Advanced Computer Networks

External Routing - BGP protocol

Prof. Andrzej Duda

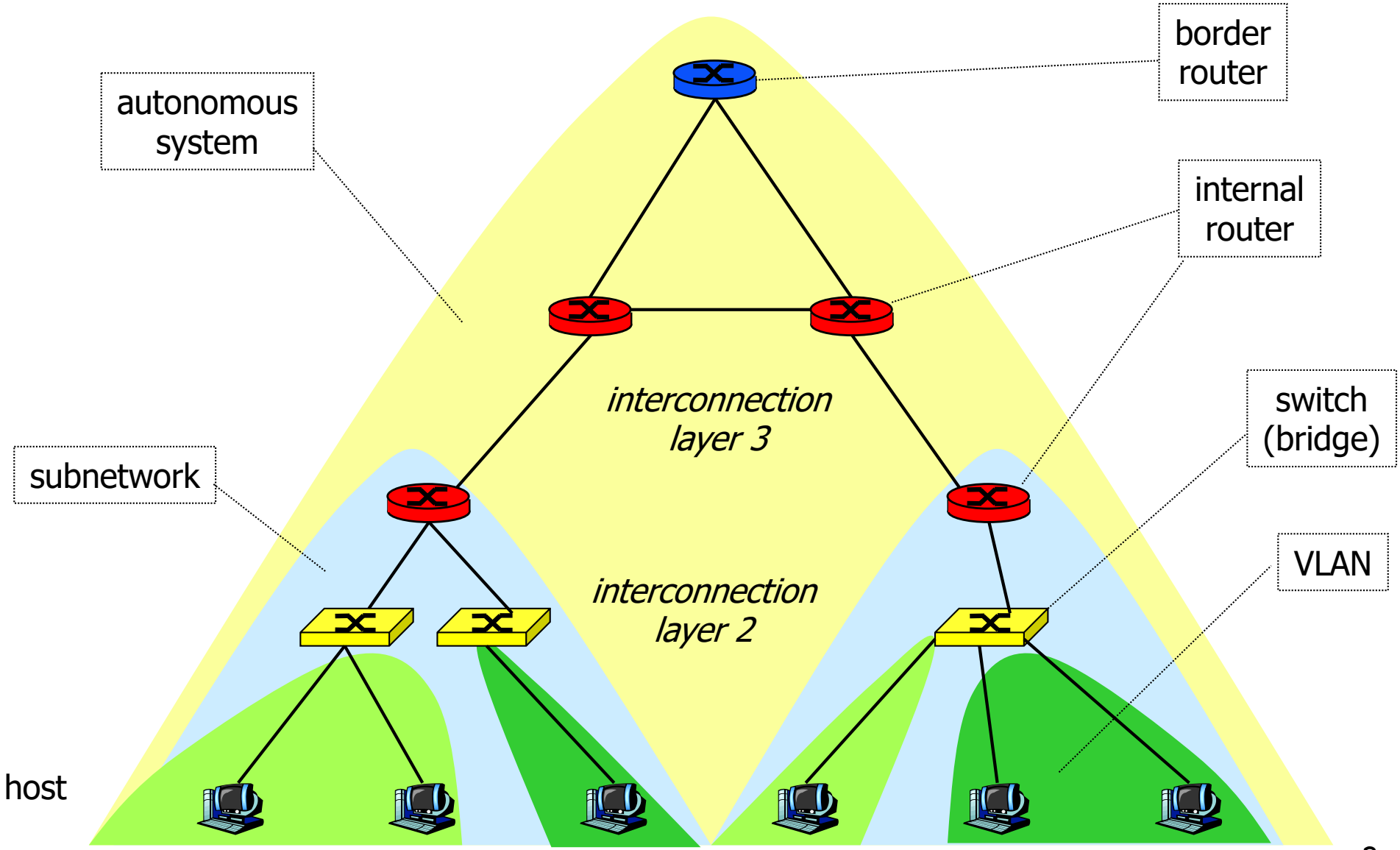
duda@imag.fr

http://duda.imag.fr

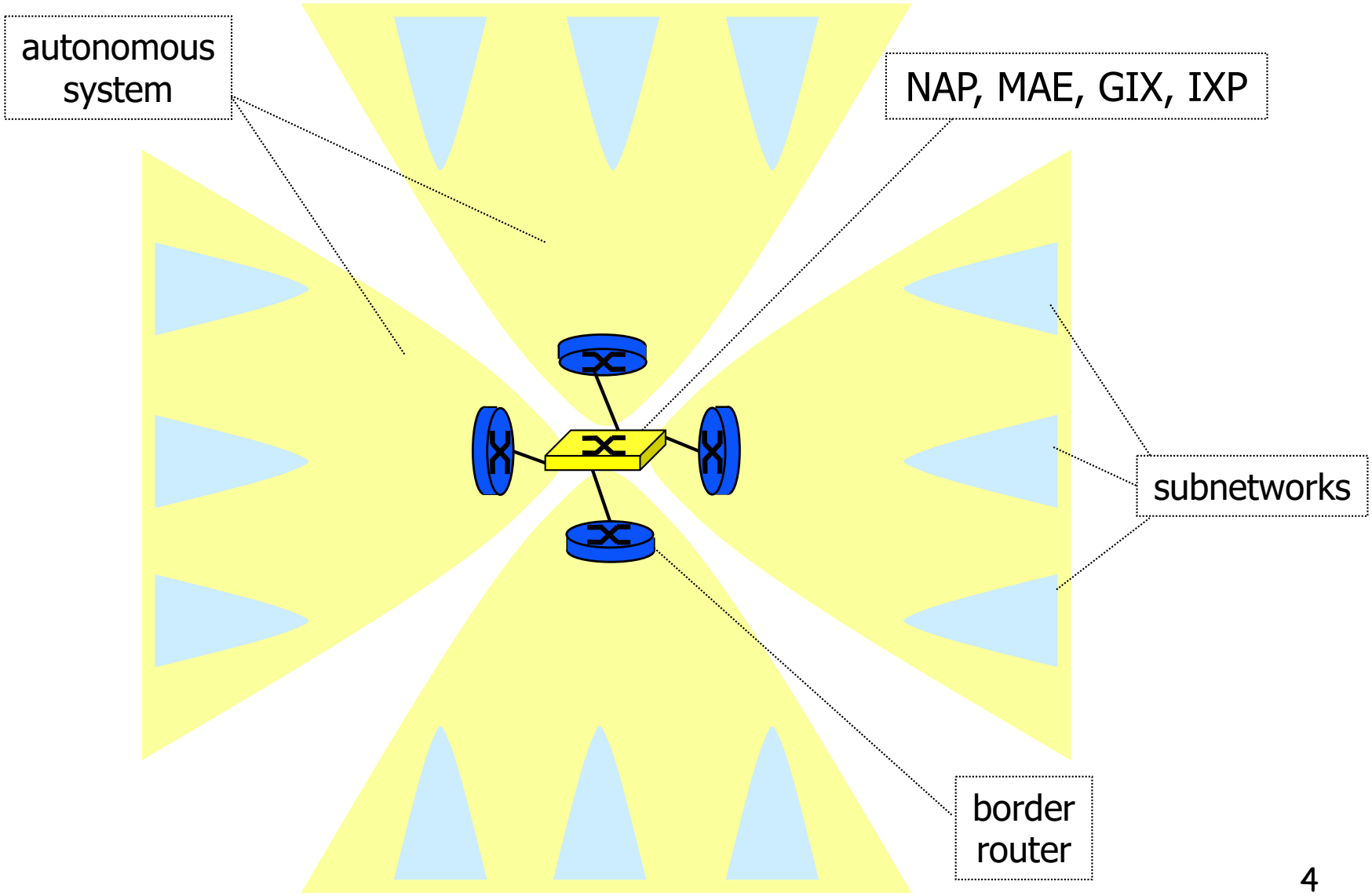
Contents

- Principles of Inter-Domain Routing
  - Autonomous systems
  - Path vector routing
  - Policy Routing
  - Route Aggregation, Anycast
- How BGP works
  - Attributes of routes, route selection
  - Interaction BGP-IGP-Packet forwarding
  - Other mechanisms
  - Filtering
  - Security: ROV-RPKI
- Examples
- Illustrations and statistics

Autonomous systems



Interconnection of AS

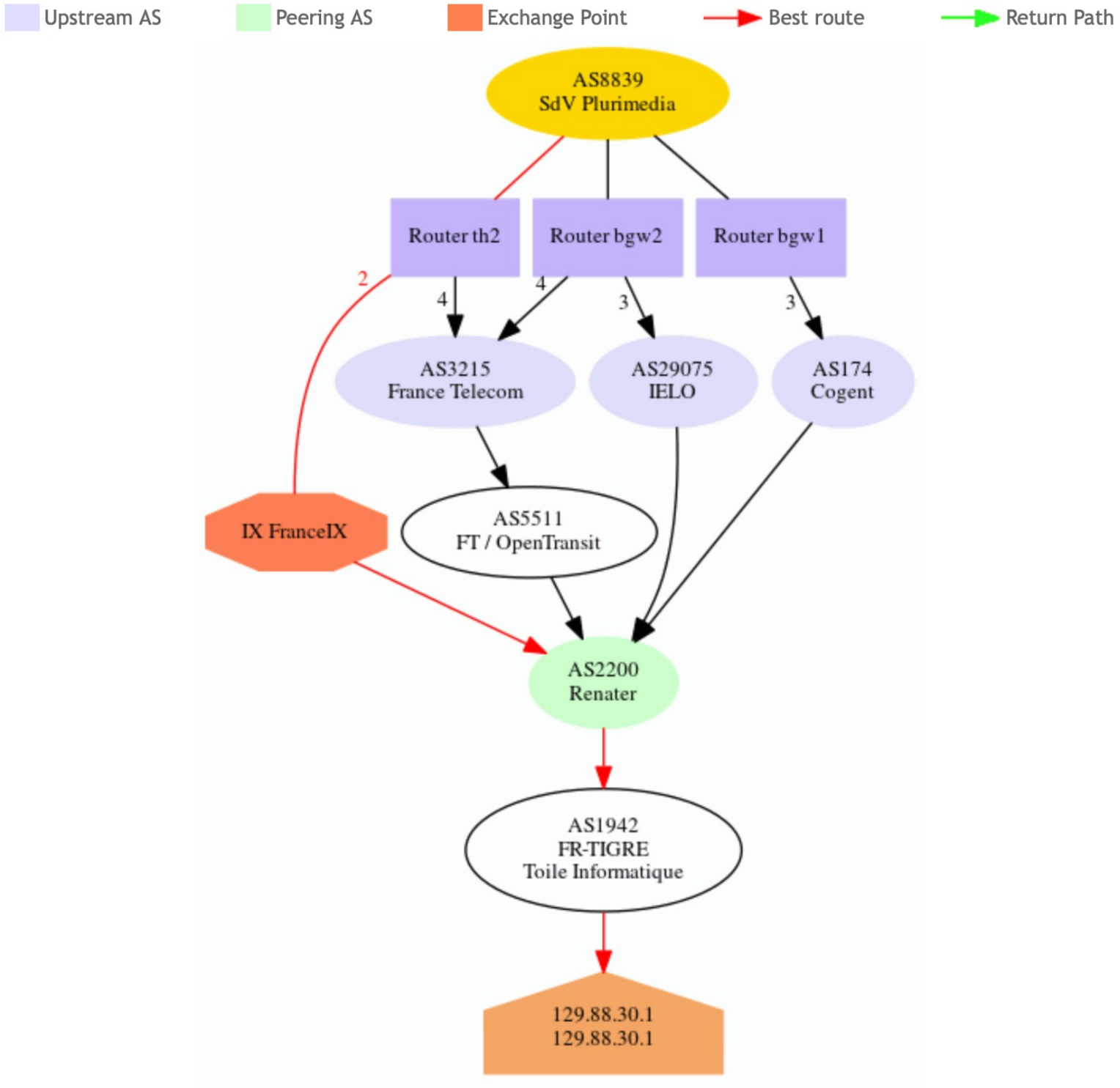


Autonomous Systems

- Routing domain under one single administration
  - one or more border routers
  - all subnetworks should be connected - run an interior gateway protocol (IGP like OSPF) to be able to forward packets within the AS
  - should learn about all other prefixes - use an exterior gateway protocol (EGP like BGP) to route packets to other AS
  - autonomy of management
- BGP – interdomain routing (domain = AS)

AS numbers

- AS number
  - 16 bits, extended to 32 bits: x.y
  - 0.y – old 16 bits numbers, 1.y - reserved
  - public: 1 - 64511
  - private: 64512 - 65535
  - ASs that do not need a number are typically those with a default route to the rest of the world
- Examples
  - AS1942 - CICG-GRENOBLE, AS2200 - Renater
  - AS559 - SWITCH Teleinformatics Services (EPFL)
  - AS5511 – OPENTRANSIT
  - AS174 – COGENT
  - AS1299 – TELIA SONERA
  - AS3356 – LEVEL 3
  - AS32934 - Facebook


Upstream AS
Peering AS
Exchange Point
Best route
Return Path
AS8839
SdV Plurimedia
Router th2
Router bgw2
Router bgw1
2
4
4
3
3
AS3215
France Telecom
AS29075
IELO
AS174
Cogent
IX FranceIX
AS5511
FT / OpenTransit
AS2200
Renater
AS1942
FR-TIGRE
Toile Informatique
129.88.30.1
129.88.30.1

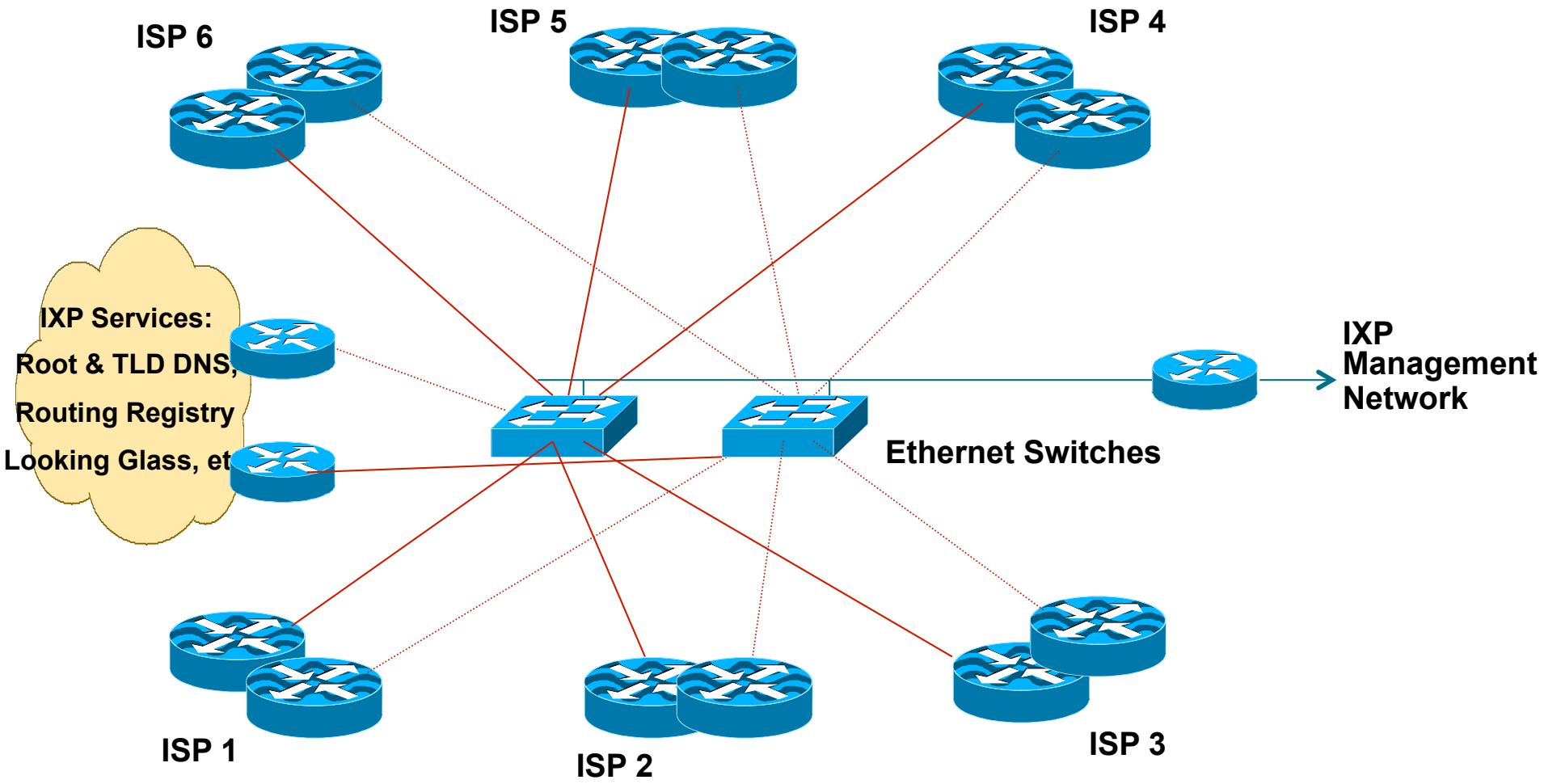
Interconnection of AS

- Border routers
  - interconnect AS
  - advertise routes to internal subnetworks
    - AS accepts the traffic
    - there is an internal route to the destination - AS is able to forward packets to the destination, otherwise - black hole
  - learn routes to external subnetworks
- Interconnection point
  - NAP (Network Access Point), MAE (Metropolitan Area Ethernet), CIX (Commercial Internet eXchange), GIX (Global Internet eXchange), IXP, SFINX, LINX
  - exchange of traffic - peering contract between ASs
- High-speed local area network connecting border routers of ASs

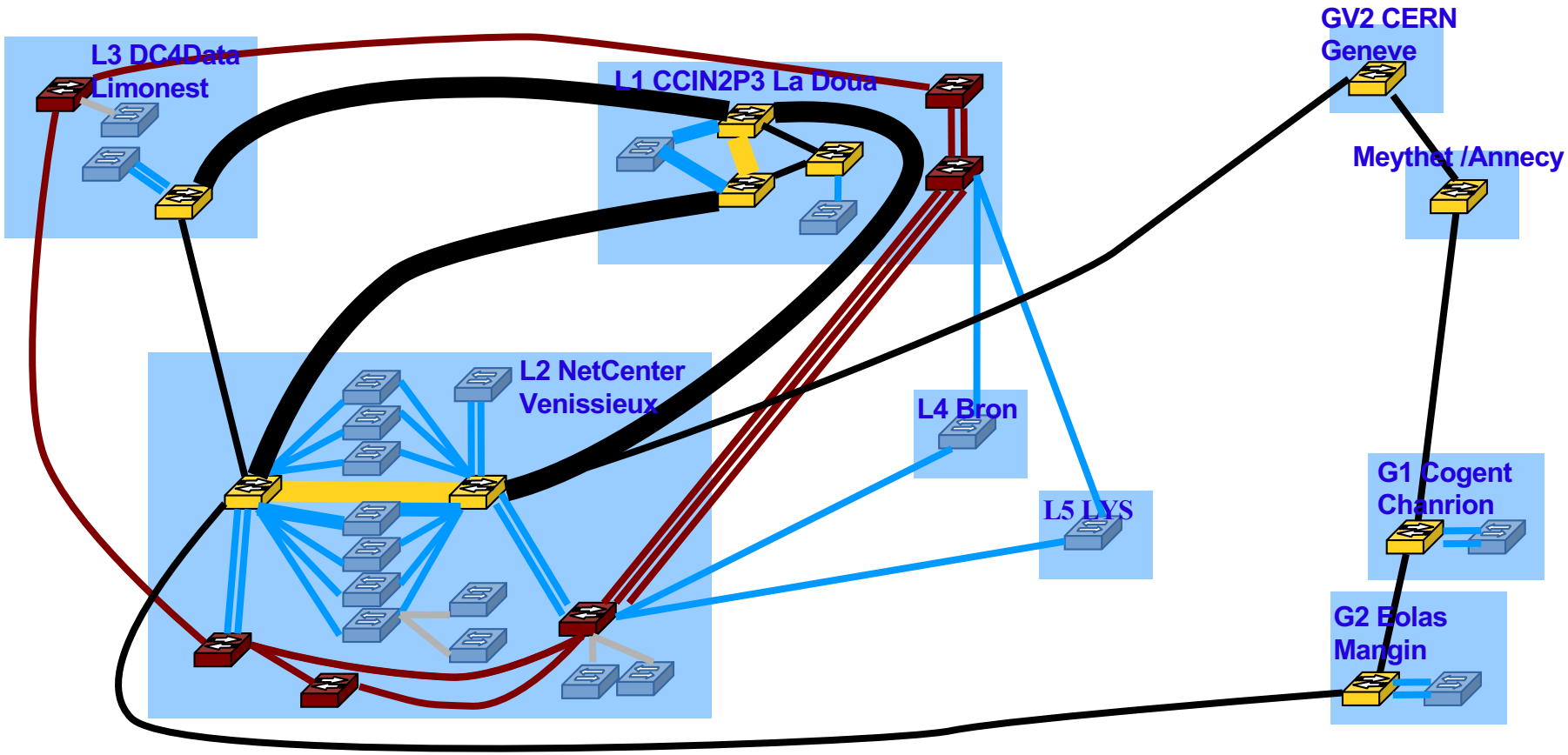
IXP – Internet Exchange Point

- Neutral location where network operators freely interconnect their networks to exchange traffic
- Ethernet switch in a neutral location
- IXP Operator provides the switch and rack space
- Network Operators bring routers, and interconnect them via the IXP fabric
- Every participant has to buy just one whole circuit from their premises to the IXP
- All Network Operators are peers – each participant configures external BGP directly with the other participants in the IXP
  - Peering with all participants or
  - Peering with a subset of participants

IXP – Internet Exchange Point

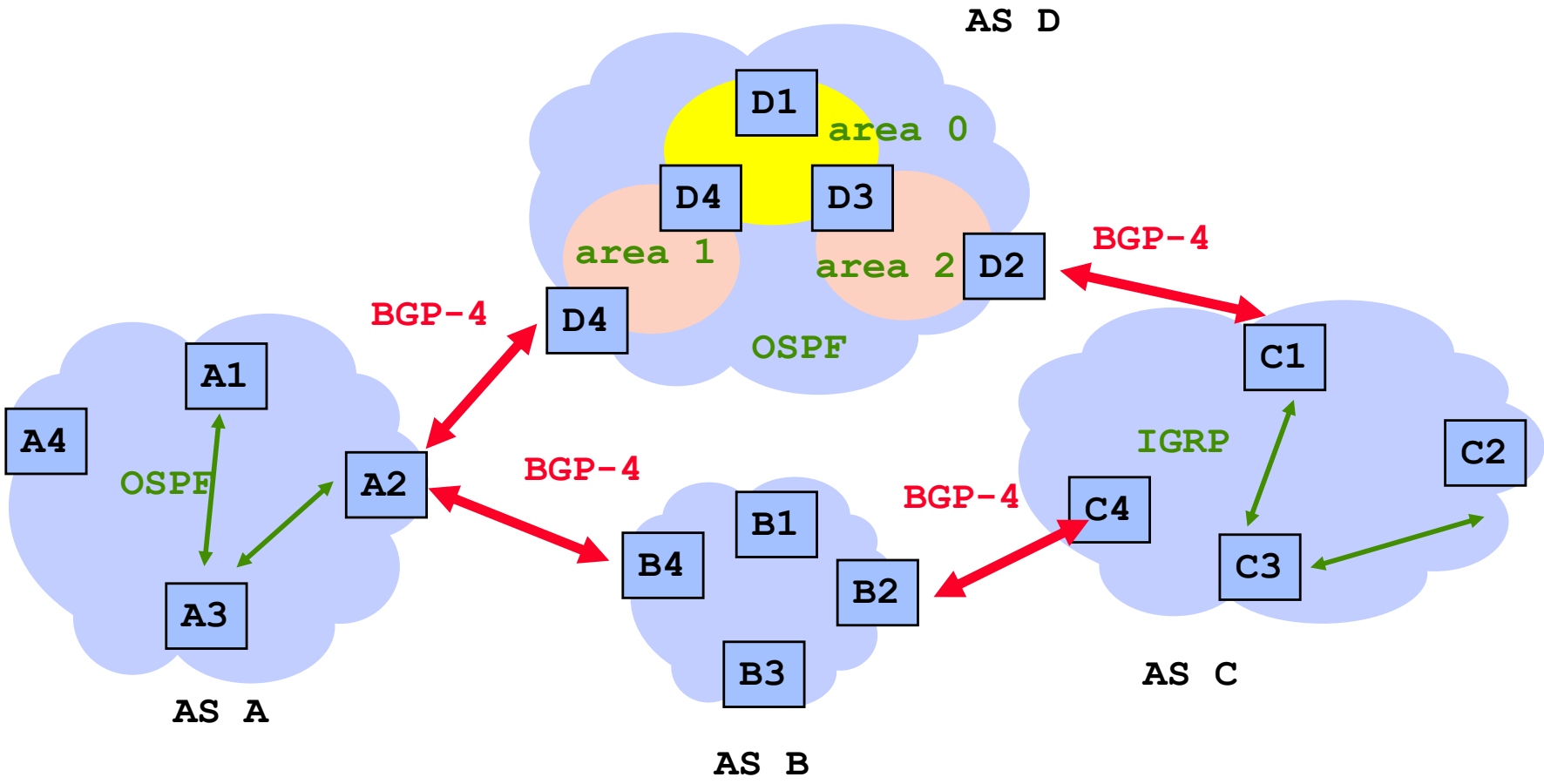


Internet Exchange Point: a rather huge Layer-2 fabric



Example interconnection

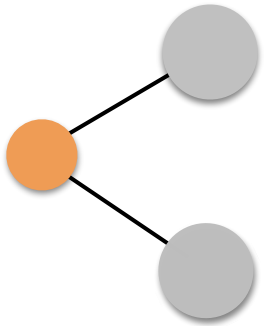
- AS can be transit (B and D), stub (A) or multihomed (C). Only non stub AS needs a number.



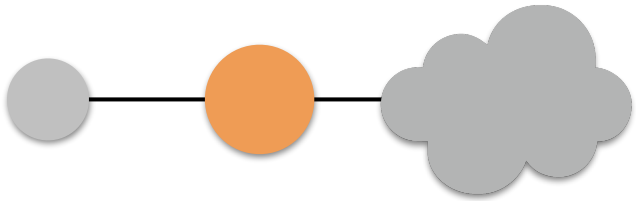
AS Types

Stub

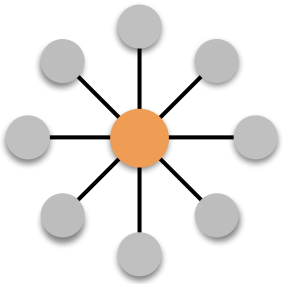
Multi homed



Transit

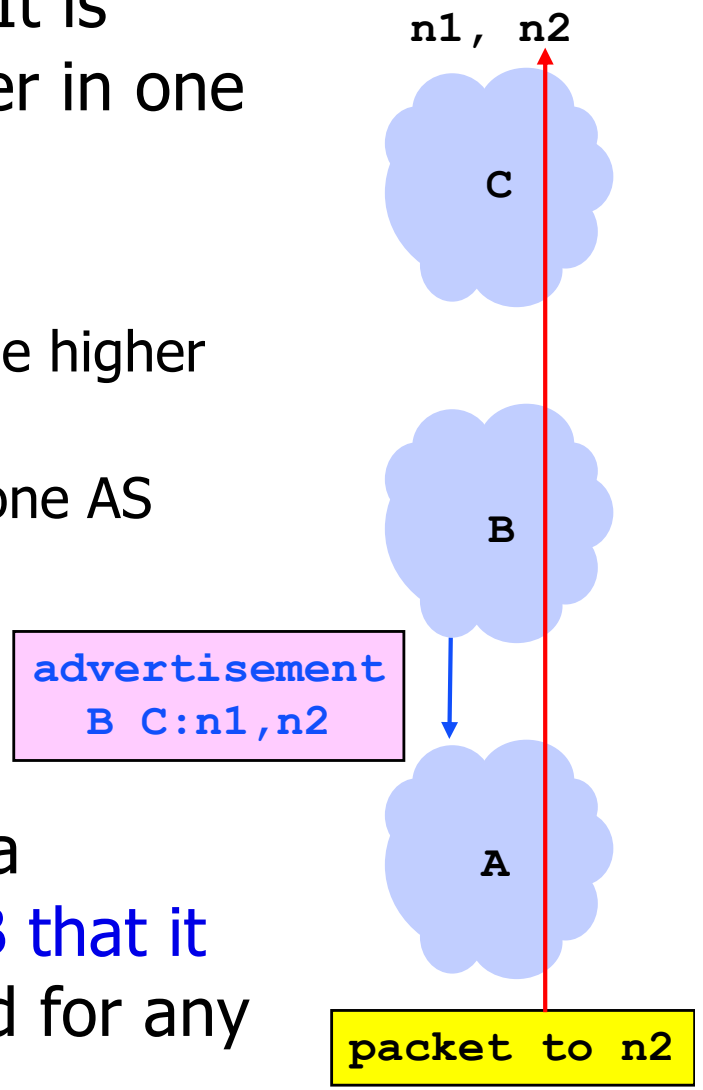


IXP - Internet Exchange Point

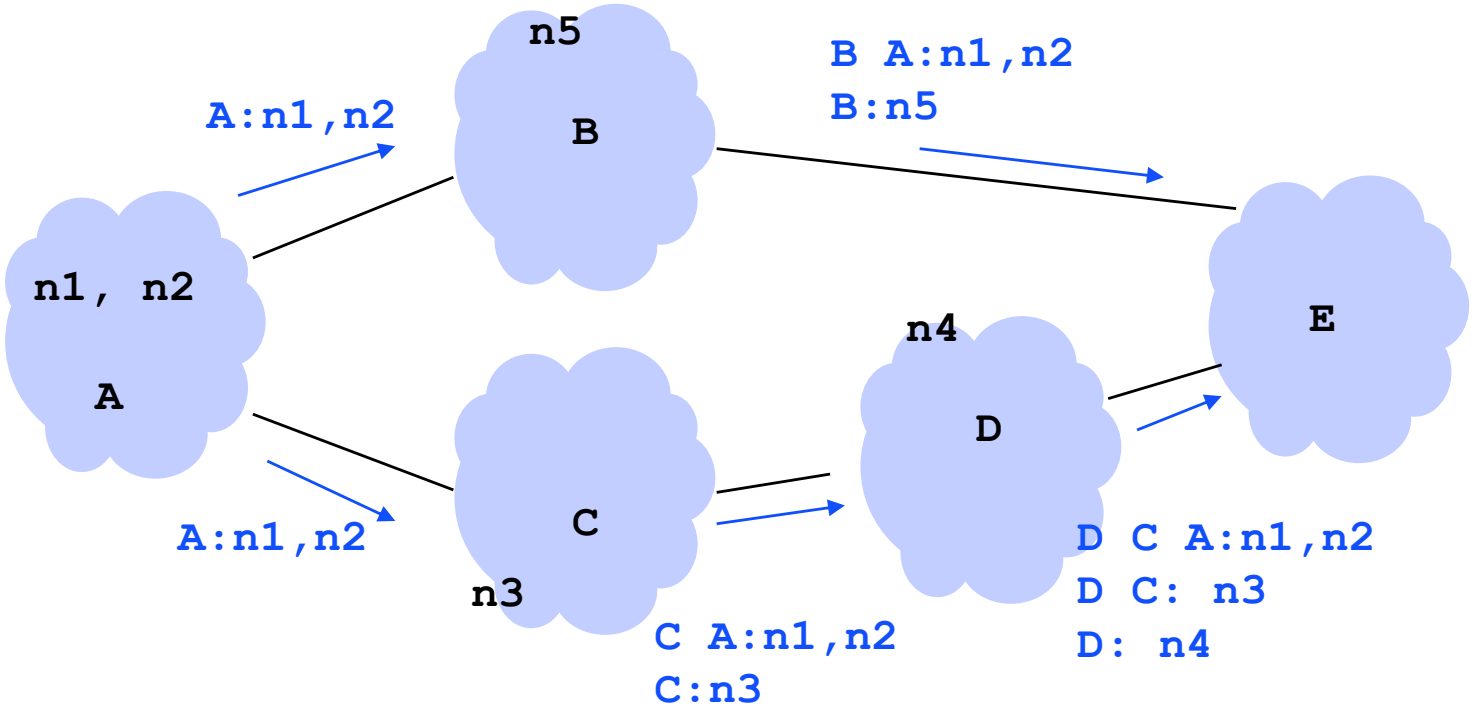


What does BGP do?

- BGP is a routing protocol between AS. It is used to establish routes from one router in one AS to any network prefix in the world
- There are two levels in BGP:
  - Inter-domain: one AS is a virtual node in the higher layer
  - Intra-domain: distribution of routes inside one AS
- The method of routing is
  - Path vector
  - With policy
- A route advertisement from B to A for a destination prefix is an agreement by B that it will forward packets sent via A destined for any destination in the prefix.



Path Vector routing



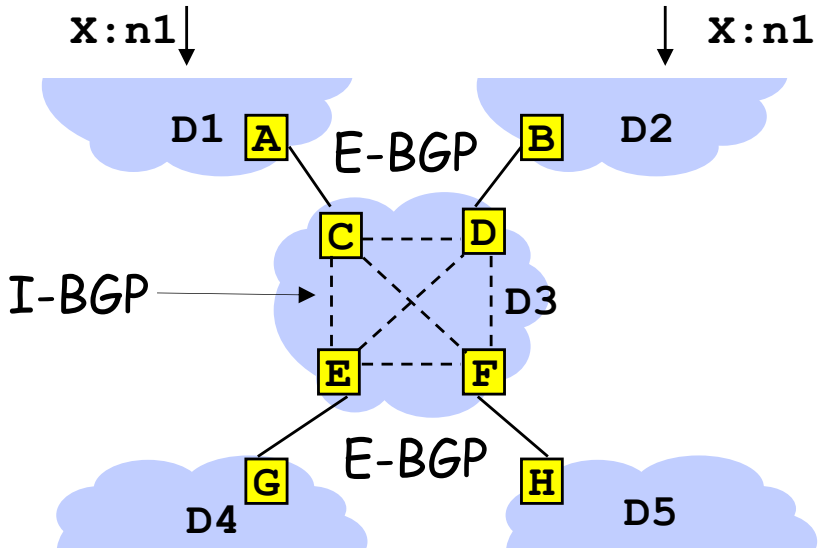
BGP table in E

dest	AS path
n1	B A
n2	B A
n3	D C
n4	D
n5	B

- AS maintains a table of best paths known so far
- Table updated using local rules
- Suitable when
  - no global meaning for costs can be assumed (heterogeneous environments)
  - global topology is fairly stable

Border Routers, E-BGP and I-BGP

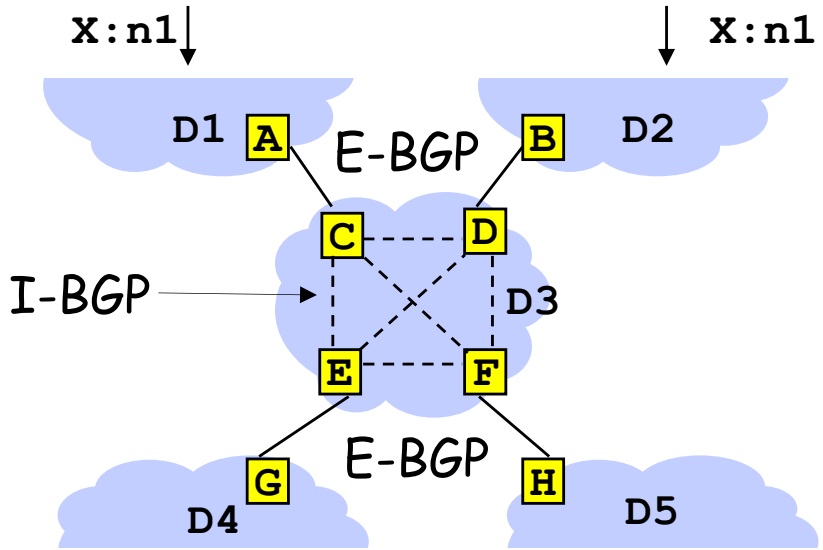
- E-BGP: BGP runs on *border routers = "BGP speakers"* belonging to one AS only
  - two border routers per boundary (OSPF - one per area boundary)
- I-BGP: BGP speakers talks to each other inside the AS using "Internal-BGP"
  - full mesh called the "BGP mesh"
  - I-BGP is the same as E-BGP except: 1) routes learned from a neighbour in the mesh are not repeated inside the mesh, 2) router does not prepend own AS number over I-BGP




```
A->C:   D1,X: n1
C->E:   D1,X: n1
C->D:   D1,X: n1
C->F:   D1,X: n1
E->G:   D3,D1,X: n1
```

Border Routers, E-BGP and I-BGP

- Which BGP updates may be sent ?
  - **C -> A: D3 D2 X: n1?**
  - **D -> E: D2 X: n1?**
  - **C -> E: D2 X: n1?**



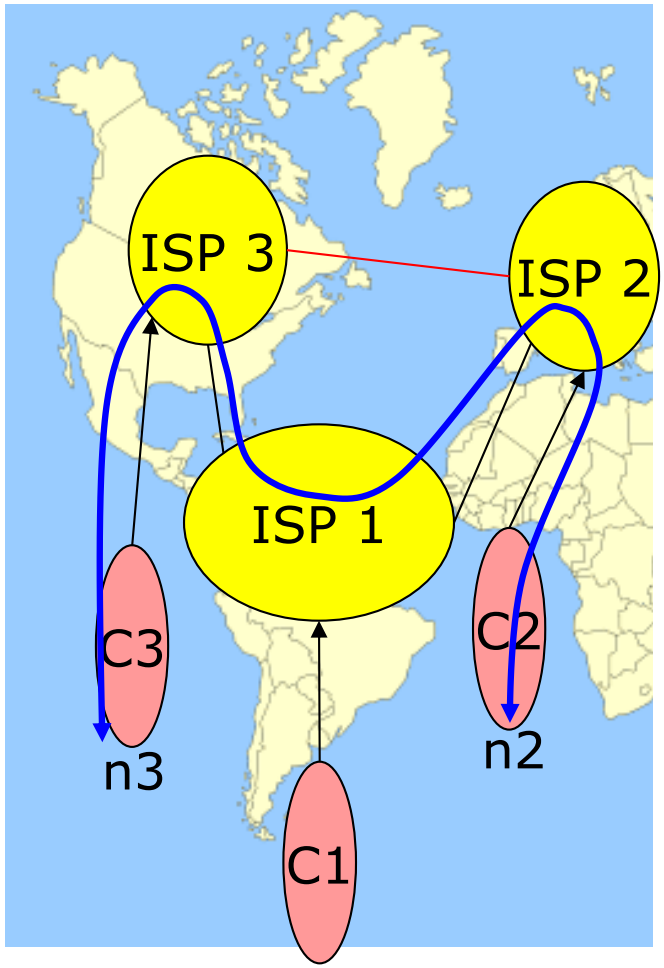
BGP General Operation

- Learns multiple candidate routes via internal and external BGP speakers
- Applies the decision process: selects THE best route and installs it in the IP routing table
- Policies applied by influencing the best route selection
- BGP speaker advertises (exports) only the routes that it uses itself
  - "hop-by-hop" routing paradigm
- From eBGP -> advertise to all other speakers over iBGP
- From iBGP -> advertise only to eBGP
  - full iBGP mesh is required!!
- Propagate ONLY the best routes

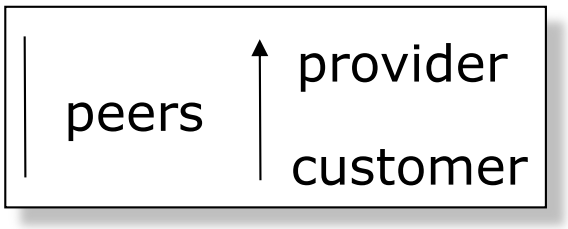
Policy Routing

- Mainly 3 types of relations depending on money flows
  - **Customer**: EPFL is customer of Switch. EPFL pays Switch
  - **Provider**: Switch is provider for EPFL; Switch is paid by EPFL
  - **Peer**: EPFL and CERN are peers: costs of interconnection is shared
- Money flow
  - Customer pays provider
  - Peers don't pay each other (shared costs)
    - Exchange roughly equal traffic (see below)
- Type of relation is negotiated in bilateral agreements there is no architecture rule, just business
- Use additional rules such as:
  - maximize performance (smallest AS path length?)
  - minimize use of my network bandwidth ("hot potato")

Typical Policy Routing Rules



- Motivating example:
  - ISP3 – ISP2: expensive transantlantic link
  - It is advantageous for ISP3 to send traffic to n2 via ISP1
  - But... ISP1 does not agree to carry traffic from C3 to C2
  - ISP1 offers a "transit service" to C1 and a "non-transit" service to ISP2 and ISP3
- Goal of "policy routing": support this arrangement
- The rules are defined by every AS and implemented in all BGP speakers in one AS



Two Principles For Typical Policies

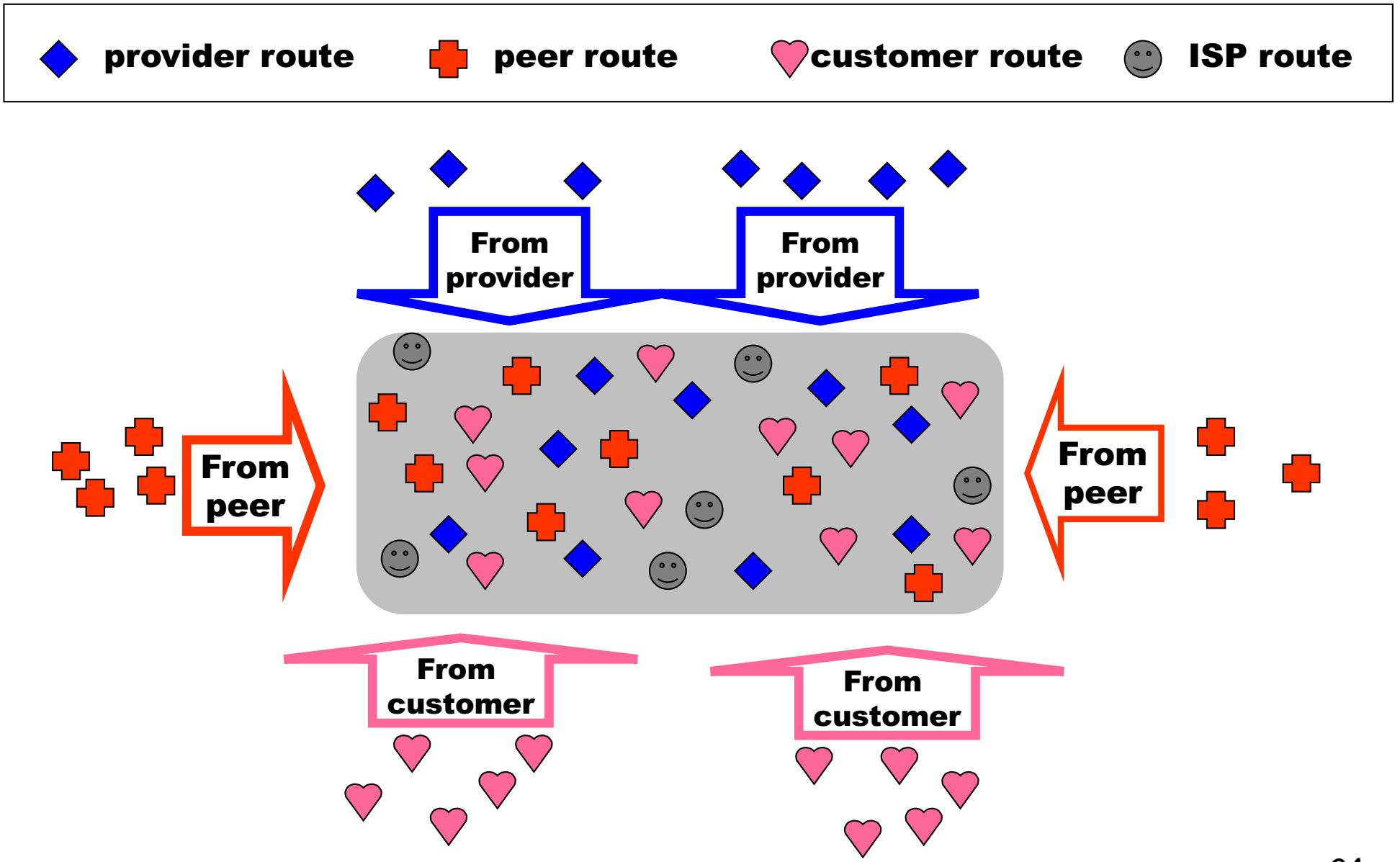
- **#1: Don't carry traffic if you are not being paid!**
  - Traffic should come from or go to customer
  - This is about what traffic I ***carry***
    - *Determined by who I tell about my routes*
- **#2: Save/make money when sending traffic**
  - Prefer sending traffic to customer
  - If can't do that, then a peer.
  - This is about where I ***send*** traffic
    - *Determined by routing choices I make*

