FCM 742 – Network Security

Network Layer

John Jay D4CS Program
Spring 2015

slides provided by Prof. Jim Kurose

1

Chapter 4: Network Layer

Chapter goals:

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

4-2

Chapter 4: Network Layer

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

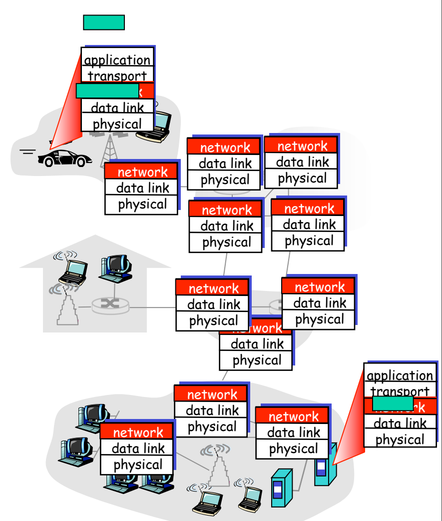
4.5 Routing algorithms
- Link state
- Distance Vector
- Hierarchical routing

4.6 Routing in the Internet
- RIP
- OSPF
- BGP

4.7 Broadcast and multicast routing

Network layer

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



Two Key Network-Layer Functions

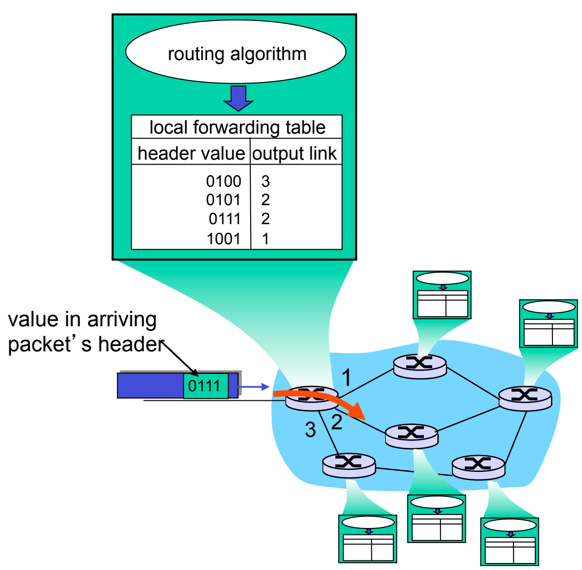
- *forwarding:* move packets from router's input to appropriate router output
- *routing:* determine route taken by packets from source to dest.
  - *routing algorithms*

analogy:

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

4-5

Interplay between routing and forwarding



4-6

Chapter 4: Network Layer

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
- BGP

4.7 Broadcast and multicast routing

Chapter 4: Network Layer

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

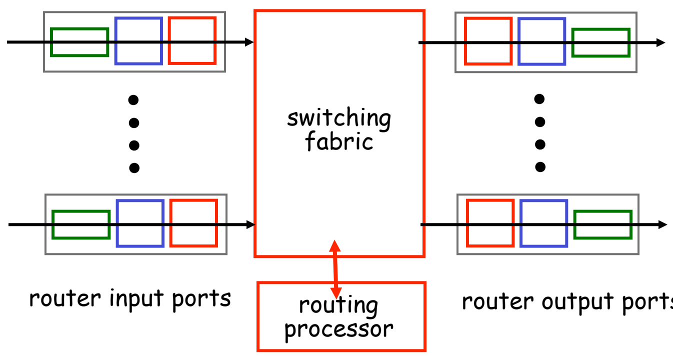
4.6 Routing in the Internet
- RIP
- OSPF
- BGP

4.7 Broadcast and multicast routing

Router Architecture Overview

two key router functions:

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



4-9

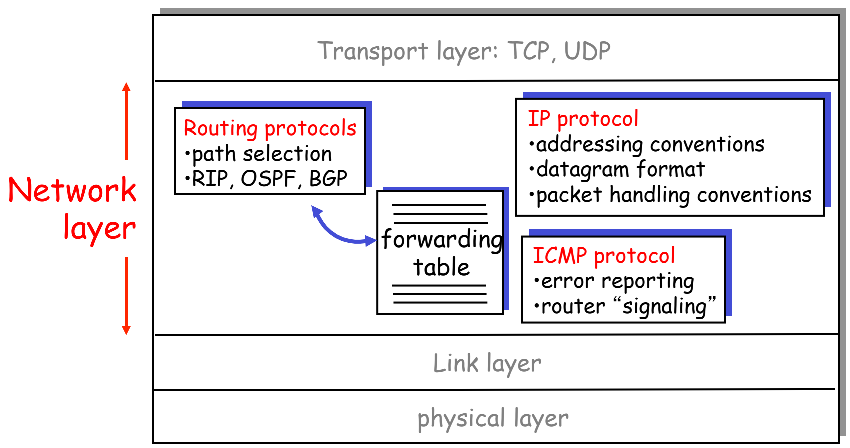
Chapter 4: Network Layer

- 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
    - BGP
- 4.7 Broadcast and multicast routing

4-10

The Internet Network layer

Host, router network layer functions:



4-11

Chapter 4: Network Layer

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
- BGP

4.7 Broadcast and multicast routing

4-12

IP datagram format

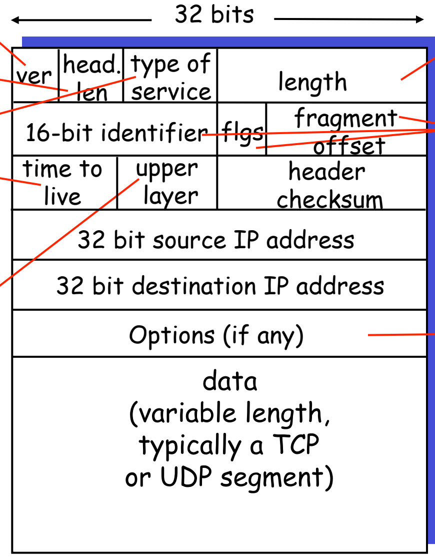
IP protocol version number

header length (bytes)

"type" of data

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.

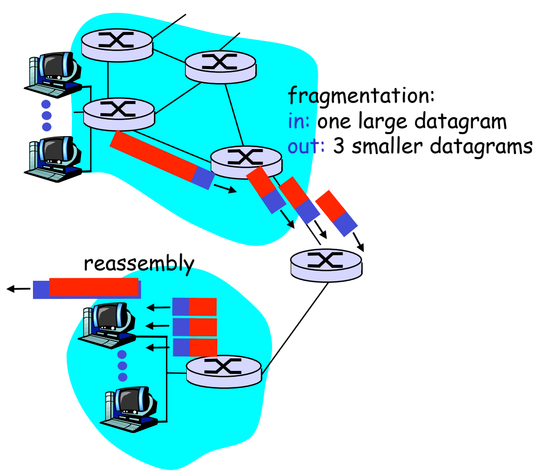
how much overhead with TCP?

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

4-13

IP Fragmentation & Reassembly

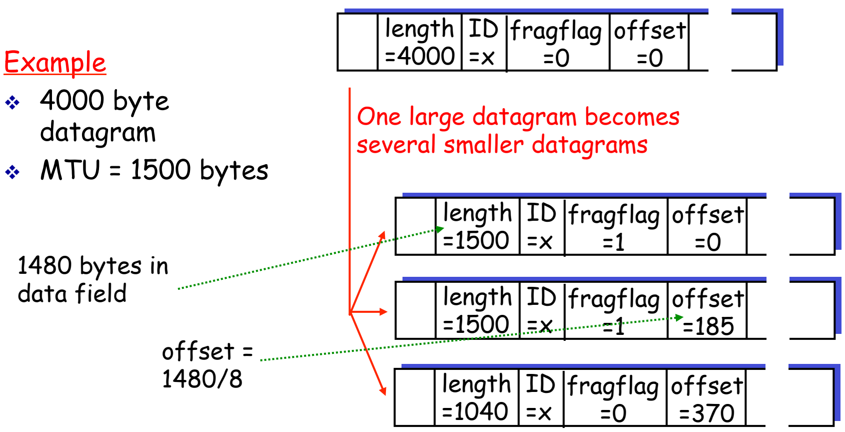
- network links have MTU (max.transfer size) - largest possible link-level frame.
  - different link types, different MTUs
- large IP datagram divided ("fragmented") within net
  - one datagram becomes several datagrams
  - "reassembled" only at final destination
  - IP header bits used to identify,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



4-14

IP Fragmentation and Reassembly

Example
- 4000 byte datagram
- MTU = 1500 bytes



4-15

Chapter 4: Network Layer

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

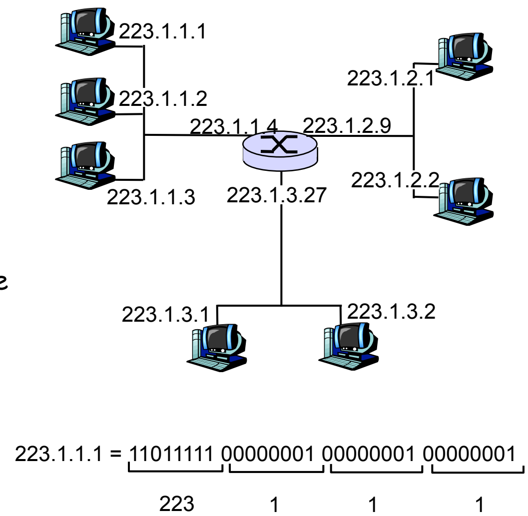
4.6 Routing in the Internet
- RIP
- OSPF
- BGP

4.7 Broadcast and multicast routing

4-16

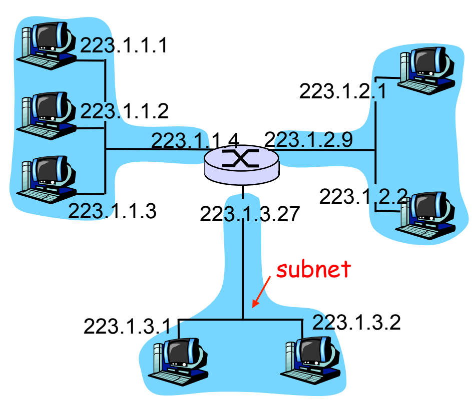
IP Addressing: introduction

- **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 interface
  - IP addresses associated with each interface



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

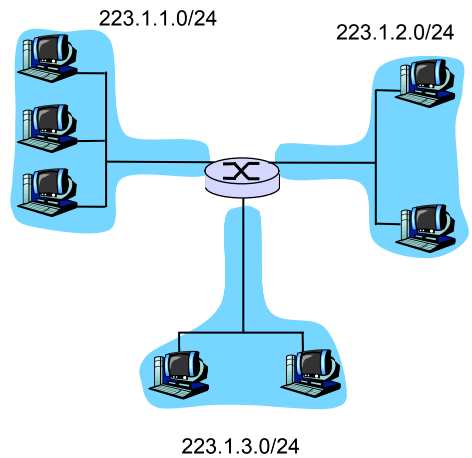


network consisting of 3 subnets

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.

