Master Course
Computer Networks
IN2097

Prof. Dr.-Ing. Georg Carle
Christian Grothoff, Ph.D.
Dr. Nils Kammenhuber

Chair for Network Architectures and Services
Institut für Informatik
Technische Universität München
http://www.net.in.tum.de

Part 1

- Introduction
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP

Part 2

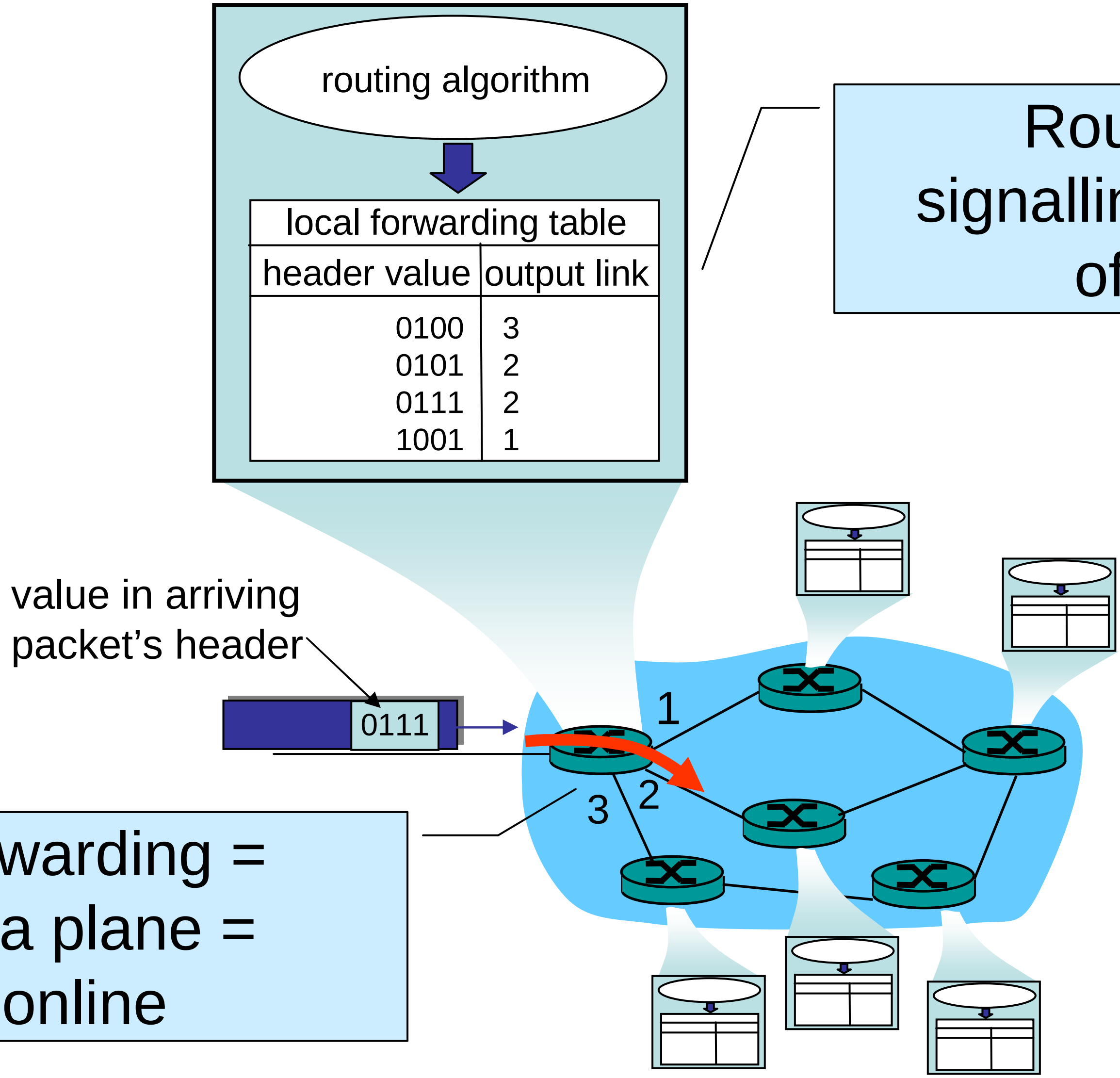
- IPv6
- NAT
- Virtual circuit and datagram networks
- What's inside a router

Part 3

- **Routing algorithms**
  - Link state
  - Distance Vector
  - Path Vector
  - Hierarchical routing
- Internet routing protocols
  - RIP
  - OSPF
  - BGP
- Business considerations
  - Policy routing
  - Traffic engineering

Short note on pronunciation of the word "routing"

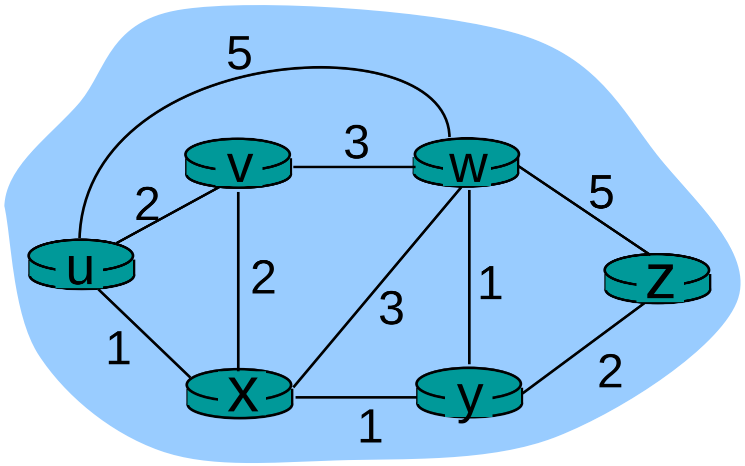
- ['ru:tɪŋ] r-oo-ting = British English
- ['raʊdɪŋ] r-ow-ding = American English
- Both are correct!



Routing = signalling plane = offline

Forwarding = data plane = online

Graph abstraction: costs



- c(x,x') =: cost of link (x,x')
  e.g.: c(w,z) = 5

- cost could always be 1,
- or inversely related to bandwidth,
- or inversely related to congestion

Cost of path $(x_1, x_2, x_3, \ldots, x_p) = c(x_1,x_2) + c(x_2,x_3) + \ldots + c(x_{p-1},x_p)$

Question: What's the least-cost path between u and z ?

Routing algorithm: algorithm that finds least-cost path

Routing Algorithm classification

Global or decentralized information?

Global:

- All routers have complete topology and link cost info
- *link state* algorithms (L-S)

Decentralized:

- Router only knows physically-connected neighbors and link costs to neighbors
- Iterative process of computation = exchange of info with neighbors
- *distance vector* algorithms (D-V)
- Variant: *path vector* algorithms

Static or dynamic?

Static:

- Routes change slowly over time

Dynamic:

- Routes change more quickly
  - periodic update
  - in response to link cost changes

Chapter 4: Network Layer

Part 1

- Introduction
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP

Part 2

- IPv6
- NAT
- Virtual circuit and datagram networks
- What's inside a router

Part 3

- **Routing algorithms**
  - **Link state**
  - Distance Vector
  - Path Vector
  - Hierarchical routing
- Internet routing protocols
  - RIP
  - OSPF
  - BGP
- Business considerations
  - Policy routing
  - Traffic engineering

A Link-State Routing Algorithm

- Net topology and link costs made known to each node
  - Accomplished via *link state broadcasts*
  - All nodes have same info
- Each node independently computes least-cost paths from one node ("source") to all other nodes
  - Usually done using Dijkstra's shortest-path algorithm
    - refer to any algorithms & data structures lecture/textbook
    - n nodes in network ⇒ $O(n^2)$ or $O(n \log n)$
  - Gives forwarding table for that node
- Result:
  - All nodes have the same information,
  - … thus calculate the same shortest paths,
  - … hence obtain consistent forwarding tables

Part 1

- Introduction
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP

Part 2

- IPv6
- NAT
- Virtual circuit and datagram networks
- What's inside a router

Part 3

- **Routing algorithms**
  - Link state
  - **Distance Vector**
  - Path Vector
  - Hierarchical routing
- Internet routing protocols
  - RIP
  - OSPF
  - BGP
- Business considerations
  - Policy routing
  - Traffic engineering

Distance Vector Algorithm

- No node knows entire topology
- Nodes only communicate with neighbours (i.e., no broadcasts)
- Nodes *jointly* calculate shortest paths
  - Iterative process
  - Algorithm == protocol
- Distributed application of Bellman-Ford algorithm
  - refer to any algorithms&data structures lecture/textbook

Distance Vector Algorithm

Bellman-Ford Equation (dynamic programming)

Let

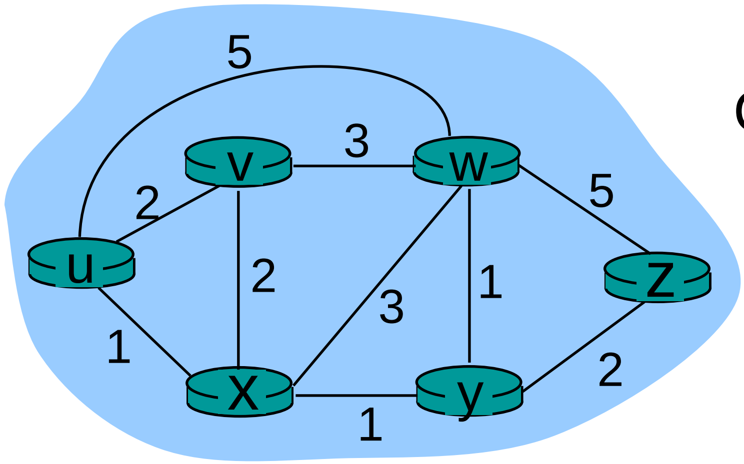
- $c(x,y)$:= cost of edge from *x* to *y*
- $d_x(y)$:= cost of least-cost path from *x* to *y*

Then

$$d_x(y) = \min \{c(x,v) + d_v(y)\}$$

where min is taken over all neighbours *v* of *x*

Bellman-Ford example



Clearly, $d_v(z) = 5$, $d_x(z) = 3$, $d_w(z) = 3$

B-F equation says:

$$d_u(z) = \min \{ c(u,v) + d_v(z),\ c(u,x) + d_x(z),\ c(u,w) + d_w(z) \}$$
$$= \min \{2 + 5,\ 1 + 3,\ 5 + 3\} = 4$$

Node that achieves minimum is next hop in shortest path → forwarding table

Distance Vector Algorithm

- Define $D_x(y)$:= estimate of least cost from *x* to *y*
- Node *x* knows cost to each neighbour *v*: $c(x,v)$
- Node *x* maintains distance vector $\mathbf{D}_x = [\ D_x(y): y \in N\ ]$
  (N := set of nodes)
- Node *x* also maintains its neighbours' distance vectors:
  - For each neighbour *v*,
    x maintains $\mathbf{D}_v = [\ D_v(y): y \in N\ ]$

Distance vector algorithm (4)

Basic idea:

- From time-to-time, each node sends its own distance vector estimate D to neighbors
  - Asynchronously
- When a node x receives new DV estimate from neighbour, it updates its own DV using B-F equation:

$$D_x(y) \leftarrow \min_v\{c(x,v) + D_v(y)\} \quad \text{for each node } y \in N$$

- Under minor, natural conditions, these estimates $D_x(y)$ *converge to the actual least cost* $d_x(y)$

Distance Vector Algorithm (5)

Iterative, asynchronous:
each local iteration caused by:

- local link cost change
- DV update message from neighbour

Distributed:

- Each node notifies neighbors *only* when its DV changes
    - neighbours then notify their neighbours if this caused *their* DV to change
    - etc.

Each node:

Forever:

wait for (change in local link cost *or* message arriving from neighbour}

recompute estimates

if (DV to any destination has changed) { *notify* neighbours }

Distance Vector Algorithm (6)

node x table

from \ cost to	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

$D_x(y) = \min\{c(x,y) + D_y(y), c(x,z) + D_z(y)\}$
$= \min\{2+0 , 7+1\} = 2$

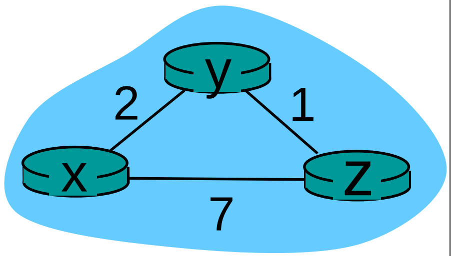
$D_x(z) = \min\{c(x,y) + D_y(z), c(x,z) + D_z(z)\}$
$= \min\{2+1 , 7+0\} = 3$

node y table

from \ cost to	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

node z table

from \ cost to	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0



time

$D_x(y) = \min\{c(x,y) + D_y(y), c(x,z) + D_z(y)\}$
$= \min\{2+0 , 7+1\} = 2$

$D_x(z) = \min\{c(x,y) + D_y(z), c(x,z) + D_z(z)\}$
$= \min\{2+1 , 7+0\} = 3$

node x table

from \ cost to	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

node y table

from \ cost to	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

from \ cost to	x	y	z
x	0	2	7
y	2	0	1
z	7	1	0

from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

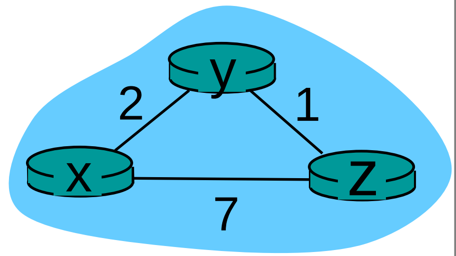
node z table

from \ cost to	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

from \ cost to	x	y	z
x	0	2	7
y	2	0	1
z	3	1	0

from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

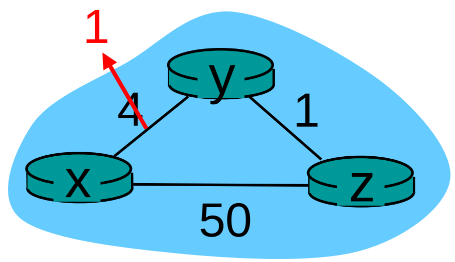
time



Distance Vector: link cost changes (1)

Link cost changes:

- node detects local link cost change
- updates routing info, recalculates distance vector
- if DV changes, notify neighbors



"good news travels fast"

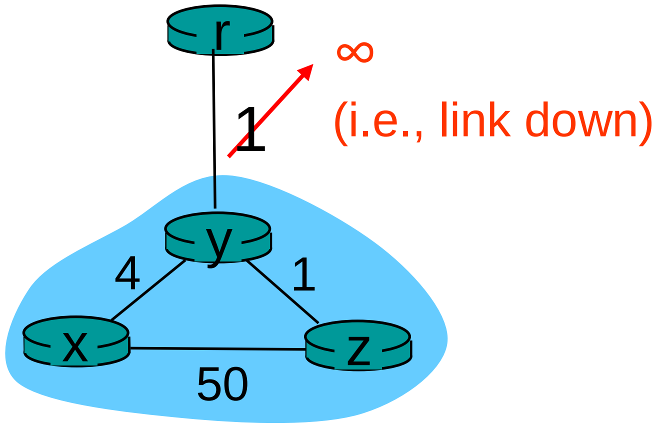
At time t_0, *y* detects the link-cost change, updates its DV, and informs its neighbors.

At time t_1, *z* receives the update from *y* and updates its table. It computes a new least cost to *x* and sends its neighbors its DV.

At time t_2, *y* receives *z*'s update and updates its distance table. *y*'s least costs do not change and hence *y* does *not* send any message to *z*.

Distance Vector: link cost changes (2)

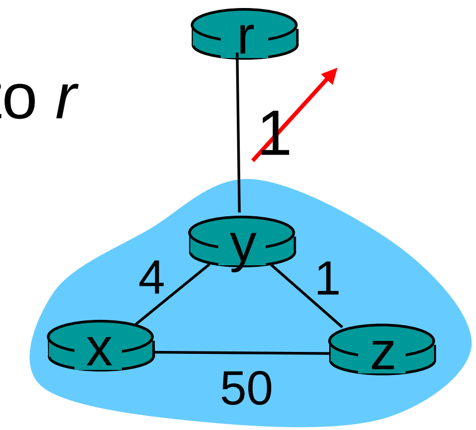
- But: bad news travels slow — "count to infinity" problem!
- In example: Many iterations before algorithm stabilizes!
  1. Cost increase for *y*→*r*:
     - *y* consults DV,
     - *y* selects "cheaper" route via *z* (cost 2+1 = 3),
     - Sends update to *z* and *x* (cost to *r* now 3 instead of 1)



  2. *z* detects cost increase for path to *r*:
     - was 1+1, is now 3+1
     - Sends update to *y* and *x* (cost to *r* now 4 instead of 2)
  3. *y* detects cost increase, sends update to *z*
  4. *z* detects cost increase, sends update to *y*
  5. ….

