Internetworking

Acknowledgements
These Slides have been adapted from the originals made available by J. Kurose and K. Ross

Goals

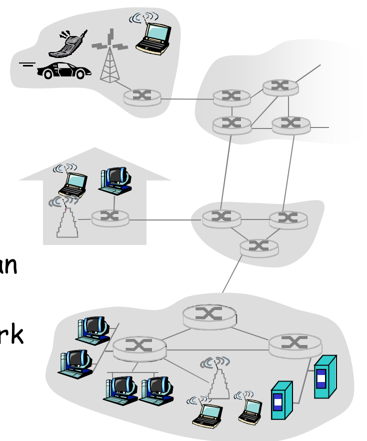
- Introduce the concept of internetwork
- understand principles behind internetworking
  - forwarding versus routing
  - how a router works
  - how data are forwarded to the final destination
  - routing (path selection)
  - dealing with scale
  - advanced topics: IPv6, mobility
- instantiation, implementation in the Internet

Internetworking

- Introduction
- What's inside a router
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - Datagram forwarding
  - Address resolution (ARP)
  - ICMP
  - IPv6
- Routing algorithms
  - Link state, Distance Vector, Hierarchical routing
- Routing in the Internet
  - RIP
  - OSPF
  - BGP

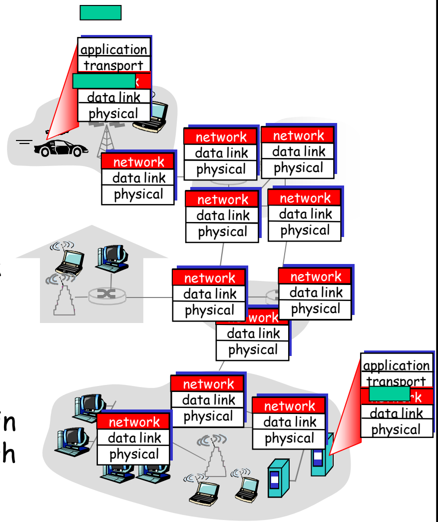
What is an inter-network (internet)?

- A network of networks
- A collection of networks
  - With different technology
    - Different physical characteristics
    - Different frame format
    - Different addressing scheme
  - Linked together by routers
- that appear as a single system
  - Users connected to the internet can communicate each other
  - Irrespective of the physical network they are attached to
- Internetworking protocol
  - Creates the internet abstraction
  - IP is the internetworking protocol for Internet



Internet Protocol (IP)

- transport data from sending to receiving host
  - passing through different nets
  - On sending side encapsulates data into datagrams
  - On receiving side, delivers data to transport layer
- Internetworking protocol in *every* host, router
  - router examines header fields in all IP datagrams passing through it



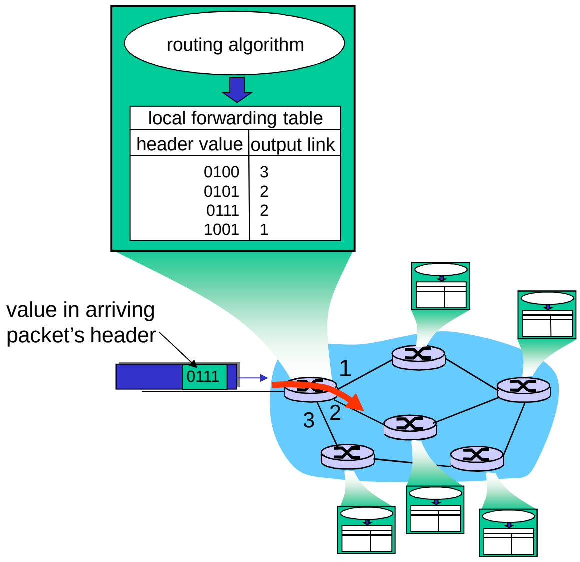
Two Key Internetworking Functions

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

Analogy: a trip to a given destination

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

Interplay between routing and forwarding

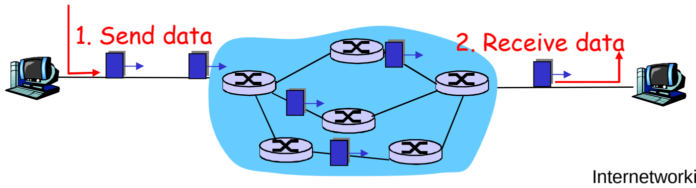


Type of Service

- **Connectionless:** each packet is managed on an individual basis
  - Also known as datagram service
- **Connection**: Virtual Circuit is preliminary established and all packets follow the same path

Datagram service

- no call setup at network layer
- routers: no state about end-to-end connections
  - no network-level concept of "connection"
- packets between same source-dest pair may take different paths
- packets forwarded using destination host address



Datagram or VC network: why?

Internet (datagram)

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

ATM (VC)

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

Service Models

- Reliable Delivery
- In-order delivery
- Guaranteed Minimal Bandwidth
- Guaranteed Bounded Delay
- Guaranteed Bounded Delay
- Guaranteed Maximum Jitter
- Security Services
  - Data confidentiality
  - Data Integrity
  - Source Authentication

Internet Quality-of-Service (QoS) model

- The QoS model provided by the Internet is known as best effort service
- Other computer networks can offer different types of QoS
  - ATM networks
    - Constant Bit Rate (CBR)
    - Variable Bit Rate (VBR)
    - Available Bit Rate (ABR)
    - Unspecified Bit Rate (UBR)

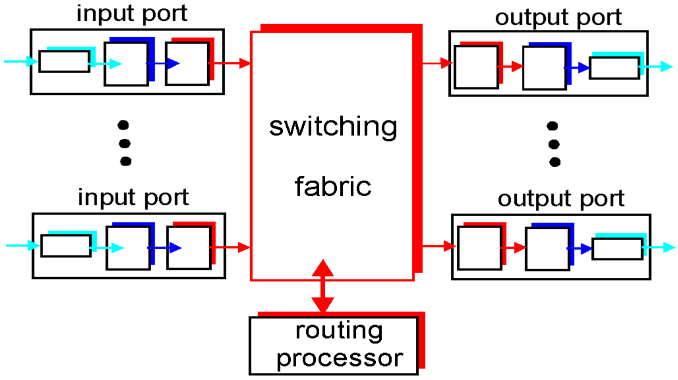
Internetworking

- Introduction
- What's inside a router
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP
  - IPv6
- Routing algorithms
  - Link state, Distance Vector, Hierarchical routing
- Routing in the Internet
  - RIP
  - OSPF
  - BGP

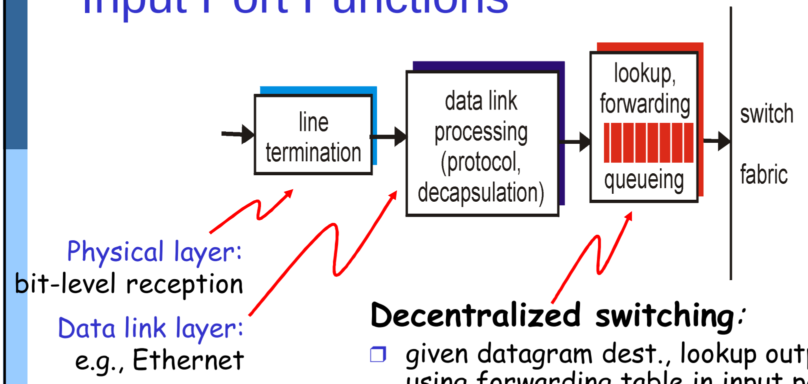
Router Architecture Overview

Two key router functions:

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



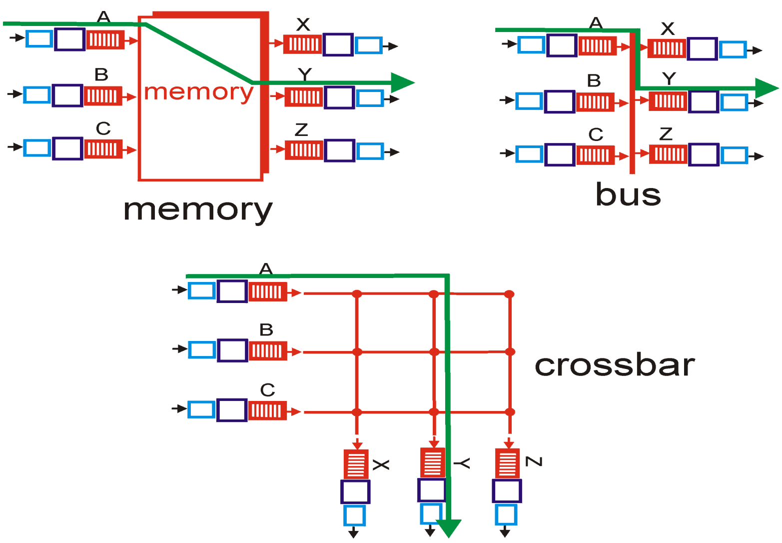
Input Port Functions



Decentralized switching:

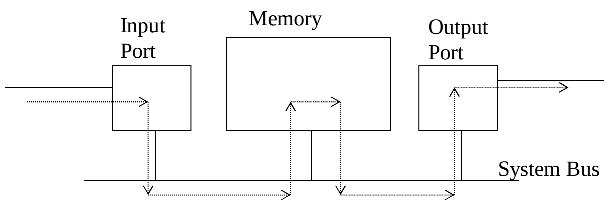
- given datagram dest., lookup output port using forwarding table in input port memory
- goal: complete input port processing at 'line speed'
- queuing: if datagrams arrive faster than forwarding rate into switch fabric

Three types of switching fabrics



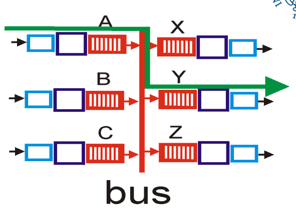
Switching Via Memory

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



- Modern Routers
  - Shared-memory multi-processors
    - Cisco Catalist 8500 switches

Switching Via a Bus

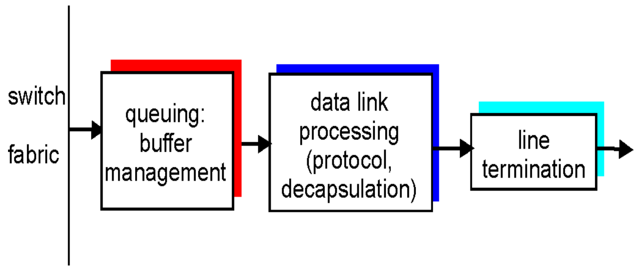


- datagram from input port memory to output port memory via a shared bus
- bus contention: switching speed limited by bus bandwidth
- 32 Gbps bus, Cisco 5600
  - sufficient speed for access and enterprise routers

Switching Via An Interconnection Network

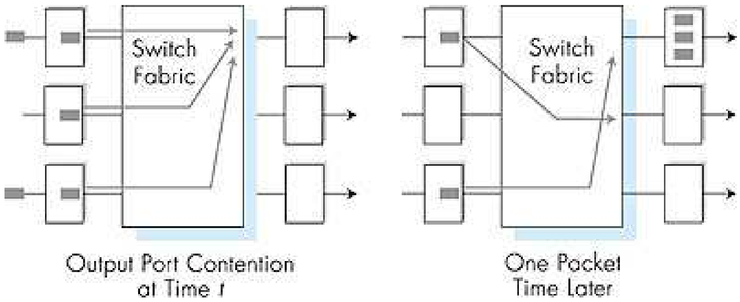
- overcome bus bandwidth limitations
- Banyan networks, other interconnection nets initially developed to connect processors in multiprocessor
- Cisco 12000: switches 60 Gbps through the interconnection network
- advanced design: fragmenting datagram into fixed length cells, switch cells through the fabric.

Output Ports



- *Buffering* required when datagrams arrive from fabric faster than the transmission rate
- *Scheduling discipline* chooses among queued datagrams for transmission
  - First Come First Served (FCFC)
  - Weighted Fair Queuing (WFQ)

Output port queueing



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

How much buffering?

- RFC 3439 rule of thumb
  - average buffering equal to “typical” RTT (say 250 msec) times link capacity C
  - e.g., C = 10 Gbps link: 2.5 Gbit buffer
- Recent recommendation
  - with N TCP flows (with large N), buffering equal to

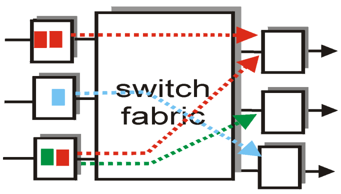
$$\frac{RTT \cdot C}{\sqrt{N}}$$

How to manage the output queue?

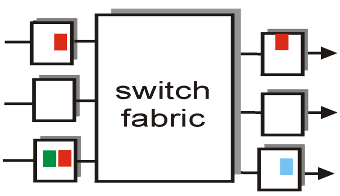
- Scheduling algorithms
  - First Come First Served (FCFS)
  - Weighted Fair Queuing (WFQ)
- What to do when a new packet arrive and there is no more space?
  - Drop the arriving packet (drop tail)
  - Drop one or more already-queued packet
  - Active Queue Management (AQM)
    - Random Early Detection (RED)

Input Port Queuing

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



output port contention at time t - only one red packet can be transferred



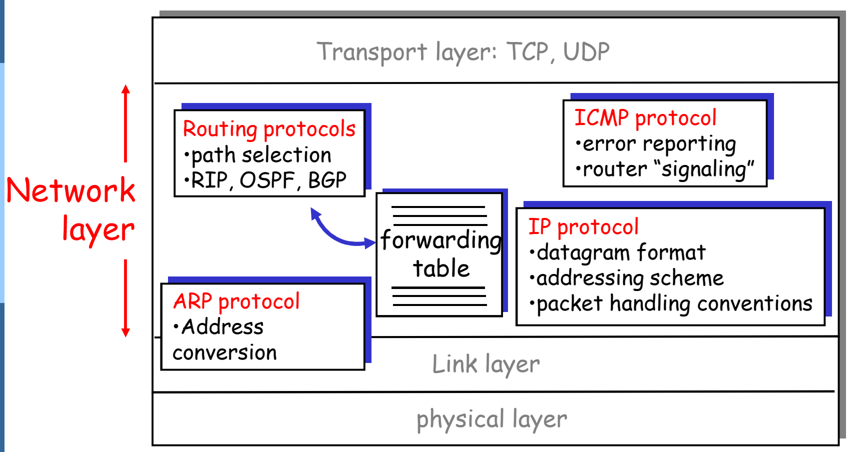
green packet experiences HOL blocking

Internetworking

- Introduction
- What's inside a router
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - Datagram forwarding
  - Address resolution (ARP)
  - ICMP
  - IPv6
- Routing algorithms
  - Link state, Distance Vector, Hierarchical routing
- Routing in the Internet
  - RIP
  - OSPF
  - BGP

The Internet Network layer

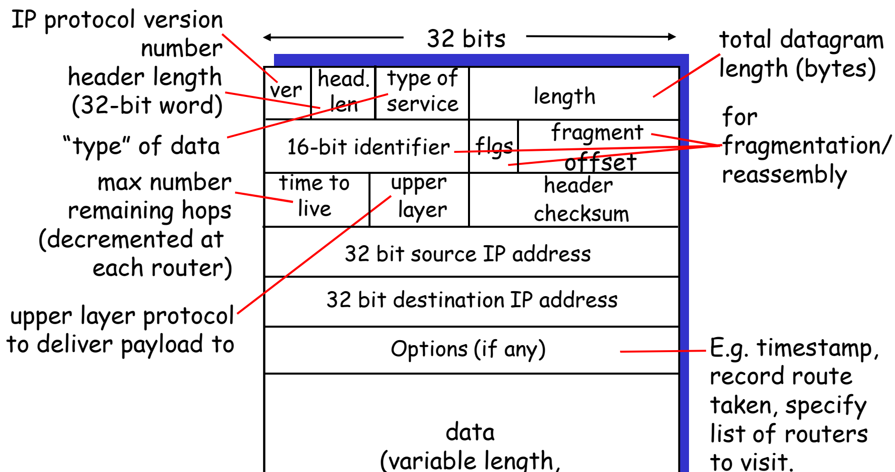
Host, router network layer functions:



Internetworking

- Introduction
- What's inside a router
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - Datagram forwarding
  - Address resolution (ARP)
  - ICMP
  - IPv6
- Routing algorithms
  - Link state, Distance Vector, Hierarchical routing
- Routing in the Internet
  - RIP
  - OSPF
  - BGP

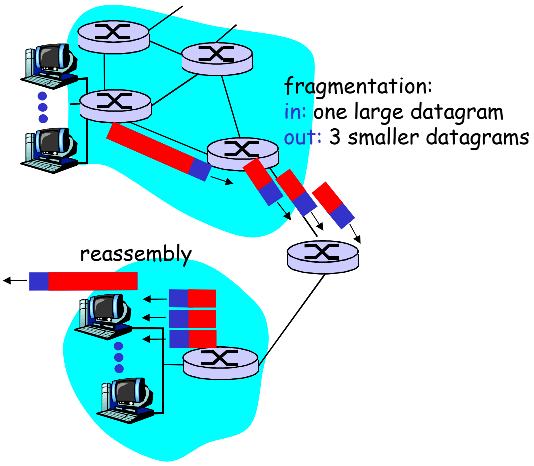
IP datagram format



how much overhead ?
- 20 bytes

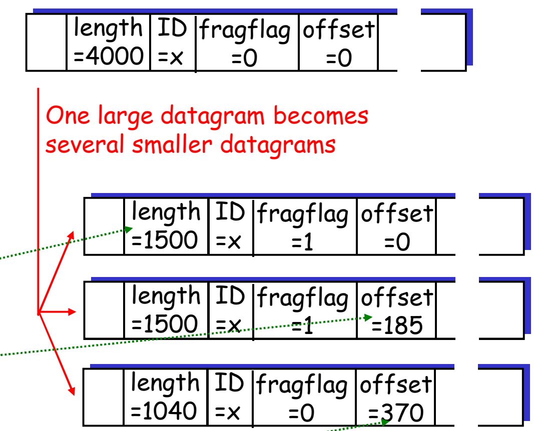
IP Fragmentation & Reassembly

- 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 and Reassembly

Example
- 4000 byte datagram
- MTU = 1500 bytes



1480 bytes in data field

offset = 1480/8

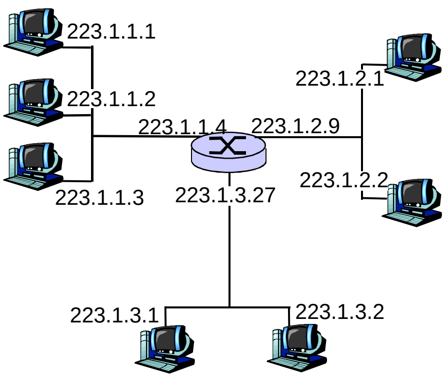
offset = 2960/8

Internetworking

- Introduction
- What's inside a router
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - Datagram forwarding
  - Address resolution (ARP)
  - ICMP
  - IPv6
- Routing algorithms
  - Link state, Distance Vector, Hierarchical routing
- Routing in the Internet
  - RIP
  - OSPF
  - BGP

