Elements of Networks

CS342, Lecture 7

Tuesday, Sept. 26th , 2006

Wellesley College
Daniel Bilar

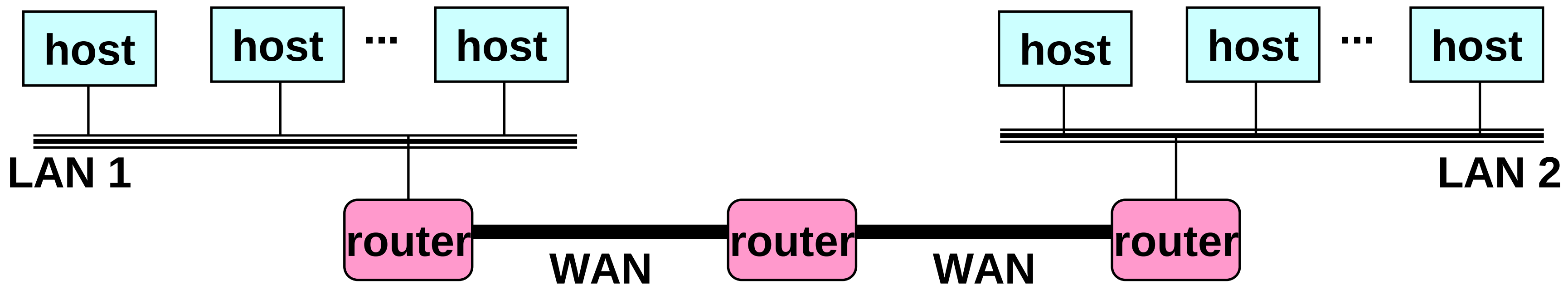
Today's class goals

- Introduction to networks (structure, addressing, protocols)
- Familiarity with OSI reference model
- Understanding the responsibilities and functions of the layers of the TCP/IP Protocol Stack
- Parsing an HTTP session and ethereal (new name: wireshark)

Internetworking

- Multiple incompatible LANs can be physically connected by specialized hardware/software called *routers*.
- The connected networks are called an *internetwork*.

  The *Internet* is one (big & successful) example of an internetwork



LAN 1 and LAN 2 might be completely different, totally incompatible LANs (e.g., Ethernet, Wi-Fi, ATM, Circuit-switched)

Internetworking Issues

- How do I designate a distant host?
  - Addressing / naming
- How do I send information to a distant host?
  - Underlying service model
    - What gets sent?
    - How fast will it go?
    - What happens if it doesn't get there?
  - Routing
    - How to get information from A to B?
- Challenges
  - Heterogeneity
    - Assembly from variety of different networks
  - Scalability
    - Ensure ability to grow to worldwide scale

OSI Reference model

- **O**pen **S**ystems **I**nterconnection- Released by International Organization for Standardization (ISO) in 1984
- Allows for a layered approach to implementation of networking
- 7 layers (for political reasons)
  - Application, Presentation, Session, Transport, Network, Data link, Physical
- ➔ Model is used, protocols are not

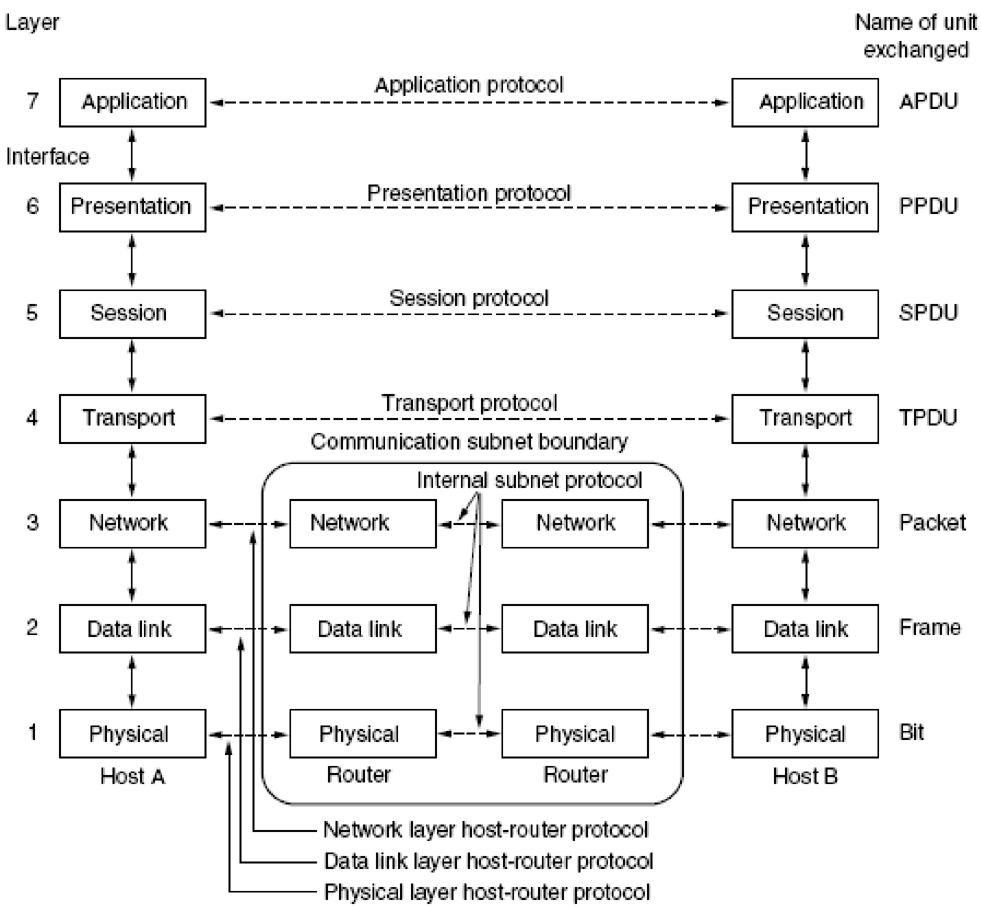


Figure 1-20. The OSI reference model.

Terminology

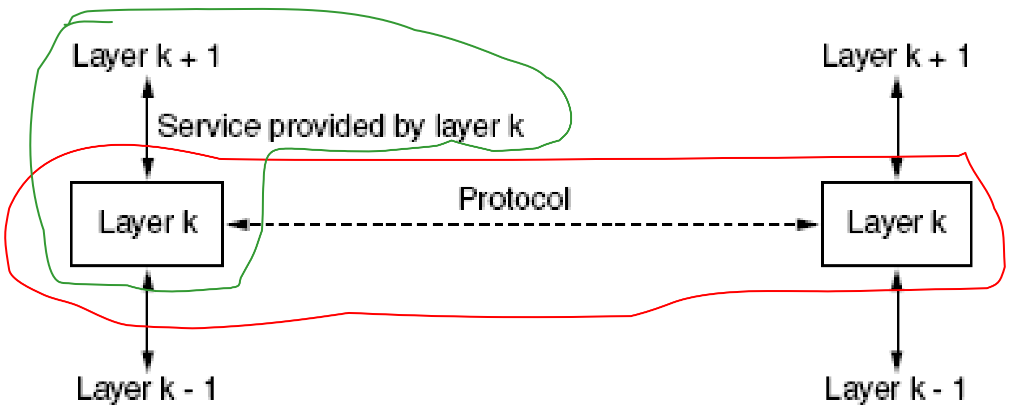


Figure 1-19. The relationship between a service and a protocol.

<u>Protocol</u>

Set of rules governing the format and meaning of messages between peers within same layer

Specifies implementation of service,

IMPLEMENTATION

<u>Service</u>

Set of 'primitives' (operations) that a layer provides to the layer above it.

Says what the layer does,

DESCRIPTION

Acccess service through

<u>Interface</u>

Set of methods, parameters and return values

Specifies how layer is accessed

INTERFACE

Why did OSI fail?

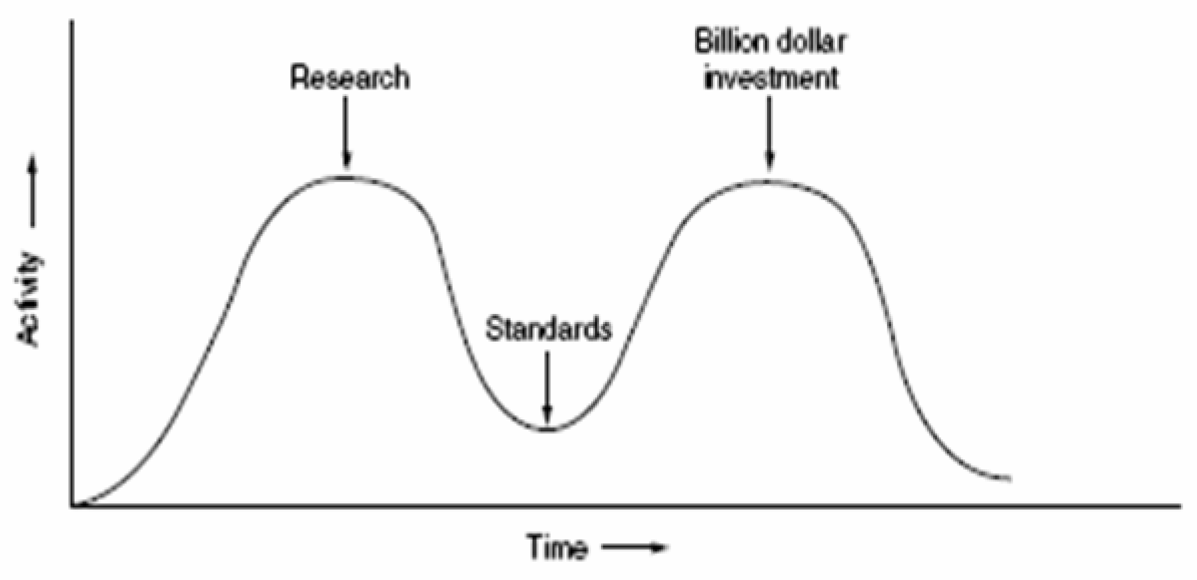


Figure 1-23. The apocalypse of the two elephants.

