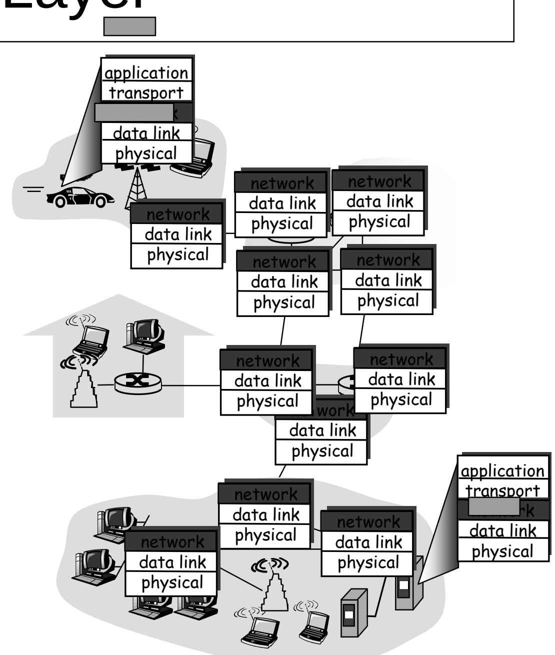
Distributed Systems

7. Network Layer

Werner Nutt

Network Layer

- Transports segments from sending to receiving host
- On sending side encapsulates segments into datagrams
- On receiving side, delivers segments to transport layer
- Network layer protocols in *every* host, router
- Router examines header fields in all IP datagrams passing through it



Key Network-Layer Functions

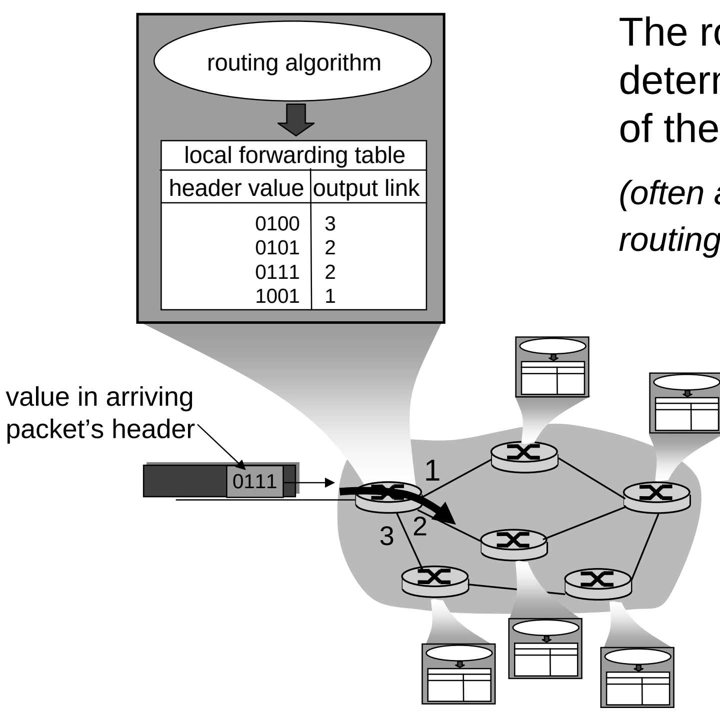
- *Forwarding:* move packets from router's input to appropriate router output
- *Routing:* determine route taken by packets from source to destination

→ *Routing Algorithms*

Analogy:

- Routing: process of planning trip from source to destination
- Forwarding: process of getting through single interchange

Routing is Implemented by Forwarding



The routing algorithm determines the content of the forwarding table

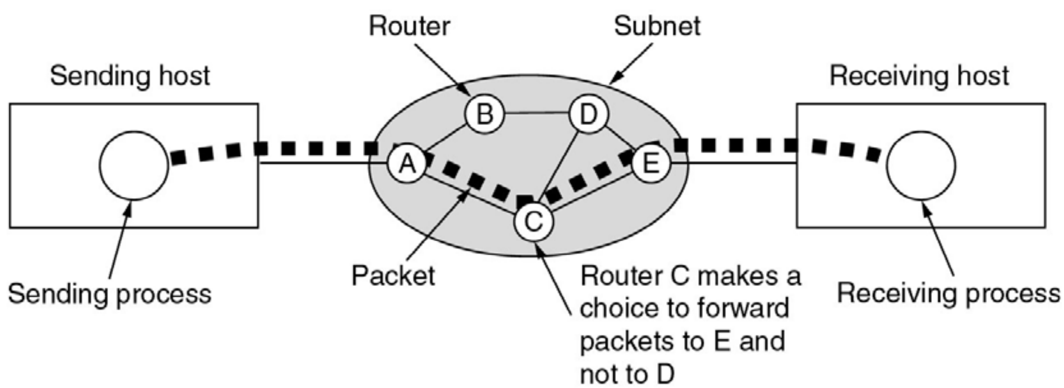
(often also called routing table)

Switching Schemes (1)

Network = nodes connected by links

- Broadcasts *(Ethernet, wireless)*
  - send messages to all nodes
  - nodes listen for (other and own) messages ("carrier sensing")
- Circuit switching *(phone networks)*
  - establish path through network
  - physical change in the network connections
- Packet routing *(Internet Protocol)*
  - "store-and-forward"
  - unpredictable delays

Data Transport Based on Packet Routing



Switching Schemes (2)

- Virtual Circuit Switching (Frame/cell relay, e.g., *ATM)*
  - small, fixed size packets (48 byte of data for ATM),
  - padded if necessary
  - “logical” circuit switching
  - bandwidth & latency guaranteed (“virtual path”)
  - forwarding based on inspection of first few bytes
  - avoids error checking at nodes (uses reliable links)