IP Addressing: introduction

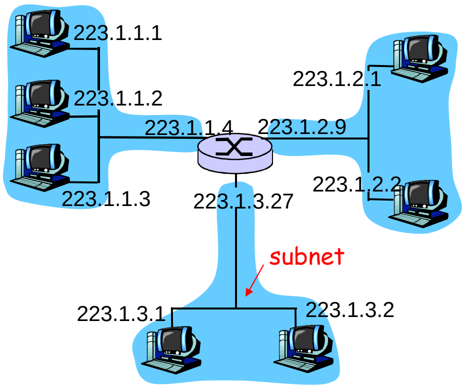
- IP address: 32-bit identifier for host, router *interface*
- *interface:* connection between host/router and physical link
  - router's typically have multiple interfaces
  - host typically has one interface
  - IP addresses associated with each interface



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

Subnets

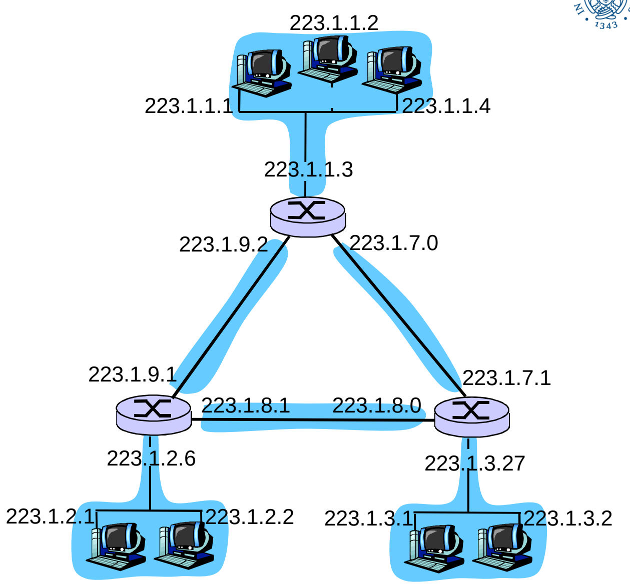
- IP address:
  - subnet part (high order bits)
  - host part (low order bits)
- *What's a subnet ?*
  - device interfaces with same subnet part of IP address
  - can physically reach each other without intervening router



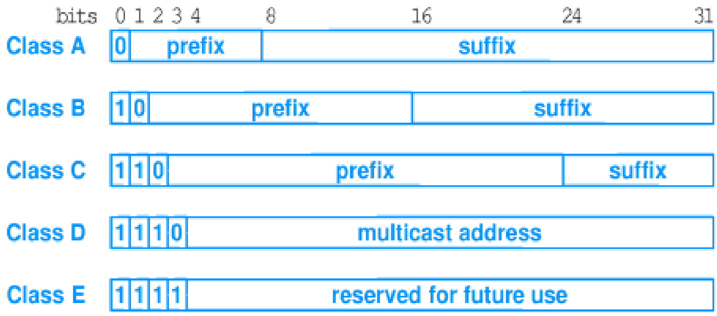
network consisting of 3 subnets

Subnets

How many?



Classes of IP Addresses



35

Classes and dotted decimal notation

Class	Range
A	0 - 127
B	128 - 191
C	192 - 223
D	224 - 239
E	240 - 255

36

Addresses in different classes

Address Class	Bits In Prefix	Maximum Number of Networks	Bits In Suffix	Maximum Number Of Hosts Per Network
A	7	128	24	16777216
B	14	16384	16	65536
C	21	2097152	8	256

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

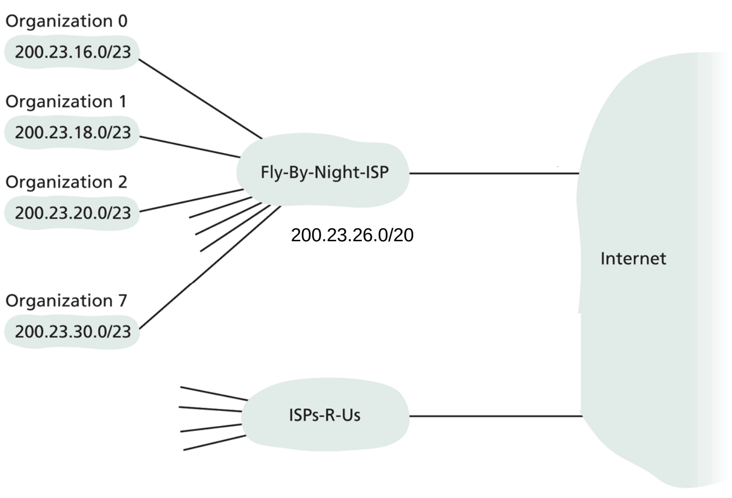
200.23.16.0/23

Reserved IP Addresses

Network Number	Host Number	Description	Notes
all 0s	all 0s	“this node”	Used at startup
x	All 0s	Network Address	Identify network *x*
x	all1s	Broadcast Address	datagram sent to all nodes of network *x*
all 1s	all1s	Restricted Broadcast Address	datagram sent to all nodes of the local network
127	--	Loopback Address	Used when developing applications

39

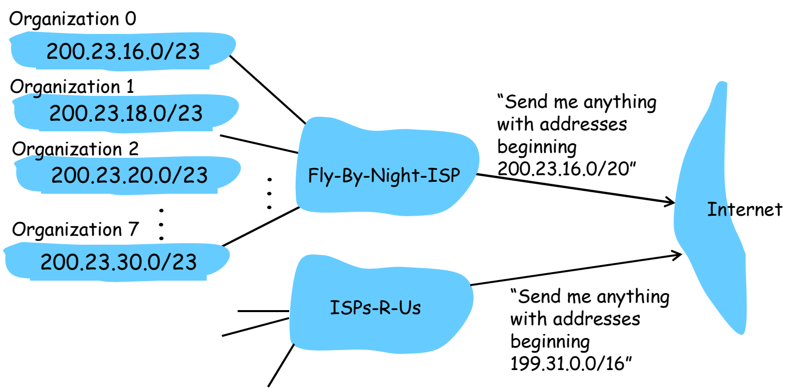
Hierarchical Addressing



Hierarchical addressing

Hierarchical addressing: route aggregation

Hierarchical addressing allows efficient advertisement of routing information:



Internetworking 41

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

Internetworking 42

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

IP addresses: how to get one?

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

- Permanent Address
  - hard-coded by system admin in a file
  - Windows: control-panel->network->configuration->tcp/ip->properties
  - UNIX: /etc/rc.config
- Temporary Address
  - 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

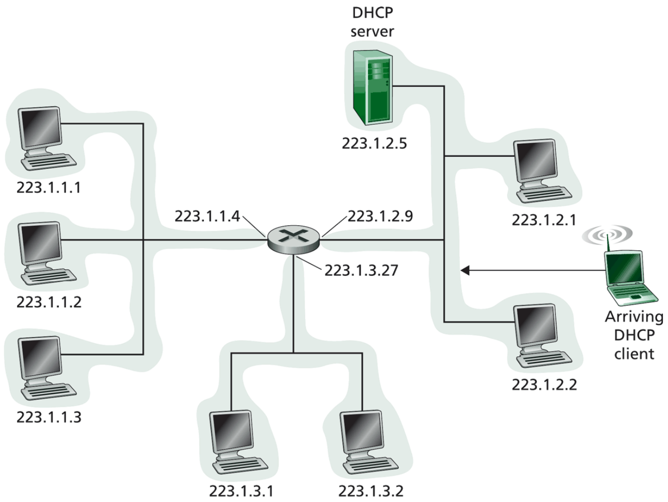
Allows reuse of addresses (only hold address while connected "on")

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

DHCP overview:

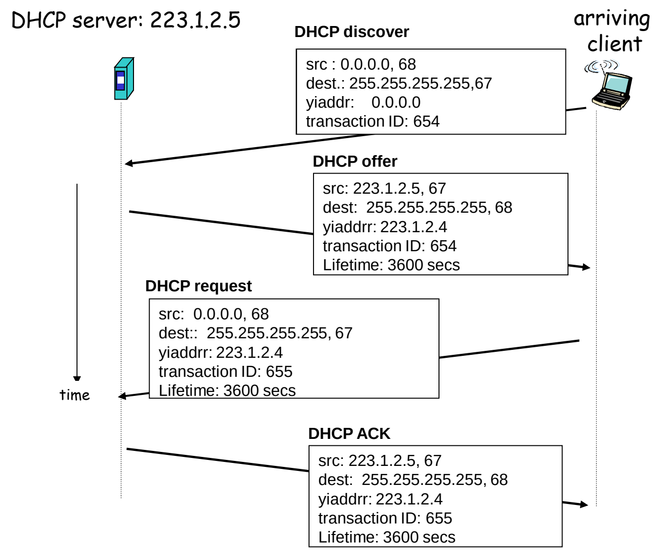
- host broadcasts "DHCP discover" msg
- DHCP server responds with "DHCP offer" msg
- host requests IP address: "DHCP request" msg
- DHCP server sends address: "DHCP ack" msg

DHCP client-server scenario



DHCP client-server scenario

DHCP client-server scenario

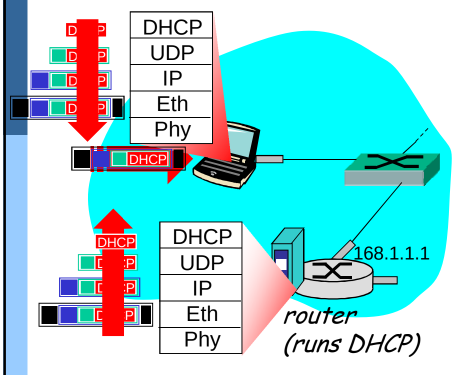


DHCP: more than IP address

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

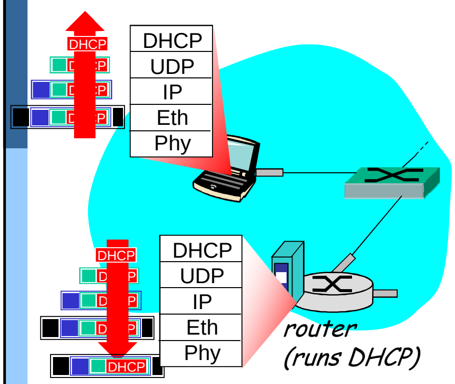
- address of first-hop router for client
- name and IP address of DNS sever
- network mask (indicating network versus host portion of address)

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 demux'ed to IP demux'ed, UDP demux'ed to DHCP

DHCP: example

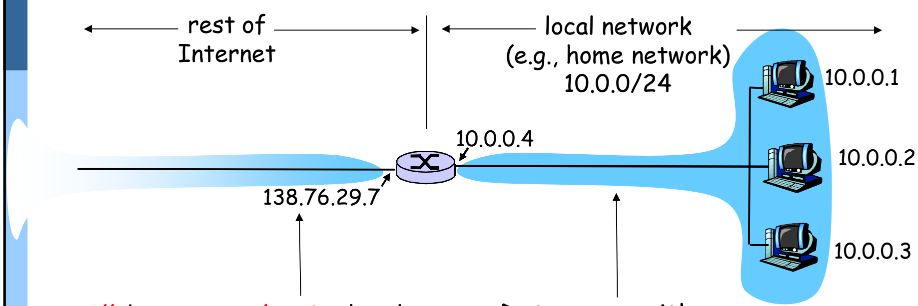


- DCP server formulates DHCP ACK containing client's IP address, IP address of first-hop router for client, name & IP address of DNS server
- encapsulation of DHCP server, frame forwarded to client, demux'ing up to DHCP at client
- client now knows its IP address, name and IP address of DSN server, IP address of its first-hop router

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



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

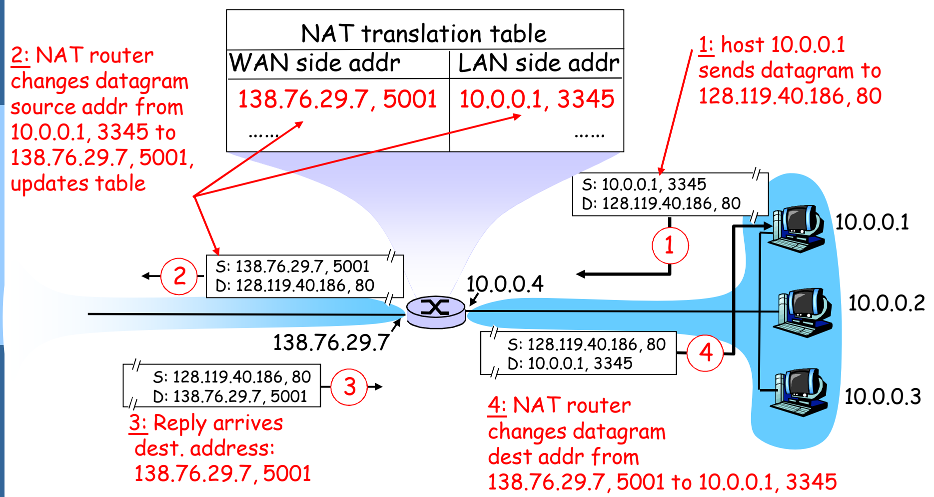
Datagrams with source or destination in this network have 10.0.0/24 address for source, destination (as usual)

NAT: Network Address Translation

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

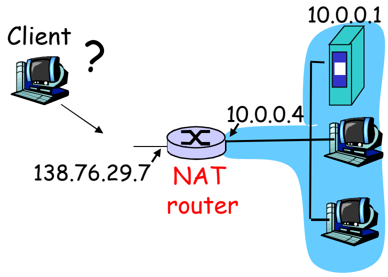


NAT: Network Address Translation

- 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, eg, P2P applications
  - address shortage should instead be solved by IPv6

NAT traversal problem

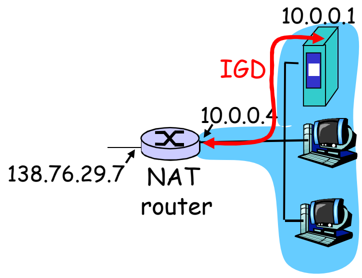
- client wants to connect to server with address 10.0.0.1
  - server address 10.0.0.1 local to LAN (client can't use it as destination addr)
  - only one externally visible NATted address: 138.76.29.7
- solution 1:
  - statically configure NAT to forward incoming connection requests at given port to server
  - e.g., (123.76.29.7, port 5001) always forwarded to 10.0.0.1 port 80



NAT traversal problem

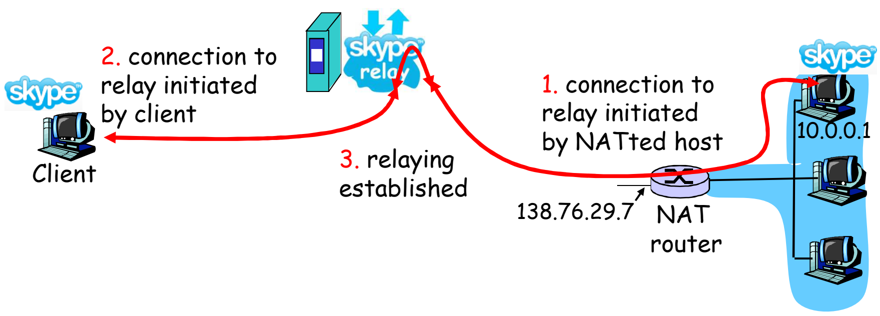
- solution 2:
  - Universal Plug and Play (UPnP) Internet Gateway Device (IGD) Protocol.
  - Allows NATted host to:
    - ❖ learn public IP address (138.76.29.7)
    - ❖ add/remove port mappings (with lease times)

i.e., automate static NAT port map configuration



NAT traversal problem

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



Internetworking

- Introduction
- What's inside a router
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - Datagram Forwarding
  - Address Resolution
  - ICMP
  - IPv6
- Routing algorithms
  - Link state, Distance Vector, Hierarchical routing
- Routing in the Internet
  - RIP
  - OSPF
  - BGP

Forwarding at intermediate router

SubnetNumber	Next Hop	Interface
128.96.34.0/25	Router R1	interface 0
128.96.34.128/25	Router R3	interface 1
128.96.33.0/24	Router R3	interface 1
...	...	...

Forwarding at intermediate router

SubnetNumber	SubnetMask	NextHop	Interface
128.96.34.0	255.255.255.128	Router R1	interface 0
128.96.34.128	255.255.255.128	Router R3	interface 1
128.96.33.0	255.255.255.0	Router R3	interface 1
...	...	...	

