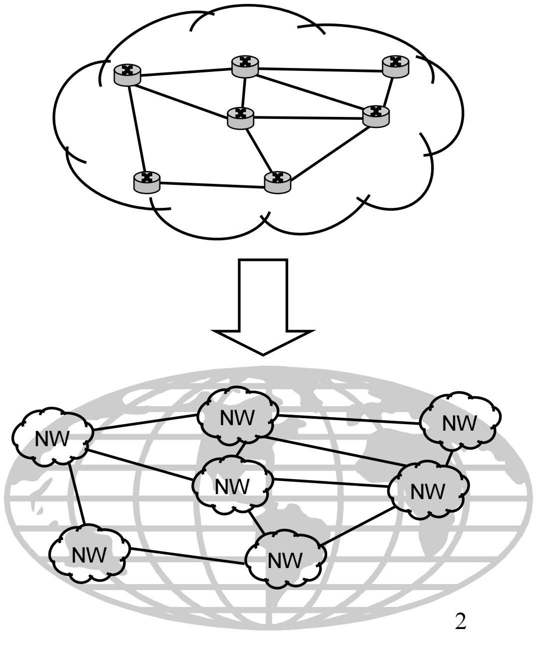
Routing in Internet

Problem

- Given set of nodes, how do routers acquire info about neighbors to construct the routing tables?
- Requirements:
    - distributed algorithms surviving link failures and topology changes
    - efficient resource usage (minimum cost routing)
    - must be able to handle highly varying traffic loads
- Issue of scale:
    - hierarchical network, backbone routers serve millions of hosts
    - routing within a “domain” done differently than between “domains”
        - intra domain vs. inter domain

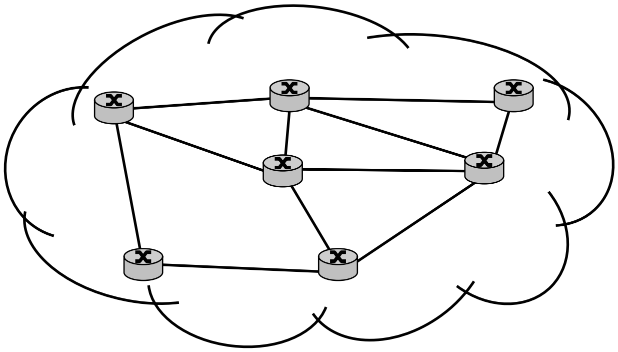


Outline

- Intradomain routing
  - distance vector routing (RIP)
  - link state routing (OSPF)
  - determining link costs
- Routing in global Internet
  - mechanisms: subnetting and classless routing (CIDR)
  - interdomain routing (BGP)
- Routing private and public IP addresses (NAT)

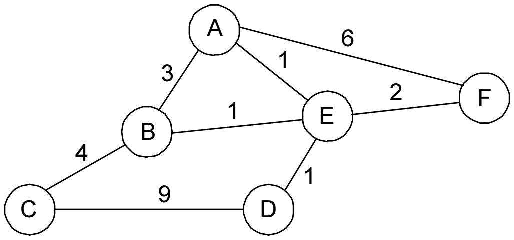
Intradomain routing

- Forwarding vs routing
  - routing: process by which routing table is built
- Intradomain routing
  - domain = routers belonging in same administrative domain (“cloud”)
  - same as IGP (Interior Gateway Protocol)
  - still not scalable to huge networks



Least cost routing

- Network as a (weighted) graph
  - vertices = routers
  - edges = network links
  - edge weight = cost of using the link



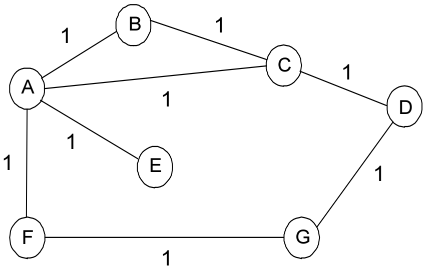
- Problem: find lowest cost path between two nodes
  - assuming given links costs (determining them treated later...)
  - using a distributed algorithm
  - two classes of algorithms: distance vector (RIP) and link state (OSPF)
- Factors
  - changing topology and varying link costs (loads)
  - topology changes at a slower time scale

