Lecture 5
Network Layer

Computer Networks

The slides are made by J.F Kurose and K.W. Ross,
adapted by Phuong Vo and Tan Le

Instructor: Le Duy Tan, Ph.D.

Email: ldtan@hcmiu.edu.vn

Chapter 5: network layer

chapter goals:

- understand principles behind network layer services:
  - network layer service models
  - forwarding versus routing
  - how a router works
  - routing (path selection)

Chapter 5: outline

5.1 introduction

5.2 what's inside a router

5.3 IP: Internet Protocol
- datagram format
- DHCP
- ICMP
- IPv6

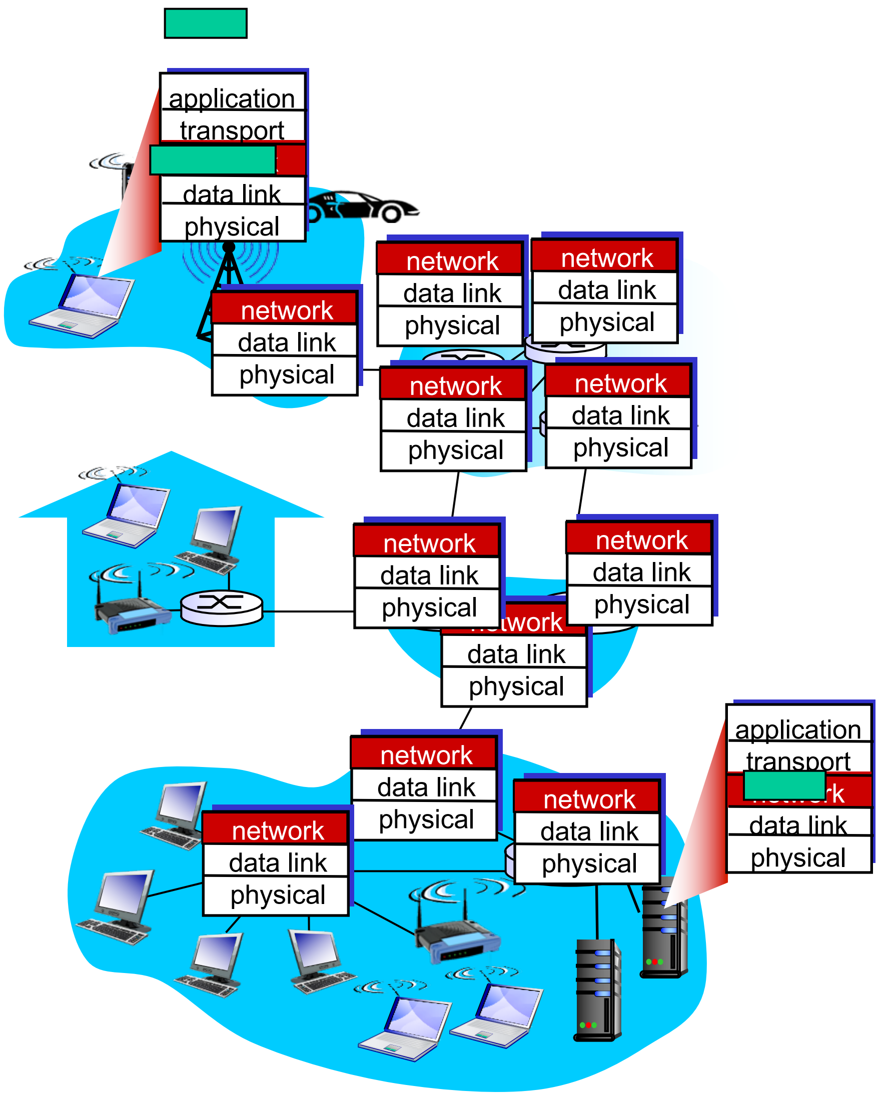
5.4 static routing

5.5 routing algorithms
- link state
- distance vector
- hierarchical routing

5.6 routing in the Internet
- RIP
- OSPF
- BGP

Network layer

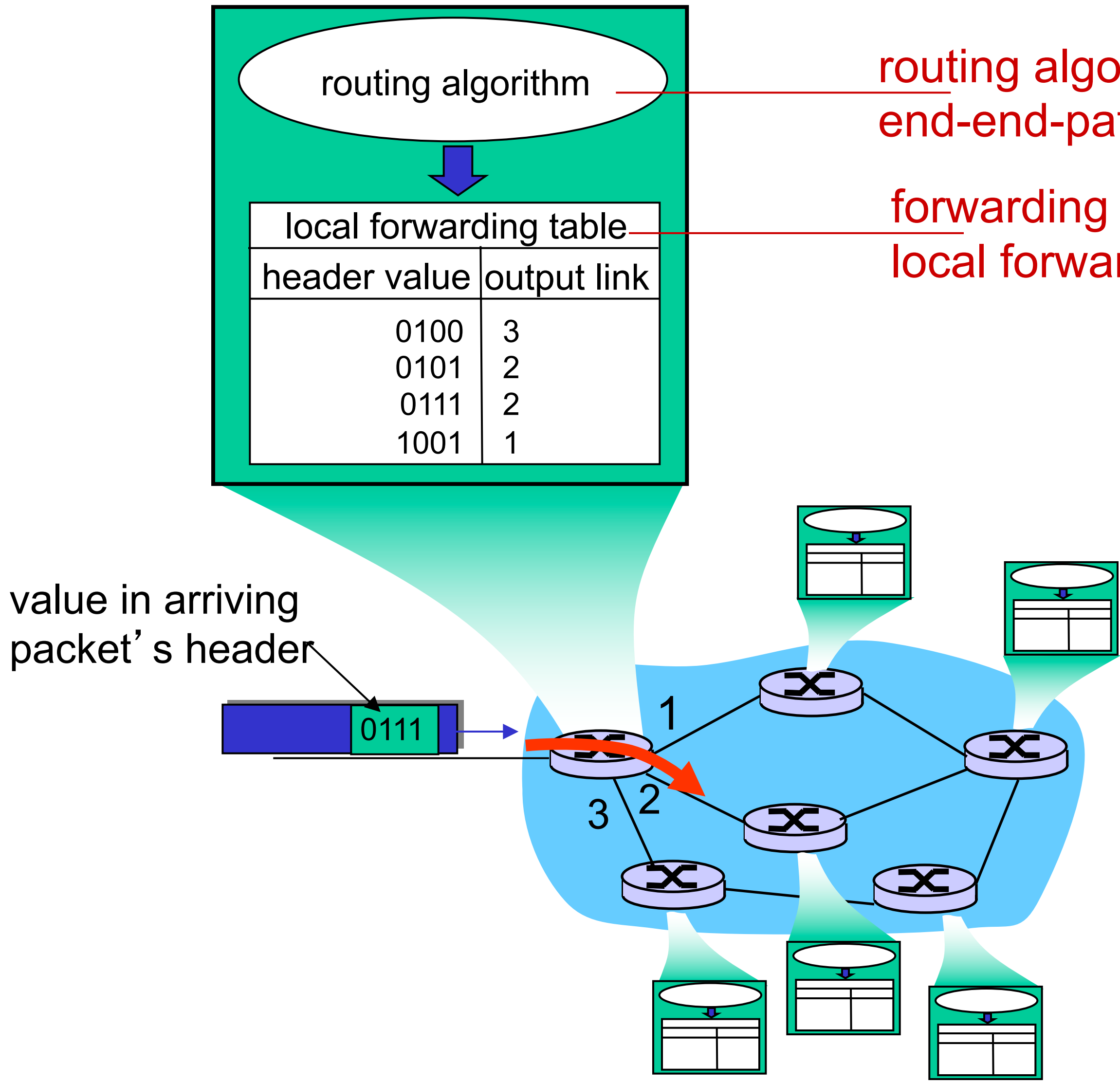
- transport segment from sending to receiving host
- on sending side encapsulates segments into datagrams
- on receiving side, delivers segments to transport layer
- network layer protocols in *every* host, router
- router examines header fields in all IP datagrams passing through it



Two key network-layer functions

- *forwarding:* move packets from router's input to appropriate router output
- *routing:* determine route taken by packets from source to dest.
  - *routing algorithms*

Interplay between routing and forwarding



routing algorithm determines end-end-path through network

forwarding table determines local forwarding at this router

Chapter 5: outline

5.1 introduction

5.2 what's inside a router

5.3 IP: Internet Protocol

- datagram format
- DHCP
- ICMP
- IPv6

5.4 static routing

5.5 routing algorithms

- link state
- distance vector
- hierarchical routing

5.6 routing in the Internet

- RIP
- OSPF
- BGP

Connection, connection-less service

- *datagram* network provides network-layer *connectionless* service
- *virtual-circuit* network provides network-layer *connection* service
- analogous to TCP/UDP connecton-oriented / connectionless transport-layer services, but:
  - *service:* host-to-host
  - *no choice:* network provides one or the other
  - *implementation:* in network core

Network service model

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

example services for individual datagrams:

- guaranteed delivery
- guaranteed delivery with less than 40 msec delay

example services for a flow of datagrams:

- in-order datagram delivery
- guaranteed minimum bandwidth to flow
- restrictions on changes in inter-packet spacing

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 address)
- *every* router on source-dest path maintains "state" for each passing connection
- link, router resources (bandwidth, buffers) may be *allocated* to VC (dedicated resources = predictable service)

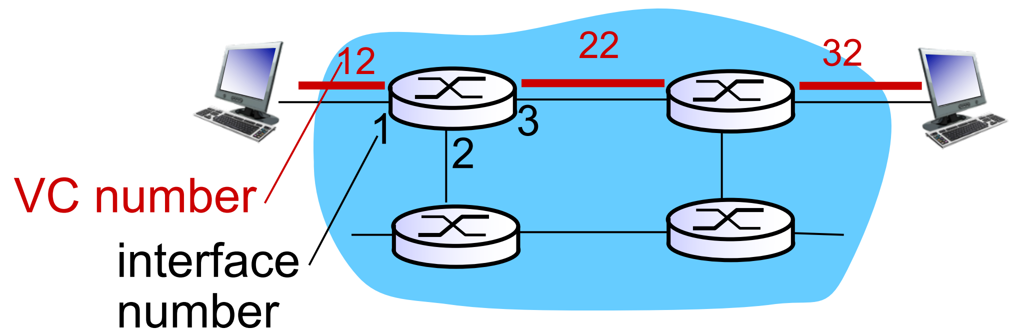
VC implementation

a VC consists of:

1. *path* from source to destination
2. *VC numbers*, one number for each link along path
3. *entries in forwarding tables* in routers along path

- packet belonging to VC carries VC number (rather than dest address)
- VC number can be changed on each link.
  - new VC number comes from forwarding table

VC forwarding table



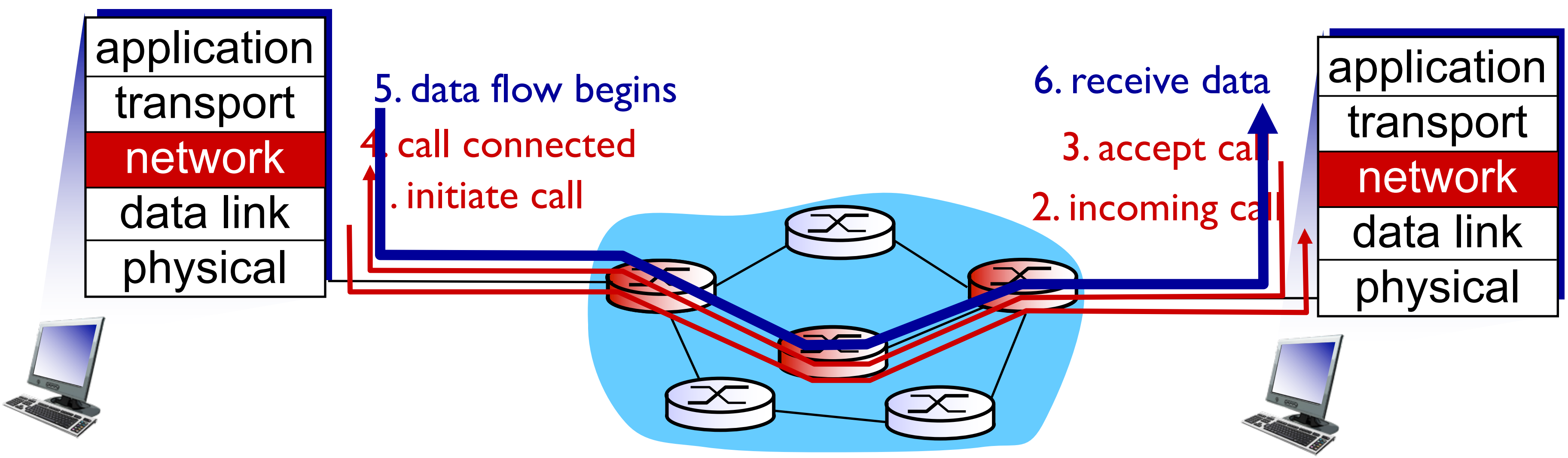
forwarding table in northwest router:

Incoming interface	Incoming VC #	Outgoing interface	Outgoing VC #
1	12	3	22
2	63	1	18
3	7	2	17
1	97	3	87
…	…	…	…

VC routers maintain connection state information!

Virtual circuits: signaling protocols

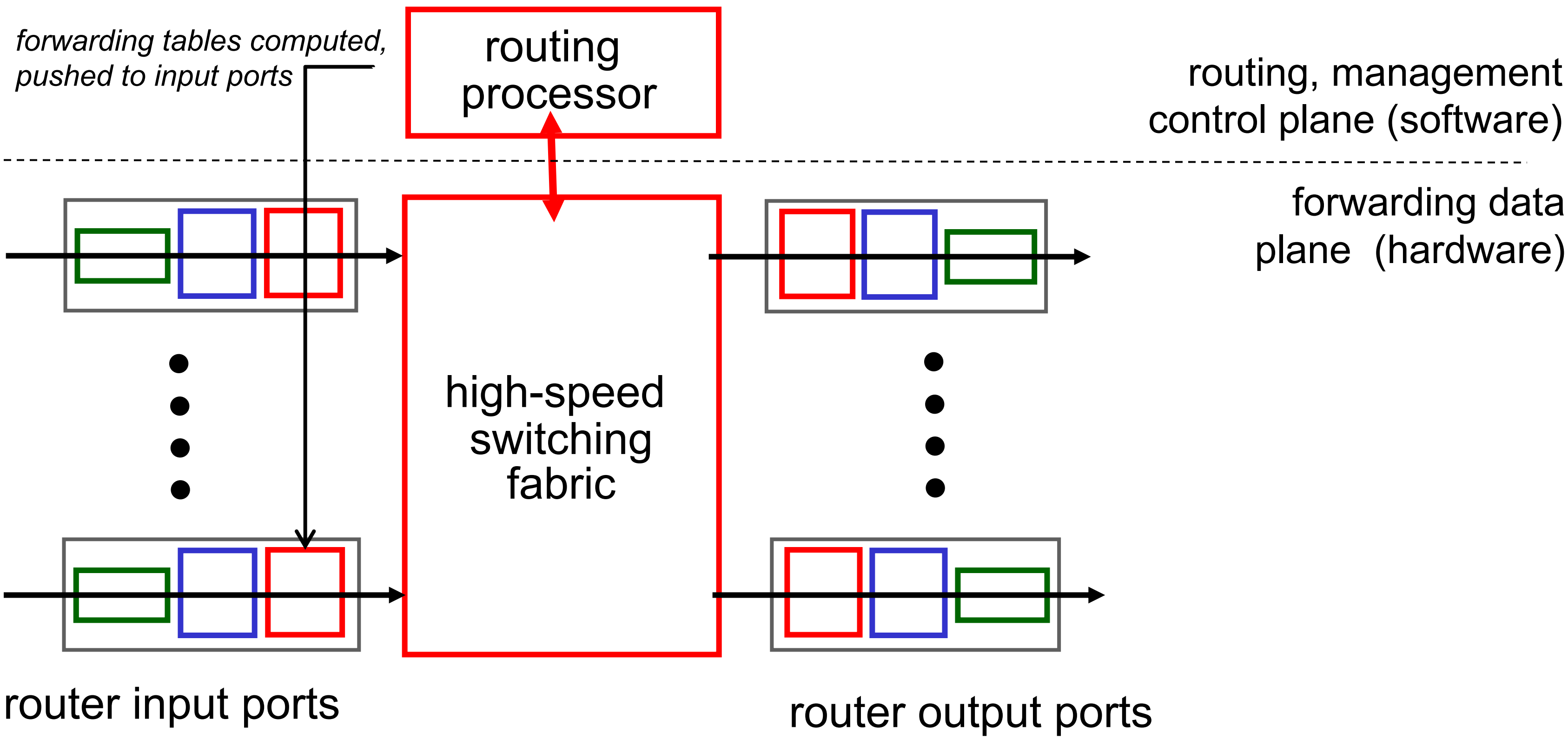
- used to setup, maintain teardown VC
- used in ATM, frame-relay, X.25
- not used in today's Internet



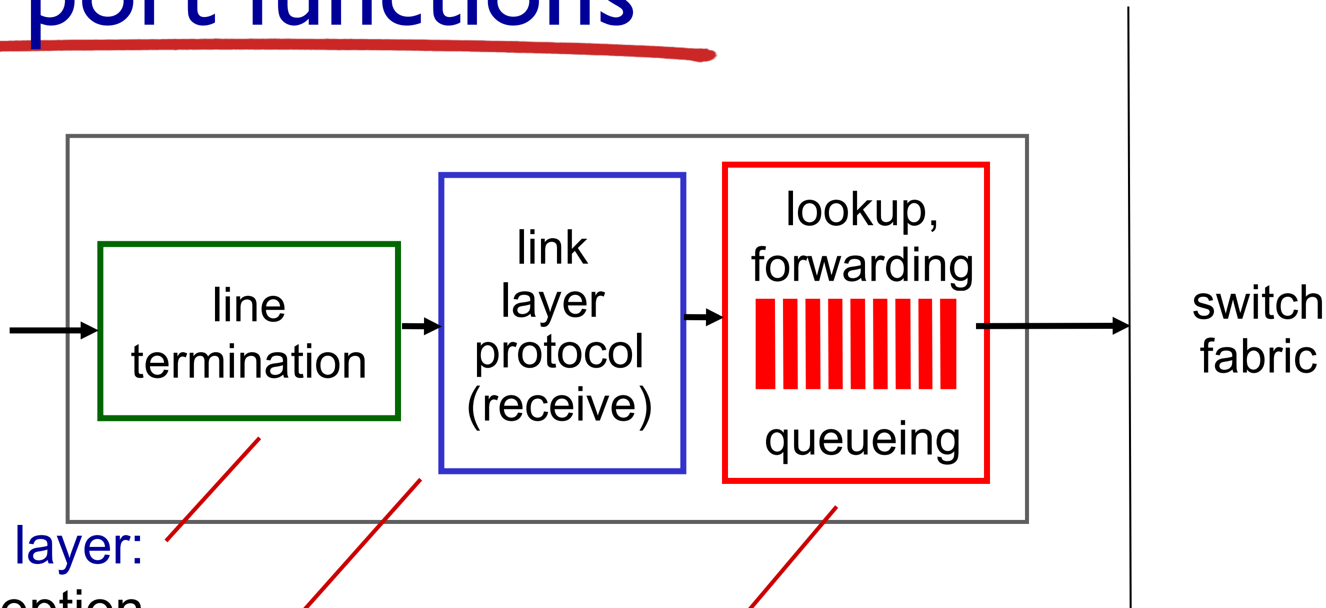
Router architecture overview

two key router functions:

- run routing algorithms/protocol (RIP, OSPF, BGP)
- *forwarding* datagrams from incoming to outgoing link



Input port functions



physical layer:
bit-level reception

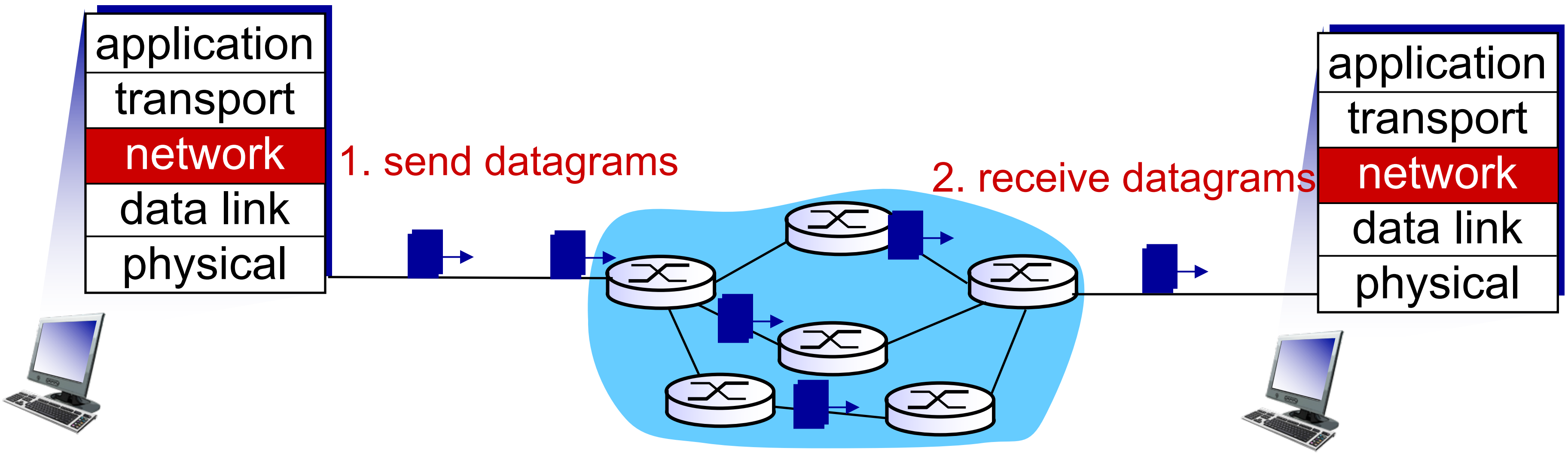
data link layer:
e.g., Ethernet
see chapter 5

decentralized switching:

- given datagram dest., lookup output port using forwarding table in input port memory *("match plus action")*
- goal: complete input port processing at 'line speed'
- queuing: if datagrams arrive faster than forwarding rate into switch fabric

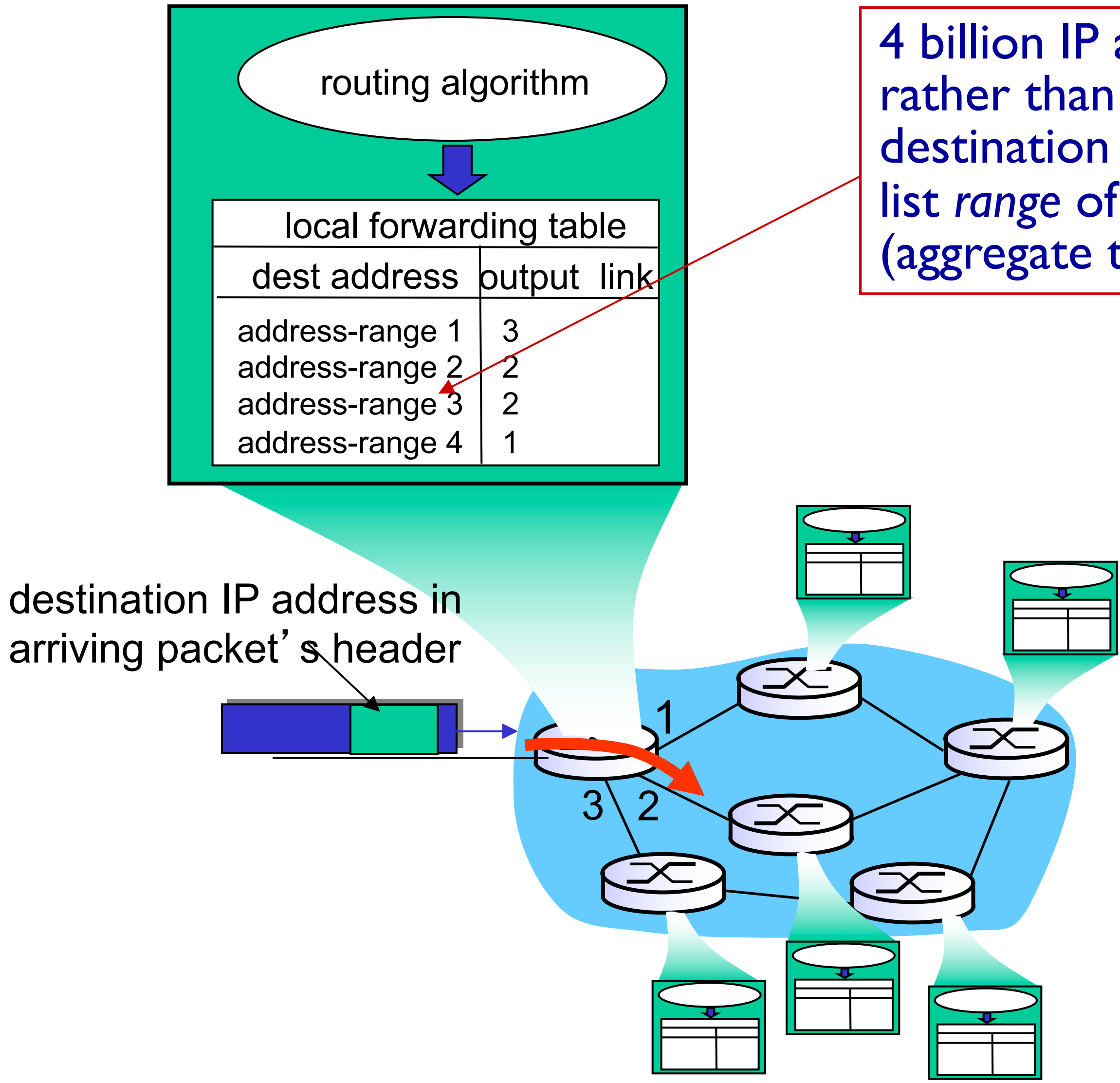
Datagram networks

- no call setup at network layer
- routers: no state about end-to-end connections
  - no network-level concept of "connection"
- packets forwarded using destination host address



Datagram forwarding table

4 billion IP addresses, so rather than list individual destination address list *range* of addresses (aggregate table entries)



Datagram forwarding table (Destination-based forwarding)

Destination Address Range	Link Interface

`11001000 00010111 00010000 00000000`
through
`11001000 00010111 00010111 11111111`
	0

`11001000 00010111 00011000 00000000`
through
`11001000 00010111 00011000 11111111`
	1

`11001000 00010111 00011001 00000000`
through
`11001000 00010111 00011111 11111111`
	2

otherwise	3

Longest prefix matching

longest prefix matching

when looking for forwarding table entry for given destination address, use *longest* address prefix that matches destination address.

Destination Address Range	Link interface
11001000 00010111 00010*** *********	0
11001000 00010111 00011000 *********	1
11001000 00010111 00011*** *********	2
otherwise	3

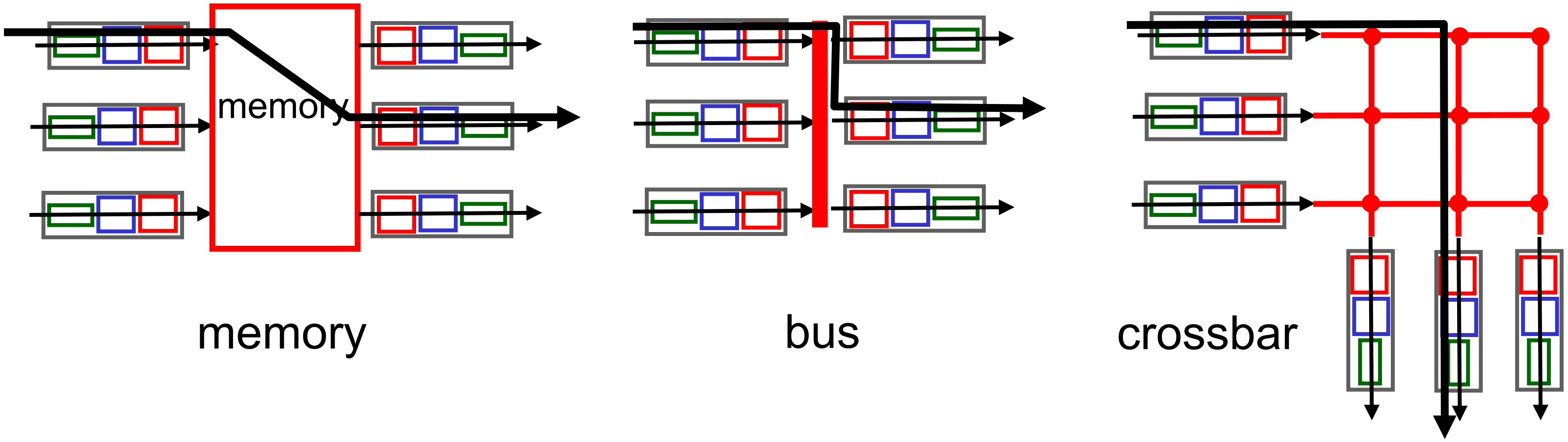
examples:

DA: 11001000 00010111 00010110 10100001 which interface?

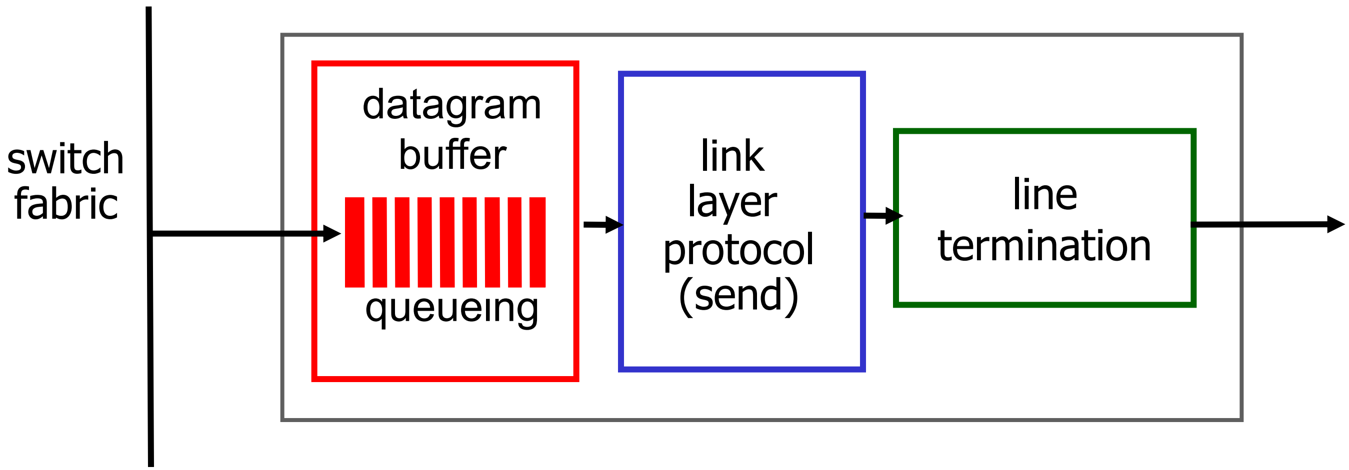
DA: 11001000 00010111 00011000 10101010 which interface?

Switching fabrics

- transfer packet from input buffer to appropriate output buffer
- switching rate: rate at which packets can be transfer from inputs to outputs
  - often measured as multiple of input/output line rate
  - N inputs: switching rate N times line rate desirable
- three types of switching fabrics



Output ports



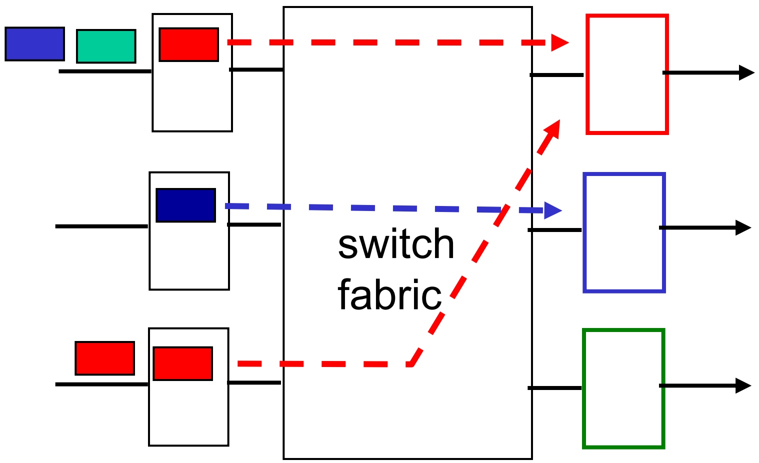
- *buffering* required from fabric faster rate

Datagram (packets) can be lost due to congestion, lack of buffers

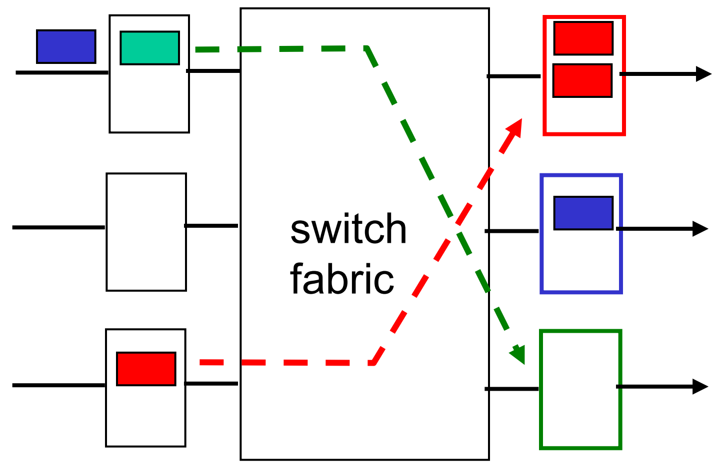
- *scheduling* datagrams

Priority scheduling – who gets best performance, network neutrality

Output port queueing



at *t*, packets more from input to output



one packet time later

- buffering when arrival rate via switch exceeds output line speed
- *queueing (delay) and loss due to output port buffer overflow!*

Chapter 5: outline

5.1 introduction

5.2 what's inside a router

5.3 IP: Internet Protocol

- datagram format
- DHCP
- ICMP
- IPv6

5.4 static routing

5.5 routing algorithms

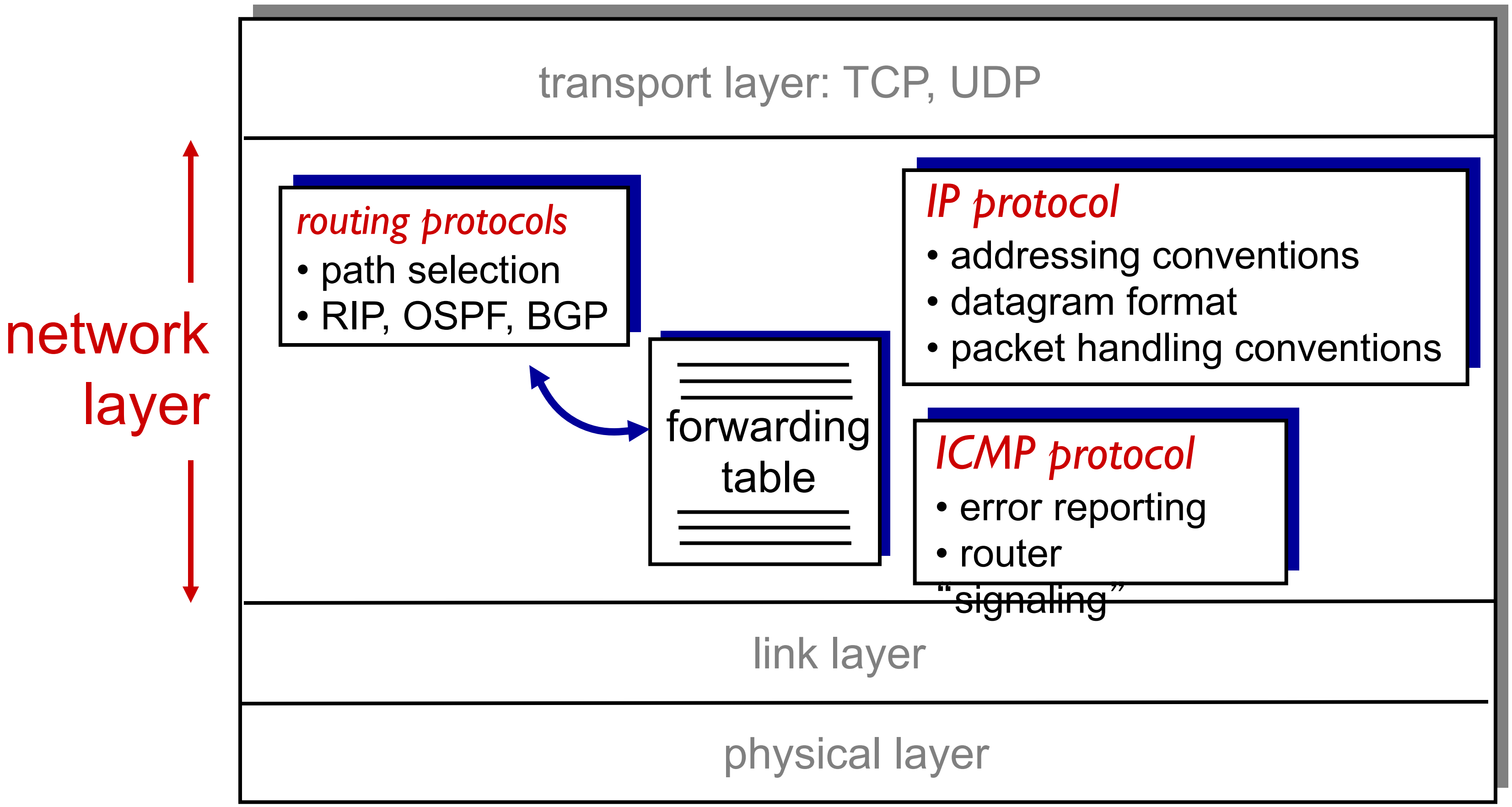
- link state
- distance vector
- hierarchical routing