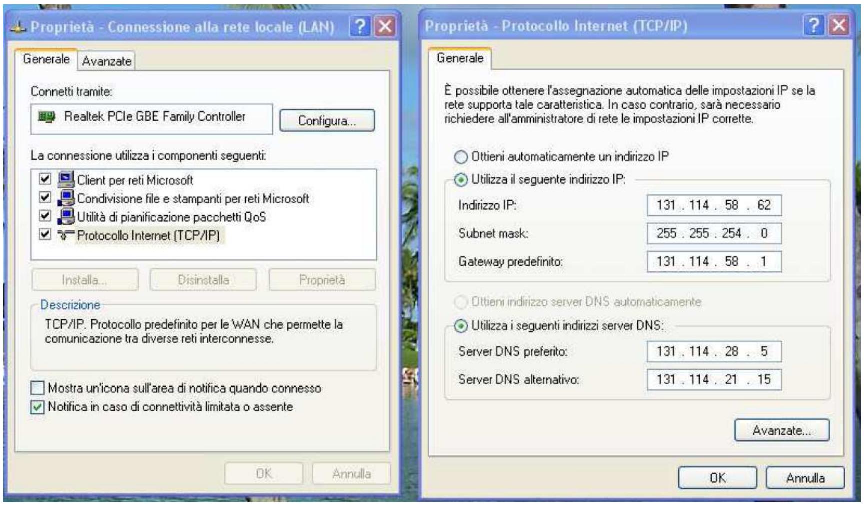
```
DHost=Destination IP Address
For each entry [i] in Table {
  DNet=(SubnetMask[i] & Dhost)
  If(DNet==SubnetNumber[i])
     then deliver datagram to NextHop[i] through Interface[i]
}
```


Forwarding at sending host

- The host knows
  - Subnet Mask (MySubnetMask)
  - Default router

```
SubnetNum=MySubnetMask & Dest_IP_Addr
If(SubnetNum ==MySubnetNum)
then deliver datagram to Dest_IP_Addr directly
else forward datagram to default router
```

Network Configuration

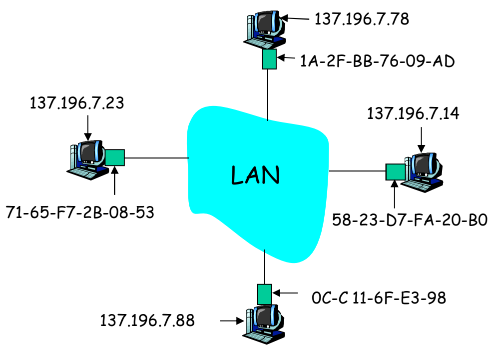


Internetworking

- Introduction
- What's inside a router
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - Datagram forwarding
  - Address resolution (ARP)
  - ICMP
  - IPv6
- Routing algorithms
  - Link state, Distance Vector, Hierarchical routing
- Routing in the Internet
  - RIP
  - OSPF
  - BGP

ARP: Address Resolution Protocol

How to determine MAC address of B knowing B's IP address?



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

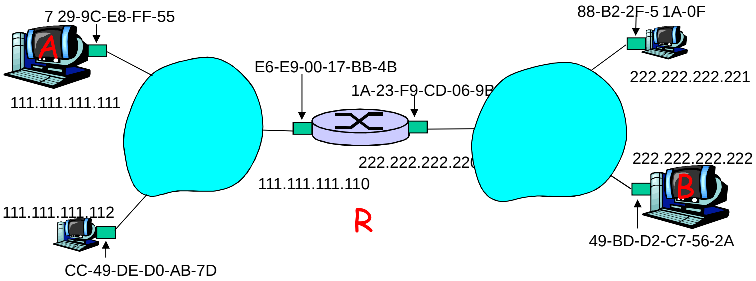
- A wants to send datagram to B, and B's MAC address not in A's ARP table.
- A broadcasts ARP query packet, containing B's IP address
  - dest MAC address = FF-FF-FF-FF-FF-FF
  - all machines on LAN receive ARP query
- B receives ARP packet
- B replies to A with its (B's) MAC address
  - frame sent to 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*

Routing to another Network

walkthrough: send datagram from A to B via R

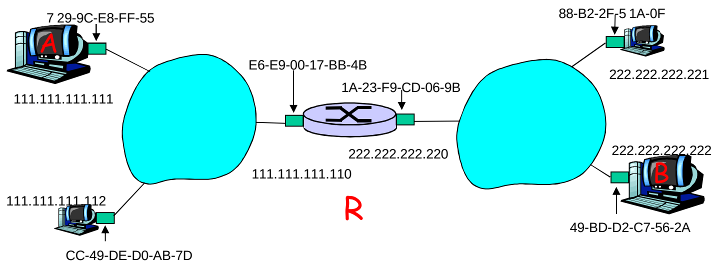
assume A knows B's IP address



- two ARP tables in router R, one for each IP network (LAN)

- A creates IP datagram with source A, destination B
- A uses ARP to get R's MAC address for 111.111.111.110
- A creates link-layer frame with R's MAC address as dest, frame contains A-to-B IP datagram
- A's NIC sends frame
- R's NIC receives frame
- R removes IP datagram from Ethernet frame, sees its destined to B
- R uses ARP to get B's MAC address
- R creates frame containing A-to-B IP datagram sends to B

This is a really important example – make sure you understand!

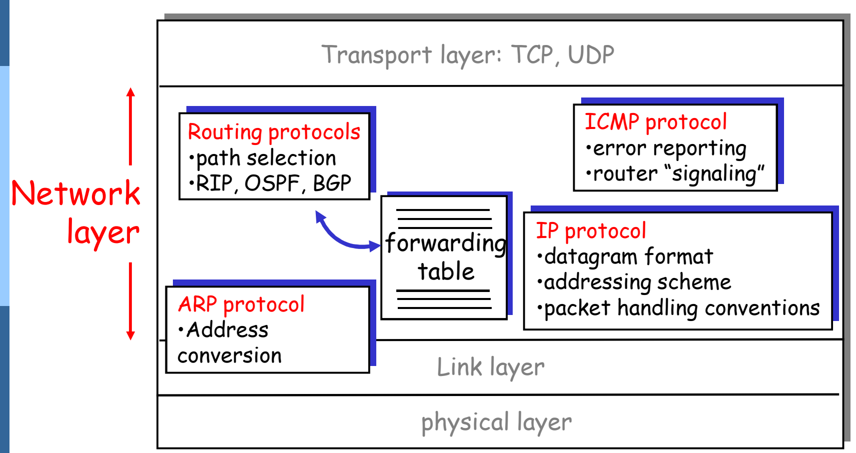


Internetworking

- Introduction
- What's inside a router
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - Datagram forwarding
  - Address resolution (ARP)
  - ICMP
  - IPv6
- Routing algorithms
  - Link state, Distance Vector, Hierarchical routing
- Routing in the Internet
  - RIP
  - OSPF
  - BGP

The Internet Network layer

Host, router network layer functions:



ICMP: Internet Control Message Protocol [RFC 792]

- 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
  - header + first 8 bytes of IP datagram causing error

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

Traceroute/Tracert

Traceroute/tracert: to www.unipi.it

Three delay measurements from source to www.unipi.it

```
Microsoft Windows 2000 [Versione 5.00.2195]
(C) Copyright 1985-1999 Microsoft Corp.

C:\>tracert www.unipi.it

Rilevazione instradamento verso www.unipi.it [131.114.190.24]
su un massimo di 30 punti di passaggio:

  1   <10 ms   <10 ms   <10 ms  rt50.univ.trieste.it [140.105.50.254]
  2   <10 ms   <10 ms   <10 ms  140.105.150.13
  3   <10 ms   <10 ms   <10 ms  utsgw48.univ.trieste.it [140.105.48.231]
  4    31 ms    31 ms    47 ms  rc-units2.ts.garr.net [193.206.132.29]
  5    31 ms    62 ms    47 ms  mi-ts-2.garr.net [193.206.134.53]
  6    47 ms    47 ms    47 ms  bo-mi-2.garr.net [193.206.134.6]
  7   125 ms   125 ms   125 ms  pi-bo-1.garr.net [193.206.134.82]
  8     *      204 ms   281 ms  unipi-rc.pi.garr.net [193.206.136.18]
  9   219 ms   312 ms   250 ms  eth03-gw.unipi.it [131.114.188.61]
 10   219 ms   187 ms   204 ms  131.114.186.1
 11   250 ms   266 ms   266 ms  solaria.adm.unipi.it [131.114.190.24]

Rilevazione completata.

C:\>
```

Traceroute and ICMP

- Source sends series of UDP segments to dest
  - First has TTL =1
  - Second has TTL=2,
  - …
  - Unlikely port number
- When n-th datagram arrives to n-th router:
  - Router discards datagram
  - And sends to source an ICMP message (type 11, code 0)
  - 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 "host unreachable" packet (type 3, code 3)
- When source gets this ICMP, stops.