Distance vector routing

- Idea in distance vector routing
  - nodes construct vector containing distances to all other nodes
  - distance vector distributed to all neighbors
  - initially each node knows only distance to immediate neighbors
  - a link that is down has “infinite” cost
  - converges typically quickly after few iterations
- More detailed:
  - each node maintains a list of triplets: (Destination, Cost, NextHop)
  - exchange updates with directly connected neighbors
    - periodically (on the order of several seconds)
    - whenever table changes (called triggered update)
    - each update is a list of pairs: (Destination, Cost)
  - update local table entry
    - always, if route update comes from entry’s “next hop” router
    - if receive a “better” route (smaller cost) from any neighbor (next-hop routers)
  - refresh existing routes; delete if they time out

Example

- Events at node B
  - learns from C that D can be reached at cost 1 ⇒ cost from B to D via C is 2 ⇒ new route accepted by B
  - learns from C that A can be reached at cost 1 ⇒ cost from B to A via C is 2 ⇒ new route not accepted by B
  - learns from A that E can be reached at cost 1 ⇒ cost from B to E via A is 2 ⇒ new route accepted by B
  - learns from A that F can be reached at cost 1 ⇒ cost from B to F via A is 2 ⇒ new route accepted by B
  - learns from C that G can be reached at cost 2 ⇒ cost from B to G via C is 3 ⇒ new route accepted by B



Initial routing table at B

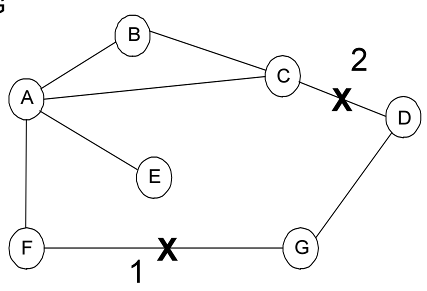
Destination	Cost	NextHop
A	1	A
C	1	C
D	Inf	-
E	Inf	-
F	Inf	-
G	Inf	-

Final routing table at B

Destination	Cost	NextHop
A	1	A
C	1	C
D	2	C
E	2	A
F	2	A
G	3	C

Routing loops

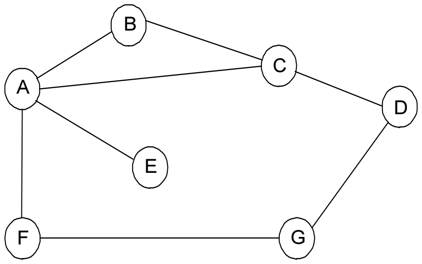
- Link failure 1: correct operation
  - F detects that link to G has failed
  - F sets distance to G to infinity and sends update to A
  - A sets distance to G to infinity since it uses F to reach G
  - A receives periodic update from C with 2-hop path to G
  - A sets distance to G to 3 and sends update to F
  - F decides it can reach G in 4 hops via A



- Link failure 2: count to infinity problem (loops)
  - also link from C to G fails
  - D advertises (C,Inf) and, at same time (periodic update), G advertises (D,2)
  - G receives (C,Inf) from D, sets (C,Inf,D) and generates (C,Inf)
  - D receives (C,2) from G, sets (C,3,G) and generates (C,3)
  - G receives (C,3) from D, sets (C,4,D) and generates (C,4)
  - D receives (C,Inf) from G, sets (C,Inf,G) and generates (C,Inf)
  - … loop, where distance increases by 1 until infinity

Loop-breaking heuristics

- Previous just an example of what can go wrong
  - can occur in more complex network scenarios
  - one basic reason is that due to timing of events it is possible that a particular node can transmit "false" information before new information has reached it
- Set infinity to 16
- Split horizon
  - node does not send those routes it learned from its neighbors
  - B uses route (E,2,A), during update B does not include (E,2) in the message to A



- Split horizon with poison reverse
  - send negative information back to neighbors to ensure that e.g. A never sends traffic to E via B
  - B sends route information back to A containing (E,Inf)
- These techniques work only for routing loops involving 2 nodes

---

Routing Information Protocol (RIP)

- RIP widely used in Internet
  - implemented in BSD version of Unix
- Straightforward implementation of distance vector routing
  - routers advertise the cost of reaching networks (instead of other routers)
  - periodic updates every 30 s
  - RIP supports multiple protocol families (not just IP)
  - RIP assumes that link costs are always equal to 1 (minimum hop route)
  - valid distances 1, ..., 15, and Infinity = 16
- RIPv2 has some scalability features

