Redes de Comunicação 2023/2024

T04 Network layer

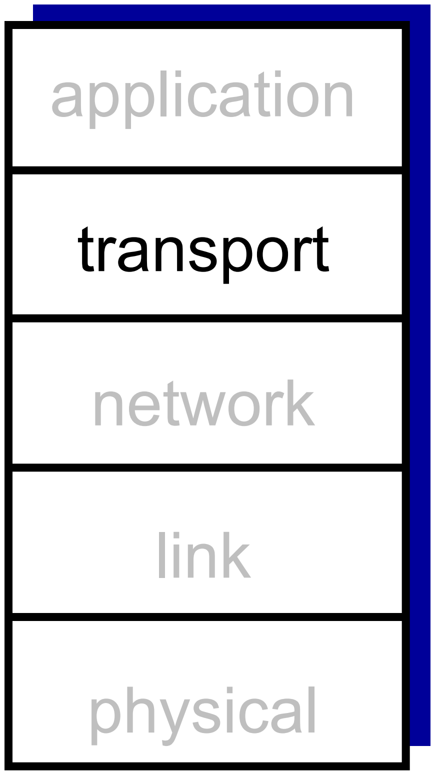
Jorge Granjal
University of Coimbra

UNIVERSIDADE D COIMBRA

T04: outline

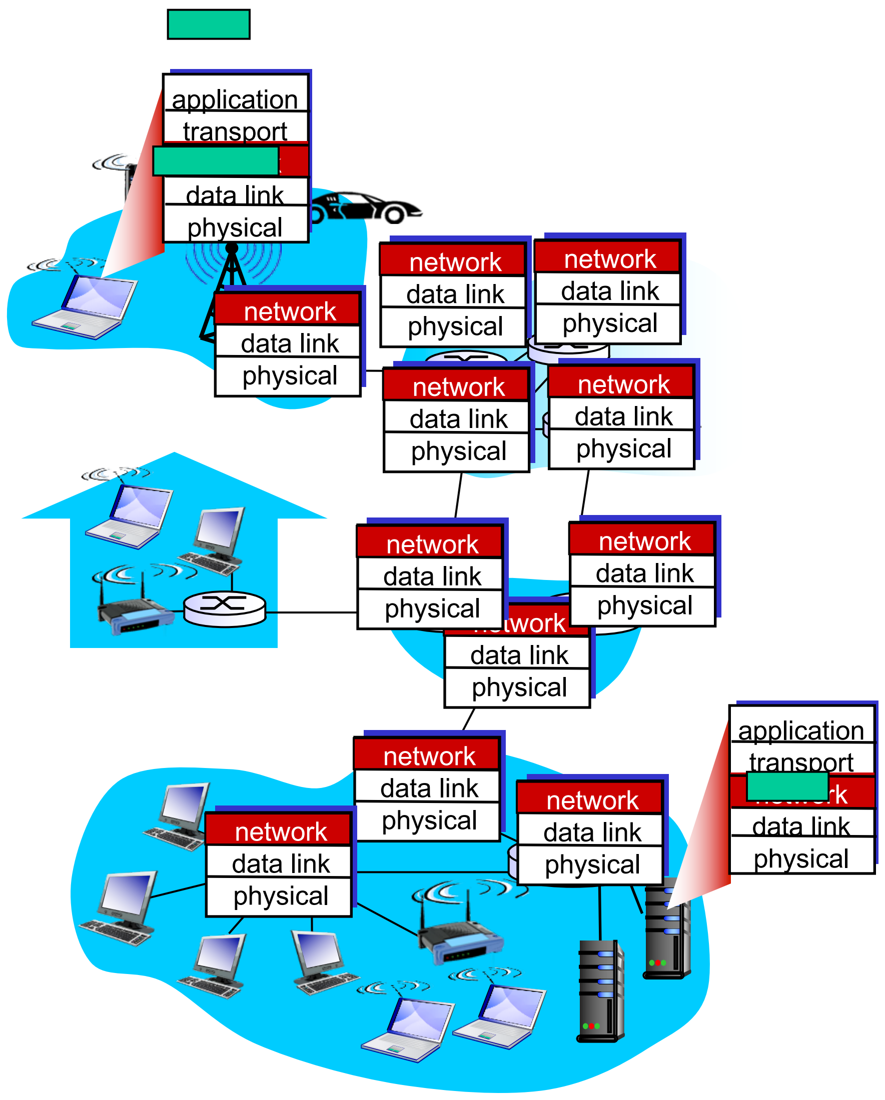
Internet (TCP/IP) protocol stack

- *application:* supporting network applications
  - FTP, SMTP, HTTP, …
- *transport:* process-process data transfer
  - TCP, UDP
- *network:* routing of datagrams from source to destination
  - IP, routing protocols
- *link:* data transfer between neighboring network elements
  - Ethernet, 802.111 (WiFi), PPP
- *physical:* bits "on the wire"



Network layer

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



Two key network-layer functions

network-layer functions:

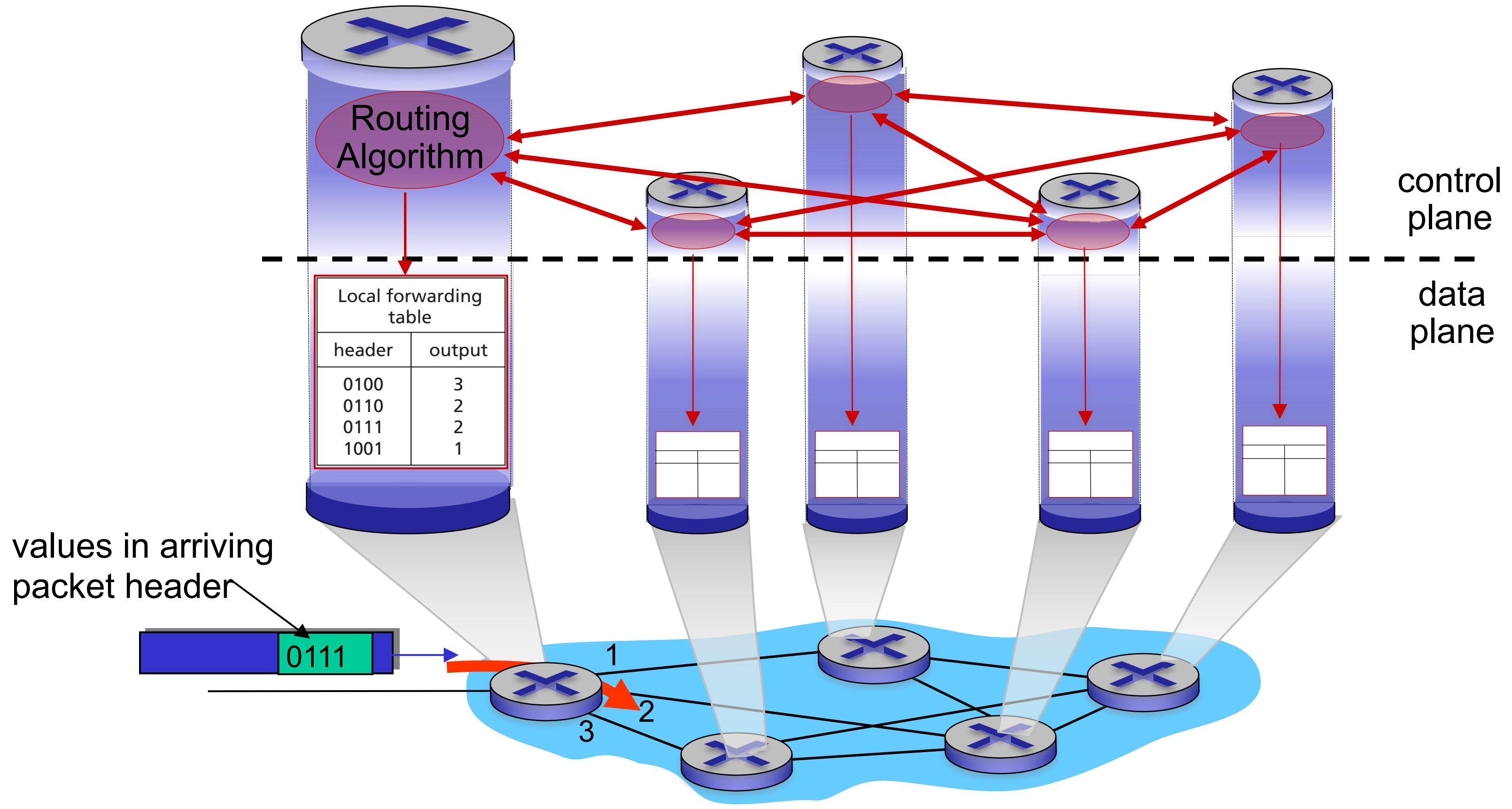
- *forwarding:* move packets from router's input interface to appropriate router output interface
- *routing:* determine route taken by packets from source to destination
  - *routing algorithms*
  - *implemented in various routers*

analogy: taking a trip

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

Routing algorithms and forwarding tables

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



Local forwarding table

header	output
0100	3
0110	2
0111	2
1001	1

Routing algorithms: important aspects

- Algorithms compute the *forwarding tables* that are used to forward packets through the network

- Router receives *routing control messages* which are used to configure its *forwarding table*

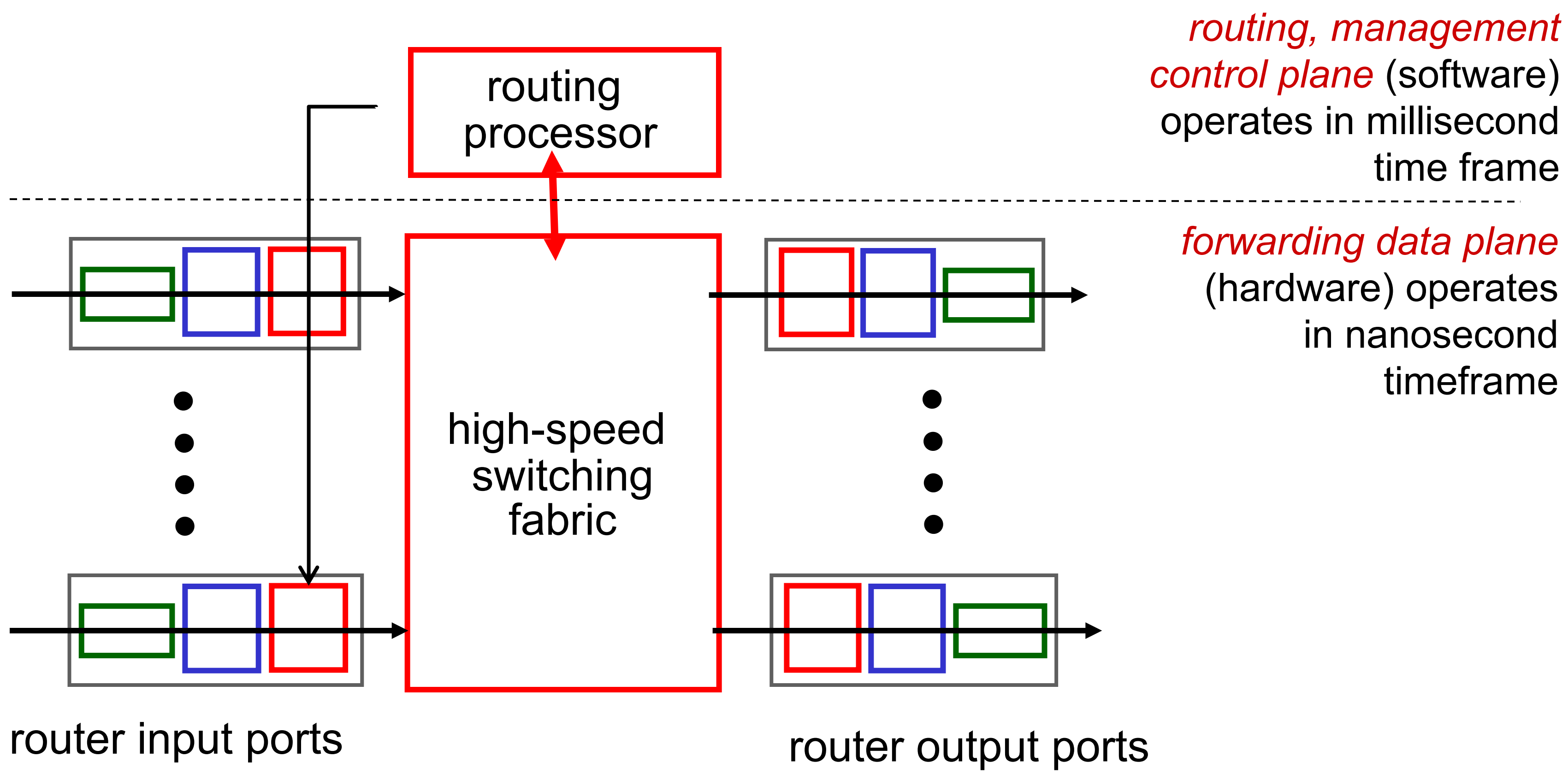
- Routing algorithms may be:
    - Centralized: algorithms executes on central site and routing information is downloaded to each of the routers
    - Decentralized: a piece of the routing algorithm runs in each router

- The term "*packet switch" may refer to either:*
    - Link-layer switch: forwarding decision is based on fields of the link-layer frame (layer 2 devices)
    - Router or layer 3 switch: forwarding decision is based on fields of the network-layer (IP) header

T04: outline

Router architecture overview

- high-level view of generic router architecture:



routing processor
routing, management control plane (software) operates in millisecond time frame
forwarding data plane (hardware) operates in nanosecond timeframe
high-speed switching fabric
router input ports
router output ports

Destination-based forwarding

forwarding table

Destination Address Range	Link Interface

`11001000 00010111 00010000 00000000`
through
`11001000 00010111 00010111 11111111`
	0

`11001000 00010111 00011000 00000000`
through
`11001000 00010111 00011000 11111111`
	1

`11001000 00010111 00011001 00000000`
through
`11001000 00010111 00011111 11111111`
	2

otherwise	3

Longest prefix matching

longest prefix matching

when looking for forwarding table entry for given destination address, use *longest* address prefix (longest subnet mask) that matches destination address.

Destination Address Range	Link interface
11001000 00010111 00010*** *********	0
11001000 00010111 00011000 *********	1
11001000 00010111 00011*** *********	2
otherwise	3

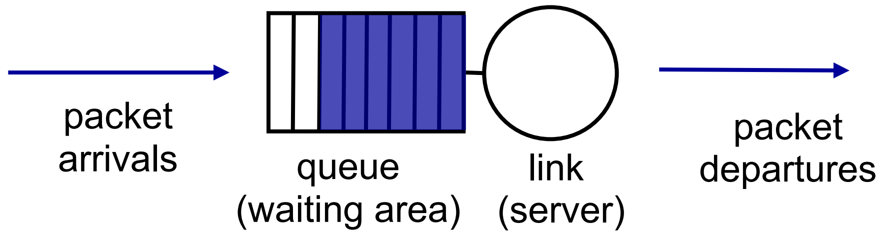
examples:

DA: 11001000 00010111 00010110 10100001 which interface?

