Chapter 4: Network Layer

Chapter goals:

- understand principles behind network layer services:
  - routing (path selection)
  - dealing with scale
  - how a router works
  - advanced topics: IPv6, multicast
- instantiation and implementation in the Internet

Overview:

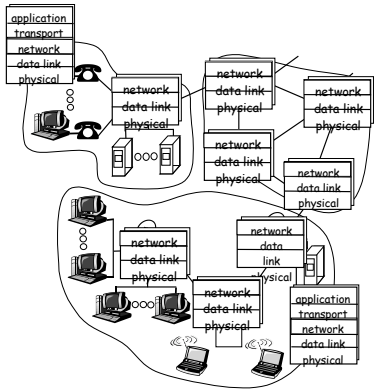
- network layer services
- routing principle: path selection
- hierarchical routing
- IP
- Internet routing protocols reliable transfer
  - intra-domain
  - inter-domain
- what's inside a router?
- IPv6
- multicast routing

Network layer functions

- transport packet from sending to receiving hosts
- network layer protocols in *every* host, router

three important functions:

- *path determination:* route taken by packets from source to dest. *Routing algorithms*
- *switching:* move packets from router's input to appropriate router output
- *call setup:* some network architectures require router call setup along path before data flows



Network service model

Q: What *service model* for "channel" transporting packets from sender to receiver?

- guaranteed bandwidth?
- preservation of inter-packet timing (no jitter)?
- loss-free delivery?
- in-order delivery?
- congestion feedback to sender?

service abstraction

The most important abstraction provided by network layer:

virtual circuit or datagram?

Virtual circuits

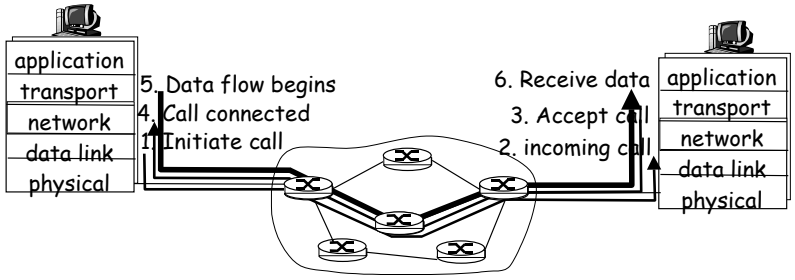
"source-to-dest path behaves much like telephone circuit"

- performance-wise
- network actions along source-to-dest path

- call setup, teardown for each call *before* data can flow
- each packet carries VC identifier (not destination host OD)
- *every* router on source-dest path s maintain "state" for each passing connection
  - transport-layer connection only involved two end systems
- link, router resources (bandwidth, buffers) may be *allocated* to VC
  - to get circuit-like perf.

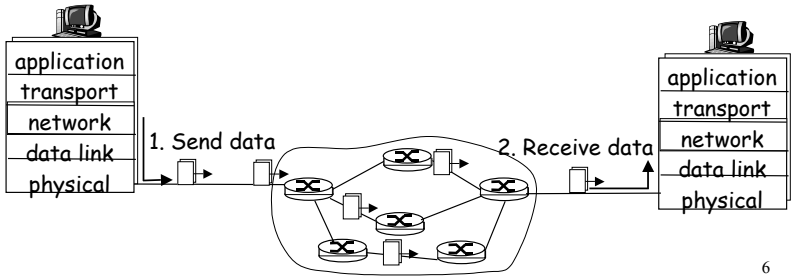
Virtual circuits: signaling protocols

- **used to setup, maintain teardown VC**
- **used in ATM, frame-relay, X.25**
- **not used in today's Internet**



Datagram networks: the Internet model

- **no call setup at network layer**
- **routers: no state about end-to-end connections**
  - no network-level concept of "connection"
- **packets typically routed using destination host ID**
  - packets between same source-dest pair may take different paths



Datagram or VC network: why?

Internet

- **data exchange among computers**
  - "elastic" service, no strict timing req.
- **"smart" end systems (computers)**
  - can adapt, perform control, error recovery
  - simple inside network, complexity at "edge"
- **many link types**
  - different characteristics
  - uniform service difficult

ATM

- **evolved from telephony**
- **human conversation:**
  - strict timing, reliability requirements
  - need for guaranteed service
- **"dumb" end systems**
  - telephones
  - complexity inside network

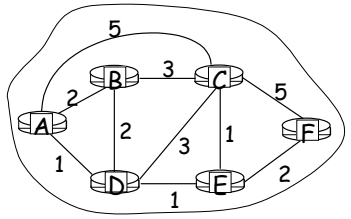
Routing

Routing protocol

Goal: determine "good" path (sequence of routers) thru network from source to dest.

Graph abstraction for routing algorithms:

- **graph nodes are routers**
- **graph edges are physical links**
  - link cost: delay, $ cost, or congestion level



"good" path:
typically means minimum cost path
other def's possible

Routing Algorithm classification

Global or decentralized information?