Outline

- Intradomain routing
  - distance vector routing (RIP)
  - link state routing (OSPF)
  - determining link costs
- Routing in global Internet
  - mechanisms: subnetting and classless routing (CIDR)
  - interdomain routing (BGP)
- Routing private and public IP addresses (NAT)

Link state routing overview

- Strategy
  - same as in distance vector routing: provide enough info to nodes so they can build least cost paths to all destinations
  - every node knows how to reach directly connected nodes
  - send to all nodes (not just neighbors) information about directly connected links (not entire routing table)
    - nodes get complete topology information
    - from topology information, compute shortest paths
- Mechanisms
  - reliable flooding of link state information (using LSPs)
  - Dijkstra’s algorithm to compute shortest paths

Reliable flooding

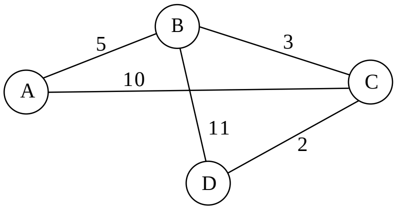
- Link State Packet (LSP)
  - id of the node that created the LSP
  - cost of link to each directly connected neighbor
  - sequence number (SEQNO)
  - time-to-live (TTL) for this packet
- Reliable flooding
  - reliable delivery of LSPs by using ACKs and retransmissions between neighbors
  - store most recent LSP from each node (based on SEQNO)
    - important to have always the most recent routing info
  - forward LSP to all neighboring nodes but the one that sent it
  - generate new LSP periodically (or triggered if directly connected link fails)
    - increment SEQNO
  - start SEQNO at 0 when reboot
  - decrement TTL of each stored LSP
    - discard when TTL=0
- After flooding is complete every node has complete topology information
  - shotrest paths can be now computed

Route calculation

- Dijkstra's shortest path algorithm
  - N : set of nodes in the graph
  - l (i, j) : non-negative cost for edge (i, j)
  - S : this (current) node
  - M : set of nodes incorporated so far
  - C(n) : cost of the path from s to node n

```
M = {s}
for each n in N - {s}
   C(n) = l(s, n)
while (N ≠ M)
   M = (M U {w}) such that C(w) is
         the minimum for all w in (N - M)
   for each n in (N - M)
      C(n) = MIN(C(n), C (w) + l(w, n ))
```

- Example: Consider node A
  1. M={A}, C(B)=5, C(C)=10, C(D)=Inf
  2. arg min C(w), w∈{B,C,D} ⇒ min = C(B) ⇒ M={A,B}
     C(C)=min(C(C), C(B)+l(B,C))=min(10,8) ⇒ B is min
     C(D)=min(C(D), C(B)+l(B,D))=min(Inf,16) ⇒ B is min
     ⇒ C(C)=8, C(D)=16
  3. arg min C(w), w∈{C,D} ⇒ min = C(C) ⇒ M={A,B,C}
     C(D)=min(C(D),C(C)+l(C,D))=min(16,10) ⇒ C is min
     ⇒ C(D)=10



- In practice, Dijkstra's algorithm realized by using forward search algorithm

Properties of link state routing

- Properties (+/-)
  - + stabilizes quickly
  - + does not generate much excess traffic
  - + responds quickly to topology changes or node failures
  - – amount of info stored in each node quite large (LSP for each node)
    - fundamental problem of scalable routing

- Distance vector vs. link state
  - in distance vector each node talks only to its neighbors and tells everything it has learned (entire routing table, even though info may not be accurate)
  - in link state, each node talks to all other nodes, but it tells them only what it knows for sure (state of its own directly connected links)

Open Shortest Path First (OSPF)

- One of the most widely used link state routing protocols

- Additional features
  - authentication of routing messages (password, cryptographic encryption)
  - provides additional hierarchy (scalability)
    - domain can be partitioned into areas
    - routing based on areas (not on all networks within an area)
  - load balancing
  - supports use of multiple cost metrics based on TOS field (QoS support)
    - not widely used

Outline

- Intradomain routing
  - distance vector routing (RIP)
  - link state routing (OSPF)
  - determining link costs
- Routing in global Internet
  - mechanisms: subnetting and classless routing (CIDR)
  - interdomain routing (BGP)
- Routing private and public IP addresses (NAT)

