Course on Computer Communication and Networks

Lecture 6
Network Layer,
Chapter 4; Part A (7/e Ch4)

EDA344/DIT 420, CTH/GU

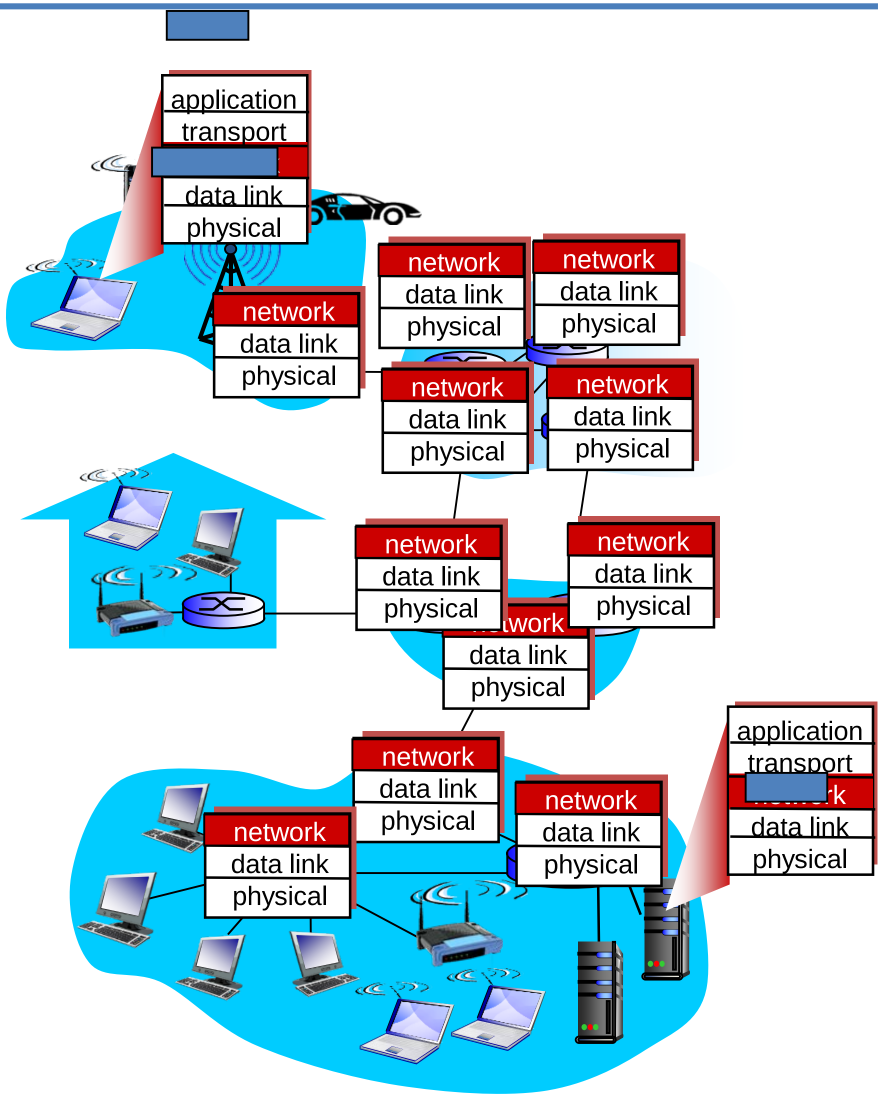
Based on the book Computer Networking: A Top Down Approach, Jim Kurose, Keith Ross, Addison-Wesley.

Network layer

Consider transporting a segment from sender to receiver

- sending side: encapsulates segments into datagrams
- receiving side: delivers segments to transport layer

- network layer protocols in *every* host, router
  - examines header fields in all datagrams passing through it



Interplay between routing and forwarding

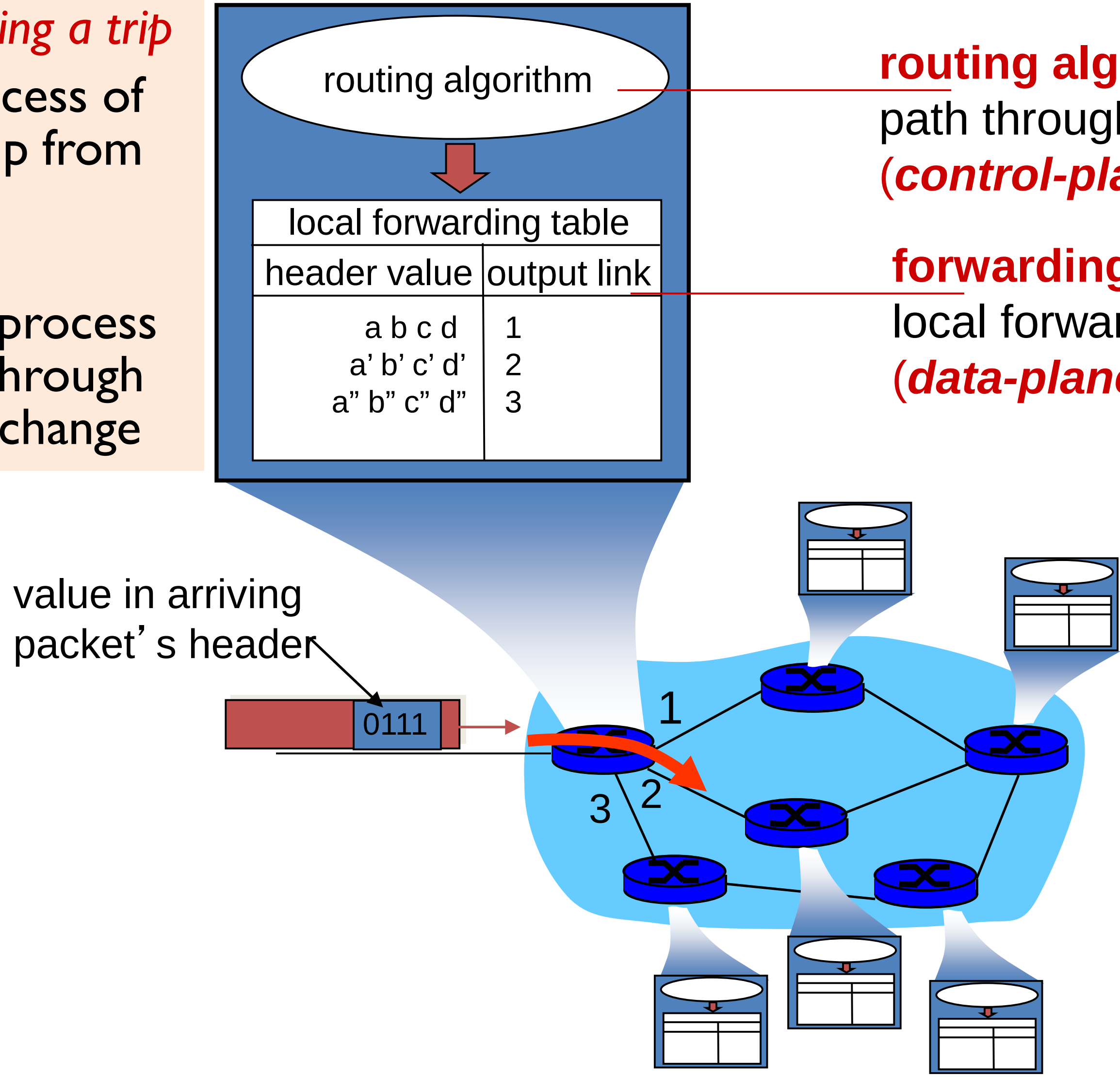
analogy: taking a trip

routing: process of planning trip from source to destination

forwarding: process of getting through single interchange

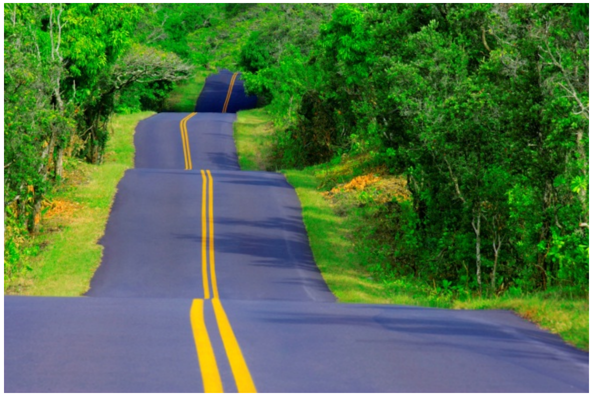
routing algorithm determines path through network (***control-plane*** functionality)

forwarding table determines local forwarding at this router (***data-plane*** functionality)



Roadmap Network Layer

- Forwarding versus routing
- Network layer service models
  - Network layer architecture (shift): Software-Defined Networks
- Inside a routerswitching fabrique
- The Internet Network layer: IP, Addressing & related
- (Next) Control, routing
  - path selection
  - instantiation, implementation in the Internet



Network service model

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

(general networking scope, ie not Internet-scope)

*example services for **individual** packets:*

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

*example services for a **flow** of packets:*

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

Connection, connection-less service

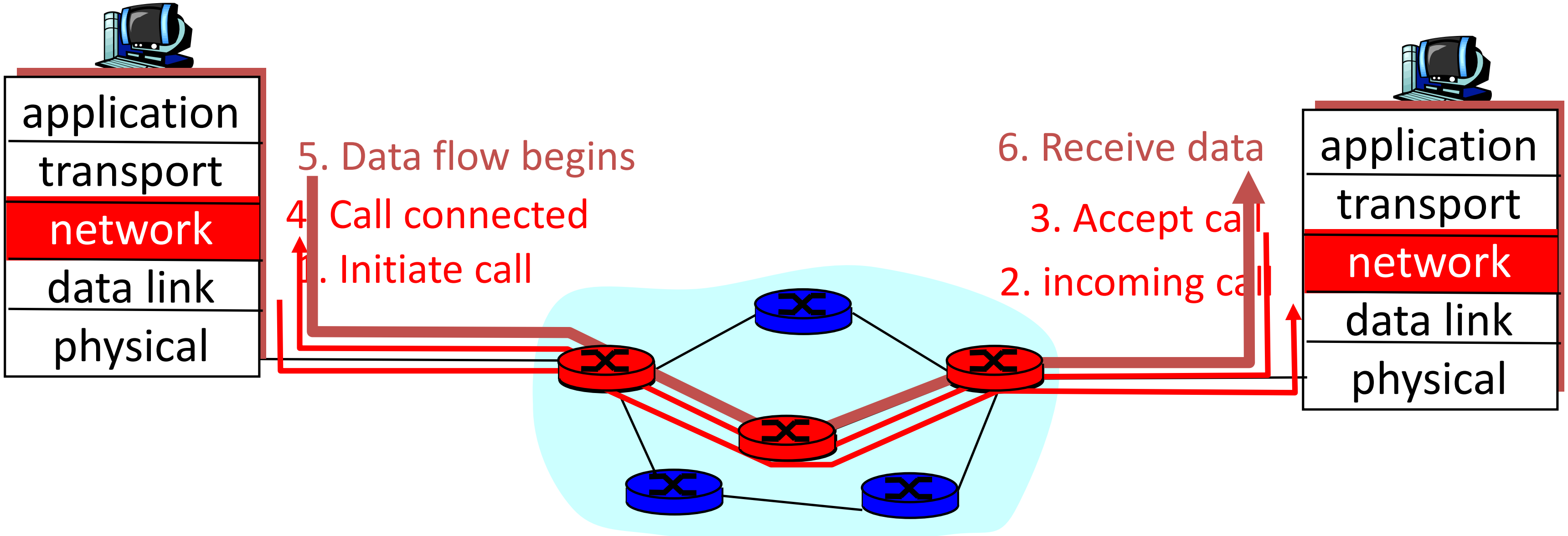
- *datagram* network provides network-layer *connectionless* service
  - classic Internet model
- *virtual-circuit* network can provide network-layer *connection-oriented* service
  - not present in Internet but efforts to simulate behaviour are being made

- analogous to TCP/UDP connection-oriented / connectionless transport-layer services, **but**:
  - *service:* host-to-host
  - *implementation:* in network core

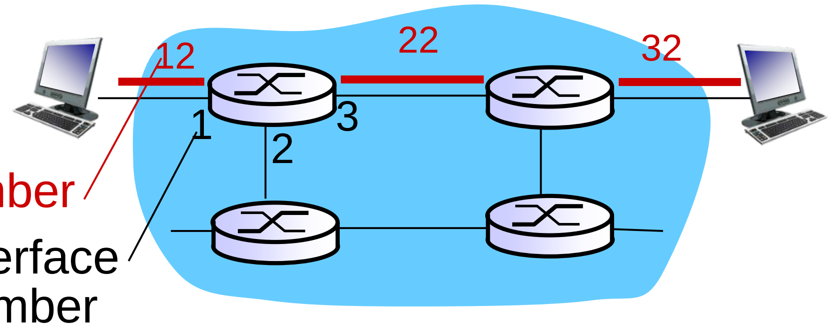
Virtual circuits:

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

- call setup, teardown for each call *before* data can flow
  - signaling protocols to setup, maintain, teardown VC (ATM, frame-relay, X.25; not in IP)
- each packet carries VC identifier (not destination host)
- *every* router maintains "state" for each passing connection
- resources (bandwidth, buffers) may be *allocated* to VC (dedicated resources = predictable service)



VC forwarding table



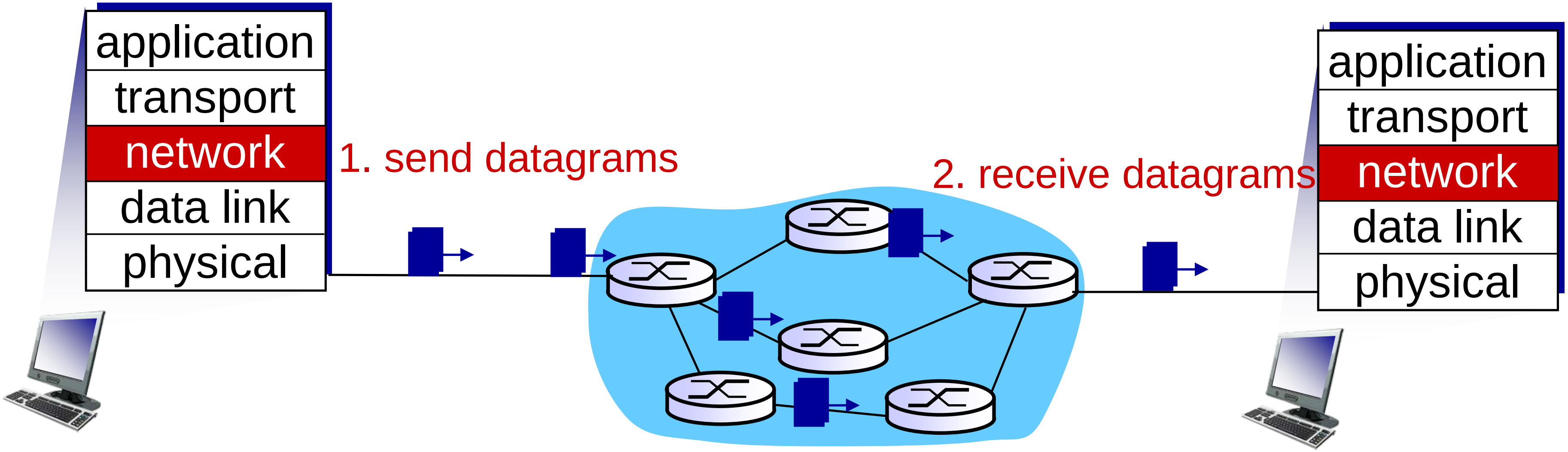
forwarding table in northwest router:

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

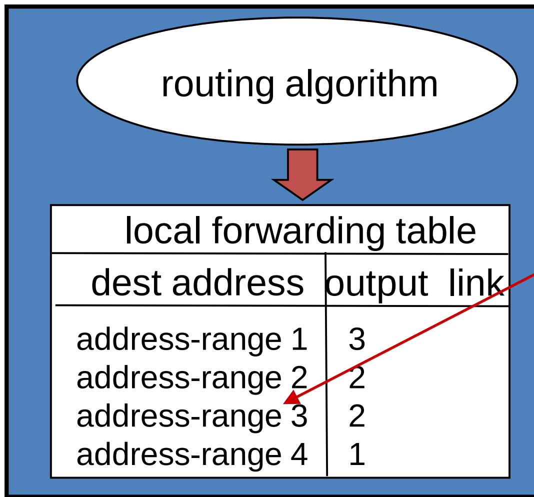
*VC routers must maintain connection **state** information!*

Datagram networks (the Internet model)

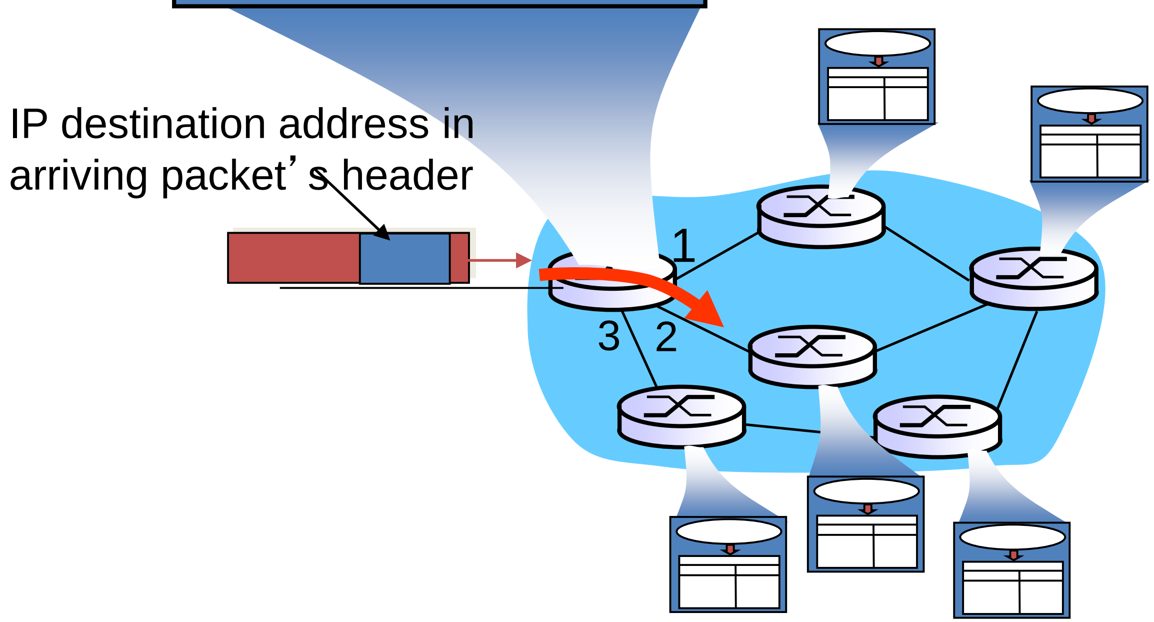
- no call setup at network layer
- routers: no state about end-to-end connections
  - no network-level concept of "connection"
- packets forwarded using destination host address



Datagram forwarding table



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



Datagram or VC network: why?

"Classic" Internet (datagram)

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

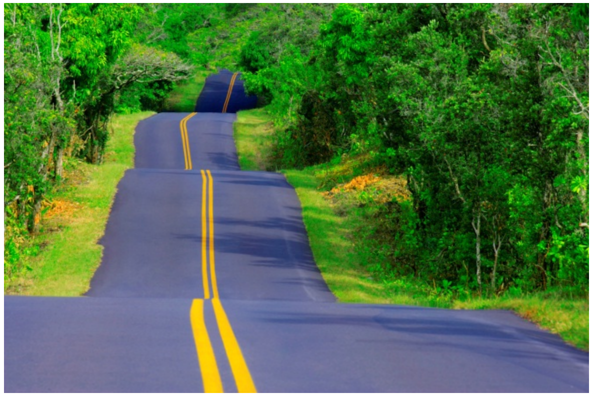
VC (eg ATM: a past's vision of the future's ww-network)

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

Re-shaping in progress ….
Software-Defined Networks

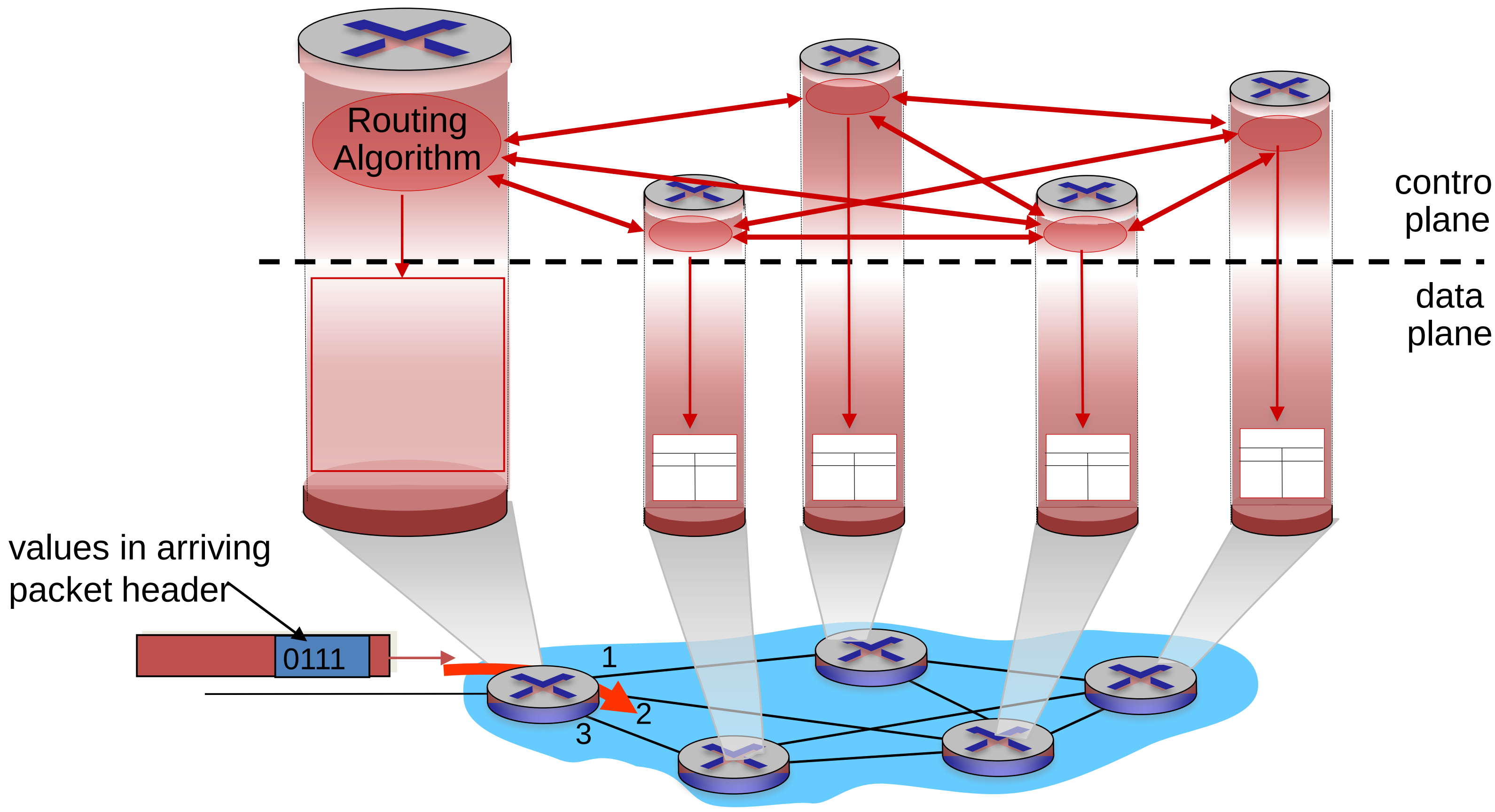
Roadmap Network Layer

- Forwarding versus routing
- Network layer service models
  - Network layer architecture (shift): Software-Defined Networks
- How a router works: switching fabrique
- The Internet Network layer: IP, Addressing & related
- (Next) Control, routing
  - path selection
  - instantiation, implementation in the Internet



Per-router control plane

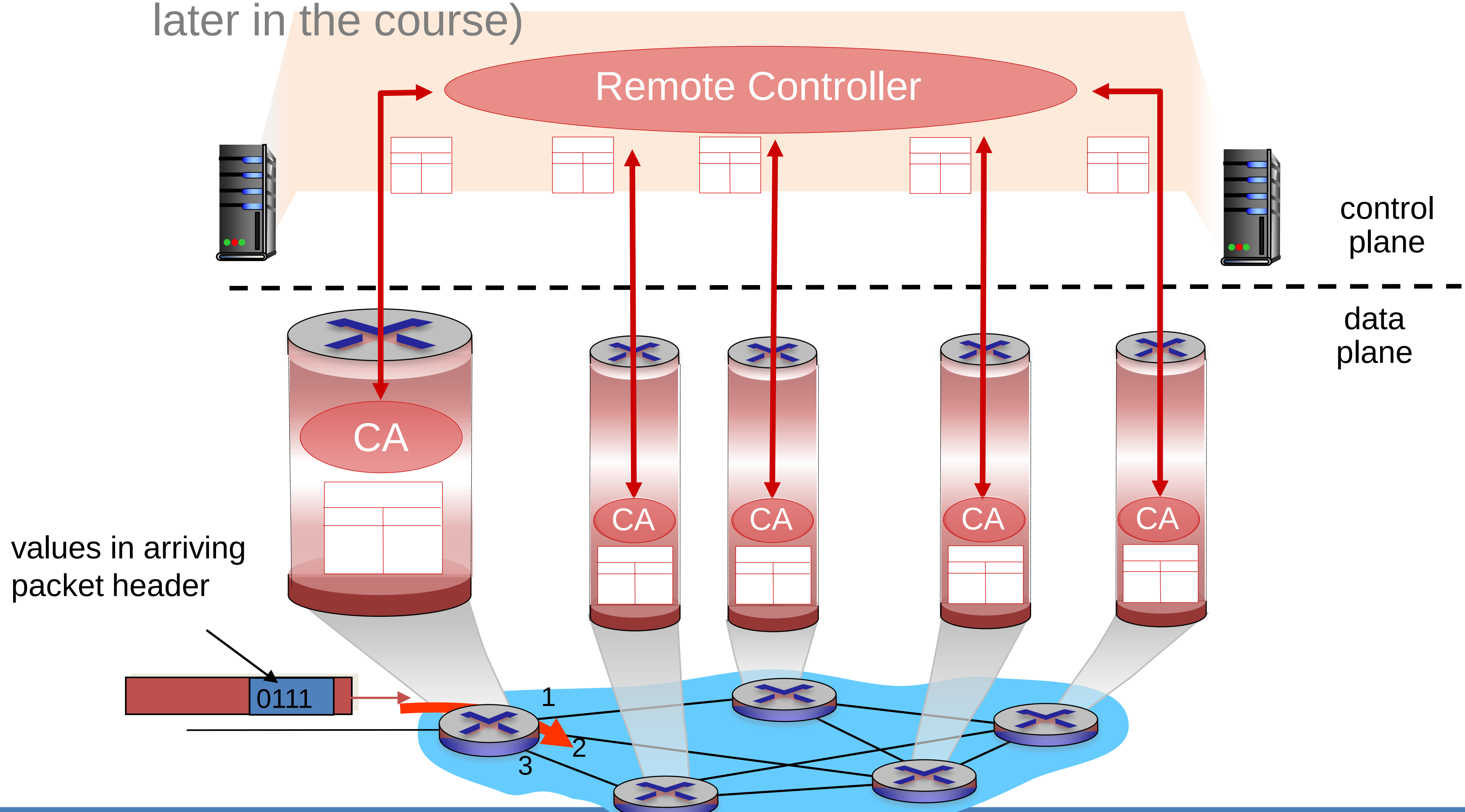
Individual routing algorithm (control) components *in each and every router* interact in the control plane



Logically centralized control plane

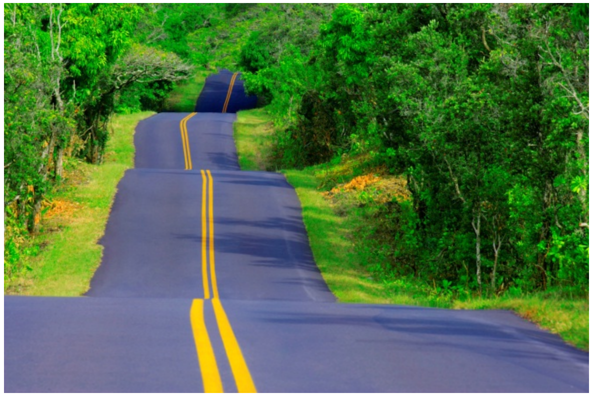
A distinct (can be remote/distributed) controller interacts with local control agents (CAs)

- this architecture (SDN) can enable new functionality (will be studied later in the course)