- ATM (= Asynchronous Transfer Mode)
  - used by ISPs to realize (A)DSL

Virtual Circuit Implementation

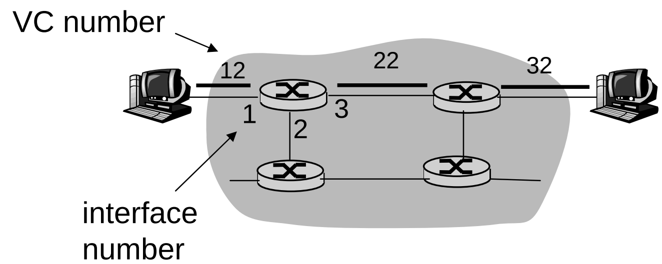
A virtual circuit (VC) consists of:

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

- A packet belonging to a VC carries VC number (rather than destination address)

- VC number can be changed on each link
  - new VC number comes from forwarding table

Virtual Circuit Forwarding Table



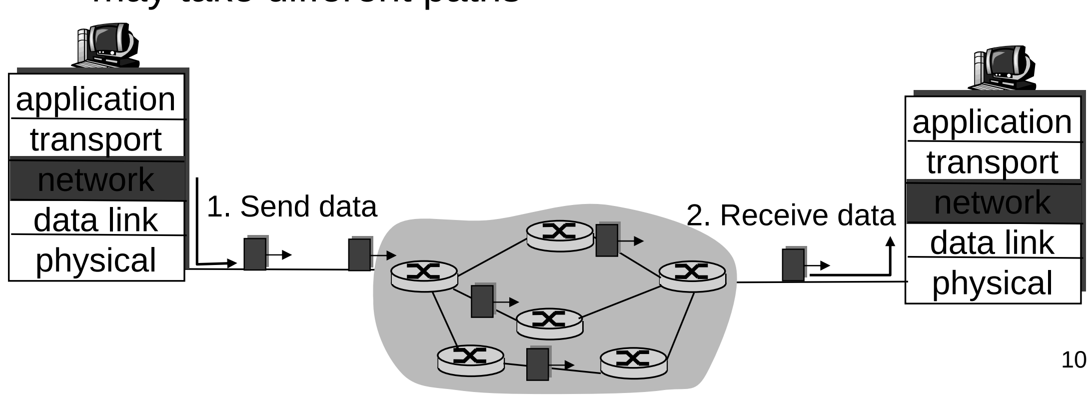
Forwarding table in router:

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

Routers maintain connection state information!
Initially, call set-up phase according to protocol!

Datagram Networks

- No call set-up at network layer
- Routers: no state about end-to-end connections
  - – no network-level concept of "connection"
- Packets forwarded using destination host address
  - – packets between same source-destination pair may take different paths



Forwarding Table

4 billion possible entries

Destination Address Range	Link Interface
11001000 00010111 00010000 00000000 through 11001000 00010111 00010111 11111111	0
11001000 00010111 00011000 00000000 through 11001000 00010111 00011000 11111111	1
11001000 00010111 00011001 00000000 through 11001000 00010111 00011111 11111111	2
otherwise	3

Longest Prefix Matching

Prefix Match	Link Interface
11001000 00010111 00010	0
11001000 00010111 00011000	1
11001000 00010111 00011	2
otherwise	3

Network numbers

Examples

DA: 11001000 00010111 00010110 10100001 — Which interface?

DA: 11001000 00010111 00011000 10101010 — Which interface?

Datagram or VC Network: Why?

Internet (datagram)

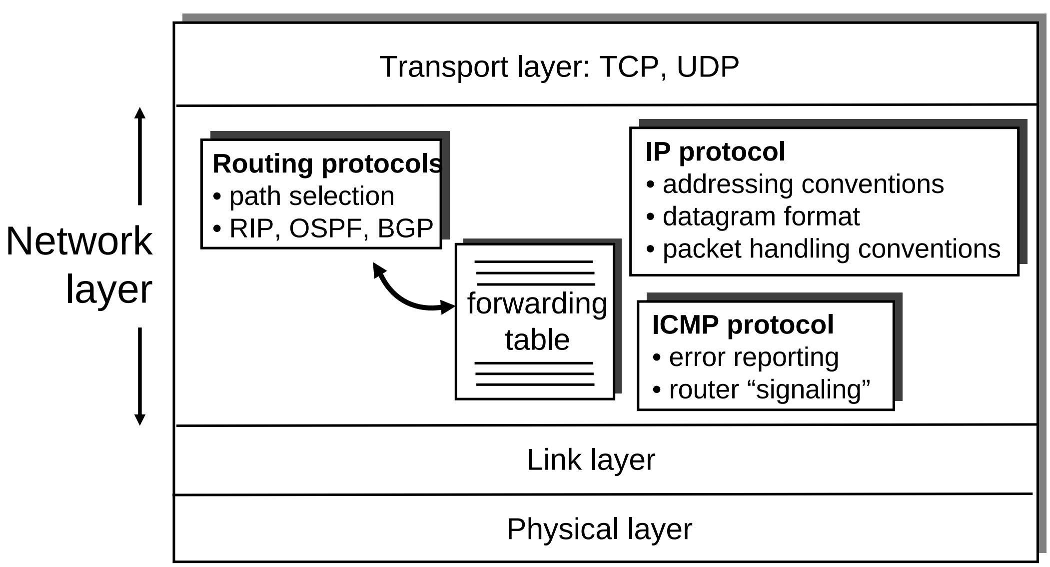
- data exchange among computers
  - "elastic" service, no strict timing requirements
- "smart" end systems (computers)
  - can adapt, perform control, error recovery
  - simple inside network, complexity at "edge"
- many link types
  - different characteristics
  - uniform service difficult

ATM (VC)

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

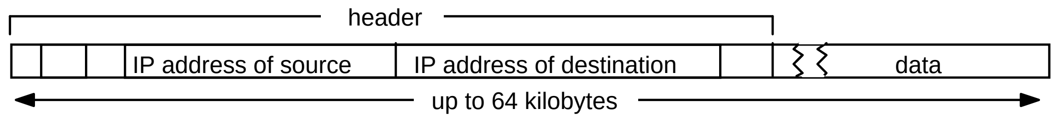
The Internet Network Layer

Host, router functions at the network layer:

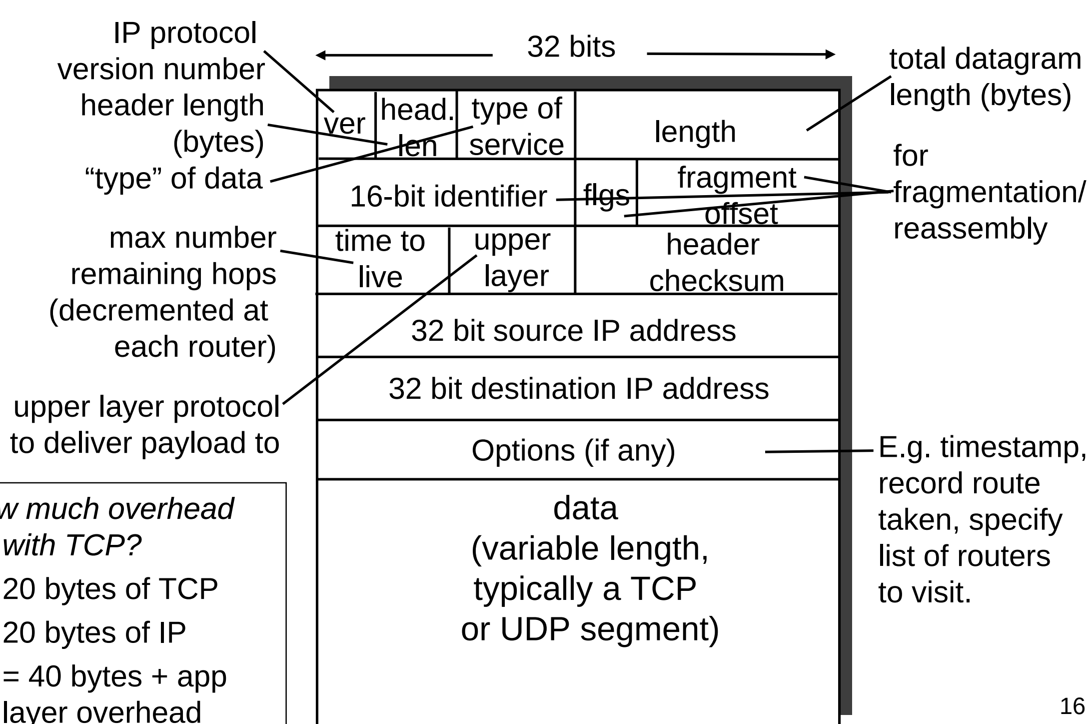


Internet Protocol (IP)

- Enables hosts to send packets to other hosts
- Layout of an IP packet



IP Datagram Format



How much overhead with TCP?

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

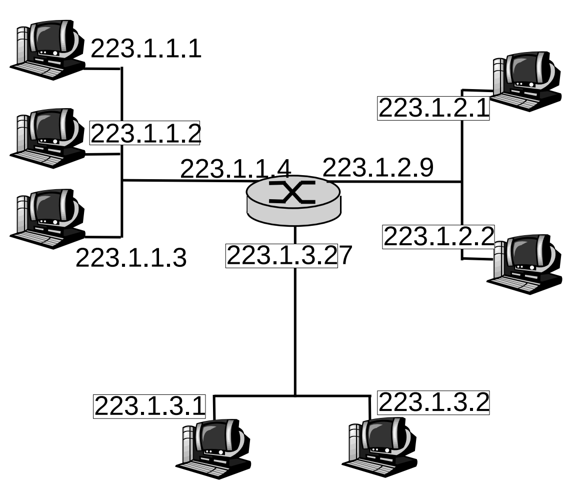
IP Addressing

Address format:

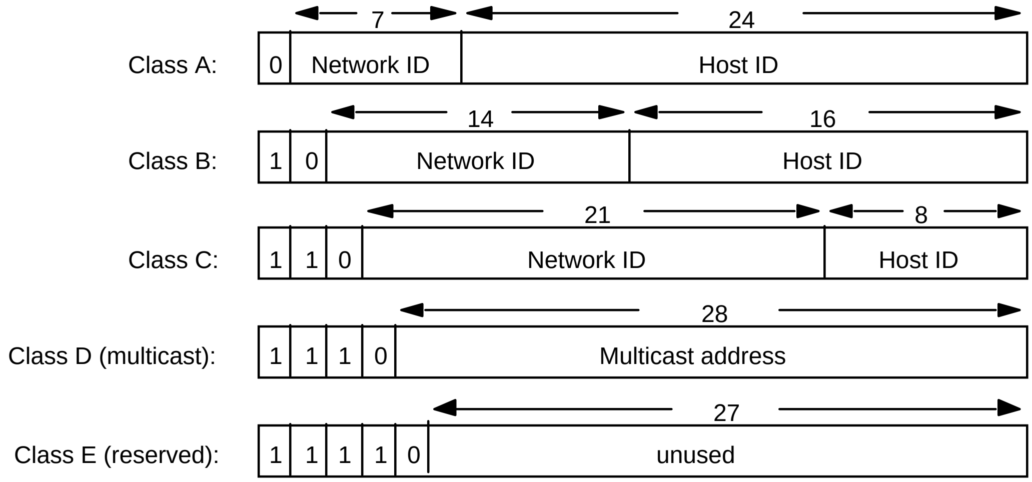
- 32 bits = 4 bytes (octets)
- Representation in “dotted decimal” notation
  193.206.186.140
- Representation in hexadecimal code
  0xc1ceba8c
- Representation in bit code
  11000001 11001110 10111010 10001100

IP Addressing (cntd)

- IP address: identifies host and router *interfaces*
- Interface: connection between host/router and physical link
  – routers typically have multiple interfaces
  – host typically has one interface
  – IP addresses associated with each interface



IP Addresses



Originally, IP addresses were divided into classes …

Hosts belong to Networks, Addresses Belong to Network Ranges

MIT Network	18.0.0.0 - 18.255.255.255
Unibz Network	193.206.186.0 - 193.206.186.255
Yahoo Network	69.147.64.0 - 69.147.127.255

