CSCI 466: Networks

Network Layer – Data Plane

Reese Pearsall

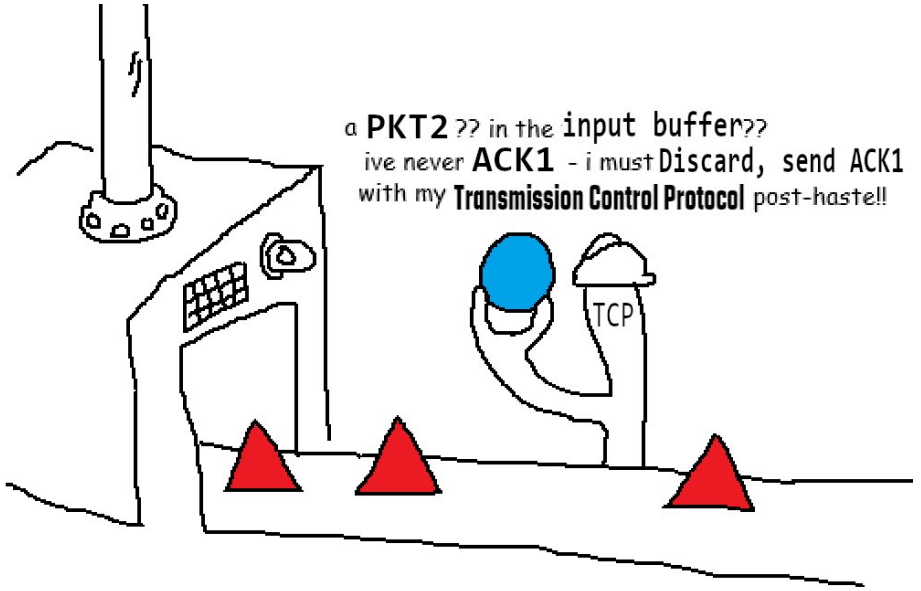
Fall 2024

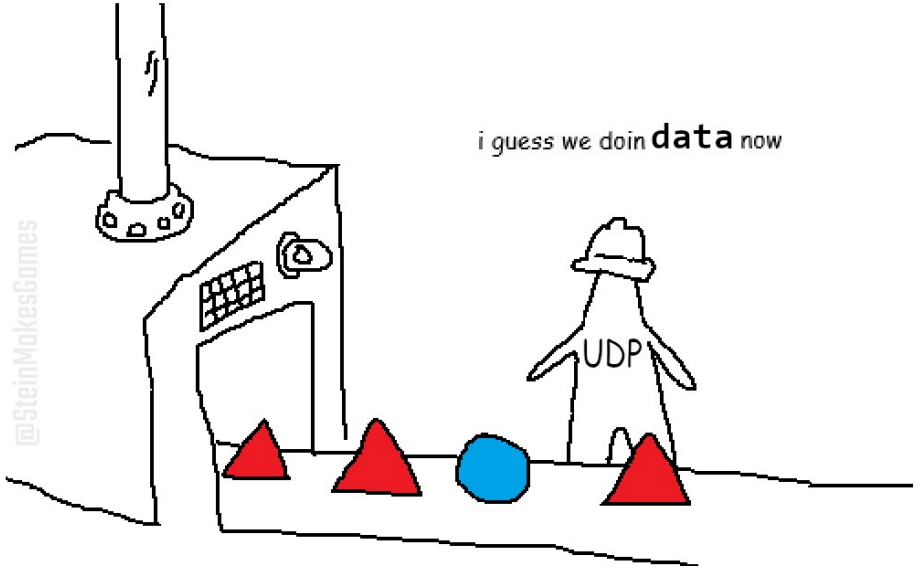
Announcements

Wireshark Lab 2 due on Wednesday

Quiz 4 also on Friday

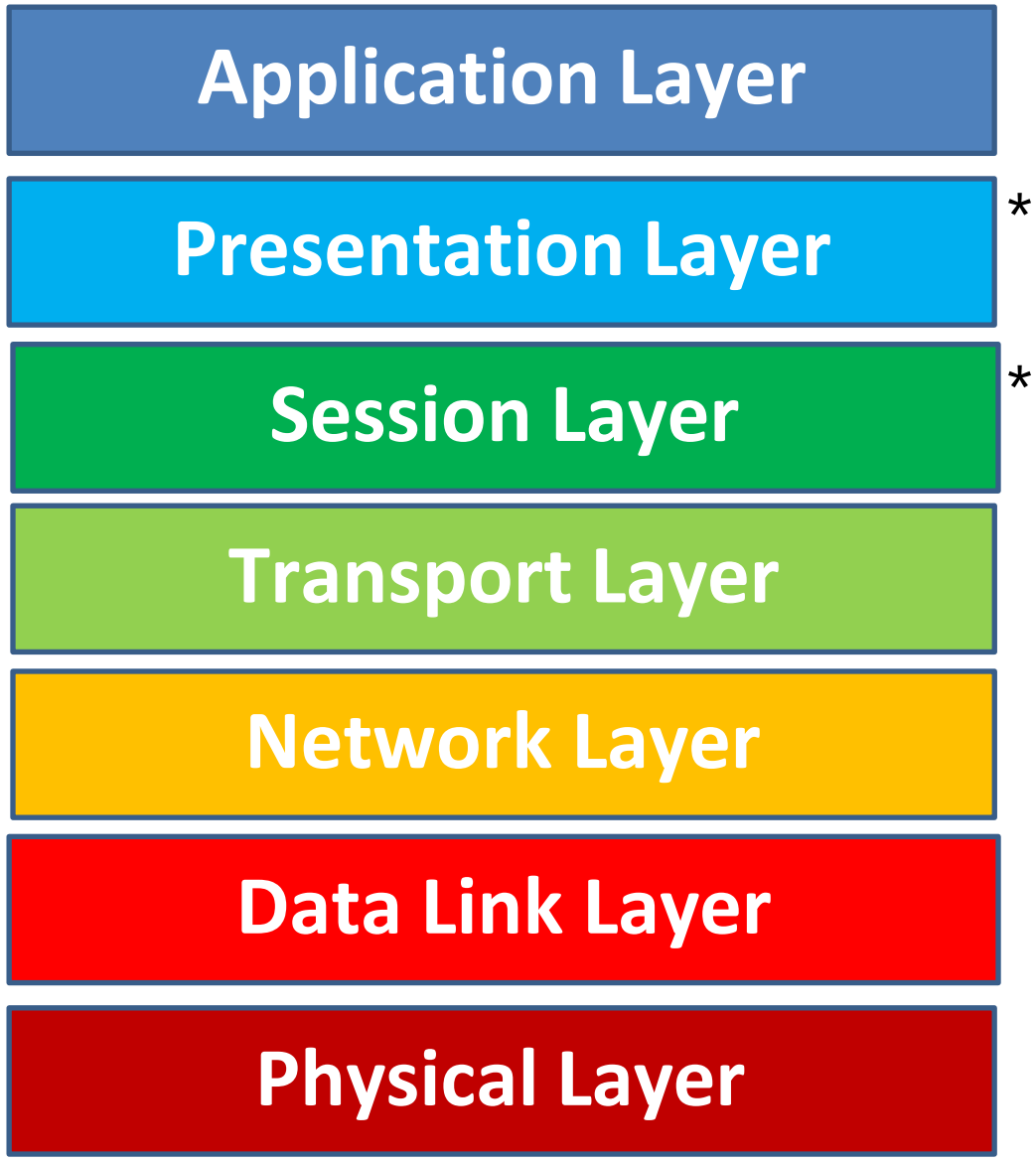
- No lecture on Friday





Announcements

OSI Model



Application Layer

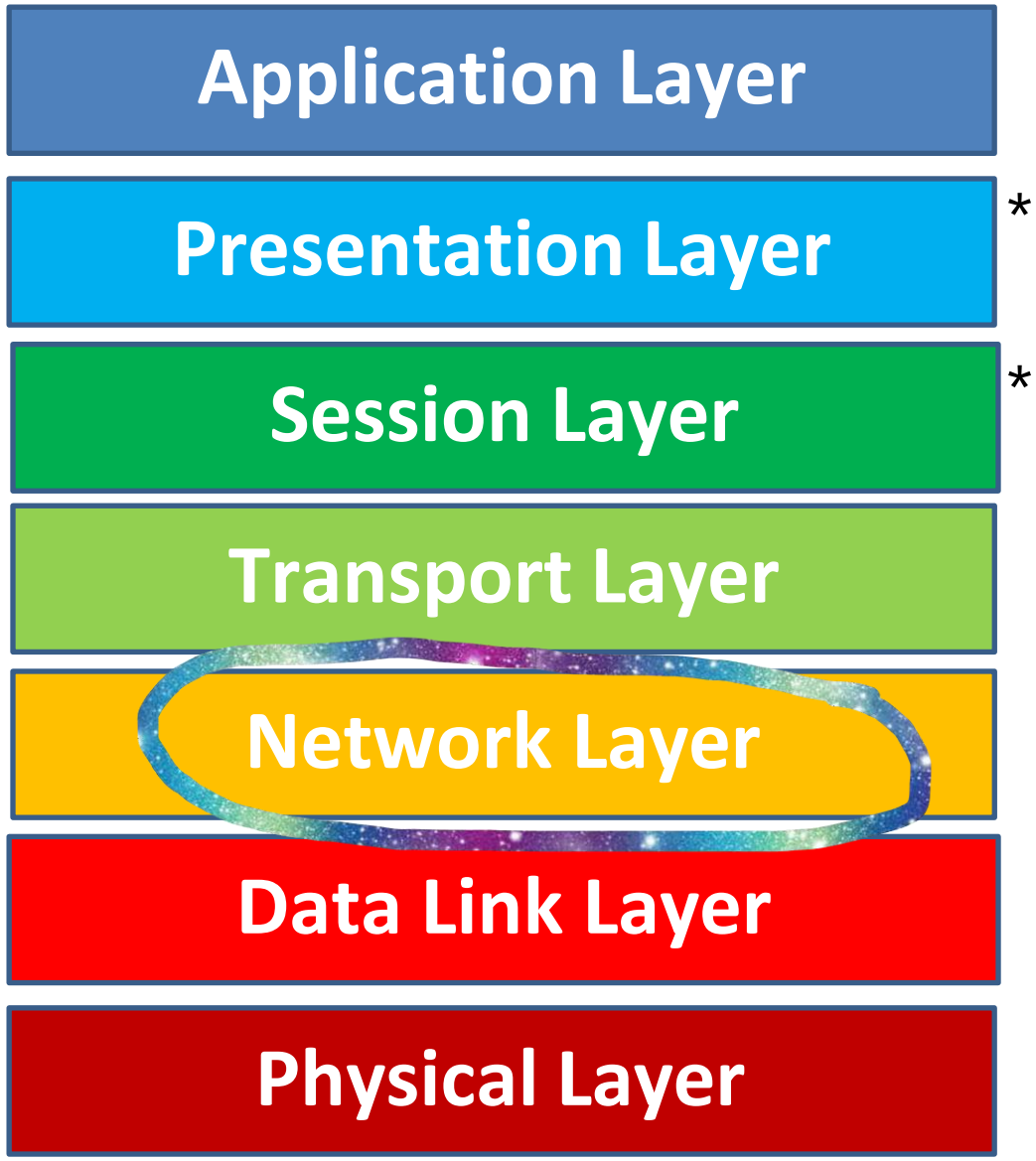
Messages from Network Applications

Physical Layer

Bits being transmitted over a medium

**In the textbook, they condense it to a 5-layer model, but 7 layers is what is most used*

OSI Model



Application Layer

Messages from Network Applications