DA: 11001000 00010111 00011000 10101010 which interface?

Scheduling mechanisms

- *scheduling:* choose next packet to send on link
- *FIFO (first in first out) scheduling:* send in order of arrival to queue
  - *discard policy:* if packet arrives to full queue: who to discard?
    - *tail drop:* drop arriving packet
    - *priority:* drop/remove on priority basis
    - *random:* drop/remove randomly



T04: outline

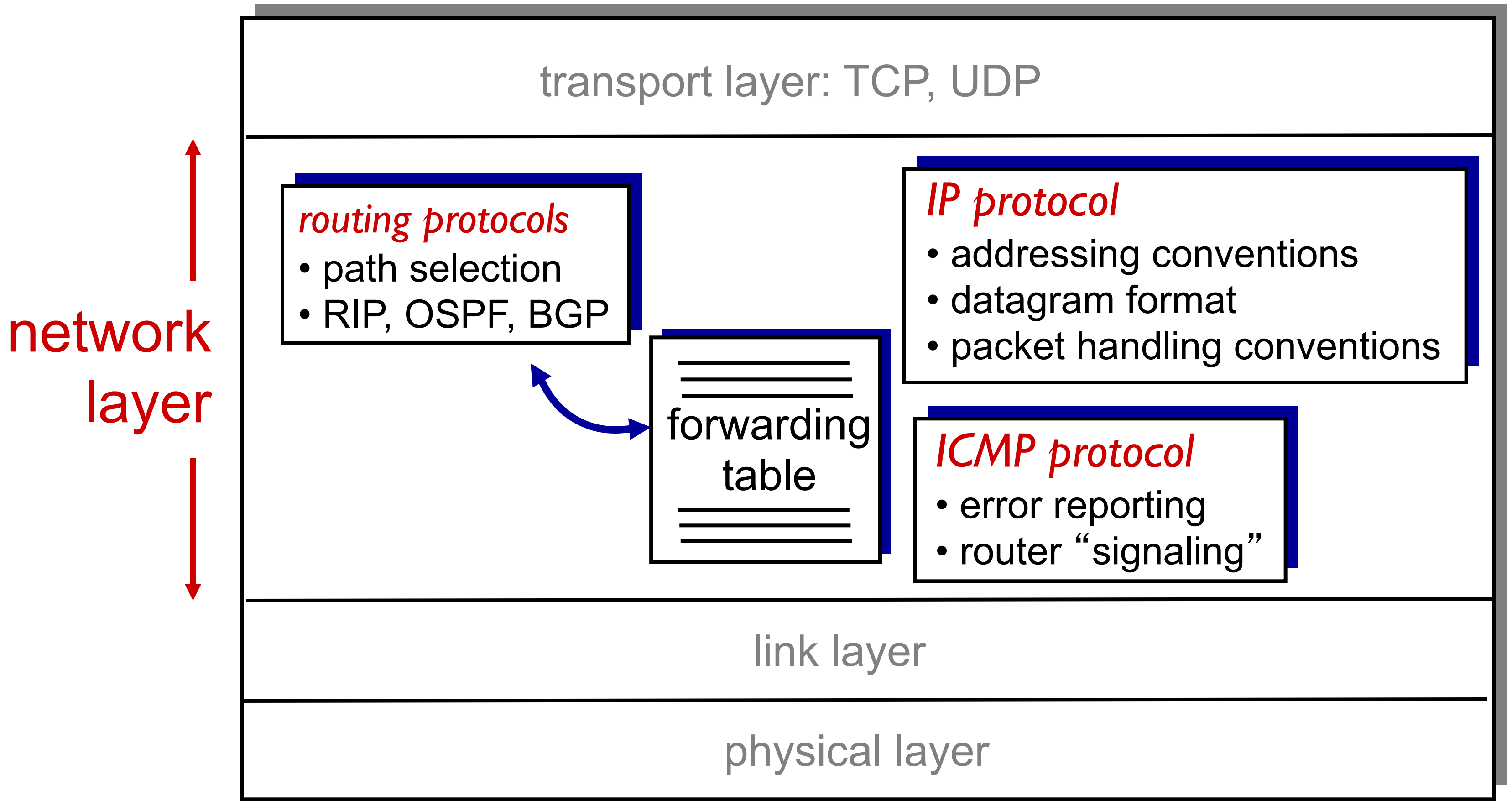
Internet's network layer service

Internet's network layer offers a **best-effort** service:

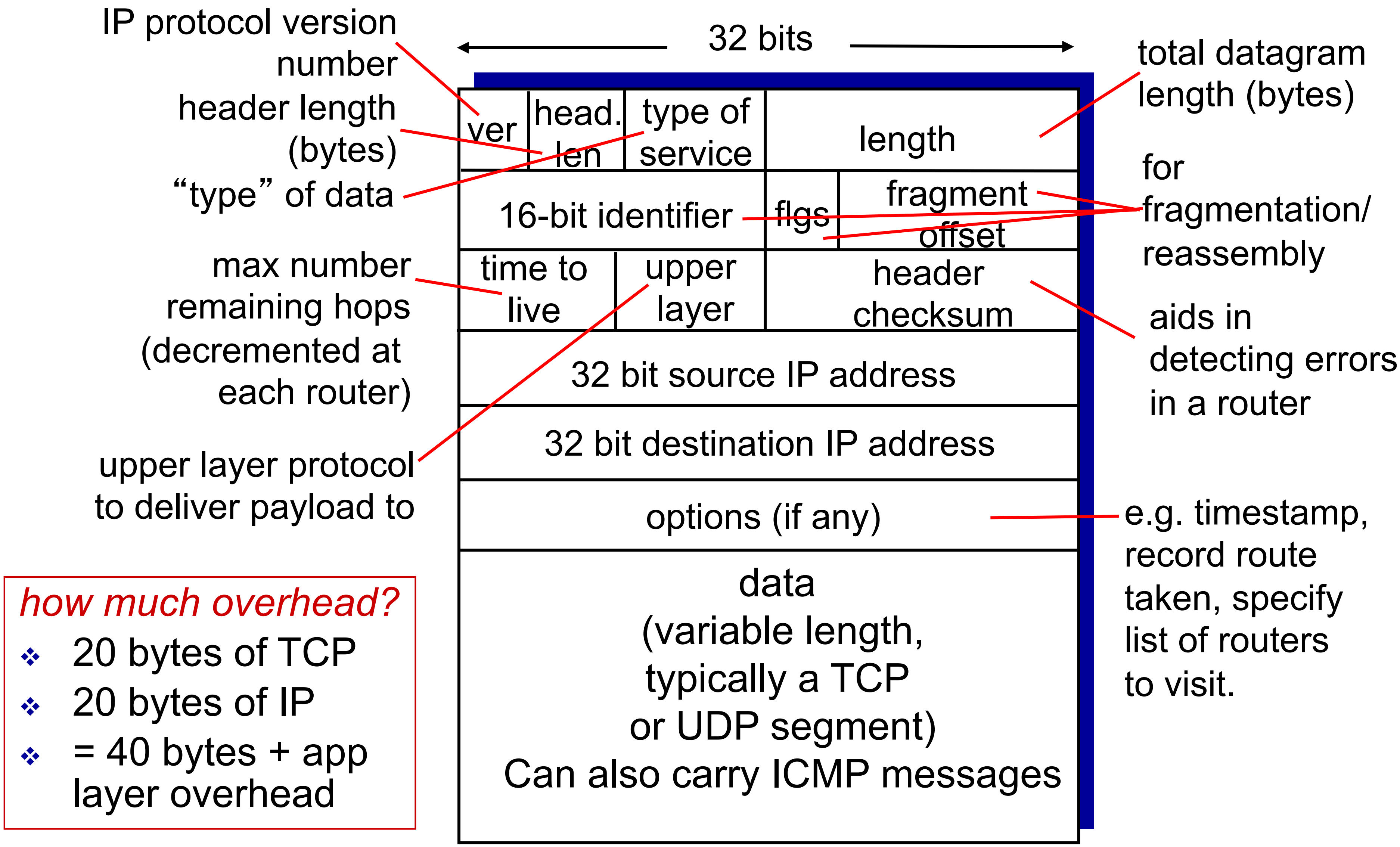
- Service model: best effort
- Bandwidth guarantee: None
- No-Loss guarantee: None
- Ordering: any order possible
- Timing between packets: not maintained
- Congestion indication: None

The Internet network layer

host, router network layer functions:



IP datagram format



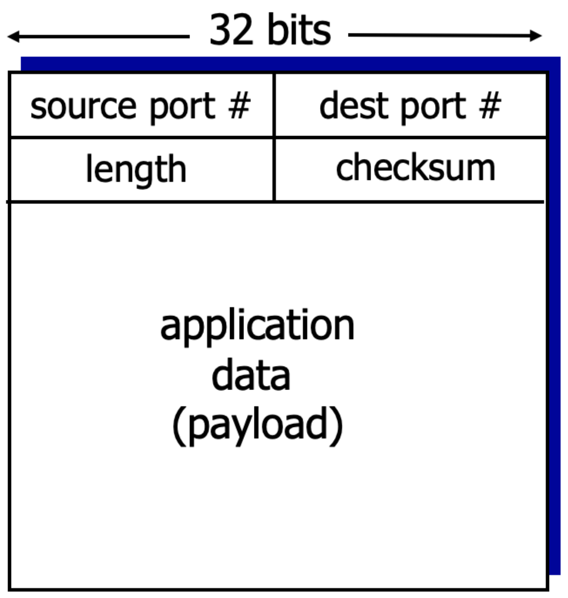
IP protocol version number
header length (bytes)
"type" of data
max number remaining hops (decremented at each router)
upper layer protocol to deliver payload to
32 bits
ver
head. len
type of service
length
16-bit identifier
flgs
fragment offset
time to live
upper layer
header checksum
32 bit source IP address
32 bit destination IP address
options (if any)
data (variable length, typically a TCP or UDP segment) Can also carry ICMP messages
total datagram length (bytes)
for fragmentation/ reassembly
aids in detecting errors in a router
e.g. timestamp, record route taken, specify list of routers to visit.

how much overhead?

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

Revisiting UDP: packet size

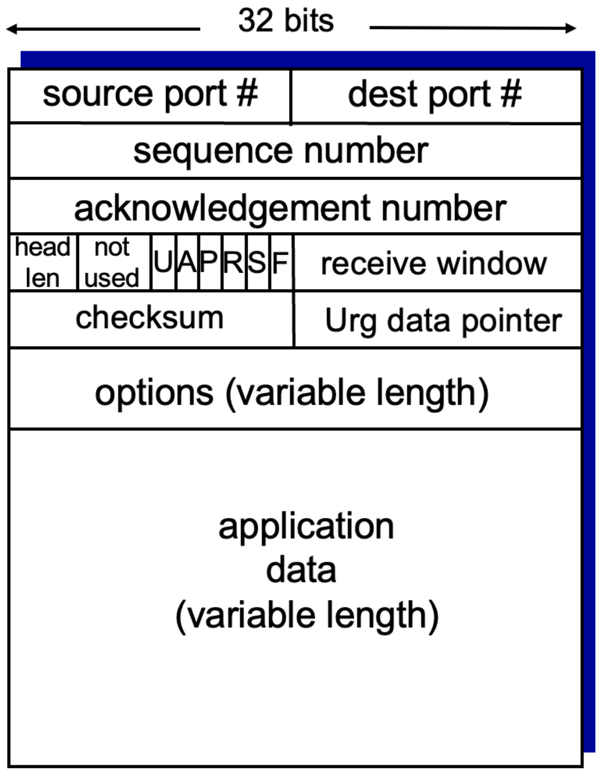
- Minimum size of a UDP packet is 8 bytes.
- The maximum size of a UDP packet is 65,535 bytes (or 2^16 - 1 bytes) minus the size of the IP (20 bytes) and UDP (8 bytes) headers.
- Most networks have a lower maximum transmission unit (MTU) size, which limits the maximum size of a UDP packet that can be transmitted without fragmentation.
- It is generally recommended to keep UDP packets small to minimize the chances of packet loss.



UDP segment format

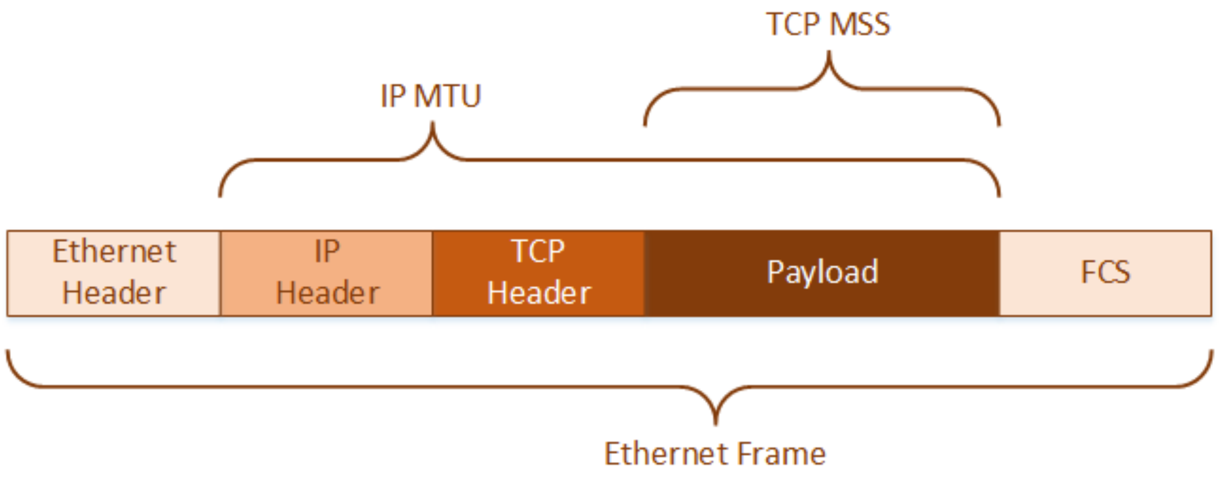
Revisiting TCP: packet size

- The minimum size of a TCP packet is 20 bytes.
- The maximum size of a TCP packet is determined by the Maximum Segment Size (MSS) value negotiated between the two endpoints during the TCP connection setup process.
- The MSS value is typically determined by the maximum size of the data that can be transmitted without fragmentation, based on the MTU of the underlying network.
- In practice, the maximum size of a TCP packet can range from 536 bytes (the minimum MSS value) to around 65,535 bytes (the maximum TCP payload size), depending on the MTU of the network and the negotiated MSS value.