Global:

- **all routers have complete topology, link cost info**
- **"link state" algorithms**

Decentralized:

- **router knows physically-connected neighbors, link costs to neighbors**
- **iterative process of computation, exchange of info with neighbors**
- **"distance vector" algorithms**

Static or dynamic?

Static:

- **route selection based on static metrics (hop count)**

Dynamic:

- **Route selection based on performance metrics (traffic load, delay, queue size)**
  - periodic update
  - in response to link cost changes

Two fundamental routing approaches

- **Link State:**
  - Each node knows the whole graph
  - Advertise each link to everybody
  - Find path by running the same algorithm
- **Distance Vector:**
  - Each node has a vector of distances to all others nodes
  - Advertise only your vector of distances to neighbors
    - *Forward info only if your vector of distance changed*
  - Determine next hop according to distance
    - *Each node only is aware of the next hop*

Distance Vector Routing Algorithm

iterative:

- **continues until no nodes exchange info.**
- ***self-terminating*: no "signal" to stop**

asynchronous:

- **nodes need *not* exchange info/iterate in lock step!**

distributed:

- **each node communicates *only* with directly-attached neighbors**

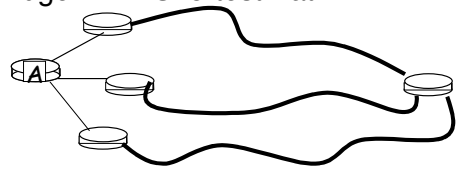
Distance Table data structure

- **each node has its own**
- **row for each possible destination**
- **column for each directly-attached neighbor to node**
- **example: in node X, for dest. Y via neighbor Z:**

$$D^X(Y,Z) = \text{distance } \textit{from } X \textit{ to } Y, \textit{ via } Z \text{ as next hop} = c(X,Z) + \min_w \{D^Z(Y,w)\}$$

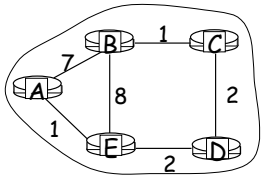
Main Idea

Edge + Shortest Path



- **My shortest path thru each neighbor to X**
  - Shortest path from a neighbor to X
  - Plus my cost to reach the neighbor
- **My shortest path overall**
  - minimum among shortest path from each neighbor

Distance Table: example



$D^E(C,D) = c(E,D) + \min_w\{D^D(C,w)\}$
$= 2+2 = 4$

From node E: to go to C
What is the cost thru D?

$D^E()$ — Via

destination	A	B	D
A	(1)	14	5
B	7	8	(5)
C	6	9	(4)
D	4	11	(2)

Distance table gives routing table

$D^E()$ — cost to destination via

destination	A	B	D
A	(1)	14	5
B	7	8	(5)
C	6	9	(4)
D	4	11	(2)

destination	Outgoing link to use, cost
A	A,1
B	D,5
C	D,4
D	D,2

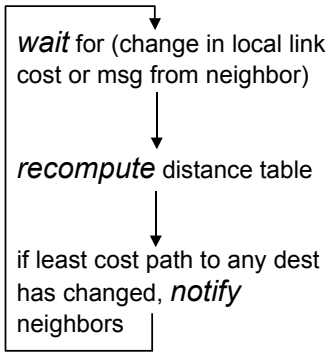
Distance table → Routing table

Distance Vector Routing: overview

Iterative, asynchronous:
each local iteration caused by:
- **local link cost change**
- **message from neighbor: its least cost path change from neighbor**

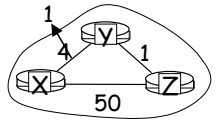
Distributed:
- **each node notifies neighbors *only* when its least cost path to any destination changes**
  - neighbors then notify their neighbors if necessary

Each node:

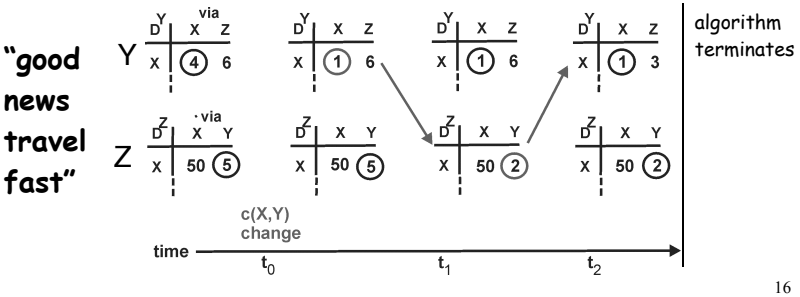


Distance Vector: link cost changes

Link cost changes:
1. node detects local link cost change updates distance table (line 15)
2. if cost change in least cost path, notify neighbors (lines 23,24)