Physical Layer

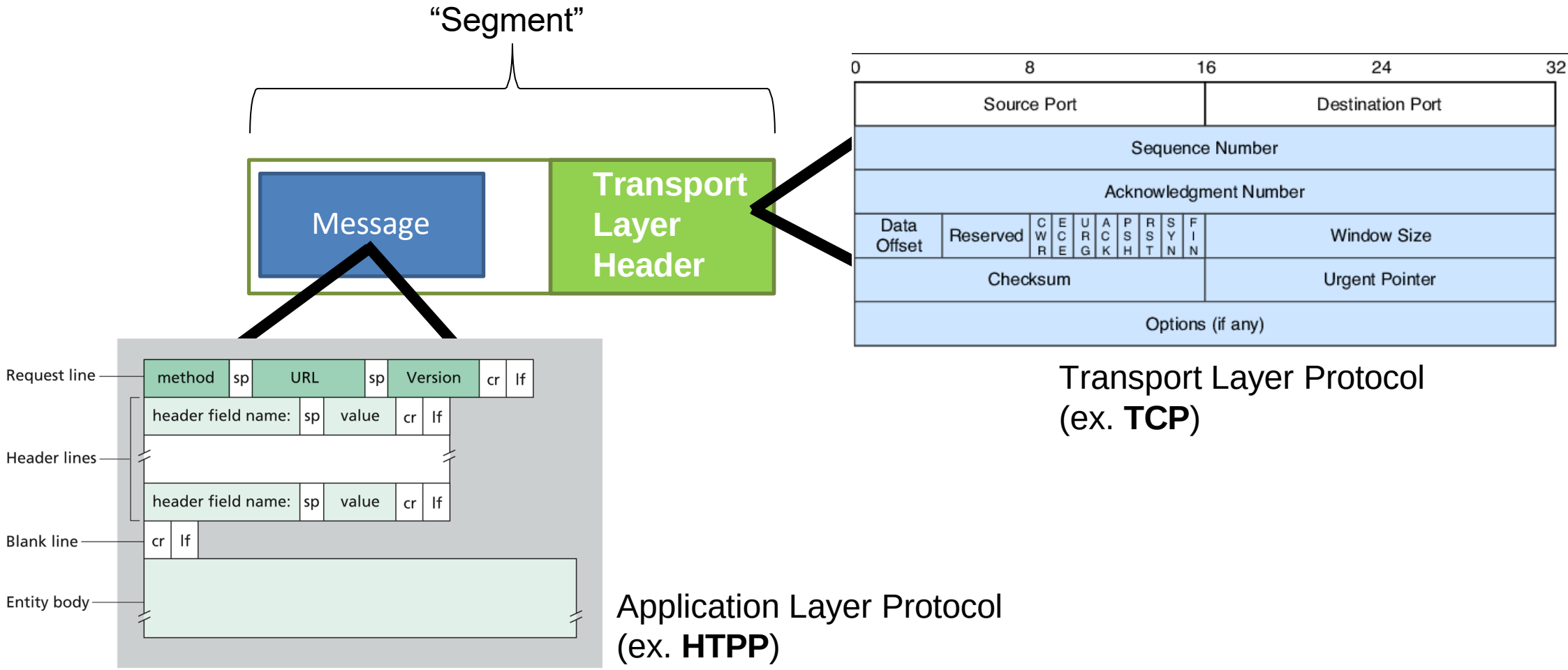
Bits being transmitted over a medium

**In the textbook, they condense it to a 5-layer model, but 7 layers is what is most used*

Application Layer

Transport Layer

Our packet of information so far…



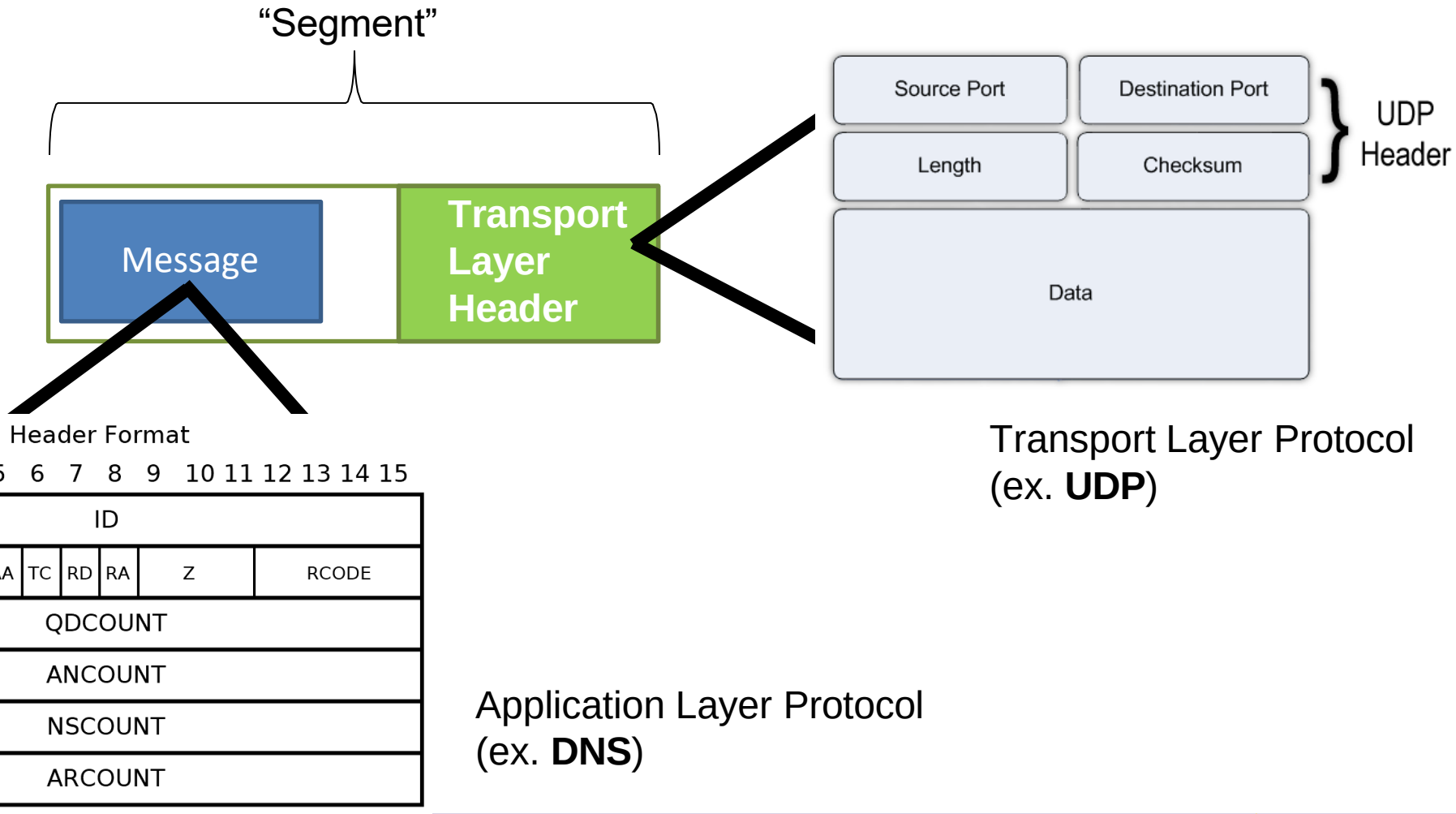
Transport Layer Protocol (ex. **TCP**)

Application Layer Protocol (ex. **HTPP**)

Application Layer

Transport Layer

Our packet of information so far…



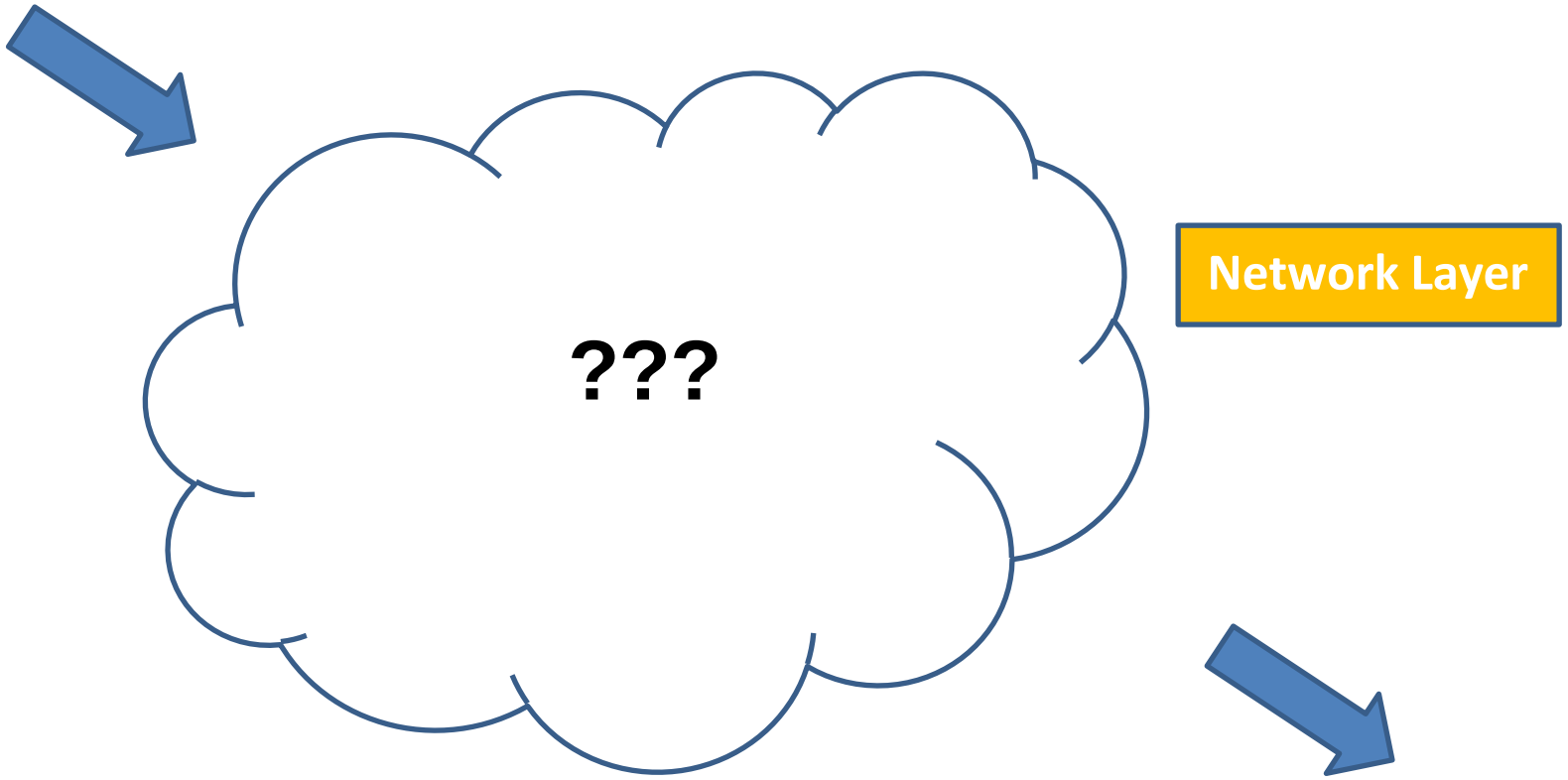
Transport Layer Protocol (ex. **UDP**)

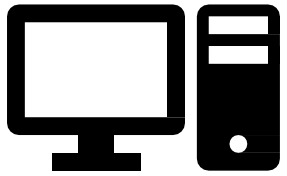
Application Layer Protocol (ex. **DNS**)