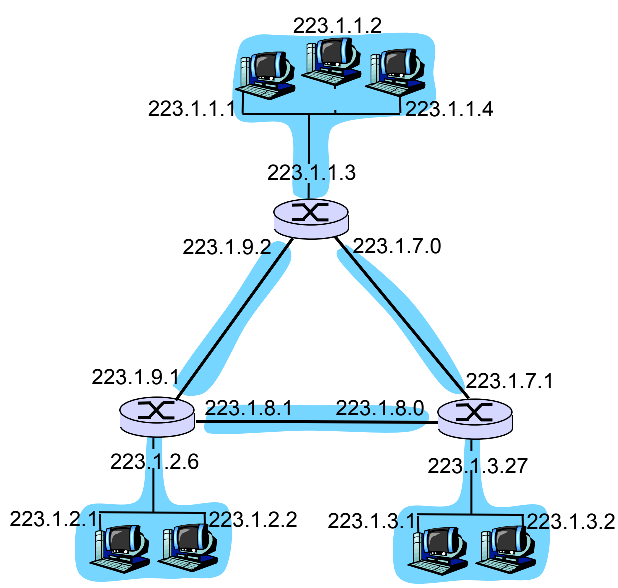


Subnet mask: /24

4-19

Subnets

How many?



4-20

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

4-21

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"

4-22

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 an "on")

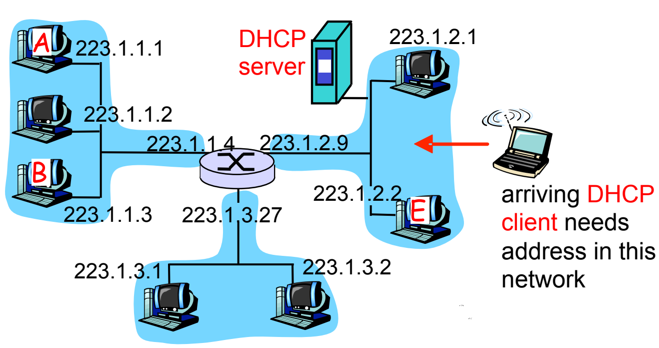
Support for mobile users who want to join network (more shortly)

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

4-23

DHCP client-server scenario



4-24

DHCP: more than 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)

4-25

IP addresses: how to get one?

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

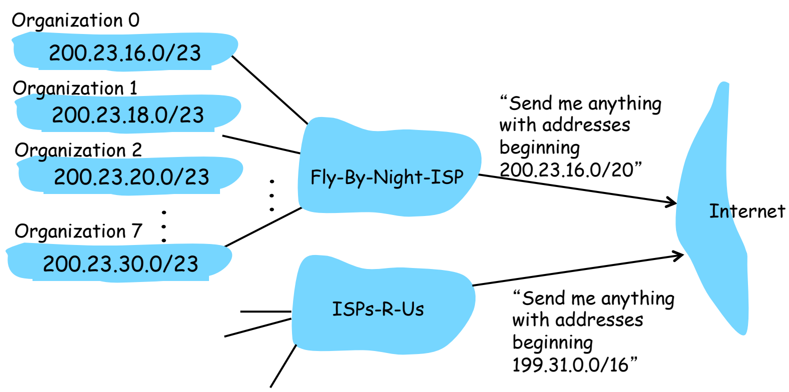
A: gets allocated portion of its provider ISP's address space

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

4-26

Hierarchical addressing: route aggregation

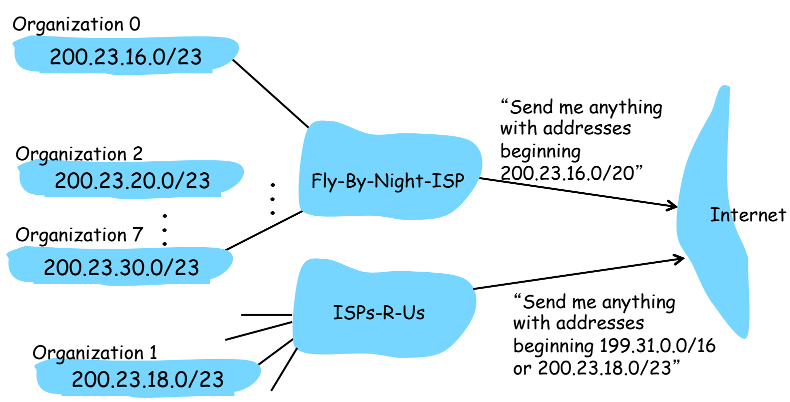
Hierarchical addressing allows efficient advertisement of routing information:



4-27

Hierarchical addressing: more specific routes

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



4-28

IP addressing: the last word...

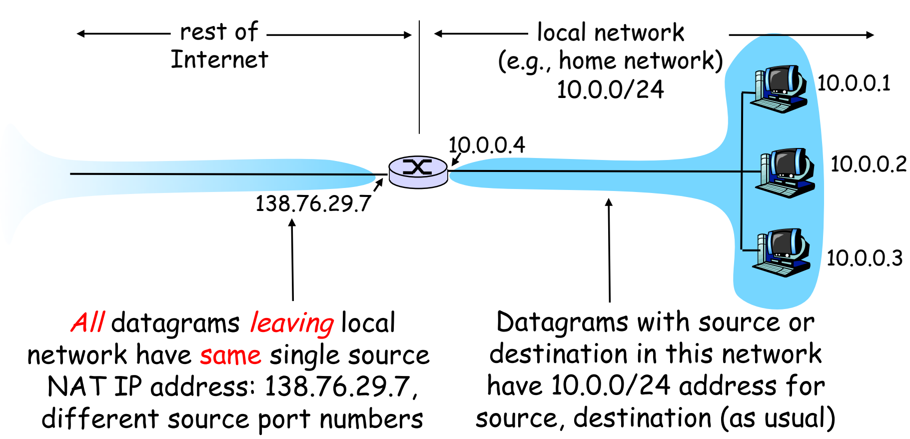
Q: How does an ISP get block of addresses?

A: ICANN: Internet Corporation for Assigned Names and Numbers

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

4-29

NAT: Network Address Translation



4-30

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).

4-31

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 #)

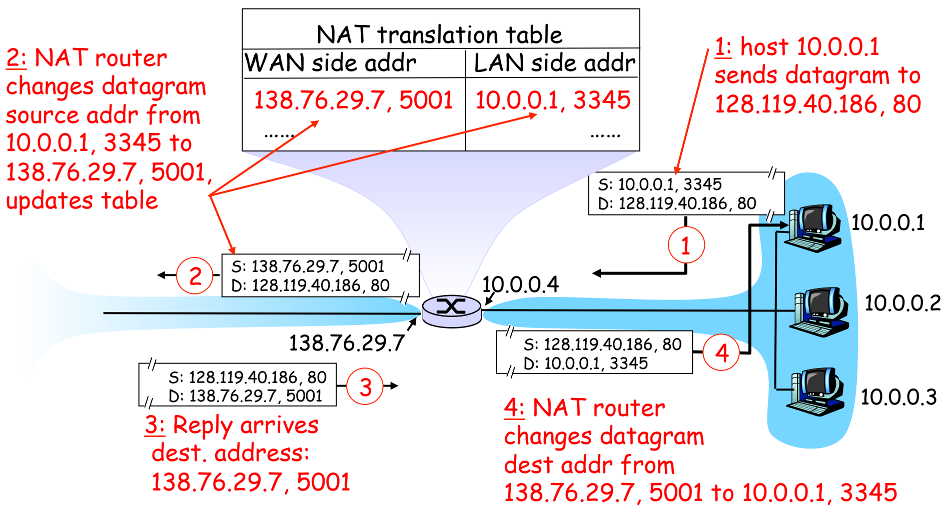
  . . . remote clients/servers will respond using (NAT IP address, new port #) as destination addr.

- *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

4-32

NAT: Network Address Translation



4-33

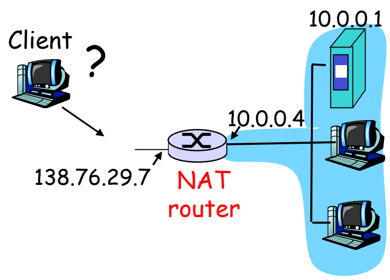
NAT: Network Address Translation

- 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

4-34

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
- solution 1: 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 2500

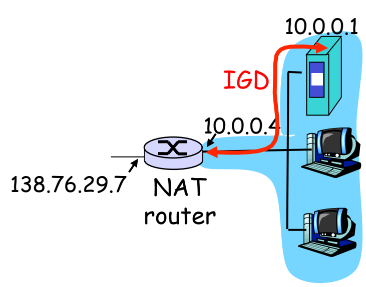


4-35

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)

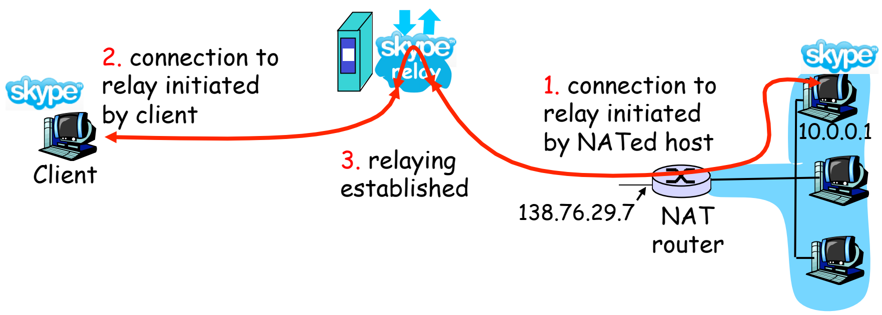
i.e., automate static NAT port map configuration



4-36

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



4-37

Chapter 4: Network Layer

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
- BGP

4.7 Broadcast and multicast routing

4-38

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

4-39

Traceroute and ICMP

- Source sends series of UDP segments to dest
  - first has TTL =1
  - second has TTL=2, etc.
  - unlikely port number
- When nth datagram arrives to nth router:
  - router discards datagram
  - and sends to source an ICMP message (type 11, code 0)
  - ICMP message includes name of router & IP address
- when ICMP message arrives, source calculates RTT
- traceroute does this 3 times

Stopping criterion

- UDP segment eventually arrives at destination host
- destination returns ICMP "port unreachable" packet (type 3, code 3)
- when source gets this ICMP, stops.