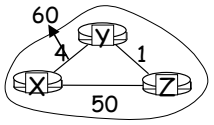


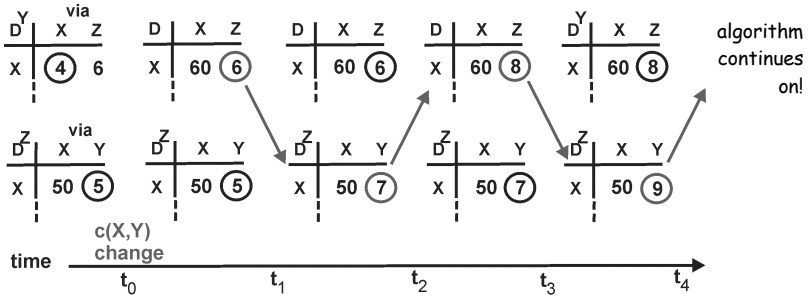
"good news travel fast"



Distance Vector: link cost changes

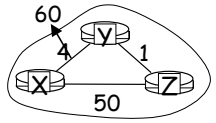
Link cost changes:
bad news travels slow -
"count to infinity" problem



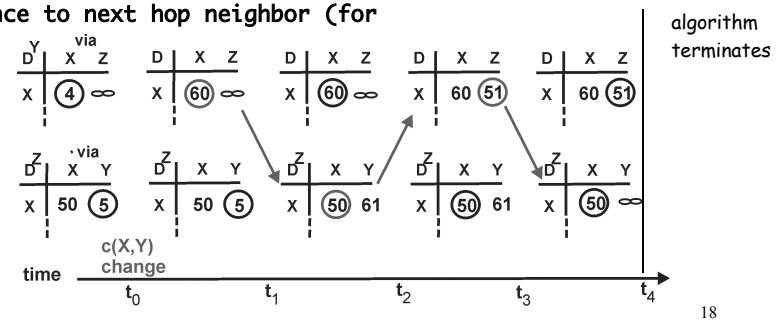


Distance Vector: poisoned reverse

Poison Reverse: If Z routes through Y to get to X :
Z tells Y its distance to X is infinite (so Y won't route to X via Z)



Split Horizon: don't advertise at all distance to next hop neighbor (for path)



Link-State Routing Algorithm

- **Net topology, link costs known to all nodes**
  - accomplished via "link state broadcast"
  - all nodes have same info
- **Each node computes locally: cost paths from to all other nodes**
  - gives routing table for that node

Dijkstra's algorithm, discussion

Algorithm complexity: n nodes

- **each iteration: need to check all nodes, w, not in N**
- **n*(n+1)/2 comparisons: O(n**2)**
- **more efficient implementations possible: O(nlogn)**

Oscillations possible:

- **if link cost = amount of carried traffic**

Link State Routing

- **Initial state : similar to distance vector i.e., state of link to neighbors known (up/down).**
- **Goal: To find the path of least cost to destination.**
- **Basic Idea --**
  - Every node knows how to reach its neighbors.
  - Disseminate the info is disseminated to every node,
  - every node has the complete map of the network
  - All nodes execute the same algorithm for paths

Two Mechanisms

- **Reliable dissemination of link state information**
  - The process is called reliable flooding.
- **Calculation of routes using the collected information**
  - The computation is based on Dijkstra's algorithm.

Reliable Flooding

- **Process of making sure that all the nodes participating in the link state routing protocol get a copy of the link-state info. from all other nodes.**
- **Each node sends out link-state info. on its directly connected links.**
- **Each node that receives this, forwards it to all links**
  - except the incoming one.

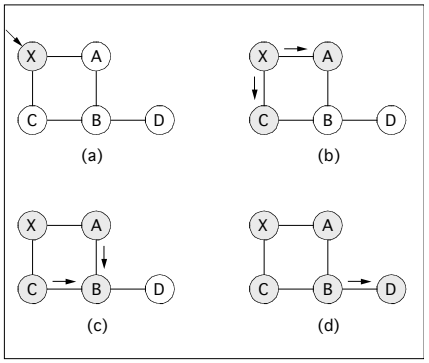
Link State Information

- **Each node creates a link-state packet (LSP) that contains:**
  - ID of the node that created LSP
  - a list of directly connected nodes and the cost to each node.
  - sequence number
  - TTL

} → **for reliability**

An Example

- X receives info from node Y.
- X checks to see if it already has an update for that link. If it does, it compares the sequence number in the new LSP to the one stored.
- If New seq no < Old sequence number, then, discard LSP.
- Else -- store LSP and send the LSP to all neighbors except the one that sent the LSP.



Dissemination of LSPs

- LSPs are sent
  - periodically (upon the expiry of a timer order of hours)
  - Or may be triggered due to a change in topology (as in RIP).
- Topology change in a directly connected link
  - Failures detected by link layer protocol by using what are known as “HELLO” packets -- probes to determine if neighbor is alive.
- Sequence numbers help in identifying new info and TTL helps in ensuring that packets don’t percolate in the network indefinitely.

