Network Layer and Data Center Topologies

Hakim Weatherspoon
Assistant Professor, Dept of Computer Science

CS 5413: High Performance Systems and Networking

September 8, 2014

Slides used and adapted judiciously from Computer Networking, A Top-Down Approach

Goals for Today

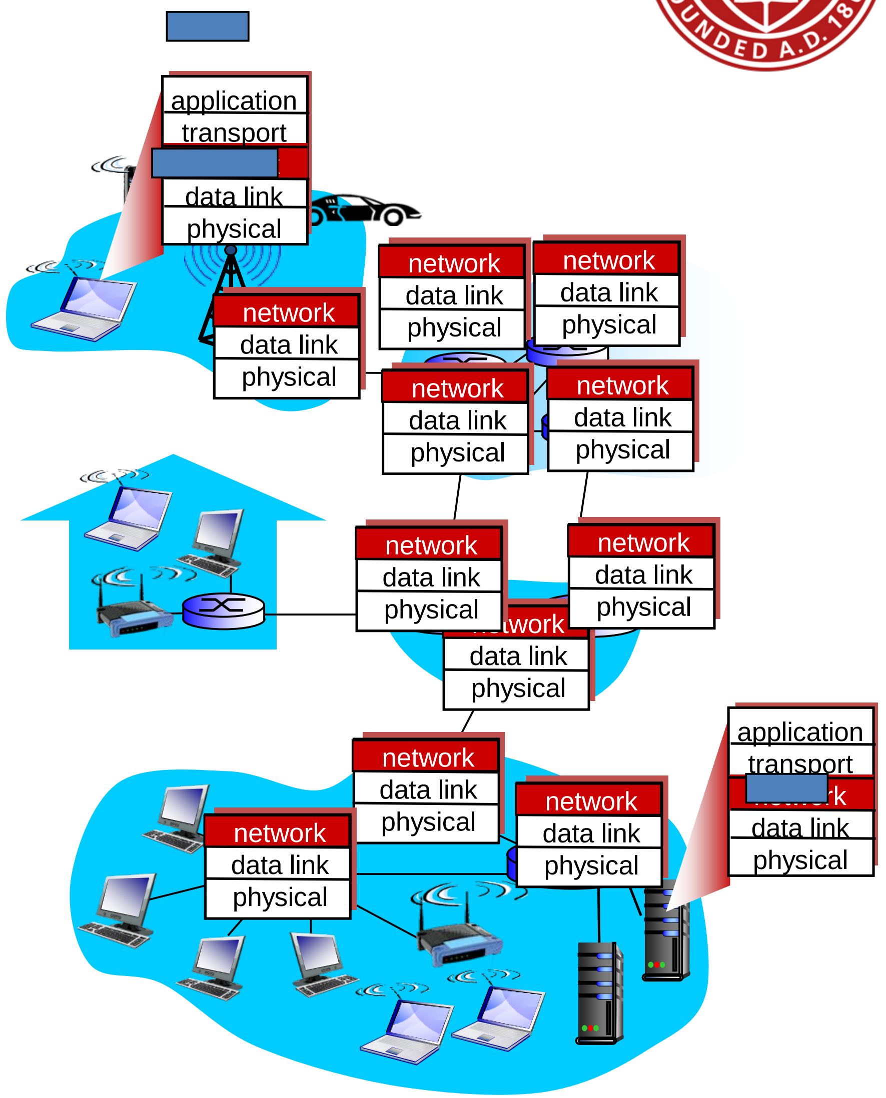
- Network Layer
  - Abstraction / services
    - Datagram vs Virtual Circuit (VC)
  - Internet Protocol
    - IP Datagram format
    - IP Addressing
    - Hierarchical Routing

- Data Center Topologies
  - FatTree

- Backup Slides
  - DHCP and NAT
  - ICMP and Traceroute
  - IPv6
  - Hierarchical Routing: 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



Network Layer

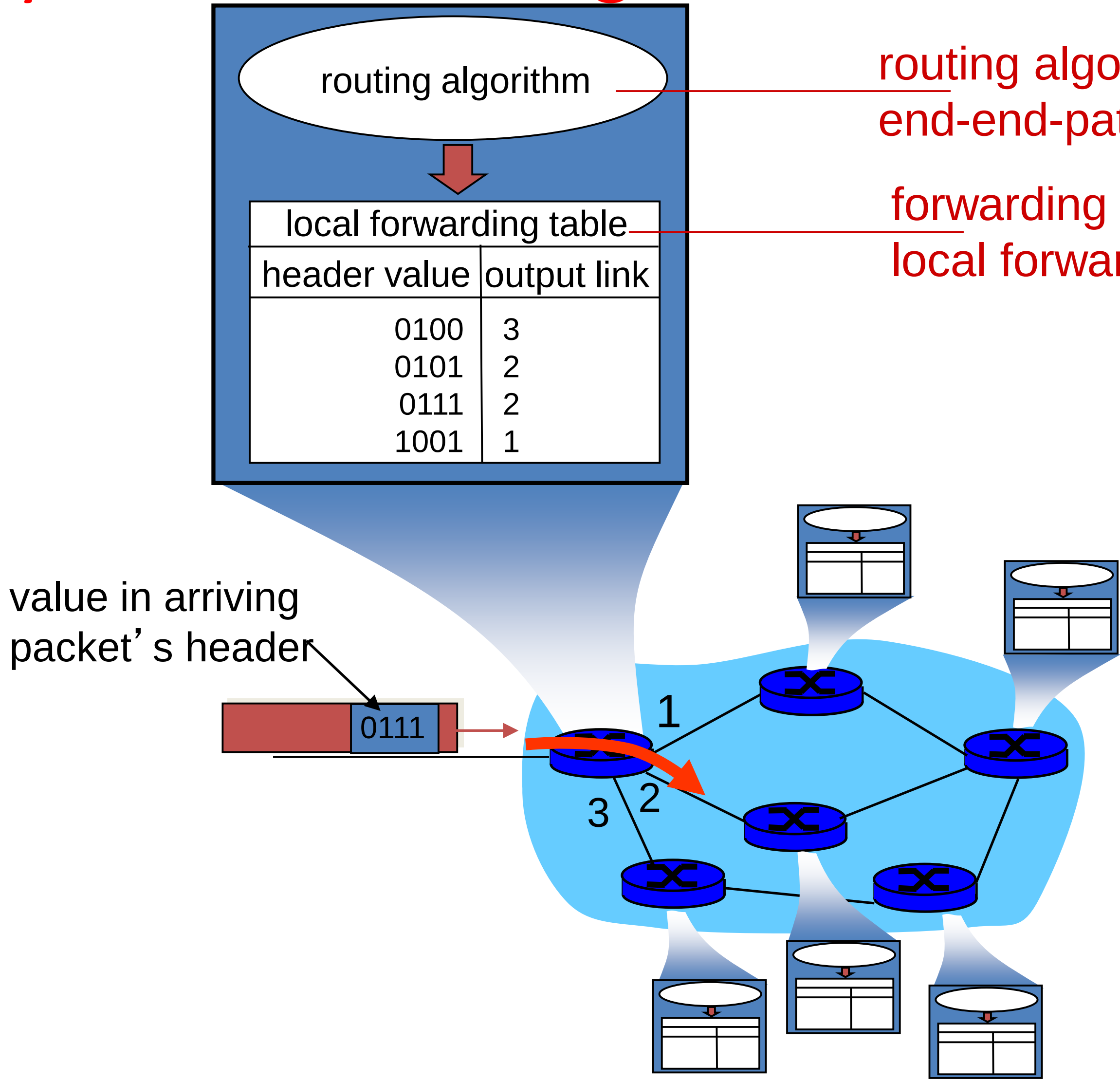
Two key functions

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

analogy:

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

Interplay between routing and forwarding



routing algorithm determines end-end-path through network

forwarding table determines local forwarding at this router

Goals for Today

- Network Layer
  - Abstraction / services
    - Datagram vs Virtual Circuit (VC)
  - Internet Protocol
    - IP Datagram format
    - Addressing
    - Hierarchical Routing

- Data Center Topologies
  - FatTree

- Backup Slides
  - DHCP and NAT
  - ICMP and Traceroute
  - IPv6
  - Hierarchical Routing: RIP, OSPF, BGP

Network Layer

Connection, Connection-less services

- *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

Virtual Circuits (VC)

"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)

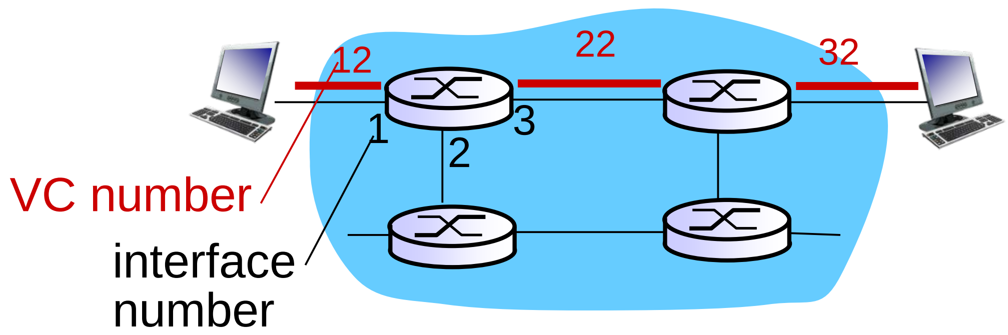
Virtual Circuits (VC) implementation

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

Virtual Circuits (VC) forwarding table



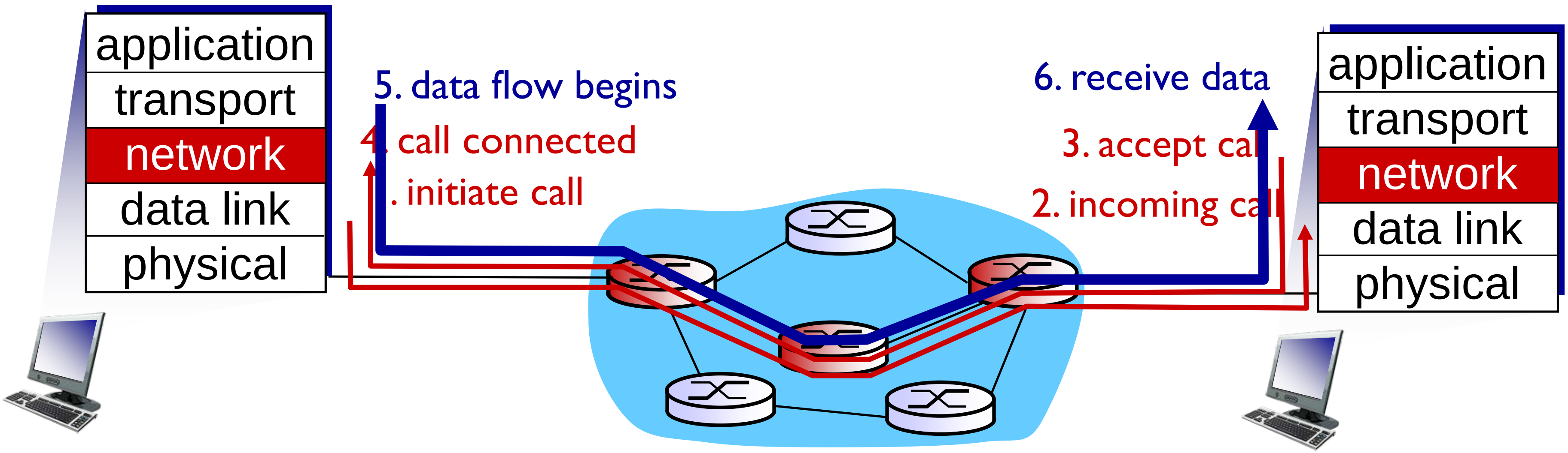
forwarding table in northwest router:

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!

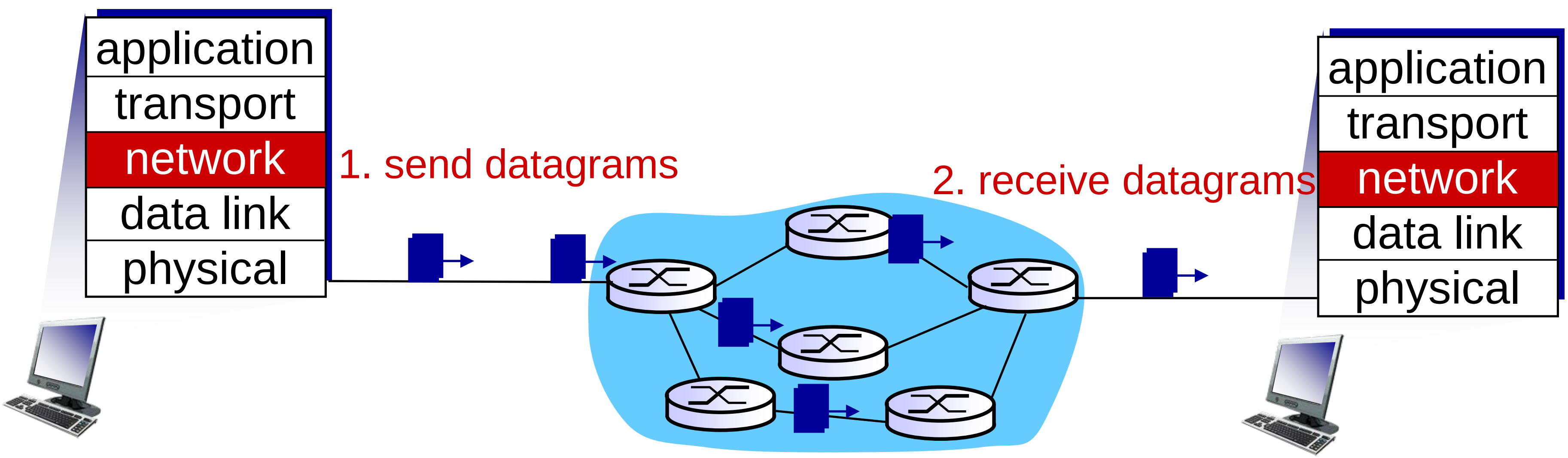
Virtual Circuits (VC) signaling protocol

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



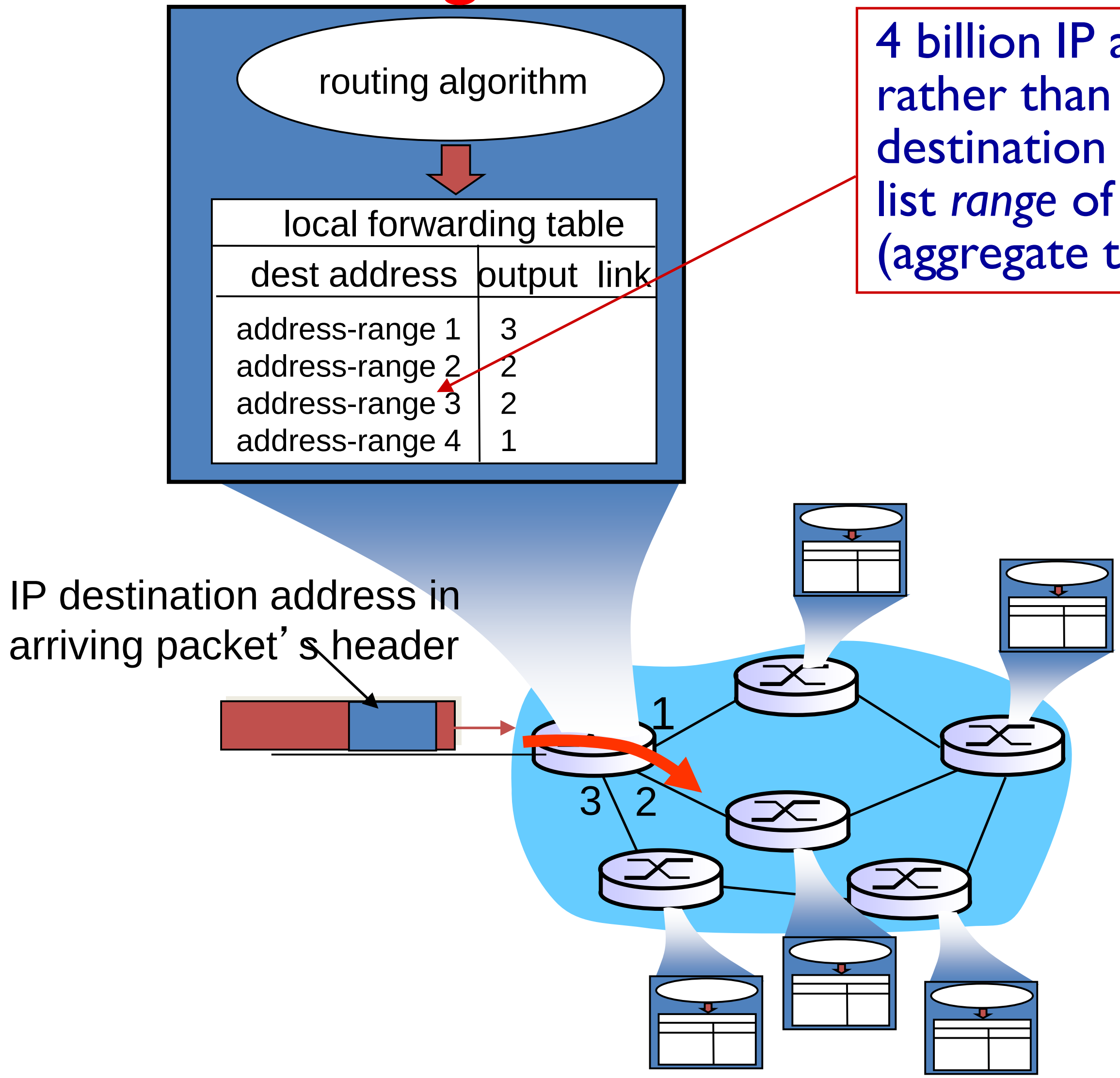
Datagram Networks

- 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



Network Layer

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 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

Q: but what happens if ranges don't divide up so nicely?

Datagram Forwarding Table: 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?

