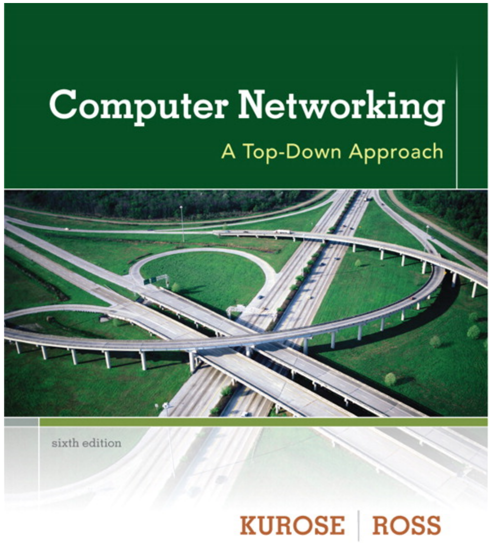
Chapter 4
Network Layer



A note on the use of these ppt slides:

We're making these slides freely available to all (faculty, students, readers). They're in PowerPoint form so you see the animations; and can add, modify, and delete slides (including this one) and slide content to suit your needs. They obviously represent a *lot* of work on our part. In return for use, we only ask the following:

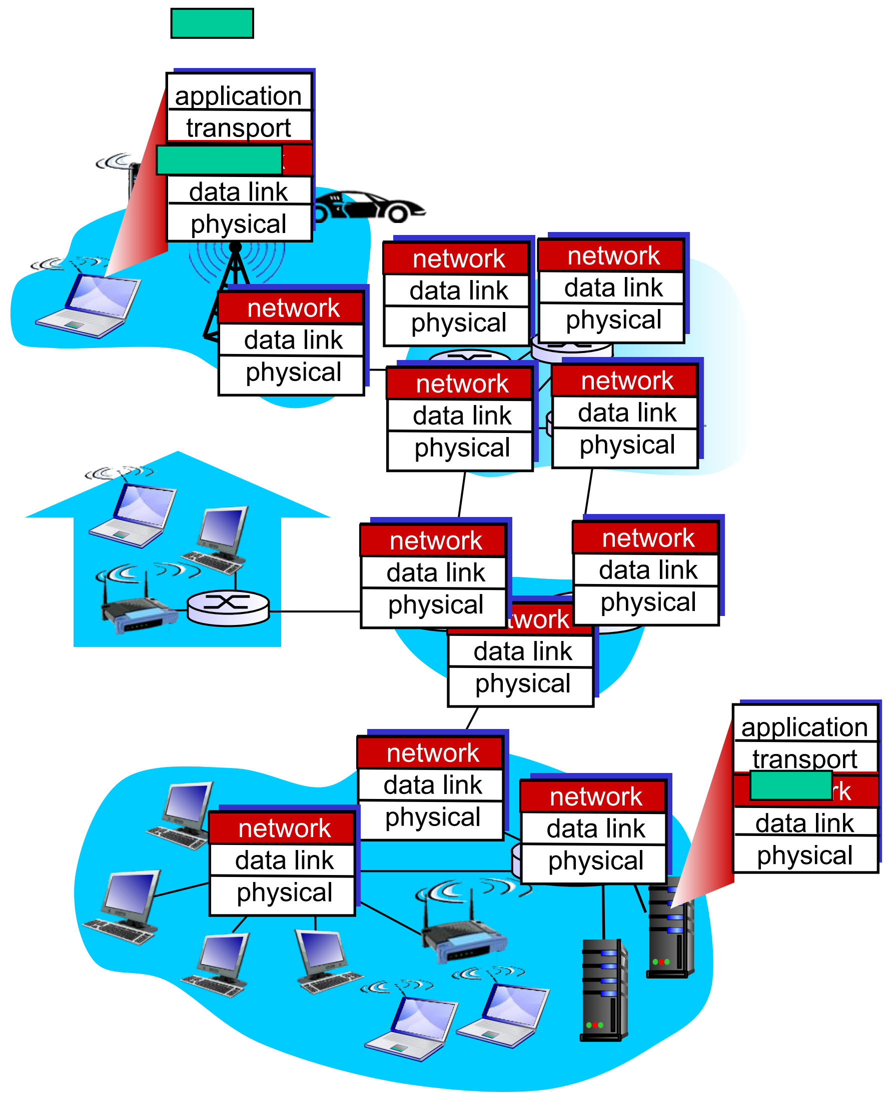
- If you use these slides (e.g., in a class) that you mention their source (after all, we'd like people to use our book!)
- If you post any slides on a www site, that you note that they are adapted from (or perhaps identical to) our slides, and note our copyright of this material.

Thanks and enjoy! JFK/KWR

Computer Networking: A Top Down Approach
6th edition
Jim Kurose, Keith Ross
Addison-Wesley
March 2012

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

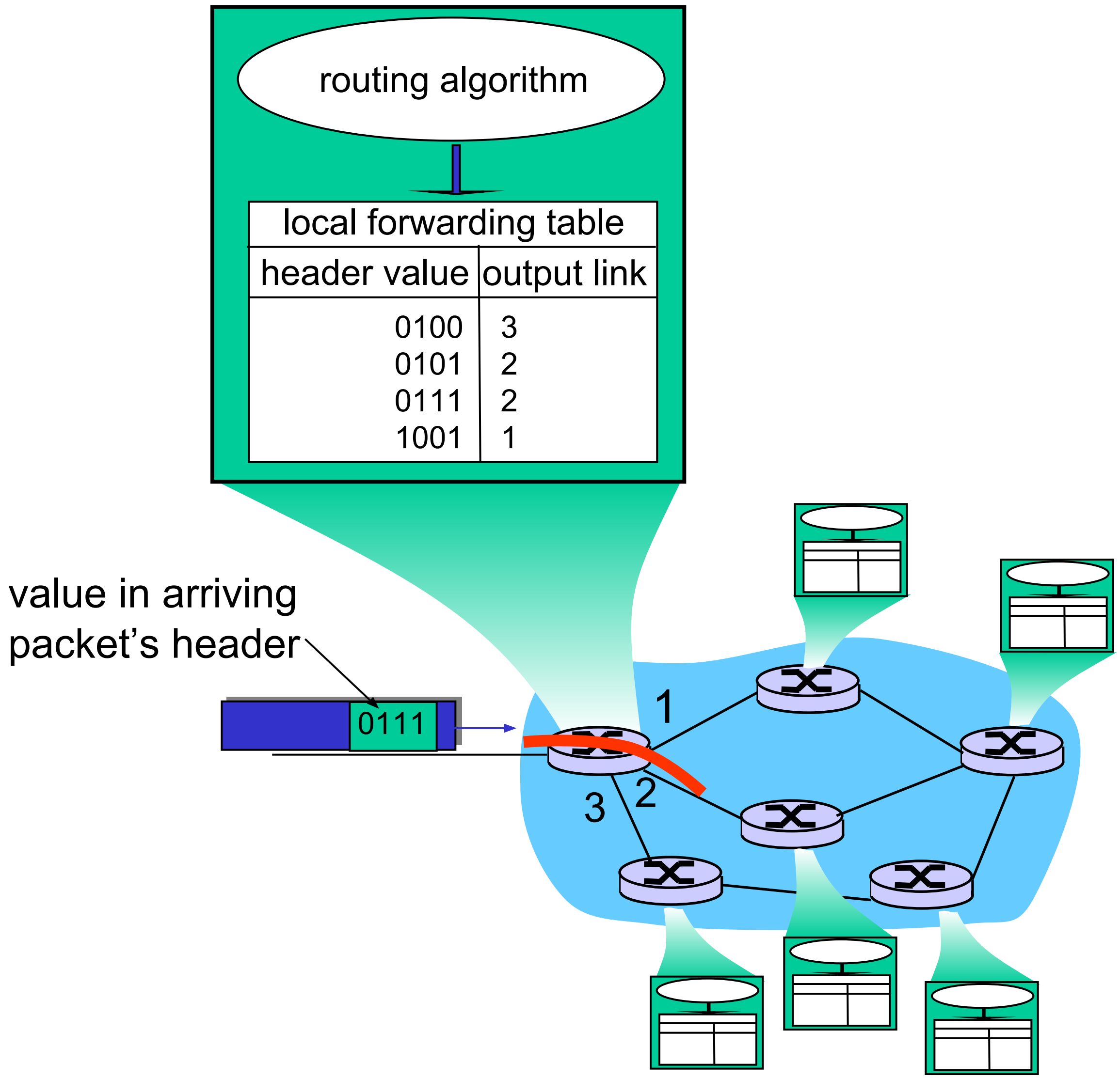
- *forwarding:* move packets from router's input to appropriate router output
- *routing:* The network layer must determine the route or path taken by packets as
  they flow from a sender to a receiver