Distance Vector: Solutions that only half work

- Finite infinity: Define some number to be ∞ (in RIP: 16 := ∞)
- Split Horizon:
  - Tell to a neighbour that is part of a best path to a destination that the destination cannot be reached
  - If *z* routes through *y* to get to *r*
    z tells *y* that its own (i.e., *y*'s) distance to *r* is infinite (so *y* won't route to *r* via *z*)



- Poisoned Reverse:
  - In addition, *actively* advertise a route as unreachable to the neighbour from which the route was learned
- (Warning: Terms often used interchangeably!)
- Often help, but cannot solve all problem instances
- Can significantly increase number of routing messages

Comparison of LS and DV algorithms

Message complexity

- LS: with *n* nodes, *E* links, O(*nE*) msgs sent
- DV: exchange between neighbors only
  - convergence time varies

Speed of Convergence

- LS: $O(n^2)$ algorithm requires O(nE) msgs
  - may have oscillations
- DV: convergence time varies
  - may be routing loops
  - count-to-infinity problem

Robustness: what happens if router malfunctions?

LS:

- node can advertise incorrect *link* cost
- each node computes only its *own* table

DV:

- DV node can advertise incorrect *path* cost
- each node's table used by others
  - error propagate thru network

Chapter 4: Network Layer

Part 1

- Introduction
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP

Part 2

- IPv6
- NAT
- Virtual circuit and datagram networks
- What's inside a router

Part 3

- **Routing algorithms**
  - Link state
  - Distance Vector
  - **Path Vector**
  - Hierarchical routing
- Internet routing protocols
  - RIP
  - OSPF
  - BGP
- Business considerations
  - Policy routing
  - Traffic engineering

Path Vector protocols

- Problem with D-V protocol:
  Path cost is “anonymous” single number
- Path Vector protocol:
  - For each destination, advertise entire path (=sequence of node identifiers) to neighbours
  - Cost calculation can be done by looking at path
  - Easy loop detection: Does my node ID already appear in the path?
- Not used very often
  - only in BGP …
  - … and BGP is much more complex than just paths!

Part 1

- Introduction
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP

Part 2

- IPv6
- NAT
- Virtual circuit and datagram networks
- What's inside a router

Part 3

- **Routing algorithms**
  - Link state
  - Distance Vector
  - Path Vector
  - **Hierarchical routing**
- Internet routing protocols
  - RIP
  - OSPF
  - BGP
- Business considerations
  - Policy routing
  - Traffic engineering

Hierarchical Routing

Our routing study thus far = idealisation

- all routers identical
- network “flat”

… *not* true in practice!

Scale = billions of destinations:

- Can’t store all destinations in routing tables!
- Routing table exchange would swamp links!

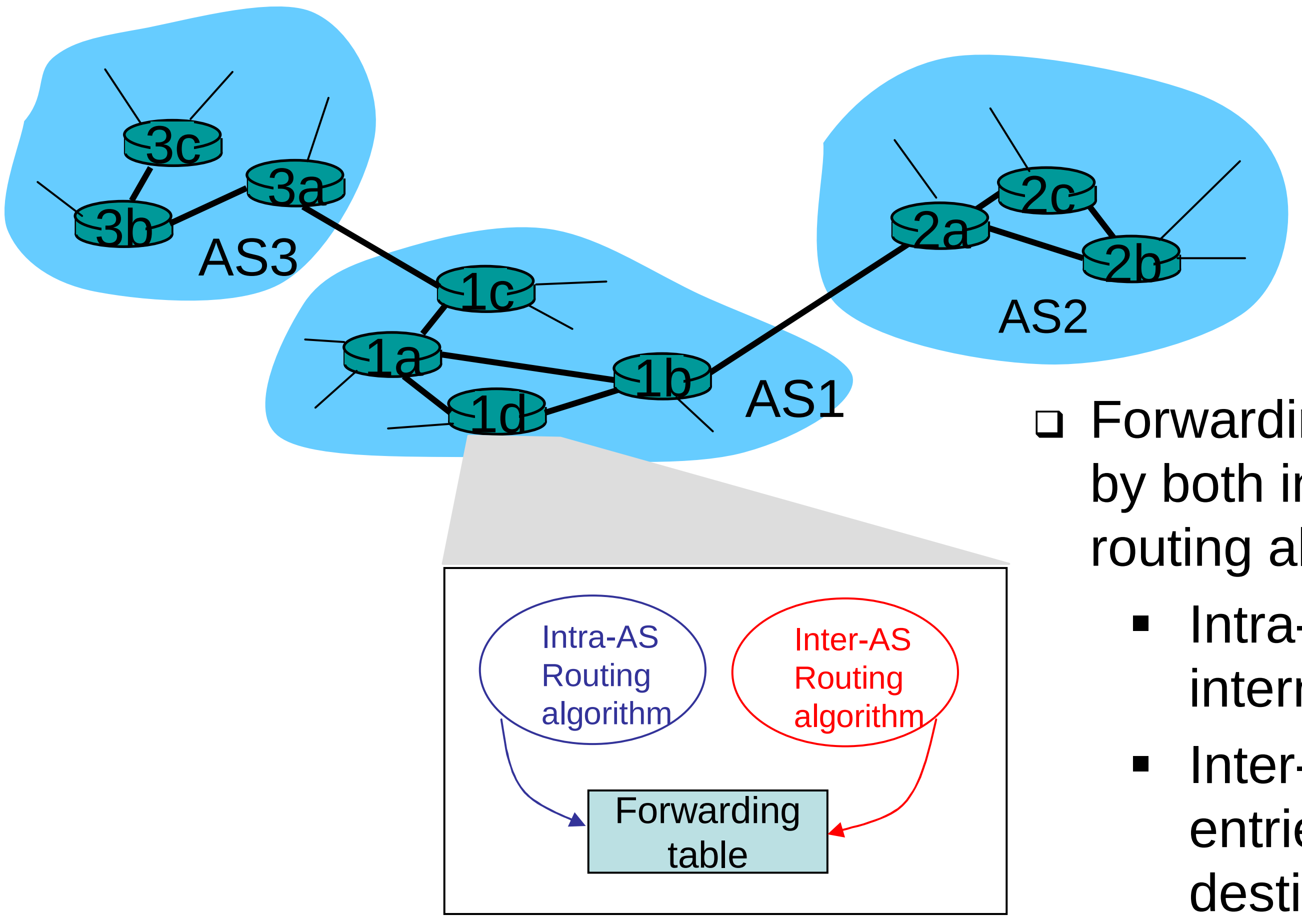
Administrative autonomy

- Internet = network of networks
- Each network admin may want to control routing in its own network — no central administration!

Hierarchical Routing

- Aggregate routers into regions called "autonomous systems" (short: AS; plural: ASes)
- Routers in same AS run same routing protocol
  - = "intr**a**-AS" routing protocol (also called "intradomain")
  - Routers in different ASes can run different intra-AS routing protocols
- ASes are connected: via gateway routers
  - Direct link to [gateway] router in another AS = "int**er**-AS" routing protocol (also called "interdomain")
  - Warning: Non-gateway routers need to know about inter-AS routing as well!

Interconnected ASes



- Forwarding table configured by both intra- *and* inter-AS routing algorithm:
  - Intra-AS sets entries for internal destinations
  - Inter-AS *and* intra-AS set entries for external destinations

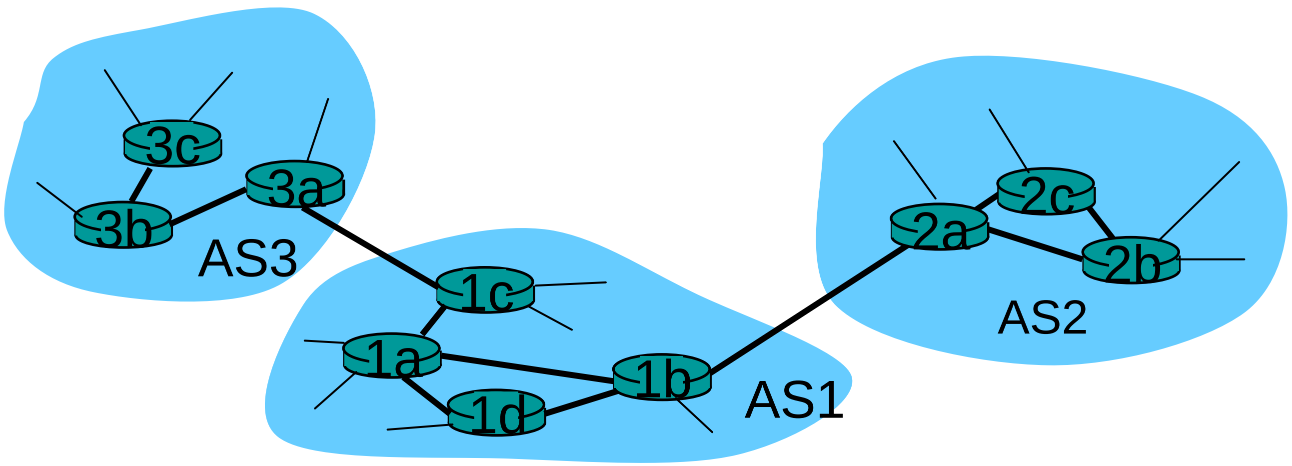
Inter-AS tasks

- Suppose router in AS1 receives datagram destined outside of AS1:
  - router should forward packet to gateway router, but which one?

<u>AS1 must:</u>

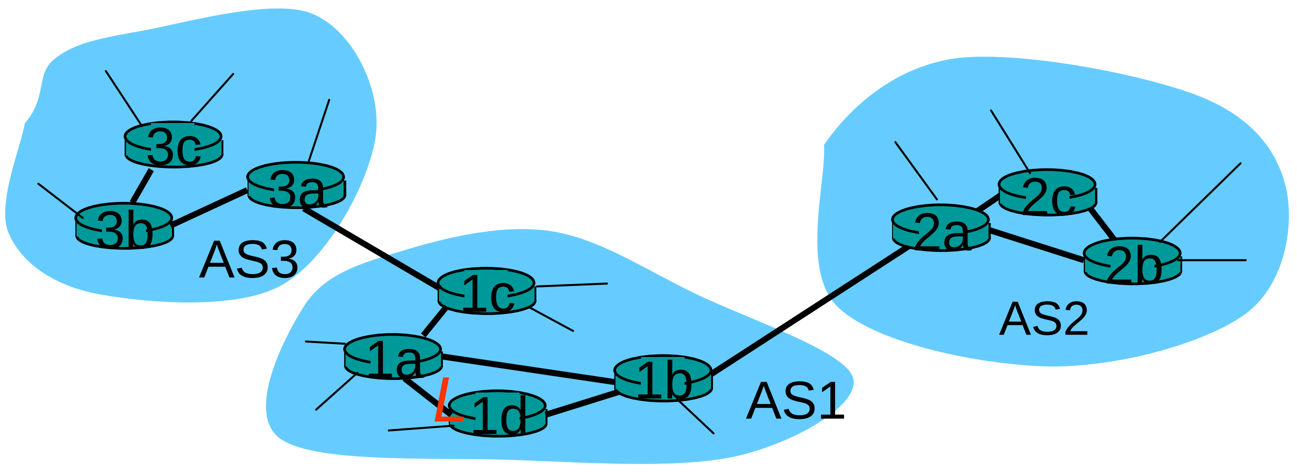
1. learn which dests are reachable through AS2, which through AS3
2. propagate this reachability info *to all* routers in AS1 (i.e., not just the gateway routers)

Job of inter-AS routing!



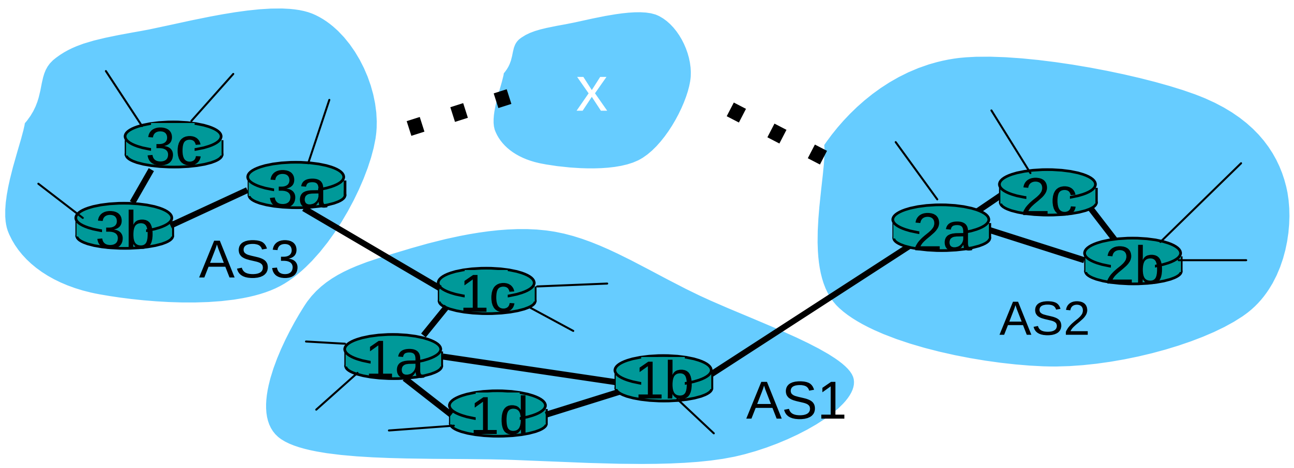
Example: Setting forwarding table in router 1d

- Suppose AS1 learns (via inter-AS protocol) that subnet *x* is reachable via AS3 (gateway 1c) but not via AS2.
- Inter-AS protocol propagates reachability info to all internal routers.
- Router 1d determines from intra-AS routing info that its interface *I* (i.e., interface to 1a) is on the least cost path to 1c.
  - installs forwarding table entry *(x,I)*



Example: Choosing among multiple ASes

- Now suppose AS1 learns from inter-AS protocol that subnet *x* is reachable from AS3 *and* from AS2.
- To configure forwarding table, router 1d must determine towards which gateway it should forward packets for destination x.
  - This is also job of inter-AS routing protocol!



