UNIVERSITY OF WATERLOO

CS 456/656
Computer Networks

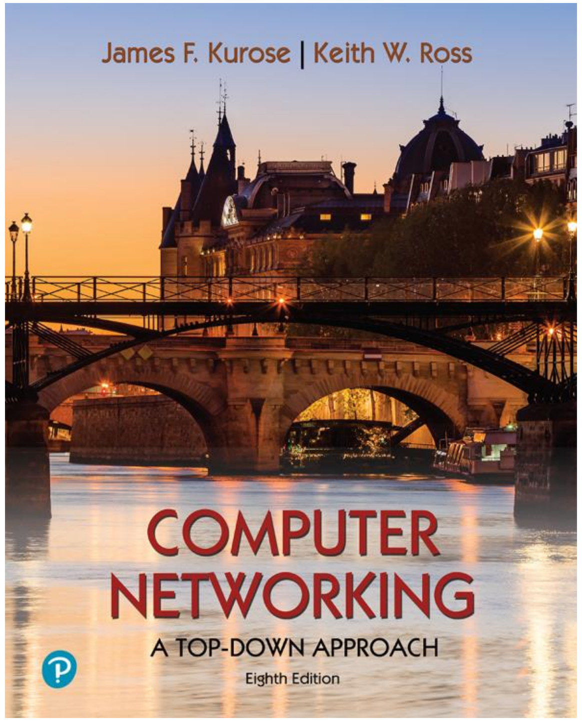
Lecture 14: Network Layer – Part 5

Mina Tahmasbi Arashloo and Bo Sun

Fall 2024

A note on the slides

Adapted from the slides that accompany this book.



Computer Networking: A Top-Down Approach
8th edition
Jim Kurose, Keith Ross
Pearson, 2020

Network Layer in the Internet

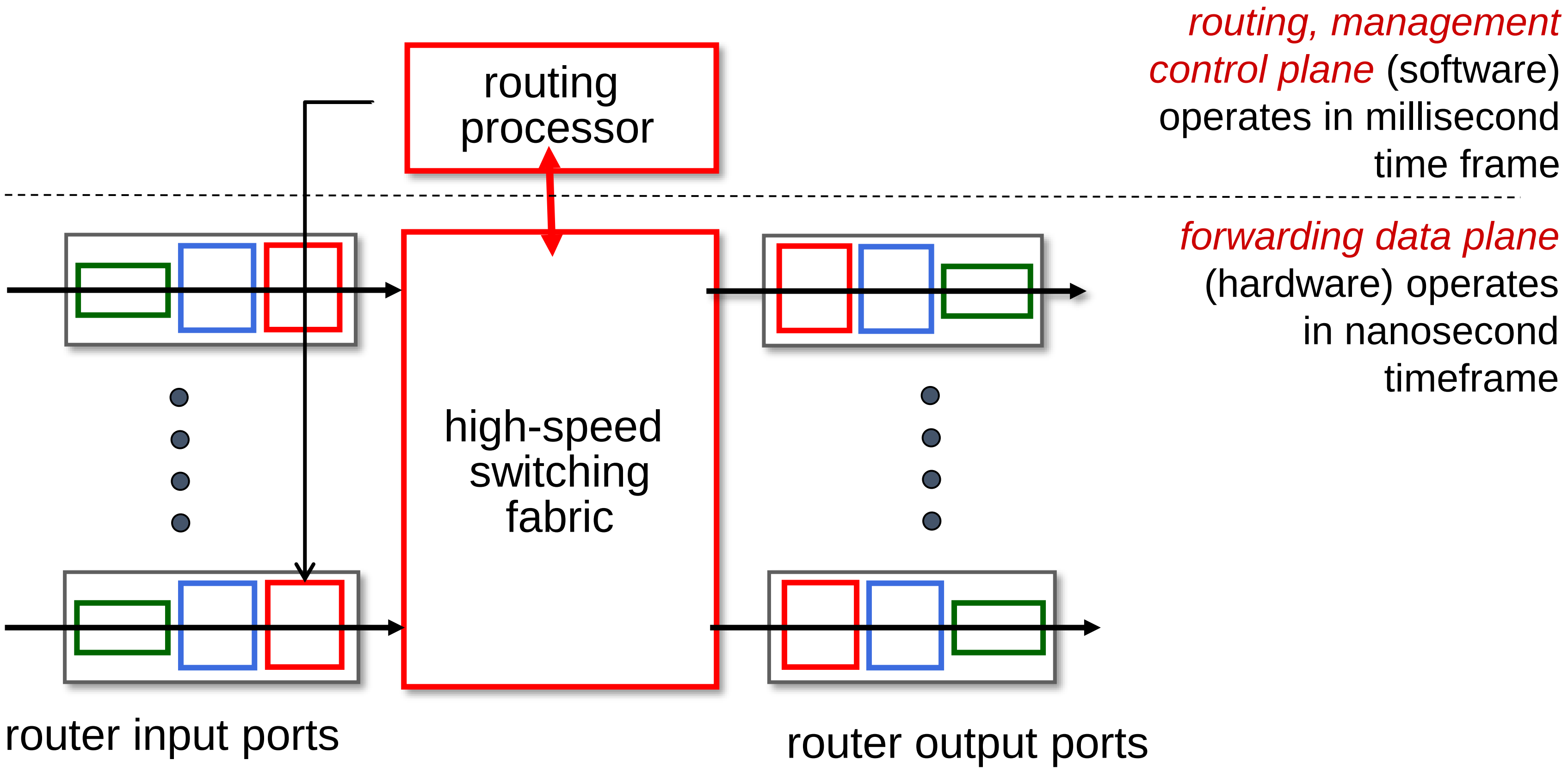
- The Internet Protocol (IP)
- Internet Routing
  - Hierarchical routing
  - Intra-ISP routing: OSPF
  - Inter-ISP routing: BGP
- Internet Forwarding

Network Layer in the Internet

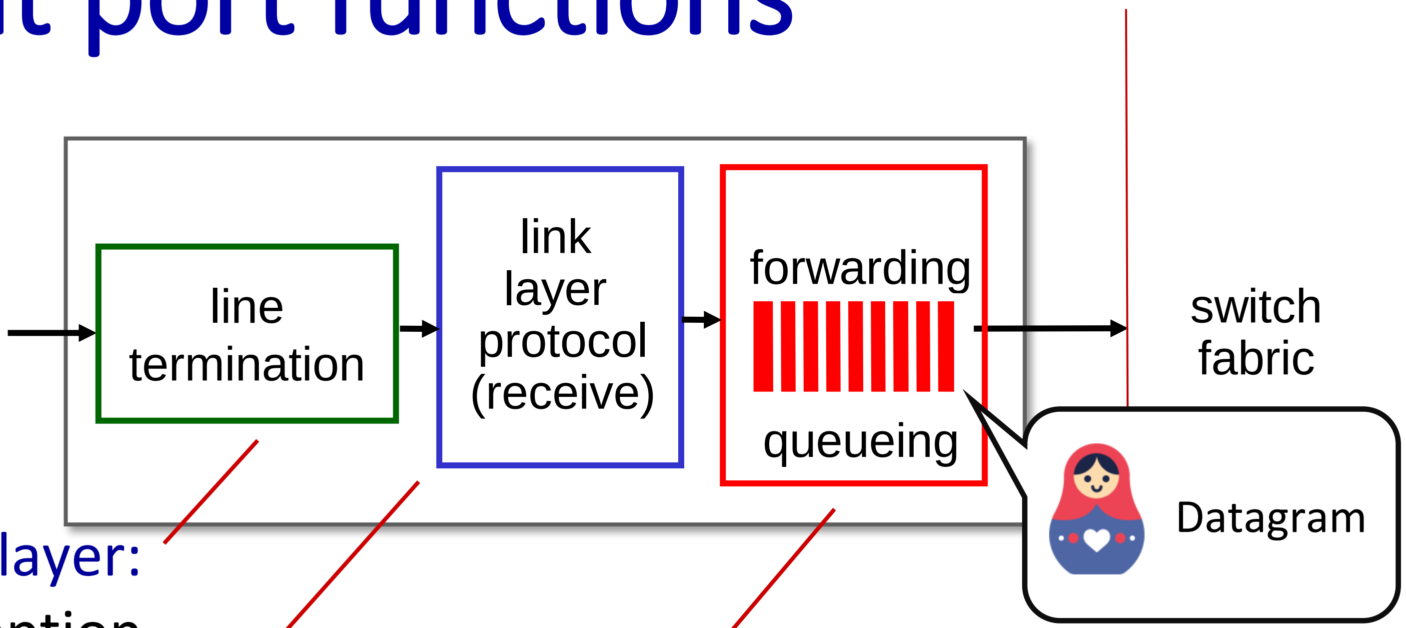
- The Internet Protocol (IP)
- Internet Routing
  - Hierarchical routing
  - Intra-ISP routing: OSPF
  - Inter-ISP routing: BGP
- Internet Forwarding
  - What is inside a router?
  - Destination-based forwarding

What is inside a router?

high-level view of generic router architecture:



Input port functions



physical layer:
bit-level reception

link layer:
e.g., Ethernet
(next layer)

- forwarding: The forwarding table we have been talking about goes here!
  - Looks up the next hop/interface the packet should go to.
  - The switching fabric will then move the packet to exit from that interface (output port)
- input port queuing: if datagrams arrive faster than forwarding rate into switch fabric, they will line up here.

Network Layer in the Internet

- The Internet Protocol (IP)
- Internet Routing
  - Hierarchical routing
  - Intra-ISP routing: OSPF
  - Inter-ISP routing: BGP
- Internet Forwarding
  - What is inside a router?
  - Destination-based forwarding

Destination-based forwarding

- This can get really large!
  - There are 2^32 IPv4 addresses
- How do we scale it?
- Notice a pattern?

Destination Address	Link Interface
10.20.30.0	0
10.20.30.1	0
…	…
10.20.30.255	0
5.4.0.0	1
5.4.0.1	1
…	…
5.4.255.255	1
…	…

Destination-based forwarding

- This can get really large!
  - There are 2^32 IPv4 addresses
- How do we scale it?
- Notice a pattern?

Destination Address	Link Interface
10.20.30.0	0
10.20.30.1	0
…	…
10.20.30.255	0
5.4.0.0	1
5.4.0.1	1
…	…
5.4.255.255	1
…	…

Destination-based forwarding

- This can get really large!
  - There are 2^32 IPv4 addresses
- How do we scale it?
- Notice a pattern?
- Why does this pattern exist?
  - Hint: think about the Internet architecture…

Destination Address	Link Interface
10.20.30.0	0
10.20.30.1	0
…	…
10.20.30.255	0
5.4.0.0	1
5.4.0.1	1
…	…
5.4.255.255	1
…	…

Destination-based forwarding

Destination Prefix	Link Interface
10.20.30.0/24	0
5.4.0.0/16	1
…	…

- How does the router use this for forwarding?
  - When a packet comes in with destination address X
  - Check whether X *matches* a prefix in the table
  - If yes, send it to the corresponding interface (action)
  - If not, drop the packet.

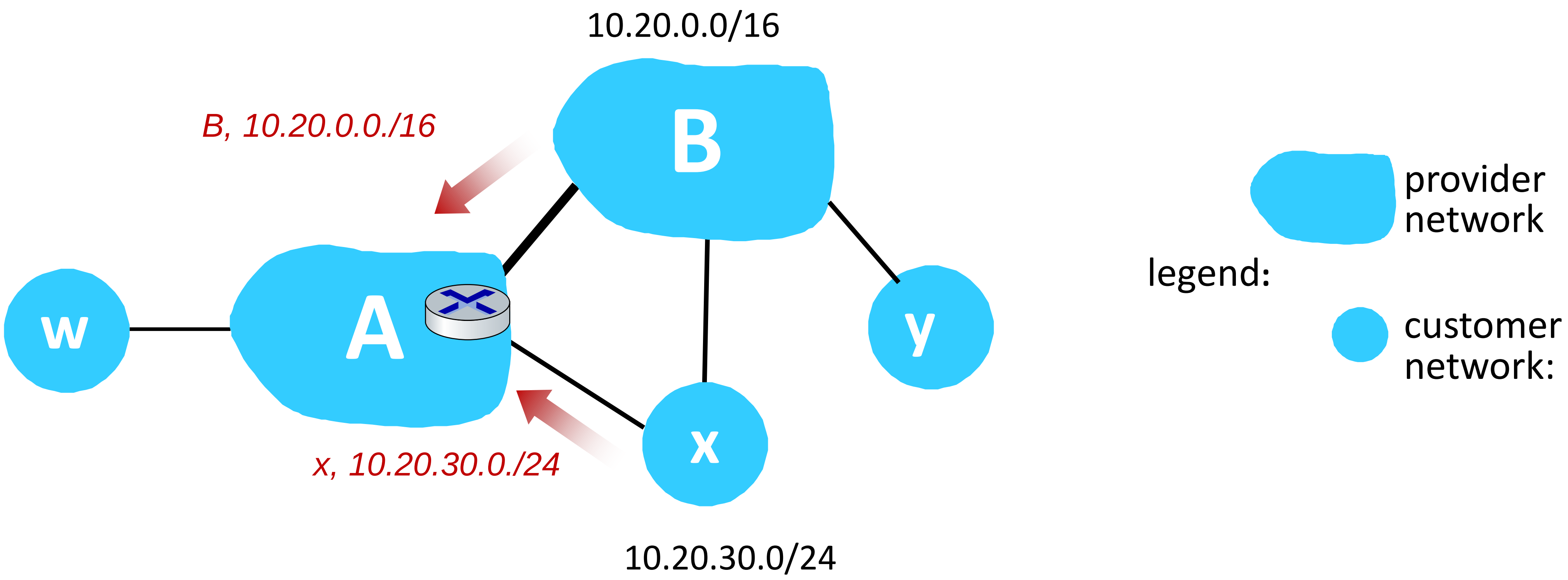
Destination-based forwarding

Destination IP address:
10.20.30.5

Destination Prefix	Link Interface
10.20.30.0/24	0
5.4.0.0/16	1
10.20.0.0/16	2
…	…

- What if a packet matches multiple entries?