Application Layer

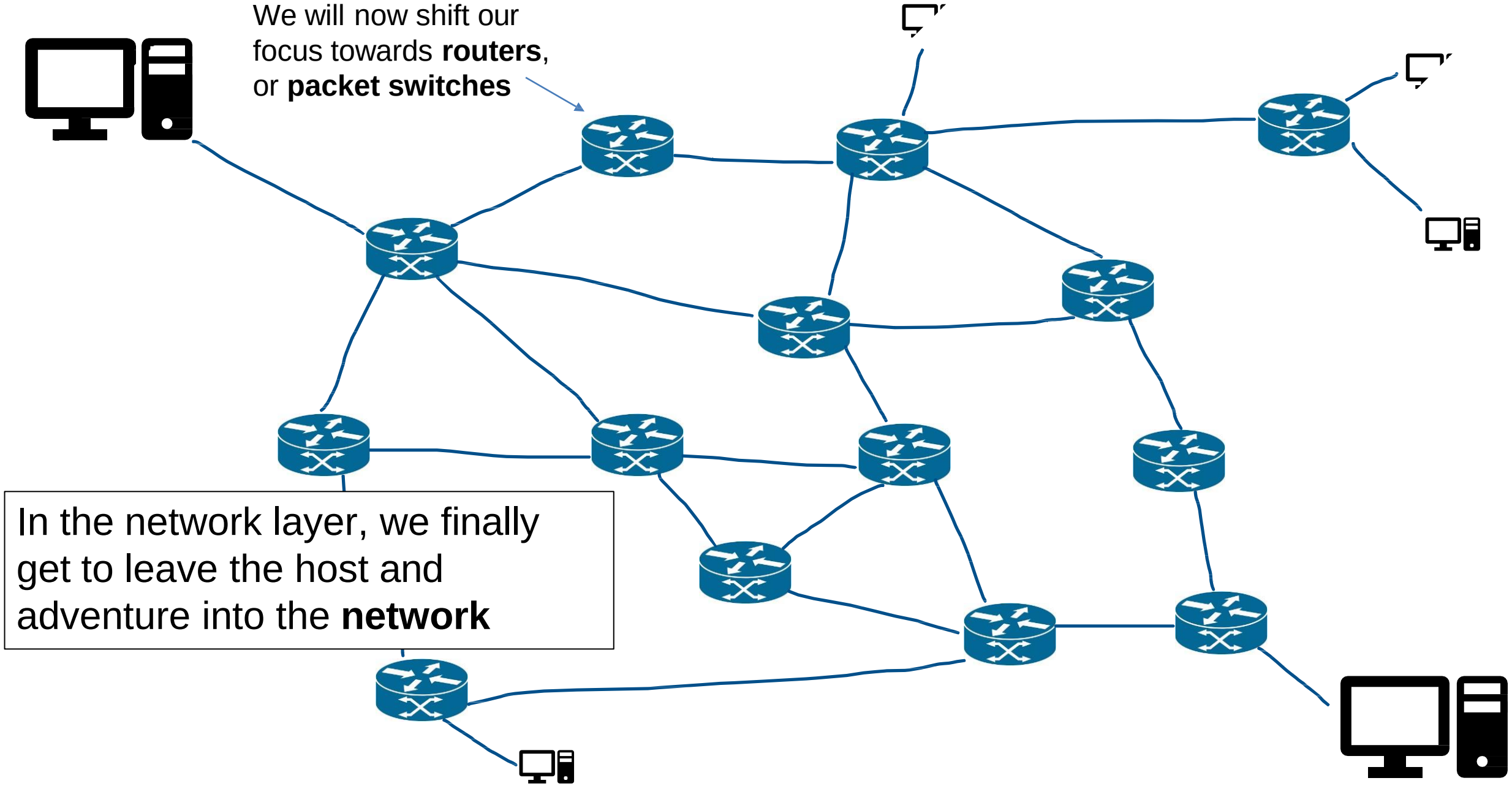
Transport Layer

We've only looked at protocols that are running on some **host**

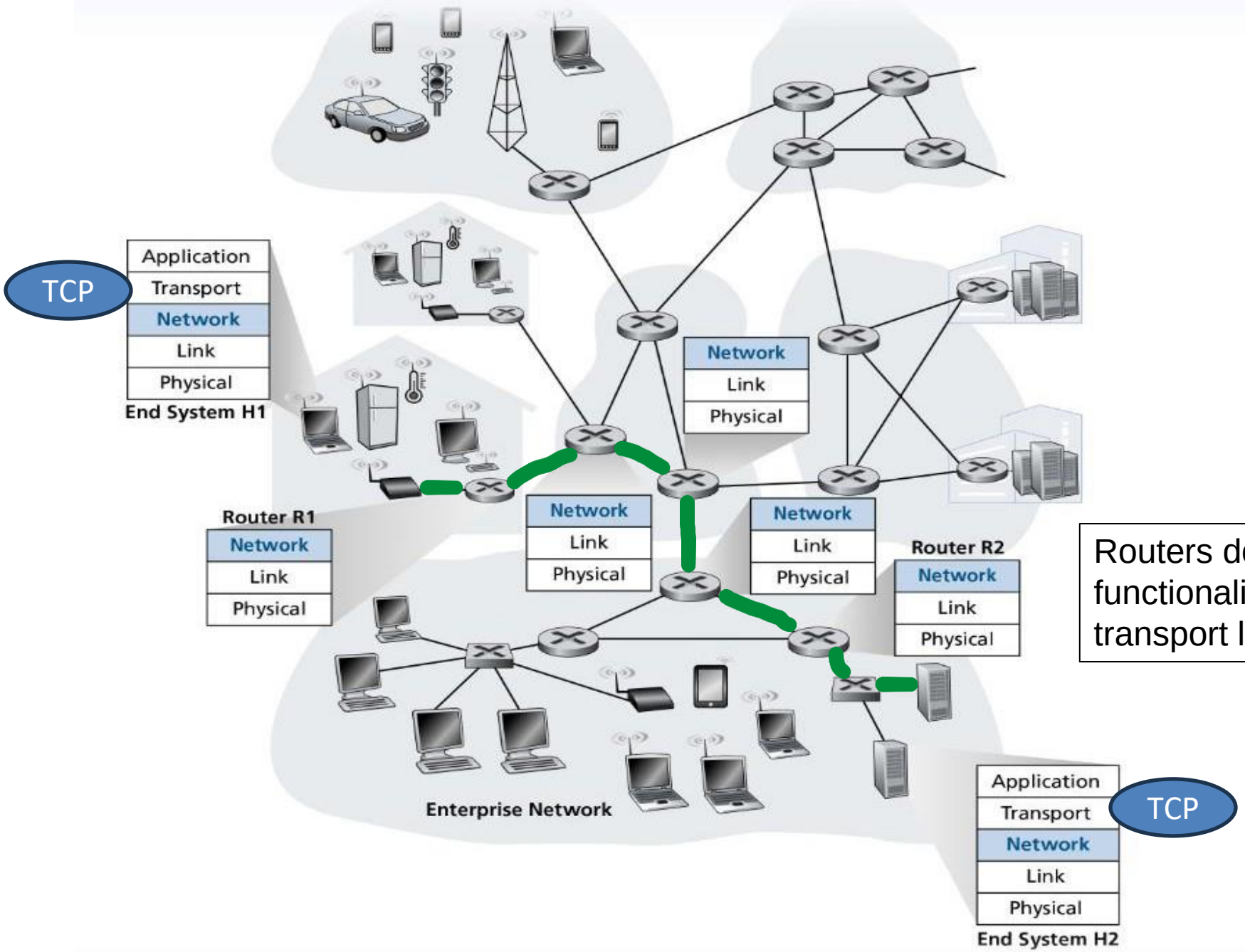




We will now shift our focus towards **routers**, or **packet switches**

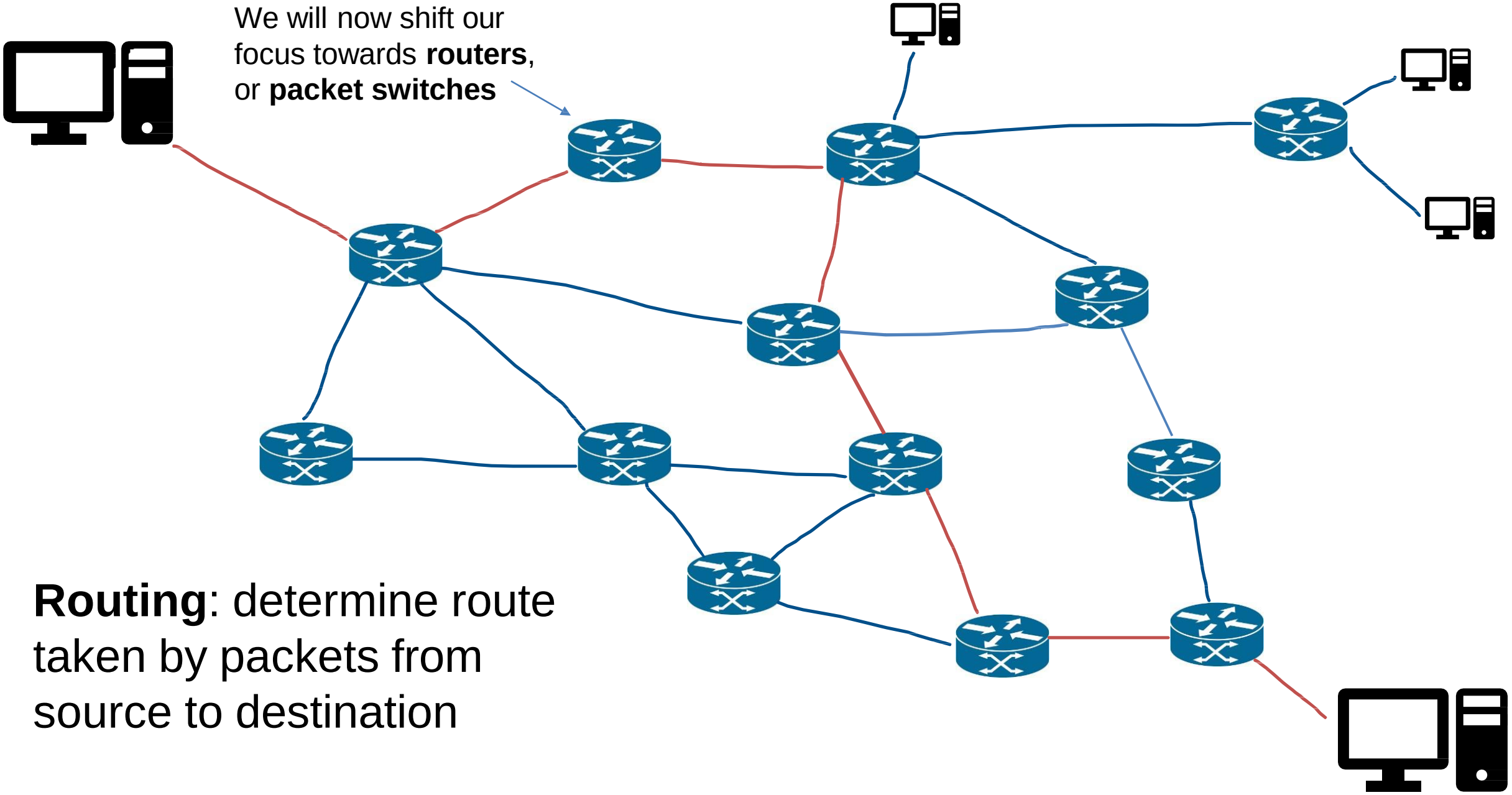


In the network layer, we finally get to leave the host and adventure into the **network**



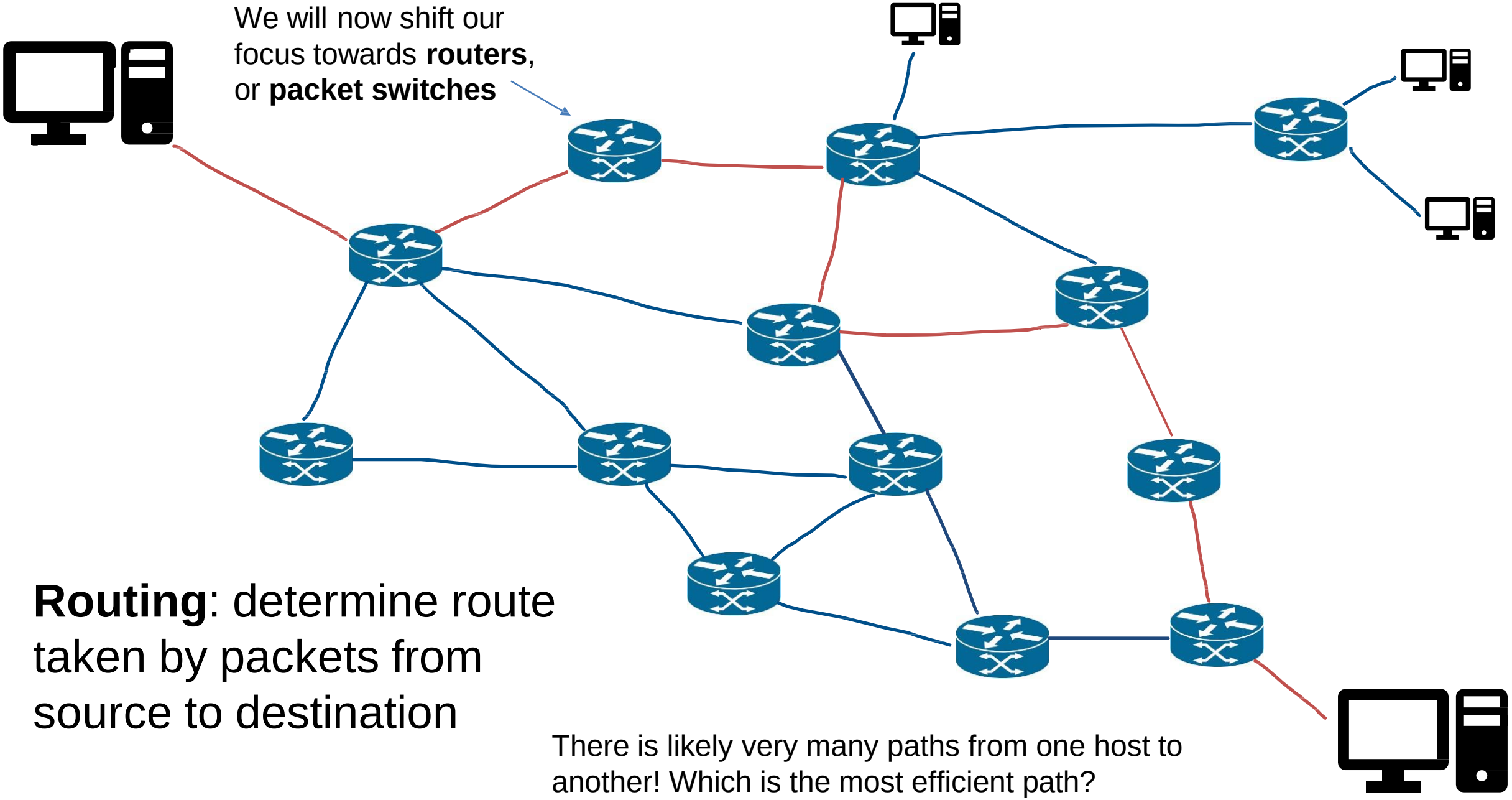
Routers do not have functionality for application & transport layer protocols

We will now shift our focus towards **routers**, or **packet switches**



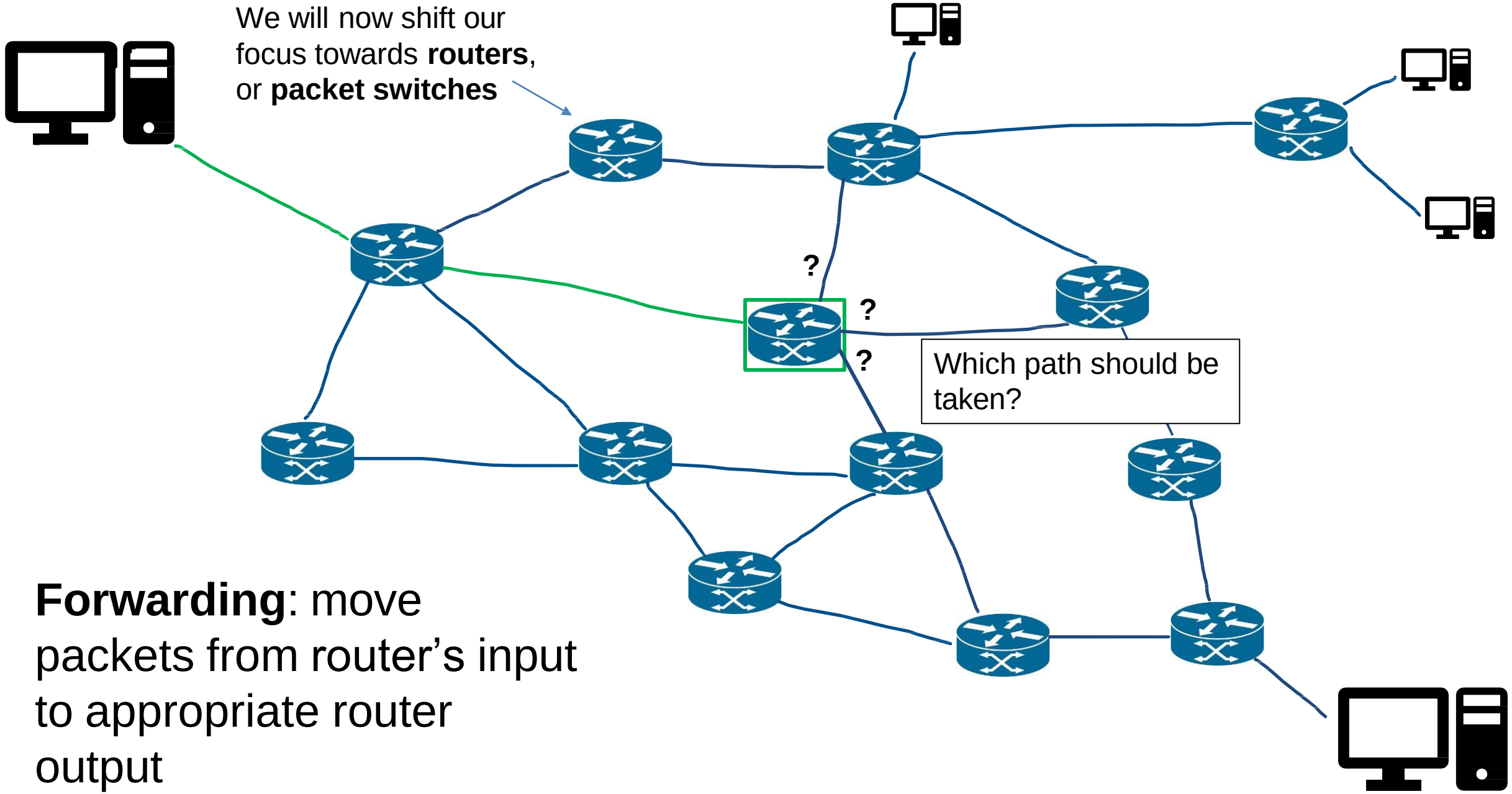
Routing: determine route taken by packets from source to destination

