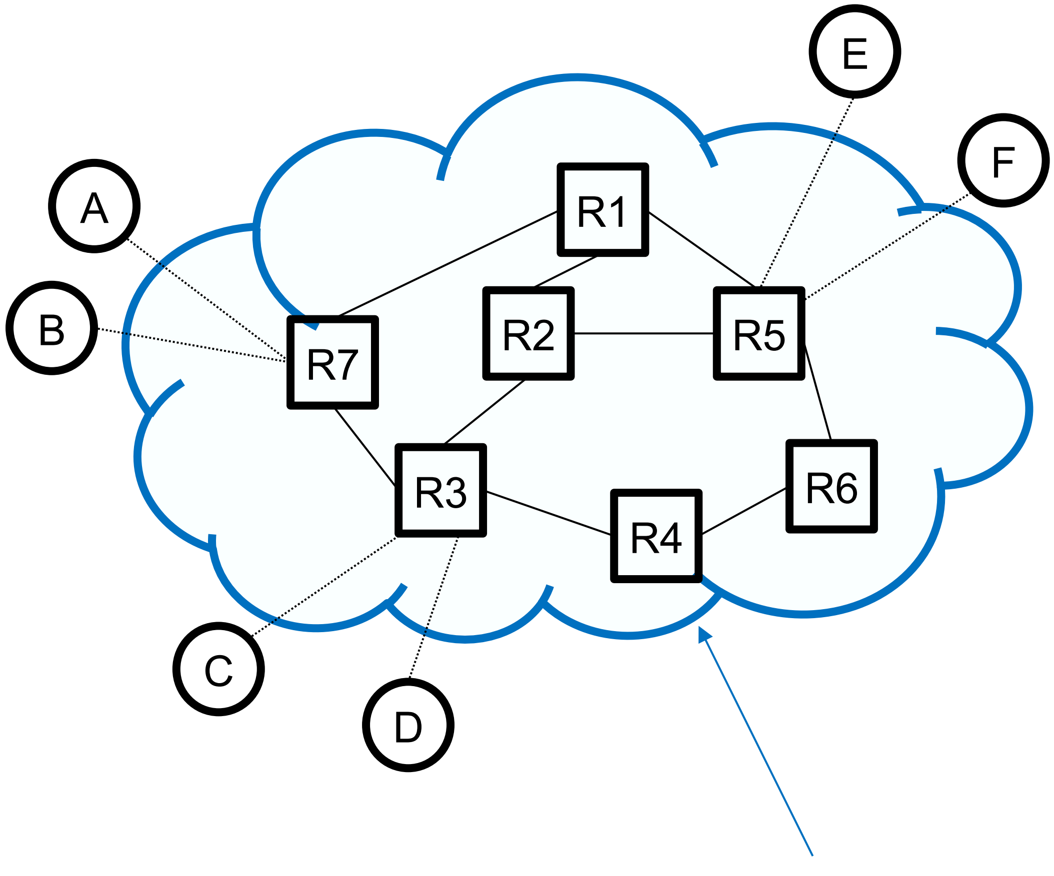
CS 168
Interdomain Routing

Fall 2024

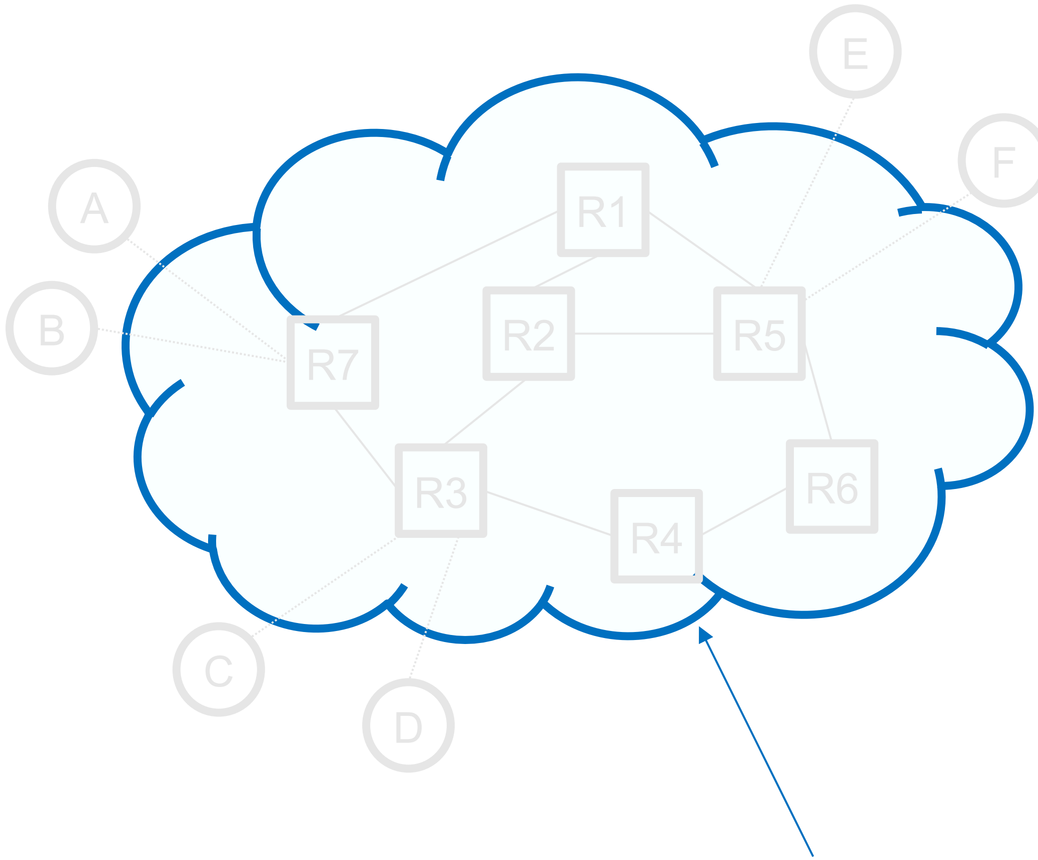
Sylvia Ratnasamy

CS168.io

Routing, so far...

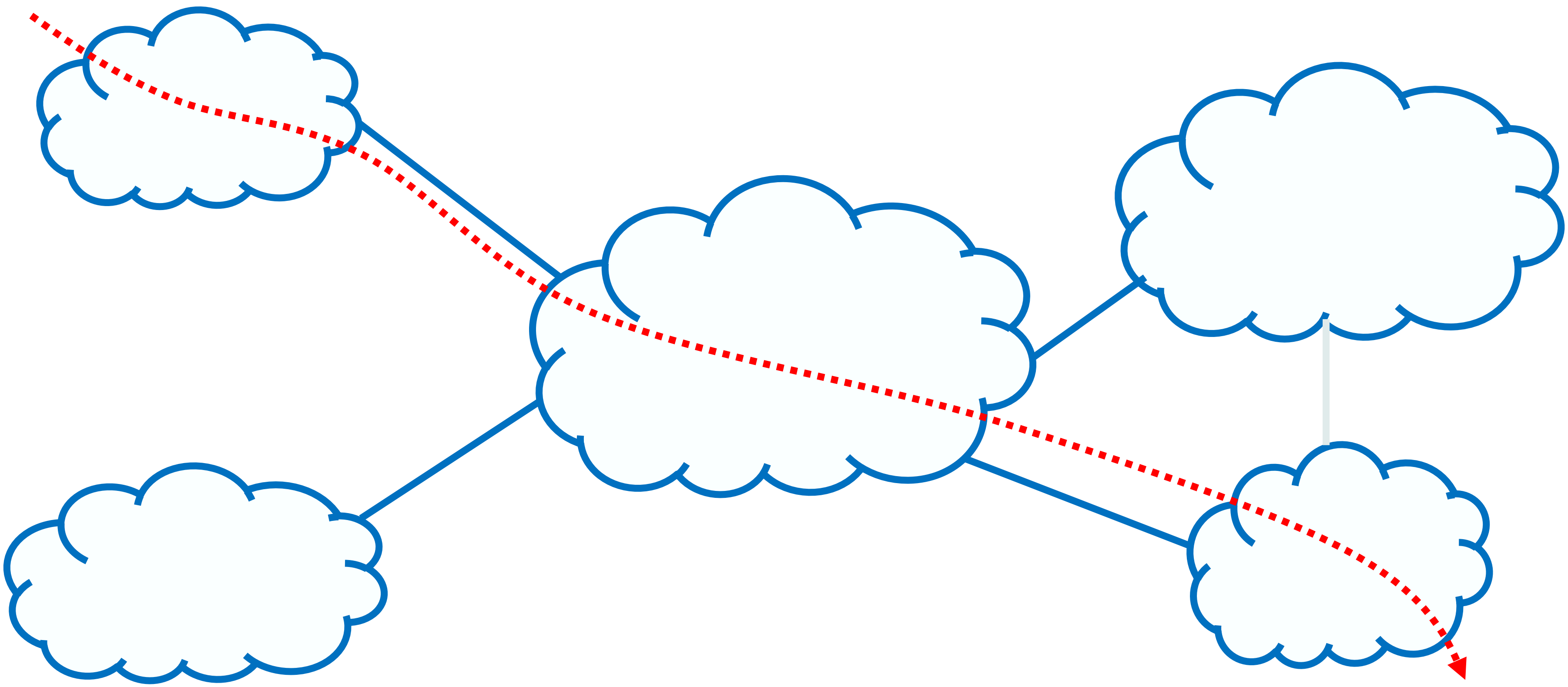


"Domain" or "Autonomous System (AS)"



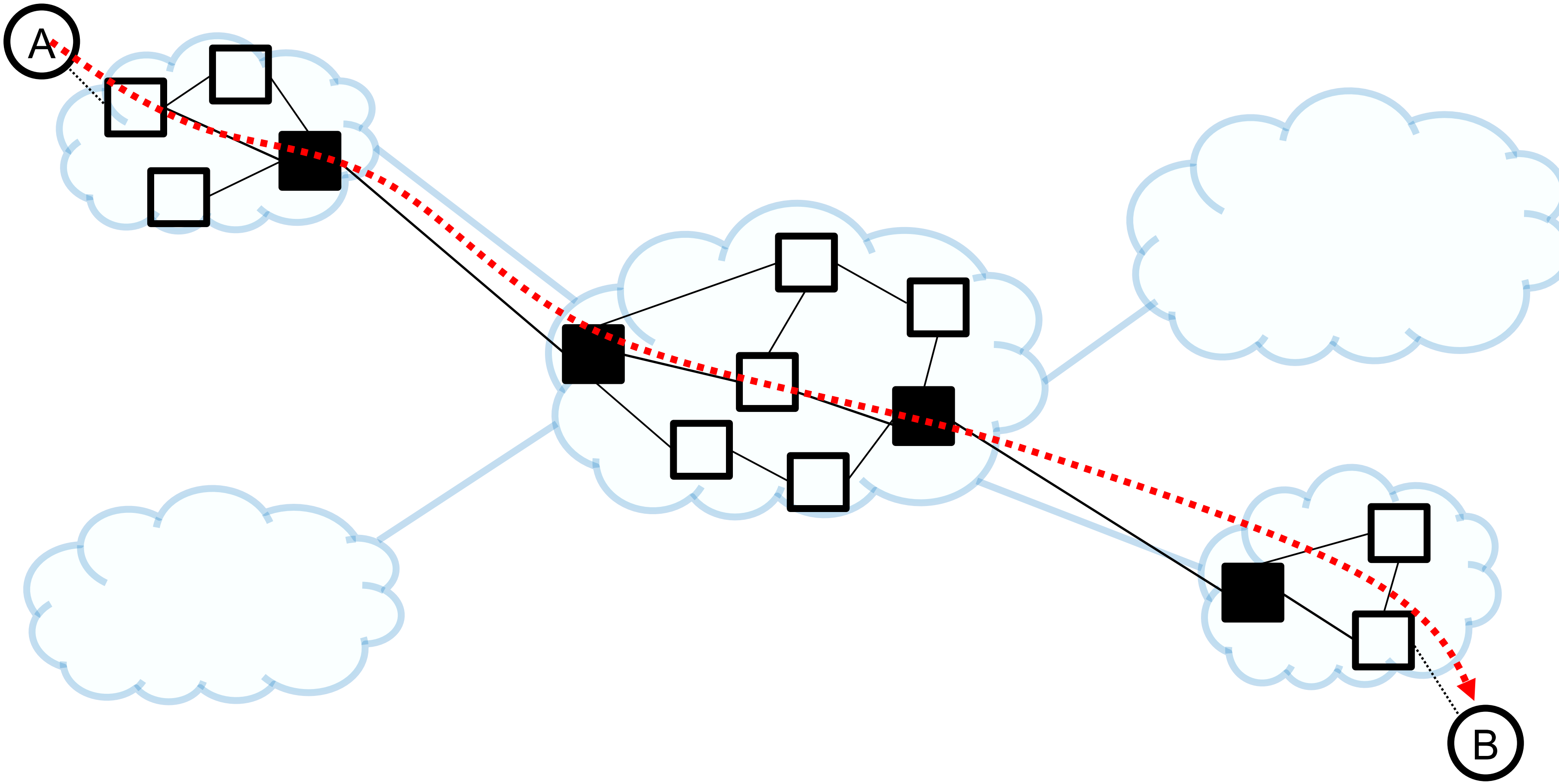
“Autonomous System (AS)” or “Domain”

This week: <u>inter</u>domain routing

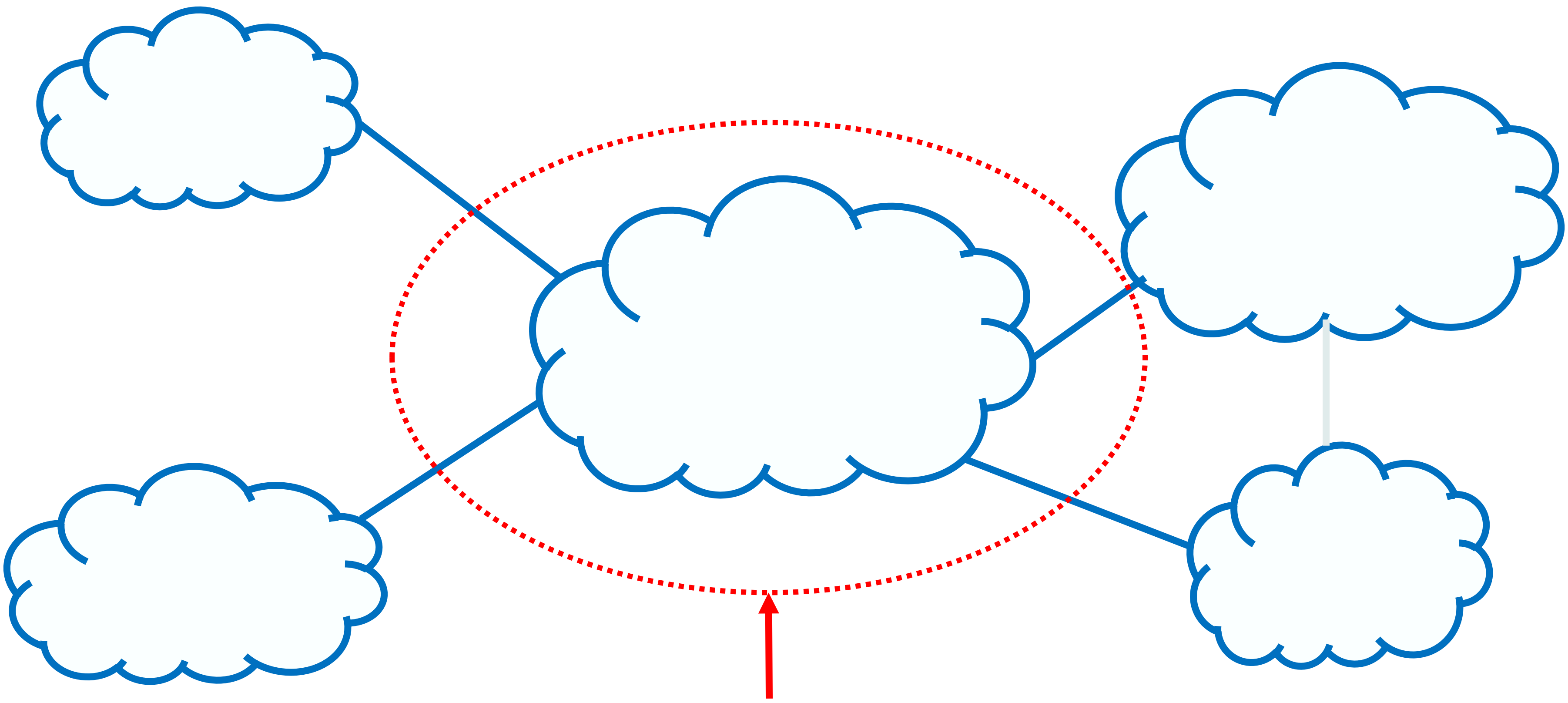


"Interdomain topology" or "AS graph"