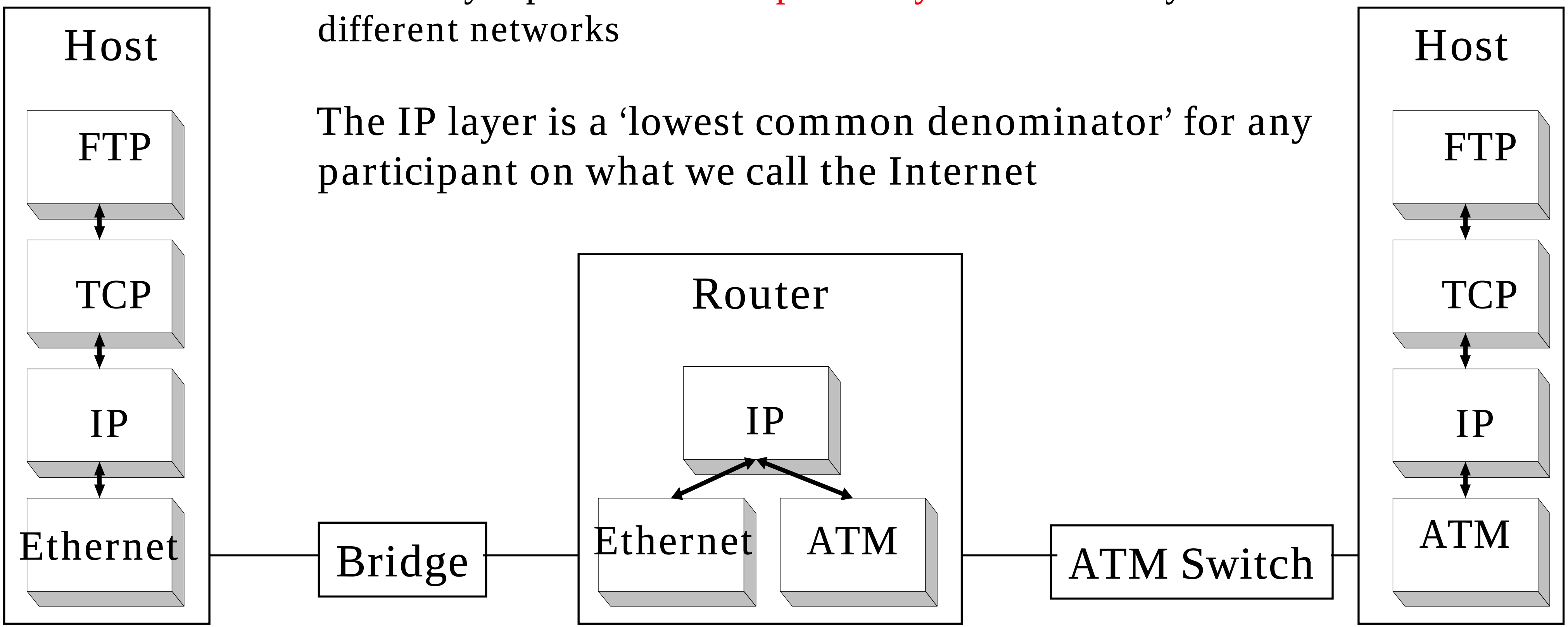
David Clark's (MIT) theory of standards

- Bad implementations
  - Huge and slow and inefficient in operation
- Bad politics
  - Perception of Euroweenies production
- Bad timing
  - "The apocalypse of the two elephants"
- Bad technology
  - Very complex
  - Q: What do you get when you cross a mobster with an international standard?
  - A: Someone who makes you an offer you can't understand

Introducing the Internet Network Layer (IP)

The IP layer provides interoperability between vastly different networks

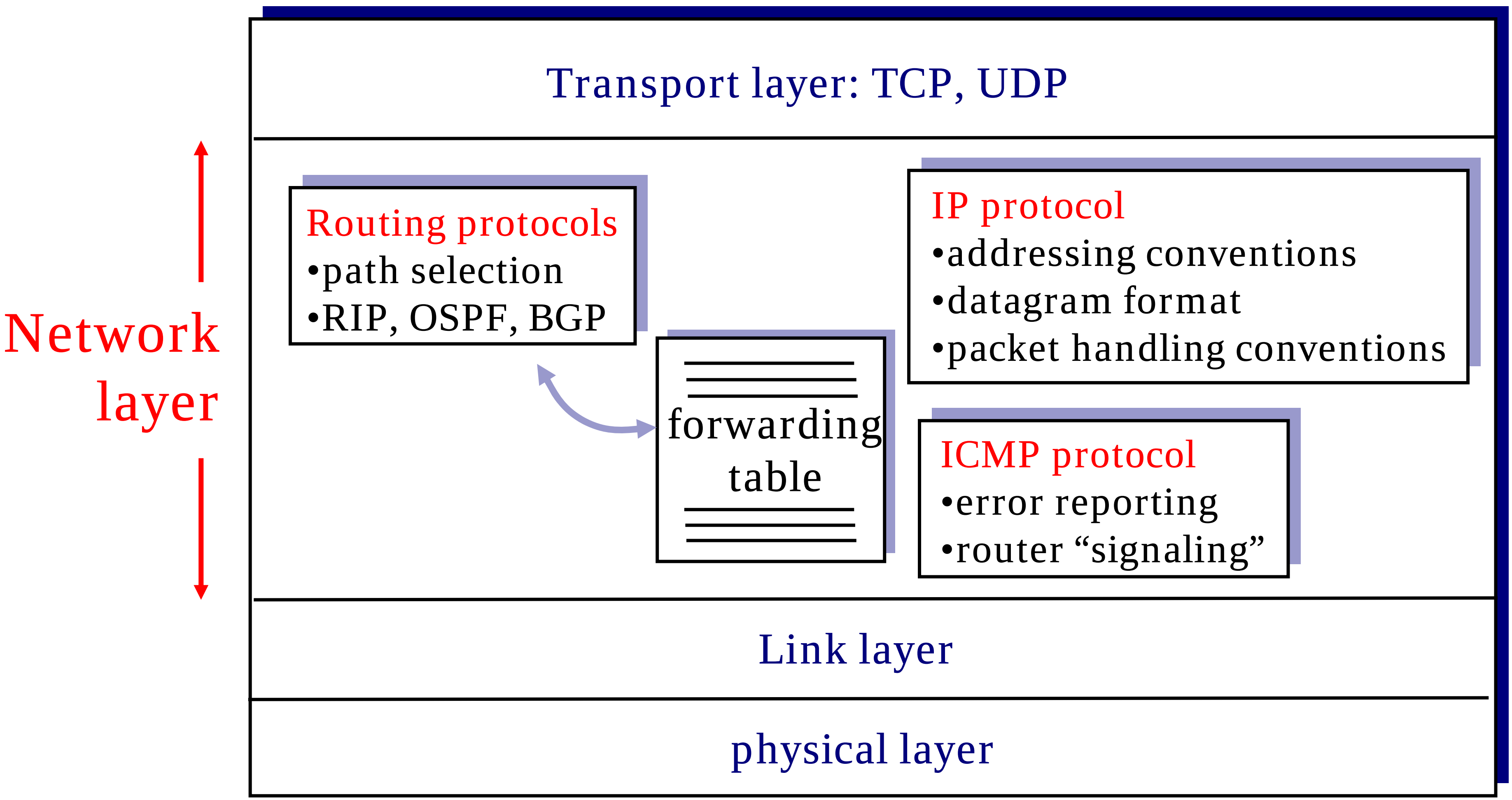
The IP layer is a ‘lowest common denominator’ for any participant on what we call the Internet



The Internet IP layer is the linchpin that holds the whole Internet architecture together!

The Internet ‘Network’ layer

Host, router network layer functions:



The Internet ‘Network’ layer’s job is to permit hosts to inject packets into *any* network and have them travel independently to the destination (potentially on a different networks)

The Internet Protocol (IP)

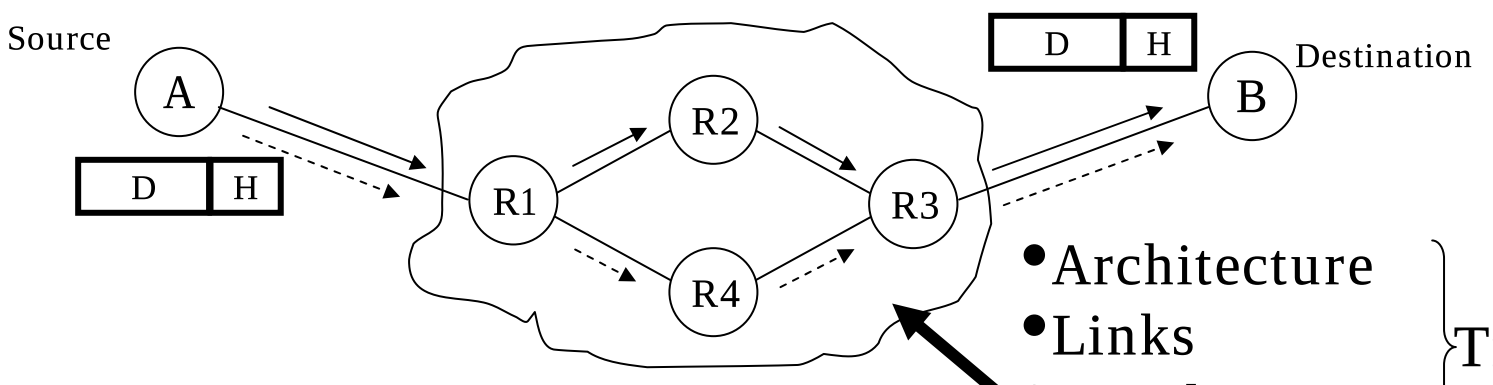
The Internet layer defines an official packet format and protocol called IP (Internet Protocol) -> An implementation of network layer

Designed for packet-switched network, each packet contains no more than 64K bytes

Connectionless: Each packet is routed independently with sender and receiver address (what is the advantage?)