We will now shift our focus towards **routers**, or **packet switches**



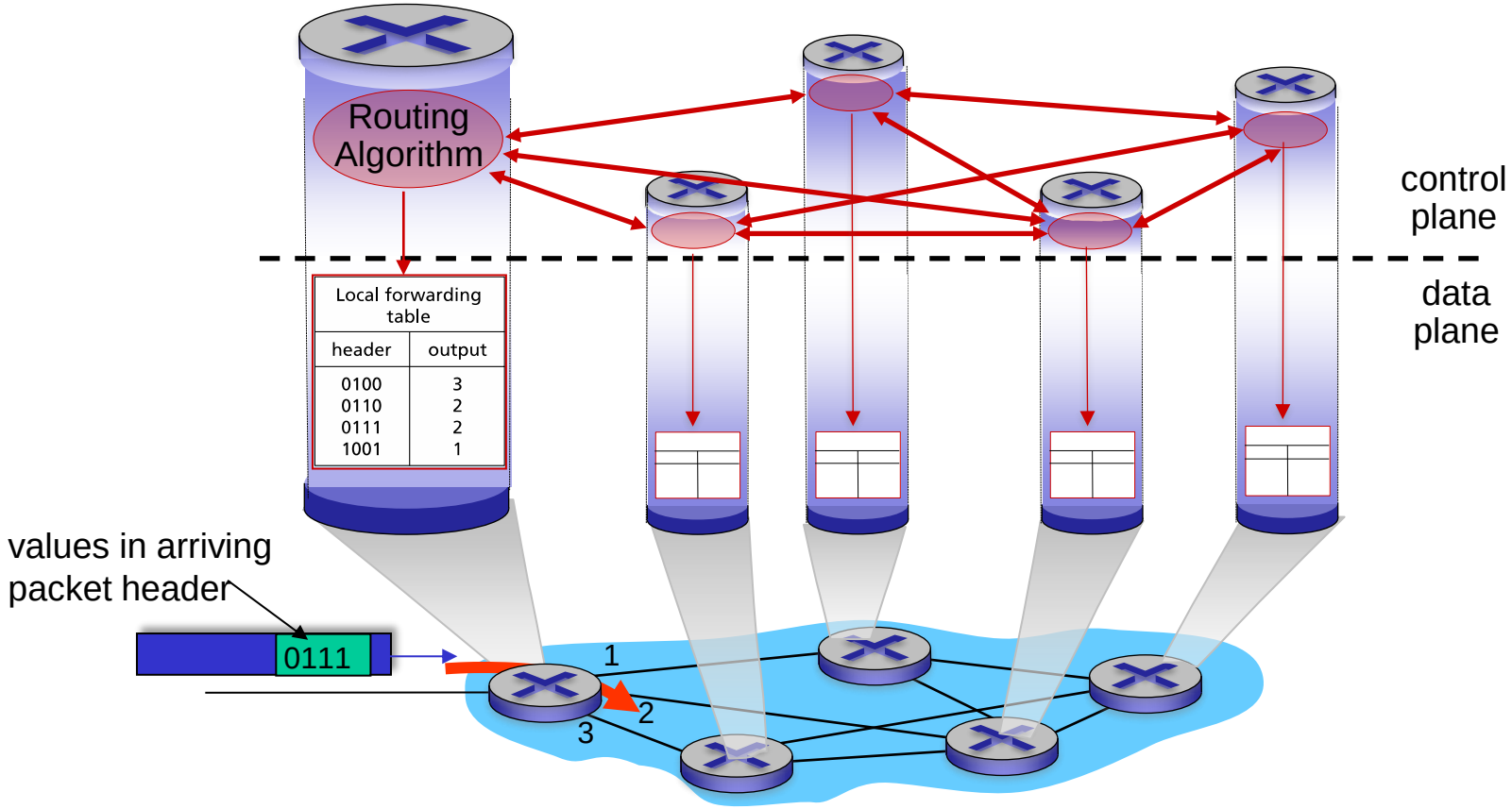
Routing: determine route taken by packets from source to destination

There is likely very many paths from one host to another! Which is the most efficient path?

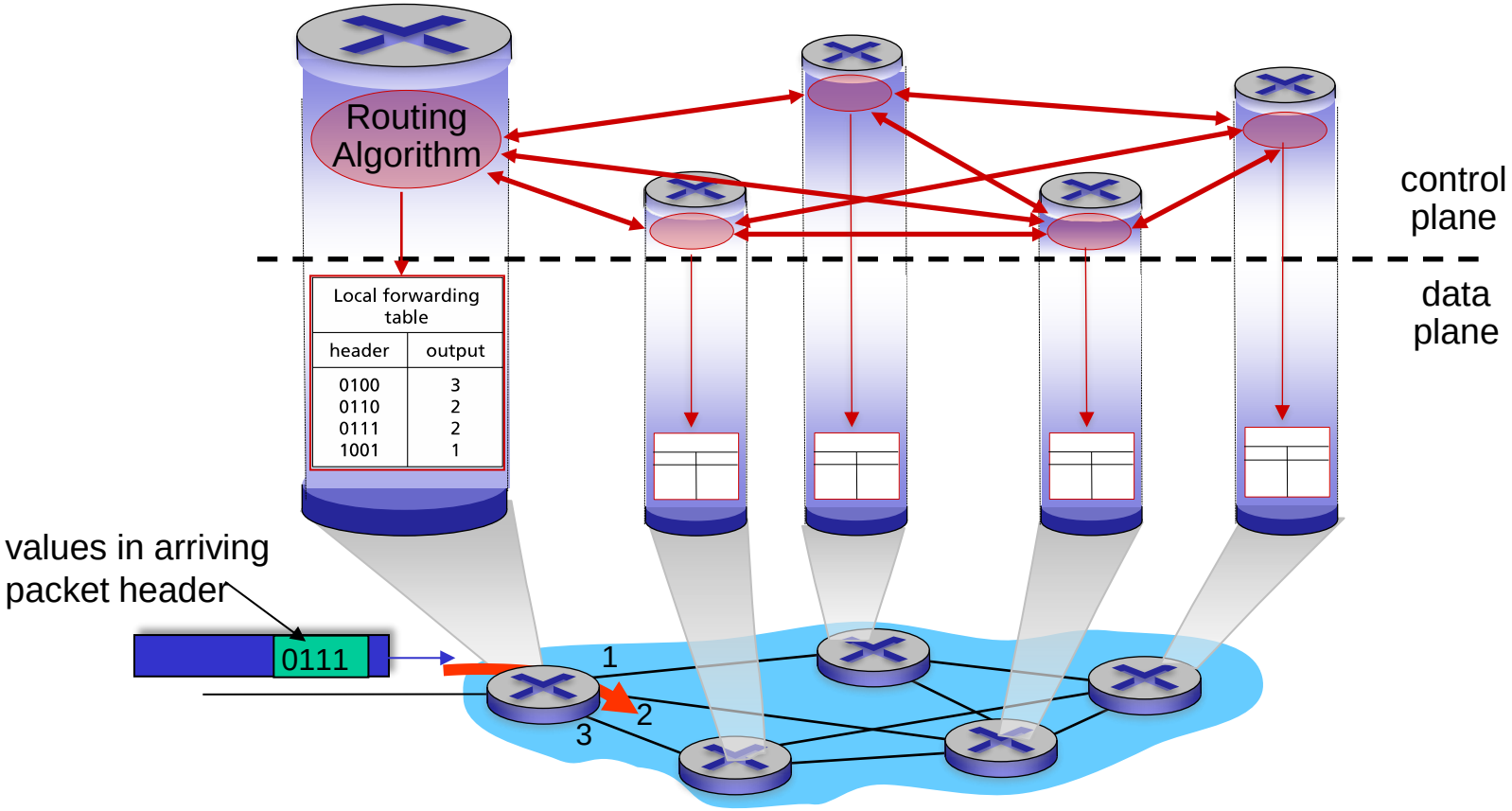


Forwarding: move packets from router's input to appropriate router output

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

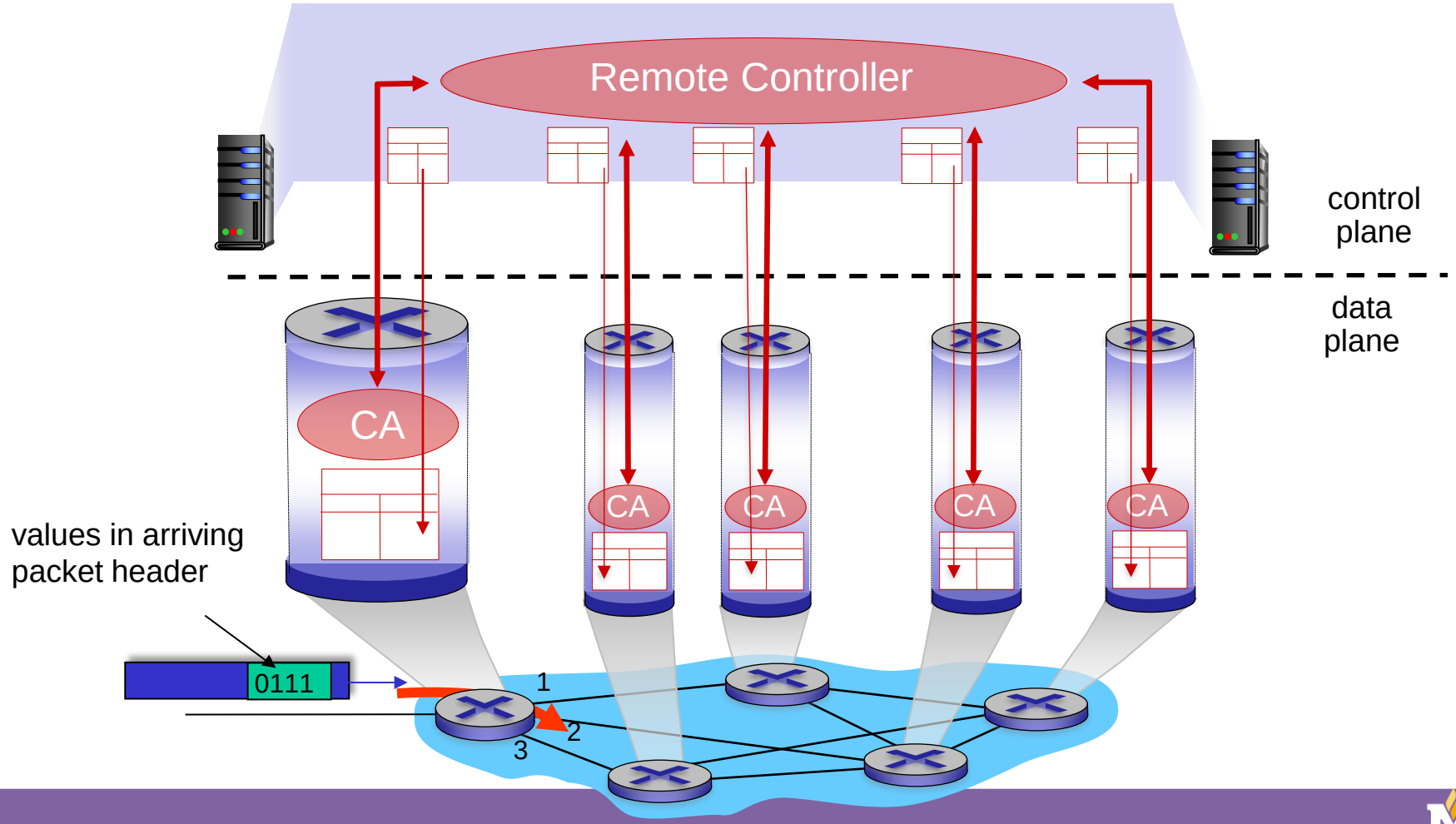


Individual routing algorithm components *in each and every router* interact in the control plane



Software-Defined Networking (SDN) control plane

Remote controller computes, installs forwarding tables in routers



Network Layer

Responsible for the delivery of data through a network

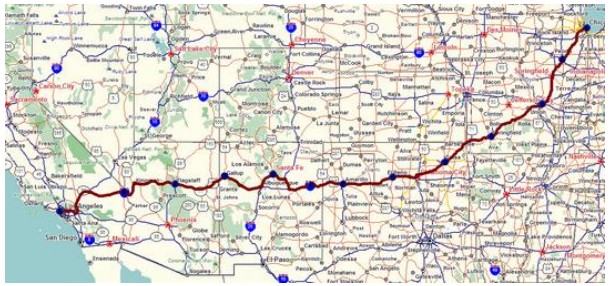
Forwarding

Data Plane

forwarding

Routing

Control Plane



routing

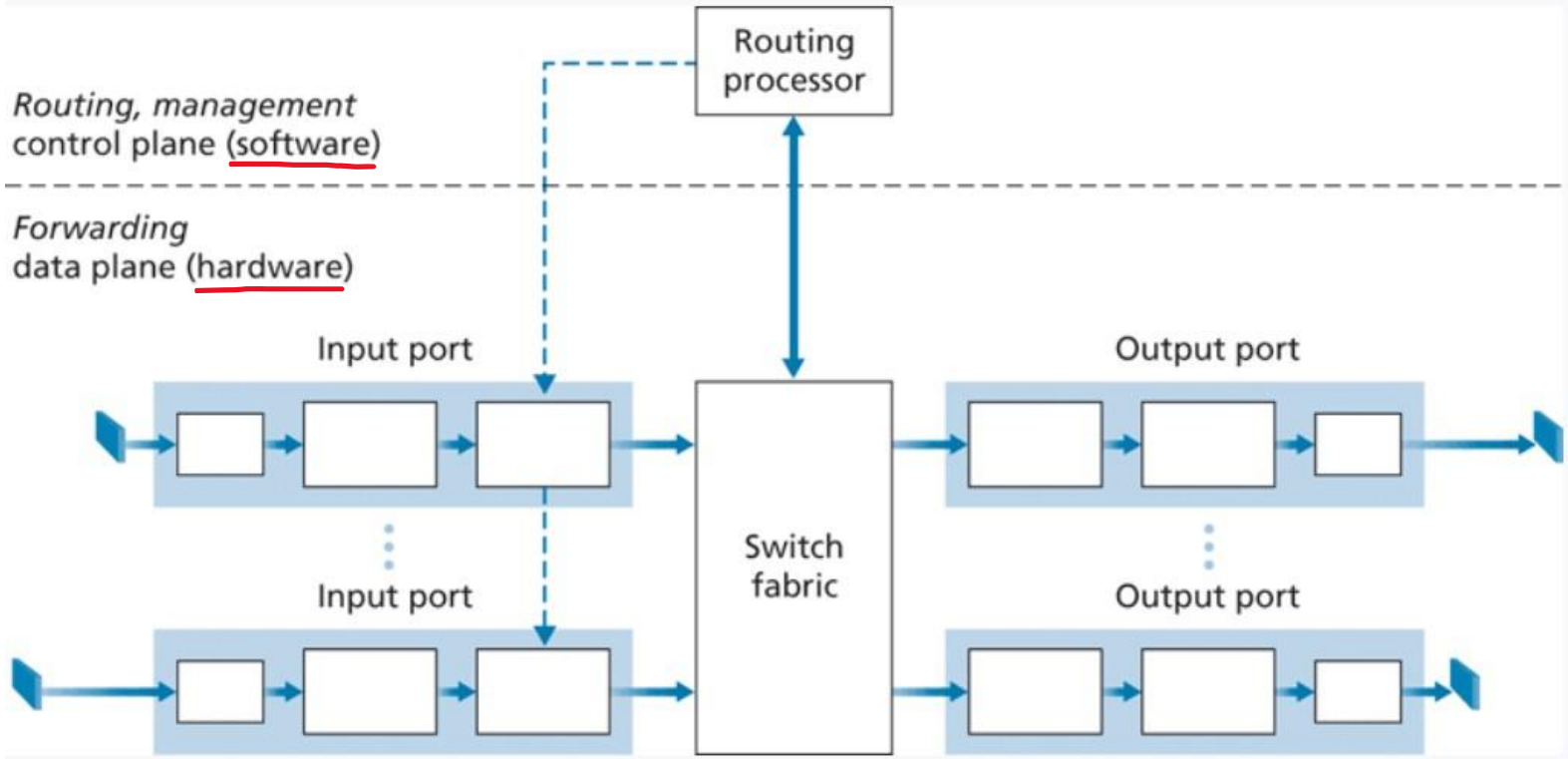
Network Layer

Internet "best effort" service model

No guarantees on:

i. successful datagram delivery to destination
ii. timing or order of delivery
iii. bandwidth available to end-end flow