4-40

Chapter 4: Network Layer

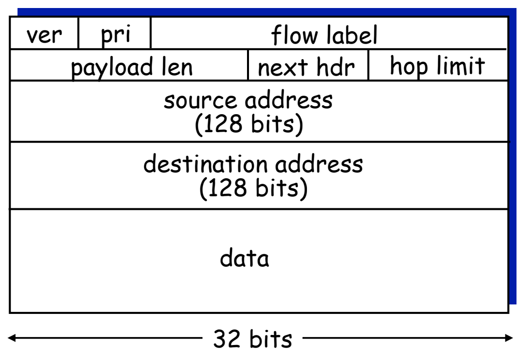
IPv6

- 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 Header (Cont)

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



4-43

Other Changes from IPv4

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

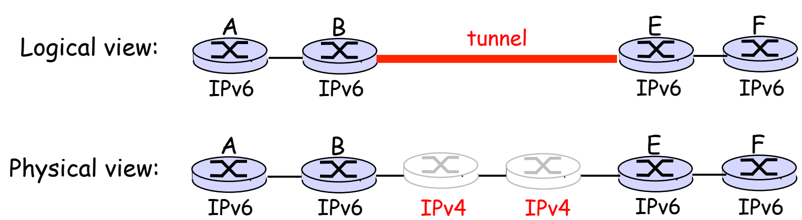
4-44

Transition From IPv4 To IPv6

- Not all routers can be upgraded simultaneous
  - no "flag days"
  - How will the network operate with mixed IPv4 and IPv6 routers?
- *Tunneling:* IPv6 carried as payload in IPv4 datagram among IPv4 routers

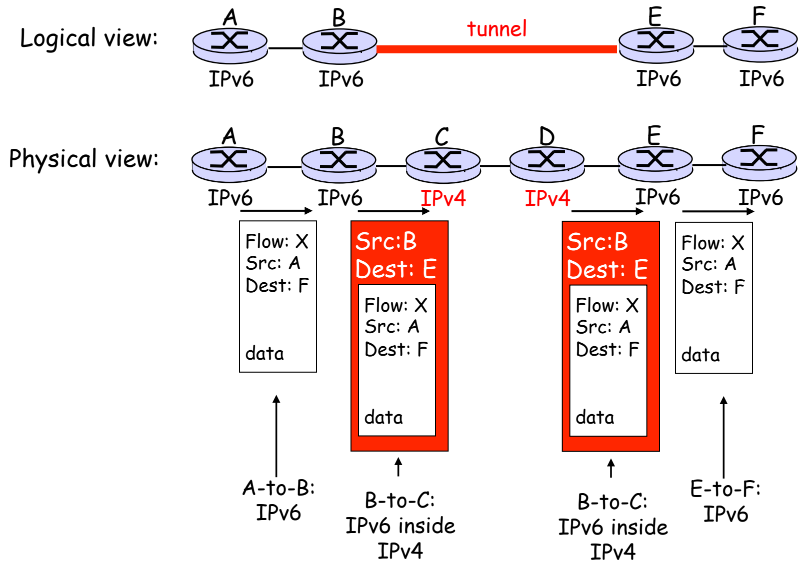
4-45

Tunneling



4-46

Tunneling



4-47

Chapter 4: Network Layer

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
- BGP

4.7 Broadcast and multicast routing

4-48

Chapter 4: Network Layer

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

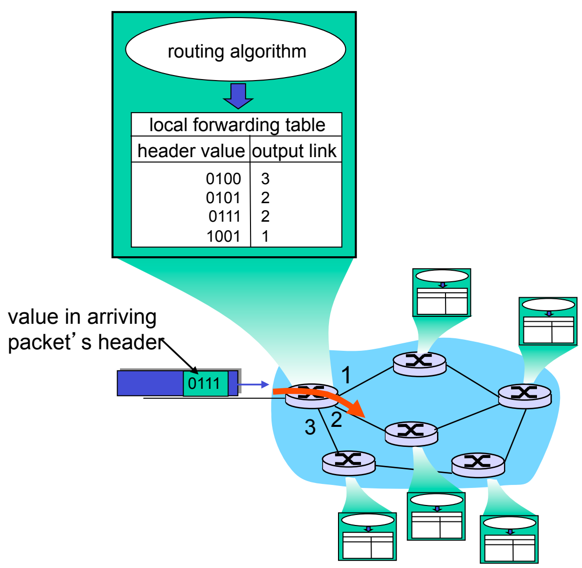
4.5 Routing algorithms
- Link state
- Distance Vector
- Hierarchical routing

4.6 Routing in the Internet
- RIP
- OSPF
- BGP

4.7 Broadcast and multicast routing

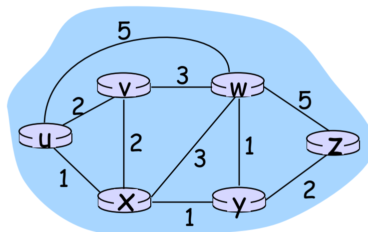
4-49

Interplay between routing, forwarding



4-50

Graph abstraction



Graph: G = (N,E)

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

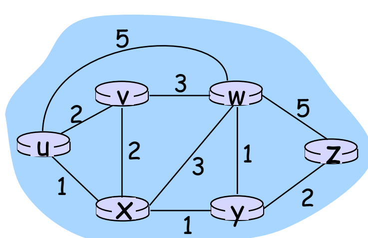
E = set of links ={ (u,v), (u,x), (v,x), (v,w), (x,w), (x,y), (w,y), (w,z), (y,z) }

Remark: Graph abstraction is useful in other network contexts

Example: P2P, where N is set of peers and E is set of TCP connections

4-51

Graph abstraction: costs



- $c(x,x')$ = cost of link (x,x')
  - e.g., $c(w,z) = 5$
- cost could always be 1, or inversely related to bandwidth, or inversely related to congestion

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

Question: What's the least-cost path between u and z ?

Routing algorithm: algorithm that finds least-cost path

4-52

Routing Algorithm classification

Global or decentralized information?

Global:

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

Decentralized:

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

Static or dynamic?

Static:

- routes change slowly over time

Dynamic:

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

4-53

Chapter 4: Network Layer

4-54

Chapter 4: Network Layer

Distance Vector Algorithm

Bellman-Ford Equation (dynamic programming)

Define

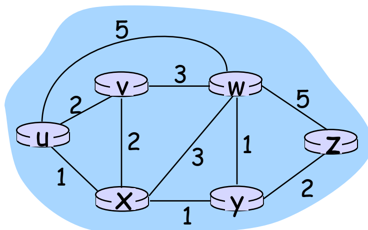
$d_x(y)$:= cost of least-cost path from x to y

Then

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

where min is 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:

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

Node that achieves minimum is next hop in shortest path ➜ forwarding table

4-57

Distance Vector Algorithm

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

4-58

Distance vector algorithm (cont.)

Basic idea:

- from time-to-time, each node sends its own distance vector estimate to neighbors
- when x receives new DV estimate from neighbor, it updates its own DV using B-F equation:

$$D_x(y) \leftarrow \min_v\{c(x,v) + D_v(y)\} \quad \text{for each node } y \in N$$

- under minor, natural conditions, the estimate $D_x(y)$ *converge to the actual least cost* $d_x(y)$

4-59

Distance Vector Algorithm (cont.)

Iterative, asynchronous:

each local iteration caused by:

- local link cost change
- DV update message from neighbor

Distributed:

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

Each node:

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

↓

recompute estimates

↓

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

4-60

Chapter 4: Network Layer

4-61

Hierarchical Routing

Our routing study thus far - idealization
- all routers identical
- network "flat"

… *not* true in practice

scale: with 200 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

4-62

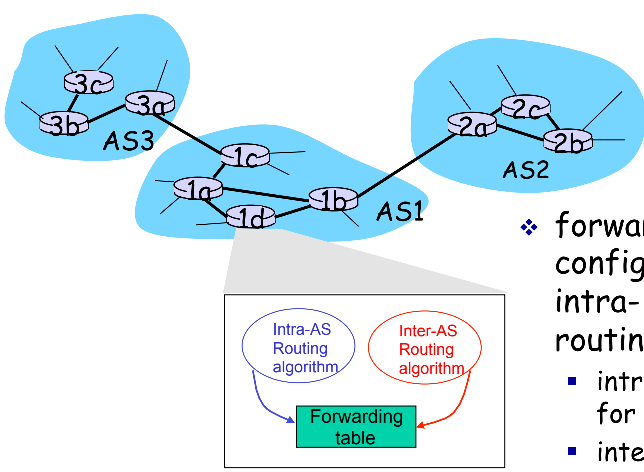
Hierarchical Routing

- 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

4-63

Interconnected ASes



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

4-64

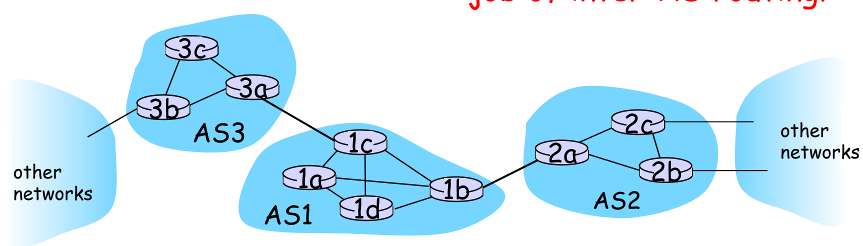
Inter-AS tasks

- 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

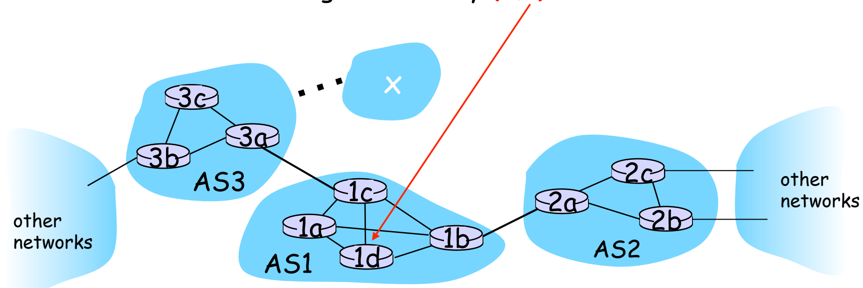
job of inter-AS routing!