5.6 routing in the Internet

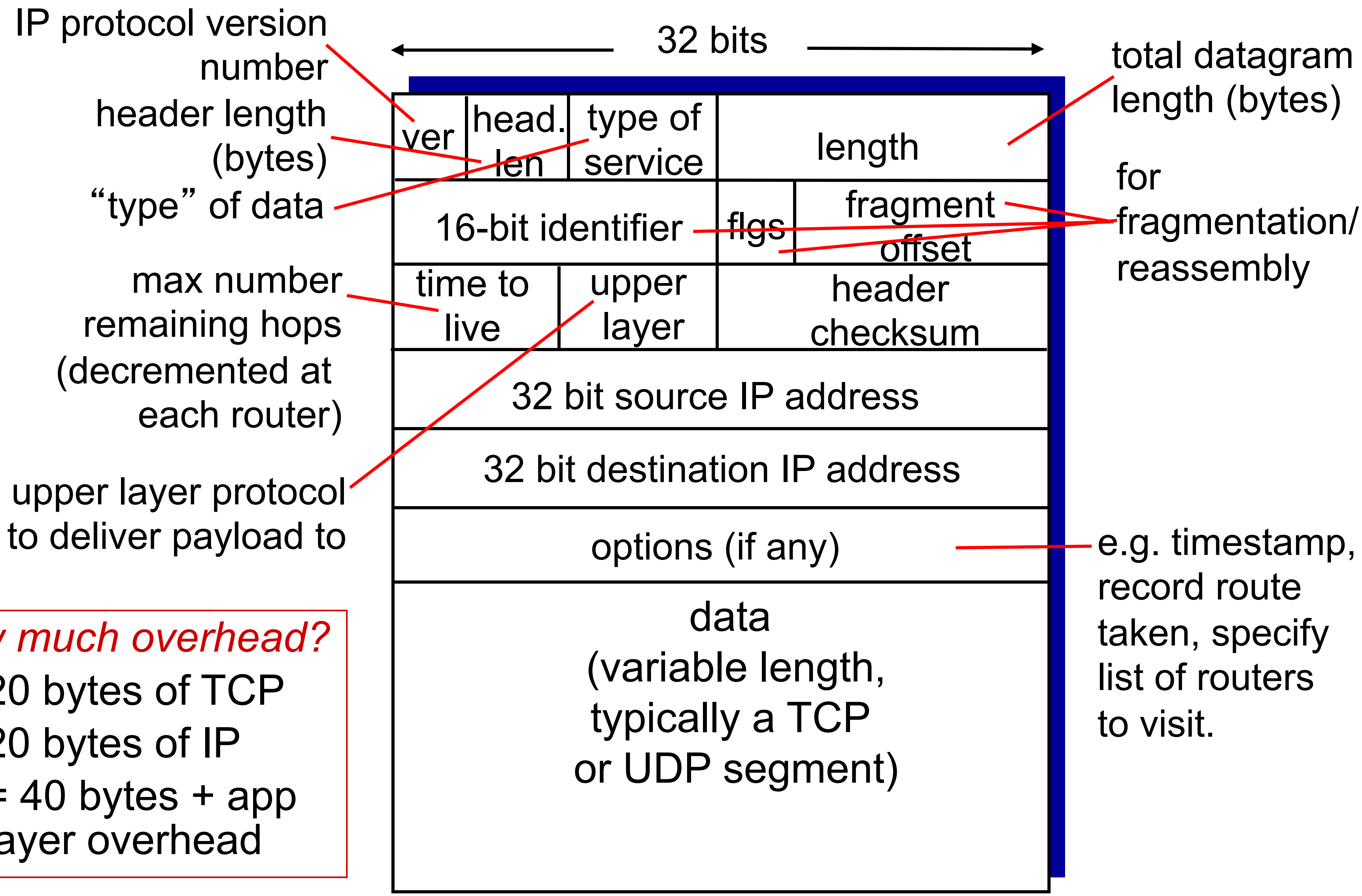
- RIP
- OSPF
- BGP

The Internet network layer

host, router network layer functions:



IP datagram format

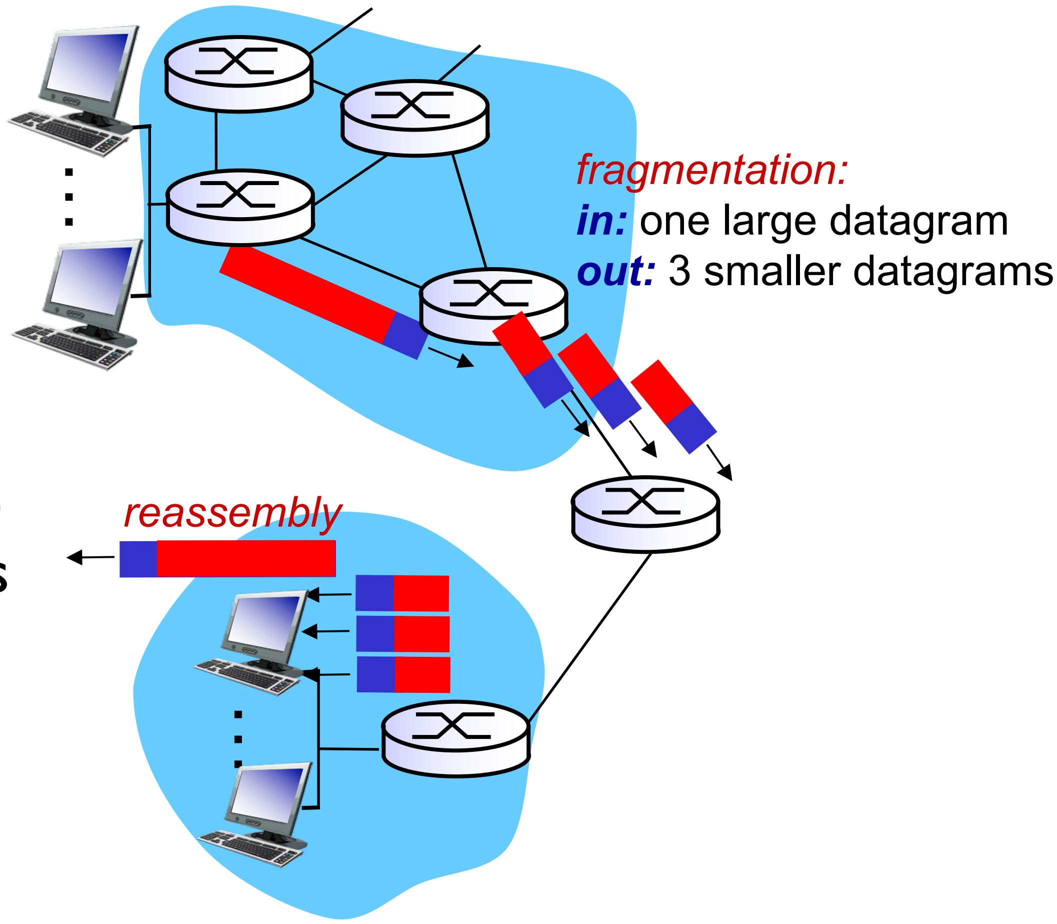


how much overhead?

- 20 bytes of TCP
- 20 bytes of IP
- = 40 bytes + app layer overhead

IP fragmentation, reassembly

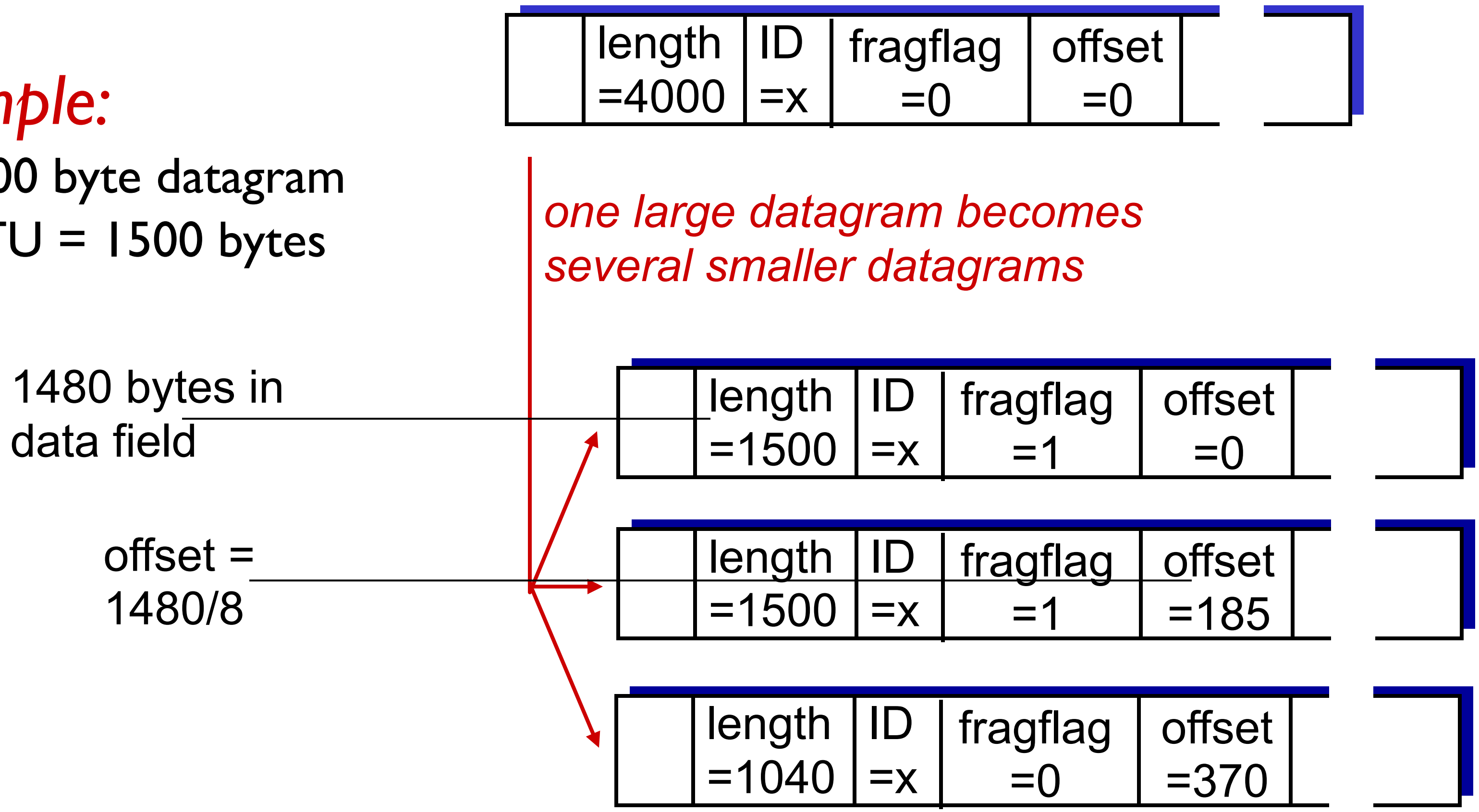
- network links have MTU (max.transfer size) - largest possible link-level frame
  - different link types, different MTUs
- large IP datagram divided ("fragmented") within net
  - one datagram becomes several datagrams
  - "reassembled" only at final destination
  - IP header bits used to identify, order related fragments



IP fragmentation, reassembly

example:

- 4000 byte datagram
- MTU = 1500 bytes



Chapter 5: outline

5.1 introduction

5.2 what's inside a router

5.3 IP: Internet Protocol
- datagram format
- DHCP
- ICMP
- IPv6

5.4 static routing

5.5 routing algorithms
- link state
- distance vector
- hierarchical routing

5.6 routing in the Internet
- RIP
- OSPF
- BGP

DHCP: Dynamic Host Configuration Protocol

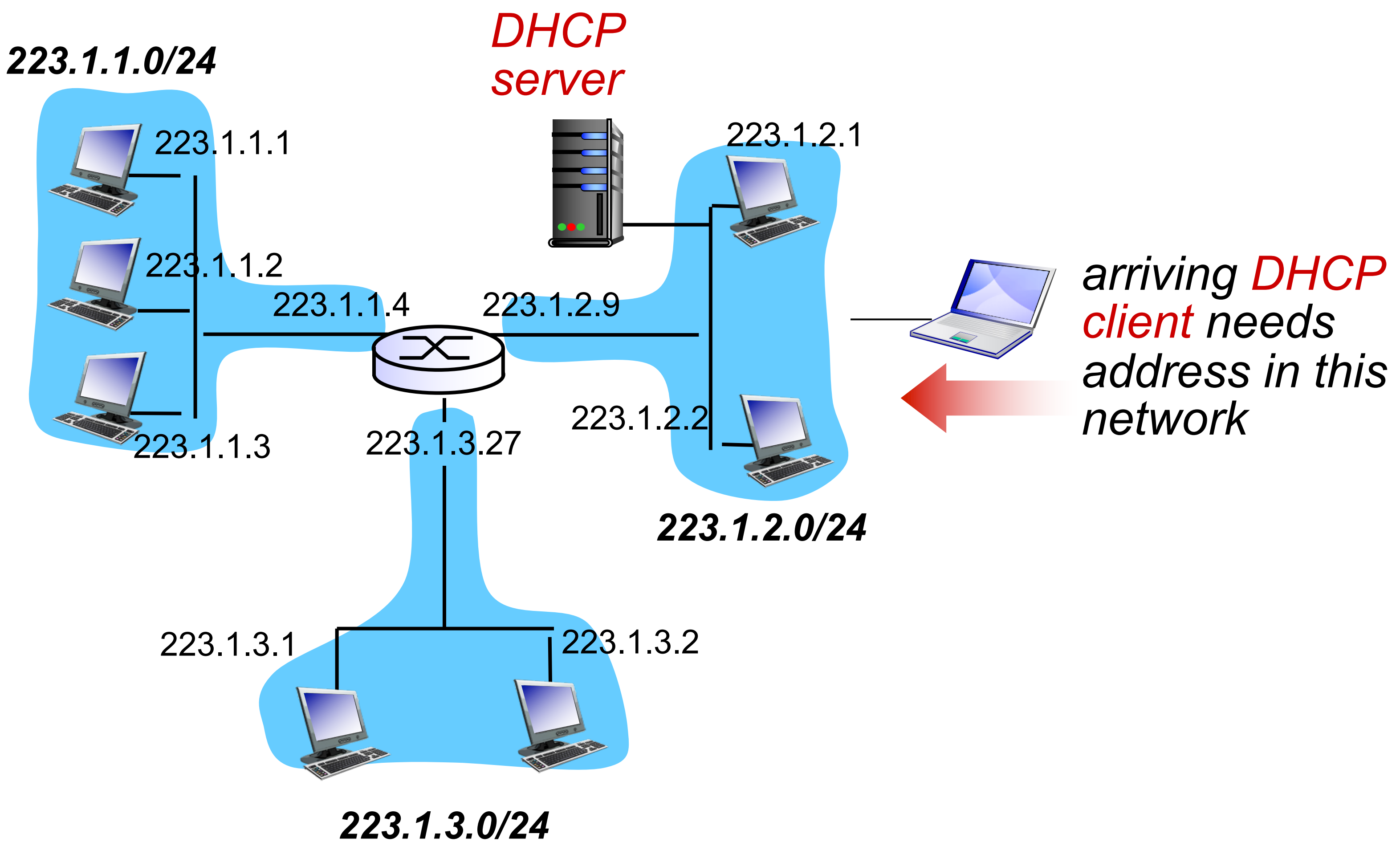
goal: allow host to *dynamically* obtain its IP address from network server when it joins network

- can renew its lease on address in use
- allows reuse of addresses (only hold address while connected/"on")
- support for mobile users who want to join network (more shortly)

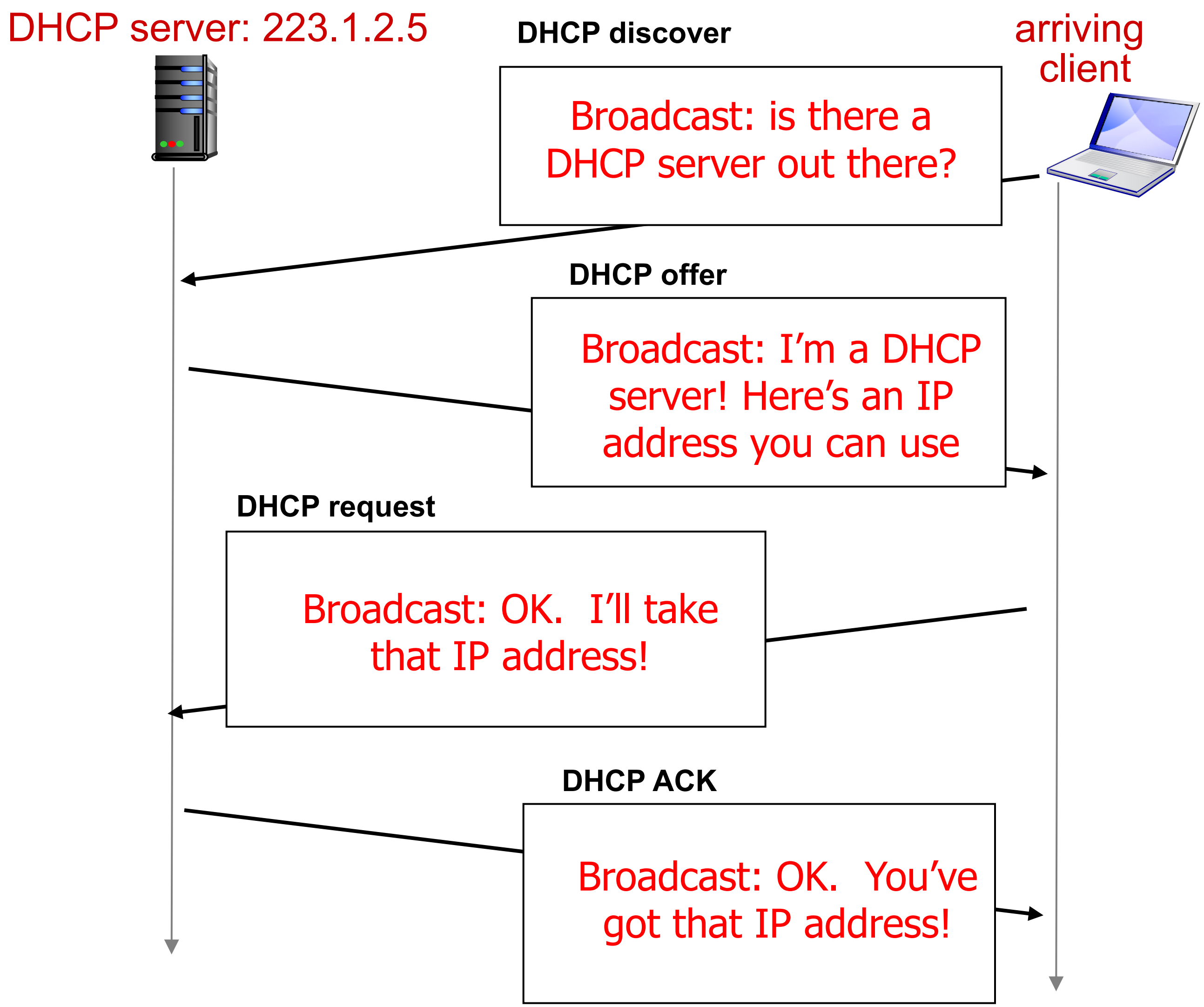
DHCP overview:

- host broadcasts "DHCP discover" msg [optional]
- DHCP server responds with "DHCP offer" msg [optional]
- host requests IP address: "DHCP request" msg
- DHCP server sends address: "DHCP ack" msg

DHCP client-server scenario



DHCP client-server scenario



DHCP: more than IP addresses

DHCP can return more than just allocated IP address on subnet:

- address of first-hop router for client
- name and IP address of DNS sever
- network mask (indicating network versus host portion of address)

Chapter 5: outline

5.1 introduction

5.2 what's inside a router

5.3 IP: Internet Protocol

- datagram format
- DHCP
- ICMP
- IPv6

5.4 static routing

5.5 routing algorithms

- link state
- distance vector
- hierarchical routing

5.6 routing in the Internet

- RIP
- OSPF
- BGP

ICMP: internet control message protocol

- used by hosts & routers to communicate network-level information
  - error reporting: unreachable host, network, port, protocol
  - echo request/reply (used by ping)
- network-layer "above" IP:
  - ICMP msgs carried in IP datagrams
- ICMP message: type, code plus first 8 bytes of IP datagram causing error

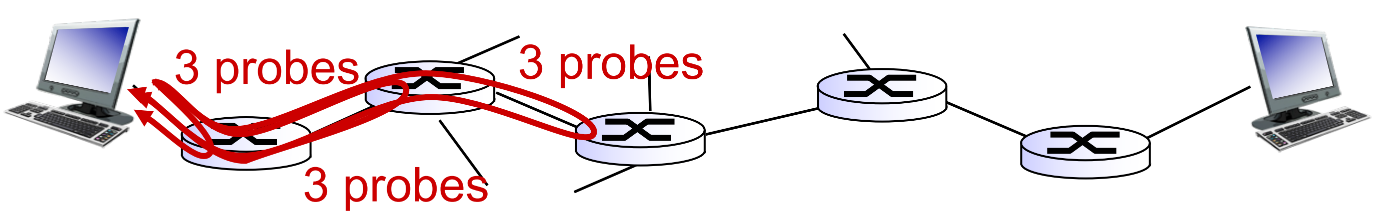
Type	Code	description
0	0	echo reply (ping)
3	0	dest. network unreachable
3	1	dest host unreachable
3	2	dest protocol unreachable
3	3	dest port unreachable
3	6	dest network unknown
3	7	dest host unknown
4	0	source quench (congestion control - not used)
8	0	echo request (ping)
9	0	route advertisement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

Traceroute and ICMP

- source sends series of UDP segments to dest
  - first set has TTL =1
  - second set has TTL=2, etc.
  - unlikely port number
- when *n*th set of datagrams arrives to nth router:
  - router discards datagrams
  - and sends source ICMP messages (type 11, code 0)
  - ICMP messages includes name of router & IP address
- when ICMP messages arrives, source records RTTs

stopping criteria:

- UDP segment eventually arrives at destination host
- destination returns ICMP "port unreachable" message (type 3, code 3)
- source stops



IPv6: motivation

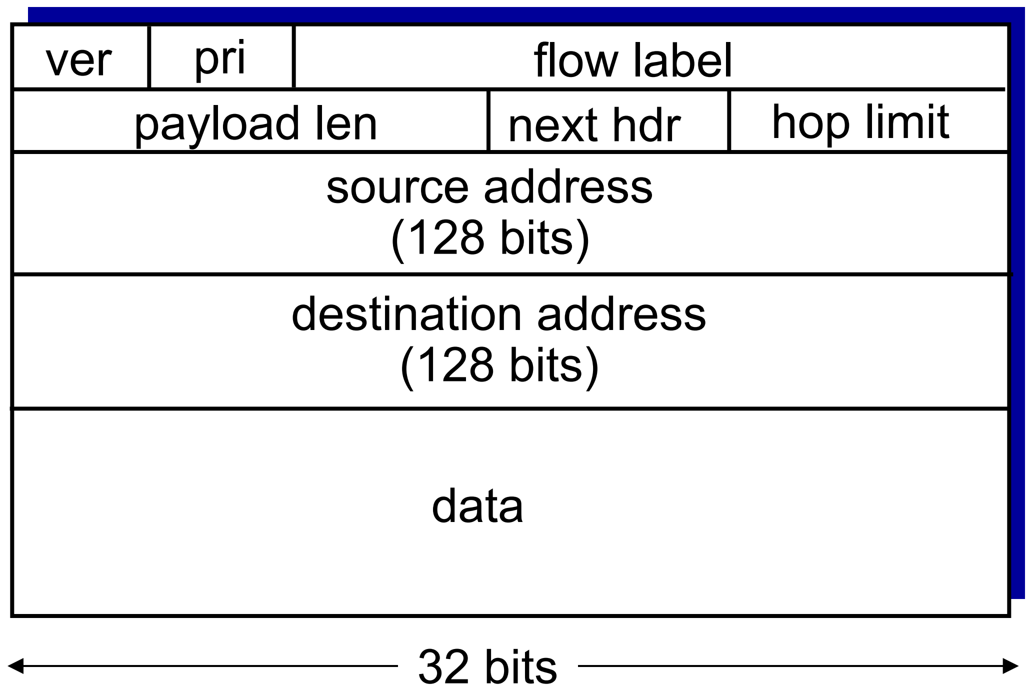
- *initial motivation:* 32-bit address space soon to be completely allocated.
- additional motivation:
  - header format helps speed processing/forwarding
  - header changes to facilitate QoS

IPv6 datagram format:

  - fixed-length 40 byte header
  - no fragmentation allowed

IPv6 datagram format

priority: identify priority among datagrams in flow
flow Label: identify datagrams in same "flow."
(concept of "flow" not well defined).
next header: identify upper layer protocol for data

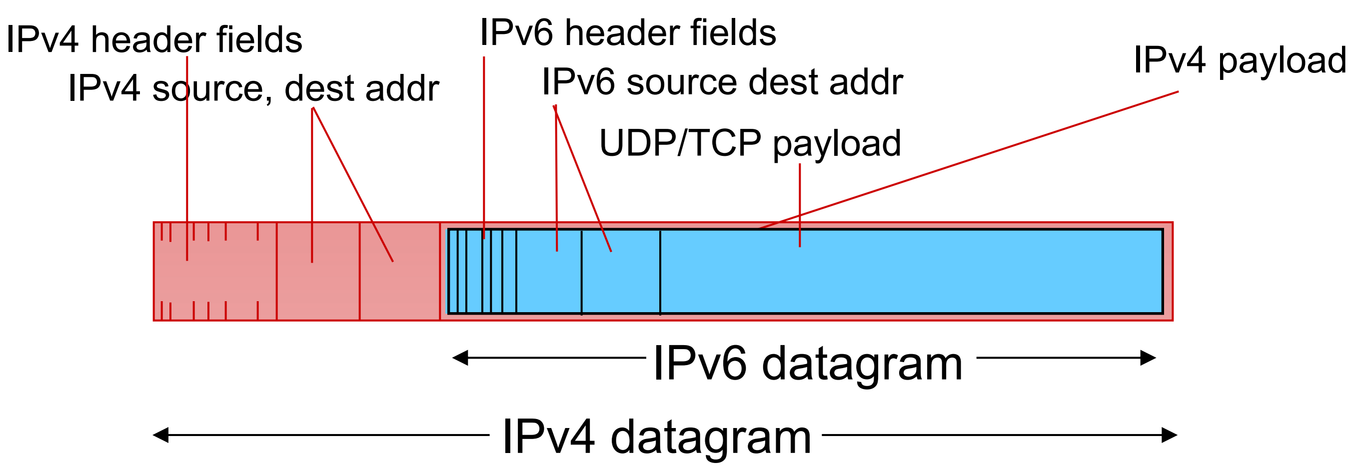


Other changes from IPv4

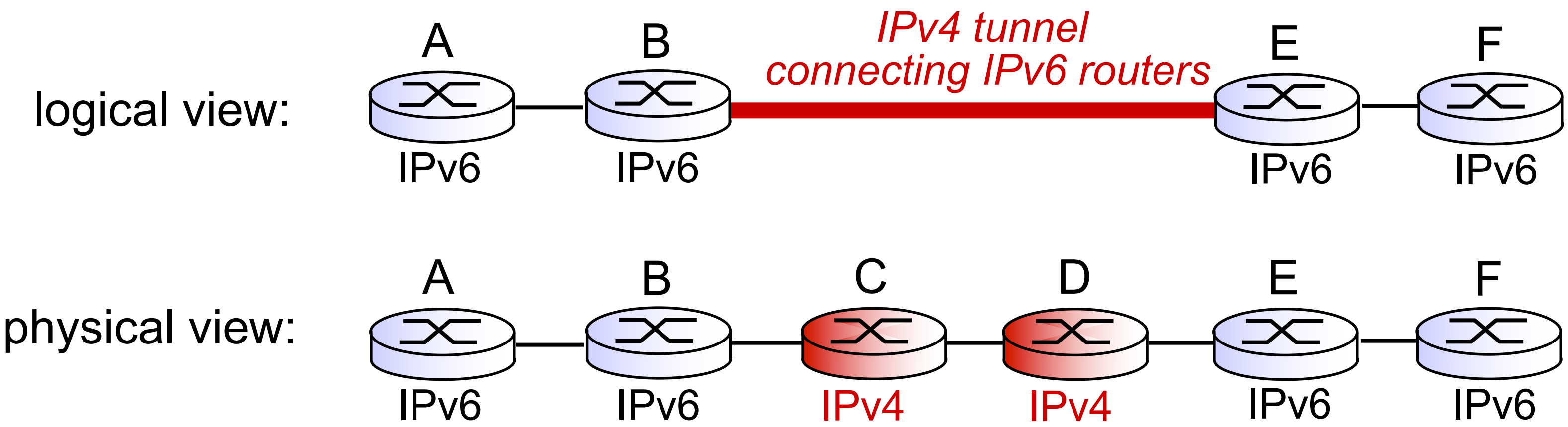
- *checksum:* removed entirely to reduce processing time at each hop
- *options:* allowed, but outside of header, indicated by "Next Header" field
- *ICMPv6:* new version of ICMP
  - additional message types, e.g. "Packet Too Big"
  - multicast group management functions

Transition from IPv4 to IPv6

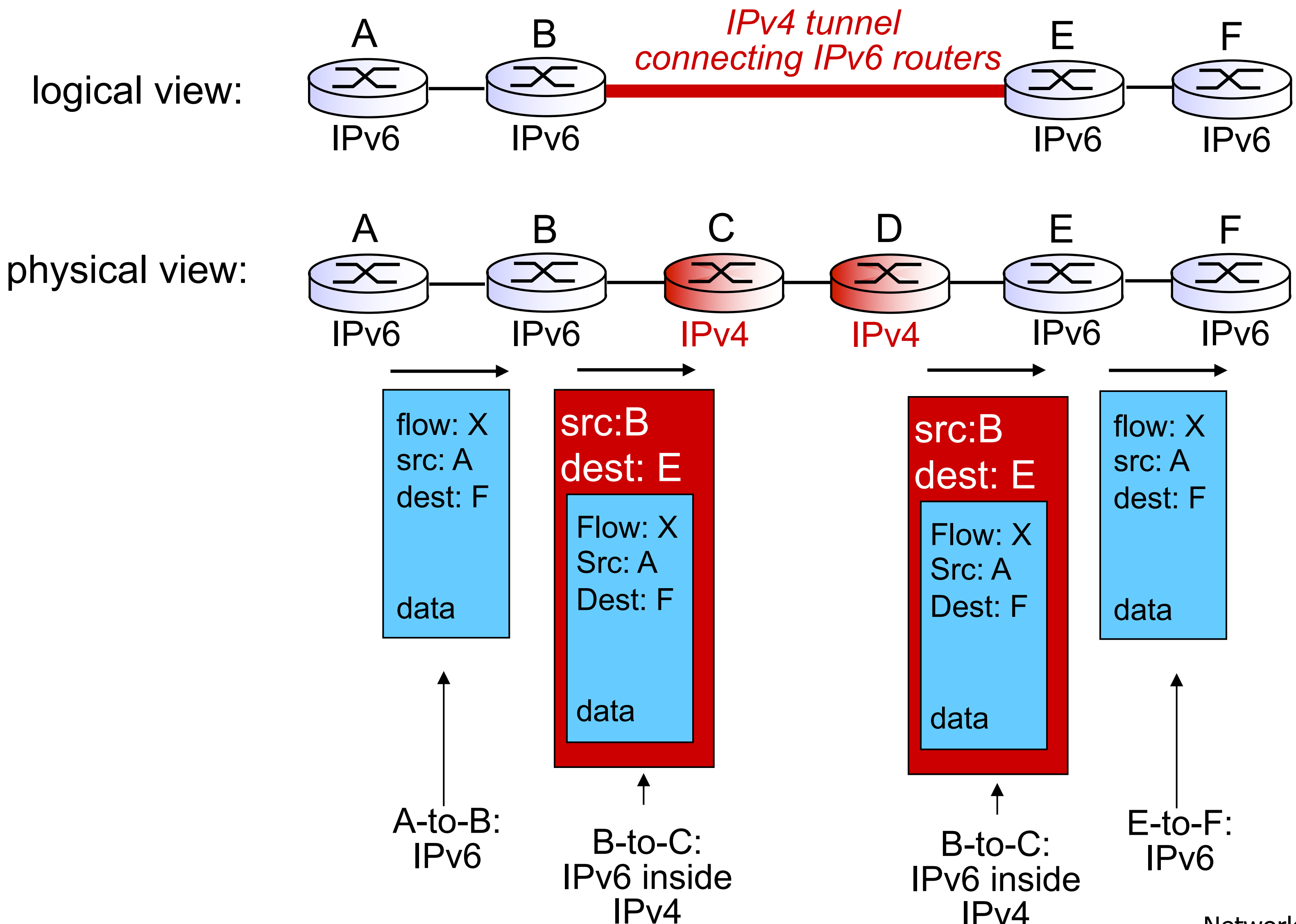
- not all routers can be upgraded simultaneously
  - no "flag days"
  - how will network operate with mixed IPv4 and IPv6 routers?
- *tunneling:* IPv6 datagram carried as *payload* in IPv4 datagram among IPv4 routers



Tunneling


logical view:
A
B
IPv4 tunnel
connecting IPv6 routers
E
F
IPv6
IPv6
IPv6
IPv6
physical view:
A
B
C
D
E
F
IPv6
IPv6
IPv4
IPv4
IPv6
IPv6

Tunneling


logical view:
A
B
IPv4 tunnel
connecting IPv6 routers
E
F
IPv6
IPv6
IPv6
IPv6
physical view:
A
B
C
D
E
F
IPv6
IPv6
IPv4
IPv4
IPv6
IPv6
flow: X
src: A
dest: F
data
src:B
dest: E
Flow: X
Src: A
Dest: F
data
src:B
dest: E
Flow: X
Src: A
Dest: F
data
flow: X
src: A
dest: F
data
A-to-B:
IPv6
B-to-C:
IPv6 inside
IPv4
B-to-C:
IPv6 inside
IPv4
E-to-F:
IPv6

IPv6: adoption

- US National Institutes of Standards estimate [2013]:
  - ~3% of industry IP routers
  - ~11% of US gov't routers

- *Long (long!) time for deployment, use*
  - 20 years and counting!
  - think of application-level changes in last 20 years: WWW, Facebook, …