This week: <u>inter</u>domain routing

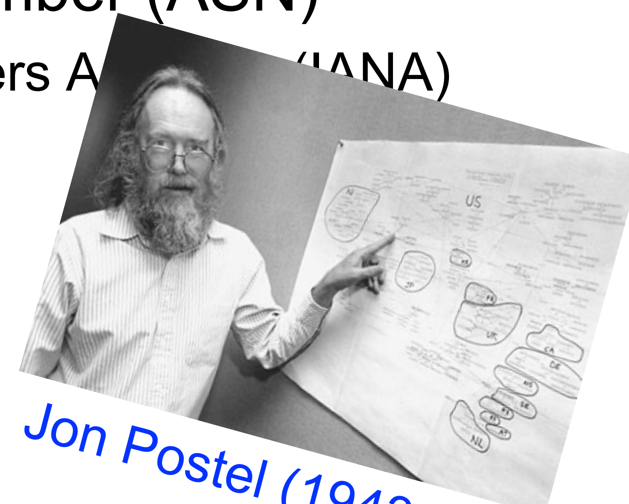


This week: interdomain routing



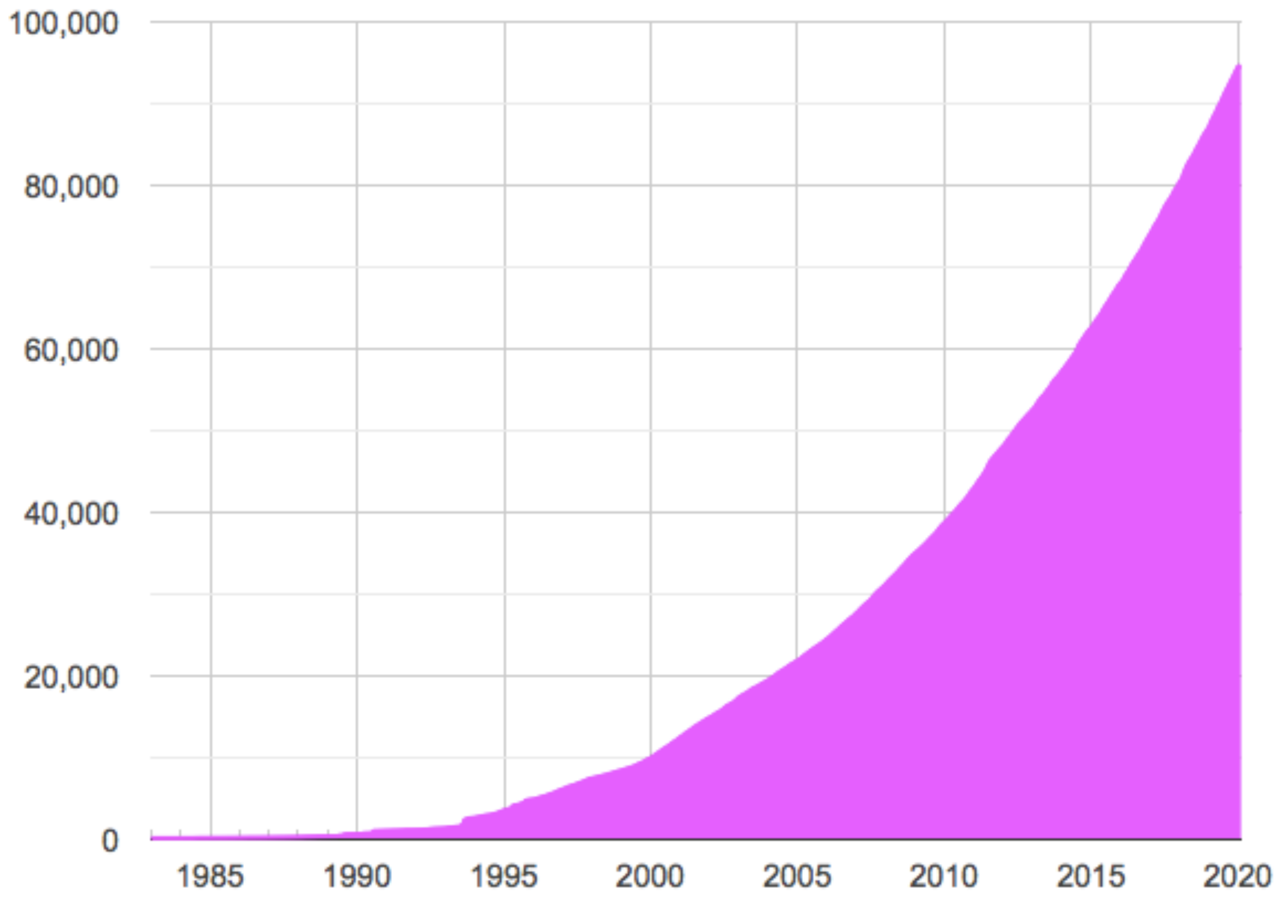
Autonomous Systems (AS)

- AS is a network under a single administrative control
  - Think AT&T, UCB, IBM, France Telecom *etc.*
- Often informally called “domains”
- Each AS is assigned a unique AS number (ASN)
  - Assigned by the Internet Assigned Numbers A (IANA)
  - E.g., ASN 25 is UCB

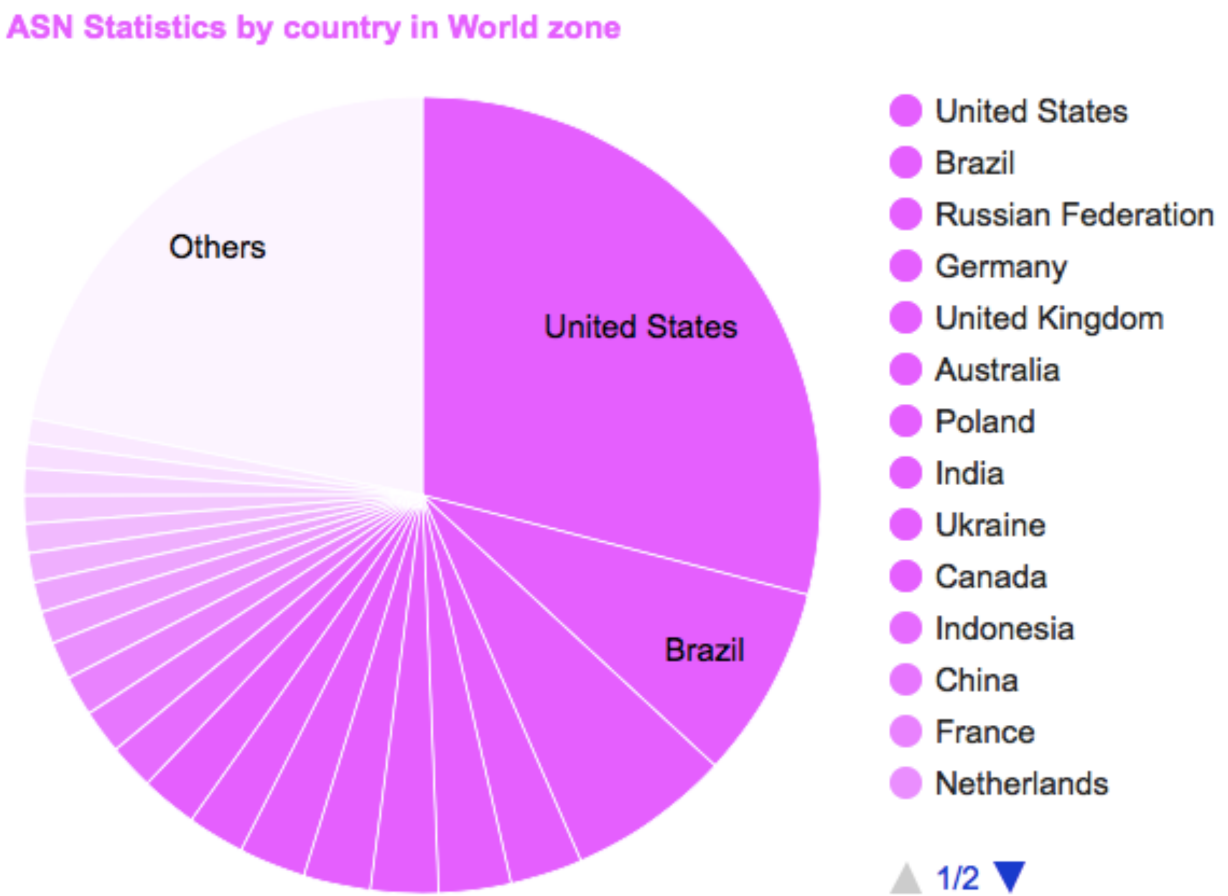


Jon Postel (1943-1998)

Autonomous Systems (AS)



Autonomous Systems (AS)



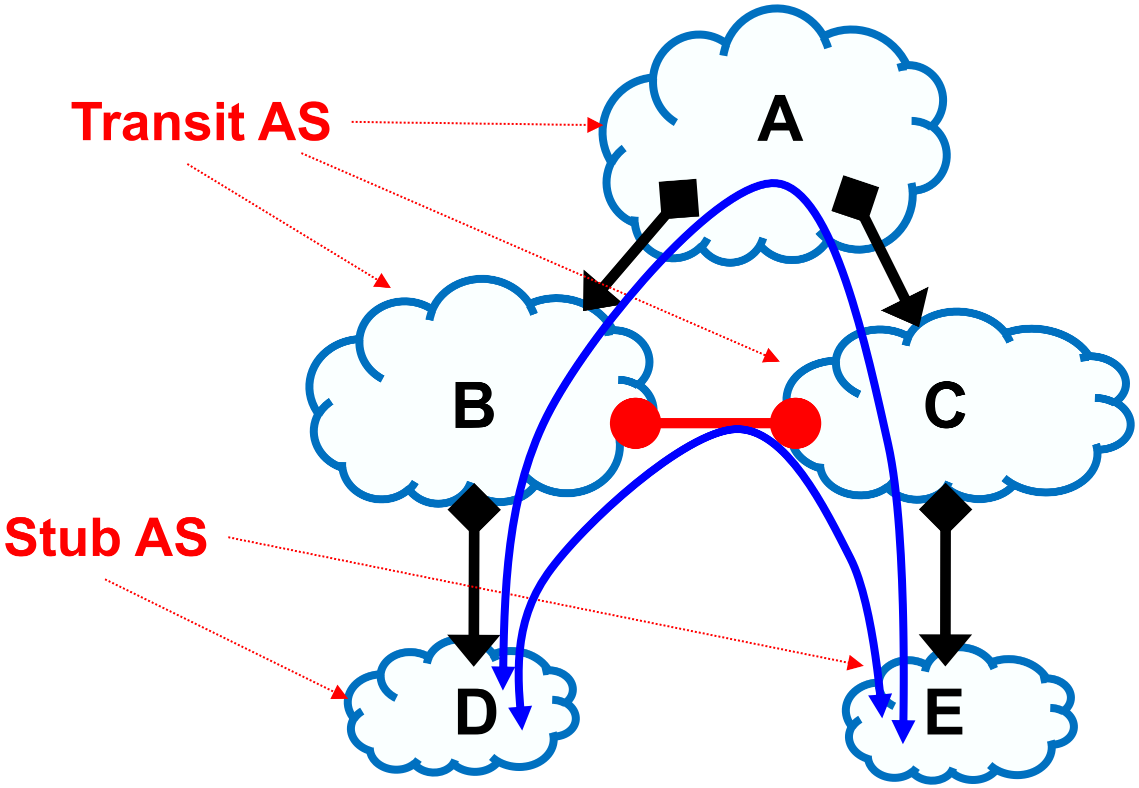
Common Kinds of ASes

- **Stub**: An AS that merely sends/receives packets on behalf of its directly connected hosts
  - Companies, universities, etc.
- **Transit**: carries packets on behalf of other *ASes*
  - Can vary greatly in scale (global, regional, etc.)

Interdomain topology is shaped by the business relationships between ASes

- Three basic kinds of relationships between ASes
  - AS X can be AS Y’s *customer*
  - AS X can be AS Y’s *provider*
  - AS X can be AS Y’s *peer*
- Business implications
  - Customer pays provider
  - Peers don’t pay each other
    - Assumed to exchange roughly equal traffic

AS graph w/ business relationships



E.g., D and E talk a lot

Peering saves B *and* C money

Relations between ASes

provider ◆——▶ customer

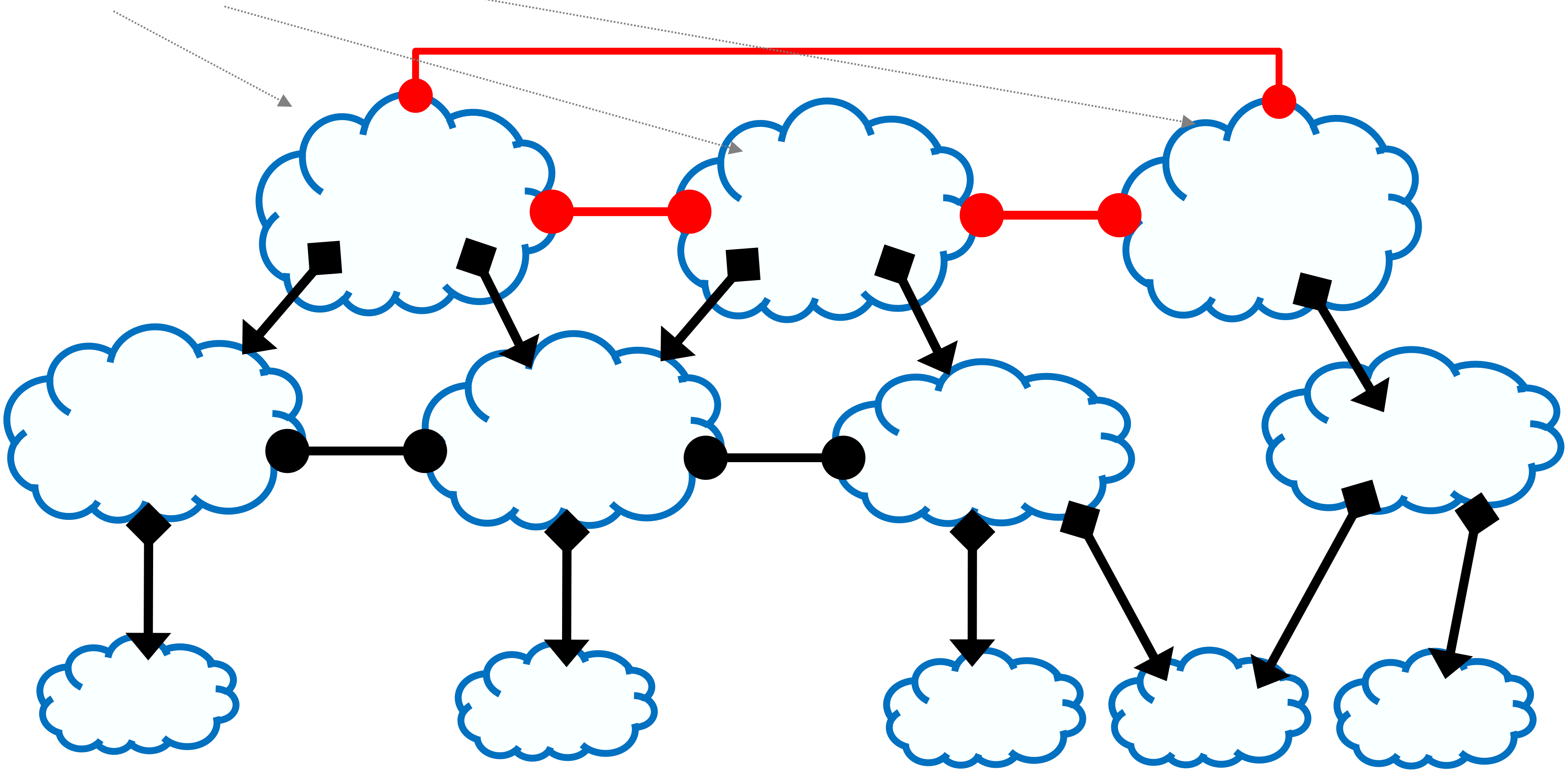
peer ●——● peer

Business Implications

- Customers pay provider
- Peers don't pay each other

AS graph w/ business relationships

“Tier 1” ASes



Outline

- Context
- Goals / Challenges
- Approach
- Detailed design
- Problems with BGP

Recall: goals for intradomain routing?

- Goals
  - Find valid routes → no loops, no deadends
  - Find “good” paths → least cost paths

Goals for <u>inter</u>domain routing?

- Still want valid routes, *etc.*
- Plus two new goals:
  - Scalability: routing must scale to the entire Internet!
  - Policy compliance: routes must reflect business goals

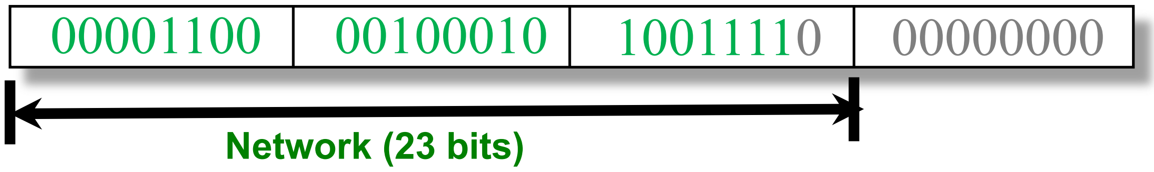
Scaling

- A router must be able to reach *any* destination
  - Given any destination address, must know the “next hop”
- Naive: Have an entry for each destination
  - Doesn’t scale!
- Recall, last lecture: host addressing key to scaling!