Best-effort: Packets could be discarded during transmission because of the exhaustion of resources or a failure at the data link or physical layer

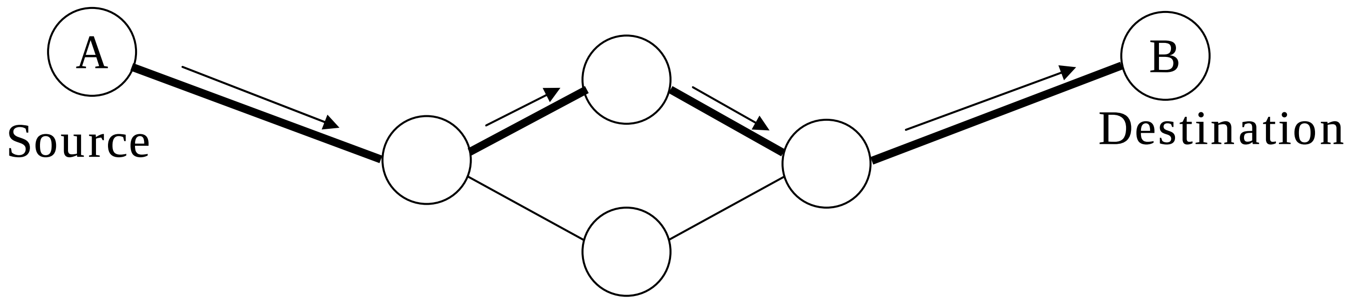
Unreliable: Reliability is ensured at higher layer, such as TCP



- Architecture
- Links
- Topology

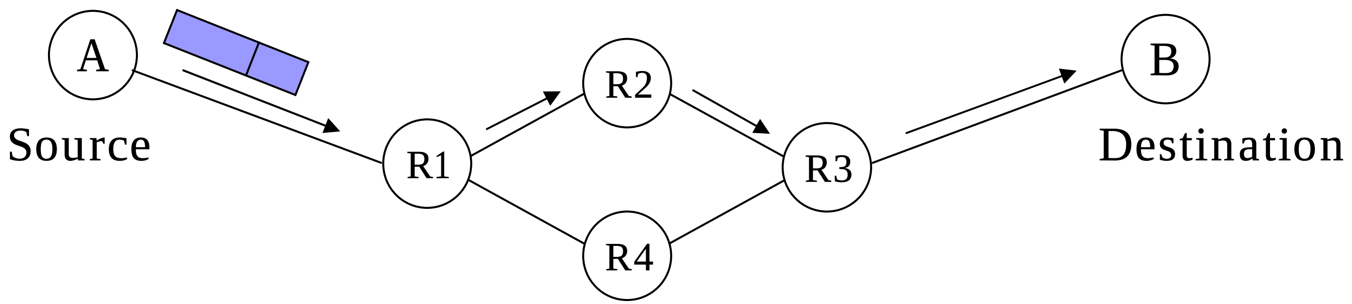
} Transparent

‘Reliable’: Circuit Switching



- Used by traditional telephone network
- A call has three phases:
  - Establish circuit from end-to-end (“dialing”),
  - Communicate,
  - Close circuit (“tear down”)
- Originally, a circuit was an end-to-end physical wire. Nowadays, a circuit is like a virtual private wire: each call has its own private, guaranteed data rate from end-to-end.

‘Unreliable’: Packet Switching



- Used by the ‘Internet’
- Each packet is individually routed packet-by-packet, using ‘routing tables’ of ‘routers’.
- Different packets may take different paths (see animation at http://computer.howstuffworks.com/internet-infrastructure.htm)

TCP/IP reference model

- From ARPANET (DoD)
- Specified in 1974 by Cerf and Kahn
- Bible is Stevens, TCP/IP Illustrated Volume 1 – the Protocols

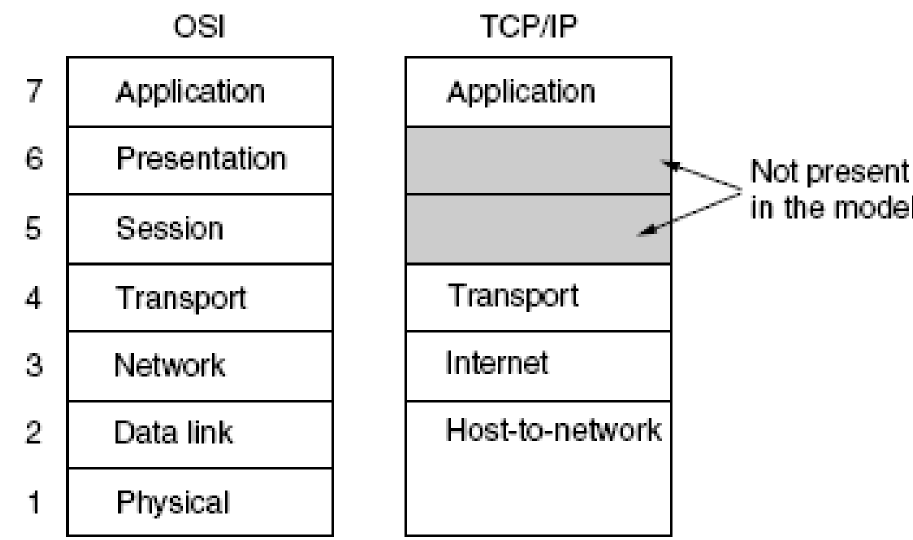


Figure 1-21. The TCP/IP reference model.

→ Protocols are widely used, model is not

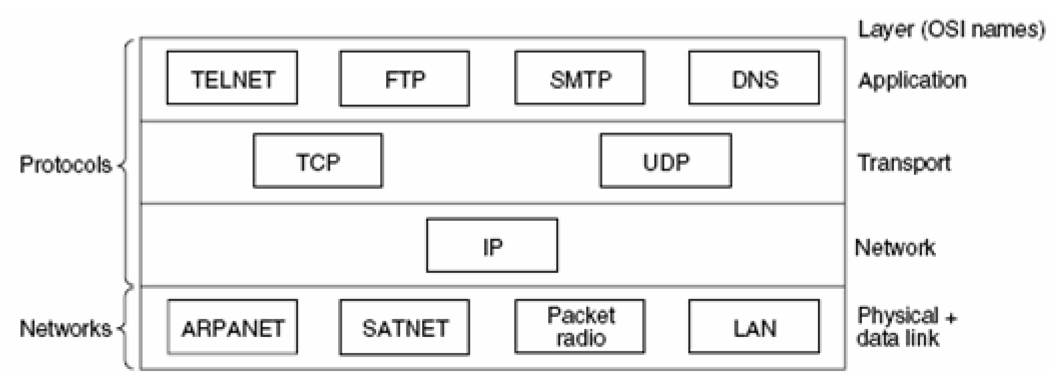


Figure 1-22. Protocols and networks in the TCP/IP model initially.

Addressing

a) Flat

- e.g., every host identified by its 48-bit MAC address
- Router would need entry for every host in the world
  - Too big (although technology can help this)
  - Too hard to maintain as hosts come and go

b) Hierarchy

- Address broken into segments of increasing specificity
  - Example (Daniel's office phone):
  - 781 (Wellesley Town) – 283 (College) – 3093 (office #)
- Route to general region and then work toward specific destination
- As people and organizations shift, only update affected routing tables

IP Addressing

- IPv4: 32-bit addresses
  - Typically, write in dotted decimal format
    - E.g., 149.130.12.213 (www.wellesley.edu)
    - Each number is decimal representation of one byte (8 bits)

0	8	16	24	31
149	130	12	213	Decimal
95	82	0C	D5	Hexadecimal
1001 0101	1000 0010	0000 1100	1101 0101	Binary

IP Addressing and Forwarding

- Routing table requirement
  - For every possible destination IP address, give next hop
  - Nearly 2^{32} (4.3×10^9) possibilities!
- Hierarchical Addressing Scheme

	x	y
pfx	network	host

  - Address split into network ID and host ID
  - All packets to given network follow same route
    - Until they reach destination network
  - Fields
    - `pfx` Prefix to specify split between network & host IDs
    - `network` 2^x possibilities
    - `host` 2^y possibilities