Implementing Customer/Provider and Peer/Peer relationships

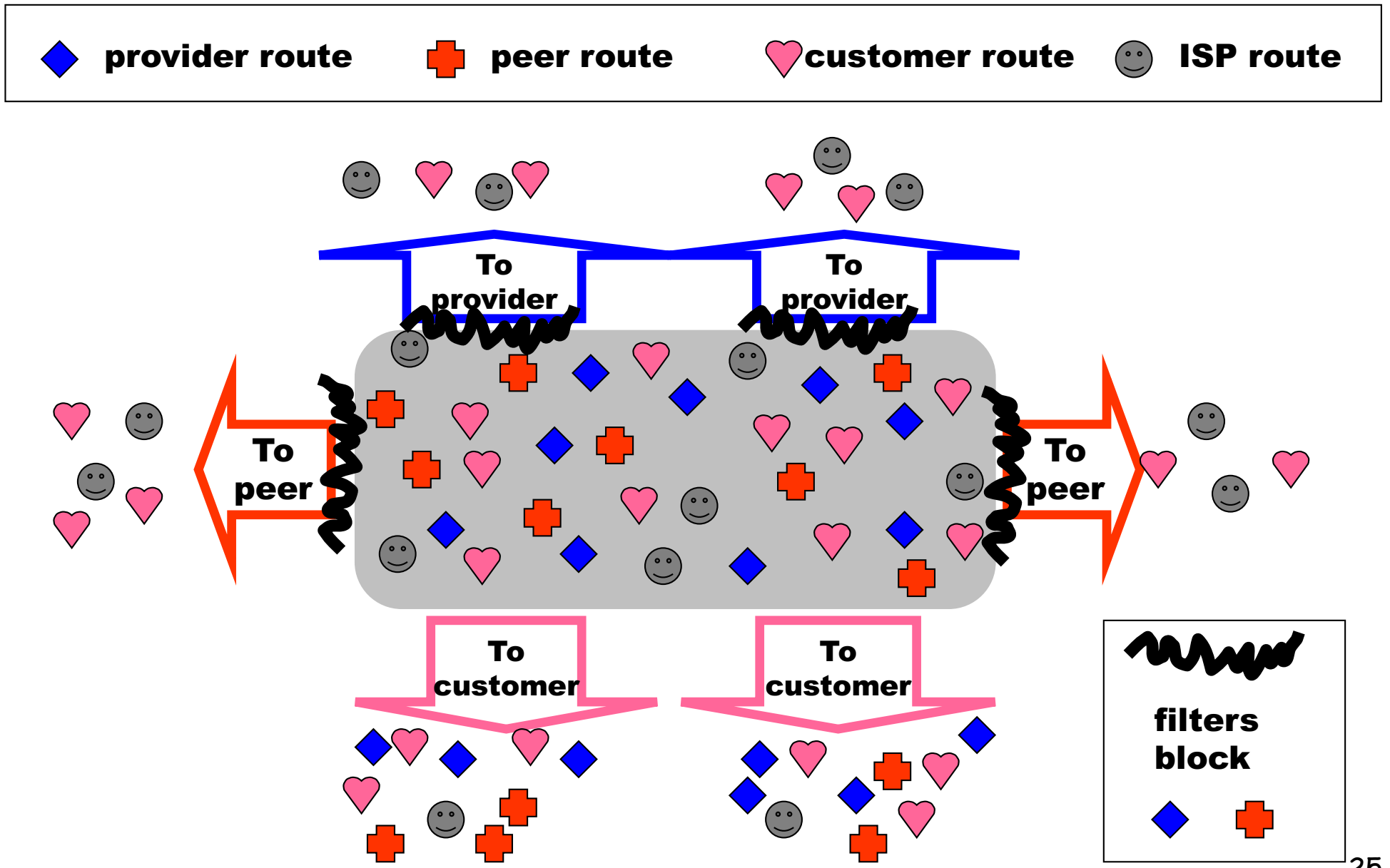
Two parts:

- Import: enforce order of route preference (money!)
  - customer > peer > provider
- Export: enforce transit relationships
  - routes coming from Peers and Providers are not propagated to Peers nor Providers

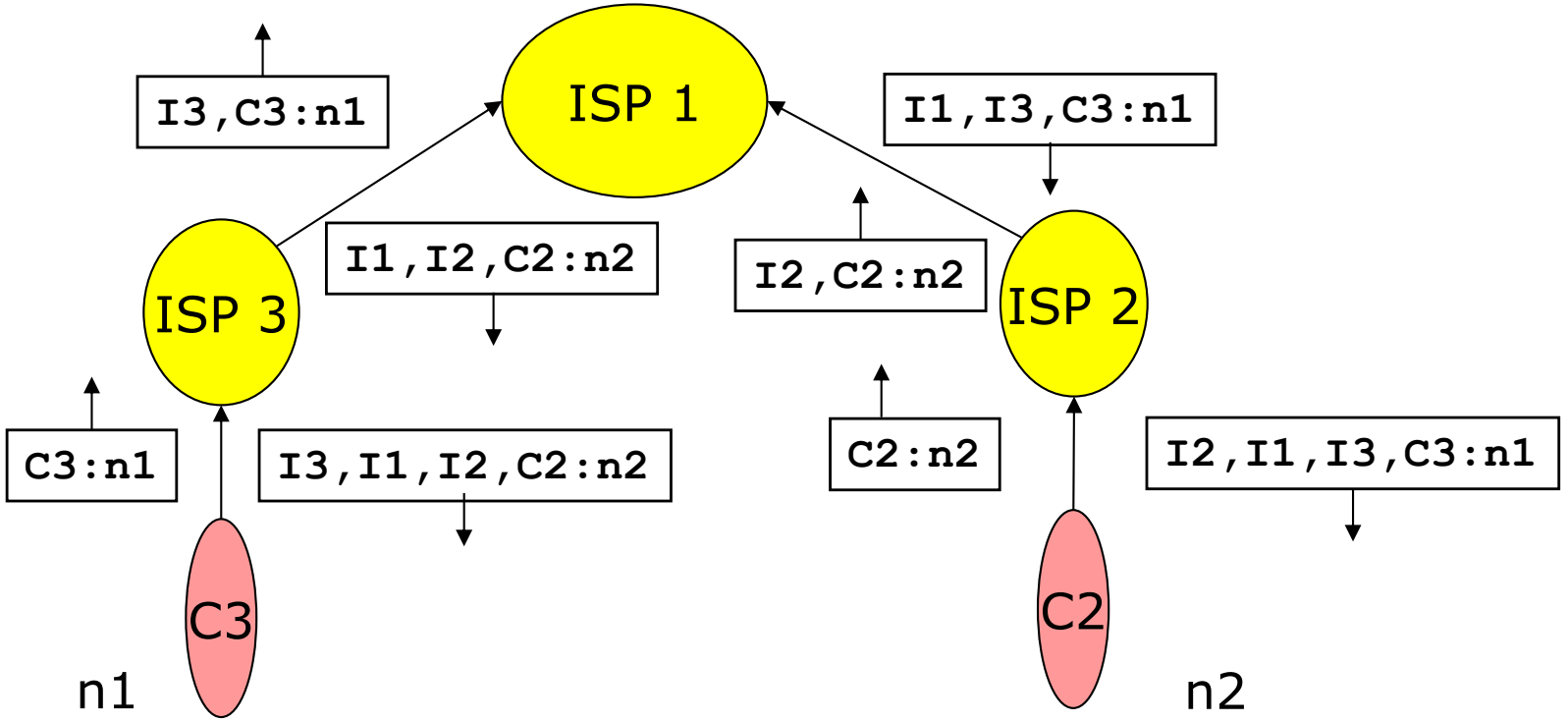
Import Routes



Export Routes



Customer-provider

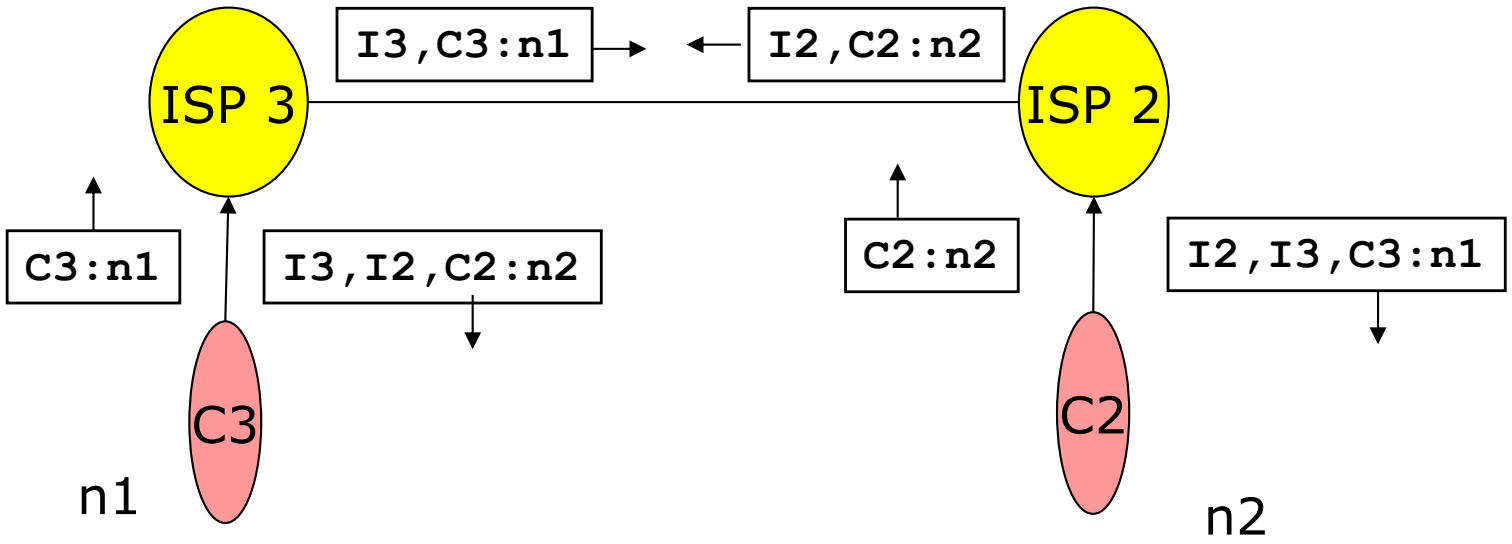


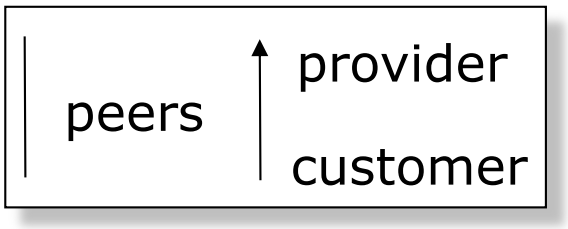
peers

provider

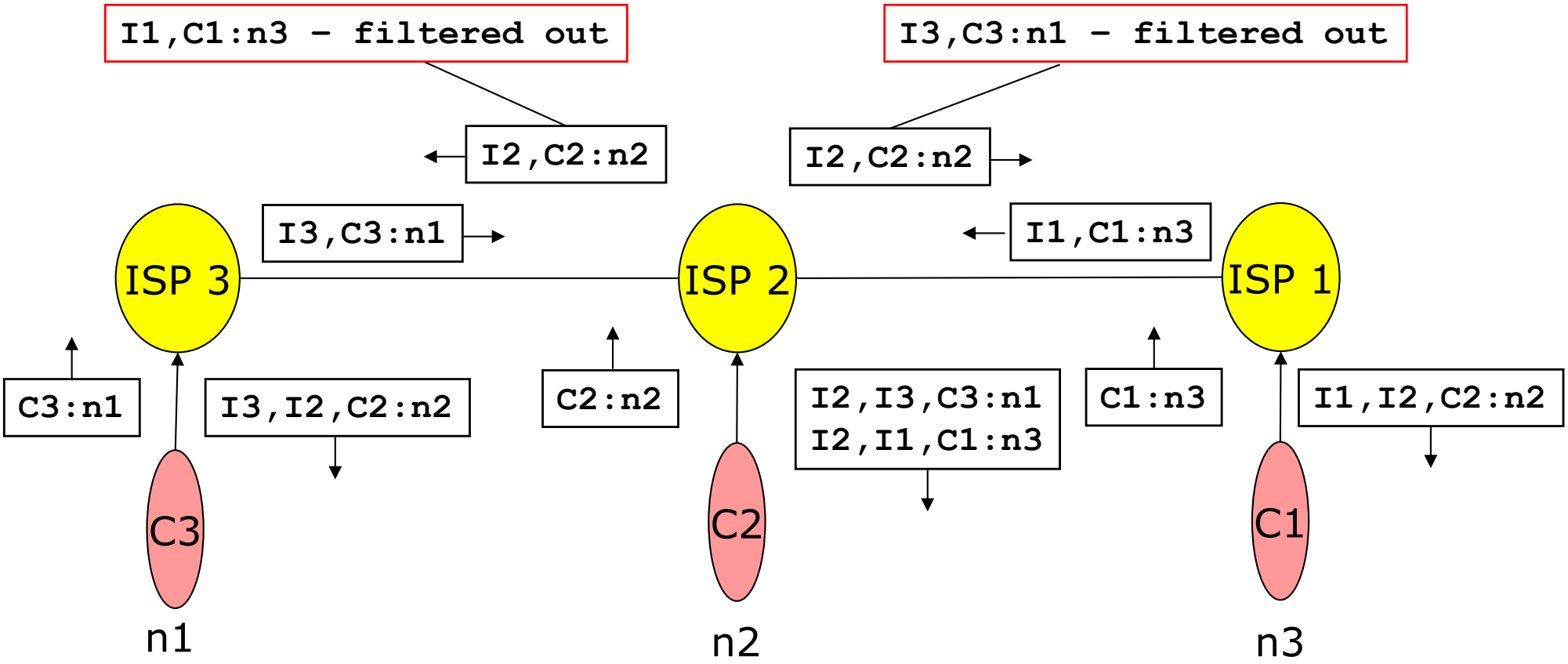
customer

Peers


I3,C3:n1
I2,C2:n2
ISP 3
ISP 2
C3:n1
I3,I2,C2:n2
C2:n2
I2,I3,C3:n1
C3
C2
n1
n2


peers
provider
customer

Peers

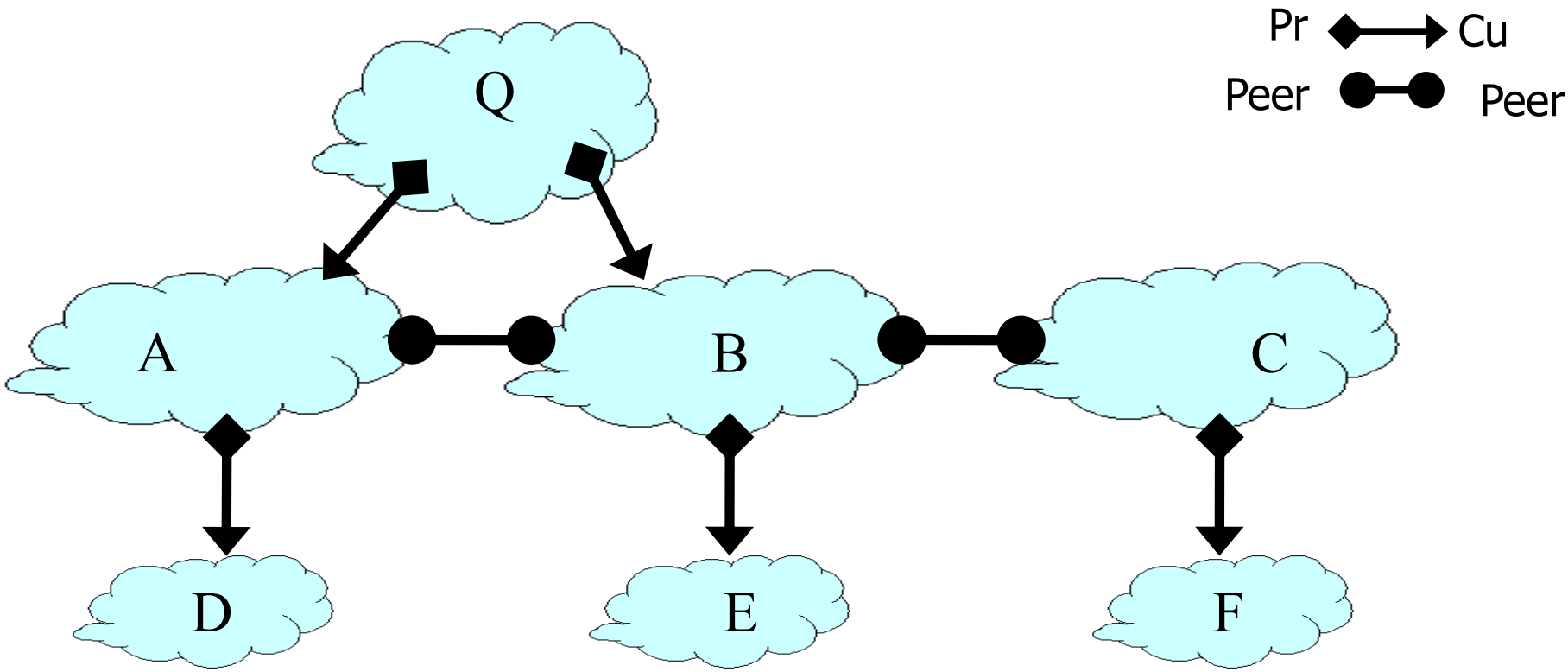


peers

provider

customer

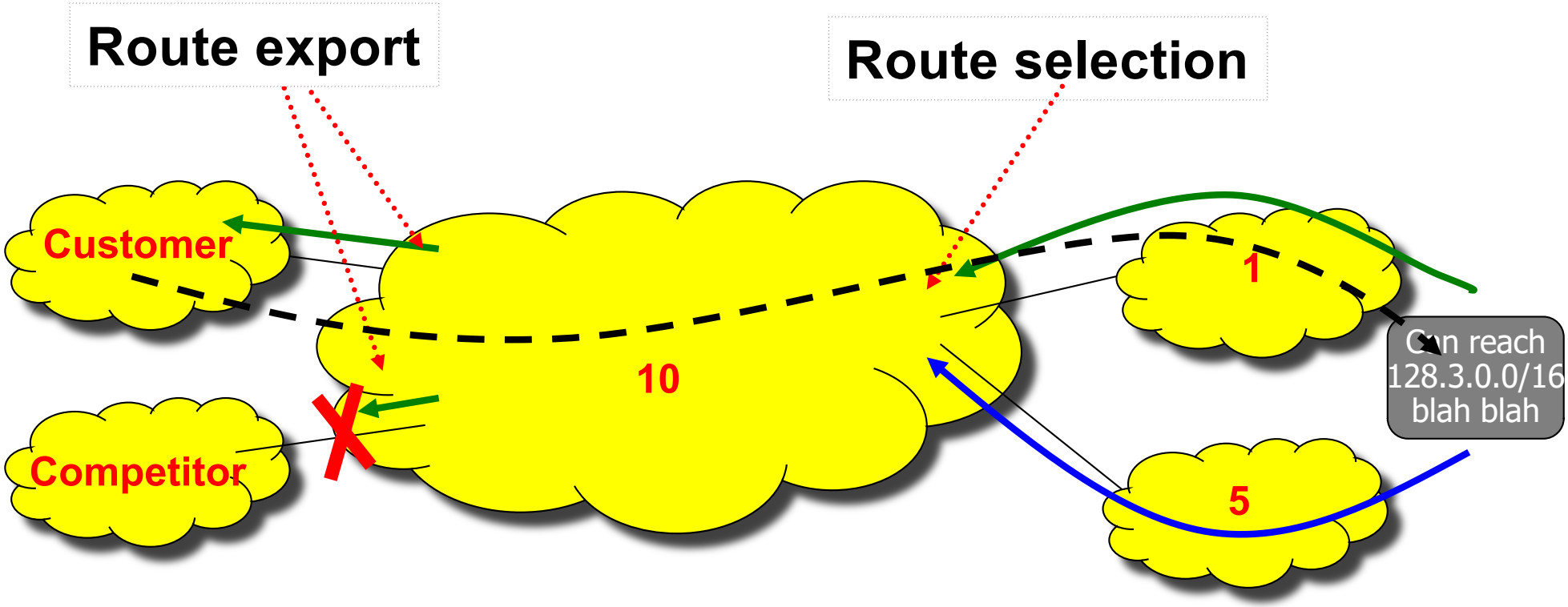
Routing Follows the Money!



- Can D reach F?

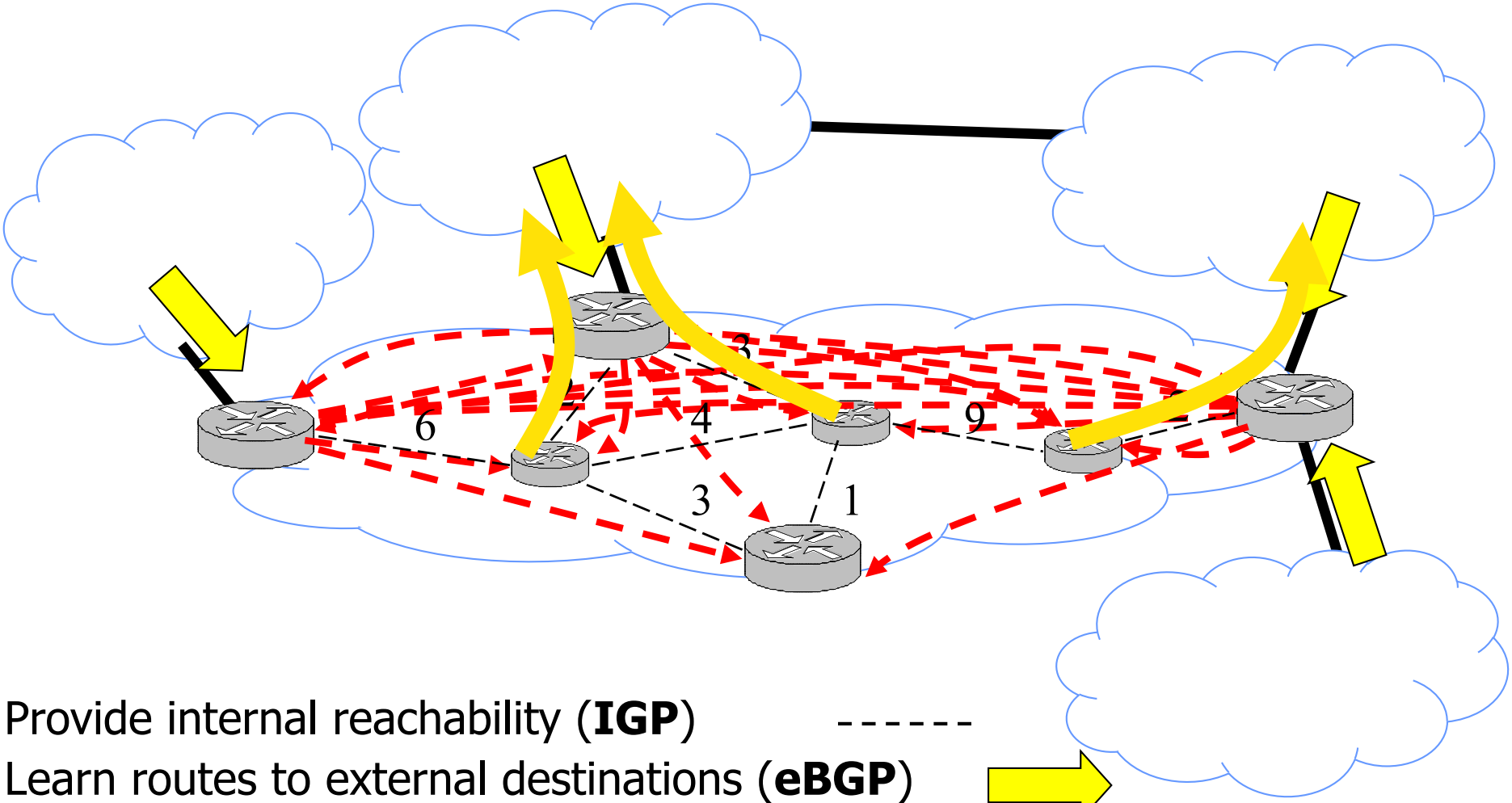
Quiz

Policy imposed in how routes are selected and exported



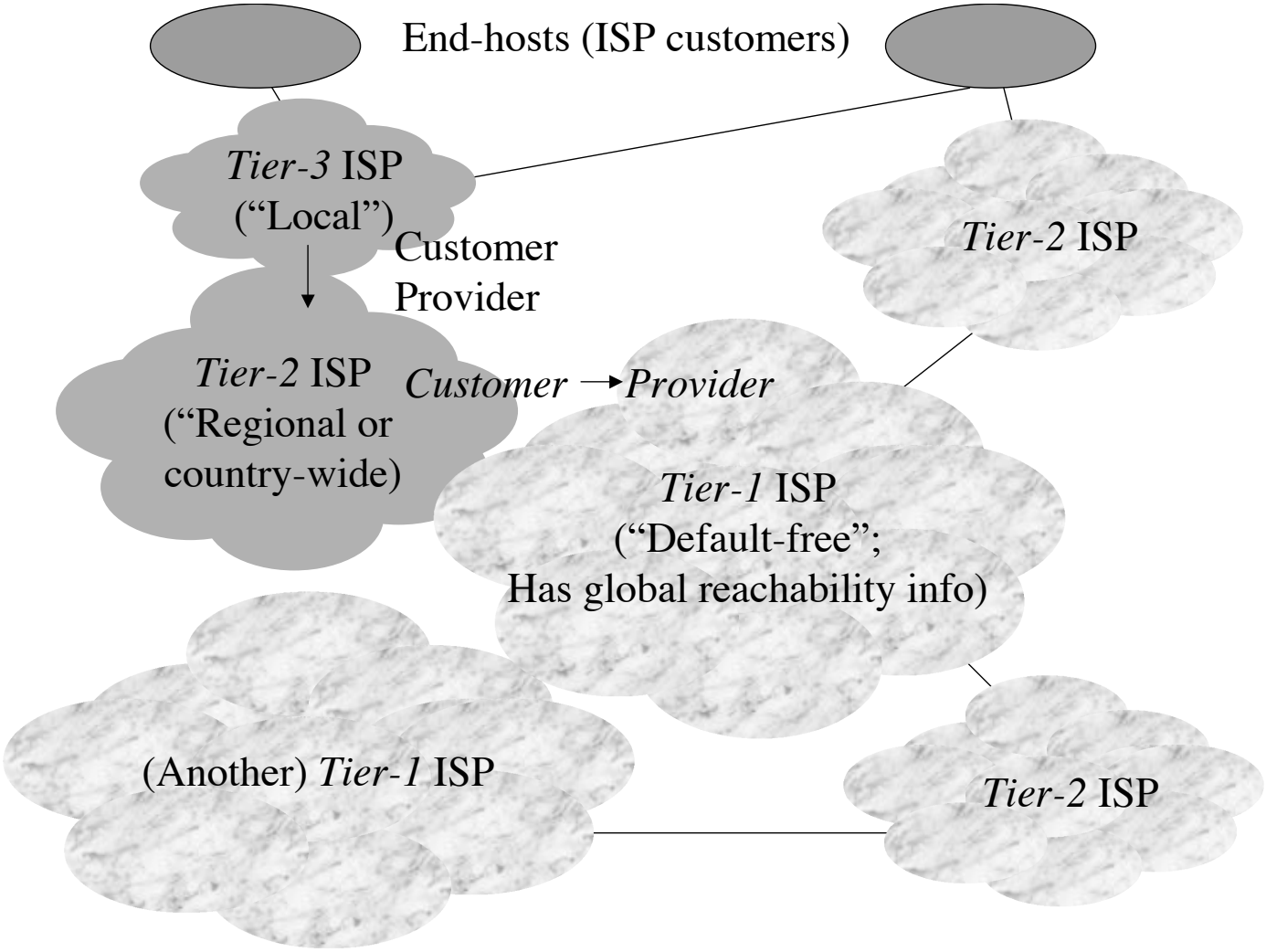
- **Selection**: Which path to use?
  - controls how traffic leaves the network
- **Export**: Which path to advertise?
  - controls whether traffic enters the network

Putting the pieces together



1. Provide internal reachability (**IGP**) ------
2. Learn routes to external destinations (**eBGP**) ➡
3. Distribute externally learned routes internally (**iBGP**) - - ▸
4. Travel shortest path to egress (IGP)

AS hierarchy

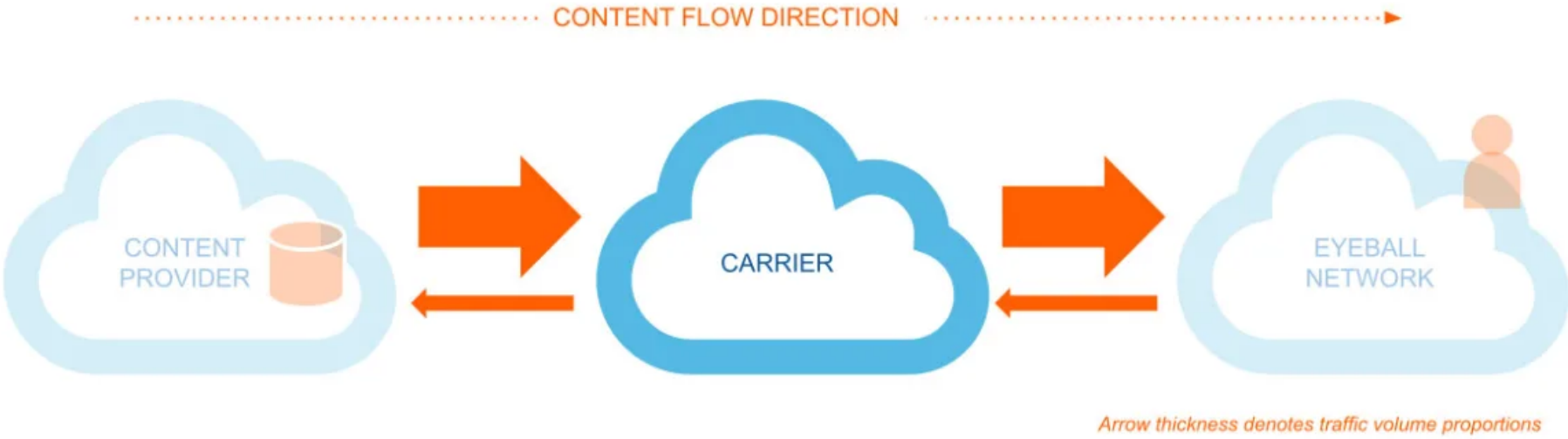


- Tier 1: don't buy service from other ISPs

Tier 1 and Typical Policies

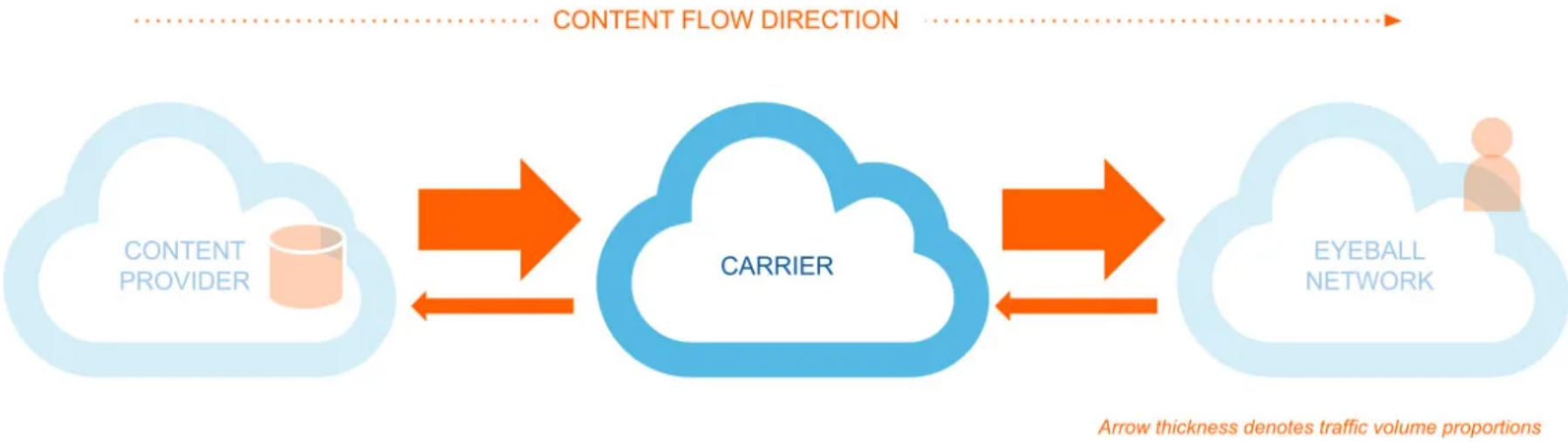
- All Tier 1 providers peer with each other
- They form a fully-connected core of the Internet
- Policy-based routing only works because:
  - Policies follow basic money rule
  - Tier 1 is a full mesh

Evolving roles of ASes



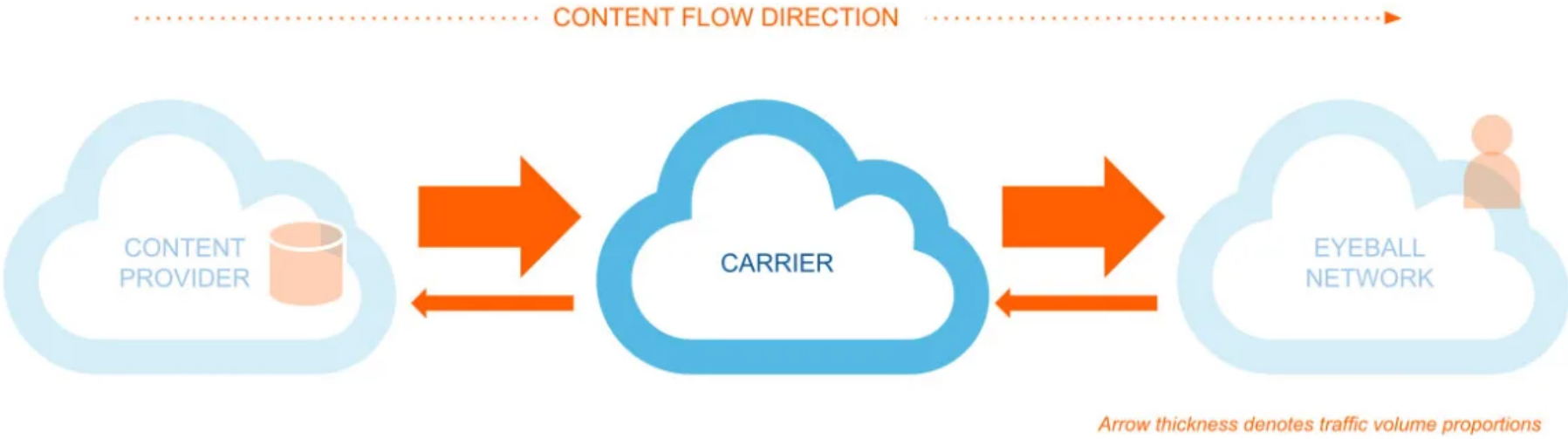
- **Eyeball networks**: residential or enterprise ISPs, users who consume content traffic
- **Content networks**: originate the majority of traffic, destined to be consumed by end-users
- **Carrier networks:** transport packets from content networks to eyeball networks, "the backbone of the Internet"

Eyeball networks



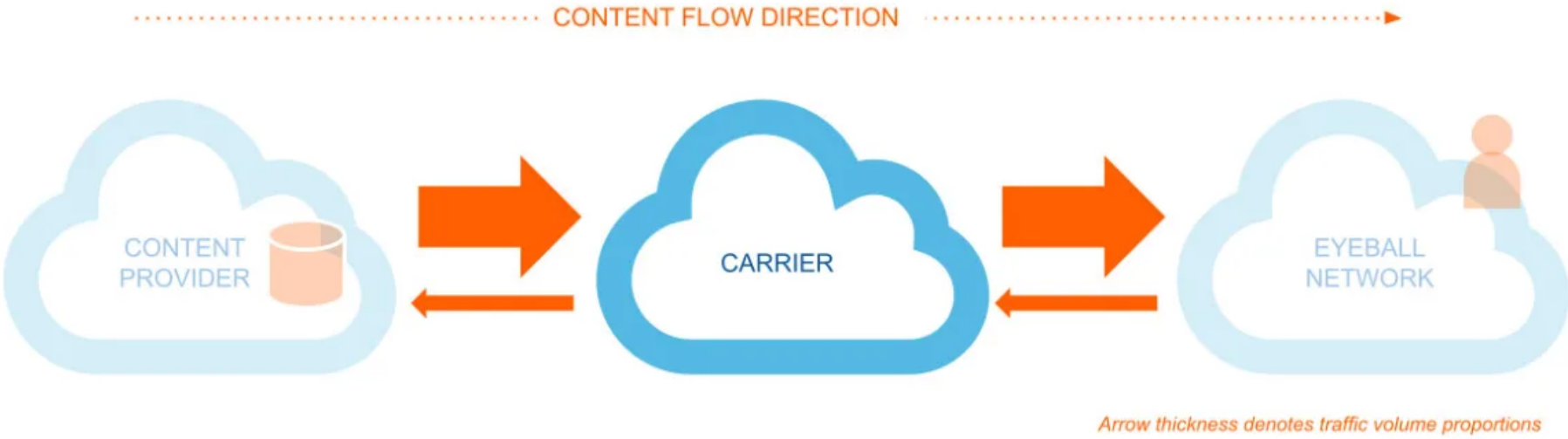
- Traffic profile mostly INBOUND
- Footprint is mostly local (country specific in most cases)
- Challenge: influence inbound traffic routing
- **Download** performance matters, i.e., the speed at which users obtain their favorite content

Content networks



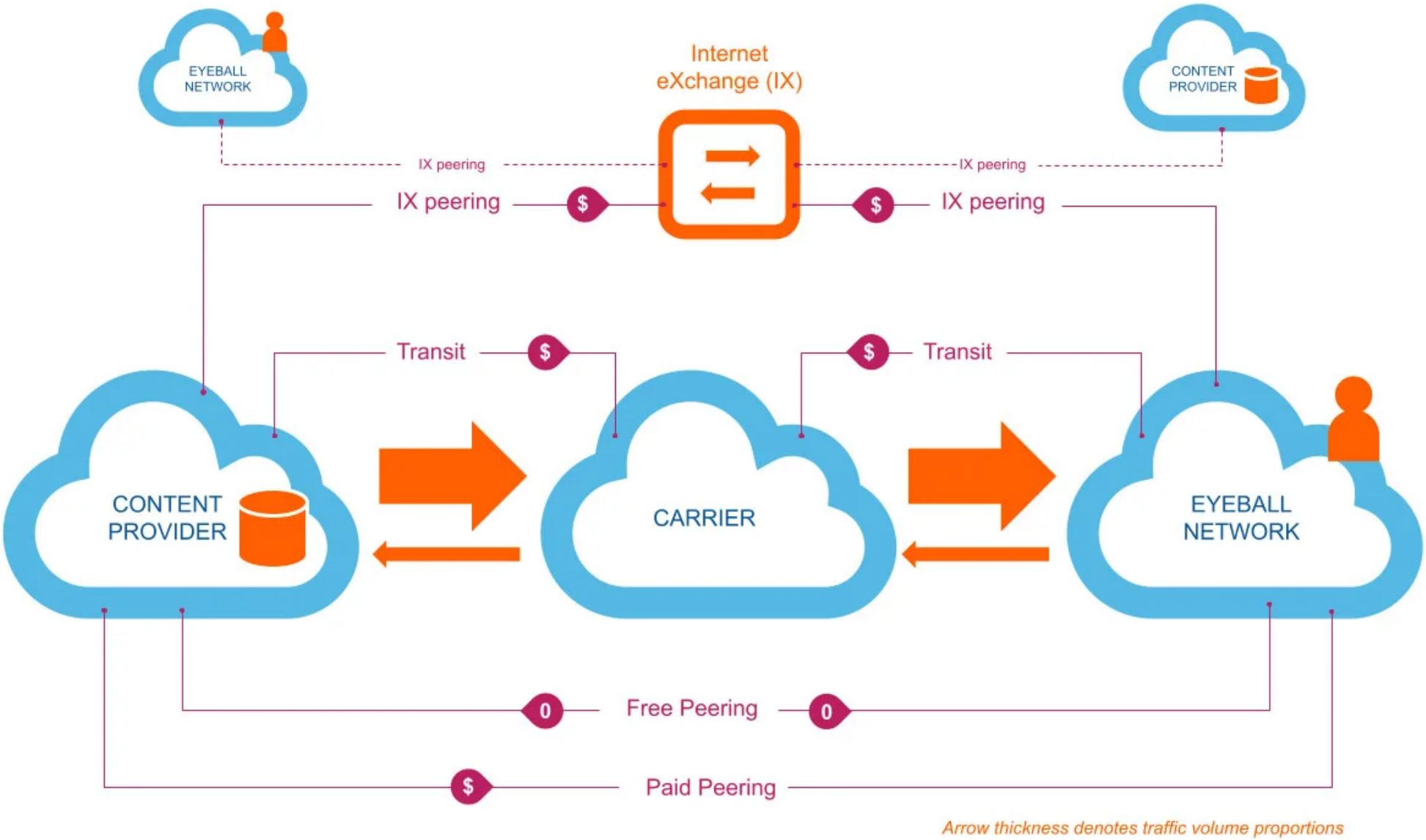
- Traffic profile is mostly OUTBOUND
- Gaming companies, social networks, hosting providers, content delivery networks (CDNs)
- BGP makes it trivial to select exit points for traffic
- Main objective: high performance towards consumers, even while the destination consumer may be multiple networks away

Carrier networks



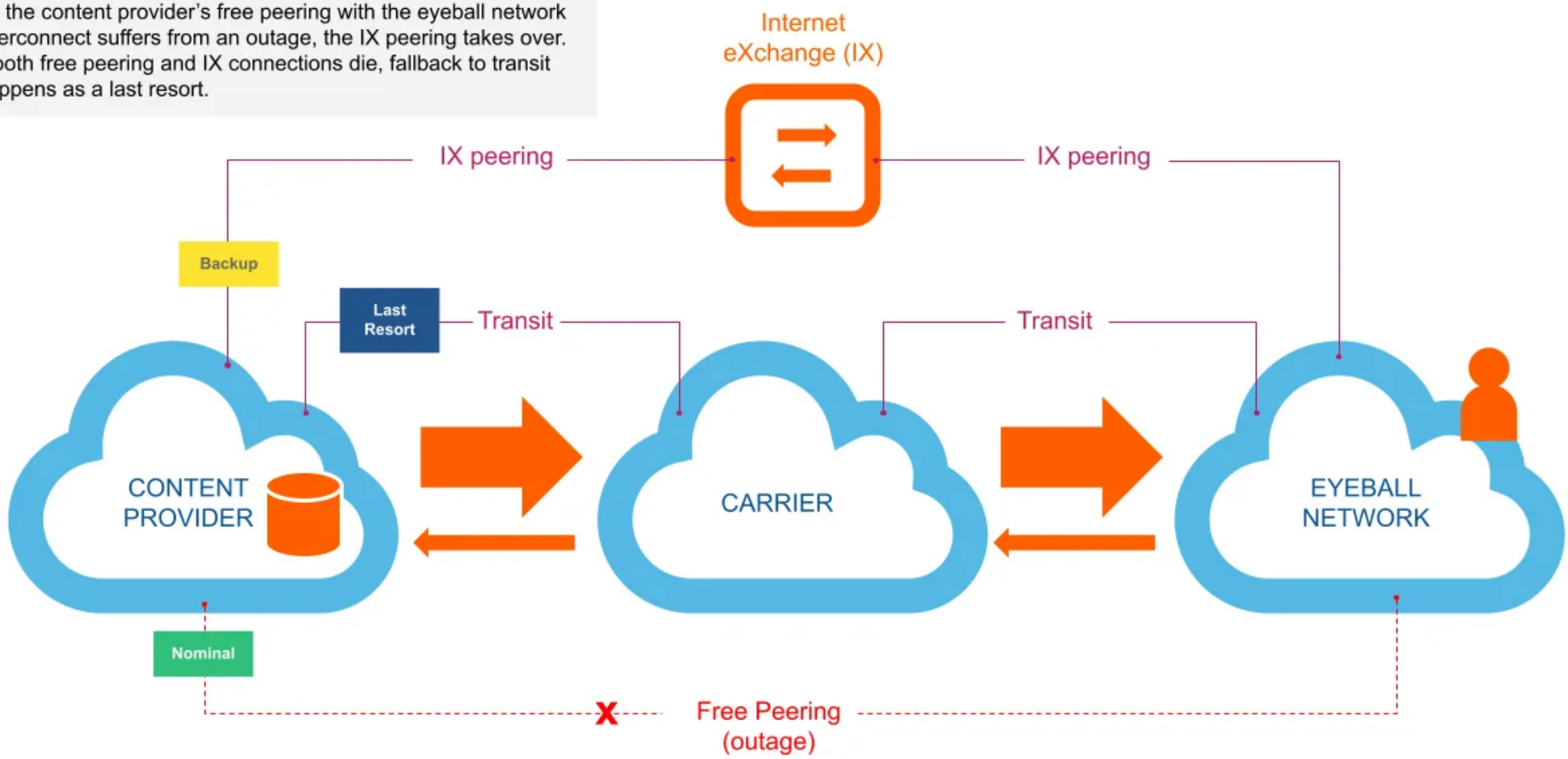
- Carriers make money by transporting content traffic to eyeball users
- Challenge: have enough eyeballs connected to attract content providers and enough content providers to attract eyeballs
- Global footprint and connect content to eyeballs over long distances (Tier-1 - carrier without any paid interconnect)

Interconnection and money-flows

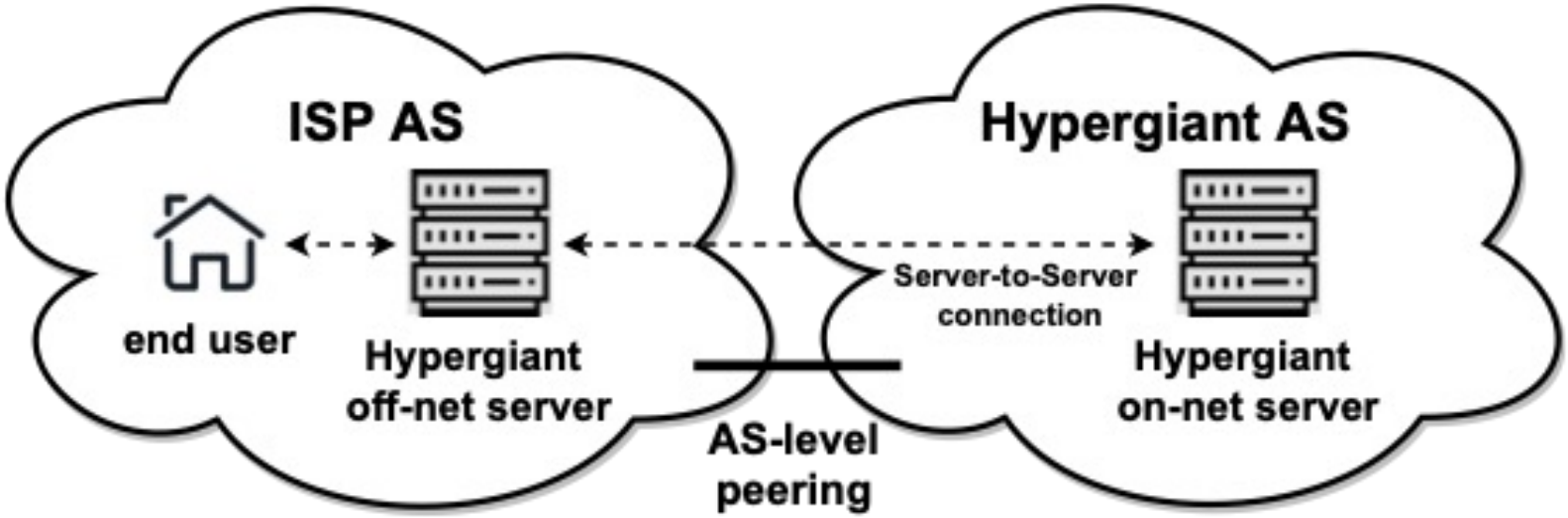


Reliability

As the content provider's free peering with the eyeball network interconnect suffers from an outage, the IX peering takes over. If both free peering and IX connections die, fallback to transit happens as a last resort.