4-65

Example: Setting forwarding table in router 1d

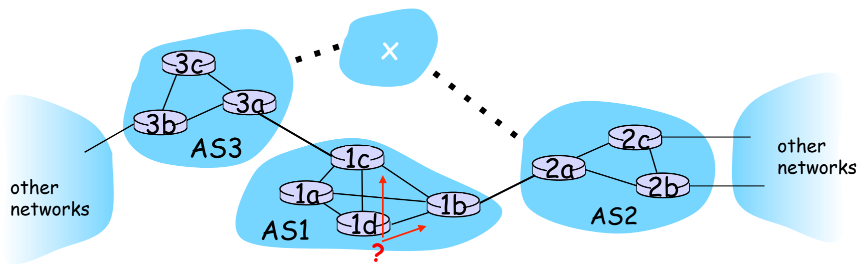
- suppose AS1 learns (via inter-AS protocol) that subnet *x* reachable via AS3 (gateway 1c) but not via AS2.
  - inter-AS protocol propagates reachability info to all internal routers
- router 1d determines from intra-AS routing info that its interface *I* is on the least cost path to 1c.
  - installs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



4-66

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!



4-67

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

4-68

Chapter 4: Network Layer

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)

Internet inter-AS routing: BGP

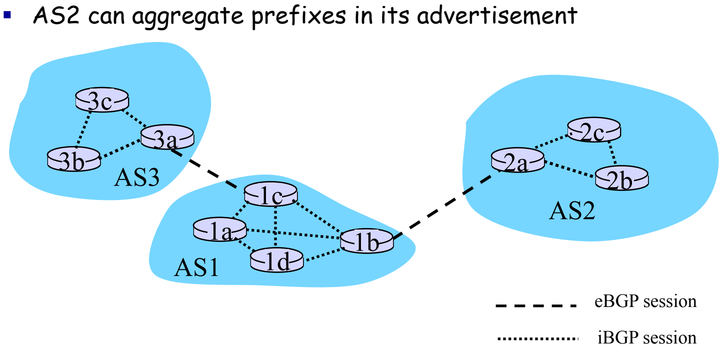
- BGP (Border Gateway Protocol): *the* de facto standard
- BGP provides each AS a means to:
  1. Obtain subnet reachability information from neighboring ASs.
  2. Propagate the reachability information to all routers internal to the AS.
  3. Determine "good" routes to subnets based on reachability information and policy.
- Allows a subnet to advertise its existence to rest of the Internet: *"I am here"*

- BGP Routing Policies in ISP networks, Matthew Caesar and Jennifer Rexford, IEEE Network, Vol 19, Issue 6, 2005

71

BGP basics

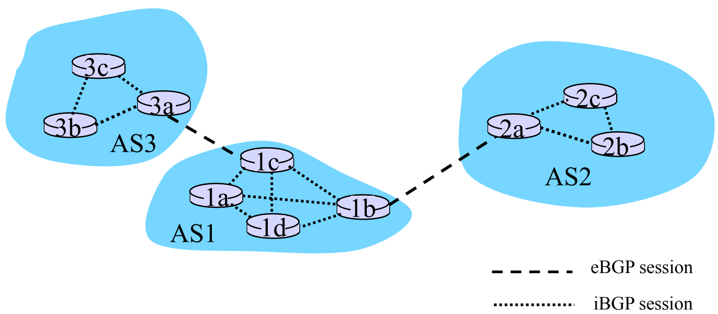
- Pairs of routers (BGP peers) exchange routing info over semi-permanent TCP conections: BGP sessions
- Note that BGP sessions do not correspond to physical links.
- When AS2 advertises a prefix to AS1, AS2 is *promising* it will forward any datagrams destined to that prefix towards the prefix.
  - AS2 can aggregate prefixes in its advertisement



72

Distributing reachability info

- With eBGP session between 3a and 1c, AS3 sends prefix reachability info to AS1.
- 1c can then use iBGP to distribute this new prefix reach info to all routers in AS1
- 1b can then re-advertise the new reach info to AS2 over the 1b-to-2a eBGP session
- When router learns about a new prefix, it creates an entry for the prefix in its forwarding table.



Path attributes & BGP routes

- When advertising a prefix, advert includes BGP attributes.
  - prefix + attributes = "route"
- Two important attributes:
  - AS-PATH: contains the ASs through which the advert for the prefix passed: AS 67 AS 17
  - NEXT-HOP: Indicates the specific internal-AS router to next-hop AS. (There may be multiple links from current AS to next-hop-AS.)
- When gateway router receives route advert, uses import policy to accept/decline.

BGP route selection

- Router may learn about more than 1 route to some prefix. Router must select route.
- Elimination rules:
  1. Local preference value attribute: policy decision
  2. Shortest AS-PATH
  3. Closest NEXT-HOP router: hot potato routing
  4. Additional criteria

75

BGP messages

- BGP messages exchanged using TCP.
- 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

76

Different Intra- and 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

Security:
- Intra-AS: tight control
- Inter-AS: very challenging, need to ensure integrity of entire path

77

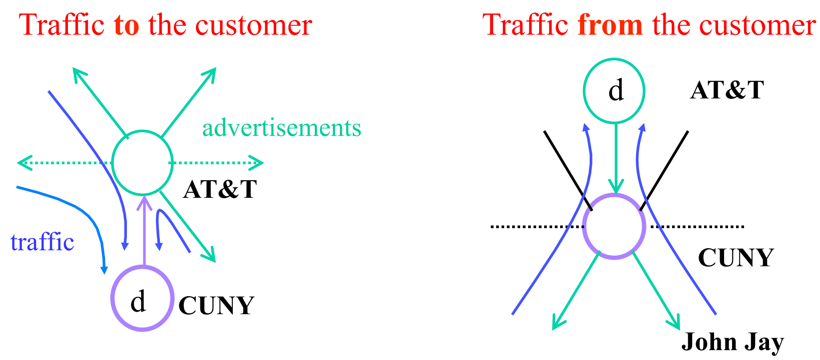
AS Relations – Business Relation

- Common relationships
  - Customer-provider
  - Peer-peer
  - Backup, sibling, …
- Implementing in BGP
  - Import policy
    - Ranking customer routes over peer routes
  - Export policy
    - Export only customer routes to peers and providers

78

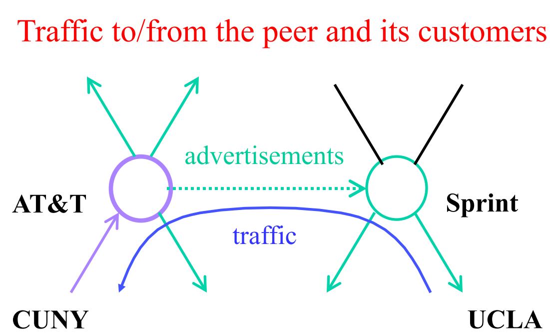
AS Relation: Customer-Provider

- Customer pays provider for access to Internet
  - Provider exports customer's routes to everybody
  - Customer exports provider's routes to customers



AS Relation: Peer-Peer

- Peers exchange traffic between customers
  - AS exports *only* customer routes to a peer
  - AS exports a peer's routes *only* to its customers



How Peering Decisions are Made?

Peer

- Reduces upstream transit costs
- Can increase end-to-end performance

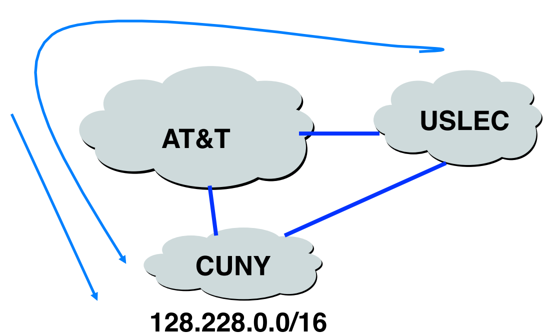
Don't Peer

- You would rather have customers
- Peers are usually your competition
- Peering relationships may require periodic renegotiation

81

AS Relation: Backup

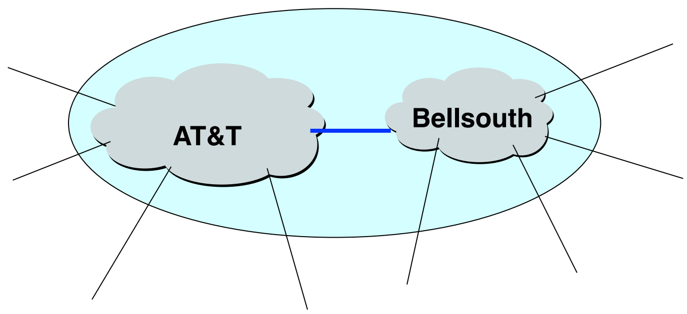
- Backup provider
  - Only used if the primary link fails
  - Routes through other paths



82

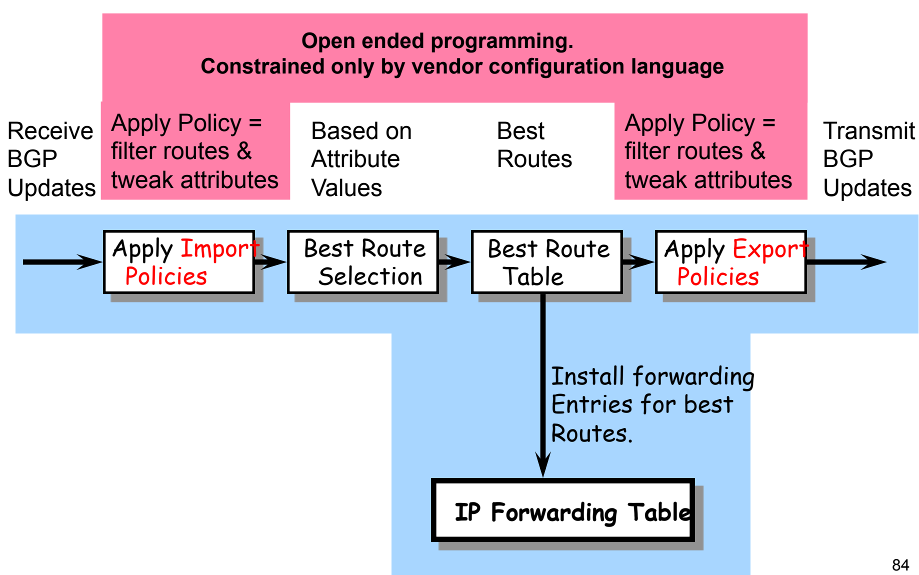
AS Relation: Sibling

- Two ASes owned by the same institution
  - E.g., two ASes that have merged
  - E.g., two ASes simply for scaling reasons
  - Essentially act as a single AS