Internetworking

- Introduction
- What's inside a router
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - Datagram forwarding
  - Address Resolution
  - ICMP
  - IPv6
- Routing algorithms
  - Link state, Distance Vector, Hierarchical routing
- Routing in the Internet
  - RIP
  - OSPF
  - BGP

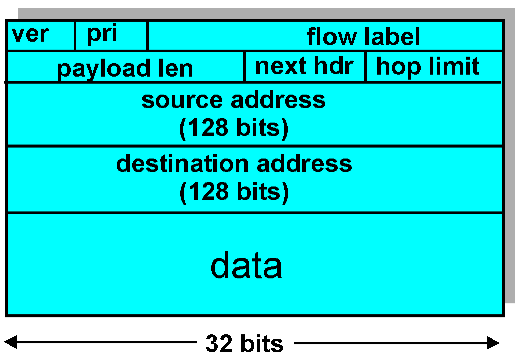
IPv6 [RFC 2460]

- **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



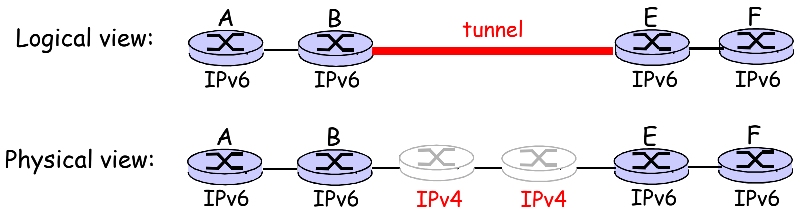
Other Changes from IPv4

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

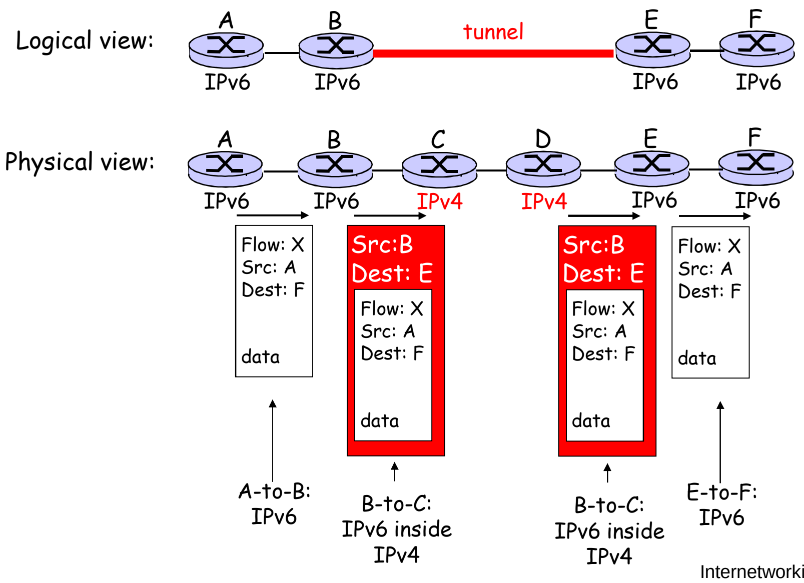
Transition From IPv4 To IPv6

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

Tunneling



Tunneling



Internetworking

- Introduction
- What's inside a router
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - Datagram forwarding
  - Address resolution (ARP)
  - ICMP
  - IPv6
- Routing algorithms
  - Link state, Distance Vector, Hierarchical routing
- Routing in the Internet
  - RIP
  - OSPF
  - BGP

Forwarding Table

SubnetNumber	SubnetMask	NextHop	Interface
128.96.34.0	255.255.255.128	Router R1	interface 0
128.96.34.128	255.255.255.128	Router R3	interface 1
128.96.33.0	255.255.255.0	Router R3	interface 1
...	...	...	

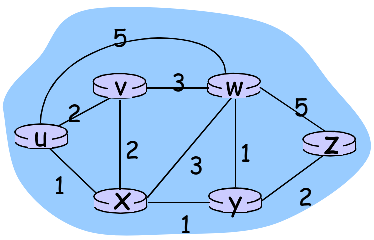
How is the Forwarding Table generated?

Routing

- Source Router
  - Default router of the source host
- Destination Router
  - Default router of the destination host

- Goal
  - Find a "good" path from the source router to the destination router

Graph abstraction



Graph: G = (N,E)

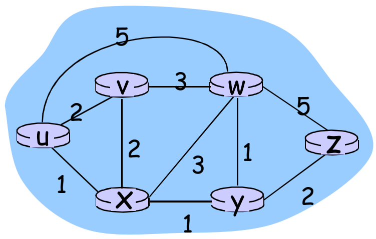
N = set of routers = { u, v, w, x, y, z }

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

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

Routing Algorithm classification

Global or decentralized?

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

Internetworking

- Introduction
- What's inside a router
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP
  - IPv6
- Routing algorithms
  - Link state, Distance Vector, Hierarchical routing
- Routing in the Internet
  - RIP
  - OSPF
  - BGP

A Link-State Routing Algorithm

Dijkstra's algorithm

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

Notation:

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

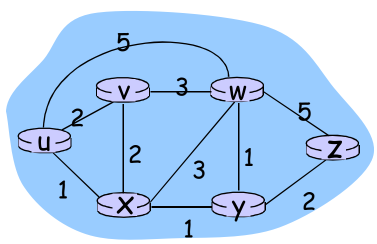
Dijsktra's Algorithm

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

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

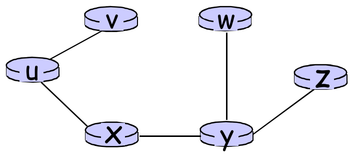

Dijkstra's algorithm: example

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



Dijkstra's algorithm: example (2)

Resulting shortest-path tree from u:



Resulting forwarding table in u:

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

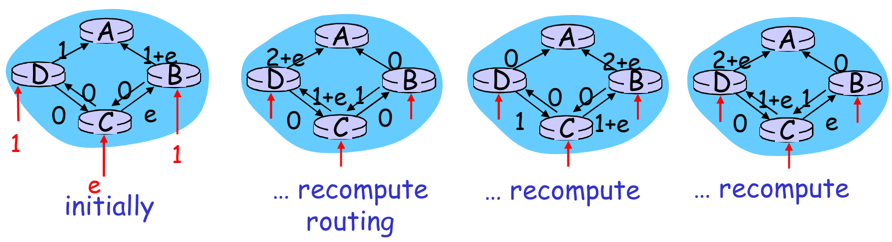
Dijkstra's algorithm, discussion

Algorithm complexity: n nodes

- each iteration: need to check all nodes, w, not in N
- n(n+1)/2 comparisons: $O(n^2)$
- more efficient implementations possible: O(nlogn)

Oscillations possible:

- e.g., link cost = amount of carried traffic



Internetworking

- Introduction
- What's inside a router
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP
  - IPv6
- Routing algorithms
  - Link state, Distance Vector, Hierarchical routing
- Routing in the Internet
  - RIP
  - OSPF
  - BGP

Distance Vector Algorithm

- Distributed
  - Each node receives information from neighboring nodes
  - performs calculation
  - Distributes the results of calculations to its neighboring nodes
- Iterative
  - The algorithm is self-terminating
- Asynchronous
  - Nodes do not need to operate synchronously

Distance Vector Algorithm

<u>Bellman-Ford Equation (dynamic programming)</u>

Define

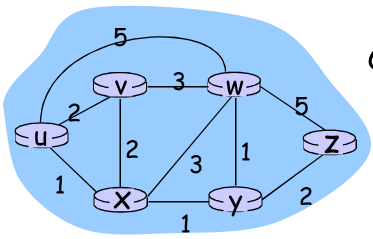
$d_x(y)$:= cost of least-cost path from x to y

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:

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

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

Distance Vector Algorithm

- $D_x(y)$ = estimate of least cost from x to y

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

Distance vector algorithm (4)

Basic idea:

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

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

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

Distance Vector Algorithm (5)

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:

Initialize data structures
Send DV to neighbors

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

recompute estimates

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

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

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

node x table

from \ cost to	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

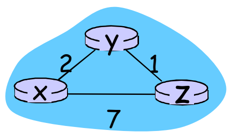
from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

node y table

from \ cost to	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

node z table

from \ cost to	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0



time

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

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

node x table

from \ cost to	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

node y table

from \ cost to	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

from \ cost to	x	y	z
x	0	2	7
y	2	0	1
z	7	1	0

from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

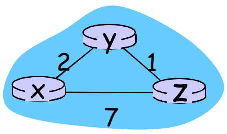
node z table

from \ cost to	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

from \ cost to	x	y	z
x	0	2	7
y	2	0	1
z	3	1	0

from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

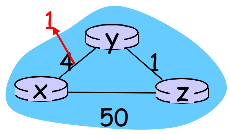
time



Distance Vector: link cost changes

Link cost changes:

- node detects local link cost change
- updates routing info, recalculates distance vector
- if DV changes, notify neighbors



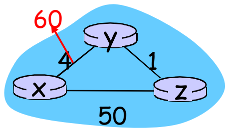
"good news travels fast"

At time t_0, y detects the link-cost change, updates its DV, and informs its neighbors.

At time t_1, z receives the update from y and updates its table. It computes a new least cost to x and sends its neighbors its DV.

At time t_2, y receives z's update and updates its distance table. y's least costs do not change and hence y does *not* send any message to z.

Distance Vector: link cost changes



Link cost changes:

- good news travels fast
- bad news travels slow - "count to infinity" problem!
- 44 iterations before algorithm stabilizes: see text

Poisoned reverse:

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

Comparison of LS and DV algorithms

Message complexity

- LS: with n nodes, E links, O(nE) msgs sent
- DV: exchange between neighbors only
  - convergence time varies

Speed of Convergence

- LS: $O(n^2)$ algorithm
  - may have oscillations
- DV: convergence time varies
  - may be routing loops
  - count-to-infinity problem

Robustness: what happens if router malfunctions?