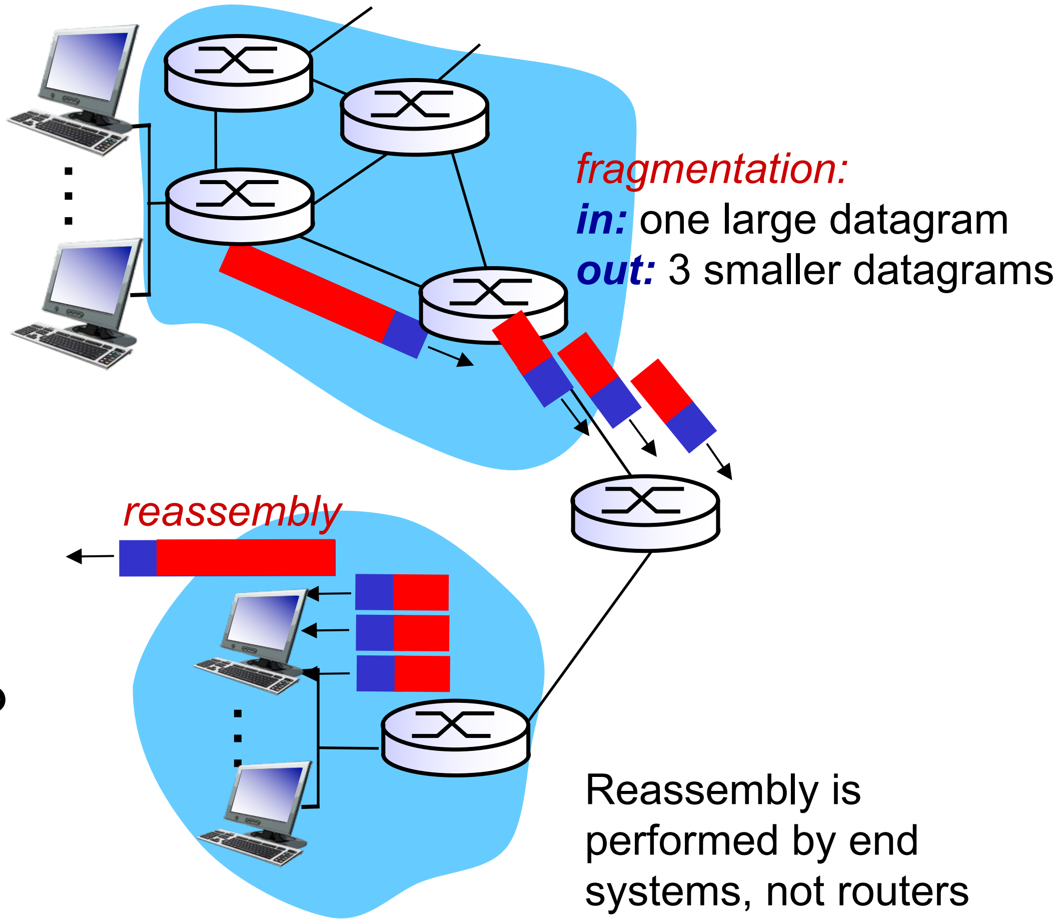
MTU in TCP

- The maximum size of an IP packet is determined by the Maximum Transmission Unit (MTU) of the underlying network.
- The MTU is the largest size of a packet that can be transmitted over a network without being fragmented. In practice, the MTU can vary depending on the type of network and the network equipment being used.
- For example, the MTU for Ethernet networks is typically 1,500 bytes, which means that the maximum size of an IP packet (including header) that can be transmitted over an Ethernet network is 1,500 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 (UDP and TCP are expecting to received complete segments)
  - IP header bits used to identify, order related fragments

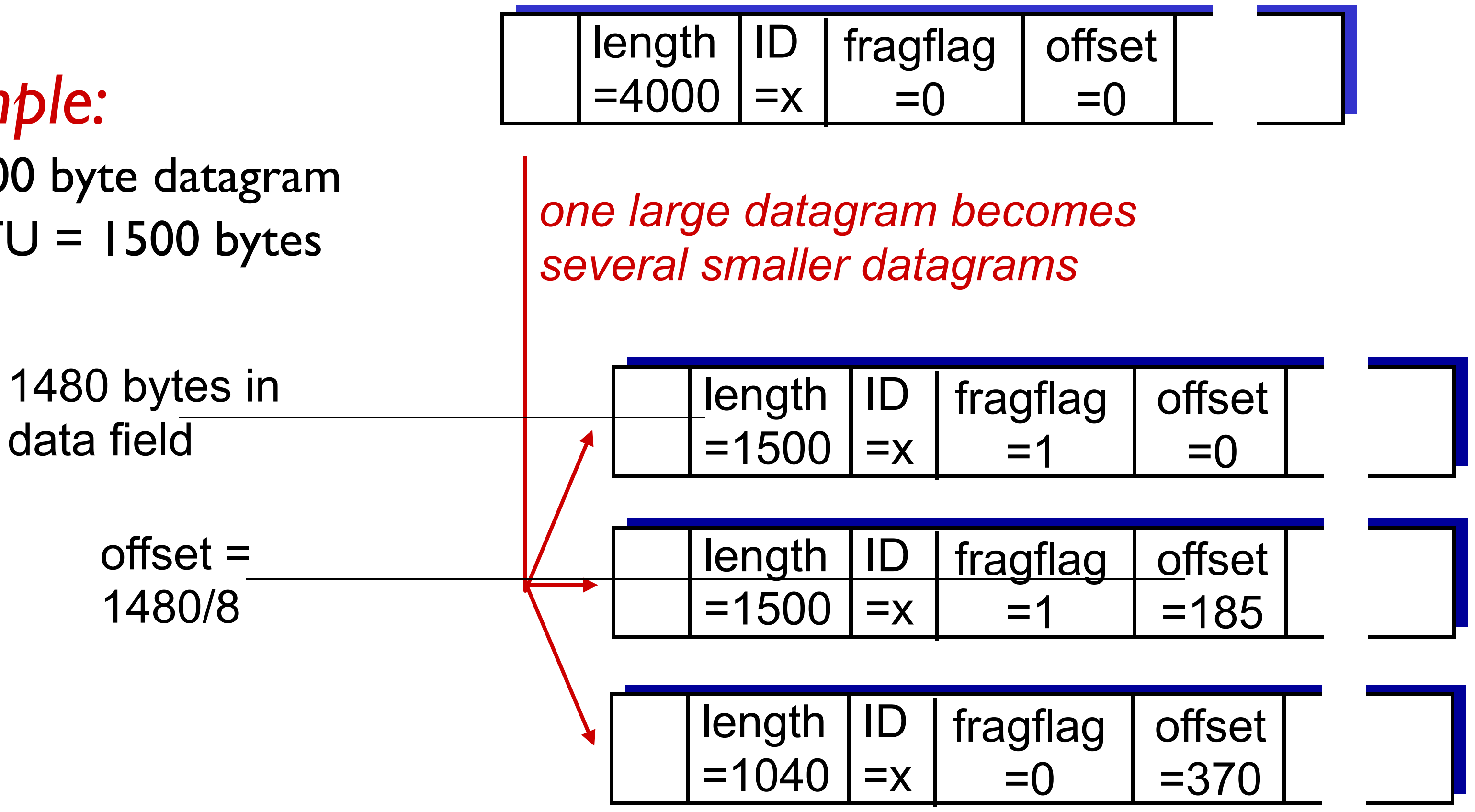


fragmentation:
in: one large datagram
out: 3 smaller datagrams
reassembly
Reassembly is performed by end systems, not routers

IP fragmentation, reassembly

example:

- 4000 byte datagram
- MTU = 1500 bytes



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"

DHCP: Dynamic Host Configuration Protocol

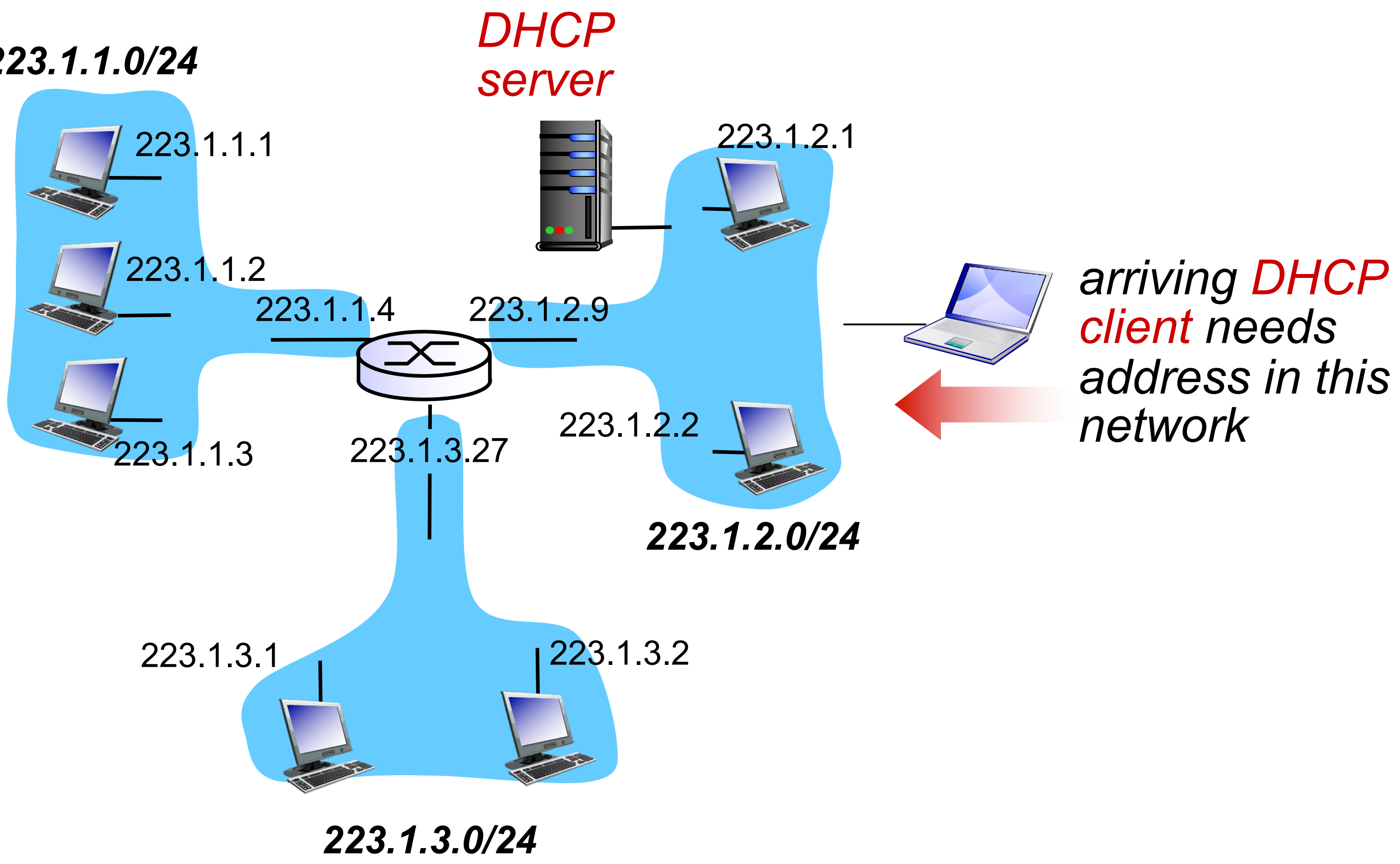
goal: allow host to *dynamically* obtain its IP address from network server when it joins network

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

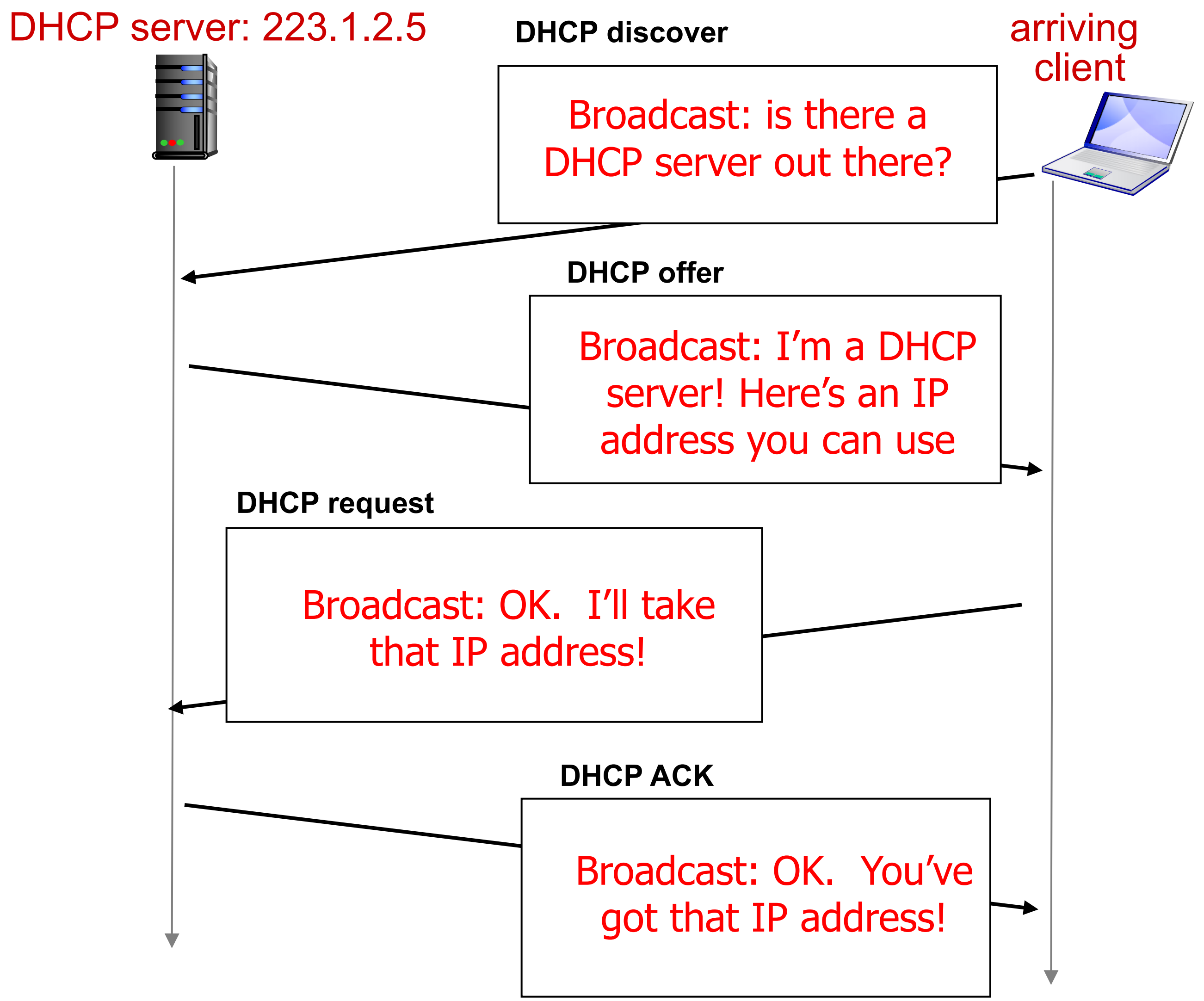
DHCP overview:

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

DHCP client-server scenario



DHCP client-server scenario

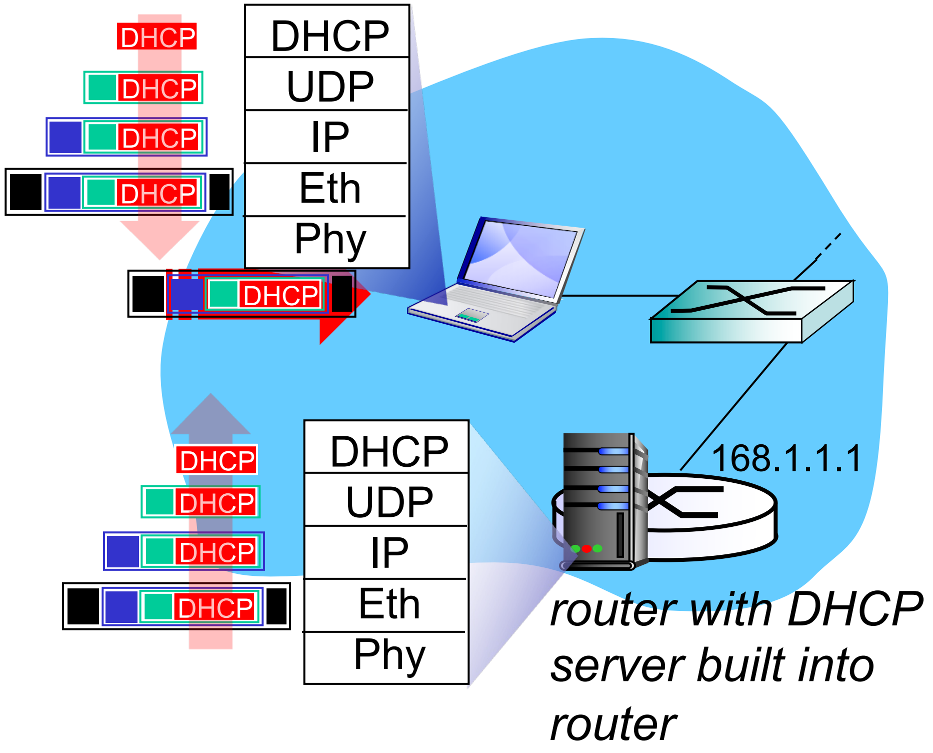

DHCP server: 223.1.2.5
arriving client
DHCP discover
Broadcast: is there a DHCP server out there?
DHCP offer
Broadcast: I'm a DHCP server! Here's an IP address you can use
DHCP request
Broadcast: OK. I'll take that IP address!
DHCP ACK
Broadcast: OK. You've got that IP address!

DHCP: more than IP addresses

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

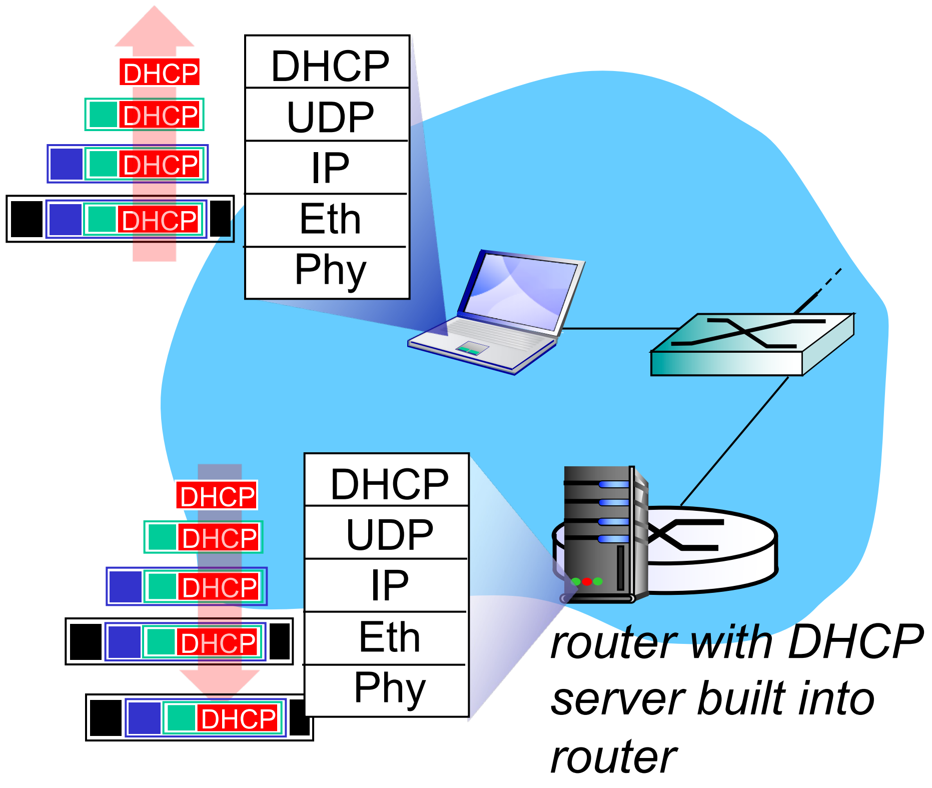
- address of first-hop router for client
- name and IP address of DNS sever
- network mask (indicating network versus host portion of address)

DHCP: example



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

DHCP: example



router with DHCP server built into router

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

IP addresses: how to get one?

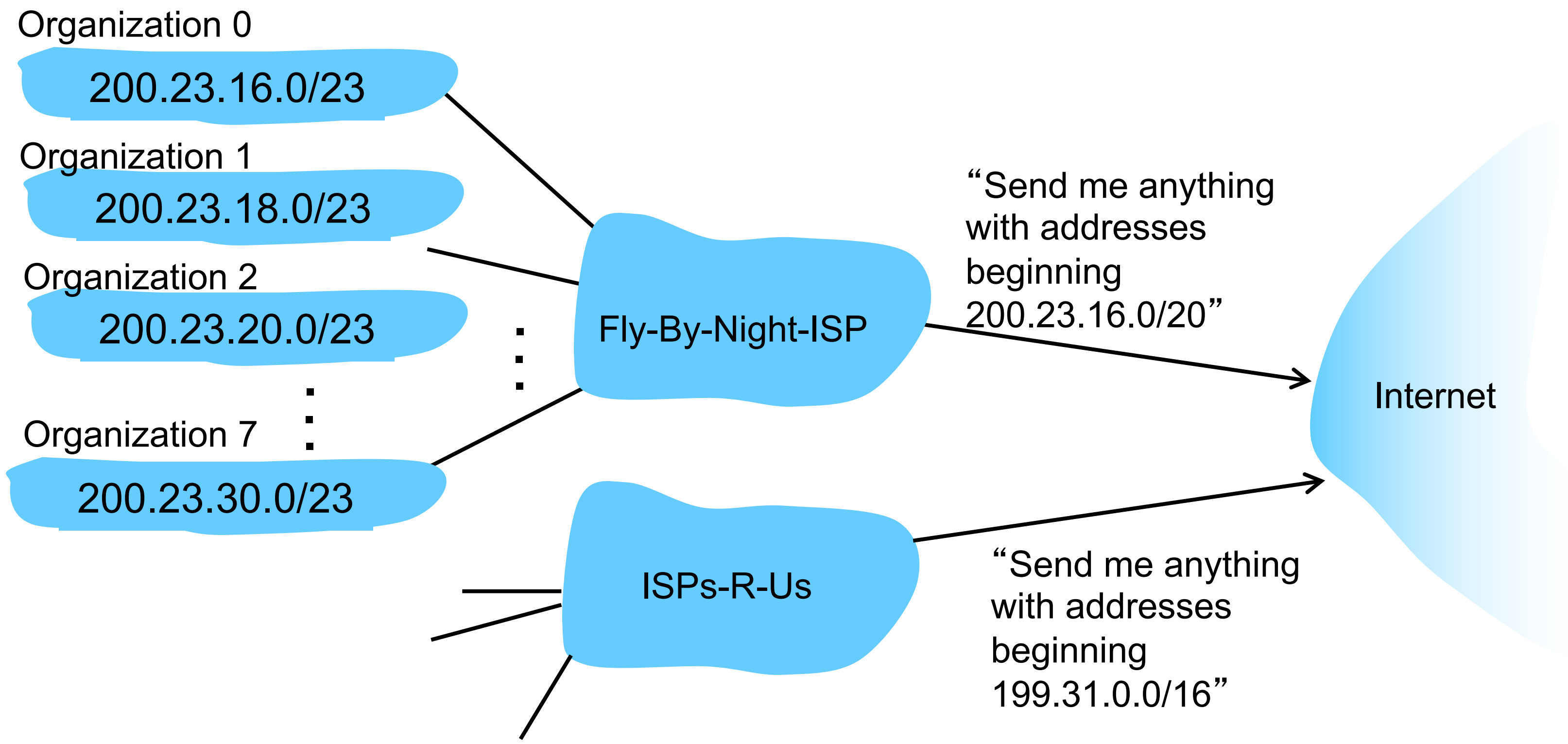
Q: how does *network* get subnet part of IP addr?

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

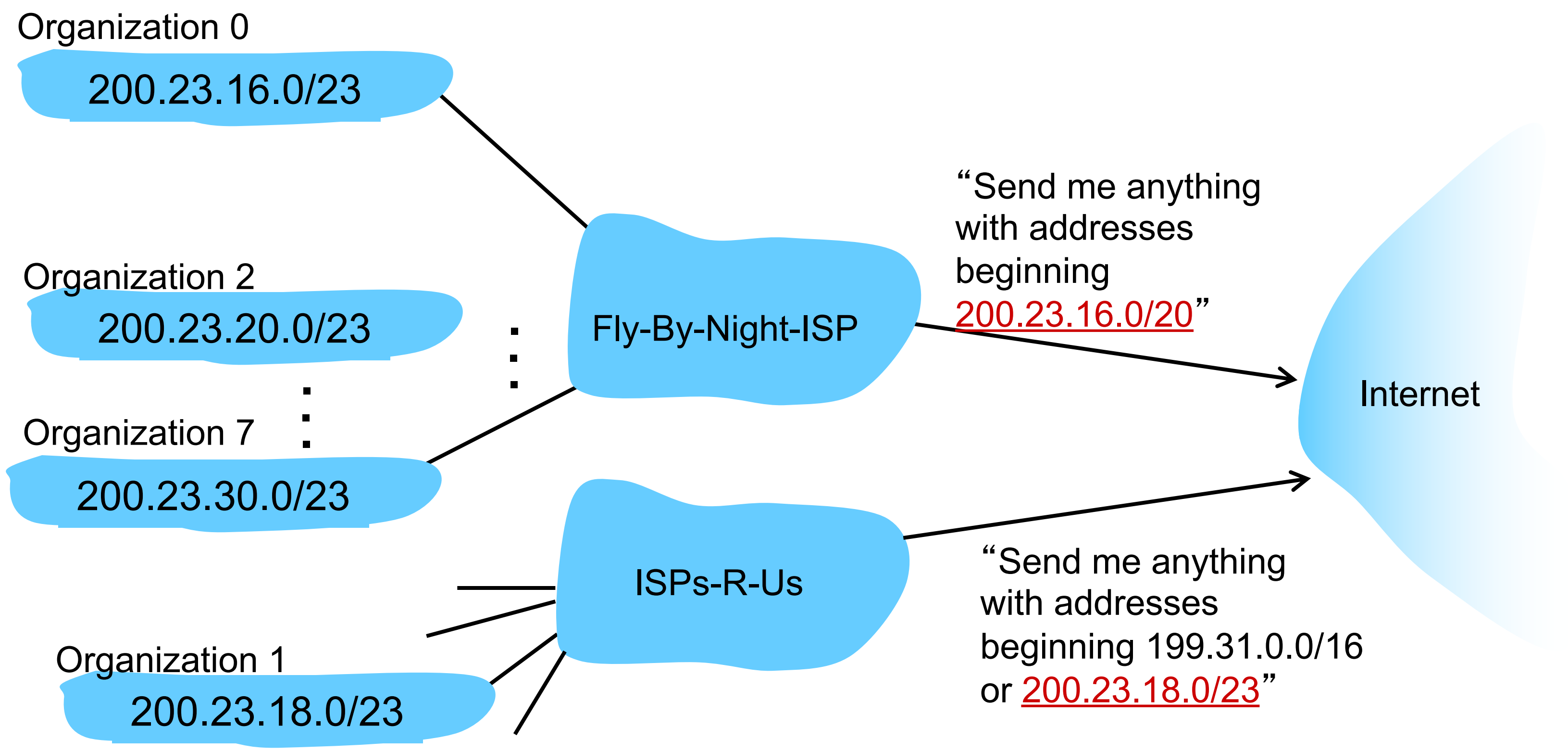
Hierarchical addressing: route aggregation

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...

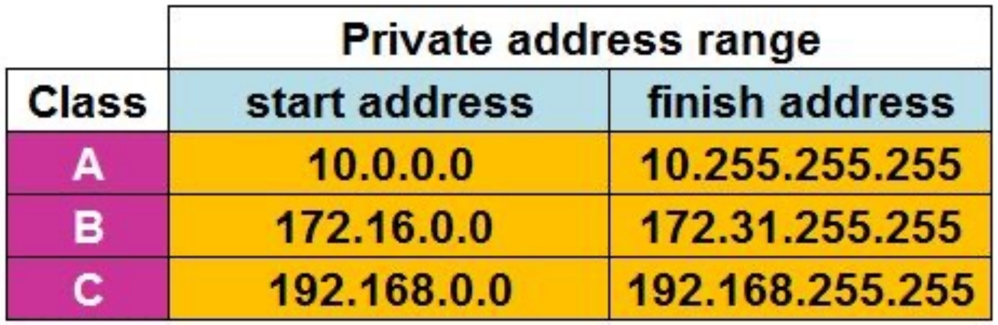
Q: how does an ISP get block of addresses?