Interplay of inter-AS and intra-AS routing

- Inter-AS routing
  - Only for destinations outside of own AS
  - Used to determine gateway router
  - Also: Steers transit traffic
    (from AS *x* to AS *y* via our own AS)
- Intra-AS routing
  - Used for destinations within own AS
  - Used to reach gateway router for outside destinations

Part 1

- Introduction
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP

Part 2

- IPv6
- NAT
- Virtual circuit and datagram networks
- What's inside a router

Part 3

- Routing algorithms
  - Link state
  - Distance Vector
  - Path Vector
  - Hierarchical routing
- **Internet routing protocols**
  - (RIP)
  - OSPF
  - BGP
- Business considerations
  - Policy routing
  - Traffic engineering

Recall: Inter-AS and intra-AS routing

- Inter-AS routing
  - Only for destinations outside of own AS
  - Used to determine gateway router
  - Also: Steers transit traffic (from AS *x* to AS *y* via our own AS)
- Intra-AS routing
  - Used for destinations within own AS
  - Used to reach gateway router for outside destinations

⇒ Routers need to run *both* types of routing protocols

Intra-AS Routing

- Also known as Interior Gateway Protocols (IGP)
- Most common Intra-AS routing protocols:
  - RIP: Routing Information Protocol — DV (typically small systems)
  - OSPF: Open Shortest Path First — hierarchical LS (typically medium to large systems)
  - IS-IS: Intermediate System to Intermediate System — hierarchical LS (typically medium-sized ASes)
  - (E)IGRP: (Enhanced) Interior Gateway Routing Protocol (Cisco proprietary) — hybrid of LS and DV

Chapter 4: Network Layer

Part 1

- Introduction
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP

Part 2

- IPv6
- NAT
- Virtual circuit and datagram networks
- What's inside a router

Part 3

- Routing algorithms
  - Link state
  - Distance Vector
  - Path Vector
  - Hierarchical routing
- **Internet routing protocols**
  - (RIP)
  - **OSPF**
  - BGP
- Business considerations
  - Policy routing
  - Traffic engineering

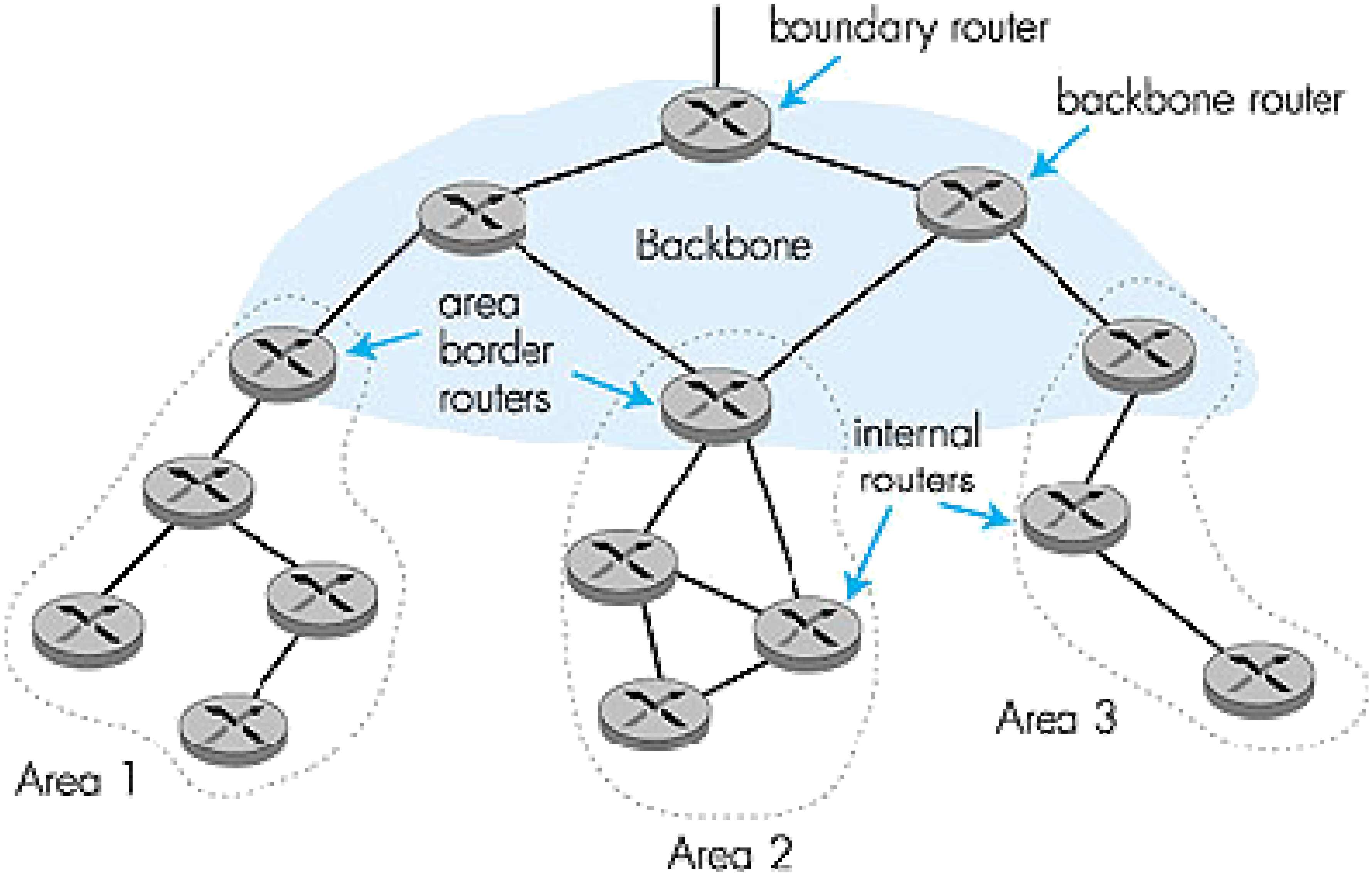
OSPF (Open Shortest Path First)

- "Open": publicly available (vs. vendor-specific, e.g., EIGRP = Cisco-proprietary)
- Uses Link State algorithm
  - LS packet dissemination (broadcasts)
  - Unidirectional edges ($\Rightarrow$costs may differ by direction)
  - Topology map at each node
  - Route computation using Dijkstra's algorithm
- OSPF advertisement carries one entry per neighbour router
- Advertisements disseminated to entire AS (via flooding)
  - (exception: hierarchical OSPF, see next slides)
  - carried in OSPF messages directly over IP (rather than TCP or UDP)

OSPF "advanced" features

- Security: all OSPF messages authenticated (to prevent malicious intrusion)
- Multiple same-cost paths allowed (only one path in RIP): *ECMP* (equal-cost multipath)
- For each link, multiple cost metrics for different Type of Service (TOS):
  e.g., satellite link cost set "low" for best effort, but high for real time
- Integrated uni- and multicast support:
  - Multicast OSPF (MOSPF) uses same topology data base as OSPF
- Hierarchical OSPF in large domains

Hierarchical OSPF



Hierarchical OSPF

- OSPF *can* create a two-level hierarchy *within* an AS
  - Similar to inter-AS and intra-AS routing in Internet
- Two levels: local *areas* and the *backbone*
  - Link-state advertisements only within local area
  - Each node has detailed area topology; but only knows direction (shortest path) to networks in other areas
- *Area border routers:* "summarize" distances to networks in own area; advertise distances to other Area Border routers
- *Backbone routers:* run OSPF routing limited to backbone
- *Boundary routers:* connect to other ASes

Chapter 4: Network Layer

Part 1

- Introduction
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP

Part 2

- IPv6
- NAT
- Virtual circuit and datagram networks
- What's inside a router

Part 3

- Routing algorithms
  - Link state
  - Distance Vector
  - Path Vector
  - Hierarchical routing
- **Internet routing protocols**
  - RIP
  - OSPF
  - **BGP**
- Business considerations
  - Policy routing
  - Traffic engineering

Internet inter-AS routing: BGP

- BGP (Border Gateway Protocol):
  The de facto standard for inter-AS routing
- BGP provides each AS a means to:
  1. Obtain subnet reachability information from neighbouring ASes.
  2. Propagate reachability information to all AS-internal routers.
  3. Determine "good" routes to subnets based on reachability information and policy.
- Allows an AS to advertise the existence of an IP prefix to rest of Internet: *"This subnet is here"*

BGP basics

- Pairs of routers (BGP peers) exchange routing info over semi-permanent TCP connections: BGP sessions
  - BGP sessions need not correspond to physical links!
- When AS2 advertises an IP prefix to AS1:
  - AS2 *promises* it will forward IP packets towards that prefix
  - AS2 can aggregate prefixes in its advertisement

How does BGP work?

- BGP = "path++" vector protocol
- BGP messages exchanged using TCP
  - Possible to run eBGP sessions not on border routers
- BGP Message types:
  - OPEN: set up new BGP session, after TCP handshake
  - NOTIFICATION: an error occurred in previous message → tear down BGP session, close TCP connection
  - KEEPALIVE: "null" data to prevent TCP timeout/auto-close; also used to acknowledge OPEN message
  - UPDATE:
    - Announcement: inform peer about new / changed route to some target
    - Withdrawal: (inform peer about non-reachability of a target)

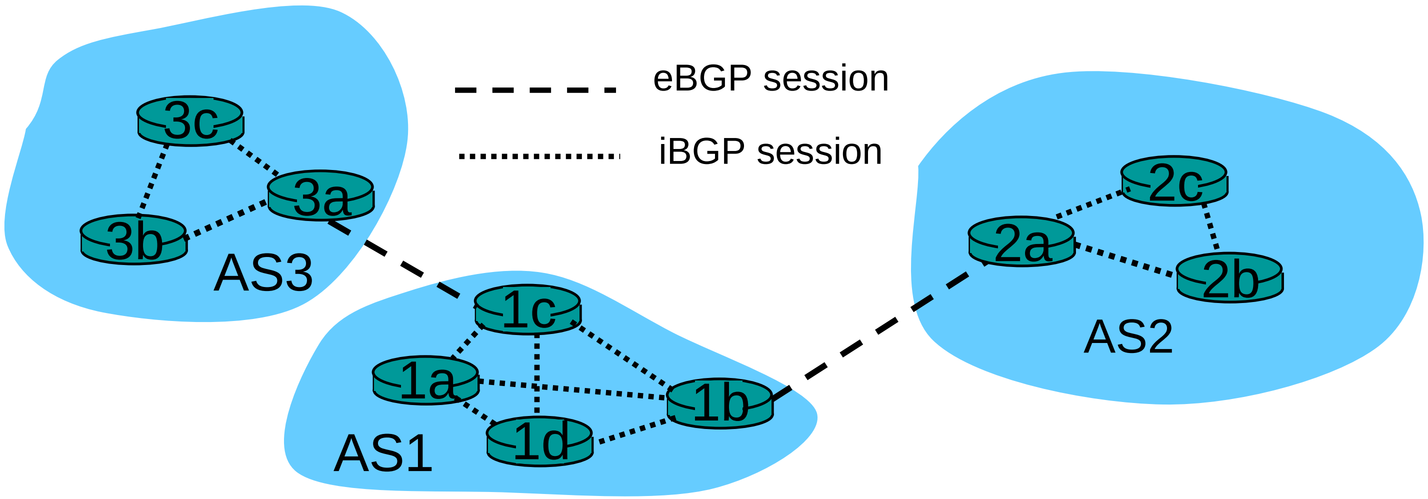
BGP updates

- Update (Announcement) message consists of
  - Destination (IP prefix)
  - AS Path (=Path vector)
  - Next hop (=IP address of our router connecting to other AS)
- …but update messages also contain a lot of further attributes:
  - Local Preference: used to prefer one gateway over another
  - Origin: route learned via { intra-AS | inter-AS | unknown }
  - MED, Community, …

⇒ Not a pure path vector protocol: More than just the path vector

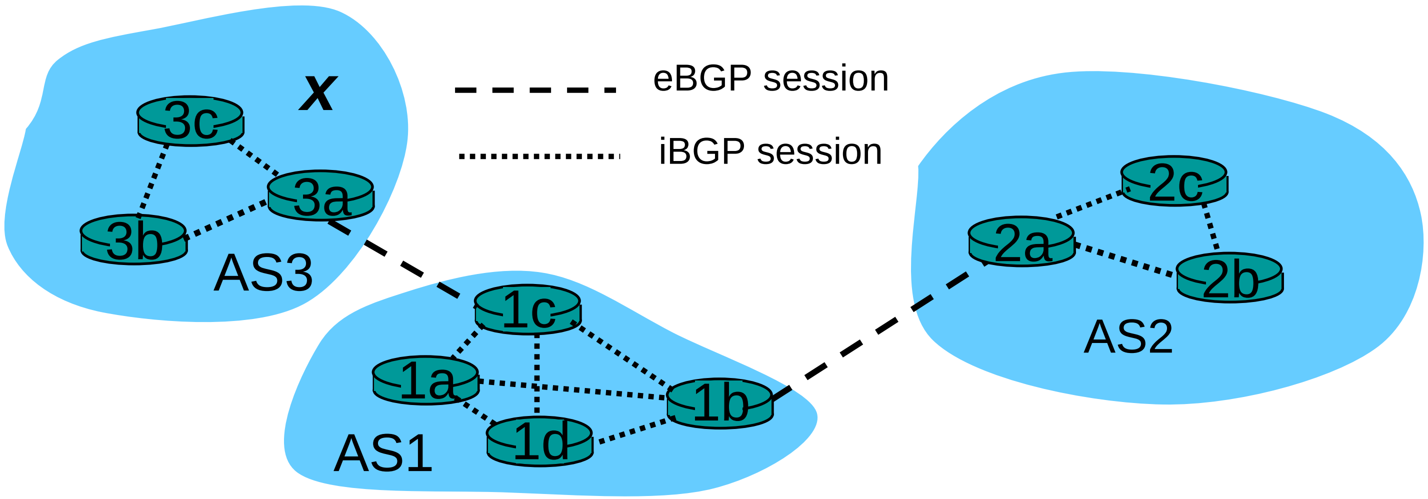
eBGP and iBGP

- External BGP: between routers in *different* ASes
- Internal BGP: between routers in *same* AS
  - Remember: In spite of intra-AS routing protocol, *all* routers need to know about external destinations (not only border routers)
- No different protocols — just slightly different configurations!



Distributing reachability info

- Using eBGP session between 3a and 1c, AS3 sends reachability info about prefix *x* to AS1.
  - 1c can then use iBGP to distribute new prefix info to all routers in AS1
  - 1b can then re-advertise new reachability info to AS2 over 1b-to-2a eBGP session
- When router learns of new prefix *x*, it creates entry for prefix in its forwarding table.



Path attributes & BGP routes

- Advertised prefix includes BGP attributes
  - prefix + attributes = “route”
- Most important attributes:
  - AS-PATH: contains ASs through which prefix advertisement has passed: e.g, AS 67, AS 17
    - ASes identified by *AS numbers*, e.g.,: lrz.de=AS12816
  - NEXT-HOP: indicates specific internal-AS router to next-hop AS. (may be multiple links from current AS to next-hop-AS)
- When gateway router receives route advertisement, it uses an import policy to accept/decline the route
  - More on this later

BGP route selection

- Router may learn about more than 1 route to some prefix
  ⇒ Router must select route.
- Elimination rules:
  1. Local preference value attribute: policy decision
  2. Shortest AS-PATH
  3. Closest NEXT-HOP router: hot potato routing
  4. Additional criteria

BGP Route Reflectors (RR)

- Every router in AS should know external routes
  - Not only local neighbours, but also neighbours connected at other routers
  - ⇒ Many/all routers in AS have to run BGP sessions
- Need to select best inter-AS routes
  - ⇒ Routers need to exchange routing information via iBGP
- O(n) BGP routers ⇒ O(n^2) iBGP sessions ↯ ↯ ↯
- **Idea:**
  - One special router = Route Reflector (RR)
  - Every **e**BGP router sends routes learned from **e**BGP via iBGP to RR
  - RR collects routes, may do policing
  - RR distributes routes to all other BGP routers in AS via iBGP
  - O(n) BGP routers, O(n) BGP sessions ☺

Chapter 4: Network Layer

Part 1

- Introduction
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP

Part 2

- IPv6
- NAT
- Virtual circuit and datagram networks
- What's inside a router

Part 3

- Routing algorithms
  - Link state
  - Distance Vector
  - Path Vector
  - Hierarchical routing
- Internet routing protocols
  - (RIP)
  - OSPF
  - BGP
- **Business considerations**
  - **Policy routing**
  - Traffic engineering

Business relationships

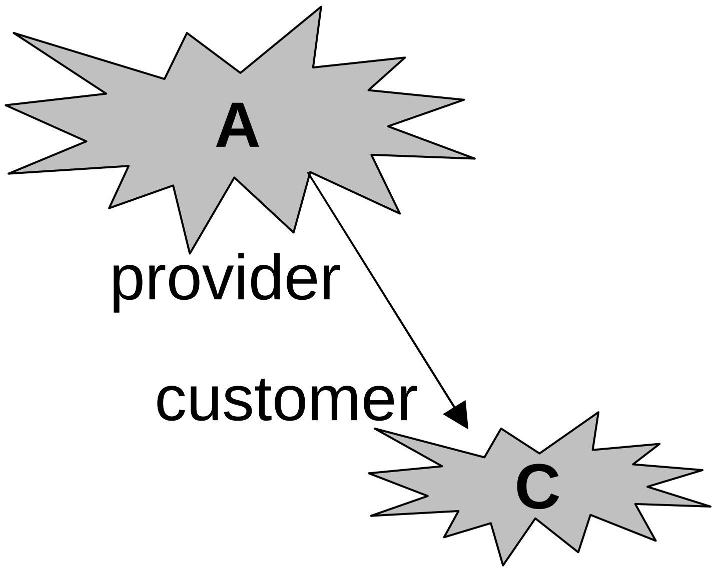
- Internet = network of networks (ASes)
  - Many thousands of ASes
  - Not every network connected to every other network
  - BGP used for routing between ASes
- Differences in economical power/importance
  - Some ASes huge, intercontinental (AT&T, Cable&Wireless)
  - Some ASes small, local (e.g., München: M"Net, SpaceNet)
- Small ASes customers of larger ASes: Transit traffic
  - Smaller AS pays for connecting link + for data = buys transit
  - Business relationship = customer—provider
- Equal-size/-importance ASes
  - Usually share cost for connecting link[s]
  - Business relationship = peering (no transit traffic)
- Warning: peering ("equal-size" AS) ≠ peers of a BGP connection (also may be customer or provider) ≠ peer-to-peer network