Metrics (1)

- Several metrics tested in development of ARPANET
  - also superiority of link state over distance vector demonstrated in ARPANET
- Original ARPANET metric
  - number of packets enqueued on each link
  - took neither latency or bandwidth into consideration
    - just moves packets towards shortest queues

Metrics (2)

- New ARPANET metric
  - stamp each incoming packet with its arrival time (AT)
  - record departure time (DT)
  - Delay = (DT - AT) + Transmit + Latency
    - (DT-AT) = (random) queuing delay
    - Transmit = packet transmission delay
    - Latency = length of the link
  - link cost = average delay over some time period (10 seconds)

- Performance
  - worked well under light load (Transmit and Latency dominate delay)
  - instability under heavy load
    - congestion ⇒ traffic routed away from link ⇒ link becomes idle ⇒ all traffic routed back ⇒ congestion ⇒ ...

Metrics (3)

- Specific problems with “New ARPANET metric”
  - range of variation is too wide
    - 9.6 Kbps highly loaded link can appear 127 times costlier than 56 Kbps lightly loaded link
    - can make a 127-hop path look better than 1-hop
  - no limit in reported delay variation

- Fine tuning (revised ARPANET metric)
  - compressed dynamic range: e.g. congested link cost max 3 x idle link cost
  - replaced delay with link utilization
    - link utilization affects metric only in moderate to high loads
    - otherwise metric dominated by constant Transmit and Latency values

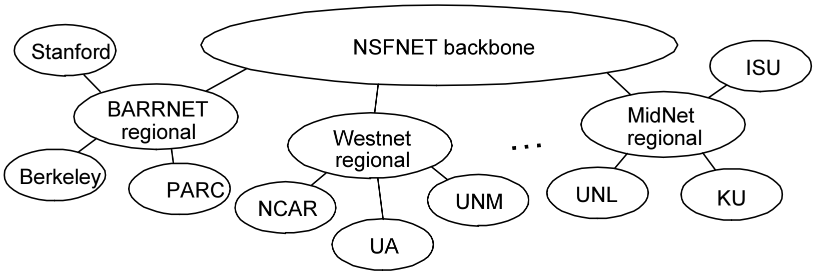
Outline

- Intradomain routing
  - distance vector routing (RIP)
  - link state routing (OSPF)
  - determining link costs
- Routing in global Internet
  - mechanisms: subnetting and classless routing (CIDR)
  - interdomain routing (BGP)
- Routing private and public IP addresses (NAT)

How to make routing scale

- Two problems:
  - routing protocol scalability
  - address space depletion
- Routing scalability
  - original Internet hierarchy: address consists of network and host part
  - thus far, for routing we assumed that routers need to know all networks
  - clearly not scalable as nof networks grows
    - routing tables do not scale
    - route propagation protocols do not scale
- Inefficient use of hierarchical address space
  - class C with 2 hosts (2/255 = 0.78% efficient)
  - class B with 256 hosts (256/65535 = 0.39% efficient)
  - class C network has only room for 256 hosts ⇒ medium sized companies prefer class B networks, but only 16 000 class B networks possible

Internet structure in the 90's (US view)



- Interconnects many different organizations
  - End user sites connected to regional service providers
  - Service providers connected to (government controlled) NSFNET backbone
- End user, service provider and back bone networks administratively independent
  - called **Autonomous Systems** (AS), each AS may run different routing protocol
- Structure can be utilized to make routing more scalable
- Task: minimize nof network numbers distributed with routing protocols and increase address assignment efficiency

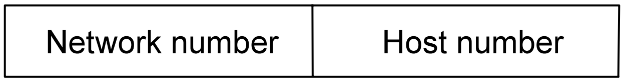
---

Subnetting

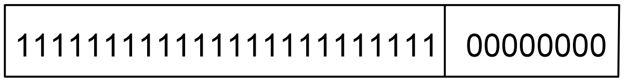
- Add another level to address/routing hierarchy: subnet
  - use one (same) IP network number for many physical networks called **subnets**
  - subnets should be geographically close to each other
    - routers in global Internet refer to the subnets with a single network number
    - i.e., there is only one route available to all subnets with same IP network number
  - example:
    - campus area with many physical networks
    - outside campus, to reach any subnet only need to know where campus is connected to rest of Internet