How do overlapping entries happen?



Destination-based forwarding

Destination IP address:
10.20.30.5

Destination Prefix	Link Interface
10.20.30.0/24	0
5.4.0.0/16	1
10.20.0.0/16	2
…	…

- What if a packet matches multiple entries?
  - Pick the one that has the longest prefix matching the packet's destination
  - In the above example, 10.20.30.5 matches
    - the first 24 bits of 10.20.30.0/24
    - the first 16 bits of 10.20.0.0/16
    - So, the router picks the first entry, and sends the packet out of interface 0

Destination-based forwarding

Destination IP address:
10.20.30.5

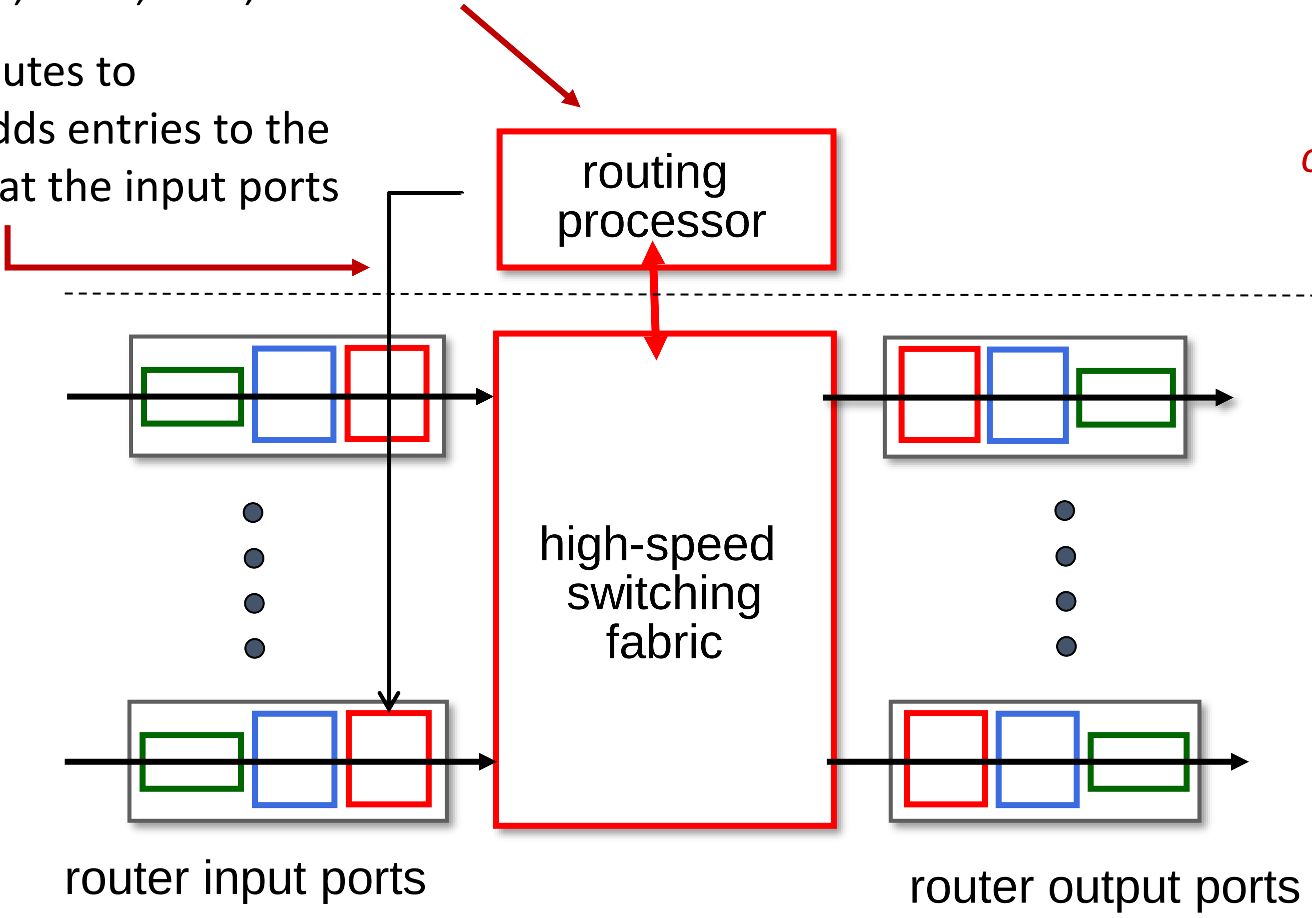
Destination <u>Prefix</u>	Link Interface
10.20.30.0/24	0
5.4.0.0/16	1
10.20.0.0/16	2
…	…

- What if a packet matches multiple entries?
  - Pick the one that has the longest prefix matching the packet's destination
  - i.e., the "most specific" prefix

Populating forwarding tables

The router's control plane runs the routing protocol(s)
e.g., OSPF, eBGP, iBGP, etc.

When it learns routes to destinations, it adds entries to the forwarding table at the input ports



routing, management control plane (software) operates in millisecond time frame

forwarding data plane (hardware) operates in nanosecond timeframe

Make sure you know

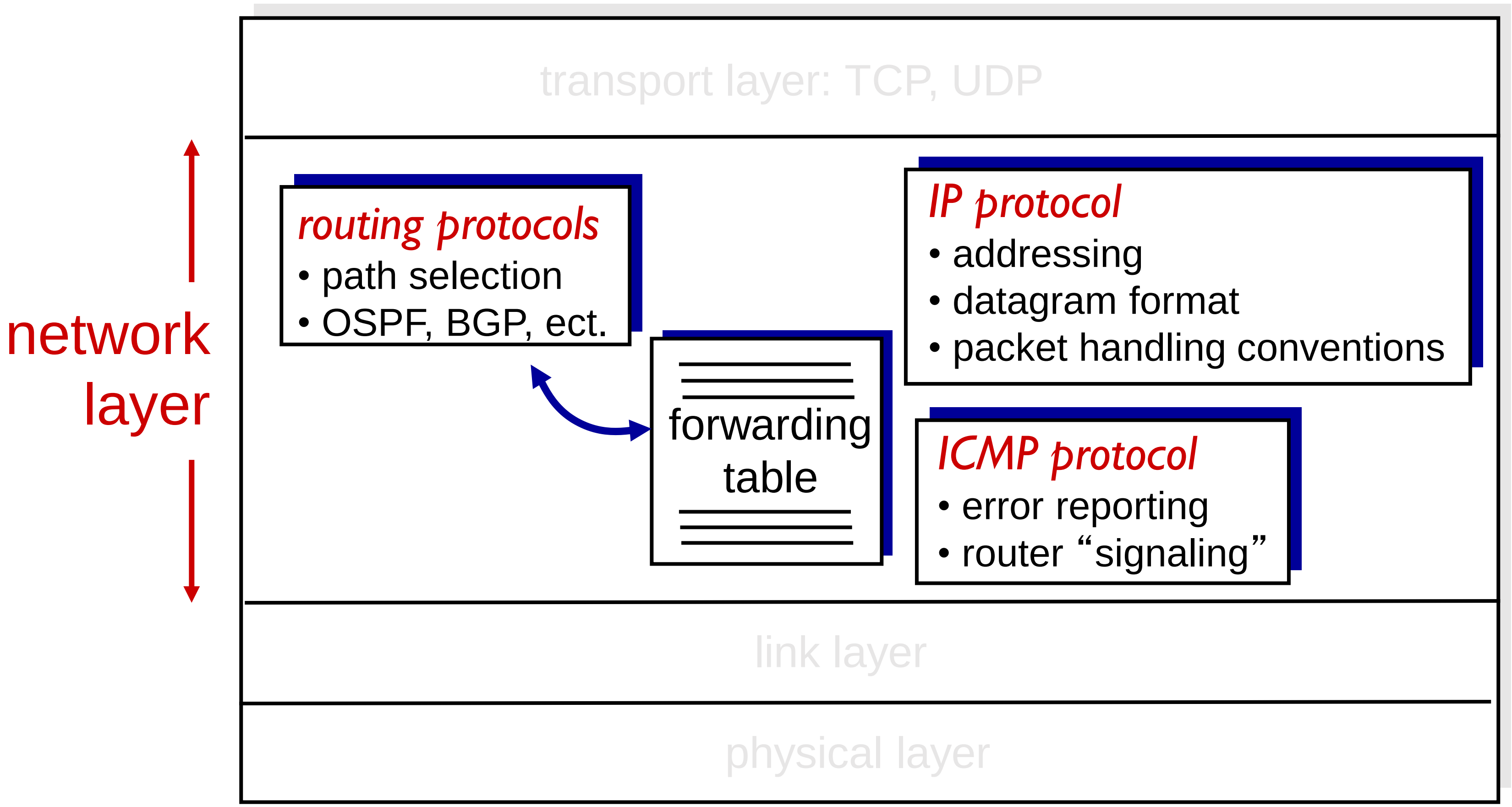
- The high-level architecture of a router
  - Input ports, switching fabric, output ports
  - What processing happens in the input ports
  - The role of the switching fabric
- What the forwarding table looks like for destination-based forwarding
- What longest-prefix matching means
- E.g., Given a packet and a table, you should be able to determine what the outgoing interface is.

Network Layer in the Internet

- The Internet Protocol (IP)
- Internet Routing
  - Hierarchical routing
  - Intra-ISP routing: OSPF
  - Inter-ISP routing: BGP
- Internet Forwarding
  - What is inside a router?
  - Destination-based forwarding
- ICMP

The Internet needs error reporting and testing

- What happens when something goes wrong during forwarding?



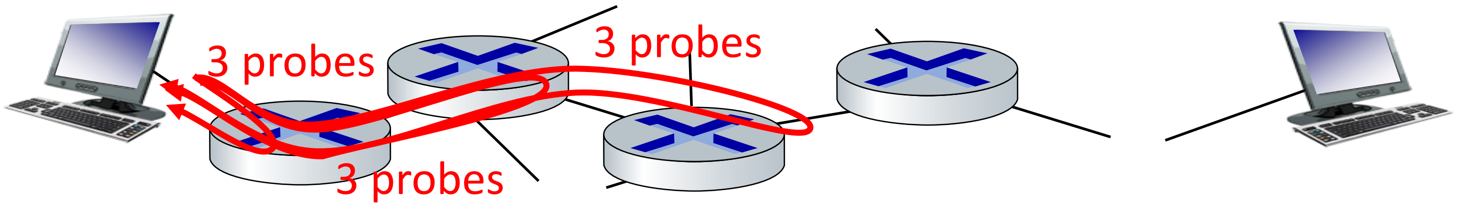
ICMP: internet control message protocol

- used by hosts and routers to communicate network-level information
  - error reporting: unreachable host, network, port, protocol
  - echo request/reply (used by ping)
- ICMP messages carried in IP datagrams
  - Still considered part of the network-layer, not transport
- *ICMP message:* type, code plus first 8 bytes of IP datagram causing error

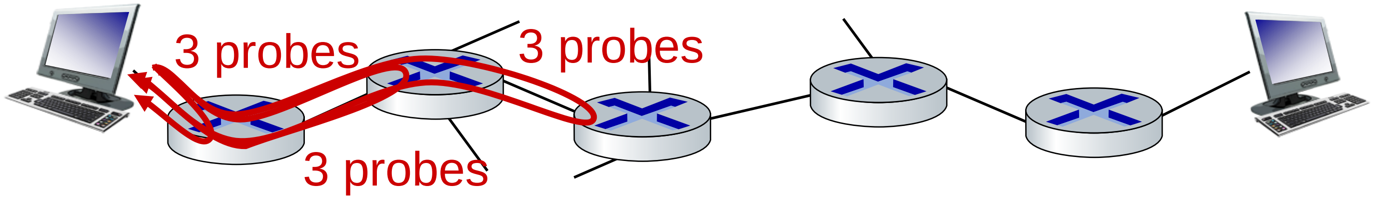
Type	Code	description
0	0	echo reply (ping)
3	0	dest. network unreachable
3	1	dest host unreachable
3	2	dest protocol unreachable
3	3	dest port unreachable
3	6	dest network unknown
3	7	dest host unknown
4	0	source quench (congestion control - not used)
8	0	echo request (ping)
9	0	route advertisement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

Traceroute and ICMP

- **traceroute** program: provides delay measurement (RTT) from source to router along end-end Internet path towards destination
  - implemented using TTL in the IP datagram header and ICMP message (type 11, code 0: TTL expired)



Traceroute and ICMP



- source sends sets of UDP segments to destination
  - 1^{st} set has TTL =1, 2^{nd} set has TTL=2, etc.
- datagram in *n*th set arrives to nth router:
  - router discards datagram and sends source ICMP message (type 11, code 0)
  - ICMP message possibly includes name of router & IP address
- when ICMP message arrives at source: record RTTs

stopping criteria:

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

Network Layer in the Internet

- The Internet Protocol (IP)
- Internet Routing
  - Hierarchical routing
  - Intra-ISP routing: OSPF
  - Inter-ISP routing: BGP
- Internet Forwarding
  - What is inside a router?
  - Destination-based forwarding
- ICMP
- Beyond destination-based routing and forwarding
  - The end-to-end argument
  - Middleboxes
  - Software-defined network (SDNs)

Network Layer in the Internet

- The Internet Protocol (IP)
- Internet Routing
  - Hierarchical routing
  - Intra-ISP routing: OSPF
  - Inter-ISP routing: BGP
- Internet Forwarding
  - What is inside a router?
  - Destination-based forwarding
- Beyond destination-based routing and forwarding
  - The end-to-end argument
  - Middleboxes
  - Software-defined network (SDNs)

Architectural Principles of the Internet

RFC 1958

"Many members of the Internet community would argue that there is no architecture, but only a tradition, which was not written down for the first 25 years (or at least not by the IAB). However, in very general terms, the community believes that the goal is connectivity, the tool is the Internet Protocol, and the intelligence is end to end rather than hidden in the network."