IP Address Classes

- Class A

	7	24
0	network	host

First octet: 1–126

mit.edu: 18.7.22.69

- Class B

	14	16
10	network	host

First octet: 128–191

wellesley.edu: 149.130.12.213

- Class C

	21	8
110	network	host

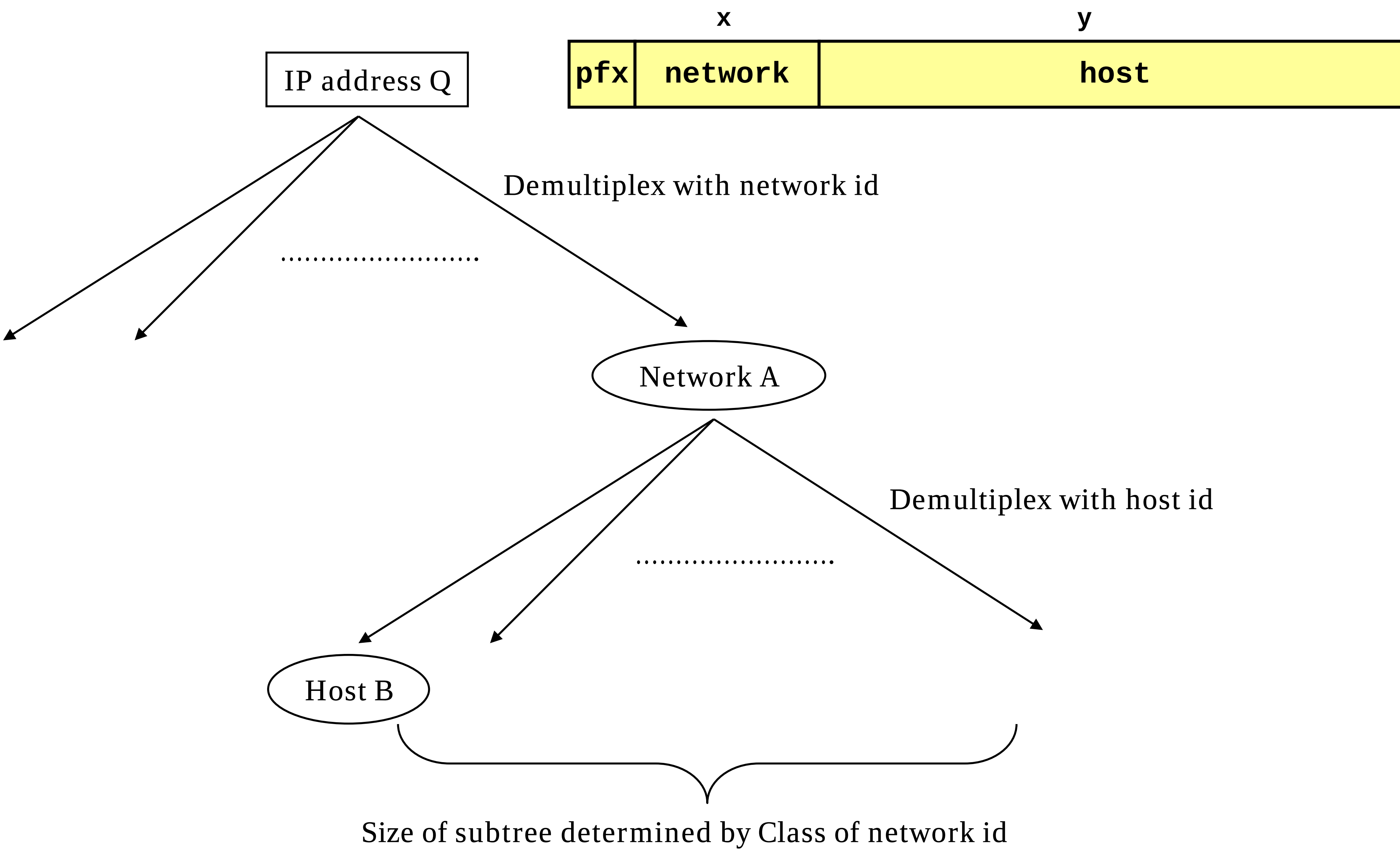
First octet: 192–223

adsl-216-63-78-18.dsl.hstntx.swbell.net: 216.63.78.18

- Classes D, E, F

  Not commonly used

Two levels: Basic IP addressing



IP Address Classes

Class	Count	Hosts
A	2^7-2 = 126 (0 & 127 reserved)	2^{24}-2 = 16,777,214 (all 0s, all 1s reserved)
B	2^{14} = 16,398	2^{16}-2 = 65,534 (all 0s, all 1s reserved)
C	2^{21} = 2,097,512	2^8-2 = 254 (all 0s, all 1s reserved)
Total	2,114,036	

- Partitioning too coarse
  - No local organization needs 16.7 million hosts
    - Large organization likely to be geographically distributed
  - Many organizations must make do with multiple class C's
- Too many different Network IDs
  - Routing tables must still have 2.1 million entries

Subnetting

- Add a third level

x z y

pfx	network	subnet	host

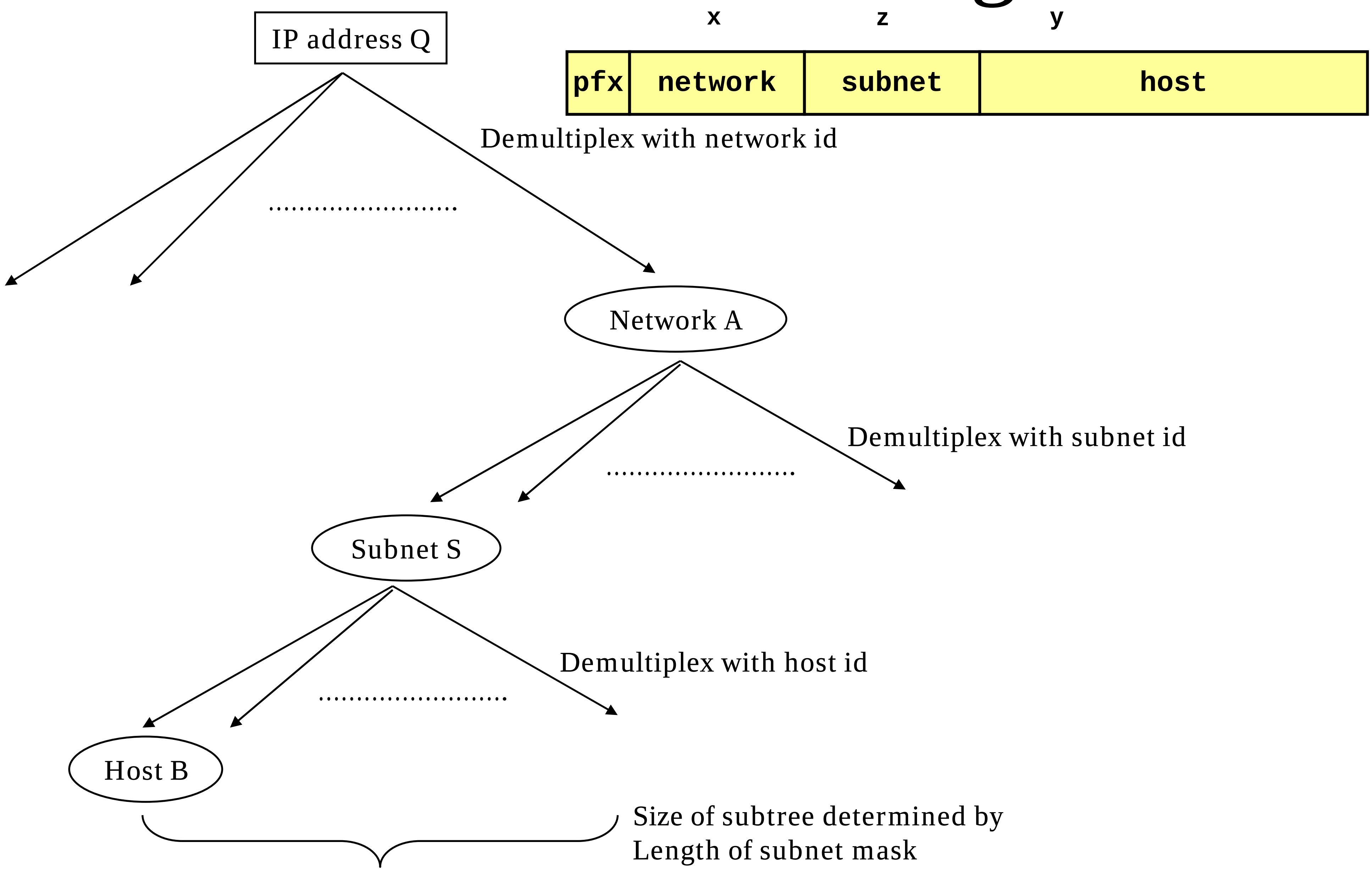
  - From the outside, appears as one monolithic network
    - Single entry in routing table
  - Within network, manage as multiple subnetworks
    - Internal routers must route according to subnet ID
- Subnet Mask
  - Way to specify break between subnet ID and host ID

1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

pfx	network	subnet	host

  - Similar masks used in many contexts

Three levels: Subnetting



Routing Table

Address Pattern	Subnet Mask	Next Hop
149.130.12.213	255.255.255.0	R1
149.130.128.0	255.255.128.0	R2
149.0.0.0	255.0.0.0	R3
0.0.0.0	0.0.0.0	R4
149.130.0.0	255.255.0.0	R5

- Address 149.130.12.213 matches 4 entries
- Longest Prefix Match

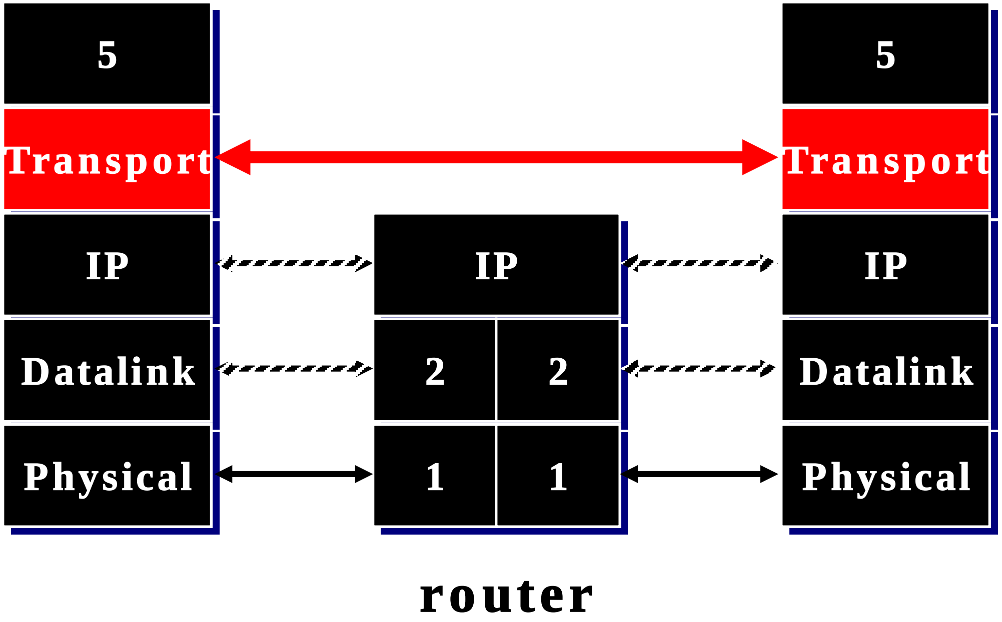
  In English: Choose most specific case

  Select entry with longest sequence of 1's in mask

  Most specific case

Transport Protocols Concern only End Hosts, not Routers

- Lowest level end-to-end protocol.
  - Header generated by sender is interpreted only by the destination
  - Routers view transport header as part of the payload
- Adds functionality to the best-effort packet delivery IP service.
  - Make up for the shortcomings of the core network



What do Transport Protocols do?