Subnet masking

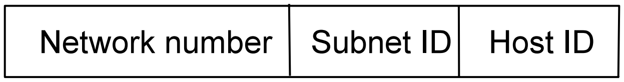
- **Subnet mask** defines a variable partition of IP address into
  - network number, subnet number and host number
  - subnets visible only within site
- Example: sharing a class B network address
  - split class B host part into subnet part and host part
  - in global Internet subnets are commonly addressed with the class B address



Class B address



Subnet mask (255.255.255.0)



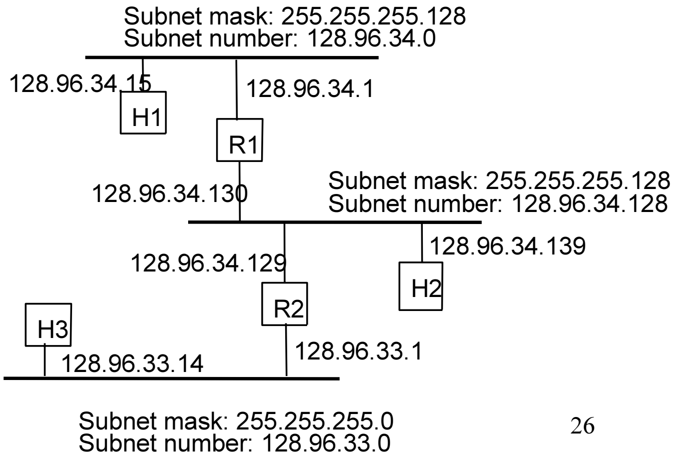
Subnetted address

Subnet example

- Hosts must be configured with **IP address and subnet mask**
- Subnet number = bitwise AND of (host addr, subnet mask)
- H1 wants to send data to H2
  - H1 takes AND(H2 IP address, H1 subnet mask)
  - result different than H1 subnet number
  - packet sent to R1
  - R1 takes AND(H2 IP address, all subnet masks)
  - R1 gets match with subnet 128.96.34.128 and forwards on Interface 1
- ARP remains largely unchanged by subnetting, but routing tables change

Forwarding table at router R1

Subnet Number	Subnet Mask	Next Hop
128.96.34.0	255.255.255.128	Interface 0
128.96.34.128	255.255.255.128	Interface 1
128.96.33.0	255.255.255.0	R2



Subnetting additional features and summary

- Additional features/consequences:
  - not necessary for all 1s in subnet mask to be contiguous (its usefulness not clear and not recommended in practise)
  - can put multiple subnets on one physical network (for administrative reasons)
  - different parts of Internet see things differently (routers inside campus see subnets, which are not visible outside)
- Benefits:
  - subnetting improves address assignment efficiency by letting us not use an entire class B or C address every time a new physical network is added
  - helps in aggregating routing information (a subnetted network appears to the outside Internet as a single network=single routing entry in tables)
- Subnetting supported by RIPv2 and OSPF-2

Supernetting

- Problem with subnetting:
  - any corporation with more than 255 hosts needs a class B address
  - ⇒ class B address depletion
- Solution: CIDR (Classless Inter-Domain Routing)
  - ... also called supernetting
  - minimizes amount of route info through aggregation and breaks rigid address boundaries between classes
  - idea: assign block of contiguous network numbers to nearby networks
    - restrict block sizes to powers of 2
- Result: we need routing protocols that support “classless” addresses
  - for example BGP-4
  - network numbers represented by (value, length) pairs, length=nw prefix length
  - all routers must understand CIDR addressing

Supernetting continued

- Observation:
  - subnetting used to share one network number among multiple physical networks
  - CIDR aggregates all network numbers assigned to an AS to one
- Possible to aggregate routes repeatedly if addresses assigned properly
  - if two corporations have adjacent 20-bit network prefixes, the service provider can advertise a single route with 19-bit prefix to both networks
- Changes in IP forwarding required by use of CIDR
  - with CIDR prefix length can be 2-32 bits
    - address format: network number/prefix length, e.g., 171.69/16
  - for a given network address it is possible to have several matching prefixes
    - address 171.69.10.3 would match prefixes 171.69 and 171.69.10
  - rule is to use the longest match for forwarding
    - longest match contains most specific information

Outline

- Intradomain routing
  - distance vector routing (RIP)
  - link state routing (OSPF)
  - determining link costs
- Routing in global Internet
  - mechanisms: subnetting and classless routing (CIDR)
  - interdomain routing (BGP)
- Routing private and public IP addresses (NAT)