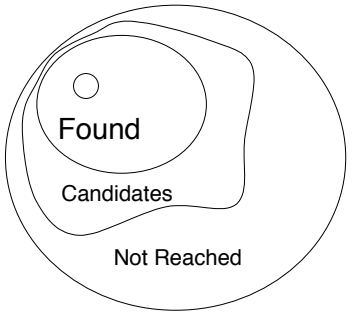
Graph abstraction

- Used for computation of shortest path using Dijkstra’s.
- Let N denote the set of nodes in the graph.
- l(i,j) denotes the non-negative cost or weight associated with the edge between nodes i and j.
- l(i,j) = ∞ if there is no edge between i and j.
- Remember -- each node has entire map of network.

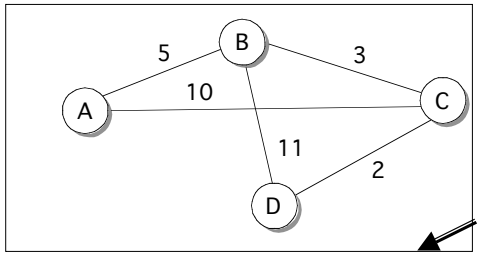
The Dijkstra’s Algorithm

- Let “s” the node that executes the algorithm.
- Algorithm maintains M --> set of nodes whose shorter path I found
- C(n) -- cost of the path from s to n.
- L(v,w) cost of link from v to w
- Reminder of Dijkstra algorithm

```
M = {s}
for each node n in N- {s}, C(n) = L(s,n);
while (N ≠M),
  find node w in N - M, with cost C(w)
  M = M U {w}
  for each n in (N-M)
      C(n) = min( C(n), C(w) + L(w,n) )
```



An Example



• Let s = A.

• Initially M = A.

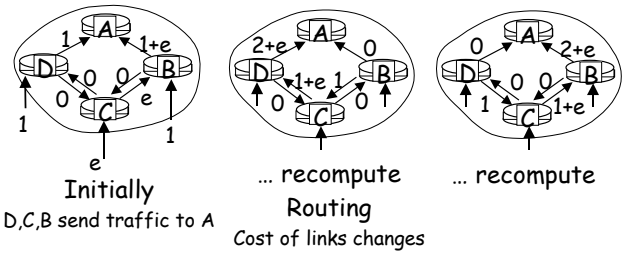
• Now, minimum cost node is B (note that only two nodes can be considered). So add B to M.

• At next step, look at neighbors of A and B (we have C and D). We add C (via B).

•Then, we consider D, (note that we consider the link via C as well -- finally that is added.

• Finally, the tree with links AB, BC and CD is created.

Routing Oscillations with dynamic weights



- **If link weight is dynamic (function of load)**
- **Paths can change after we route**
  - Example: D,C,B send traffic to A continuously
  - Once we route, best path changes!

Two fundamental routing approaches

- **Link State:**
  - Each node knows the whole graph
  - Advertise each link to everybody
  - Find path by running the same algorithm
- **Distance Vector:**
  - Each node has a vector of distances to all others
  - Advertise only your vector of distances to neighbors
    - *Forward info only if your vector of distance changed*
  - Determine next hop according to distance
    - *Each node only is aware of the next hop*

Summary: Distributed Routing Techniques

Link State	Vectoring
Topology information is flooded within the routing domain	Each router knows little about network topology
Best end-to-end paths are computed locally at each router.	Only best next-hops are chosen by each router for each destination network.
Best end-to-end paths determine next-hops.	Best end-to-end paths result from composition of all next-hop choices
Advertises: link info	Advertises: path info, distance
Works only if policy is shared and uniform	Does not require uniform policies at all routers
Examples: OSPF, IS-IS	Examples: RIP, BGP

Routing in the Global Internet

Hierarchical Routing

Our routing study thus far - idealization
all routers identical
network "flat"
... *not* true in practice

scale: with 50 million destinations:

- **can't store all dest's 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**

Hierarchical Routing

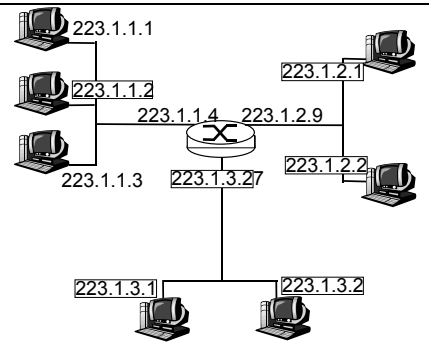
- **Aggregate routers into regions, "autonomous systems" (AS)**
- **Routers in same AS run same routing protocol**
  - "intra-AS" routing protocol
  - different AS can run different intra-AS protocol
- **Aggregatable addresses**

gateway routers

- **special routers in AS**
- **run intra-AS routing protocol with all other routers in AS**
- ***also* responsible for routing to destinations outside AS**
  - run *inter-AS routing* protocol with other gateway routers

IP Addressing: introduction

- **IP address: 32-bit identifier for host, router *interface***
- ***interface:* connection between host, router and physical link**
  - router's typically have multiple interfaces
  - host may have multiple interfaces
  - IP addresses associated with interface, not host, router