- How do we describe network ranges?
- Note: all addresses in a range
  - agree on their first N bits (network prefix)
  - vary on the remaining 32-N bits (host address)
- CIDR Notation (CIDR = Classless Interdomain Routing)
  - MIT Network 18.0.0.0/8
  - Unibz Network 193.206.186/24
  - Yahoo Network 69.147.64.0/18

IP Addressing: CIDR

- **CIDR**: **C**lassless **I**nter**D**omain **R**outing
  - subnet portion of address of arbitrary length
  - address format: a.b.c.d/x, where x is # bits in subnet portion of address

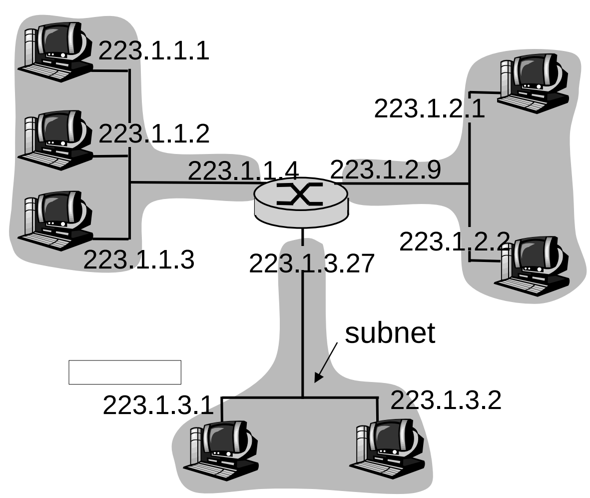
subnet part | host part

11001000 00010111 00010000 00000000

200.23.16.0/23

Subnets

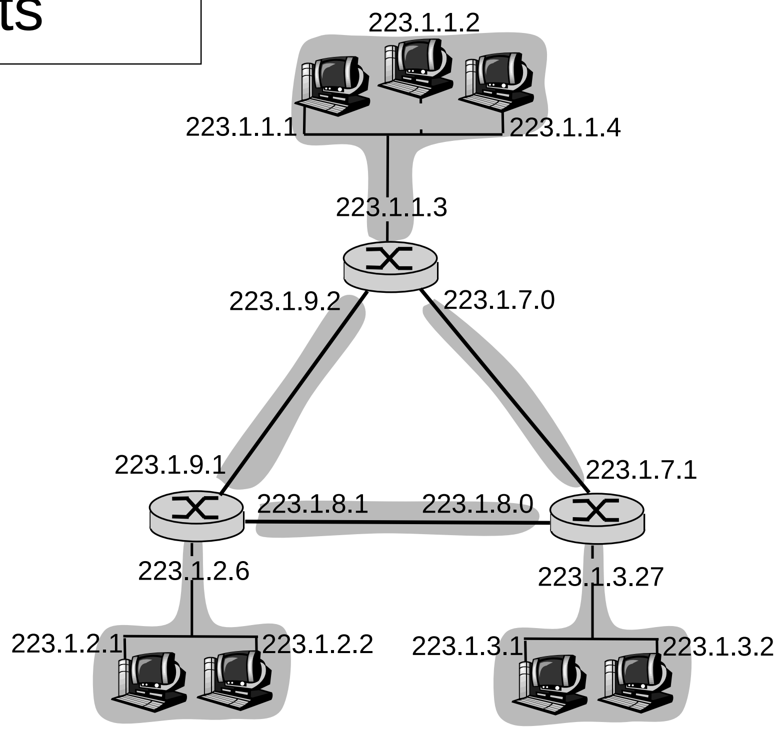
- IP address:
  - subnet part (high order bits)
  - host part (low order bits)
- *What's a subnet ?*
  - device interfaces with same subnet part of IP address
  - can physically reach each other without intervening router



Network consisting of 3 subnets

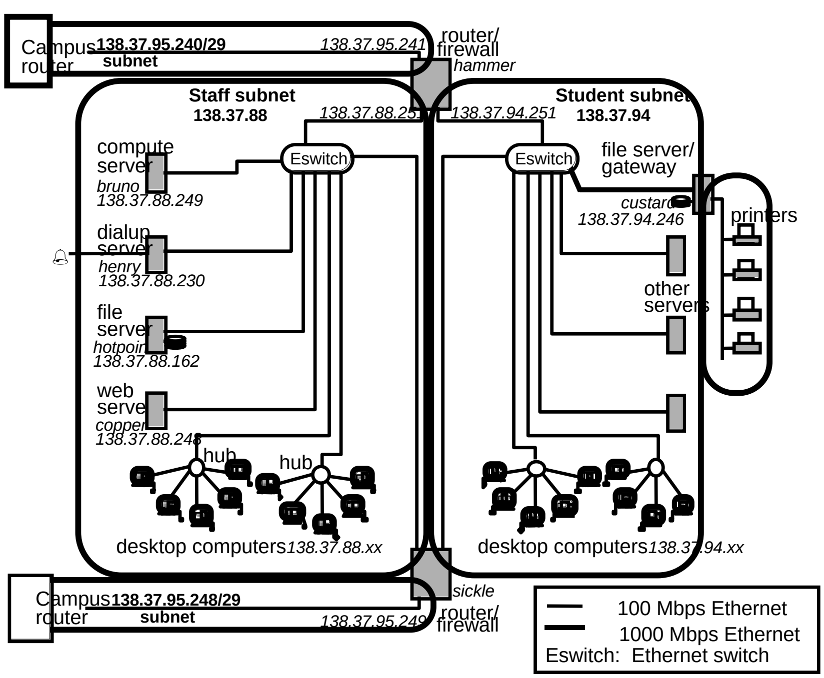
Subnets

How many?



A University LAN With Public IP Addresses

Simplified view of the Queen Mary and Westfield College Computer Science network



(Ethernet at FUB is switched)

Subnet Masks

Suppose: A packet for destination 193.206.186.140 arrives at router

How does the router know to which network the packet should go?