Three cornerstone beliefs:

- simple connectivity
- IP protocol: that narrow waist
- intelligence, complexity at network edge

Architectural Principles of the Internet

RFC 1958

"Many members of the Internet community would argue that there is no architecture, but only a tradition, which was not written down for the first 25 years (or at least not by the IAB). However, in very general terms, the community believes that the goal is connectivity, the tool is the Internet Protocol, and the intelligence is end to end rather than hidden in the network."

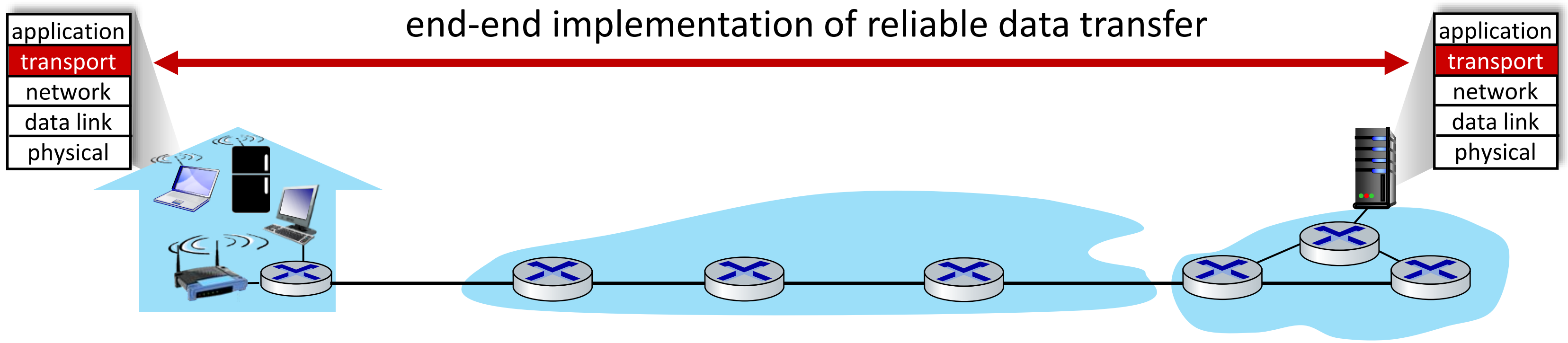
Three cornerstone beliefs:

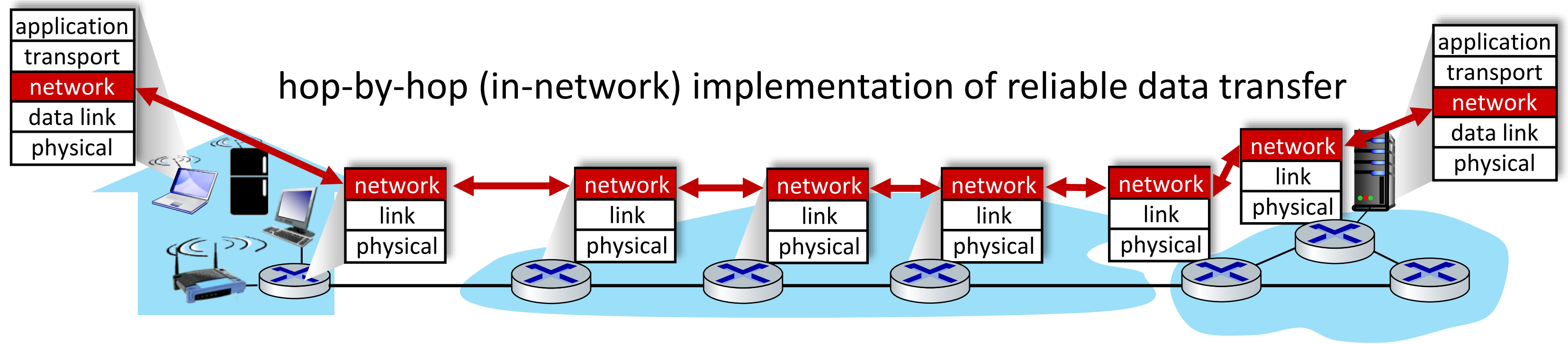
- simple connectivity
- IP protocol: that narrow waist
- intelligence, complexity at network edge

The end-to-end argument

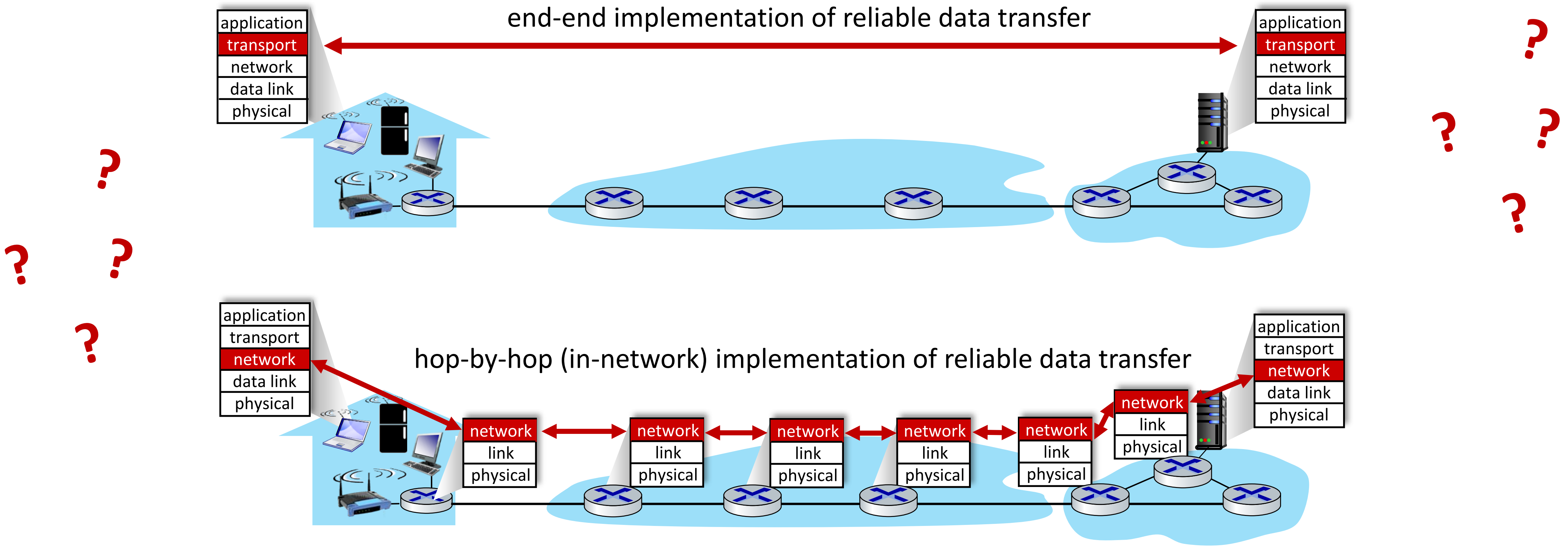
- We have talked about how some of the more complex network functionality, e.g., reliable data transfer and congestion control, is implemented in the transport layer at the end points
  - as opposed to inside the network (e.g., in the routers)
- But that didn’t necessarily have to be the case
- One could come up with a network design where these functionalities are implemented inside the network
  - In fact, Infiniband networks have made different choices in this regard compared to the current Internet

The end-to-end argument





The end-to-end argument



The end-to-end argument

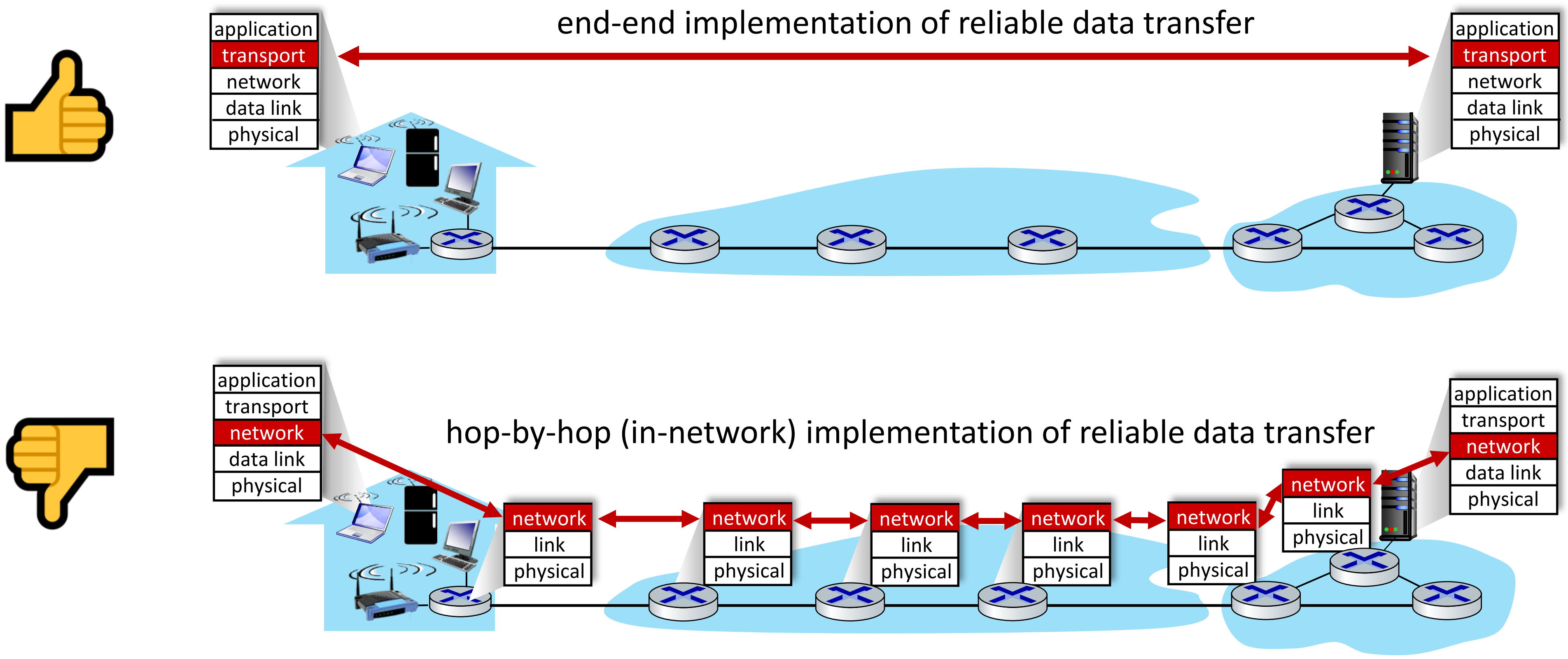
> "The function in question can completely and correctly be implemented only with the knowledge and help of the application standing at the end points of the communication system. Therefore, providing that questioned function as a feature of the communication system itself is not possible. (Sometimes an incomplete version of the function provided by the communication system may be useful as a performance enhancement.)

We call this line of reasoning against low-level function implementation the "end-to-end argument."