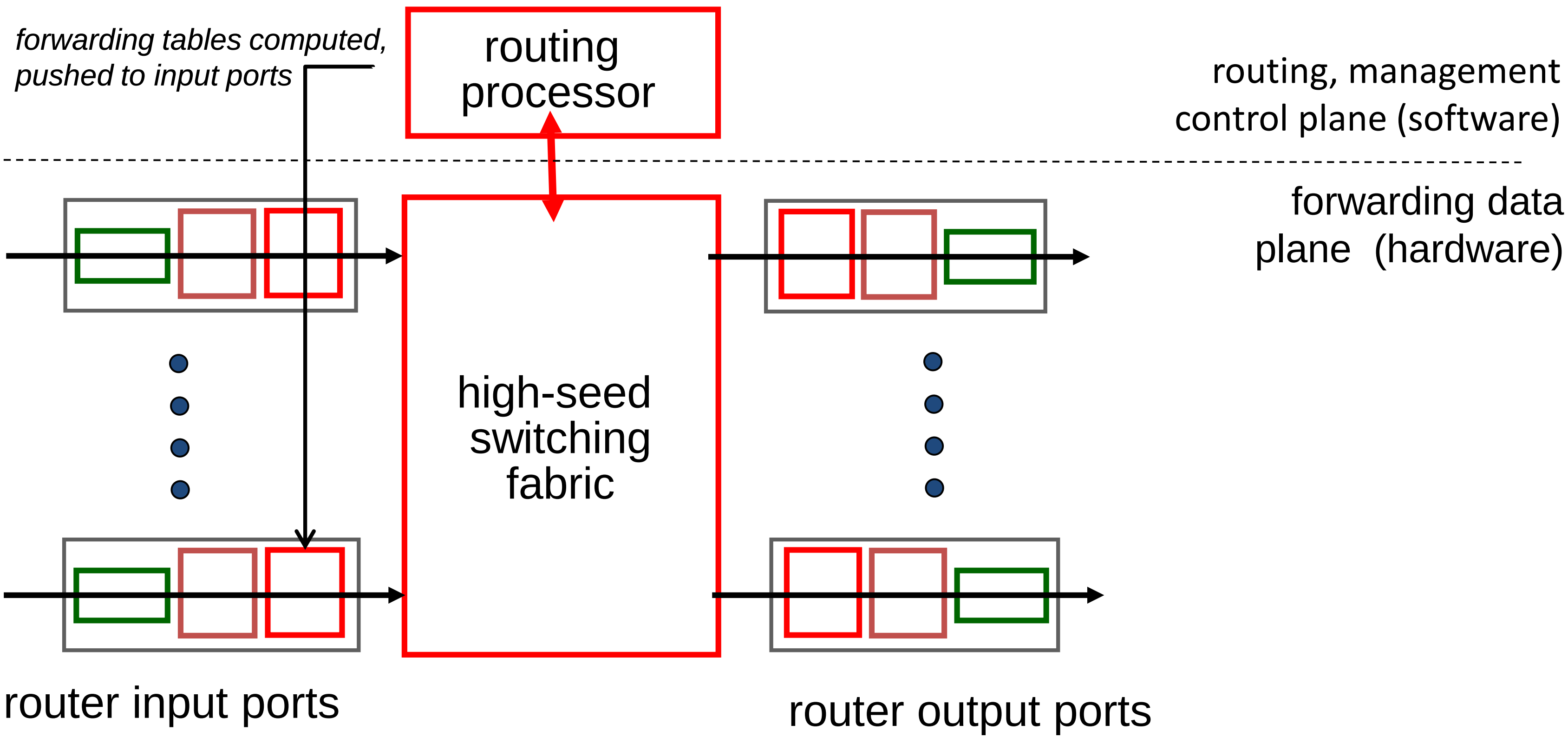


Roadmap Network Layer

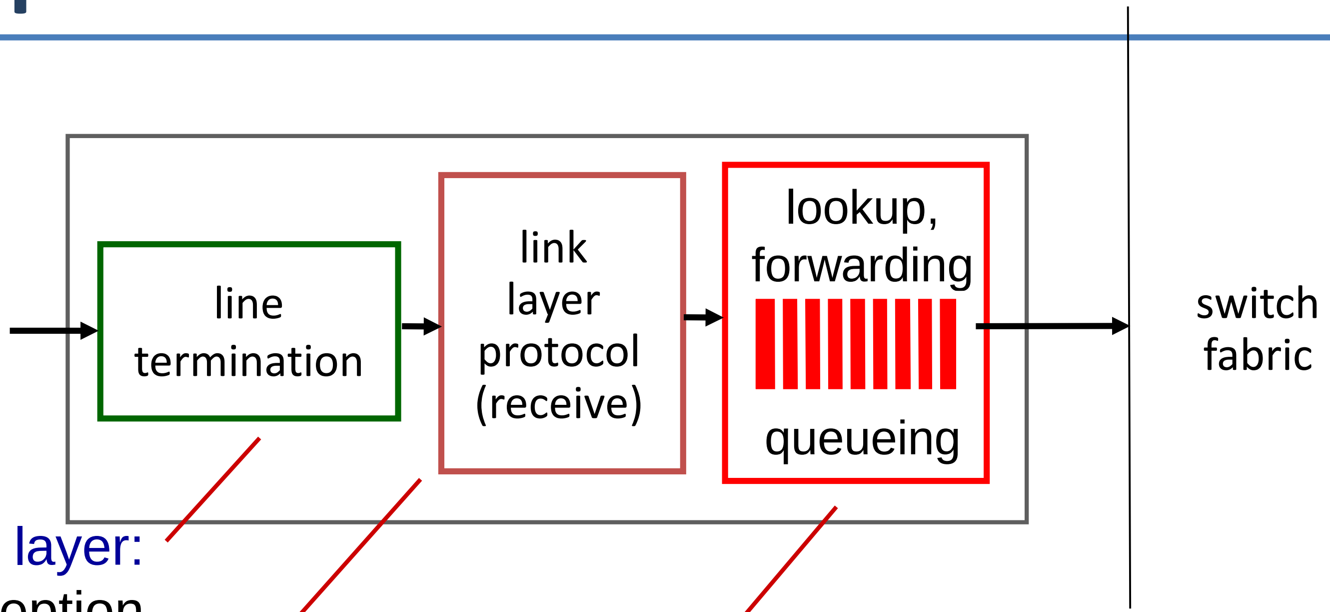
- Forwarding versus routing
- Network layer service models
  - Network layer architecture (shift): Software-Defined Networks
- **Inside a router**
- The Internet Network layer: IP, Addressing & related
- (Next) Control, routing
  - path selection
  - instantiation, implementation in the Internet



Router architecture overview



Input port functions



physical layer:
bit-level reception

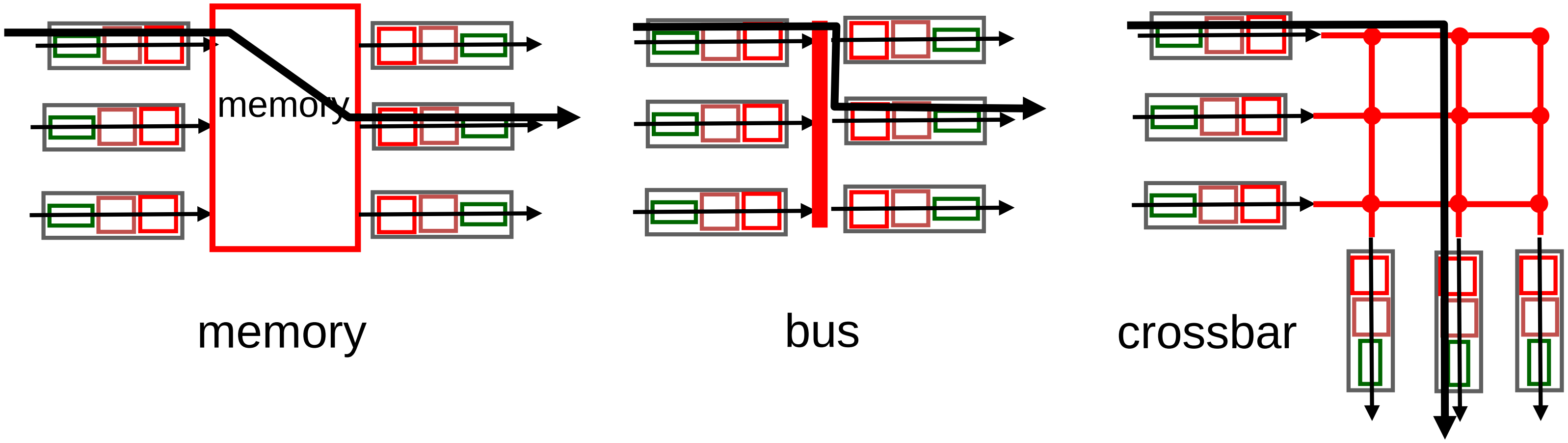
data link layer:
e.g., Ethernet
see chapter 5

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

Switching fabrics

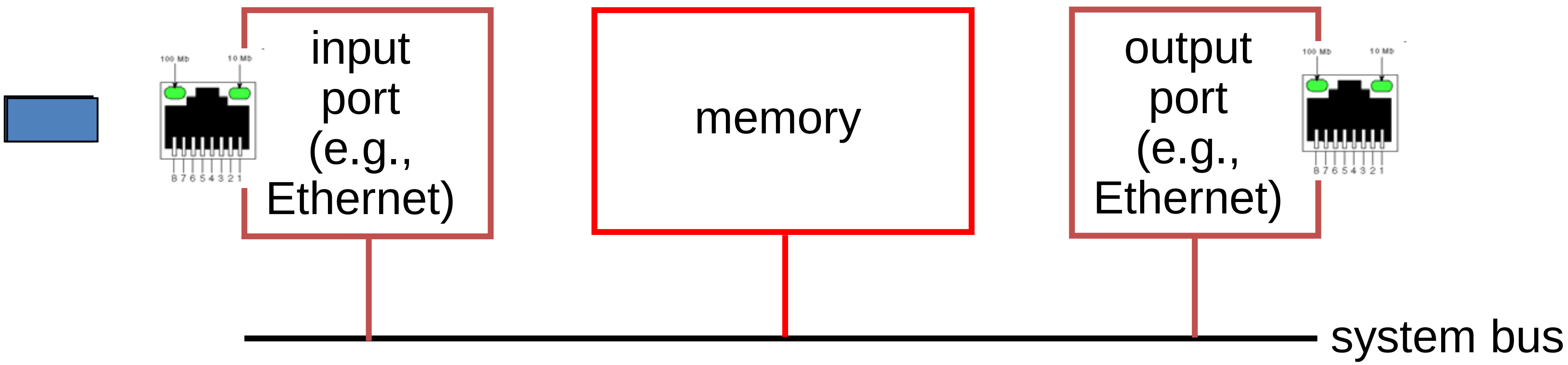
- transfer packet from input buffer to appropriate output buffer
- switching rate: rate at which packets can be transfer from inputs to outputs
  - often measured as multiple of input/output line rate
  - N inputs: switching rate N times line rate desirable
- three types of switching fabrics:



Switching via memory

first generation routers:

- traditional computers with switching under direct control of CPU
- packet copied to system's memory
- speed limited by memory bandwidth (2 bus crossings per datagram)

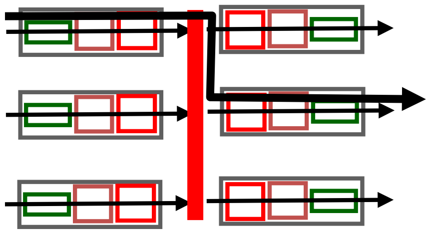


Switching via a bus

datagram from input port memory

to output port memory via a shared bus

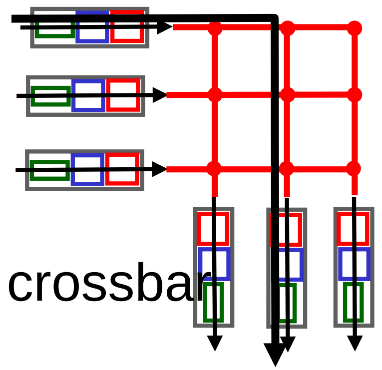
- *bus contention:* switching speed limited by bus bandwidth
- 32 Gbps bus, Cisco 5600: sufficient speed for access and enterprise routers

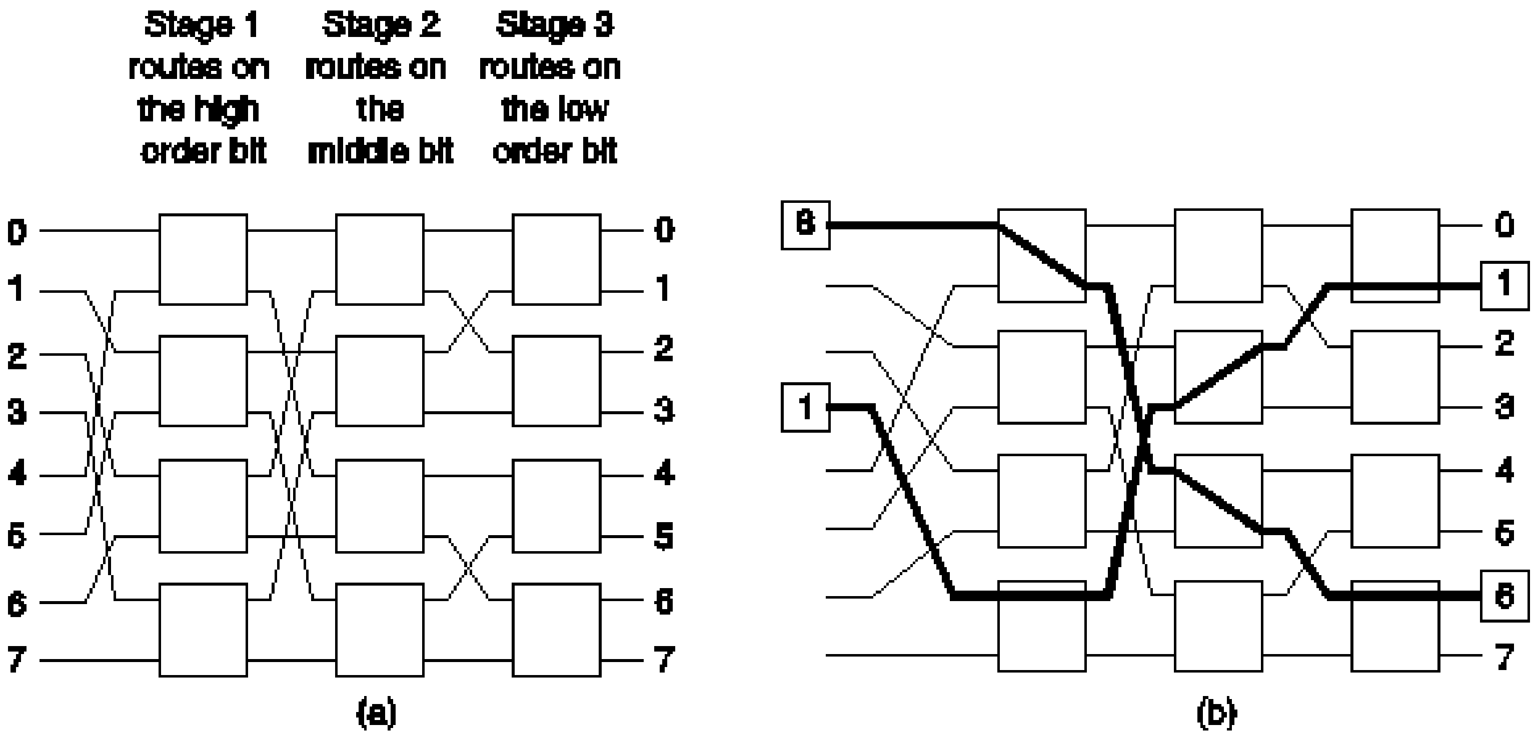


bus

Switching Via an Interconnection Network

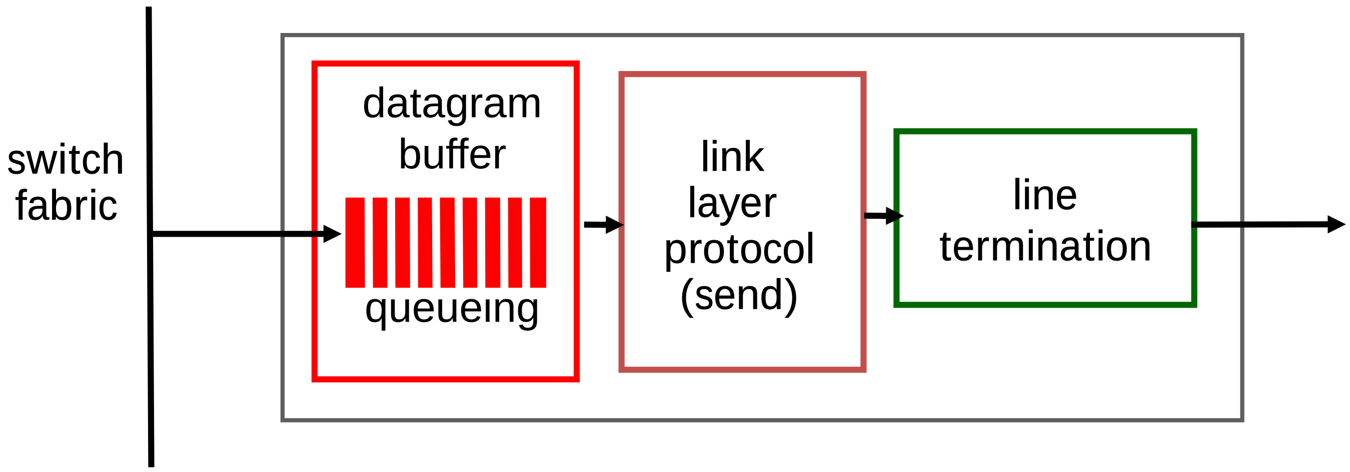
- Overcome bus bandwidth limitations
- Banyan networks, other interconnection nets (also used in processors-memory interconnects in multiprocessors)
  - Cisco 12000: switches at 60 Gbps
  - Example Banyan interconnect: using 3-bit link address





Output ports

This is very important!