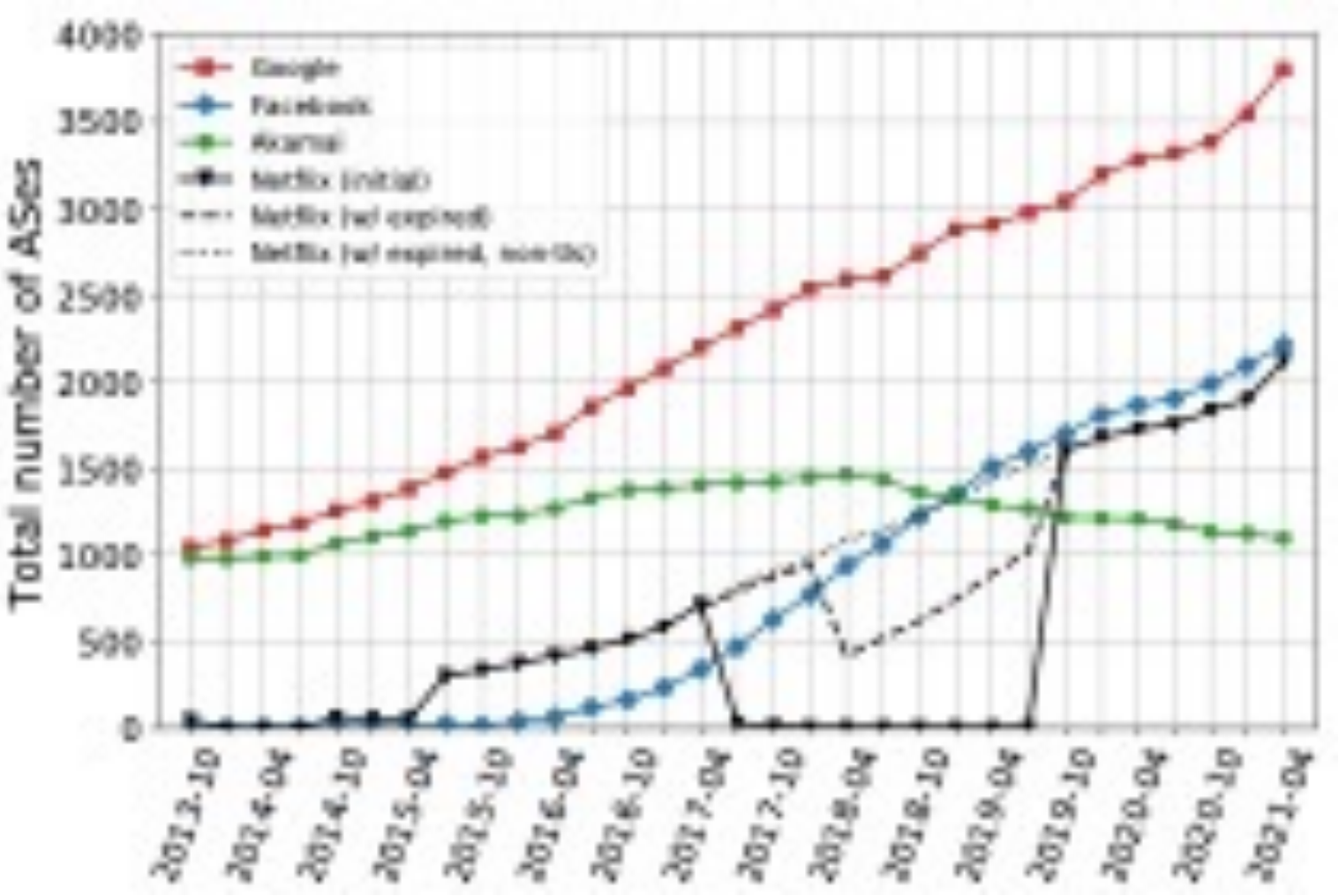


Content Hypergiants



- Hypergiants: Google, Netflix, Facebook, and Akamai
  - deliver content to billions of users around the world
  - half of Internet traffic originated from only 5 HGs in 2019
- Peer directly with eyeball networks, install servers inside eyeball or other networks
  - off-nets - outside (off) the HG's own network,
  - on-nets - HG hosts on its own network

Hypergiant Growth

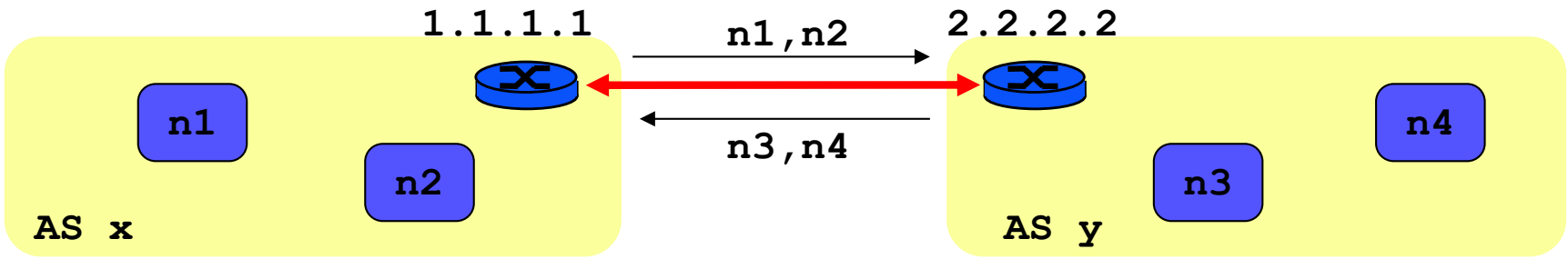


- Off-net footprint growth for top-4 HGs over time

BGP (Border Gateway Protocol)

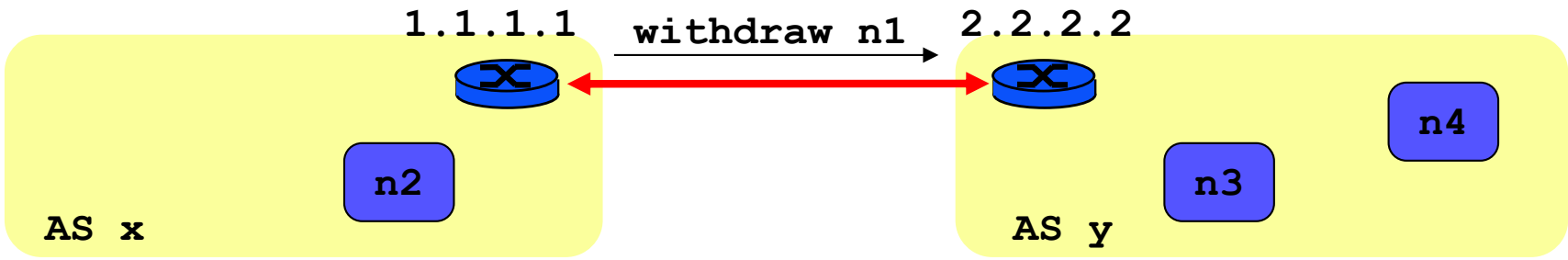
- BGP-4, RFC 1771
- AS border router - BGP speaker
  - peer-to peer relation with another AS border router
  - connected communication
    - on top of a TCP connection, port 179 (vs. datagram (RIP, OSPF))
  - external connections (E-BGP)
    - with border routers of different AS
  - internal connections (I-BGP)
    - with border routers of the same AS
  - BGP only transmits modifications (UPDATE)

BGP principles



- Establish BGP session
- Update
  - list of destinations reachable via each router
  - path attributes such as degree of preference for a particular route
- **BGP Announcement = prefix + attribute values**

BGP principles

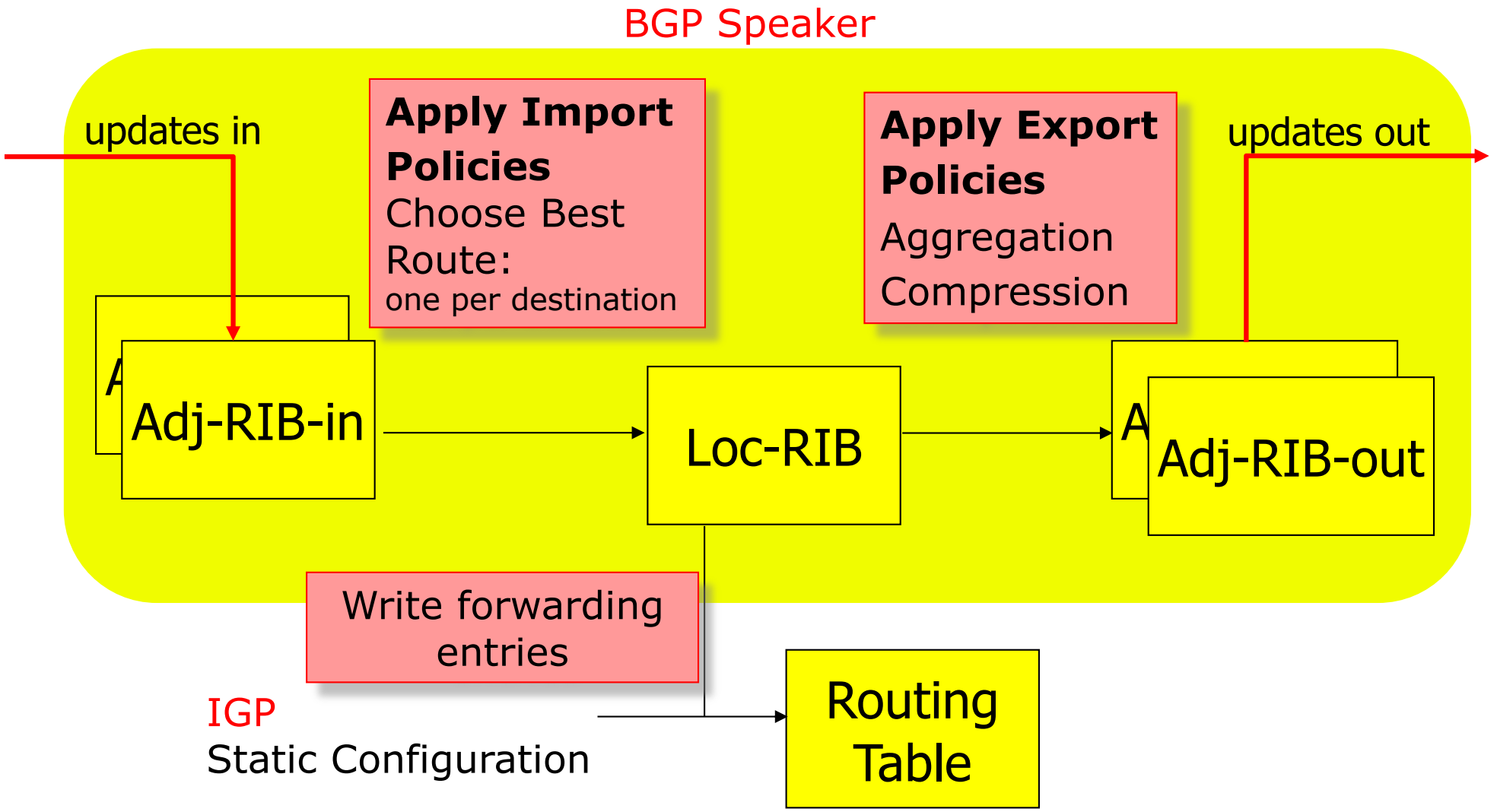


- **n1** no longer reachable
- Incremental update
  - withdraw **n1**

Operation of a BGP speaker

- Receives and stores candidate routes from its BGP peers and from itself
- Applies the decision process to select at most one route per destination prefix
- Exports the selected routes to BGP neighbors, after applying export policy rules and possibly aggregation.
- Stores result in Adj-RIB-out (one per BGP peer) and sends updates when Adj-RIB-out changes (addition or deletion).
- Only routes learnt from E-BGP are sent to an I-BGP neighbor.

Inside BGP

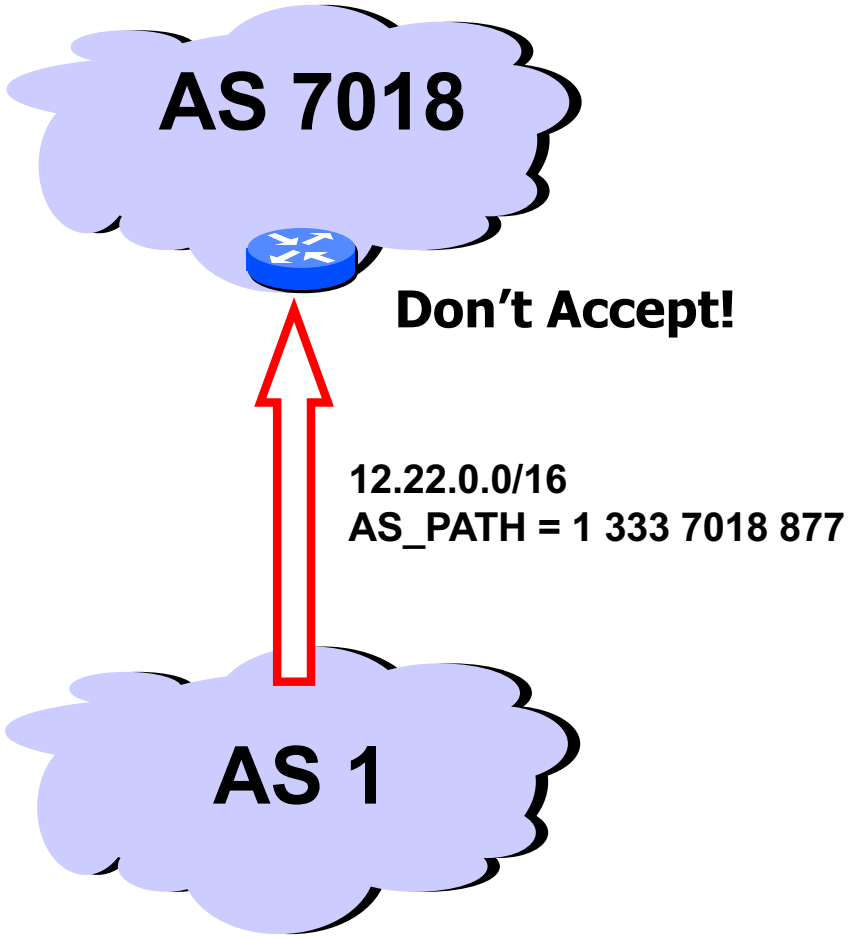

BGP Speaker
updates in
Apply Import Policies
Choose Best Route:
one per destination
Apply Export Policies
Aggregation
Compression
updates out
Adj-RIB-in
Loc-RIB
Adj-RIB-out
Write forwarding entries
IGP
Static Configuration
Routing Table

BGP announcement

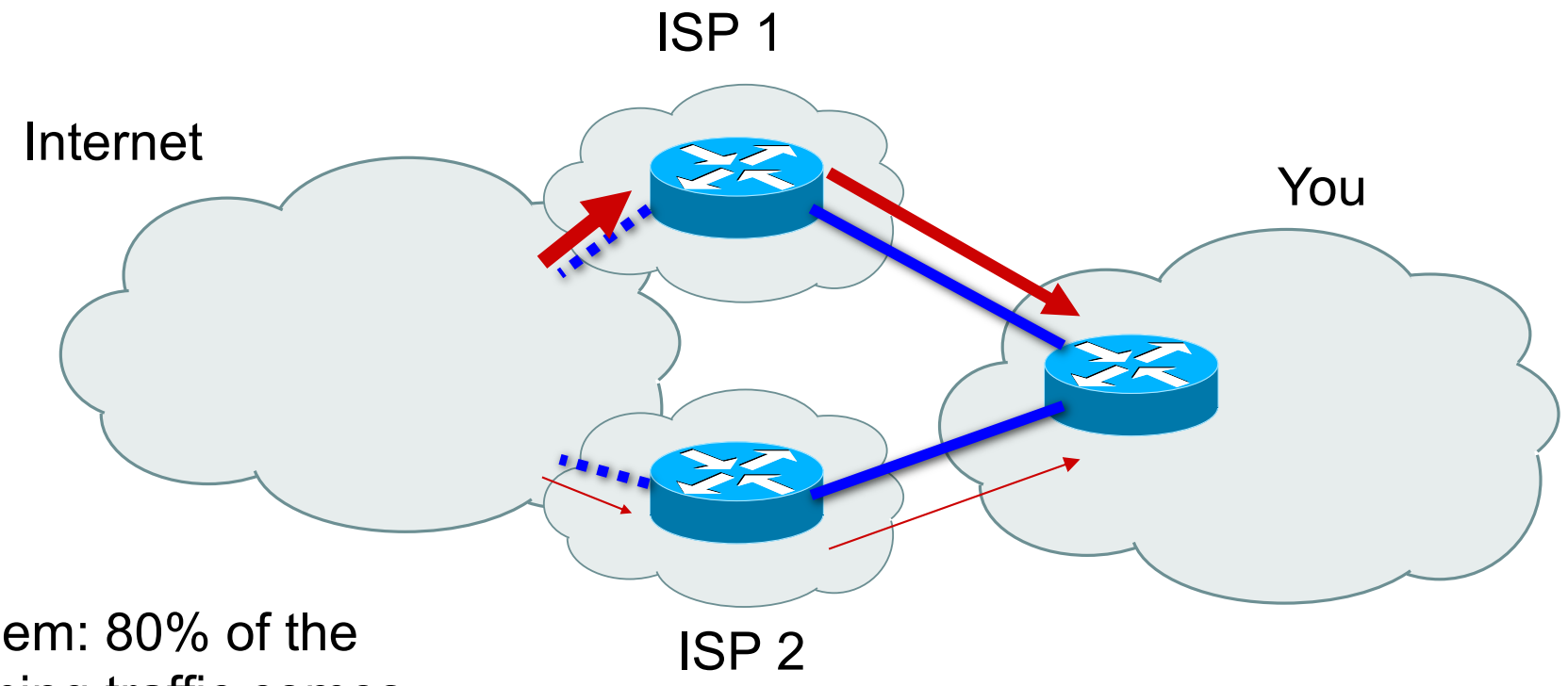
- **Route** - NLRI - Network Layer Reachability Information, contains:
- destination (subnetwork prefix)
- attributes
  - Well-known Mandatory
    - **AS_PATH**
    - **NEXT_HOP**
    - **ORIGIN** (route learnt from IGP, BGP or static)
  - Well-known Discretionary
    - **LOCAL_PREF**
    - ATOMIC_AGGREGATE (= route cannot be dis-aggregated)
  - Optional Transitive
    - **MULTI_EXIT_DISC** (MED) (see later)
    - AGGREGATOR (who aggregated this route)
  - Optional Nontransitive
    - **WEIGHT**

AS_PATH - Loop Prevention

- AS-PATH contains the list of AS the update had to traverse.
- AS-PATH is updated by the sending router with its own AS number.
- BGP uses the AS-PATH to detect routing loops:
  - BGP at AS YYY will never accept a route with AS_PATH containing YYY



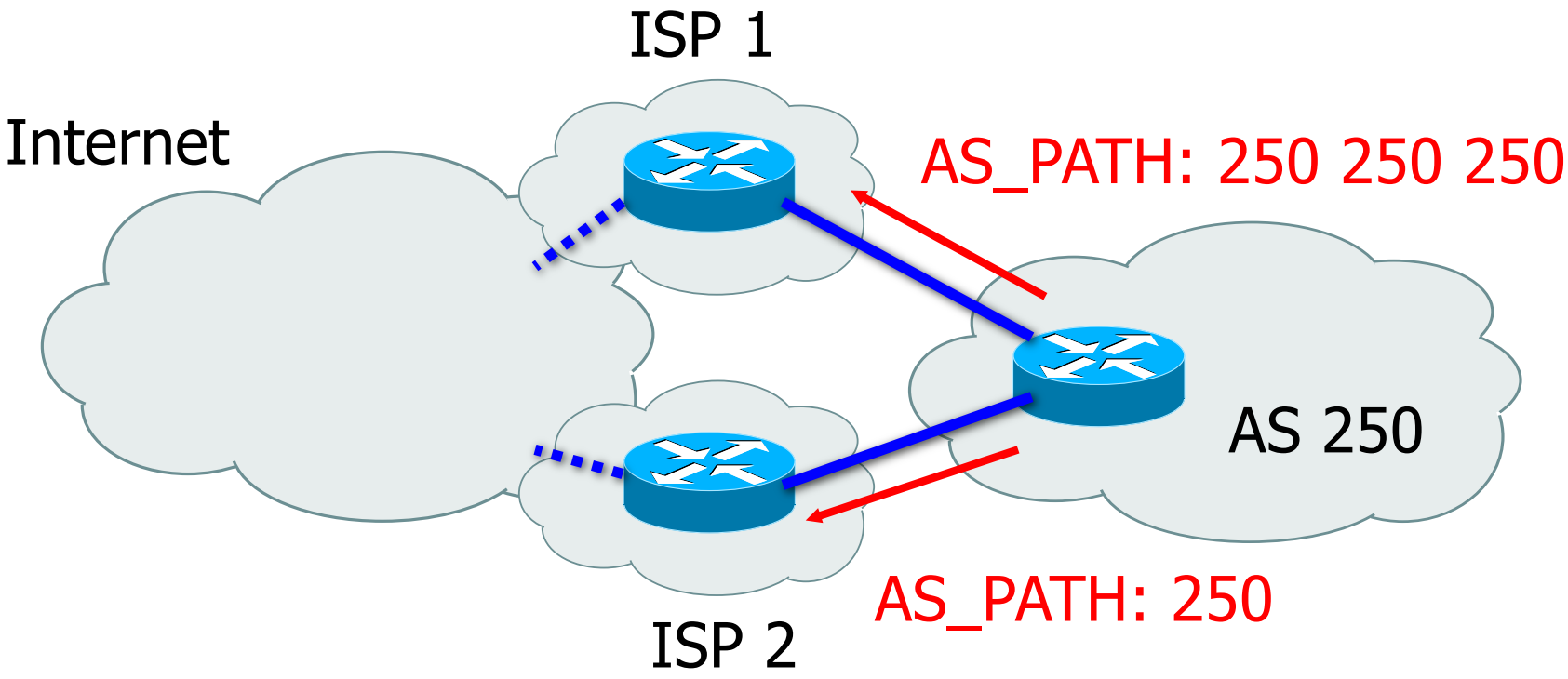
AS_PATH manipulation
AS-PATH prepending



Problem: 80% of the incoming traffic comes from ISP 1

AS_PATH manipulation
AS_PATH prepending

set as-path prepend 250 250



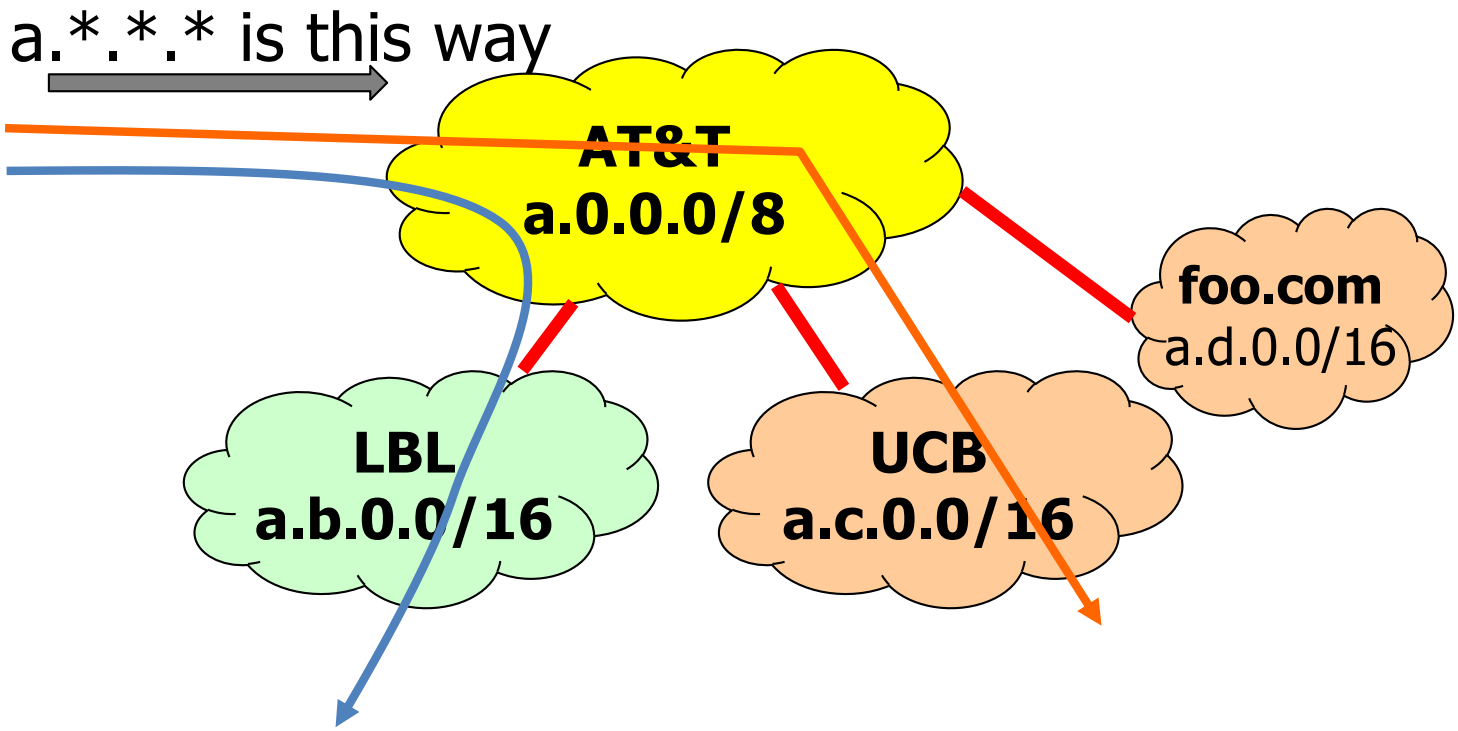
Prefix Aggregation

- AS that does not have a default route (i.e. all transit ISPs) must know all routes in the world (> 200 000 prefixes)
- Aggregation is a way to reduce the number of routes

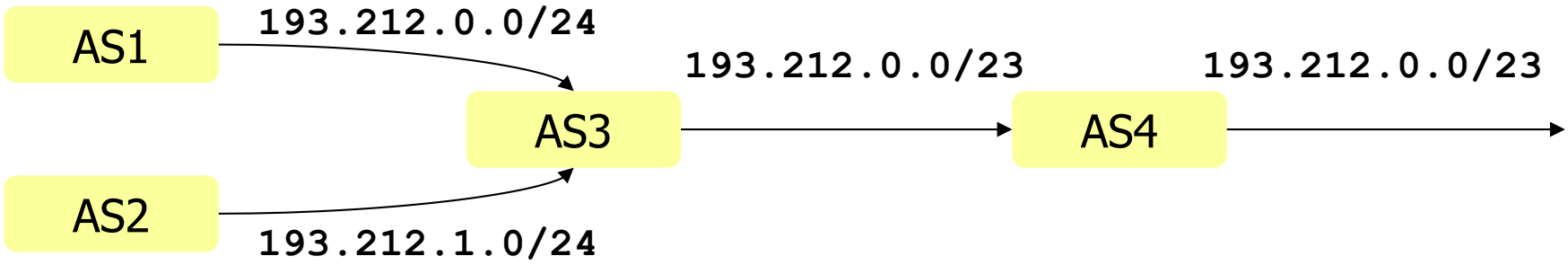
- AGGREGATOR attribute - last AS that formed the aggregate route
- ATOMIC AGGREGATE attribute
  - indicates a more specific route exists
- AS_PATH attribute
  - identifies ASes in reverse order
- AS segments
  - AS_SET - Unordered set of ASes
  - AS_SEQUENCE - Ordered set of ASes

Prefix aggregation

- For scalability, BGP may aggregate routes for different prefixes

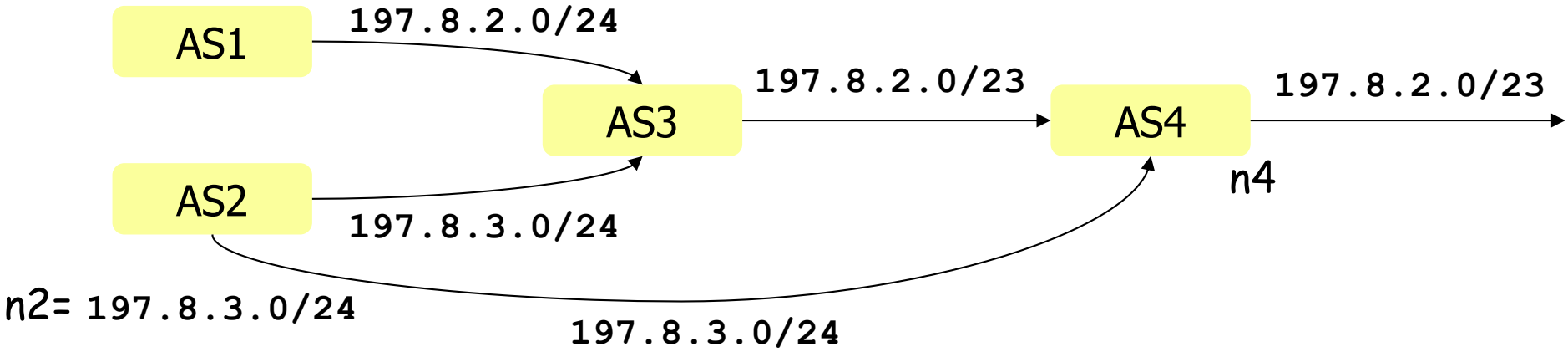


Aggregation Example 1



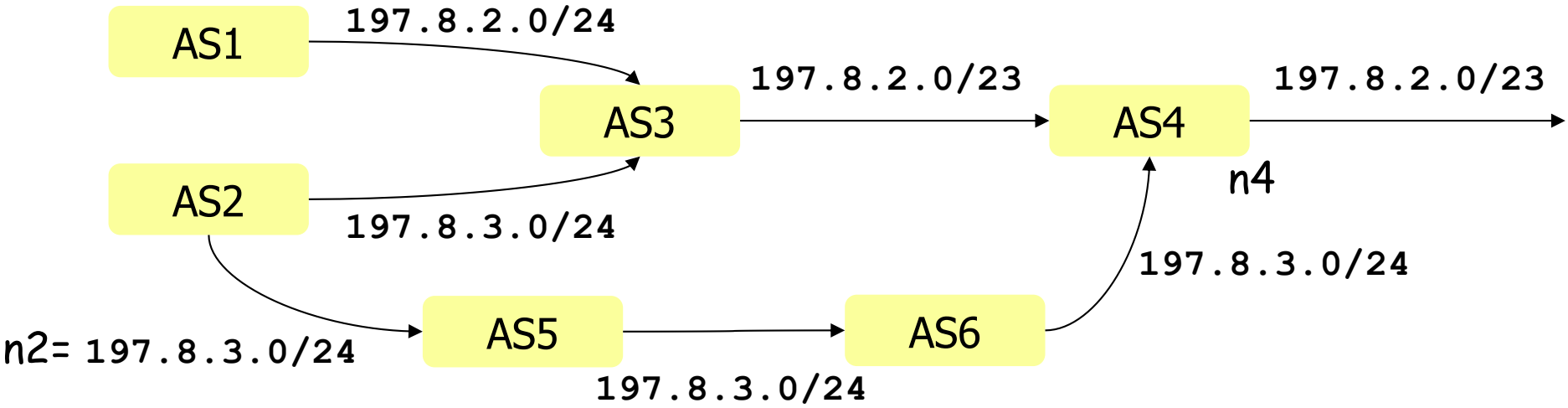
- Assume AS3 aggregates the routes received from AS1 and AS2
  - **AS1: 193.212.0.0/24 AS_PATH: 1**
  - **AS2: 193.212.1.0/24 AS_PATH: 2**
  - **AS3: 193.212.0.0/23 AS_PATH: 3 {1 2}**
  - **AS4: 193.212.0.0/23 AS_PATH: 4 3 {1 2}**

Aggregation Example 2



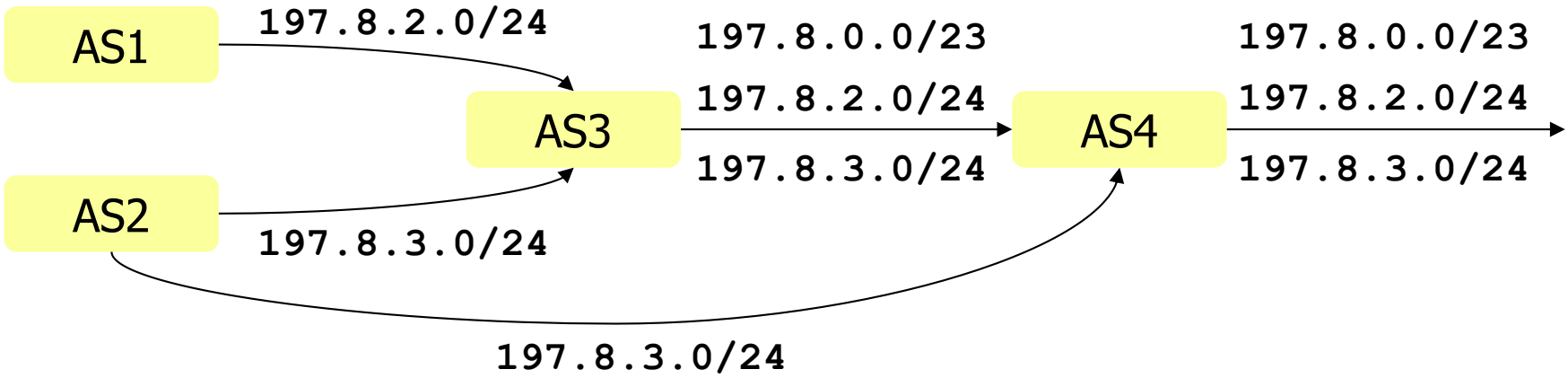
- AS4 receives
  - **197.8.2.0/23** **AS_PATH: 3 {1 2}**
  - **197.8.3.0/24** **AS_PATH: 2**
- What happens to packets from n4 to n2?
  - if AS4 puts two entries: **197.8.2.0/23, 197.8.3.0/24**
  - if AS4 puts one entry: **197.8.2.0/23**

Aggregation Example 3



- AS4 receives
  - **197.8.2.0/23** **AS_PATH: 3 {1 2}**
  - **197.8.3.0/24** **AS_PATH: 6 5 2**
- What happens to packets from n4 to n2?
  - if both routes are used: **197.8.2.0/23, 197.8.3.0/24**
  - if the shortest AS path is used: **197.8.2.0/23**

Example Without Aggregation



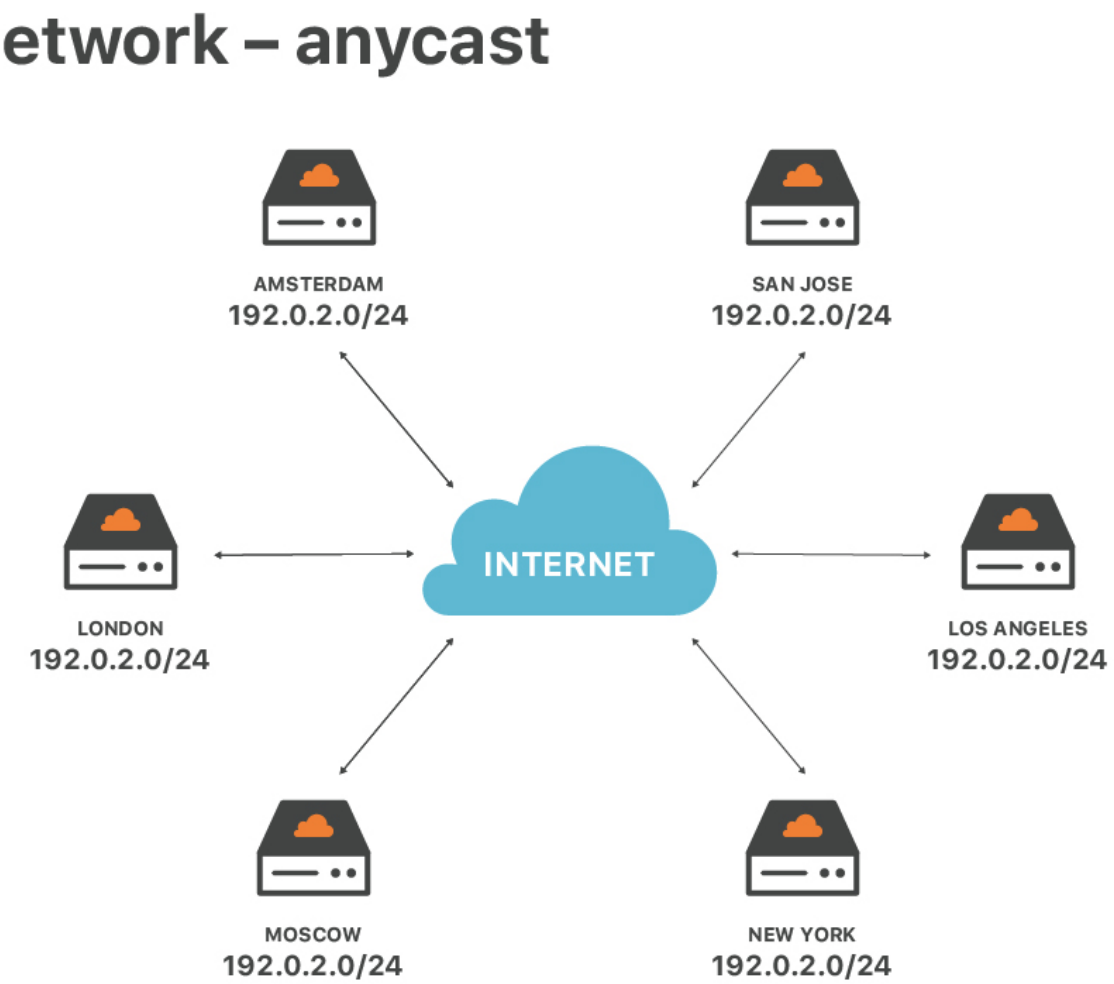
- AS3 has **197.8.0.0/23**
- If AS3 does not aggregate, what are the routes announced by AS 4?
  - **197.8.0.0/23** **AS_PATH: 4 3**
  - **197.8.2.0/24** **AS_PATH: 4 3 1**
  - **197.8.3.0/24** **AS_PATH: 4 2**
- There is no benefit since all routes go via AS 4 anyhow. AS4 should aggregate to **197.8.0.0/22**.