Datagram versus Virtual Circuits (VC)

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”***

ATM (VC)

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

Goals for Today

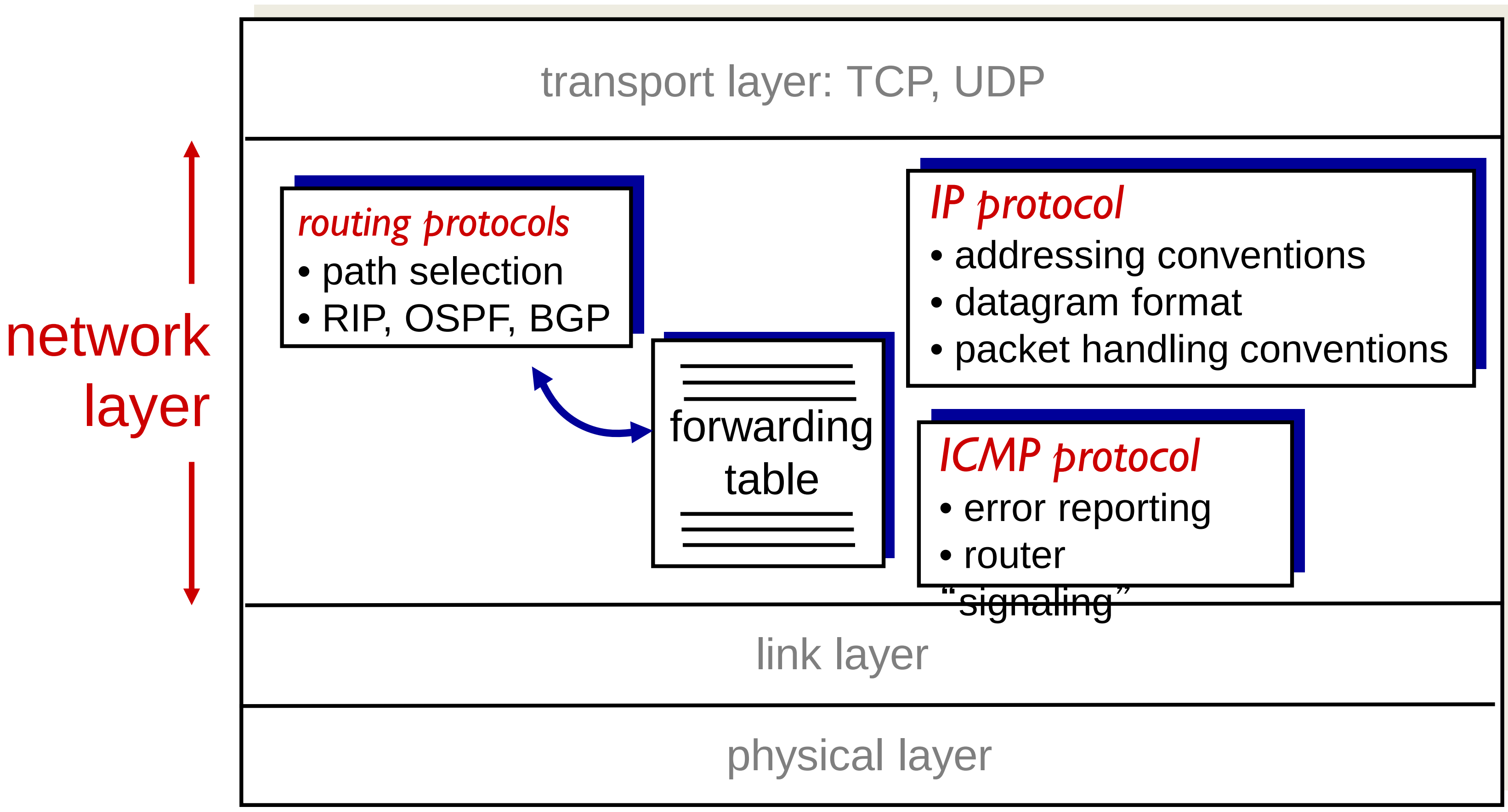
- Network Layer
  - Abstraction / services
    - Datagram vs Virtual Circuit (VC)
  - Internet Protocol
    - IP Datagram format
    - IP Addressing
    - Hierarchical Routing

- Data Center Topologies
  - FatTree

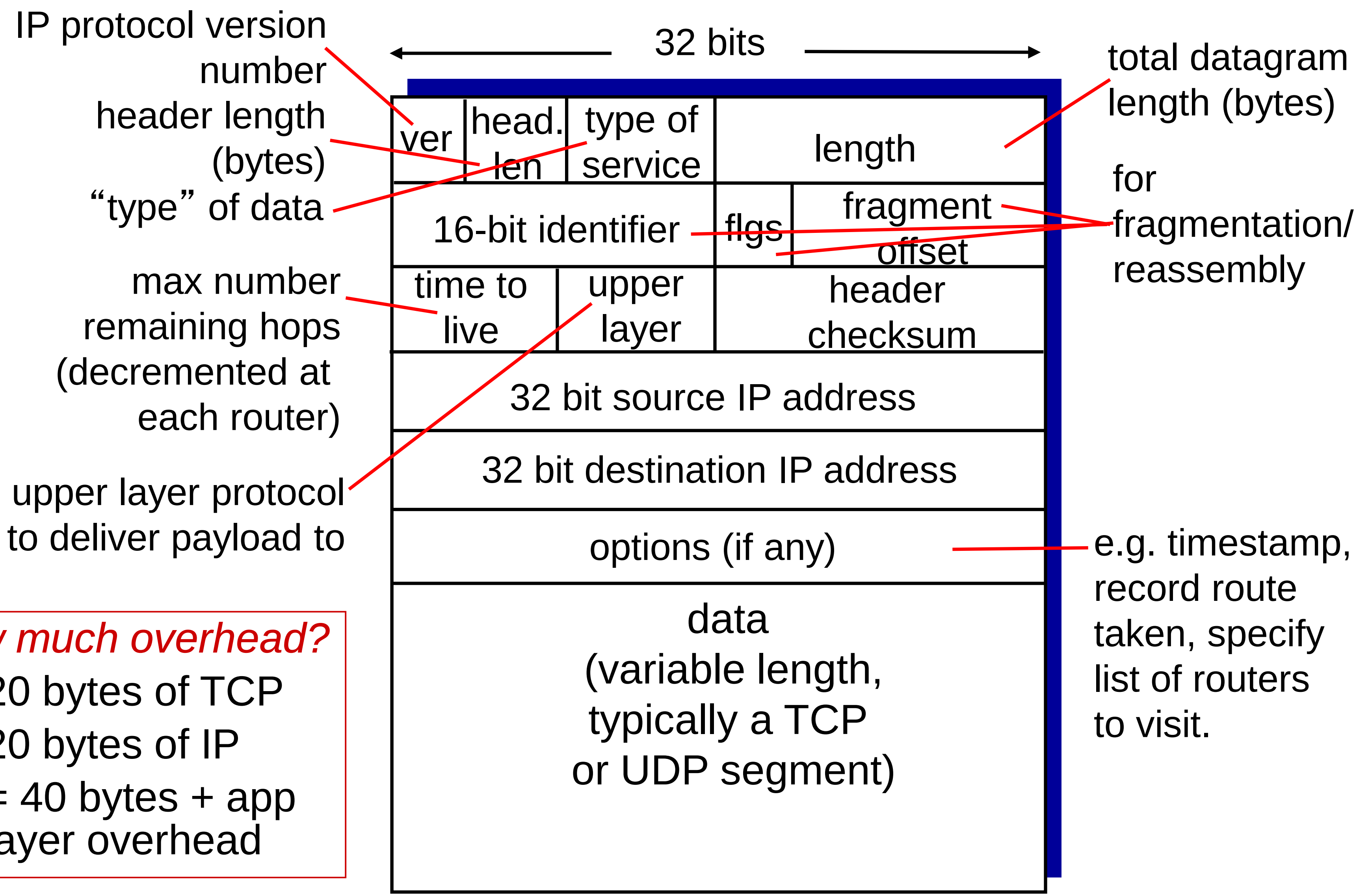
- Backup Slides
  - DHCP and NAT
  - ICMP and Traceroute
  - IPv6
  - Hierarchical Routing: RIP, OSPF, BGP

The Internet Protocol Network Layer

host, router network layer functions:



IP Datagram format

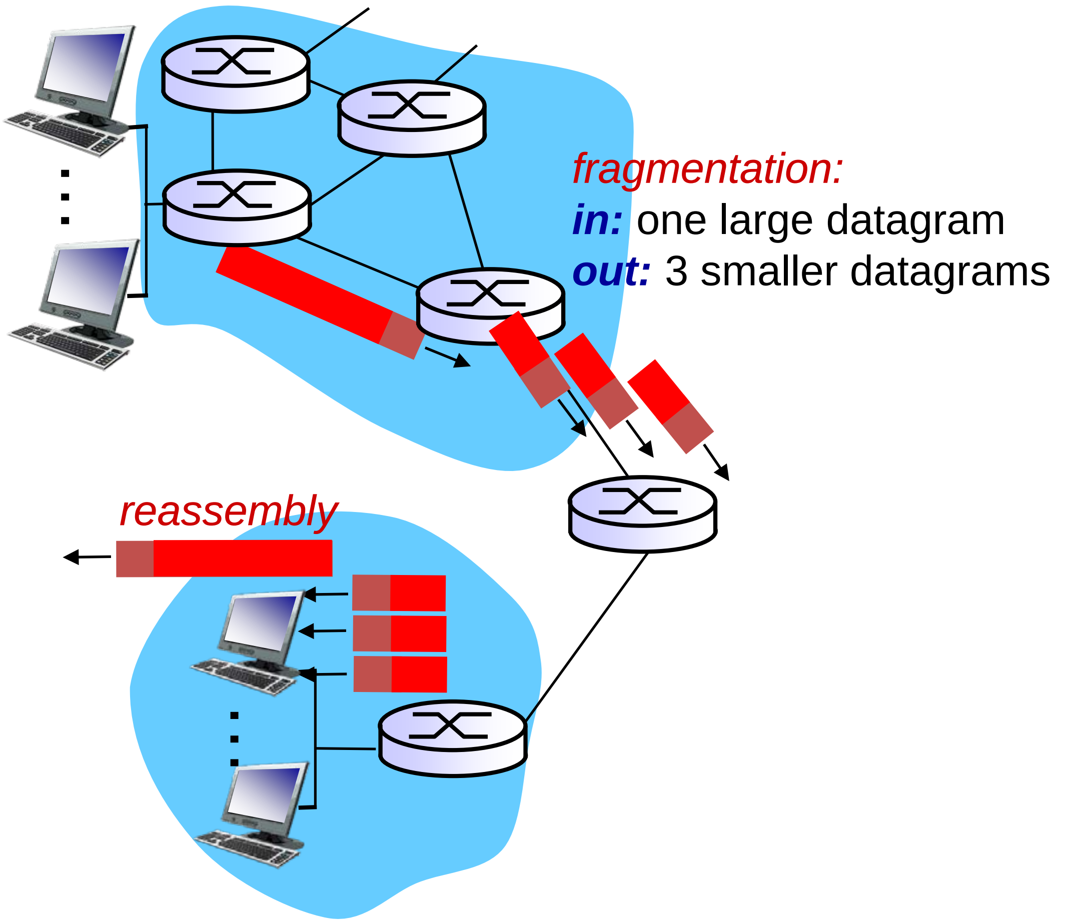


how much overhead?

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

IP Fragmentation/Reassembly

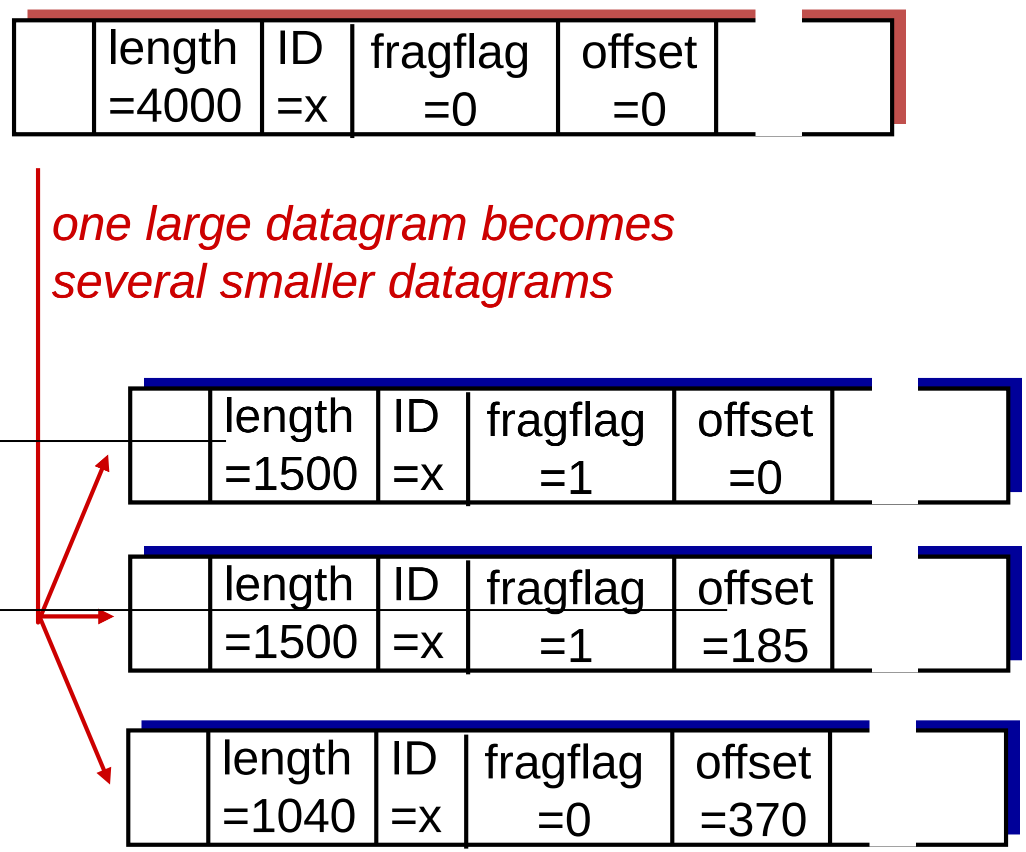
- network links have MTU (max.transfer size) - largest possible link-level frame
  - different link types, different MTUs
- large IP datagram divided ("fragmented") within net
  - one datagram becomes several datagrams
  - "reassembled" only at final destination
  - IP header bits used to identify, order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



IP Fragmentation/Reassembly

example:

- 4000 byte datagram
- MTU = 1500 bytes



1480 bytes in data field

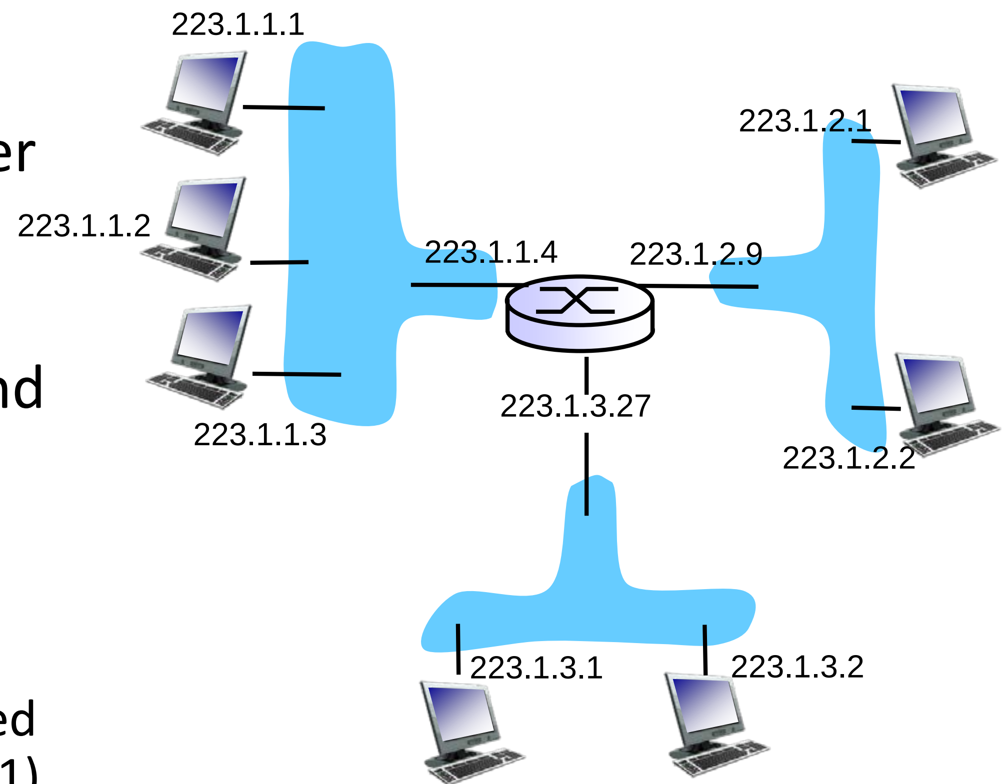
offset = 1480/8

Goals for Today

- Network Layer
  - Abstraction / services
    - Datagram vs Virtual Circuit (VC)
  - Internet Protocol
    - IP Datagram format
    - IP Addressing
    - Hierarchical Routing
- Data Center Topologies
  - FatTree
- Backup Slides
  - DHCP and NAT
  - ICMP and Traceroute
  - IPv6
  - Hierarchical Routing: RIP, OSPF, BGP

IP Addressing

- *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*

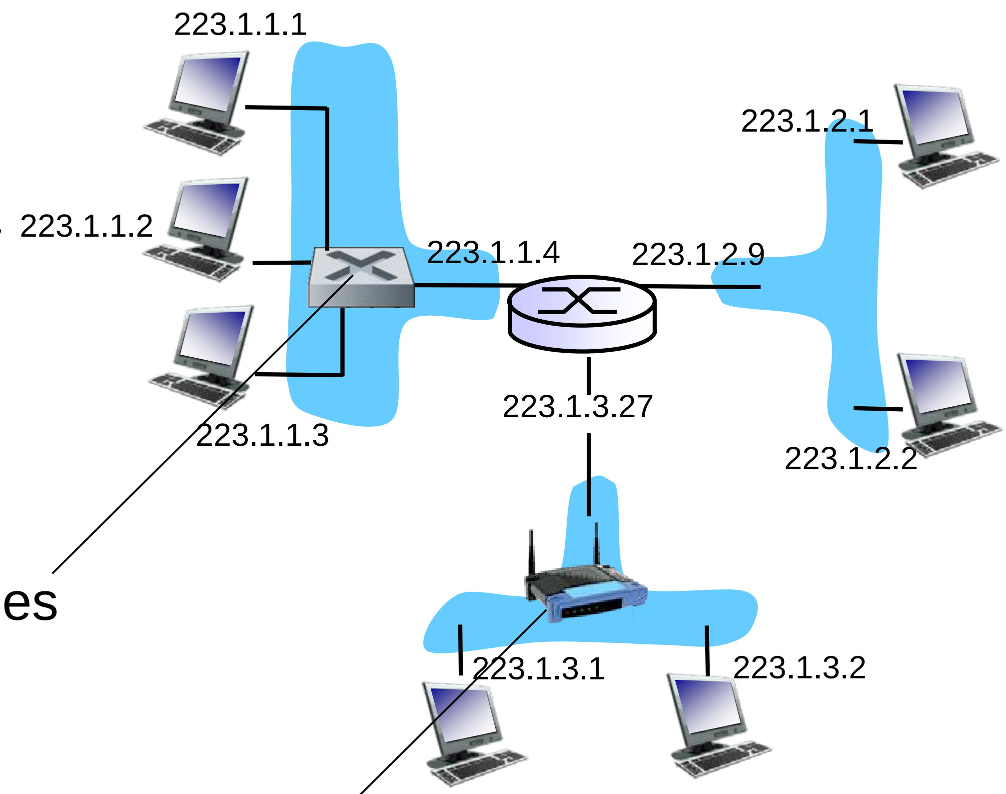


$$223.1.1.1 = \underbrace{11011111}_{223}\ \underbrace{00000001}_{1}\ \underbrace{00000001}_{1}\ \underbrace{00000001}_{1}$$

IP Addressing

Q: how are interfaces actually connected?

A: we'll learn about that in chapter 5, 6.