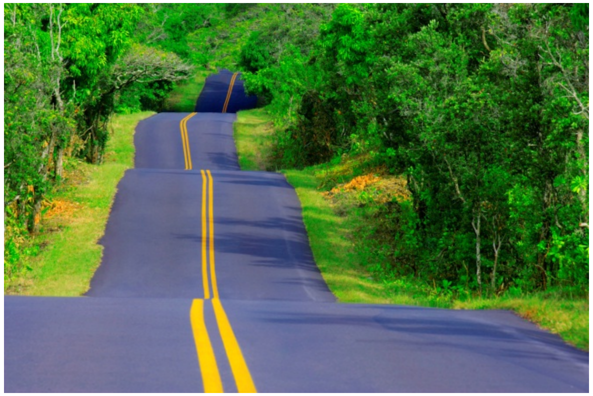
- *buffering* required when datagrams arrive from fabric faster than the transmission rate
- *scheduling discipline* chooses among queued datagrams for transmission

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

Priority scheduling – who gets best performance, network neutrality

Roadmap Network Layer

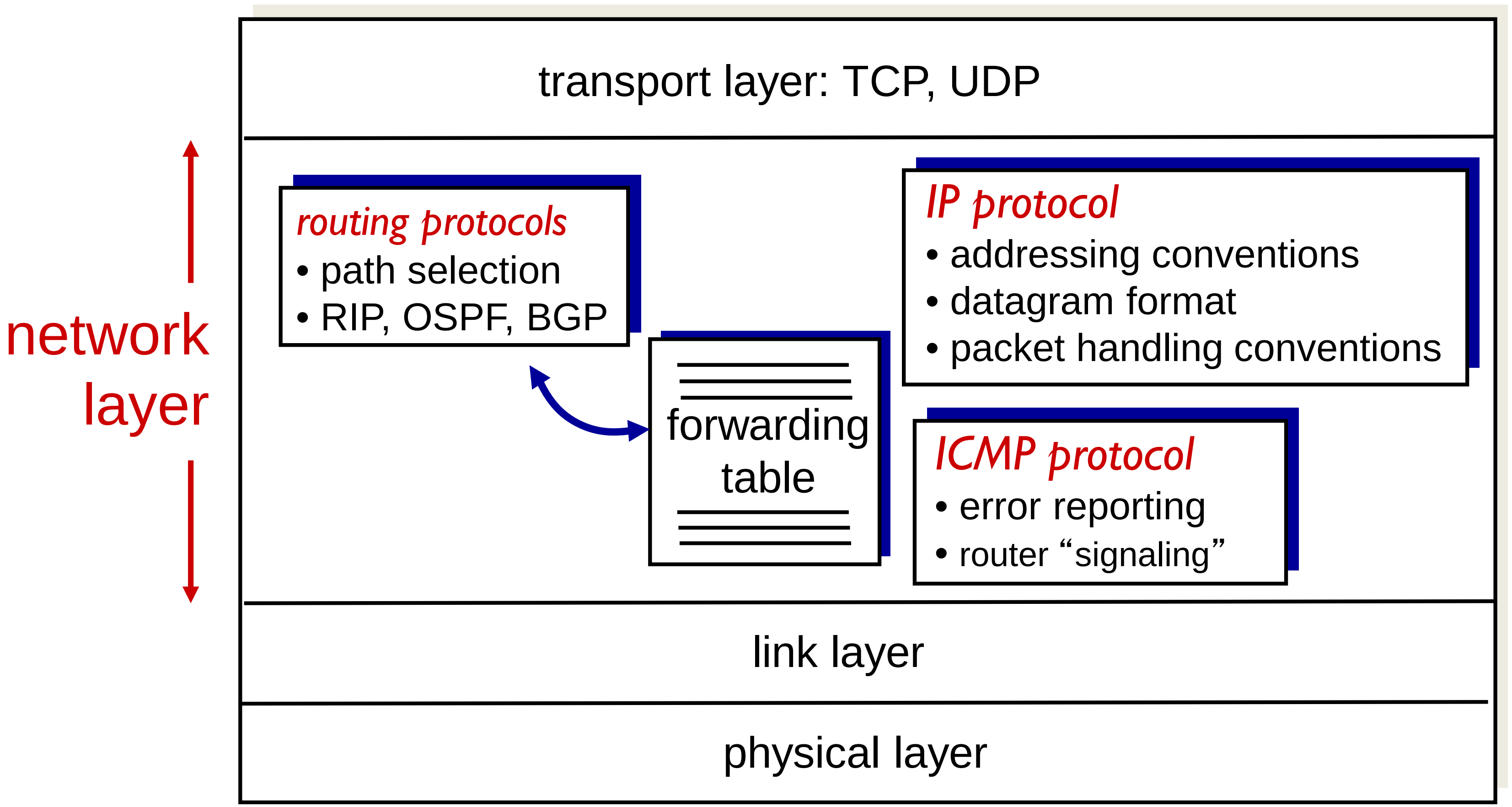
- Forwarding versus routing
- Network layer service models
  - Network layer architecture (shift): Software-Defined Networks
- How a router works
- The Internet Network layer: IP, Addressing & related



- (Next) Control, routing
  - path selection
  - instantiation, implementation in the Internet

The Internet network layer

host, router network layer functions:



IPv4 datagram format

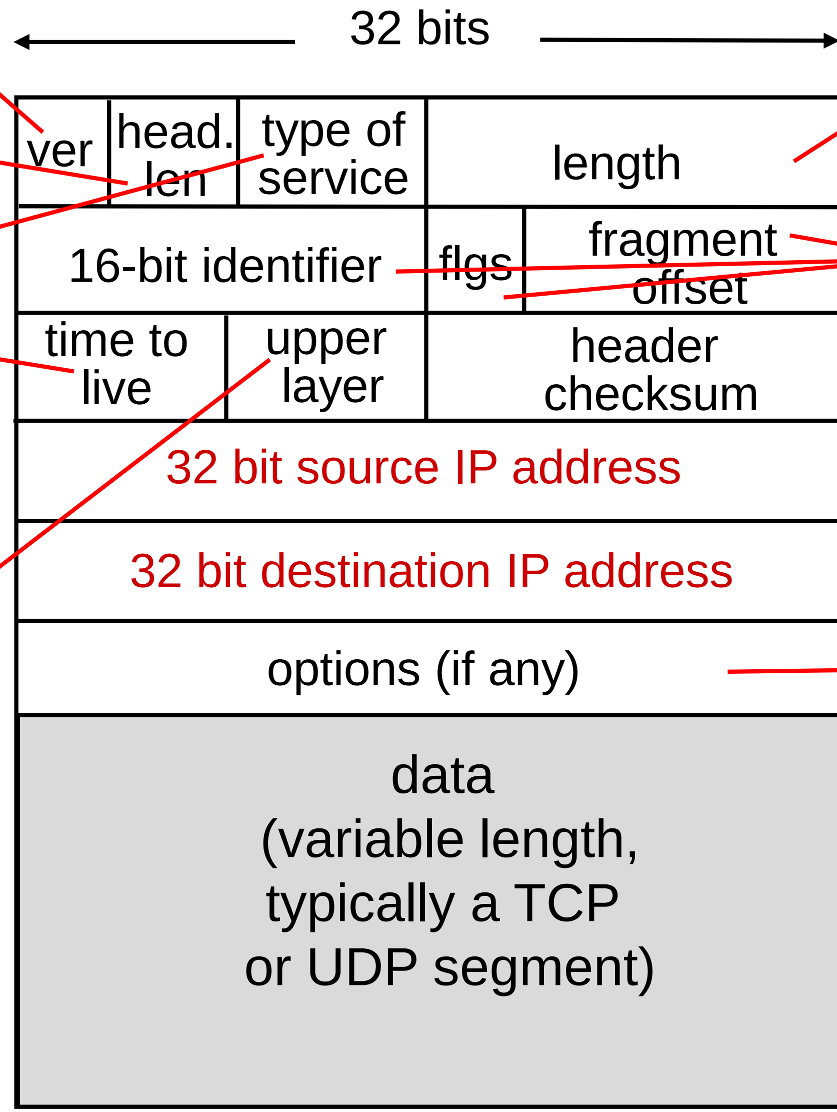
IP protocol version number

header length (bytes)

"type" of data (prio)

max number remaining hops (decremented at each router)

upper layer protocol to deliver payload to



total datagram length (bytes)

for fragmentation/ reassembly

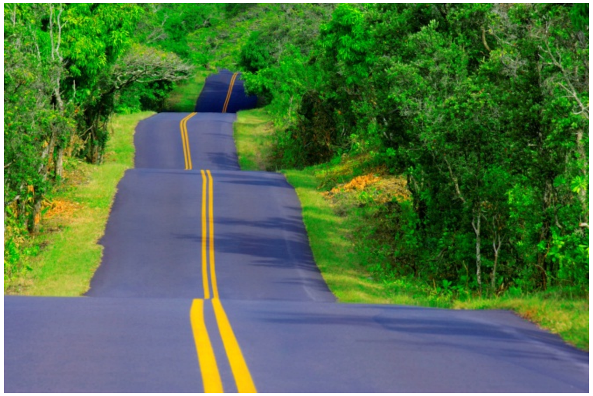
e.g. timestamp, record route taken, specify list of routers to visit.

how much overhead?

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

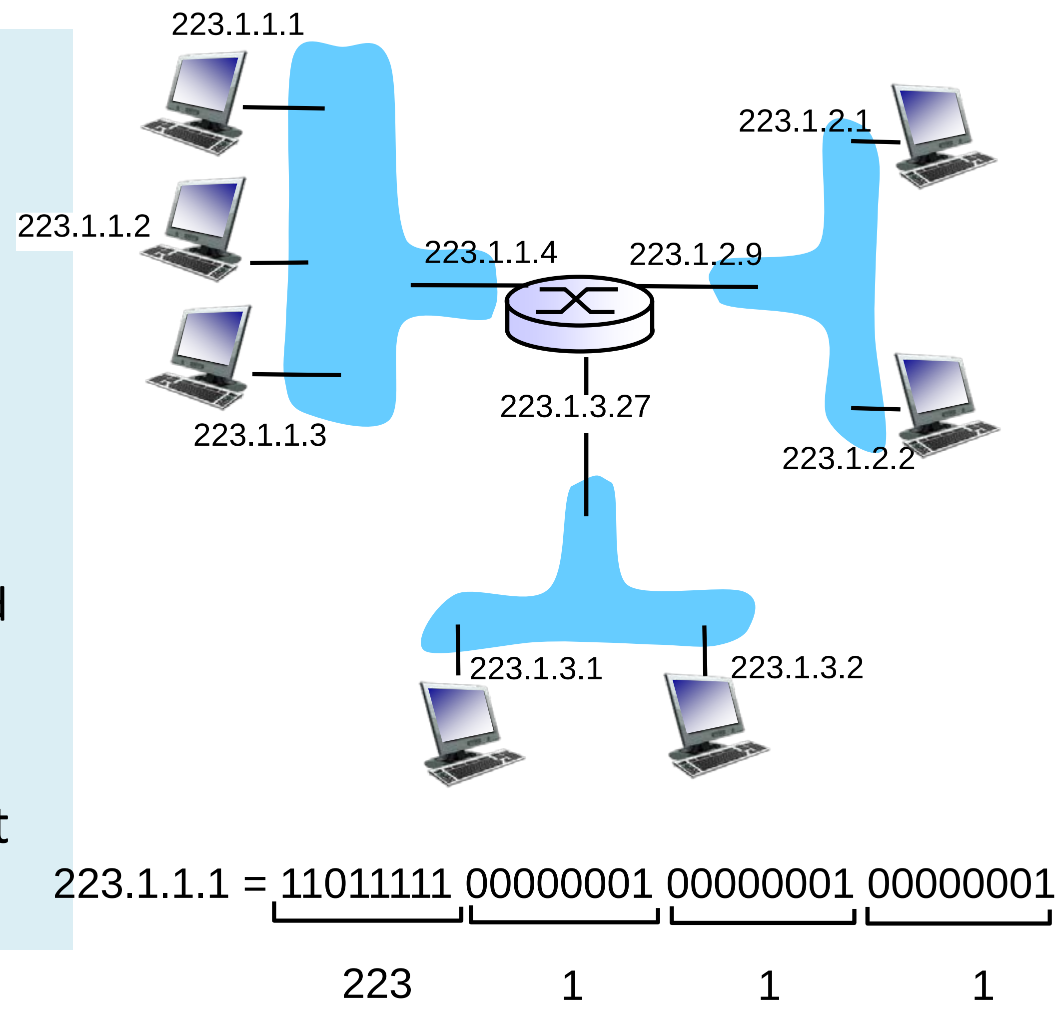
Roadmap Network Layer

- Forwarding versus routing
- Network layer service models
  - Network layer architecture (shift): Software-Defined Networks
- How a router works
- The Internet Network layer: IP, Addressing & related
  - Hierarchical addressing
- (Next) Control, routing
  - path selection
  - instantiation, implementation in the Internet



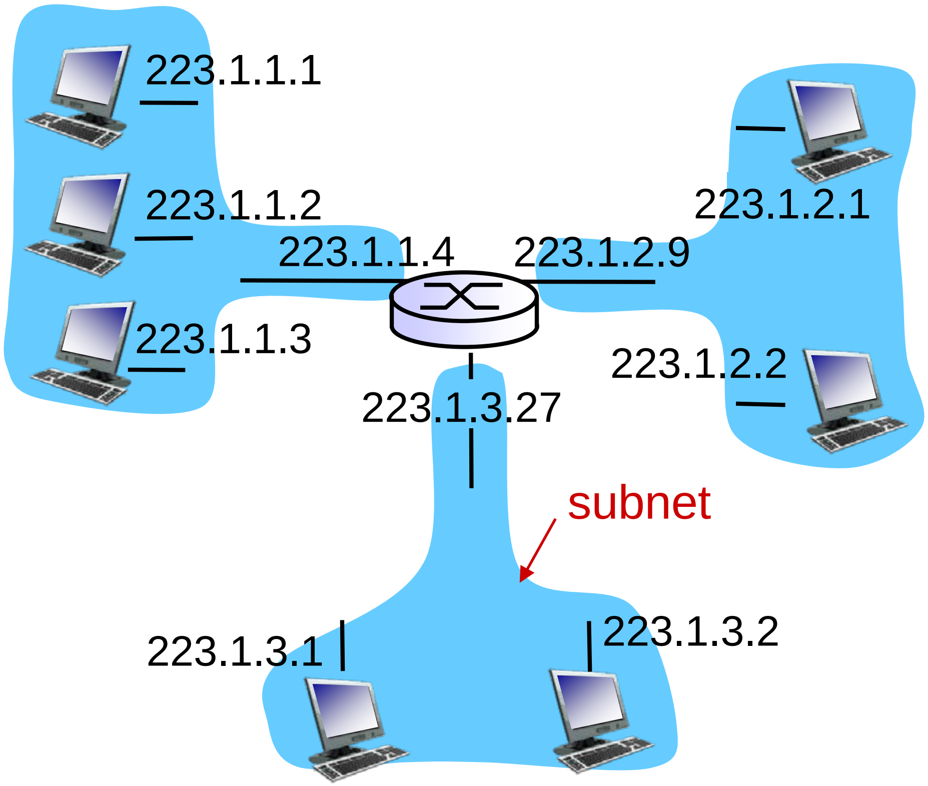
IP addressing: introduction

- *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 and wireless 802.11)
- *IP addresses associated with each interface* (ie not the host)



Subnets

- IP address:
  - –subnet part - high order bits (variable number)
  - –host part - low order bits
- *what's a subnet ?*
  - –device interfaces with same subnet part of IP address
  - –can physically reach each other *without intervening router*