Chapter 5: outline

5.1 introduction

5.2 what's inside a router

5.3 IP: Internet Protocol

- datagram format
- DHCP
- ICMP
- IPv6

5.4 static routing

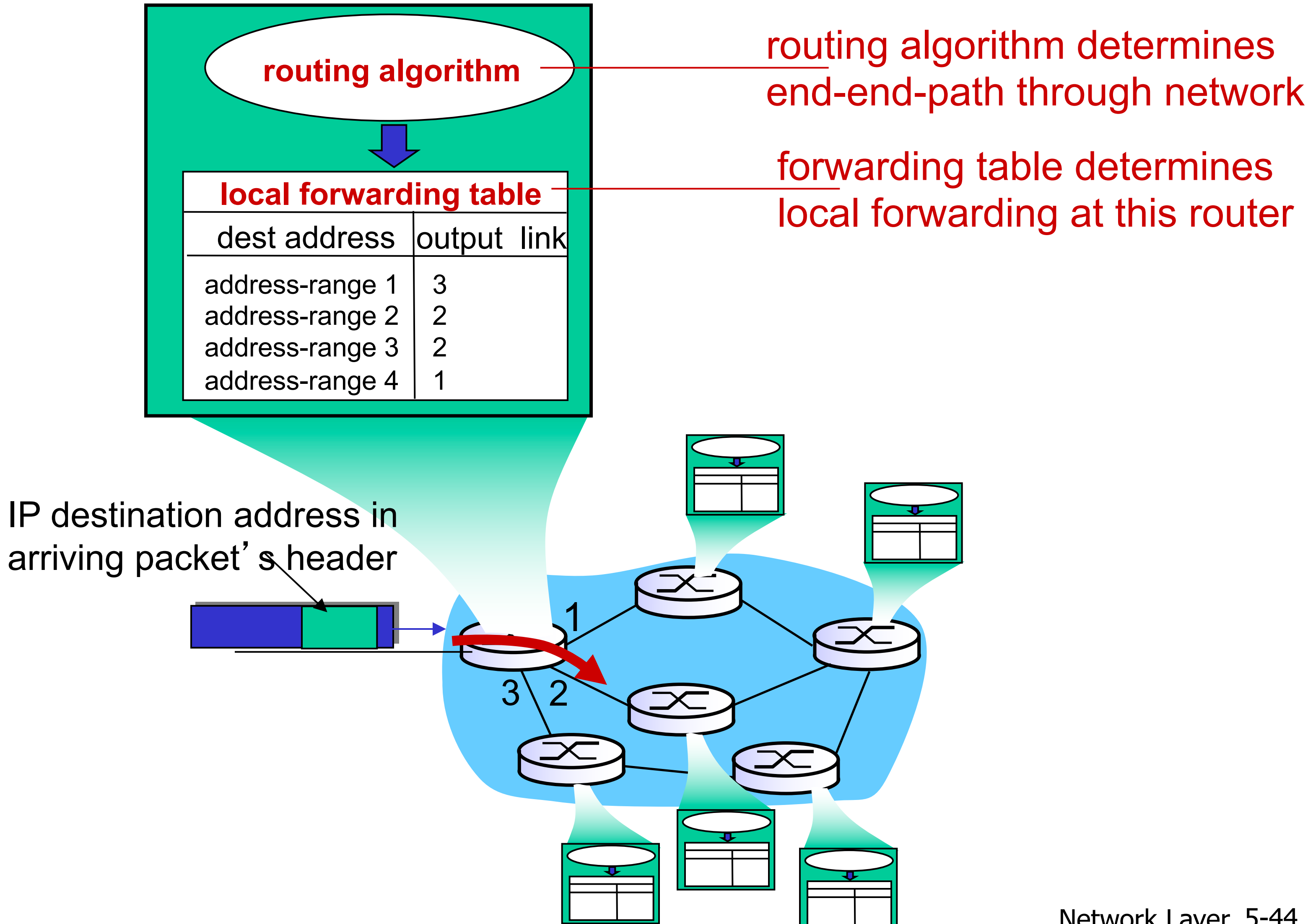
5.5 routing algorithms

- link state
- distance vector
- hierarchical routing

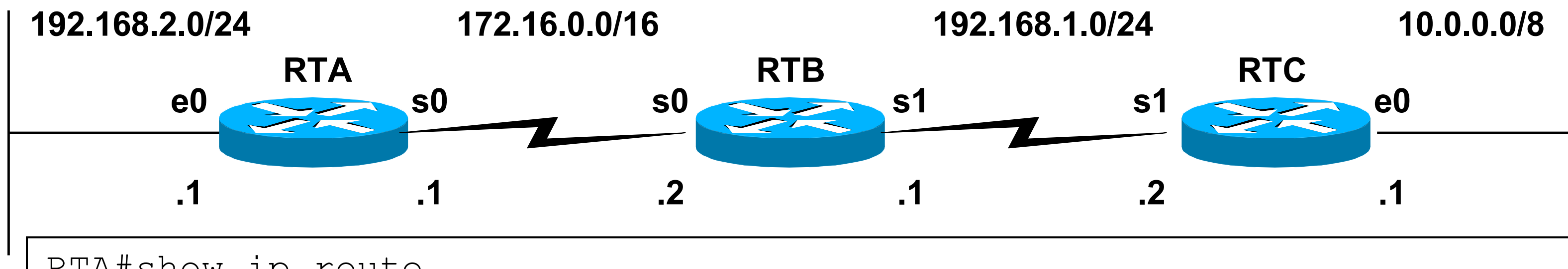
5.6 routing in the Internet

- RIP
- OSPF
- BGP

Interplay between routing, forwarding



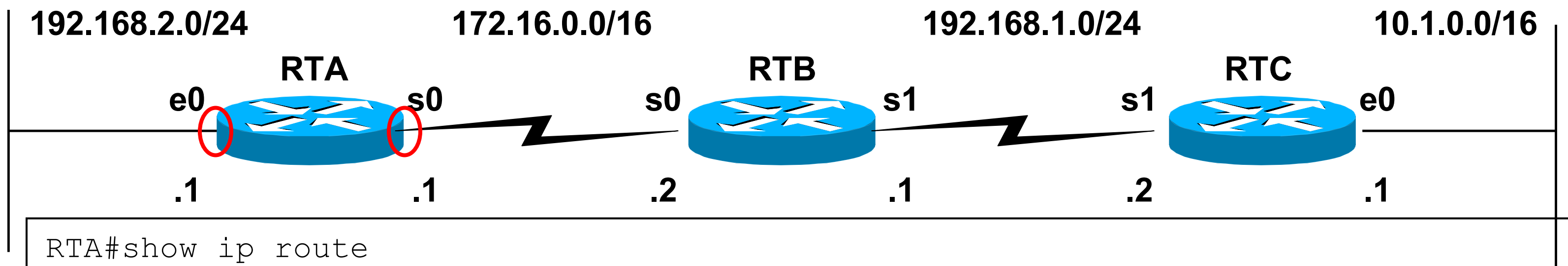
Directly Connected Networks and the IP Routing Table(*)




```
RTA#show ip route
Codes: C - connected,.. <Other codes and gateway information omitted>
C    172.16.0.0/16 is directly connected, Serial0
C    192.168.2.0/24 is directly connected, Ethernet0

RTB#show ip route
Codes: C - connected,.. <Other codes and gateway information omitted>
C    172.16.0.0/16 is directly connected, Serial0
C    192.168.1.0/24 is directly connected, Serial1

RTC#show ip route
Codes: C - connected,.. <Other codes and gateway information omitted>
C    10.0.0.0/8 is directly connected, Ethernet0
C    192.168.1.0/24 is directly connected, Serial1
```

(*) *This slide and next 05 slides is from CCNA 3.0 curriculum.*

Directly Connected Networks and the IP Routing Table



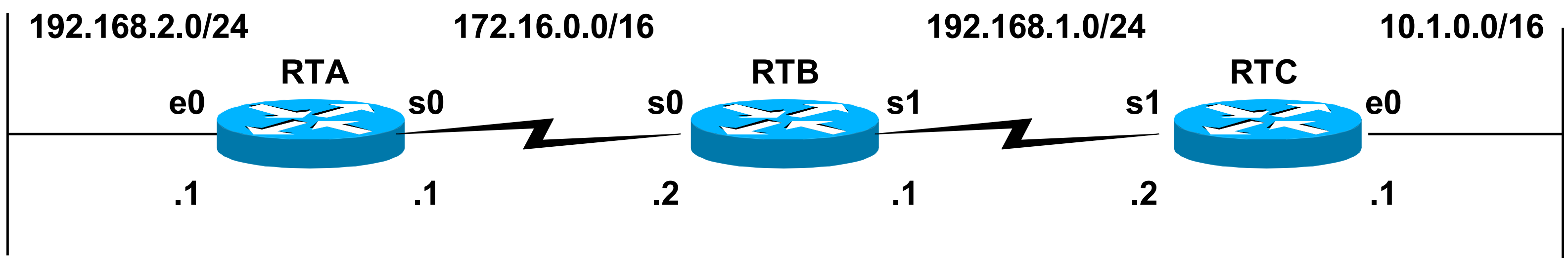

```
RTA#show ip route
C    172.16.0.0/16 is directly connected, Serial0
C    192.168.2.0/24 is directly connected, Ethernet0
RTA#ping 172.16.0.1
!!!!!

RTA#ping 172.16.0.2
!!!!!

RTA#ping 192.168.1.1
.....

RTA#ping 192.168.1.2
.....

RTA#ping 10.1.0.1
.....
```

Static Routing

Router(config)#**ip route** *destination-prefix destination-prefix-mask* {*address* | *interface*} [*distance*] [**tag** *tag*] [**permanent**]

Parameter	Description
destination-prefix	The IP network or subnetwork address for the destination
destination-prefix-mask	Subnet mask for the destination IP address
address	IP address of the next hop that can be used to reach that network
interface	Network interface to use
distance	Optional, an administrative distance
tag tag	Optional, tag value that can be used as a match value for controlling redistribution using route maps
permanent	Optional, specification that the route will not be removed, even if the interface shuts down

Static Routing



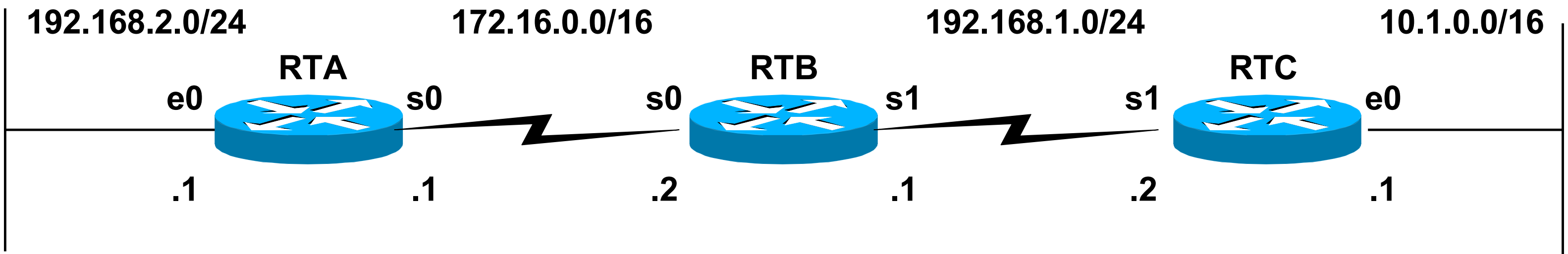

```
RTA(config)#ip route 192.168.1.0 255.255.255.0 172.16.0.2
```

Network/subnet route

Intermediate-Address (usually "next-hop")

```
RTA#show ip route
Codes: C - connected, S - static,
C    172.16.0.0/16 is directly connected, Serial0
S    192.168.1.0/24 [1/0] via 172.16.0.2
C    192.168.2.0/24 is directly connected, Ethernet0
```

Static Routing




```
RTA(config)#ip route 192.168.1.0 255.255.255.0 serial 0
```

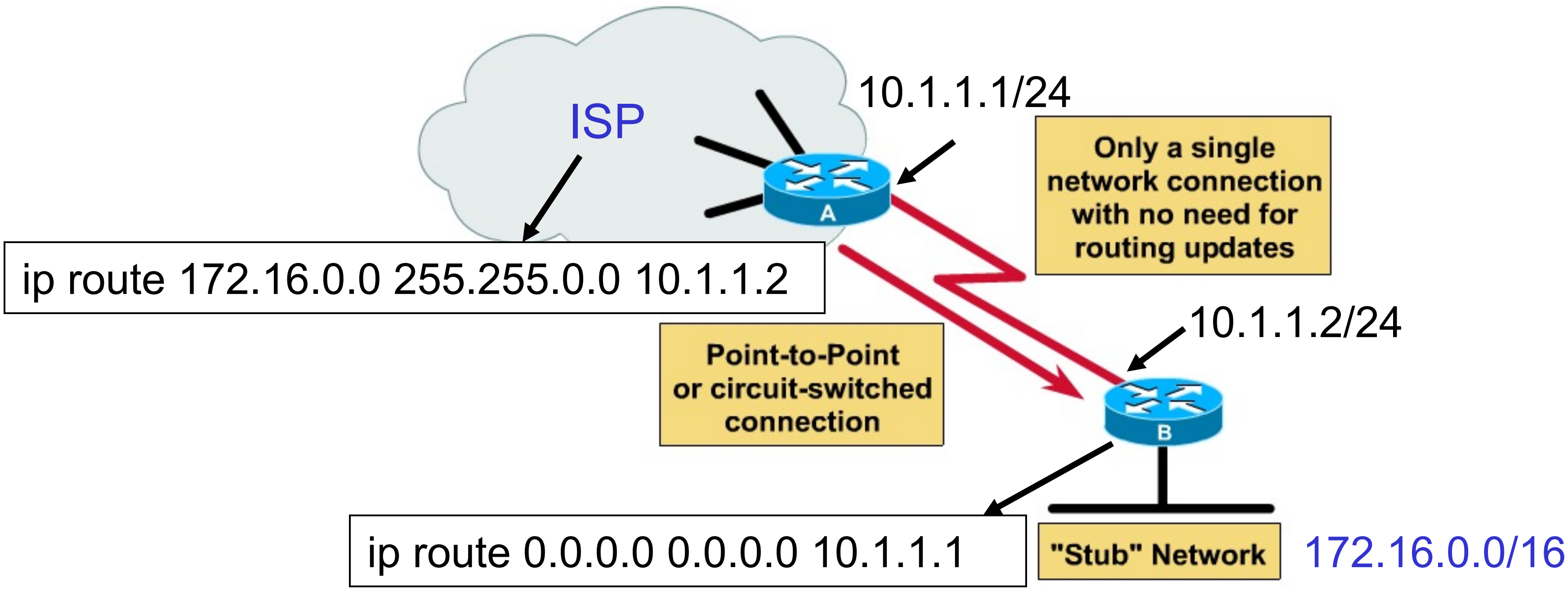
Network/subnet route

Outgoing interface

```
RTA#show ip route
Codes: C - connected, S - static,
C     172.16.0.0/16 is directly connected, Serial0
S     192.168.1.0/24 is directly connected, Serial0
C     192.168.2.0/24 is directly connected, Ethernet0
```

Common uses for Static Routes

Default Static Routing Example




```
RTB#show ip route
Gateway of last resort is 10.1.1.1 to network 0.0.0.0
C    172.16.0.0/16 is directly connected, Ethernet0
     10.0.0.0/24 is subnetted, 1 subnets
C       10.1.1.0 is directly connected, Serial1
S*   0.0.0.0/0 [1/0] via 10.1.1.1
```

- Any packets not matching the routes 172.16.0.0/16 or 10.1.1.0/24 are sent to the router 10.1.1.1 – where it is now their "problem."

Chapter 5: outline

5.1 introduction

5.2 what's inside a router

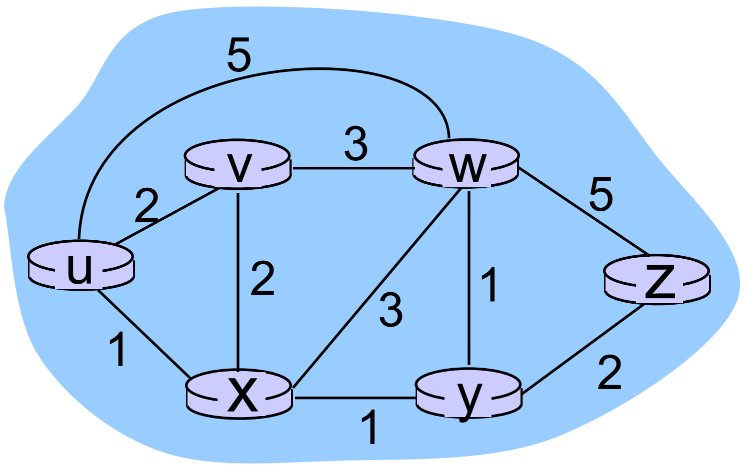
5.3 IP: Internet Protocol
- datagram format
- DHCP
- ICMP
- IPv6

5.4 static routing

5.5 routing algorithms
- link state
- distance vector
- hierarchical routing

5.6 routing in the Internet
- RIP
- OSPF
- BGP

Graph abstraction



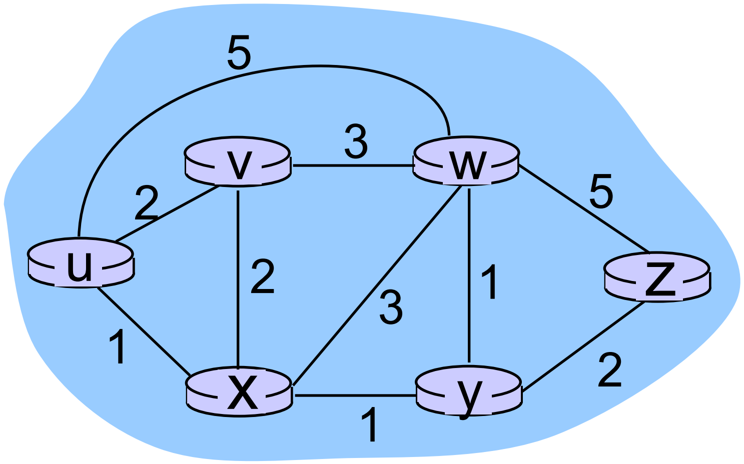
graph: G = (N,E)

N = set of routers = { u, v, w, x, y, z }

E = set of links ={ (u,v), (u,x), (v,x), (v,w), (x,w), (x,y), (w,y), (w,z), (y,z) }

aside: graph abstraction is useful in other network contexts, e.g., P2P, where *N* is set of peers and *E* is set of TCP connections

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 related to congestion

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

key question: what is the least-cost path between u and z ?
routing algorithm: algorithm that finds that least cost path

Routing algorithm classification

Q: 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

Q: static or dynamic?

static:

- routes change slowly over time

dynamic:

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

Chapter 5: outline

5.1 introduction

5.2 what's inside a router

5.3 IP: Internet Protocol

- datagram format
- DHCP
- ICMP
- IPv6

5.4 static routing

5.5 routing algorithms

- link state
- distance vector
- hierarchical routing

5.6 routing in the Internet

- RIP
- OSPF
- BGP

A Link-State Routing Algorithm

Dijkstra's algorithm

- net topology, link costs known to all nodes
  - accomplished via "link state broadcast"
  - all nodes have same info
- computes least cost paths from one node ('source") to all other nodes
  - gives *forwarding table* for that node
- iterative: after k iterations, know least cost path to k dest.'s

notation:

- c(x,y): link cost from node x to y; = ∞ if not direct neighbors
- D(v): current value of cost of path from source to dest. v
- p(v): predecessor node along path from source to v
- N': set of nodes whose least cost path definitively known

Dijsktra's Algorithm

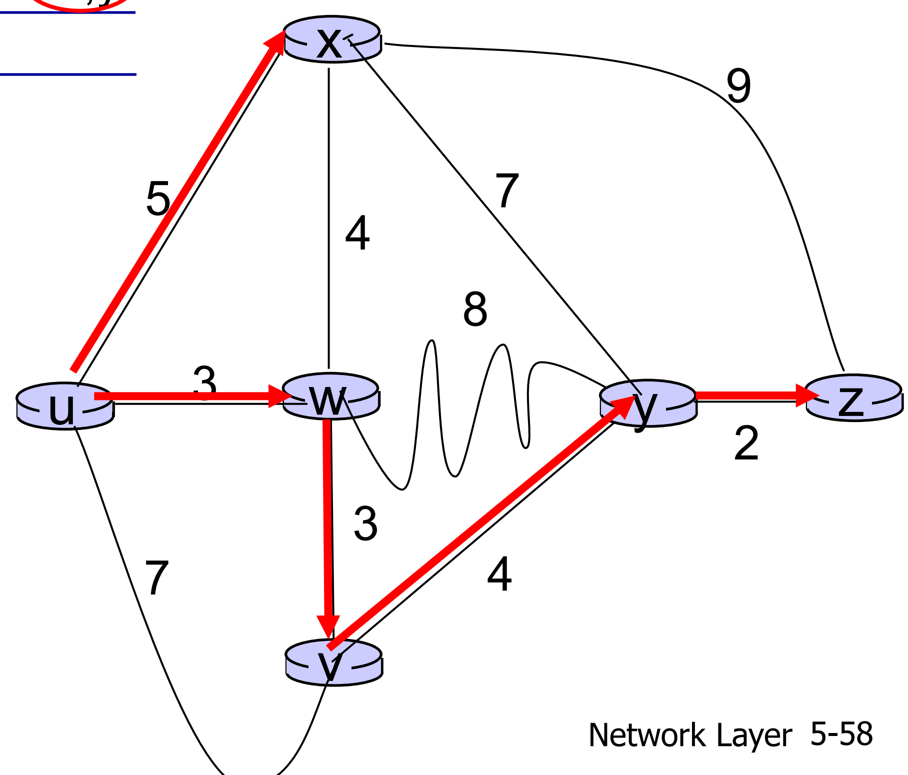
```
Initialization:
  N' = {u}
  for all nodes v
    if v adjacent to u
        then D(v) = c(u,v)
    else D(v) = ∞

Loop
   find w not in N' such that D(w) is a minimum
   add w to N'
   update D(v) for all v adjacent to w and not in N' :
       D(v) = min( D(v), D(w) + c(w,v) )
   /* new cost to v is either old cost to v or known
    shortest path cost to w plus cost from w to v */
until all nodes in N'
```