Saltzer, Reed, Clark 1981

The end-to-end argument

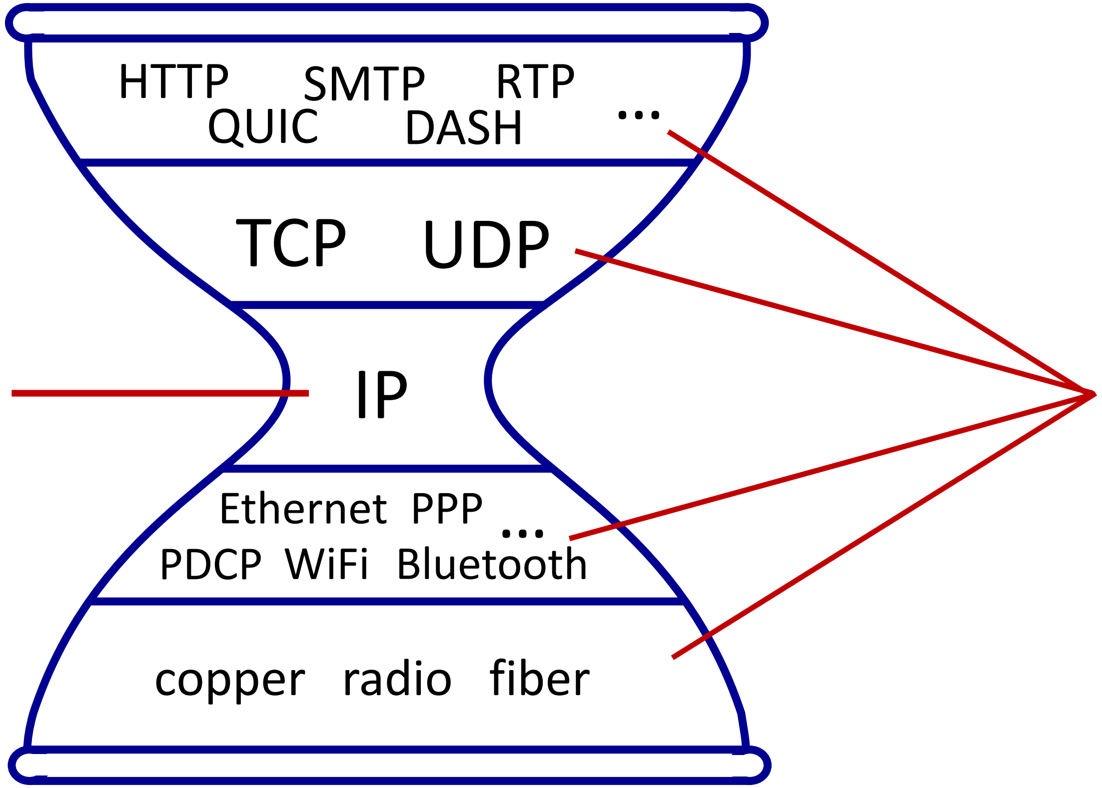
- In the Internet, the choice ended up being…



The IP hourglass

Internet's "thin waist":

- *one* network layer protocol: IP
- *must* be implemented by every (billions) of Internet-connected devices



many protocols in physical, link, transport, and application layers

Side note: end-to-end argument is not set in stone

- This was a design decision made by the designers of the Internet protocol stack to meet their requirements for the Internet
- Other networks with different requirements can make different choices
- For example, the InfiniBand protocol stack offers reliable data transfer at the network layer
  - No packets will be lost!
  - More limited in scale
  - Can’t easily inter-operate with the Internet

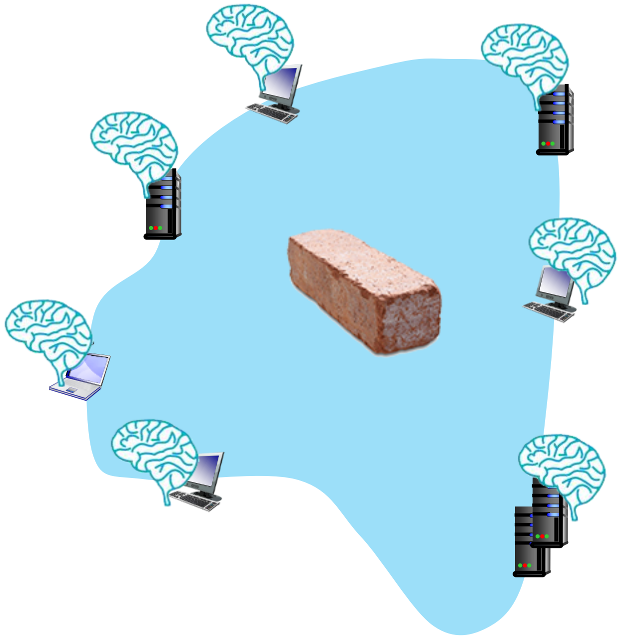
Make sure you know

- What the end-to-end argument is
  - With an example
- How it affected the design of the Internet and its protocol stack

Network Layer in the Internet

- The Internet Protocol (IP)
- Internet Routing
  - Hierarchical routing
  - Intra-ISP routing: OSPF
  - Inter-ISP routing: BGP
- Internet Forwarding
  - What is inside a router?
  - Destination-based forwarding
- ICMP
- Beyond destination-based routing and forwarding
  - The end-to-end argument
  - Middleboxes
  - Software-defined network (SDNs)

What if we need to do more in the network?



Internet (until early 2000s)
- "intelligence" at the end point

- Following the end-to-end argument, routers in the network would only do destination-based forwarding
  - No parsing beyond network layer header
  - No changing anything in the packet
    - Except for, say, flags like ECN or TTL
- But, it turned out, we may need more from the network after all.

Middleboxes

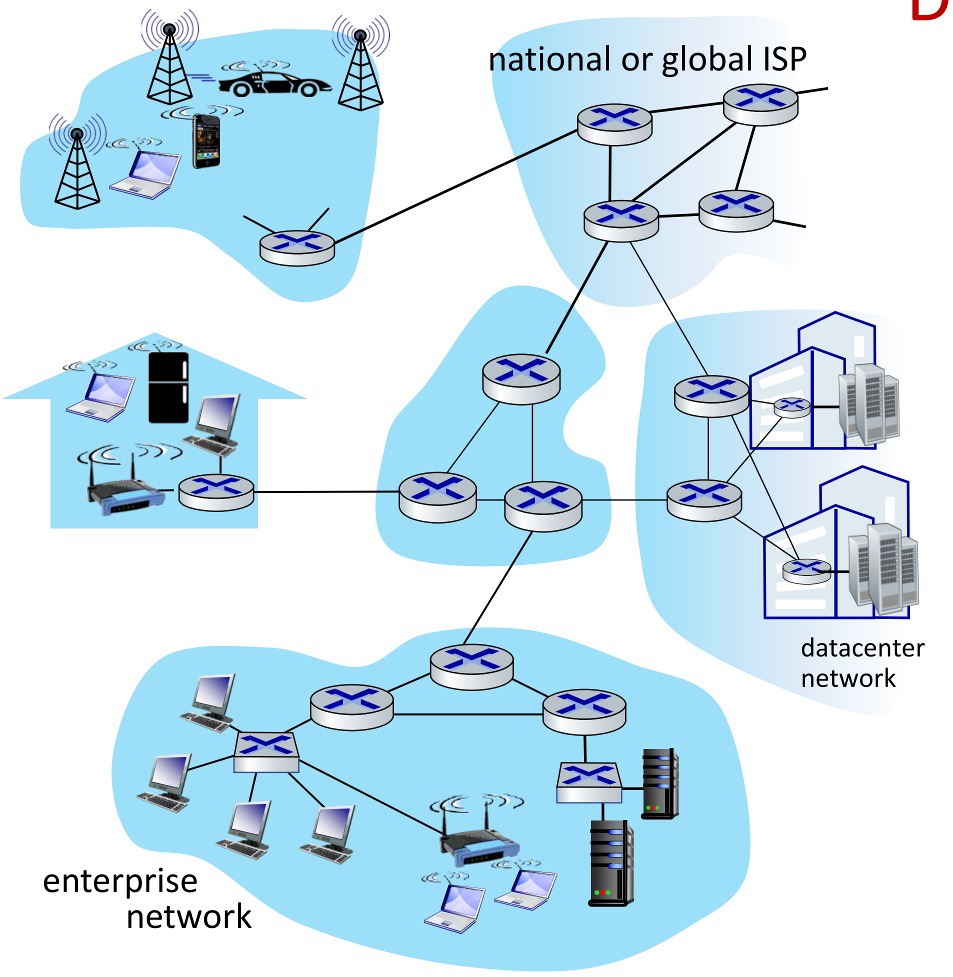
Middlebox (RFC 3234)

"any intermediary box performing functions apart from normal, standard functions of an IP router on the data path between a source host and destination host"

Middleboxes everywhere!

NAT: modifies IP addresses and ports

Load balancers: pick the server packets should go to



Firewalls, Intrusion Detection System (IDS): drop suspicious traffic

Caches: keep copies of popular data

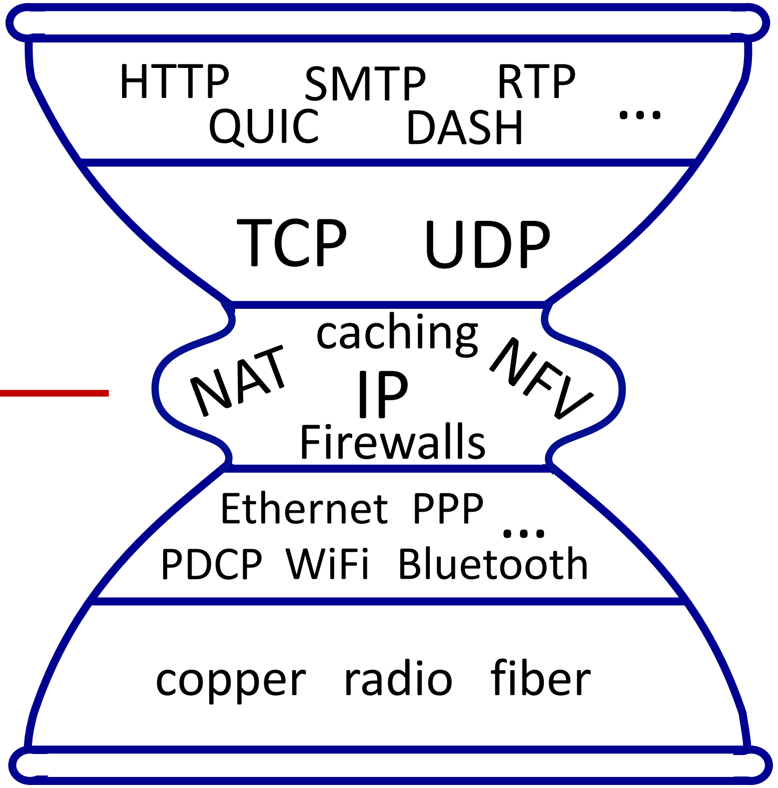
Middleboxes (or Network Functions)

- initially: proprietary (closed) hardware solutions
- move towards "whitebox" hardware
  - Hardware providing an open API (programming interface)
- Sometimes, high-speed implementation in software
  - network functions virtualization (NFV): specialized packet processing in, say, a VM.

The (more realistic) IP hourglass

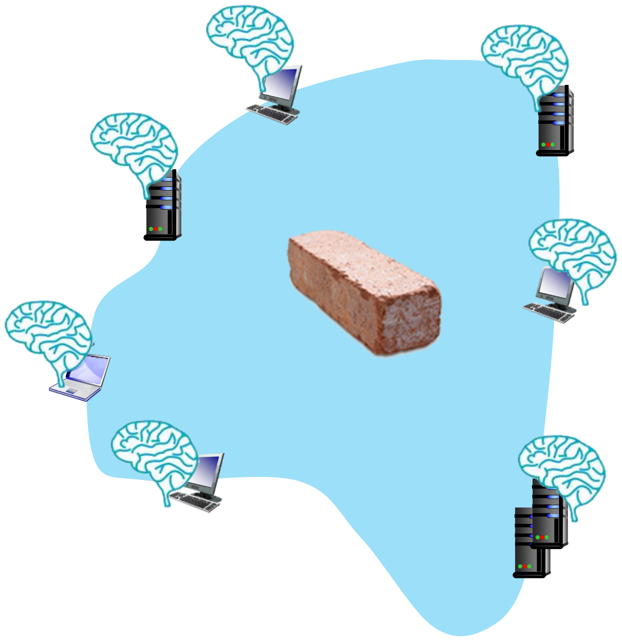
Internet's new waist?