Conclusion on Aggregation

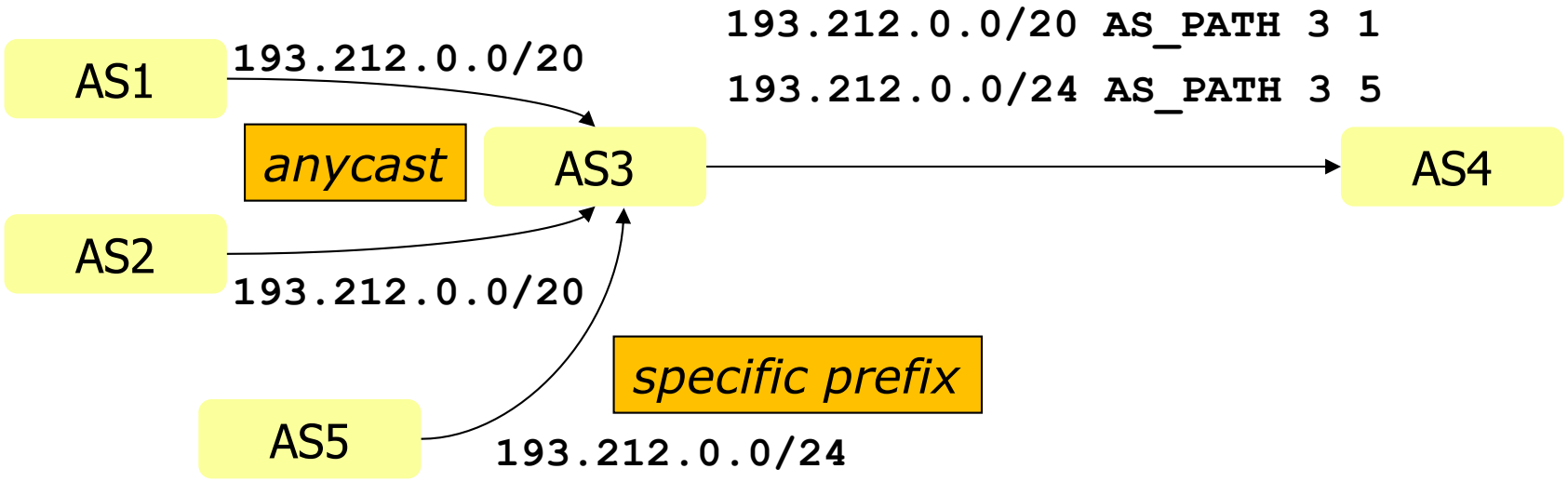
- Aggregation should be performed whenever possible
  - when all aggregated prefixes have the same path (example 1)
  - when all aggregated prefixes have the same path before the aggregation point (examples 2 to 4)

- An AS can decide to
  - Aggregate several routes when exporting them
  - But still maintain different routing entries inside its domain (example 2)

Anycast

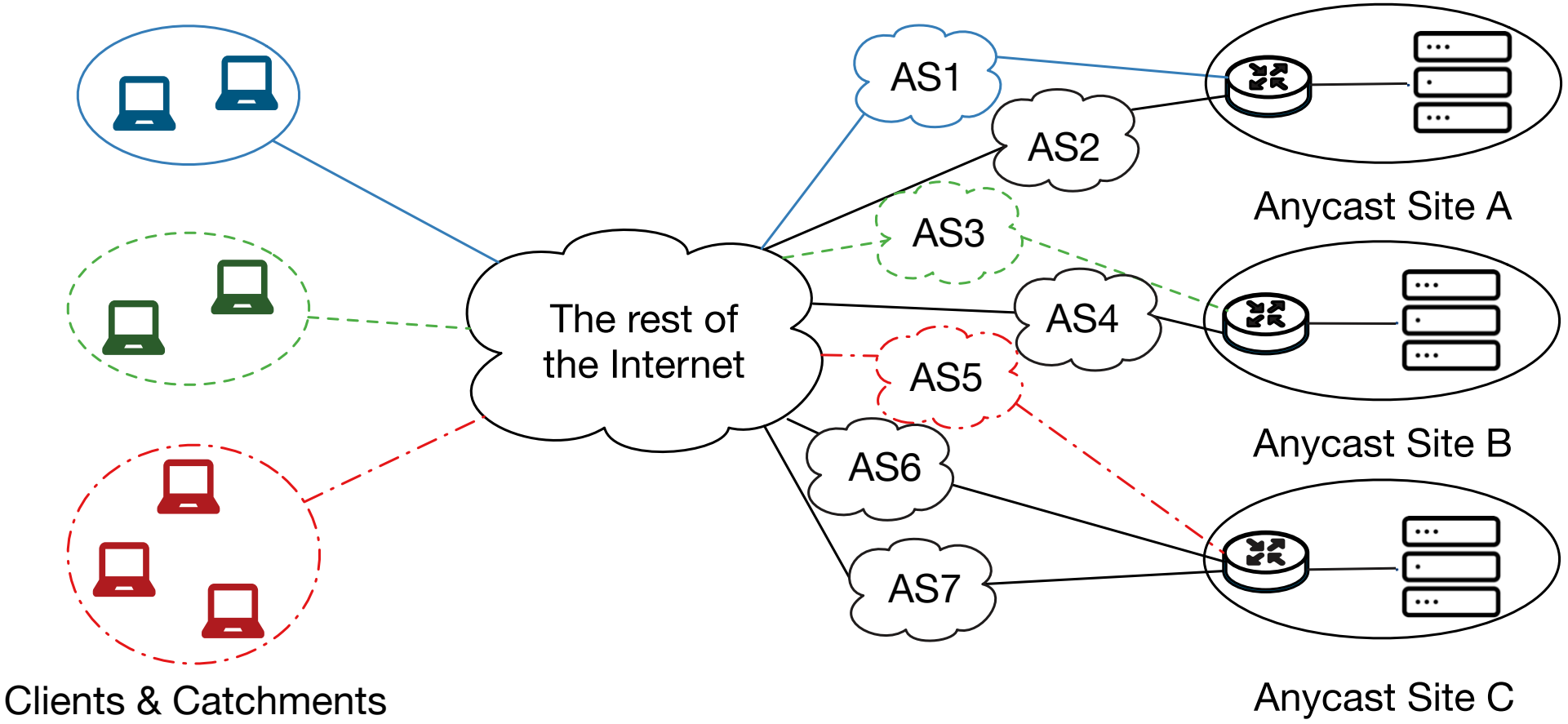

Edge Network – anycast
AMSTERDAM
192.0.2.0/24
SAN JOSE
192.0.2.0/24
INTERNET
LONDON
192.0.2.0/24
LOS ANGELES
192.0.2.0/24
MOSCOW
192.0.2.0/24
NEW YORK
192.0.2.0/24

Anycast, specific prefixes



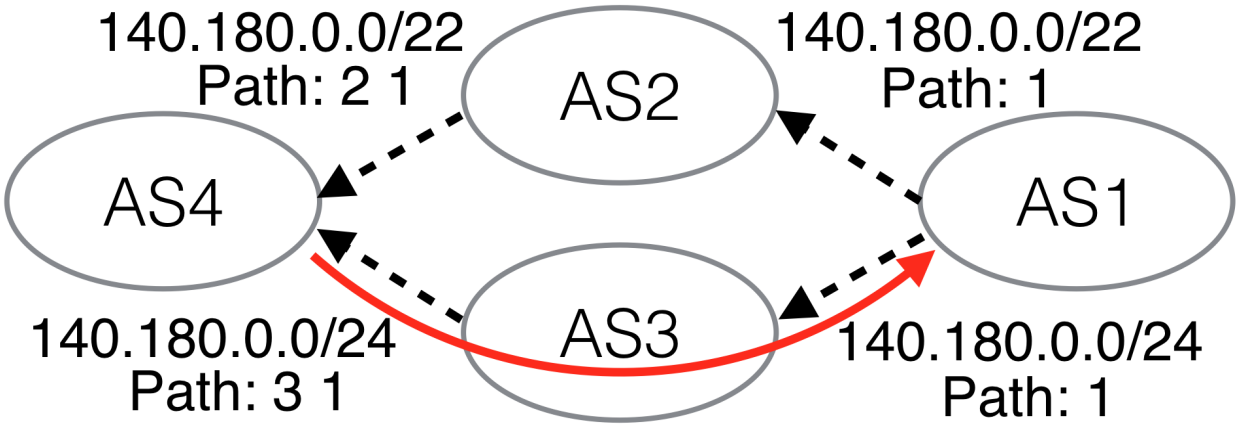
- Both AS1 and AS2 announce **193.212.0.0/20**, AS3 chooses the "closest" one – AS1
- AS5 announces **193.212.0.0/24**, a more specific prefix than **193.212.0.0/20**, AS3 propagates both prefixes

Anycast



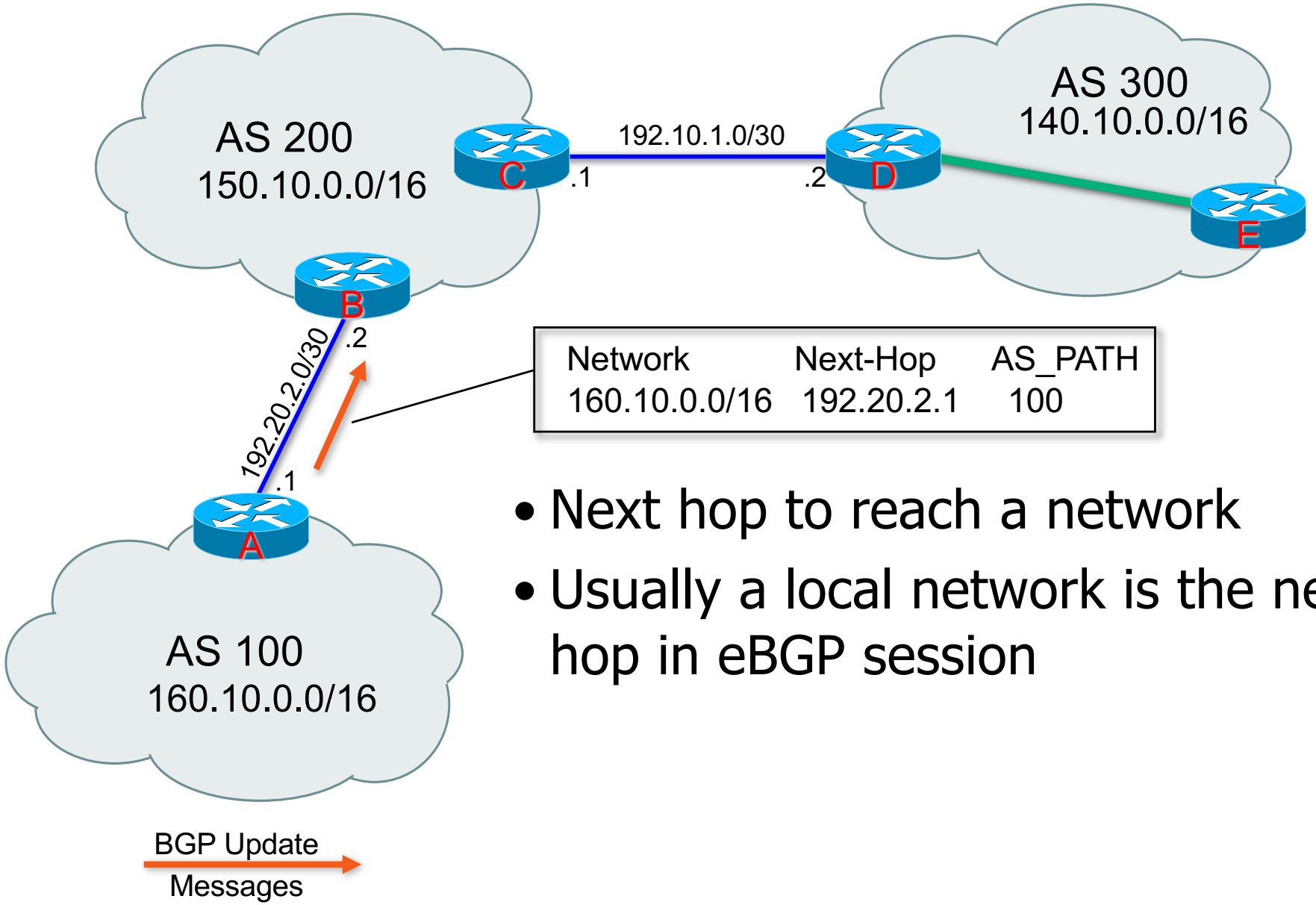
- Site A in London, site B in San Jose, site C in Warsaw

Longest Matching Prefix



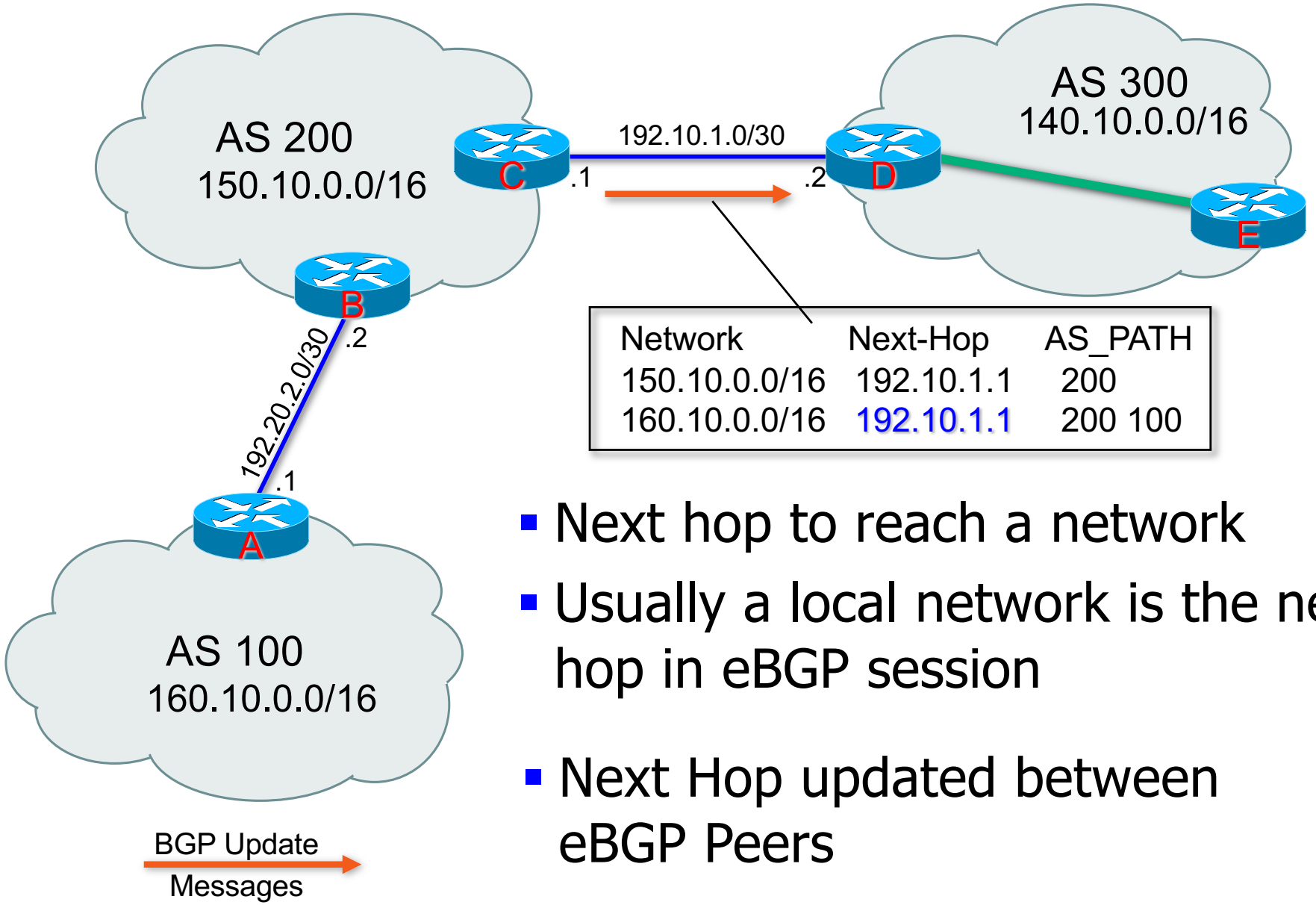
- ASes forward packets according to longest matching prefix
  - AS1 announces 140.180.0.0/22 via neighbor AS2, and 140.180.0.0/24 via neighbor AS3.
  - AS4 forwards packets to 140.180.0.0/24 via AS3
- Longest prefix successfully propagated is /24:
  - prefixes longer than /24 are filtered

Next Hop Attribute



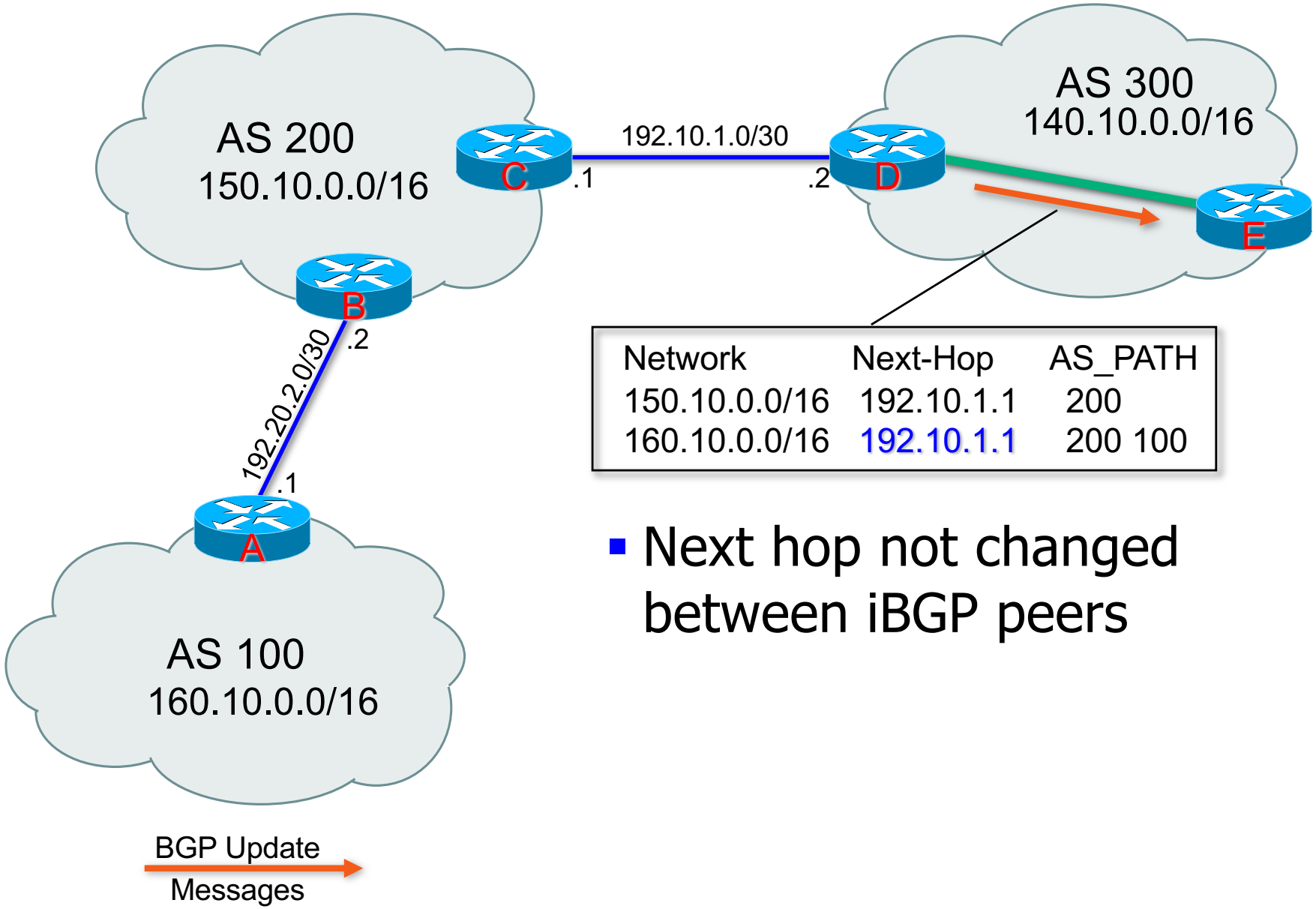
- Next hop to reach a network
- Usually a local network is the next hop in eBGP session

Next Hop Attribute



- Next hop to reach a network
- Usually a local network is the next hop in eBGP session
- Next Hop updated between eBGP Peers

Next Hop Attribute

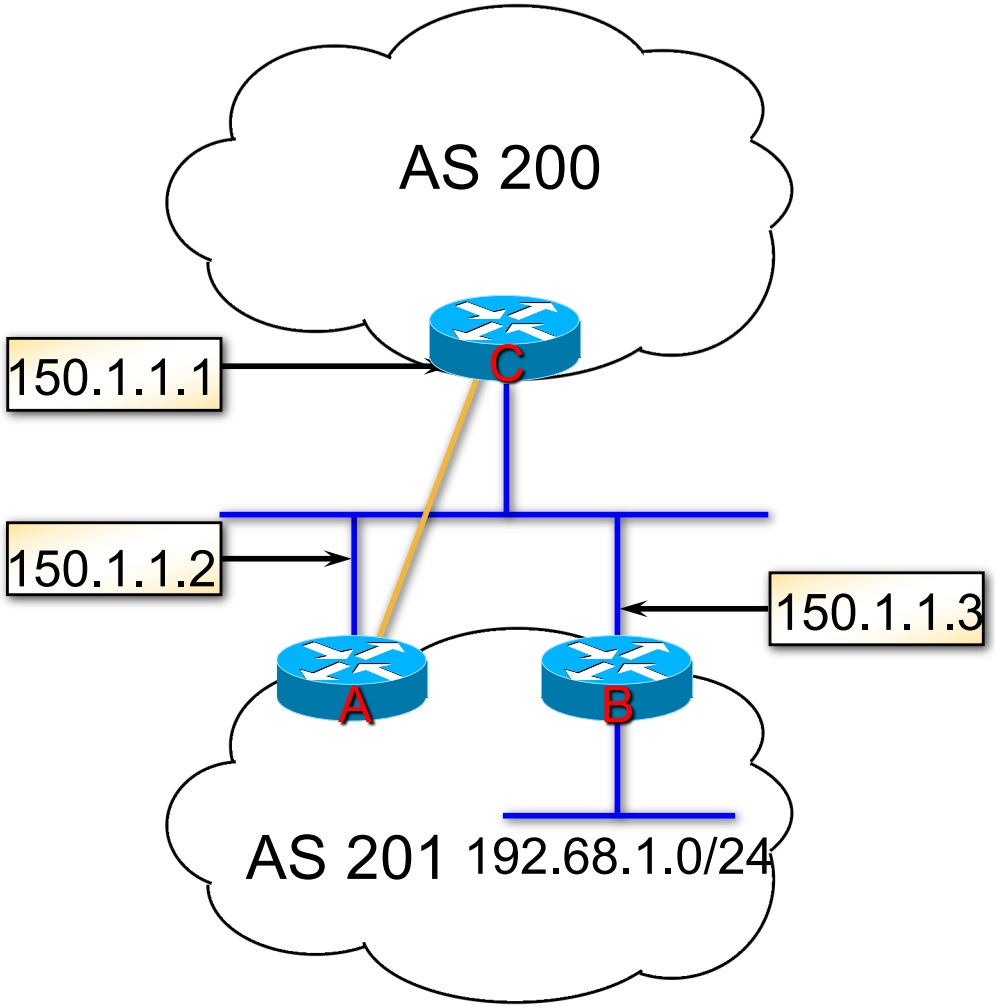


Network	Next-Hop	AS_PATH
150.10.0.0/16	192.10.1.1	200
160.10.0.0/16	192.10.1.1	200 100

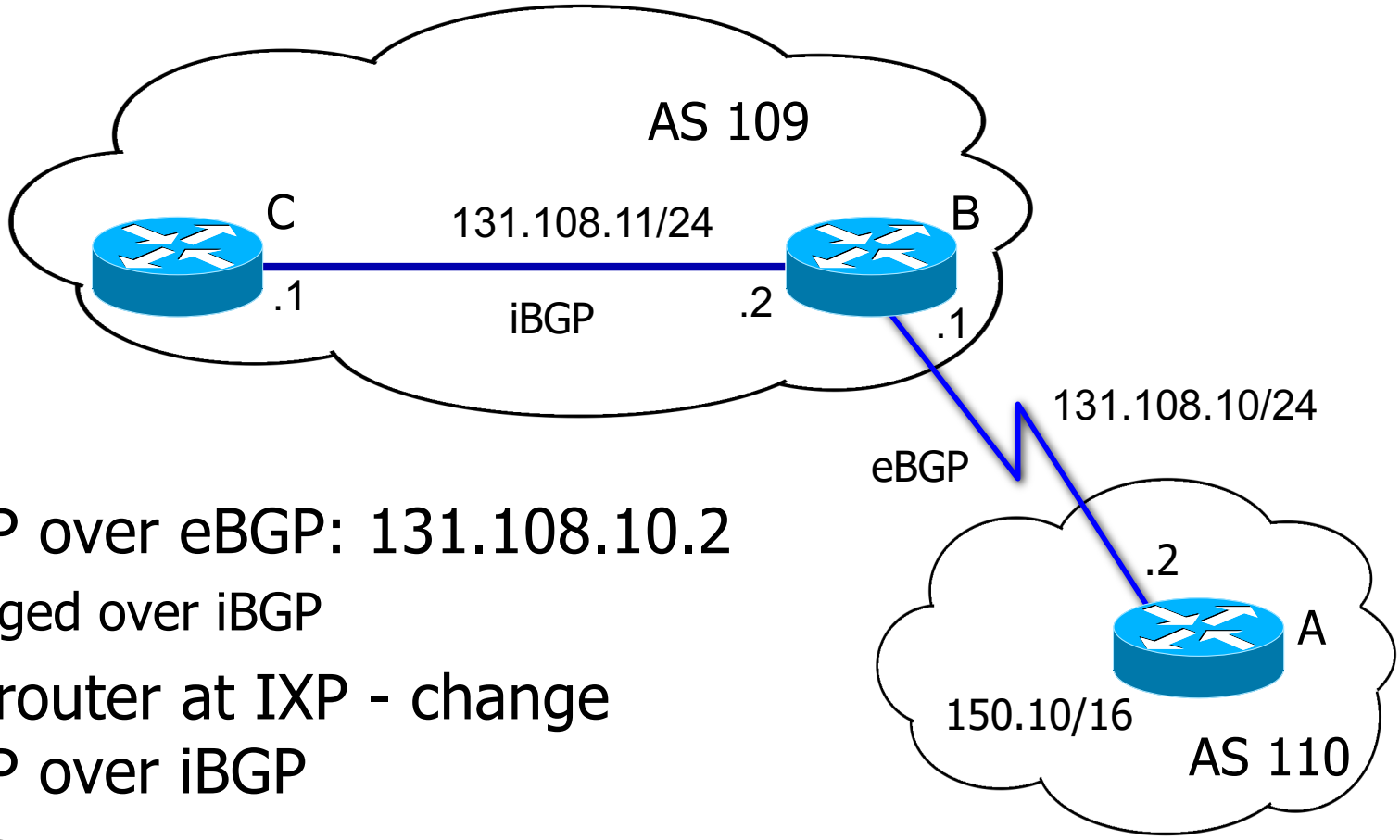
- Next hop not changed between iBGP peers

Third-Party NEXT_HOP

- Example:
  - A and B are in the same AS
  - Router A will advertise 192.68.1.0/24 with a NEXT_HOP of 150.1.1.3
- More efficient!

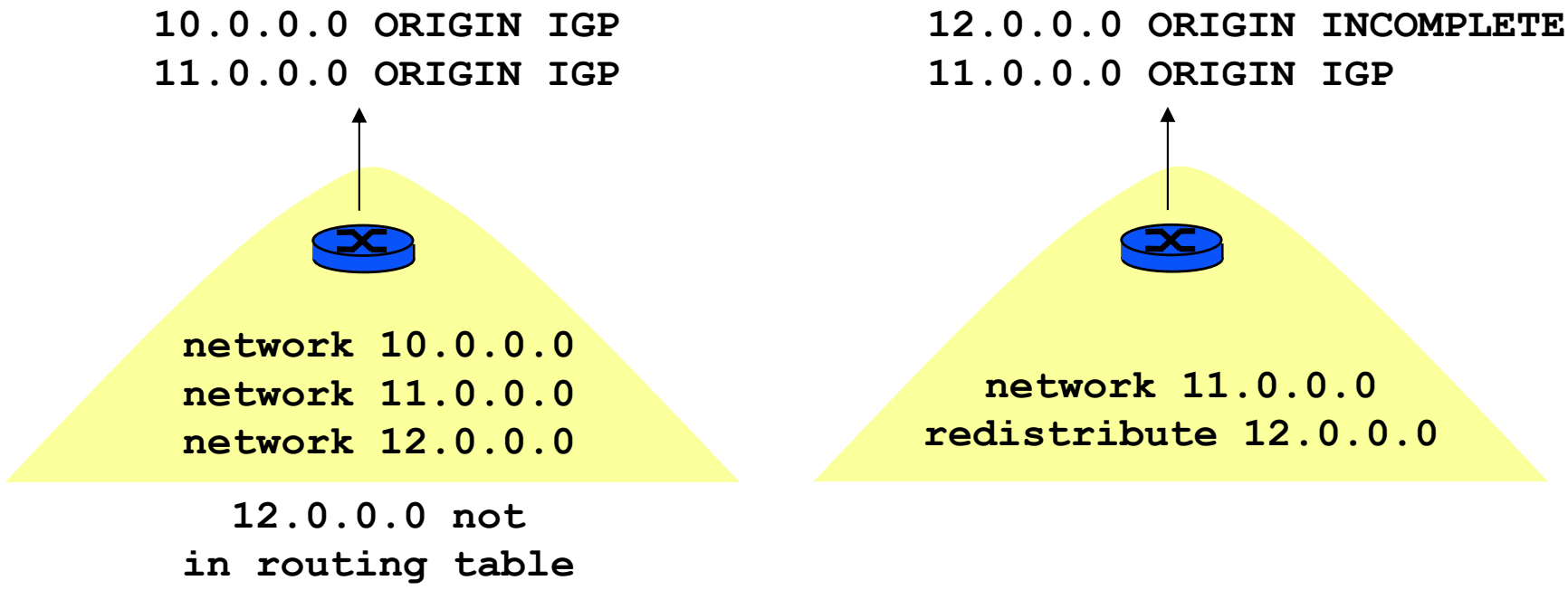


Next-hop-self NEXT_HOP



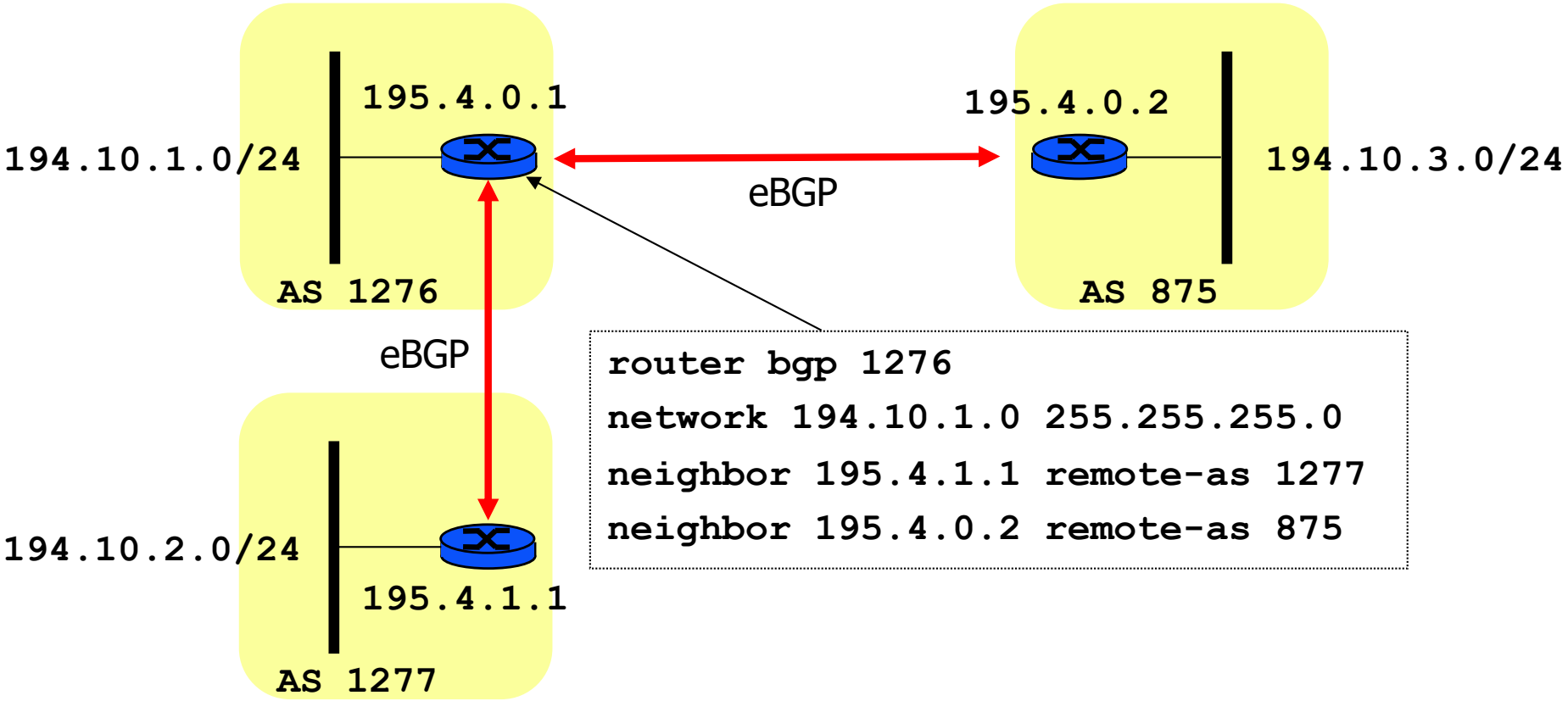
- NEXT_HOP over eBGP: 131.108.10.2
  - not changed over iBGP
- Small AS, router at IXP - change NEXT_HOP over iBGP
- At router B:
  - **`neighbor 131.108.11.1 next-hop-self`**
  - NEXT_HOP becomes 131.108.11.2

ORIGIN



- Source of information
  - IGP (i): static route explicitly injected into BGP by **`network`** directive
    - exists in the routing table
  - EGP (e): route learned via BGP
  - INCOMPLETE (?): another origin (by **`redistribute`** directive)

Example

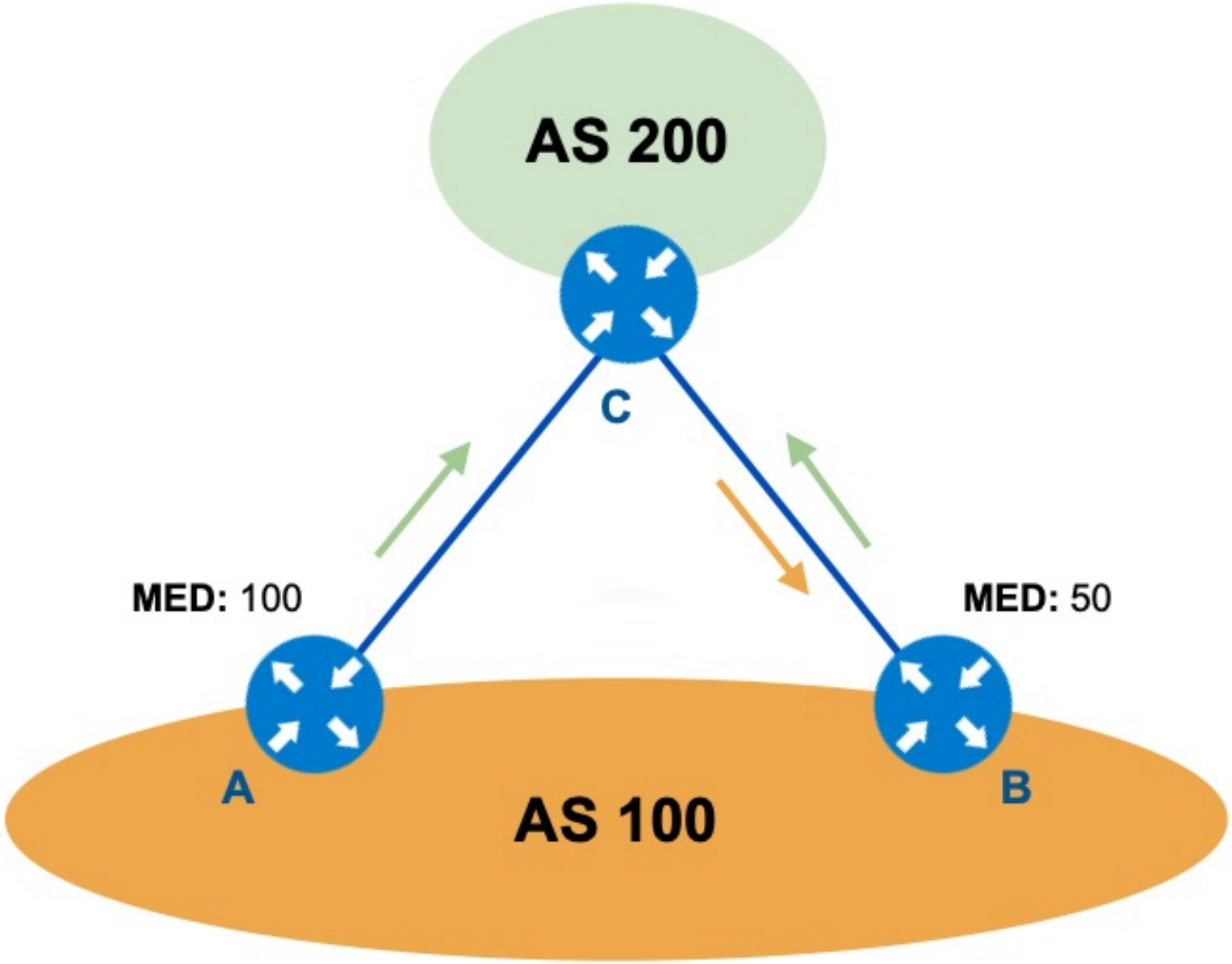


- AS 1276: network 194.10.1/24 ORIGIN=IGP

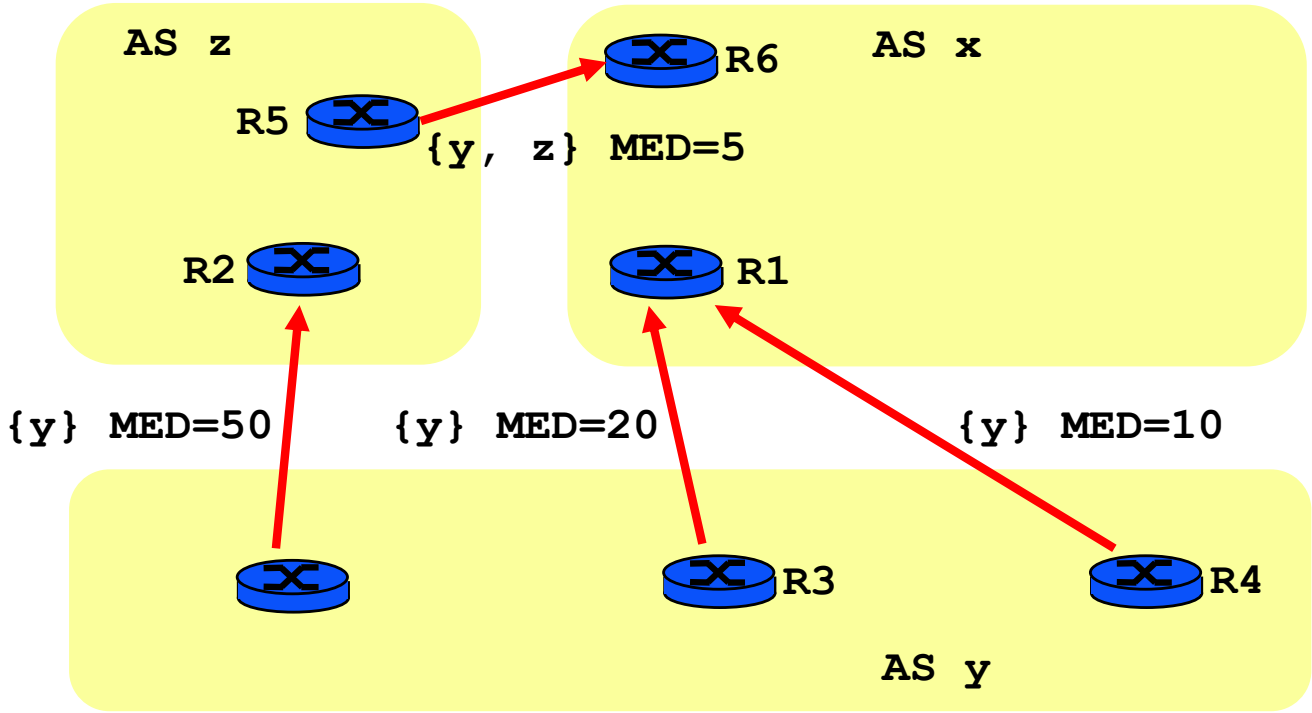
Preference attributes

- When multiple routes exist, choose one route to put into the BGP routing table
- Preference information
  - passed to other ASs - **MED**
  - local to an AS - **LOCAL_PREF**
  - local to a BGP router - **WEIGHT**

MULTI_EXIT_DISC (MED)

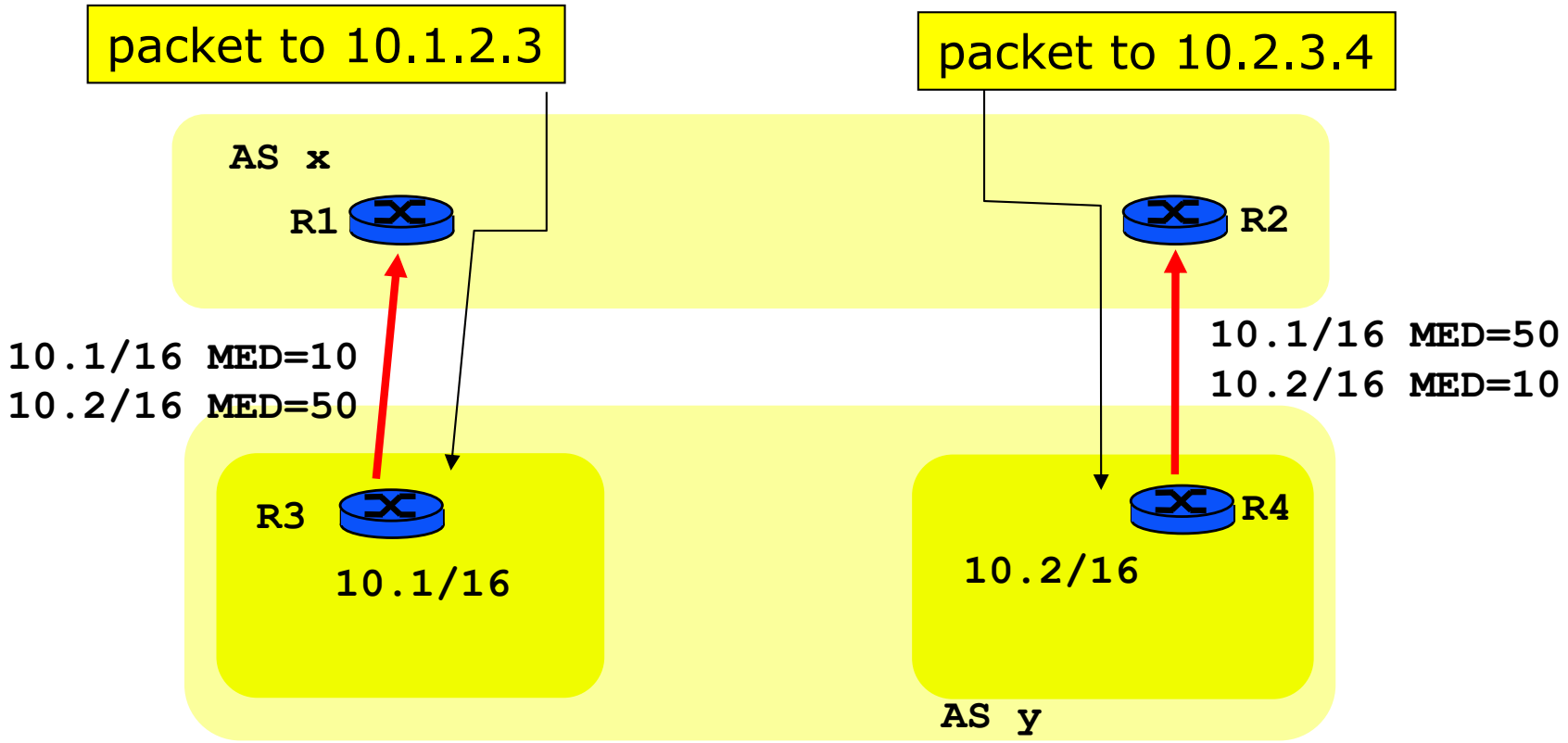


MULTI_EXIT_DISC (MED)



- Indication (to external peers) of the preferred path into AS
  - AS y advertises its prefixes with MED 10, 20, 50
  - AS x will accept the prefix with the smallest MED
- Compared only for routes from the same AS
  - unless **`bgp always-compare-med`** is enabled

MULTI-EXIT-DISC (MED)



- One AS connected to another over several links
  - ex: multinational company connected to worldwide ISP
  - AS y advertises its prefixes with different MEDs (low = preferred)
  - If AS x accepts to use MEDs put by AS y: traffic goes on preferred link

MED Example

- Q1: by which mechanisms will R1 and R2 make sure that packets to ASy use the preferred links?
  - R1 and R2 exchange their routes to AS y via I-BGP
  - R1 has 2 routes to 10.1/16, one of them learnt over E-BGP; prefers route via R1; injects it into IGP
  - R1 has 2 routes to 10.2/16, one of them learnt over E-BGP; prefers route via R2; does not inject a route to 10.2/16 into IGP
- Q2: router R3 crashes; can 10.1/16 still be reached ? explain the sequence of actions.
  - R1 clears routes to AS y learnt from R3 (keep-alive mechanism)
  - R2 is informed of the route suppression by I-BGP
  - R2 has now only 1 route to 10.1/16 and 1 route to 10.2/16;. keeps both routes in its local RIB and injects them into IGP since both were learnt via E-BGP
  - traffic to 10.1/16 now goes to R2

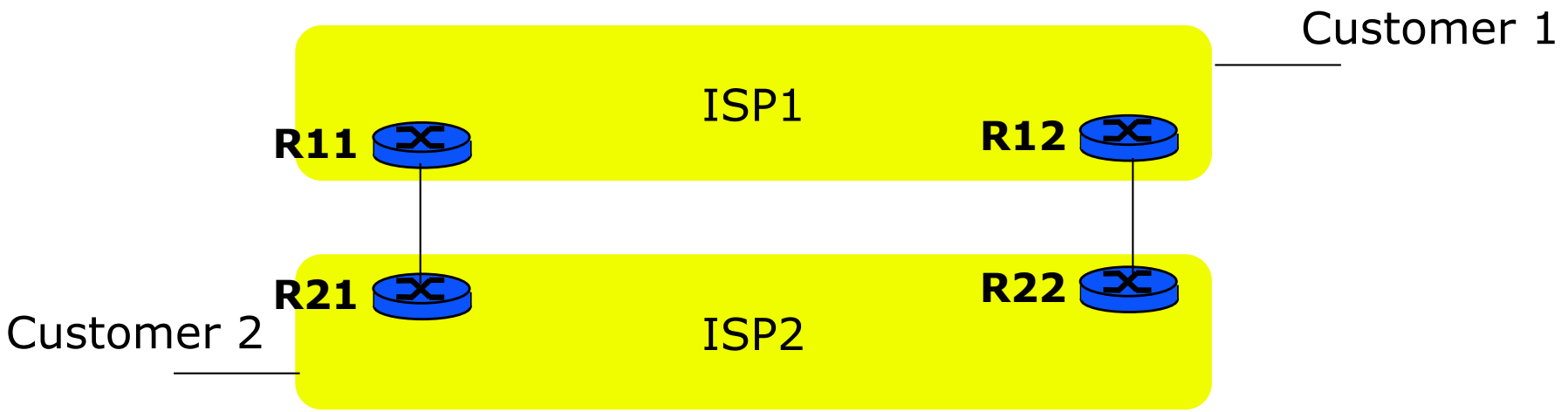
MED Question

- Q1: Assume now AS x and AS y are peers (ex: both are ISPs). Explain why AS x is not interested in taking MED into account.