Router Architecture Overview



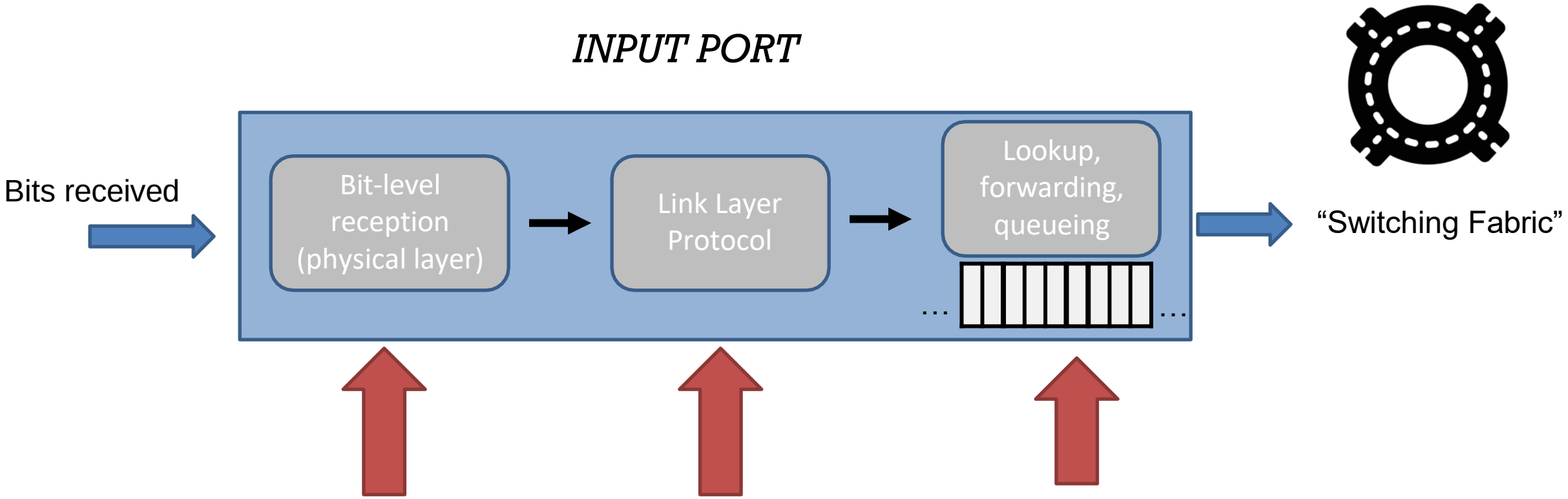
1. Destination-based forwarding

- Forwarding decisions are based on the **destination** of the packet

2. Generalized forwarding

- Forwarding decisions based on any set of header field values

Router Architecture Overview



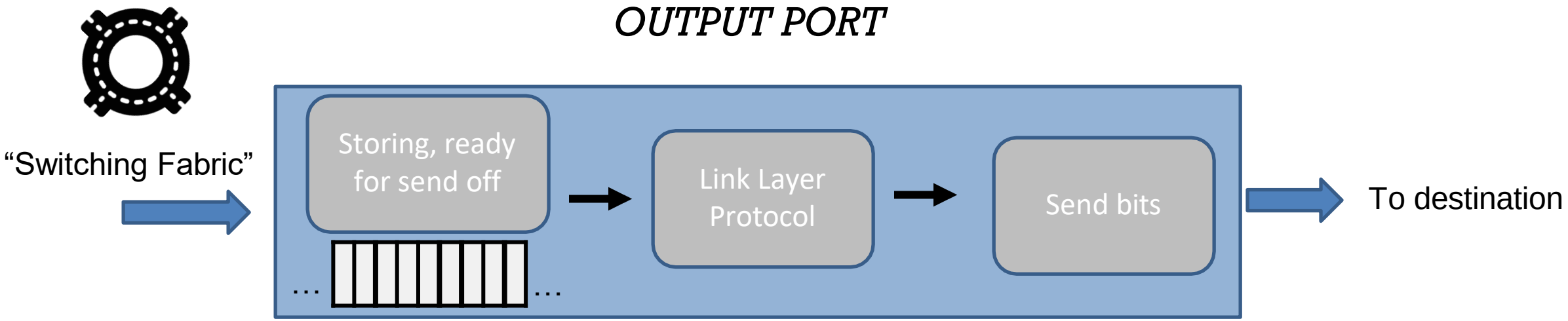
Such as over the wire or through the air

Next layer of OSI model
Ex. Ethernet

Evaluate header field values, lookup output port using forwarding table in memory

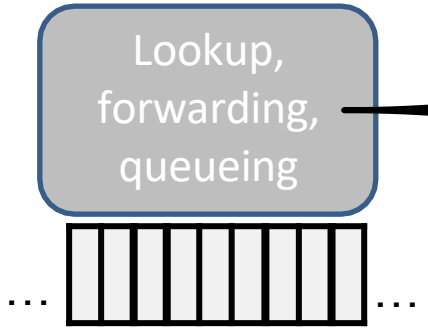
Queueing will occur if datagrams arrive faster than forwarding rate into the switching fabric

Router Architecture Overview



Forwarding Techniques

- Connection-less
- does not require startup,
- Has no idea where the final destination is



Routing Table

Address range	Interface (output link)
128.11.52.0 – 128.11.52.255	1
153.90.2.0 – 153.90.2.255	2
153.90.2.87 – 153.90.2.89	3

This routing table could get very big…
IP addresses need 32/128 bits of memory each

Forwarding Techniques

- Connection-less
- does not require startup,
- Has no idea where the final destination is

Routing Table

Address range	Interface (output link)
128.11.52.0 – 128.11.52.255	1
153.90.2.0 – 153.90.2.255	2
153.90.2.87 – 153.90.2.89	3

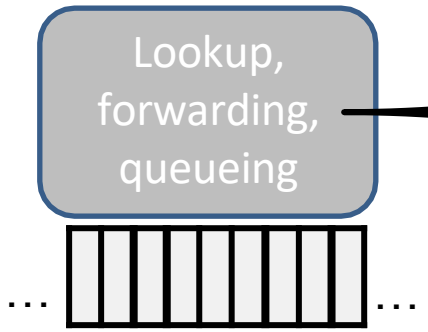
Ranges of IP Addresses

Lookup, forwarding, queueing

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

Prefix of IP addresses

Forwarding Techniques



Longest prefix matching

Address range	Interface (output link)
11001000 00010111 00010*** *********	1
11001000 00010111 00011000 *********	2
11001000 00010111 00011*** ********	3
otherwise	4

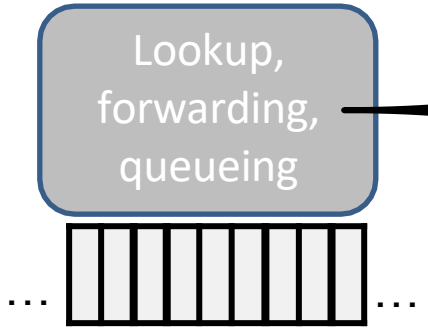
when looking for forwarding table entry for given destination address, use *longest* address prefix that matches destination address

examples:

DA: 11001000 00010111 000**10110 10100001** which interface?

DA: 11001000 00010111 000**11000 10101010** which interface?

Forwarding Techniques



Longest prefix matching

Address range	Interface (output link)
11001000 00010111 00010*** *********	1
11001000 00010111 00011000 *********	2
11001000 00010111 00011*** ********	3
otherwise	4

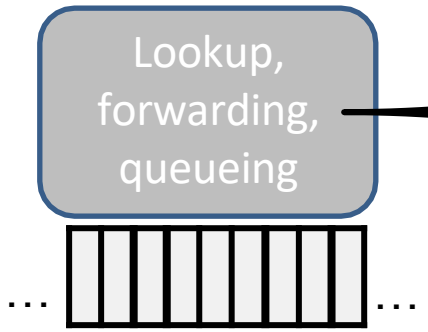
when looking for forwarding table entry for given destination address, use *longest* address prefix that matches destination address

examples:

DA: 11001000 00010111 00010110 10100001 which interface?

DA: 11001000 00010111 00011000 10101010 which interface?

Forwarding Techniques



Longest prefix matching

Address range	Interface (output link)
11001000 00010111 00010*** *********	1
11001000 00010111 00011000 *********	2
11001000 00010111 00011*** ********	3
otherwise	4

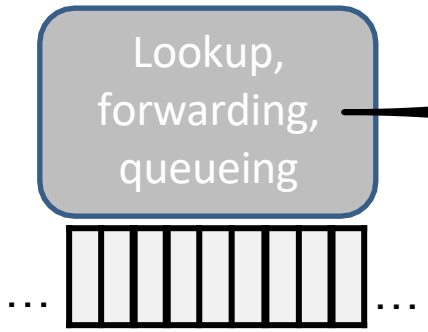
when looking for forwarding table entry for given destination address, use *longest* address prefix that matches destination address

examples:

DA: 11001000 00010111 000**10110 10100001** which interface?

DA: 11001000 00010111 000**11000 10101010** which interface?

Forwarding Techniques



Longest prefix matching

Address range	Interface (output link)
11001000 00010111 00010*** *********	1
11001000 00010111 00011000 *********	2
11001000 00010111 00011*** ********	3
otherwise	4

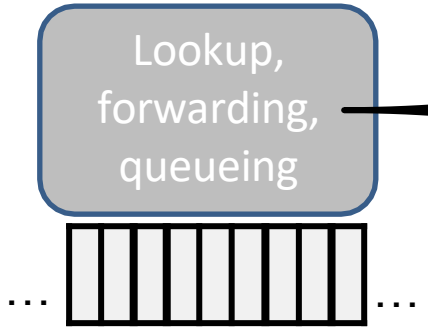
when looking for forwarding table entry for given destination address, use *longest* address prefix that matches destination address

examples:

DA: 11001000 00010111 00010110 10100001 which interface?

DA: 11001000 00010111 00011000 10101010 which interface?

Forwarding Techniques



Longest prefix matching

Address range	Interface (output link)
11001000 00010111 00010*** *********	1
11001000 00010111 00011000 *********	2
11001000 00010111 00011*** ********	3
otherwise	4

when looking for forwarding table entry for given destination address, use *longest* address prefix that matches destination address

These lookups need to happen in nanoseconds for our network to function

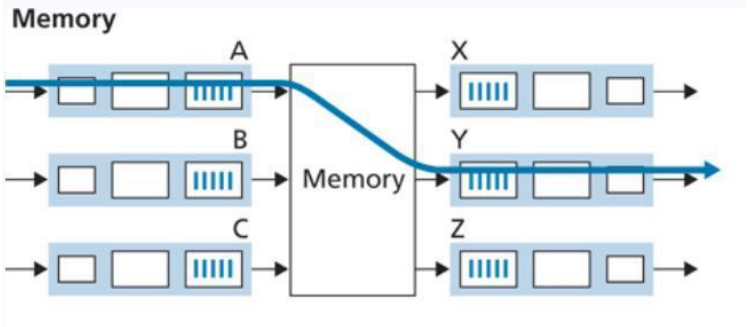
Ternary content addressable memories (TCAMs) are used in modern routers to do **LPM**

Cisco routers can carry millions of TCAM entries in their routers

Fabric Switch and Switching

Switching fabric: Mechanism that forwards data from an input port to output port

Switching via memory:



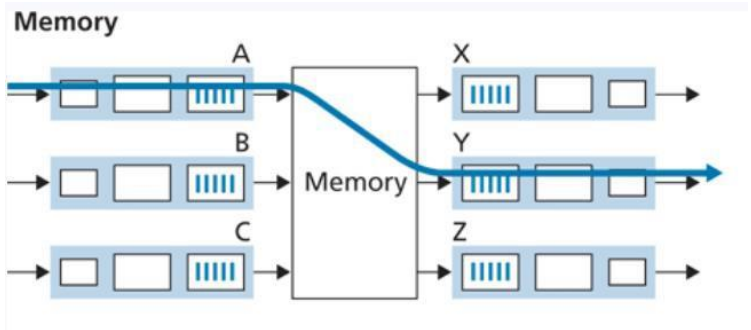
Handled by a CPU (routing processor)