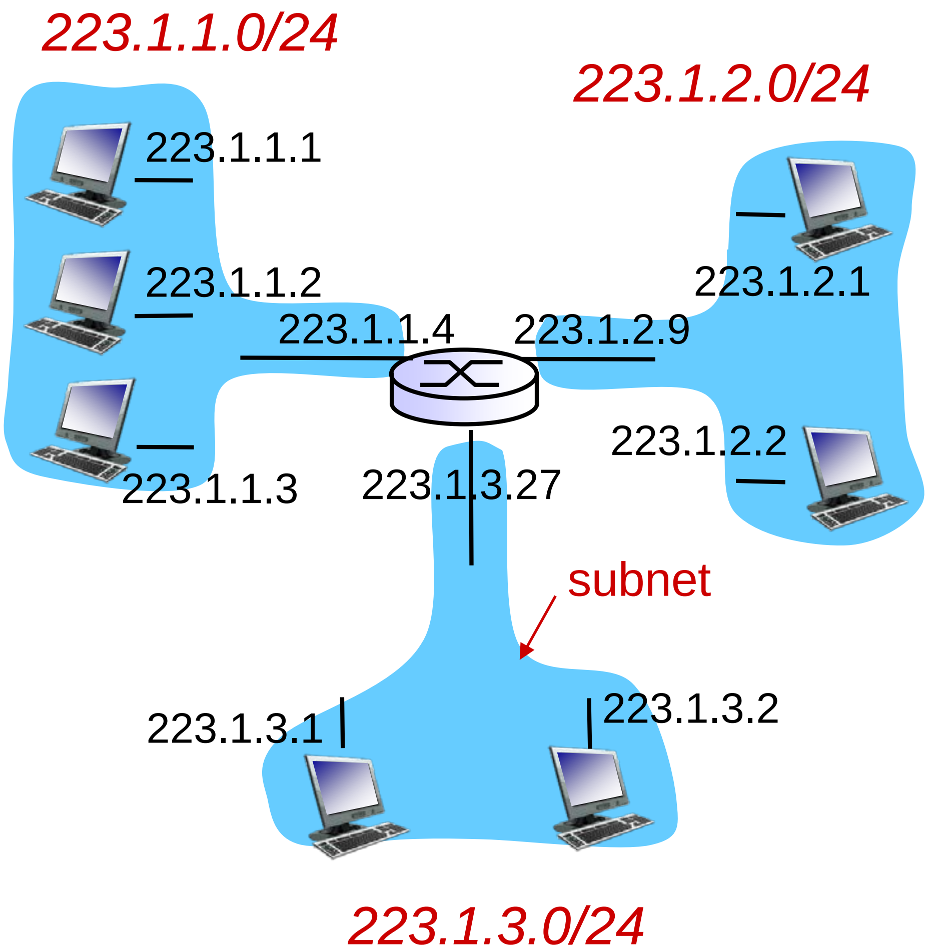


network consisting of 3 subnets

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: eg /24

defines how to find the subnet part of the address …

IP addressing: CIDR

CIDR: Classless InterDomain Routing

- subnet portion of address of arbitrary length
- address format: a.b.c.d/x, where x is # bits in subnet portion of address

subnet part ← → host part

11001000 00010111 00010000 00000000

200.23.16.0/23

Subnets, masks, calculations

Example subnet: 192.168.5.0/24

	Binary form	Dot-decimal notation
IP address	11000000.10101000.00000101.10000010	192.168.5.130
Subnet mask	11111111.11111111.11111111.00000000 --------24 first bits set to 1------	255.255.255.0
Network prefix: *(bitwise AND of address, mask)*	11000000.10101000.00000101.00000000	192.168.5.0
Host part (obtained with similar calculation, with a "mask" where the 32 – 24 last bits set to 1)	00000000.00000000.00000000.10000010	0.0.0.130

CIDR Address Masks

CIDR Notation	Dotted Decimal	CIDR Notation	Dotted Decimal
/1	128.0.0.0	/17	255.255.128.0
/2	192.0.0.0	/18	255.255.192.0
/3	224.0.0.0	/19	255.255.224.0
/4	240.0.0.0	/20	255.255.240.0
/5	248.0.0.0	/21	255.255.248.0
/6	252.0.0.0	/22	255.255.252.0
/7	254.0.0.0	/23	255.255.254.0
/8	255.0.0.0	/24	255.255.255.0
/9	255.128.0.0	/25	255.255.255.128
/10	255.192.0.0	/26	255.255.255.192
/11	255.224.0.0	/27	255.255.255.224
/12	255.240.0.0	/28	255.255.255.240
/13	255.248.0.0	/29	255.255.255.248
/14	255.252.0.0	/30	255.255.255.252
/15	255.254.0.0	/31	255.255.255.254
/16	255.255.0.0	/32	255.255.255.255

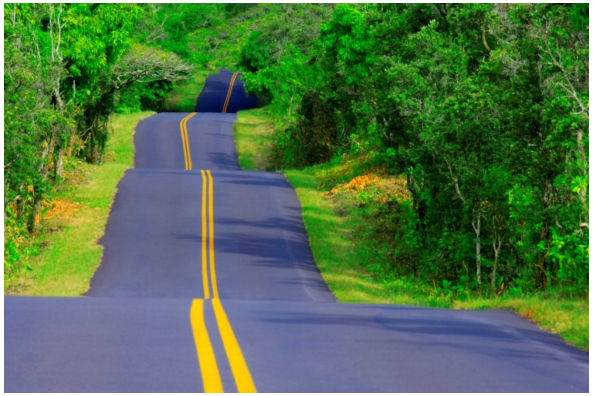
Classless Address: example

- An ISP has an address block 122.211.0.0/16
- A customer needs max. 6 host addresses,
- ISP can e.g. allocate: 122.211.**176.208**/29
  - 3 bits enough for host part
- subnet mask 255.255.**255.248**

	Dotted Decimal	Last 8 bits
Network	122.211.176.**208**	**11010**000
1st address	122.211.176.**209**	**11010**001
………….	………………….	…………
6th address	122.211.176.**214**	**11010**110
Broadcast	122.211.176.**215**	**11010**111

Roadmap Network Layer

- Forwarding versus routing
- Network layer service models
  - Network layer architecture (shift): Software-Defined Networks
- How a router works
- The Internet Network layer: IP, Addressing & related
  - Hierarchical addressing
  - How to get addresses



- (Next) Control, routing
  - path selection
  - instantiation, implementation in the Internet

IP addresses: how to get one (for an end-host)?

- 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:
  - 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: more than an IP address

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)

IP addresses: how to get one (net-part)?

Q: how does *network* get subnet part of IP addr?

A: gets allocated portion of its provider ISP's address space; eg:

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**

3 bits, 8 networks

IP Addressing: the last word...

Q: How does an ISP get block of addresses?

A: **ICANN**: http://www.icann.org/

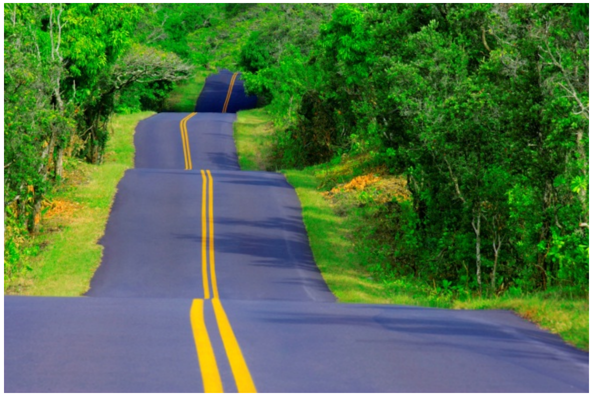
Internet **C**orporation for **A**ssigned **N**ames and **N**umbers

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

Users are assigned IP addresses by Internet Service Providers (ISPs). ISPs obtain allocations of IP addresses from a Local Internet Registry (LIR) or National Internet Registry (NIR), or from their appropriate Regional Internet Registry (RIR, 5 worldwide).

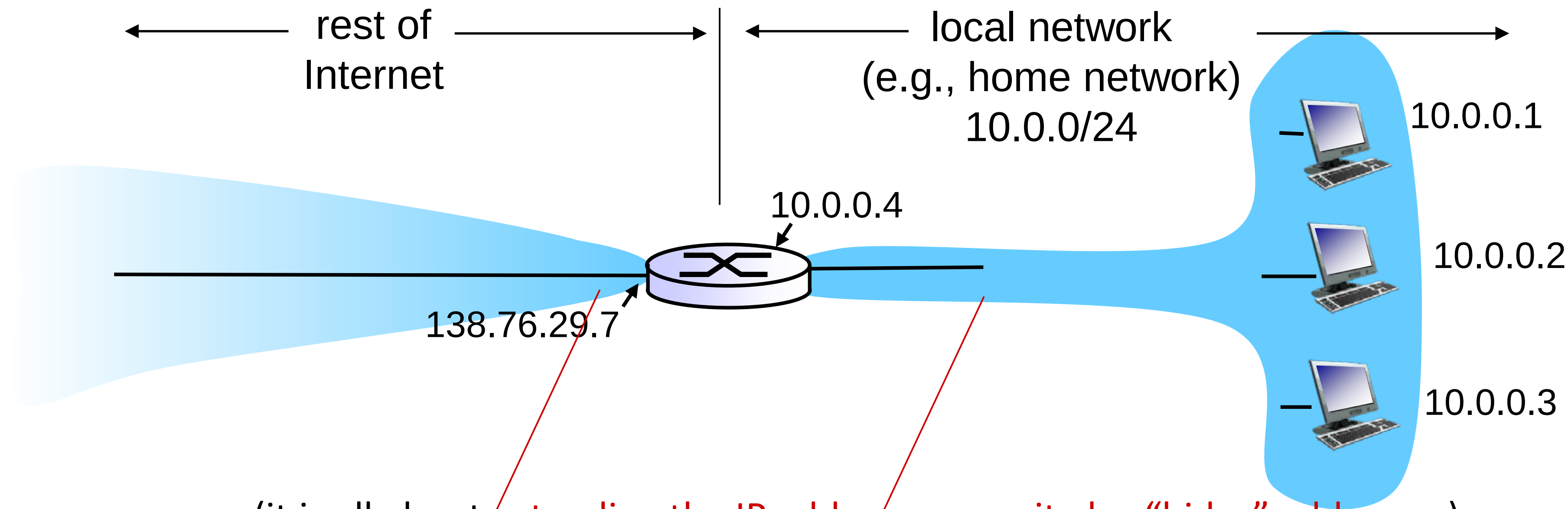
Roadmap Network Layer

- Forwarding versus routing
- Network layer service models
  - Network layer architecture (shift): Software-Defined Networks
- How a router works
- The Internet Network layer: IP, Addressing & related
  - Hierarchical addressing
  - How to get addresses
  - NAT
- (Next) Control, routing
  - path selection
  - instantiation, implementation in the Internet



(Well, it was not really the last word…)

NAT: network address translation

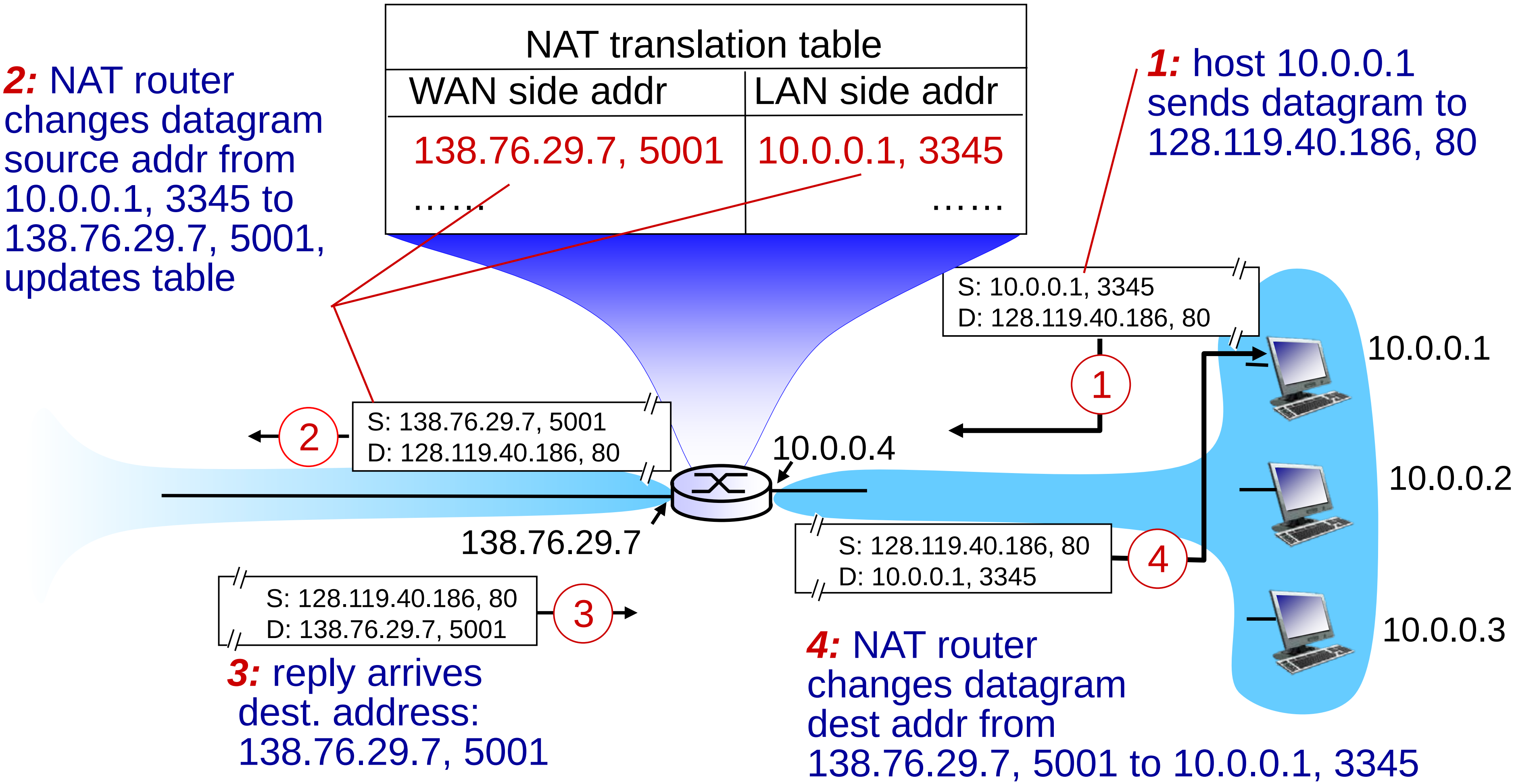


(it is all about extending the IP address space; it also "hides" addresses)

all datagrams *leaving* local network have *same* single source NAT IP address: 138.76.29.7,different source port numbers

datagrams with source or destination in this network have 10.0.0/24 address for source, destination (as usual)

NAT: network address translation

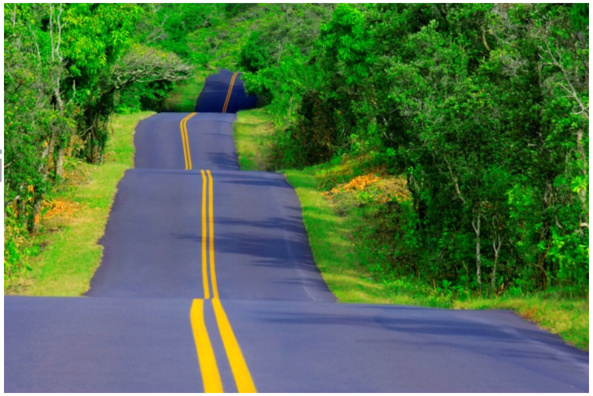


NAT: network address translation

- 16-bit port-number field:
  - 64k simultaneous connections with a single LAN-side address!
- NAT is controversial:
  - routers should in principle process up to layer 3
  - violates end-to-end argument
    - NAT possibility must be taken into account by app designers, e.g., P2P applications
  - address shortage should instead be solved by **IPv6**

Roadmap Network Layer

- Forwarding versus routing
- Network layer service models
  - Network layer architecture (shift): Software-Defi Networks
- How a router works
- The Internet Network layer: IP, Addressing & related
  - Hierarchical addressing
  - How to get addresses
  - NAT
  - IPv6
- (Next) Control, routing
  - path selection
  - instantiation, implementation in the Internet



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:

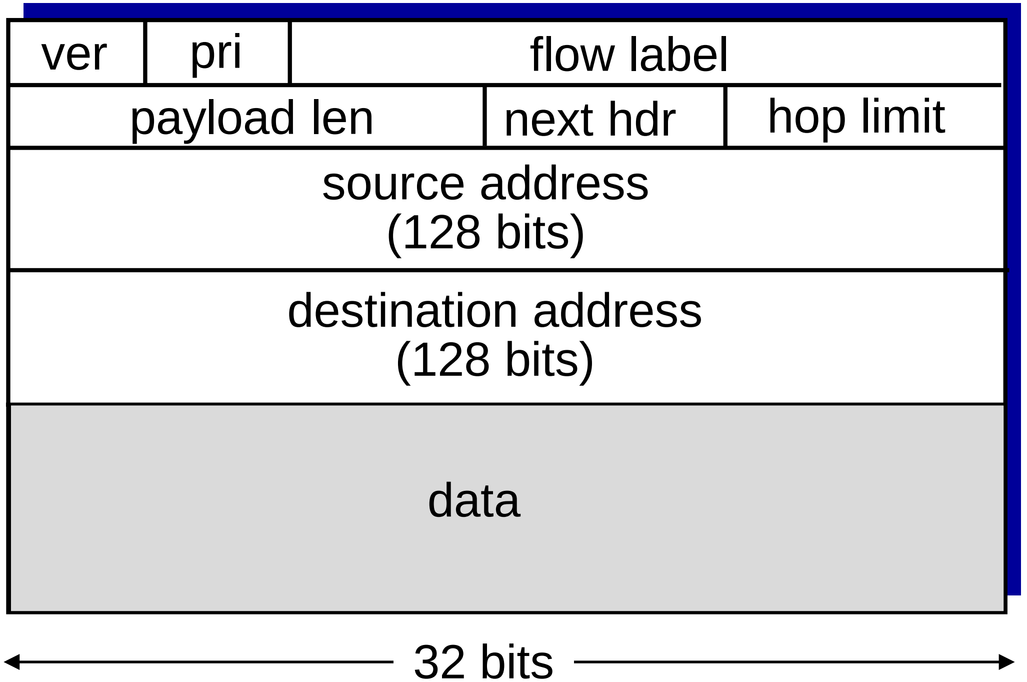
- fixed-length 40 byte header
- no fragmentation allowed
- *128-bit addresses* ($2^{128} = 10^{38}$ hosts)
- Standard subnet size: 2^{64} hosts

IPv6 datagram format

priority: identify priority among datagrams in flow
flow Label: identify datagrams in same "flow."
(concept of "flow" not well defined).