- Multiplexing/demultiplexing for multiple applications
  - Port abstraction
- Connection establishment
  - Logical end-to-end connection
  - Connection state to optimize performance
- Error control
  - Hide unreliability of the network layer from applications
  - Many types of errors: corruption, loss, duplication, reordering.
- End-to-end flow control
  - Avoid flooding the receiver
- Congestion control
  - Avoid flooding the network

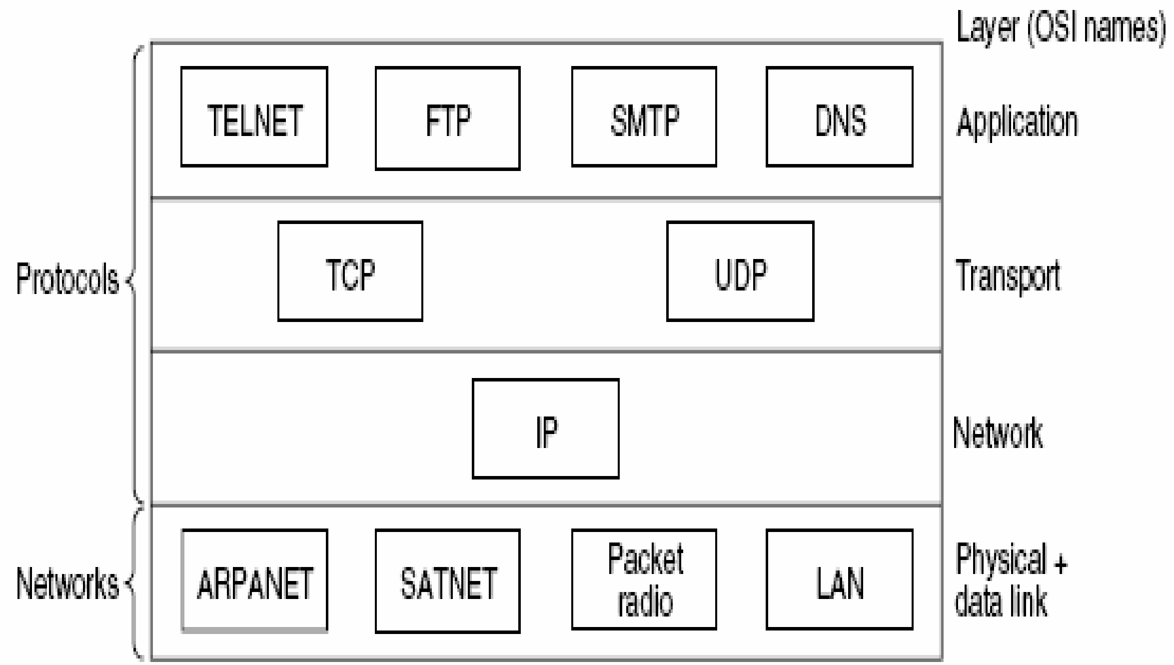


Figure 1-22. Protocols and networks in the TCP/IP model initially.

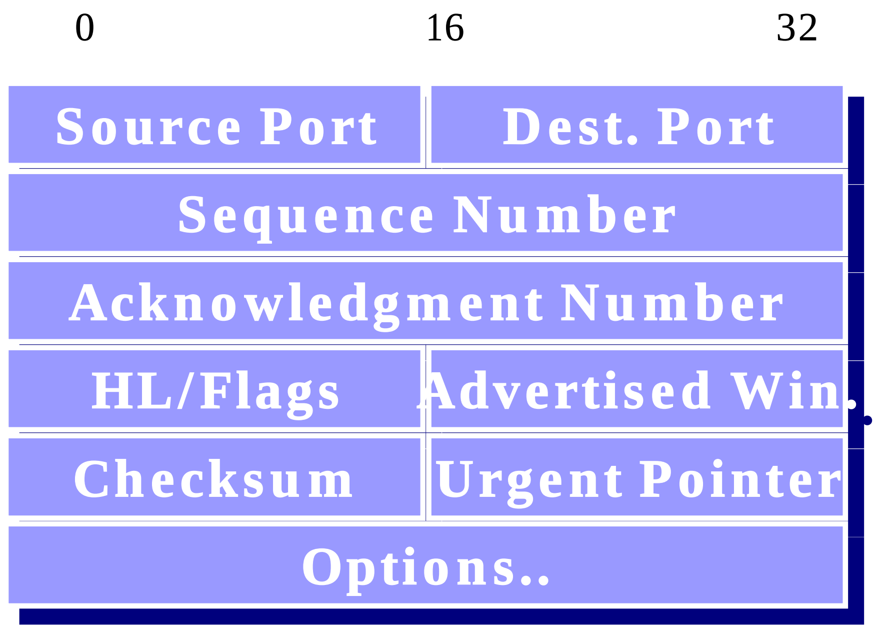
Universal Datagram Protocol (UDP)

- An implementation of transport layer on top of IP
- Unreliable data transmission
  - No guaranteed on delivery
  - Packets could be received out of order
- Add port identification numbers and payload checksum to IP
  - Ports allow multiplexing of data streams
- Highly efficient because of low overhead
  - Suitable for delivering data that is small amount and needs to be sent frequently
  - Typically used for latency-sensitive or low-overhead applications (video, time, DNS, etc.)

0	16	32
Source Port	**Dest. Port**	
Length	**Checksum**	

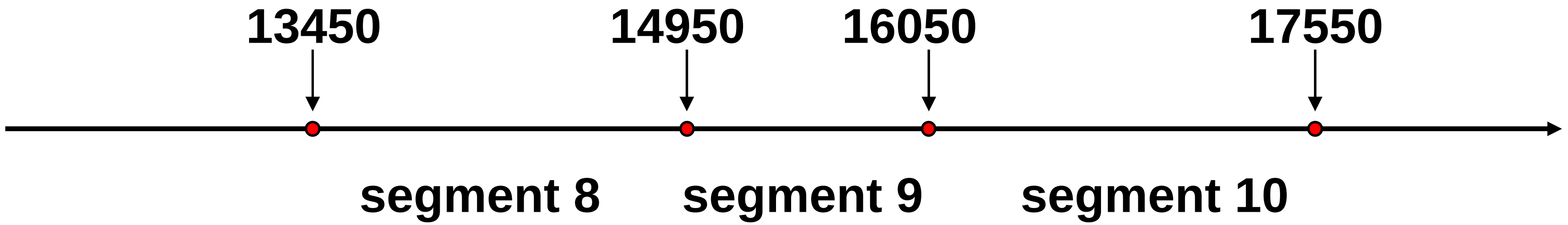
Transmission Control Protocol (TCP)

- An implementation of transport layer on top of IP
- Reliable data transmission that can be used to send a sequence of bytes
  - Provide guaranteed delivery and ordering of bytes, i.e., data are always received in their original order
- Port numbers, like UDP
- Checksums payload
- Flow control
  - Sensitive to packet loss and round-trip time
- Error recovery: retransmit lost/corrupted packets