Routers have two pieces of information per network entry

- Network address 193.206.186.0
- 32 bit mask 255.255.255.0

(represents number of significant bits as in CIDR notation)

Algorithm:

For each network entry
compute: (destination address) AND (subnet mask)
if result = network address, then destination in network

Special Addresses

- Address ranges for private networks *(no routing over the Internet!?)*:

  10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16

- Network address: lowest number in range
- Broadcast address: highest number in range
- Gateway address: often second highest number in range
- Loopback network: 127.0.0.0/8
  virtual interface connection a host to itself
- Localhost: 127.0.0.1

IP Address Quiz

- How many possible subnet masks are there?
- What are the possible numbers that can occur in a mask position?
- What is the network mask of the Stanford Univ. network (171.64.0.0/14)?
- How many addresses are there on the Stanford network?
- Which of the following addresses could belong to a host at Stanford:
  171.74.212.31 ?
  171.68.0.31 ?
  171.67.212.44 ?
- Host **`actarus.inf.unibz.it`** has the address 10.10.20.5 and mask 255.255.252.0.

  What is the broadcast address on that host's network?
  What is (probably) the gateway address?

IP Addresses: How to Get One?

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

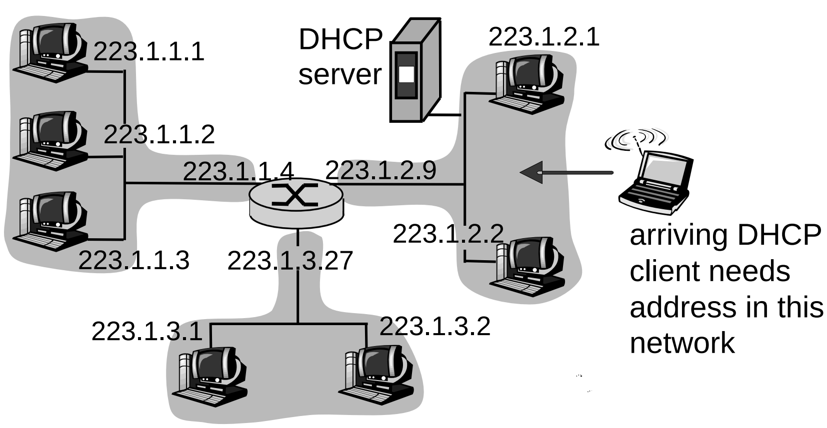
- Hard-coded by system administrator in a file
  - Windows: control-panel->network connections-> local area connections -> properties
  - LINUX (Debian/Ubuntu): /etc/network/interfaces
- DHCP: Dynamic Host Configuration Protocol: dynamically get address from as server
  - “plug-and-play”

DHCP: Dynamic Host Configuration Protocol

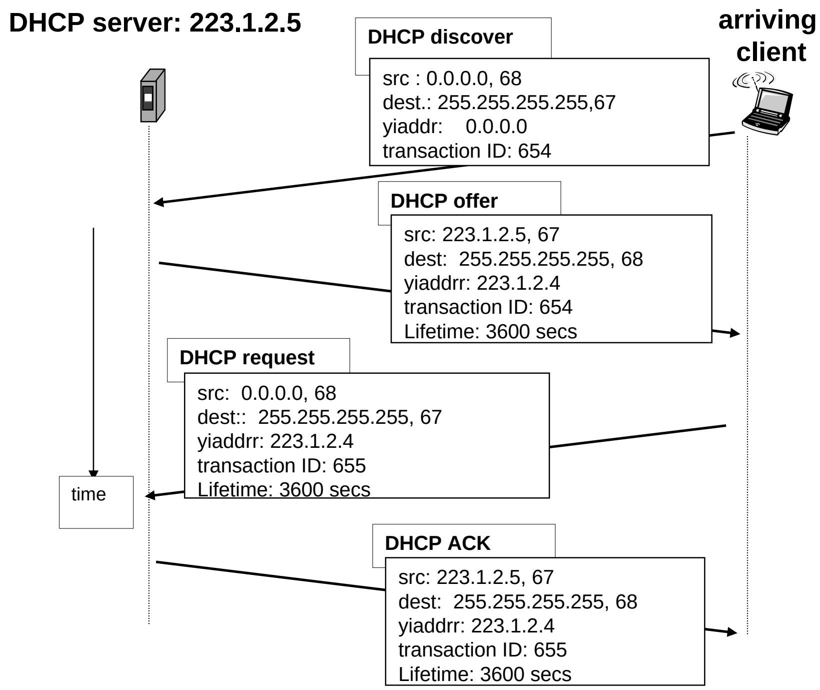
Goal: Allow a host to *dynamically* obtain its IP address from a network server when it joins network.

- Can renew its lease on address in use
- Allows reuse of addresses
  (only hold address while connected and “on”)
- Support for mobile users who want to join network
- DHCP overview:
  - host broadcasts “DHCP discover” message *[optional]*
  - server responds with “DHCP offer” message *[optional]*
  - host requests IP address: “DHCP request” message
  - server sends address: “DHCP ack” message

DHCP Client-Server Scenario



DHCP Client-Server Scenario



DHCP: More Than IP Address

DHCP returns more than just
the allocated IP address on subnet:

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

IP Addresses: How to Get One?