Cannot forward data in parallel

Fabric Switch and Switching

Switching fabric: Mechanism that forwards data from an input port to output port

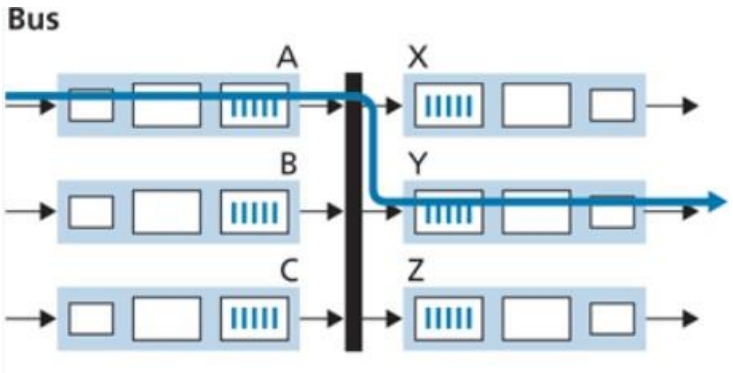
Switching via memory:



Handled by a CPU (routing processor)

Cannot forward data in parallel

Switching via bus:



Datagrams are prepended with a header

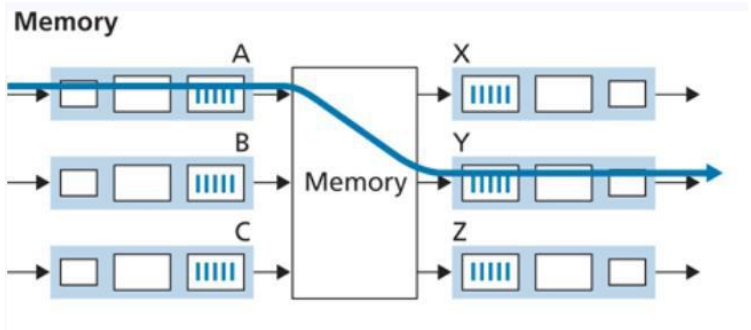
A “bus” transports input port datagrams to output ports

“keep going around the roundabout until you find your port”

Fabric Switch and Switching

Switching fabric: Mechanism that forwards data from an input port to output port

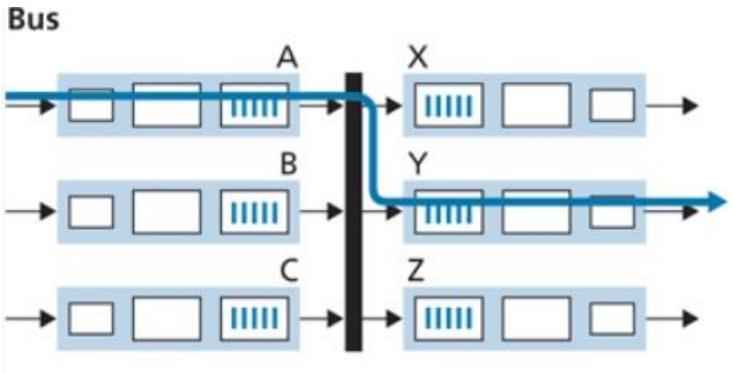
Switching via memory:



Handled by a CPU (routing processor)

Cannot forward data in parallel

Switching via bus:



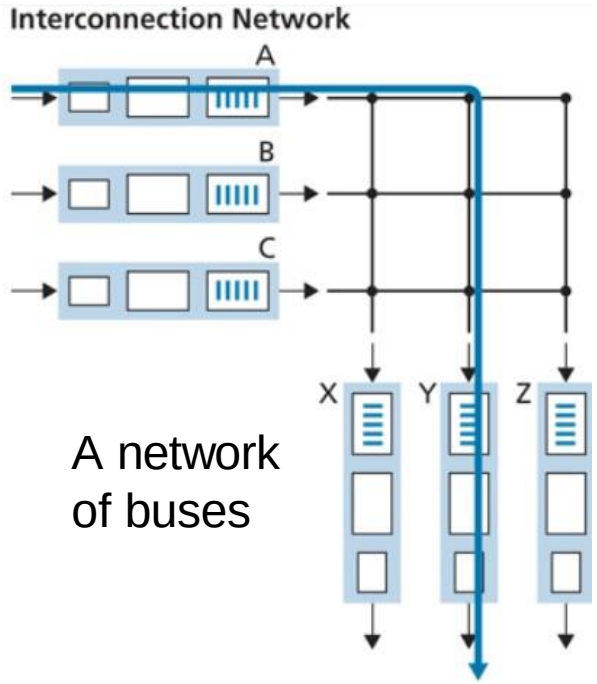
Datagrams are prepended with a header

A “bus” transports input port datagrams to output ports

“keep going around the roundabout until you find your port”

One shared bus, but can only fit one datagram

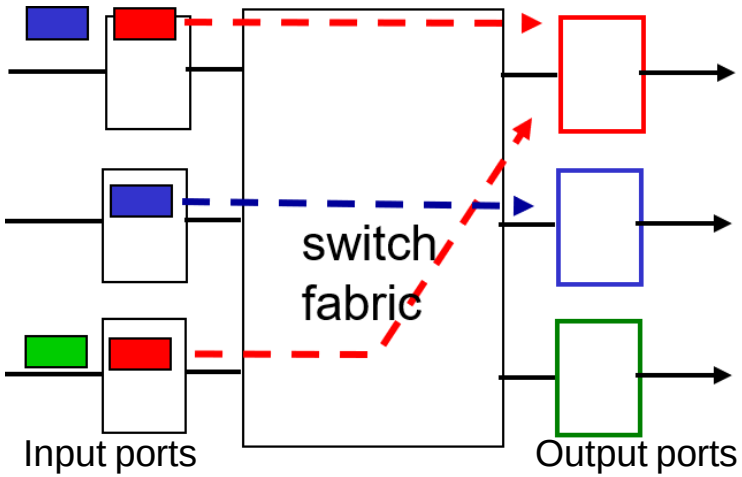
Switching via network:



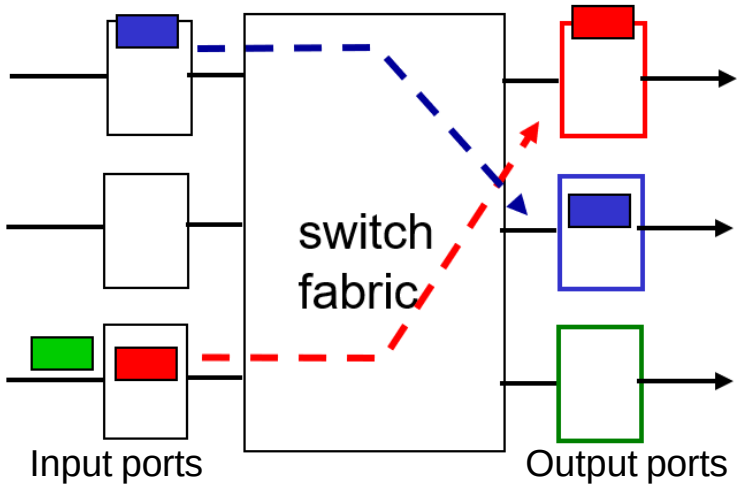
A network of buses

Input Queueing

fabric slower than input ports combined ➔*queueing may occur at input queues*



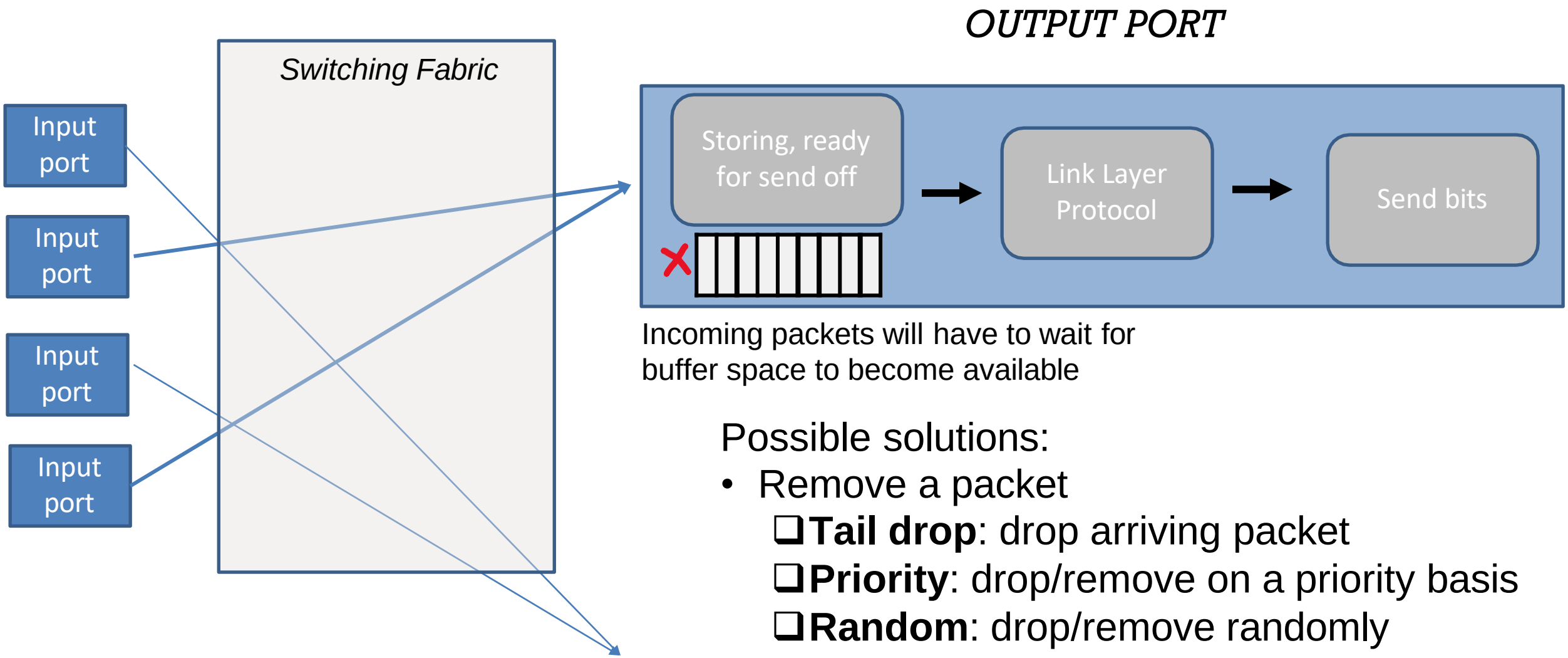
output port contention:
only one red datagram can be transferred.
lower red packet is blocked



one packet time later:
green packet experiences HOL blocking

Head-of-the-Line (HOL) blocking: queued datagram at front of queue prevents others in queue from moving forward

Output Queueing



Incoming packets will have to wait for buffer space to become available

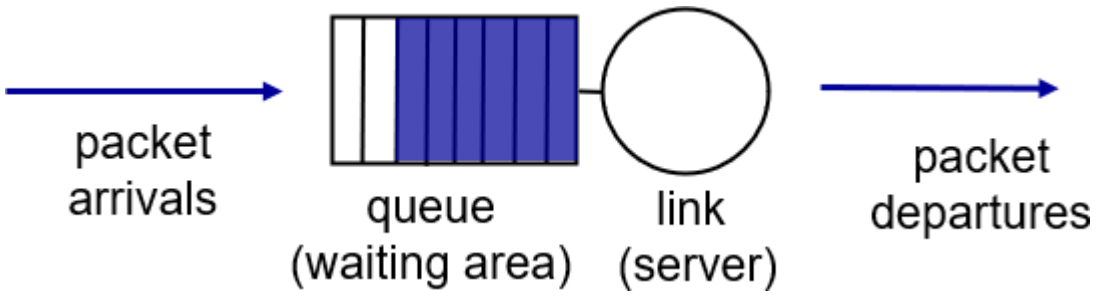
Possible solutions:

- Remove a packet
  - ❑ **Tail drop**: drop arriving packet
  - ❑ **Priority**: drop/remove on a priority basis
  - ❑ **Random**: drop/remove randomly

Packet Scheduling

Scheduling is used to determine the next packet to send on the link

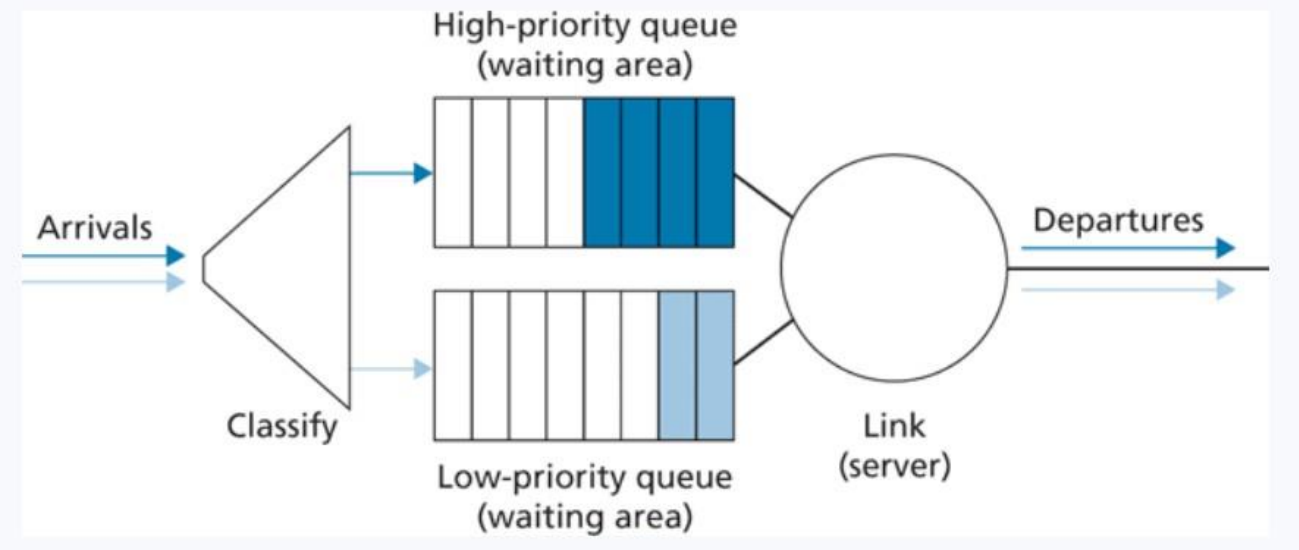
FIFO (first in first out): Send in order of arrival to queue