  routing algorithms

analogy:

- *routing:* process of planning trip from source to dest
- *forwarding:* process of getting through single interchange

Interplay between routing and forwarding



Chapter 4: outline

4.1 introduction

4.2 virtual circuit and datagram networks

4.3 what's inside a router

4.4 IP: Internet Protocol
- datagram format
- IPv4 addressing
- ICMP
- IPv6

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

4.6 routing in the Internet
- RIP
- OSPF

4.7 broadcast and multicast routing

Connection, connection-less service

- *datagram* network provides network-layer *connectionless* service
- *virtual-circuit* network provides network-layer *connection* service

VC implementation

a VC consists of:

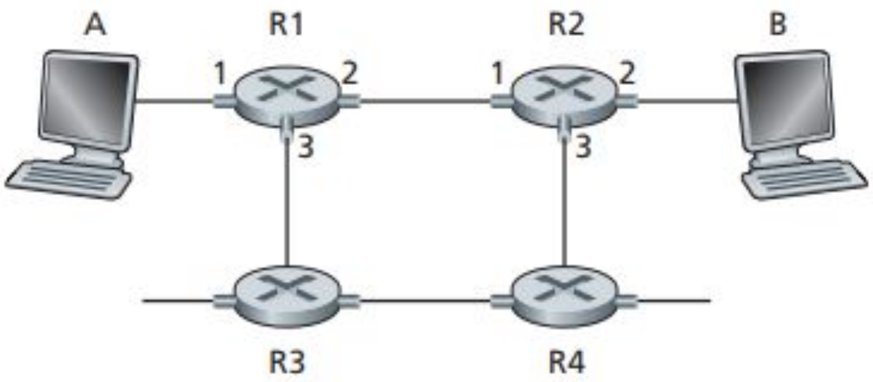
1. *path* from source to destination
2. *VC numbers*, one number for each link along path
3. *entries in forwarding tables* in routers along path

- It has 3 phases:

1. Set up phase 2. Data transmission phase 3. Teardown Phase

- Packet belonging to a virtual circuit will carry a VC number in its header. Each intervening router must
replace the VC number of each traversing packet with a new VC number. The new VC number is obtained from the forwarding table.
- Host A requests that the network establish a VC between itself and Host B. The network chooses the path A-R1-R2-B and assigns VC numbers 12, 22, and 32 to the three links in this path for this virtual circuit. In this case, when a packet in this VC leaves Host A, the value in the VC number field in the packet header is 12; when it leaves R1, the value is 22; and when it leaves R2, the value is 32.

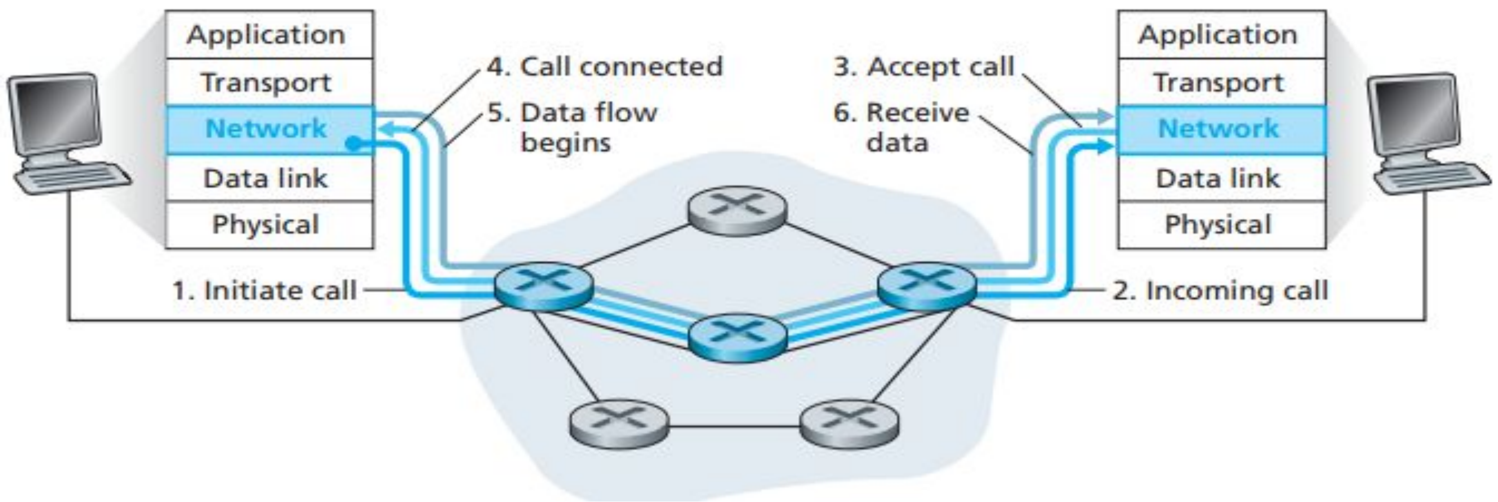
Incoming Interface	Incoming VC #	Outgoing Interface	Outgoing VC #
1	12	2	22
2	63	1	18
3	7	2	17
1	97	3	87
...	...	...	...