Interdomain routing

- Internet organized as a collection of interconnected ASs
  - each AS administratively independent from other ASs
  - ASs provide an additional way to hierarchically aggregate routing information
- Routing problem decomposition
  - routing inside an AS (intradomain routing)
    - AS can use any routing protocol as its intradomain routing protocol (RIP, OSPF, even static routing)
  - routing between ASs (interdomain routing)
    - routing deals with sharing reachability information between ASs

---

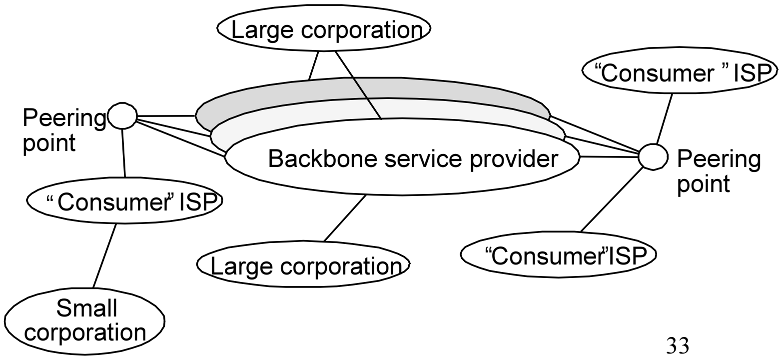
Interdomain routing continued

- Route information propagation:
  - "know a smarter router" (called default route)
  - hosts know local router
  - local routers know site routers
  - site routers know core router
  - core routers know everything
  - idea: by using default routes, routers do not necessarily need to know much about routes leading outside a given AS
- Main problem:
  - managing the amount of route information in backbone routers
- First approach for interdomain routing: EGP
  - designed for tree-structured Internet
  - concerned with reachability, not optimal routes
  - Problem: modern Internet no longer tree structured!

Internet structure today (US view)

- Internet consists of multiple backbones (service provider networks)
- Different sites (ASs) connected to the Internet in arbitrary ways
  - large corporations can connect to one or more backbones
- ISPs mainly exist to provide consumers access to the Internet
- Providers connect via peering points:
  - "an interconnection of public networks that allows customers of one network to exchange traffic to customers directly on the second ISP's network"

AS Types
- stub AS: has a single connection to one other AS
  - carries local traffic only
- multihomed AS: has connections to more than one AS
  - refuses to carry transit traffic
- transit AS: has connections to more than one AS
  - carries both transit and local traffic



BGP-4: Border Gateway Protocol overview

- Interdomain routing protocol for modern Internet: BGP-4
  - assumes Internet consists of arbitrarily connected ASs

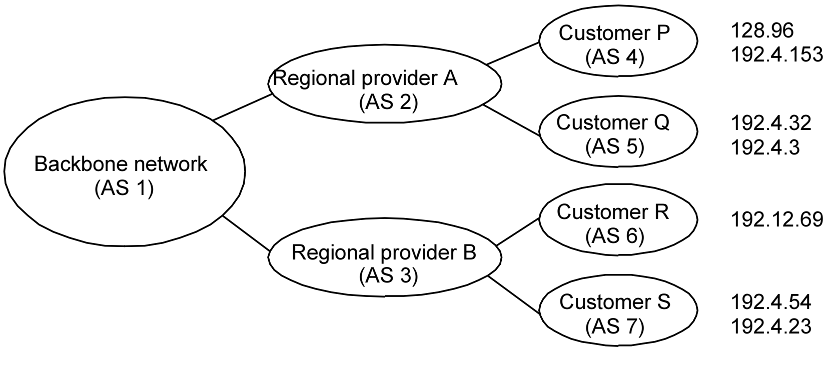
- Interdomain routing problem
  - goal to find loop free paths (reachability more important than optimality)
  - why not optimal?
    - scale (>50 000 routes in back bone)
    - ASs independent (can use any routing protocol and metric)
    - trust: provider A may not trust provider B's route information
    - policy routing: provider A wants to prefer some routes over others