Business and policy routing (1)

- Basic principle #1
  - Prefer routes that incur financial gain
- Basic principle #2
  - Announce routes that incur financial gain if others use them
    - Others = customers
  - Announce routes that reduce costs if others use them
    - Others = peers
  - Do not announce routes that incur financial loss
    (…as long as alternative paths exist)

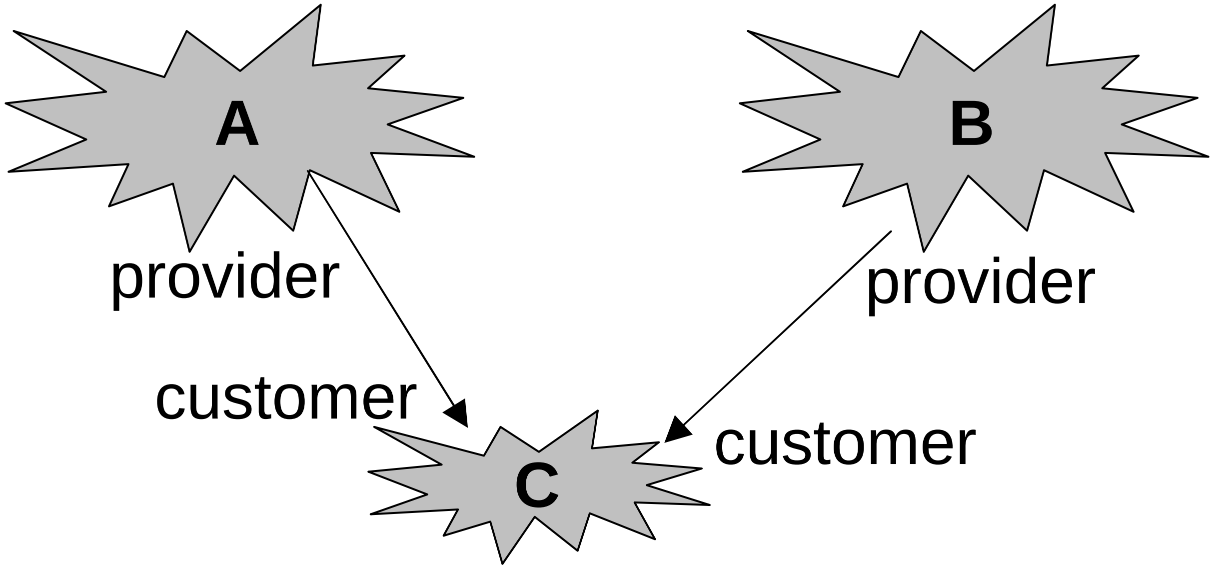
Business and policy routing (2)

- A tells C all routes it uses to reach other ASes
  - The more traffic comes from C, the more money A makes



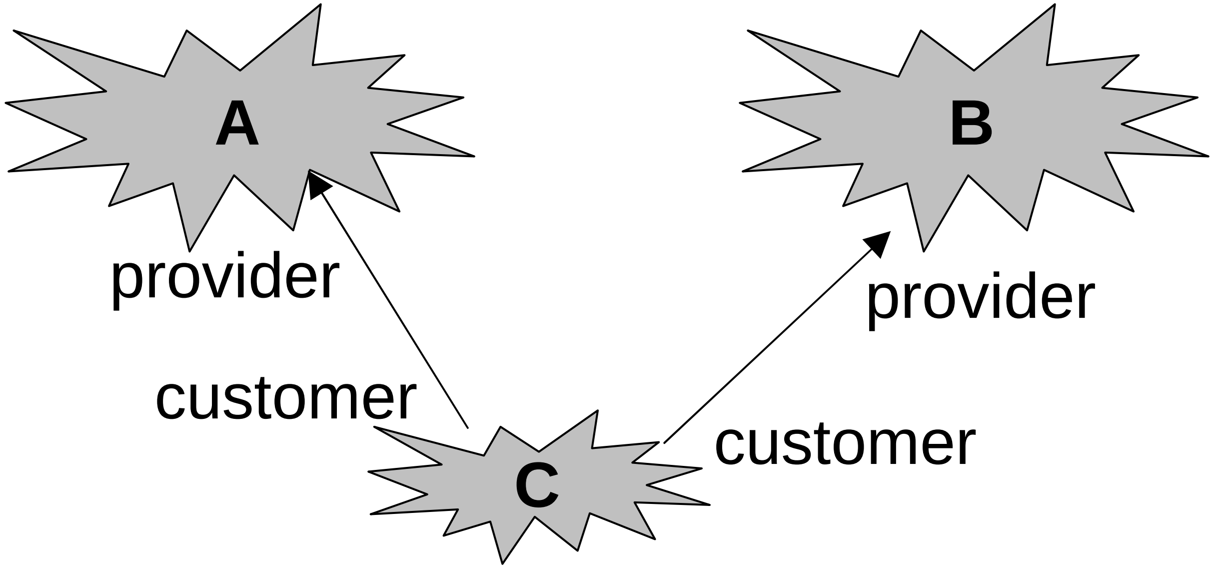
Business and policy routing (3)

- A and B tell C all routes they use to reach other ASes
  - The more traffic flows from C to A, the more money A makes
  - The more traffic flows from C to B, the more money B makes



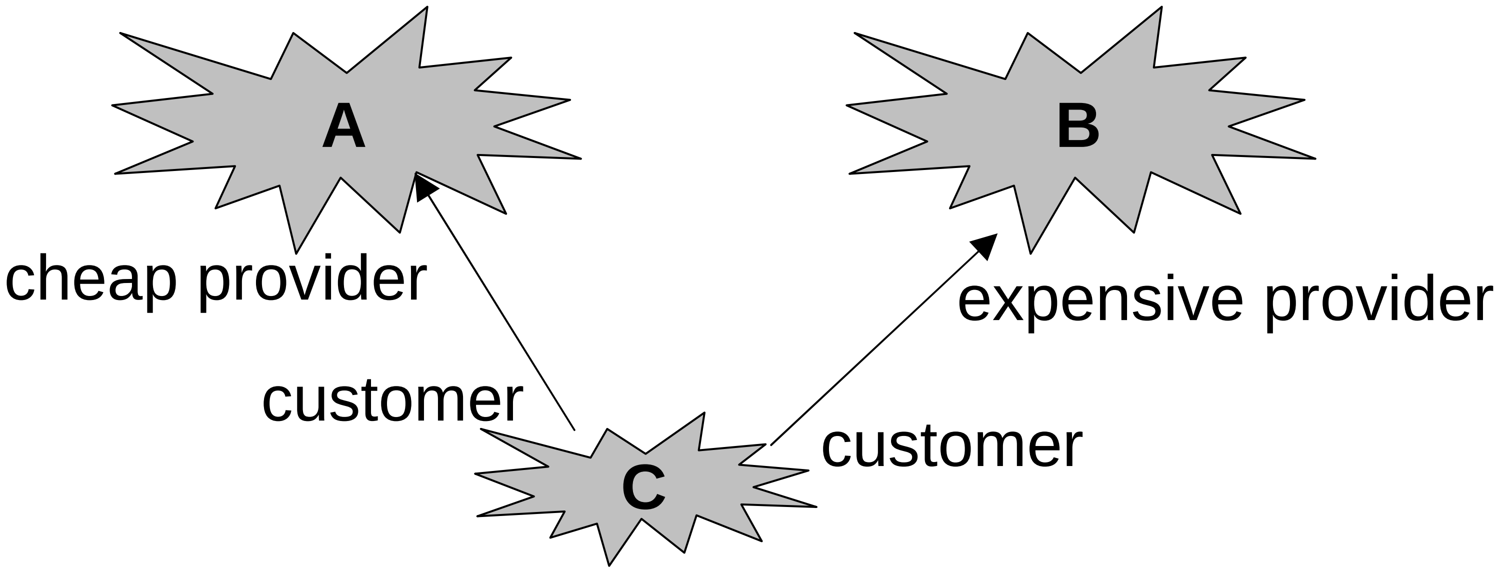
Business and policy routing (4)

- C tells A its own prefixes; C tells B its own prefixes
  - C wants to be reachable from outside
- C does not tell A routes learned from/via B
  C does not tell B routes learned from/via A
  - C does not want to pay money for traffic …↔A ↔C ↔B ↔…



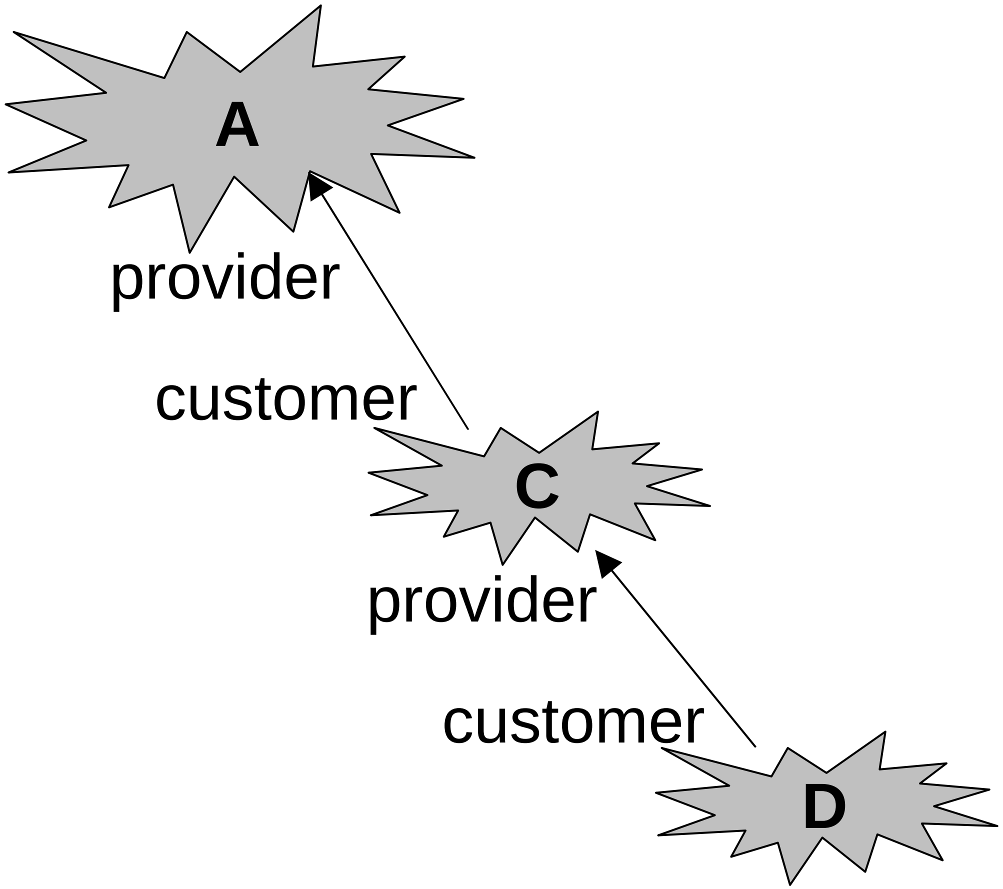
Business and policy routing (5): AS path prepending

- C tells A its own prefixes
- C may tell B its own prefixes
  - …but inserts “C” multiple times into AS path
  - Result: Route available, but longer path = less attractive
  - Technique is called *AS path prepending*



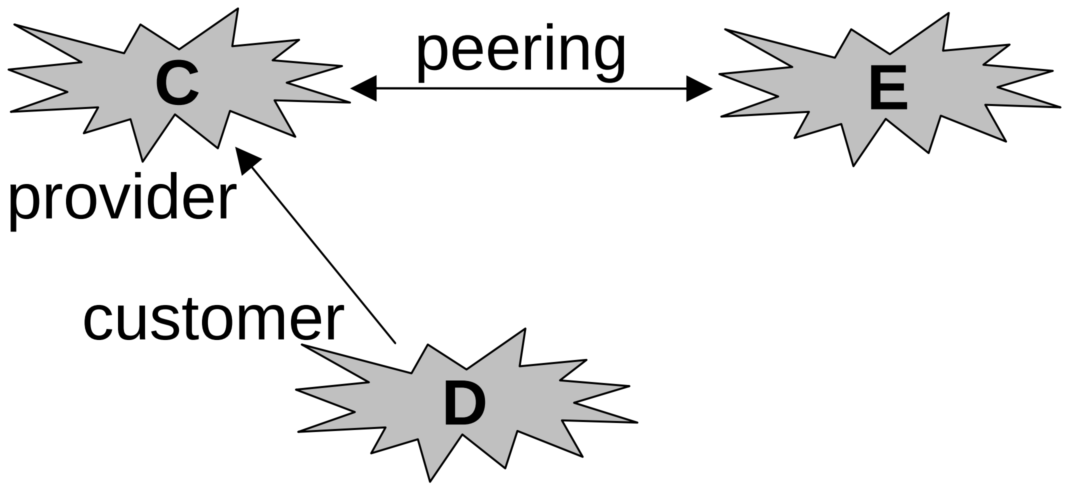
Business and policy routing (6)

- C tells A about its own prefixes
- C tells A about its route to D's prefixes:
  loses money to A, but gains money from D



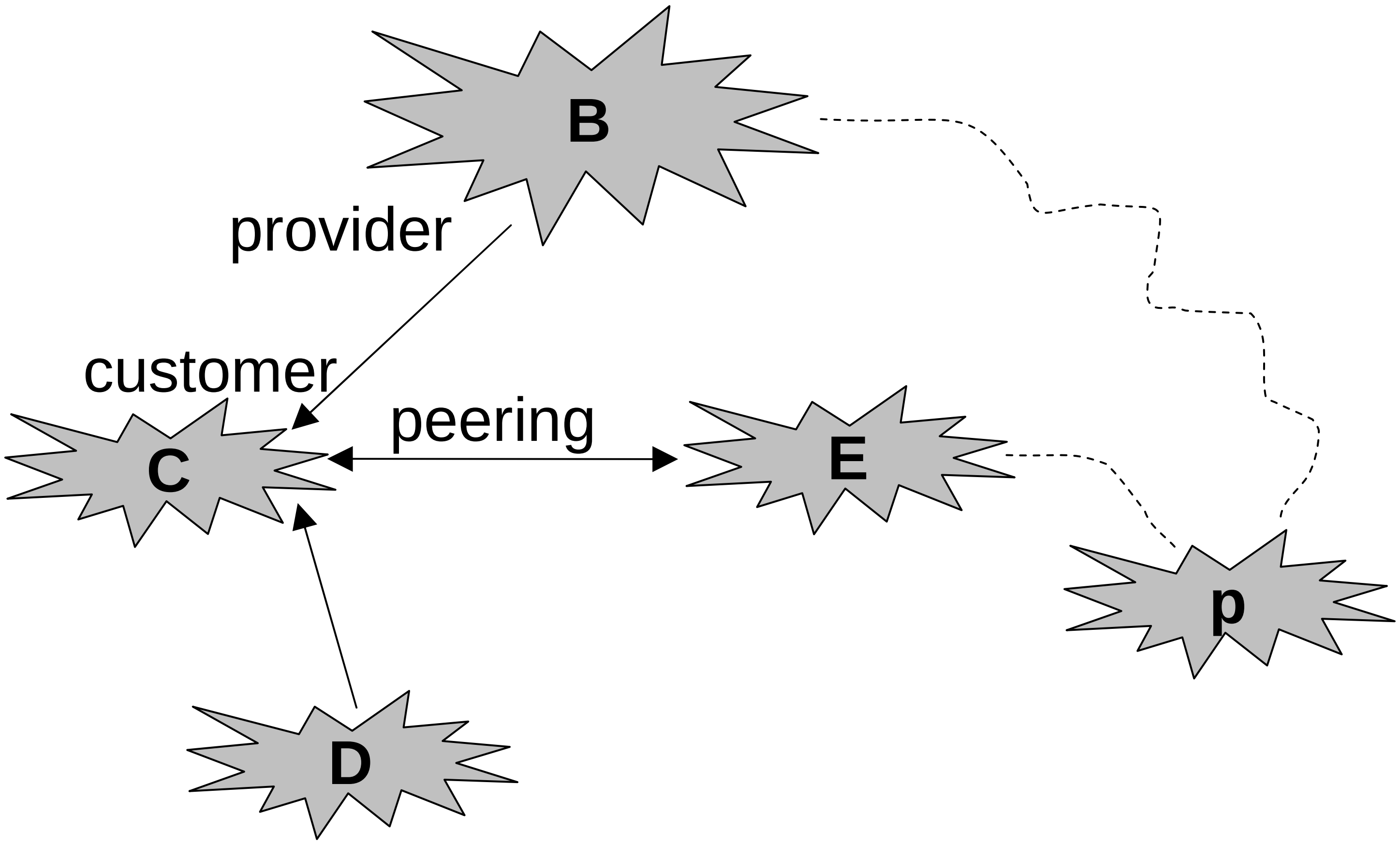
Business and policy routing (7)