TCP: Sequence Number Space

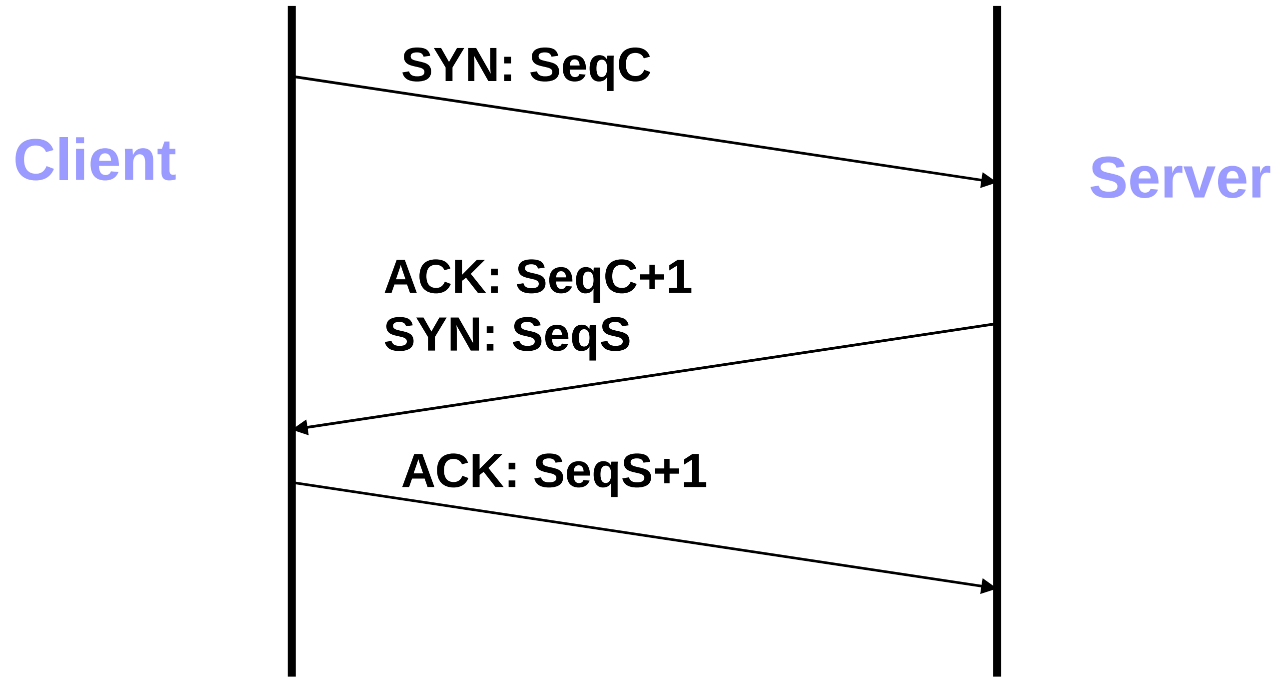
- Each byte in byte stream is numbered.
  - 32 bit value
  - Wraps around
  - Initial values selected at start up time
- TCP breaks up the byte stream in packets ("segments")
  - Packet size is limited to the Maximum Segment Size
  - Set to prevent packet fragmentation
- Each segment has a sequence number.
  - Indicates where it fits in the byte stream



TCP: Important Flags

- SYN: Synchronize
  - Used when setting up connection
- FIN: Finish
  - Used when tearing down connection
- ACK
  - Acknowledging received data

Establishing TCP Connection

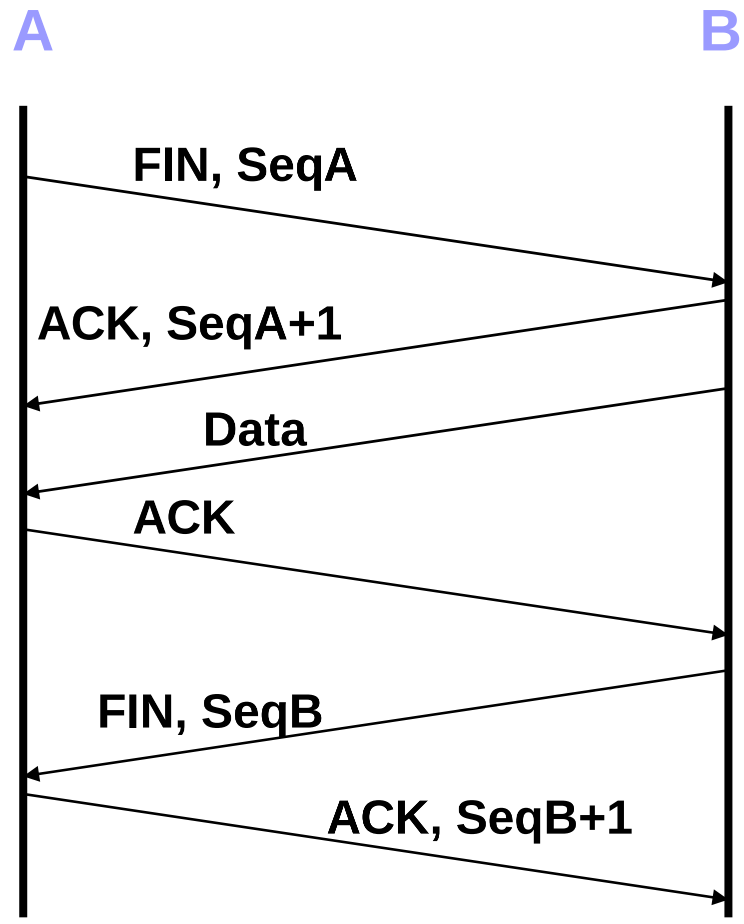


"Three-Way Handshake"

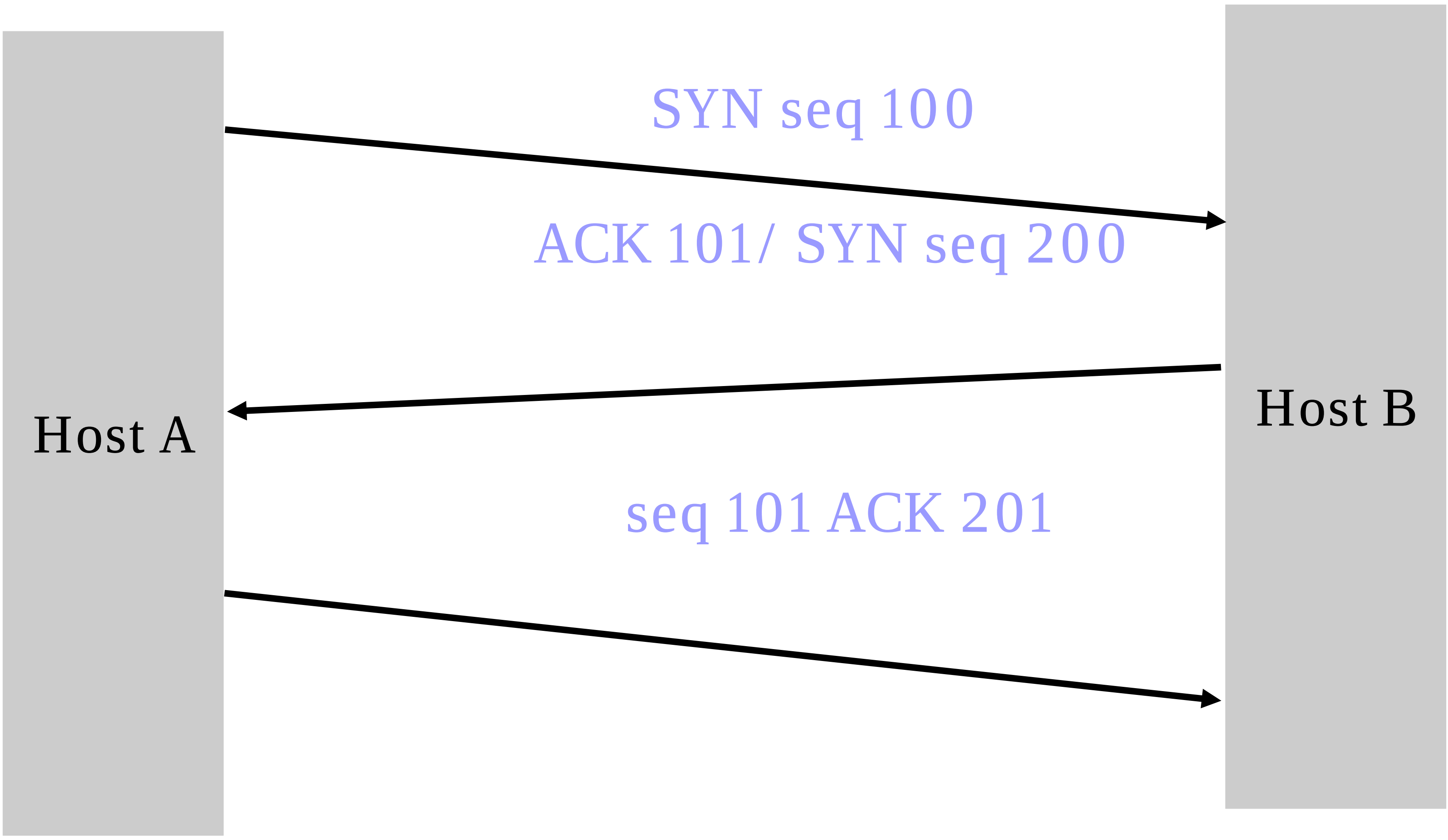
- Each side notifies other of starting sequence number it will use for sending
- Each side acknowledges other's sequence number
  - SYN-ACK: Acknowledge sequence number + 1
- Can combine second SYN with first ACK

Protocol: Tearing Down Connection

- Either Side Can Initiate Tear Down
  - Send FIN signal
  - "I'm not going to send any more data"
- Other Side Can Continue Sending Data
  - Half open connection
  - Must continue to acknowledge
- Acknowledging FIN
  - Acknowledge last sequence number + 1