A: wired Ethernet interfaces connected by Ethernet switches

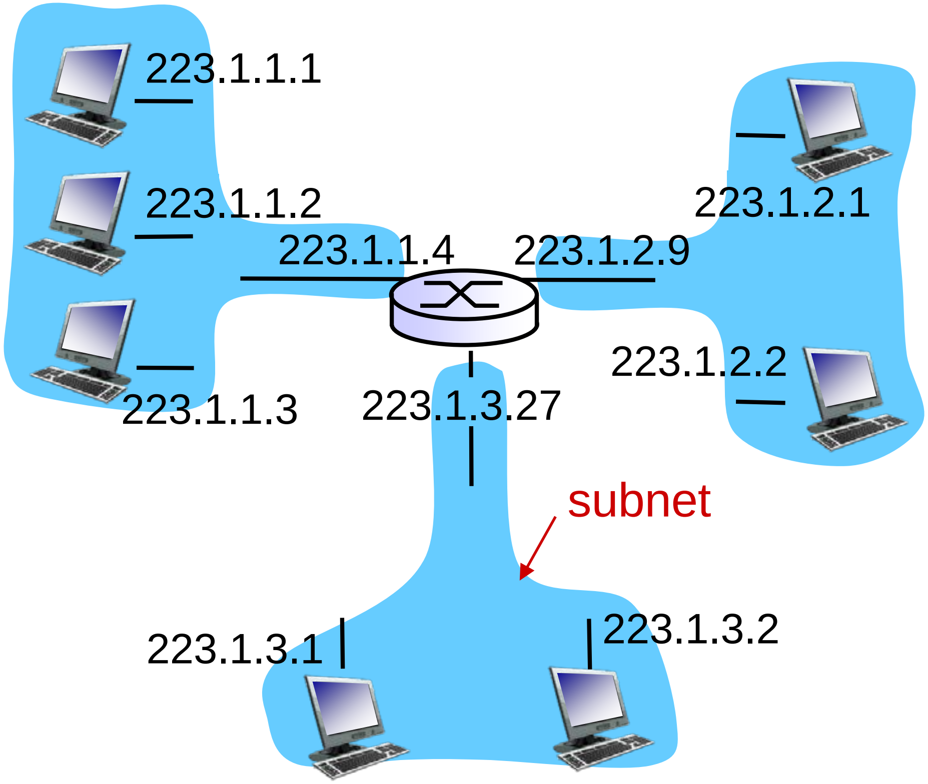
For now: don't need to worry about how one interface is connected to another (with no intervening router)

A: wireless WiFi interfaces connected by WiFi base station

The Internet Protocol Network Layer

Subnets

- IP address:
  - –subnet part - high order bits
  - –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*

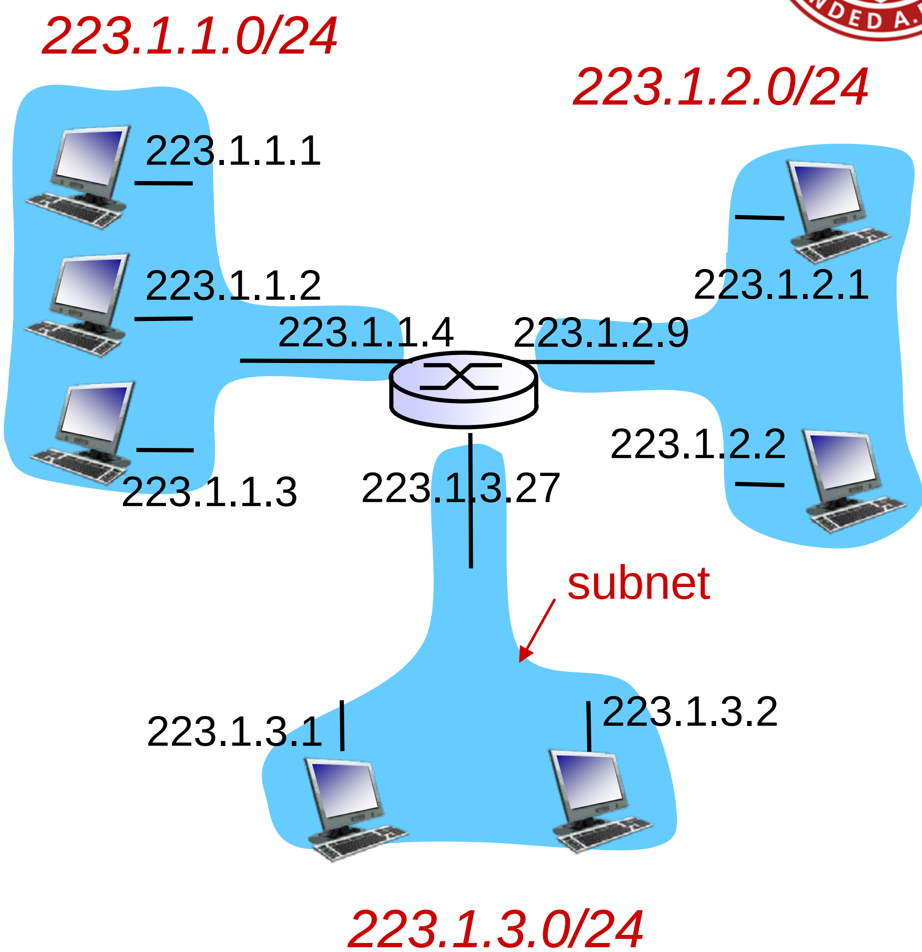


network consisting of 3 subnets

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*

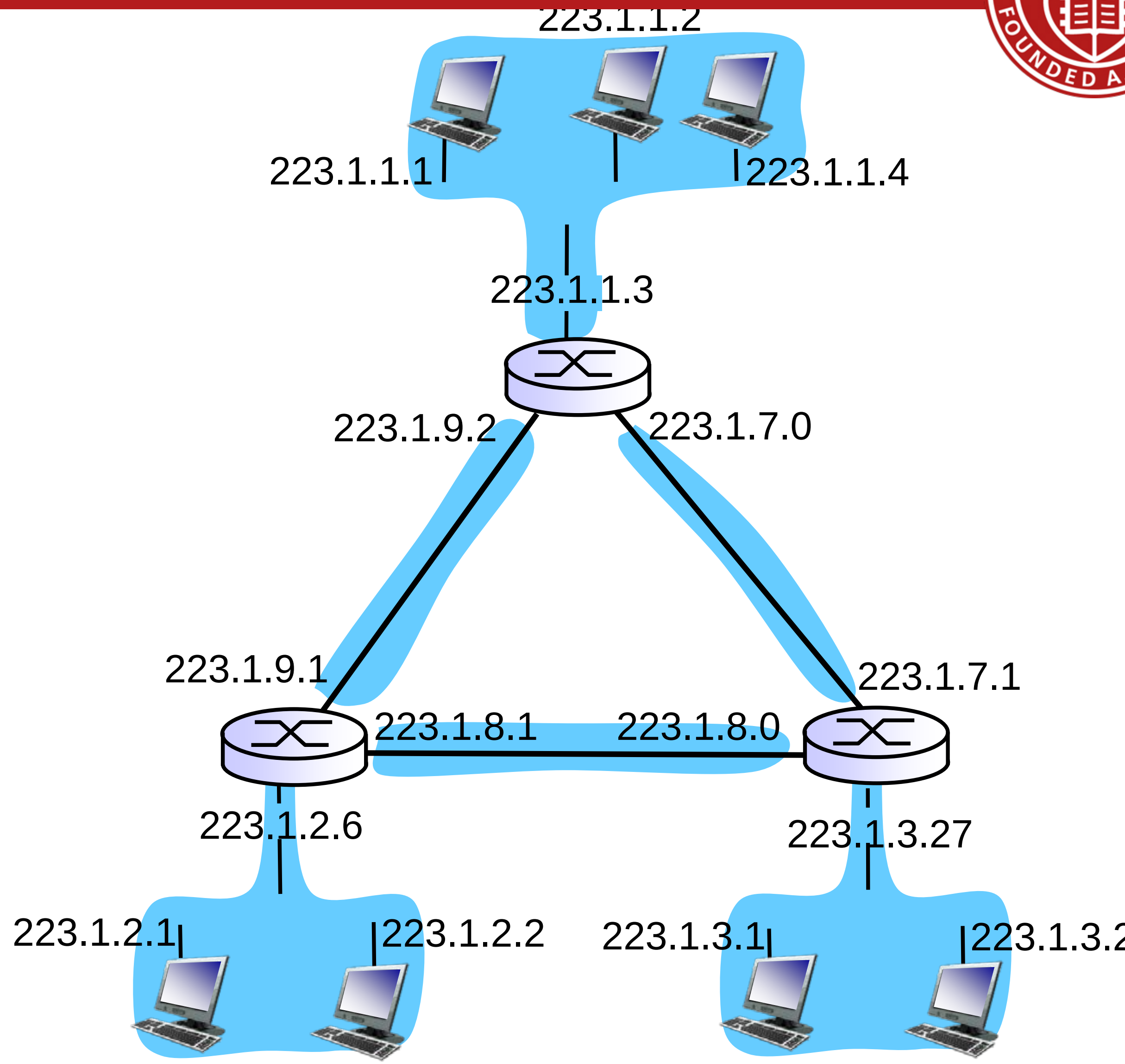


subnet mask: /24

The Internet Protocol Network Layer

Subnets

how many?



IP Addressing: CIDR (Classess InterDomain Routing)

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

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"

Goals for Today

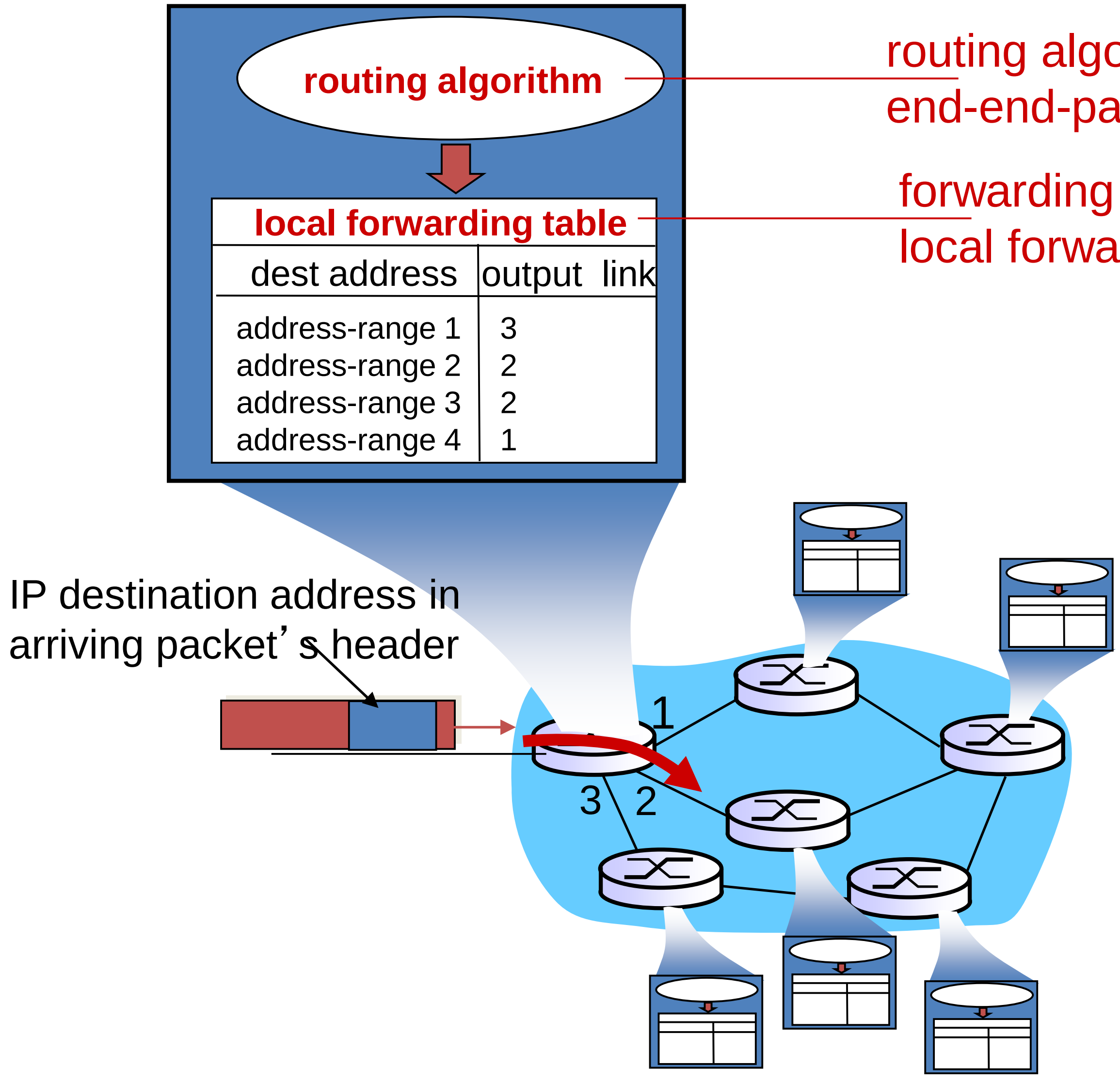
- Network Layer
  - Abstraction / services
    - Datagram vs Virtual Circuit (VC)
  - Internet Protocol
    - IP Datagram format
    - IP Addressing / subnets
    - Routing Algorithms
    - Hierarchical Routing

- Data Center Topologies
  - FatTree

- Backup Slides
  - DHCP and NAT
  - ICMP and Traceroute
  - IPv6
  - Hierarchical Routing: RIP, OSPF, BGP

The Internet Protocol Network Layer

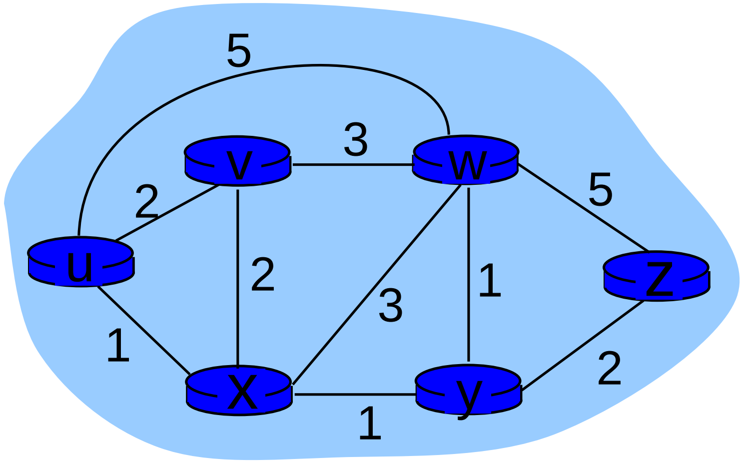
Interplay between routing and forwarding



routing algorithm determines end-end-path through network

forwarding table determines local forwarding at this router

Graph Abstractions



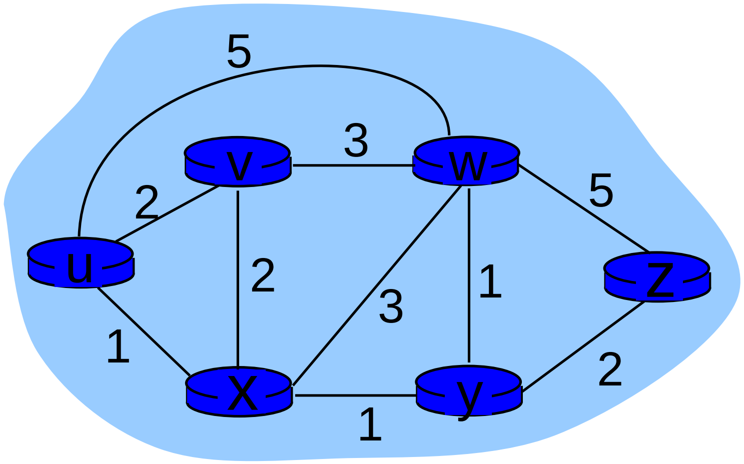
graph: G = (N,E)

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 Abstractions: 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 inversely 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 Classifications

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

Goals for Today

- Network Layer
  - Abstraction / services
    - Datagram vs Virtual Circuit (VC)
  - Internet Protocol
    - IP Datagram format
    - IP Addressing / subnets
    - Routing Algorithms
    - Hierarchical Routing

- Data Center Topologies
  - FatTree

- Backup Slides
  - DHCP and NAT
  - ICMP and Traceroute
  - IPv6
  - Hierarchical Routing: 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

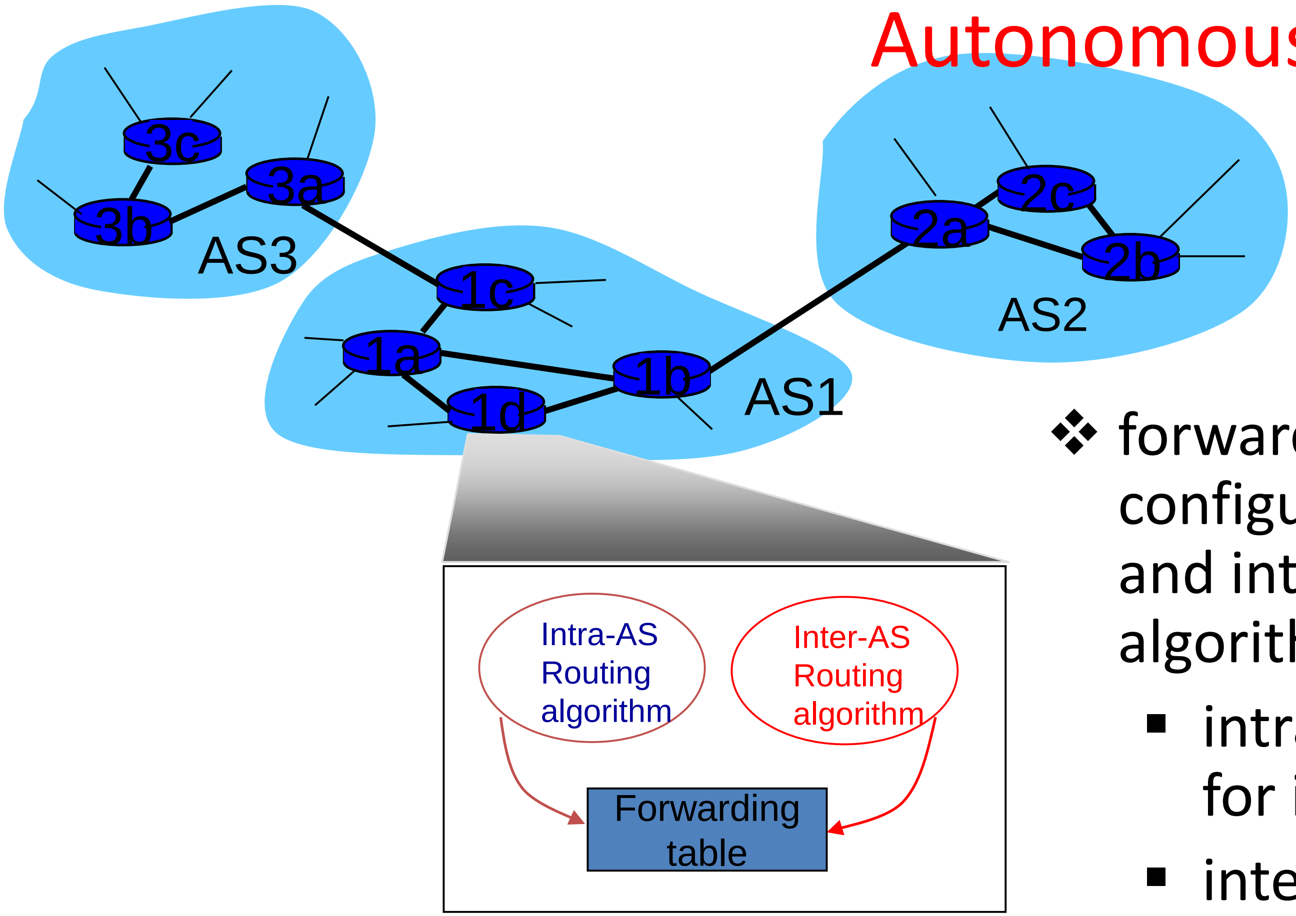
Hierarchical Routing

- 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

Hierarchical Routing: Interconnected Autonomous Systems (AS)



- 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

Hierarchical Routing: 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)

Hierarchical Routing: 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"*

The Internet Protocol Network Layer

Hierarchical Routing: Inter-AS routing—BGP

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

Hierarchical Routing: Inter-AS routing—BGP

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

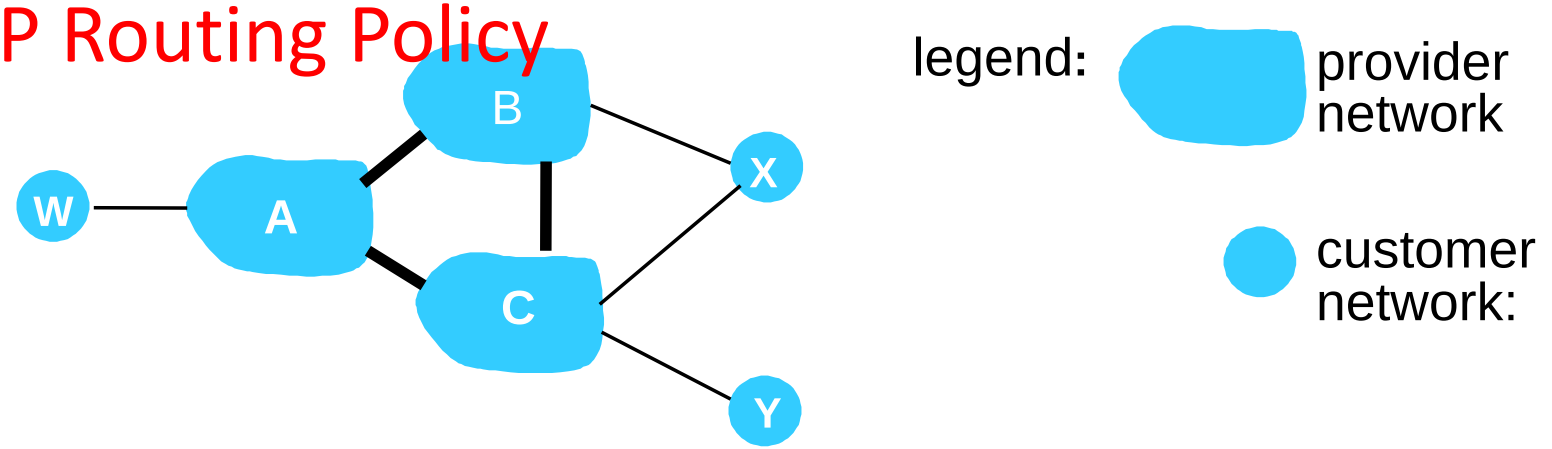
Hierarchical Routing: Inter-AS routing—BGP