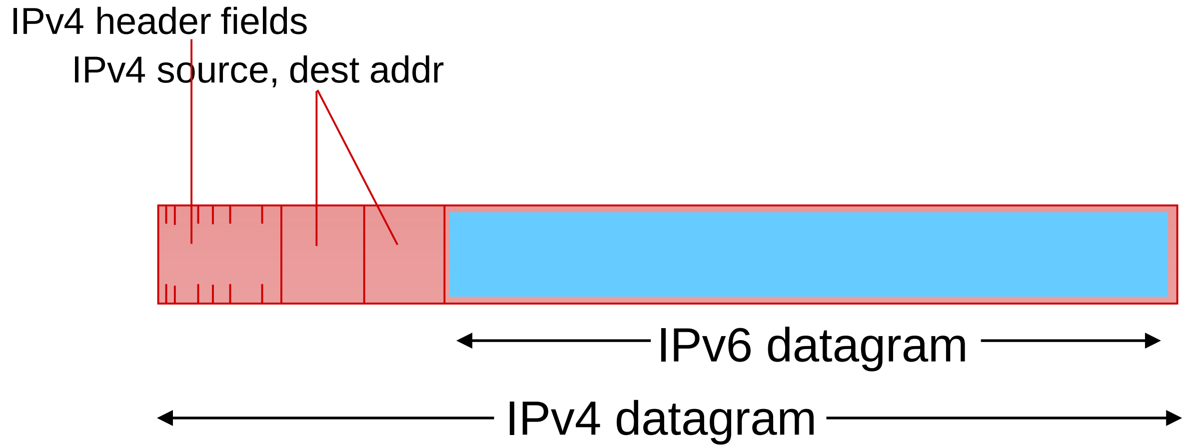
checksum: removed entirely to reduce processing time at each hop

options: allowed, but outside of header, indicated by "Next Header" field

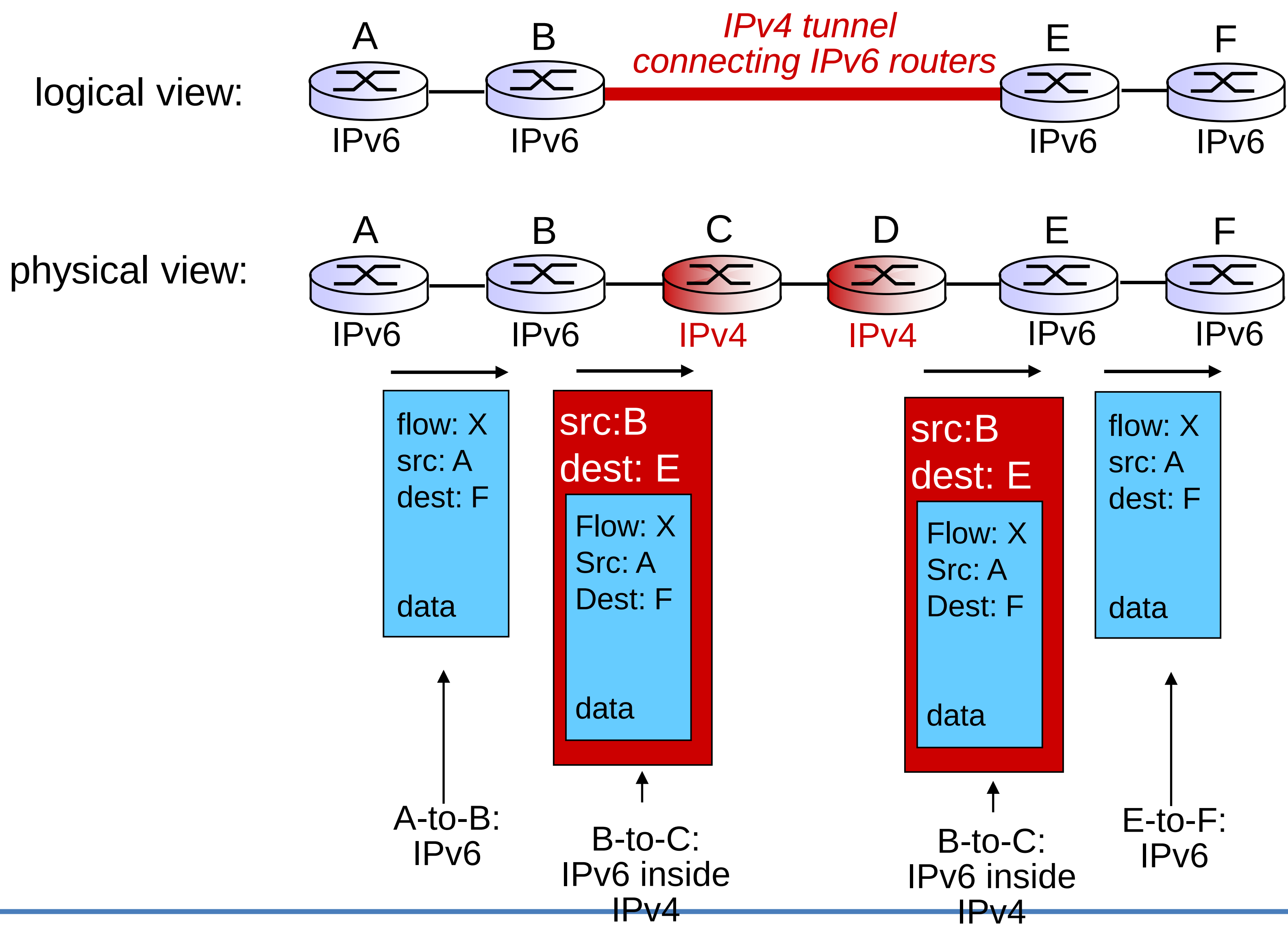


Transition from IPv4 to IPv6

- not all routers can be upgraded simultaneously
  - how will network operate with mixed IPv4 and IPv6 routers?
- *tunneling:* IPv6 datagram carried as *payload* in IPv4 datagram among IPv4 routers



Tunneling (6in4 – static tunnel)



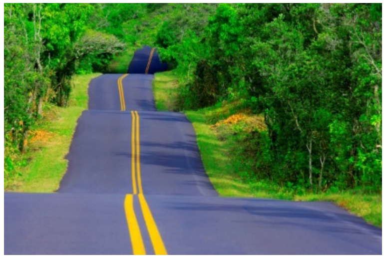
IPv6: adoption

- Google: 8% of clients access services via IPv6
- NIST: 1/3 of all US government domains are IPv6 capable

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

Roadmap Network Layer

- Forwarding versus routing
- Network layer service models
  - Network layer architecture (shift): Software-Defined Networks
- How a router works
- The Internet Network layer: IP, Addressing & related
  - Hierarchical addressing
  - How to get addresses
  - NAT
  - IPv6
- (Next) Control, routing
  - path selection
  - instantiation, implementation in the Internet



Reading instructions Network Layer
(incl. Next lecture)

- **KuroseRoss book**

Careful	**Quick**
5/e,6/e: 4.1-4.6 7/e: 4.1-4.3, 5.2-5.4, 5.5, 5.6, *[new- SDN, data and control plane 4.4, 5.5: in subsequent lectures, connecting to multimedia/streaming Study material through the pingpong-system]*	5/e,6/e: 4.7, 7/e: 5.7

Review questions for this part

- network layer **service models**

tagram routing (simplicity, cost,
they may enable)

ı routing and forwarding

'where do queueing delays happen
an packets be dropped at a router?

ıasking?

ations

om source to destination

Some complementary material /video-links

- IP addresses and subnets
  http://www.youtube.com/watch?v=ZTJIkjgyuZE&list=PLE9F3F05C381ED8E8&feature=plcp

- How does PGP choose its routes
  http://www.youtube.com/watch?v=RGe0qt9Wz4U&feature=plcp

Some taste of layer 2: no worries if not all details fall in place, need the lectures also to grasp them.

- Hubs, switches, routers
  http://www.youtube.com/watch?v=reXS_e3fTAk&feature=related
-
- What is a broadcast + MAC address
  http://www.youtube.com/watch?v=BmZNcjLtmwo&feature=plcp

- Broadcast domains:
  http://www.youtube.com/watch?v=EhJO1TCQX5I&feature=plcp

Extra slides

Network layer service models:

Network Architecture	Service Model	Guarantees ? Bandwidth	Loss	Order	Timing	Congestion feedback
Internet	best effort	none	no	no	no	no (inferred via loss)
ATM	CBR	constant rate	yes	yes	yes	no congestion
ATM	VBR	guaranteed rate	yes	yes	yes	no congestion
ATM	ABR	guaranteed minimum	no	yes	no	yes
ATM	UBR	none	no	yes	no	no

- Internet model being extented: Intserv, Diffserv
  - (will study these later on)

VC implementation

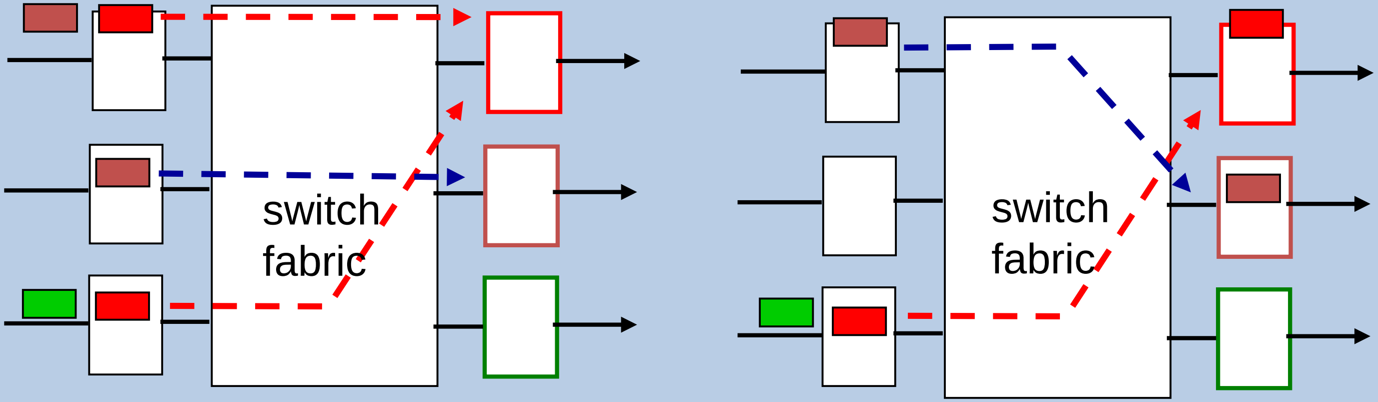
a VC consists of:

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

Input port queuing

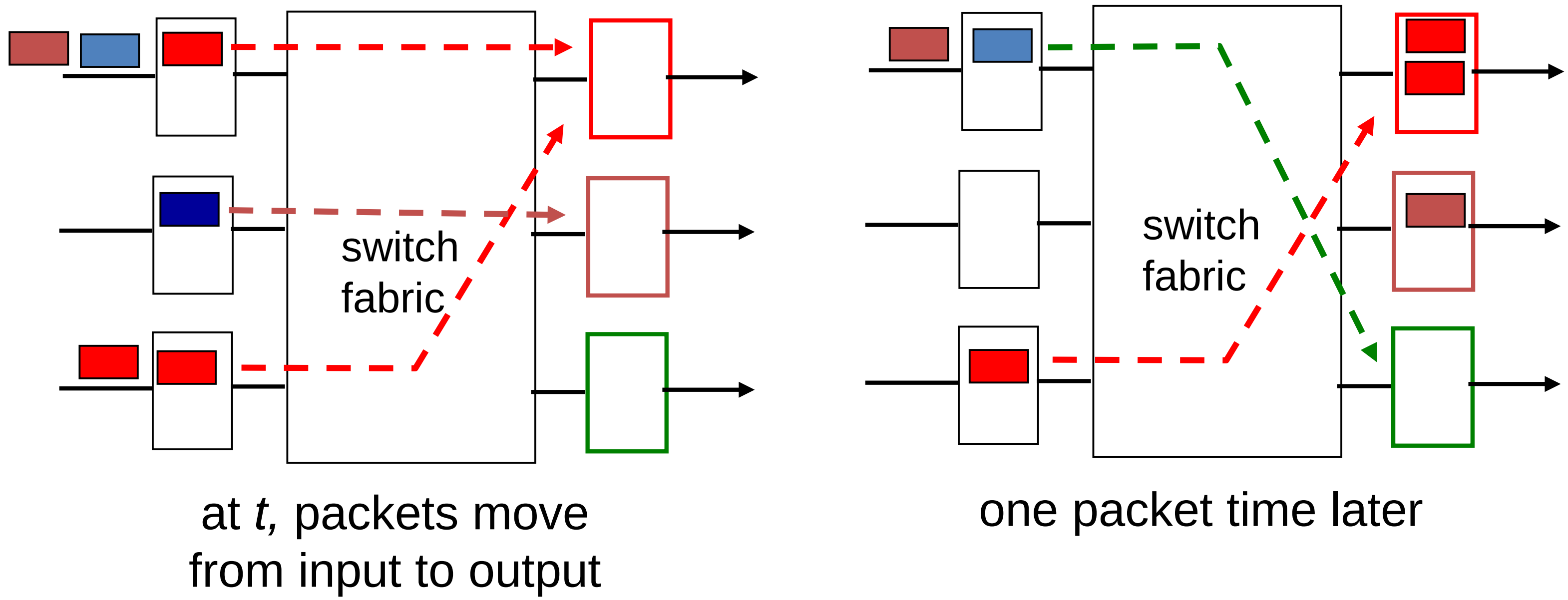
- fabric slower than input ports combined -> queueing may occur at input queues
  - *queueing delay and loss due to input buffer overflow!*
- Head-of-the-Line (HOL) blocking: queued datagram at front of queue prevents others in queue from moving forward



output port contention:
only one red datagram can be transferred.
lower red packet is blocked