Recall: IP addresses are hierarchical

- Hierarchical in structure and allocation
  - Address partitioned into a network prefix host suffix
  - Prefix represents *all* hosts in that network

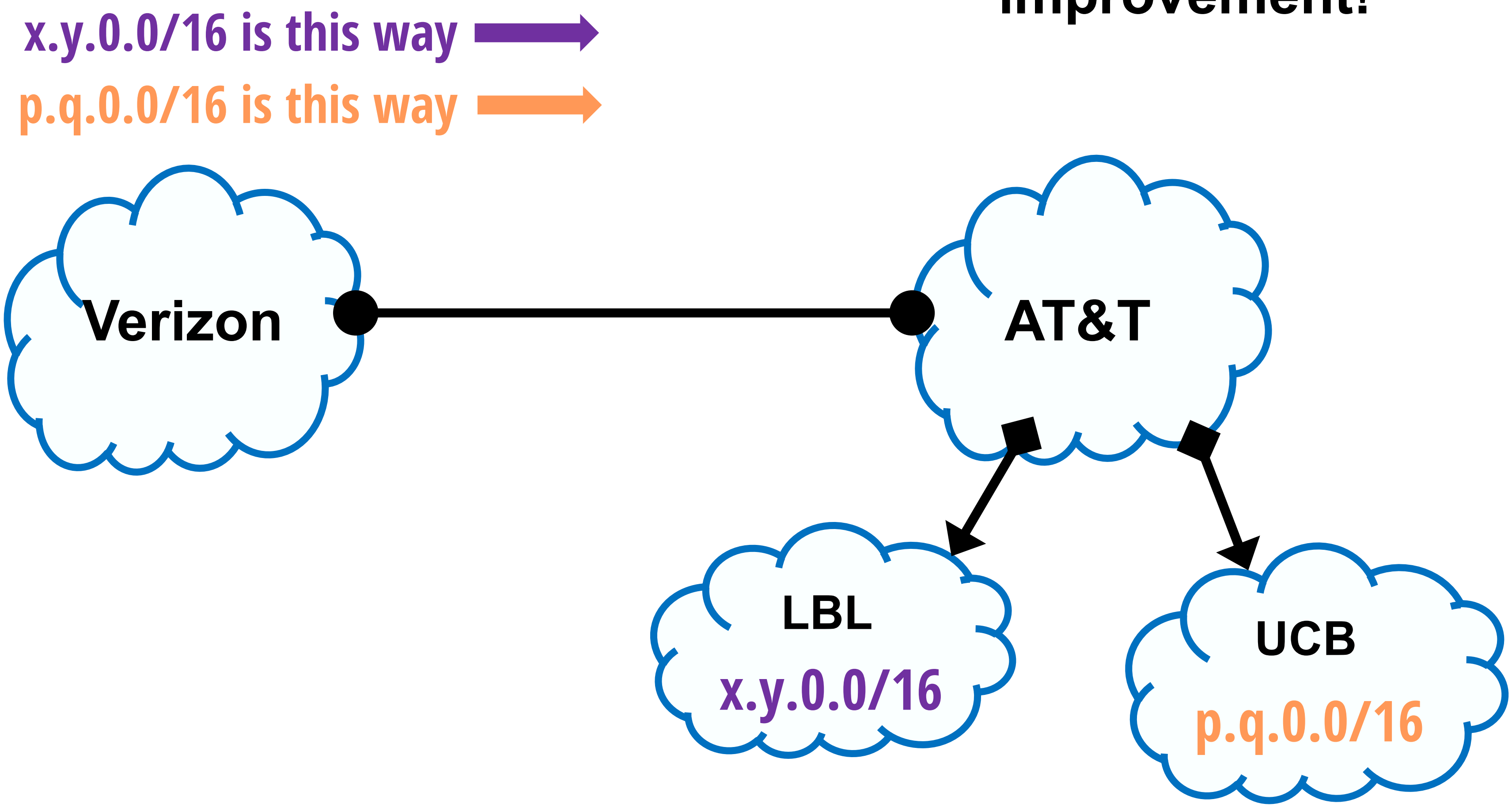
- Destinations in interdomain routing are prefixes



This prefix is: 12.34.158.0/23

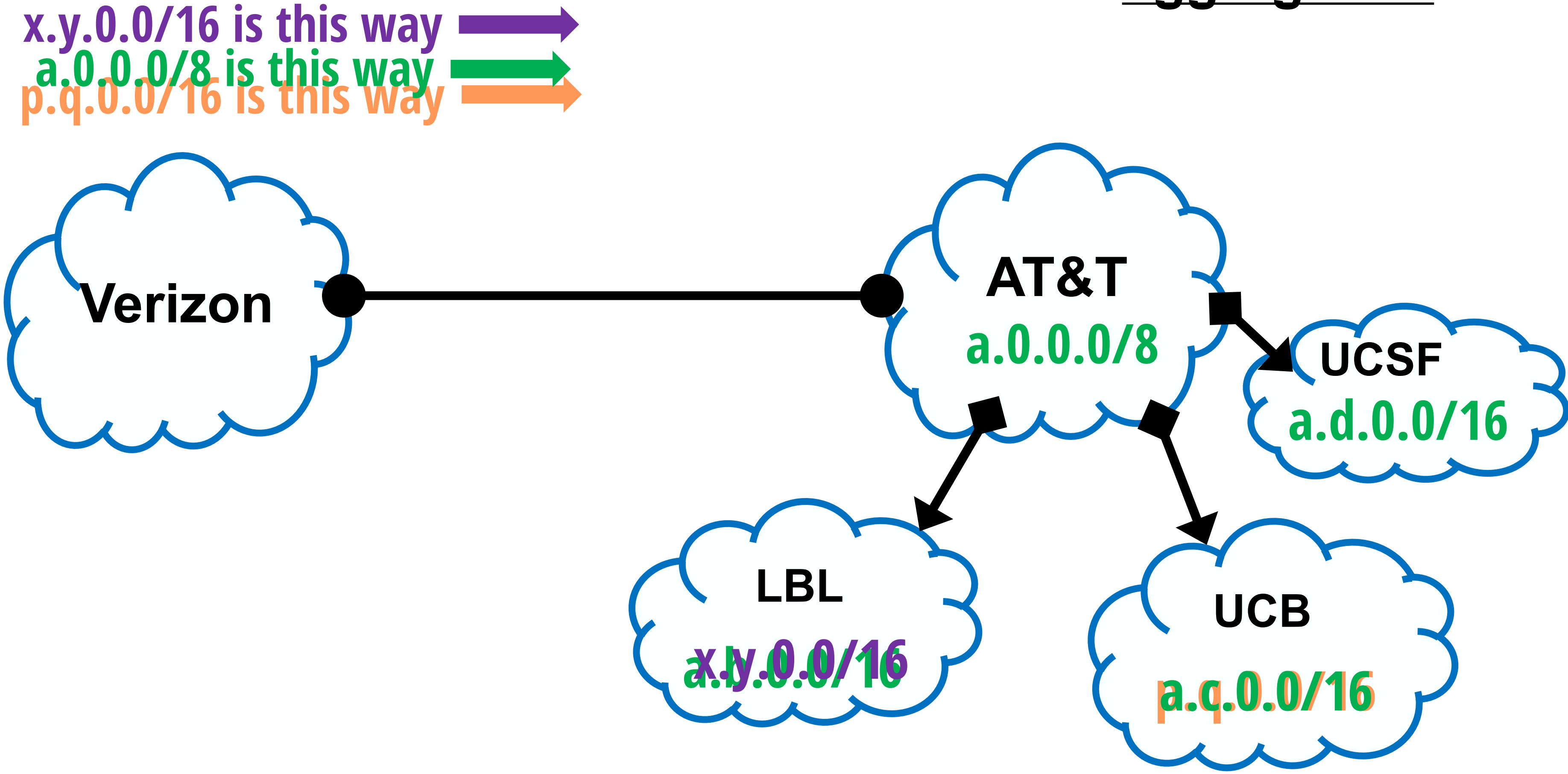
Back to our AS Graph ...

Already a huge improvement!

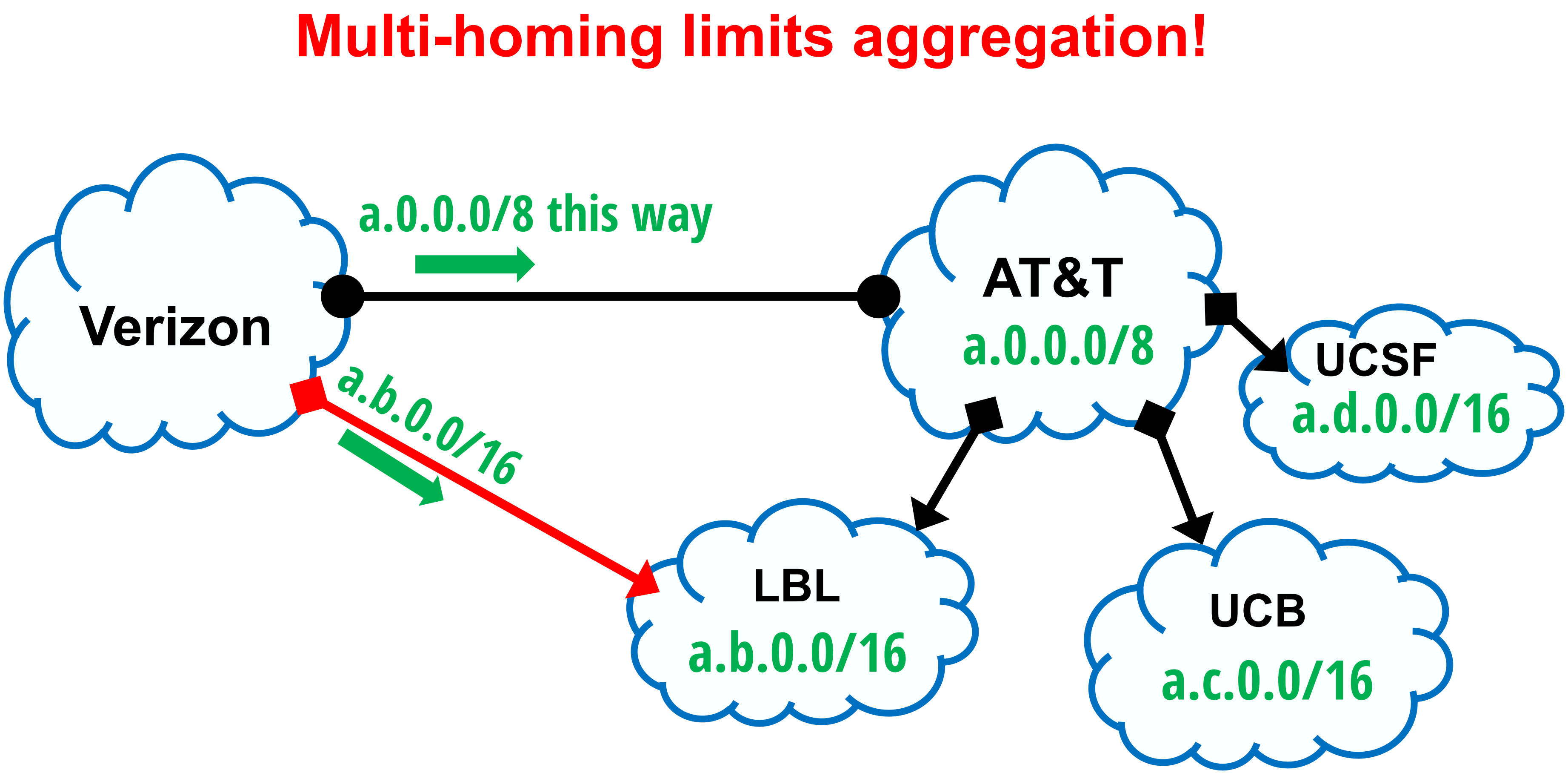


Back to our AS Graph ...

Hierarchical allocation enables <u>aggregation</u>!



Back to our AS Graph ...



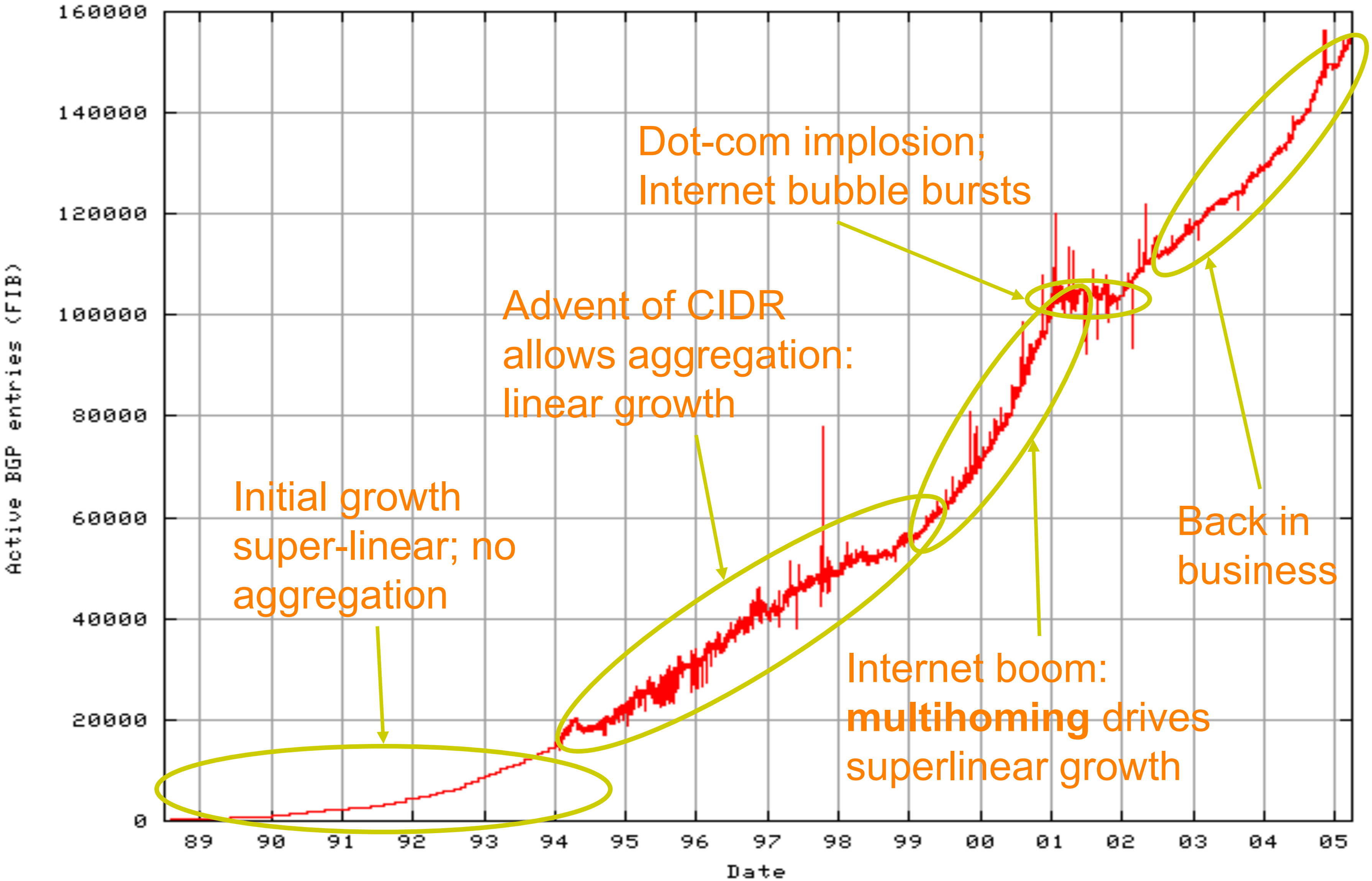
Verizon needs routing entries for both a.0.0.0/8 and a.b.0.0/16

IP addressing → scalable routing?

- Adress aggregation helps routing scalability
- But we may not be able to aggregate addresses for "multi-homed" networks
  - Multi-homed → more than one provider
- Two competing forces in scalable routing
  - aggregation reduces number of routing entries
  - multi-homing increases number of entries

https://blog.apnic.net/2023/01/06/bgp-in-2022-the-routing-table/

Growth in Routed Prefixes (1989-2005)



Goals for interdomain routing?

- Two new goals:
  - Scalability: routing must scale to the entire Internet!
  - Policy compliance: routes must reflect business goals

Administrative preferences shape interdomain routing

- ASes want freedom to pick routes based on policy

Policy

- *“I don’t want to carry AS#2046’s traffic through my network”*
- *“Prefer it if my traffic is carried by AS#10 instead of AS#4”*
- *”Avoid AS#54 whenever possible”*
- *On Mondays I like AS#12, on Tuesdays AS#13*
- Not expressible as Internet-wide “least cost”!

Two Principles For Typical Policies

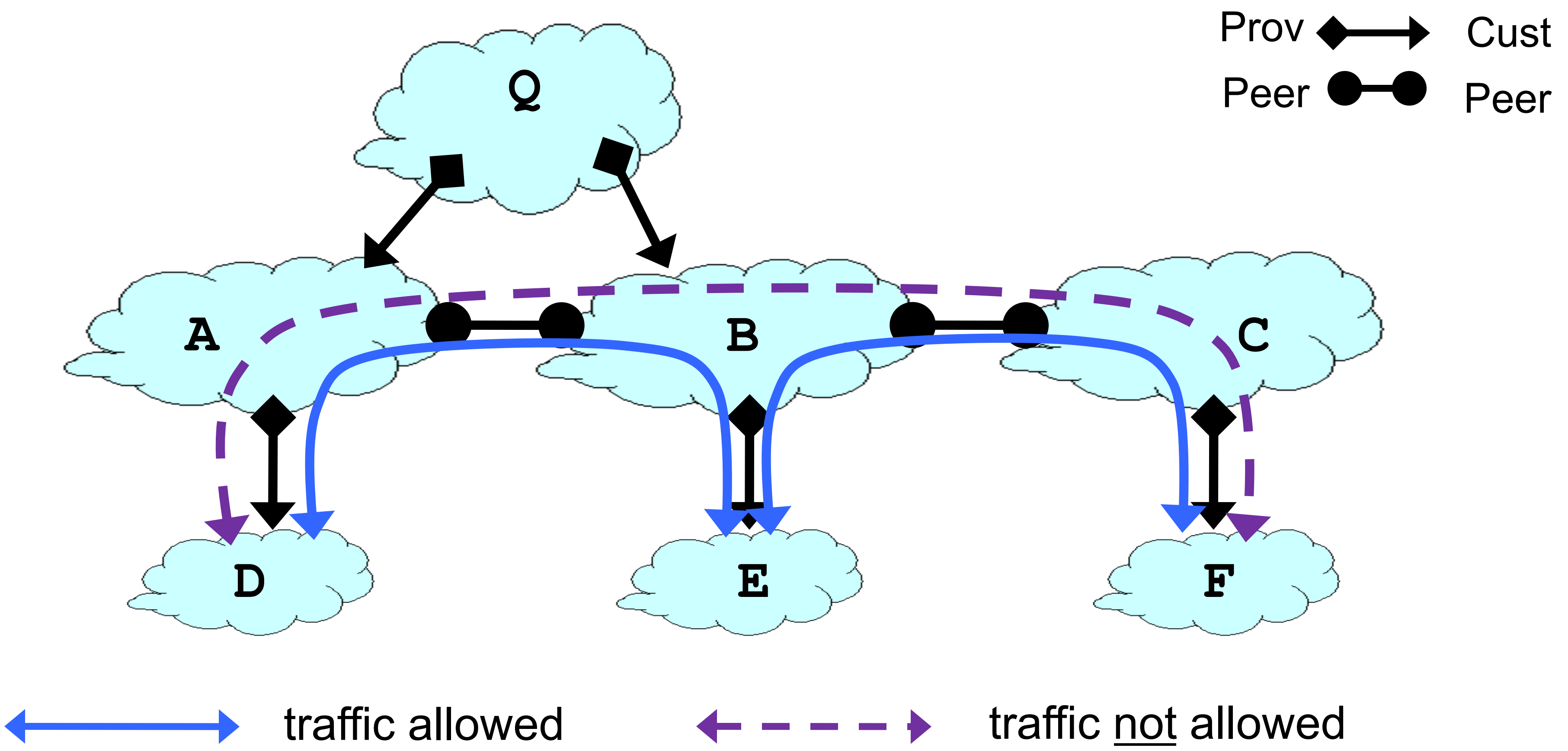
1) Don't accept to carry traffic if you are not being paid!