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

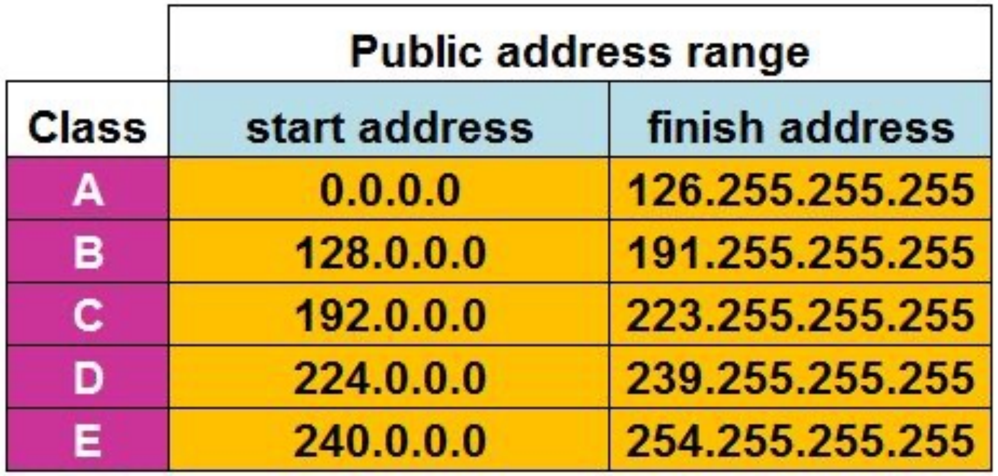
- allocates addresses
- manages DNS
- assigns domain names, resolves disputes

ICANN

NAT: private IP address ranges



	Private address range	
Class	start address	finish address
A	10.0.0.0	10.255.255.255
B	172.16.0.0	172.31.255.255
C	192.168.0.0	192.168.255.255



	Public address range	
Class	start address	finish address
A	0.0.0.0	126.255.255.255
B	128.0.0.0	191.255.255.255
C	192.0.0.0	223.255.255.255
D	224.0.0.0	239.255.255.255
E	240.0.0.0	254.255.255.255

NAT: network address translation

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

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

NAT: network address translation

implementation: NAT router must:

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

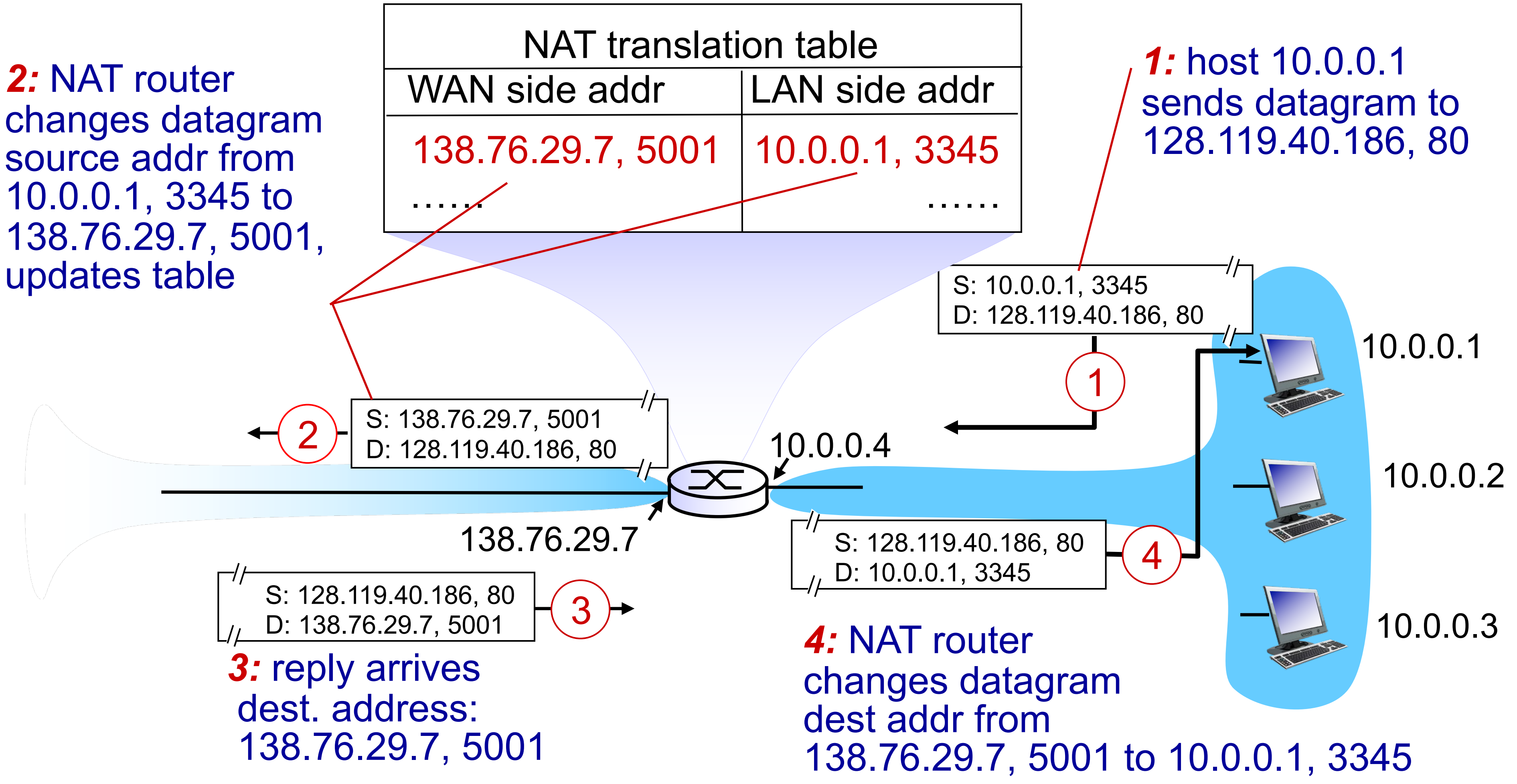
  . . . remote clients/servers will respond using (NAT IP address, new port #) as destination addr

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

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

NAT: SNAT

NAT translation table	
WAN side addr	LAN side addr
138.76.29.7, 5001	10.0.0.1, 3345
……	……



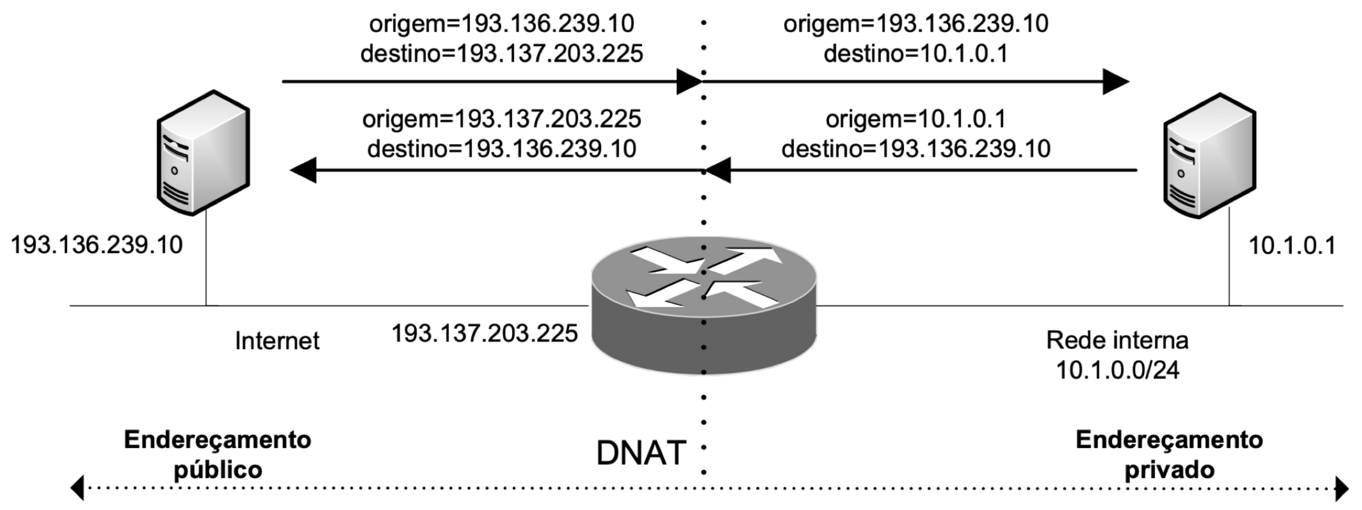
2: NAT router changes datagram source addr from 10.0.0.1, 3345 to 138.76.29.7, 5001, updates table
1: host 10.0.0.1 sends datagram to 128.119.40.186, 80
S: 10.0.0.1, 3345
D: 128.119.40.186, 80
1
10.0.0.1
10.0.0.2
10.0.0.3
2
S: 138.76.29.7, 5001
D: 128.119.40.186, 80
10.0.0.4
138.76.29.7
S: 128.119.40.186, 80
D: 10.0.0.1, 3345
4
S: 128.119.40.186, 80
D: 138.76.29.7, 5001
3
3: reply arrives dest. address: 138.76.29.7, 5001
4: NAT router changes datagram dest addr from 138.76.29.7, 5001 to 10.0.0.1, 3345

NAT: SNAT

- Also known as SNAT (Source NAT)
- 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
  - address shortage should be solved by IPv6
  - violates end-to-end argument
    - NAT possibility must be taken into account by app designers, e.g., P2P applications
  - NAT traversal: what if client wants to connect to server behind NAT?

NAT Traversal: DNAT

- What if client wants to connect to server behind NAT?
- DNAT (Destination NAT): destination IP and port # changed for packets from the external to the internal network



T04: outline

Network-layer functions

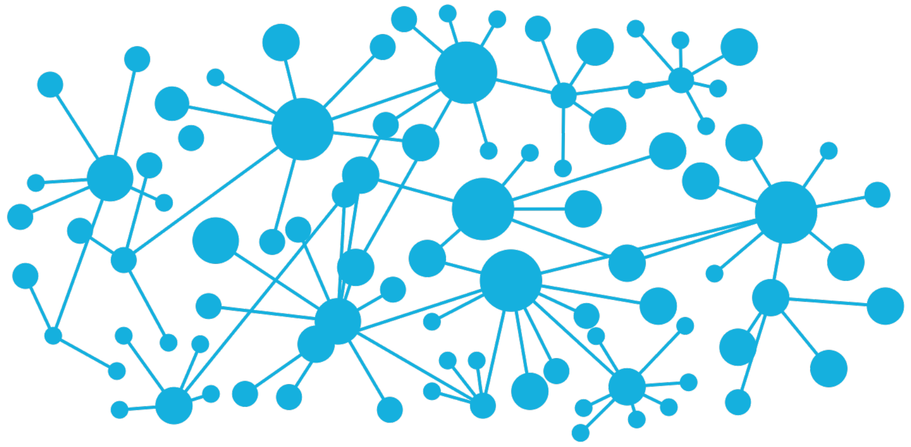
Recall: two network-layer functions:

- *forwarding:* move packets from router's input to appropriate router output

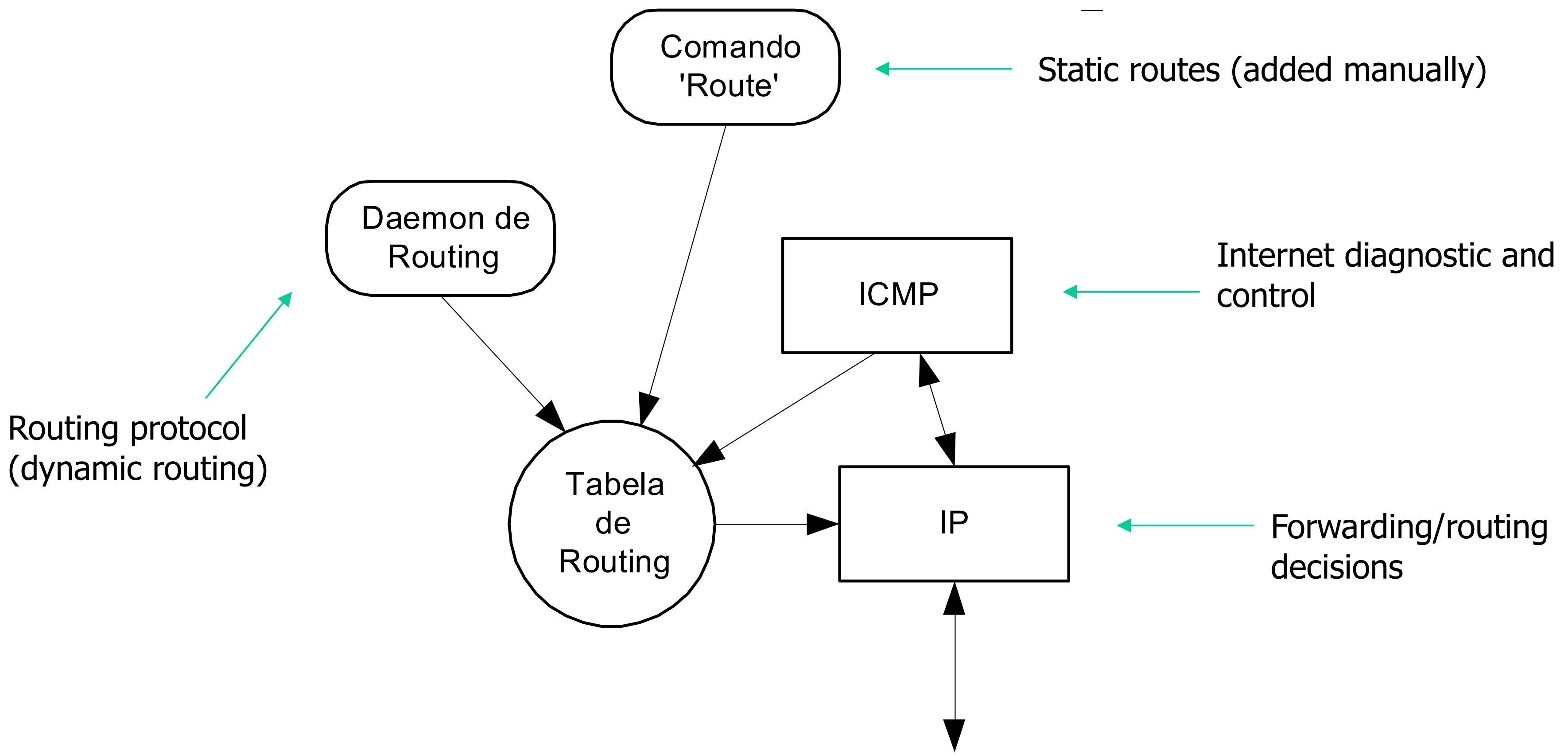
data plane

- *routing:* determine route taken by packets from source to destination

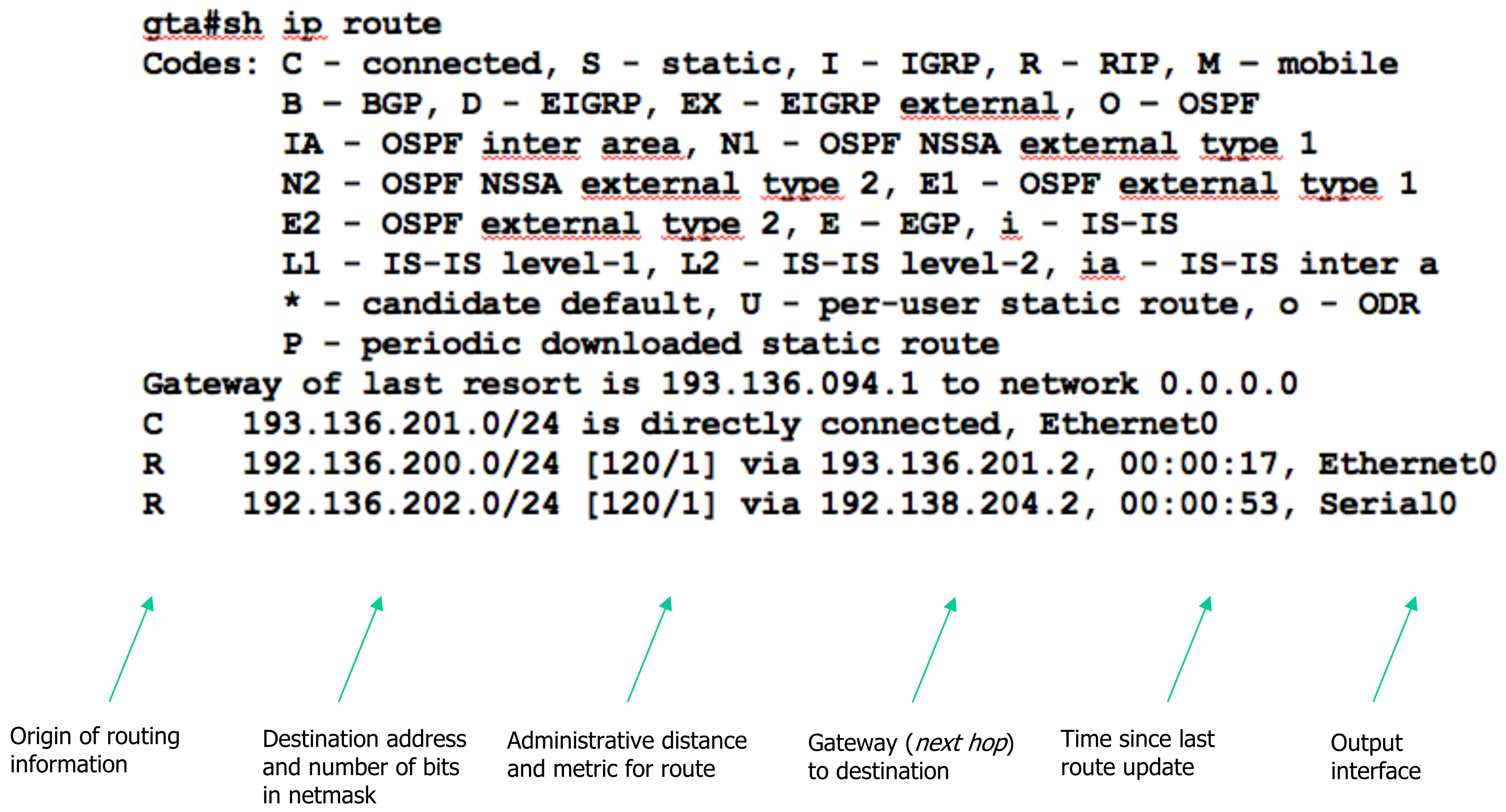
control plane



IP Routing


Comando 'Route'
Static routes (added manually)
Daemon de Routing
ICMP
Internet diagnostic and control
Routing protocol (dynamic routing)
Tabela de Routing
IP
Forwarding/routing decisions