one packet time later:
green packet experiences HOL blocking

Output port queueing



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

Example contemporary routers

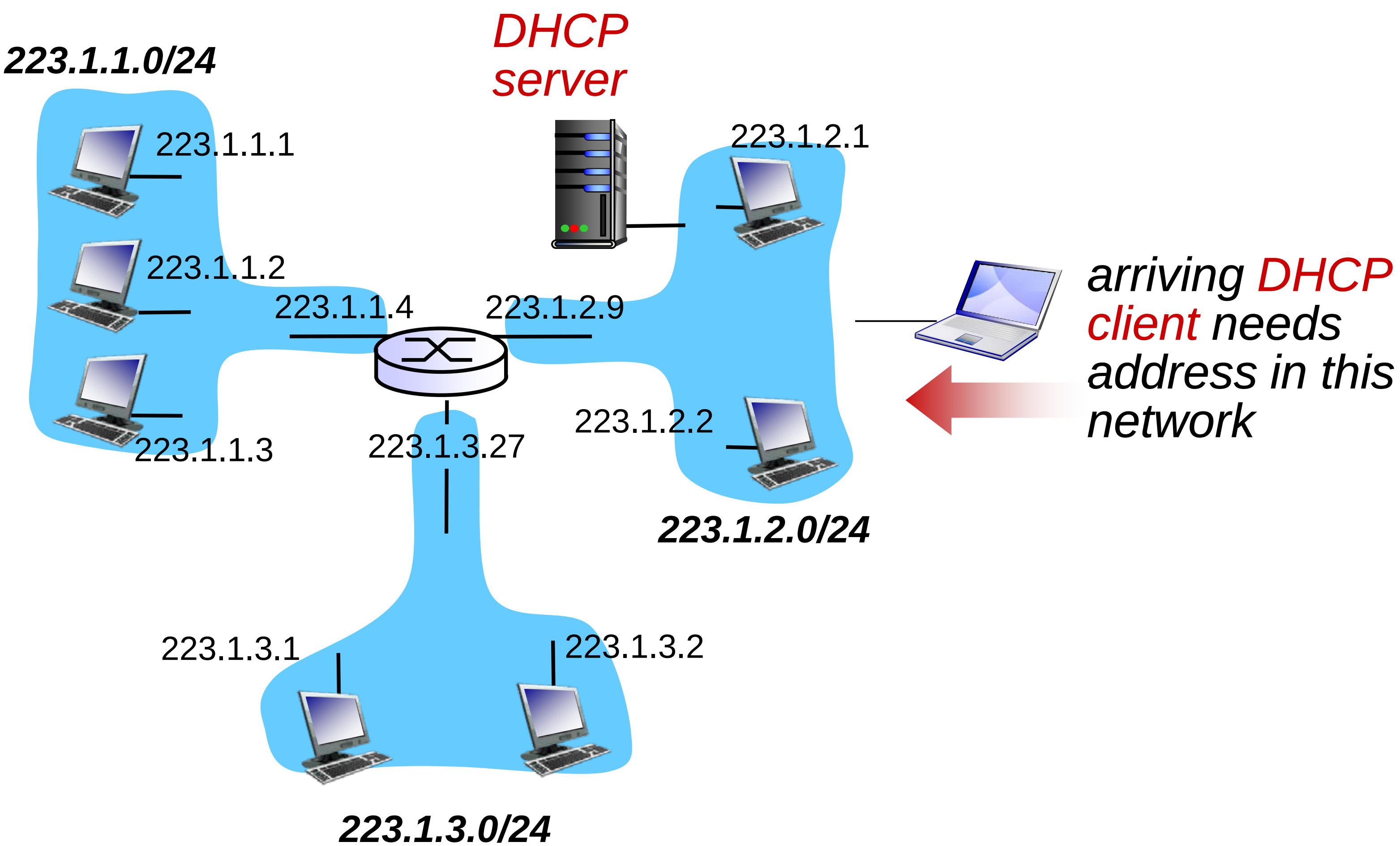
Cisco Catalyst 3750E
Stackable (can combine units)
1 Gbit/s ports
64 Gbit/s bandwidth
13 Mpps (packets per second)
12,000 address entries

Price: from 100 kSEK

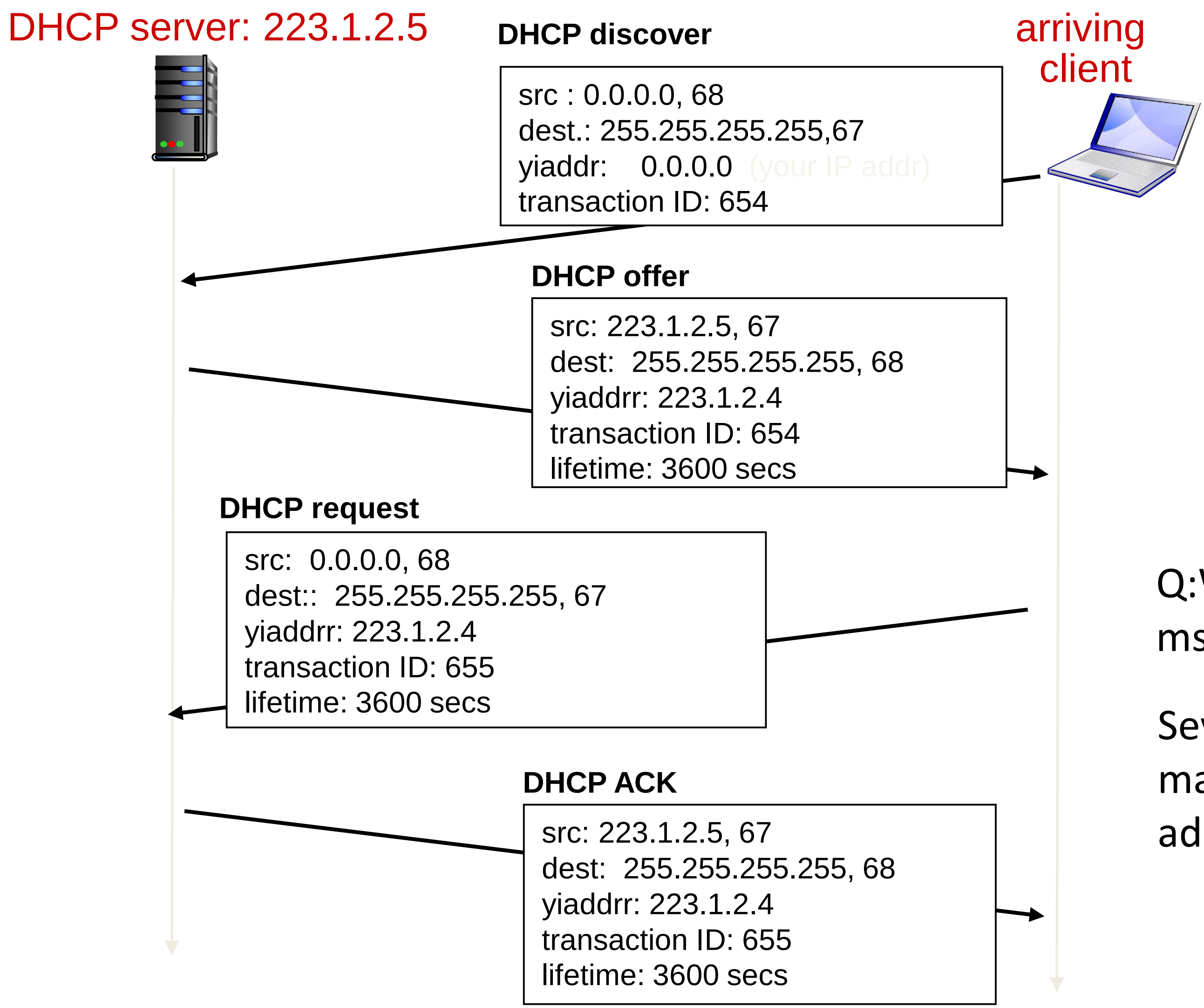
HP ProCurve 6600-24G-4XG Switch
1 Gbit/s, 10 Gbps
Up to 75 Mpps (64-byte packets)
Latency: < 2.4 µs (FIFO 64-byte packets)
10,000 entries

Price approx. 50 kSEK

DHCP client-server scenario



DHCP client-server scenario

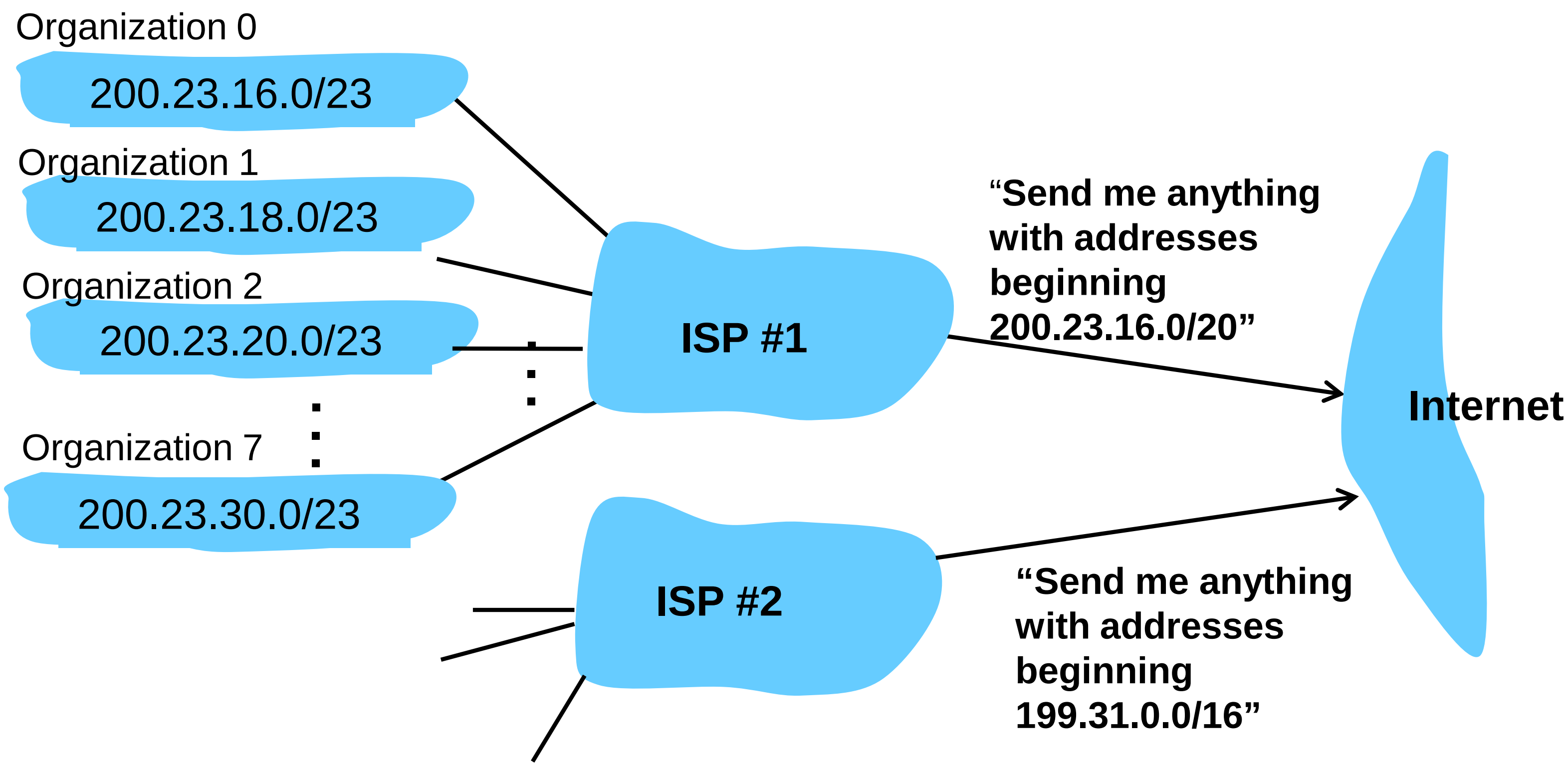


Q:Why a request msg?

Several DHCP servers may answer and offer addresses

Hierarchical Addressing: Route Aggregation

- Hierarchical addressing allows efficient advertisement of routing information
- The “outside” does not need to know about subnets.



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?

NAT: network address translation

motivation: local network uses just one IP address as far as outside world is concerned:

- range of addresses not needed from ISP: just one IP address for all devices
- can change addresses of devices in local network without notifying outside world
- can change ISP without changing addresses of devices in local network
- devices inside local net not explicitly addressable, visible by outside world (a security plus)

NAT: network address translation

implementation: NAT router must:

outgoing datagrams: replace (source IP address, port #) of every outgoing datagram to (NAT IP address, new port #)

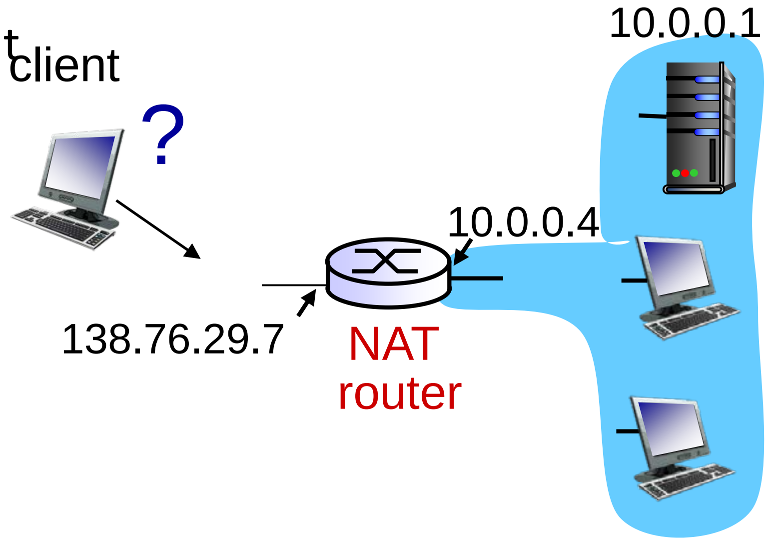
. . . remote clients/servers will respond using (NAT IP address, new port #) as destination addr

remember (in NAT translation table) every (source IP address, port #) to (NAT IP address, new port #) translation pair

incoming datagrams: replace (NAT IP address, new port #) in dest fields of every incoming datagram with corresponding (source IP address, port #) stored in NAT table

NAT traversal problem

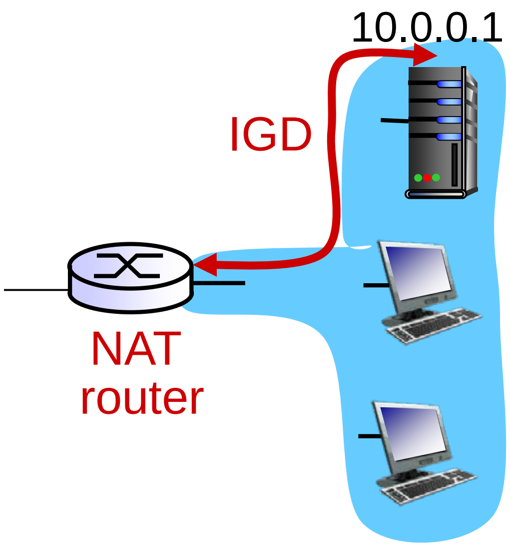
- client wants to connect to server with address 10.0.0.1
  - server address 10.0.0.1 local to LAN (client can't use it as destination addr)
  - only one externally visible address: 138.76.29.7
- *solution1:* statically configure NAT to forward incoming connection requests at given port to server
  - e.g., (123.76.29.7, port 2500) always forwarded to 10.0.0.1 port 25000
- *Solution 2:* automate the above through a protocol (universal plug-and-play)
- *Solution 3*: through a proxy/relay (will discuss in connection to p2p applications)