- middleboxes, operating inside the network



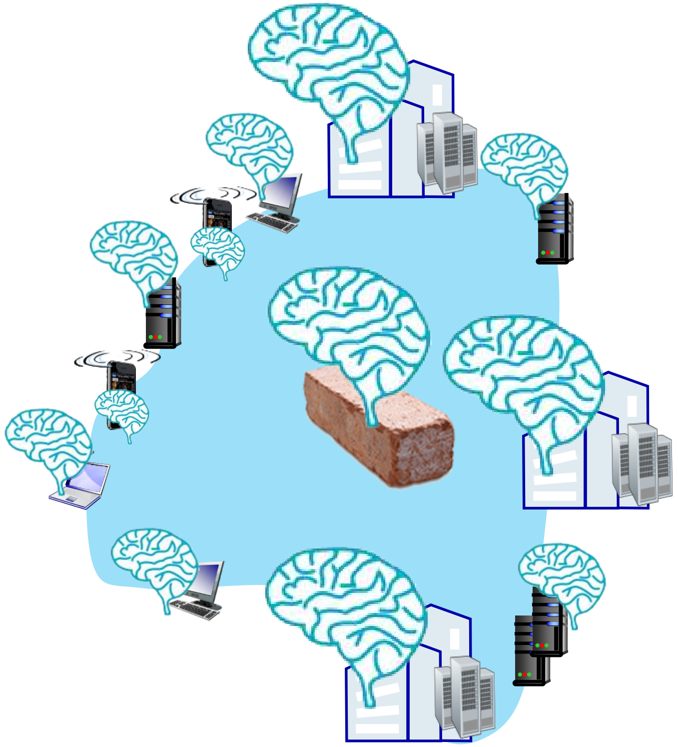
Do middleboxes break the end-to-end argument?

What if we need to do more in the network?



Internet (until early 2000s)

- intelligence, computing at edge



Internet (now)

- Middleboxes/NFV
- Software defined networks (SDNs)
- programmable network devices

Make sure you know

- What a middleboxes is
- Know some example middleboxes and what they do
- How the functionality inside the network has changed since its early days

Network Layer in the Internet

- The Internet Protocol (IP)
- Internet Routing
  - Hierarchical routing
  - Intra-ISP routing: OSPF
  - Inter-ISP routing: BGP
- Internet Forwarding
  - What is inside a router?
  - Destination-based forwarding
- ICMP
- Beyond destination-based routing and forwarding
  - The end-to-end argument
  - Middleboxes
  - Software-defined network (SDNs)

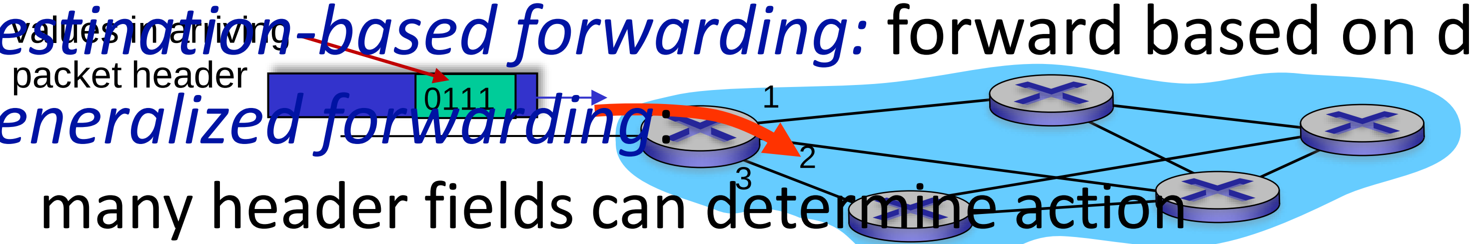
A more general packet processing model

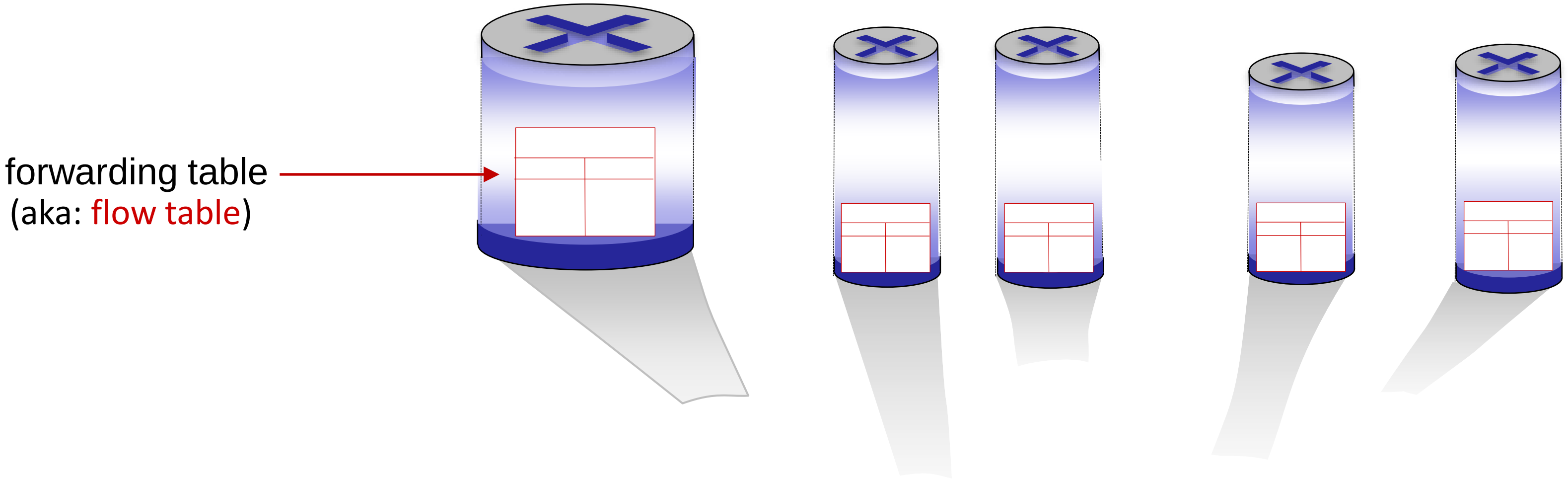
- Routers → Forwarding table that maps destination IP addresses of incoming packets to the outgoing interface
- Middleboxes → Each has their own custom approach to packet processing that goes beyond what routers traditionally do
  - Make decisions based on headers in other layers (e.g., transport)
  - Modify packet header and/or content
  - …
- OpenFlow → A proposal for generalized packet processing model devices in the network
  - One of the original building blocks of software-defined networking (SDN)

Generalized forwarding: match plus action

Review: each router contains a forwarding table (aka: flow table)

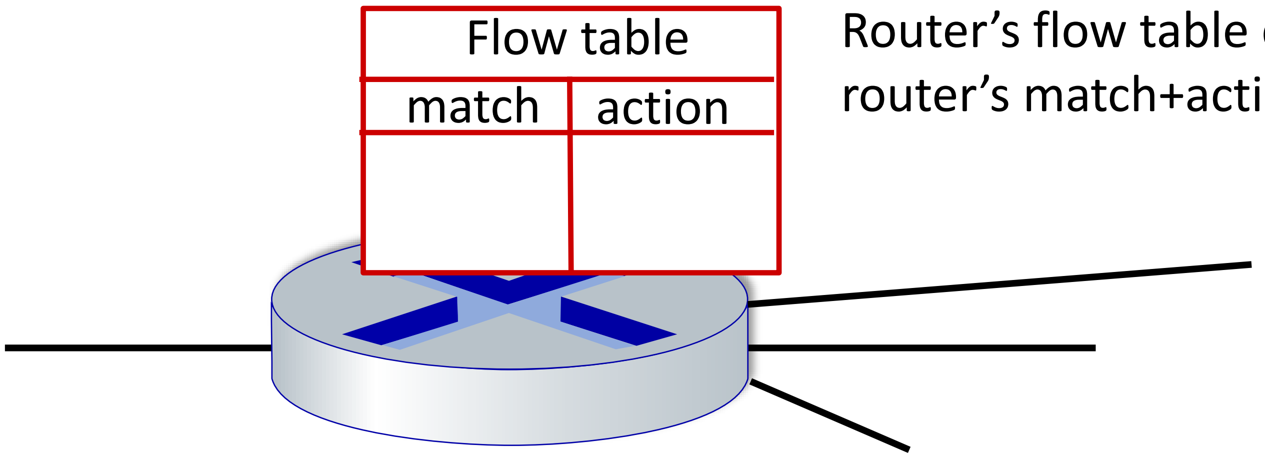
- "match plus action" abstraction: match bits in arriving packet, take action
  - *destination-based forwarding:* forward based on dest. IP address
  - *generalized forwarding:*
    - many header fields can determine action
    - many action possible: drop/copy/modify/log packet





Flow table abstraction

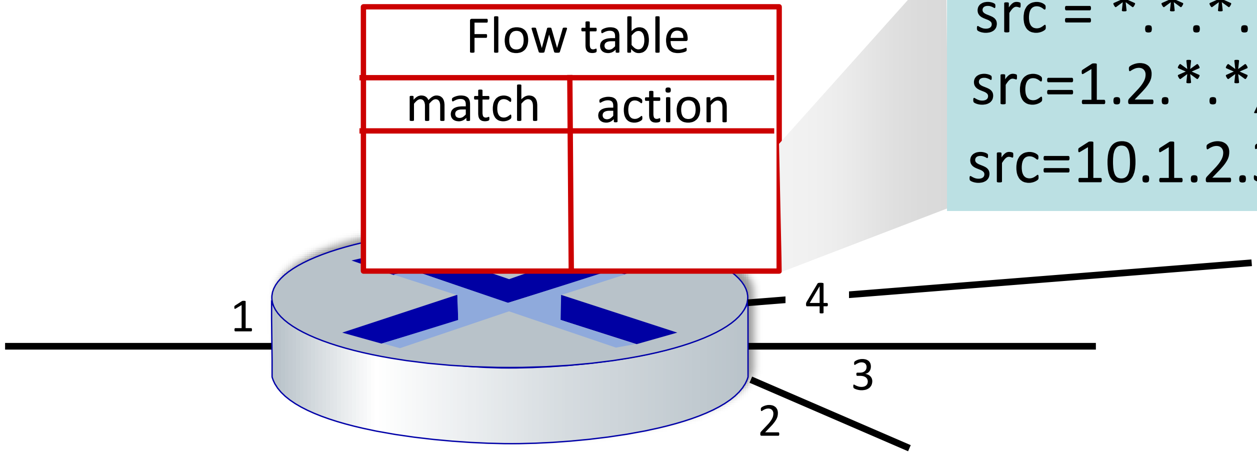
- flow: defined by header field values (in link-, network-, transport-layer fields)
- generalized forwarding: simple packet-handling rules
  - match: pattern values in packet header fields
  - actions: for matched packet: drop, forward, modify, matched packet or send matched packet to controller
  - priority: disambiguate overlapping patterns
  - counters: #bytes and #packets



Router's flow table define router's match+action rules

Flow table abstraction

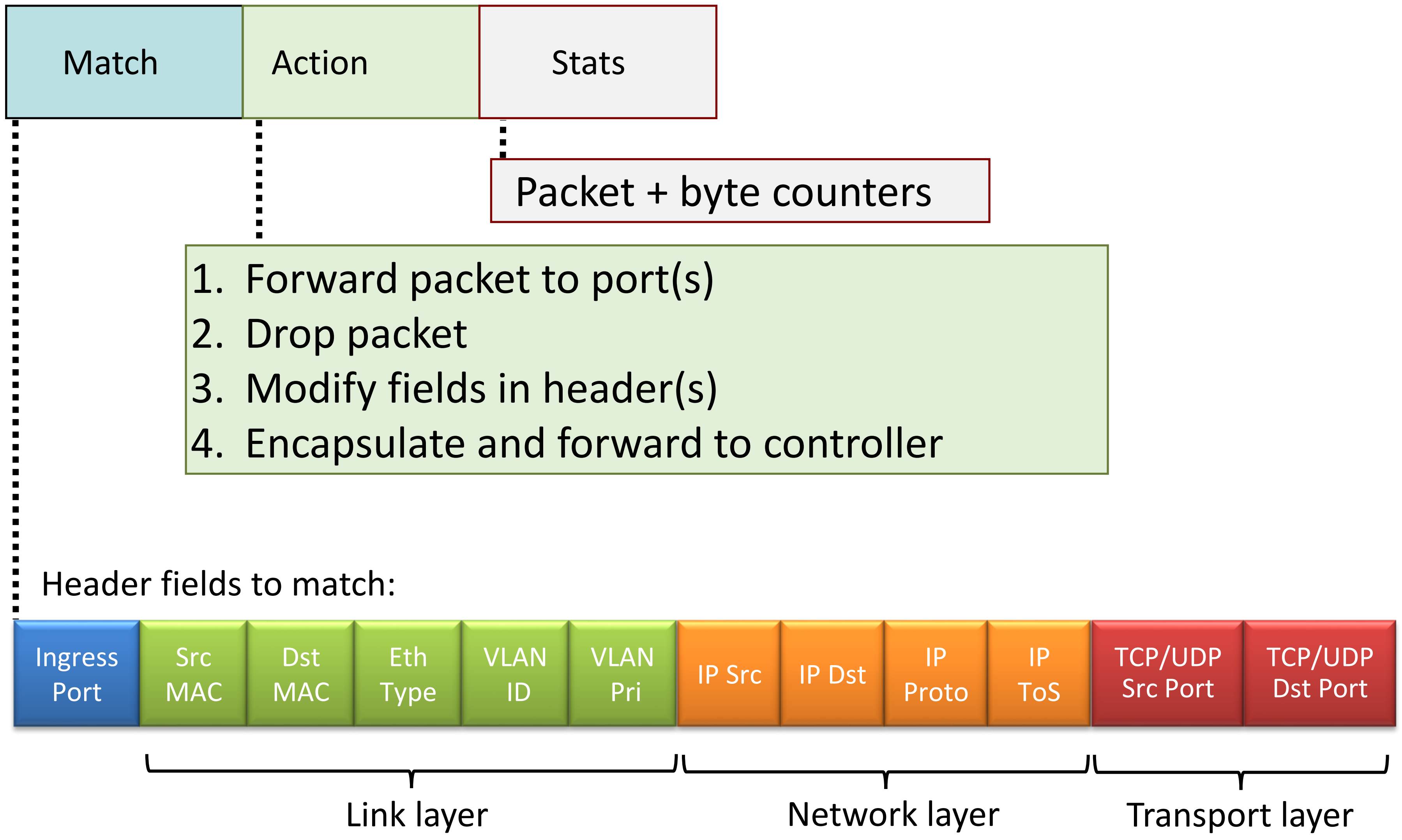
- flow: defined by header fields
- generalized forwarding: simple packet-handling rules
  - match: pattern values in packet header fields
  - actions: for matched packet: drop, forward, modify, matched packet or send matched packet to controller
  - priority: disambiguate overlapping patterns
  - counters: #bytes and #packets



src = *.*.*.*, dest=3.4.*.*	forward(2)
src=1.2.*.*, dest=*.*.*.*	drop
src=10.1.2.3, dest=*.*.*.*	send to controller