83

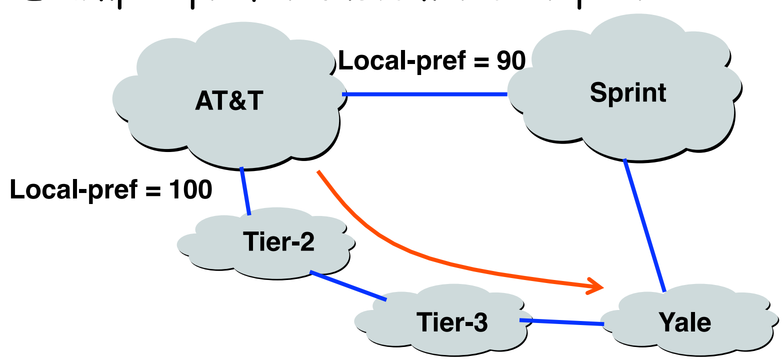
BGP Policy: Influencing Decisions



84

Import Policy: Local Preference

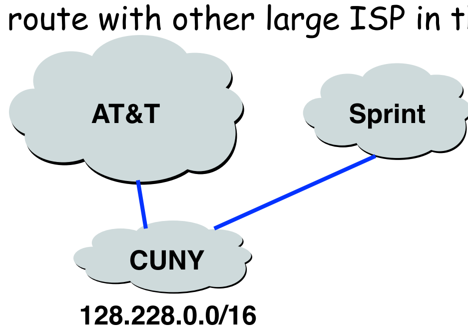
- Favor one path over another
  - Override the influence of AS path length
  - Apply local policies to prefer a path
- Example: prefer customer over peer



85

Import Policy: Filtering

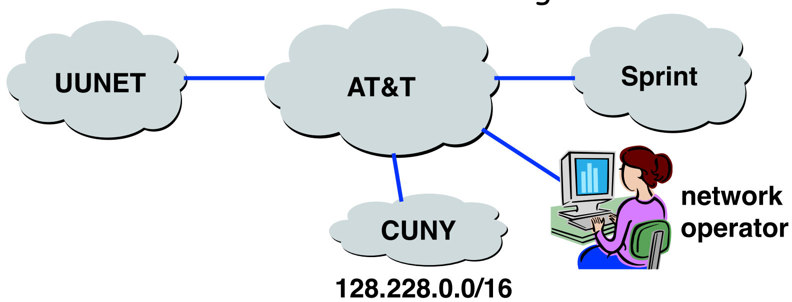
- Discard some route announcements
  - Detect configuration mistakes and attacks
- Examples on session to a customer
  - Discard route if prefix not owned by the customer
  - Discard route with other large ISP in the AS path



86

Export Policy: Filtering

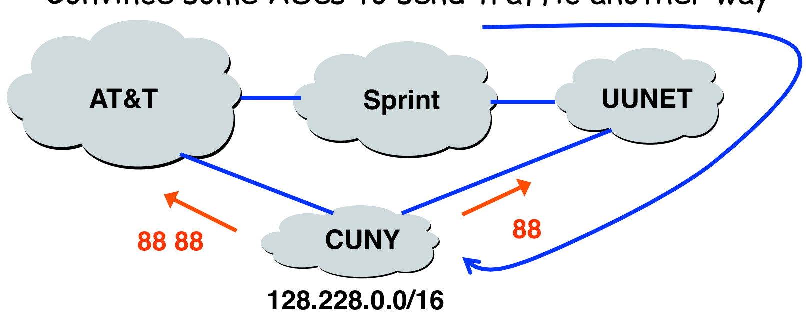
- Discard some route announcements
  - Limit propagation of routing information
- Examples
  - Don't announce routes from one peer to another
  - Don't announce routes for management hosts



87

Export Policy: Attribute Manipulation

- Modify attributes of the active route
  - To influence the way other ASes behave
- Example: AS prepending
  - Artificially inflate AS path length seen by others
  - Convince some ASes to send traffic another way



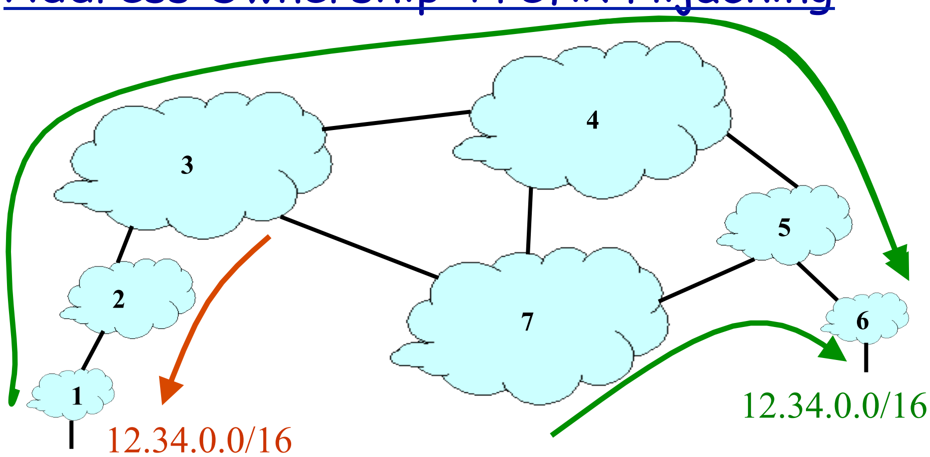
88

Security Goals for BGP

- Secure message exchange between neighbors
  - Confidential BGP message exchange
    - Can ASes exchange messages w/o someone watching?
  - No denial of service
    - Prevent overload, session reset, tampered messages?
- Validity of the routing information
  - Origin authentication
    - Is the prefix owned by the AS announcing it?
  - AS path authentication
    - Is AS path the sequence of ASes the update traversed?
  - AS path policy
    - Does AS path adhere to the routing policies of each AS?

89

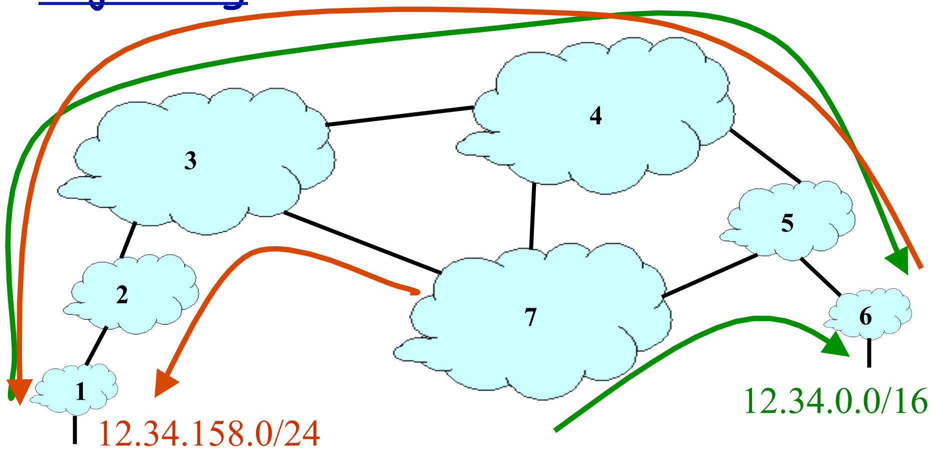
Address Ownership: Prefix Hijacking



- Consequences for the affected ASes
  - Blackhole: data traffic is discarded
  - Snooping: data traffic is inspected, and then redirected
  - Impersonation: data traffic is sent to bogus destinations

90

Address Ownership: Subprefix Hijacking



- Originating a more-specific prefix
  - Every AS picks the bogus route for that prefix
  - Traffic follows the longest matching prefix

Chapter 4: Network Layer

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
- BGP

4.7 Broadcast and multicast routing

Chapter 4: summary

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
- BGP

4.7 Broadcast and multicast routing

Chapter 5: The Data Link Layer

Our goals:

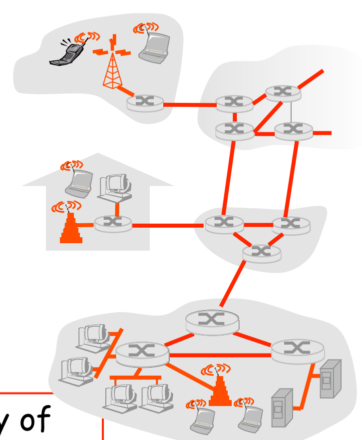
- understand principles behind data link layer services:
  - error detection, correction
  - sharing a broadcast channel: multiple access
  - link layer addressing
  - reliable data transfer, flow control: *done!*
- instantiation and implementation of various link layer technologies

Link Layer

Link Layer: Introduction

Terminology:

- hosts and routers are nodes
- communication channels that connect adjacent nodes along communication path are links
  - wired links
  - wireless links
  - LANs
- layer-2 packet is a frame, encapsulates datagram



data-link layer has responsibility of transferring datagram from one node to physically adjacent node over a link

Link layer: context

- datagram transferred by different link protocols over different links:
  - e.g., Ethernet on first link, frame relay on intermediate links, 802.11 on last link
- each link protocol provides different services
  - e.g., may or may not provide rdt over link

transportation analogy

- trip from Princeton to Lausanne
  - limo: Princeton to JFK
  - plane: JFK to Geneva
  - train: Geneva to Lausanne
- tourist = datagram
- transport segment = communication link
- transportation mode = link layer protocol
- travel agent = routing algorithm

Link Layer Services

- *framing, link access:*
  - encapsulate datagram into frame, adding header, trailer
  - channel access if shared medium
  - "MAC" addresses used in frame headers to identify source, dest
    - different from IP address!
- *reliable delivery between adjacent nodes*
  - we learned how to do this already (chapter 3)!
  - seldom used on low bit-error link (fiber, some twisted pair)
  - wireless links: high error rates
    - Q: why both link-level and end-end reliability?