BGP Messages

- BGP messages exchanged between peers over TCP connection
- BGP messages:
  - OPEN: opens TCP connection to peer and authenticates sender
  - UPDATE: advertises new path (or withdraws old)
  - KEEPALIVE: keeps connection alive in absence of UPDATES; also ACKs OPEN request
  - NOTIFICATION: reports errors in previous msg; also used to close connection

Hierarchical Routing: Inter-AS routing—BGP

BGP Routing Policy

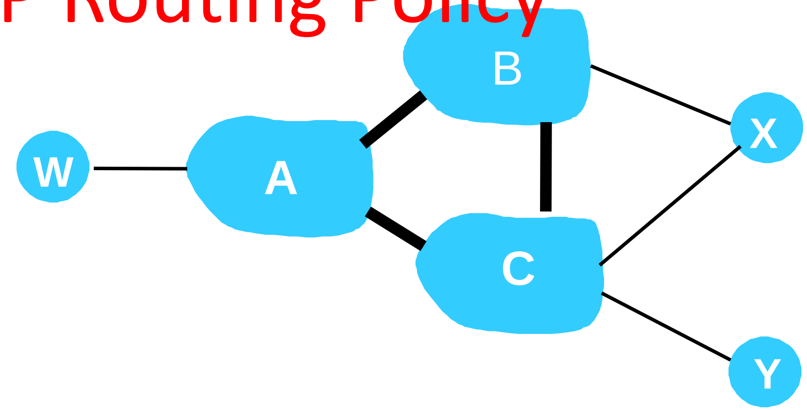


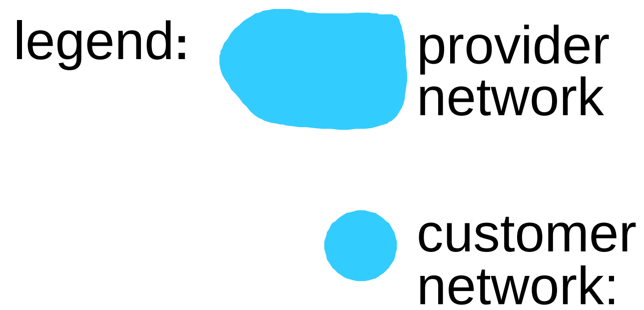
- A,B,C are *provider networks*
- X,W,Y are customer (of provider networks)
- X is *dual-homed:* attached to two networks
  - X does not want to route from B via X to C
  - .. so X will not advertise to B a route to C

The Internet Protocol Network Layer

Hierarchical Routing: Inter-AS routing—BGP

BGP Routing Policy





- A advertises path AW to B
- B advertises path BAW to X
- Should B advertise path BAW to C?
  - No way! B gets no "revenue" for routing CBAW since neither W nor C are B's customers
  - B wants to force C to route to w via A
  - B wants to route *only* to/from its customers!

Hierarchical Routing: Intra vs 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

Goals for Today

- Network Layer
  - Abstraction / services
    - Datagram vs Virtual Circuit (VC)
  - Internet Protocol
    - IP Datagram format
    - IP Addressing / subnets
    - Routing Algorithms
    - Hierarchical Routing
- Data Center Topologies
  - FatTree
- Backup Slides
  - DHCP and NAT
  - ICMP and Traceroute
  - IPv6
  - Hierarchical Routing: RIP, OSPF, BGP

Data Center Topology: FatTree

- A scalable, commodity data center network architecture, M. Al-Fares, A. Loukissas, and A. Vahdat. *ACM SIGCOMM Computer Communication Review*, Volume 38, Issue 4 (August 2008), pages 63-74.

Data Center Topology: FatTree

Overview

- Structure and Properties of a Data Center
- Desired properties in a DC Architecture
- Fat tree based solution

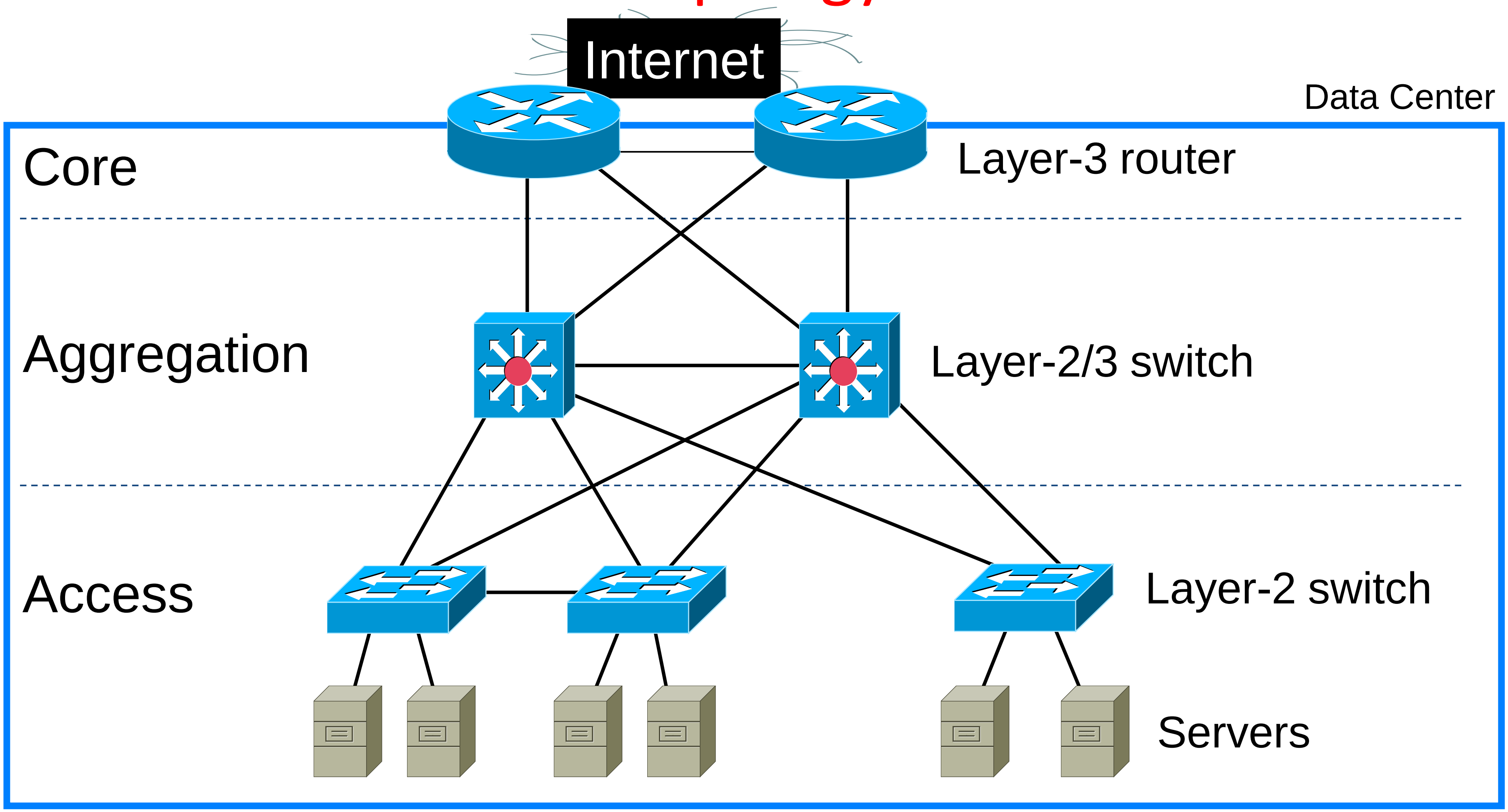
Data Center Topology: FatTree

Background

- *Topology:*
  - 2 layers: 5K to 8K hosts
  - 3 layer: >25K hosts
  - Switches:
    - Leaves: have N GigE ports (48-288) + N 10 GigE uplinks to one or more layers of network elements
    - Higher levels: N 10 GigE ports (32-128)
- *Multi-path Routing:*
  - Ex. ECMP
    - without it, the largest cluster = 1,280 nodes
    - Performs static load splitting among flows
    - Lead to oversubscription for simple comm. patterns
    - Routing table entries grows multiplicatively with number of paths, cost ++, lookup latency ++

Data Center Topology: FatTree

Common Data Center Topology



Data Center Topology: FatTree

Issues with Traditional Data Center Topology

- Single point of failure
- Over subscript of links higher up in the topology
  - Trade off between cost and provisioning

Data Center Topology: FatTree

Issues with Traditional Data Center Topology

- *Oversubscription:*
  - Ratio of the worst-case achievable aggregate bandwidth among the end hosts to the total bisection bandwidth of a particular communication topology
  - Lower the total cost of the design
  - Typical designs: factor of 2:5:1 (400 Mbps)to 8:1(125 Mbps)
- *Cost:*
  - Edge: $7,000 for each 48-port GigE switch
  - Aggregation and core: $700,000 for 128-port 10GigE switches
  - Cabling costs are not considered!

Data Center Topology: FatTree

Properties of Desired Solution

- Backwards compatible with existing infrastructure
  - No changes in application
  - Support of layer 2 (Ethernet)
- Cost effective
  - Low power consumption & heat emission
  - Cheap infrastructure
- Allows host communication at line speed

Data Center Topology: FatTree

Properties of Desired Solution: Tradeoffs

- **Leverages specialized hardware and communication protocols, such as InfiniBand, Myrinet.**
  - These solutions can scale to clusters of thousands of nodes with high bandwidth
  - Expensive infrastructure, incompatible with TCP/IP applications
- **Leverages commodity Ethernet switches and routers to interconnect cluster machines**
  - Backwards compatible with existing infrastructures, low-cost
  - Aggregate cluster bandwidth scales poorly with cluster size, and achieving the highest levels of bandwidth incurs non-linear cost increase with cluster size

Data Center Topology: FatTree

Proposed Solution: FatTree (Clos Network)

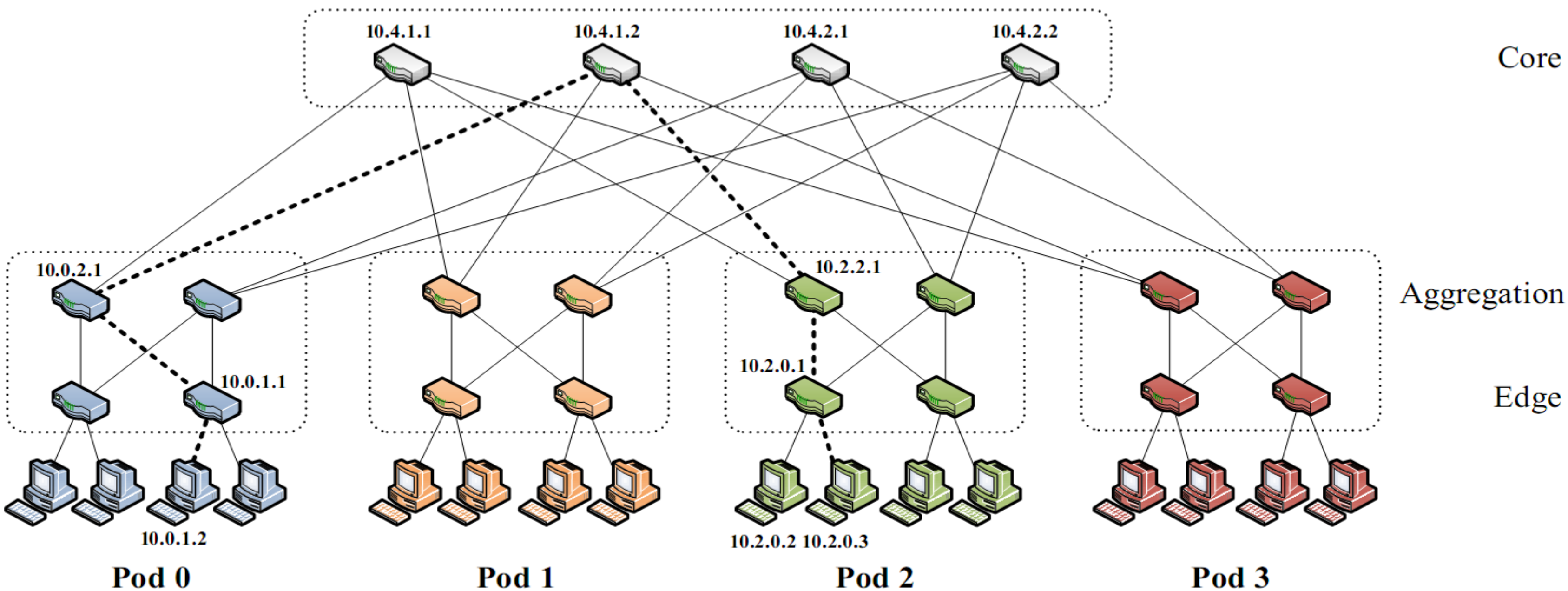
- Adopt a special instance of a Clos topology
- Similar trends in telephone switches led to designing a topology with high bandwidth by interconnecting smaller commodity switches.

Data Center Topology: FatTree

FatTree Based Data Center Architecture

- *Inter-connect racks (of servers) using a fat-tree topology*

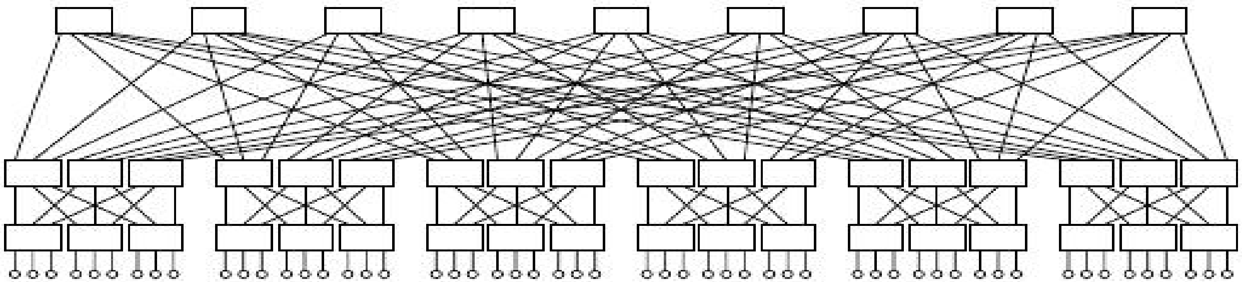
  K-ary fat tree: three-layer topology (edge, aggregation and core)
  - each pod consists of $(k/2)^2$ servers & 2 layers of k/2 k-port switches
  - each edge switch connects to k/2 servers & k/2 aggr. switches
  - each aggr. switch connects to k/2 edge & k/2 core switches
  - $(k/2)^2$ core switches: each connects to k pods



Data Center Topology: FatTree

FatTree Based Data Center Architecture

- **Why Fat-Tree?**
  - Fat tree has identical bandwidth at any bisections
  - Each layer has the same aggregated bandwidth
- Can be built using cheap devices with uniform capacity
  - Each port supports same speed as end host
  - All devices can transmit at line speed if packets are distributed uniform along available paths
- Great scalability: k-port switch supports $k^3/4$ servers



Data Center Topology: FatTree

Problems with FatTree

- Layer 3 will only use one of the existing equal cost paths
  - Bottlenecks up and down the fat-tree
    - Simple extension to IP forwarding
- Packet re-ordering occurs if layer 3 blindly takes advantage of path diversity ; further load may not necessarily be well-balanced
- Wiring complexity in large networks
  - Packing and placement technique

Data Center Topology: FatTree

FatTree Modified

- Enforce a special (IP) addressing scheme in DC
  - unused.PodNumber.switchnumber.Endhost
  - Allows host attached to same switch to route only through switch
  - Allows inter-pod traffic to stay within pod

Data Center Topology: FatTree

FatTree Modified

- Use two level look-ups to distribute traffic and maintain packet ordering
  - First level is prefix lookup
    - used to route down the topology to servers
  - Second level is a suffix lookup
    - used to route up towards core
    - maintain packet ordering by using same ports for same server
    - Diffuses and spreads out traffic

Prefix	Output port
10.2.0.0/24	0
10.2.1.0/24	1
0.0.0.0/0	

Suffix	Output port
0.0.0.2/8	2
0.0.0.3/8	3

Before Next time

- Project Proposal
  - **due this Friday**
  - Project group meeting Tuesday, 4:15pm, in 122 Gates Hall
  - Meet with groups, TA, and professor
- Lab1
  - Lab1 help session in MEng Lab, Wednesday, Sept 10, during lecture time
  - Single threaded TCP proxy
  - **Due this Friday**

- No required reading and review due
- But, review chapter 5 from the book, Data Link and Physical Layer
  - We will also briefly discuss data center topologies

- Check website for updated schedule

Goals for Today

- Network Layer
  - Abstraction / services
    - Datagram vs Virtual Circuit (VC)
  - Internet Protocol
    - IP Datagram format
    - IP Addressing
    - Hierarchical Routing

- Data Center Topologies
  - FatTree

- Backup Slides
  - DHCP and NAT
  - ICMP and Traceroute
  - IPv6
  - Hierarchical Routing: RIP, OSPF, BGP

DHCP (Dynamic Host Configuration Protocol)