- C tells peering partner E about its own prefixes and route to D: no cost on link to E, but gains money from D



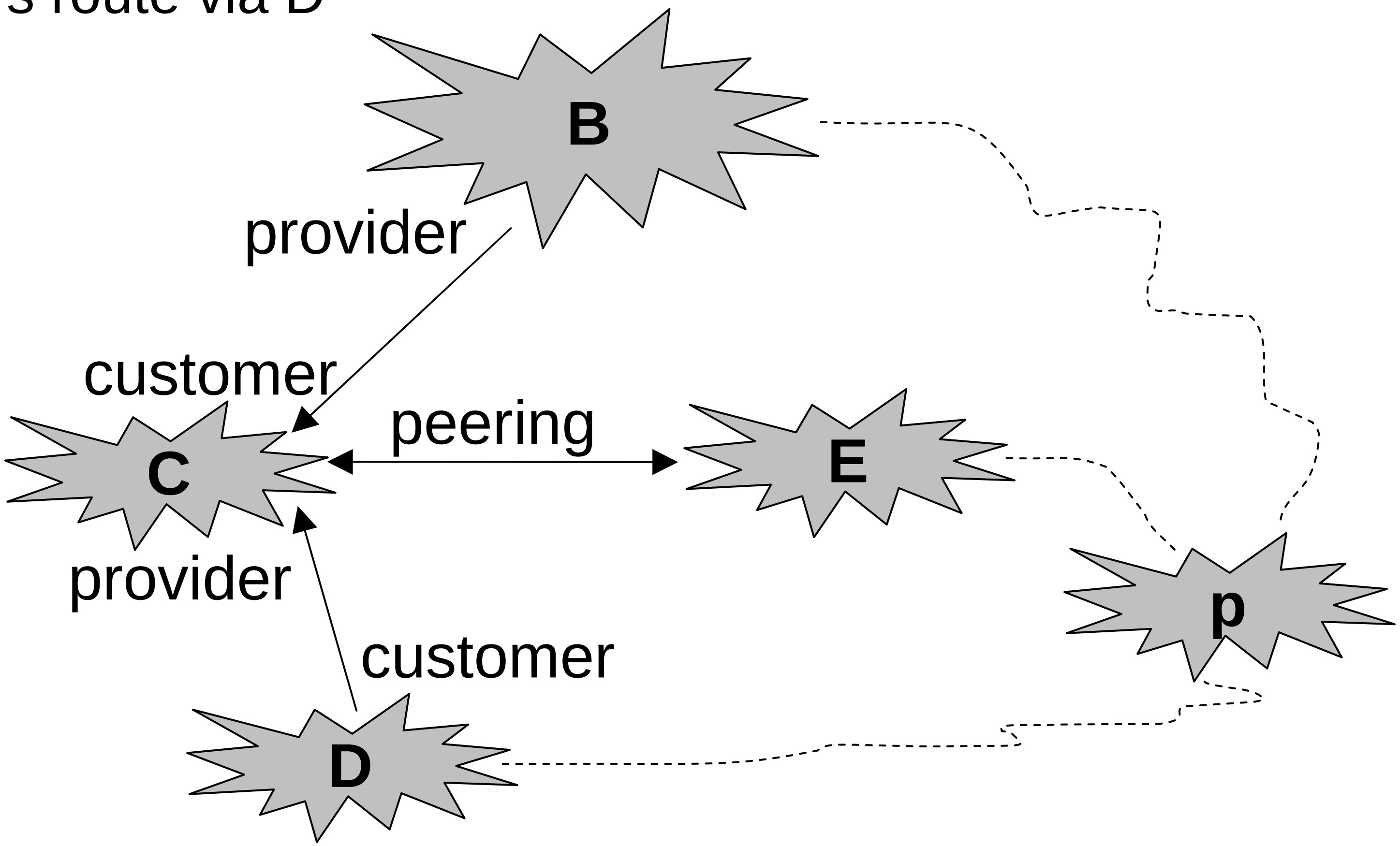
Business and policy routing (8)

- B tells C about route to prefix p (lose money)
- E tells C about route to prefix p (± 0)
- C prefers route via E



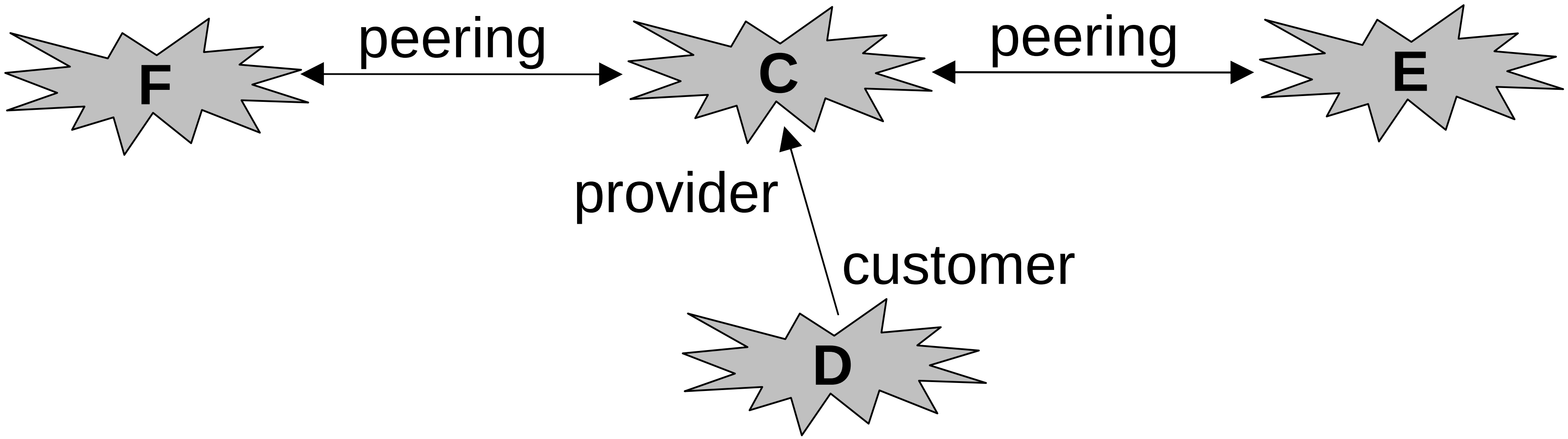
Business and policy routing (8)

- B tells C about route to prefix p (lose money)
- E tells C about route to prefix p (± 0)
- D tells C about route to prefix p (gain money)
- C prefers route via D



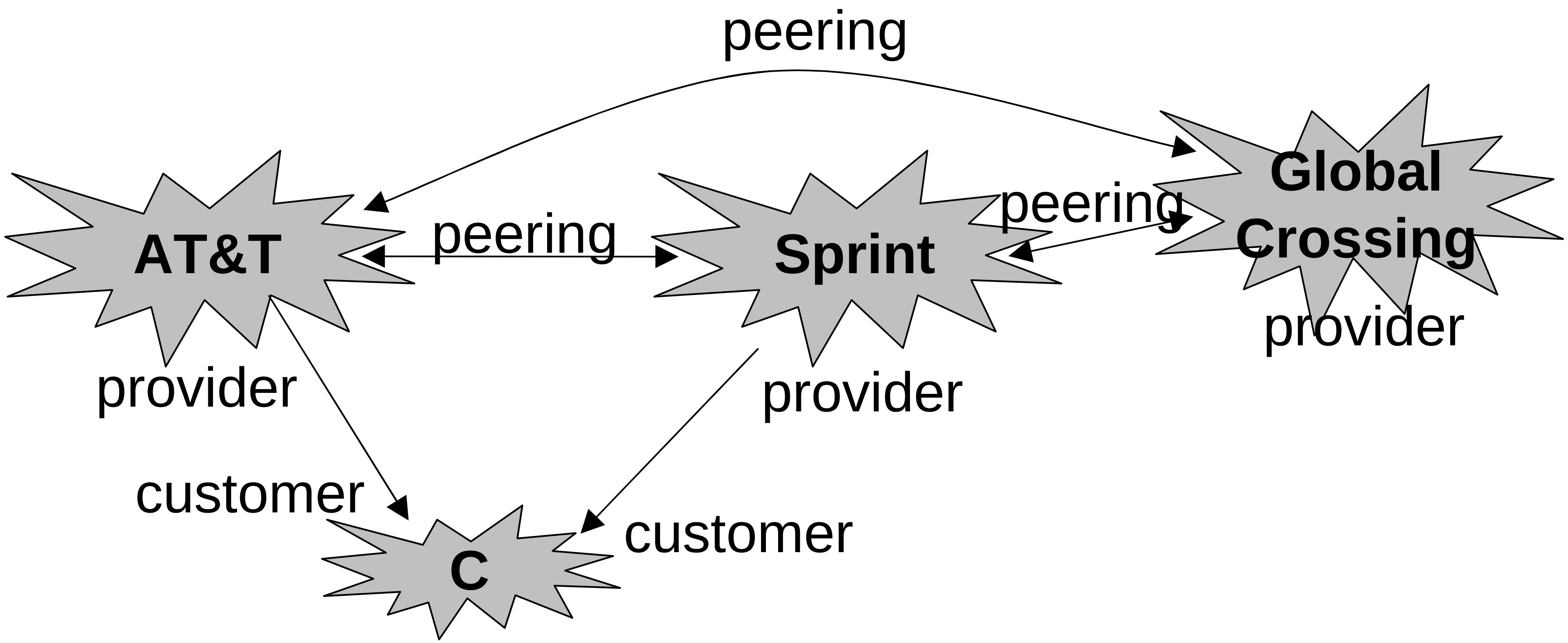
Business and policy routing (9)

- C announces to F and E: its own prefixes and D's routes
- C does *not* announce to E: routes going via F
  - Otherwise: E could send traffic towards F but wouldn't pay anything, F wouldn't pay either, and C's network gets loaded with additional traffic
- C does *not* announce to F: routes going via E
  - Same reason



Business and policy routing (10): "Tiers" / "DFZ"

- Big players have no providers, only customers and peers
  - "Tier-1" providers
  - or "Default-Free Zone" (have no default route to "provider")
- Each Tier-1 peers with each other



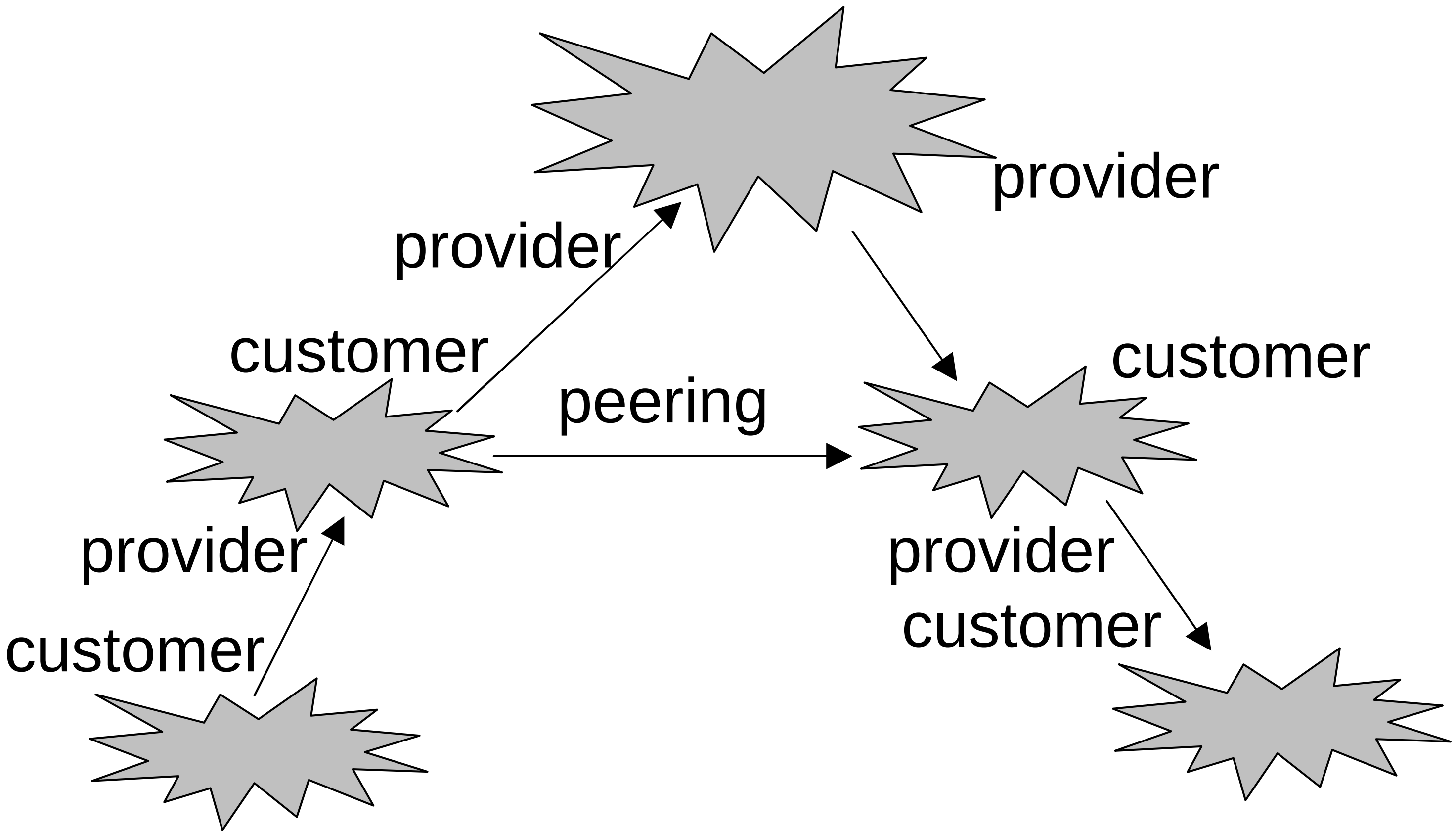
Tier-1, Tier-2, Tier-3 etc.