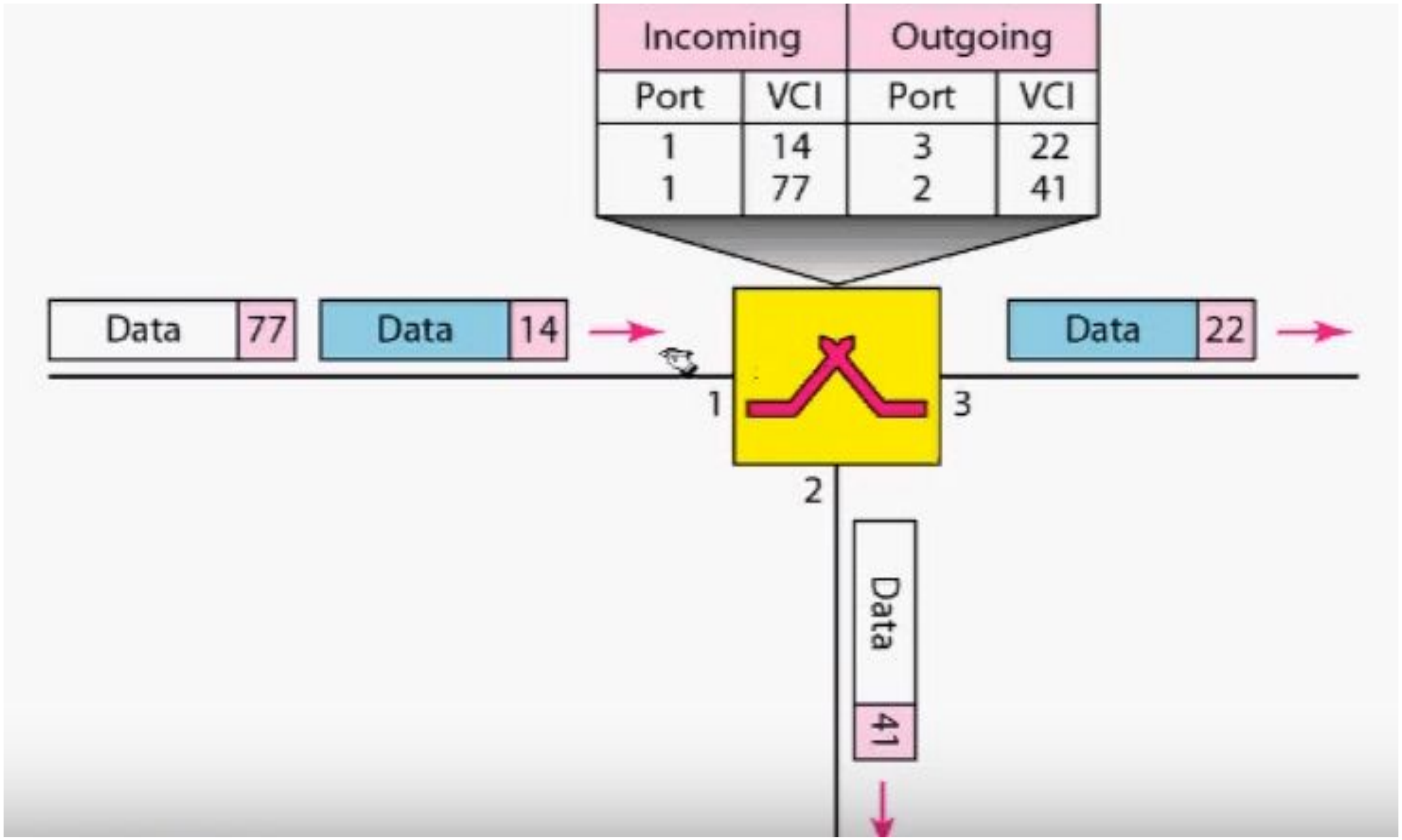
- There are three identifiable phases in a virtual circuit:
  • *VC setup.* During the setup phase, the sending transport layer contacts the network layer, specifies the receiver's address, and waits for the network to set upt he VC. The network layer determines the path between sender and receiver, that is, the series of links and routers through which all packets of the VC will travel.

  • *Data transfer.* As shown in Figure 4.4, once the VC has been established, packets can begin to flow along the VC.
  • *VC teardown.* This is initiated when the sender (or receiver) informs the network layer of its desire to terminate the VC.