- Traffic should come from or go to customer
- This is about what traffic I *carry*

2) Make/save money when sending traffic

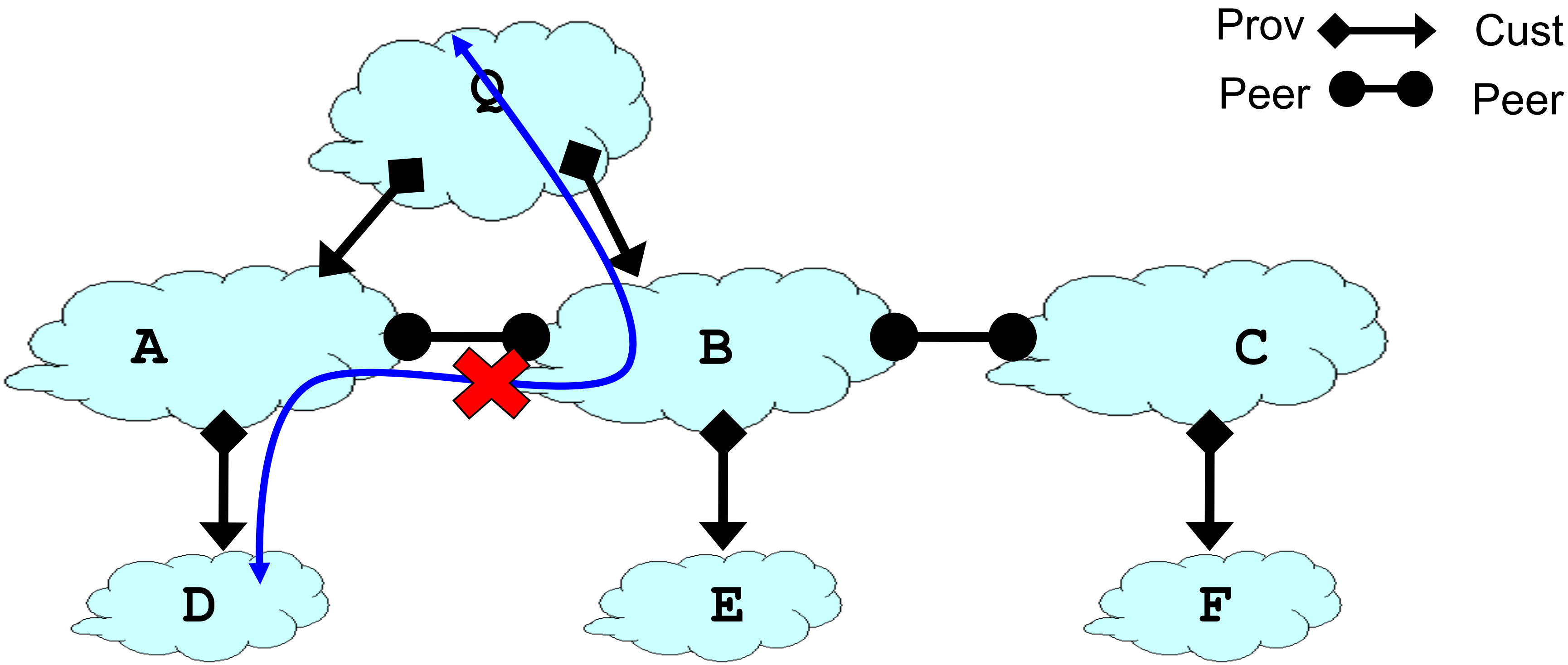
- Prefer sending traffic to customer
- If can't do that, then a peer
- Only send via a provider if I have to
- This is about where I *send* traffic

Routing Follows the Money!



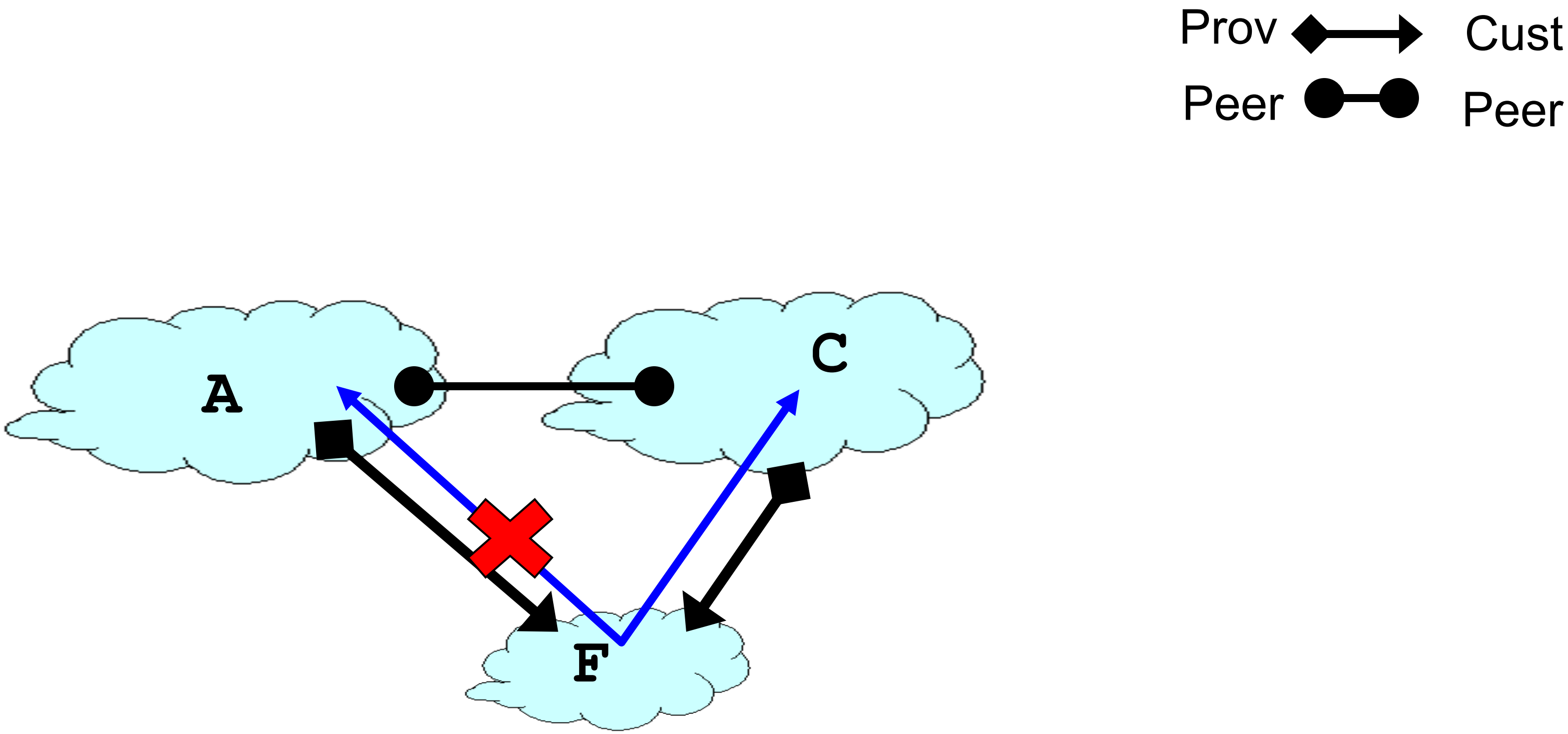
Peers do not provide transit between other peers

Routing Follows the Money!



An AS only carries traffic to/from its own customers over a peering link

Routing Follows the Money!



Routes are “valley free” (will return to this later)

Administrative preferences shape interdomain routing

- ASes want freedom to pick routes based on policy
- ASes want autonomy
- ASes want privacy

Autonomy and Privacy

- ASes want autonomy
  - Want the freedom to choose their own policies
- ASes want privacy
  - Don't want to *explicitly* announce these choices to others
- Policy is "what" we want to achieve; autonomy and privacy are requirements on "how" we achieve it

Recap: Interdomain Setup

- Inter-domain topology
  - Nodes are Autonomous Systems (ASes)
  - Destinations are IP prefixes (12.0.0.0/8)
  - Links represent physical links *and* biz relationships
- Business relationships between ASes impact which routes are acceptable
- Interdomain routing design must support these **policy** choices while preserving domains' **autonomy** and **privacy**
- **Border Gateway Protocol (BGP)** is current design

BGP: a new routing paradigm

- The idea of routing through a network is an old one
  - Dijkstra's (1956); Bellman-Ford (1958); ...
  - All designed to find "least cost" paths

- The notion of "autonomous systems" with their private policies was new
  - BGP was hastily designed in response to this need

- Has proven effective but with some serious warts

Outline

- Context
- Goals / Challenges
- Approach
- BGP: detailed design
- Limitations

Choice of Routing Algorithm

Link State (LS) *vs.* Distance Vector (DV)?

- LS offers no privacy – broadcasts all network information
- LS limits autonomy -- need agreement on metric, algorithm

- DV is a decent starting point
  - But wasn't designed to implement policy
  - Per-destination routing updates as a hook to implement policy?

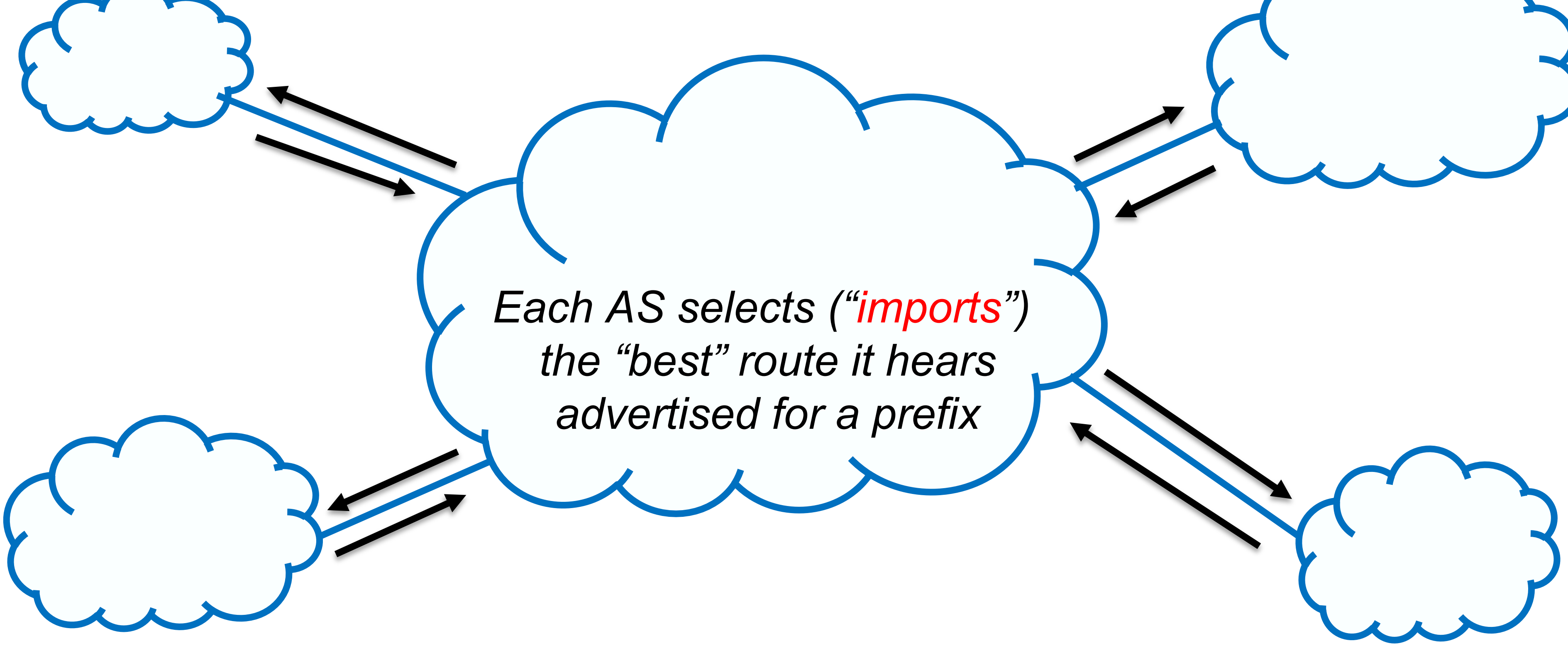
BGP extends DV to accommodate policy

Outline

- Context
- Goals / Challenges
- Approach
  - From DV to BGP
  - How policy is implemented (detail-free version)
- Detailed design
- Problems with BGP

BGP: Basic Idea

An AS advertises ("exports") its best routes to one or more IP prefixes



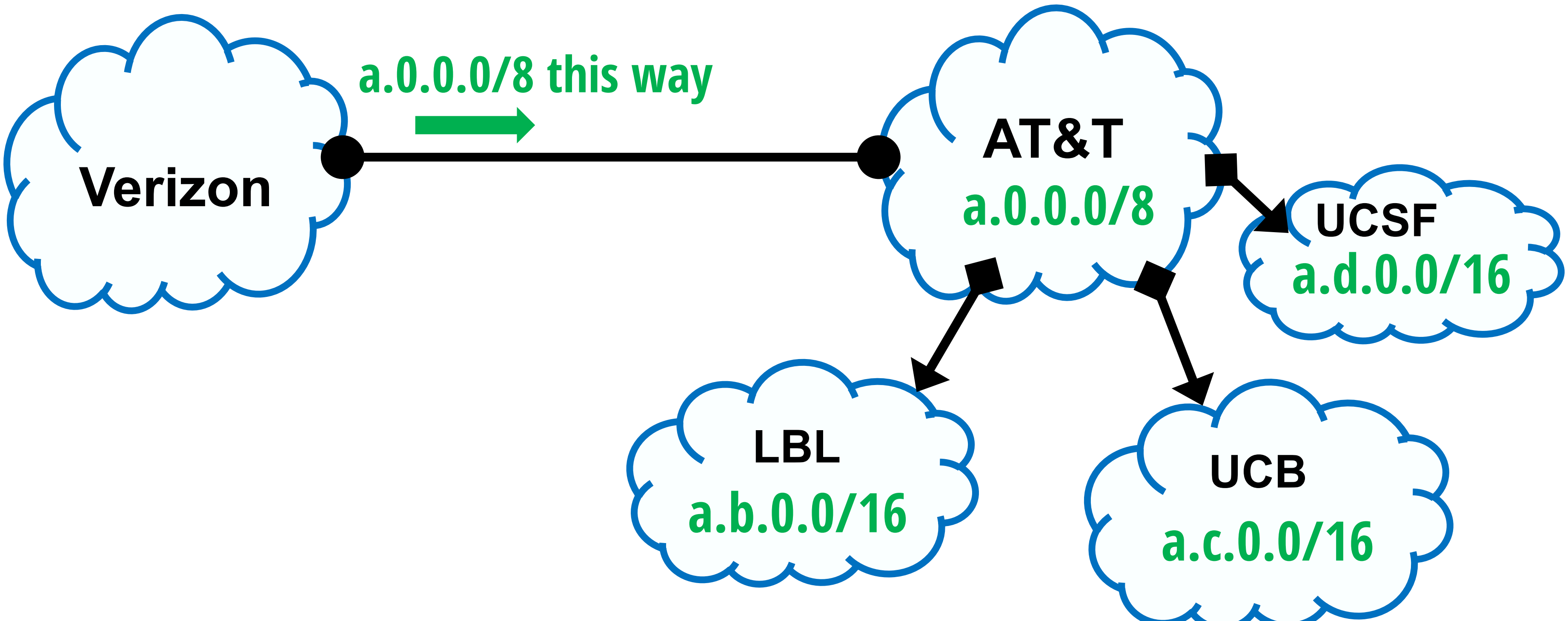
Policy will determine which route advertisements are selected and which are advertised (more later)

BGP inspired by Distance Vector

- Per-destination (prefix) route advertisements
- No global sharing of network topology info.
- Iterative and distributed convergence on paths
- With four crucial differences!