- Tier-1/DFZ = only peerings, no providers
- Tier-2 = only peerings and Tier-1 providers
- Tier-3 = at least one Tier-2 as a provider
- Tier-*n*: defined recursively
  - $n \geq 4$: Rare in Western Europe, North America, East Asia
- "Tier-1.5" = almost a Tier-1 but pays money for some links
  - Example: Deutsche Telekom pays money to Sprint, but peers with other Tier-1 providers
  - Marketing purposes: Tier-1 sounds better

Valley-free routing

Results: Packets always travel…

1. upstream: sequence of C→P links (possibly length = 0)
2. then possibly across *one* peering link
3. then downstream: sequence of P→C links (possibly length = 0)



Siblings

- Not everything is provider/customer or peering
- Sibling = mutual transit agreement
  - Provide connectivity to the rest of the Internet for each other
  - ≈ very extensive peering
- Examples
  - Two small ASes close to each other that cannot afford additional Internet services
  - Merging two companies
    - Merging two ASes into one = difficult,
    - Keeping two ASes and exchaning everything for free = easier

To peer or not to peer, this is the question

Peer:

- Reduce upstream costs
- Possibly increases performance
- Perhaps only way to connect your customers (Tier-1)

Don't peer

- You don't gain any money
- Peers are usually your competitors
- What if it turns out the peering is more beneficial to you peer than to you? ⇒ Require periodic regenotiation

Where to peer

- Private peering
- At public peering locations (IX, Internet Exchange Point)
  - “A house full of routers that many providers connect to”
  - E.g., DE-CIX, AMS-IX, LINX

BGP/Policy routing Summary

- Import Policy = Which routes to use
  - Select path that incurs most money
  - Special/political considerations (e.g., Iranian AS does not want traffic to pass Israeli AS; other kinds of censorship)
- Export Policy = Which routes to propagate to other ASes
  - Not all possible routes propagate:
    Export only…
    - If it incurs revenue
    - If it reduces cost
    - If it is inevitable
  - Propagation driven by business considerations
  - Propagation not driven by technical considerations!
    Example: Slower route via peer may be preferred over faster route via provider

BGP policy routing: Technical summary

1. Receive BGP update
2. Apply import policies
   - ❑ Filter routes
   - ❑ Tweak attributes (advanced topic…)
3. Best route selection based on attribute values
   - ❑ Install forwarding tables entries for best routes
   - ❑ Possibly transfer to Route Reflector
4. Apply export policies
   - ❑ Filter routes
   - ❑ Tweak attributes
5. Transmit BGP updates

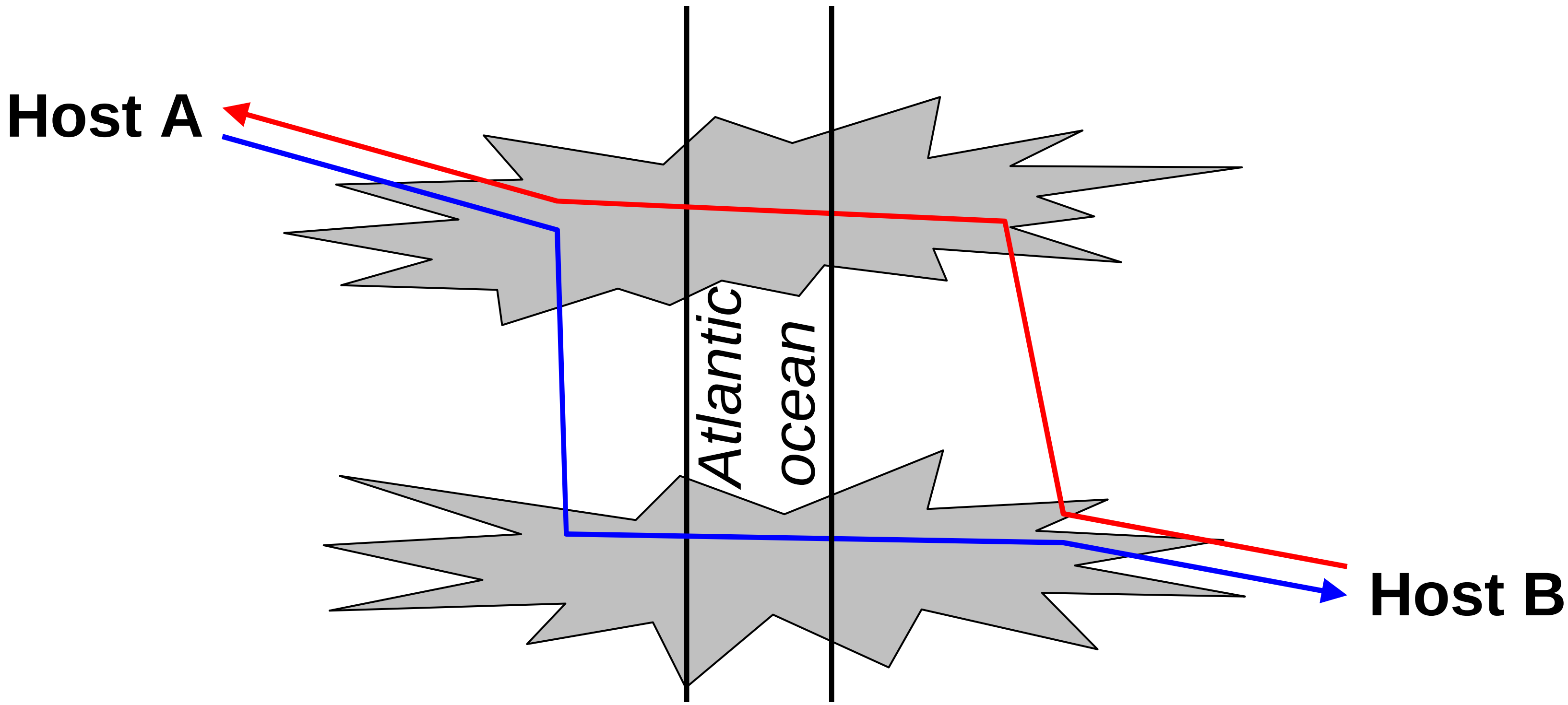
Part 1
- Introduction
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP

Part 2
- IPv6
- NAT
- Virtual circuit and datagram networks
- What's inside a router

Part 3
- Routing algorithms
  - Link state
  - Distance Vector
  - Path Vector
  - Hierarchical routing
- Internet routing protocols
  - (RIP)
  - OSPF
  - BGP
- **Business considerations**
  - Policy routing
  - **Hot-potato routing**
  - Traffic engineering

Hot-potato routing

- Interaction between Inter-AS and Intra-AS routing
  - Business: If traffic is destined for other AS, get rid of it ASAP
  - Technical: Intra-AS routing finds shortest path to gateway
- Multiple transit points ⇒asymmetrical routing



Chapter 4: Network Layer

Part 1

- Introduction
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP

Part 2

- IPv6
- NAT
- Virtual circuit and datagram networks
- What's inside a router

Part 3

- Routing algorithms
  - Link state
  - Distance Vector
  - Path Vector
  - Hierarchical routing
- Internet routing protocols
  - (RIP)
  - OSPF
  - BGP
- **Business considerations**
  - Policy routing
  - **Traffic engineering**

Routing: Optimization purposes

- Inter-AS routing
  - Optimality = select route with highest revenue/least loss
- Intra-AS routing
  - Optimality = configure routing such that network can host as much traffic as possible

Traffic Engineering

1. Collect traffic statistics: Traffic Matrix
   - How much traffic flowing from A to B?
   - Difficult to measure! (drains router performance); thus often estimated: research area
2. Optimize routing
   - E.g., calculate good choice of OSPF weights
   - Goal: minimize maximum link load in entire network; keep average link load below 50%
     - why? Fractal TCP traffic leads to spikes!
3. Deploy new routing
   - Performance may deteriorate during update
   - E.g., routing loops during OSPF convergence

Dynamic traffic engineering

Why not dynamic?

- Routing loops during convergence
- Packet reordering:
  - Packet P1 arrives later than Packet P2
  - TCP will think that P1 got lost! ⇒ congestion control!
- Thus: Congestion control in end hosts, not in network

Note on exercise sheet

- Measurement exercise sheet ≠ 2nd project (on measurement)
- BUT:
  It gives some theoretical foundations for measurement project

Chapter 4: Network Layer

Part 1

- Introduction
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP

Part 2

- IPv6
- NAT
- Virtual circuit and datagram networks
- What's inside a router

Part 3

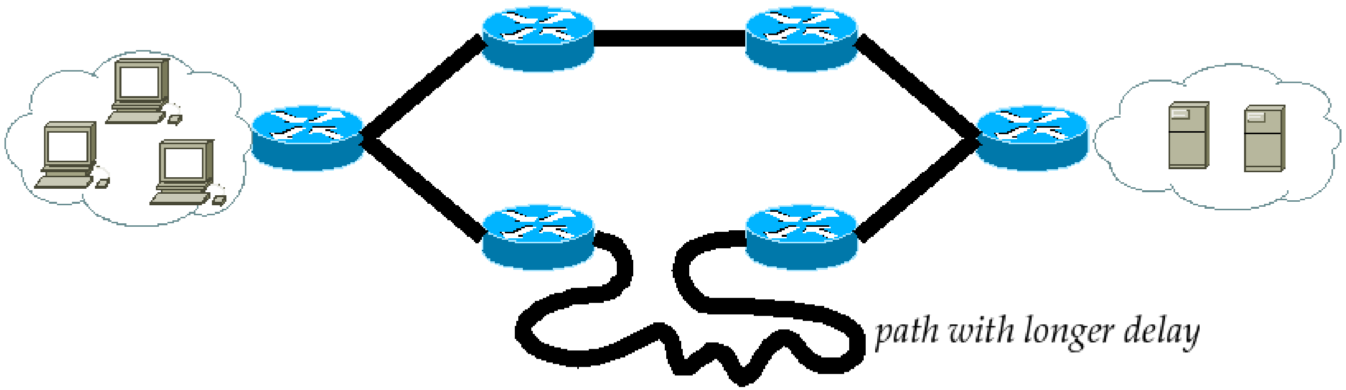
- Routing algorithms
  - Link state
  - Distance Vector
  - Hierarchical routing
- Routing in the Internet
  - OSPF
  - BGP
  - Business considerations
  - **Multipath and TCP**
- Multicast routing
- *NAT (different slide set)*
- Weaknesses and shortcomings

Multipath routing

- Routing = finding best-cost route
- What if more than one exists?
- Some routing protocols allow Equal-Cost Multipath (ECMP) routing, e.g., OSPF
  - $\geq$ 2 routes of same cost exist to destination prefix? $\rightarrow$ Evenly distribute traffic across these routes

Multipath routing: TCP problem

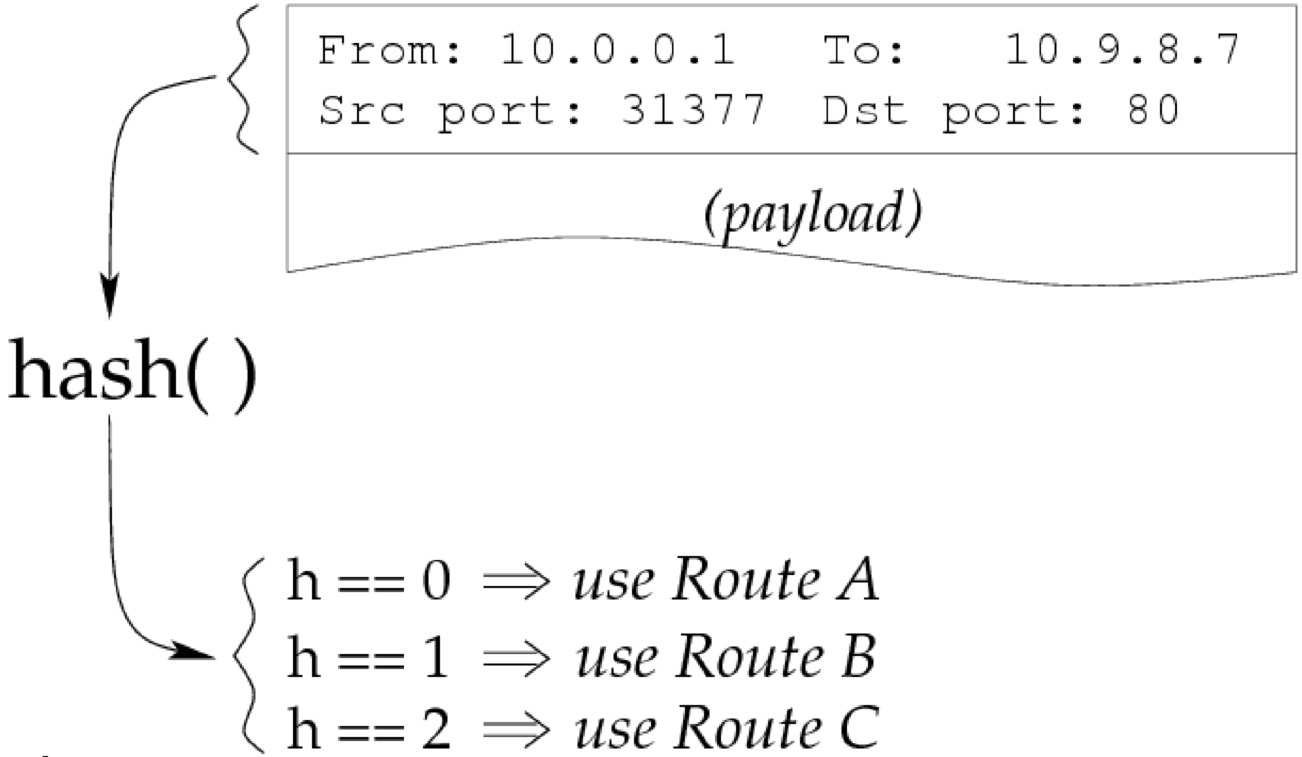
- How to distribute traffic? Naïve approaches:
  - Round-robin
  - Distribute randomly
- Equal cost does not mean equal latency:



- Again: Problem with TCP = Packet reordering!
  - Packets sent: P1, P2
  - Packets received: P2, P1
  - Receiver receives P2 → believes P1 to be lost → triggers congestion control mechanisms → performance degrades

Multipath routing: Solution

- Hash "randomly"…
- …but use packet headers as "random" values:



- Result:
  - Packets from same TCP connection yield same hash value
  - No reordering possible

Chapter 4: Network Layer

Part 1

- Introduction
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP

Part 2

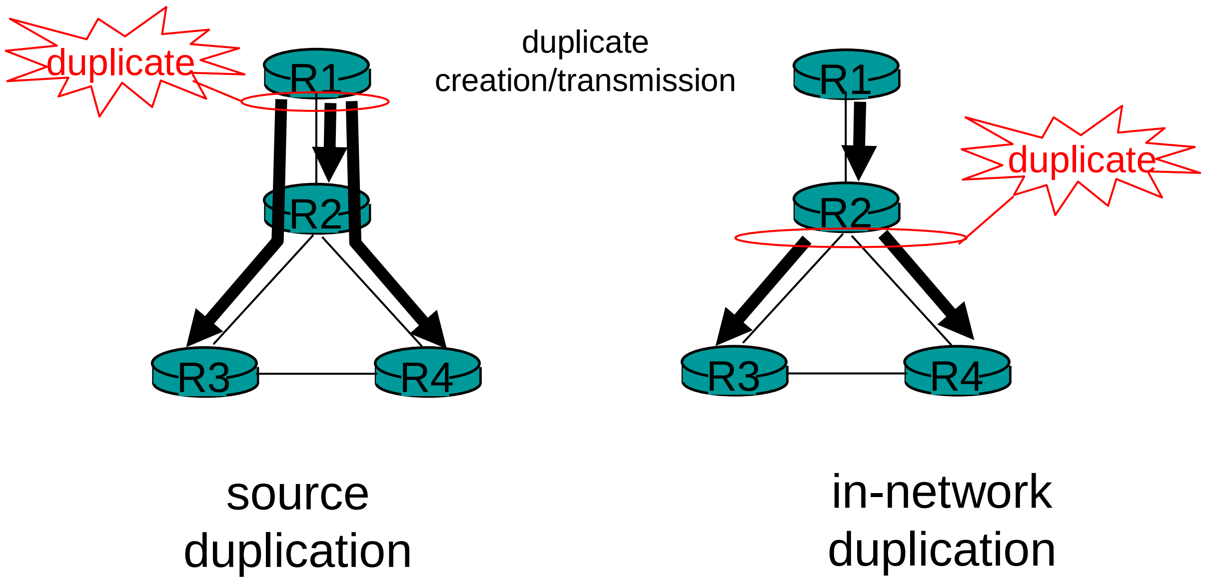
- IPv6
- NAT
- Virtual circuit and datagram networks
- What's inside a router

Part 3

- Routing algorithms
  - Link state
  - Distance Vector
  - Hierarchical routing
- Routing in the Internet
  - OSPF
  - BGP
  - Business considerations
  - Multipath routing and TCP
- **Multicast routing**
- *NAT (different slide set)*
- Weaknesses and shortcomings

Broadcast Routing

- Deliver packets from source to all other nodes
- Source duplication is inefficient:



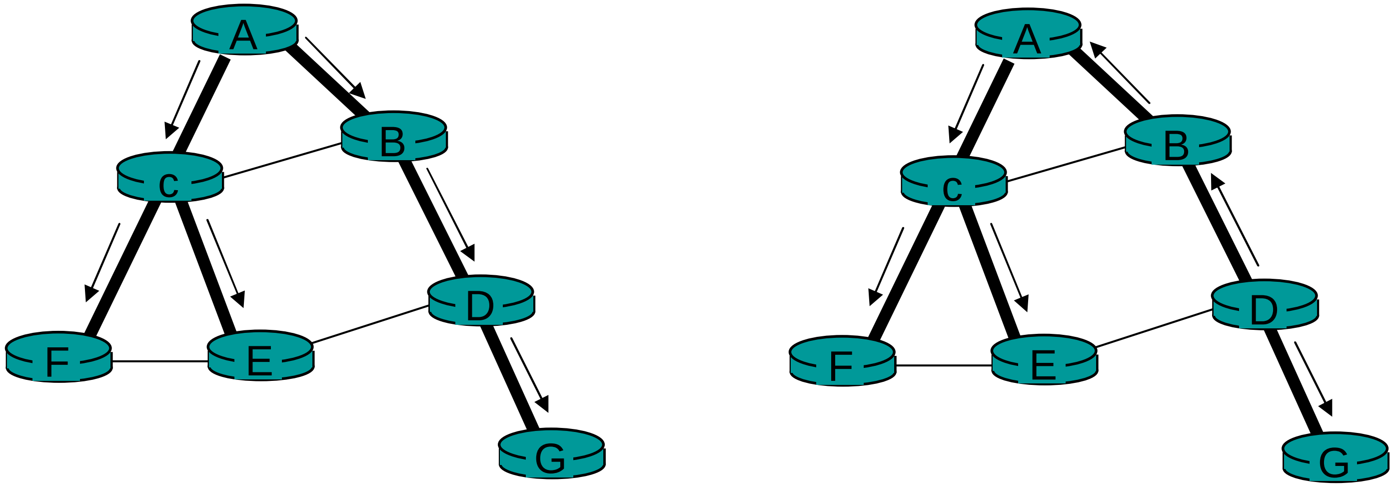
- Source duplication: how does source determine recipient addresses?