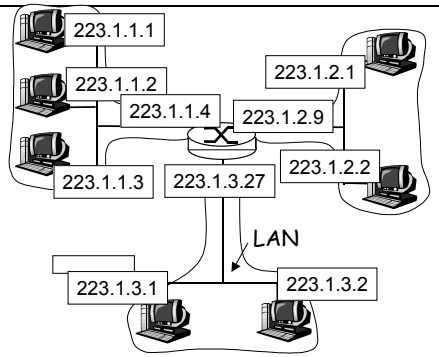


223.1.1.1 = 11011111 00000001 00000001 00000001

223 1 1 1

IP Addressing

- **IP address:**
  - network part (high order bits)
  - host part (low order bits)
- ***What's a network ?***
  (from IP address perspective)
  - device interfaces with same network part of IP address
  - can physically reach each other without intervening router

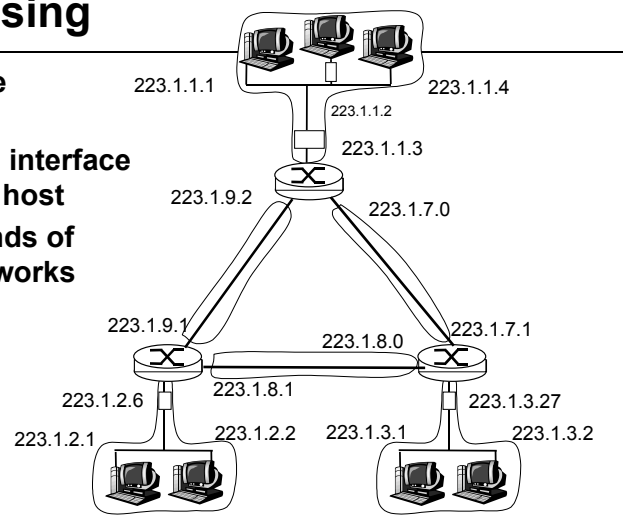


network consisting of 3 IP networks (for IP addresses starting with 223, first 24 bits are network address)

IP Addressing

How to find the networks?

- **Detach each interface from router, host**
- **create "islands of isolated networks**

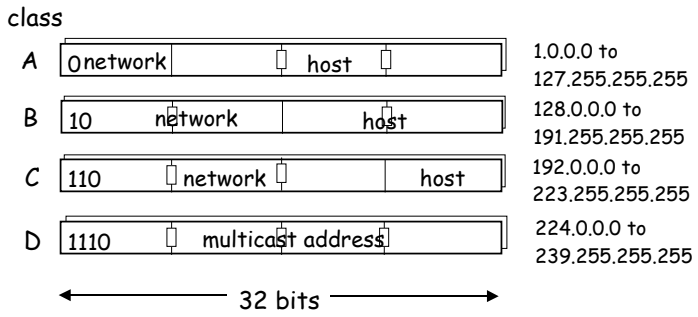


Interconnected system consisting of six networks

IP Addresses

given notion of "network", let's re-examine IP addresses:

"class-full" addressing:

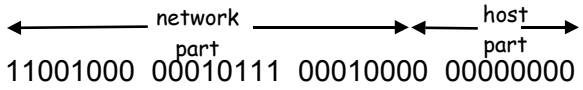


IP addressing: need for change

- **Problems with classful addressing:**
  - inefficient use of address space, address space exhaustion
  - e.g., class B net allocated enough addresses for 65K hosts, even if only 2K hosts in that network

IP addressing: CIDR

- **CIDR: Classless InterDomain Routing**
- **network portion of address of arbitrary length**
- **address format: a.b.c.d/x, where x is # bits in network portion of address**



200.23.16.0/23

IP Addresses and Prefixes

- **IP addresses have 32 bits: 4 octets of bits (IPv4)**
- **A prefix is a group of IP addresses**
- **128.32.101.5 is an IP address (32 bits)**
- **128.32.0.0/16 is a prefix of the 16 first bits:**
  - 128.32.0.0 – 128.32.255.255 (2^16 addresses)
- **128.32.4.0/24 is a prefix of the 24 first bits - longer**

IP addresses: how to get one?

Hosts (host portion):

- **hard-coded by system admin in a file**
- **DHCP: Dynamic Host Configuration Protocol: dynamically get address: "plug-and-play"**
  - host broadcasts "DHCP discover" msg
  - DHCP server responds with "DHCP offer" msg
  - host requests IP address: "DHCP request" msg
  - DHCP server sends address: "DHCP ack" msg

IP addresses: how to get one?