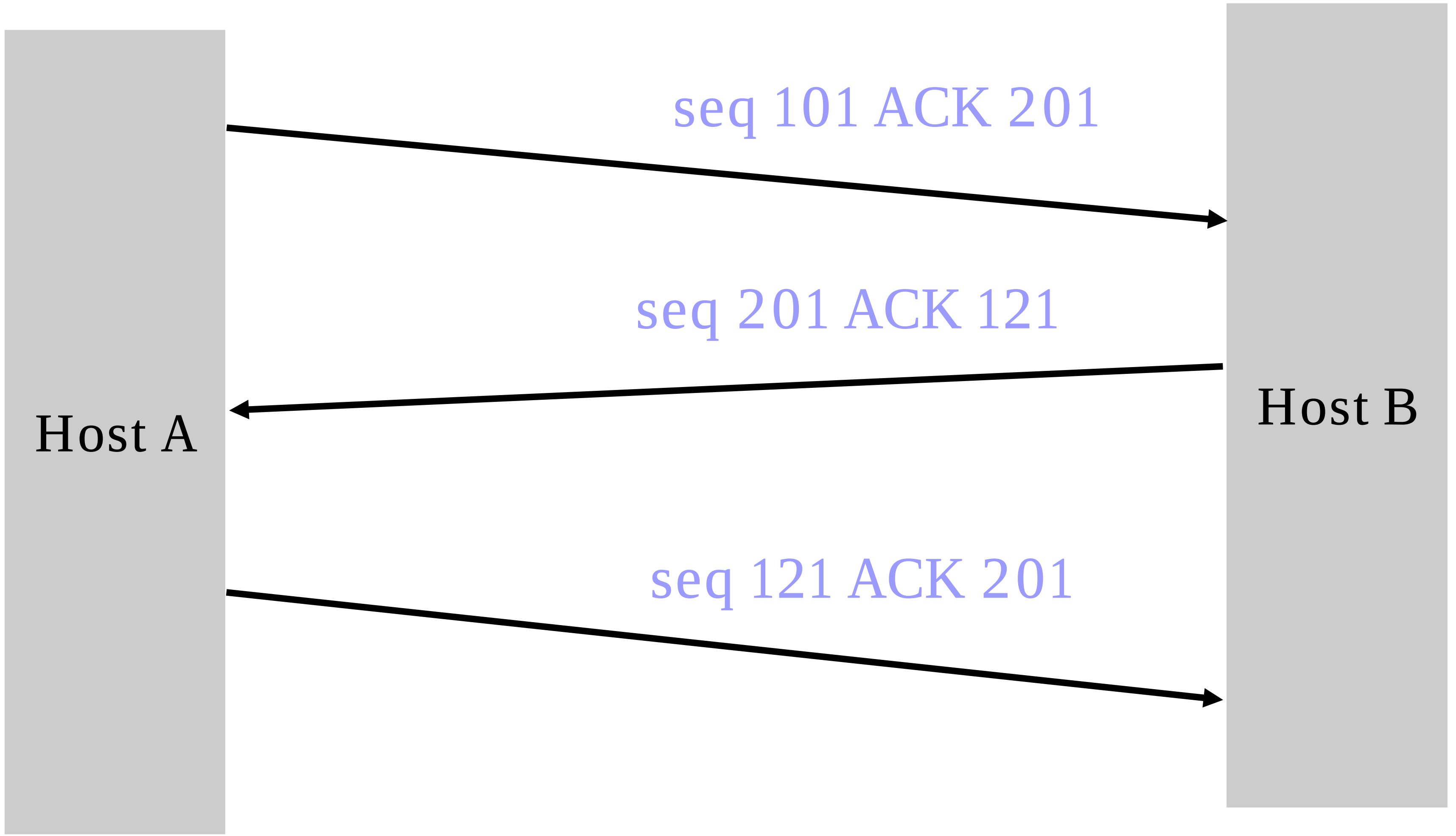


1. Establish a TCP Connection

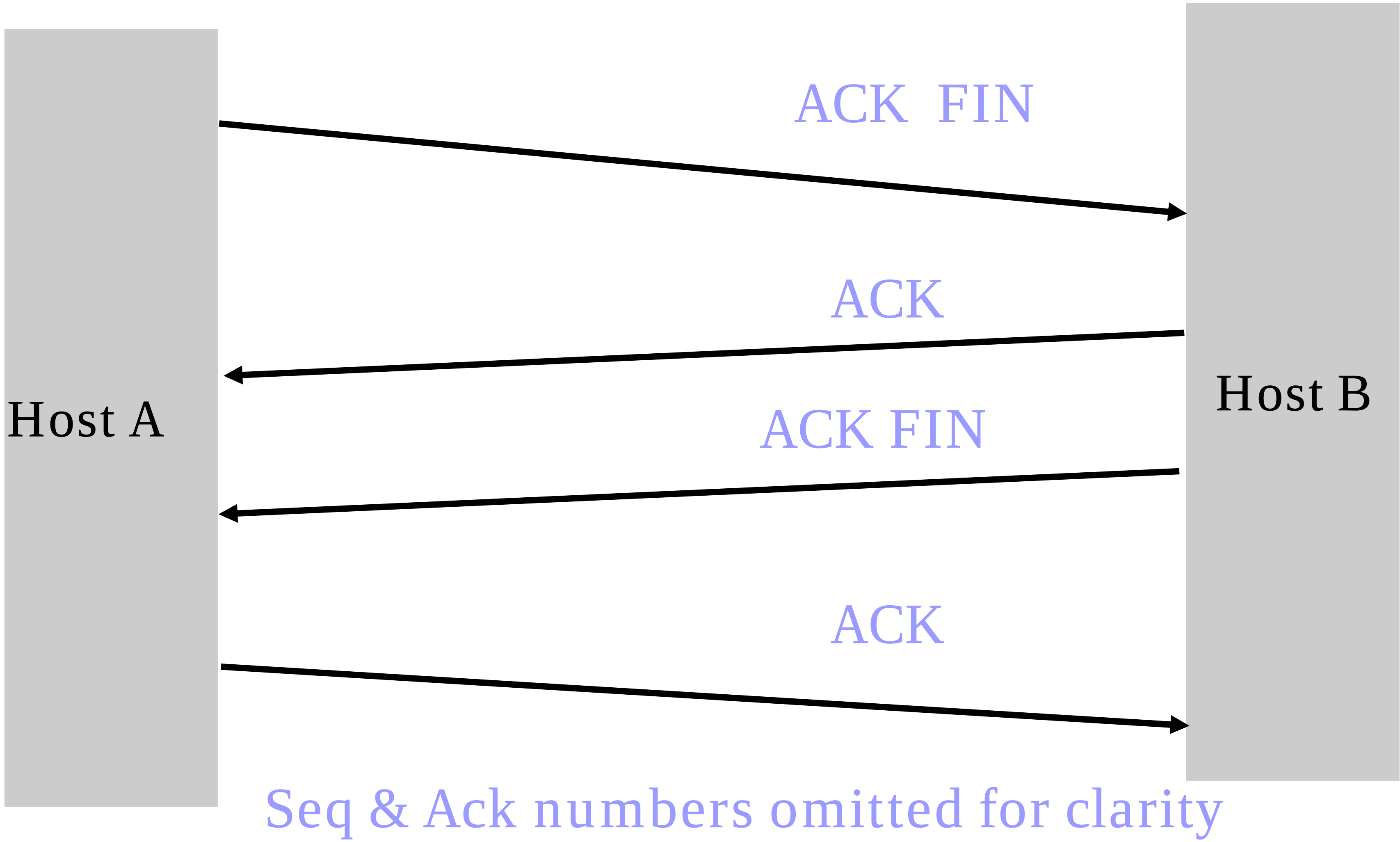


The "three-way handshake"

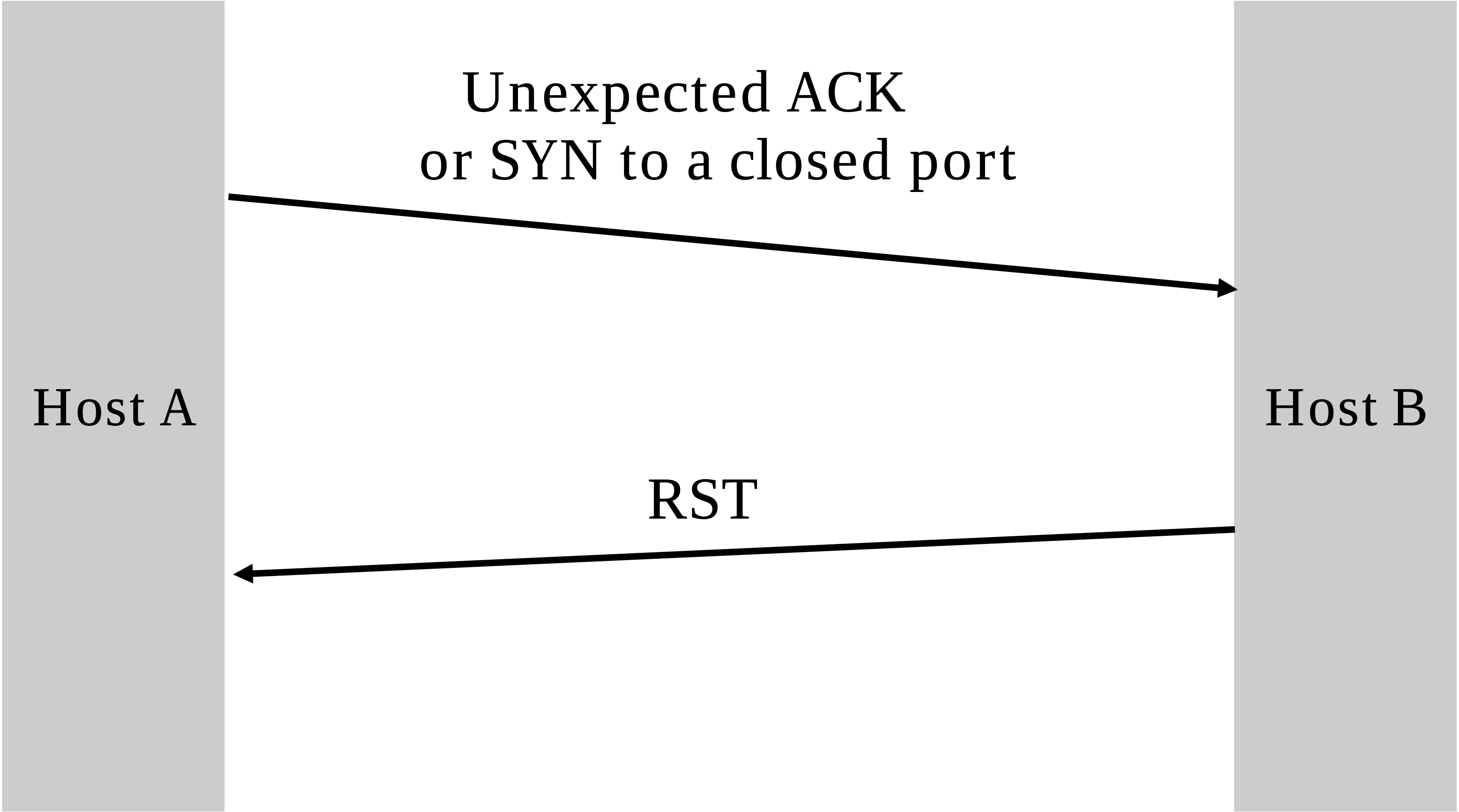
2. Transmit data over TCP



3a. Terminate TCP Connection gracefully



3b. Terminate TCP Connection abruptly



A Reset flag terminates communication w/o further ado

Tcpdump

- Tcpdump(8):

  Tcpdump outputs the headers of packets on a network interface that match the boolean expression.