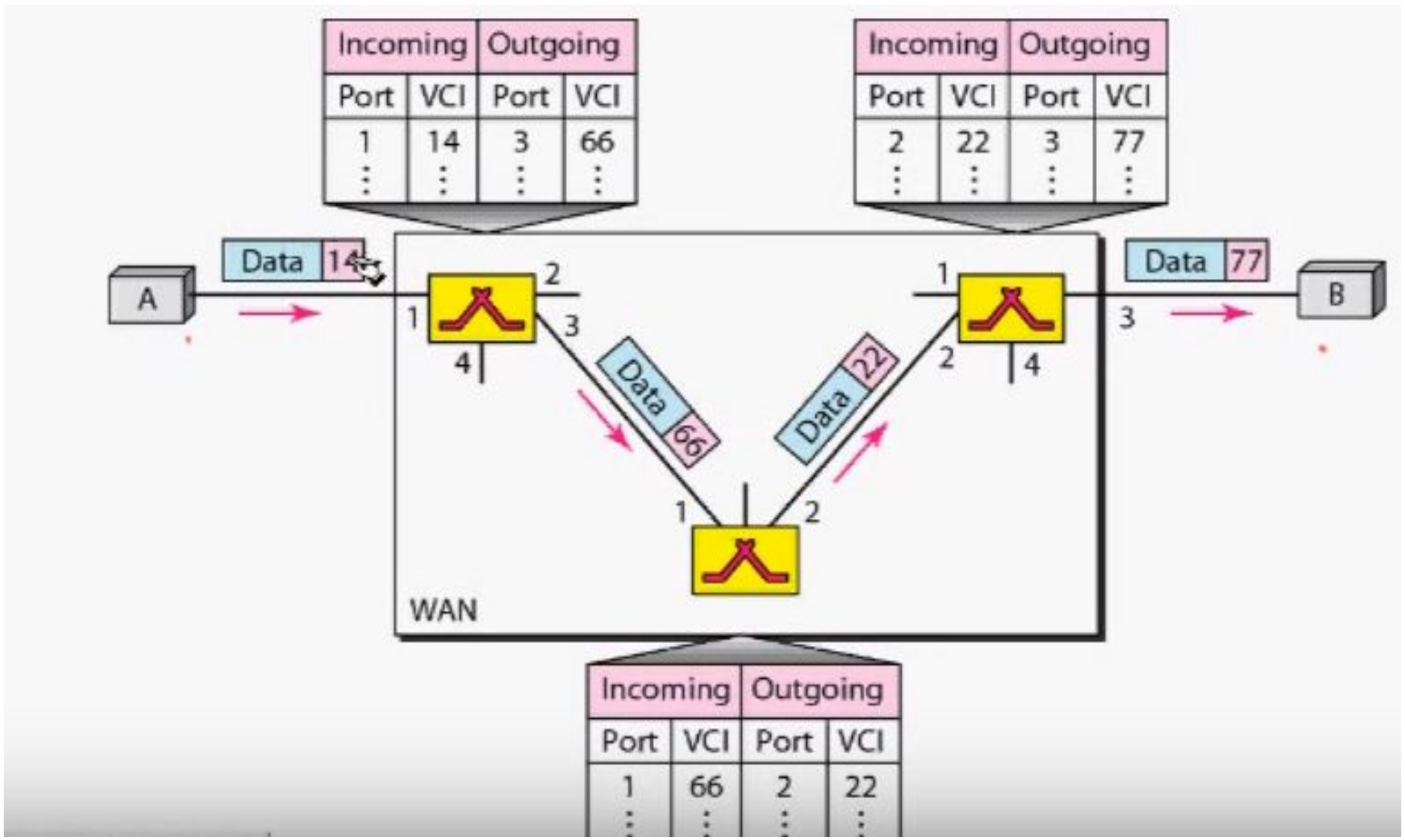
Virtual circuit setup



VC implementation

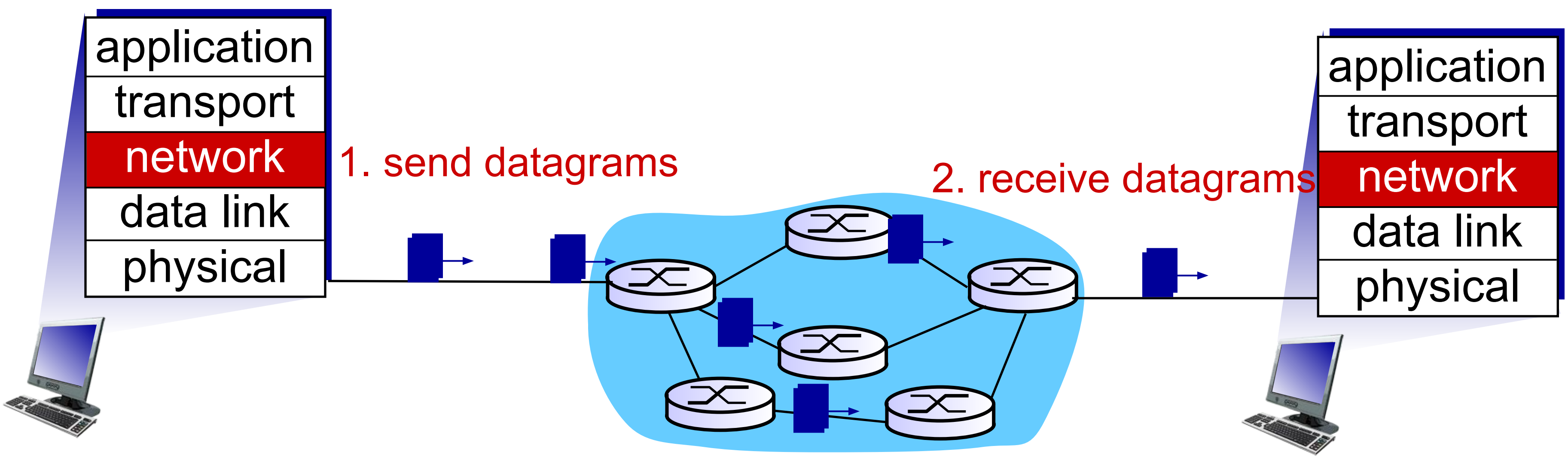


VC implementation



Datagram networks

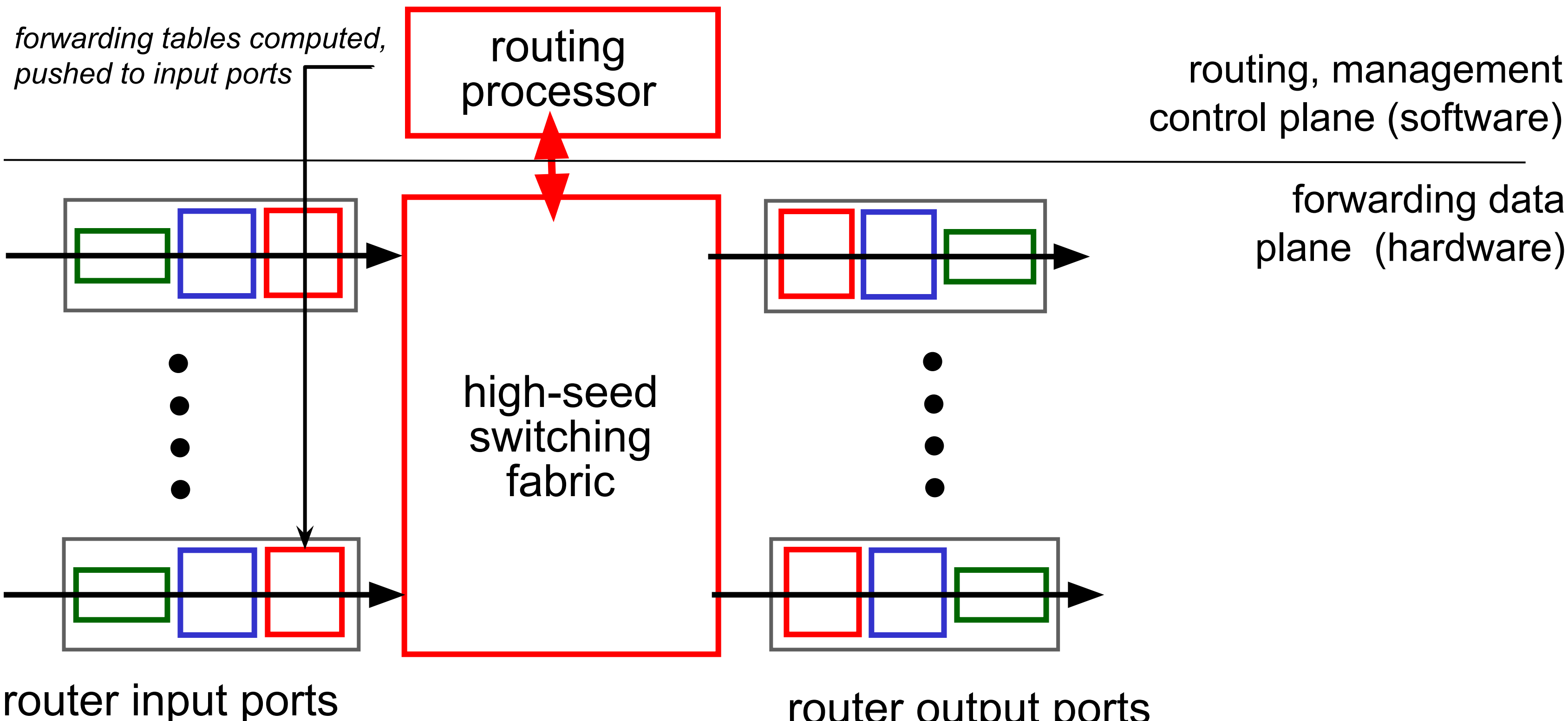
- In a **datagram network**, each time an end system wants to send a packet, it stamps the packet with the address of the destination end system and then pops the packet into the network
- no call setup at network layer