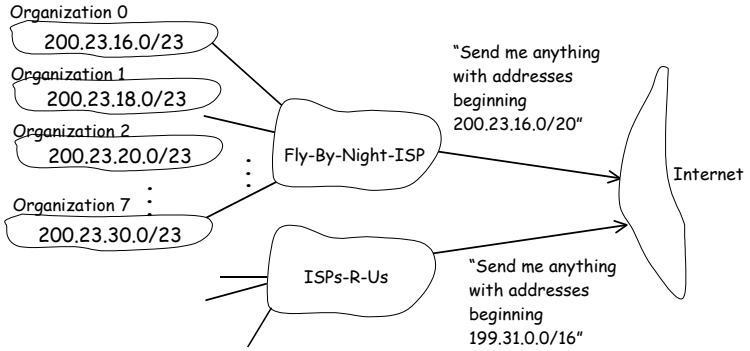
Network (network portion):

- **get allocated portion of ISP's address space:**

ISP's block	11001000 00010111 00010000 00000000	200.23.16.0/20
Organization 0	11001000 00010111 00010000 00000000	200.23.16.0/23
Organization 1	11001000 00010111 00010010 00000000	200.23.18.0/23
Organization 2	11001000 00010111 00010100 00000000	200.23.20.0/23
...	…..	…. ….
Organization 7	11001000 00010111 00011110 00000000	200.23.30.0/23

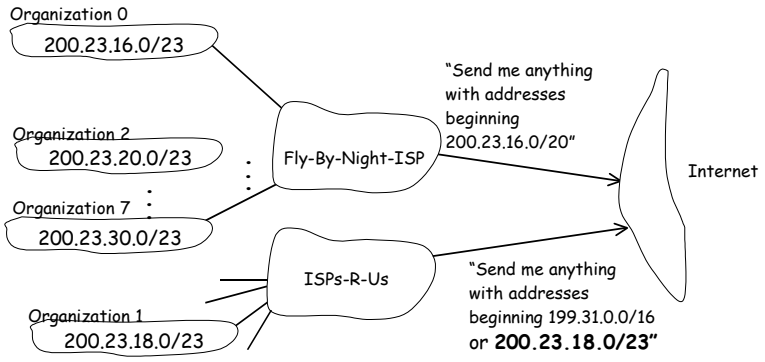
Hierarchical addressing: route aggregation

Hierarchical addressing allows efficient advertisement of routing information:



Hierarchical addressing: more specific routes

ISPs-R-Us has a more specific route to Organization 1



Routing is Based on Prefixes

- **A BGP Routing table has prefixes for entries**
- **For a IP address of a packet, find <u>longest match</u>**
- **Example: packet IP 128.32.101.1**
- **Matching:**
- **128.1.1.4 matches the first 8 bits – no match!**
- **128.32.0.0/16 match for 16 bits**
- **128.32.101.0/24 is a longer match**

Prefix Matching in More Detail

- **For a IP address of a packet, find <u>longest match</u>**
- **Example: Compare**
  - packet IP 128.32.101.1
  - With 128.32.0.0/16
  - IP : 01000000. 001000000. 01100101 .00000001
  - Mask : 11111111. 111111111. 00000000 .00000000
  - AND : 01000000. 001000000. 00000000 .00000000
  - Prefix : 01000000. 001000000. 00000000. 00000000
  - Equal? Yes
  - We have a match of length 16

IP addressing: the last word...

Q: How does an ISP get block of addresses?

A: ICANN: Internet Corporation for Assigned Names and Numbers

- allocates addresses
- manages DNS
- assigns domain names, resolves disputes

ICMP: Reporting errors

- **What happens if things cannot proceed as expected?**
- **ICMP: Internet Control Message Protocol**
- **Used by hosts, routers, gateways to communication network-level information**
  - error reporting: unreachable host, network, port, protocol
  - echo request/reply (used by ping)

ICMP: Internet Control Message Protocol

- **Network-layer "above" IP:**
  - ICMP msgs carried in IP datagrams
- **ICMP message: type, code plus first 8 bytes of IP datagram causing error**

Type	Code	description
0	0	echo reply (ping)
3	0	dest. network unreachable
3	1	dest host unreachable
3	2	dest protocol unreachable
3	3	dest port unreachable
3	6	dest network unknown
3	7	dest host unknown
4	0	source quench (congestion control - not used)
8	0	echo request (ping)
9	0	route advertisement
10	0	router discovery
11	0	TTL expired
12	0	bad IP header

Let's see how things work in practice

Getting a datagram from source to dest.

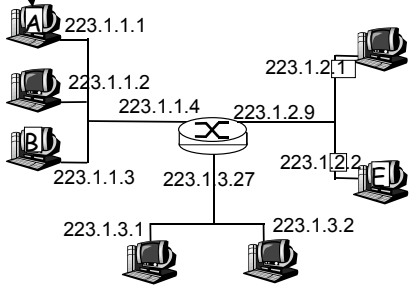
IP datagram:

misc fields	source IP addr	dest IP addr	data

datagram remains unchanged, as it travels source to destination
addr fields of interest here

routing table in A

Dest. Net.	next router	Nhops
223.1.1		1
223.1.2	223.1.1.4	2
223.1.3	223.1.1.4	2



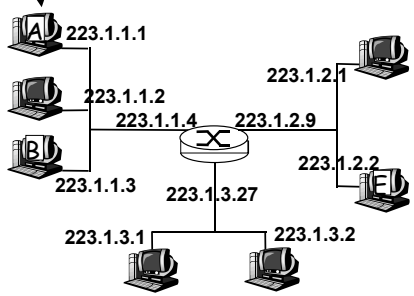
Getting a datagram from source to dest.

misc fields	223.1.1.1	223.1.1.3	data

Starting at A, given IP datagram addressed to B:
look up net. address of B
find B is on same net. as A
link layer will send datagram directly to B inside link-layer frame

B and A are directly connected

Dest. Net.	next router	Nhops
223.1.1		1
223.1.2	223.1.1.4	2
223.1.3	223.1.1.4	2



Getting a datagram from source to dest.