Link Layer Services (more)

- *flow control:*
  - pacing between adjacent sending and receiving nodes
- *error detection*:
  - errors caused by signal attenuation, noise.
  - receiver detects presence of errors:
    - signals sender for retransmission or drops frame
- error correction:
  - receiver identifies *and corrects* bit error(s) without resorting to retransmission
- *half-duplex and full-duplex*
  - with half duplex, nodes at both ends of link can transmit, but not at same time

Link Layer

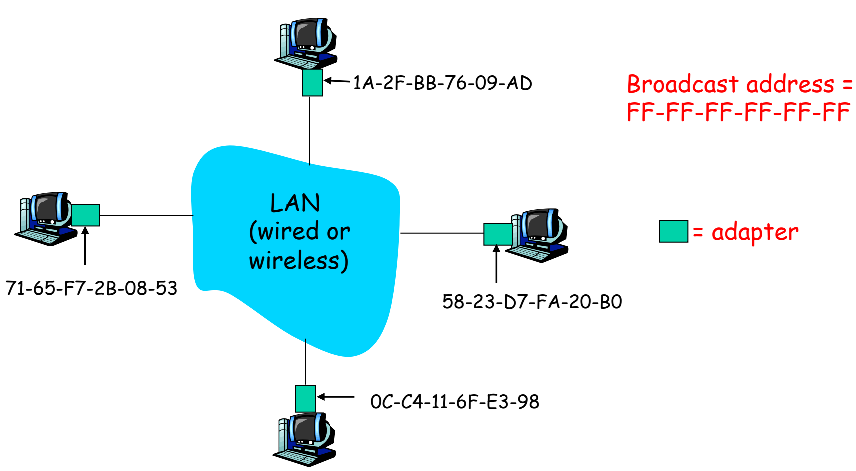
MAC Addresses and ARP

- 32-bit IP address:
  - *network-layer* address
  - used to get datagram to destination IP subnet
- MAC (or LAN or physical or Ethernet) address:
  - function: *get frame from one interface to another physically-connected interface (same network)*
  - 48 bit MAC address (for most LANs)
    - burned in NIC ROM, also sometimes software settable

Data Link Layer 5-101

LAN Addresses and ARP

Each adapter on LAN has unique LAN address



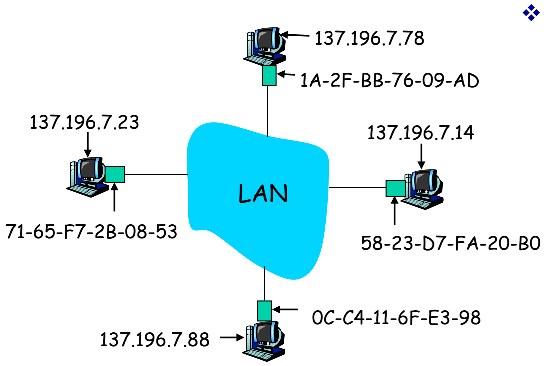
Data Link Layer 5-102

LAN Address (more)

- MAC address allocation administered by IEEE
- manufacturer buys portion of MAC address space (to assure uniqueness)
- analogy:
  - (a) MAC address: like Social Security Number
  - (b) IP address: like postal address
- MAC flat address ➜ portability
  - can move LAN card from one LAN to another
- IP hierarchical address NOT portable
  - address depends on IP subnet to which node is attached

ARP: Address Resolution Protocol

Question: how to determine MAC address of B knowing B's IP address?



- Each IP node (host, router) on LAN has ARP table
- ARP table: IP/MAC address mappings for some LAN nodes

  < IP address; MAC address; TTL>

  - TTL (Time To Live): time after which address mapping will be forgotten (typically 20 min)

ARP protocol: Same LAN (network)

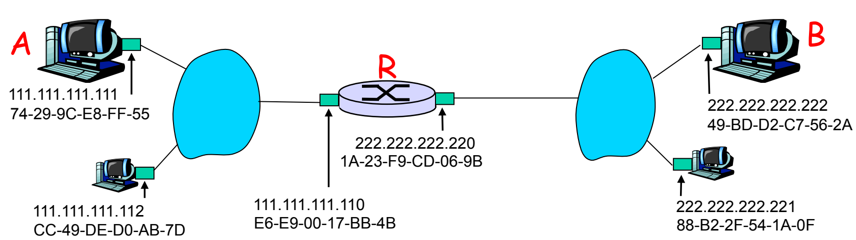
- A wants to send datagram to B, and B's MAC address not in A's ARP table.
- A broadcasts ARP query packet, containing B's IP address
  - dest MAC address = FF-FF-FF-FF-FF-FF
  - all machines on LAN receive ARP query
- B receives ARP packet, replies to A with its (B's) MAC address
  - frame sent to A's MAC address (unicast)
- A caches (saves) IP-to-MAC address pair in its ARP table until information becomes old (times out)
  - soft state: information that times out (goes away) unless refreshed
- ARP is "plug-and-play":
  - nodes create their ARP tables *without intervention from net administrator*

Data Link Layer 5-105

Addressing: routing to another LAN

walkthrough: send datagram from A to B via R.

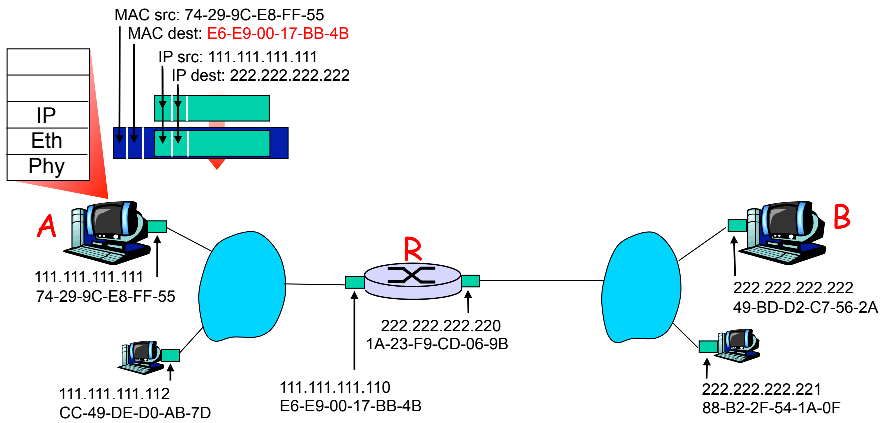
- focus on addressing - at both IP (datagram) and MAC layer (frame)
- assume A knows B's IP address
- assume A knows IP address of first hop router, R (how?)
- assume A knows MAC address of first hop router interface (how?)



Data Link Layer 5-106

Addressing: routing to another LAN

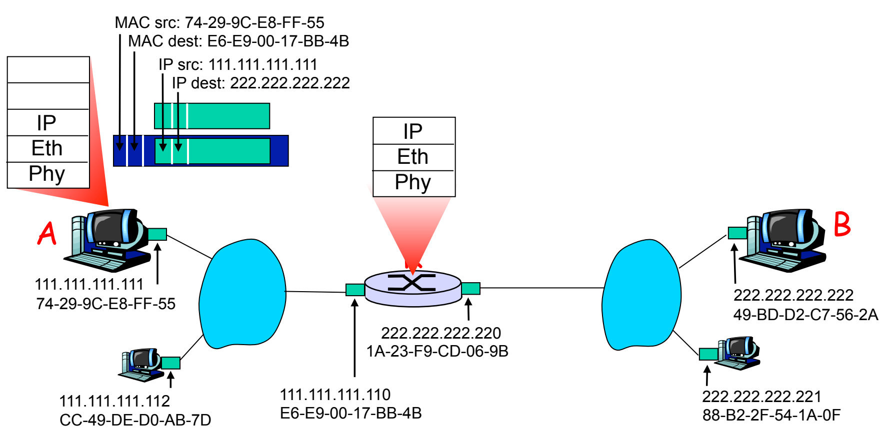
- A creates IP datagram with IP source A, destination B
- A creates link-layer frame with R's MAC address as dest, frame contains A-to-B IP datagram



Data Link Layer 5-107

Addressing: routing to another LAN

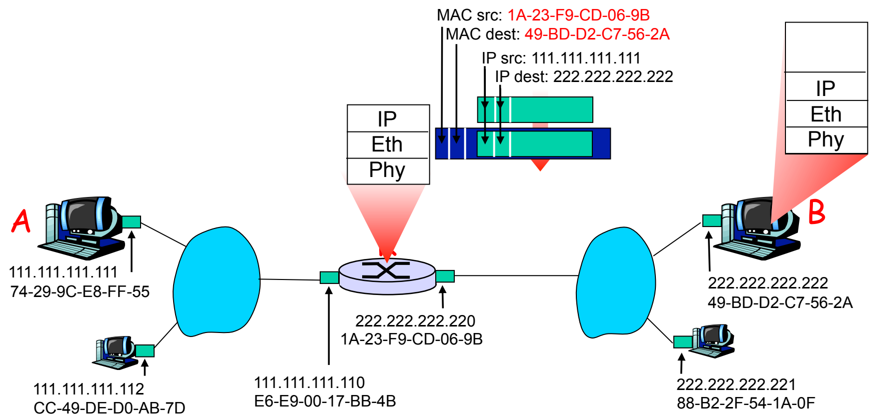
- frame sent from A to R
- frame received at R, datagram removed, passed up to IP



Data Link Layer 5-108

Addressing: routing to another LAN

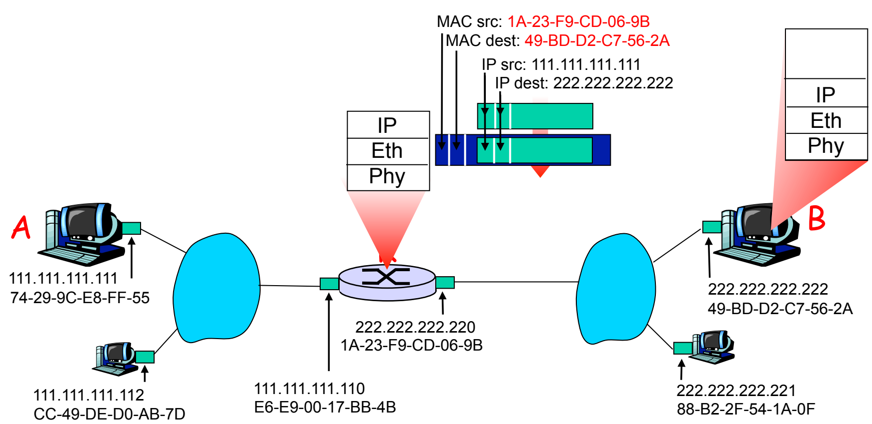
- R forwards datagram with IP source A, destination B
- R creates link-layer frame with B's MAC address as dest, frame contains A-to-B IP datagram