Dijkstra's algorithm: example

Step	N'	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u	7,u	3,u	5,u	∞	∞
1	uw	6,w		5,u	11,w	∞
2	uwx	6,w			11,w	14,x
3	uwxv				10,v	14,x
4	uwxvy					12,y
5	uwxvyz					

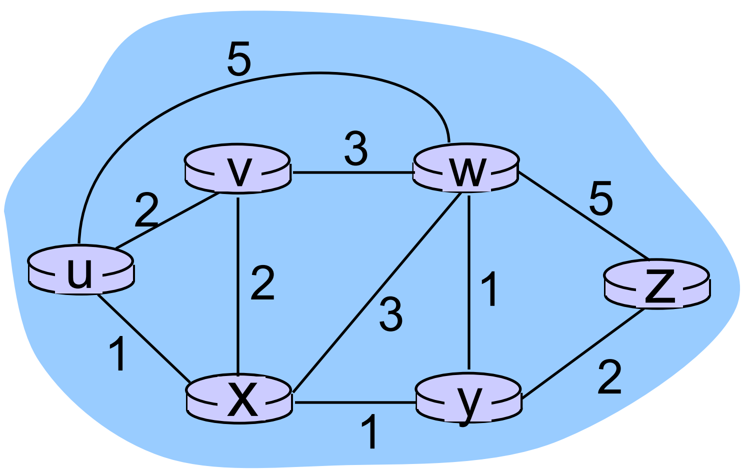
notes:

- construct shortest path tree by tracing predecessor nodes
- ties can exist (can be broken arbitrarily)



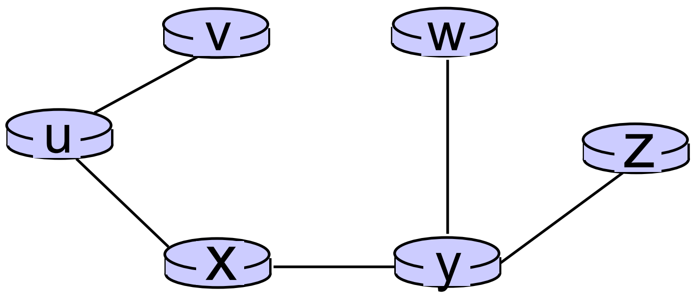
Dijkstra's algorithm: another example

Step	N'	D(v),p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
0	u	2,u	5,u	1,u	∞	∞
1	ux	2,u	4,x		2,x	∞
2	uxy	2,u	3,y			4,y
3	uxyv		3,y			4,y
4	uxyvw					4,y
5	uxyvwz					



Dijkstra's algorithm: example (2)

resulting shortest-path tree from u:



resulting forwarding table in u:

destination	link
v	(u,v)
x	(u,x)
y	(u,x)
w	(u,x)
z	(u,x)

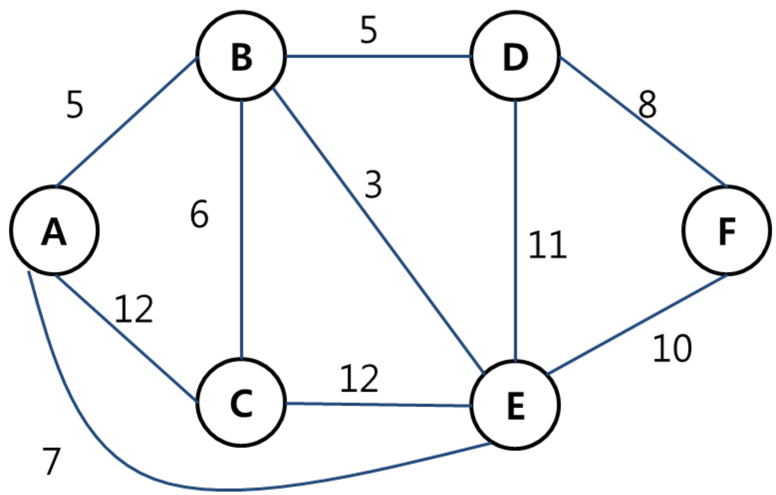
Dijkstra's algorithm, discussion

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)

Example

1) Find the shortest paths from the source node A to the other nodes using Dijkstra's algorithm in the following network topology? (Show all steps towards your solution.)



2) Draw the shortest-path tree?

Chapter 5: outline

5.1 introduction

5.2 what's inside a router

5.3 IP: Internet Protocol

- datagram format
- DHCP
- ICMP
- IPv6

5.4 static routing

5.5 routing algorithms

- link state
- distance vector
- hierarchical routing

5.6 routing in the Internet

- RIP
- OSPF
- BGP

Distance vector algorithm

Bellman-Ford equation (dynamic programming)

let

$d_x(y)$:= cost of least-cost path from x to y

then

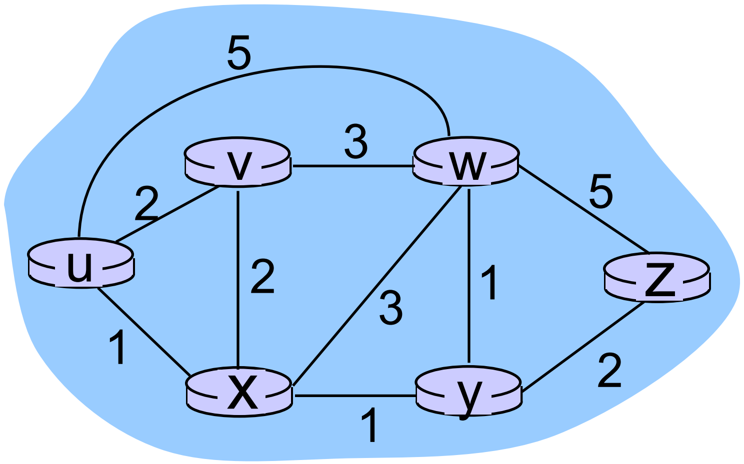
$$d_x(y) = \min_v \{c(x,v) + d_v(y)\}$$

cost from neighbor v to destination y

cost to neighbor v of x

min taken over all neighbors v of x

Bellman-Ford example



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

B-F equation says:

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

node achieving minimum is next
hop in shortest path, used in forwarding table

Distance vector algorithm

- $D_x(y)$ = estimate of least cost from x to y
  - x maintains distance vector $\mathbf{D}_x = [D_x(y): y \in N]$
- node x:
  - knows cost to each neighbor v: $c(x,v)$
  - maintains its neighbors' distance vectors. For each neighbor v, x maintains $\mathbf{D}_v = [D_v(y): y \in N]$

Distance vector algorithm

key idea:

- from time-to-time, each node sends its own distance vector estimate to neighbors
- when x receives new DV estimate from neighbor, 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, the estimate $D_x(y)$ *converge to the actual least cost* $d_x(y)$

Distance vector algorithm

iterative, asynchronous:
each local iteration caused by:

- local link cost change
- DV update message from neighbor

distributed:

- each node notifies neighbors *only* when its DV changes
  - neighbors then notify their neighbors if necessary

each node:

wait for (change in local link cost or msg from neighbor)

recompute estimates

if DV to any dest has changed, *notify* neighbors

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

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

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

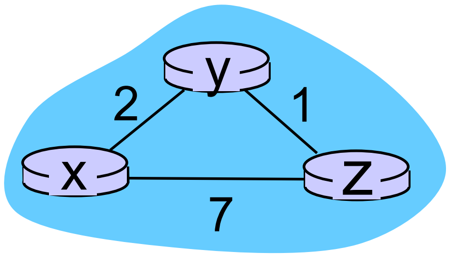
node y table

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

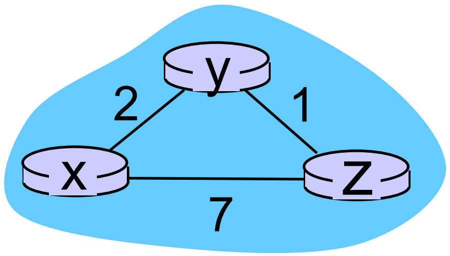
node z table

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

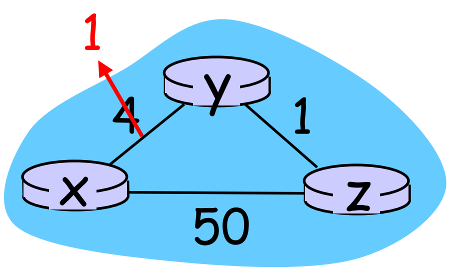
time



Distance vector: link cost changes

link cost changes:

- node detects local link cost change
- updates routing info, recalculates distance vector
- if DV changes, notify neighbors
- e.g., *focus on the y's and z's entries to destination x:*



"good news travels fast"

t_0 : *y detects link-cost change, updates its DV, informs its neighbors.*

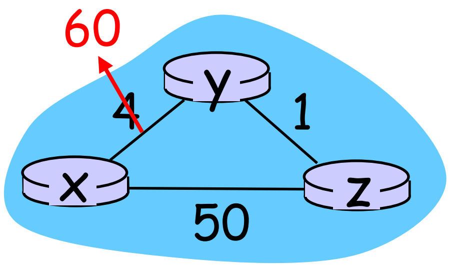
t_1 : *z receives update from y, updates its table, computes new least cost to x , sends its neighbors its DV.*

t_2 : *y receives z's update, updates its distance table. y's least costs do not change, so y does not send a message to z.*

Distance vector: link cost changes

link cost changes:

- node detects local link cost change
- 44 iterations before algorithm stabilizes: see text
- *bad news travels slow* - "count to infinity" problem!



poisoned reverse:

- If Z gets to X via Y :
  - Z tells Y its (Z's) distance to X is infinite (so Y won't route to X via Z)
- will this completely solve count to infinity problem?

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

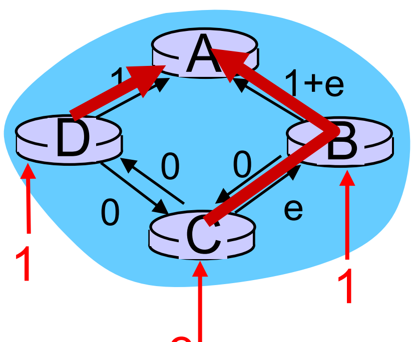
DV:

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

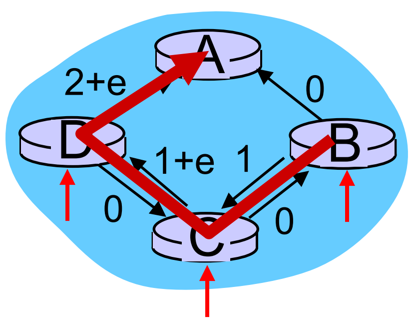
Oscillations with delay-based/congestion link metric

oscillations possible:

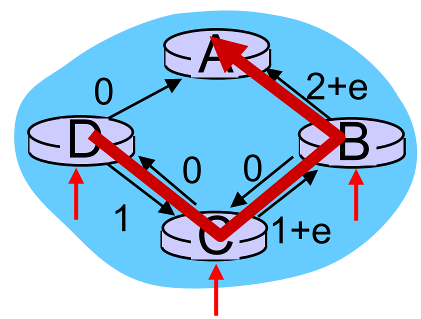
- e.g., support link cost equals amount of carried traffic:
  (for any routing protocols with delay-based/congestion link metric)



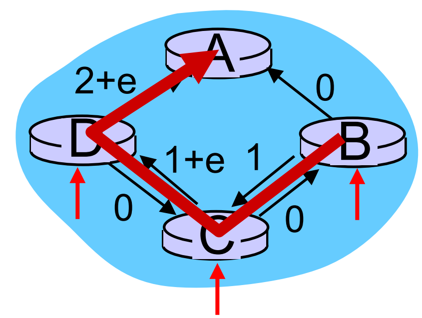
initially



given these costs, find new routing…. resulting in new costs



given these costs, find new routing…. resulting in new costs



given these costs, find new routing…. resulting in new costs

Chapter 5: outline

5.1 introduction

5.2 what's inside a router

5.3 IP: Internet Protocol

- datagram format
- DHCP
- ICMP
- IPv6

5.4 static routing

5.5 routing algorithms

- link state
- distance vector
- hierarchical routing

5.6 routing in the Internet

- RIP
- OSPF
- BGP

Hierarchical routing

our routing study thus far - idealization

- all routers identical
- network "flat"

… *not* true in practice

scale: with 600 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

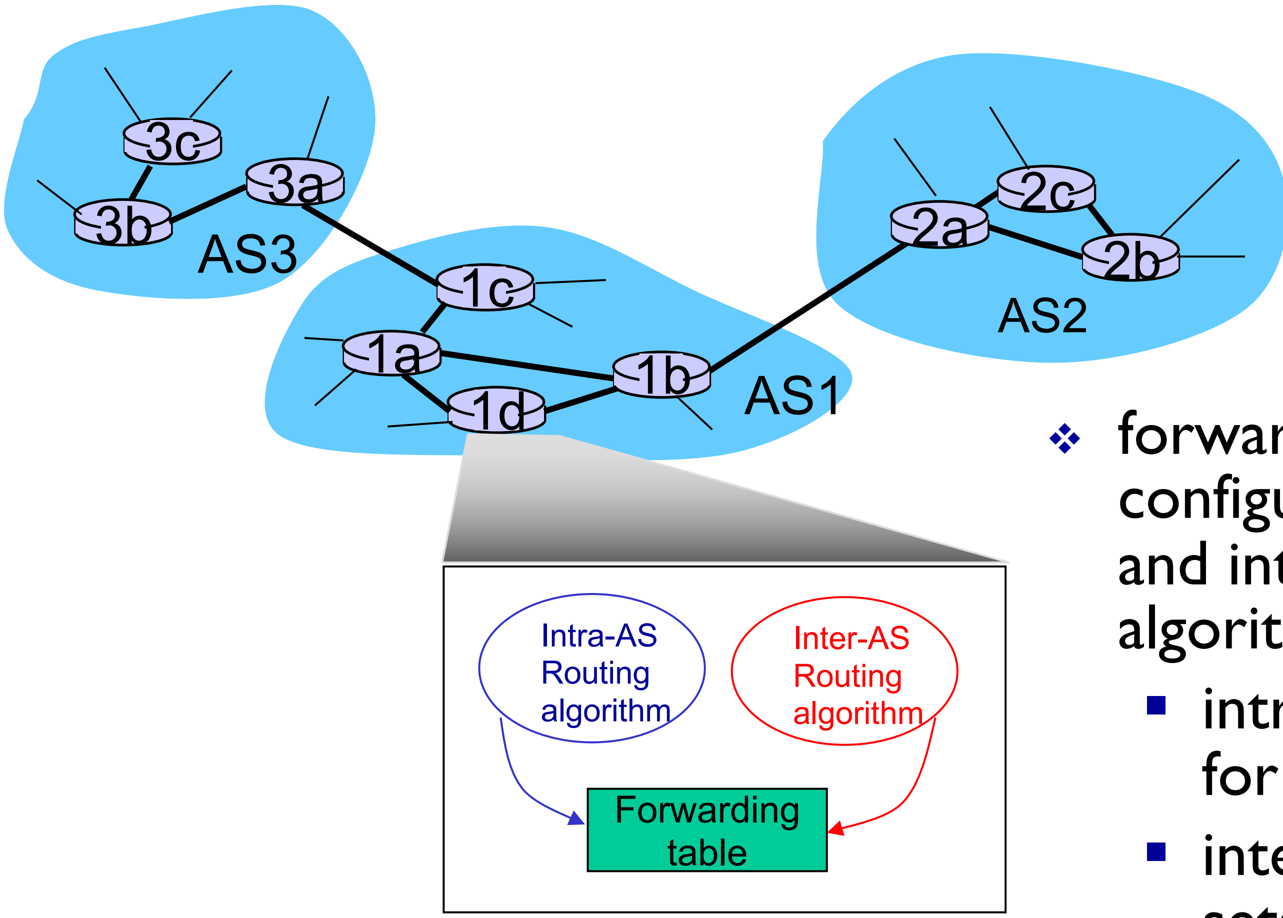
Hierarchical routing

- aggregate routers into regions, "autonomous systems" (AS)
- routers in same AS run same routing protocol
  - "intra-AS" routing protocol
  - routers in different AS can run different intra-AS routing protocol

gateway router:

- at "edge" of its own AS
- has link to router in another AS

Interconnected ASes



- forwarding table configured by both intra- and inter-AS routing algorithm
  - intra-AS sets entries for internal dests
  - inter-AS & intra-AS sets entries for external dests

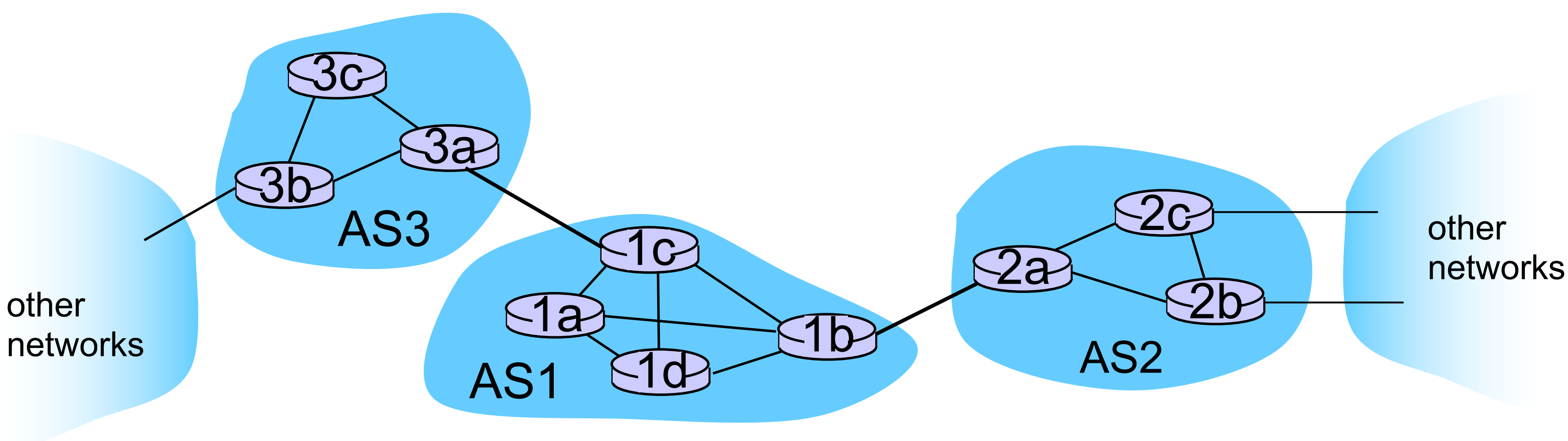
Inter-AS tasks

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

AS1 must:

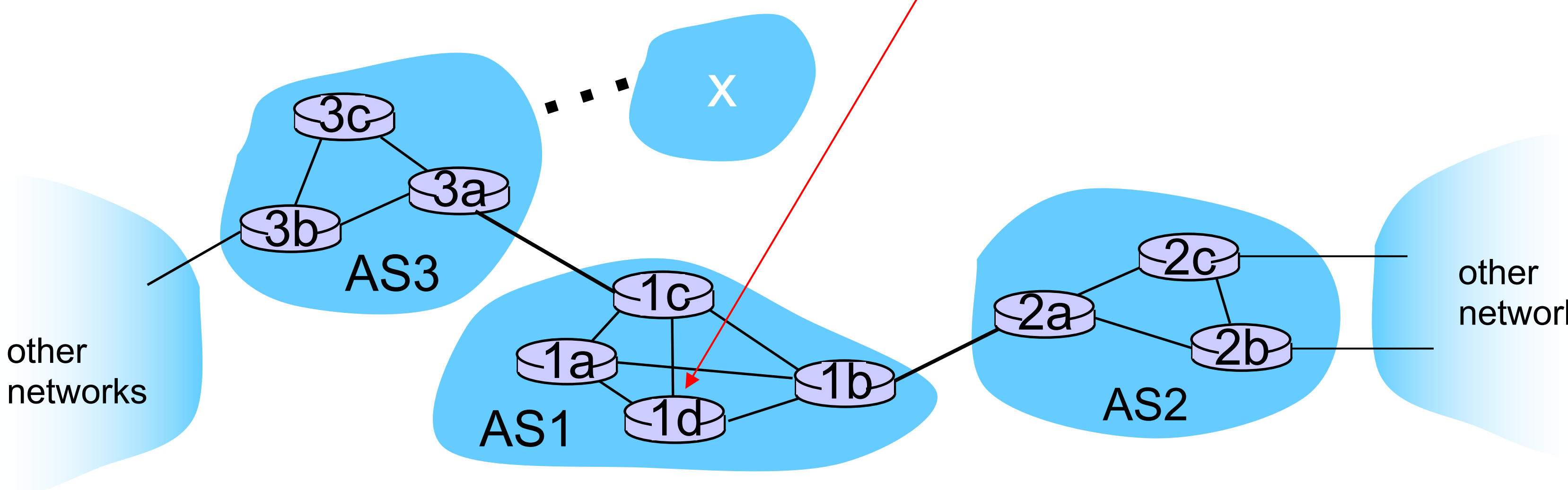
1. learn which dests are reachable through AS2, which through AS3
2. propagate this reachability info to all routers in AS1

job of inter-AS routing!



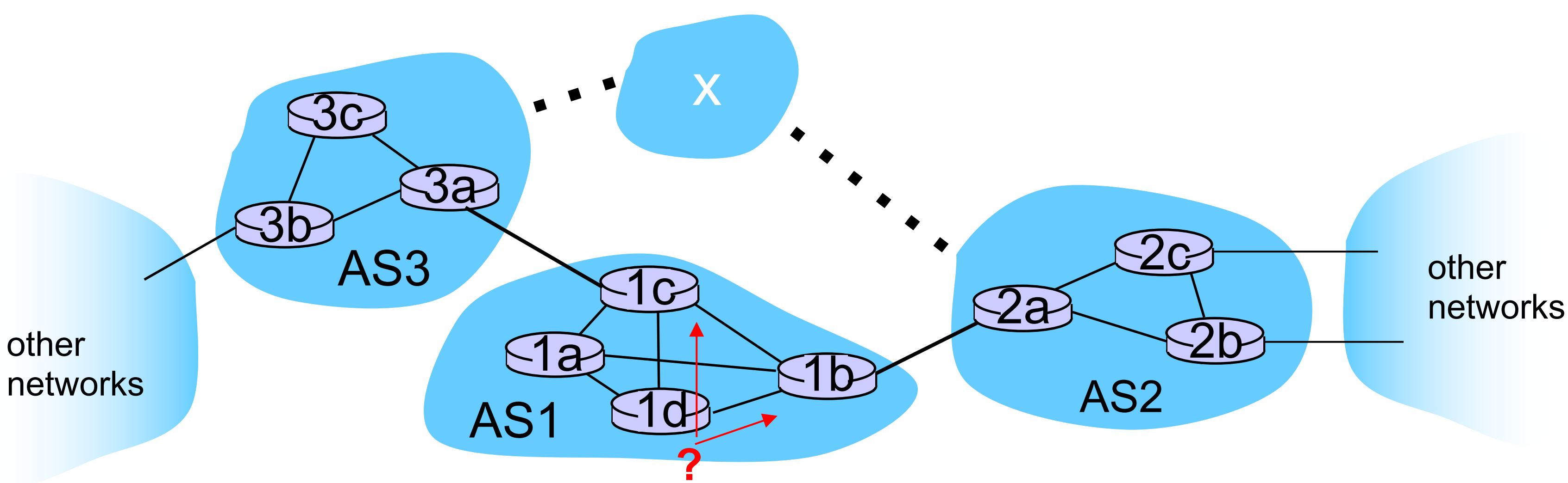
Example: setting forwarding table in router 1d

- suppose AS1 learns (via inter-AS protocol) that subnet *x* 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* is on the least cost path to 1c
  - installs forwarding table entry *(x,I)*



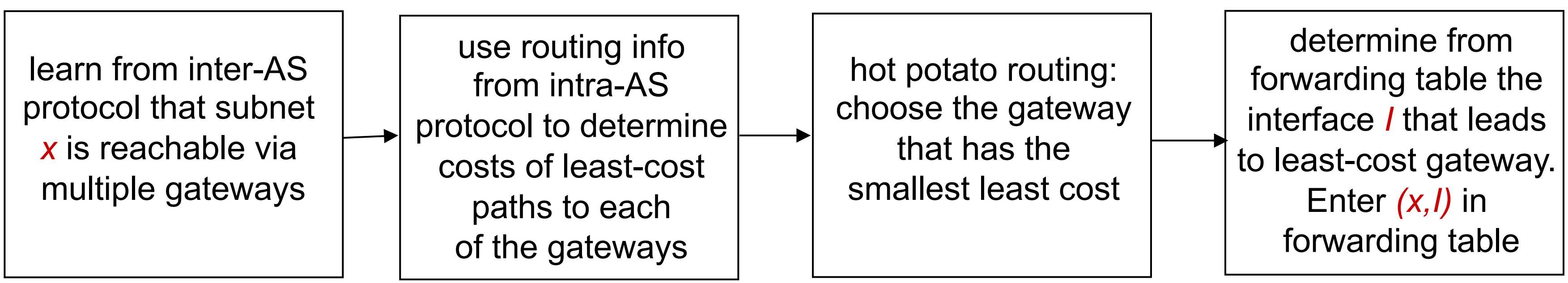
Example: choosing among multiple ASes

- 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 which gateway it should forward packets towards for dest x
  - this is also job of inter-AS routing protocol!



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 dest x
  - this is also job of inter-AS routing protocol!
- *hot potato routing: send* packet towards closest of two routers.



Chapter 5: outline

5.1 introduction

5.2 what's inside a router

5.3 IP: Internet Protocol

- datagram format
- DHCP
- ICMP
- IPv6

5.4 static routing

5.5 routing algorithms

- link state
- distance vector
- hierarchical routing

5.6 routing in the Internet

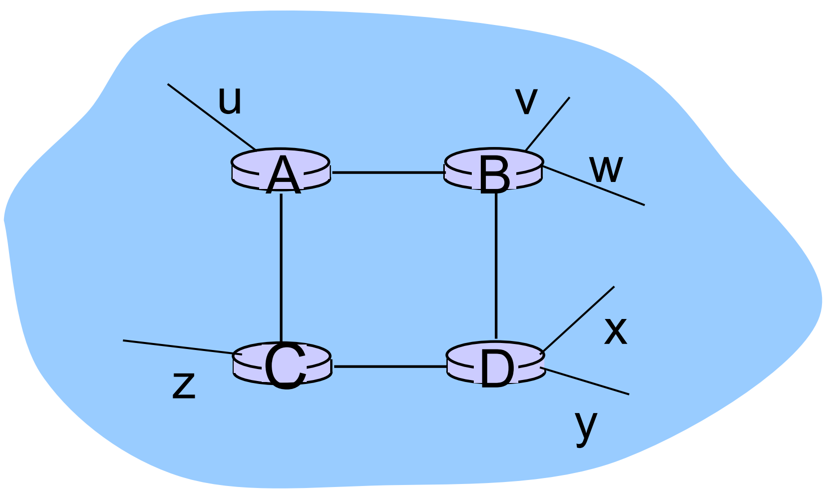
- RIP
- OSPF
- BGP

Intra-AS Routing

- also known as *interior gateway protocols (IGP)*
- most common intra-AS routing protocols:
  - RIP: Routing Information Protocol
  - OSPF: Open Shortest Path First
  - IGRP: Interior Gateway Routing Protocol (Cisco proprietary)

RIP (Routing Information Protocol)

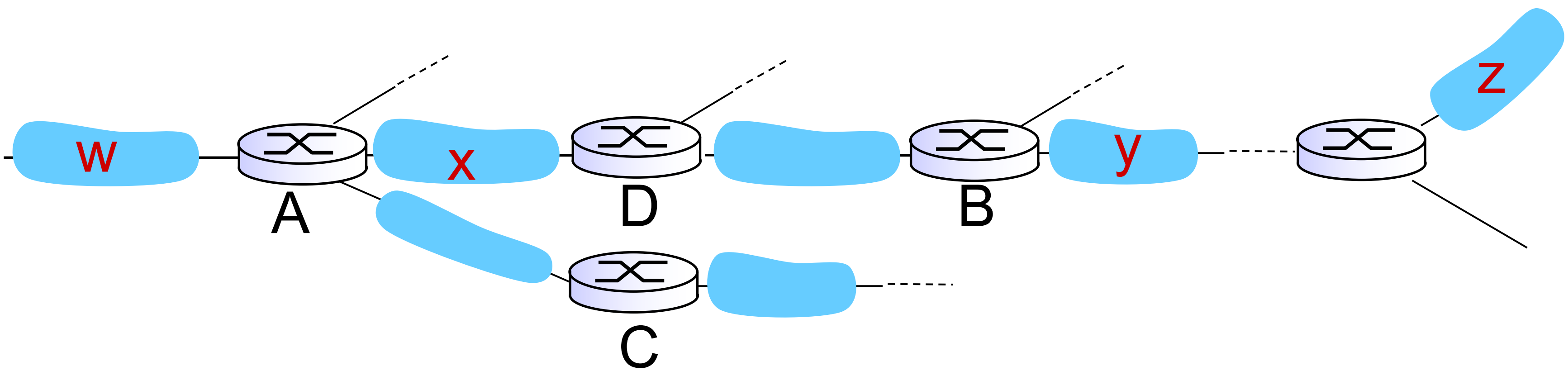
- included in BSD-UNIX distribution in 1982
- distance vector algorithm
  - distance metric: # hops (max = 15 hops), each link has cost 1
  - DVs exchanged with neighbors every 30 sec in response message (aka advertisement)
  - each advertisement: list of up to 25 destination *subnets* *(in IP addressing sense)*



from router A to destination *subnets:*

subnet	hops
u	1
v	2
w	2
x	3
y	3
z	2

RIP: example



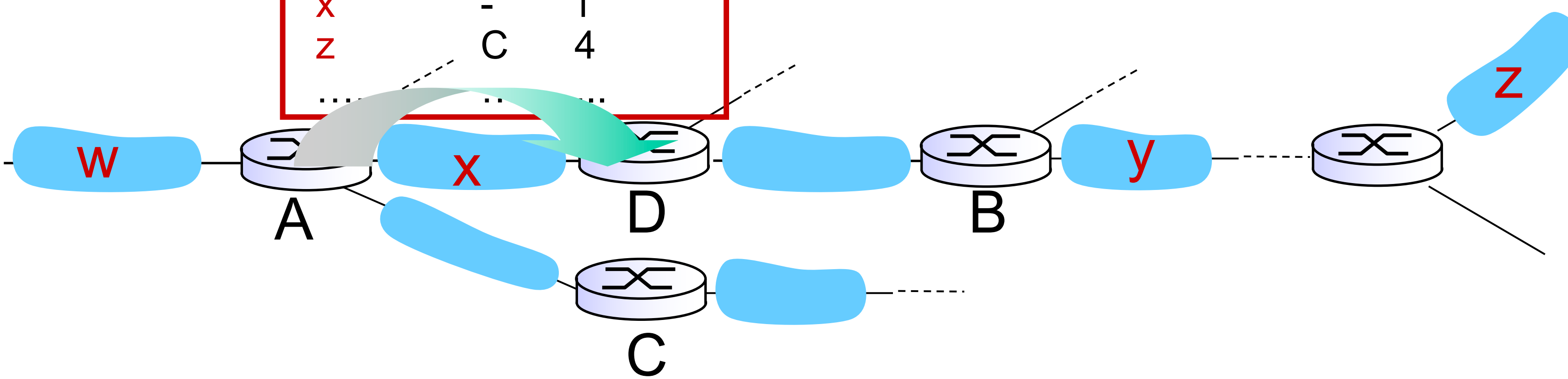
routing table in router D

destination subnet	next router	# hops to dest
w	A	2
y	B	2
z	B	7
x	--	1
….	….	….

RIP: example

A-to-D advertisement

dest	next	hops
w	-	1
x	-	1
z	C	4
….	…	…



routing table in router D

destination subnet	next router	# hops to dest
w	A	2
y	B	2
z	~~B~~ A	~~7~~ 5
x	--	1
….	….	….

RIP: link failure, recovery

if no advertisement heard after 180 sec --> neighbor/link declared dead

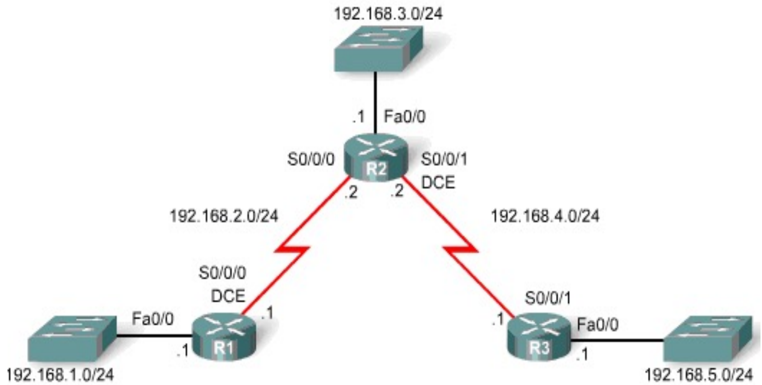
- routes via neighbor invalidated
- new advertisements sent to neighbors
- neighbors in turn send out new advertisements (if tables changed)
- link failure info quickly (?) propagates to entire net
- *poison reverse* used to prevent ping-pong loops (infinite distance = 16 hops)

Basic RIP Configuration on Cisco's router(*)

- Specifying Networks:

  Use the ***network*** command to:

  - Enable RIP on all interfaces that belong to this network
  - Advertise this network in RIP updates sent to other routers every 30 seconds




```
R1(config)#router rip
R1(config-router)#network 192.168.1.0
R1(config-router)#network 192.168.2.0
```

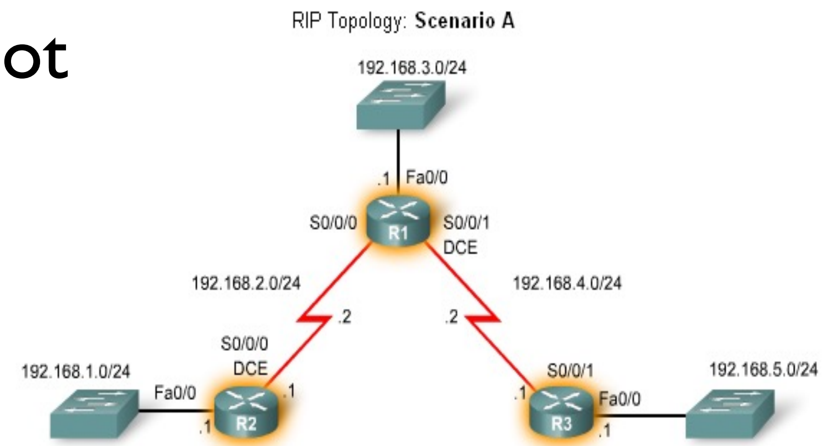
```
R2(config)#router rip
R2(config-router)#network 192.168.2.0
R2(config-router)#network 192.168.3.0
R2(config-router)#network 192.168.4.0
```

```
R3(config)#router rip
R3(config-router)#network 192.168.4.0
R3(config-router)#network 192.168.5.0
```

(*)*The following 4 slides are from Cisco's CCNA 3.1.*

Verification and Troubleshooting

- To verify and troubleshoot routing
  - -show ip route
  - -show ip protocols
  - -debug ip rip



R 192.168.5.0/24 [120/2] via 192.168.2.2, 00:00:23, Serial 0/0/0

Interpreting a RIP Route in the Routing Table

R	Identifies the source of the route as RIP.
192.168.5.0	Indicates the address of the remote network.
/24	The subnet mask used for this network
[120/2]	The administrative distance (120) and the metric (2 hops)
via 192.168.2.2	Specifies the address of the next-hop router (R2) to send traffic to for the remote network.
00:00:23	Specifies the amount of time since the route was updated (here, 23 seconds). Another update is due in 7 seconds.
Serial0/0/0	192.168.4.2

Verification and Troubleshooting

- **Passive interface** command: Used to prevent a router from sending updates through an interface

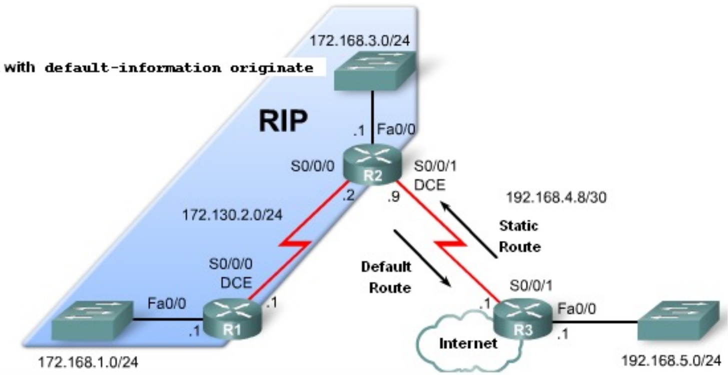
```
R2(config)#router rip
R2(config-router)#passive-interface FastEthernet 0/0
R2(config-router)#end
R2#show ip protocols
Routing Protocol is "rip"
    Sending updates every 30 seconds, next due in 14 seconds
    Invalid after 180 seconds, hold down 180, flushed after 240
    Outgoing update filter list for all interfaces is
    Incoming update filter list for all interfaces is
    Redistributing: rip
    Default version control: send version 1, receive any version
        Interface             Send  Recv  Triggered RIP  Key-chain
        Serial0/0/0           1     1 2
        Serial0/0/1           1     1 2
    Automatic network summarization is in effect
    Routing for Networks:
        192.168.2.0
        192.168.3.0
        192.168.3.0
        192.168.4.0
    Passive Interface(s):
       FastEthernet0/0
    Routing Information Sources:
    Gateway         Distance      Last Update
       192.168.2.1          120       00:00:27
       192.168.4.1          120       00:00:23
Distance: (default is 120)
```

Notice FastEthernet 0/0 is no longer listed under "Default version contol:"
However, R2 is still routing for 192.168.3.0 and now lists FastEthernet under "Passive Interfaces:"

Default Route and RIP

- Propagating the Default Route in RIP
- *Default-information originate* command
  - used to specify that the router is to originate default information, by propagating the static default route in RIP.