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

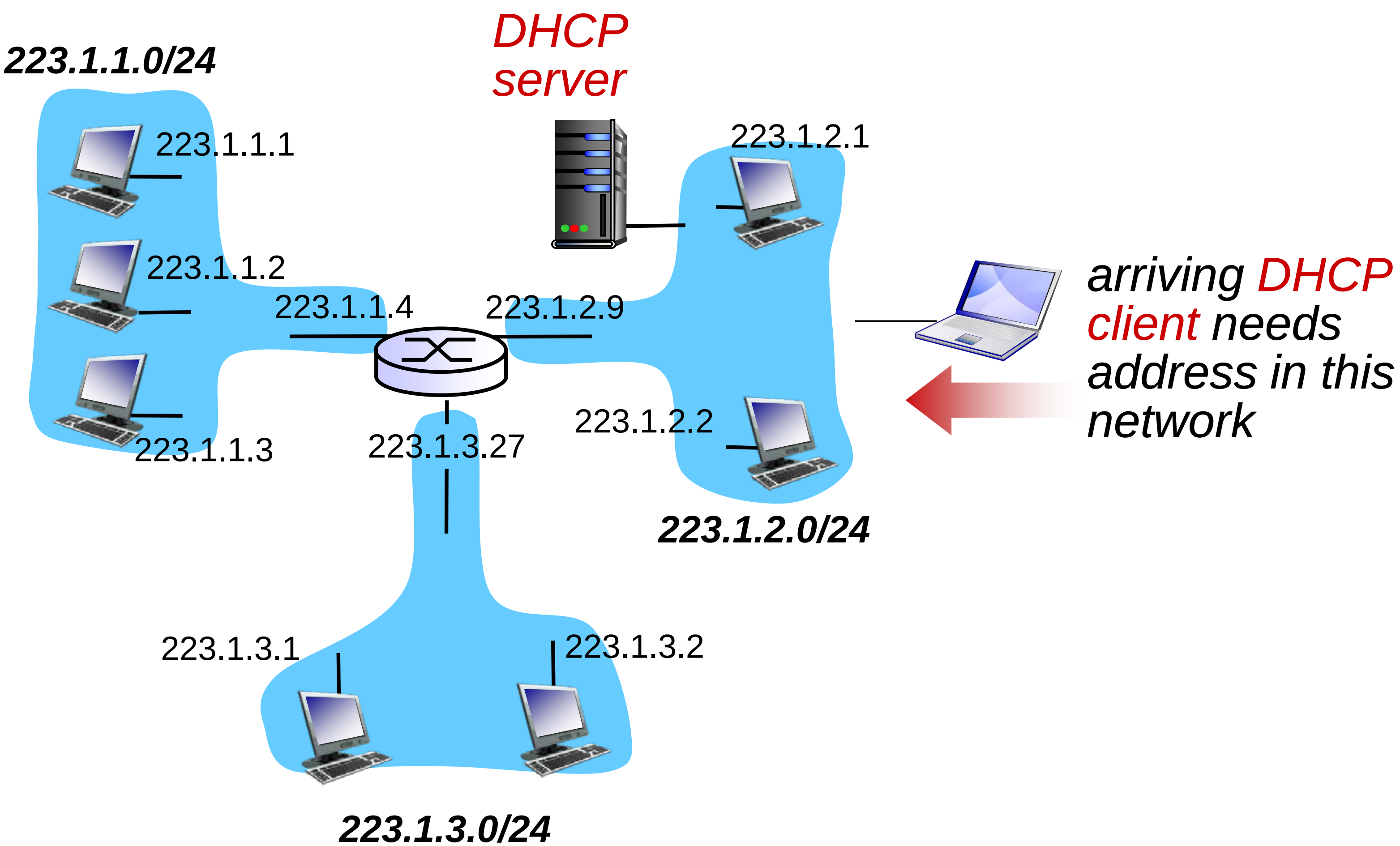
- can renew its lease on address in use
- allows reuse of addresses (only hold address while connected/"on")
- support for mobile users who want to join network (more shortly)

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

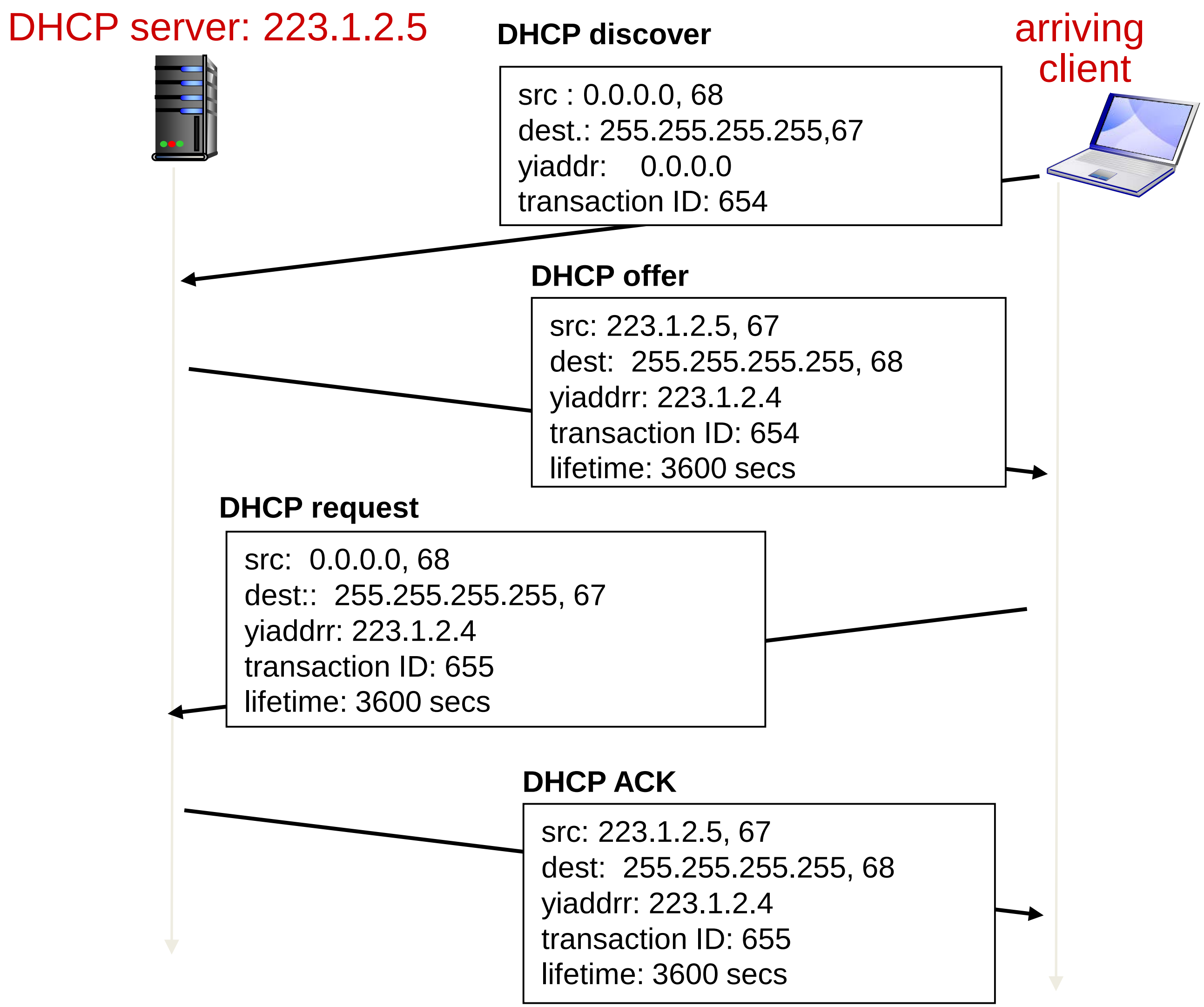
DHCP (Dynamic Host Configuration Protocol)

Client-Server Scenario



DHCP (Dynamic Host Configuration Protocol)

Client-Server Scenario

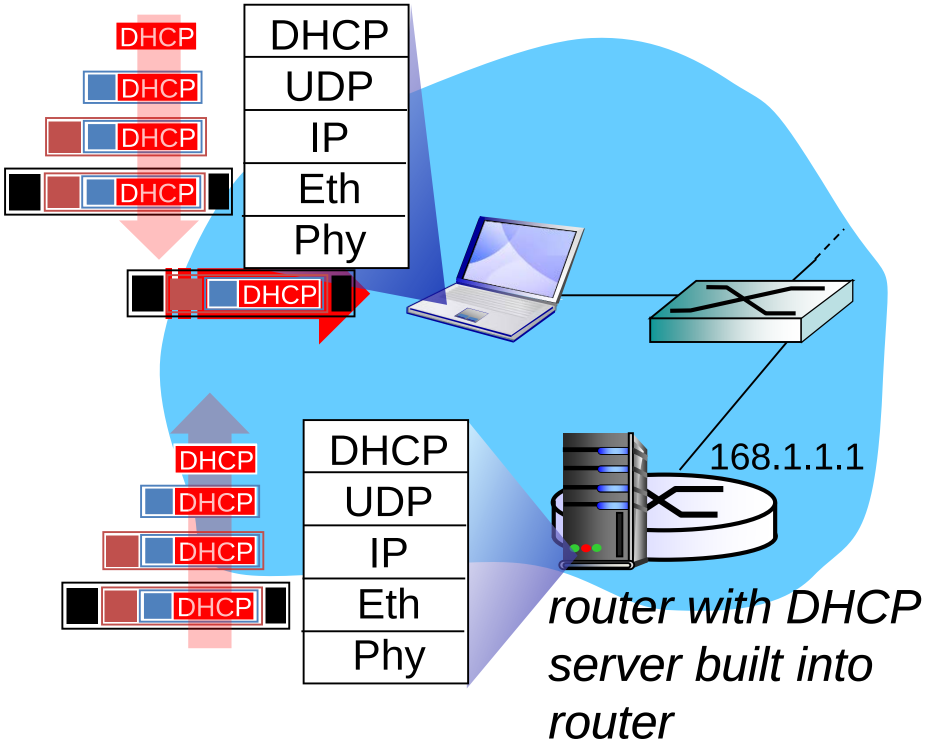


DHCP (Dynamic Host Configuration Protocol)

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)

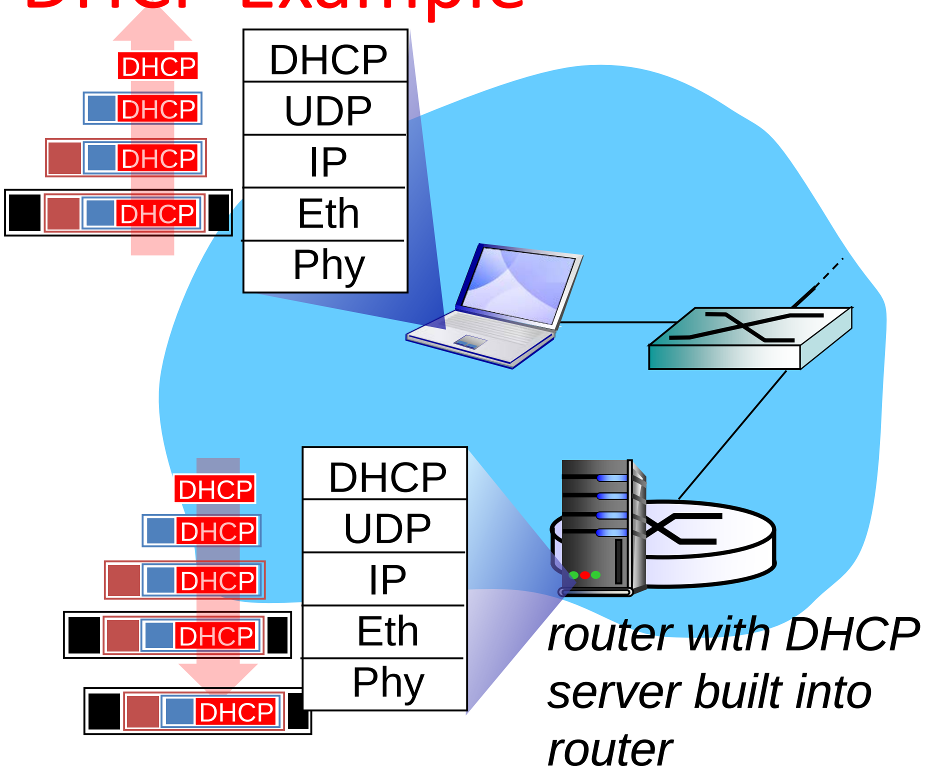
DHCP Example



- connecting laptop needs its IP address, addr of first-hop router, addr of DNS server: use DHCP
- DHCP request encapsulated in UDP, encapsulated in IP, encapsulated in 802.1 Ethernet
- Ethernet frame broadcast (dest: FFFFFFFFFFFF) on LAN, received at router running DHCP server
- Ethernet demuxed to IP demuxed, UDP demuxed to DHCP

DHCP (Dynamic Host Configuration Protocol)

DHCP Example



- DCP server formulates DHCP ACK containing client's IP address, IP address of first-hop router for client, name & IP address of DNS server

- encapsulation of DHCP server, frame forwarded to client, demuxing up to DHCP at client
- client now knows its IP address, name and IP address of DSN server, IP address of its first-hop router

IP Addressing: Hierarchical Addressing

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

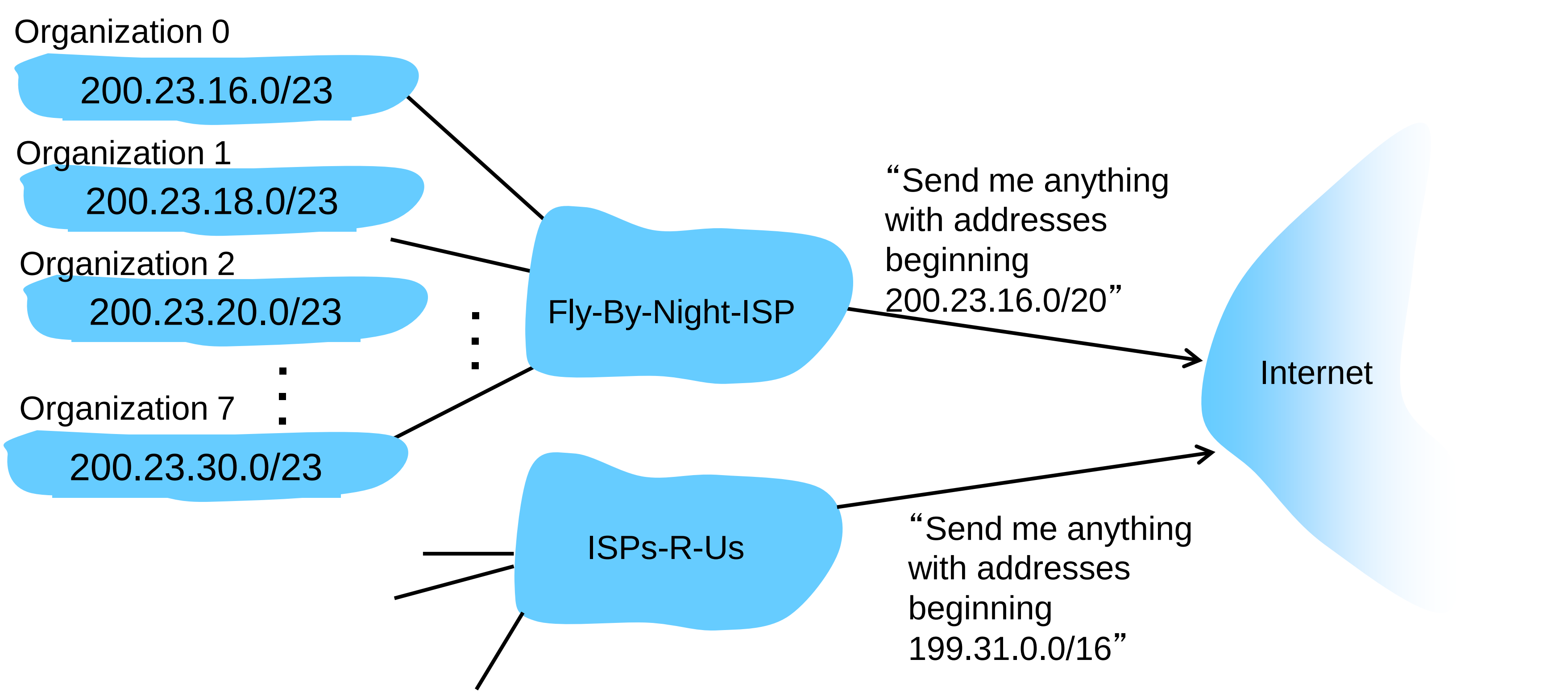
A: gets allocated portion of its provider ISP's address space

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

IP Addressing: Hierarchical Addressing

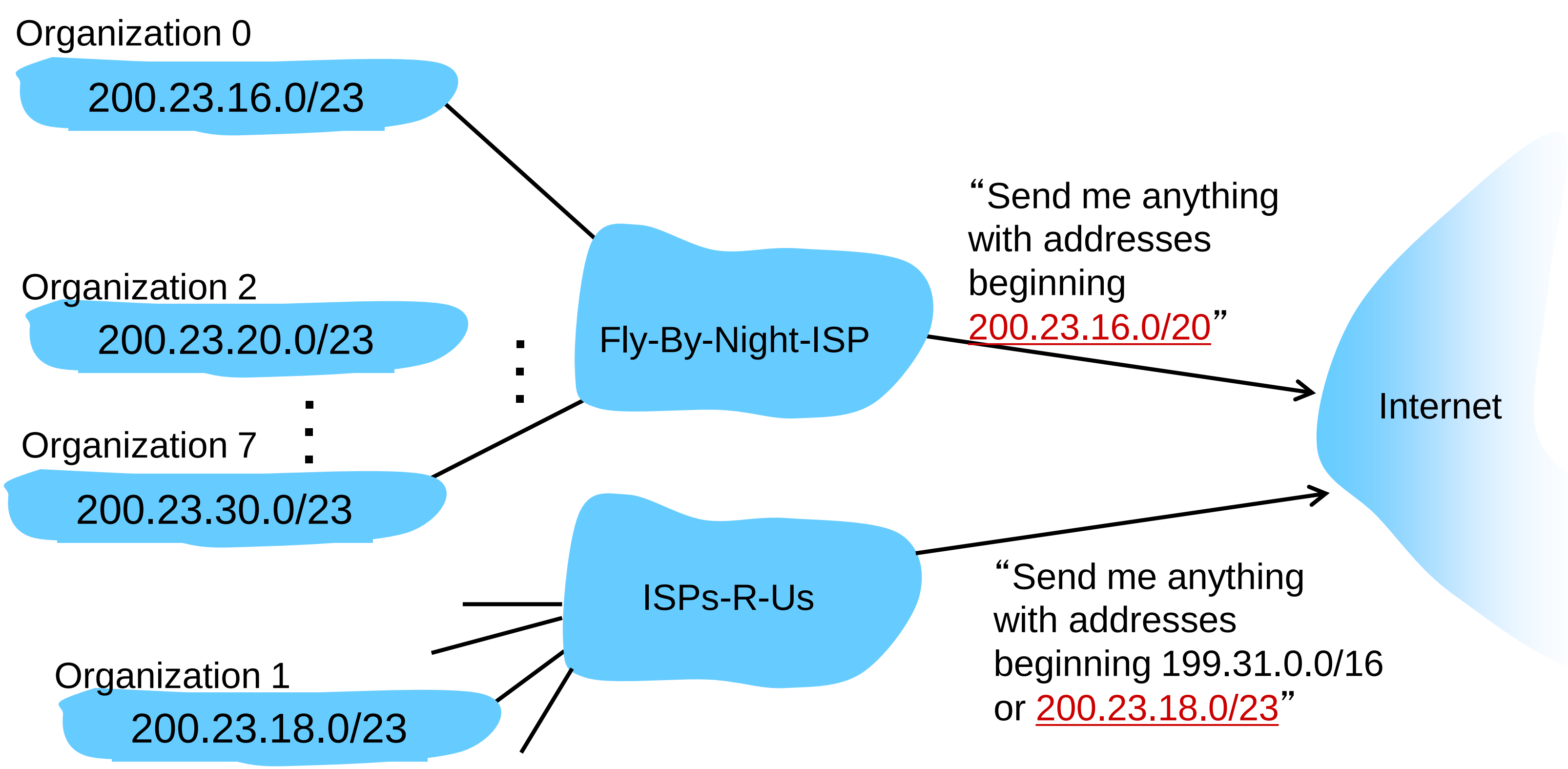
Hierarchical Addressing: Route Aggregation

hierarchical addressing allows efficient advertisement of routing information:



IP Addressing: Hierarchical Addressing

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



IP Addressing: Hierarchical Addressing

Q: how does an ISP get block of addresses?