* : wildcard

OpenFlow: flow table entries



OpenFlow: examples

Destination-based forwarding:

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	VLAN Pri	IP Src	IP Dst	IP Prot	IP ToS	TCP s-port	TCP d-port	Action
*	*	*	*	*	*	*	51.6.0.8	*	*	*	*	port6

IP datagrams destined to IP address 51.6.0.8 should be forwarded to router output port 6

Firewall:

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	VLAN Pri	IP Src	IP Dst	IP Prot	IP ToS	TCP s-port	TCP d-port	Action
*	*	*	*	*	*	*	*	*	*	*	22	drop

Block (do not forward) all datagrams destined to TCP port 22 (ssh port #)

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	VLAN Pri	IP Src	IP Dst	IP Prot	IP ToS	TCP s-port	TCP d-port	Action
*	*	*	*	*	*	128.119.1.1	*	*	*	*	*	drop

Block (do not forward) all datagrams sent by host 128.119.1.1

OpenFlow: examples

Layer 2 destination-based forwarding:

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	VLAN Pri	IP Src	IP Dst	IP Prot	IP ToS	TCP s-port	TCP d-port	Action
*	*	22:A7:23: 11:E1:02	*	*	*	*	*	*	*	*	*	port3

layer 2 frames with destination MAC address 22:A7:23:11:E1:02 should be forwarded to output port 3

OpenFlow abstraction

- match+action: abstraction unifies different kinds of devices

Router
- *match:* longest destination IP prefix
- *action:* forward out a link

Switch
- *match:* destination MAC address
- *action:* forward or flood

Firewall
- *match*: IP addresses and TCP/UDP port numbers
- *action:* permit or deny

NAT
- *match:* IP address and port
- *action:* rewrite address and port

OpenFlow abstraction

- match+action: abstraction unifies different kinds of devices

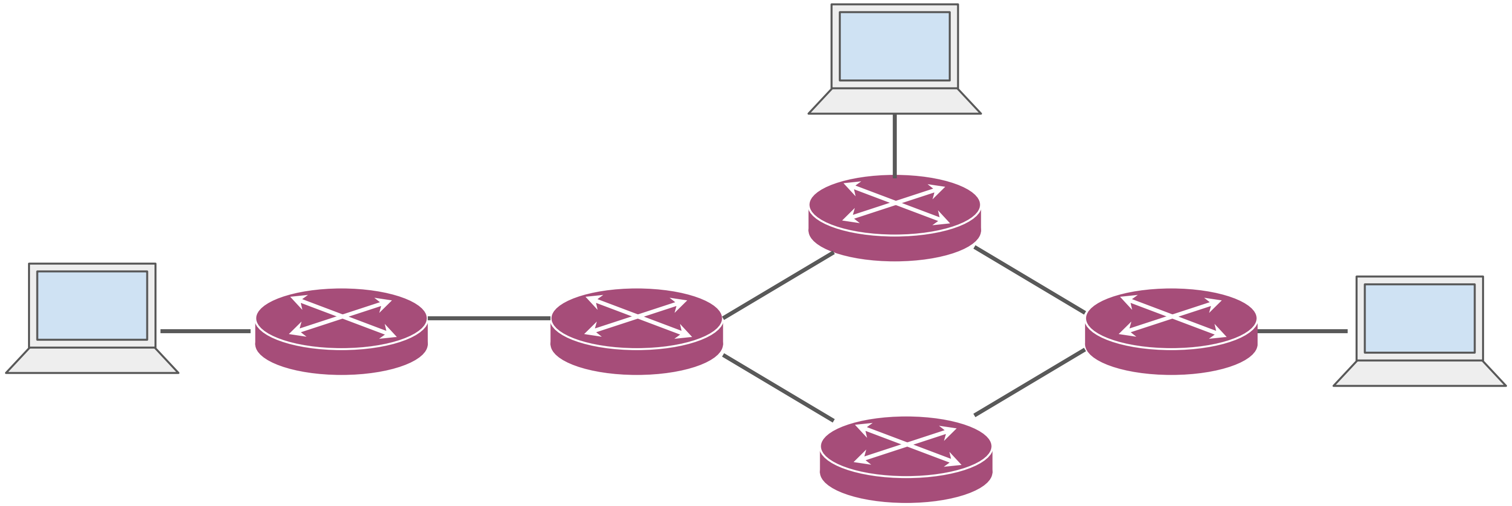
In SDN, we call all of them (packet) switches

Even outside of SDN context, it is now quite common to use the name switches to refer to a generic network device.

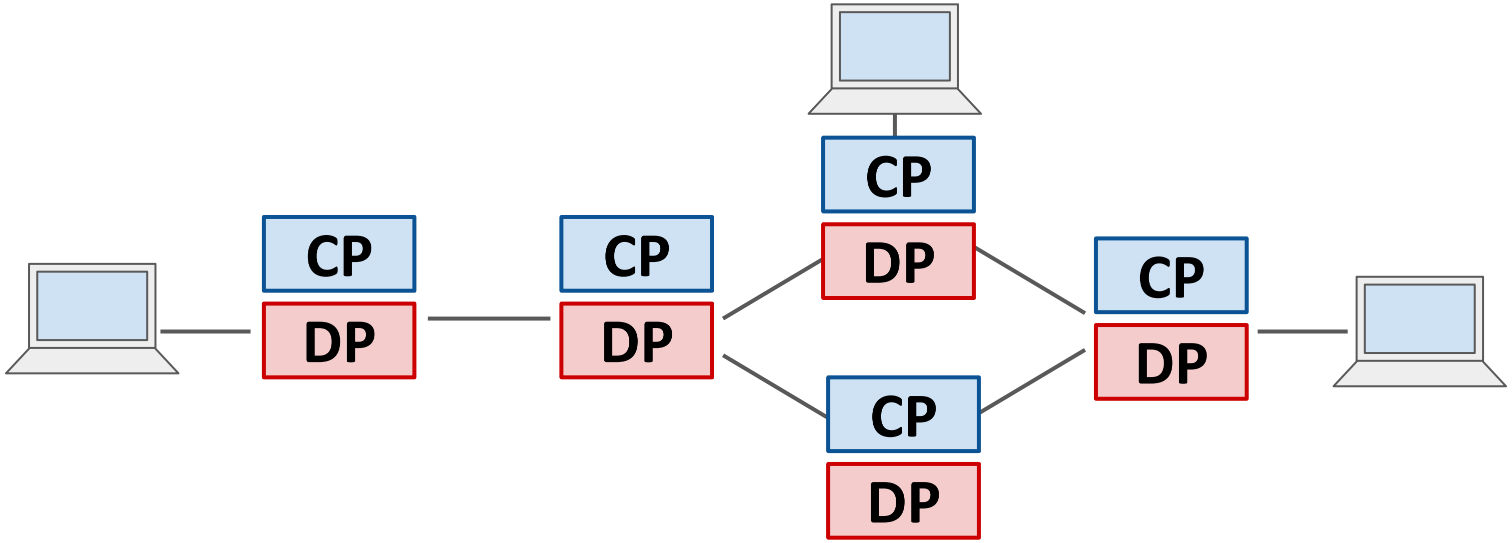
How does routing work in this generalized forwarding model?

- The routing protocols we had learned about so far were distributed
  - Every router runs an instance of the protocol
  - They communicate to figure out the best forwarding paths based on costs
- They were also designed with destination-based forwarding in mind.
- Software-Defined Networking takes a different approach…

Software-Defined Networking (SDN)

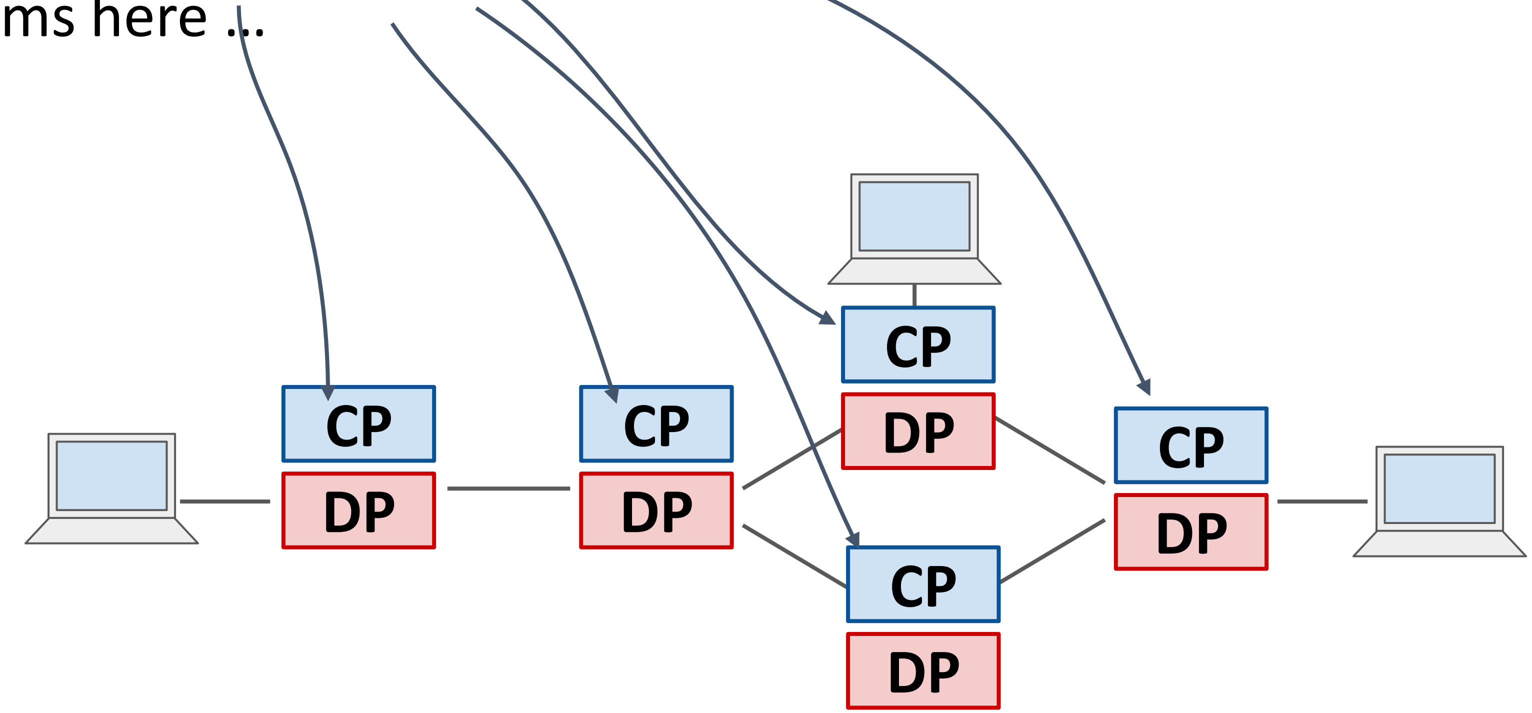


Software-Defined Networking (SDN)



Software-Defined Networking (SDN)

