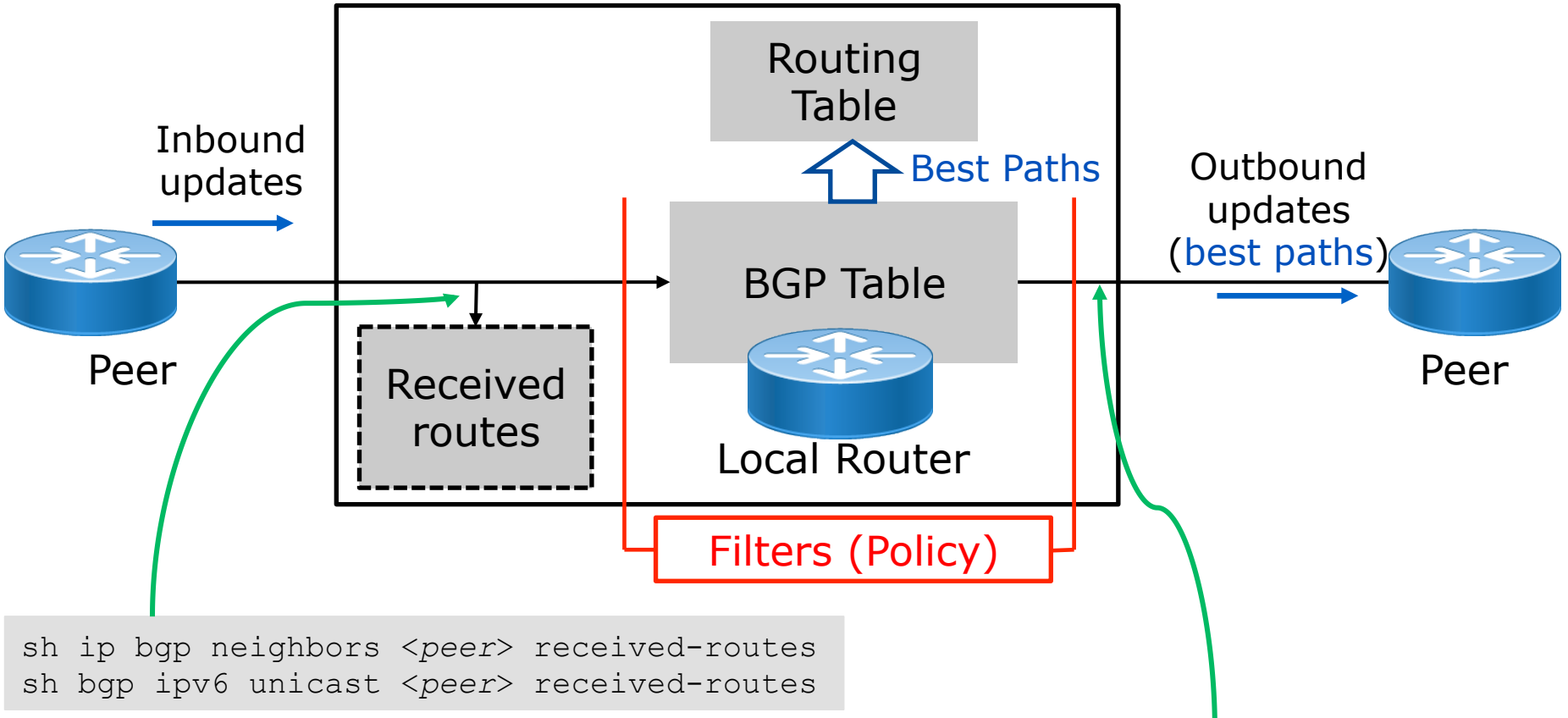
- Examples:
  - Huawei: refresh bgp ipv6 <peer-address> export
  - Cisco: clear ip bgp <peer-address> soft in

Soft Reconfiguration

- All current routers should support route refresh
  - If the router (local or peer) does not have route refresh capability, use soft-reconfiguration

- With soft-reconfiguration, the router stores a copy of the received routes in addition to the BGP table (allowed by policy filters)
  - Thus, requires additional memory!

Soft Reconfiguration




```
sh ip bgp neighbors <peer> received-routes
sh bgp ipv6 unicast <peer> received-routes
```

```
sh ip bgp neighbors <peer> advertised-routes
sh bgp ipv6 unicast <peer> advertised-routes
```

Soft Reconfiguration

```
router bgp 17821
 neighbor 1.1.1.1 remote-as 100
 neighbor 1.1.1.1 soft-reconfiguration inbound
```

– Whenever there is change in policy

```
clear ip bgp 1.1.1.1 soft [in|out]
```

- If “in”, runs the stored received routes through the new filter
- If ”out”, sends the BGP table through the filters
- Does not tear down the BGP session!

– Route refresh capability cannot be used if soft-reconfig is used!


Questions