A: ICANN: Internet Corporation for Assigned Names and Numbers http://www.icann.org/

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

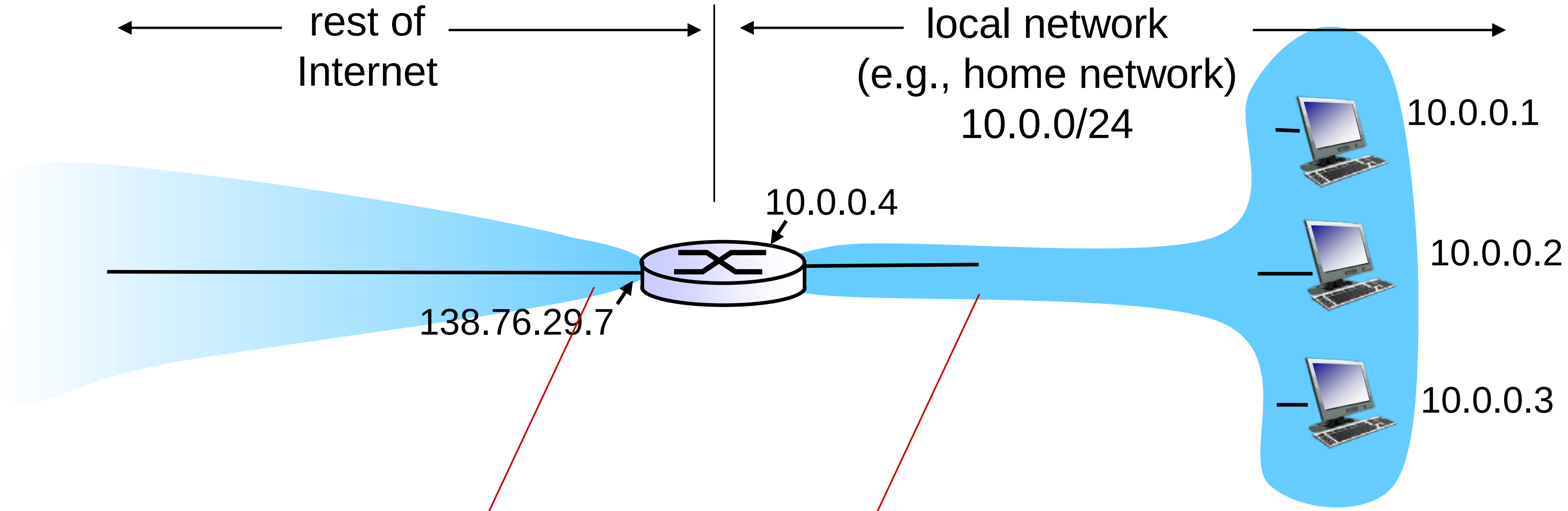
Goals for Today

- Network Layer
  - Abstraction / services
    - Datagram vs Virtual Circuit (VC)
  - Internet Protocol
    - IP Datagram format
    - IP Addressing
    - Hierarchical Routing

- Data Center Topologies
  - FatTree

- Backup Slides
  - DHCP and NAT
  - ICMP and Traceroute
  - IPv6
  - Hierarchical Routing: RIP, OSPF, BGP

NAT (Network Address Translation)



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)

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:

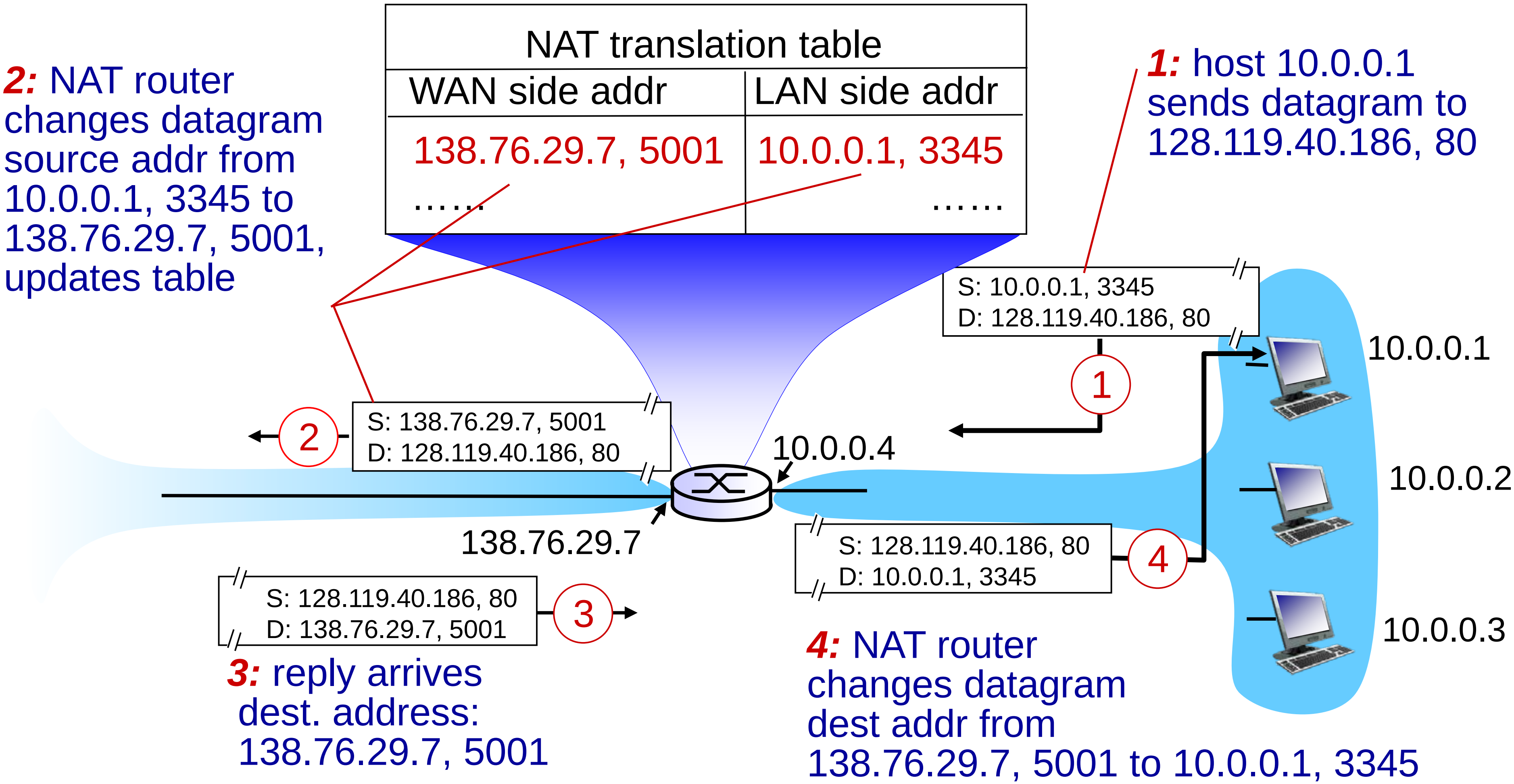
- *outgoing datagrams: replace* (source IP address, port #) of every outgoing datagram to (NAT IP address, new port #)

  . . . 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 (Network Address Translation)



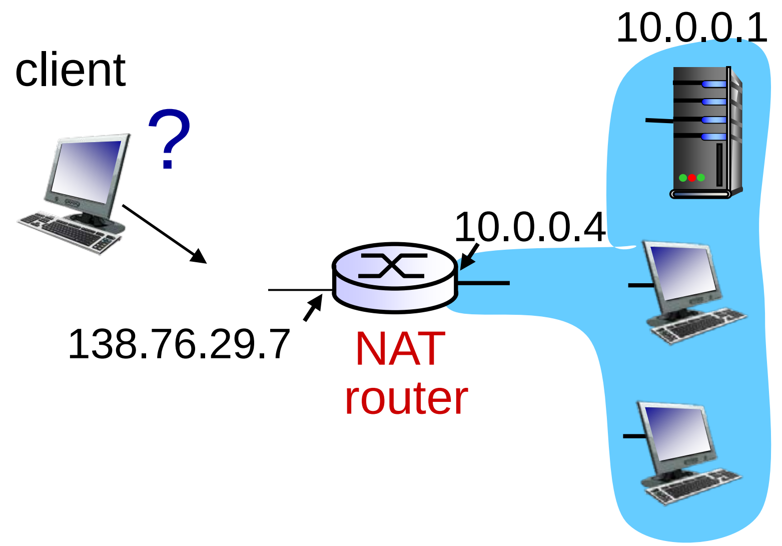
NAT (Network Address Translation)

- 16-bit port-number field:
  - 60,000 simultaneous connections with a single LAN-side address!
- NAT is controversial:
  - routers should only 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

NAT (Network Address Translation)

NAT Traversal Problem

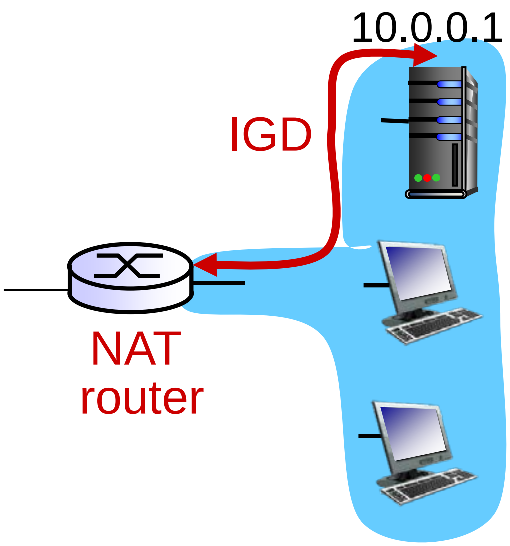
- client wants to connect to server with address 10.0.0.1
  - server address 10.0.0.1 local to LAN (client can't use it as destination addr)
  - only one externally visible NATed address: 138.76.29.7
- *solution1:* statically configure NAT to forward incoming connection requests at given port to server
  - e.g., (123.76.29.7, port 2500) always forwarded to 10.0.0.1 port 25000



NAT (Network Address Translation)

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)

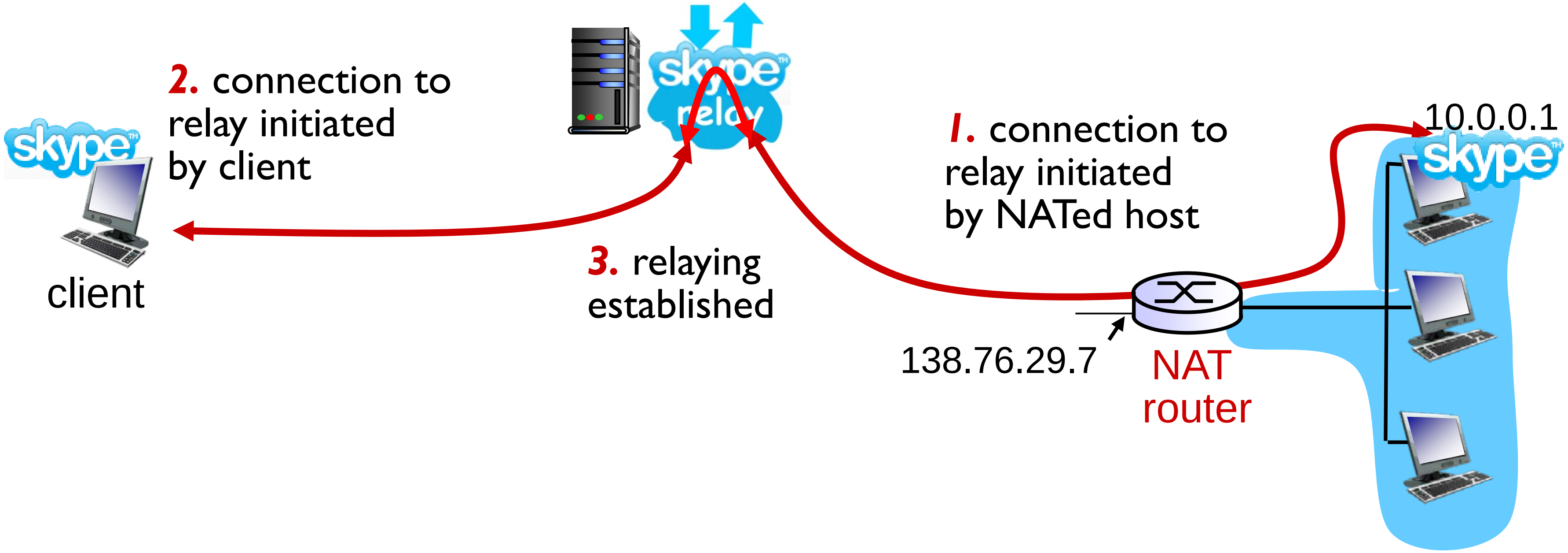


i.e., automate static NAT port map configuration

NAT (Network Address Translation)

NAT Traversal Problem

- *solution 3:* relaying (used in Skype)
  - NATed client establishes connection to relay
  - external client connects to relay
  - relay bridges packets between to connections



Goals for Today

- Network Layer
  - Abstraction / services
    - Datagram vs Virtual Circuit (VC)
  - Internet Protocol
    - IP Datagram format
    - IP Addressing
    - Hierarchical Routing

- Data Center Topologies
  - FatTree

- Backup Slides
  - DHCP and NAT
  - ICMP and Traceroute
  - IPv6
  - Hierarchical Routing: 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

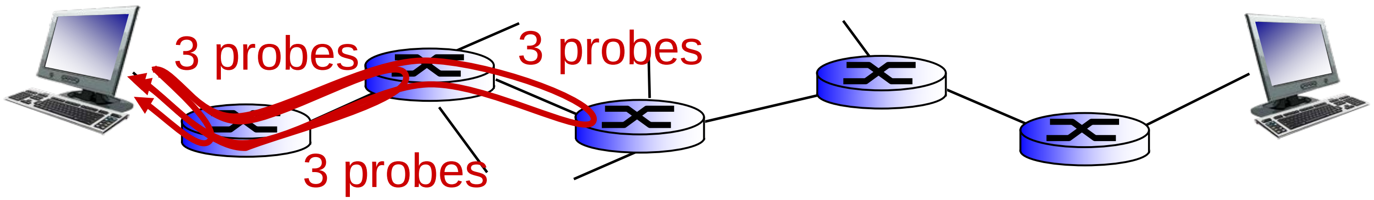
ICMP (Internet control message protocol)

ICMP and Traceroute

- 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



Goals for Today

- Network Layer
  - Abstraction / services
    - Datagram vs Virtual Circuit (VC)
  - Internet Protocol
    - IP Datagram format
    - IP Addressing
    - Hierarchical Routing

- Data Center Topologies
  - FatTree

- Backup Slides
  - DHCP and NAT
  - ICMP and Traceroute
  - IPv6
  - Hierarchical Routing: RIP, OSPF, BGP

IPv6

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

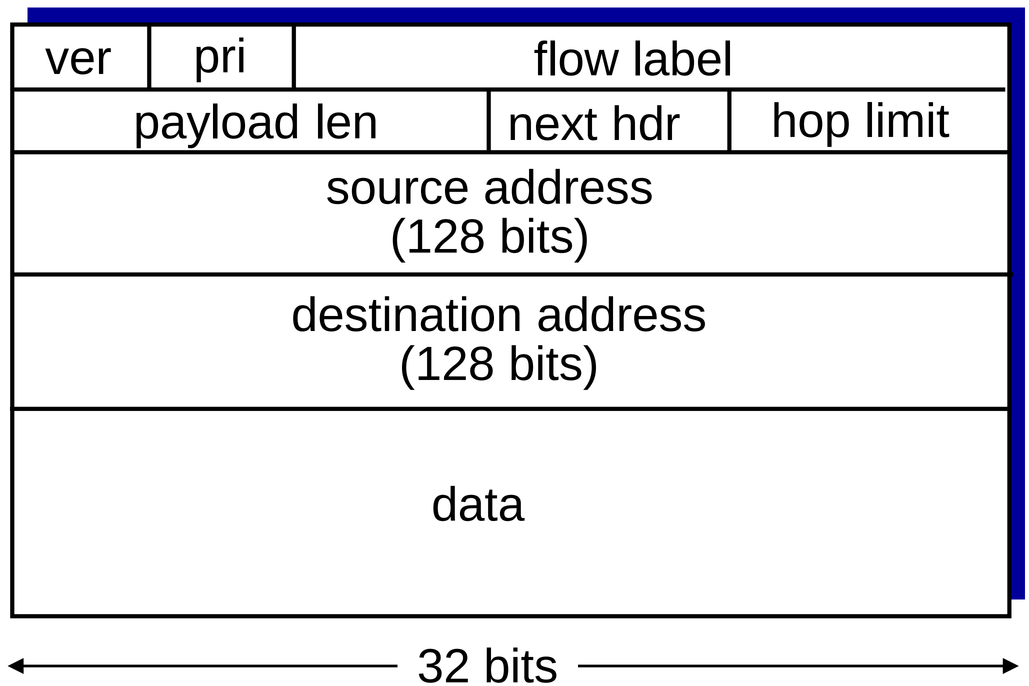
IPv6 datagram format:

- fixed-length 40 byte header
- no fragmentation allowed

IPv6

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



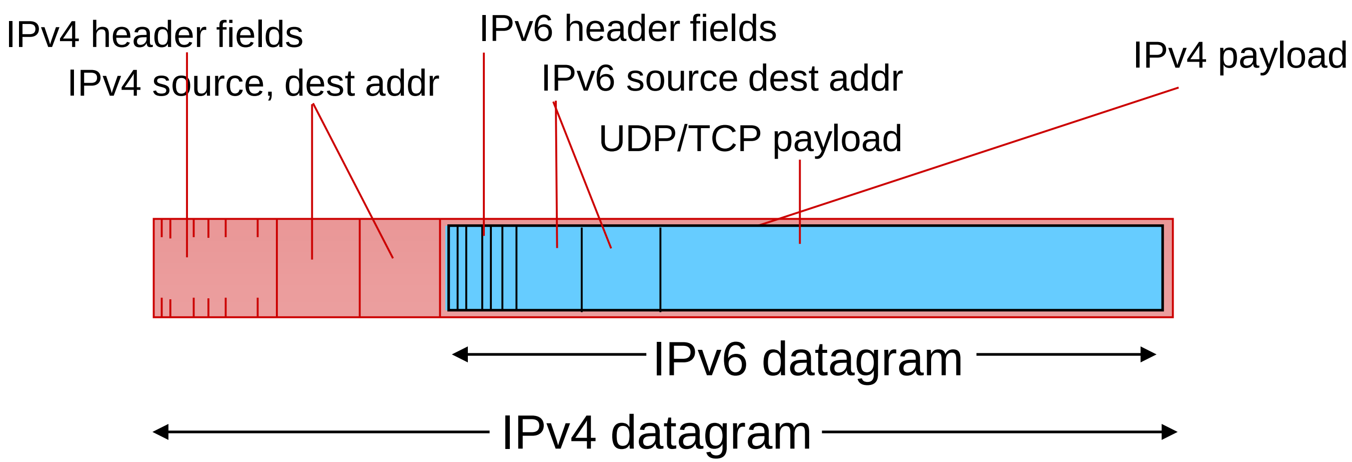
IPv6

Changes from IPv4

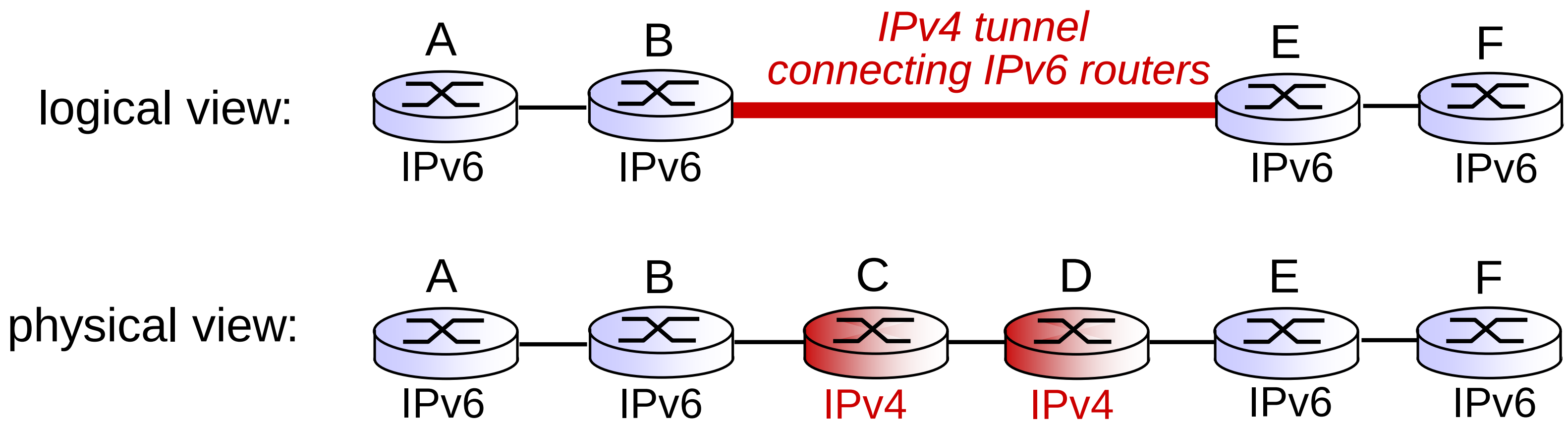
- *checksum*: removed entirely to reduce processing time at each hop
- *options:* allowed, but outside of header, indicated by "Next Header" field
- *ICMPv6:* new version of ICMP
  - additional message types, e.g. "Packet Too Big"
  - multicast group management functions