Routing table (Cisco)



```
gta#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile
       B - BGP, D - EIGRP, EX - EIGRP external, O - OSPF
       IA - OSPF inter area, N1 - OSPF NSSA external type 1
       N2 - OSPF NSSA external type 2, E1 - OSPF external type 1
       E2 - OSPF external type 2, E - EGP, i - IS-IS
       L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter a
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route
Gateway of last resort is 193.136.094.1 to network 0.0.0.0
C    193.136.201.0/24 is directly connected, Ethernet0
R    192.136.200.0/24 [120/1] via 193.136.201.2, 00:00:17, Ethernet0
R    192.136.202.0/24 [120/1] via 192.138.204.2, 00:00:53, Serial0
```

Origin of routing information
Destination address and number of bits in netmask
Administrative distance and metric for route
Gateway (*next hop*) to destination
Time since last route update
Output interface

Routing algorithm classification

Q: global or decentralized information?

global:

- all routers have **complete topology**, link cost info
- “link state” algorithms

decentralized:

- router knows **only physically-connected neighbors** and link costs to neighbors
- iterative process of computation, exchange of info with neighbors
- “distance vector” algorithms

Q: static or dynamic?

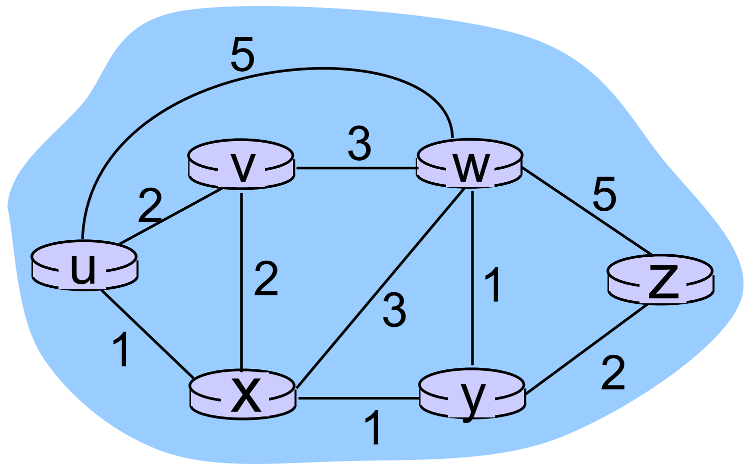
static:

- routes change slowly over time

dynamic:

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

Graph abstraction of the network



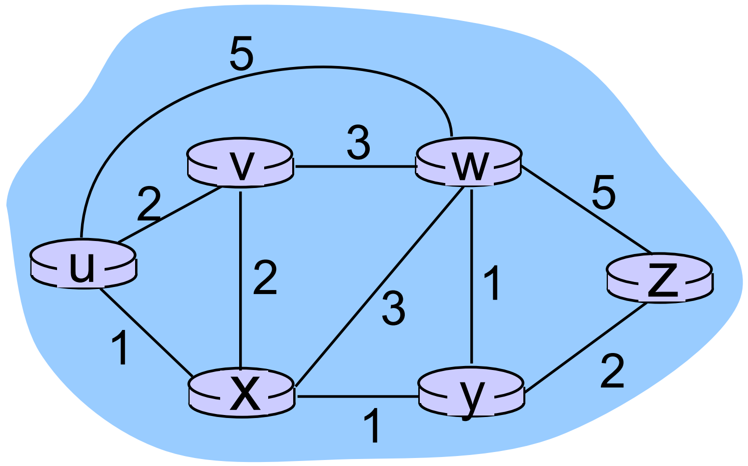
graph: G = (N,E)

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

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

aside: graph abstraction is useful in other network contexts, e.g., P2P, where *N* is set of peers and *E* is set of TCP connections

Graph 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, etc.

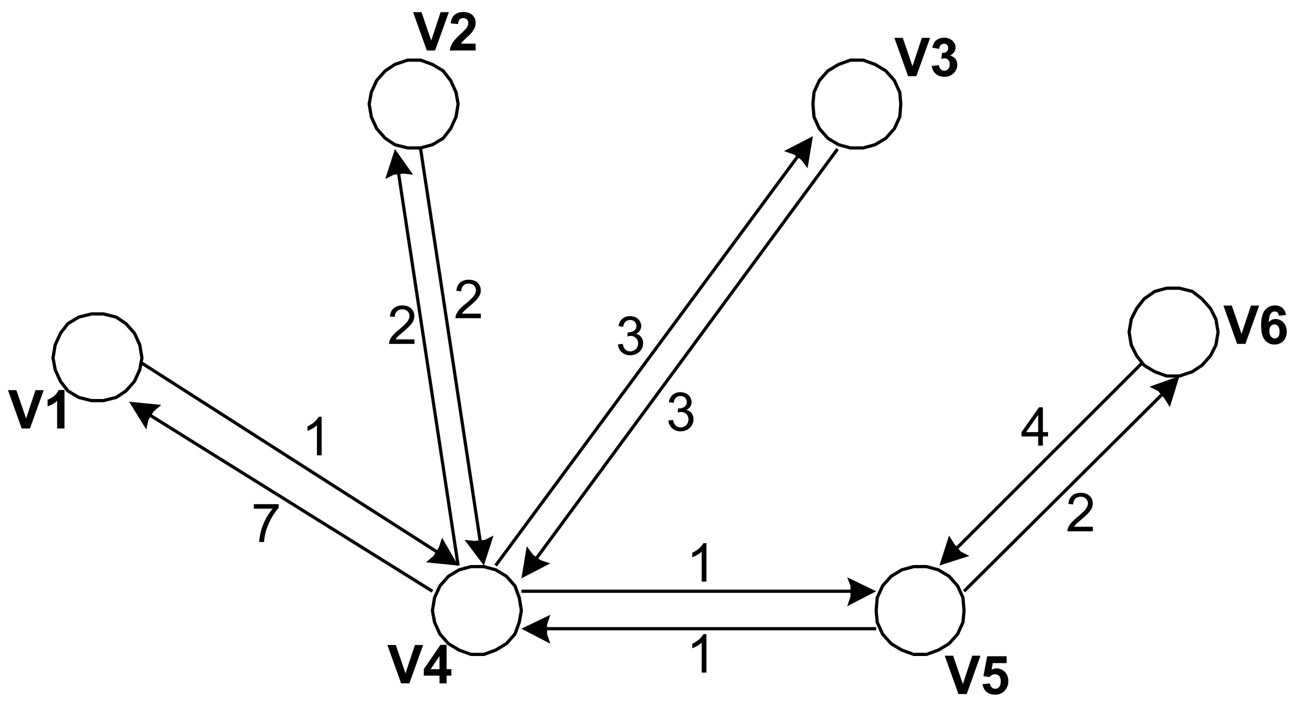
- costs can also be different for the two directions of communications

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

key question: what is the least-cost path between u and z ?
routing algorithm: algorithm that <u>finds that least cost path</u>

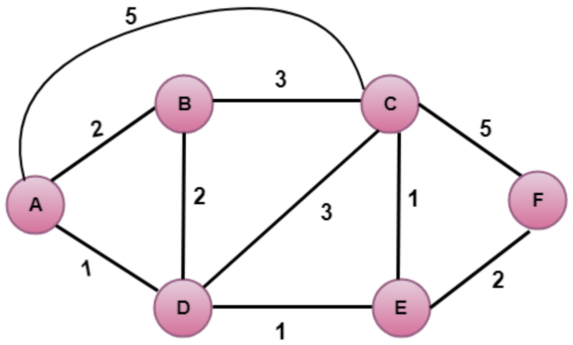
Spanning tree (of a graph)

A **spanning tree** is a subset of a graph which has all the vertices covered and with minimum possible number of edges



Dijkstra algorithm

- Created by Edsger Dijkstra (1959)
- Algorithm to find shortest path from a given vertex to all other known vertices (routers)
- The resulting set of paths forms a spanning tree
- For a graph with V vertices requires V-1 iterations
- In each iteration the number of operations is proportional to V
- Iterative: of $O(V^2)$ complexity for a graph with V vertices
- Used in link state routing protocols: requires information about *topology* and *link costs*



Dijkstra algorithm

Given:

- N: set of vertices in graph
- S: origin vertex
- T: set of vertices (already) added by the algorithm
- w(i,j): cost of path from i to j
- L(n): cost of the path with lower cost from s to n, already added by the algorithm

The algorithm uses 3 steps

- Step 1: initialization
- Step 2 and 3 are repeated until T = N (until spanning tree starting at origin vertex is formed)

Dijkstra algorithm

Step 1 – Initialization

1.1 T = {s}

1.2 L(n) = w(s,n)

L(s) = **0**

L(i) = **w(s,i)** (to neighbouring nodes of *s*)

L(n) = **infinite** (to other nodes which are not neighbors of *s*)

Dijkstra algorithm

Step 2 – chooses next vertex to add

2.1 Find x not belonging to T for which L(x) = min L(j), for all j not in T

2.2 Add x to T

2.3 Add link to x to T

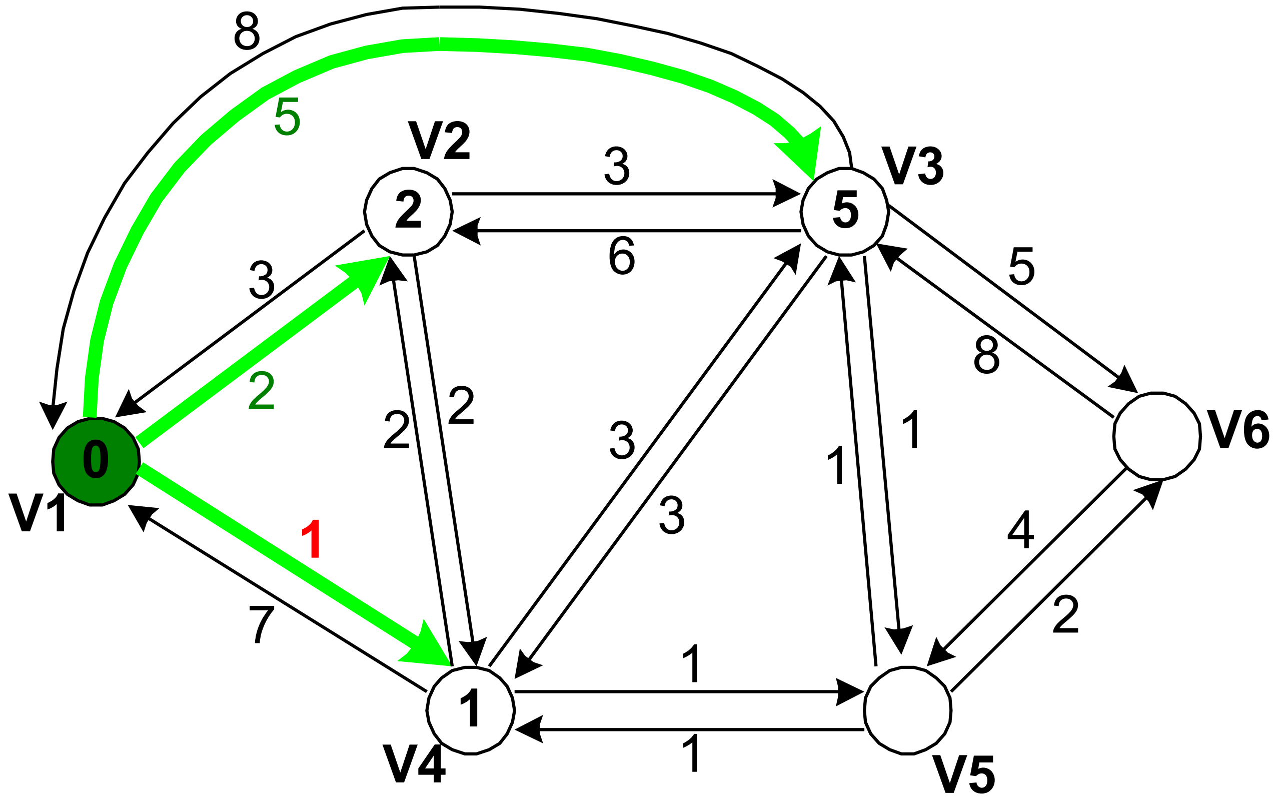
Step 3 – update least cost paths

L(n) = min [L(n), L(x)+w(x,n)],
for all n not in T

Dijkstra algorithm (example)