Differences between BGP and DV
(1) BGP may *aggregate* destinations

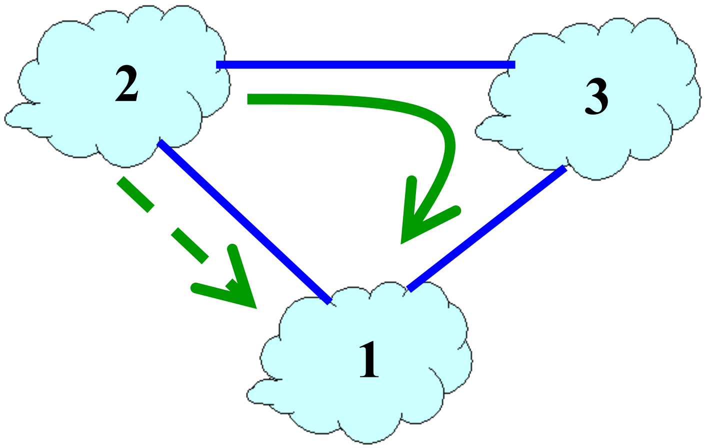
- For scalability, BGP may aggregate routes for different prefixes



Differences between BGP and DV

(2) Not picking shortest path routes

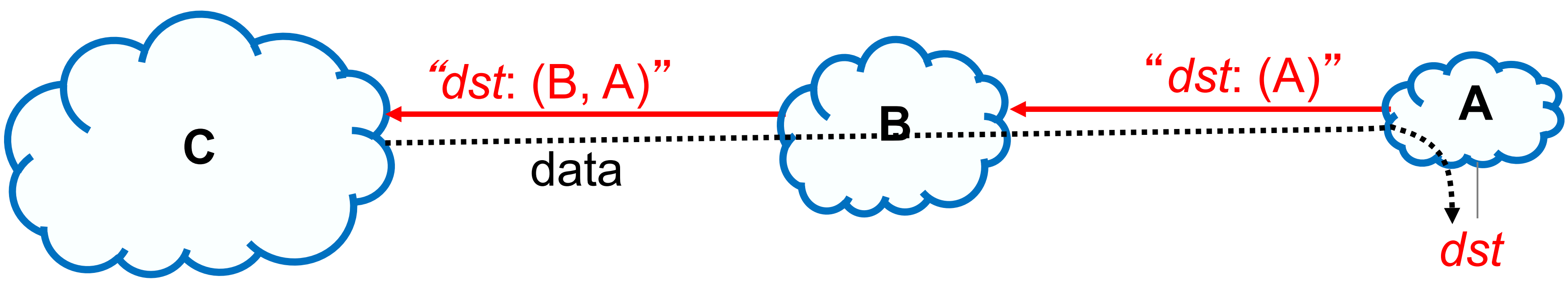
- BGP selects the best route based on policy, not least cost
- How do we avoid loops?



Node 2 may prefer "2, 3, 1" over "2, 1"

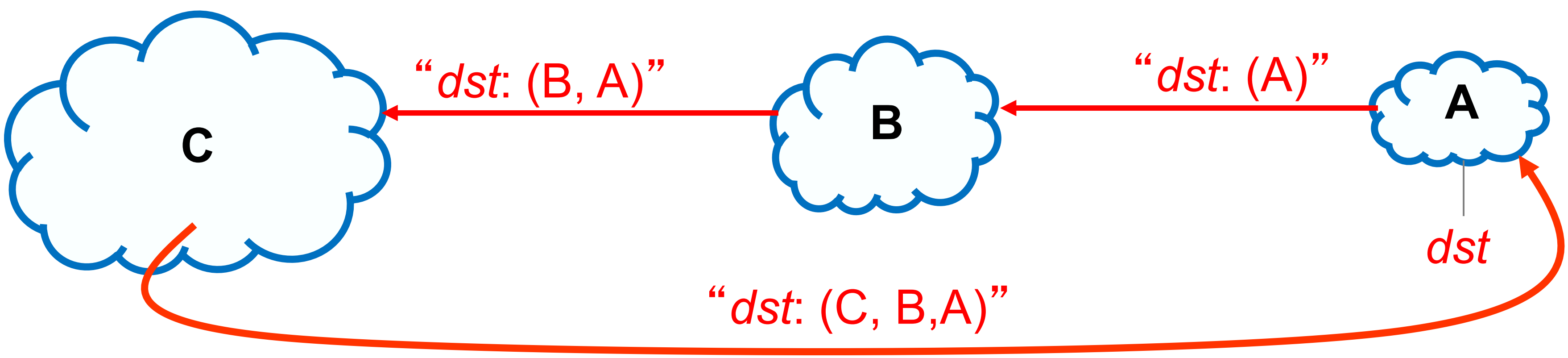
Differences between BGP and DV
(3) Distance-vector → Path-vector

- Key idea: advertise the entire path
  - Distance vector: send *cost metric* per destination
  - **Path vector**: send the *entire AS path* for each destination



Loop Detection w/ Path Vector

- AS can easily detect and discard paths w/ loops
  - E.g., A sees itself in the path "C, B, A"
  - E.g., A simply discards the advertisement

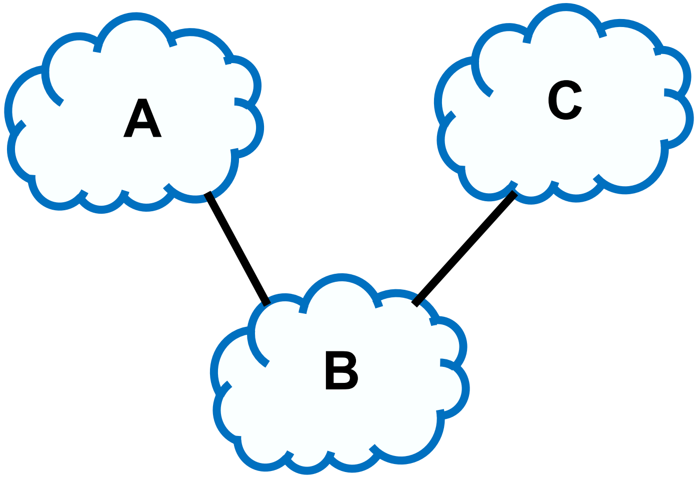


Differences between BGP and DV
(3) Distance-vector → Path-vector

- Key idea: advertise the entire path
  - Distance vector: send *cost metric* per destination
  - Path vector: send the *entire AS path* for each destination
- Benefits
  - Loop avoidance is easy
  - Can base policies on the entire path

Differences between BGP and DV
(4) Selective route advertisement

- For policy reasons, an AS may choose not to advertise a route to a destination
- Hence, reachability is not guaranteed even if graph is connected



Example: B does not want to carry traffic between A and C

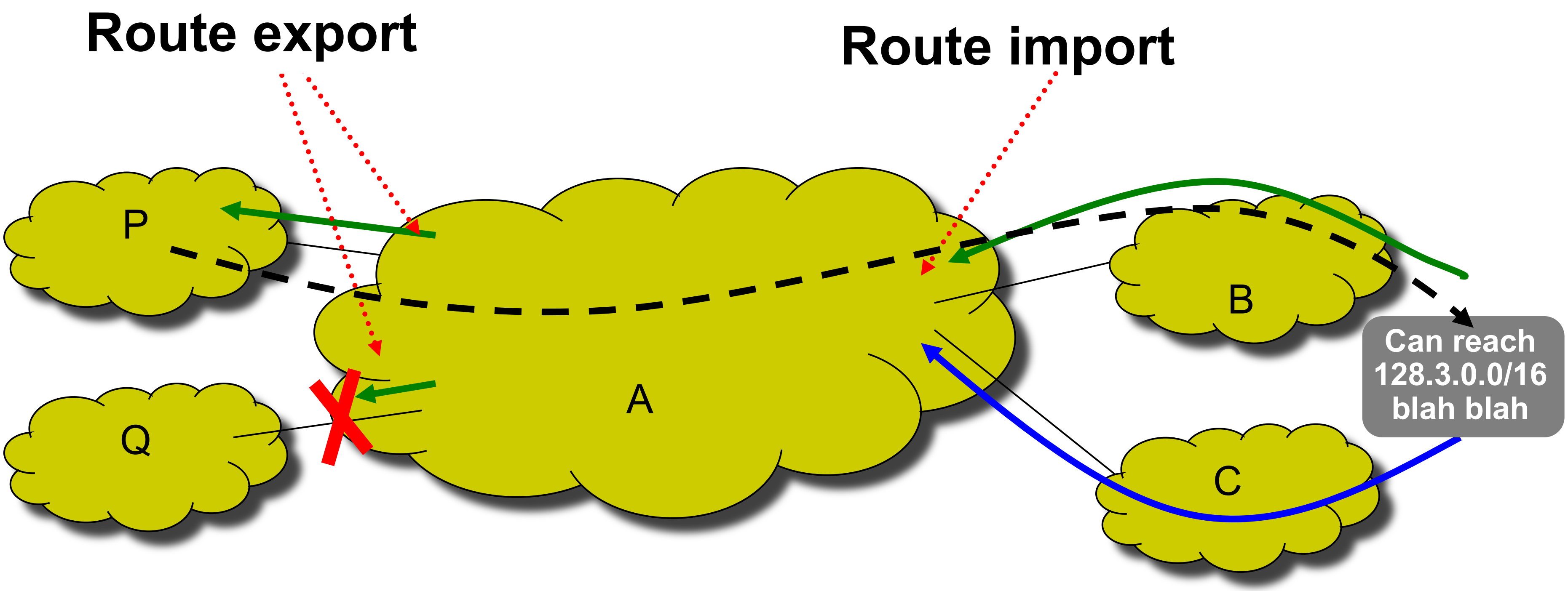
Recap: four differences

- BGP may aggregate destinations and routes
- Route selection not based on shortest path
- Advertise the entire path (path vector)
- Selective route advertisement

Outline

- Context
- Goals
- **Approach**:
  - BGP extends Distance-Vector
  - How policy is implemented (detail-free version)
- Detailed design
- Limitations

Policy imposed in how routes are import and exported



- **Import** (aka selection): Which path to use?
  - controls whether/how traffic **leaves** the network
- **Export**: Which path to advertise?
  - controls whether/how traffic **enters** the network

Repeating Two Crucial Points

- Import (selection): Which path to use?
  - Determines **where** your traffic ***goes***
- Export: Which path to advertise?
  - Determines **which** traffic you ***carry***

Gao-Rexford Rules

- Rules that describe common – not required! – practice in import/export policies
- Essential to understanding why the Internet works
  - Because it wouldn't if policies were completely general

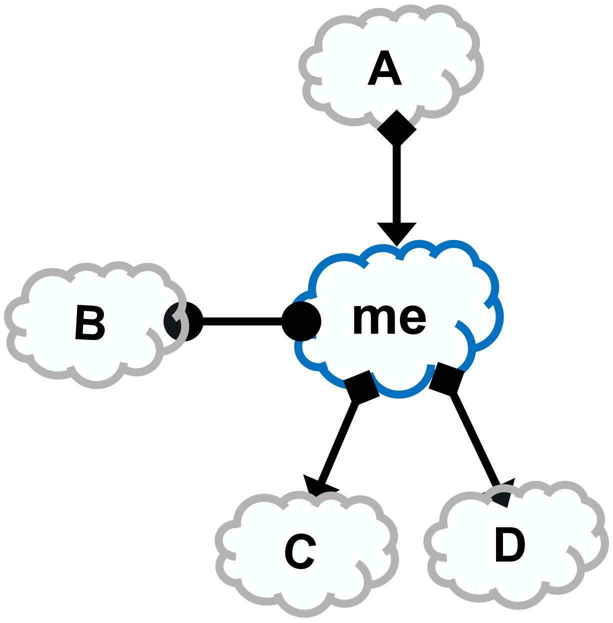
Gao-Rexford Rule: Import policy

- When importing (selecting) a route to a destination, pick route advertised by customer > peer > provider
- In practice, ASes use additional rules to break ties
- Typical example, in decreasing order of priority:
  - make/save money (G-R rule)
  - maximize performance
  - minimize use of my network bandwidth
  - ....

Gao-Rexford Rules: Export policy

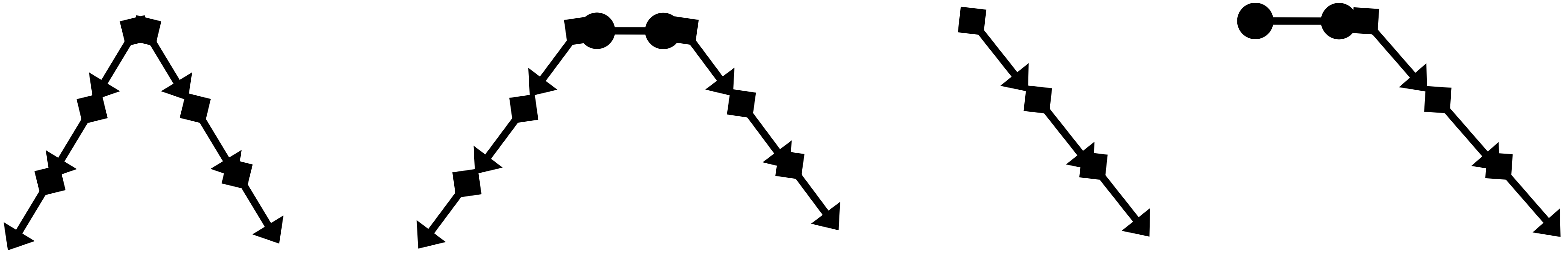
- Question: where should I export a route?
  - Recall: ASes that I export a route to, will send traffic *to* me

Destination prefix advertised by…	Export route to…
Customer	Everyone (providers, peers, other customers)
Peer	Customers
Provider	Customers



Gao-Rexford Rules: Property

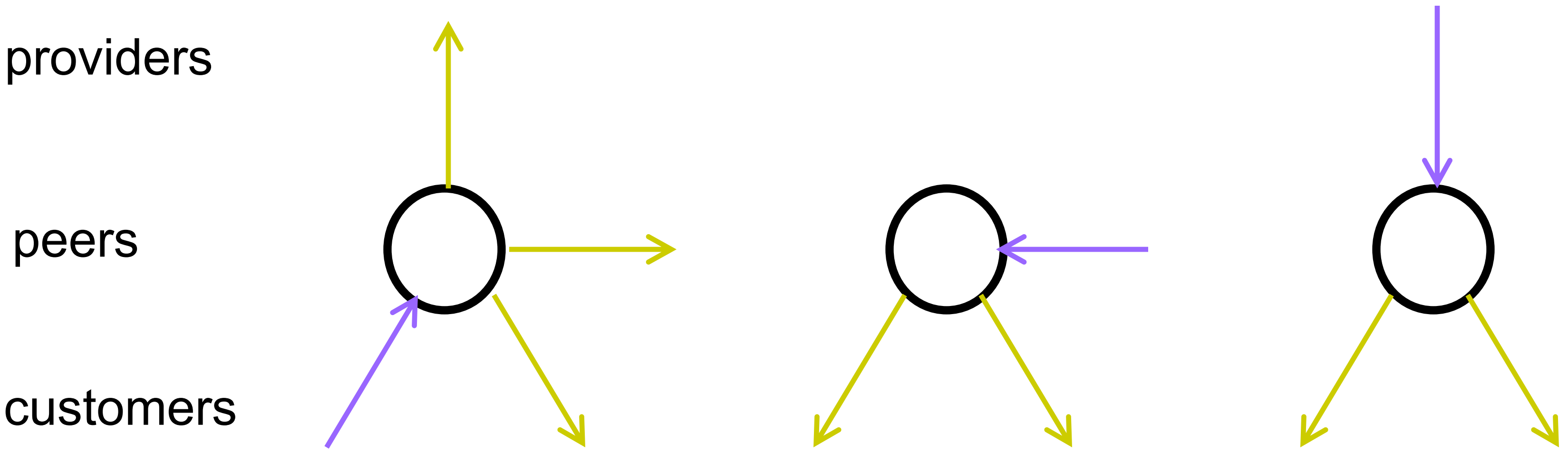
If all ASes follow G-R, routes are "valley free"



"valley free" == "single peaked"

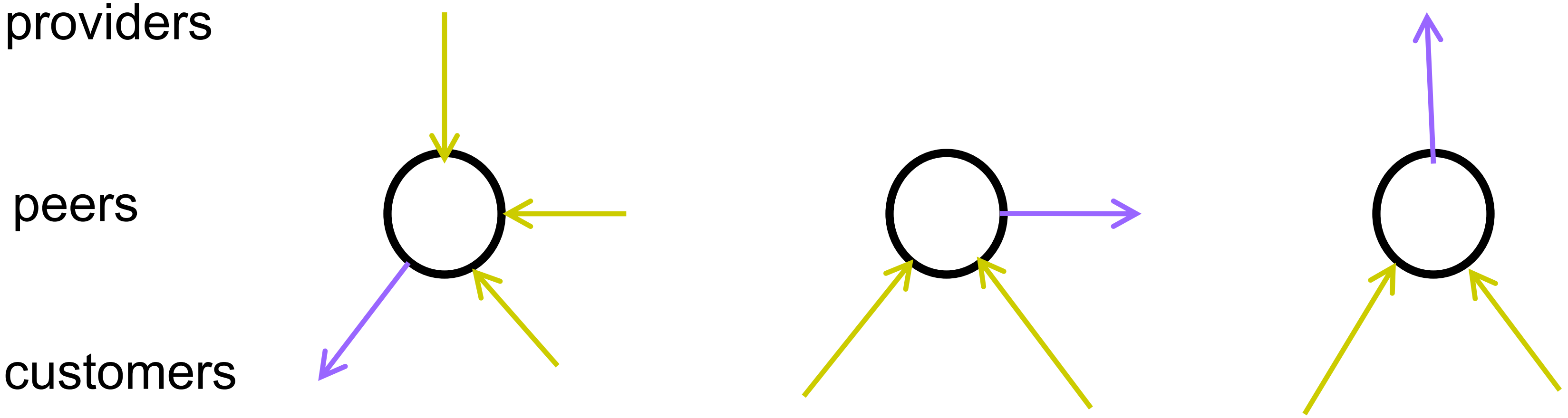
Why Valley-Free?

If all ASes follow G-R, routes are "valley free"



Why Valley-Free?