Transition to IPv6 from IPv4

- 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

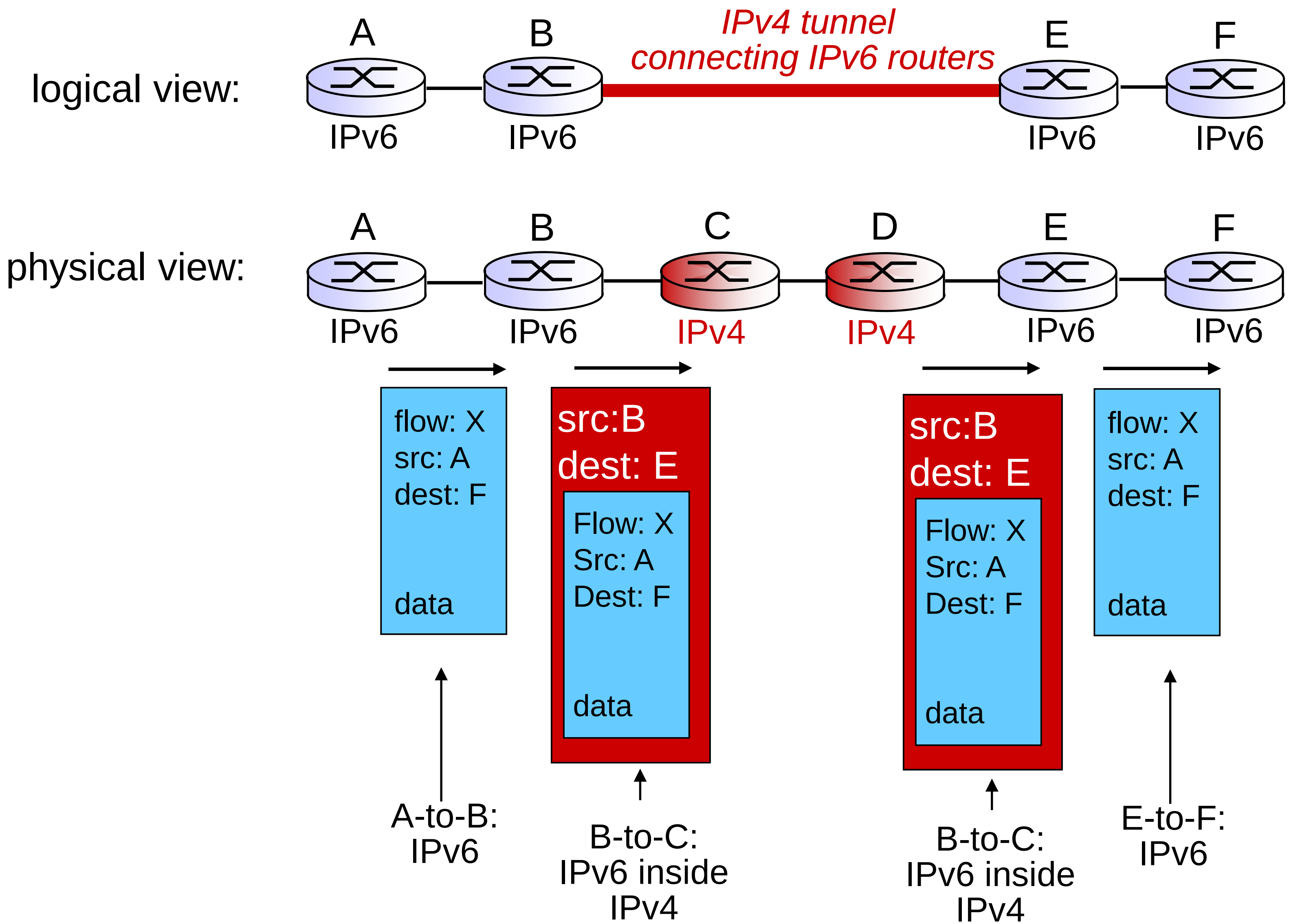


Transition to IPv6 from IPv4 via Tunneling



IPv6

Transition to IPv6 from IPv4 via Tunneling



Goals for Today

- Network Layer
  - Abstraction / services
    - Datagram vs Virtual Circuit (VC)
  - Internet Protocol
    - IP Datagram format
    - IP Addressing
    - Hierarchical Routing

- Data Center Topologies
  - FatTree

- Backup Slides
  - DHCP and NAT
  - ICMP and Traceroute
  - IPv6
  - Hierarchical Routing: 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

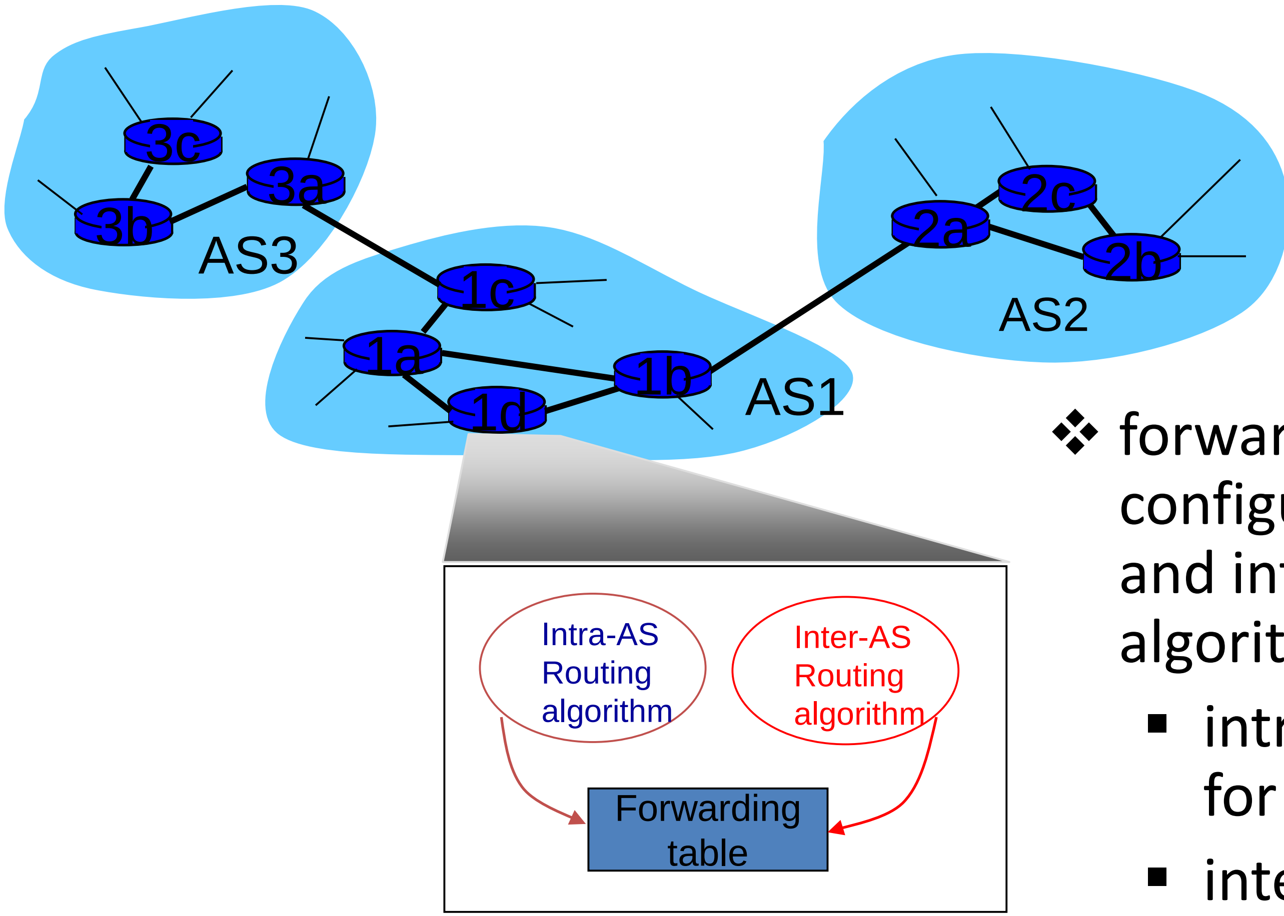
- aggregate routers into regions, "autonomous systems" (AS)
- routers in same AS run same routing protocol
  - "intra-AS" routing protocol
  - routers in different AS can run different intra-AS routing protocol

gateway router:

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

Hierarchical Routing

Interconnected Autonomous Systems (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

Hierarchical Routing

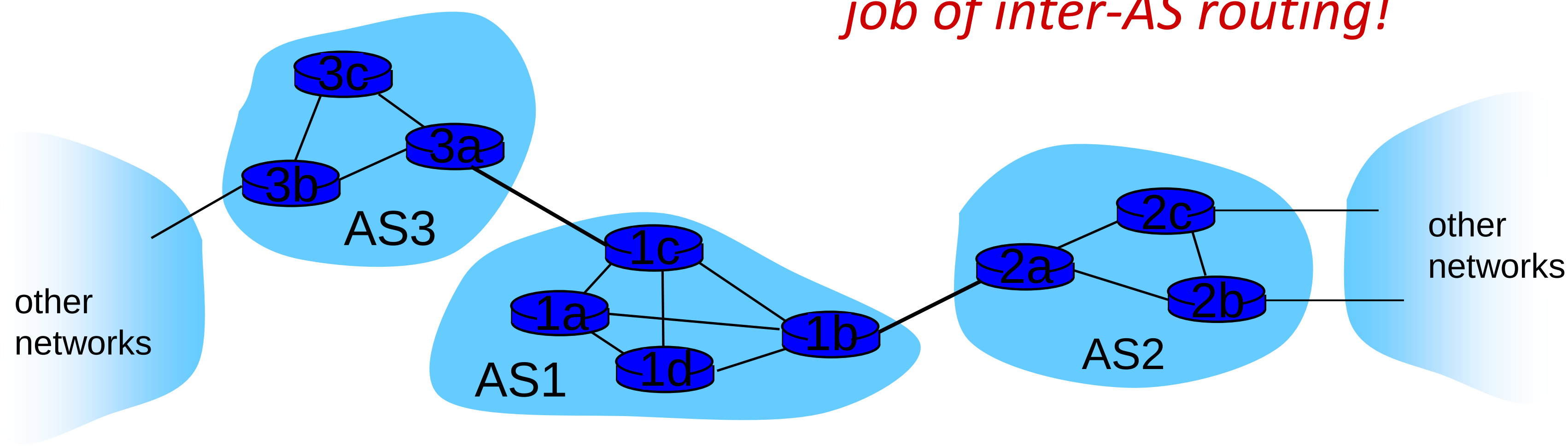
Inter-AS tasks

- 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

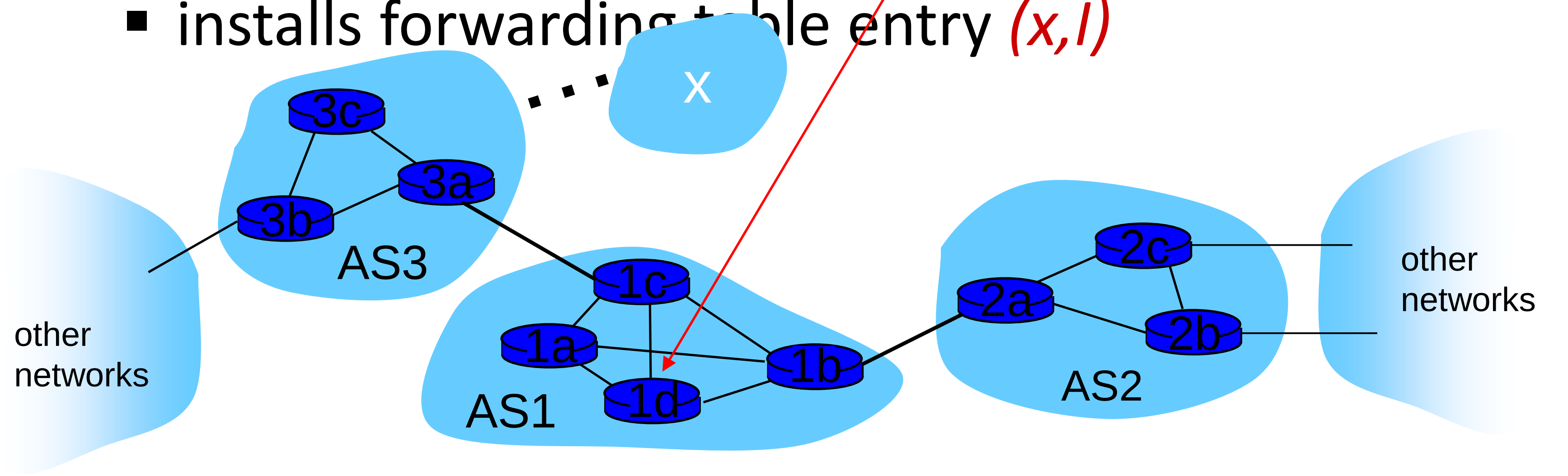
job of inter-AS routing!



Hierarchical Routing

Example: Setting forwarding table in router 1d

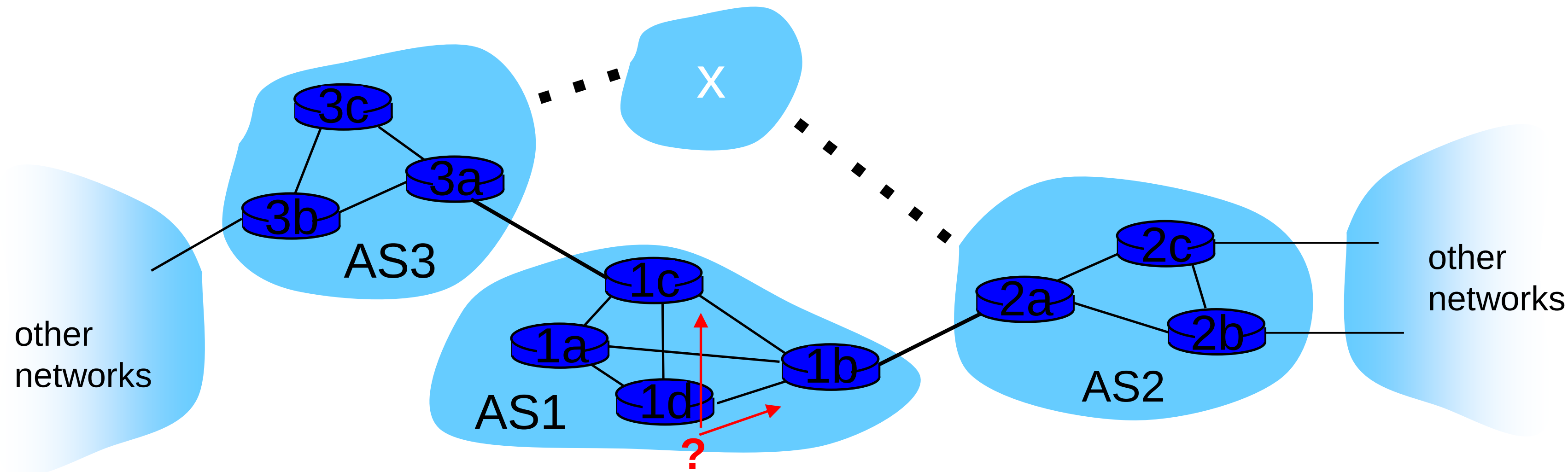
- suppose AS1 learns (via inter-AS protocol) that subnet *x* reachable via AS3 (gateway 1c), but not via AS2
  - inter-AS protocol propagates reachability info to all internal routers
- router 1d determines from intra-AS routing info that its interface *I* is on the least cost path to 1c
  - installs forwarding table entry *(x,I)*



Hierarchical Routing

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!



Hierarchical Routing

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.

learn from inter-AS protocol that subnet *x* is reachable via multiple gateways	→	use routing info from intra-AS protocol to determine costs of least-cost paths to each of the gateways	→	hot potato routing: choose the gateway that has the smallest least cost	→	determine from forwarding table the interface *I* that leads to least-cost gateway. Enter *(x,I)* in forwarding table

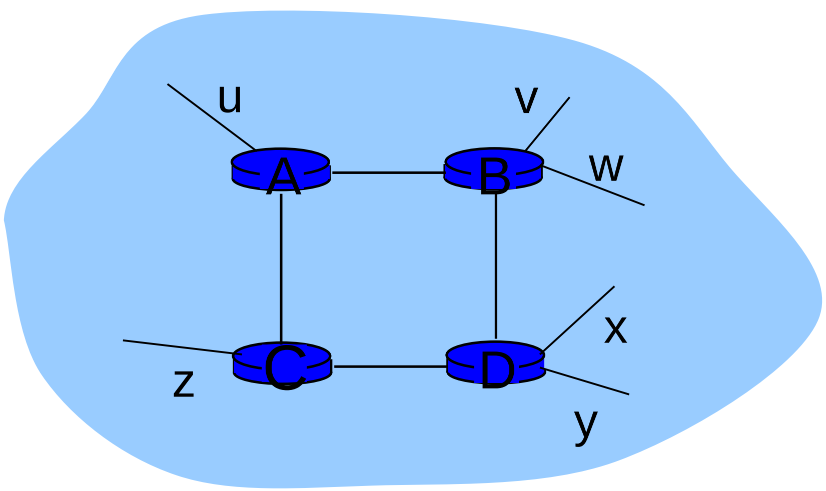
Hierarchical Routing

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)

Intra-AS Routing: 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)*

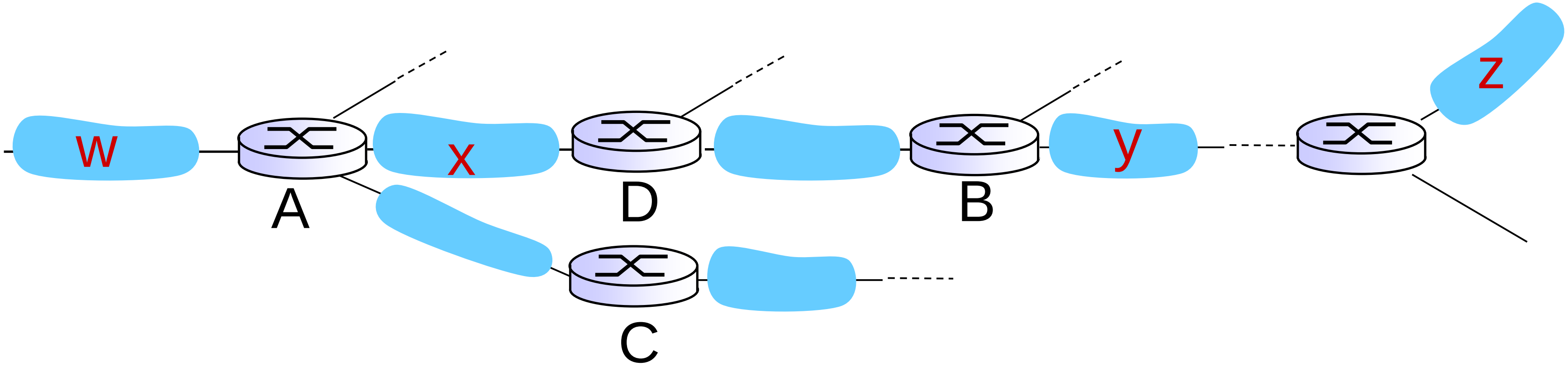


from router A to destination *subnets:*

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

Hierarchical Routing

Intra-AS Routing: RIP (Routing Information Protocol)



routing table in router D

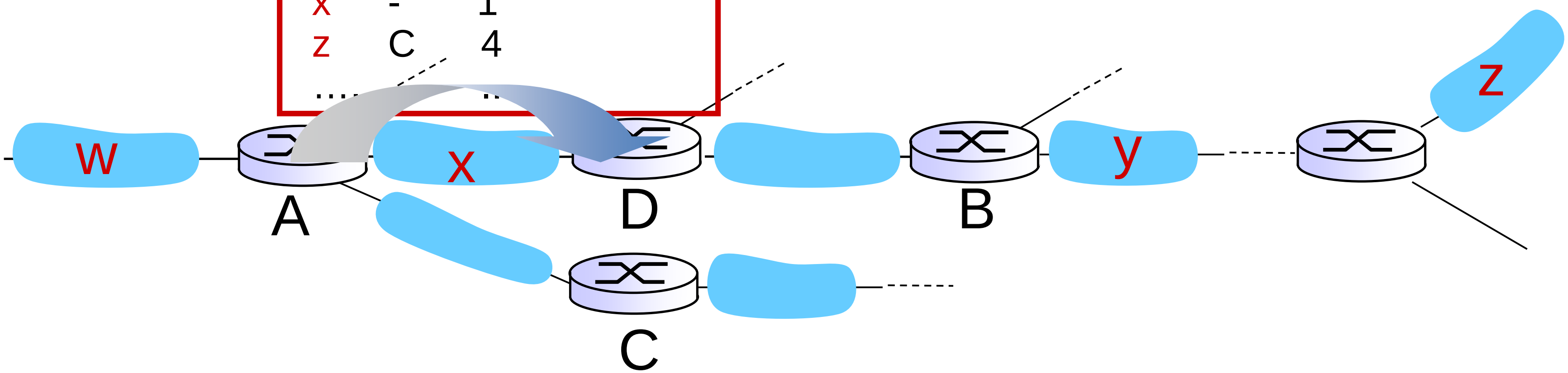
destination subnet	next router	# hops to dest
w	A	2
y	B	2
z	B	7
x	--	1
….	….	….