Data Link Layer 5-109

Addressing: routing to another LAN

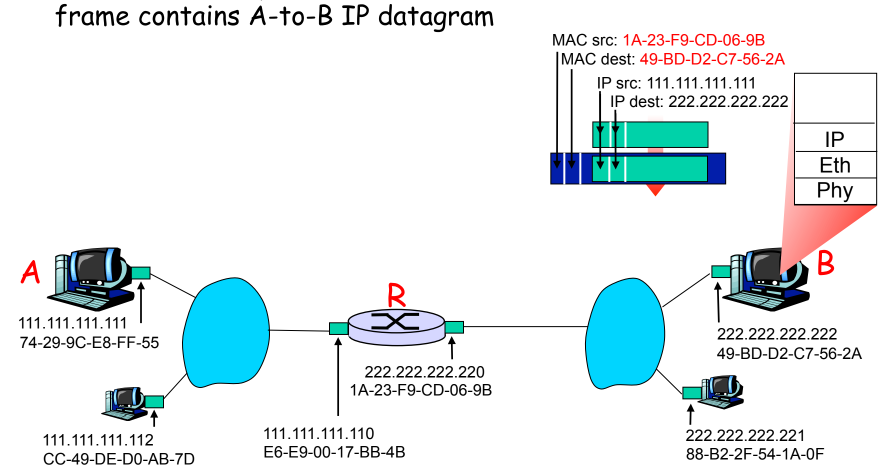
- R forwards datagram with IP source A, destination B
- R creates link-layer frame with B's MAC address as dest, frame contains A-to-B IP datagram



Data Link Layer 5-110

Addressing: routing to another LAN

- R forwards datagram with IP source A, destination B
- R creates link-layer frame with B's MAC address as dest, frame contains A-to-B IP datagram



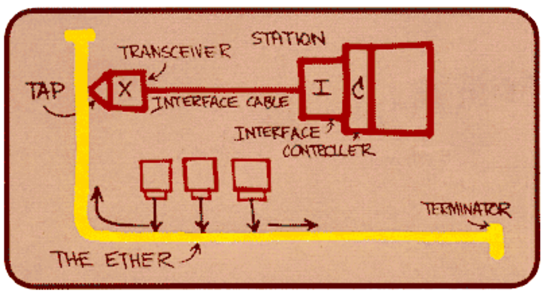
Link Layer

5.1 Introduction and services
5.2 Error detection and correction
5.3Multiple access protocols
5.4 Link-Layer Addressing
5.5 Ethernet
5.6 Link-layer switches
5.7 PPP
5.8 Link virtualization: MPLS
5.9 A day in the life of a web request

Ethernet

"dominant" wired LAN technology:

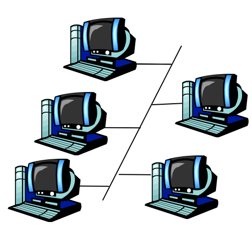
- cheap $20 for NIC
- first widely used LAN technology
- simpler, cheaper than token LANs and ATM
- kept up with speed race: 10 Mbps – 10 Gbps



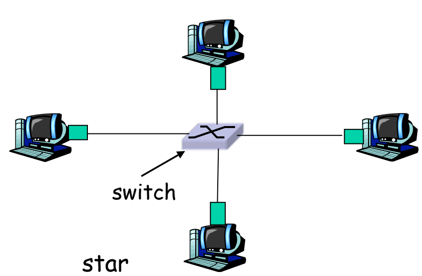
Metcalfe's Ethernet sketch

Star topology

- bus topology popular through mid 90s
  - all nodes in same collision domain (can collide with each other)
- today: star topology prevails
  - active *switch* in center
  - each "spoke" runs a (separate) Ethernet protocol (nodes do not collide with each other)



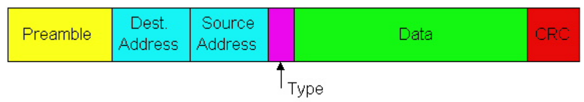
bus: coaxial cable



star

Ethernet Frame Structure

Sending adapter encapsulates IP datagram (or other network layer protocol packet) in Ethernet frame

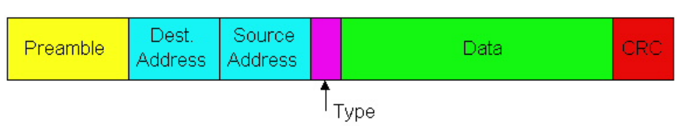


Preamble:

- 7 bytes with pattern 10101010 followed by one byte with pattern 10101011
- used to synchronize receiver, sender clock rates

Ethernet Frame Structure (more)

- Addresses: 6 bytes
  - if adapter receives frame with matching destination address, or with broadcast address (e.g. ARP packet), it passes data in frame to network layer protocol
  - otherwise, adapter discards frame
- Type: indicates higher layer protocol (mostly IP but others possible, e.g., Novell IPX, AppleTalk)
- CRC: checked at receiver, if error is detected, frame is dropped



Ethernet: Unreliable, connectionless

- connectionless: No handshaking between sending and receiving NICs
- unreliable: receiving NIC doesn't send acks or nacks to sending NIC
  - stream of datagrams passed to network layer can have gaps (missing datagrams)
  - gaps will be filled if app is using TCP
  - otherwise, app will see gaps
- Ethernet's MAC protocol: unslotted CSMA/CD

Link Layer

5.1 Introduction and services

5.2 Error detection and correction

5.3 Multiple access protocols

5.4 Link-layer Addressing

5.5 Ethernet

5.6 Link-layer switches, LANs, VLANs

5.7 PPP

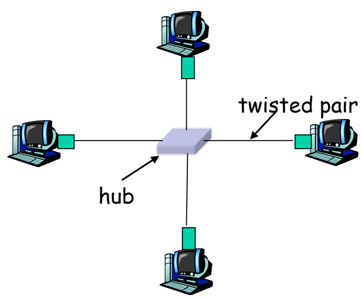
5.8 Link virtualization: MPLS

5.9 A day in the life of a web request

Hubs

… physical-layer ("dumb") repeaters:

- bits coming in one link go out *all* other links at same rate
- all nodes connected to hub can collide with one another
- no frame buffering
- no CSMA/CD at hub: host NICs detect collisions



Switch

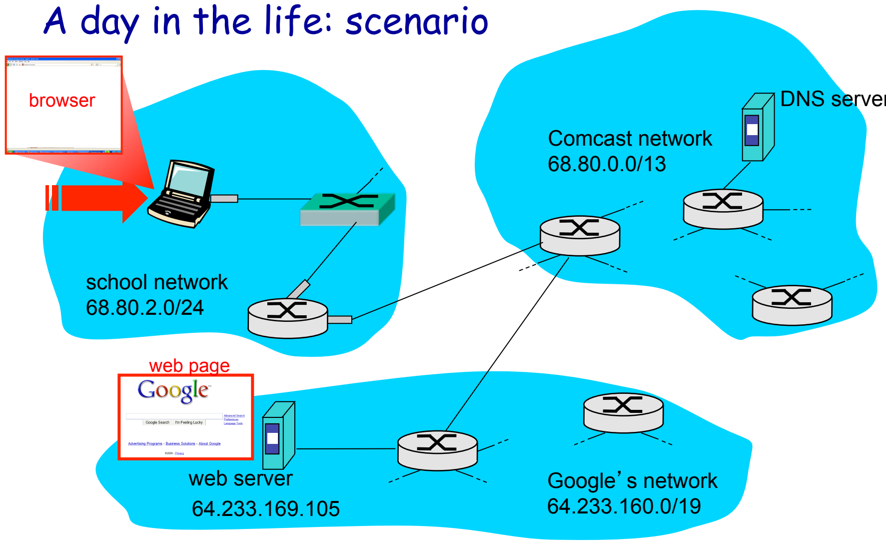
- link-layer device: smarter than hubs, take *active* role
  - store, forward Ethernet frames
  - examine incoming frame's MAC address, selectively forward frame to one-or-more outgoing links when frame is to be forwarded on segment, uses CSMA/CD to access segment
- *transparent*
  - hosts are unaware of presence of switches
- *plug-and-play, self-learning*
  - switches do not need to be configured

Link Layer

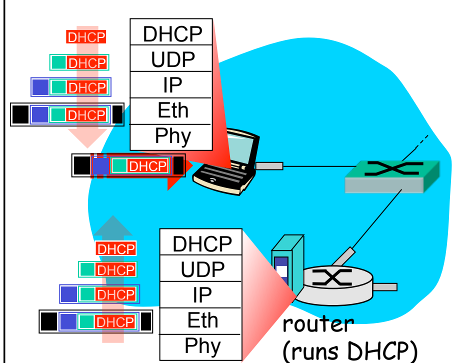
Synthesis: a day in the life of a web request

- journey down protocol stack complete!
  - application, transport, network, link
- putting-it-all-together: synthesis!
  - *goal:* identify, review, understand protocols (at all layers) involved in seemingly simple scenario: requesting www page
  - *scenario:* student attaches laptop to campus network, requests/receives www.google.com

A day in the life: scenario

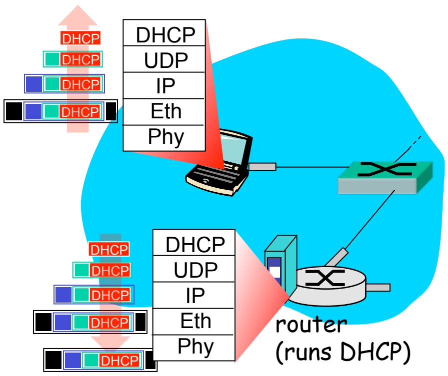


A day in the life… connecting to the Internet



- connecting laptop needs to get its own 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