If all ASes follow G-R, routes are "valley free"



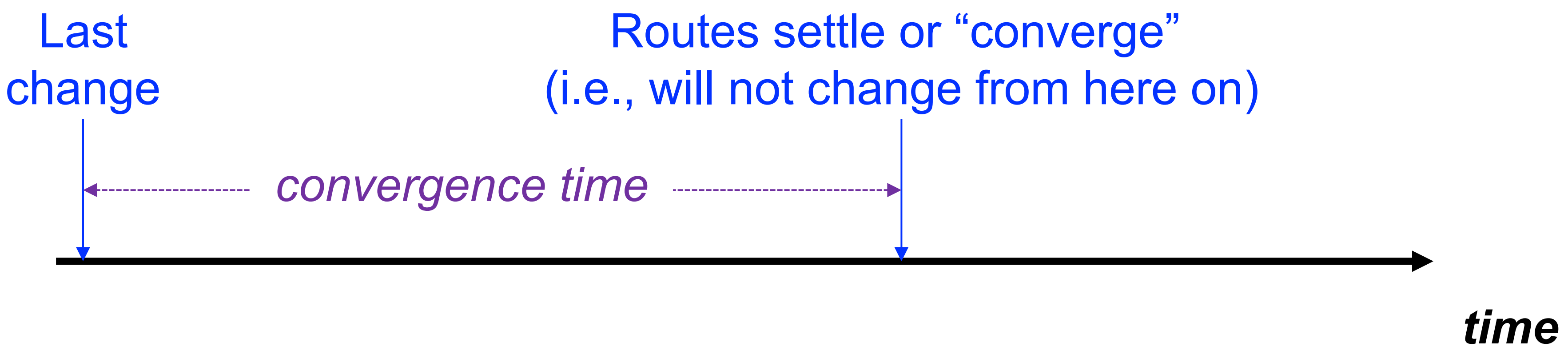
Proof: based on observing that once traffic arrives from a provider (above) or peer (side), it can only go down

Gao-Rexford Rules: Implication

- Under two assumptions about the AS graph (coming up), if all ASes follow Gao-Rexford, we can guarantee:
  - **Reachability**: any two ASes can communicate
  - **Convergence**: all routers agree on paths
- The above hold in **steady state**

Steady State and Convergence

- Steady state essentially means no changes
  - No addition/removal/failure of nodes, links, destinations
  - No change in policies, *etc.*



Two assumptions

#1 The graph of customer-provider relationships is acyclic

- Cannot have A→B→...→C and then C→A (cust → prov)
- Means one can arrange providers in a hierarchy
- Note: OK if peering relationships are cyclic (A-B, B-C, C-A)

#2 Starting from any AS, and following the chain of providers leads to a Tier 1 AS

- Tier 1: group of provider ASes that all peer with each other

Gao-Rexford Rules: Implication

- Under two assumptions about the AS graph (coming up), if all ASes follow Gao-Rexford, we can guarantee:
  - **Reachability**: any two ASes can communicate
  - **Convergence**: all routers agree on paths
- The above hold in steady state
- The above are <u>not</u> guaranteed for general policies!
  - (You'll see an example of this in section)

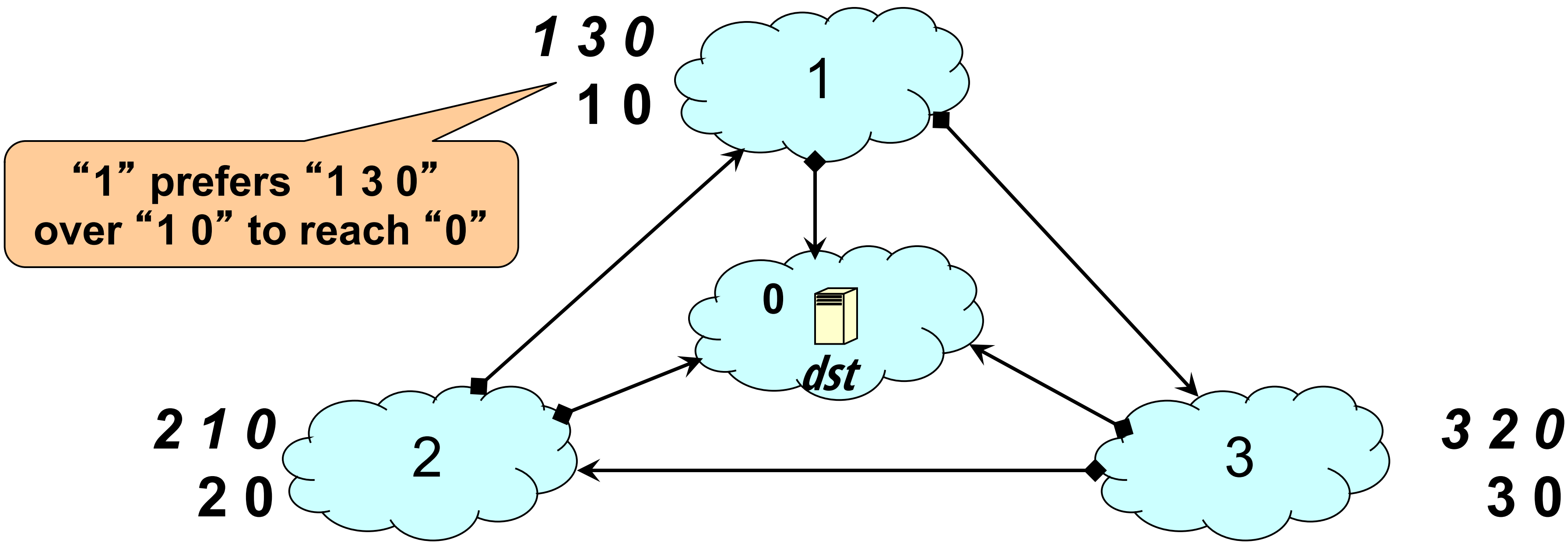
Recap

- Policy is implemented by choosing which routes we import and which ones we export
- Gao-Rexford rules tell us which routes to import/export in order to make/save money
- Good stuff happens when you follow G-R rules

Questions?

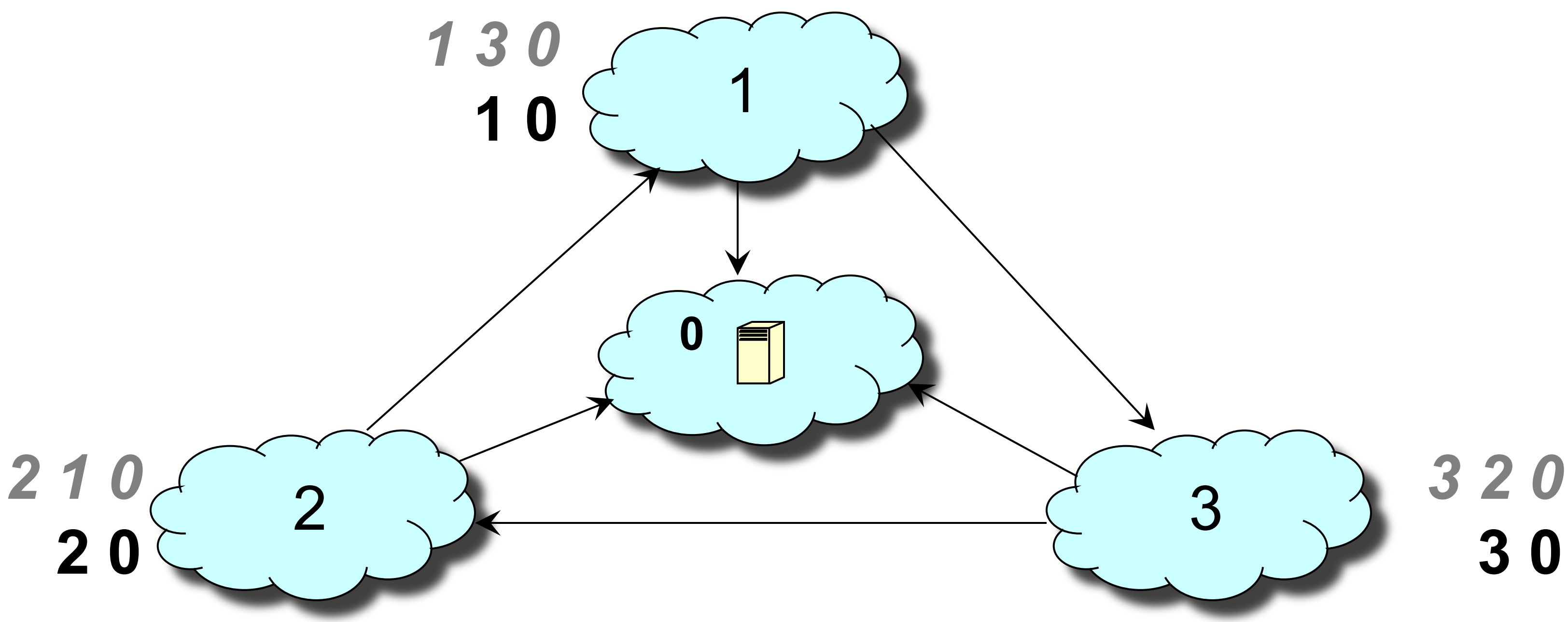
Backup

Example of Policy Oscillation (failure to converge)