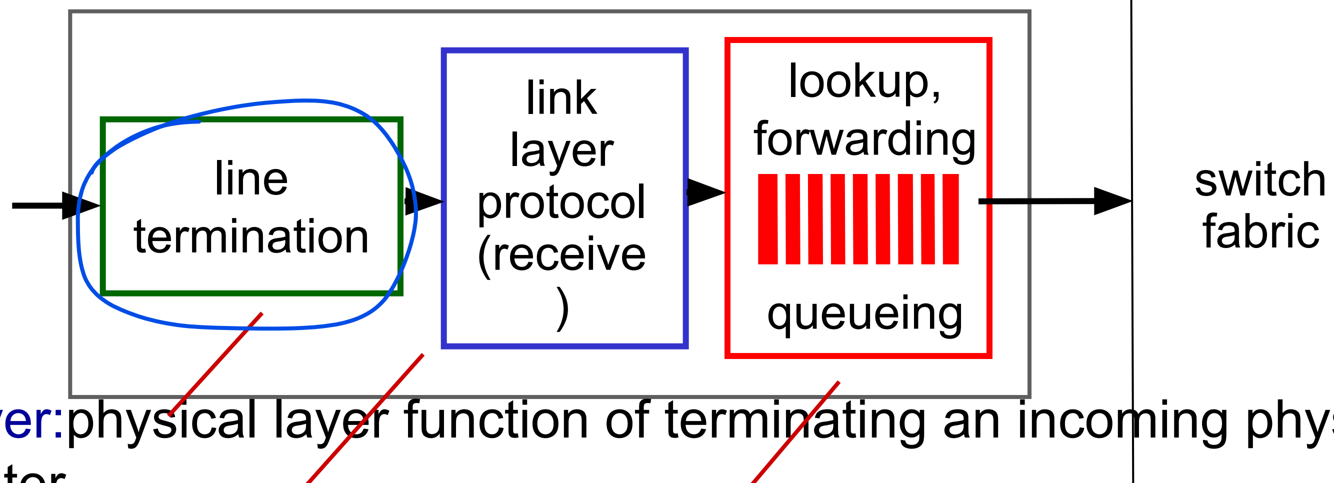
Router architecture overview

two key router functions:

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



Input port functions



physical layer: physical layer function of terminating an incoming physical link at a router

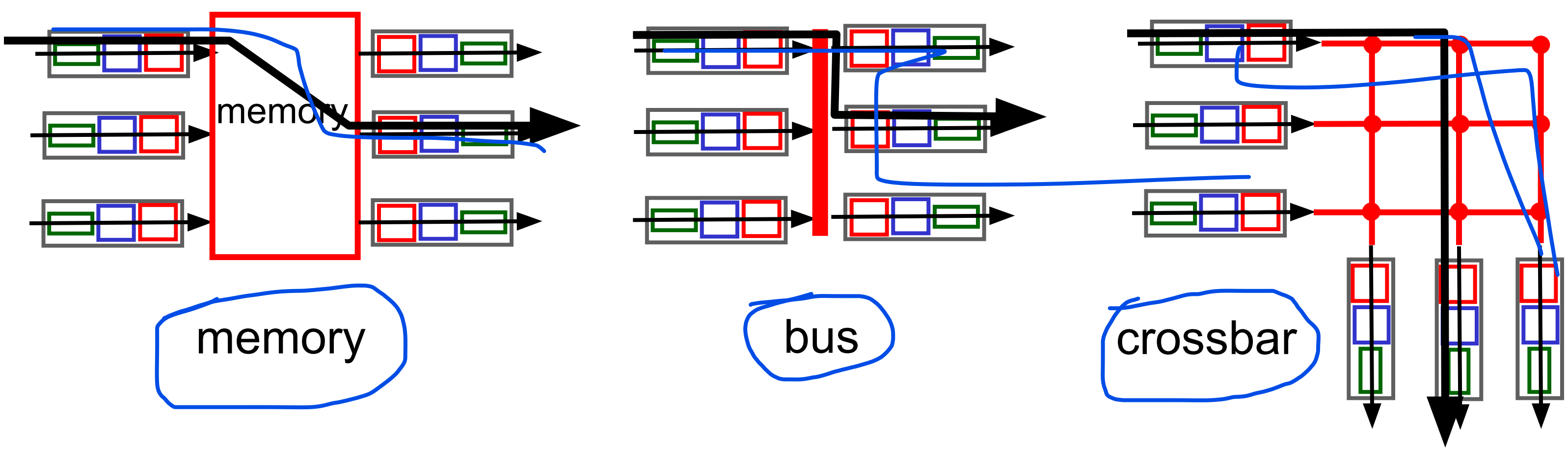
data link layer:
to interoperate with the link layer at the other side of the incoming link

decentralized switching:

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

Switching fabrics

- transfer packet from input buffer to appropriate output buffer
- switching rate: rate at which packets can be transfer from inputs to outputs
- three types of switching fabrics

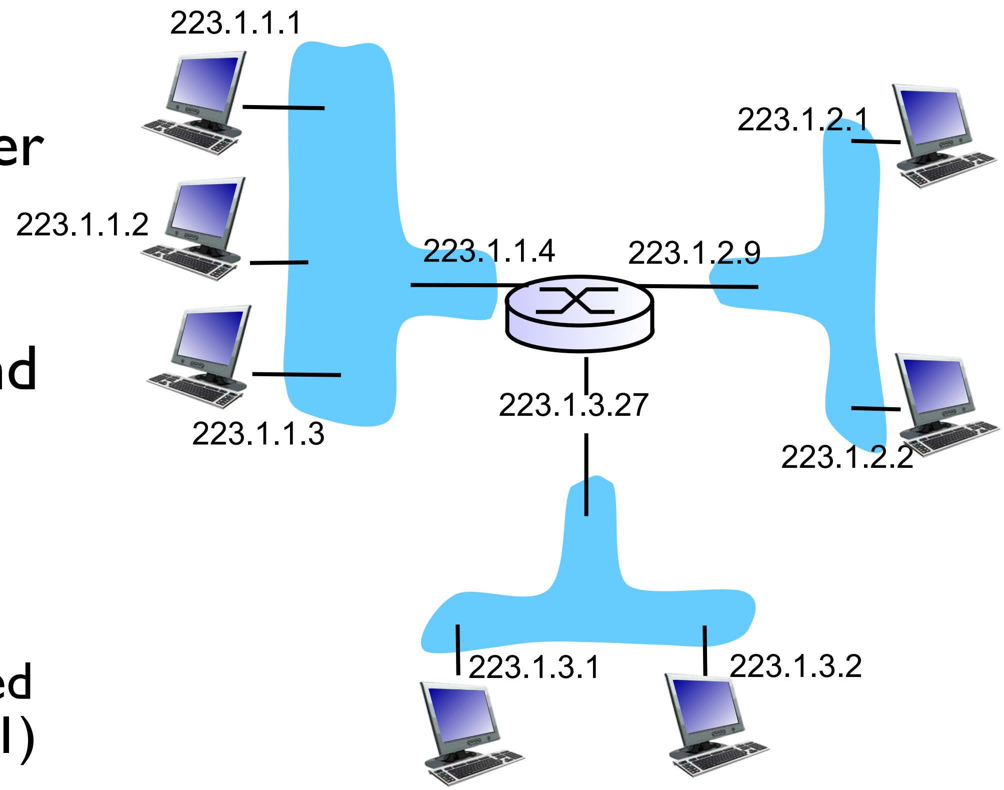


Output ports

- An output port stores packets received from the switching fabric and transmits these packets on the outgoing link by performing the necessary link-layer and physical-layer functions

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 typically has one or two interfaces (e.g., wired Ethernet, wireless 802.11)
- *IP addresses associated with each interface*