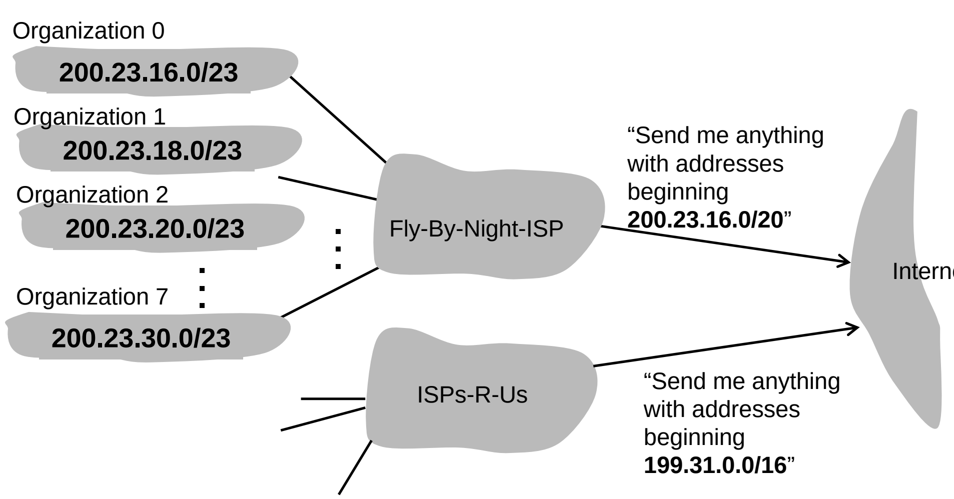
Q: How does a *network* get the subnet part of the IP addresses (i.e., its network address)?

A: From the range allocated to its 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

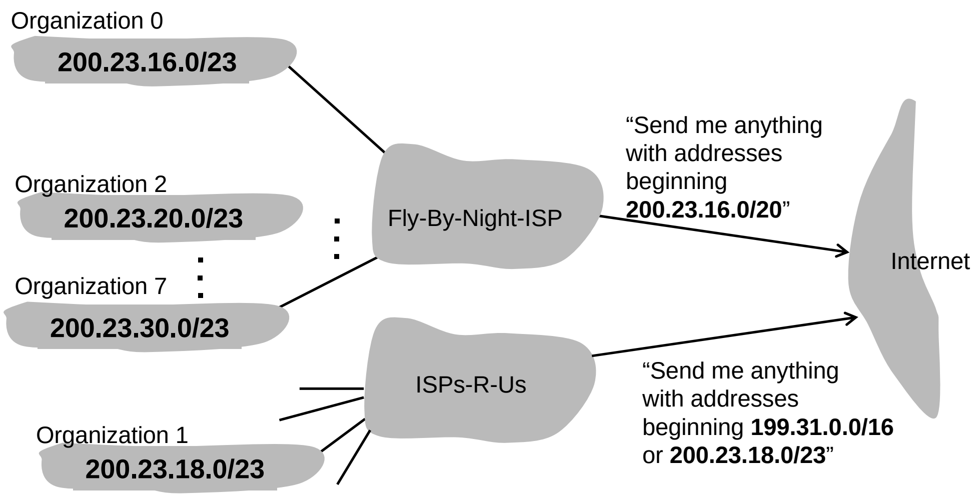
Hierarchical Addressing: Route Aggregation

Hierarchical addressing allows efficient advertisement of routing information:



Hierarchical Addressing: More Specific Routes

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



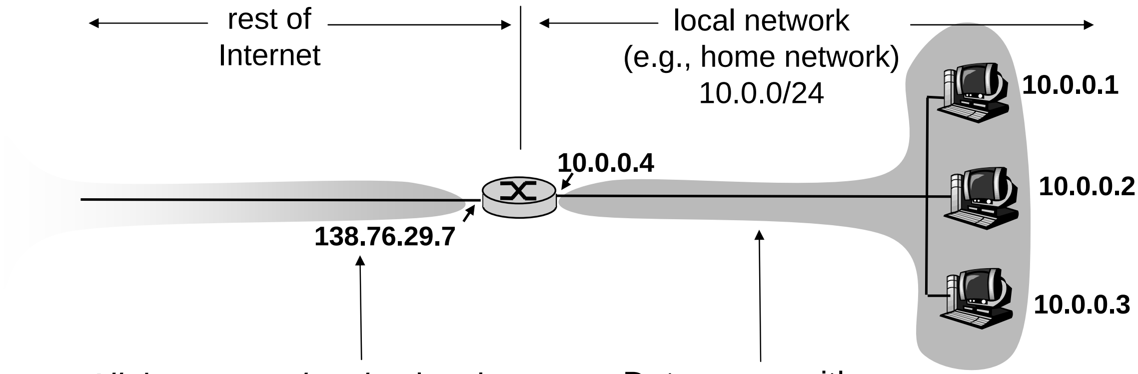
IP Addressing: The Last Word...

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

NAT: Network Address Translation



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

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

NAT: Network Address Translation

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

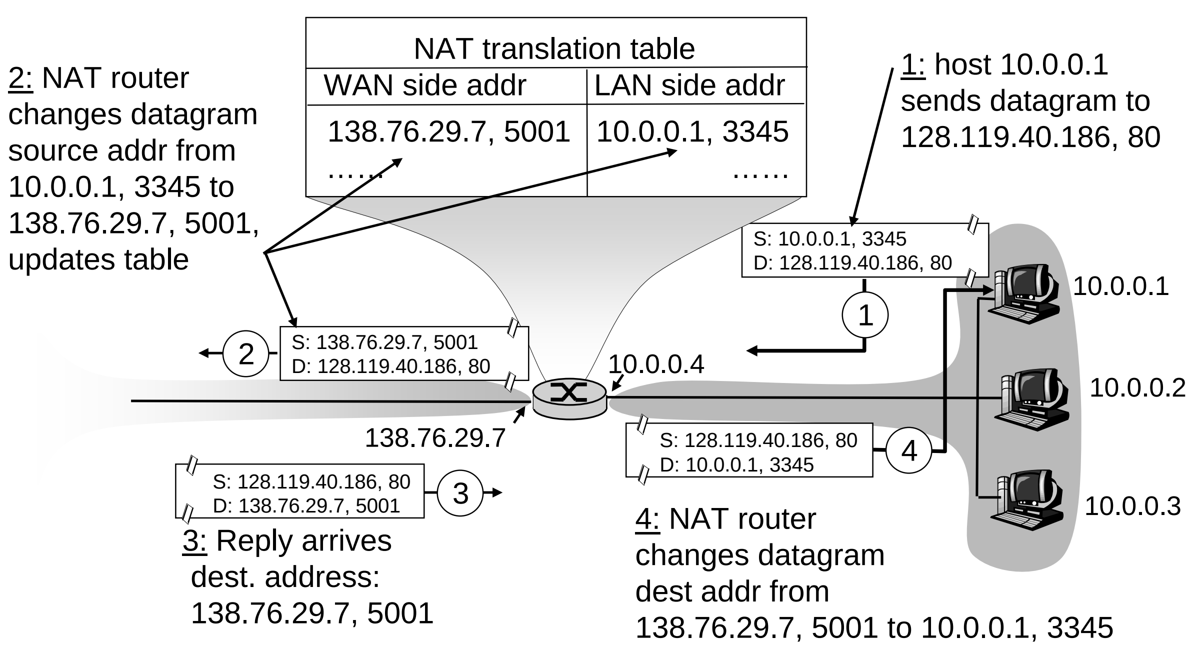
- range of addresses not needed from ISP: just one IP address for all devices
- can change addresses of devices in local network without notifying outside world
- can change ISP without changing addresses of devices in local network
- devices inside local network not explicitly addressable, visible by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