LS:

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

DV:

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

Internetworking

- Introduction
- What's inside a router
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP
  - IPv6
- Routing algorithms
  - Link state, Distance Vector, Hierarchical routing
- Routing in the Internet
  - RIP
  - OSPF
  - BGP

Hierarchical Routing

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

… *not* true in practice

scale: with 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

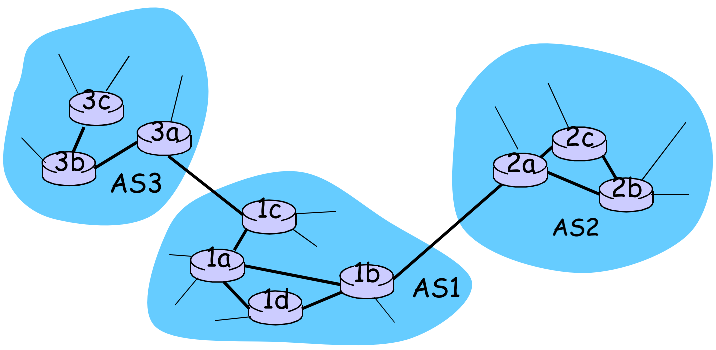
Hierarchical Routing

- 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

- Direct link to router in another AS

Interconnected ASes



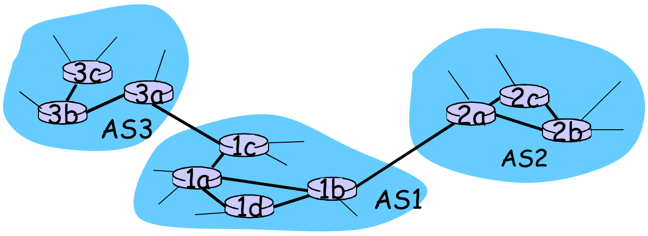
Inter-AS tasks

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

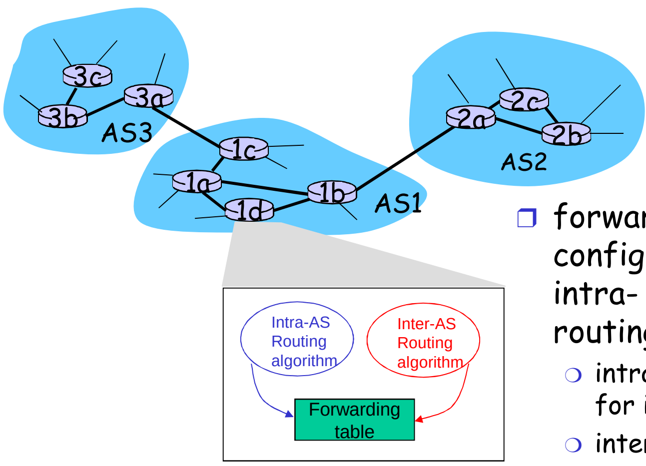
AS1 must:

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!



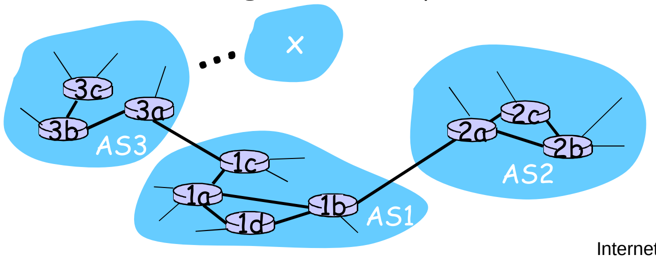
Forwarding Table



- 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

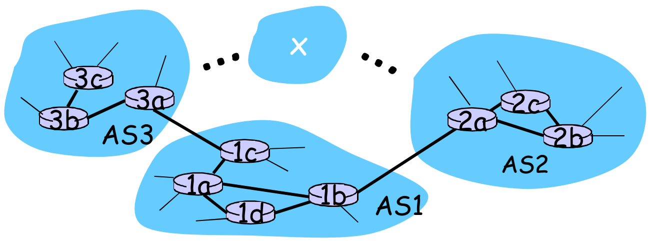
Example 1: Setting forwarding table in router 1d

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



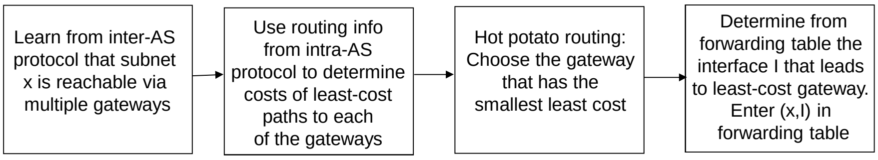
Example 2: Choosing among multiple ASes

- now suppose AS1 learns from inter-AS protocol that subnet *x* is reachable from AS3 (1c) *and* from AS2 (1b)
- to configure forwarding table, router 1d must determine towards which gateway it should forward packets for dest x.
  - Hot-potato routing
    - The router chooses the gateway router having the least-cost path from itself



Example 2: Choosing among multiple ASes

- now suppose AS1 learns from inter-AS protocol that subnet *x* is reachable from AS3 (1c) *and* from AS2 (1b)
- to configure forwarding table, router 1d must determine towards which gateway it should forward packets for dest x.
  - Hot-potato routing
    - The router chooses the gateway router having the least-cost path from itself



Internetworking

- Introduction
- What's inside a router
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP
  - IPv6
- Routing algorithms
  - Link state, Distance Vector, Hierarchical routing
- Routing in the Internet
  - RIP
  - OSPF
  - BGP

Intra-AS Routing

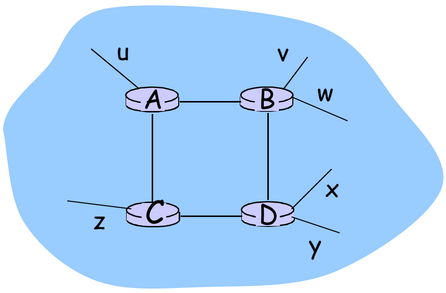
- also known as Interior Gateway Protocols (IGP)
- most common Intra-AS routing protocols:
  - RIP: Routing Information Protocol
  - OSPF: Open Shortest Path First
  - IGRP: Interior Gateway Routing Protocol (Cisco proprietary)
  - EIGRP: Extended Interior Gateway Routing Protocol (Cisco proprietary)

Internetworking

- Introduction
- What's inside a router
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP
  - IPv6
- Routing algorithms
  - Link state, Distance Vector, Hierarchical routing
- Routing in the Internet
  - RIP
  - OSPF
  - BGP

RIP (Routing Information Protocol) [RFC 1058, 2453]

- distance vector algorithm
- included in BSD-UNIX Distribution in 1982
- distance metric: # of hops (max = 15 hops)



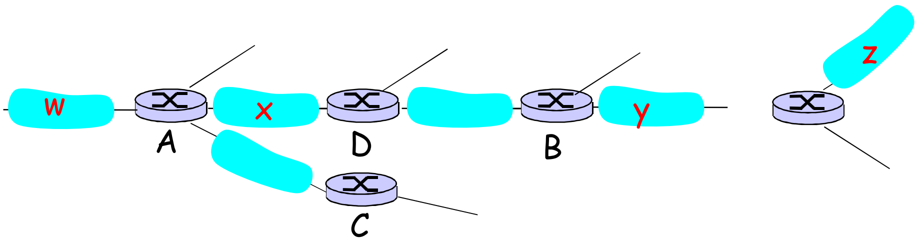
From router A to subnets:

Destination	Hops
u	1
v	2
w	2
x	3
y	3
z	2

RIP advertisements

- *distance vectors:* exchanged among neighbors every 30 sec via RIP Response Message (also called advertisement)
- each advertisement: list of up to 25 destination subnets within AS

RIP: Example



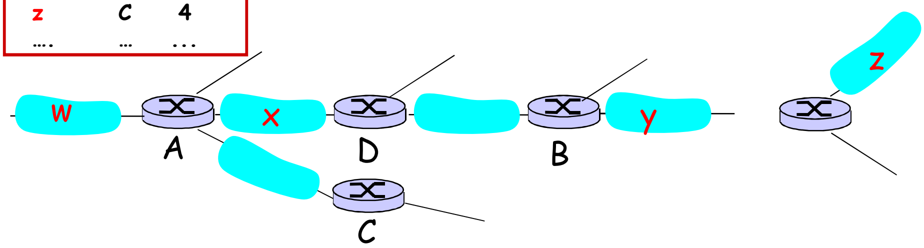
Destination Network	Next Router	Num. of hops to dest.
w	A	2
y	B	2
z	B	7
x	--	1
….	….	....

Routing table in D

RIP: Example

Dest	Next	hops
w	-	1
x	-	1
z	C	4
….	…	...

Advertisement from A to D



Destination Network	Next Router	Num. of hops to dest.
w	A	2
y	B	2
z	~~B~~ A	~~7~~ 5
x	--	1
….	….	....