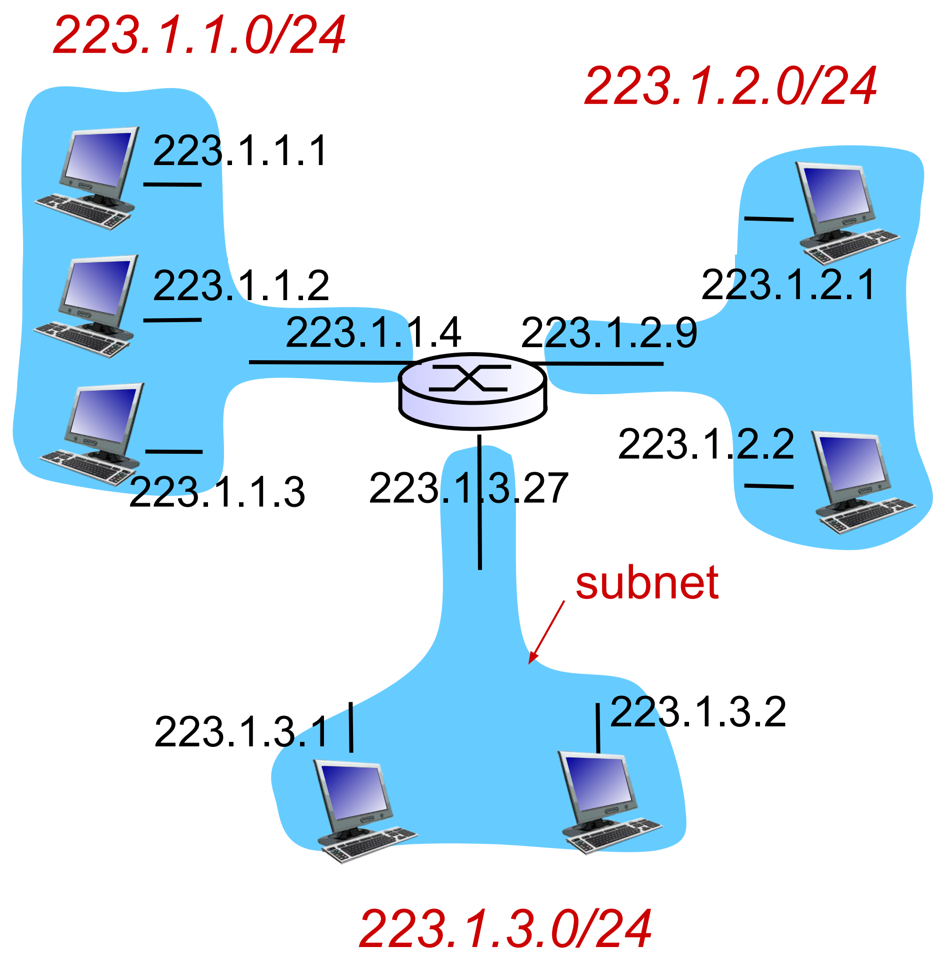


$$223.1.1.1 = \underbrace{11011111}_{223}\ \underbrace{00000001}_{1}\ \underbrace{00000001}_{1}\ \underbrace{00000001}_{1}$$

Subnets

recipe

- to determine the subnets, detach each interface from its host or router, creating islands of isolated networks
- each isolated network is called a *subnet*



subnet mask: /24

IP addresses: how to get one?

Q: How does a *host* get IP address?

- hard-coded by system admin in a file
  - Windows: control-panel->network->configuration->tcp/ip->properties
  - UNIX: /etc/rc.config
- DHCP: Dynamic Host Configuration Protocol: dynamically get address from as server
  - "plug-and-play"

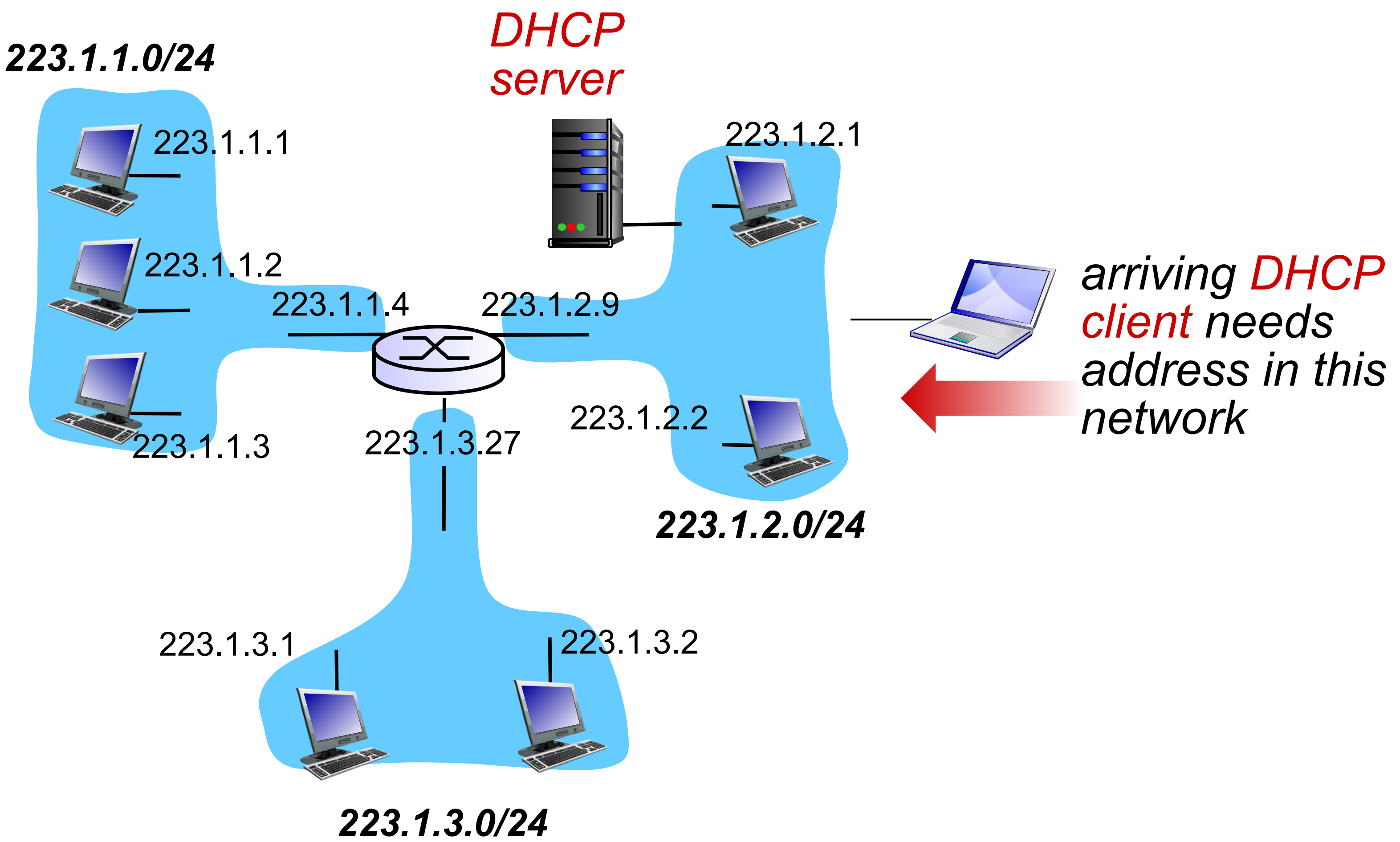
DHCP: Dynamic Host Configuration Protocol