NAT: Network Address Translation

Implementation: NAT router must:

- *outgoing datagrams: replace* (source IP address, port #) of every outgoing datagram with (NAT IP address, new port #)
  . . . remote clients/servers will respond using
  (NAT IP address, new port #) as destination address
- *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 destination fields of every incoming datagram with corresponding (source IP address, port #) stored in NAT table

NAT: Network Address Translation



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

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)
  - 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

Routing Information on a Host

Every IP capable host needs to know about at least two classes of destinations

- locally connected computers
- everywhere else

Routing table on actarus:

```
wnutt@actarus:~$ netstat -r
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
10.10.20.0      *               255.255.252.0   U         0 0          0 eth0
10.10.112.0     *               255.255.240.0   U         0 0          0 eth1
default         10.10.23.254    0.0.0.0         UG        0 0          0 eth0
```


Address Resolution

Running ARP (= Address Resolution Protocol),
a host finds out the MAC address belonging to an IP address

ARP Steps

- actarus wants to send an IP packet to 10.10.23.254
- actarus broadcasts an ARP request:
  "I have IP address 10.10.20.5 and MAC address 00:11:85:e8:ff:8f, who has IP address 10.10.23.254?"
- Gateway sends an ARP reply to actarus:
  "I have IP address 10.10.23.254 and MAC address 00:10:db:bd:ce:87"

Optimization

- Hosts keep a cache
- Hosts overhearing a request update their cache
- ARP announcements: hosts send an ARP request to themselves (why?)

IP Routing (1)

Problem: Host H1 wants to send a packet to host H2

Case 1: H2 is on the same LAN (e.g., Ethernet) as H1

Approach:

- H1 finds out the Ethernet address of H2 (MAC address)
  (physical address, unique in the world
  for every Ethernet-enabled device)
- Ethernet module of H1 sends out the packet
  in Ethernet format

IP Routing (2)

Case 2: H2 is on a different LAN

Approach:

- H1 sends packet to its local gateway (say, G1)
- G1 sends packet across intermediate networks to the network of H2

If a gateway receives a packet, where should it send it?

Routing Problem

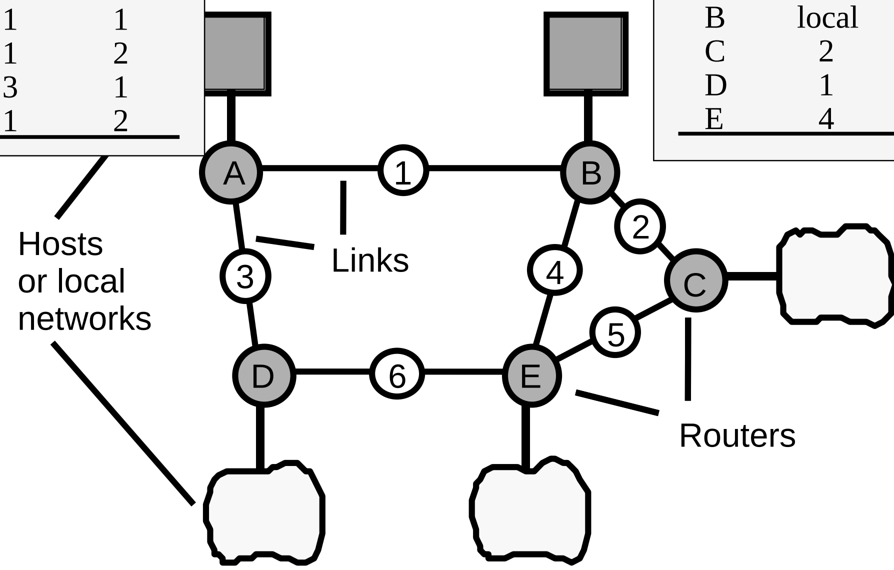
- What is a good path from H1 to H2?
- What is the next step on the path?

Computers forwarding packets through a network are called routers

Routing: Example

Routings from A		
To	*Link*	*Cost*
A	local	0
B	1	1
C	1	2
D	3	1
E	1	2

Routings from B		
To	*Link*	*Cost*
A	1	1
B	local	0
C	2	1
D	1	2
E	4	1



Routing Tables

Routings from A		
To	*Link*	*Cost*
A	local	0
B	1	1
C	1	2
D	3	1
E	1	2

Routings from B		
To	*Link*	*Cost*
A	1	1
B	local	0
C	2	1
D	1	2
E	4	1

Routings from C		
To	*Link*	*Cost*
A	2	2
B	2	1
C	local	0
D	5	2
E	5	1

Routings from D		
To	*Link*	*Cost*
A	3	1
B	3	2
C	6	2
D	local	0
E	6	1

Routings from E		
To	*Link*	*Cost*
A	4	2
B	4	1
C	5	1
D	6	1
E	local	0

Sample Routes

- Send from C to A:
  - – to link 2, arrive at B
  - – to link 1, arrive at A

- Send from C to A if B's table is modified to:

Routings from B

To	*Link*	*Cost*
B	local	0
C	2	1
E	4	1
default	5	-

- – to link 5, arrive at E
- – to link 4, arrive at B
- – to link 1, arrive at A

Note the extra hop.