In-network duplication

- Flooding: when node receives broadcast packet, sends copy to all neighbours
  - Problems: cycles & broadcast storm
- Controlled flooding: node only broadcasts packet if it hasn't broadcast same packet before
  - Node keeps track of packet IDs already broadcasted
    (need memory! expensive!)
  - Or reverse path forwarding (RPF): only forward packet if it arrived on shortest path between node and source
- Spanning tree
  - No redundant packets received by any node

Spanning Tree

- First construct a spanning tree
  - cf. algorithms + data structures lecture / textbook
- Nodes forward copies only along spanning tree

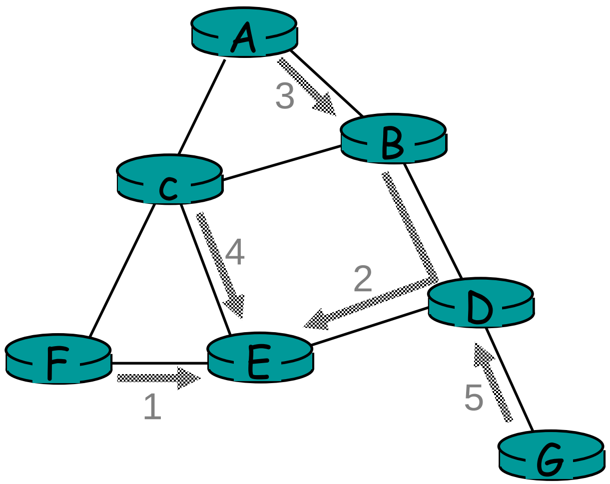


(a) Broadcast initiated at A

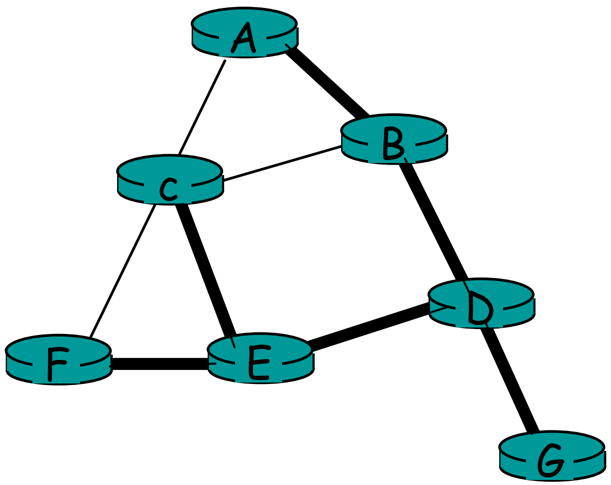
(b) Broadcast initiated at D

Spanning Tree: Creation

- Center node
- Each node sends unicast join message to center node
  - Message forwarded until it arrives at a node already belonging to spanning tree



(a) Stepwise construction of spanning tree



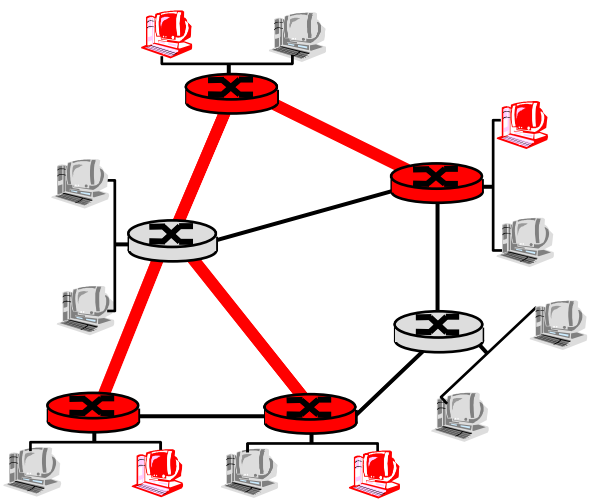
(b) Constructed spanning tree

Multicast groups

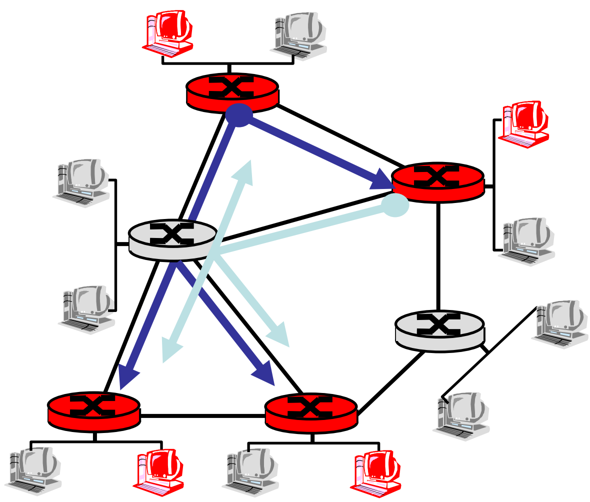
- One single shortest-path graph for all kind of unicast traffic
  - (let's not consider QoS routing here…)
- One single tree for multicast traffic?
  - Consider TV:
    - German TV in D-A-CH area
    - Korean TV in Korea
  - Why multicast German TV in entire Korean IP network and vice versa?
- Multicast group:
  - Different multicast routing trees for different resources
  - Internet: 1 Multicast group = 1 "special" IP address
    - Special = from specific Multicast prefix
    - IP address does not specify one host, but entire group: multiple receivers; multiple senders

Multicast Routing: Problem Statement

- ***Goal:*** find a tree (or trees) connecting routers having local multicast group members
  - *tree:* not all paths between routers used
  - *source-based:* different tree from each sender to rcvrs
  - *shared-tree:* same tree used by all group members



Shared tree



Source-based trees

Approaches for building multicast trees

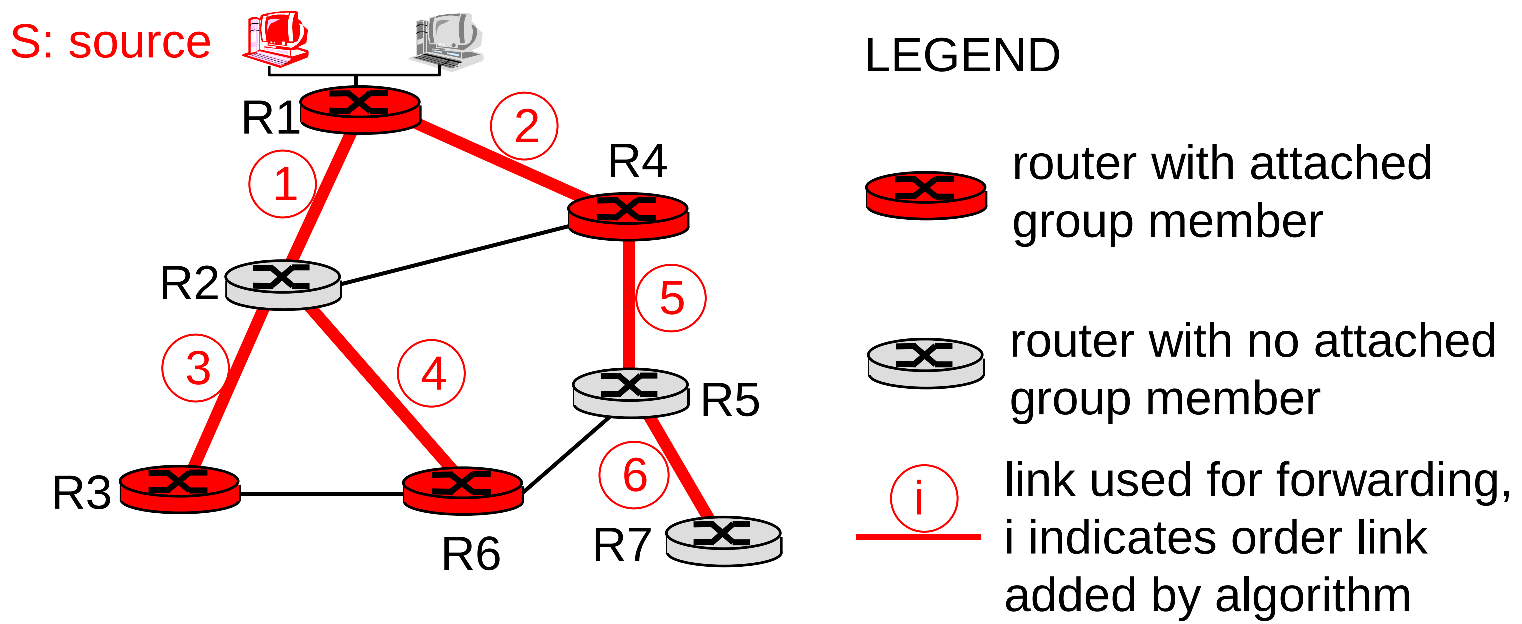
Approaches:

- source-based tree: one tree per source
  - shortest path trees
  - reverse path forwarding
- group-shared tree: group uses one tree
  - minimal spanning (Steiner)
  - center-based trees

…we first look at basic approaches, then specific protocols adopting these approaches

Shortest Path Tree

- Multicast forwarding tree: tree of shortest path routes from source to all receivers
  - Dijkstra's algorithm



Reverse Path Forwarding

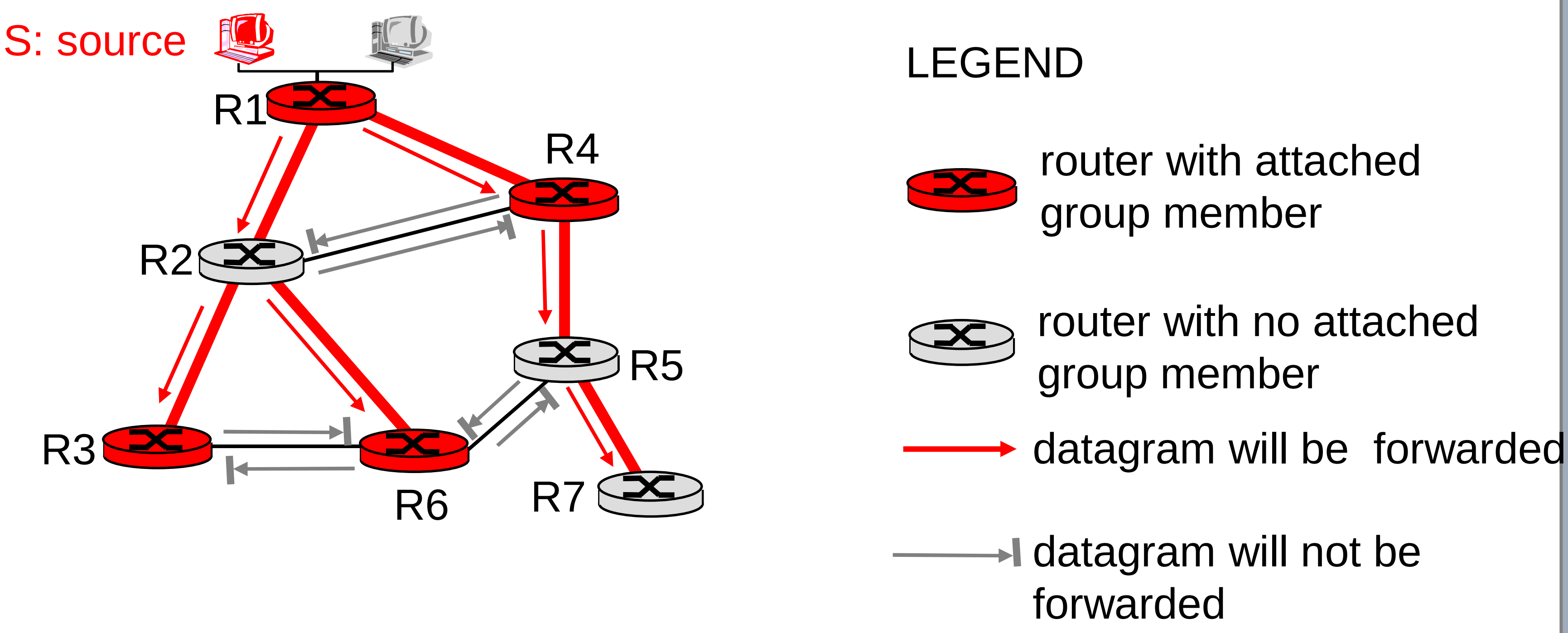
- Rely on router's knowledge of unicast shortest path from it to sender
- Each router has simple forwarding behavior:

if (mcast datagram received on incoming link on shortest path back to center)

then flood datagram onto all outgoing links

else ignore datagram

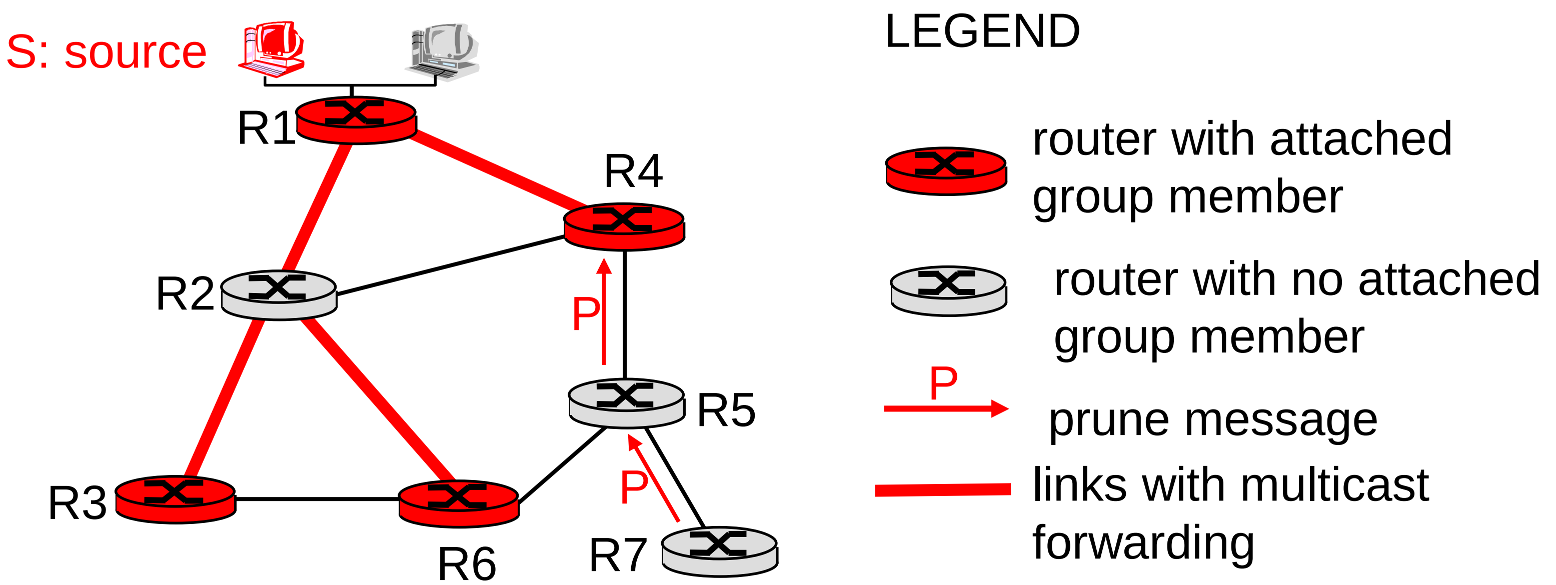
Reverse Path Forwarding: example



- result is a source-specific *reverse* SPT
  – may be a bad choice with asymmetric links