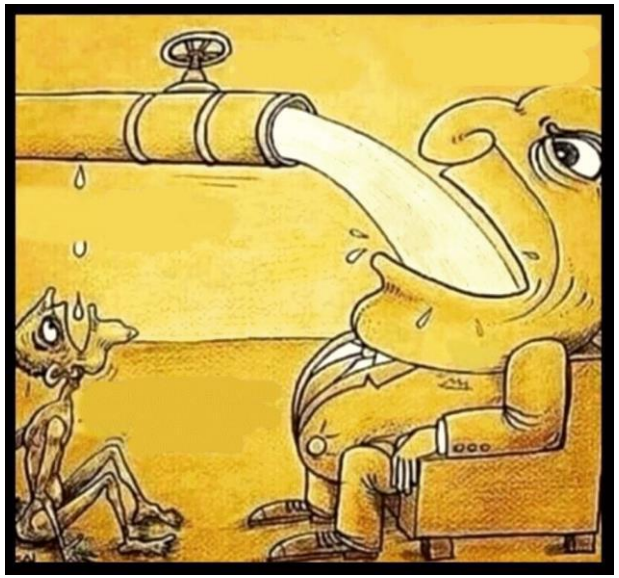


Packet Scheduling

Scheduling is used to determine the next packet to send on the link

Priority: packets are classified into priority classes. High priority = sent over link first

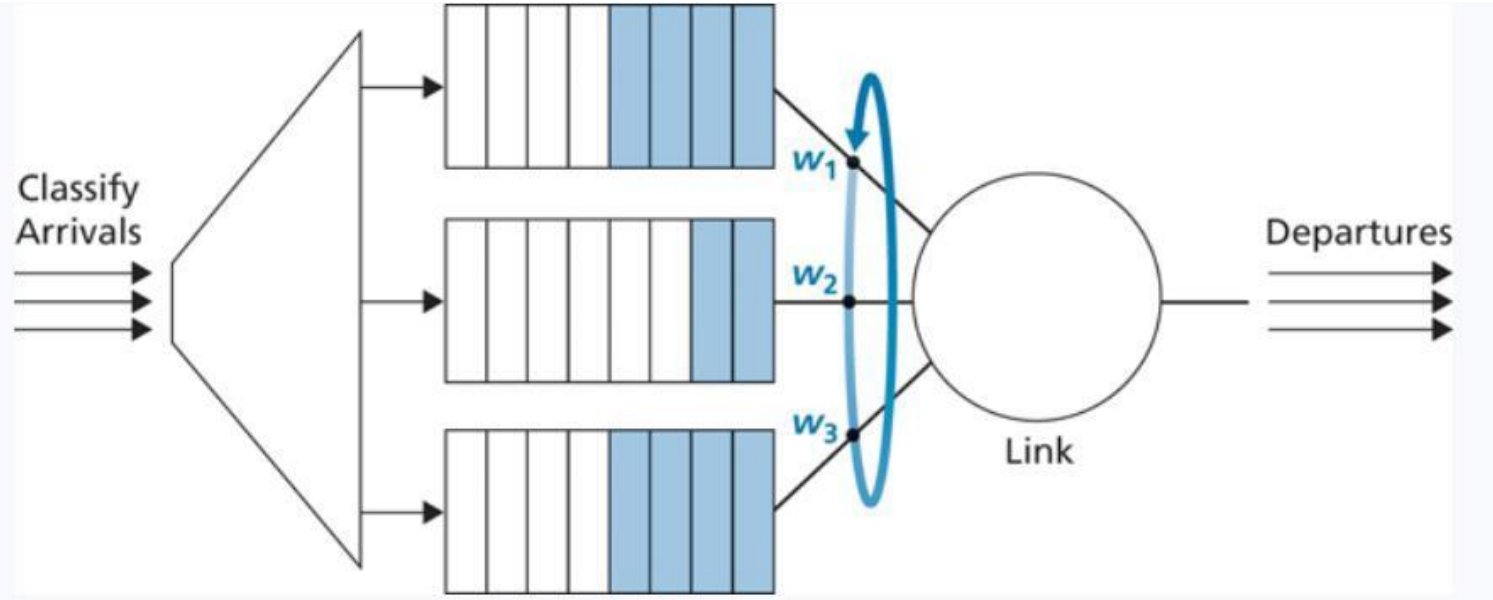




Packet Scheduling

Scheduling is used to determine the next packet to send on the link

Round robin weighted fair queueing: packets are classified into priority classes. Each class gets to send one packet during a “cycle”

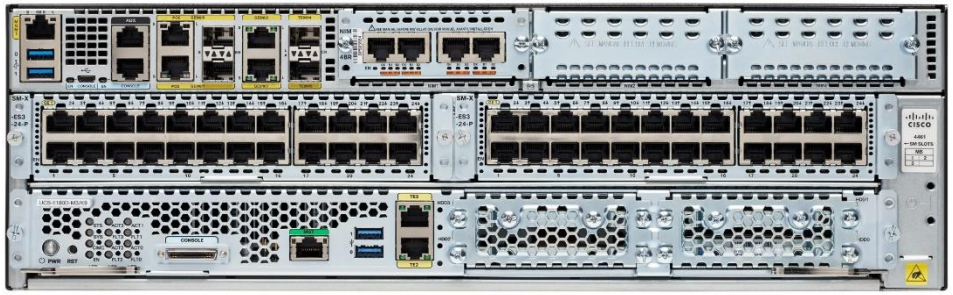


Packet from the priority group gets sent

Packet from the middle class gets sent

Packet from the plebian class gets sent

A Router



A Router

A router has software, hardware, and some kind of OS running on it

Routing-Protocols
- OSPF
- BGP
- RIP

Network Protocols
- DHCP
- DNS
- NAT

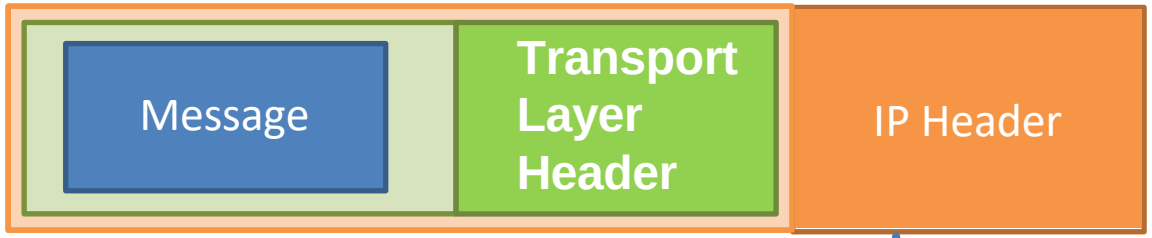
Network Services
- Firewall
- VPN
- Logging

IP PROTOCOL

IP addressing, IPv4, and IPv6

https://www.rfc-editor.org/rfc/rfc791

Packets traversing through the network layer are referred to as a **datagram**. Each packet gets an IPv4/IPv6 header



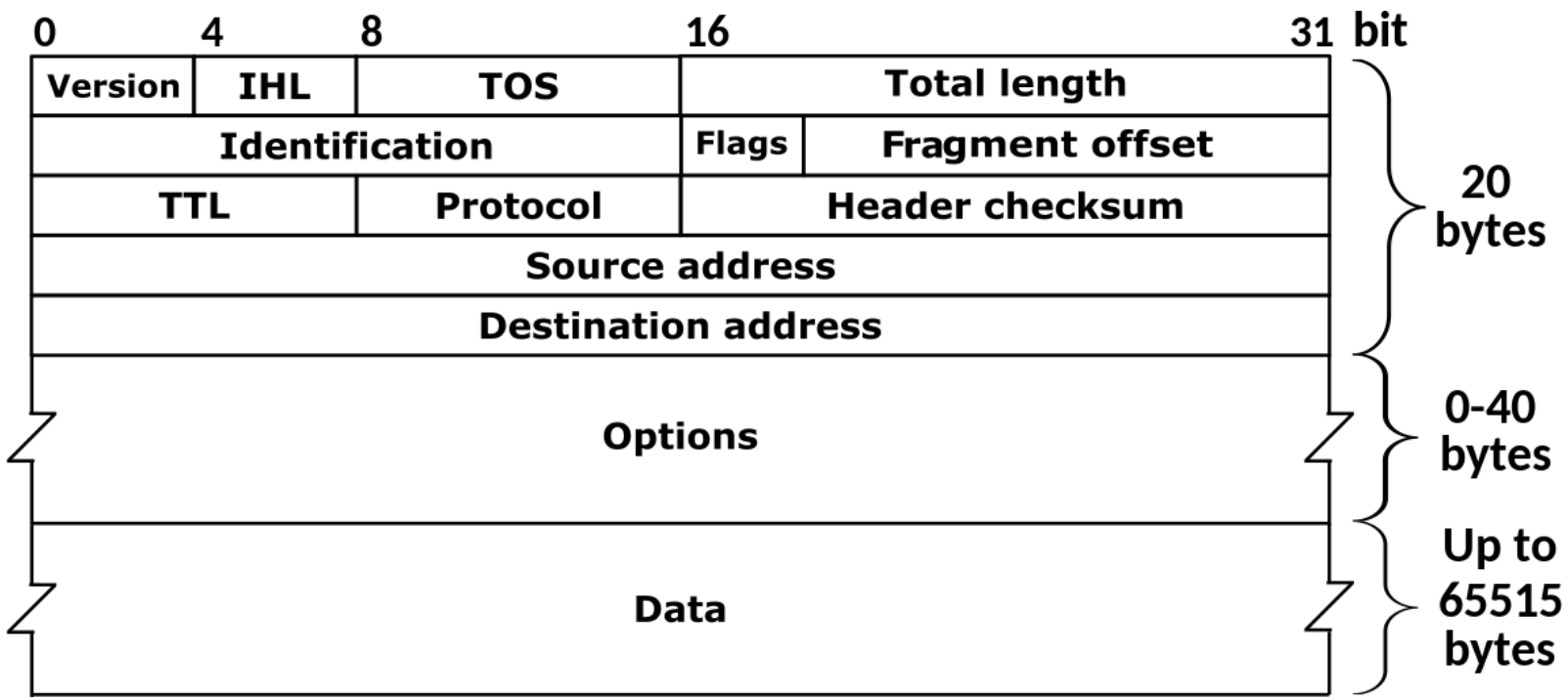
IPv4: 32-bit addresses (decimal)

192.149.252.76

IPv6: 128-bit addresses (hexademical)

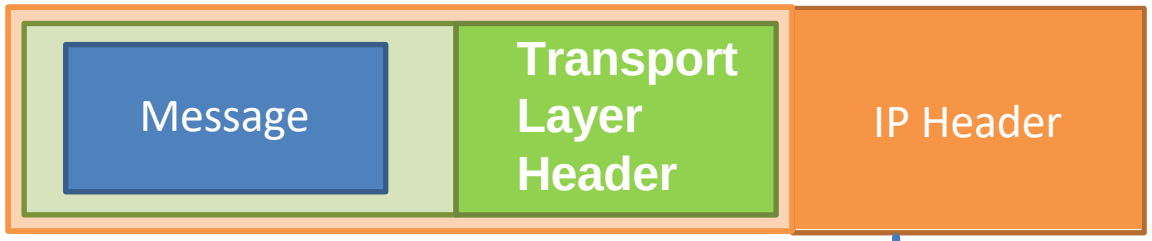
3ffe:1900:fe21:4545::

Why do we need IPv6?

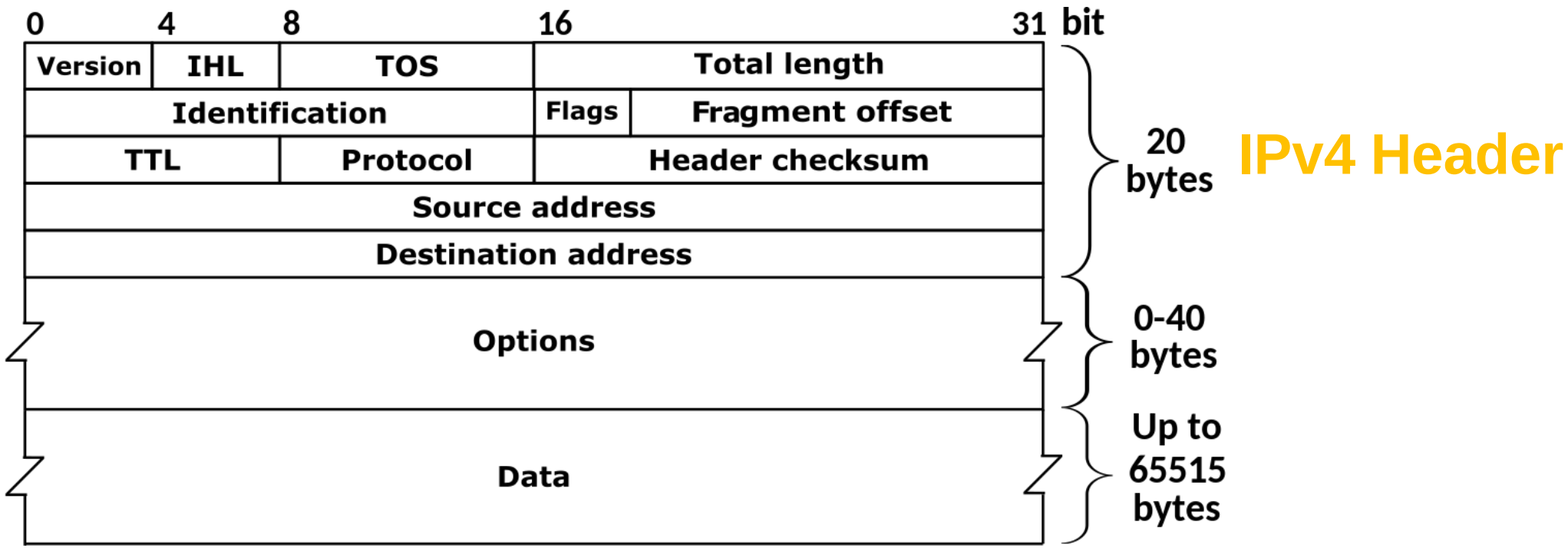


IPv4 Header

Packets traversing through the network layer are referred to as a **datagram**. Each packet gets an IPv4/IPv6 header



TCP header = 20 bytes
IP header = 20 bytes



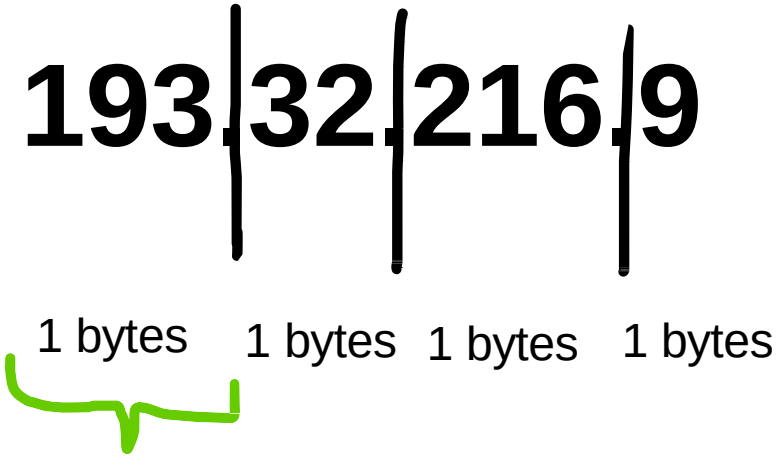
IPv4 Header

IP Address: Globally unique* 32 bit (4 byte) **dotted decimal** number assigned to interfaces on hosts and routers

193.32.216.9

IP Address: Globally unique* 32 bit (4 byte) **dotted decimal** number assigned to interfaces on hosts and routers

(1 byte = 8 bits)



128	64	32	16	8	4	2	1
1	**1**	0	0	0	0	0	**1**

128 + 64 + 1 = **193**

193 = `11000001` in binary

IP Address: Globally unique* 32 bit (4 byte) **dotted decimal** number assigned to interfaces on hosts and routers

(1 byte = 8 bits)

193.32.216.9

=

11000001 00100000 11011000 00001001

How many addresses are possible with a 32 bit number? **~4 billion possible IPv4 addresses**

How do IPs get obtained/assigned?