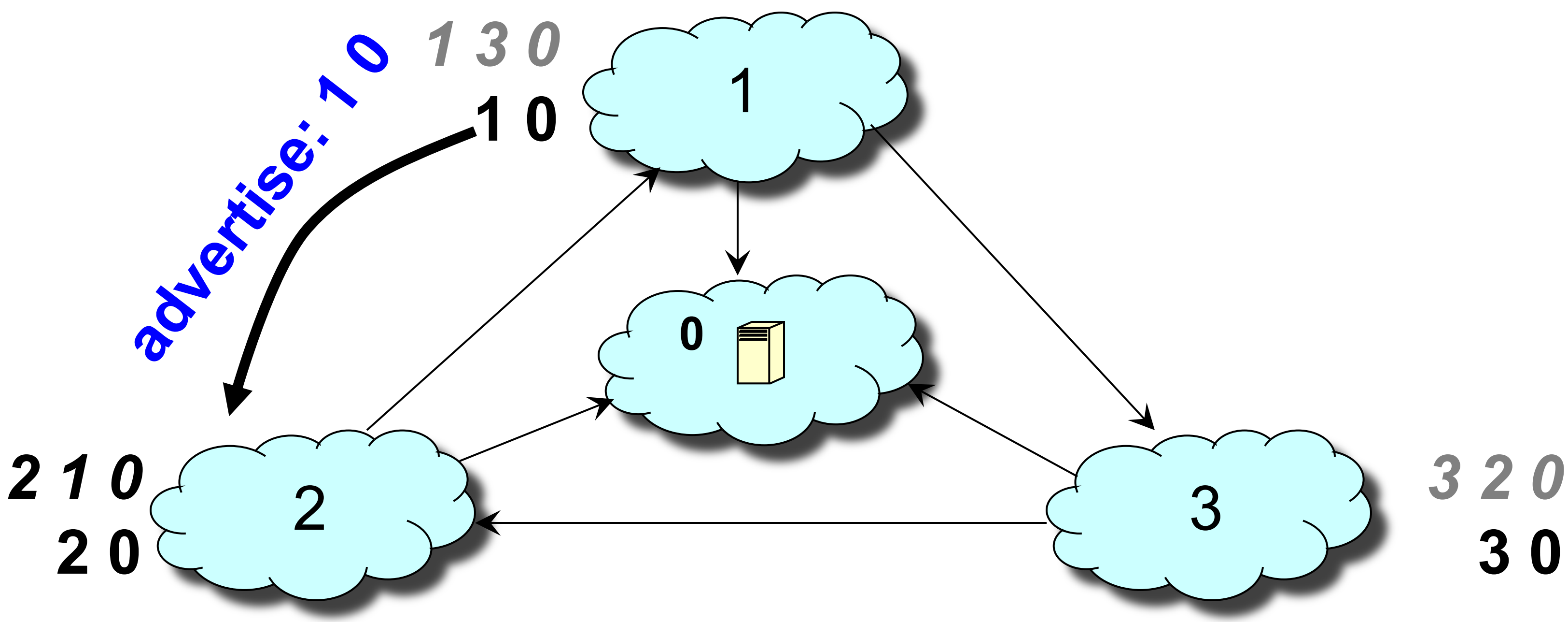
Step-by-Step of Policy Oscillation

Initially: nodes 1, 2, 3 know only shortest path to 0

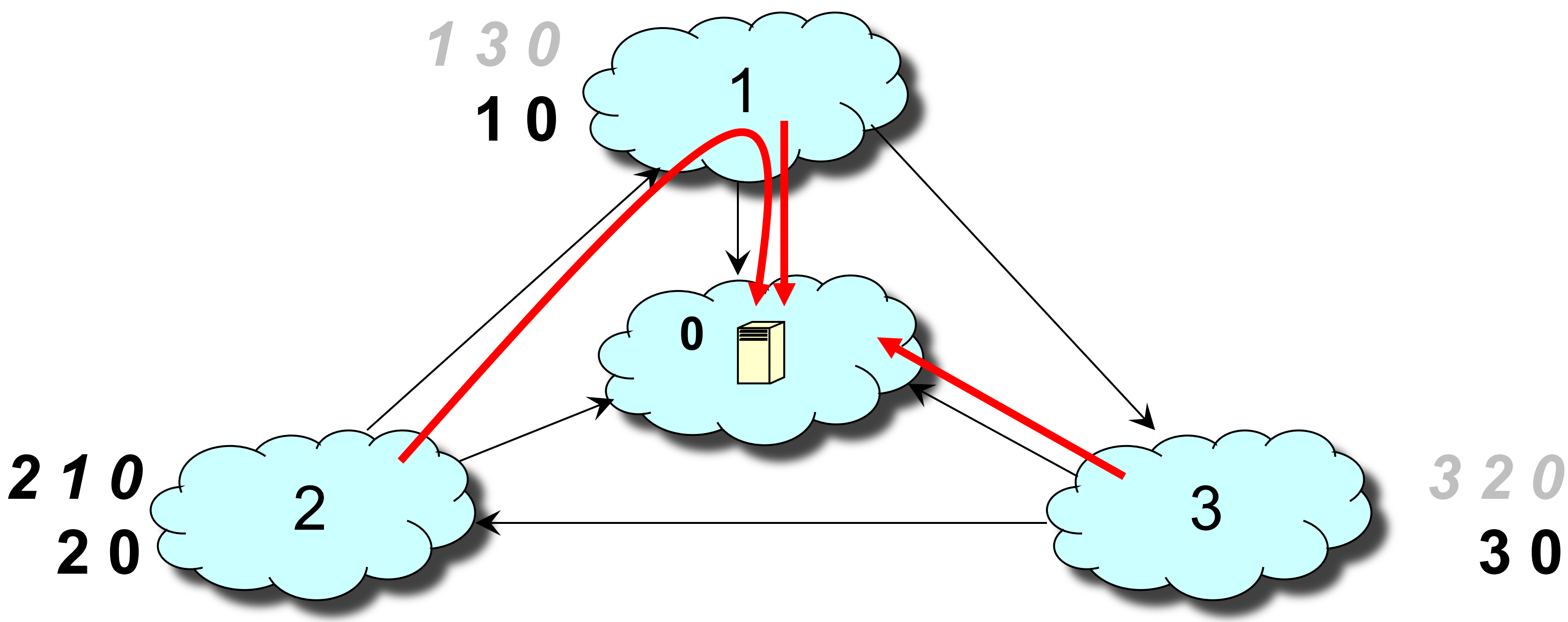


Step-by-Step of Policy Oscillation

1 **advertises** its path 1 0 to 2

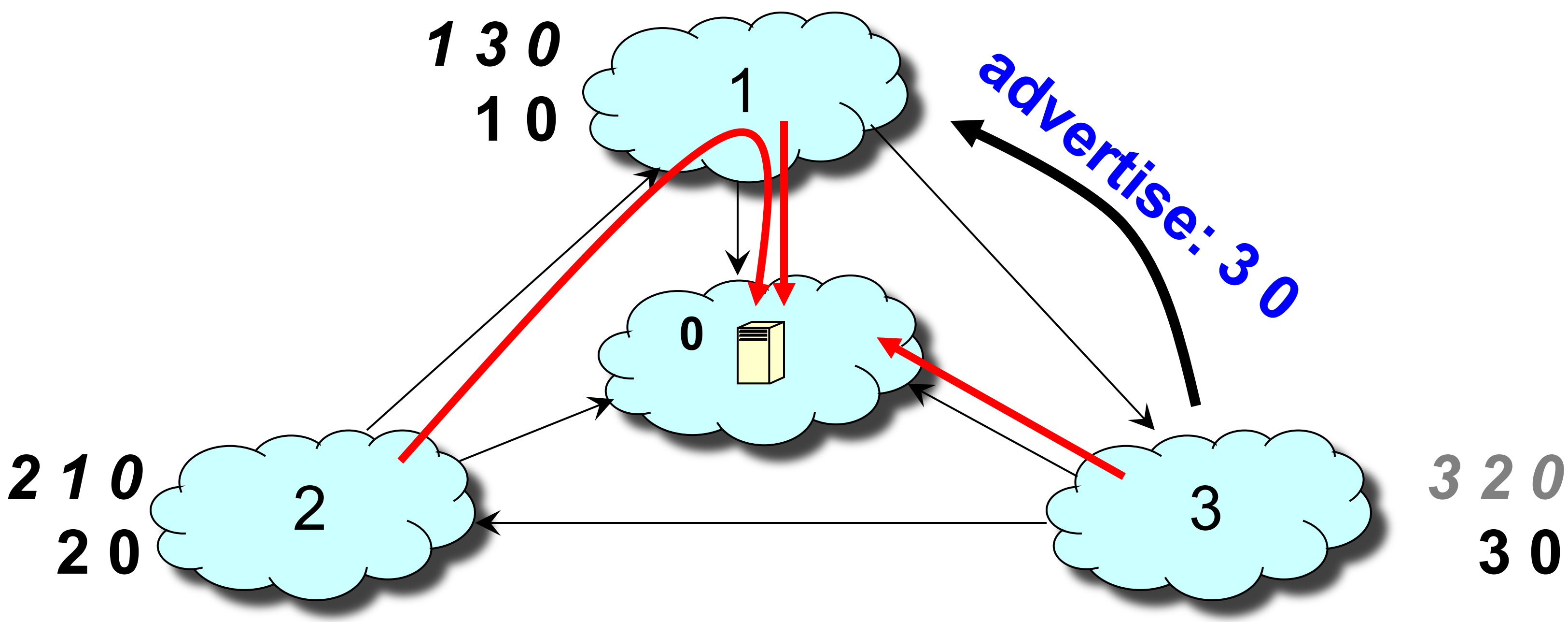


Step-by-Step of Policy Oscillation

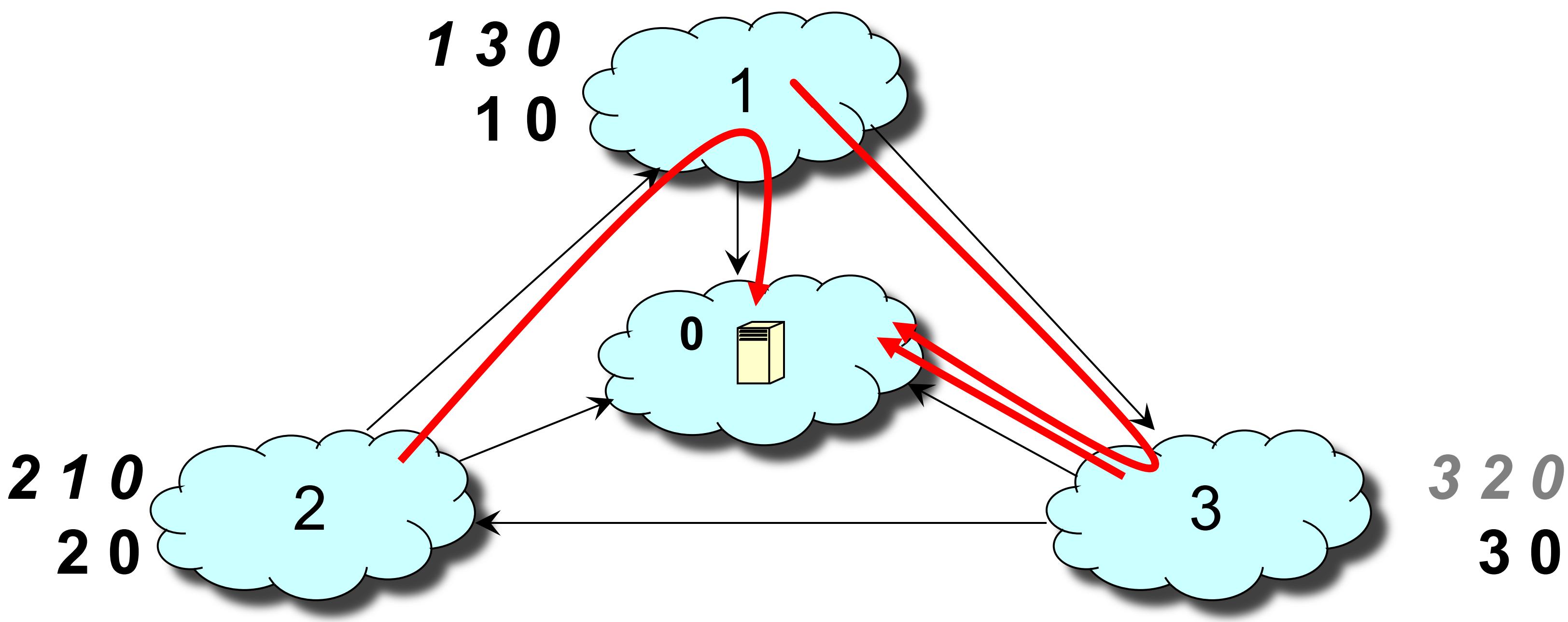


Step-by-Step of Policy Oscillation

3 **advertises** its path 3 0 to 1

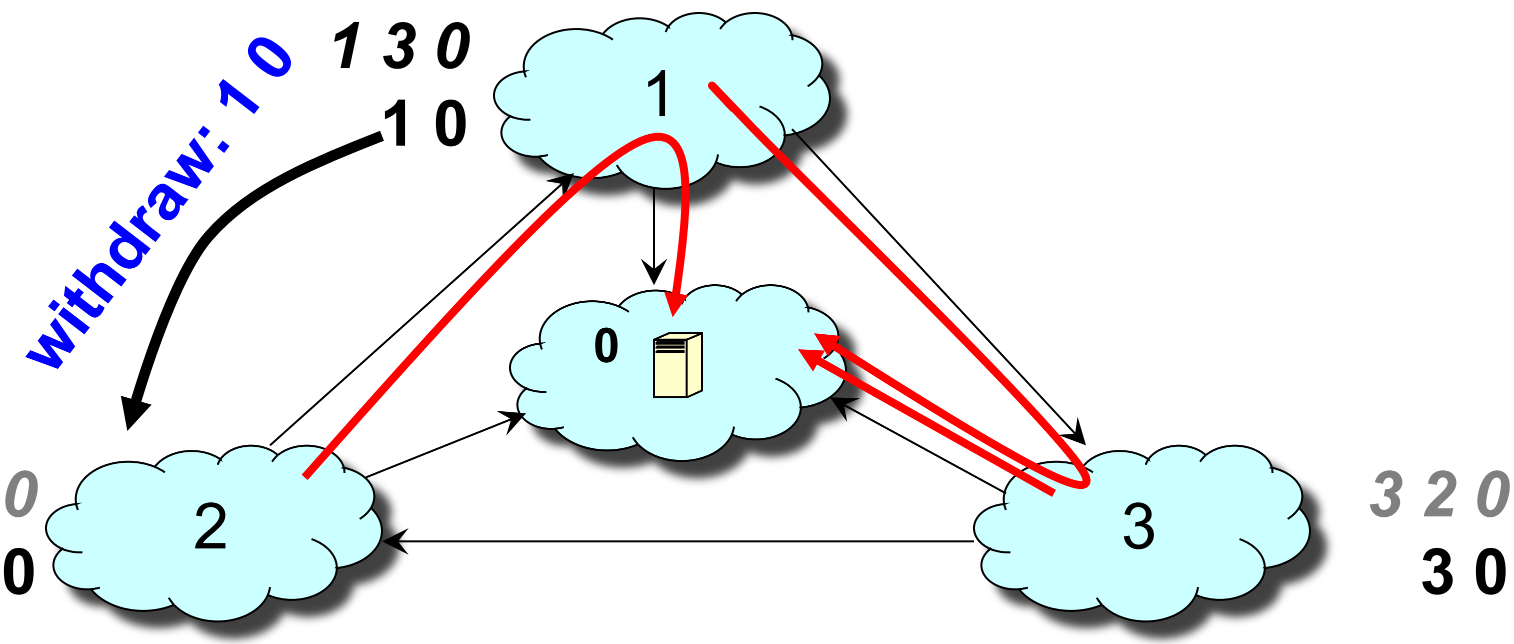


Step-by-Step of Policy Oscillation

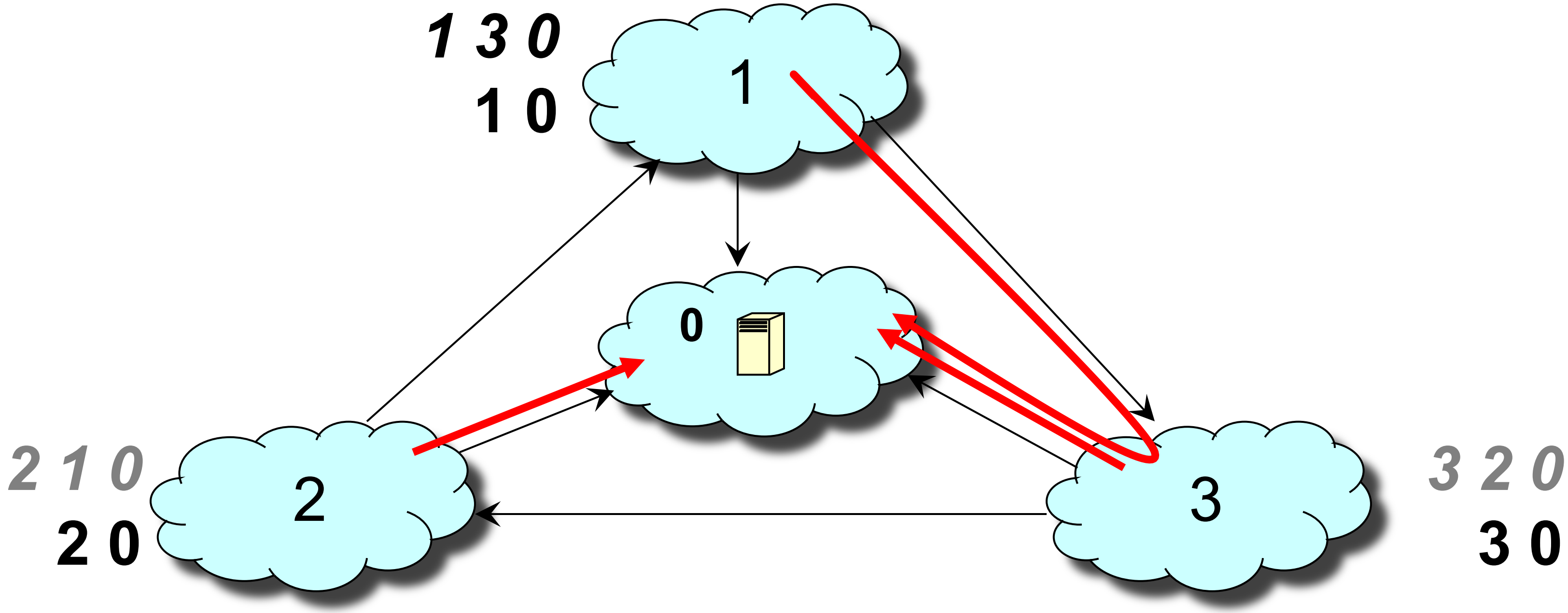


Step-by-Step of Policy Oscillation

1 **withdraws** its path 1 0 from 2

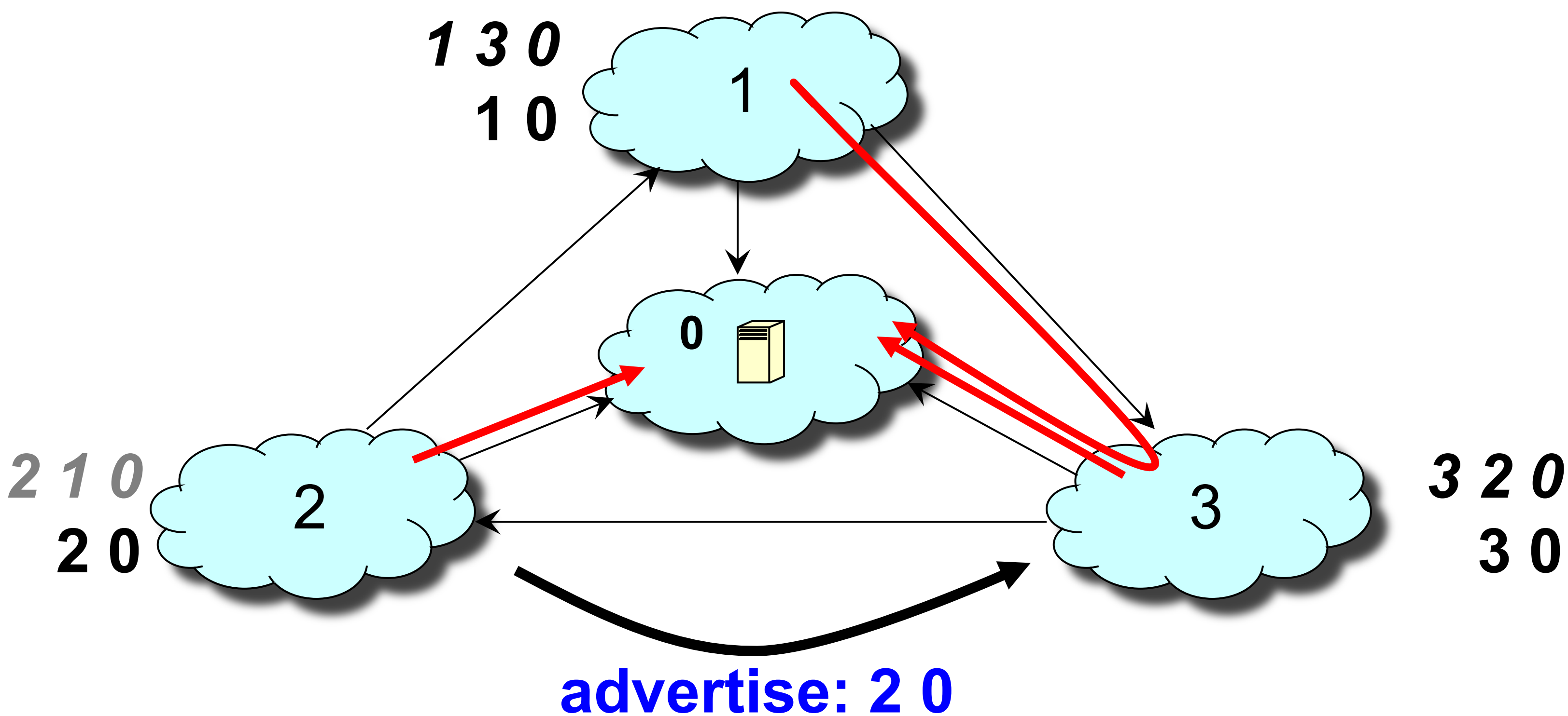


Step-by-Step of Policy Oscillation

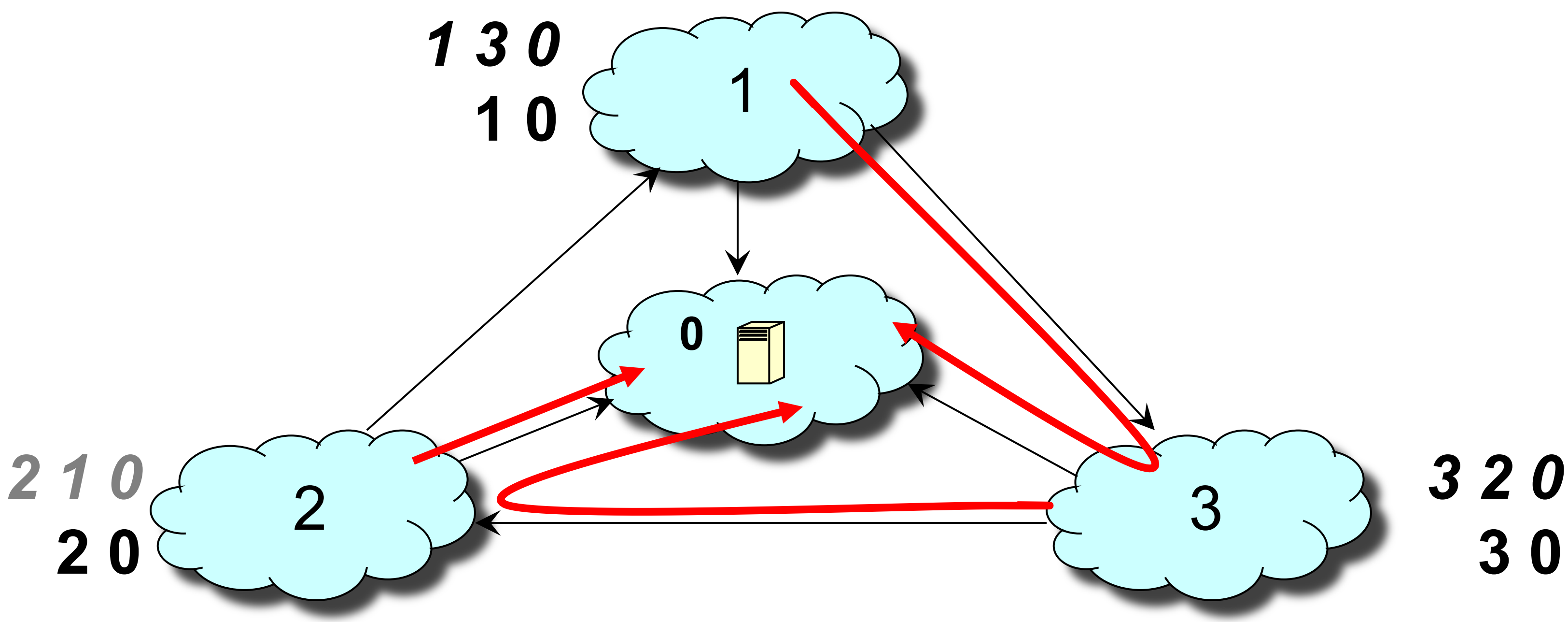