  A: AS x is interested in sending traffic to AS y to the nearest exit, avoiding transit inside AS x as much as possible. Thus AS x will choose the nearest route to AS y and will ignore MEDs
- Q2: By which mechanisms can AS x pick the nearest route to AS y?

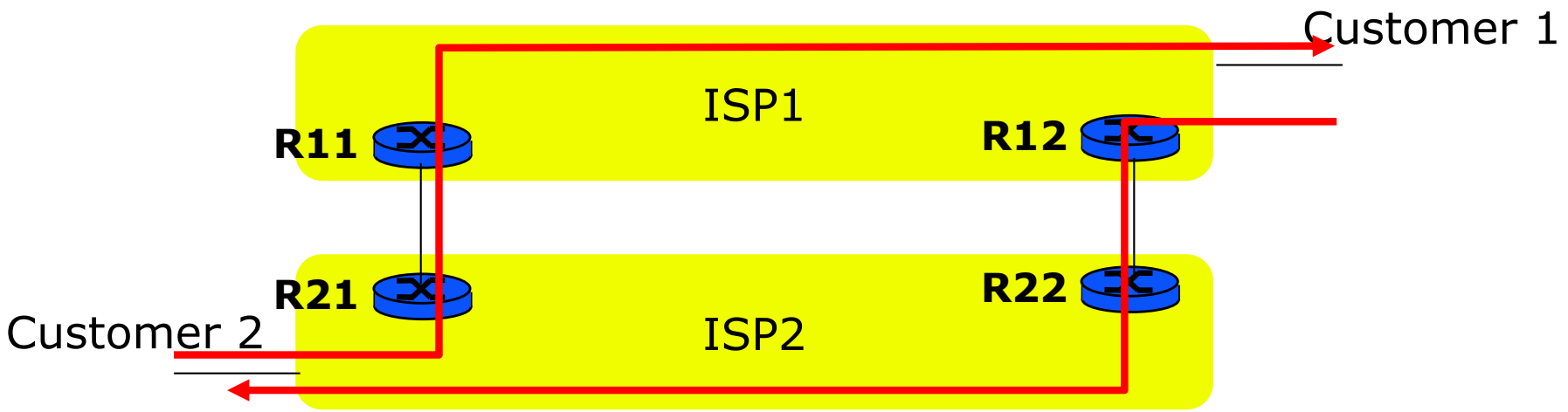
  A: it depends on the IGP. With OSPF: all routes to AS y are injected into OSPF by means of type 5 LSAs. These LSAs say: send to router R3 or R4. Every OSPF router inside AS x knows the cost (determined by OSPF weights) of the path from self to R3 and R4. Packets to 10.1/16 and 10.2/16 are routed to the nearest among R3 and R4 (nearest = lowest OSPF cost).

Example MED: Hot Potato Routing



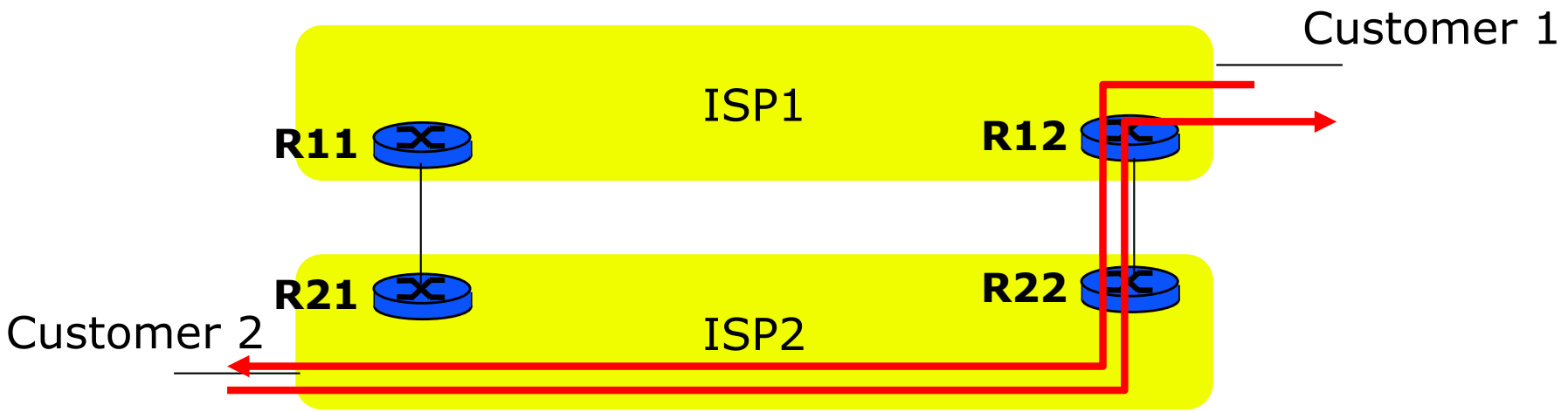
- Packets from Customer 2 to Customer 1
  - Both R21 and R22 have a route to Customer 1
  - Shortest path routing favors R21
  - Q1: by which mechanism is that done?
- Q2: what is the path followed in the reverse direction?

Example MED: Hot Potato Routing



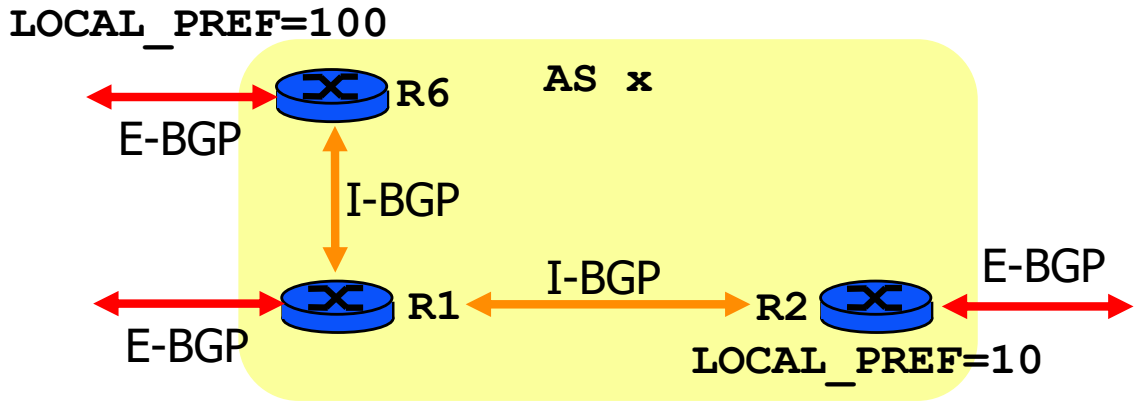
- Packets from Customer 2 to Customer 1
  - Both R21 and R22 have a route to Customer 1
  - Shortest path routing favors R21
  - Q1: by which mechanism is that done?
  - A: « Choice of the best route » (criterion 7), assuming all routers in ISP2 run BGP
- Q2: what is the path followed in the reverse direction?
  - A: see picture. Note the asymmetric routing

Cold Potato Routing – follow MED



- Packets from Customer 2 to Customer 1
  - Use the internal routing of ISP2 and enter ISP1 as indicated by MED
- No asymmetric routing

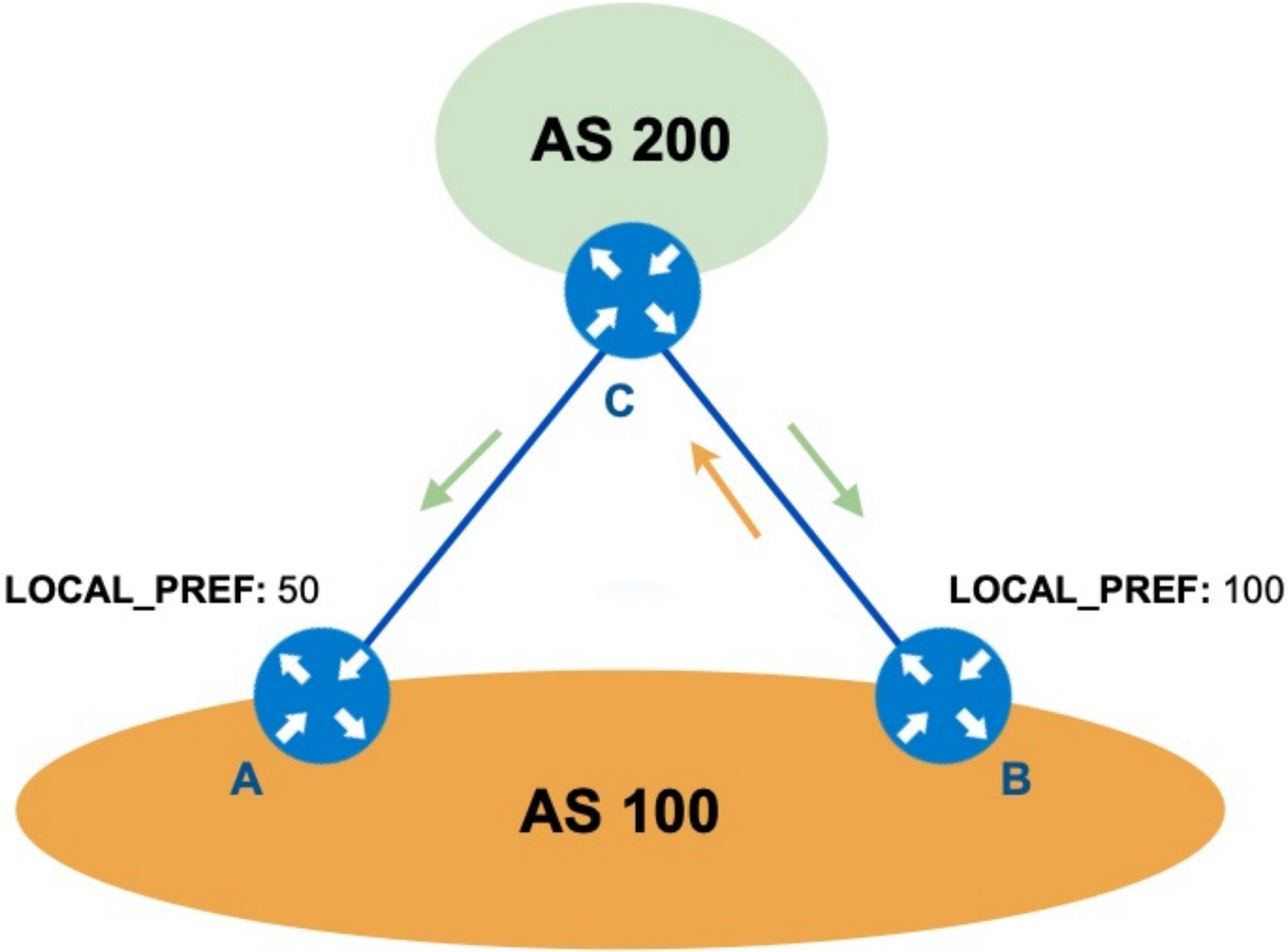
LOCAL_PREF



- Used to select the best route through an *AS path,* local to an AS; carried only in iBGP messages
- Assigned by border router when receiving route over E-BGP (100 by default)
  - Propagated without change over I-BGP
- Example
  - R6 associates pref=100, R2 pref=10
  - R1 chooses the largest preference

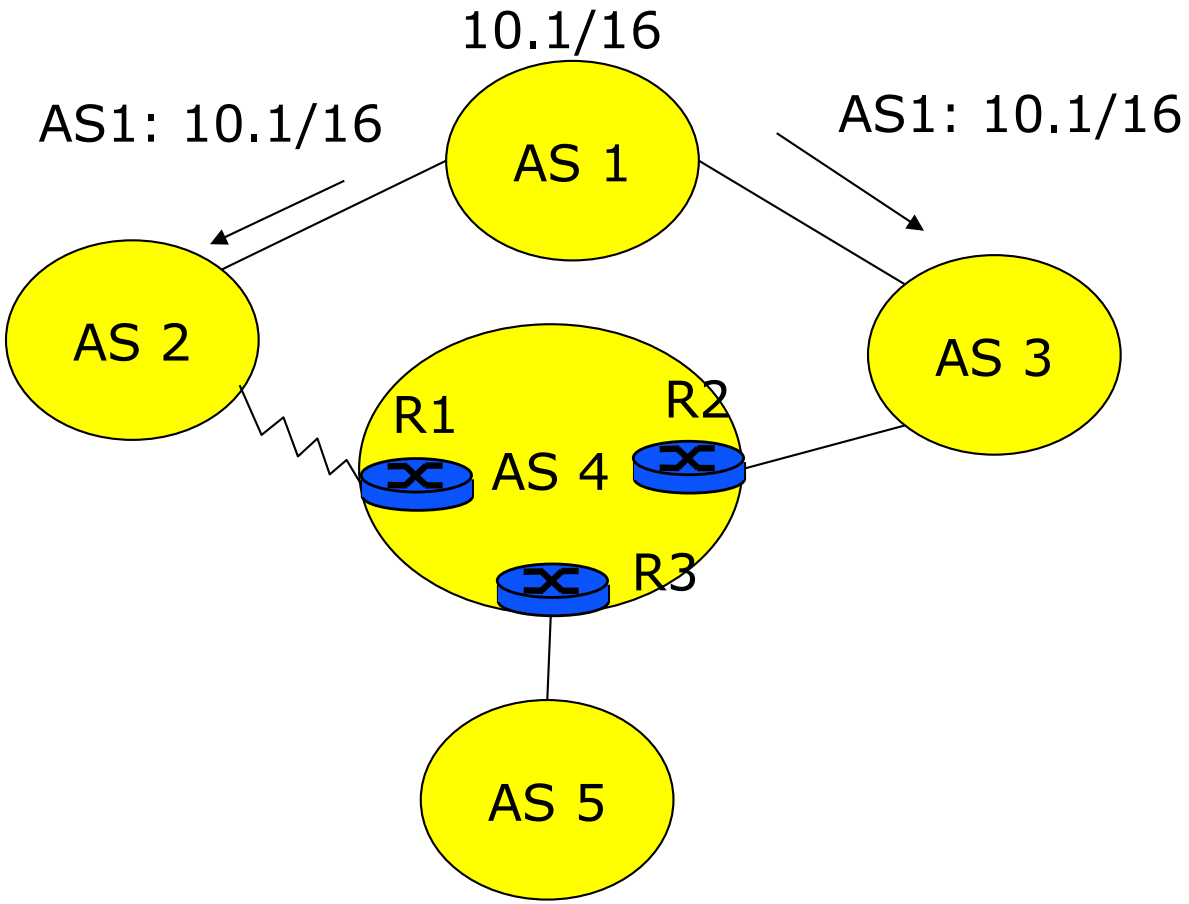
`bgp default local-preference` *`pref-value`*

LOCAL_PREF



LOCAL_PREF Example

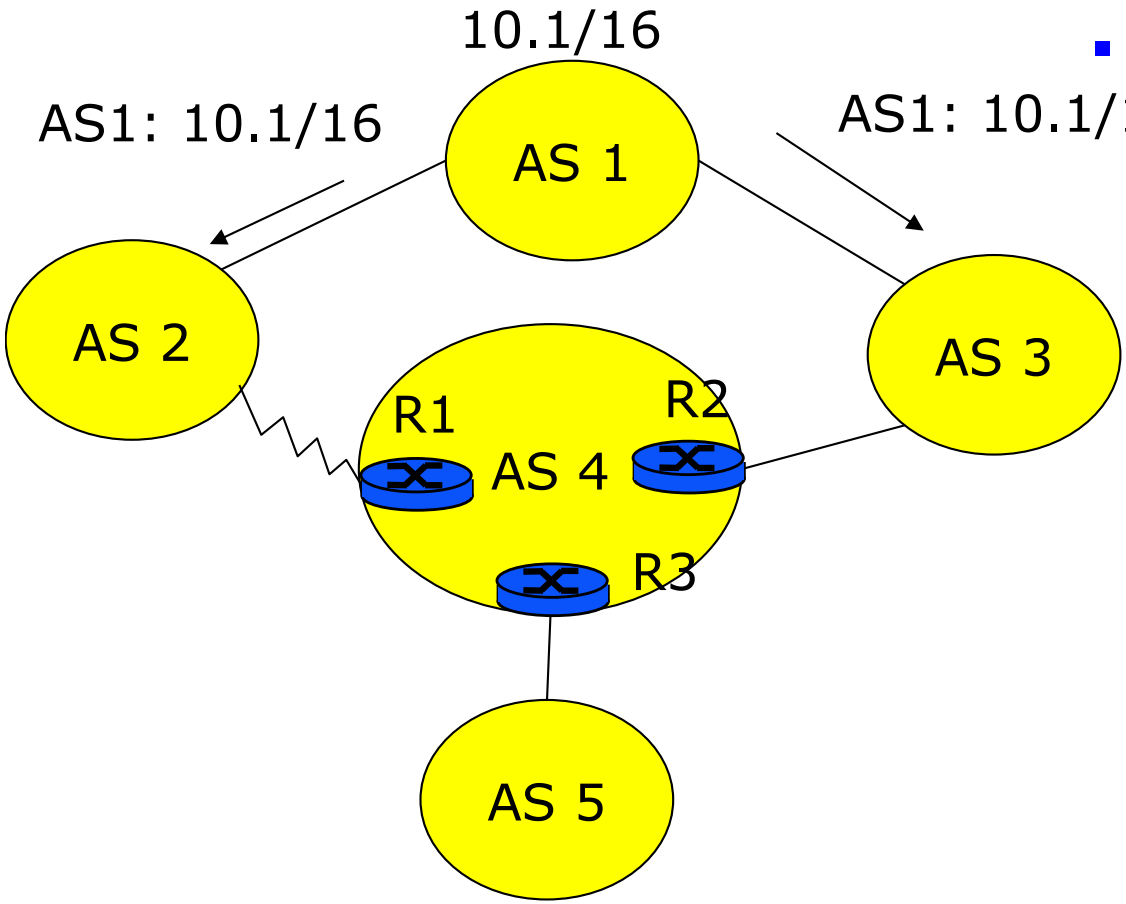
- Q1: The link AS2-AS4 is expensive. How should AS 4 set local-prefs on routes received from AS 3 and AS 2 in order to route traffic preferably through AS 3 ?
- Q2:Explain the sequence of events for R1, R2 and R3.



LOCAL_PREF Example

- Q1: The link AS2-AS4 is expensive. How should AS 4 set local-prefs on routes received from AS 3 and AS 2 in order to route traffic preferably through AS 3 ?

  A: for example: set LOCAL_PREF to 100 to all routes received from AS 3 and to 50 to all routes received from AS 2



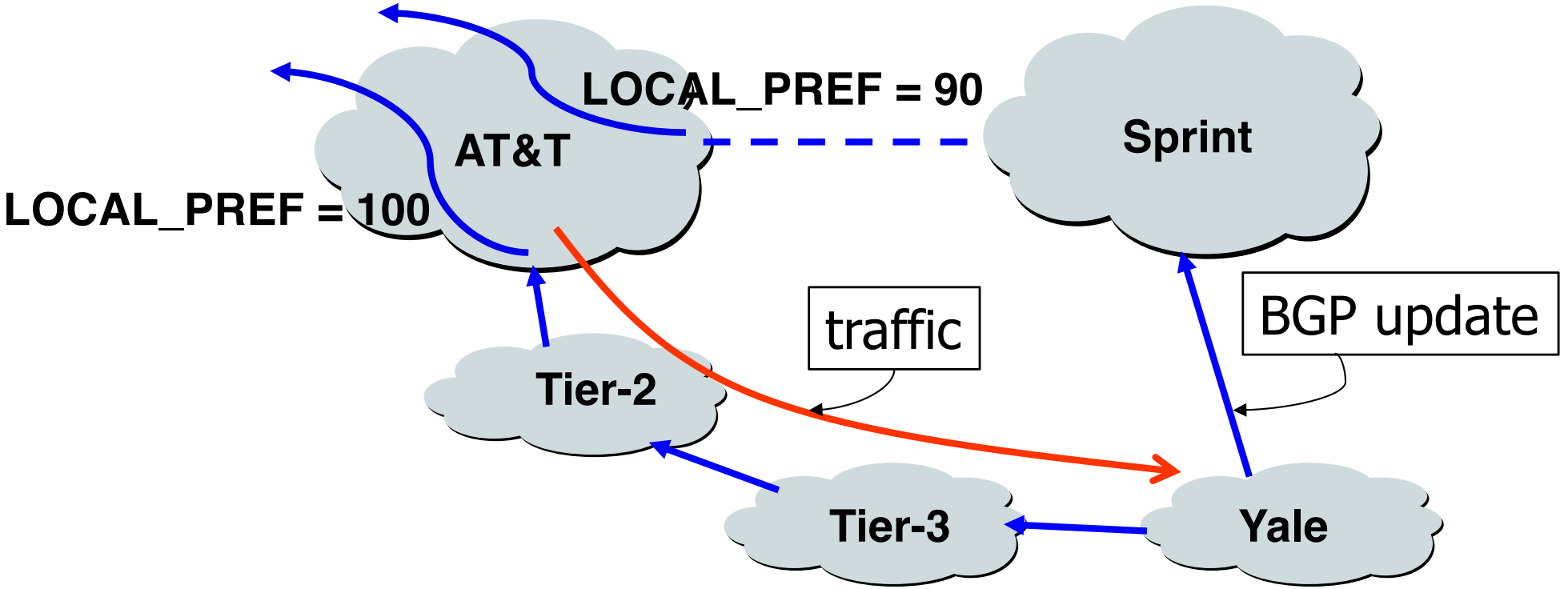
- Sequence of events
  - R1 receives the route AS2 AS1 10.1/16 over E-BGP; sets LOCAL_PREF to 50
  - R2 receives the route AS3 AS1 10.1/16 over E-BGP; sets LOCAL_PREF to 100
  - R3 receives AS2 AS1 10.1/16, LOCAL_PREF=50 from R1 over I-BGP and AS3 AS1 10.1/16, LOCAL_PREF=100 from R1 over I-BGP
  - R3 selects AS3 AS1 10.1/16, LOCAL_PREF=100 and installs it into local-RIB
  - R3 announces only AS3 AS1 10.1/16 to AS 5

LOCAL_PREF Question

- Q: Compare MED to LOCAL_PREF
  A:
  - MED is used between ASs (i.e. over E-BGP); LOCAL_PREF is used inside one AS (over I-BGP)
  - MED is used to tell one provider AS which *entry link* to prefer; LOCAL_PREF is used to tell the rest of the world which *AS path* we want to use, by not announcing the other ones.

"Prefer Customer" Import Policy: Local Preference

- Favor one path over another (even over a shorter path)
  - Override the influence of AS path length
  - Apply local policies to prefer a path
- Example: AT&T will prefer customer over peer – routes over iBGP with LOCAL_PREF=100

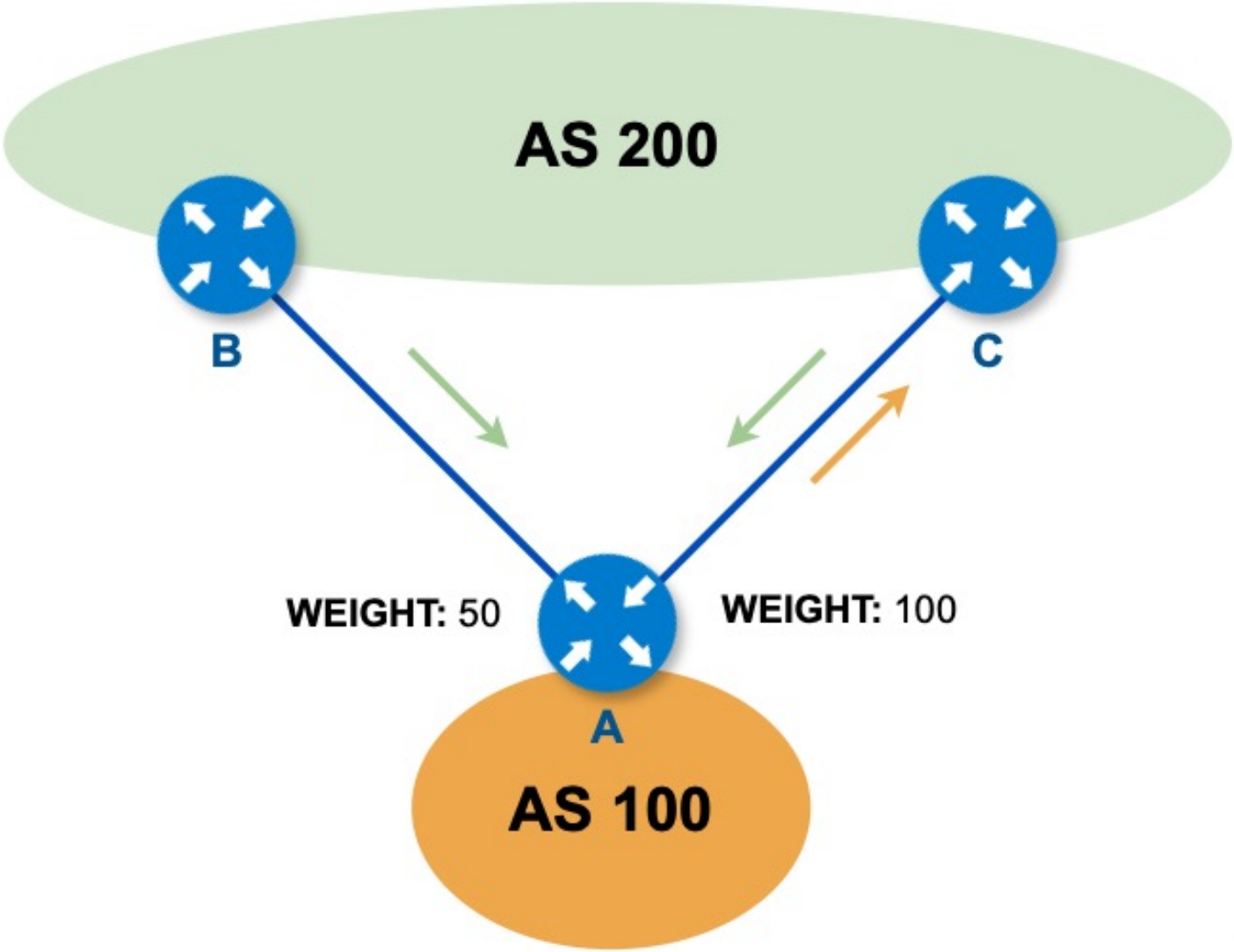


WEIGHT

- Cisco specific (sort of router internal local preference)
- Associate a weight with a neighbor
- For a local choice at a BGP router

  `neighbor` *`IP-address`* **`weight`** *`weight-value`*
- The route passing via the neighbor of the largest weight will be chosen
- Local to the router
  - Not propagated

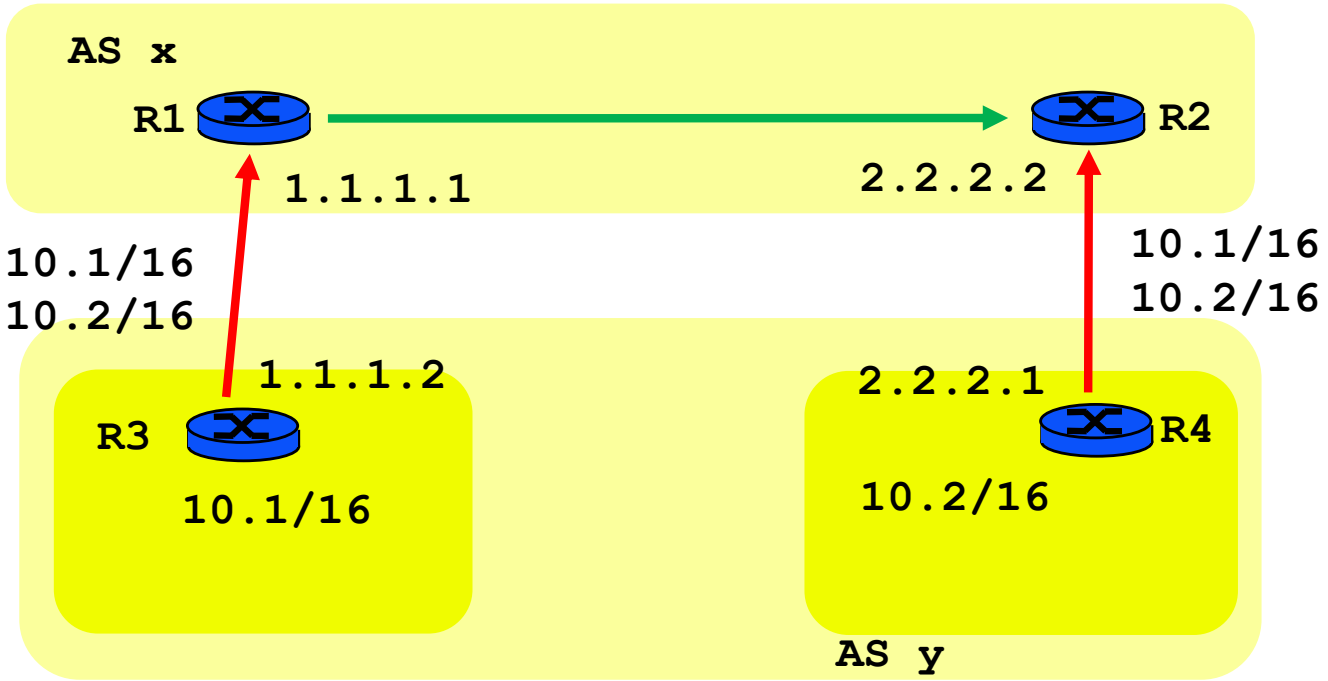
WEIGHT



Choice of the best route

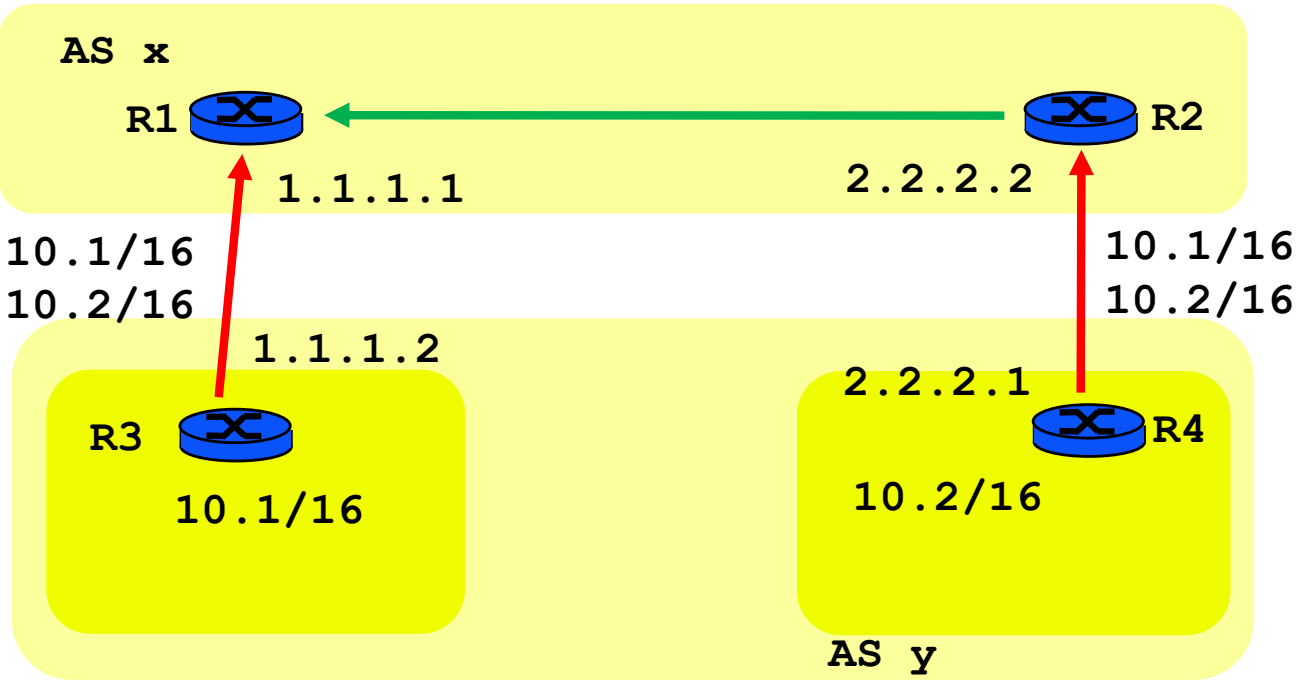
- Done by **decision process**; route installed in Loc-RIB
- Choose one best route to exactly the same prefix
  - Only one route to 2.2/16 can be chosen
  - But there can be different routes to 2.2.2/24 and 2.2/16
- Route validation: check if NEXT_HOP is accessible
- Decreasing priority (configurable, skip some steps)
  1. Highest WEIGHT (CISCO)
  2. Highest LOCAL_PREF (customer > peer > provider)
  3. Shortest AS_PATH
  4. ORIGIN attribute IGP > EGP > INCOMPLETE
  5. Lowest MULTI_EXIT_DISC
  6. Source of the route: E-BGP > I-BGP (hot potato routing)
  7. Shortest IGP distance to NEXT_HOP (hot potato routing)
  8. Lowest Next-Hop Router-ID

Example



- R3 → R1, R1 Adj-RIB-in:
  - `10.1/16 AS-PATH=y NEXT-HOP=1.1.1.2, best`
  - `10.2/16 AS-PATH=y NEXT-HOP=1.1.1.2, best`
- Adj-RIB-out, R1→ R2:
  - `10.1/16 AS-PATH=y NEXT-HOP=1.1.1.2, to R2`
  - `10.2/16 AS-PATH=y NEXT-HOP=1.1.1.2, to R2`

Example



- R2 -> R1, R1 Adj-RIB-in:
  - `10.1/16 AS-PATH=y NEXT-HOP=1.1.1.2, best`
  - `10.1/16 AS-PATH=y NEXT-HOP=2.2.2.1, from R2`
  - `10.2/16 AS-PATH=y NEXT-HOP=1.1.1.2, best`
  - `10.2/16 AS-PATH=y NEXT-HOP=2.2.2.1, from R2`
- Routes from R3 learnt over eBGP, preferred over iBGP
- No change in BGP tables, no message sent by R1.

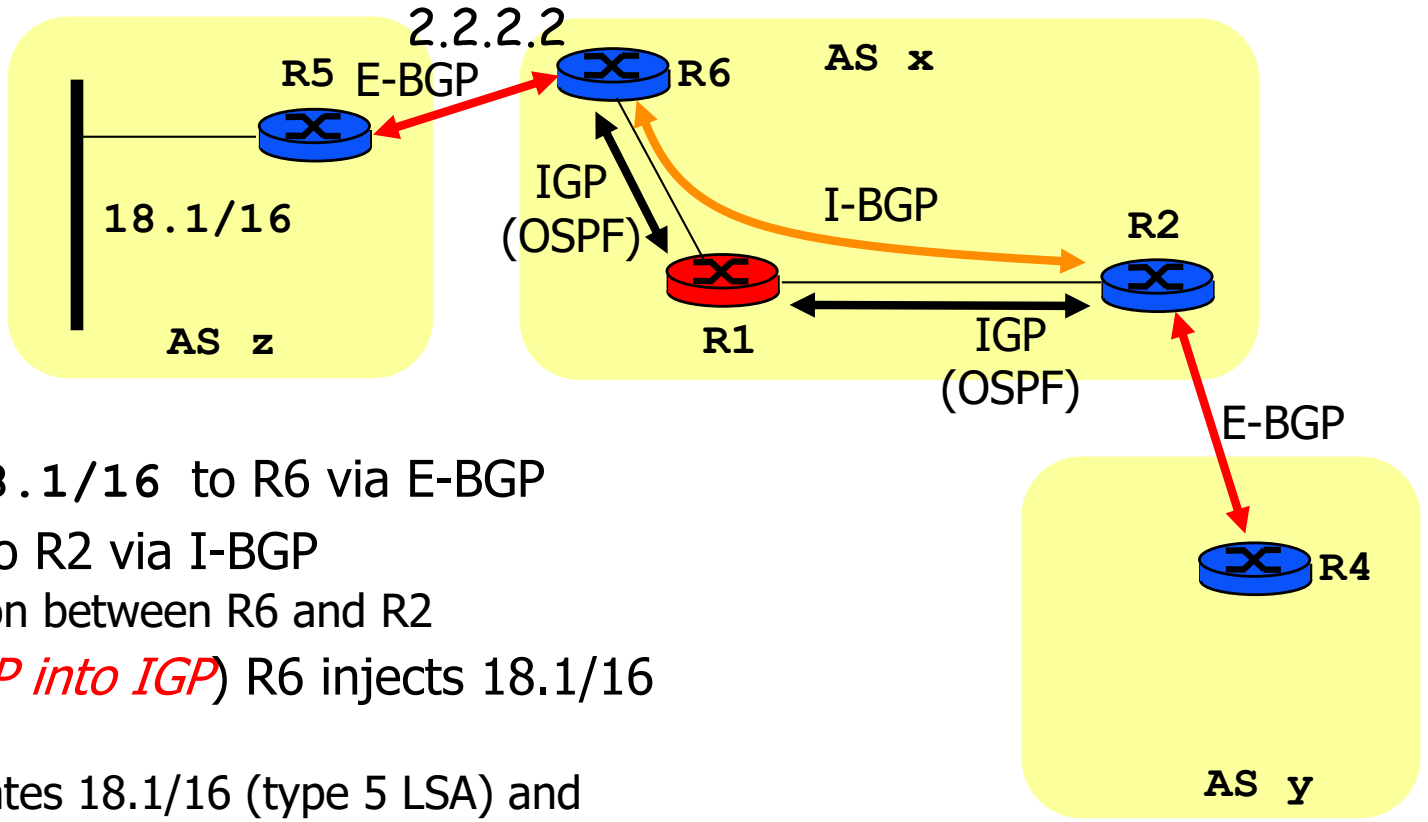
What happens when a BGB router loses its best route to some destination ?