Reverse Path Forwarding: pruning

- Forwarding tree contains subtrees with no multicast group members
  - no need to forward datagrams down subtree
  - "prune" msgs sent upstream by router with no downstream group members



Shared-Tree: Steiner Tree

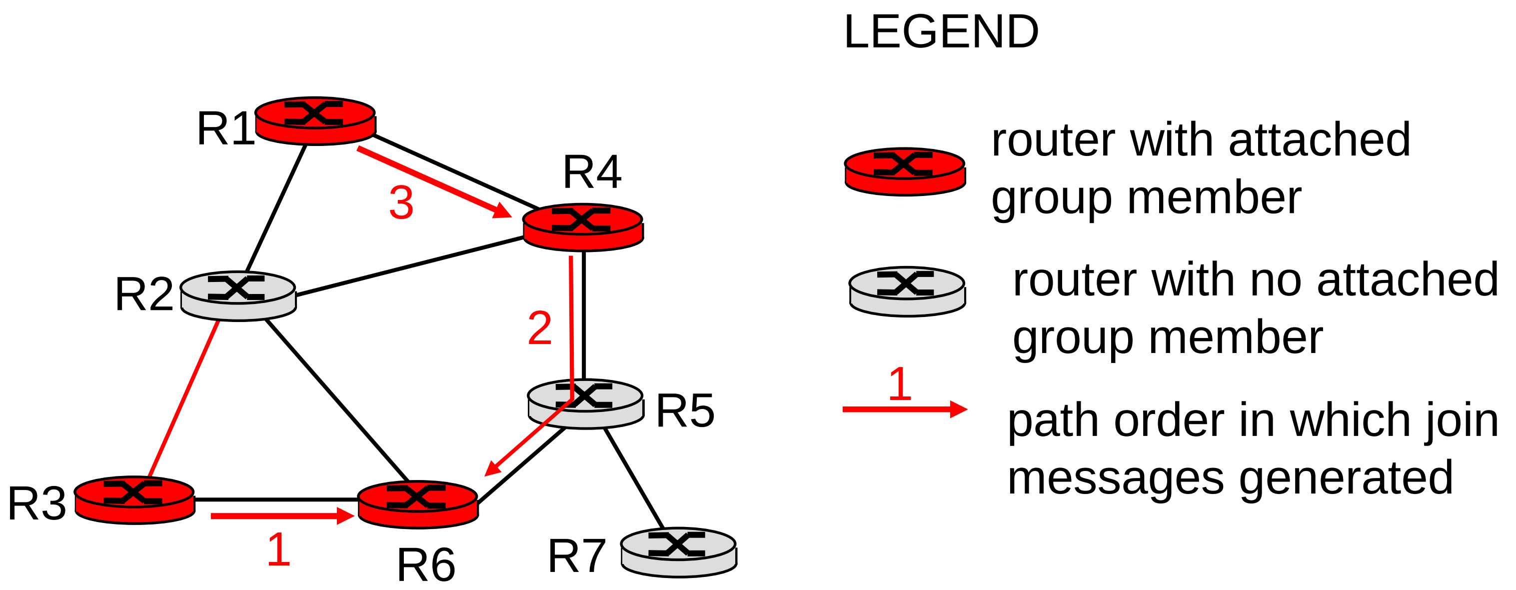
- Steiner Tree: minimum cost tree connecting all routers with attached group members
- Problem is NP-complete
- Excellent heuristics exists; active research area in theoretical computer science
- But not used in practice:
  - Computational complexity
  - Information about entire network needed
  - Monolithic: rerun whenever a router needs to join/leave

Center-based trees

- Single delivery tree shared by all members
- One router identified as *"center"* of tree
- To join:
  - edge router sends unicast *join-msg* addressed to center router
  - *join-msg* "processed" by intermediate routers and forwarded towards center
  - *join-msg* either hits existing tree branch for this center, or arrives at center
  - path taken by *join-msg* becomes new branch of tree for this router

Center-based trees: an example

Suppose R6 chosen as center:



Multicast routing in the Internet (1)

- Multicast routing protocols
  - DVMRP: distance-vector multicast routing protocol
  - MOSPF: Multipath OSPF
  - PIM: Protocol Independent Multicast
- But the end hosts!?
  - End hosts send/receive Multicast traffic,…
  - …but do not run routing protocols!
- IGMP (Internet Group Management Protocol): IPv4
  - Host can join/leave multicast group
  - Sits on top of IPv4
- MLD (Multicast Listener Discovery): IPv6
  - Router discovers multicast listeners
  - Embedded into IPv6

Multicast routing in the Internet (2)

- Who uses it? Almost nobody!

No application makes use of it

Not available to "normal" end users

Nobody asks for it

Providers normally don't offer it

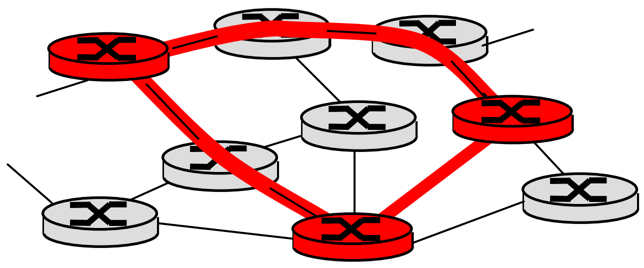
Accounting quarrels: who pays for what?

Technical issues: need rapid+efficient join/leave across AS boundaries…

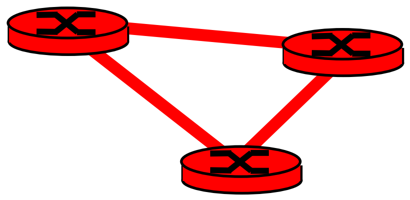
- Available as special solution in some isolated contexts
  - Triple Play: Internet + telephone (VoIP) + TV over IP
  - Software updates in large companies

Tunneling

Q: How to connect "islands" of multicast routers in a "sea" of unicast routers?



physical topology



logical topology

- Multicast datagram encapsulated inside "normal" (non-multicast-addressed) datagram
- Normal IP datagram sent through "tunnel" via regular IP unicast to receiving multicast router
- Receiving mcast router unencapsulates to get multicast datagram

PIM: Protocol Independent Multicast

- Not dependent on any specific underlying unicast routing algorithm (works with all)
- Two different multicast distribution scenarios :

Dense:

- group members densely packed, in “close” proximity
- bandwidth more plentiful

Sparse:

- #of networks with group members small with respect to # of interconnected networks
- group members “widely dispersed”
- bandwidth not plentiful

Consequences of Sparse/Dense Dichotomy:

Dense

- Group membership by routers *assumed* until routers explicitly prune
- *Data-driven* construction on multicast tree
- Bandwidth and non-group-router processing may *waste* resources

Sparse:

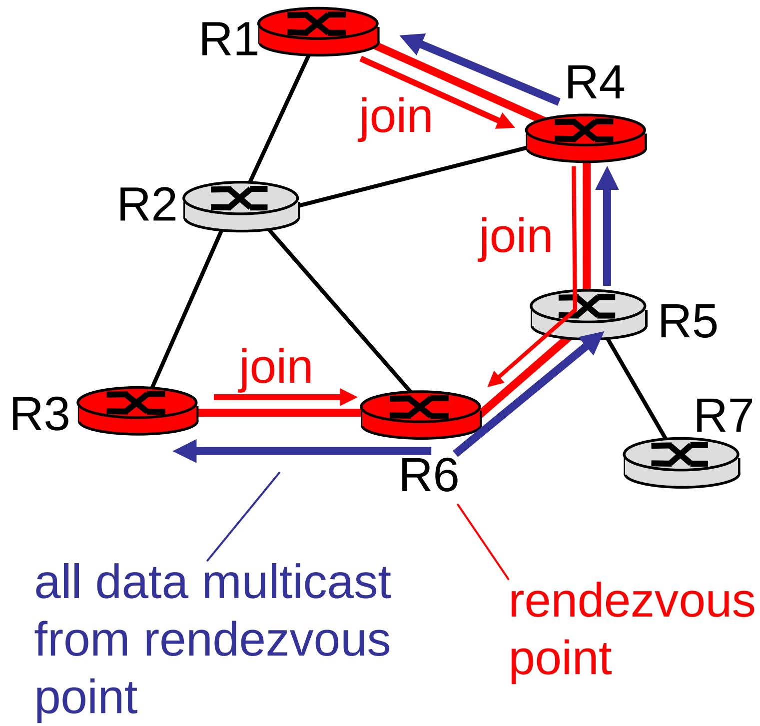
- No membership until routers explicitly join
- *Receiver- driven* construction of multicast tree (e.g., center-based)
- Bandwidth and non-group-router processing *conservative*

PIM: Dense mode

- "Flood and prune":
  Use Reverse Path Forwarding to flood information across network
- Uninterested leaf routers *may* prune network
  - Mechanism to detect if leaf router

PIM: Sparse Mode

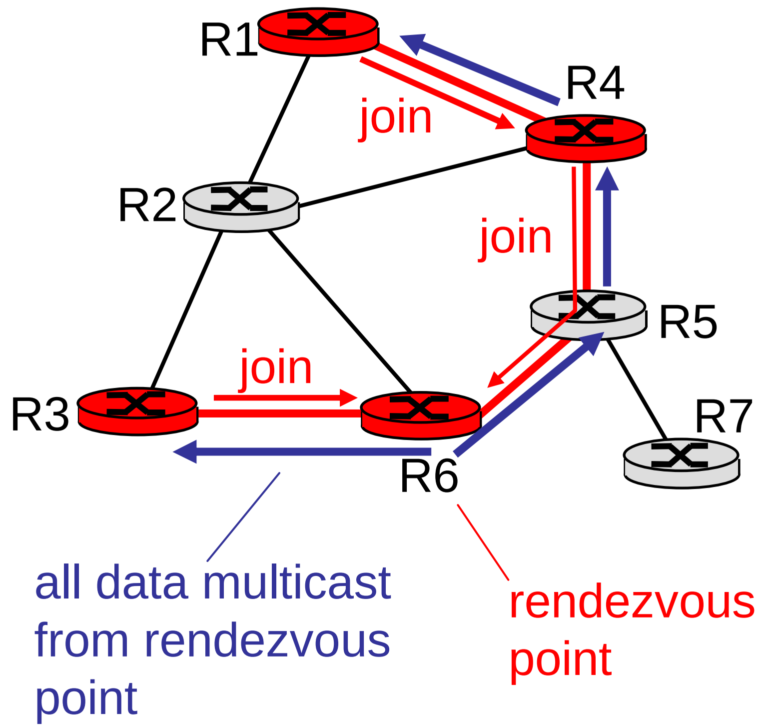
- Center-based approach
- Router sends *join* msg to rendezvous point (RP)
  - intermediate routers update state and forward *join*
- After joining via RP, router can switch to source-specific tree
  - increased performance: less concentration, shorter paths



PIM: Sparse Mode

sender(s):

- unicast data to Rendezvous Point (!)
- RP then distributes down RP-rooted tree
- RP can extend multicast tree upstream to source
- RP can send *stop* msg if no attached receivers
  - "no one is listening!"



Chapter 4: Network Layer

Part 1

- Introduction
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP

Part 2

- IPv6
- NAT
- Virtual circuit and datagram networks
- What's inside a router

Part 3

- Routing algorithms
  - Link state
  - Distance Vector
  - Hierarchical routing
- Routing in the Internet
  - OSPF
  - BGP
  - Business considerations
  - Multipath routing and TCP
- Multicast routing
- ***NAT (different slide set)***
- Weaknesses and shortcomings

Part 1

- Introduction
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP

Part 2

- IPv6
- NAT
- Virtual circuit and datagram networks
- What's inside a router

Part 3

- Routing algorithms
  - Link state
  - Distance Vector
  - Hierarchical routing
- Routing in the Internet
  - OSPF
  - BGP
  - Business considerations
  - Multipath routing and TCP
- Multicast routing
- *NAT (different slide set)*
- **Weaknesses and shortcomings**

Network Layer: Weaknesses and shortcomings (1)

- Violation of layering principles
  - NAT: share IP address (L3) based on TCP/UDP ports (L4)
  - Firewalls: router (L3) blocks traffic based on TCP/UDP (L4)
  - Intelligent firewalls, transparent proxies: IP traffic (L3) intercepted and mangled depending on application! (L5–7)
  - Multi-path routing (L3): take care of TCP (L4)
  - Dynamic routing (L3): complicated due to TCP (L4)
  - Layer 1, Layer 2 must not drop too many packets (e.g., wireless, satellite, etc.):
    TCP (L4) believes losses to be caused by congestion

Network layer: Weaknesses and Shortcomings (2)

- Security
  - Denial of service attacks:
    Undesired traffic dropped at receiver, not in network
  - Other attacks: hard to trace, no sender signature
  - BGP misconfiguration can create havoc
    - Example: Pakistan created YouTube black hole
  - BGP implementation errors can create havoc
    - Example: Czech provider creates huge AS path
      => Many routers crash world-wide
      => Wildly oscillates
  - Question: What about concerted attack on BGP…? ☹ ☹ ☹
- Routing = destination-based
  - No complete choice of paths
  - Restricts solutions for traffic engineering

Network layer: Weaknesses and shortcomings (3)

- No network congestion control:
  Dynamic routing / dynamic traffic engineering = difficult!
  - Tried out in ARPANET: Oscillations everywhere
  - Today: Interaction with TCP congestion control feedback loop → even worse!
- Convergence speed (link/router failures)
  - OSPF: 200ms … several seconds
    - Routing loops may occur during convergence = black holes
  - BGP: seconds to several minutes!
    - Never really converges: there's always something going on
- More and more prefixes in routing tables
  - 300,000 and growing
  - IPv6 does *not* help! (in contrast…)

Network Layer: Weaknesses and shortcomings (4)

- Manageability
  - Routing = complex to set up
  - Even more complex to manage/debug
    - What/who caused the error? – Difficult to answer!
- End hosts: increasingly mobile
  - WLAN → UMTS? = IP address changes!
- Multicast is not deployed
- Quality of service
  - Different applications have different service demands
    - File transfer: max bandwidth
    - Chat, VoIP, games: min delay
    - E-Mail: min cost
  - QoS = different *classes of service*
  - Works in theory and lab — but is not deployed!
    (same reasons as with multicast)

Future Internet

- Obviously, the Internet as we know it needs to change
- Research term: “Future Internet”
  - Lots of €€$$¥¥££ spent on this
  - Everyone is doing Future Internet research nowadays…
- Revolutionary approach (“clean slate”)
  - Throw everything old away, make it new from scratch
  - Tackle multiple problems at once
- Evolutionary approach
  - Change disturbing aspects separately,
  - one at a time
  - In coexistence with today’s network landscape

THANK YOU