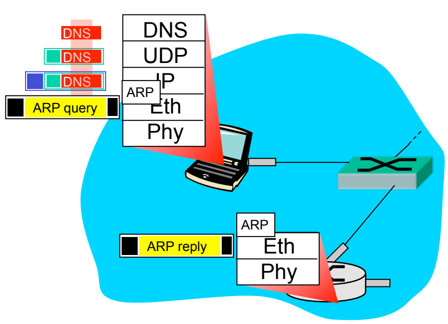
A day in the life… connecting to the Internet



- DHCP server formulates ***DHCP ACK*** containing client's IP address, IP address of first-hop router for client, name & IP address of DNS server
- encapsulation at DHCP server, frame forwarded (**switch learning**) through LAN, demultiplexing at client
- DHCP client receives DHCP ACK reply

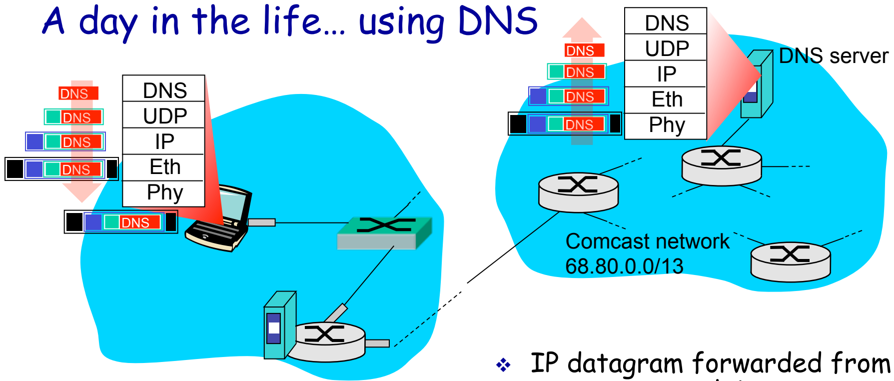
Client now has IP address, knows name & addr of DNS server, IP address of its first-hop router

A day in the life… ARP (before DNS, before HTTP)



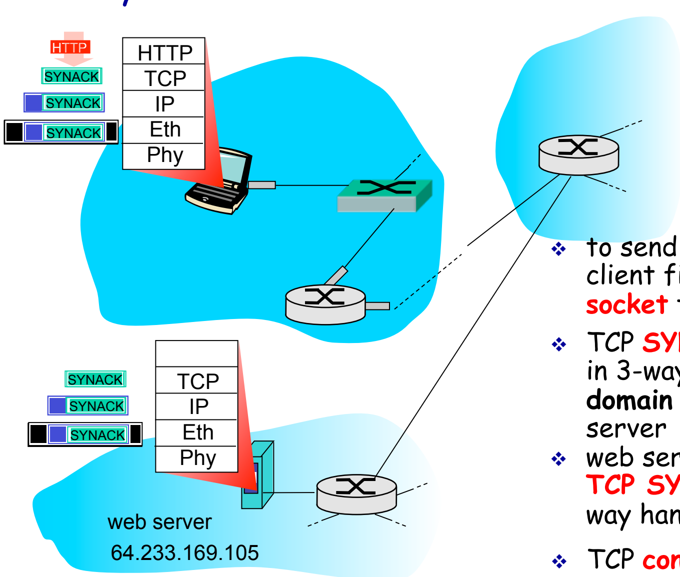
- before sending ***HTTP*** request, need IP address of www.google.com: ***DNS***
- DNS query created, encapsulated in UDP, encapsulated in IP, encapsulated in Eth. In order to send frame to router, need MAC address of router interface: **ARP**
- **ARP query** broadcast, received by router, which replies with **ARP reply** giving MAC address of router interface
- client now knows MAC address of first hop router, so can now send frame containing DNS query

A day in the life… using DNS



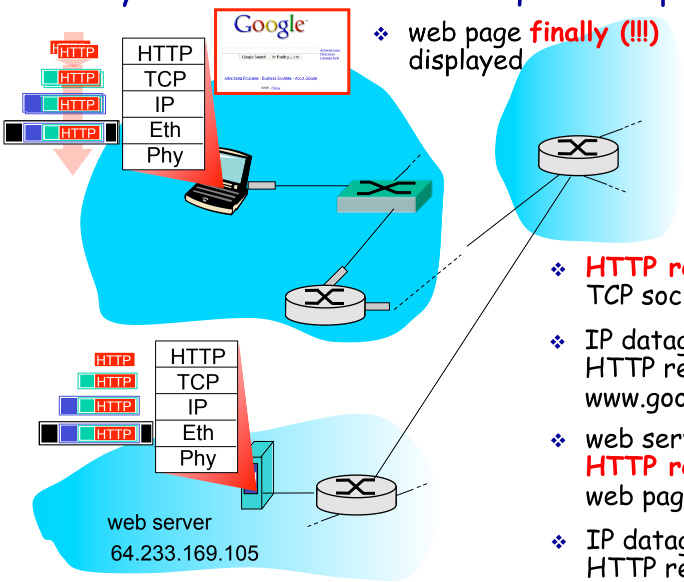
- IP datagram containing DNS query forwarded via LAN switch from client to 1st hop router
- IP datagram forwarded from campus network into comcast network, routed (tables created by **RIP, OSPF, IS-IS** and/or **BGP** routing protocols) to DNS server
- demuxed to DNS server
- DNS server replies to client with IP address of www.google.com

A day in the life… TCP connection carrying HTTP



- to send HTTP request, client first opens **TCP socket** to web server
- TCP **SYN segment** (step 1 in 3-way handshake) **inter-domain routed** to web server
- web server responds with **TCP SYNACK** (step 2 in 3-way handshake)
- TCP **connection established!**

A day in the life… HTTP request/reply



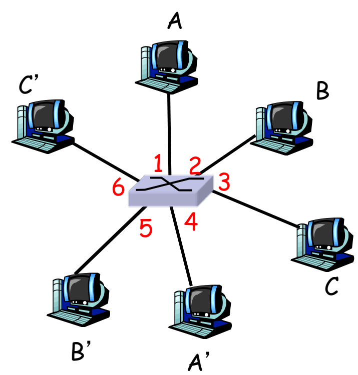
- web page **finally (!!!)** displayed
- **HTTP request** sent into TCP socket
- IP datagram containing HTTP request routed to www.google.com
- web server responds with **HTTP reply** (containing web page)
- IP datagram containing HTTP reply routed back to client

Chapter 5: Summary

- principles behind data link layer services:
  - error detection, correction
  - sharing a broadcast channel: multiple access
  - link layer addressing
- instantiation and implementation of various link layer technologies
  - Ethernet
  - switched LANS, VLANs
  - PPP
  - virtualized networks as a link layer: MPLS
- synthesis: a day in the life of a web request

Switch: allows *multiple* simultaneous transmissions

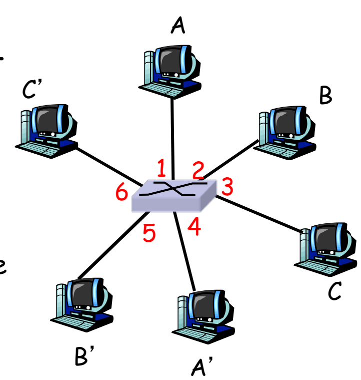
- hosts have dedicated, direct connection to switch
- switches buffer packets
- Ethernet protocol used on *each* incoming link, but no collisions; full duplex
  - each link is its own collision domain
- *switching:* A-to-A' and B-to-B' simultaneously, without collisions
  - not possible with dumb hub



switch with six interfaces (1,2,3,4,5,6)

Switch Table

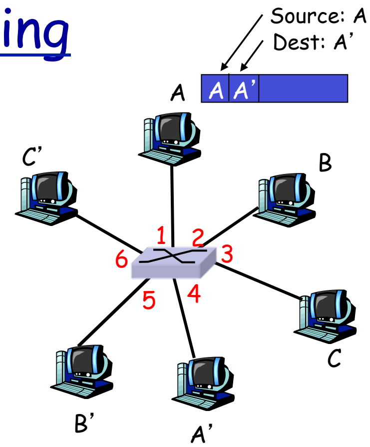
- *Q:* how does switch know that A' reachable via interface 4, B' reachable via interface 5?
- *A:* each switch has a switch table, each entry:
  - (MAC address of host, interface to reach host, time stamp)
- looks like a routing table!
- *Q:* how are entries created, maintained in switch table?
  - something like a routing protocol?



switch with six interfaces (1,2,3,4,5,6)

Switch: self-learning

- switch *learns* which hosts can be reached through which interfaces
  - when frame received, switch "learns" location of sender: incoming LAN segment
  - records sender/location pair in switch table



MAC addr	interface	TTL
A	1	60

Switch table
(initially empty)

Data Link Layer 5-133

Switch: frame filtering/forwarding

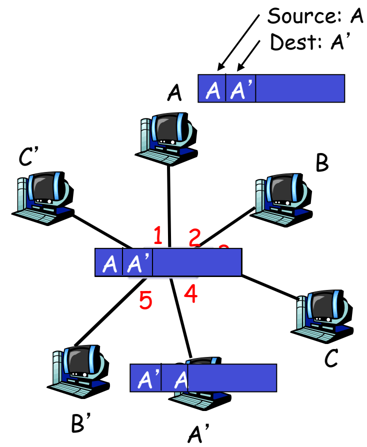
When frame received:

1. record link associated with sending host
2. index switch table using MAC dest address
3. if entry found for destination
   then {
     if dest on segment from which frame arrived
       then drop the frame
       else forward the frame on interface indicated
     }
   else flood

forward on all but the interface on which the frame arrived

Data Link Layer 5-134

Self-learning, forwarding: example



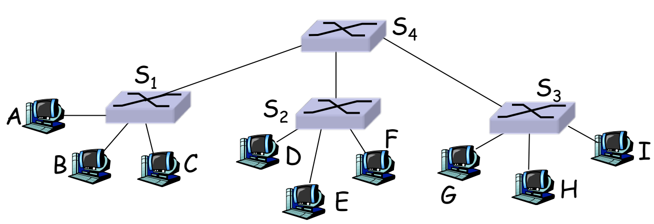
- frame destination unknown: flood
- destination A location known: selective send

MAC addr	interface	TTL
A	1	60
A'	4	60

Switch table (initially empty)

Interconnecting switches

- switches can be connected together



- Q: sending from A to G - how does S_1 know to forward frame destined to G via S_4 and S_3?
- A: self learning! (works exactly the same as in single-switch case!)

Institutional network