Step-by-Step of Policy Oscillation

2 **advertises** its path 2 0 to 3

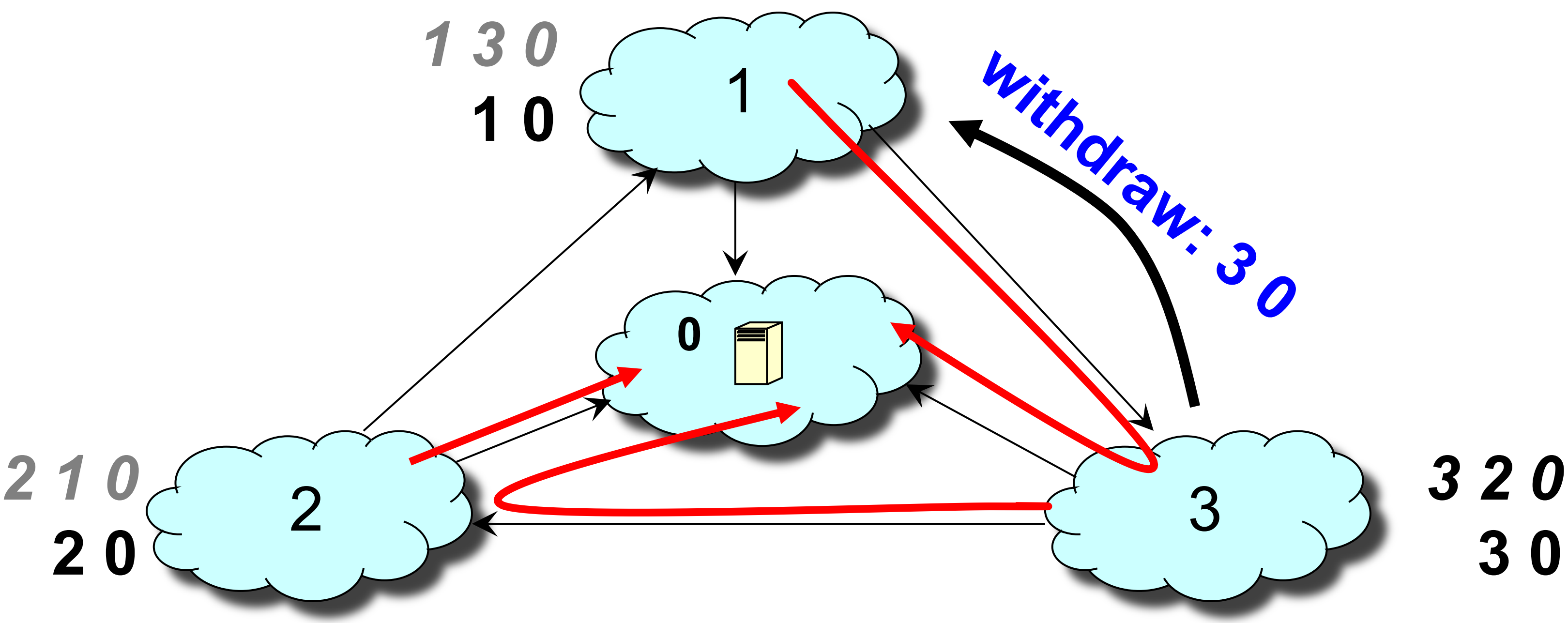


Step-by-Step of Policy Oscillation

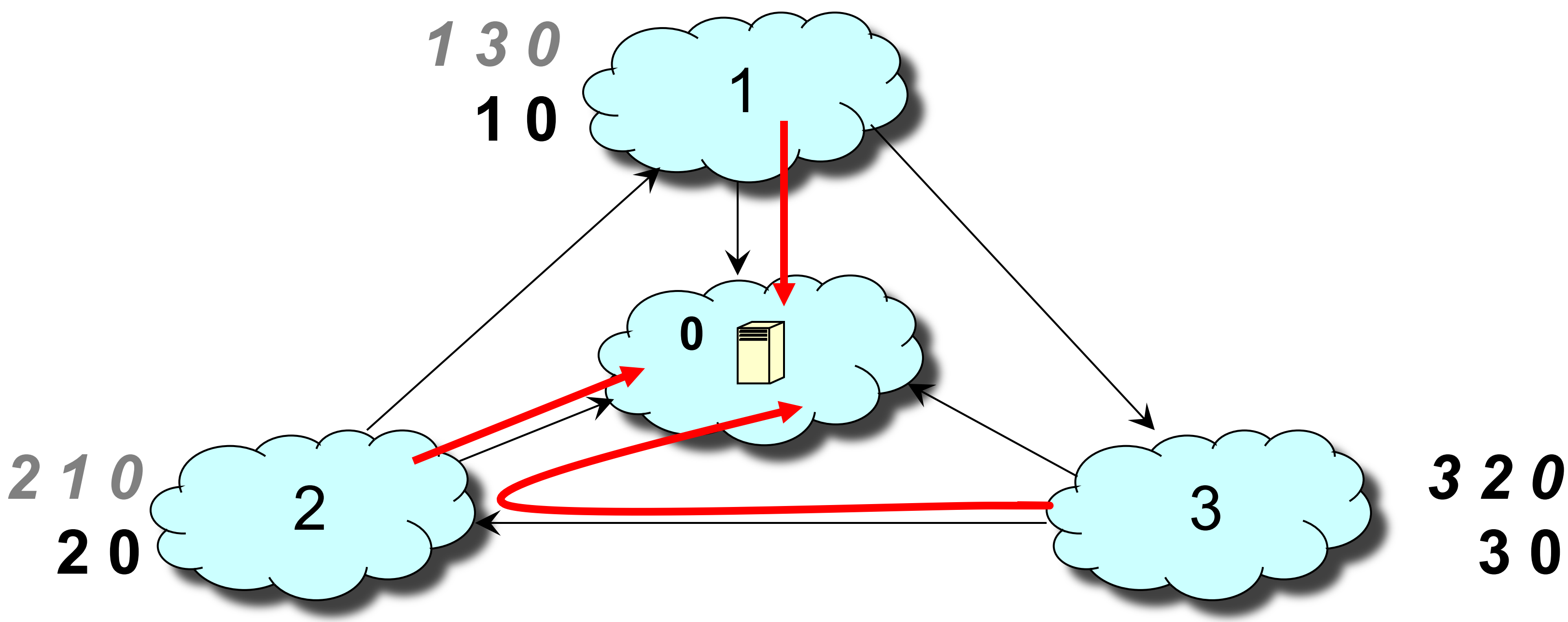


Step-by-Step of Policy Oscillation

3 **withdraws** its path 3 0 from 1

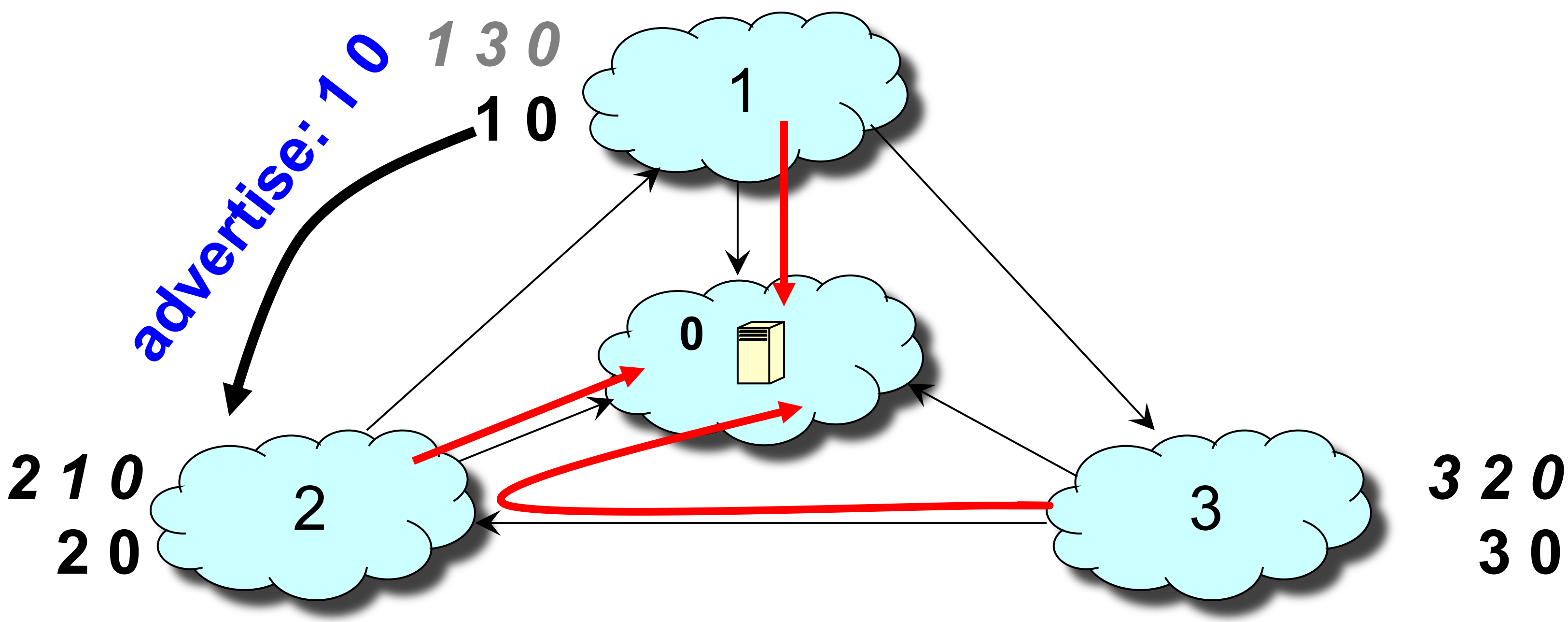


Step-by-Step of Policy Oscillation

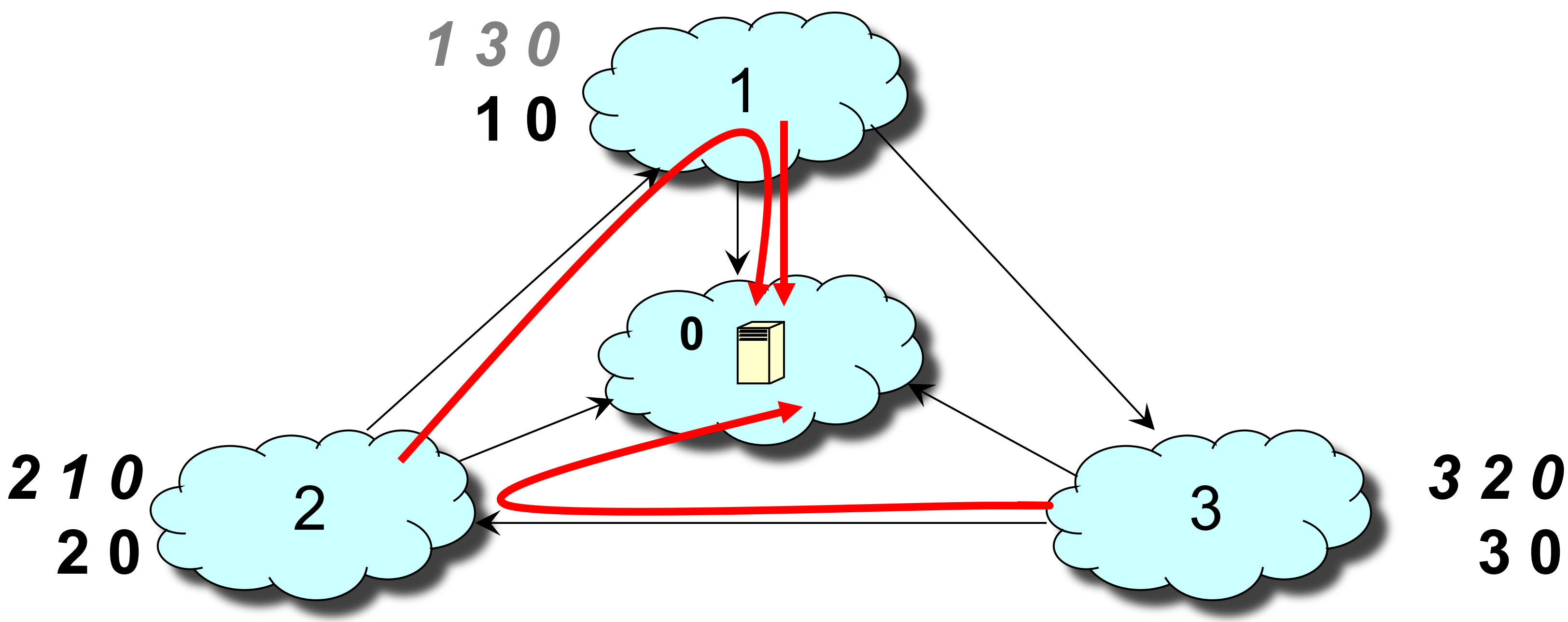


Step-by-Step of Policy Oscillation

1 **advertises** its path 1 0 to 2

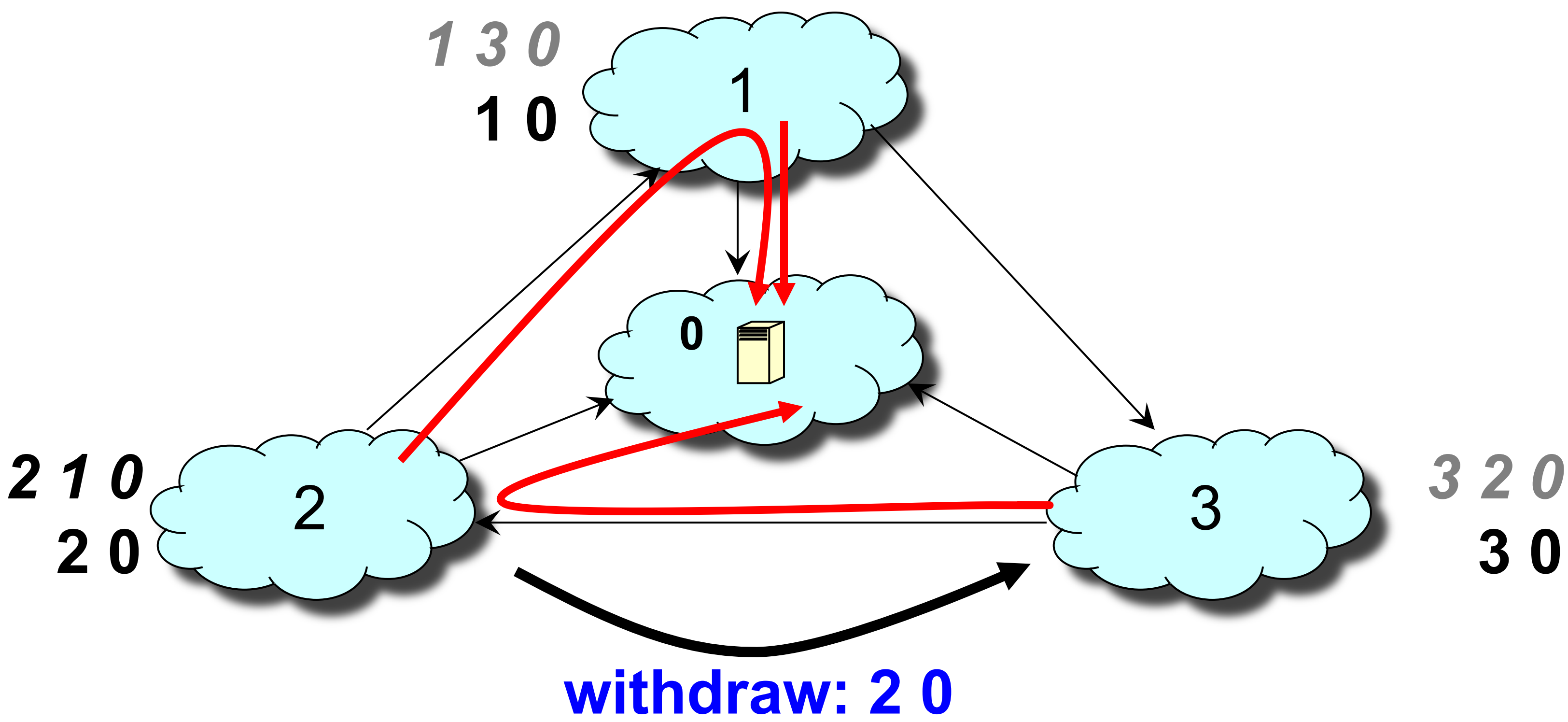


Step-by-Step of Policy Oscillation

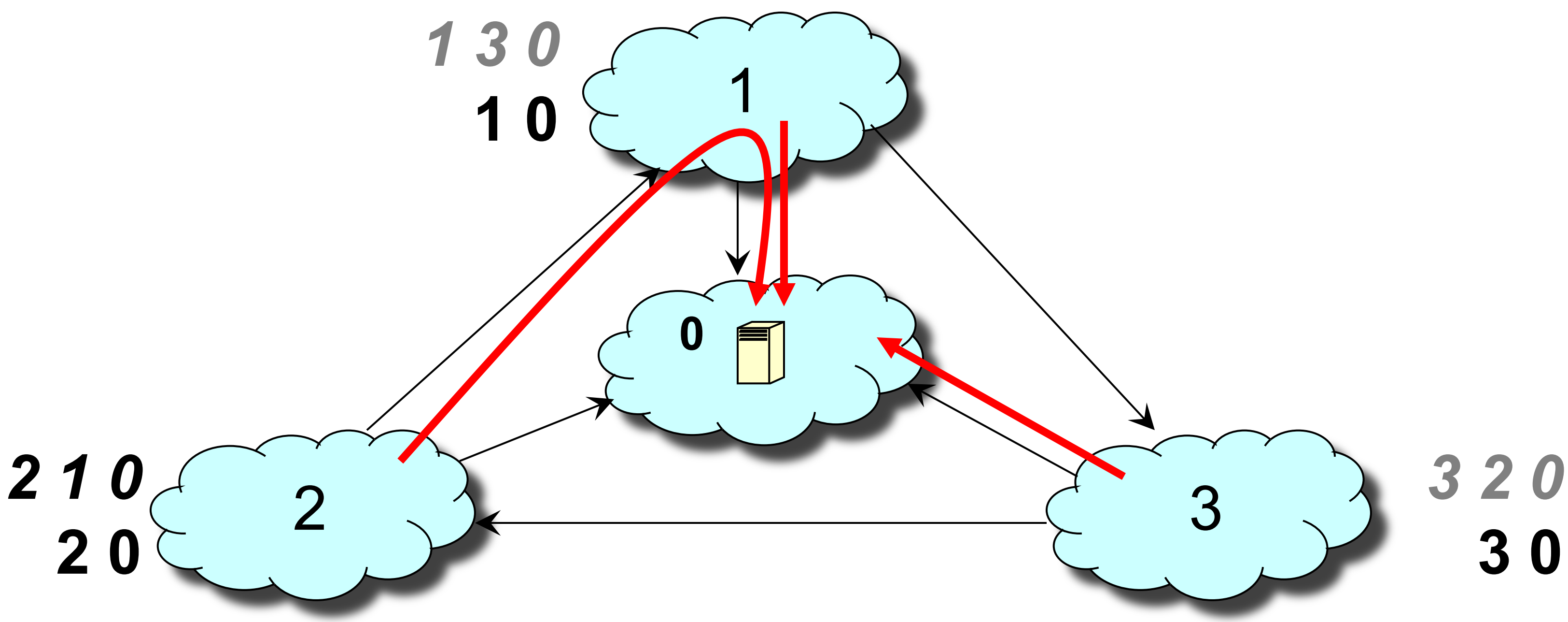


Step-by-Step of Policy Oscillation

2 **withdraws** its path 2 0 from 3



Step-by-Step of Policy Oscillation



Back to where we started! Routes never converge