NAT traversal problem

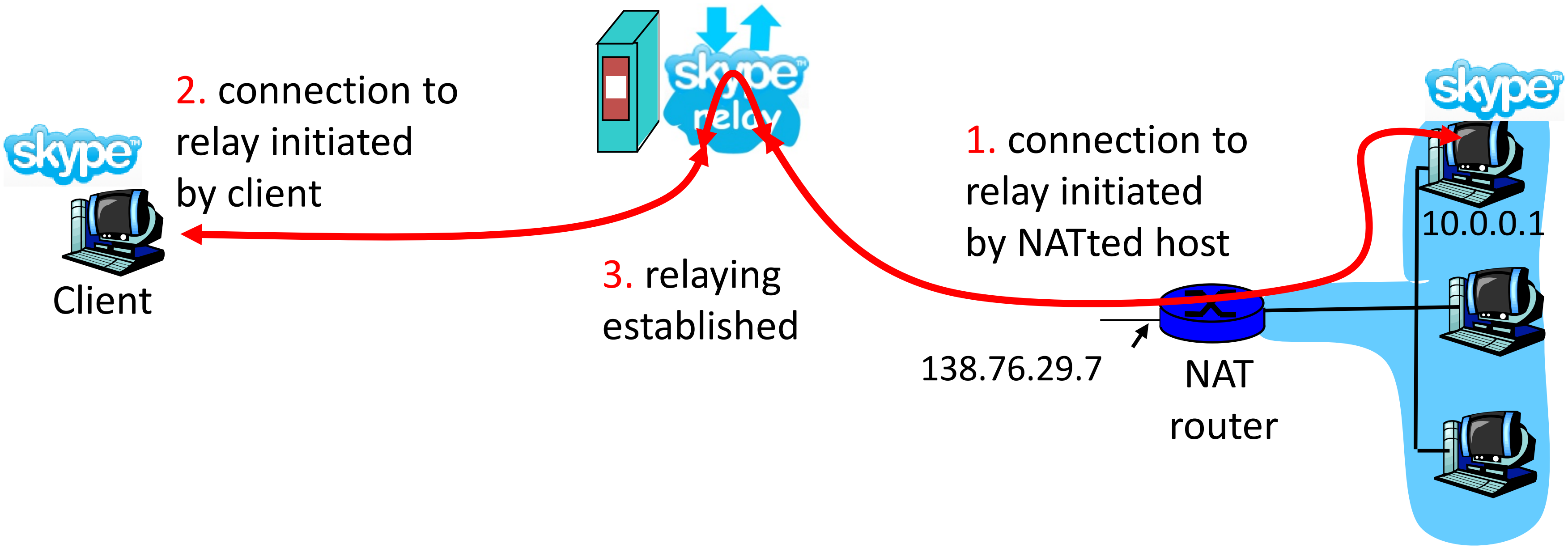
- *solution 2:* Universal Plug and Play (UPnP) Internet Gateway Device (IGD) Protocol. Allows NATed host to:
  - learn public IP address (138.76.29.7)
  - add/remove port mappings (with lease times)

  i.e., automate static NAT port map configuration



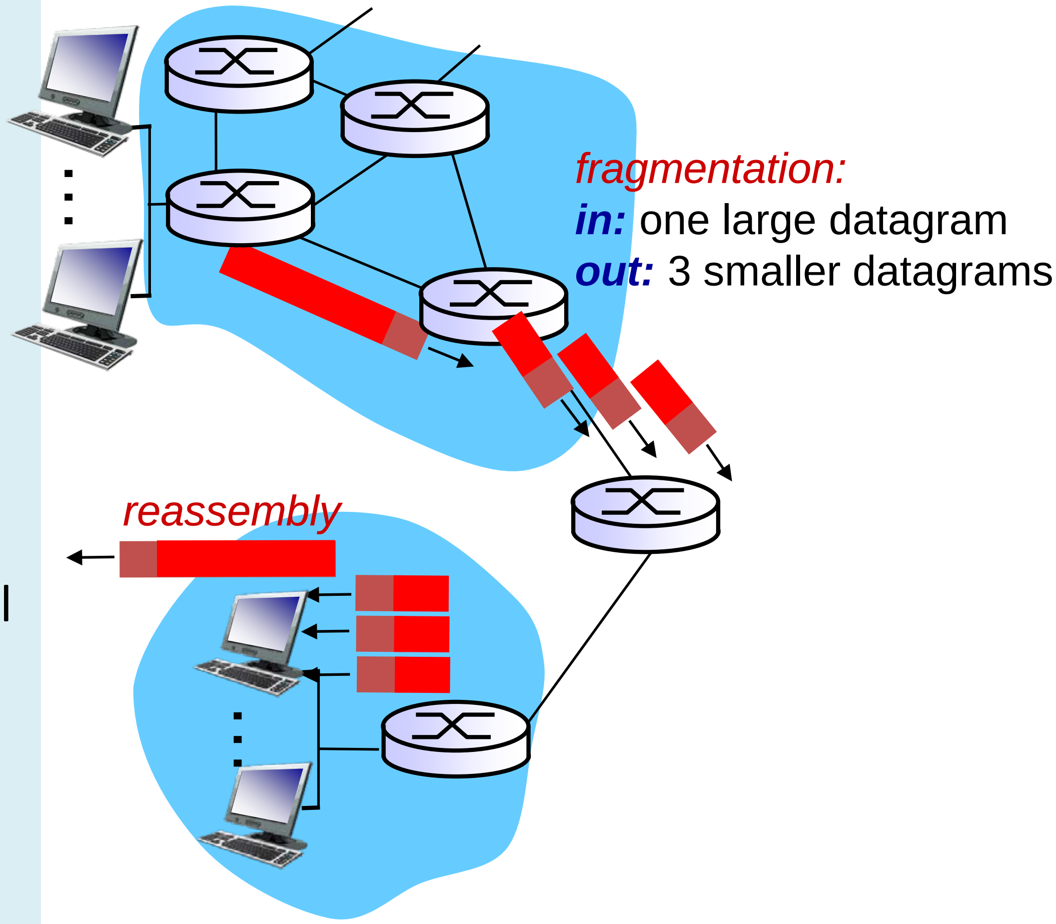
NAT traversal problem

- solution 3 (application): relaying (used in Skype)
  - NATed server establishes connection to relay
  - External client connects to relay
  - relay bridges packets between two connections



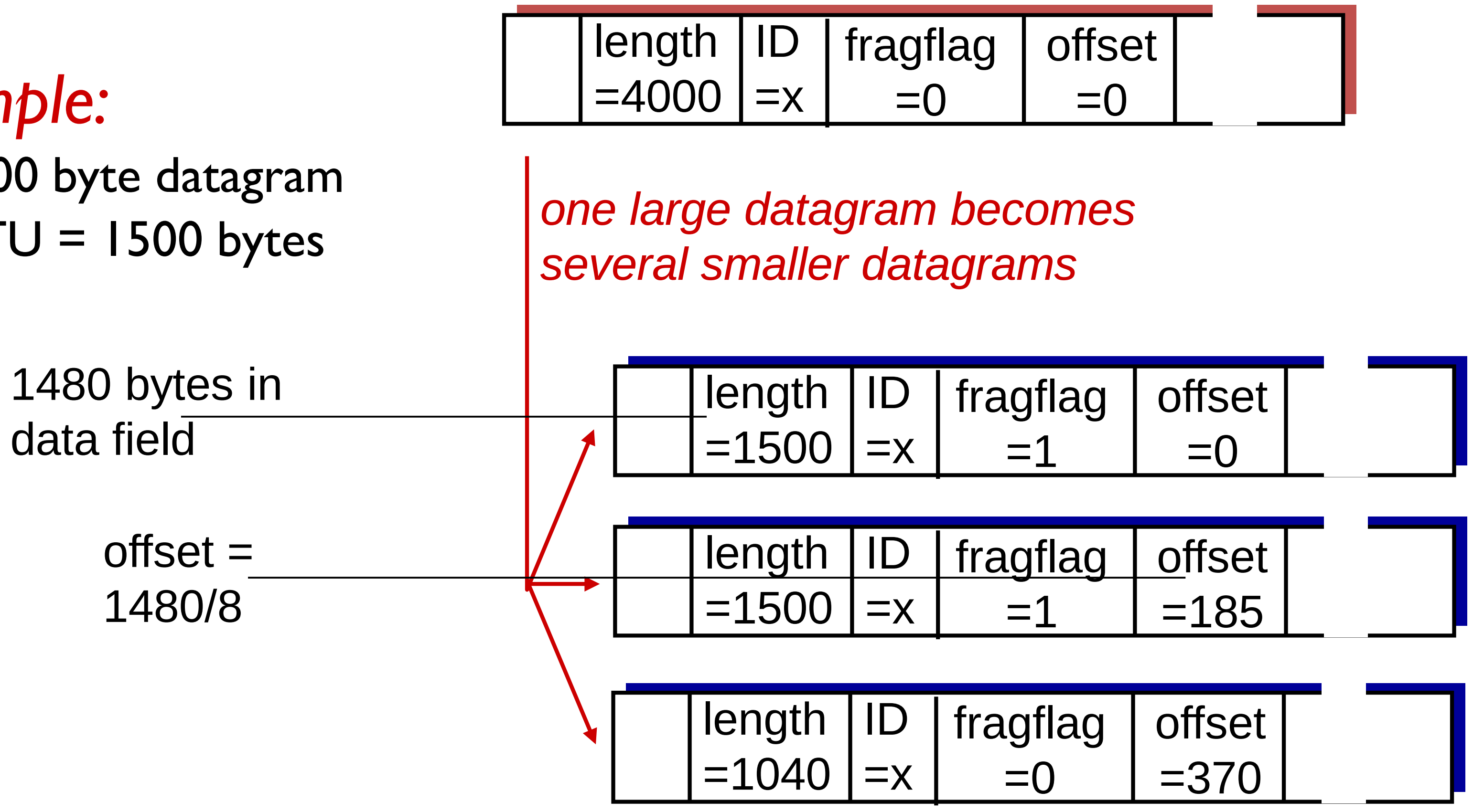
IP fragmentation, reassembly

- 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 to identify + order related fragments



IP fragmentation, reassembly

example:
- 4000 byte datagram
- MTU = 1500 bytes



Getting a datagram from source to dest.

Getting a datagram from source to dest.

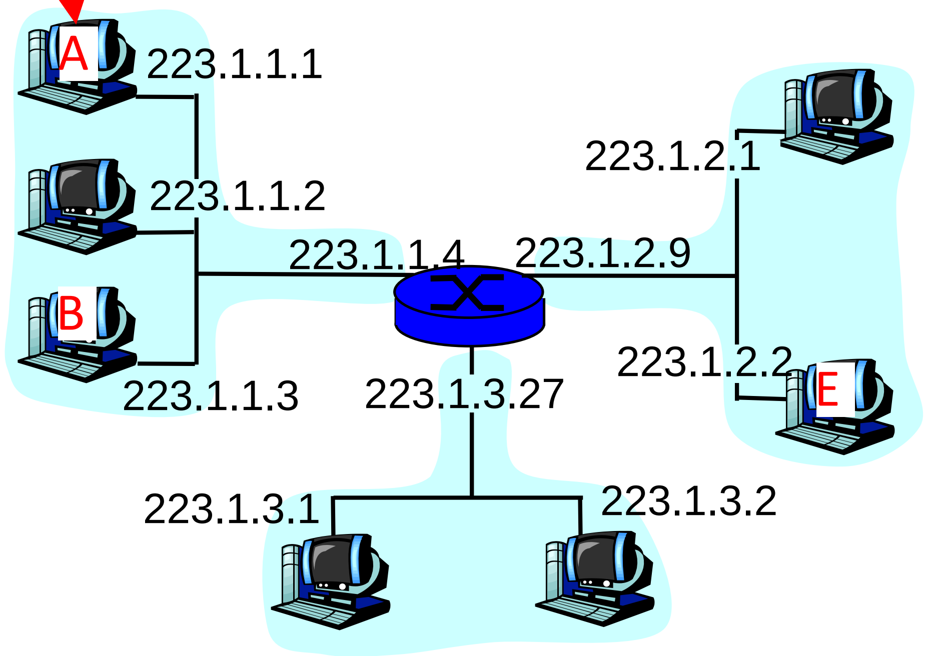
forwarding table in A

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

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



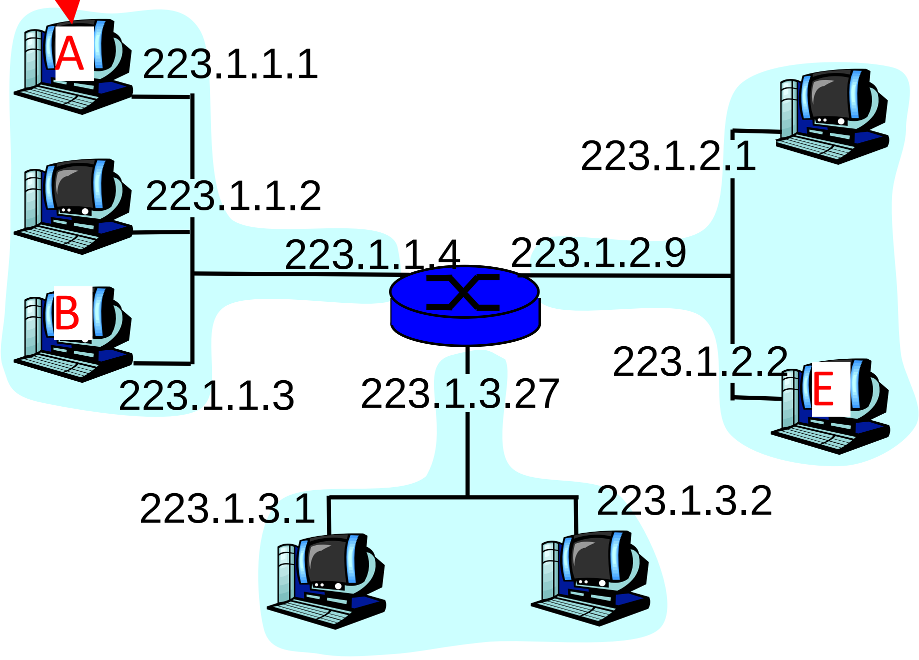
Getting a datagram from source to dest.

misc fields	223.1.1.1	223.1.1.3	data

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

Starting at A, given IP datagram addressed to B:

- look up net. address of B
- find B is on same net. as A (B and A are directly connected)
- link layer will send datagram directly to B (inside link-layer frame)



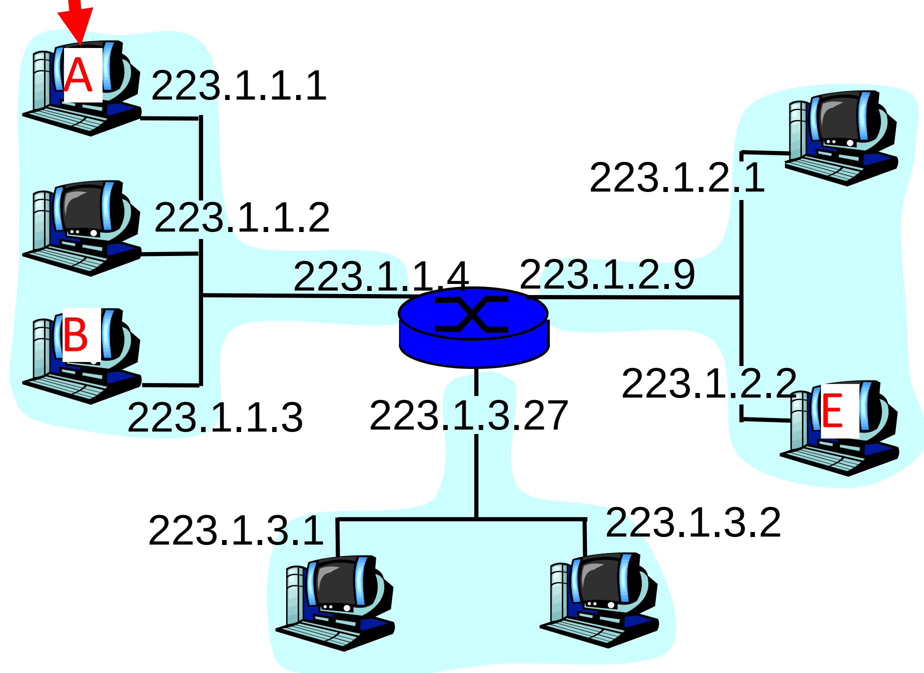
Getting a datagram from source to dest.

misc fields	223.1.1.1	223.1.2.3	data

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

Starting at A, dest. E:

- look up network address of E
- E on *different* network
- routing table: next hop router to E is 223.1.1.4
- link layer is asked to send datagram to router 223.1.1.4 (inside link-layer frame)
- datagram arrives at 223.1.1.4
- continued…..



Getting a datagram from source to dest.

misc fields	223.1.1.1	223.1.2.3	data

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

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!)