BGP-4 overview continued

- Each AS has:
  - one or more border routers (called gateways)
    - routers through which packets enter and leave AS
  - one border router chosen as “BGP speaker”, communicates with other ASs
  - BGP speaker advertises:
    - local networks
    - other reachable networks (transit AS only)
- Each non-stub AS has a unique id
  - 16 bit numbers assigned by central authority
- BGP advertises complete paths as a list of ASs to reach a particular network
  - necessary for policy routing and loop detection (if speaker sees own id in any path list $\Rightarrow$ loop)
  - possible to make negative advertisements (to with draw routes)
  - update format: prefix/length, e.g., 192.4.16/20

BGP Example

- Speaker for AS2 advertises reachability to P and Q
  - network 128.96, 192.4.153, 192.4.32, and 192.4.3, can be reached directly from AS2
- Speaker for backbone advertises
  - networks 128.96, 192.4.153, 192.4.32, and 192.4.3 can be reached along the path (AS1, AS2).
- Speaker can also cancel previously advertised paths (link failures etc.)



BGP and intradomain routing

- BGP-4 in short
  - BGP-4 specifies how reachability info is exchanged among ASs
  - BGP speakers get enough info to compute loop free routes, but how to choose the best is not specified
- How all other routers in an AS get the route information of gateway router(s)
  - in a stub AS, other routers need only know “default” router (=border router)
  - in a multihomed AS (regional provider), border router A can inject routing info about a customer AS into the AS intradomain routing protocol
    - “A has link to network 192.4.54/24 of cost X”
    - other routers inside provider AS learn that to reach above prefix, send packets to router A
  - in the backbone problem is that there is too much route info to be injected
    - Interior-BGP used to distribute route info from AS speakers to other backbone routers

---

Routing in today’s Internet summary

- Internet is a collection of interconnected ASs
- Routing divided into intra-/interdomain
- Intradomain inside an AS
  - usually based on OSPF
    - shortest path routing based on a link state algorithm
  - about metrics: based on simple static metrics (~ 1/link_bandwidth)
    - dynamic metrics too “unstable”; earlier just history of development of Internet routing metrics (before Internet became public)
- Interdomain routing between Ass
  - BGP-4 is the currently used interdomain routing protocol
  - based on advertising loopless paths to other ASs
  - addresses of each AS based on CIDR addressing format
  - subnetting can be used to further divide CIDR based network address into smaller chunks (sub-domains)

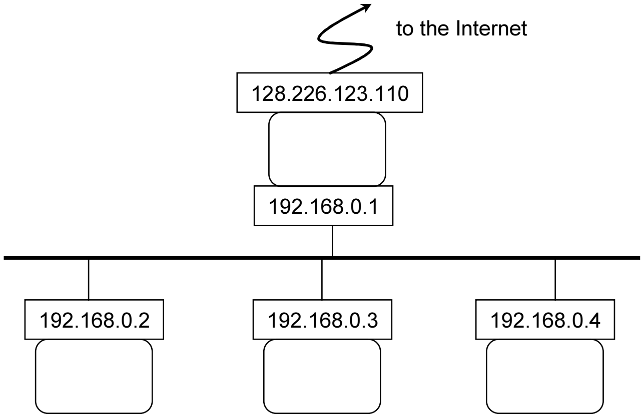
Outline

- Intradomain routing
  - distance vector routing (RIP)
  - link state routing (OSPF)
  - determining link costs
- Routing in global Internet
  - mechanisms: subnetting and classless routing (CIDR)
  - interdomain routing (BGP)
- Routing private and public IP addresses (NAT)

Private network addresses

- Problem:
  - even with dynamic IP assignment, one may end up having more hosts than IP numbers assigned
  - public IP addresses cost money ⇒ need to minimize nof such addresses
- Solution:
  - cheat and use unassigned numbers
  - on the Internet there is an agreement that some addresses are not routed to the backbone (RFC1918)
    - 10.0.0.0/8
    - 192.168.0.0/16
    - 172.16.0.0/12
  - make sure that these numbers are only visible inside of your subnet!
- These addresses are called private networks and are used for NAT (Network Address Translation)
  - NAT technology widely used in current Internet
  - Linux: IP Masquerading
  - Windows: Internet Connection Sharing
  - Elsewhere: Network Address Translation

NAT (Network Address Translation)



- One machine has a legitimate IP address and is connected to the Internet
- Internally, other machines are assigned private addresses
- NAT machine establishes “proxy” connections on behalf of the other machines
- Only NAT machine is visible to the outside