Step 1:Initialization

T = {V1}

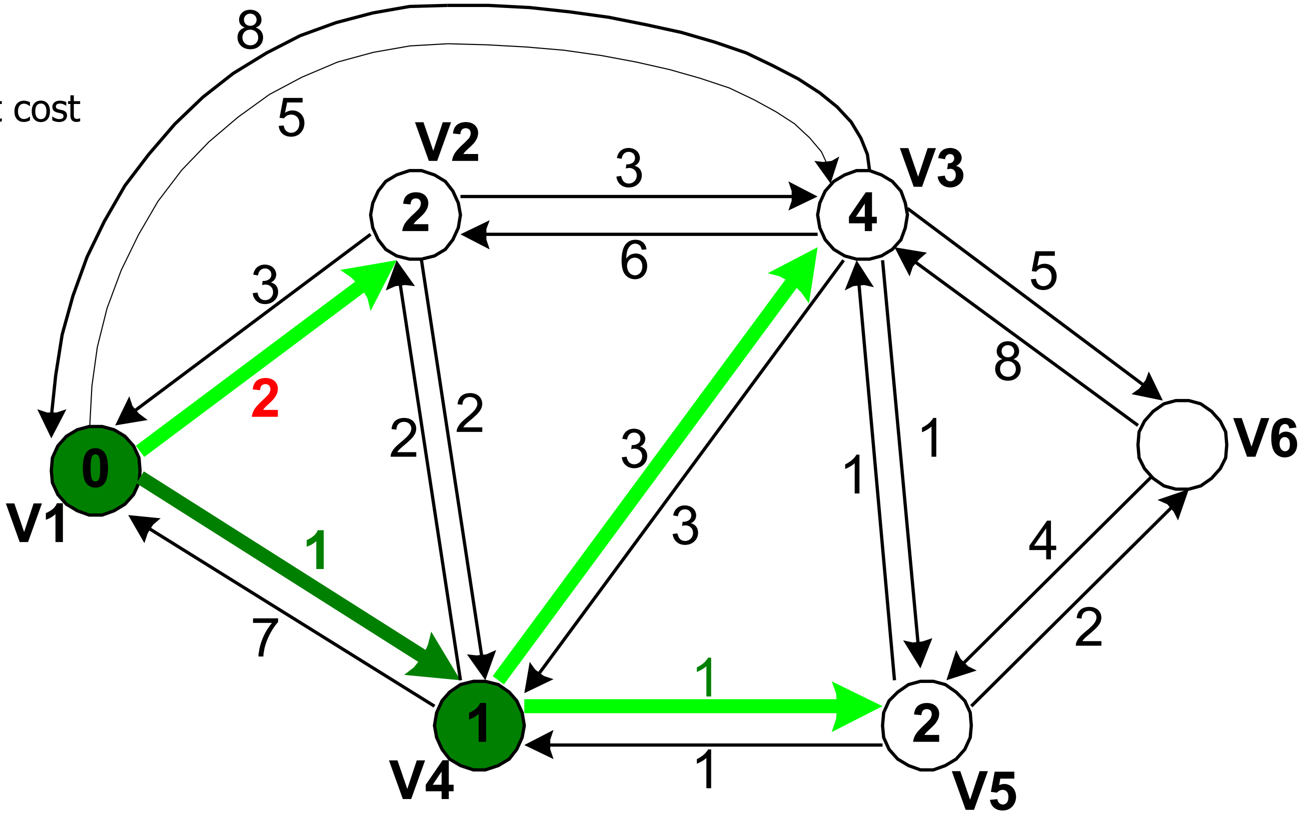


Dijkstra algorithm (example)

Step 2: add V4 to T

T = {V1,V4}

Step 3: update least cost paths

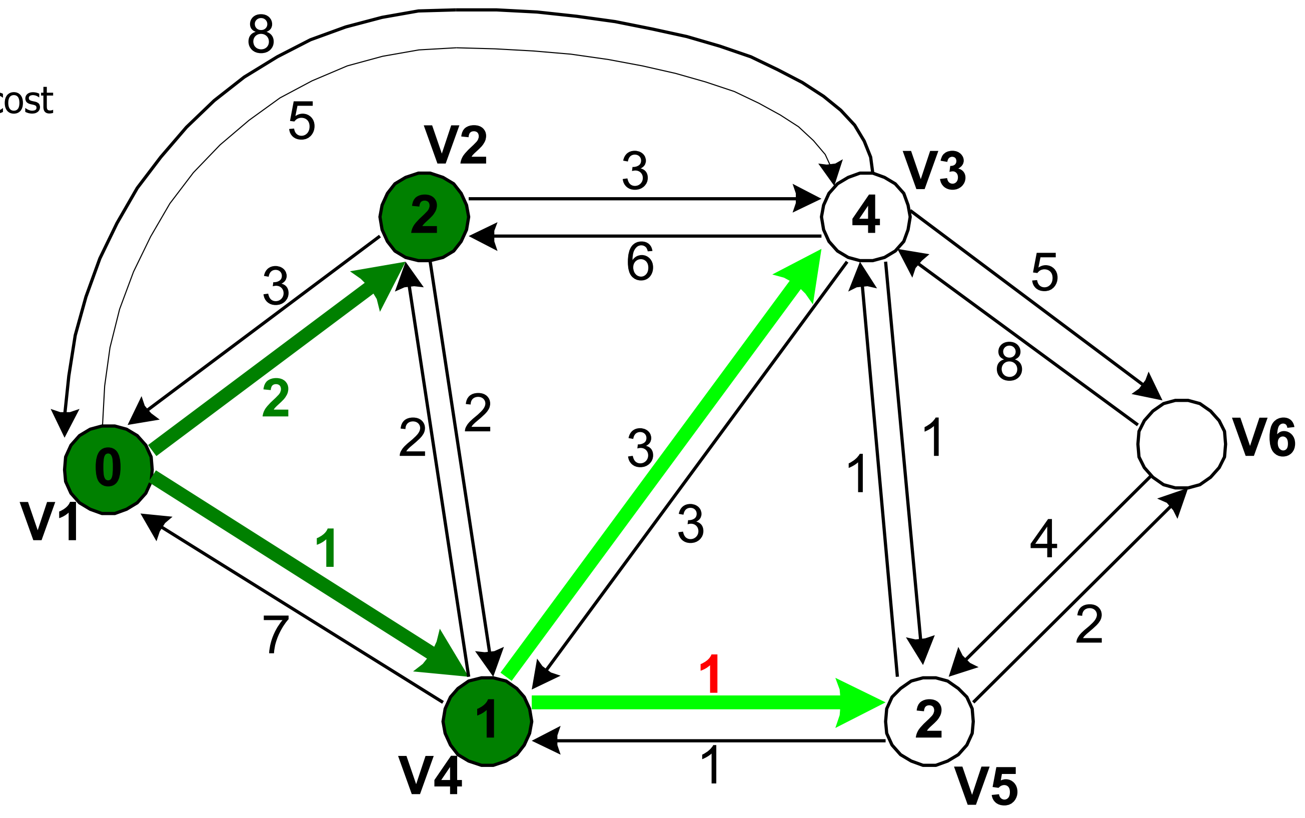


Dijkstra algorithm (example)

Step 2: add V2 to T

T = {V1,V4,V2}

Step 3: update least cost paths

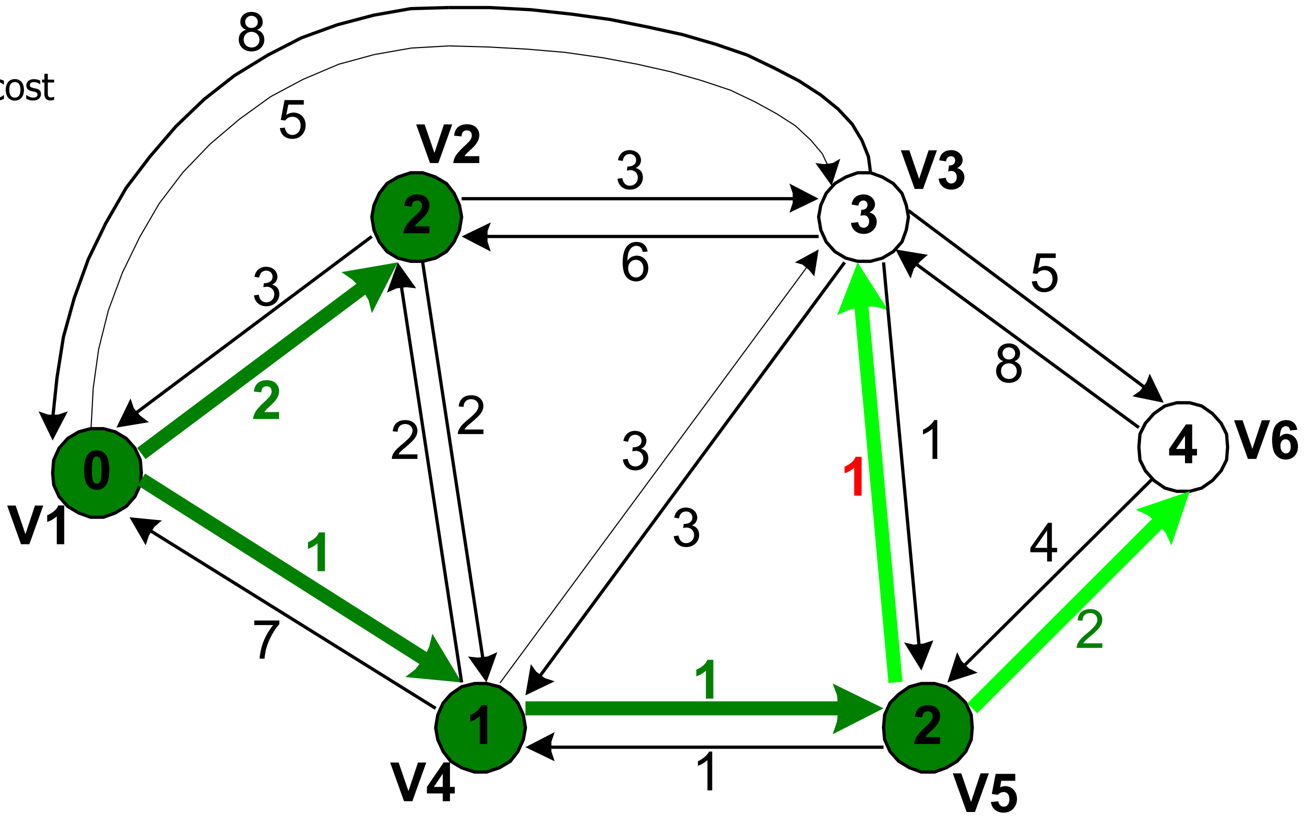


Dijkstra algorithm (example)

Step 2: add V5 to T

T = {V1,V4,V2,V5}

Step 3: update least cost paths

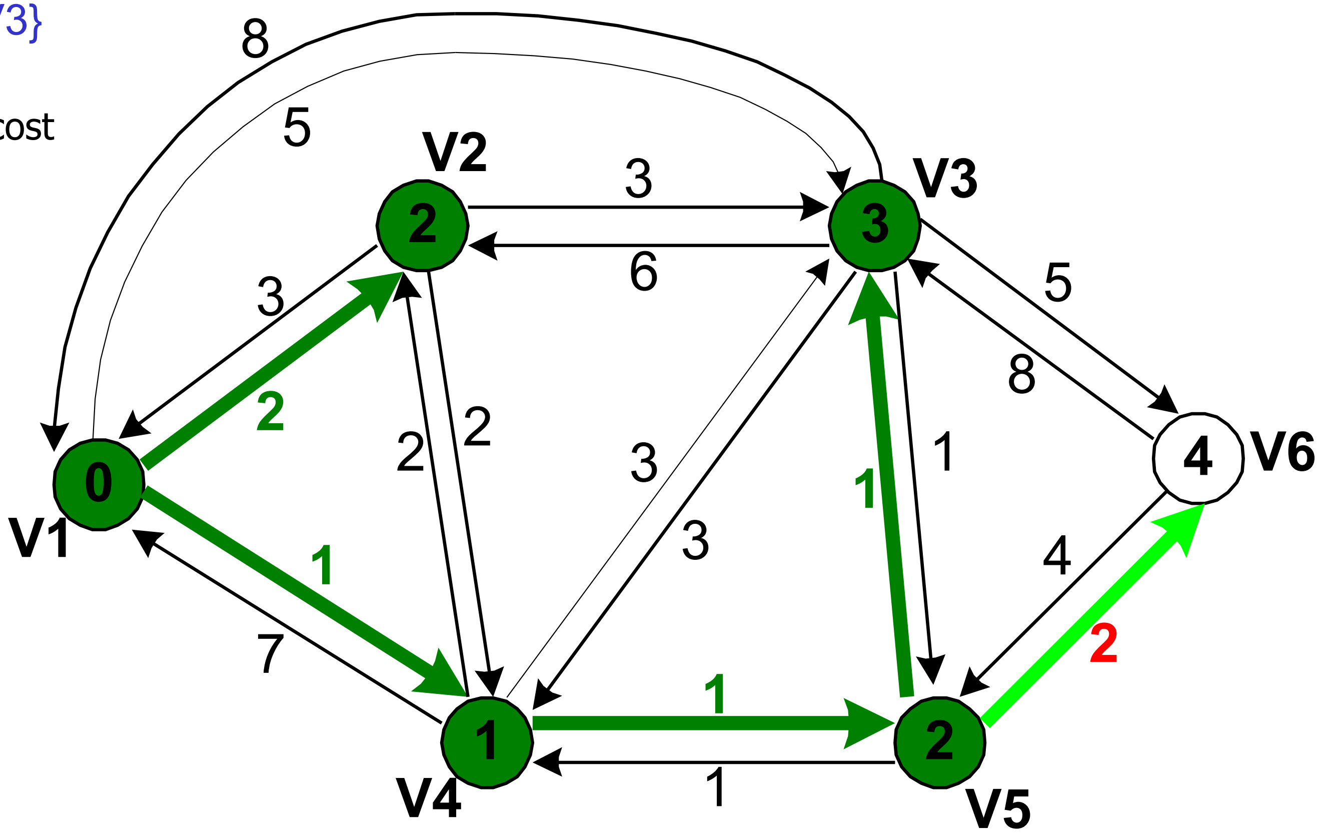


Dijkstra algorithm (example)