- It sends a WITHDRAW update to the BGP peers to whom it had sent this route, as soon as possible

Interaction BGP—IGP—Packet Forwarding

- How BGP routers inform all the routers in their AS about prefixes they learn?
- There are main two interactions between BGP and internal routing that you have to know
- *Redistribution*: routes learnt by BGP are passed to IGP (ex: OSPF)
  - Called "redistribution of BGP into OSPF"
  - OSPF propagates the routes using type 5 LSAs to all routers in OSPF cloud
- *Injection*: routes learnt by BGP are written into the forwarding table of this router
  - Routes do not propagate; this helps only this router

Redistribution Example



- R5 advertises `18.1/16` to R6 via E-BGP
- R6 transmits it to R2 via I-BGP
  - TCP connection between R6 and R2
- (*redistribute BGP into IGP*) R6 injects 18.1/16 into IGP (OSPF)
  - OSPF propagates 18.1/16 (type 5 LSA) and updates forwarding tables
  - After OSPF converges, R1, R2 now have a route to 18.1/6
- R2 advertises route to R4 via E-BGP
  - (*synchronize with IGP*) R2 must wait for the OSPF entry to 18.1/16 before advertising via E-BGP
- Packet to `18.1/16` from AS y finds forwarding table entries in R2, R1 and R6

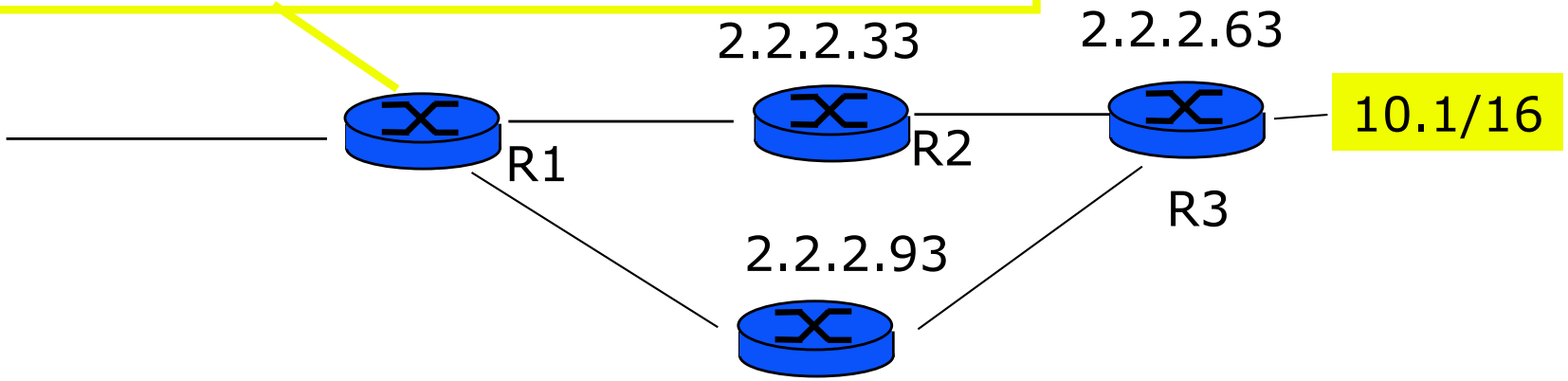
Re-Distribution Considered Harmful

- In practice, operators avoid re-distribution of BGP into IGP
  - Large number of routing entries in IGP
  - Reconvergence time after failures is large if IGP has many routing table entries
- A classical solution is based on *recursive table lookup*
  - When IP packet is submitted to router, the forwarding table may indicate a "NEXT-HOP" which is not on-link with router
  - A second table lookup needs to be done to resolve the next-hop into an on-link neighbour
    - in practice, second lookup is done in advance – not in real time– by preprocessing the routing table

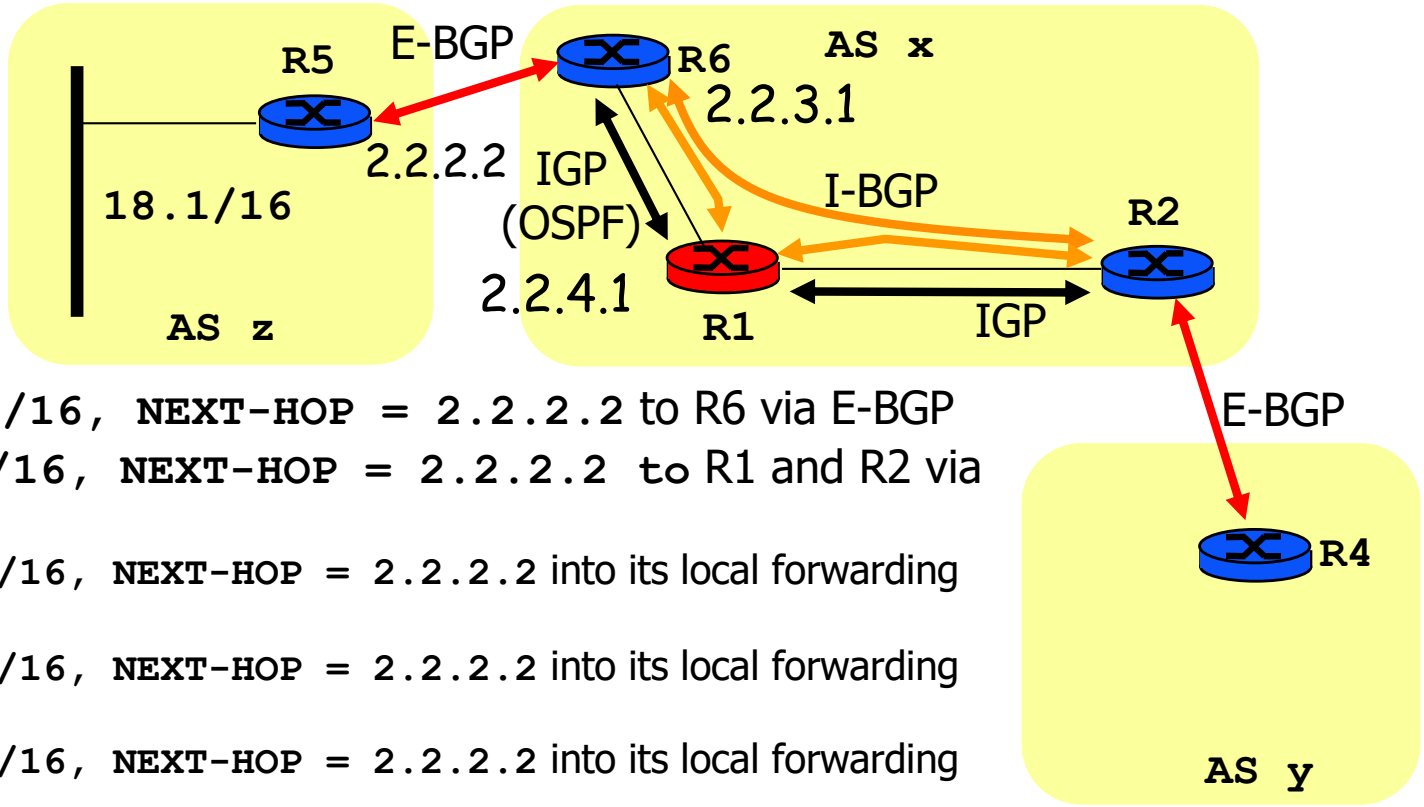
Example: Recursive Table Lookup

- At R1, data packet to 10.1.x.y is received
- The forwarding table at R1 is looked up
  - Q: what are the next events ?
  - A: first, the next-hop 2.2.2.63 is found; a second lookup for 2.2.2.63 is done; the packet is sent to MAC address x09:F1:6A:33:76:21

	To	*NEXT-HOP*	*layer-2 addr*
BGP	10.1/16	2.2.2.63	N/A
IGP	2.2.2/24	2.2.2.33	x09:F1:6A:33:76:21

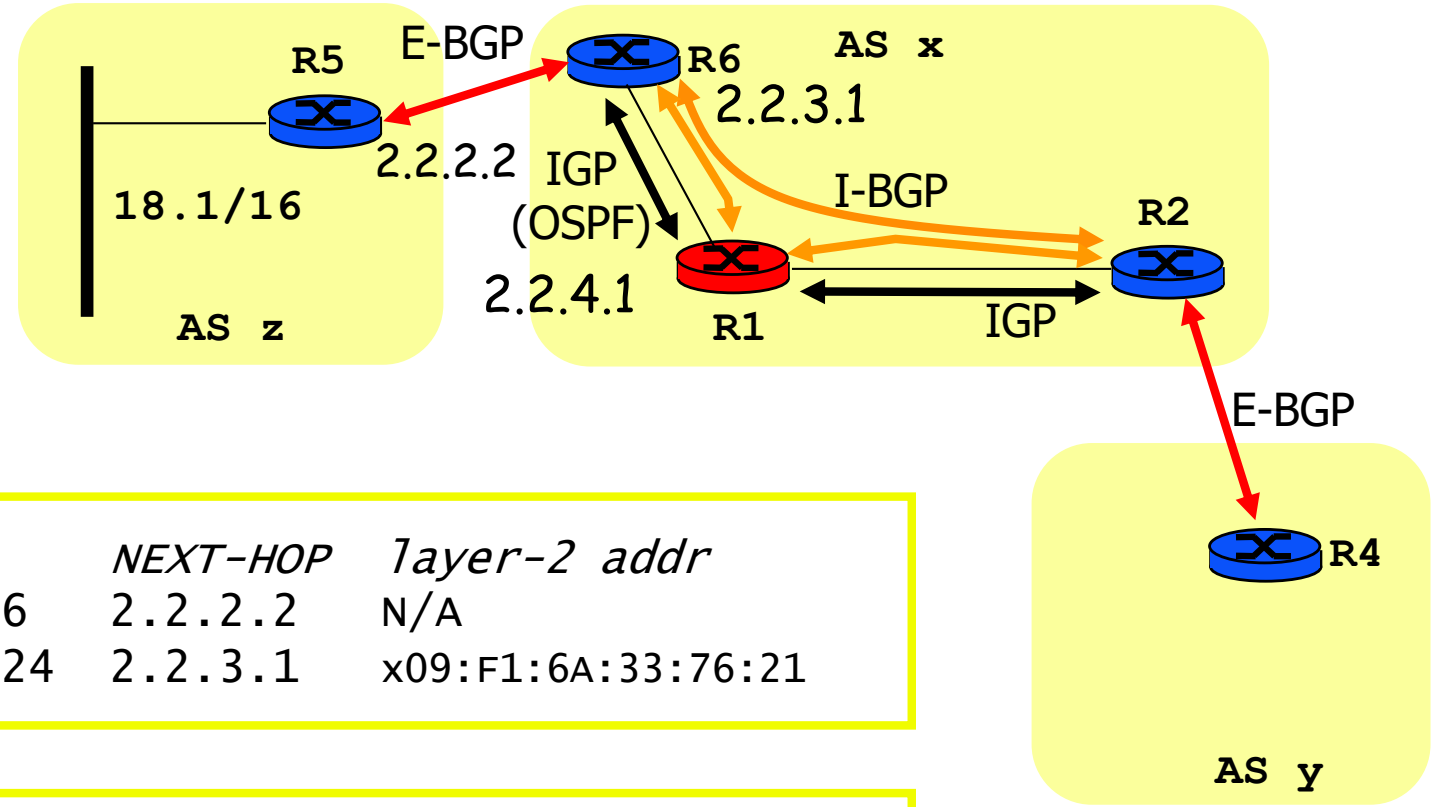


Practical Solution: run BGP everywhere



- R5 advertises `18.1/16, NEXT-HOP = 2.2.2.2` to R6 via E-BGP
- R6 transmits `18.1/16, NEXT-HOP = 2.2.2.2 to` R1 and R2 via I-BGP
  - R6 *injects* `18.1/16, NEXT-HOP = 2.2.2.2` into its local forwarding table
  - R1 *injects* `18.1/16, NEXT-HOP = 2.2.2.2` into its local forwarding table
  - R2 *injects* `18.1/16, NEXT-HOP = 2.2.2.2` into its local forwarding table
- Independently, IGP finds that at R2 packets to 2.2.2.2 should be sent to R1 (route to 2.2.2.2 goes through R1)
- Data packet to 18.1.2.3 is received by R2
  - At R2, recursive table lookup determines that packet should be forwarded to R1 (2.2.4.1)
  - At R1, recursive table lookup determines that packet should be forwarded to R6 (2.2.3.1)
  - At R6, table lookup determines that packet should be forwarded to 2.2.2.2

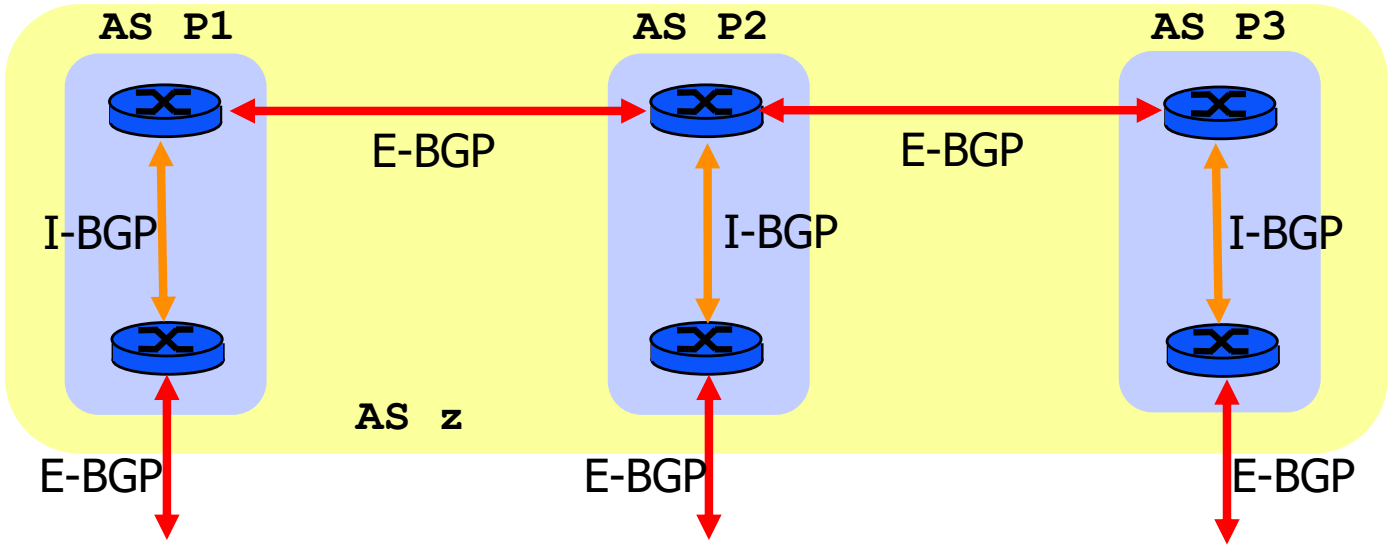
Practical Solution: run BGP everywhere



R1	*To*	*NEXT-HOP*	*layer-2 addr*
BGP	18.1/16	2.2.2.2	N/A
IGP	2.2.2/24	2.2.3.1	x09:F1:6A:33:76:21

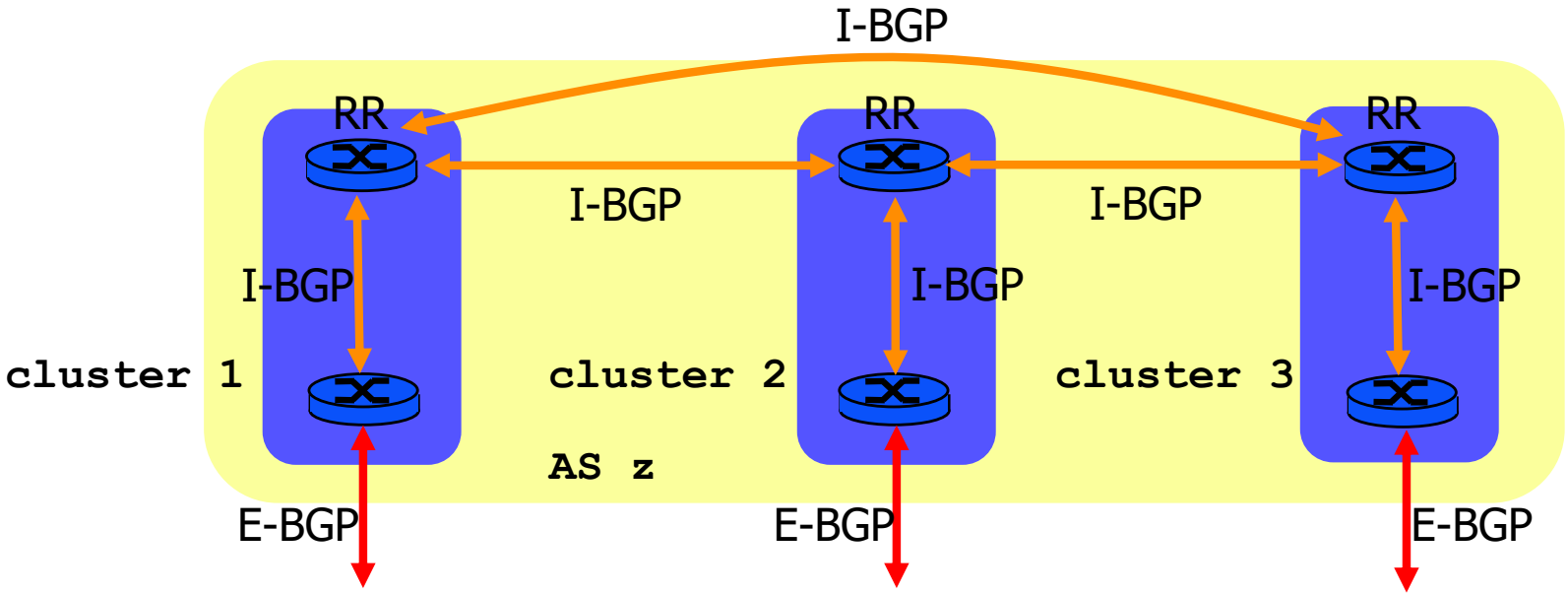
R2	*To*	*NEXT-HOP*	*layer-2 addr*
BGP	18.1/16	2.2.2.2	N/A
IGP	2.2.2/24	2.2.4.1	x09:F1:6A:33:66:12

Avoid I-BGP Mesh: Confederations



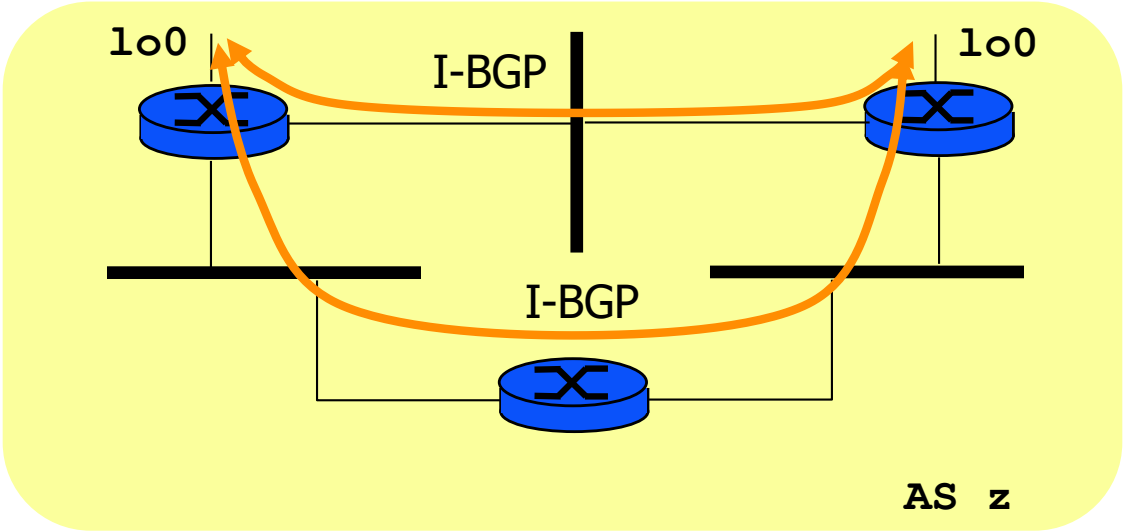
- AS decomposed into sub-AS
  - private AS number
  - similar to OSPF areas
    - I-BGP inside sub-AS (full interconnection)
    - E-BGP between sub-AS

Avoid I-BGP Mesh: Route reflectors



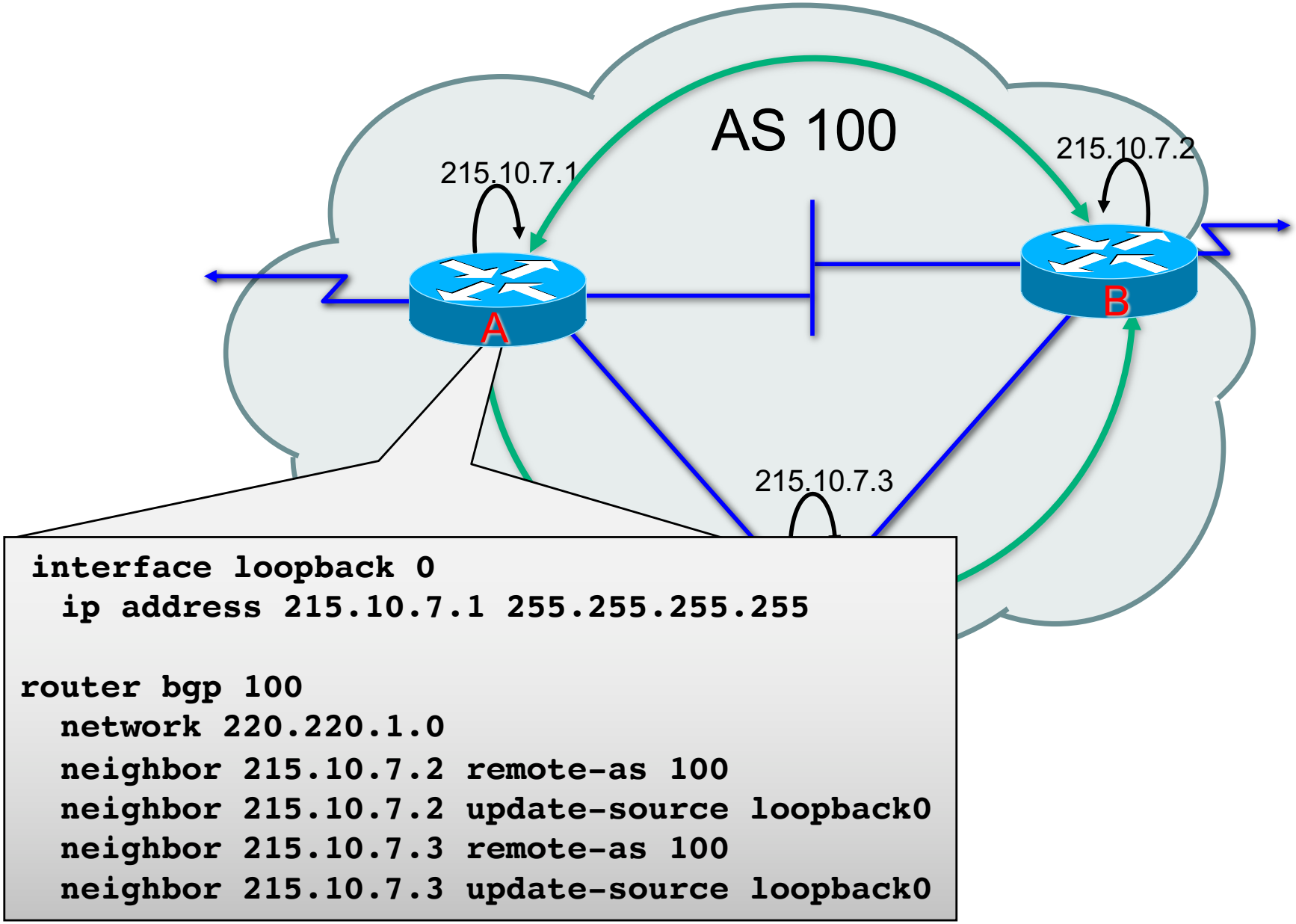
- Cluster of routers
  - one I-BGP session between one client and RR
  - CLUSTER_ID
- Route reflector
  - re-advertises a route learnt via I-BGP
  - to avoid loops
    - ORIGINATOR_ID attribute associated with the advertisement

I-BGP configuration

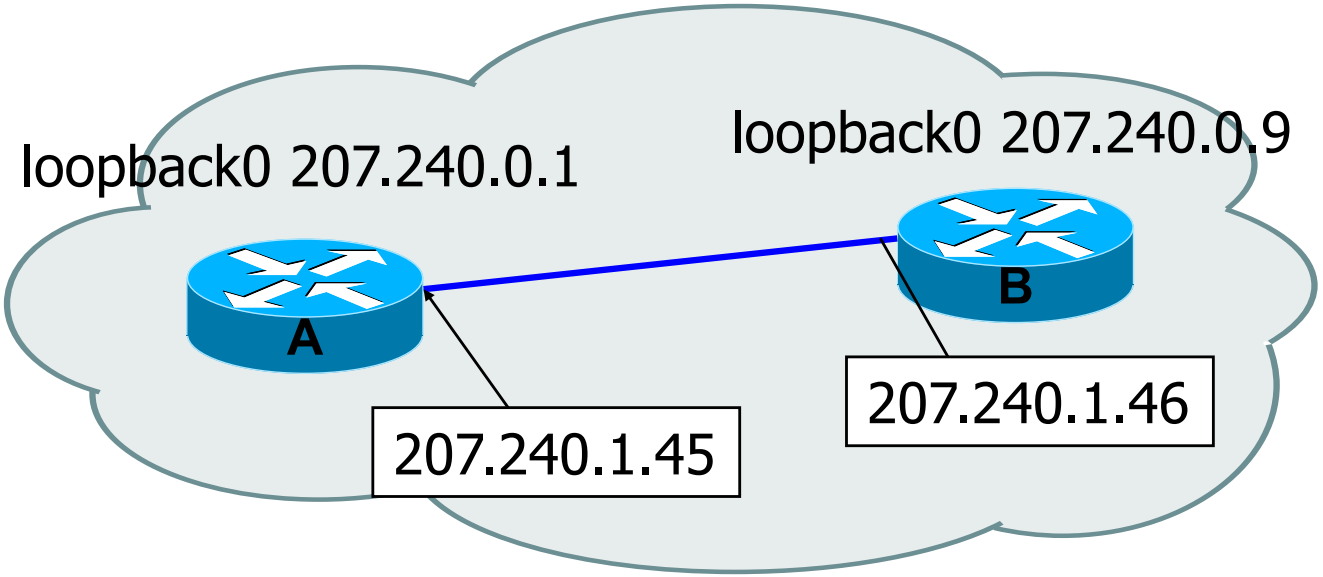


- I-BGP configured on loopback interface (lo0)
  - interface always up
  - IP address associated with the interface
  - IGP routing guarantees packet forwarding to the interface
- BGP router identifier (ID) - highest IP address on the router

Configuring BGP Peers

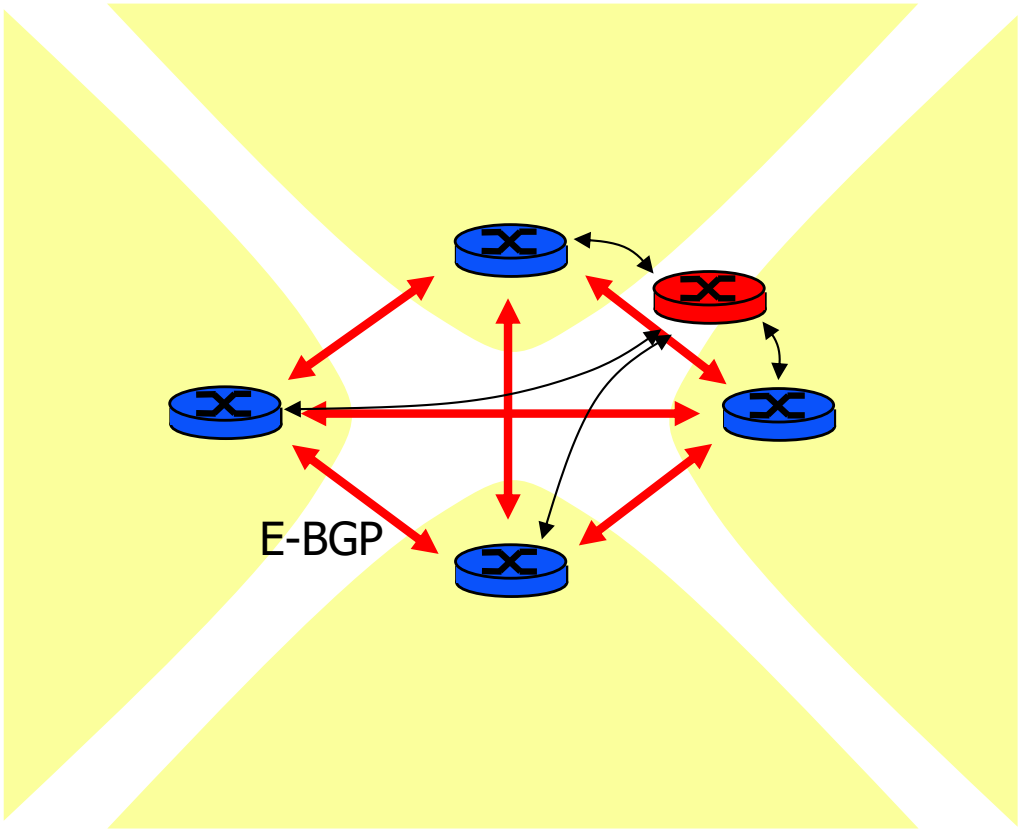


Update-Source Loopback0



- Source address of packets sent from router A to router B would be 207.240.1.45
- **`update-source loopback0`**: set the source address to that of the specified interface for all BGP packets sent to that peer

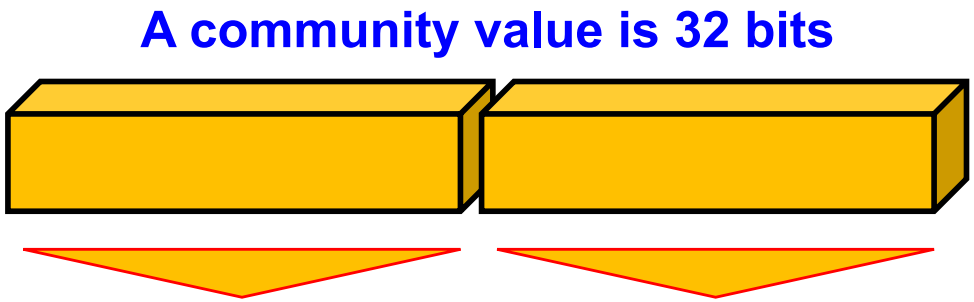
Avoid E-BGP mesh: Route server



- At interconnection point
- Instead of $n(n-1)/2$ peer to peer E-BGP connections n connections to Route Server
- To avoid loops ADVERTISER attribute indicates which router in the AS generated the route

Colored Routes - Communities

- **Community Attribute:**
  - **mark routes that share a common property**
  - **signal routes that needs to be processed in a predefined way**



Used for signaling within and between ASes

Very powerful BECAUSE it has no (predefined) meaning

Community Attribute = a list of community values

AS-no:x, x - value (0-65535)

one route can belong to multiple communities

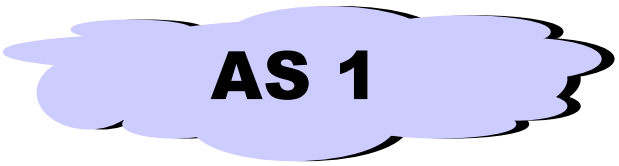
Communities Example

- 1:100 
  - Customer routes
- 1:200 
  - Peer routes
- 1:300 
  - Provider Routes

Import

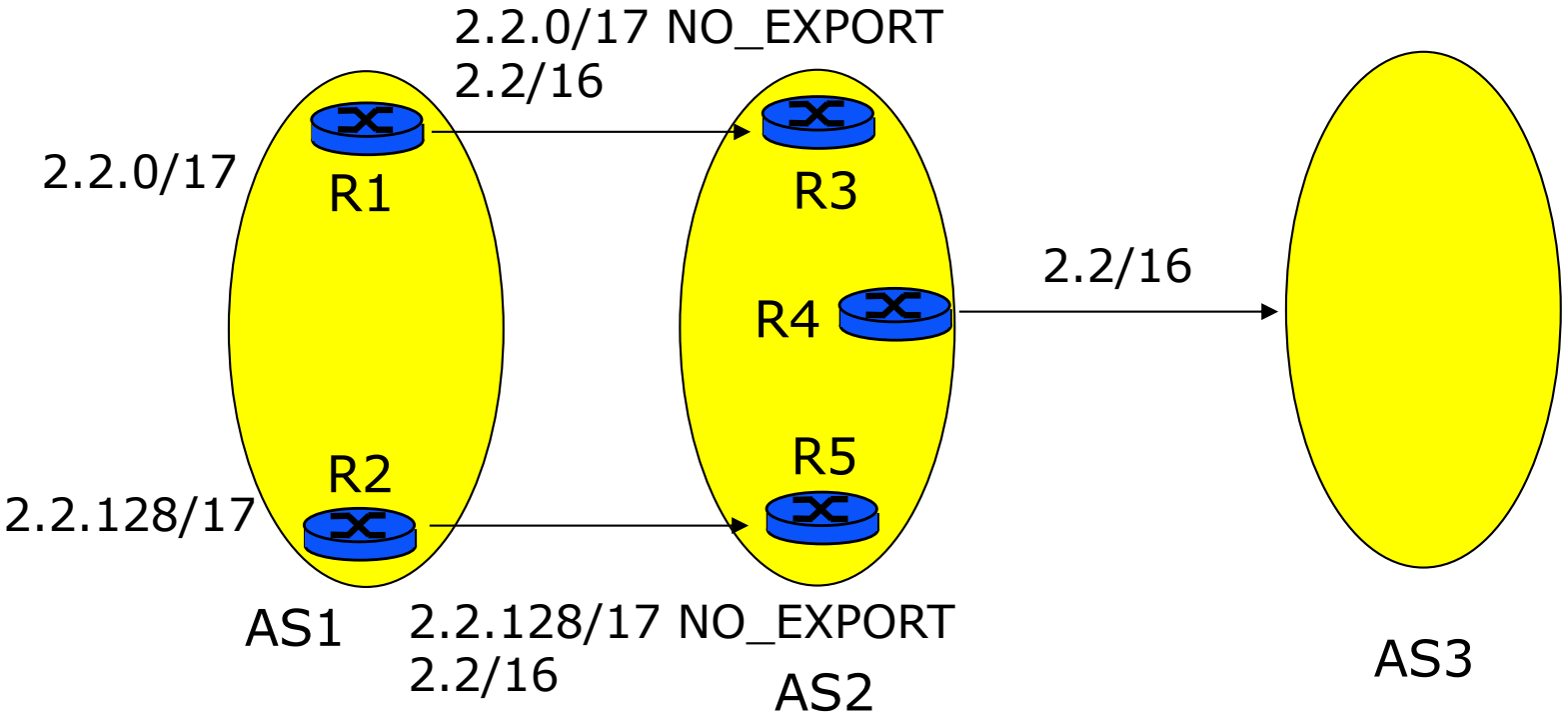
- To Customers
  - 1:100, 1:200, 1:300
- To Peers
  - 1:100
- To Providers
  - 1:100

Export



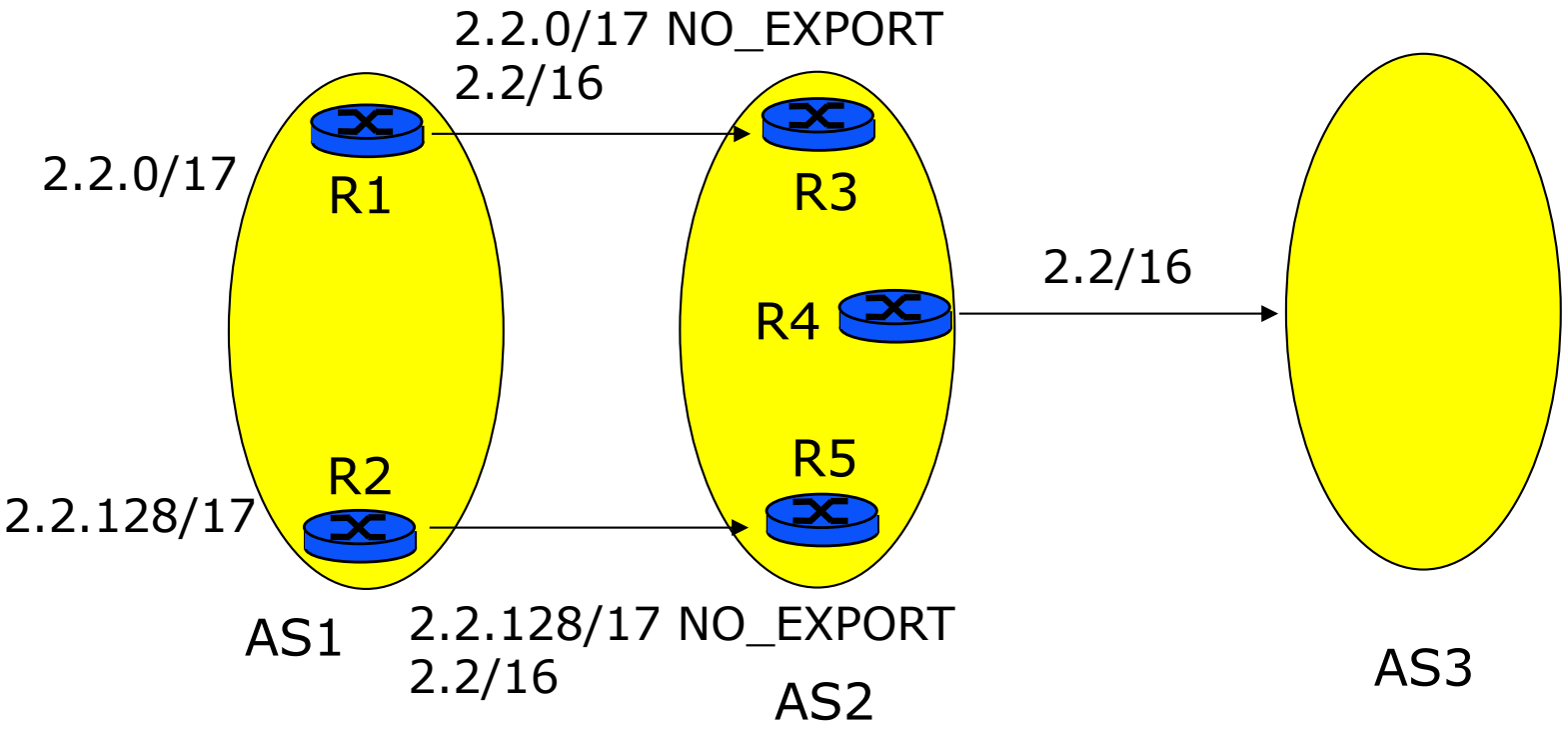
NO_EXPORT

- Written on E-BGP by one AS, transmitted on I-BGP by accepting AS, not forwarded
- Example: AS2 has different routes to AS1 but AS2 sends only one aggregate route to AS3
  - simplifies the aggregation rules at AS2
  - What is the route followed by a packet sent to 2.2.48 received by R4 ?

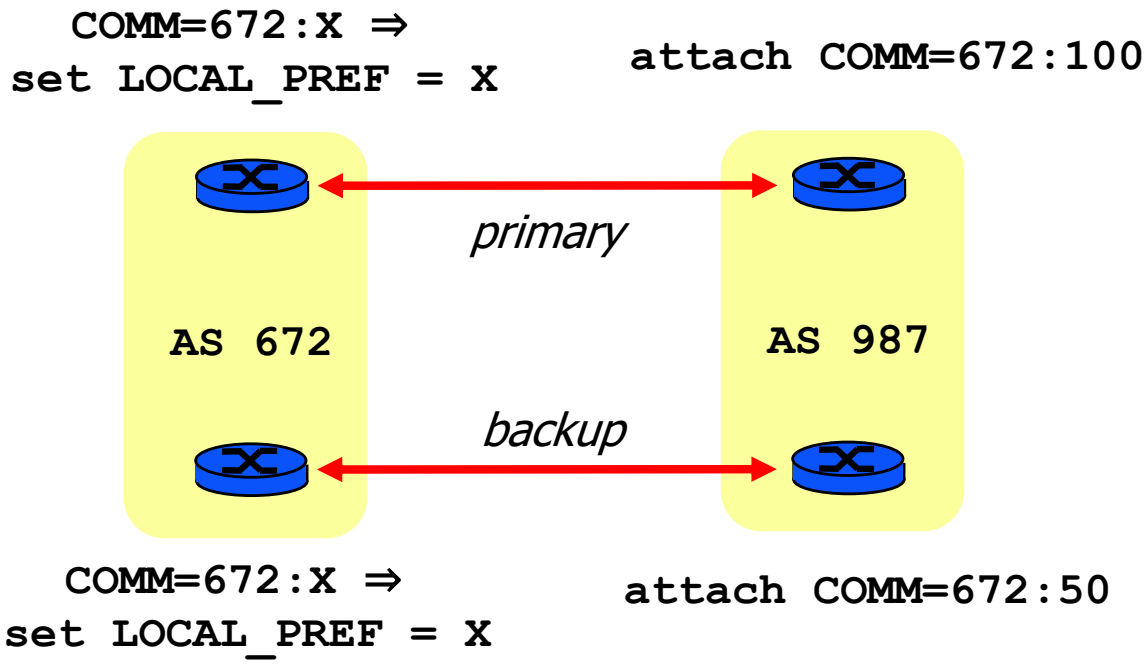


NO_EXPORT

- Q: What is the route followed by a packet sent to 2.2.48 received by R4 ?
- A: the packet is sent via R3 and R1

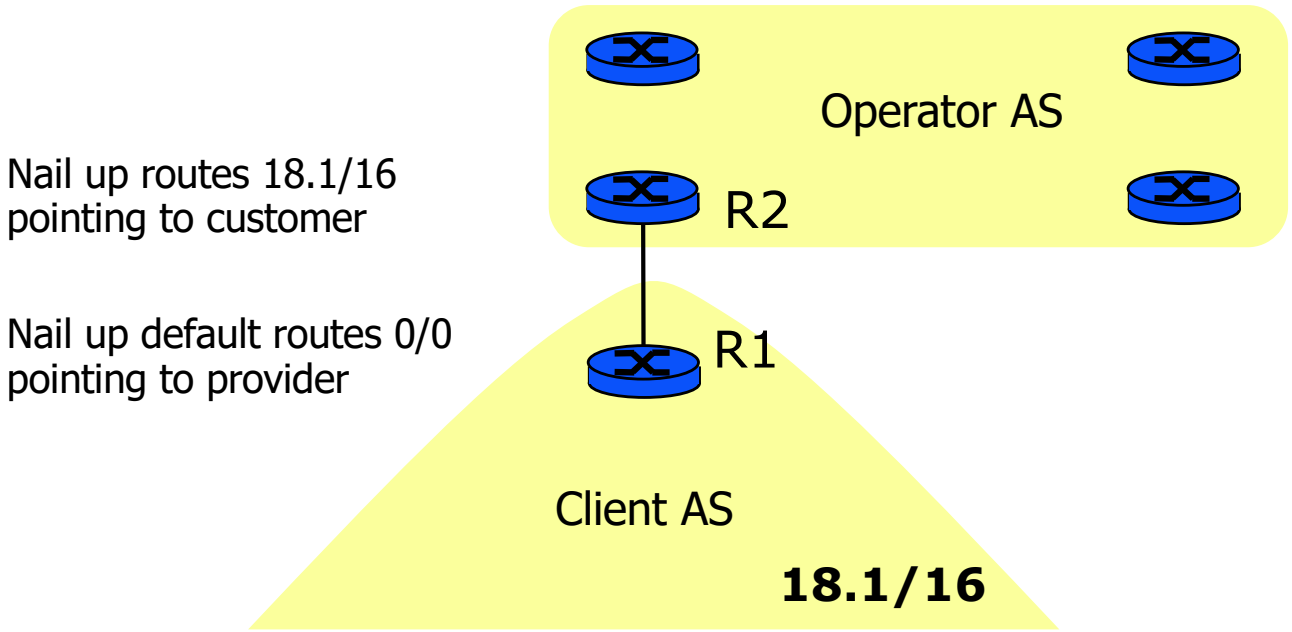


COMMUNITY



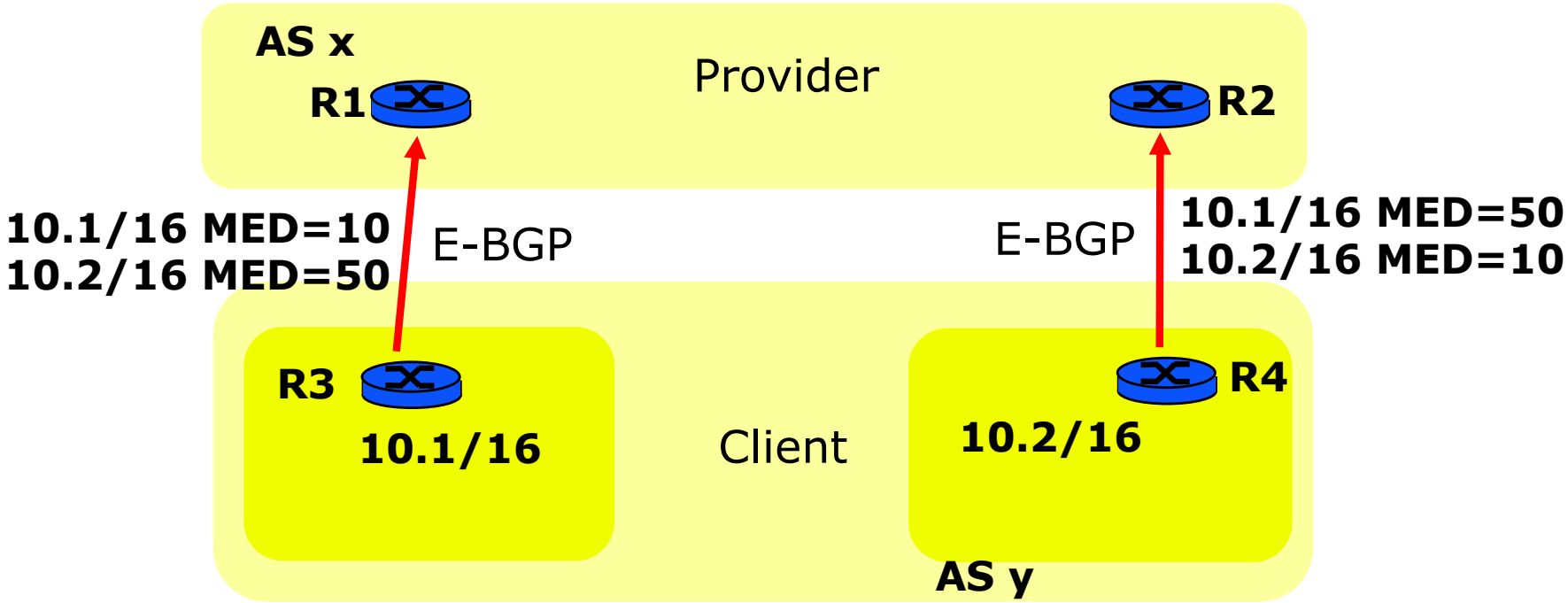
- Set LOCAL_PREF according to community values

Ex1: Stub AS



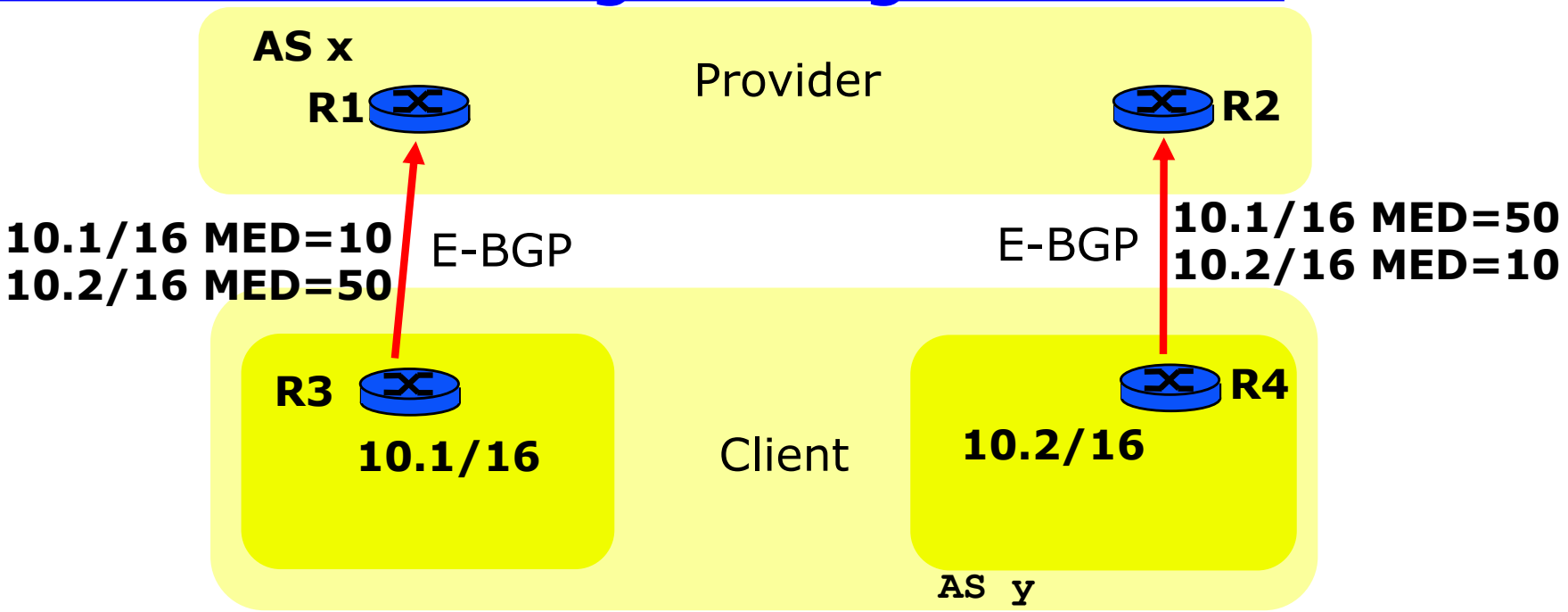
- BGP not needed between Client and Operator
- No AS number for client
- R2 learns all prefixes in Client by static configuration or IGP on link R1—R2
- Example: IMAG and CICG-GRENOBLE
- what if R1 fails ?