Approaches to Routing Algorithms

Decentralised

- a router communicates with its immediate neighbors

➔Distance Vector algorithm (Bellman, Ford, Fulkerson)

- – realised in Router Information Protocol (RIP)

Global

- a router knows all routers in the network, their links, and the cost of sending a packet over a link

  (also called: link state protocols)

➔ Shortest Path algorithm (Dijkstra),

- – realised in Open Shortest Path First (OSPF) protocol

Distance Vector Routing: Principles

- Each router R maintains a routing table *(= distance vector)*, which records for each other router how far away it is from R *(e.g., how many hops)*
- The initial table of R has only one element: (R,local,0)
- Periodically, or when there is a change in its neighbourhood, a router sends its table to its neighbours
- When receiving a table, a router updates its local table
- When a link to a neighbour fails, the cost of the link is set to ∞

How does a router know that a link has failed?

Distance Vector Algorithm: Idea

Update: Every *t* seconds or when local table changes, send the full table to each accessible neighbor.

Propagation: When receiving an update from neighbor N

- if N knows a path to a new destination D, send messages for D to N
- if N knows a cheaper path to D, send messages for D to N
- if N is closer to D *(i.e., messages for D are sent to N)*, update cost for D

(Idea: N has better information about D)

See next slide for details

Distance Vector Algorithm (Pseudo Code)

Send: Every *t* seconds or when local table *Tl* changes,
send *Tl* on each non-faulty outgoing link

Receive: Whenever a routing table *Tr* is received on link *n*:

```
for all rows Rr in Tr {// modify Rr for subsequent comparisons
    if (Rr.link ≠ n) {
        Rr.cost = Rr.cost + 1;
        Rr.link = n;
        if (Rr.destination is not in Tl) add Rr to Tl;
         // add new destination to Tl
        else for all rows Rl in Tl {
            if (Rr.destination = Rl.destination and
               (Rr.cost < Rl.cost or Rl.link = n)) Rl = Rr;
            // Rr.cost < Rl.cost : remote node has better route
            // Rl.link = n : remote node is more authoritative
        }
    }
}
```


Distance Vector Routing: Convergence

- After initialisation, all routers reach a state where all tables are correct
  (i.e., show next hop along shortest path)
- Similarly, after a new router has joined
- However, convergence is slow

Distance Vector Routing: Looping

- When links fail, tables may be updated in a way that leads to loops

  rare situation, caused by delayed messages

- Routers in a loop continuously update their tables, increasing the cost (“count to infinity”)

- Solution (among others): make infinity small

  RIP: $\infty = 16$

Distance Vector Routing: Protocols

- RIP was the first Internet routing protocol

- Not scalable

- Replaced by a link state protocol

Link State Routing: Principles

- A router knows its neighbourhood, i.e.,
  - the routers it is linked to
  - the cost of the links
- Periodically, it broadcasts a map of its neighbourhood
  (the neighbourhood maps have timestamps)
- Each router
  - builds a global map, using the latest neighbourhood maps
  - computes the shortest path to each other router
- Routing table:
  - for each R, show the first hop on the shortest path to R

Dijkstra's Algorithm (1)

Input:

- graph G = (V,E)
- weight function w: E $\rightarrow$ R
- start node s $\in$ V

Output:

- function d: V $\rightarrow$ R
  - v.d is the distance from s to v (= length of shortest path)
- function pred: V \ {s} $\rightarrow$ V
  - v.pred is the predecessor of v
    on the shortest path from s to v

Dijkstra's Algorithm (2)

Input: $G = (V,E)$, $w: E \to R$, $s \in V$

Output: $d: V \to R$, pred: $V \setminus \{s\} \to V$

Ideas:

- Initial pessimistic estimates:
  – v.d = ∞ for all v ∈ V
  – v.pred = null
- Loop:
  – improve estimate of d
  – find candidate for pred
  – determine vertex v such that v.d is exact (and also v.pred)

Dijkstra's Algorithm (Pseudo Code)

```
Input: V, E, w, s

S = ∅, Q = V;          // Initialisation

For each vertex v ∈ V {
   v.d = ∞;
   v.pred = null }
s.d = 0;
While Q is not empty {     // Algorithm
   u = extractMin(Q);       // extract a vertex u for which
                            // u.d is minimal
   S = S ∪ {u};
   For each edge (u,v) outgoing from u {
      if (u.d + w(u,v) < v.d) {   // Relax v.d
        v.d = u.d + w(u,v);
        v.pred = u}
      }
   }
```

Dijkstra's Algorithm: Discussion

- If u = extractMin(Q), then the estimates for u are correct
- Shortest path from s to v:
  follow pred links
- Runtime
  - each vertex and each edge are visited only once
    ➔ total runtime = O(|E| + |V| x runtime(extractMin))
  - runtime of extractMin depends on implementation: O(log V) possible
    ➔ total runtime = O((|E| + |V|) x log(V))
- Incremental versions: needed to update routing tables

Routing: How Can All this Work?

The Internet is too large to be captured in one routing table

➔ Divide and Conquer

The Internet is divided into Autonomous Systems (ASs)
(= network with common routing protocol, e.g., RIP or OSPF)

Hierarchical Routing

- Granularity of Internet routing = ASs
- Internal traffic of an AS: finegrained routing
- Outbound traffic: send to (suitable) gateway
- At AS level: apply Boundary Gateway Protocol (BGP)
- Inbound traffic = internal traffic

Which Route Do My Packets Take?

- Unix/Linux: **`traceroute`**
- Windows: **`tracert`**

Example: **`tracert www.yahoo.com`**

How does it work?

- A packet has a time to live (TTL)
  Initially: TTL = 64 hops
- If a packet dies (TTL = 0 hops), most routers send error message back to source (ICMP “time exceeded” packet)
- Iteratively, send packets with TTL = 1, TTL = 2, …