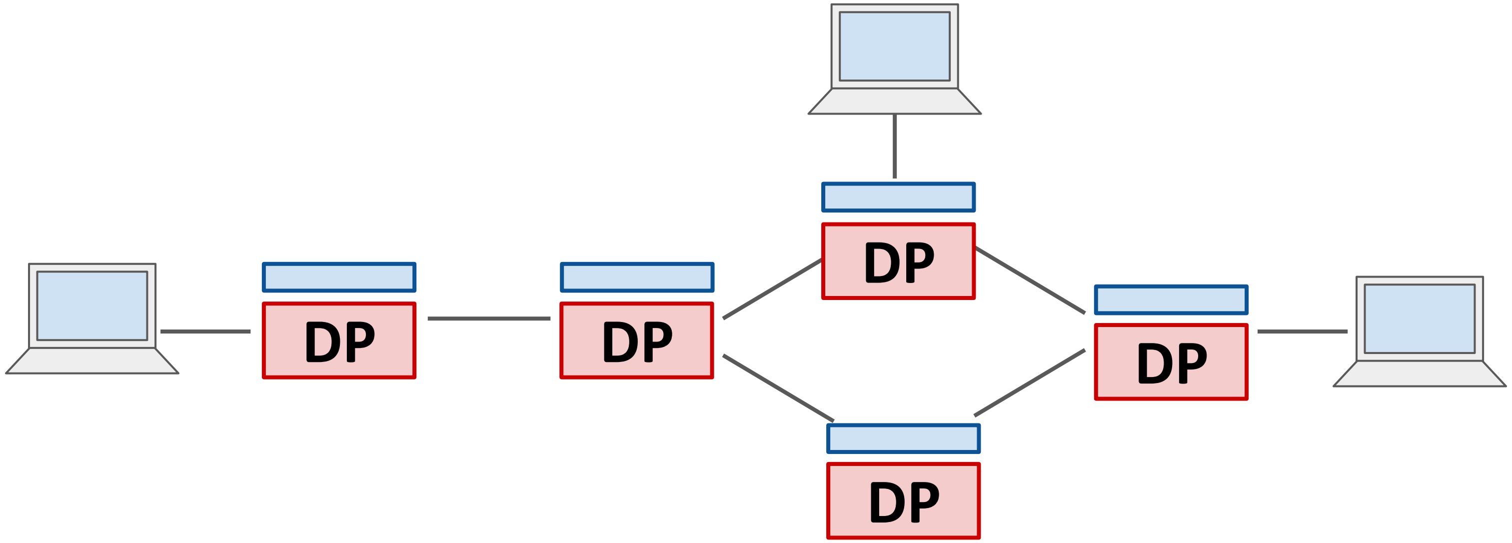
Instead of running distributed control-plane algorithms here …



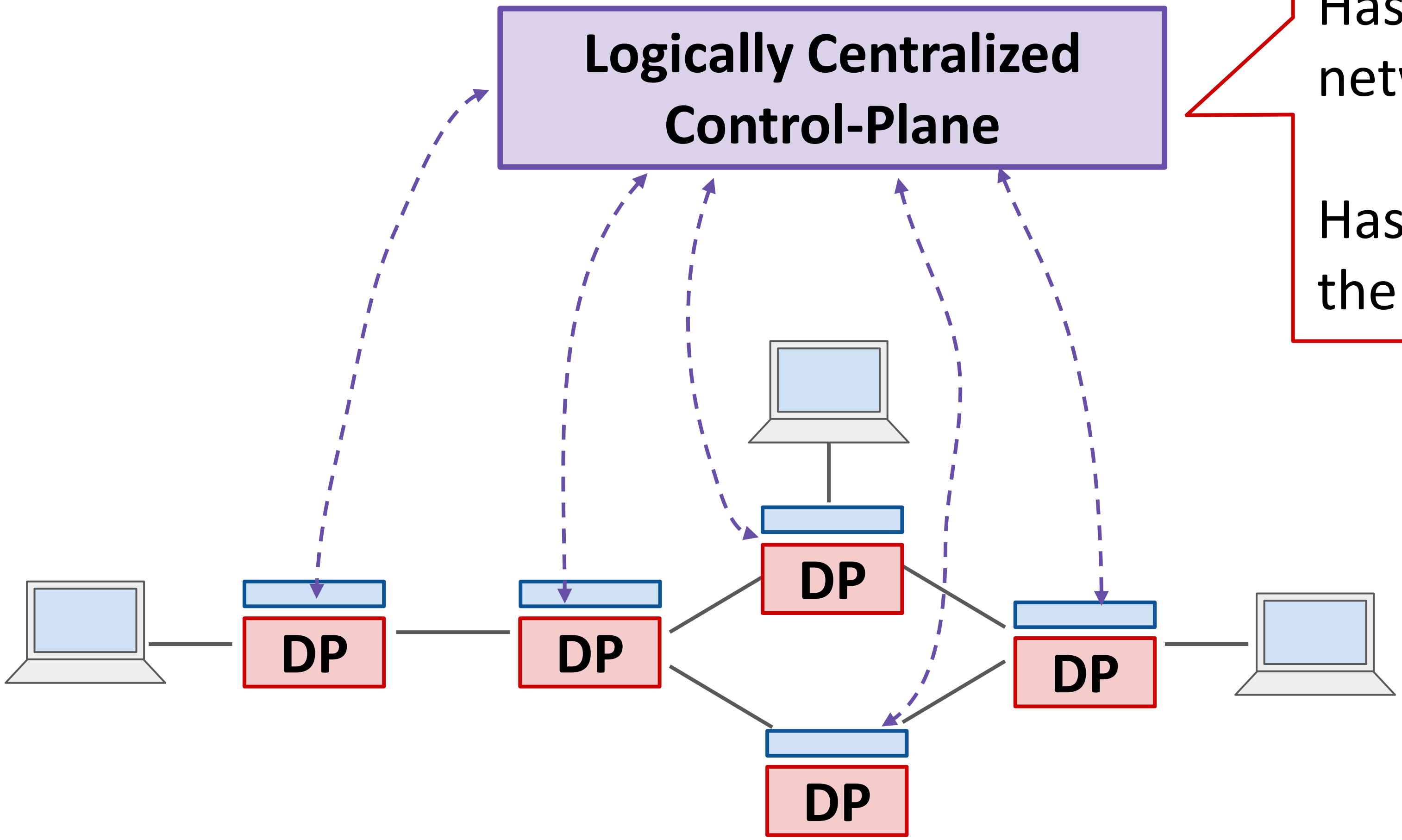
Software-Defined Networking (SDN)

Run them here, and directly tell the router data plane how to forward traffic.

Logically Centralized Control-Plane



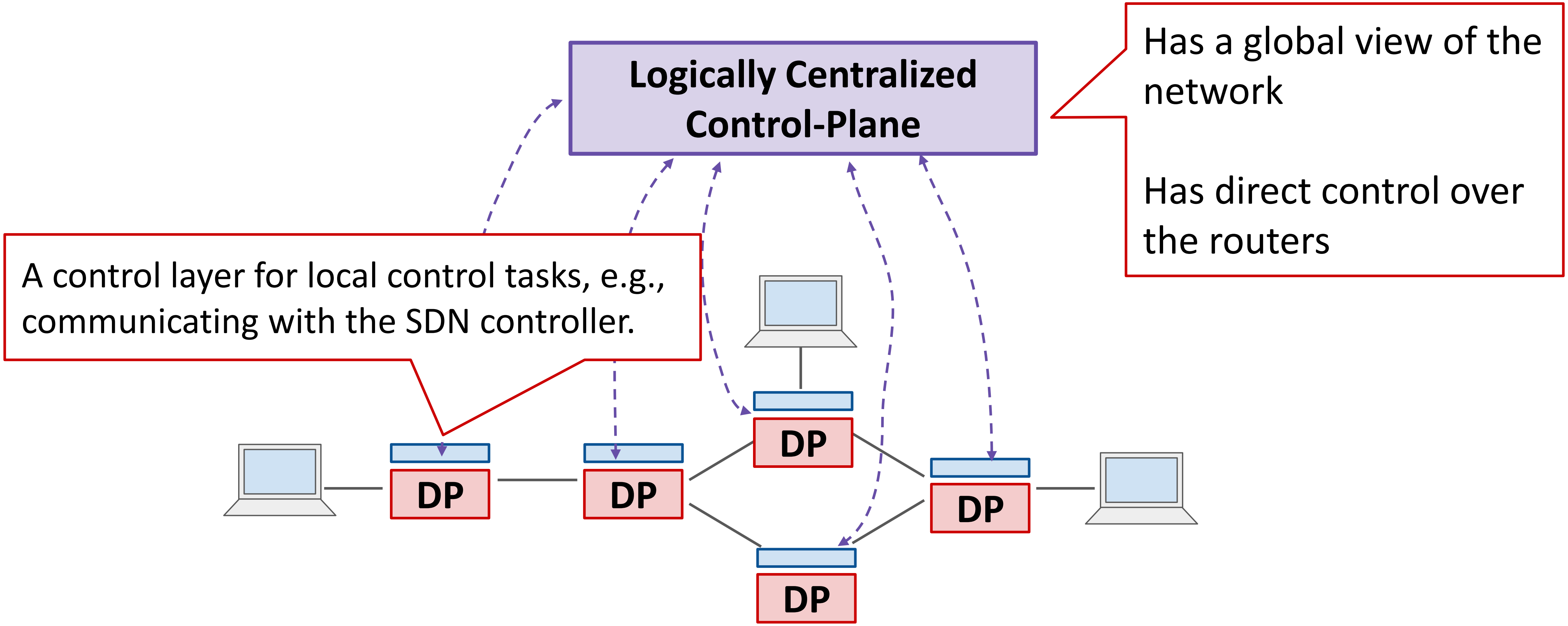
Software-Defined Networking (SDN)