Ex2: Dual Homing to Single Provider



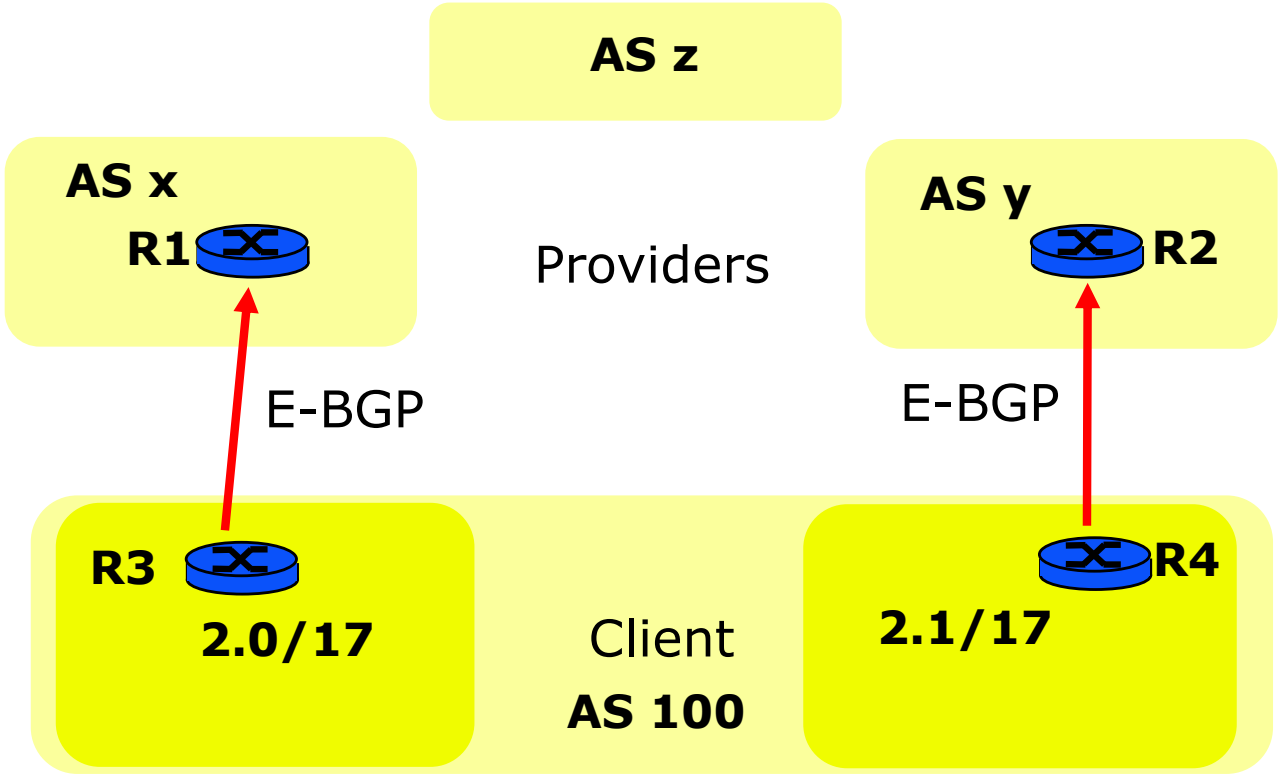
- With numbered Client AS
  - Use MED to share traffic from ISP to Client on two links
  - Use Client IGP configuration to share traffic from Client on two links
  - Q1: is it possible to avoid distributing BGP routes into Client IGP ?
  - Q2: is it possible to avoid assigning an AS number to Client ?
  - Q3: is it possible to avoid BGP between Client and Provider ?

Ex2: Dual Homing to Single Provider



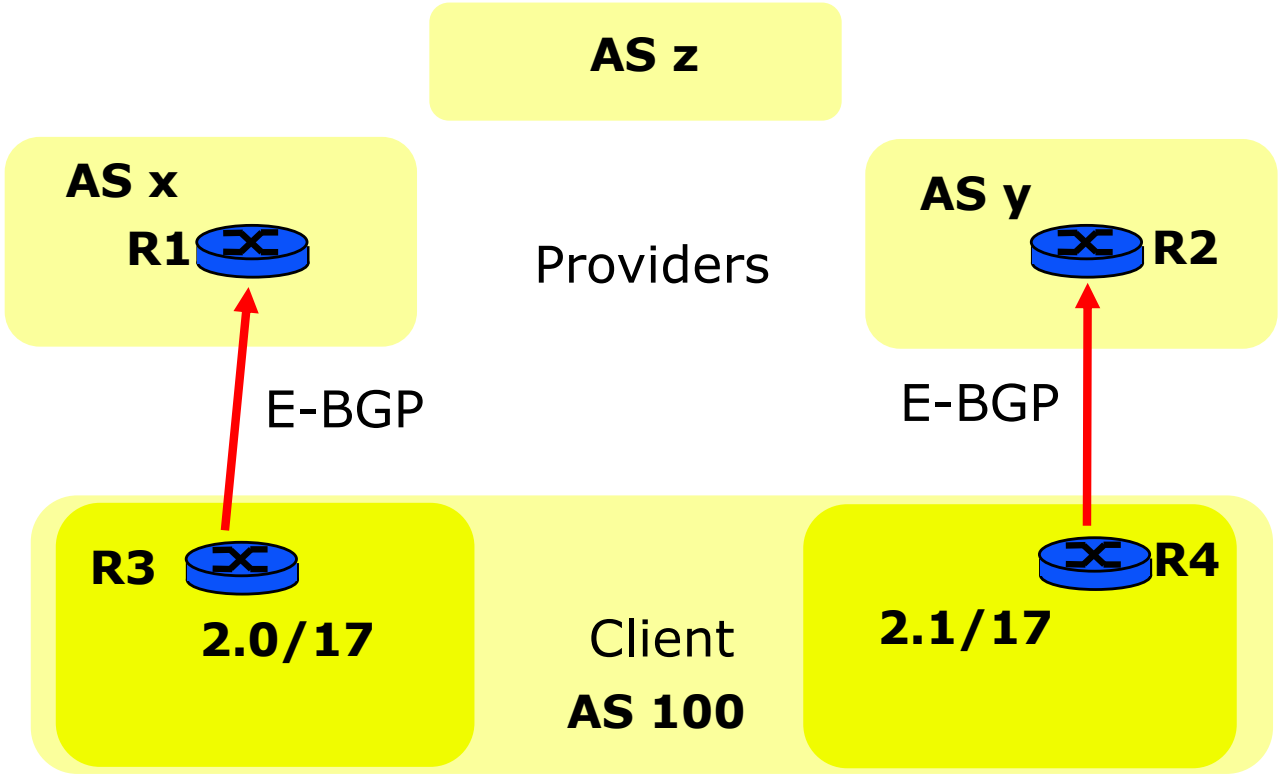
- Q1: is it possible to avoid distributing BGP routes into Client IGP ?
- A: yes, for example: configure R3 and R4 as default routers in Client AS; traffic from Client AS is forwarded to nearest of R3 and R4. If R3 or R4 fails, to the remaining one
- Q2: is it possible to avoid assigning an AS number to Client ?
- A: Yes, it is sufficient to assign to Client a private AS number: Provider translates this number to its own.
- Q3: is it possible to avoid BGP between Client and Provider ?
- A: Yes, by running a protocol like RIP between Client and Provider and redistributing Client routes into Provider IGP. Thus Provider pretends to the rest of the world that the prefixes of Client are its own.

Ex3: Dual Homing to Several Providers



- Client has its own address space and AS number
- Q: how can routes be announced between AS 100 and AS x? AS x and AS z?
- Q: assume Client wants most traffic to favor AS y How can that be done?

Ex3: Dual Homing to Several Providers



- Client has its own address space and AS number
- Q: how can routes be announced between AS 100 and AS x? AS x and AS z?
  A: R3 announces 2.0/17 and 2.0/16; traffic from AS x to 2.0/17 will flow via AS x; if R3 fails, it will use the longer prefix and flow via AS y.
  AS x announces 2.0/17 and 2.0/16 to AS z
- Q: assume Client wants most traffic to prefer AS y. How can that be done?
  A: R3 announces an artificially inflated path: 100 100 100 100 : 2.0/17. AS z will favour the path via AS y which has a shorter AS path length

Route filtering

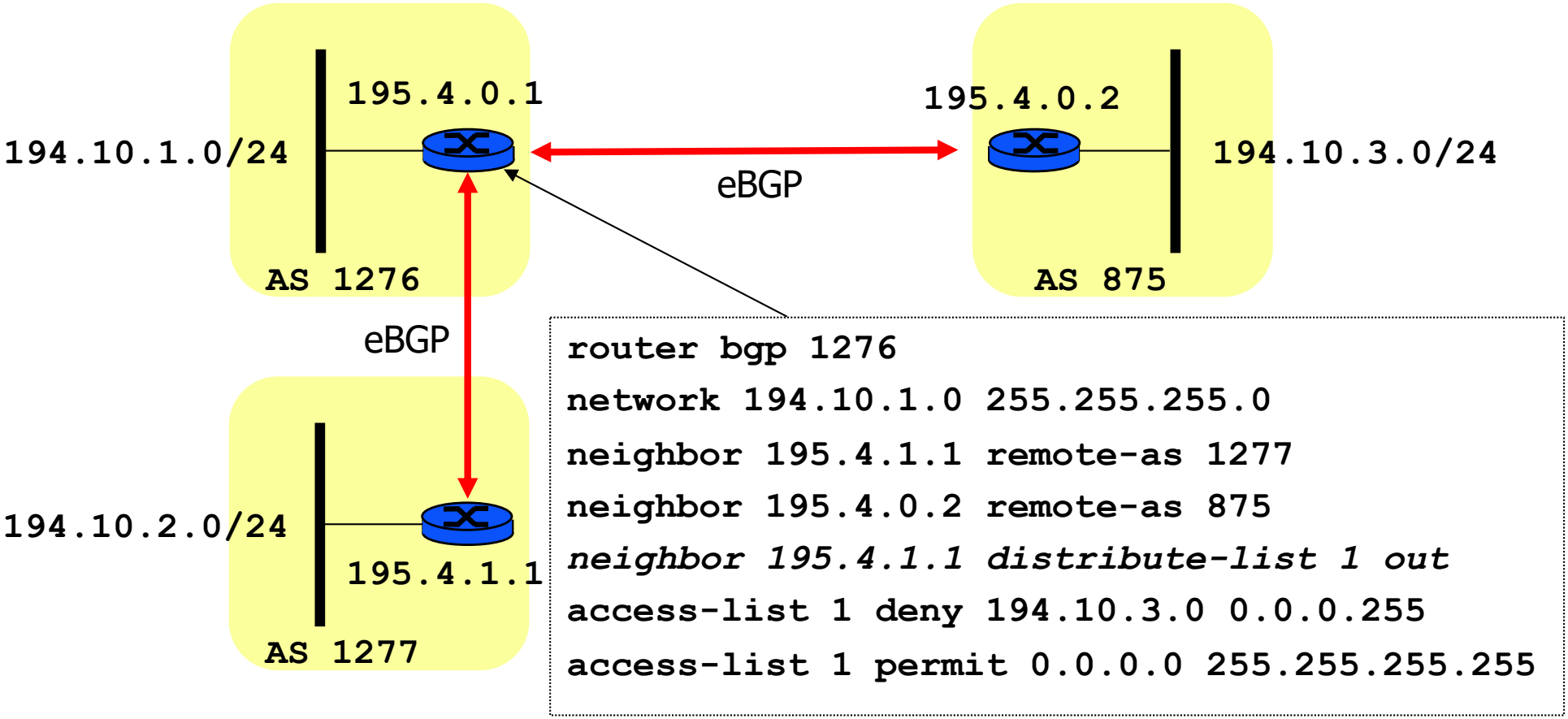
- Associate an access list with a neighbor

neighbor *ID* **distribute-list** *no-of-the-list* **[in/out]**

- Define an access list
  - non-significant-bits (inverse of the netmask)
  - if no action specified at the end of the list, apply "deny everything else"

access-list *No-of-the-list* **[deny/permit]**
IP-address non-significant-bits

Route filtering



- AS 1276 does not want to forward traffic to `194.10.3.0/24` of AS 875 - it does not re-advertise this prefix

Path filtering

- Associate a filter list with a neighbor

neighbor *ID* **filter-list** *no-of-the-list* **[in/out]**

- Define a filter list

ip as-path access-list *no-of-the-list* **[deny/permit]**
regular-expression

- Regular expressions

^	**beginning of the path**
$	**end of the path**
.	**any character**
?	**one character**
_	**matches ^ $ ( ) 'space'**
*****	**any number of characters (zero included)**
+	**any number of characters (at least one)**

Path filtering

- Examples

  `^$` - local routes only (empty AS_PATH)

  `.*` - all routes (all paths AS_PATH)

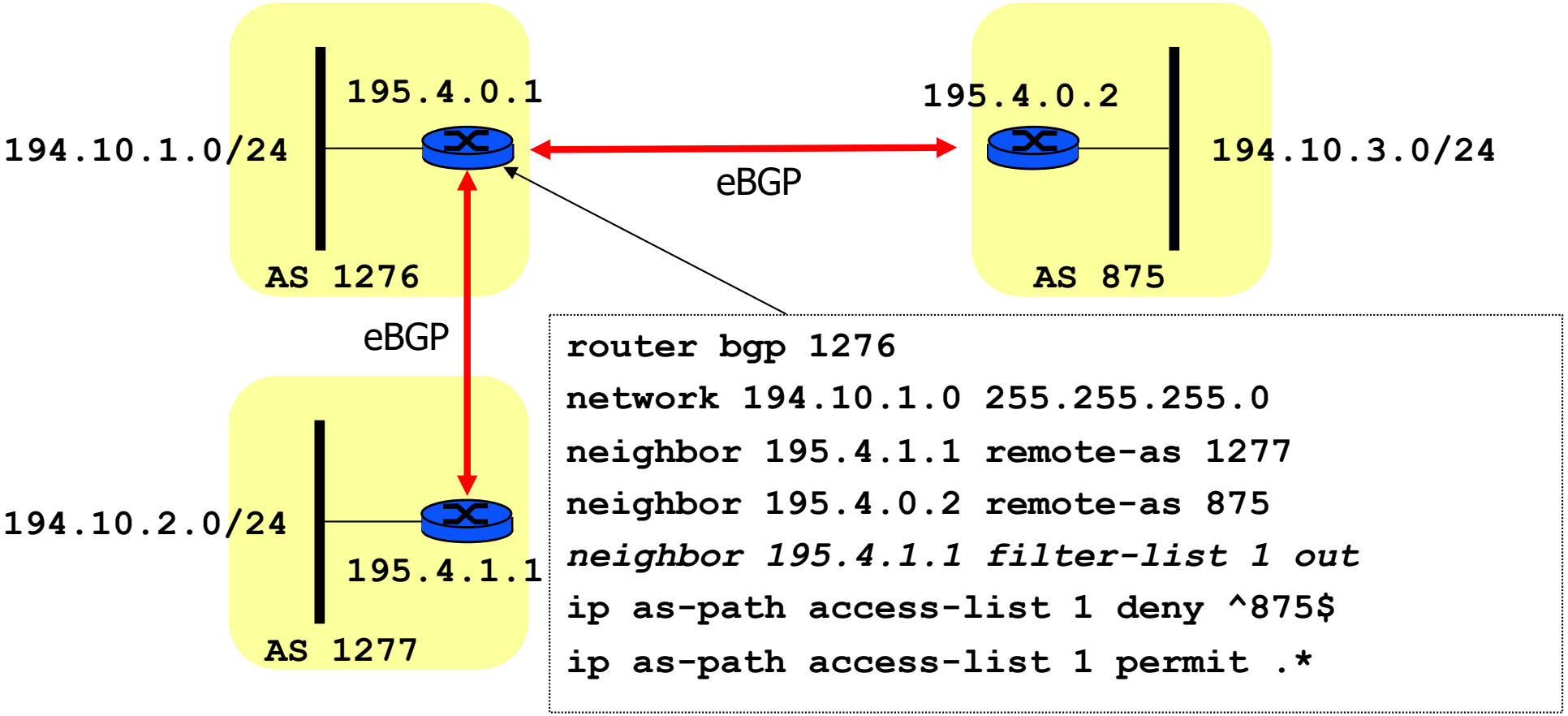
  `^300$` - AS_PATH = 300

  `^300_` - all routes coming from 300 (e.g. AS_PATH = 300 200 100)

  `_300$` - all routes originated at 300 (e.g. AS_PATH = 100 200 300)

  `_300_` - all routes passing via 300 (e.g. AS_PATH = 200 300 100)

Path filtering



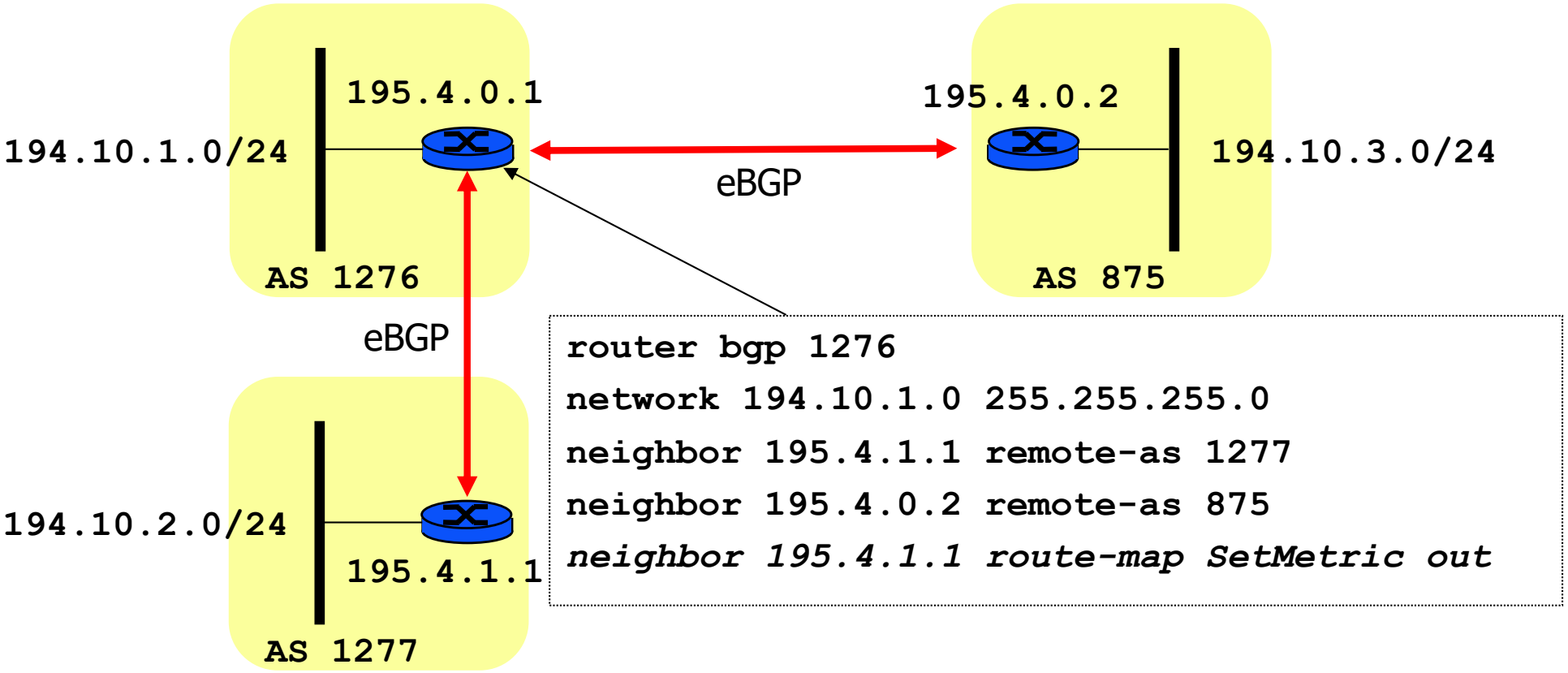
- AS 1276 does not want to forward traffic for all internal routes of AS 875

Route maps

```
route-map map-tag [permit|deny] instance-no
first-instance-conditions: set match
next-instance-conditions: set match
...
route-map SetMetric permit 10
match ip address 1
set metric 200

route-map SetMetric permit 20
set metric 300

access-list 1 permit 194.10.3.0 0.0.0.255
```

- Set metric 200 (MED) on route **194.10.3/24**

Route maps



- Set metric 200 on route **194.10.3/24**, 300 otherwise

Route maps

```
neighbor 192.68.5.2 route-map SetLocal in

route-map SetLocal permit 10
set local-preference 300
```

- Set LOCAL_PREF to 300

```
neighbor 172.16.2.2 route-map AddASnum out

route-map AddASnum permit 10
set as-path prepend 801 801
```

- Prepend AS 801 801 to AS_PATH (makes it longer)

Facebook DNS authoritative servers

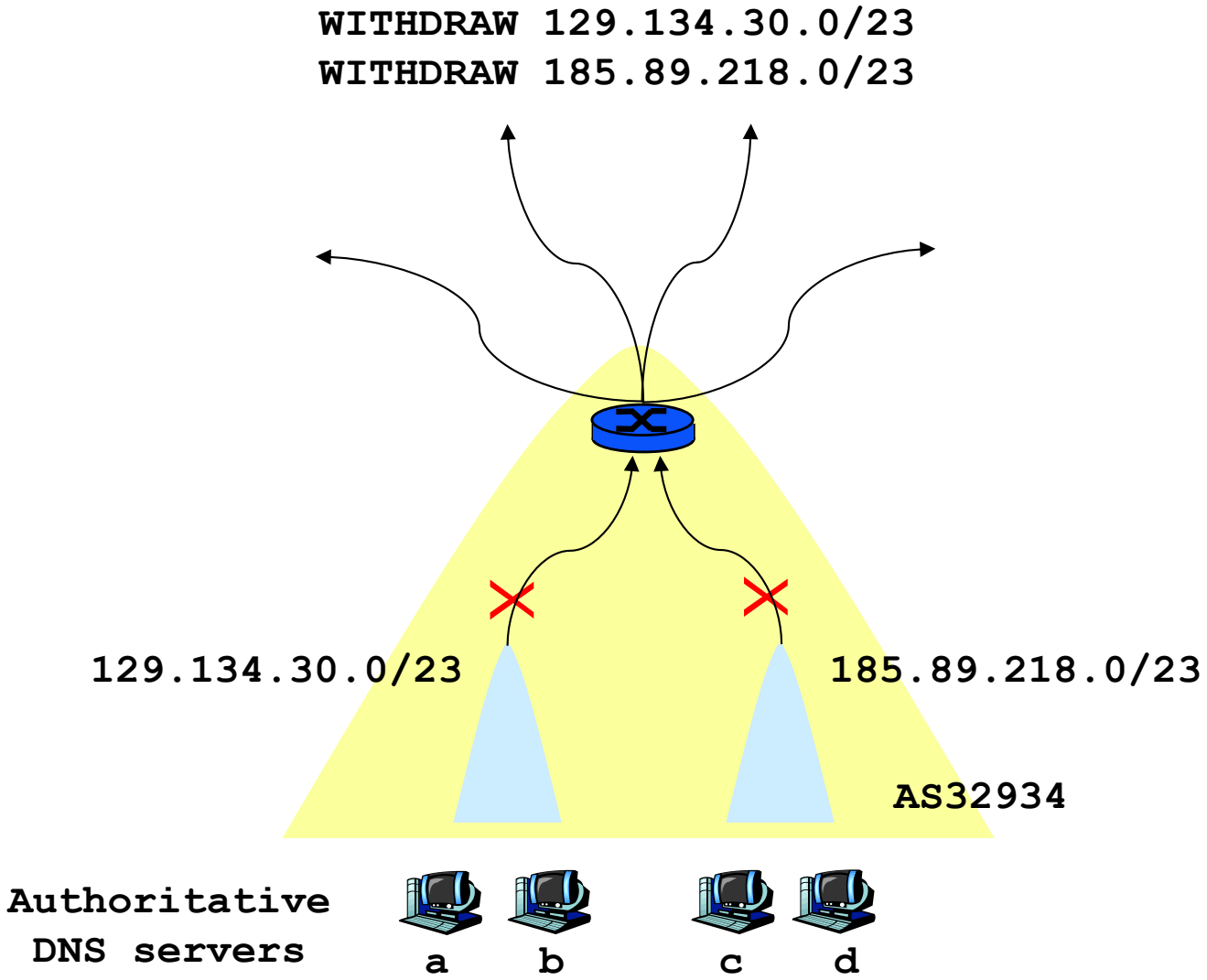
- `facebook.com. 1083 IN NS c.ns.facebook.com.`
- `facebook.com. 1083 IN NS b.ns.facebook.com.`
- `facebook.com. 1083 IN NS a.ns.facebook.com.`
- `facebook.com. 1083 IN NS d.ns.facebook.com.`

- `a.ns.facebook.com. 83708 IN A 129.134.30.12`
- `b.ns.facebook.com. 57194 IN A 129.134.31.12`
- `c.ns.facebook.com. 57881 IN A 185.89.218.12`
- `d.ns.facebook.com. 51247 IN A 185.89.219.12`

BGP Table at AS1299 - now

- show bgp ipv4 unicast 185.89.218.0/23
- BGP routing table entry for 185.89.218.0/23
- Paths: (26 available, best #25)
- Path #25:
- Received by speaker 0
- 32934
- 62.115.36.59 from 62.115.36.59 (129.134.59.179)
- Origin IGP, metric 0, localpref 200, valid, external, best, group-best, multipath, import-candidate
- Communities:
  - 1299:430 (RPKI state Valid)
  - 1299:1000 1299:35000 1299:35200

Disconnected Prefixes of DNS Servers



BGP Table at Cloudflare – Oct. 4, 2021

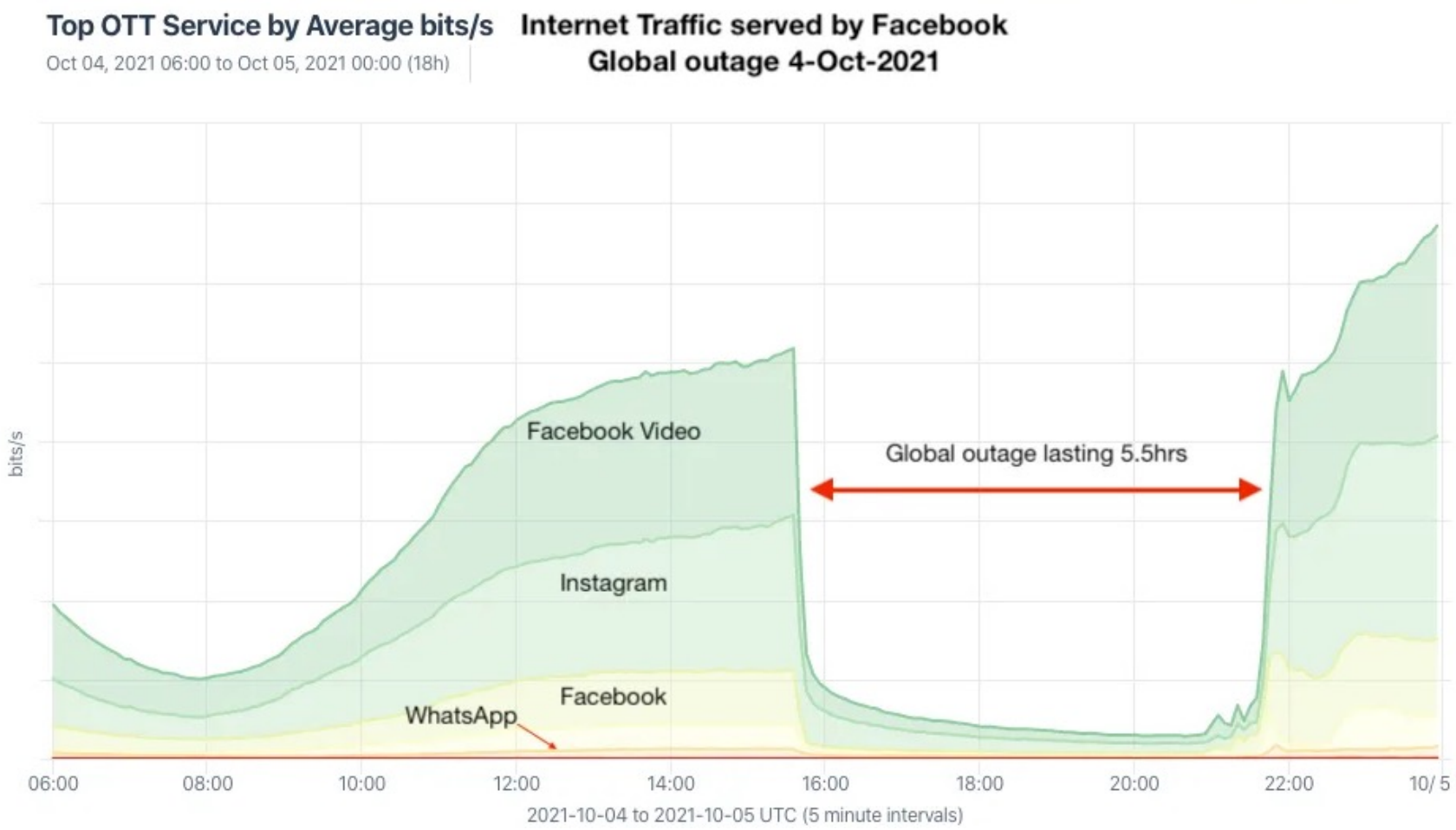
- **route-views>show ip bgp 185.89.218.0/23**
- **% Network not in table**

- **route-views>show ip bgp 129.134.30.0/23**
- **% Network not in table**

All DNS servers

```
user@host$ dig @1.1.1.1 facebook.com # CloudFlare
;; ->>HEADER<<- opcode: QUERY, status: SERVFAIL, id: 5153
;facebook.com.                  IN      A
user@host$ dig @8.8.8.8 facebook.com # Google Public DNS
;; ->>HEADER<<- opcode: QUERY, status: SERVFAIL, id: 43224
;facebook.com.                  IN      A
user@host$ dig @208.67.222.222 facebook.com # OpenDNS
;; ->>HEADER<<- opcode: QUERY, status: SERVFAIL, id: 7643
;facebook.com.                  IN      A
user@host$ dig @176.103.130.130 facebook.com # AdGuard
;; ->>HEADER<<- opcode: QUERY, status: SERVFAIL, id: 5434
;facebook.com.                  IN      A
```

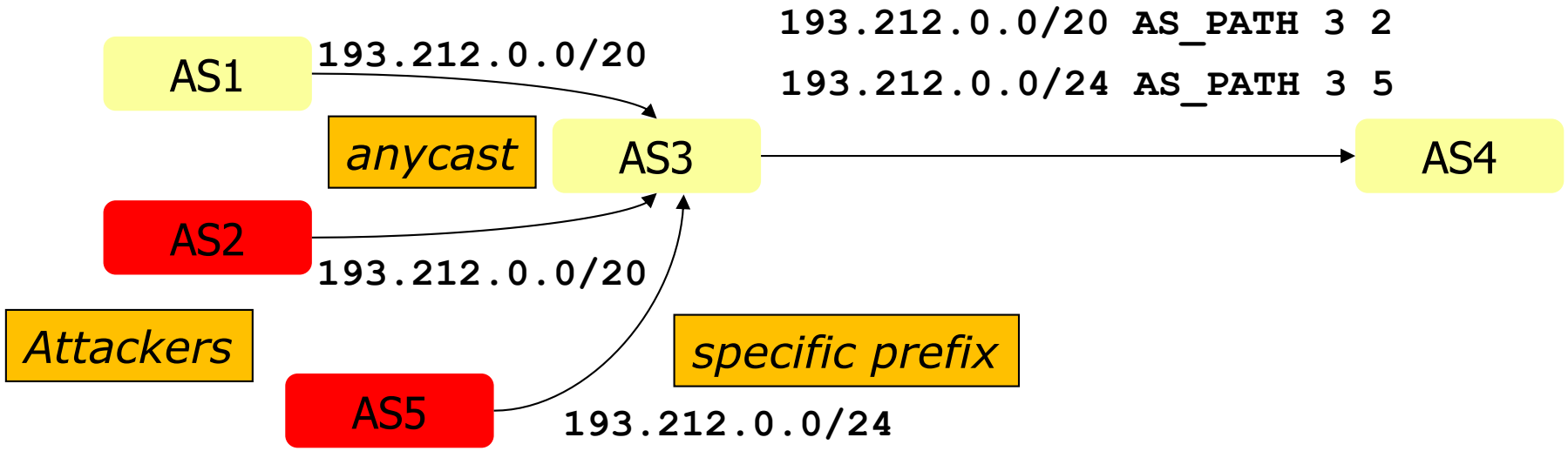
All traffic to Facebook



DNS servers at hypergiants

```
[stoege@puffy205 RD:0 $ for i in `dig +short facebook.com NS`; do aslo $i; done; echo
129.134.30.12     AS32934 | US | FACEBOOK
185.89.218.12     AS32934 | US | FACEBOOK
185.89.219.12     AS32934 | US | FACEBOOK
129.134.31.12     AS32934 | US | FACEBOOK

[stoege@puffy205 RD:0 $ for i in `dig +short google.com NS`; do aslo $i; done; echo
216.239.38.10     AS15169 | US | GOOGLE
216.239.36.10     AS15169 | US | GOOGLE
216.239.32.10     AS15169 | US | GOOGLE
216.239.34.10     AS15169 | US | GOOGLE

[stoege@puffy205 RD:0 $ for i in `dig +short cloudflare.net NS`; do aslo $i; done; echo
198.41.222.131    AS13335 | US | CLOUDFLARENET
198.41.222.31     AS13335 | US | CLOUDFLARENET
198.41.223.131    AS13335 | US | CLOUDFLARENET
198.41.223.31     AS13335 | US | CLOUDFLARENET
173.245.59.31     AS13335 | US | CLOUDFLARENET

[stoege@puffy205 RD:0 $ for i in `dig +short microsoft.com NS`; do aslo $i; done; echo
13.107.160.205    AS8068 | US | MICROSOFT-CORP-MSN-AS-BLOCK
40.90.4.205       AS8068 | US | MICROSOFT-CORP-MSN-AS-BLOCK
64.4.48.205       AS8068 | US | MICROSOFT-CORP-MSN-AS-BLOCK
13.107.24.205     AS8068 | US | MICROSOFT-CORP-MSN-AS-BLOCK

[stoege@puffy205 RD:0 $ for i in `dig +short twitter.com NS`; do aslo $i; done; echo
205.251.192.179   AS16509 | US | AMAZON-02
208.78.70.34      AS33517 | US | DYNDNS
204.13.250.34     AS33517 | US | DYNDNS
205.251.194.151   AS16509 | US | AMAZON-02
208.78.70.34      AS33517 | US | DYNDNS
204.13.251.34     AS33517 | US | DYNDNS
208.78.71.34      AS33517 | US | DYNDNS
204.13.250.34     AS33517 | US | DYNDNS
205.251.199.195   AS16509 | US | AMAZON-02
205.251.196.198   AS16509 | US | AMAZON-02

stoege@puffy205 RD:0 $
```

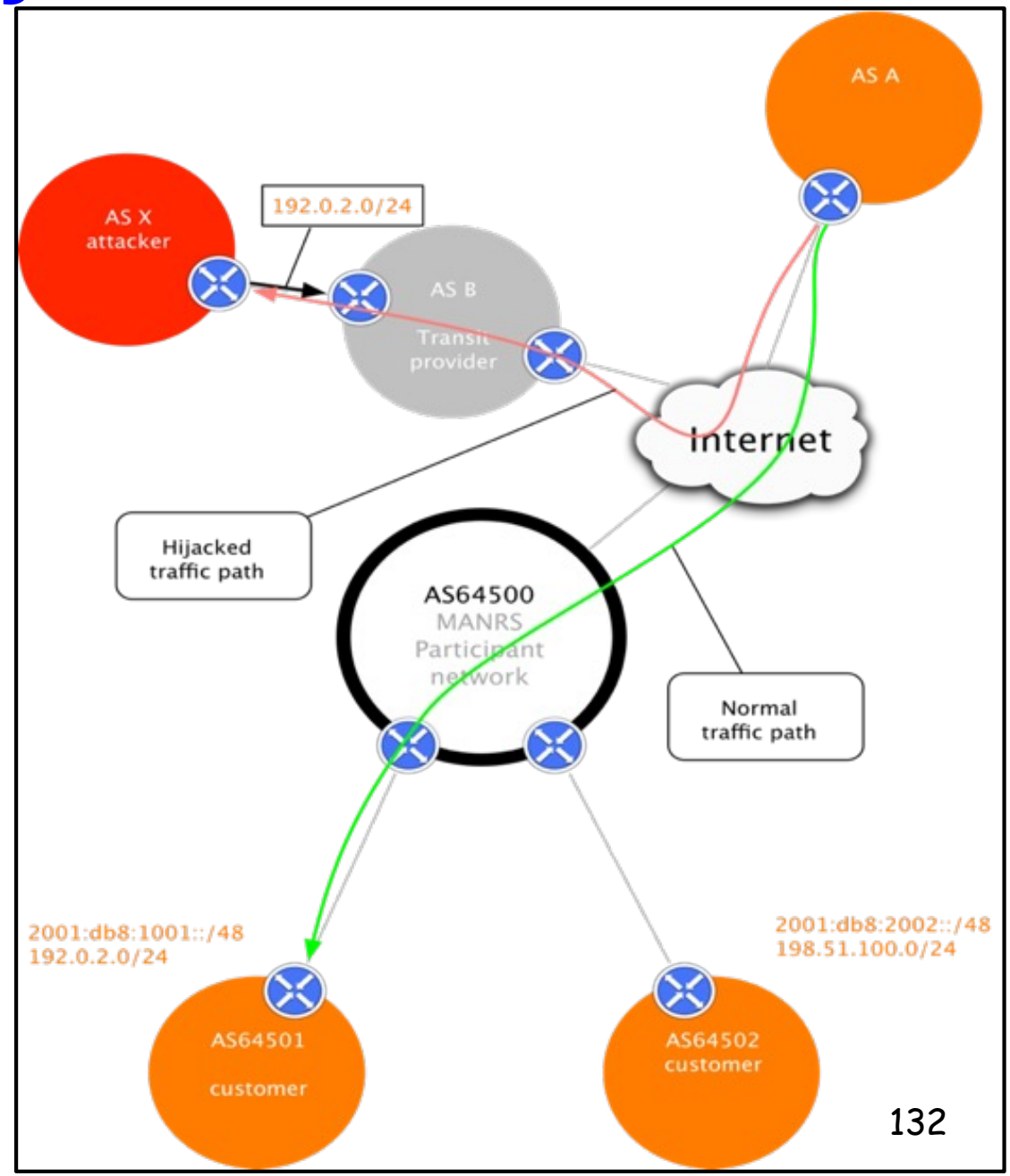
Security – prefix hijacking



- Proper origination of a prefix into BGP
  - By the AS who owns the prefix
  - By upstream ASes
- Prefix hijacking
  - another AS originates the prefix
  - BGP does not verify that the AS is authorized

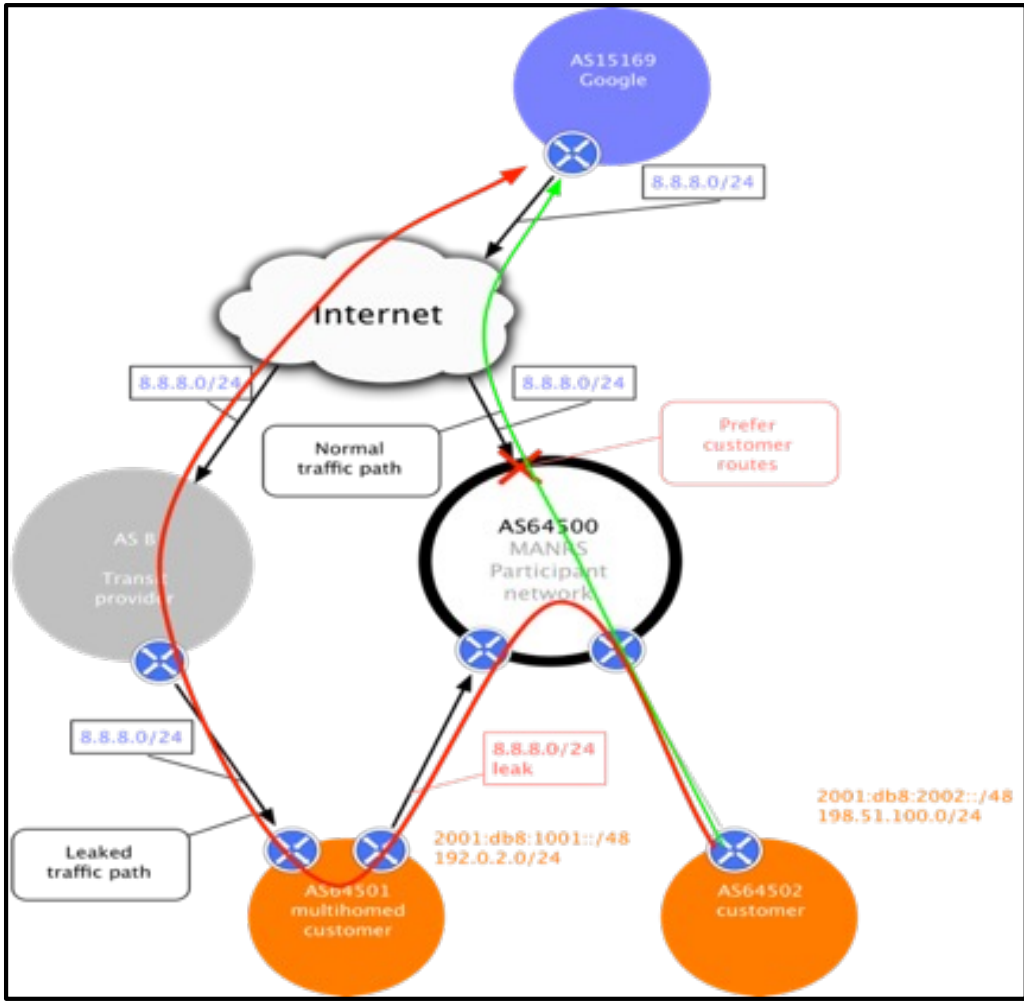
Prefix/Route Hijack

- **Route hijacking,** also known as "BGP hijacking" when a network operator or attacker (accidentally or deliberately) impersonates another network operator or pretending that a server or network is their client

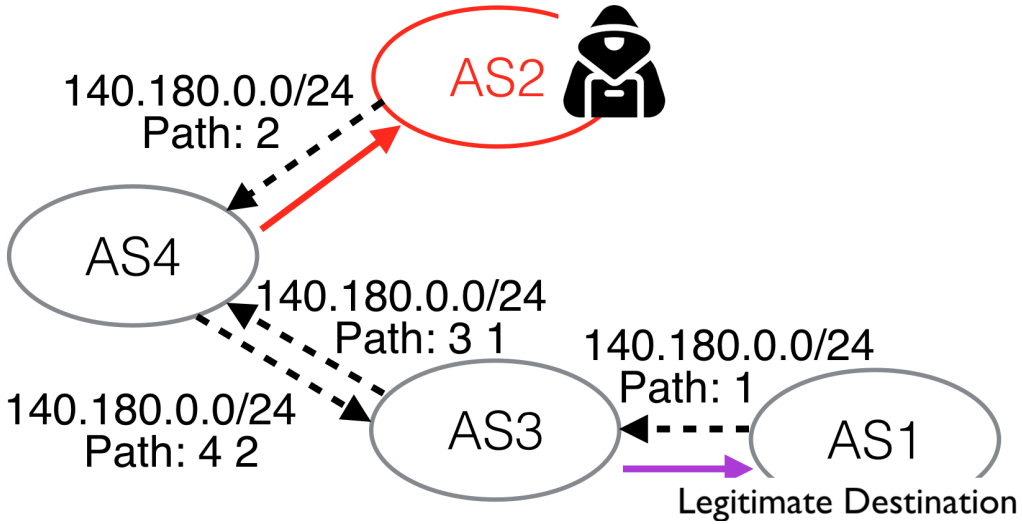


Route Leak

- **A route leak**
  - a network operator with multiple upstream providers accidentally announces to one of its upstream providers that is has a route to a destination through the other upstream provider.
- Makes the network an intermediary network between the two upstream providers. With one sending traffic now through it to get to the other.