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

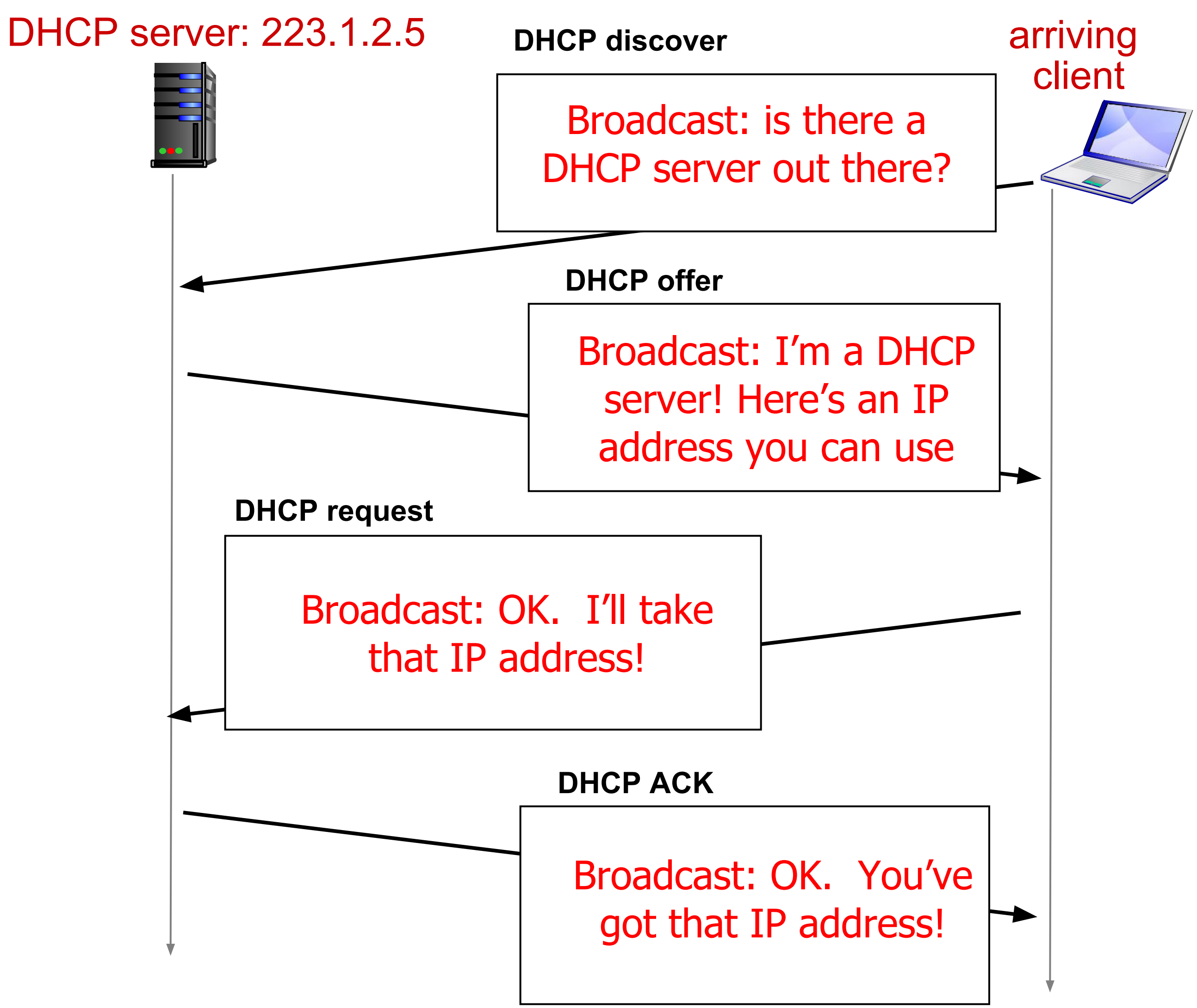
DHCP overview:

- 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

DHCP client-server scenario



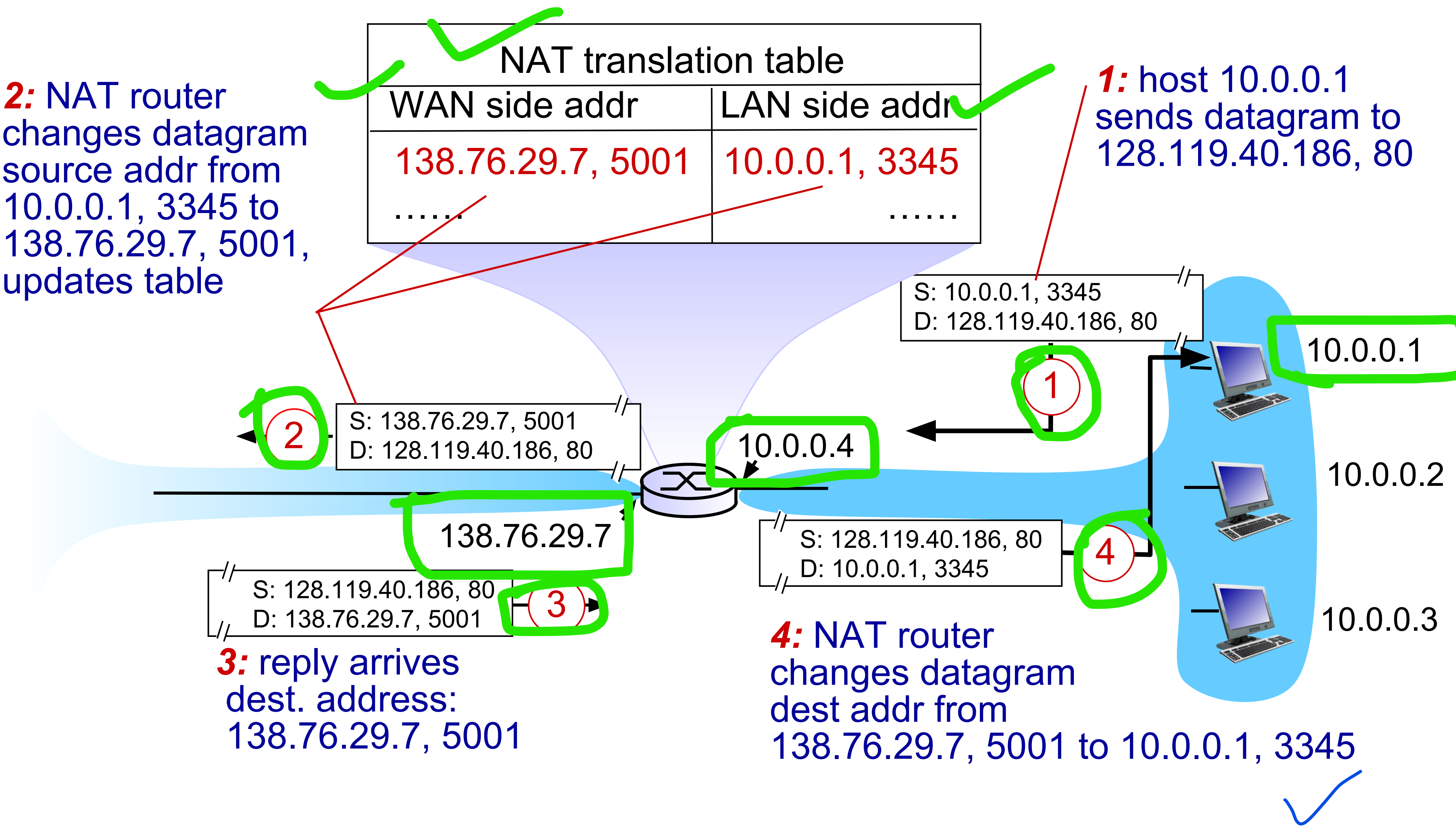
DHCP client-server scenario



NAT: network address translation

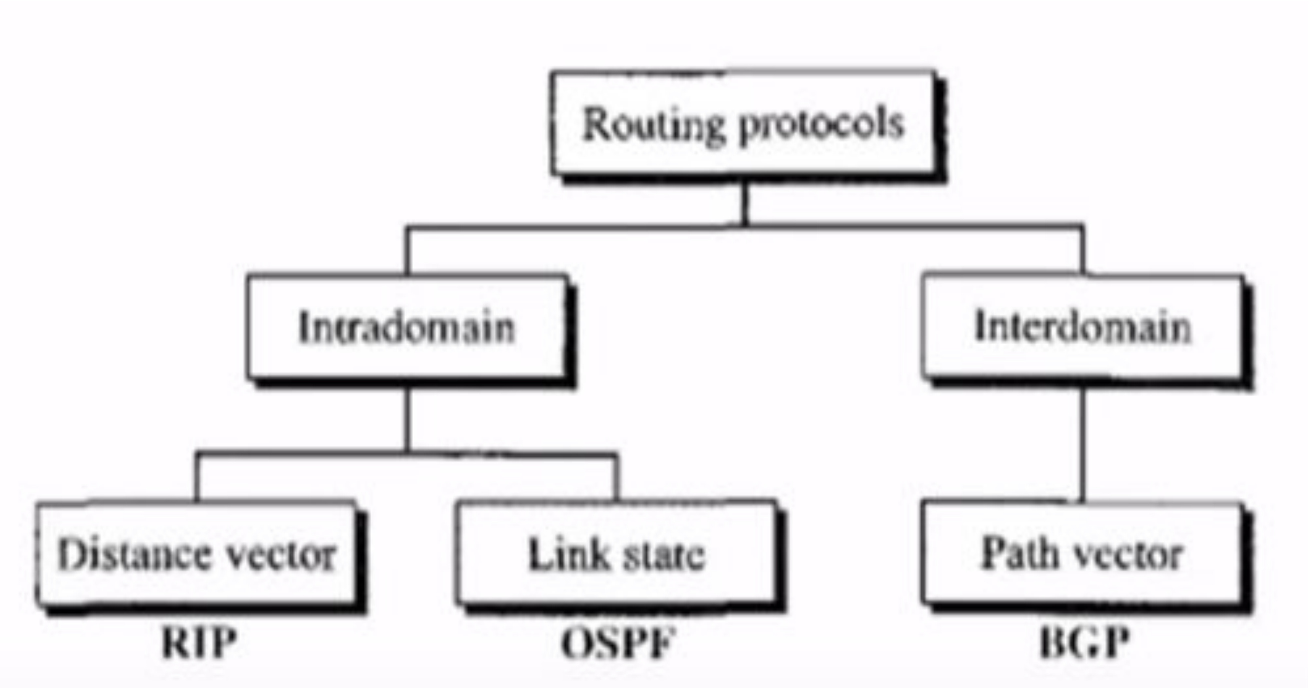
- *Network Address Translation* (NAT) is the process where a network device, assigns a public address to a computer (or group of computers) inside a private network.

NAT: network address translation



Chapter 4: outline

Routing Protocol



A Link-State Routing Algorithm

Dijkstra's algorithm

Nodes compute their forwarding table in the same distributed setting as for distance vector:

1. Nodes know only the cost to their neighbors; not the topology
2. Nodes can talk only to their neighbors using messages
3. All nodes run the same algorithm concurrently
4. Nodes/links may fail, messages may be lost

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

Proceeds in two phases:

1. Nodes flood topology in the form of link state packets
   - Each node learns full topology
2. Each node computes its own forwarding table
   - By running Dijkstra (or equivalent)

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'
```

Distance vector algorithm

Bellman-Ford equation (dynamic programming)

let

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

then

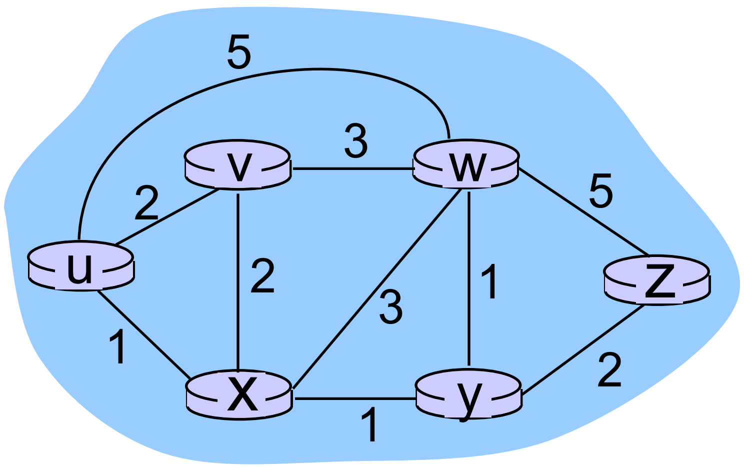
$$d_x(y) = \min_v \{c(x,v) + d_v(y)\}$$

cost from neighbor v to destination y

cost to neighbor v

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:

$$d_u(z) = \min \{ c(u,v) + d_v(z), \; c(u,x) + d_x(z), \; c(u,w) + d_w(z) \}$$

$$= \min \{2 + 5, \; 1 + 3, \; 5 + 3\} = 4$$

$0 + 2 < \infty$

Previous + ed

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

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

LS = Link State
DV = Distance Vector

robustness: what happens if router malfunctions?

LS:

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

DV:

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

Chapter 4: outline

4.1 introduction

4.2 virtual circuit and datagram networks

4.3 what's inside a router

4.4 IP: Internet Protocol
- datagram format
- IPv4 addressing
- ICMP
- IPv6

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

4.6 routing in the Internet
- RIP
- OSPF

4.7 broadcast and multicast routing

- IPv6 has three types of addresses: unicast, multicast and anycast.
- Unicast is a type of communication where data is sent from one computer to another computer. Unicast is a one-to-one type of network communication. This type of communication is the option when clients need different data from network server.
- In Unicast type of communication, there is only one sender, and only one receiver.
- Example for IPv6 Unicast type of network communication:
- 1) Browsing a website. (Webserver is the sender and your computer is the receiver.)
- 2) Downloading a file from a FTP Server. (FTP Server is the sender and your computer is the receiver.)

- Multicast is a type of communication where multicast traffic addressed for a group of devices on the network. IPv6 multicast traffic are sent to a group and only members of that group receive the Multicast traffic. Example: Service Location Protocol
- Anycast is a type of IPv6 network communication in which IPv6 datagrams from a source are routed to the nearest device (in terms of routing distance) from a group servers which provide the same service.