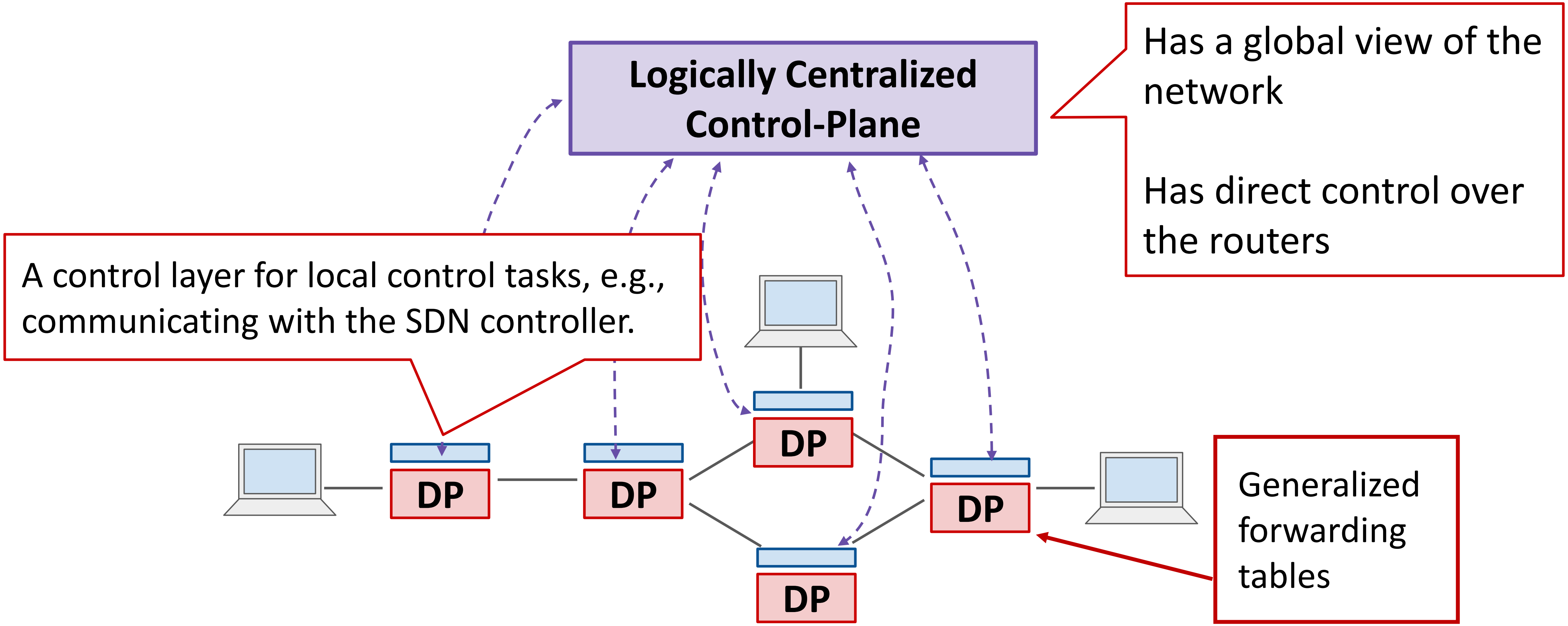
Has a global view of the network

Has direct control over the routers

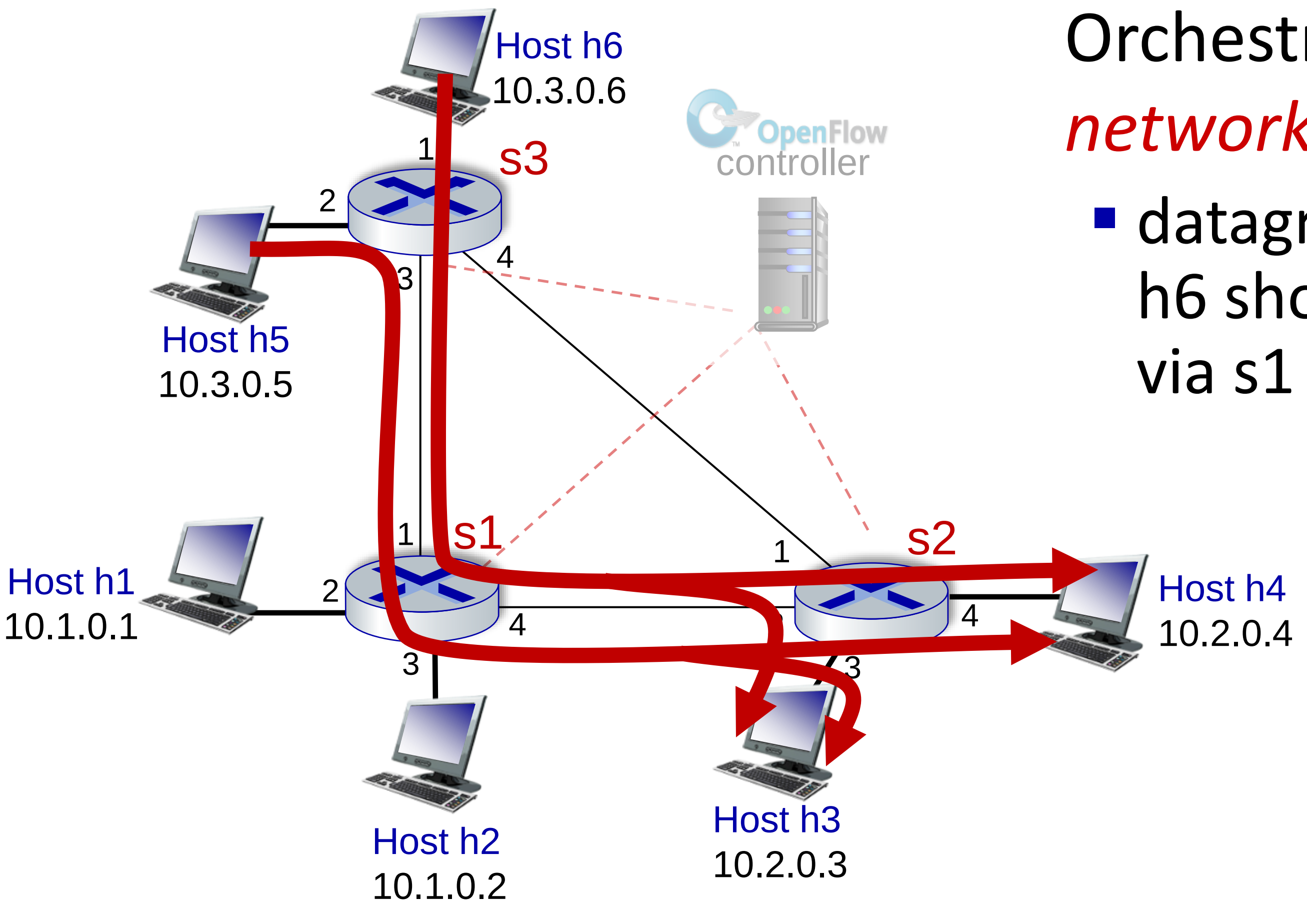
Software-Defined Networking (SDN)



Software-Defined Networking (SDN)



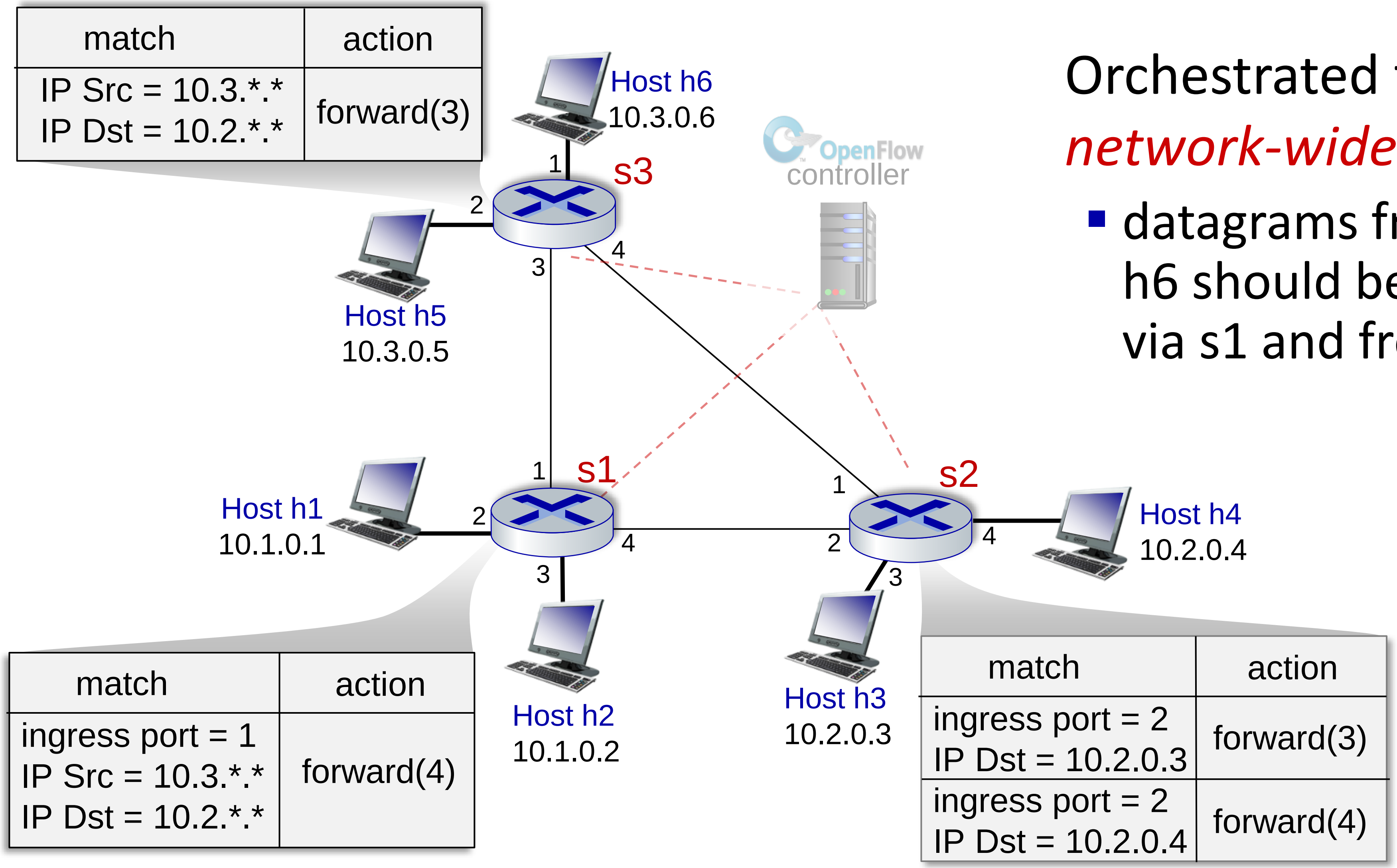
OpenFlow example



Orchestrated tables can create *network-wide* behavior, e.g.,:

- datagrams from hosts h5 and h6 should be sent to h3 or h4, via s1 and from there to s2

OpenFlow example



match	action
IP Src = 10.3.*.* IP Dst = 10.2.*.*	forward(3)

match	action
ingress port = 1 IP Src = 10.3.*.* IP Dst = 10.2.*.*	forward(4)

match	action
ingress port = 2 IP Dst = 10.2.0.3	forward(3)
ingress port = 2 IP Dst = 10.2.0.4	forward(4)

Orchestrated tables can create *network-wide* behavior, e.g.,:

- datagrams from hosts h5 and h6 should be sent to h3 or h4, via s1 and from there to s2

Generalized forwarding: summary

- "match plus action" abstraction: match bits in arriving packet header(s) in any layers, take action
  - matching over many fields (link-, network-, transport-layer)
  - local actions: drop, forward, modify, or send matched packet to controller
  - "program" *network-wide* behaviors
- simple form of "network programmability"
  - program the network with OpenFlow rules
  - *historical roots:* active networking
  - *today:* more generalized programming: P4 (see p4.org).

Remarks on Software-Defined Networking (SDN)

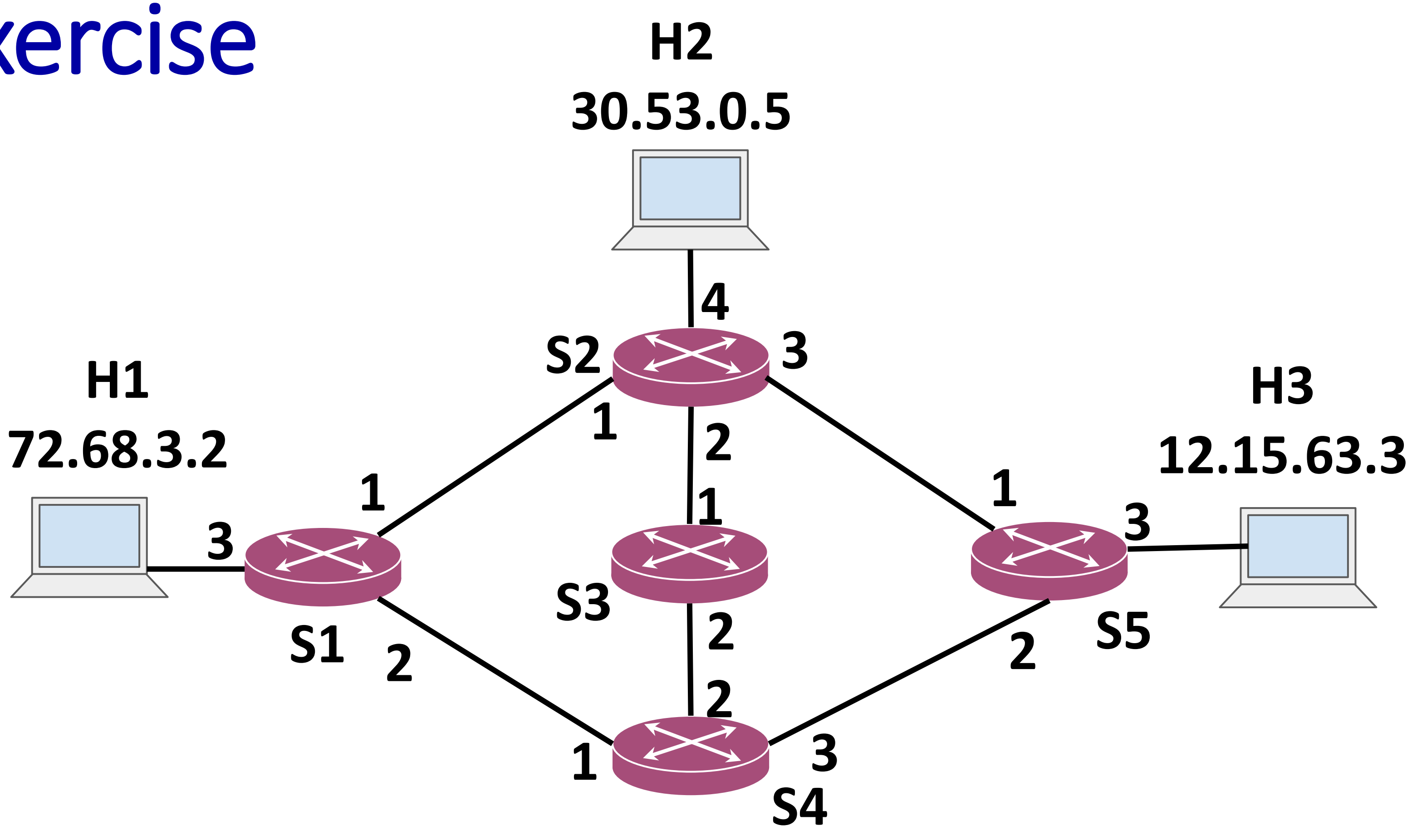
- SDN provides global visibility and direct control
- Generalized forwarding model is one of the enablers
- Why software-defined?
- Because the "software" running on the SDN controller will "define" the behavior of the network
  - As opposed to the interactions of several instances of a distributed protocol.

Make sure you know

- What generalized forwarding mean
  - In contrast with (traditional) destination-based forwarding
- What an OpenFlow flow table looks like and how it is used to process incoming packets
- How a controller can setup rules in flow tables to determine how packets are processed in the network
- E.g., given a set of paths for different traffic flows, you should be able to write down the entries in each switch flow table.

SDN/OpenFlow Exercise

- Switches should not drop any packets destined to H1, H2, or H3 unless specified by the network policy.
- Web traffic between H1 and H3 should pass S3 (suppose the webserver is running on port 80).
- H2 should not be able to send packets to H1
- Traffic between H2 and H3 should not have any links in common with traffic between H1 and H3.
- What do the OpenFlow rules on the switches look like?



Network Layer in the Internet

- The … based
- Inte … ng
  - Hi … ment
  - In
  - In … twork (SDNs)
- Inte
  - W
  - De
- ICM

We are done with the network layer!

Next Up: Link Layer and Local Area Networks (LANs)