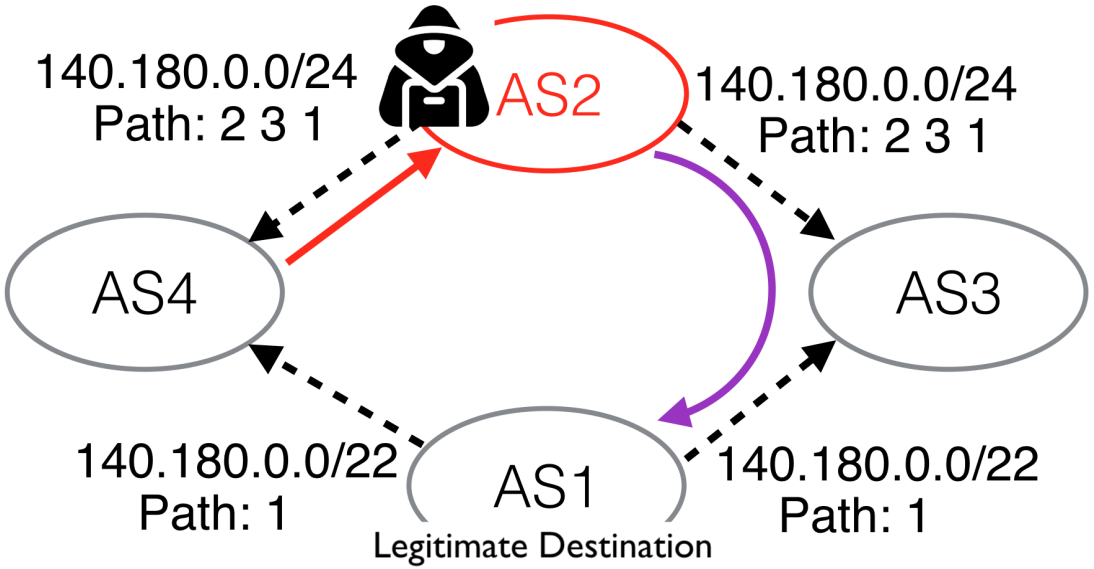


Targeting selective traffic



- AS2 (attacker) announces an equally-specific prefix as AS1 (legitimate destination)
- AS4 prefers the path to AS2, while AS3 prefers the path to AS1
- Only AS4 is affected by the attack

AS Path Poisoning Attack

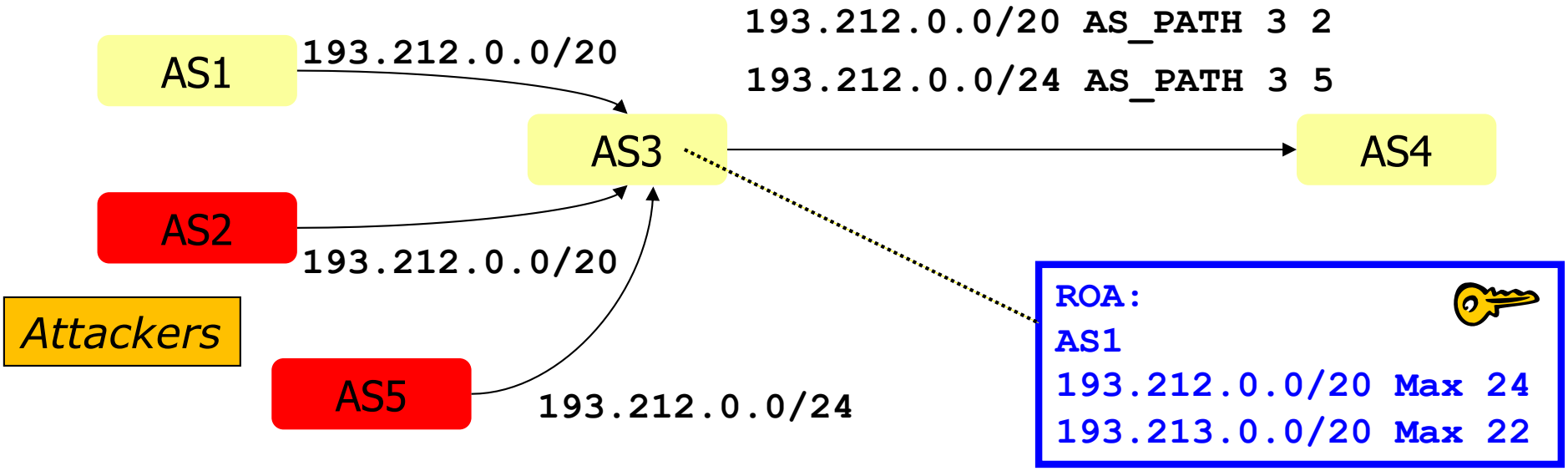


- AS2 (attacker) "poisons" the path by appending AS3 and AS1 (legitimate destination) in the path, which preserves a legitimate route from AS2 to AS1.

Pakistan Youtube incident

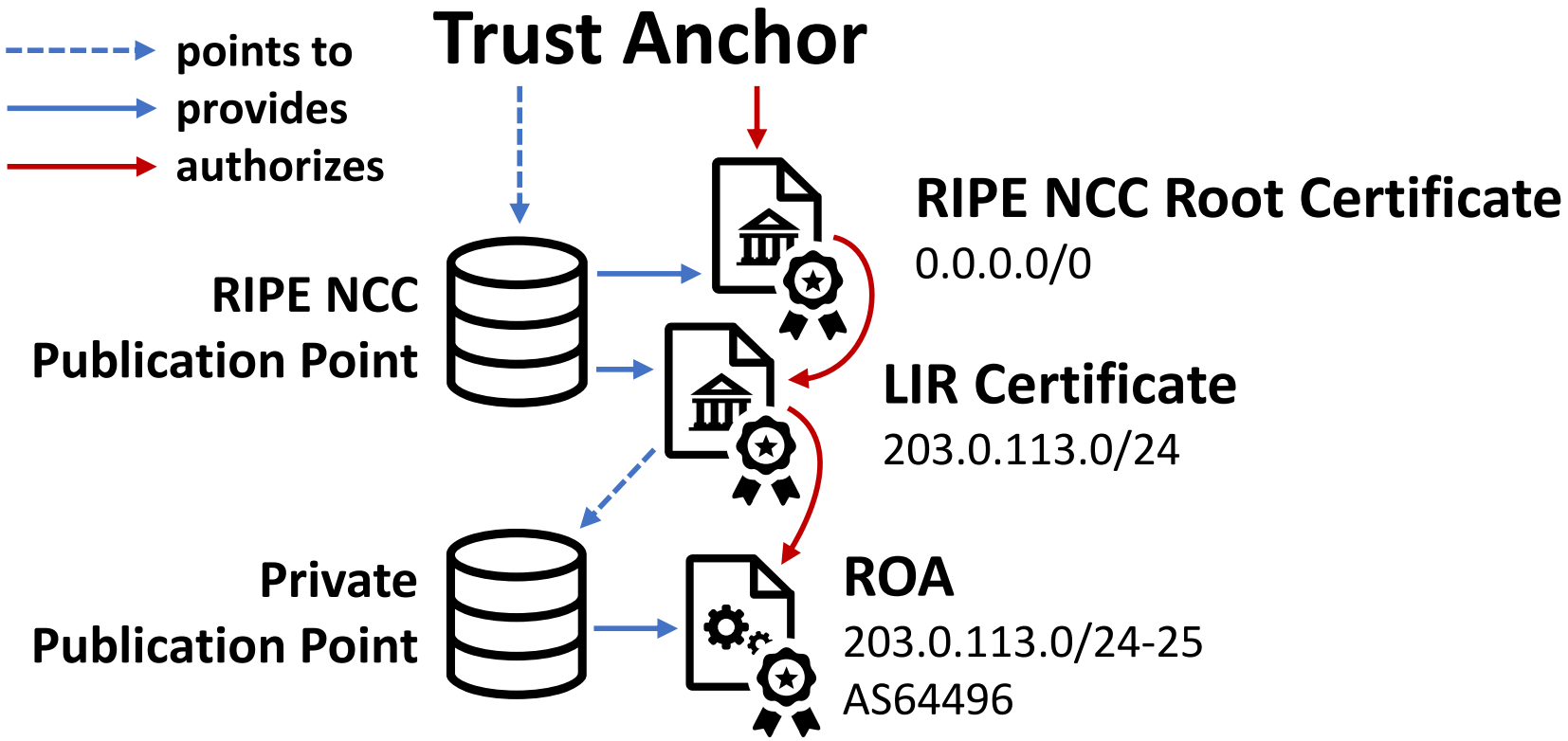
- Youtube's has prefix 208.65.152.0/22
- Pakistan's government order Youtube blocked
- Pakistan Telecom (AS 17557) announces 208.65.153.0/24 in the wrong direction (outwards!)
- Longest prefix match caused worldwide outage

ROA - Route Origin Authorizations



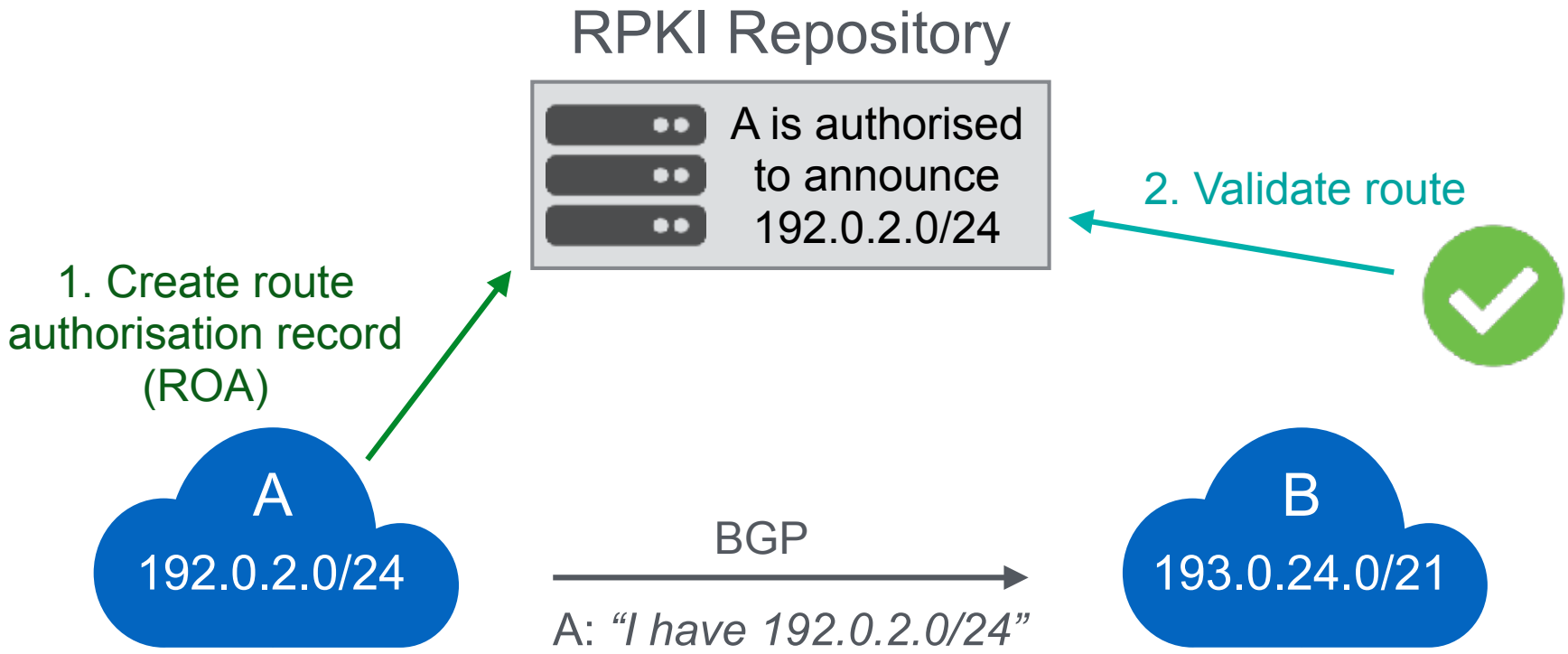
- ROA - file signed with a private key by resource holder
  - AS at the origin of one or several prefixes
  - Prefix and maximum prefix length accepted for this ROA
- RPKI – Resource Public Key Infrastructure
  - X.509 certificates, private key used for the signature
  - chain of trust: IANA->RIPE(RIR)->LIR

RPKI



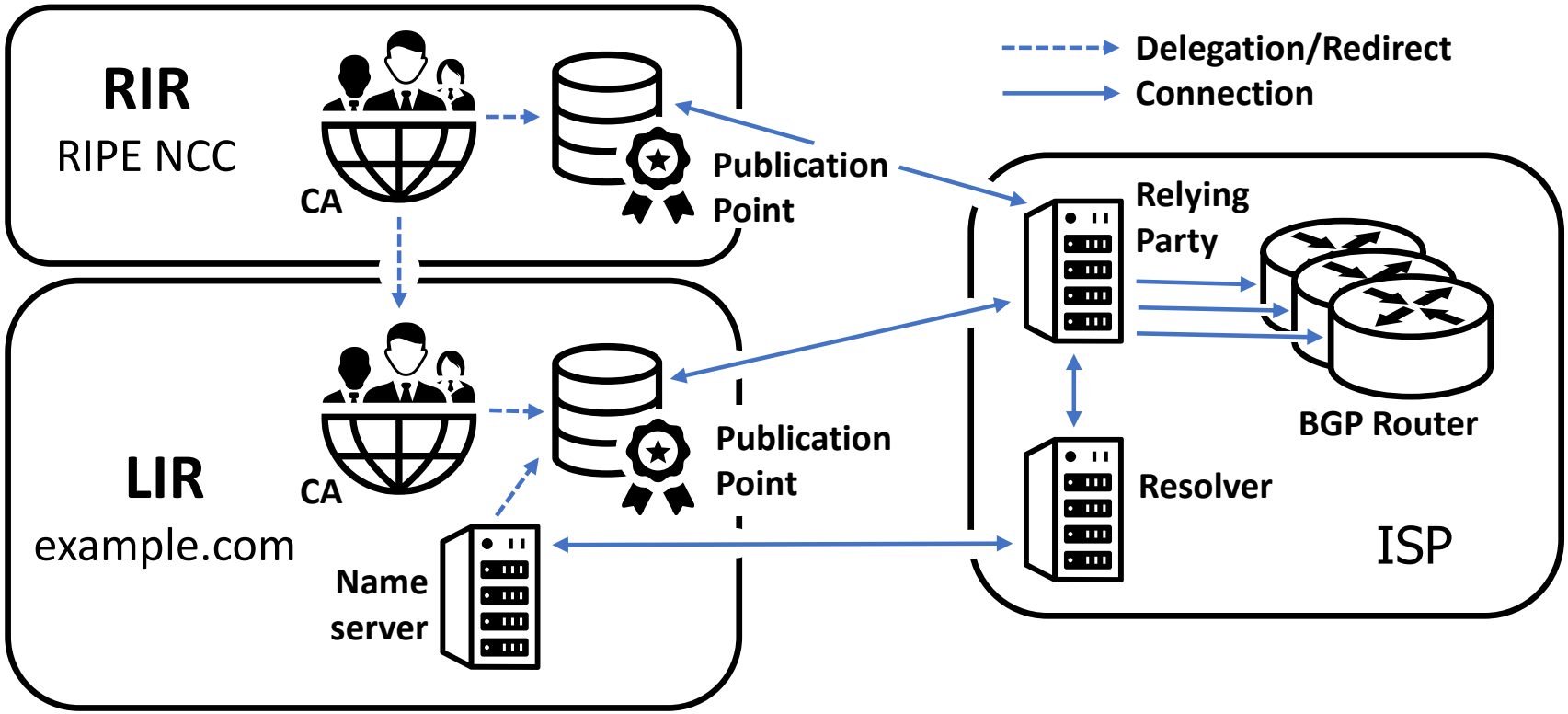
- Like server certificates, another CAs (Certificate Authorities)
- LIR – Local Internet Registry

Validators - Relying Party



- Downloads the RPKI repository from the RIRs
- Validates the chain of trust of all ROAs and associated CAs
- Talks to your routers using RTR Protocol

RPKI



- Relying parties (Validators) apply ROV to filter routes
- Routers use RTR (RPKI to Router Protocol) to download the current result set from the Relying Party

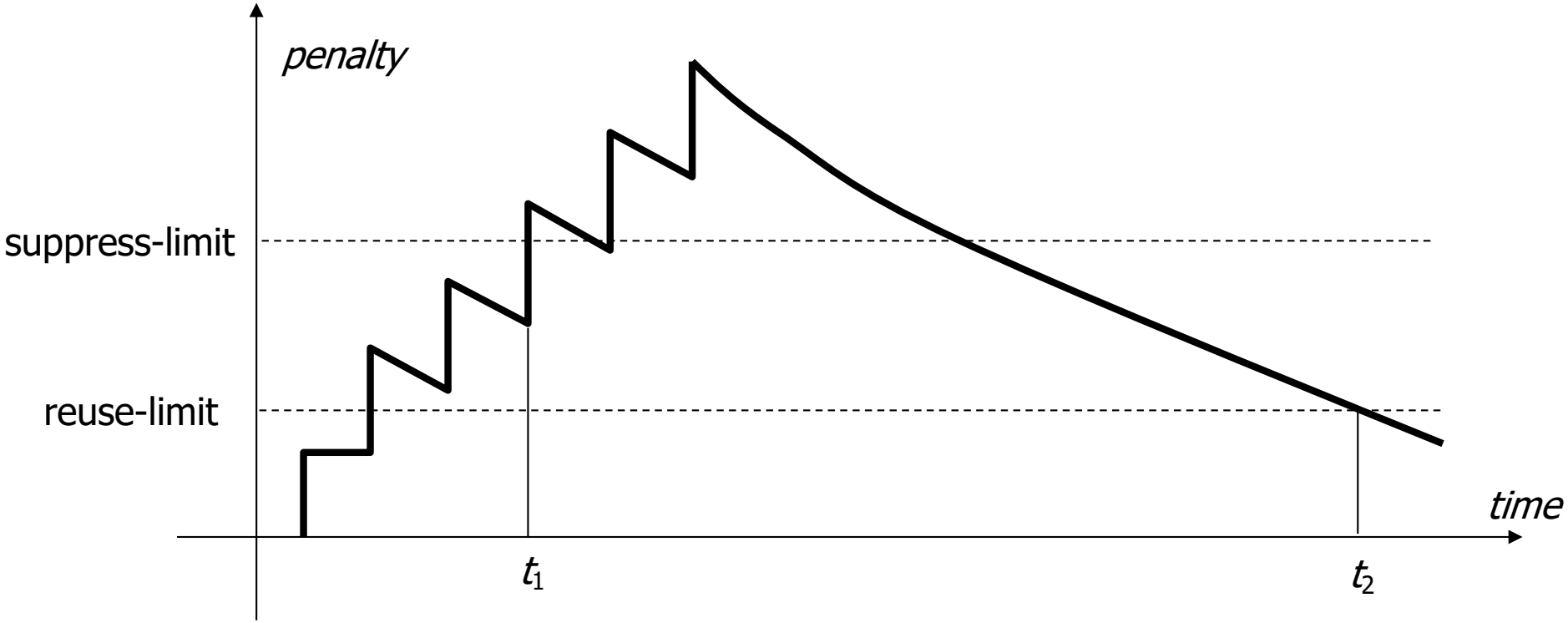
RPKI

- Protects unauthorized origination attacks (errors)
  - Stops ISPs to announce routes with a direct AS path to the upstream
- What it does not stop today
  - AS path poisoning attack (an attacker just needs to prepend an AS path to a valid origin AS in order to divert traffic)
- Danger of accidentally rejecting valid routes
  - ROV-enforcing ASes may discard legitimate BGP route-advertisements, thus disconnecting from thousands of legitimate destinations

Route dampening

- Route modification propagates everywhere
  - successive UPDATE and WITHDRAW of a route
- Sometimes routes are *flapping*
  - successive UPDATE and WITHDRAW
  - caused for example by BGP speaker that often crashes and reboots
- Solution:
  - decision process eliminates flapping routes
- How
  - withdrawn routes are kept in Adj-RIN-in
  - if comes up again soon (ie : flap), route receives a penalty
  - penalty fades out exponentially (halved at each half-life-time)
  - used to suppress or restore routes
- Thresholds: suppress-limit, reuse-limit

Route dampening



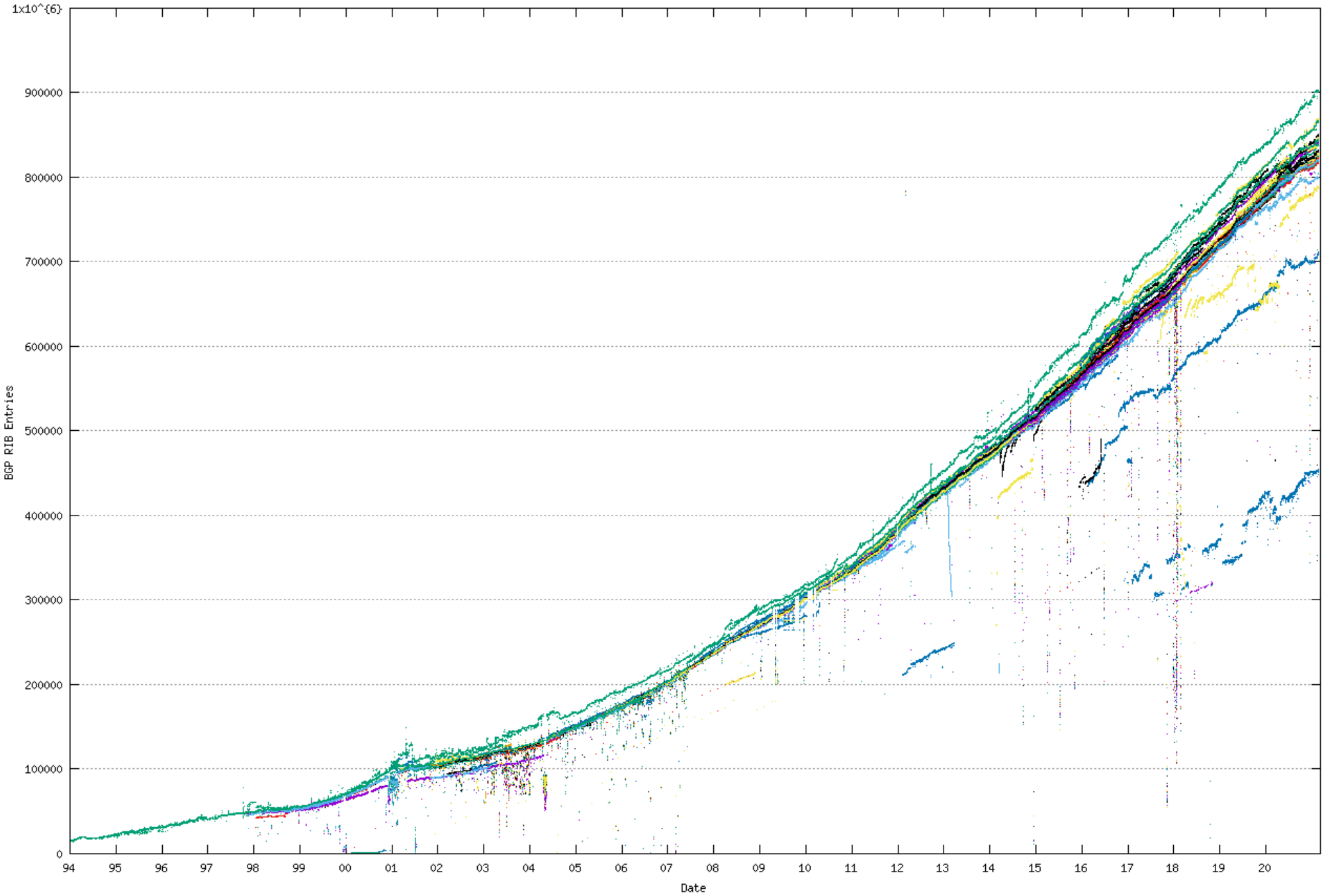
- Route suppressed at t_1, restored at t_2

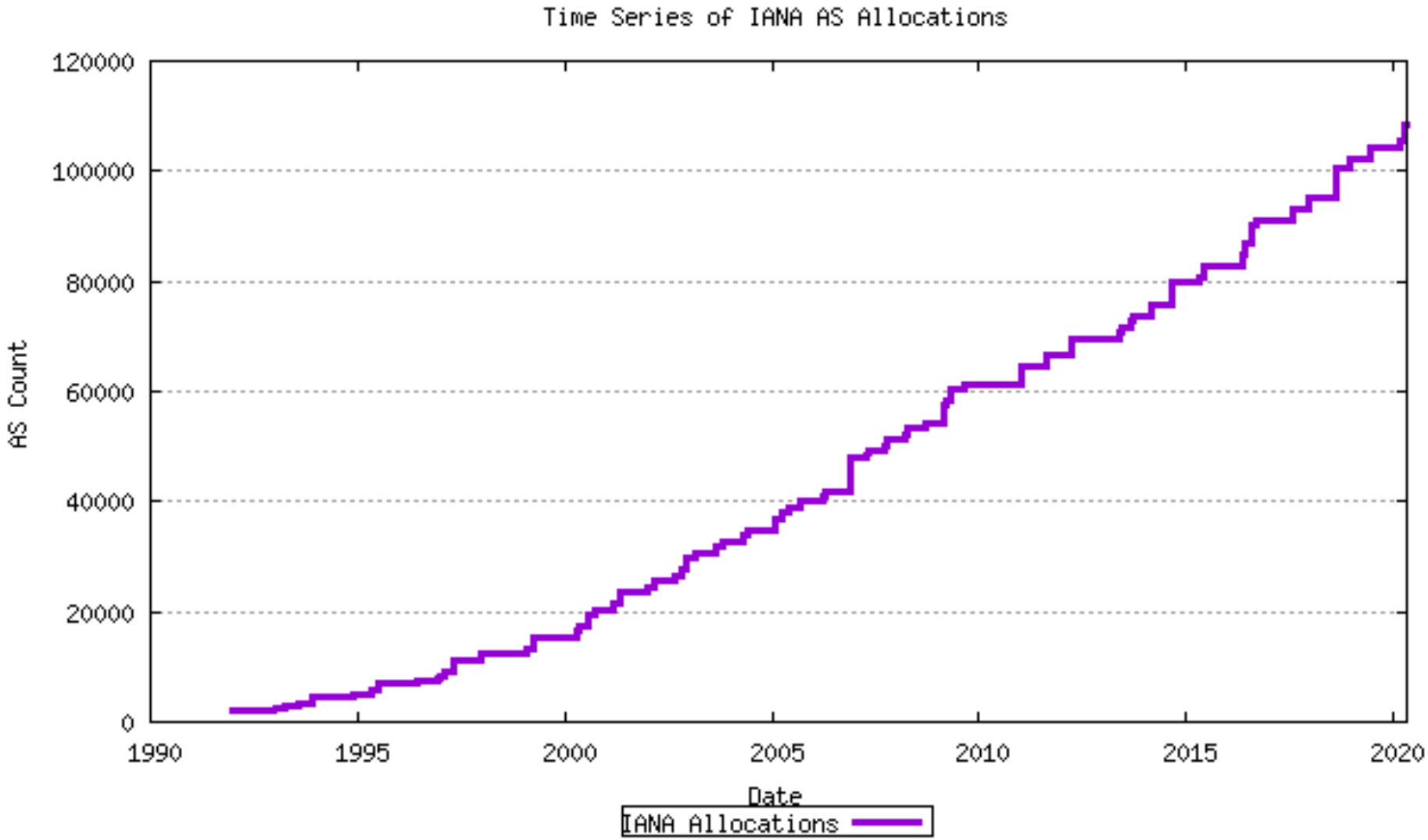
Some statistics

- Number of routes
  - 1988-1994: exponential increase
  - 1994-1995: CIDR
  - 1995-1998: linear increase (10000/year)
  - 1999-2000: return to exponential increase (42% per year)
  - since 2001: return to linear increase, ~120,000
- Number of ASs
  - 51% per year for 4 last years
  - 14000 AS effectively used
- Number of IP addresses
  - 162,128,493 (Jul 2002)
  - 7% per year

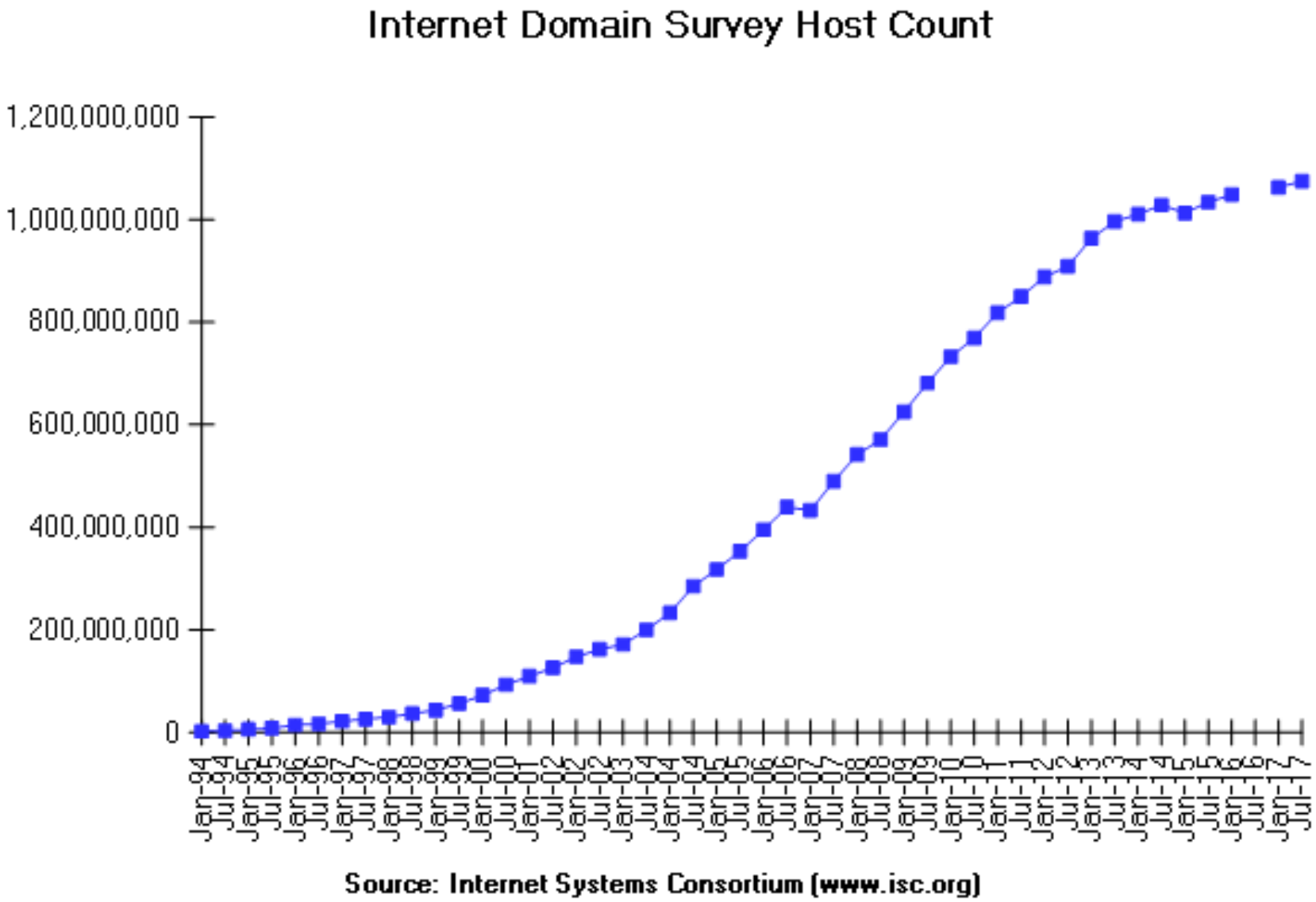
Default-free Zone

- Internet routers which have explicit routing information about the rest of the Internet


1x10^{6}
900000
800000
700000
600000
500000
400000
300000
200000
100000
0
BGP RIB Entries
94
95
96
97
98
99
00
01
02
03
04
05
06
07
08
09
10
11
12
13
14
15
16
17
18
19
20
Date


Time Series of IANA AS Allocations
AS Count
120000
100000
80000
60000
40000
20000
0
1990
1995
2000
2005
2010
2015
2020
Date
IANA Allocations

Number of hosts



BGP statistics

BGP routing table entries examined:	117013
Total ASes present in the Internet Routing Table:	14042
Origin-only ASes present in the Internet Routing Table:	12159
Transit ASes present in the Internet Routing Table:	1883
Transit-only ASes present in the Internet Routing Table:	63
Average AS path length visible in the Internet Routing Table:	5.3
Max AS path length visible:	23
Number of addresses announced to Internet:	1182831464
Equivalent to 70 /8s, 128 /16s and 147 /24s	
Percentage of available address space announced:	31.9
Percentage of allocated address space announced:	58.5

Prefix length distribution

/1:0 /2:0 /3:0 /4:0 /5:0 /6:0
/7:0 /8:17 /9:5 /10:8 /11:12 /12:46
/13:90 /14:239 /15:430 /16:7308 /17:1529 /18:2726
/19:7895 /20:7524 /21:5361 /22:8216 /23:9925 /24:64838
/25:185 /26:221 /27:126 /28:105 /29:85 /30:93
/31:0 /32:29

AS 559 - SWITCH

AS559 SWITCH-AS SWITCH Teleinformatics Services

Adjacency: 3 Upstream: 2 Downstream: 1

Upstream Adjacent AS list

AS1299 TCN-AS Telia Corporate Network

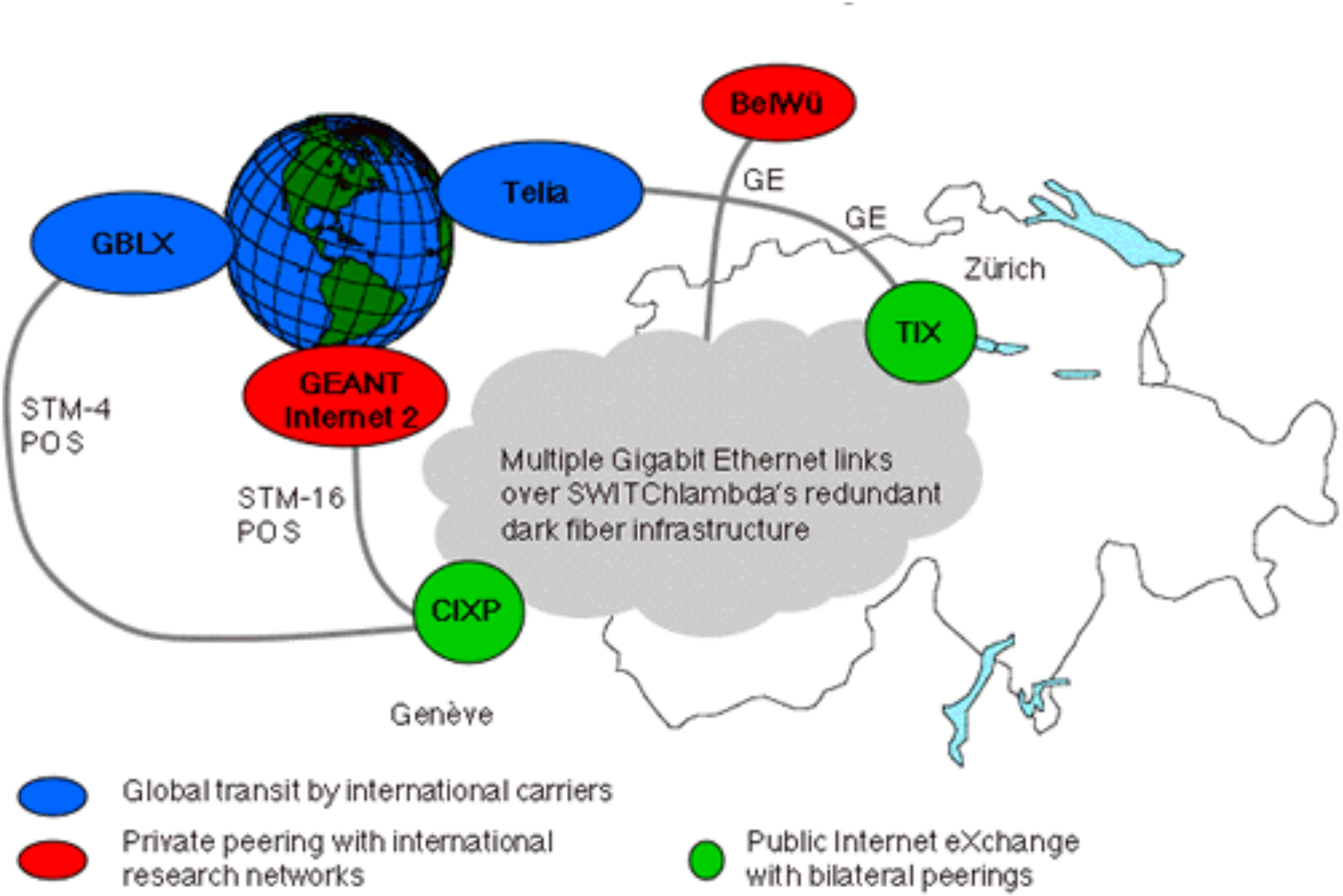
AS3549 GBLX Global Crossing

Downstream Adjacent AS list

AS4128 RG-SPARE RGnet, Inc.

Prefix	(AS Path)
128.178.0.0/15	1 3549 559
129.129.0.0/16	1 3549 559
129.132.0.0/16	1 3549 559

Switch


BelWü
GE
GE
Telia
GBLX
Zürich
TIX
GEANT
Internet 2
STM-4
POS
STM-16
POS
Multiple Gigabit Ethernet links
over SWITCHlambda's redundant
dark fiber infrastructure
CIXP
Genève
Global transit by international carriers
Private peering with international
research networks
Public Internet eXchange
with bilateral peerings

AS 1942 - CICG-GRENOBLE

AS1942 AS1942 FR-CICG-GRENOBLE

Adjacency: 1 Upstream: 1 Downstream: 0

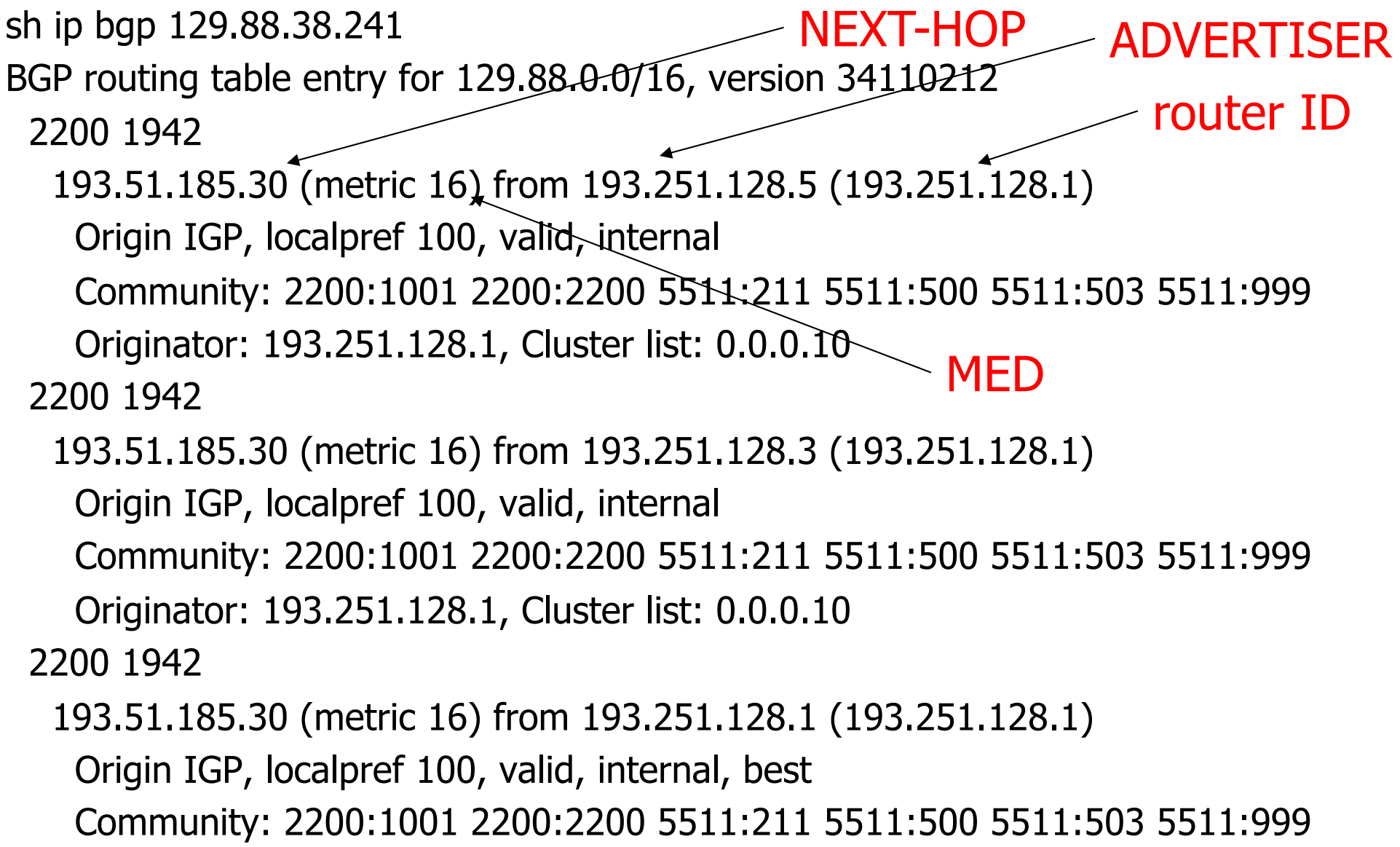
Upstream Adjacent AS list

AS2200 AS2200 RENATER 2

Prefix	(AS Path)
129.88.0.0/16	1239 5511 2200 1942
130.190.0.0/16	1239 5511 2200 1942
147.171.0.0/16	1239 5511 2200 1942
147.173.0.0/16	1239 5511 2200 1942

2200 - Renater-2, 5511 - OpenTransit (FT), 1239 - Sprint

Looking glass at genbb1.opentransit.net



From genbb1.opentransit.net

Tracing the route to horus.imag.fr (129.88.38.1)

1 P8-0-0.GENAR1.Geneva.opentransit.net (193.251.242.130) 0 msec 0 msec 0 msec
2 P6-0-0.GENAR2.Geneva.opentransit.net (193.251.150.30) 0 msec 4 msec 0 msec
3 P4-3.BAGBB1.Bagnolet.opentransit.net (193.251.154.97) 8 msec 8 msec 8 msec
4 193.51.185.30 [AS 2200] 16 msec 16 msec 16 msec
5 grenoble-pos1-0.cssi.renater.fr (193.51.179.238) [AS 2200] 16 msec 20 msec 16 ms
6 tigre-grenoble.cssi.renater.fr (195.220.98.58) [AS 2200] 20 msec 20 msec 20 msec
7 r-campus.grenet.fr (193.54.184.45) [AS 1942] 20 msec 16 msec 16 msec
8 r-imag.grenet.fr (193.54.185.123) [AS 1942] 20 msec 20 msec 20 msec
9 horus.imag.fr (129.88.38.1) [AS 1942] 16 msec 20 msec 20 msec

Looking glass at genbb1.opentransit.net

sh ip bgp 128.178.50.92
BGP routing table entry for 128.178.0.0/15, version 30024182
  1299 559
    193.251.252.22 (metric 13) from 193.251.128.5 (193.251.128.4)
      Origin IGP, metric 100, localpref 85, valid, internal
      Community: 5511:666 5511:710
      Originator: 193.251.128.4, Cluster list: 0.0.0.10
  1299 559
    193.251.252.22 (metric 13) from 193.251.128.3 (193.251.128.4)
      Origin IGP, metric 100, localpref 85, valid, internal
      Community: 5511:666 5511:710
      Originator: 193.251.128.4, Cluster list: 0.0.0.10
  1299 559
    193.251.252.22 (metric 13) from 193.251.128.1 (193.251.128.4)
      Origin IGP, metric 100, localpref 85, valid, internal, best
      Community: 5511:666 5511:710
      Originator: 193.251.128.4, Cluster list: 0.0.0.10

From genbb1.opentransit.net

Tracing the route to empc19.epfl.ch (128.178.50.92)

1 P5-1.PASBB1.Pastourelle.opentransit.net (193.251.150.25) 8 msec
P4-1.PASBB1.Pastourelle.opentransit.net (193.251.242.134) 8 msec
P5-1.PASBB1.Pastourelle.opentransit.net (193.251.150.25) 8 msec

2 P8-0.PASBB2.Pastourelle.opentransit.net (193.251.240.102) 8 msec 8 msec 8 msec

3 Telia.GW.opentransit.net (193.251.252.22) 8 msec 12 msec 8 msec

4 prs-bb1-pos0-3-0.telia.net (213.248.70.1) [AS 1299] 8 msec 8 msec 8 msec

5 ffm-bb1-pos2-1-0.telia.net (213.248.64.190) [AS 1299] 16 msec 16 msec 16 msec

6 zch-b1-pos6-1.telia.net (213.248.65.42) [AS 1299] 48 msec 32 msec 48 msec

7 dante-01287-zch-b1.c.telia.net (213.248.79.190) [AS 1299] 44 msec 36 msec 44 msec

8 swiEZ2-G3-2.switch.ch (130.59.36.249) [AS 559] 36 msec 44 msec 36 msec

9 swiLS2-G2-3.switch.ch (130.59.36.33) [AS 559] 36 msec 36 msec 36 msec

10 * * *

Demo

- BGP Reports Potaroo.net
- BGPView.io – info prefix 129.88.0.0/16
- Traceroute.org, Looking Glass, Telia
- SDV Looking Glass
- RIPE BGPlay
- CAIDA AS Core
- Renater Services Communautés – 1, 3, 4, 5
- AS-Graph - https://bgp.nat.moe/

Conclusion

- BGP
  - essential to the current structure of the Internet
  - increasing number of AS of Hypergiants
  - complex - policy management, filtering
  - bad configuration – propagation in the entire Internet