Step 2: add V3 to T

T = {V1,V4,V2,V5,V3}

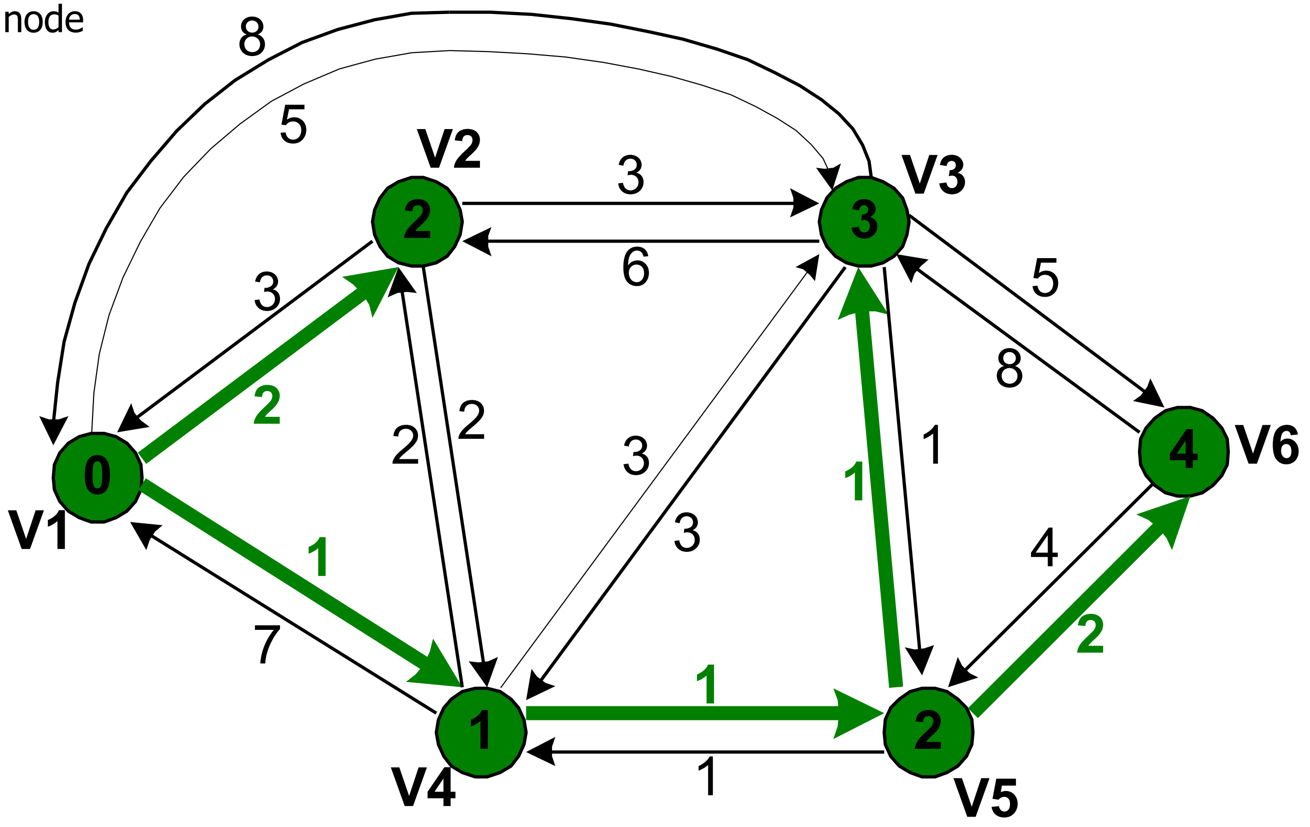
Step 3: update least cost paths



Dijkstra algorithm (example)

T = {V1,V4,V2,V5,V3,V6} = N

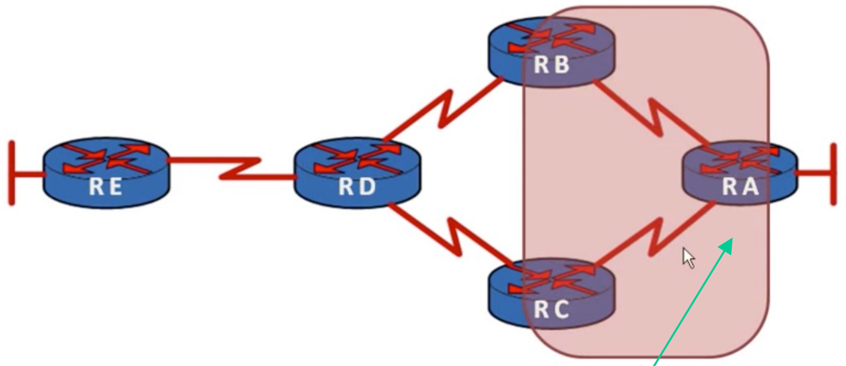
Spanning tree from node V1 is formed



Distance-Vector Protocols

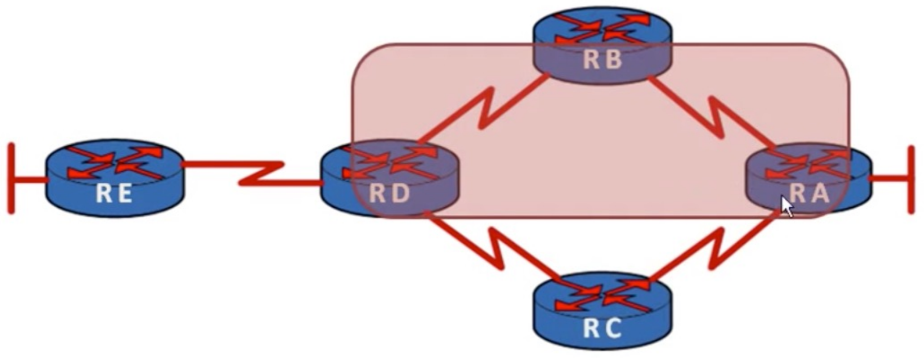
- Each router only knows directly-connected neighbours (reachable via the same physical link), and link costs to its neighbours

- Contrary to link-state protocols, each router possesses a limited vision of the topology



RA only "knows" RB and RC and trust those neighbours to route traffic towards other networks

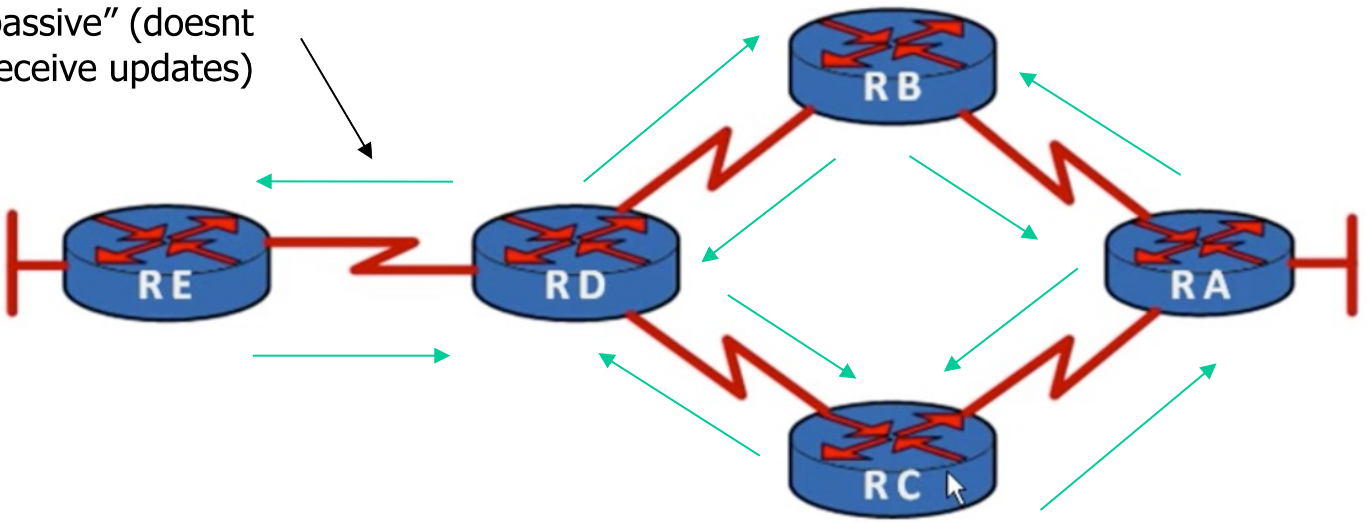
Same thing for RB in relation to RD and RA



Distance-Vector Protocols

- Each router sends periodic *updates* to its neighbours (even if no changes in the network have to be reported)
- The updates include the complete contents of the routing table of the router

Interface can also be "passive" (doesnt receive updates)



Routing protocols (link-state and distance-vector)

Protocolo	Tipo de encaminhamento		Algoritmo
	interior	exterior	
RIP (*Routing Information Protocol*), v1	●		***distance-vector***
RIP (*Routing Information Protocol*), v2	●		***distance-vector***
IGRP (*Interior Gateway Routing Protocol*)	●		***distance-vector***
EIGRP (*Enhanced Interior Gateway Routing Protocol*)	●		***distance-vector +link-state***
OSPF (*Open Shortest Path First*)	●		***link-state***
EGP (*Exterior Gateway Protocol*)		●	–
BGP (*Border Gateway Protocol*)		●	–

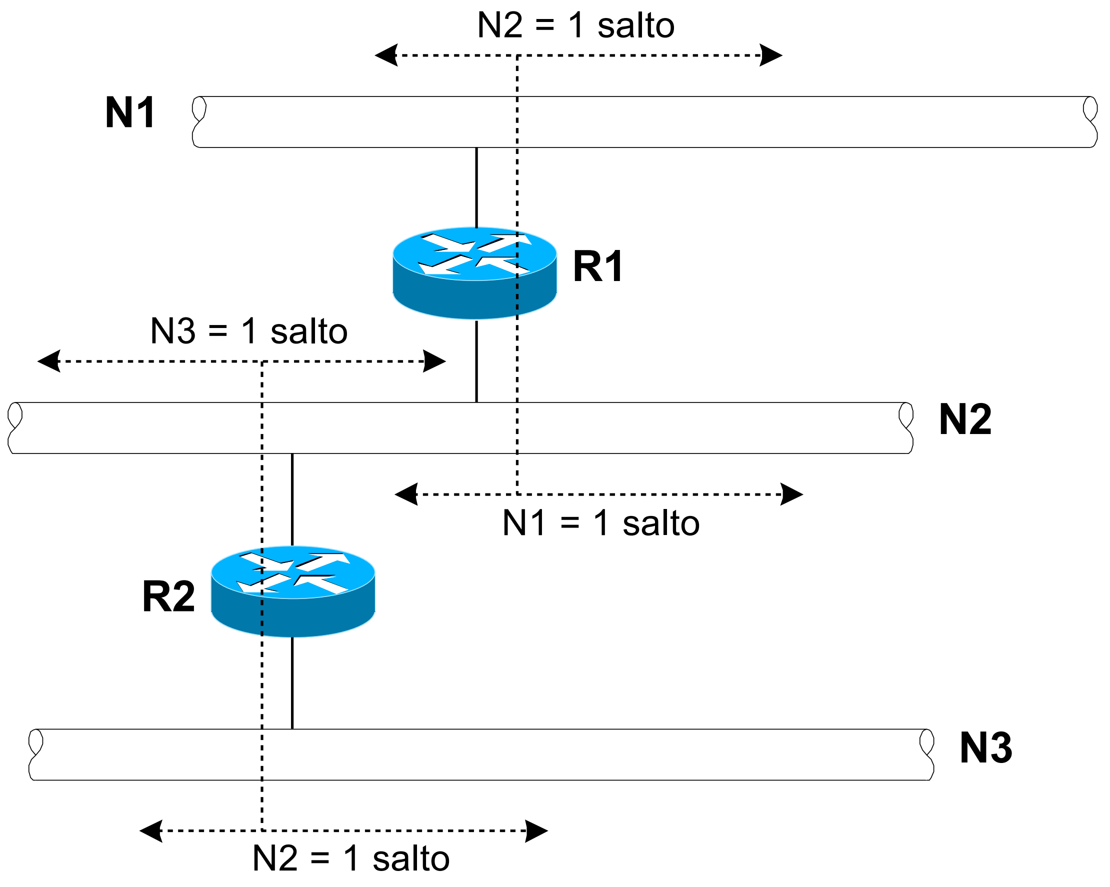
Router Information Protocol

- RIP-2 – Defined in RFC 1723 (1994)
- Router *broadcasts* routing information at each 30 s
- Sends immediate update upon detecting change on a link
- Router uses information received from its neighbours to calculate the shortest paths to all reachable (and known) destinations
- Uses *hop count* as metric
- Maximum hop count of 15 (16 is considered “infinite”, or “unreachable”)
- Reports are broadcasted to neighbours
- A route expires if no update is received for 180s

Routing Information Protocol

- May store up to 6 equal cost paths to same destination
- Supports load balancing using paths with the same cost
- Timer values used by the protocol:
  - **Update interval**: 30s (periodic update of routes sent to neighbours)
  - **Invalid timer**: 180s (time since last update for route, upon which route is marked as invalid and put "on hold": *hop count* 16 or "infinite")
  - **Flush timer**: 240 s (time since last update upon which route is flushed or deleted)

Routing Information Protocol



Routing Information Protocol

Example (update of routes using "distance vector" strategy):

A

Rede dest.	Router seg.	Dist.
1	-	1
2	B	2
3	B	4
4	D	2
5	D	5

Estado inicial

B	C	D
3	7	4
1	6	3
4	5	2
3	5	1
5	5	2

Vectores recebidos

A

Rede dest.	Router seg.	Dist.
1	-	1
2	B	2
3	D	3
4	D	2
5	D	3

Estado final

T04: outline

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

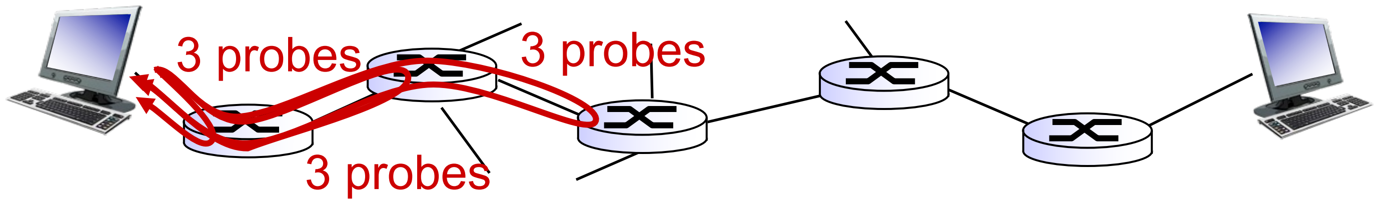
Traceroute and ICMP

- source sends series of UDP segments to destination
  - first set has TTL =1
  - second set has TTL=2, etc.
  - unlikely port number
- when datagram in *n*th set arrives to nth router:
  - router discards datagram and sends source ICMP message (type 11, code 0)
  - ICMP message include name of router & IP address

- when ICMP message arrives, source records RTTs

stopping criteria:

- UDP segment eventually arrives at destination host
- destination returns ICMP "port unreachable" message (type 3, code 3)
- source stops



T04: summary

4.1 Overview of Network layer
- data plane
- control plane

4.2 What's inside a router

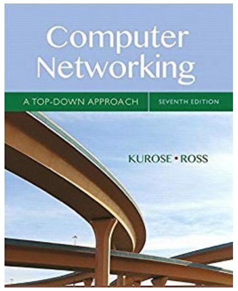
4.3 IP: Internet Protocol
- datagram format
- fragmentation
- IPv4 addressing
- network address translation

4.4 Routing protocols
- link state
- Distance vector
- Internal and external routing

4.5 ICMP: The Internet Control Message Protocol

T04: Bibliography

J. Kurose and K. Ross, “Computer Networking – a top-down approach”, Pearson. Chapter 4: The network Layer



Redes de Comunicação 2023/2024

T04
Network Layer
Extra material

Jorge Granjal
University of Coimbra

UNIVERSIDADE D COIMBRA

T04: Firewalls

- A firewall is a network security device that monitors incoming and outgoing network traffic and decides whether to allow or block specific communications.
- Firewalls support a first line of defense in network security. They establish a barrier between secured and controlled internal (trusted) networks.

[Introduction of Firewall in Computer Network]

[DDoS Attack Explained]