Propagating a Default with `default-information originate`




```
R2(config)#router rip
R2(config-router)#default-information originate
R2(config-router)#end
R2#debug ip rip
RIP protocol debugging is on
RIP: sending v1 update to 255.255.255.255 via Serial0/0/0 (172.30.2.2)
RIP: build update entries
            subnet 0.0.0.0 metric 1
            subnet 172.30.3.0 metric 2
R2#undebug all
All possible debugging has been turned off
```

R2 is now sending a "quad-zero" route to R1.

```
R1#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B- BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route

Gateway of last resort is 172.30.2.2 to network 0.0.0.0

   172.30.0.0/24 is subnetted, 3 subnets
C  172.30.2.0 is directly connected, Serial0/0/0
R  172.30.3.0 [120/1] via 172.30.2.2 00:00:16, Serial0/0/0
C  172.30.1.0 is directly connected, Fast Ethernet0/0
R* 0.0.0.0/0 [120/1] via 172.30.2.2, 00:00:16, Serial0/0/0
```

R1 has a "gateway of last resort", a candidate default route.

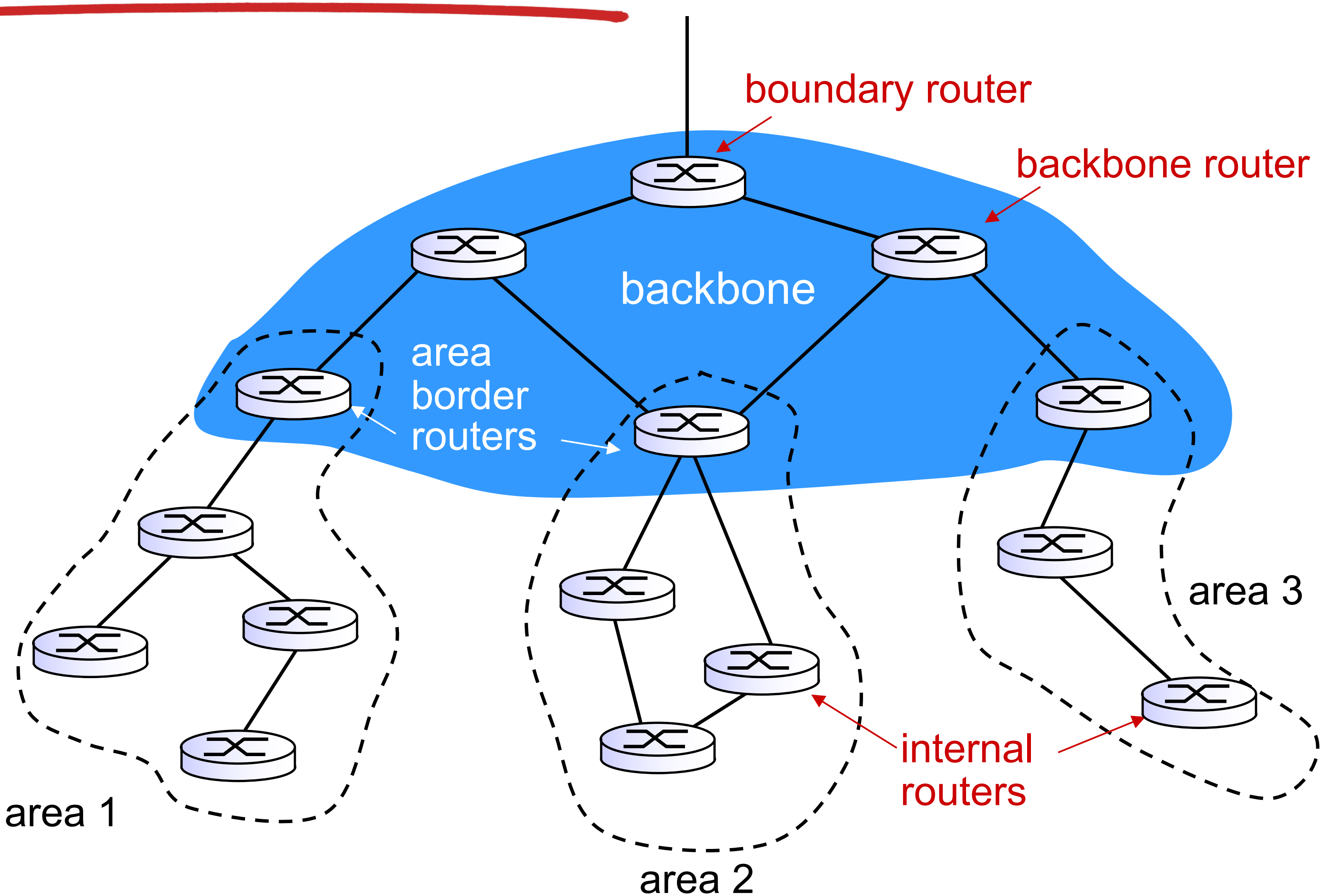
OSPF (Open Shortest Path First)

- "open": publicly available
- uses link state algorithm
  - LS packet dissemination
  - topology map at each node
  - route computation using Dijkstra's algorithm
- OSPF advertisement carries one entry per neighbor
- advertisements flooded to *entire* AS
  - carried in OSPF messages directly over IP (rather than TCP or UDP
- *IS-IS routing* protocol: nearly identical to OSPF

OSPF "advanced" features (not in RIP)

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

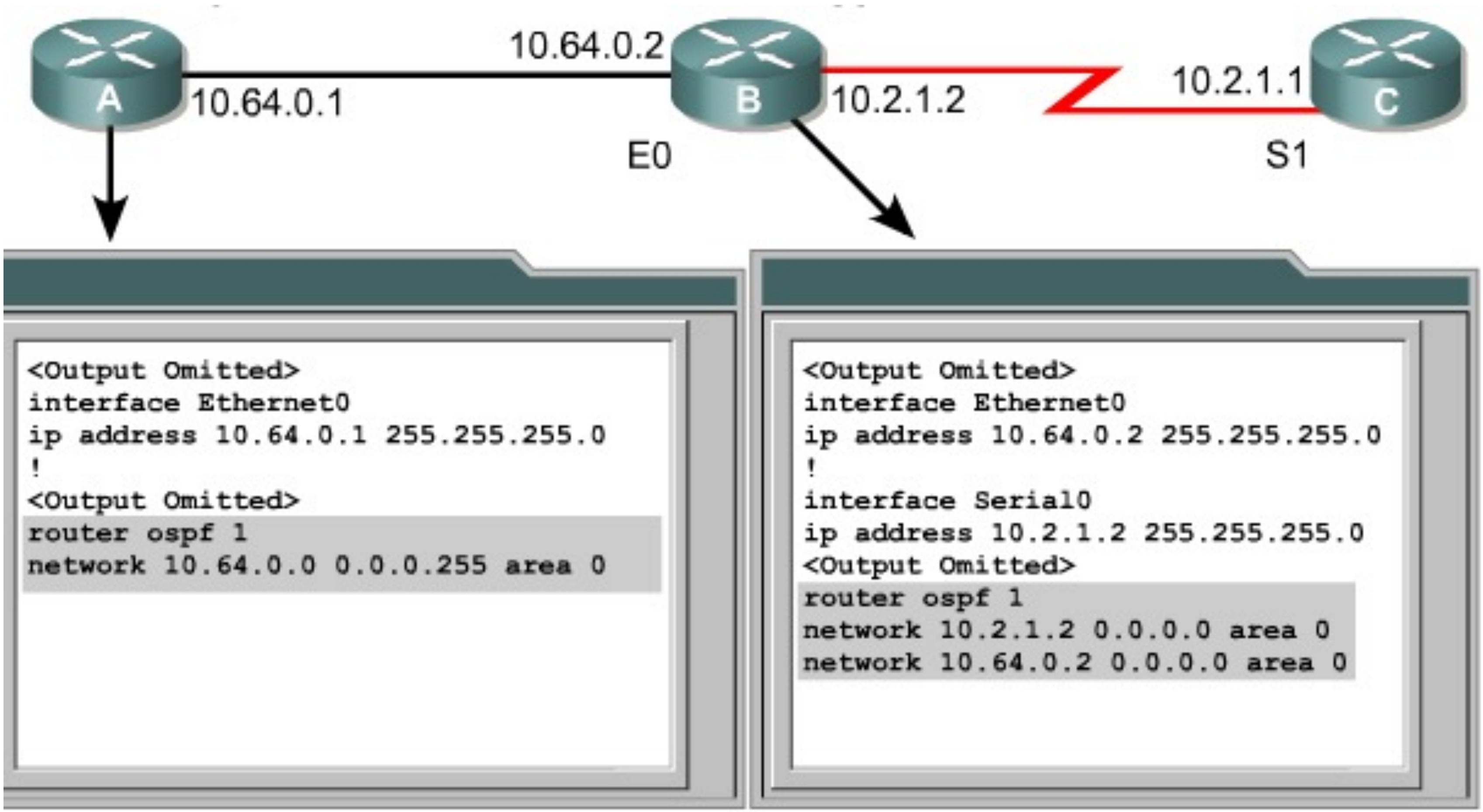
Hierarchical OSPF



Hierarchical OSPF

- *two-level hierarchy:* local area, backbone.
  - link-state advertisements only in area
  - each nodes has detailed area topology; only know direction (shortest path) to nets in other areas.
- *area border routers:* "summarize" distances to nets in own area, advertise to other Area Border routers.
- *backbone routers:* run OSPF routing limited to backbone.
- *boundary routers:* connect to other AS's.

Basic OSPF Configuration on Cisco’s router



(*) *The following 3 slides are from Cisco’s CCNA 3.1.*

Basic OSPF Configuration on Cisco's router

Network area Command	Description
address	Can be the network address, subnet, or the address of the interface. Instructs router to know which links to advertise, which links to listen to advertisements on, and what networks to advertise.
wildcard-mask	An inverse mask used to determine how to read the address. The mask has wildcard bits where 0 is a match and 1 is "do not care"; for example, 0.0.255.255 indicates a match in the first two bytes. (the equivalent REGULAR subnet mask would be a 16 bit mask of 255.255.0.0) If specifying the interface address, use mask 0.0.0.0.
area-id	Specifies the area to be associated with the address. Can be a number or can be similar to an IP address A.B.C.D. For a backbone area, the ID must equal 0.

Verifying OSPF Configuration

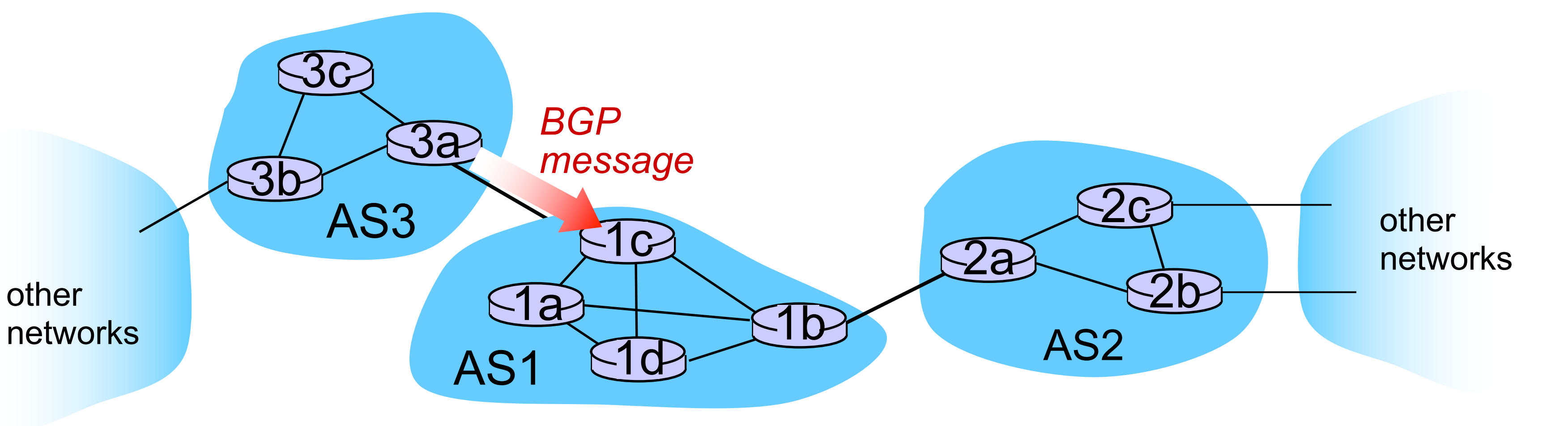
- `show ip protocol`
- `show ip route`
- `show ip ospf interface`
- `show ip ospf`
- `show ip ospf neighbor detail`
- `show ip ospf database`

Internet inter-AS routing: BGP

- BGP (Border Gateway Protocol): *the* de facto inter-domain routing protocol
  - "glue that holds the Internet together"
- BGP provides each AS a means to:
  - eBGP: obtain subnet reachability information from neighboring ASs.
  - iBGP: propagate reachability information to all AS-internal routers.
  - determine "good" routes to other networks based on reachability information and policy.
- allows subnet to advertise its existence to rest of Internet: *"I am here"*

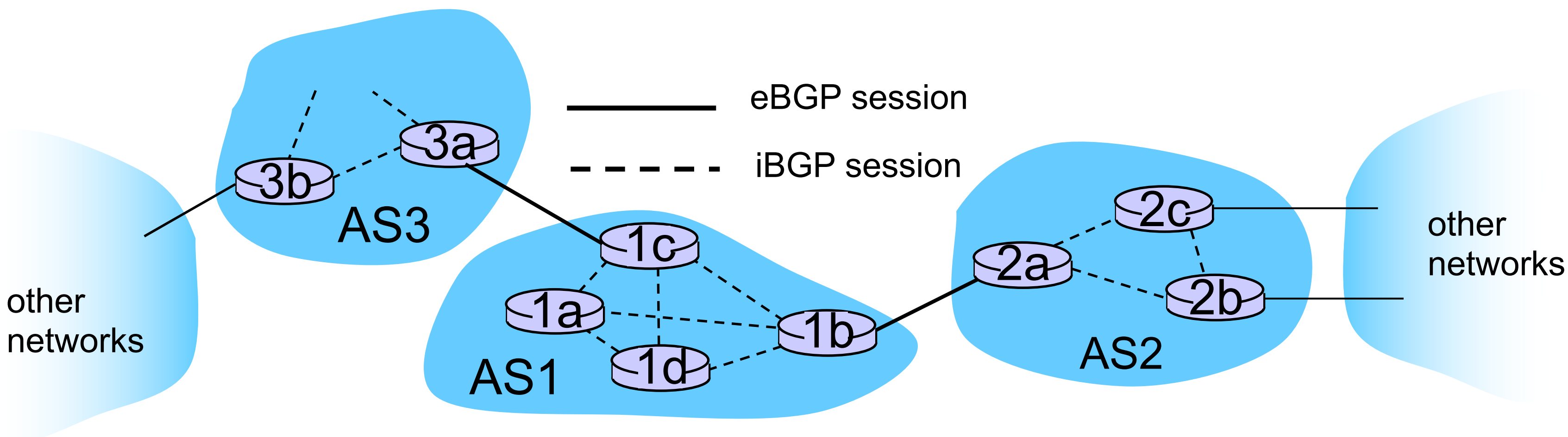
BGP basics

- BGP session: two BGP routers ("peers") exchange BGP messages:
  - advertising *paths* to different destination network prefixes ("path vector" protocol)
  - exchanged over semi-permanent TCP connections
- when AS3 advertises a prefix to AS1:
  - AS3 *promises* it will forward datagrams towards that prefix
  - AS3 can aggregate prefixes in its advertisement



BGP basics: distributing path information

- using eBGP session between 3a and 1c, AS3 sends prefix reachability info to AS1.
  - 1c can then use iBGP do 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, it creates entry for prefix in its forwarding table.



Path attributes and BGP routes

- advertised prefix includes BGP attributes
  - prefix + attributes = “route”
- two important attributes:
  - AS-PATH: contains ASs through which prefix advertisement has passed: e.g., AS 67, AS 17
  - NEXT-HOP: indicates specific internal-AS router to next-hop AS. (may be multiple links from current AS to next-hop-AS)
- gateway router receiving route advertisement uses import policy to accept/decline
  - e.g., never route through AS x
  - *policy-based* routing

BGP route selection

- router may learn about more than 1 route to destination AS, selects route based on:
  1. local preference value attribute: policy decision
  2. shortest AS-PATH
  3. closest NEXT-HOP router: hot potato routing
  4. additional criteria

Why different Intra-, Inter-AS routing ?

policy:

- inter-AS: admin wants control over how its traffic routed, who routes through its net.
- intra-AS: single admin, so no policy decisions needed

scale:

- hierarchical routing saves table size, reduced update traffic

performance:

- intra-AS: can focus on performance
- inter-AS: policy may dominate over performance

Summary

5.1 introduction

5.2 what's inside a router

5.3 IP: Internet Protocol

- datagram format
- DHCP
- ICMP
- IPv6

5.4 static routing

5.5 routing algorithms

- link state
- distance vector
- hierarchical routing

5.6 routing in the Internet

- RIP
- OSPF
- BGP

- understand principles behind network layer services:
  - network layer service models, forwarding versus routing how a router works, routing (path selection)
- instantiation, implementation in the Internet