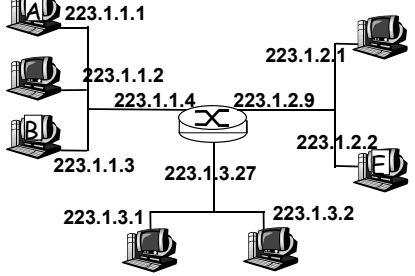
misc fields	223.1.1.1	223.1.2.2	data

Starting at A, dest. E:
look up network address of E
E on *different* network

A, E not directly attached

routing table: next hop router to E is 223.1.1.4
link layer sends datagram to router 223.1.1.4 inside link-layer frame
datagram arrives at 223.1.1.4
continued…..

Dest. Net.	next router	Nhops
223.1.1		1
223.1.2	223.1.1.4	2
223.1.3	223.1.1.4	2



Getting a datagram from source to dest.

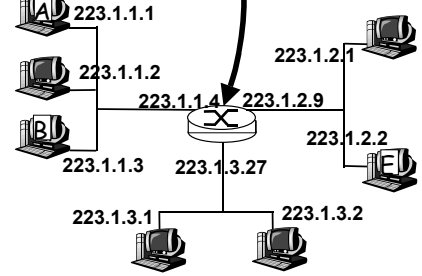
misc fields	223.1.1.1	223.1.2.2	data

Arriving at 223.1.4, destined for 223.1.2.2
look up network address of E
E on *same* network as router's interface 223.1.2.9

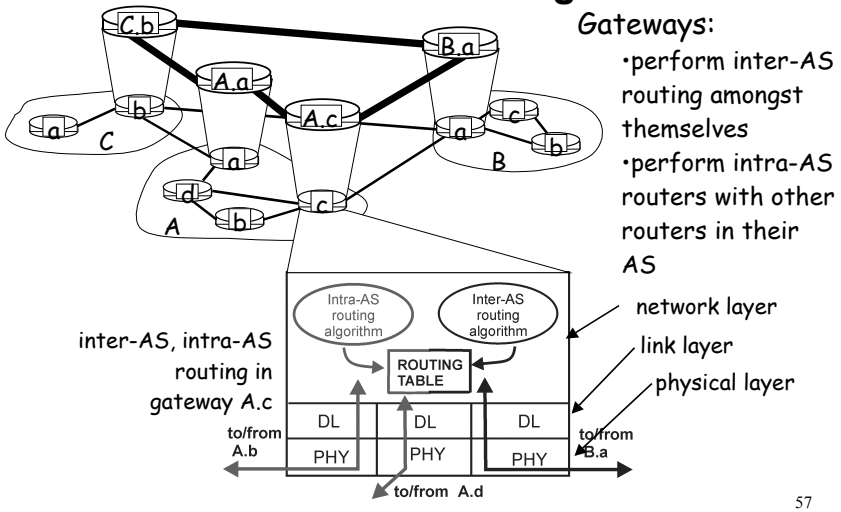
router, E directly attached

link layer sends datagram to 223.1.2.2 inside link-layer frame via interface 223.1.2.9
datagram arrives at 223.1.2.2!!!
(hooray!)

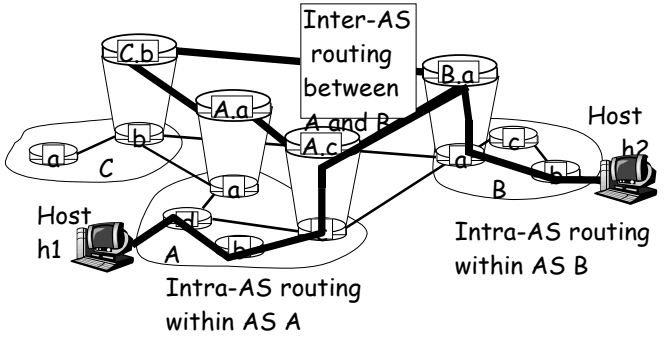
Dest. network	next router	Nhops	interface
223.1.1	-	1	223.1.1.4
223.1.2	-	1	223.1.2.9
223.1.3	-	1	223.1.3.27