Routing table in D

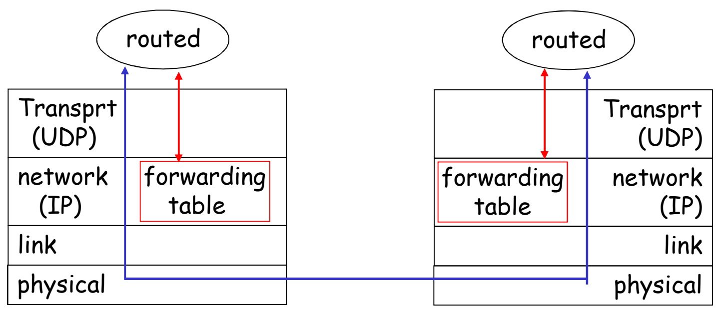
RIP: Link Failure and Recovery

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 propagates to entire net
- *poisoned reverse* used to prevent ping-pong loops (infinite distance = 16 hops)

RIP Table processing

- RIP routing tables managed by **application-level** process called routed (daemon)
- advertisements sent in UDP packets – port 520



Internetworking

- Introduction
- What's inside a router
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP
  - IPv6
- Routing algorithms
  - Link state, Distance Vector, Hierarchical routing
- Routing in the Internet
  - RIP
  - OSPF
  - BGP

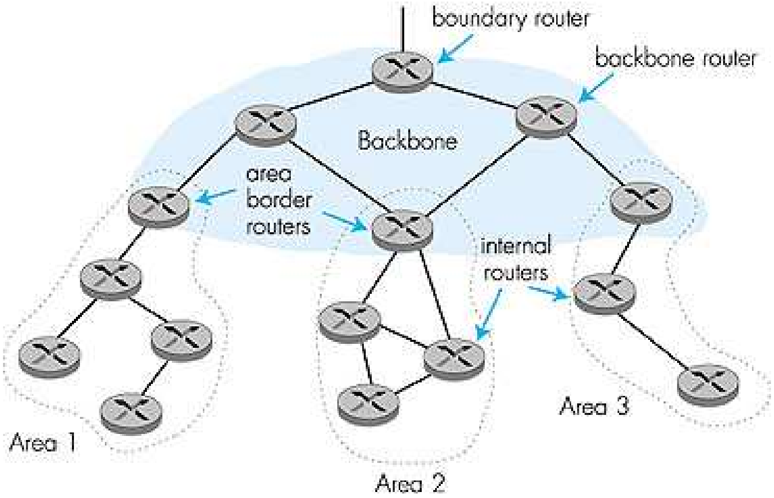
OSPF (Open Shortest Path First) [RFC 2328]

- "open": publicly available
- uses Link State algorithm
  - LS packet dissemination
  - topology map at each node
  - route computation using Dijkstra's algorithm
- advertisements disseminated to entire AS (via flooding)
  - carried in OSPF messages directly over IP (rather than TCP or UDP
  - At least every 30 minutes and whenever a change in the link state occurs

OSPF "advanced" features (not in RIP)

- security: OSPF messages can be authenticated (to prevent malicious intrusion)
  - No Authentication (default), Simple, MD5
- 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; high for real time)
- integrated uni- and multicast support:
  - Multicast OSPF (MOSPF) uses same topology data base as OSPF
- hierarchical OSPF in large domains.

Hierarchical OSPF



Hierarchical OSPF

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

Internetworking

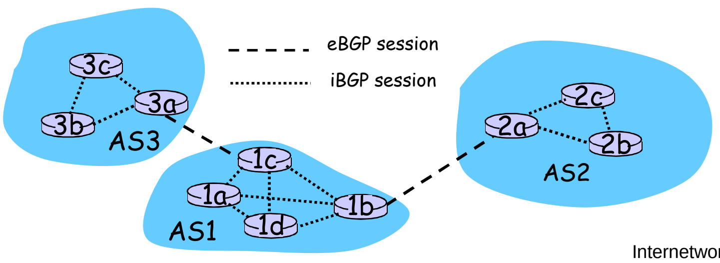
- Introduction
- What's inside a router
- IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP
  - IPv6
- Routing algorithms
  - Link state, Distance Vector, Hierarchical routing
- Routing in the Internet
  - RIP
  - OSPF
  - BGP

Internet inter-AS routing: BGP4 [RFC 4271]

- 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 reachability information to all AS-internal routers.
  3. Determine "good" routes to subnets based on reachability information and policy.
- allows subnet to advertise its existence to rest of Internet: *"I am here"*

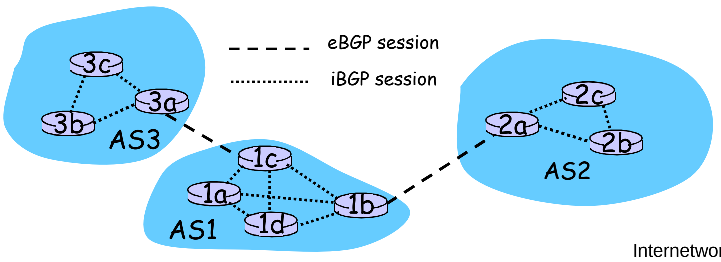
BGP basics

- pairs of routers (BGP peers) exchange routing info over TCP connections: BGP sessions
  - BGP sessions need not correspond to physical links.
- when AS2 advertises a prefix to AS1:
  - AS2 *promises* it will forward datagrams towards that prefix.
  - AS2 can aggregate prefixes in its advertisement



Distributing reachability info

- using eBGP session between 3a and 1c, AS3 sends prefix reachability info to AS1.
  - 1c can then use iBGP do distribute new prefix info to all routers in AS1
  - 1b can then re-advertise new reachability info to AS2 over 1b-to-2a eBGP session
- when router learns of new prefix, it creates entry for prefix in its forwarding table.



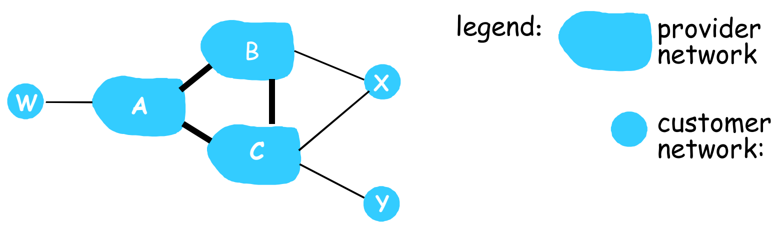
Path attributes & BGP routes

- advertised prefix includes BGP attributes.
  - prefix + attributes = "route"
- two important attributes:
  - AS-PATH: contains ASs through which prefix advertisement has passed: e.g, AS3, AS1
  - NEXT-HOP: the router interface that begins the AS path (e.g., router 3a)
    - Internal routers determine the least-cost path to Next-Hop (through intra-AS routing) to configure their FT
- when gateway router receives route advertisement, uses import policy to accept/decline.

BGP route selection

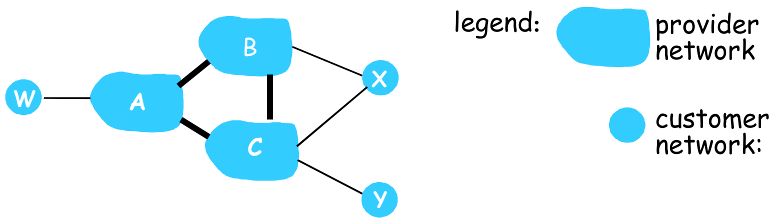
- router may learn about more than 1 route to some prefix. Router must select route.
- elimination rules:
  1. local preference value attribute: policy decision
  2. shortest AS-PATH
  3. closest NEXT-HOP router: hot potato routing
  4. additional criteria

BGP routing policy



- A,B,C are provider networks
- X,W,Y are customer (of provider networks) stub ASs
- 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

BGP routing policy (2)



- 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!

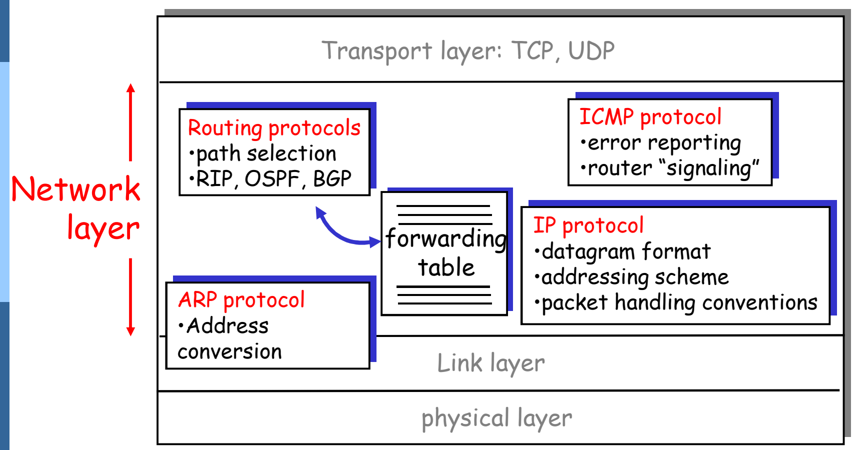
Why different Intra- and Inter-AS routing ?

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

Summary


Transport layer: TCP, UDP
Network layer
Routing protocols
•path selection
•RIP, OSPF, BGP
ICMP protocol
•error reporting
•router "signaling"
forwarding table
IP protocol
•datagram format
•addressing scheme
•packet handling conventions
ARP protocol
•Address conversion
Link layer
physical layer