The Internet Corporation for Assigned Names and Numbers (ICANN) is responsible for managing and allocating IP address space for ISPs and organizations

(they also manage the DNS root servers!)

When an organization gets a range of IP addresses to use, how to we give assign them to devices?

Do we do it manually?

we could….

DHCP

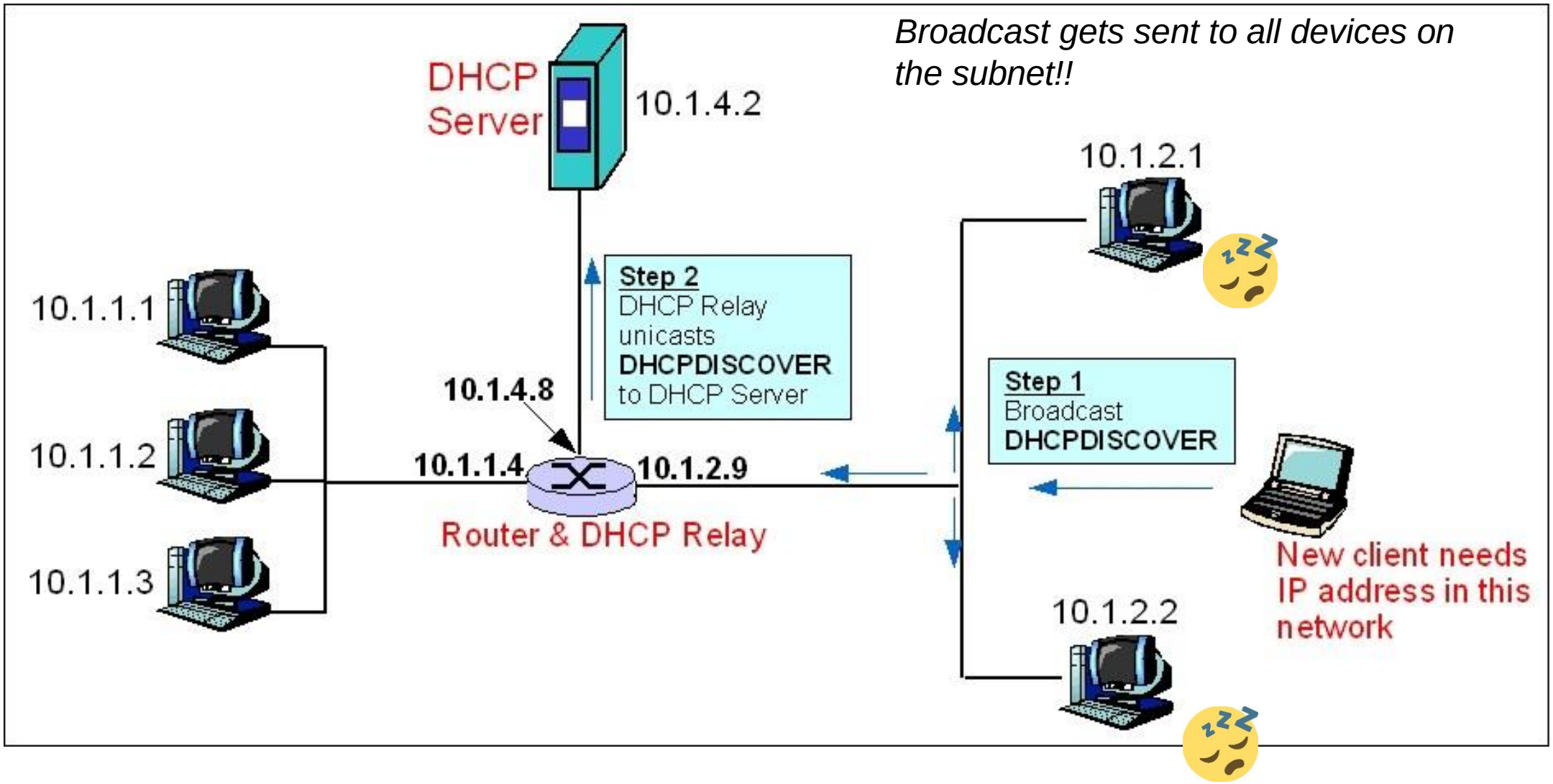
Dynamic Host Configuration Protocol (DHCP) is a **plug-and-play**, client-server protocol that allows a host to obtain an IP address automatically

When a host is automatically assigned an IP address, it might keep that one forever, or the IP addresses can be temporary

(more common)

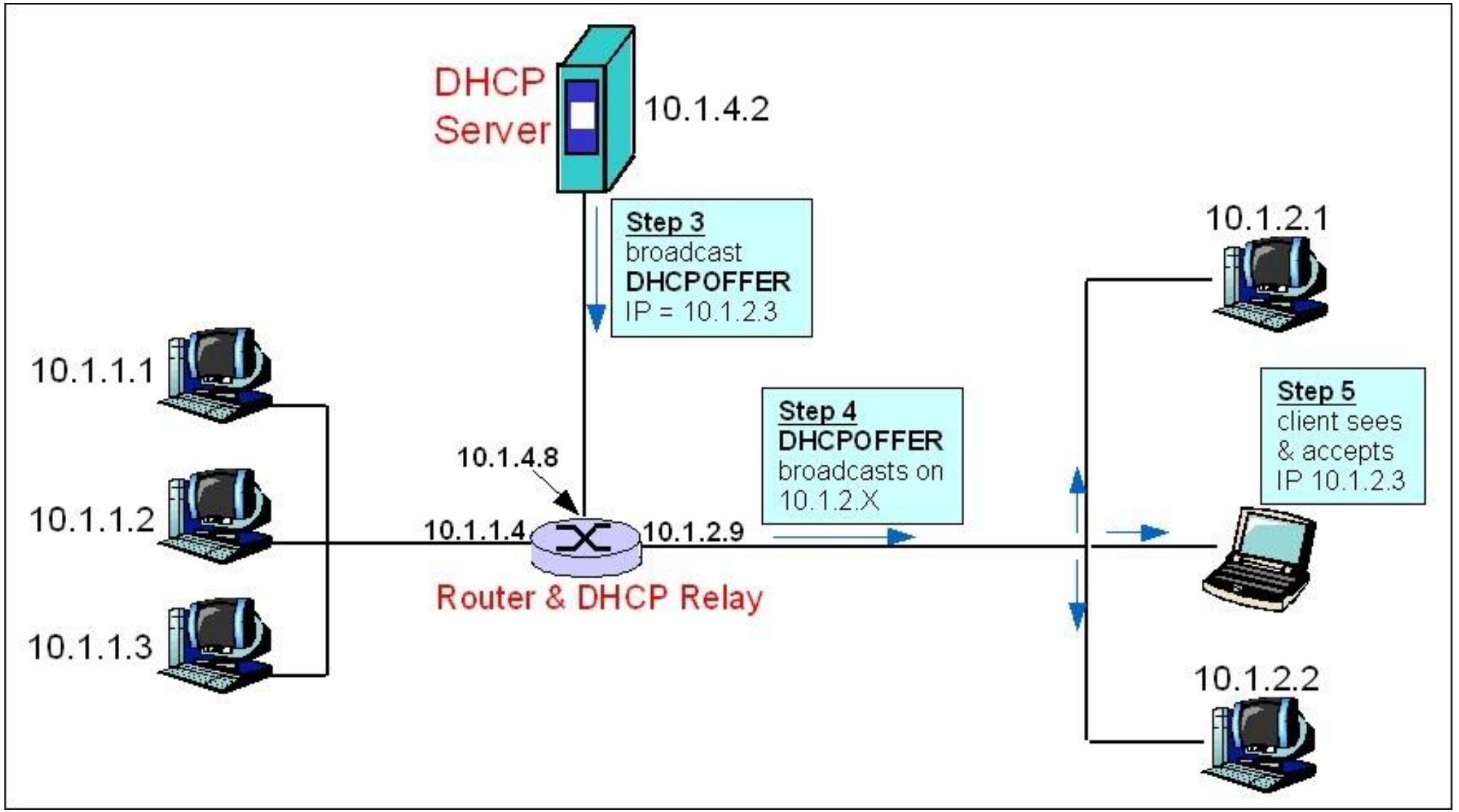
When DHCP finishes, the new host will have an IP address and contact info for the local DNS server

This process is similar to a TCP handshake!

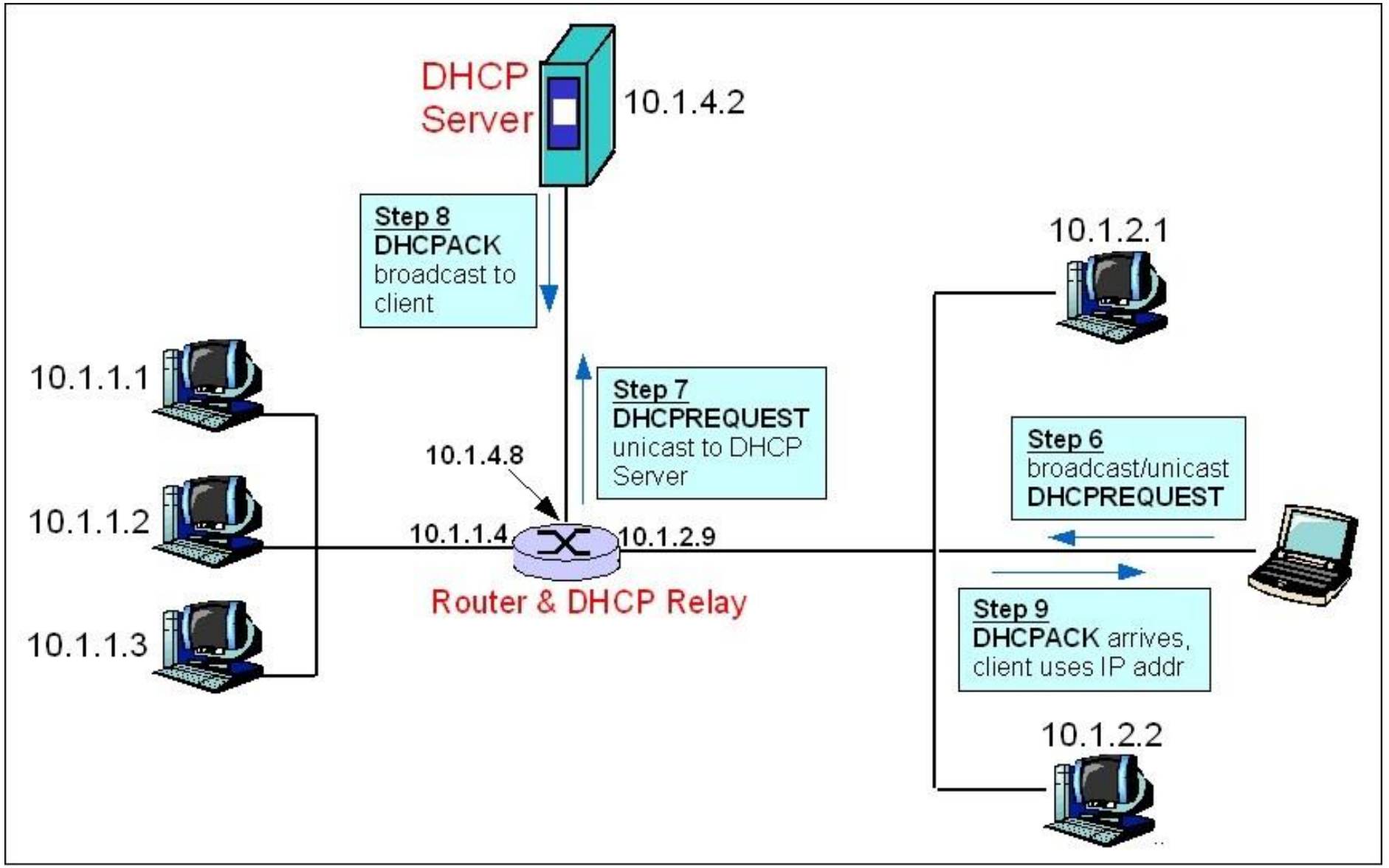


But devices that are **not** the dhcp server will ignore it

This process is similar to a TCP handshake!



This process is similar to a TCP handshake!



IP/DHCP in Wireshark