Intra-AS and Inter-AS routing



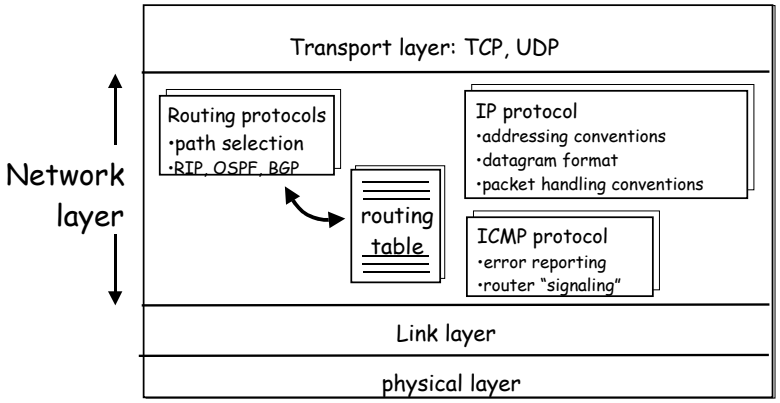
Intra-AS and Inter-AS routing



We'll examine specific inter-AS and intra-AS Internet routing protocols shortly

The Internet Network layer

Host, router network layer functions:



Measurements in the Internet

- **Difficulties in measuring**
- **Measuring tools (traceroute)**
- **Misc issues**

Measuring and Modeling Is not Easy

- **Constantly changing environment**
- **How much data is enough**
  - Recently: we need to measure more than 24h!
- **How frequently should I be measuring?**
- **Are the measurements representative?**

Operation versus Measurements

- **Operators do not care about**
  - Measurements
  - Academic Research
- **Why?**
  - Takes away resources
  - Can create problems
  - Complicates their lives
- **Luckily, there are measurement centers**
  - CAIDA, NLANR, routeviews, RIPE

Types of Measurement Tools

- **Application level:**
  - Install application agents at two measuring entries
  - REALITI tool from UCR
  - More control over process
- **Network level:**
  - Use the Internet control functionality (ICMP)
  - Trick the network to provide information

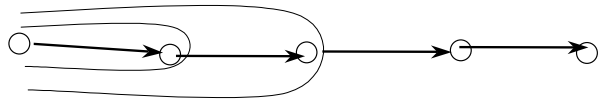
Ping: the tool

- **Uses ICMP ECHO_REQUEST datagram to elicit an ICMP ECHO_RESPONSE from a host or gateway**
- **Reports**
  - Round trip time
  - Packets loss
- **Many available options: packet type, size etc**
- **Limitation: >1sec measurement frequency**
- **Read manual: man ping**

Traceroute: the tool

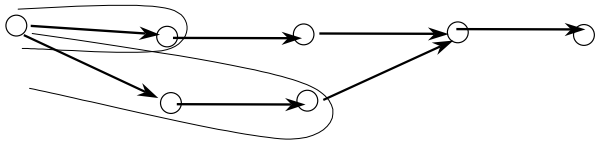
- **Traceroute measures**
  - the path and the round trip time
- **Traceroute: ingenious (ab)use of the network layer by Van Jacobson**
- **Main ideas:**
  - send “bad” packets to receive ICMP: “packet died”
  - Recursive probing to identify the path
  - Send three packets at a time
- **Read manual: man traceroute**

The ingenuity of traceroute



- **Send a packet for every hop of the path**
- **Set TTL = 1, packet expires, ICMP returns**
- **Increase TTL by one, and repeat**
- **At the destination, port number is wrong: return an ICMP packet, port not found**

Traceroute: Some Limitations



- **In traceroute, you may be exploring multiple paths without knowing it**
- **Delays for each part of the path correspond to different measurements: ie they don’t sum up**

Identifying The Router Topology

- **Several efforts rely on multiple traceroute**
  - Govindan et al INFOCOM 2000
  - Cheswick and Burch Internet Mapping Project
- **Main idea:**
  - Do thousands of traceroutes
  - Collect all adjacent nodes
  - Generate a graph

Router Graphs: A Complication

- **Routers have multiple IP addresses**
  - One for each interface
- **How do we resolve this?**
- **Only heuristics exist [Govindan]**
- **Heuristic: Send packets to one interface and hope that they will respond with the other interface**
  - Typically, router responds with IP of interface the packet came on

End

Proposals

- **Overall: decent**
- **Not enough motivation/background**
  - All related papers
- **Not enough thought on what you will do**
  - Spend an evening thinking what you will do and how
- **Try to clarify goals early and talk to me**
- **Photocopy them, and give me the original**
  - It's the contract

Practical Tips

- **The earlier, the better**
- **Talk to me early**
- **Look for tools, data, previous work**
  - It can save you a lot of time in the long run
- **Try to focus on a topic**

Side Note

- **Important things in research:**
  - Asking the right questions
  - Identifying the right topic
  - Context of research
    - *Motivation*
    - *Previous and related work*
    - *Importance of problem*
  - Thoroughness of work

Quiz

- **What is the main role of routing?**
- **Describe a centralized routing protocol**
  - Main operation functions
  - Comment on complexity, robustness
- **Describe a table driven routing protocol**
  - Main functions
  - Comment on complexity
- **How can the Internet routing scale?**
  - Describe the elements that make it work
  - What does a router need to keep in its memory
  - Describe a routing table
- **What happens to a packet when it arrives at a router?**
  - What kind of “hardware” does it go through?
  - Where does the delay and loss come into play?