Hierarchical Routing

Intra-AS Routing: RIP (Routing Information Protocol)

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	7
x	--	1
….	….	….

A

5

Hierarchical Routing

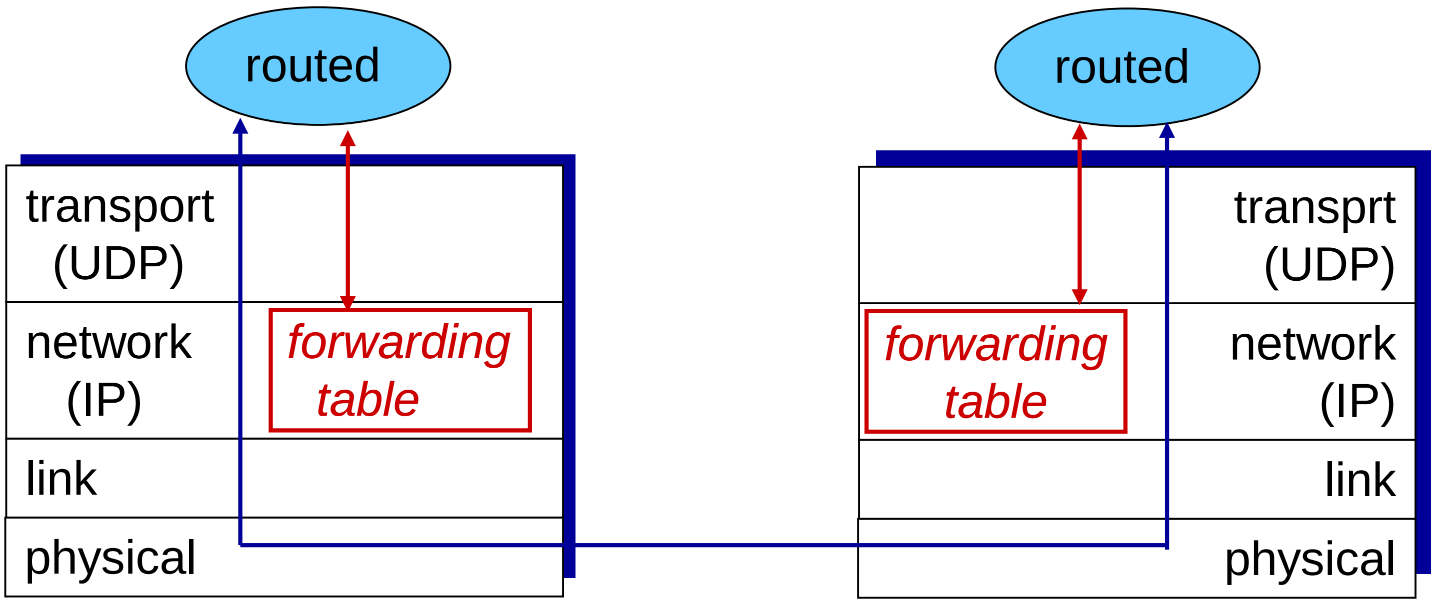
Intra-AS Routing: RIP—Link failure and recovery

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)

Intra-AS Routing: RIP—Table processing

- RIP routing tables managed by *application-level* process called route-d (daemon)
- advertisements sent in UDP packets, periodically repeated



Hierarchical Routing

Intra-AS Routing: 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

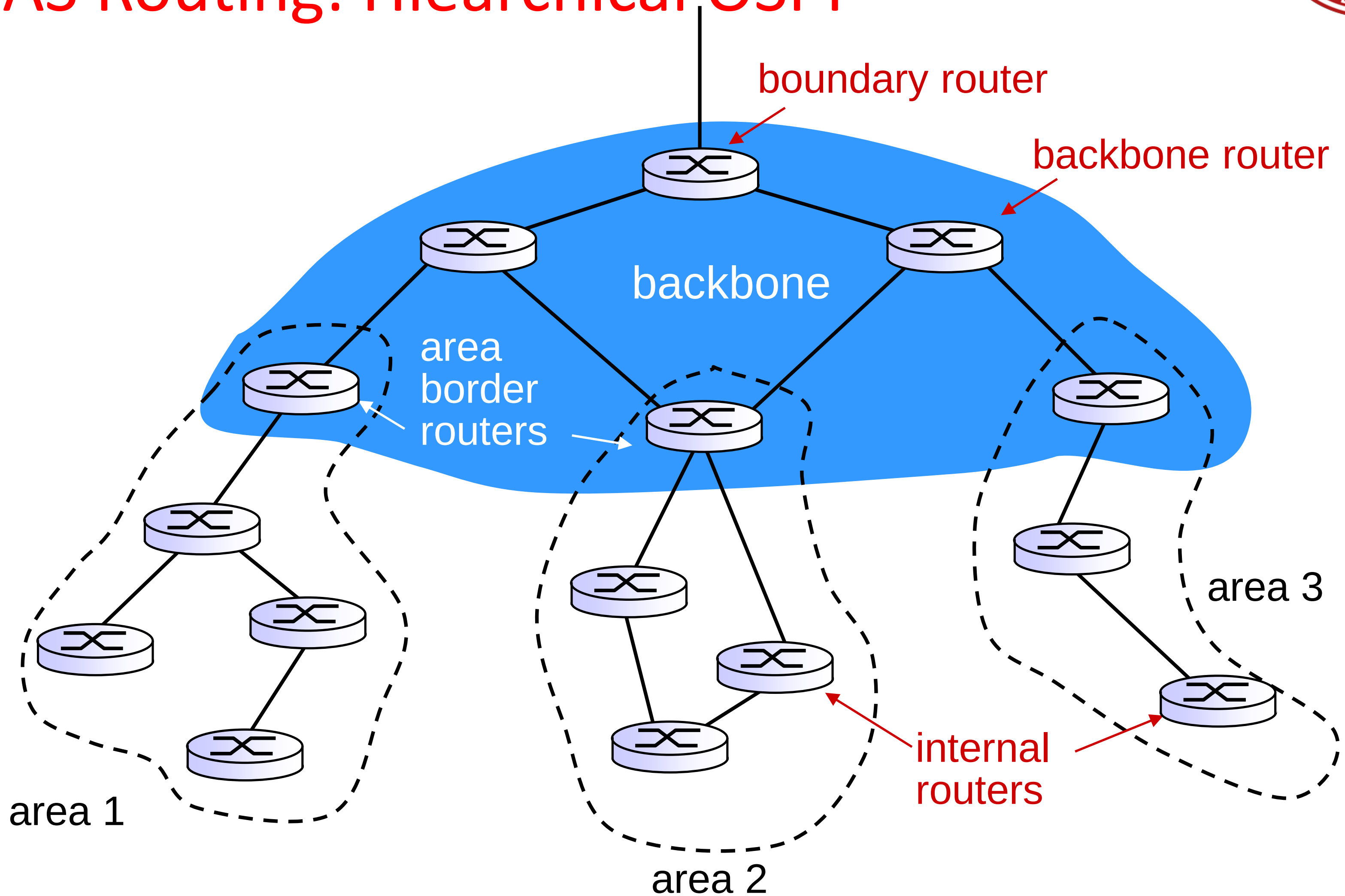
Hierarchical Routing

Intra-AS Routing: 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 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 Routing

Intra-AS Routing: Hiearchical OSPF



Intra-AS Routing: 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.

Inter-AS Routing—BGP

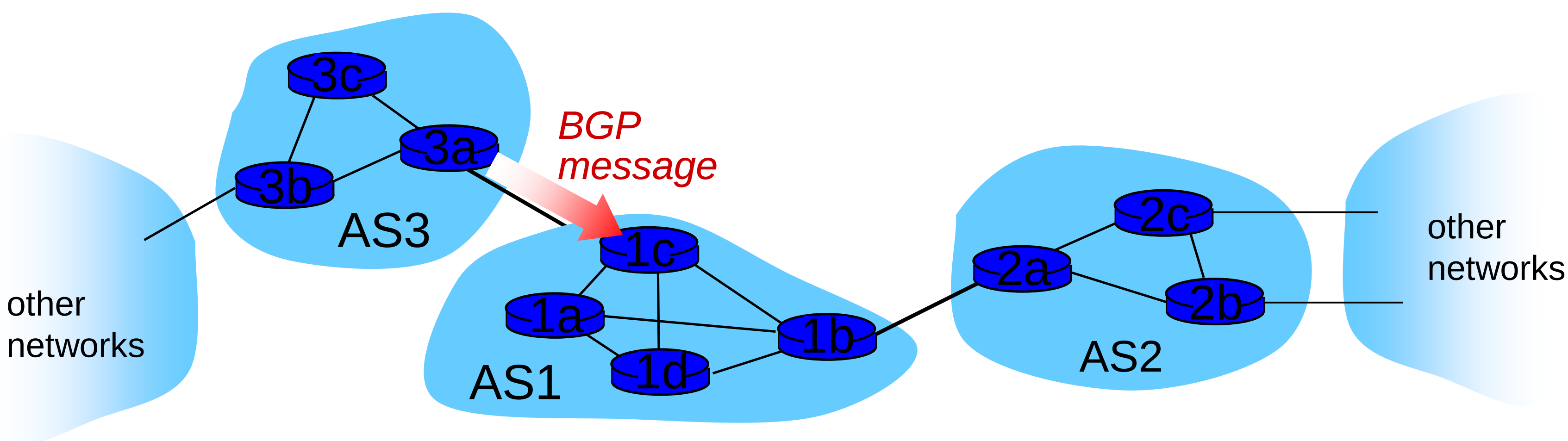
- BGP (Border Gateway Protocol): *the* de facto inter-domain routing protocol
  - "glue that holds the Internet together"
- BGP provides each AS a means to:
  - eBGP: obtain subnet reachability information from neighboring ASs.
  - iBGP: propagate reachability information to all AS-internal routers.
  - 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"*

Hierarchical Routing

Inter-AS Routing—BGP

- **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

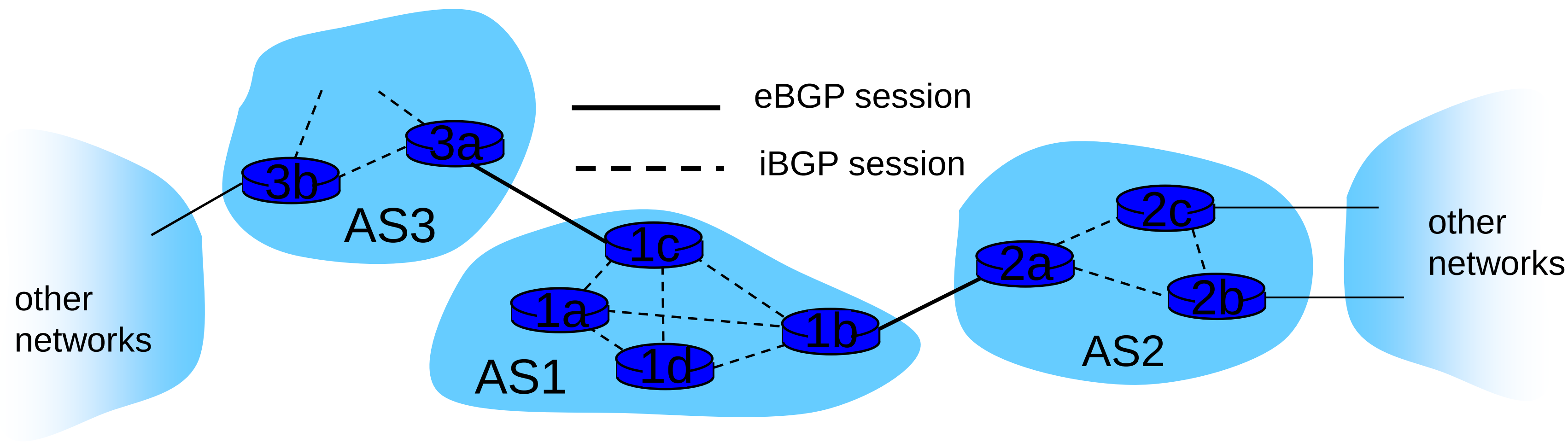
- when AS3 advertises a prefix to AS1:
  - AS3 *promises* it will forward datagrams towards that prefix
  - AS3 can aggregate prefixes in its advertisement



Hierarchical Routing

Inter-AS Routing—BGP 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.



Hierarchical Routing

Inter-AS Routing—BGP routes and Path attributes

- 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

Inter-AS Routing—BGP Route Selection

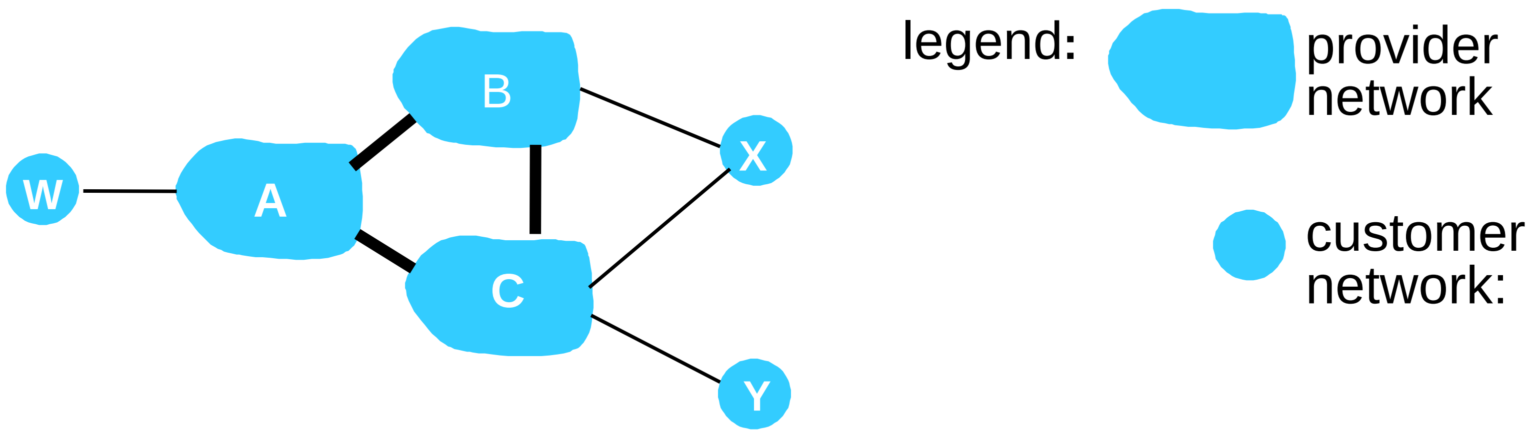
- router may learn about more than 1 route to destination AS, selects route based on:
  1. local preference value attribute: policy decision
  2. shortest AS-PATH
  3. closest NEXT-HOP router: hot potato routing
  4. additional criteria

Inter-AS Routing—BGP Messages

- BGP messages exchanged between peers over TCP connection
- BGP messages:
  - OPEN: opens TCP connection to peer and authenticates sender
  - UPDATE: advertises new path (or withdraws old)
  - KEEPALIVE: keeps connection alive in absence of UPDATES; also ACKs OPEN request
  - NOTIFICATION: reports errors in previous msg; also used to close connection

Hierarchical Routing

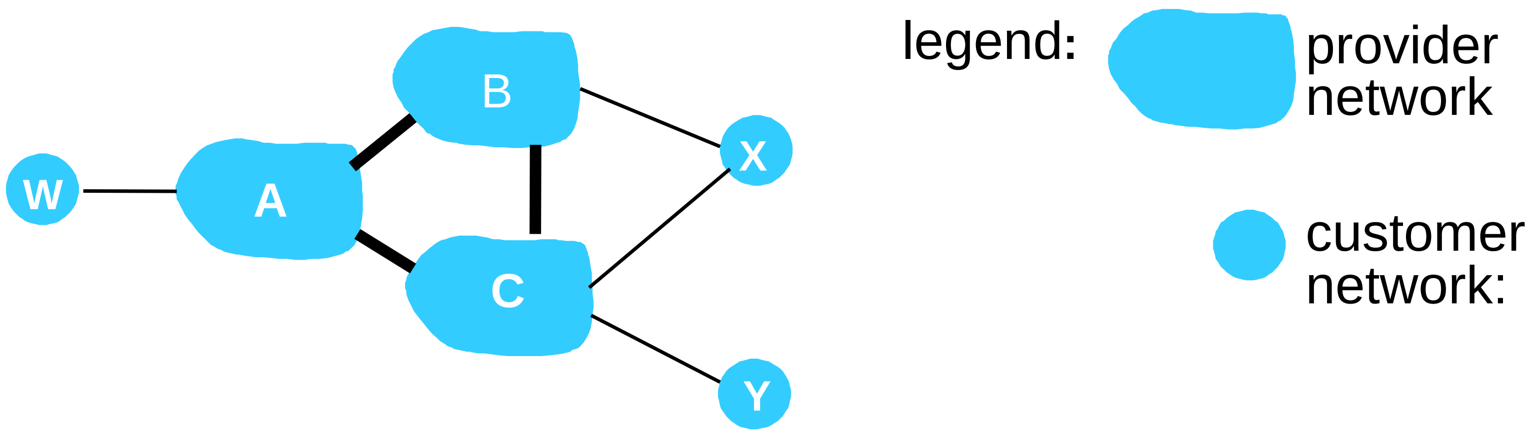
Inter-AS Routing—BGP Routing Policy



- A,B,C are *provider networks*
- X,W,Y are customer (of provider networks)
- X is *dual-homed:* attached to two networks
  - X does not want to route from B via X to C
  - .. so X will not advertise to B a route to C

Hierarchical Routing

Inter-AS Routing—BGP Routing Policy



- A advertises path AW to B
- B advertises path BAW to X
- Should B advertise path BAW to C?
  - No way! B gets no "revenue" for routing CBAW since neither W nor C are B's customers
  - B wants to force C to route to w via A
  - B wants to route *only* to/from its customers!

Hierarchical Routing

Intra- vs 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