  Output in ascii, hexadecimal or binary file

Tcpdump
TCP Connection establishment

03:08:39.91 : 192.168.1.4.3749 > 10.198.102.190.telnet: S **3588754520**:3588754520(0) win 5840 <mss 1460,sackOK,nop> (DF)

03:08:39.94 : 10.198.102.190.telnet > 192.168.1.4.3749: S **1624898076**:1624898076(0) ack *3588754521* win 1024 <mss 536>

03:08:39.99 : 192.168.1.4.3749 > 10.198.102.190.telnet: . 3588754521:3588754521(0) ack *1624898077* win 5840 (DF)

Tcpdump
TCP Connection Termination

03:08:46.76: 192.168.1.4.3749 > 10.198.102.190.telnet: F **3588754563**:3588754563(0) ack 1624898686 win 6804 (DF)

03:08:46.77: 10.198.102.190.telnet > 192.168.1.4.3749: F **1624898686**:1624898686(0) ack *3588754564* win 982

03:08:46.83: 10.198.102.190.telnet > 192.168.1.4.3749: F **1624898686**:1624898686(0) ack *3588754564* win 982

03:08:46.84: 192.168.1.4.3749 > 10.198.102.190.telnet: . 3588754564:3588754564(0) ack *1624898687* win 6804 (DF)

Tcpdump
Abrupt TCP Connection Termination

03:38:12.488670 > 192.168.1.4.3754 > 192.168.1.1.telnet: S 1166553013:1166553013(0) win 5840 <mss 1460,sackOK,timestamp 52171146 0,nop,wscale 0> (DF)

4500 003c 0000 4000 4006 b766 c0a8 0104
c0a8 0101 0eaa 0017 4588 2fb5 0000 0000
a002 16d0 147d 0000 0204 05b4 0402 080a
031c 118a 0000 0000 0103 0300

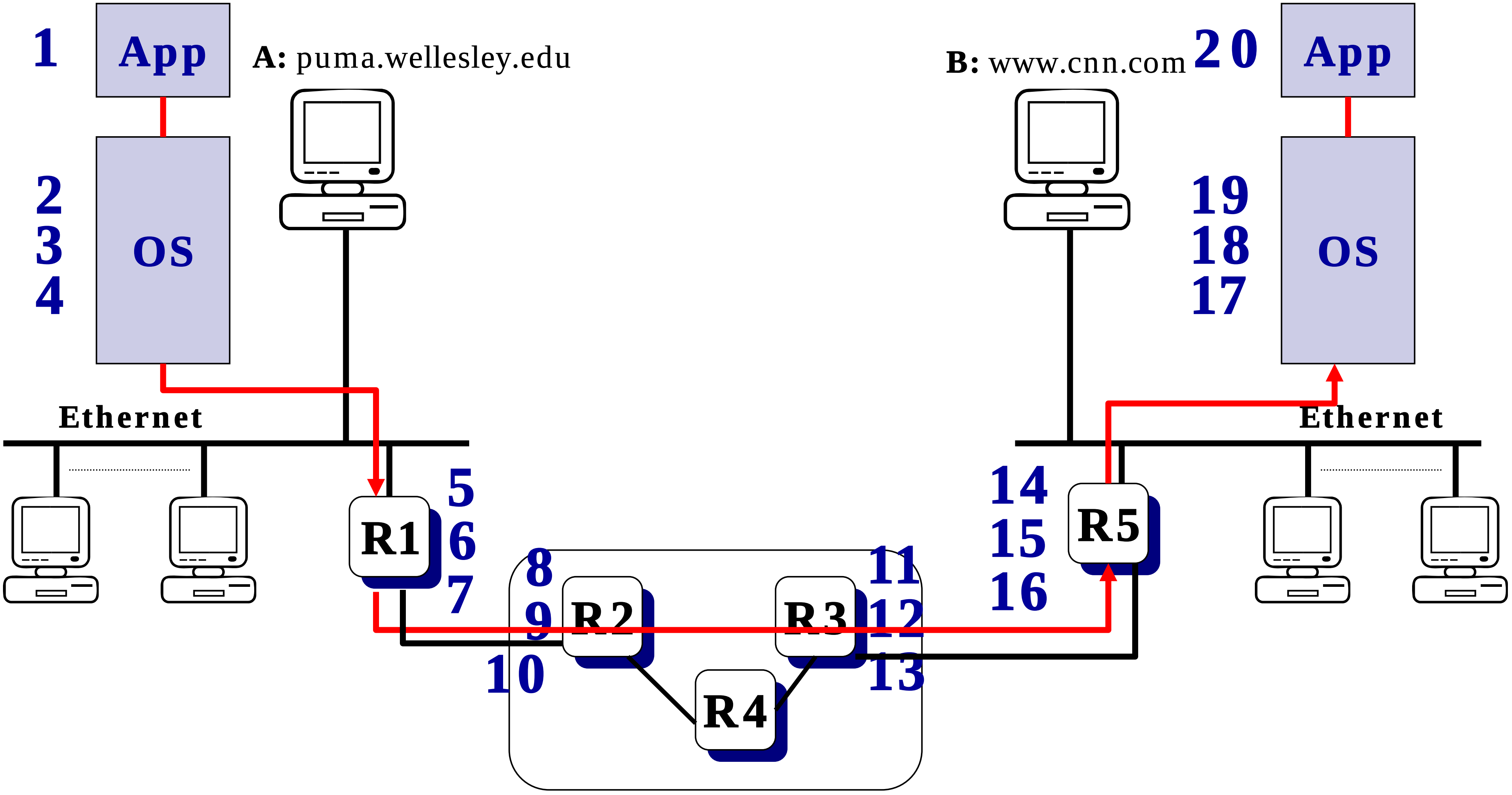
ip[9] = 0x06 *(tcp)*

03:38:12.489026 < 192.168.1.1.telnet > 192.168.1.4.3754: R 0:0(0) ack 1166553014 win 0

4500 0028 3447 0000 ff06 0433 c0a8 0101
c0a8 0104 0017 0eaa 0000 0000 4588 2fb6
5014 0000 a87b 0000 0000 0000 0000

tcp[13] = 0x14

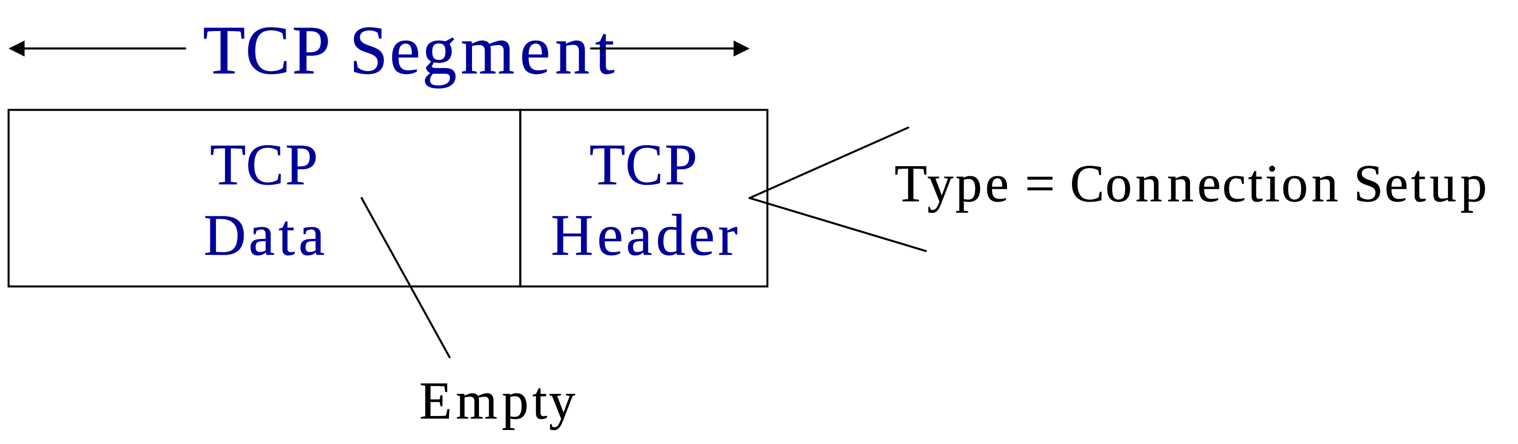
Example Connection: HTTP



At puma, we are firing up a web browser and going to www.cnn.com

In the sending host, puma

1. **Application-Programming Interface (API)**
   Application (Firefox) forms requests TCP connection with the web server at www.cnn.com
2. **Transmission Control Protocol (TCP)**
   Creates TCP “Connection setup” packet
   TCP requests IP packet to be sent to “B”

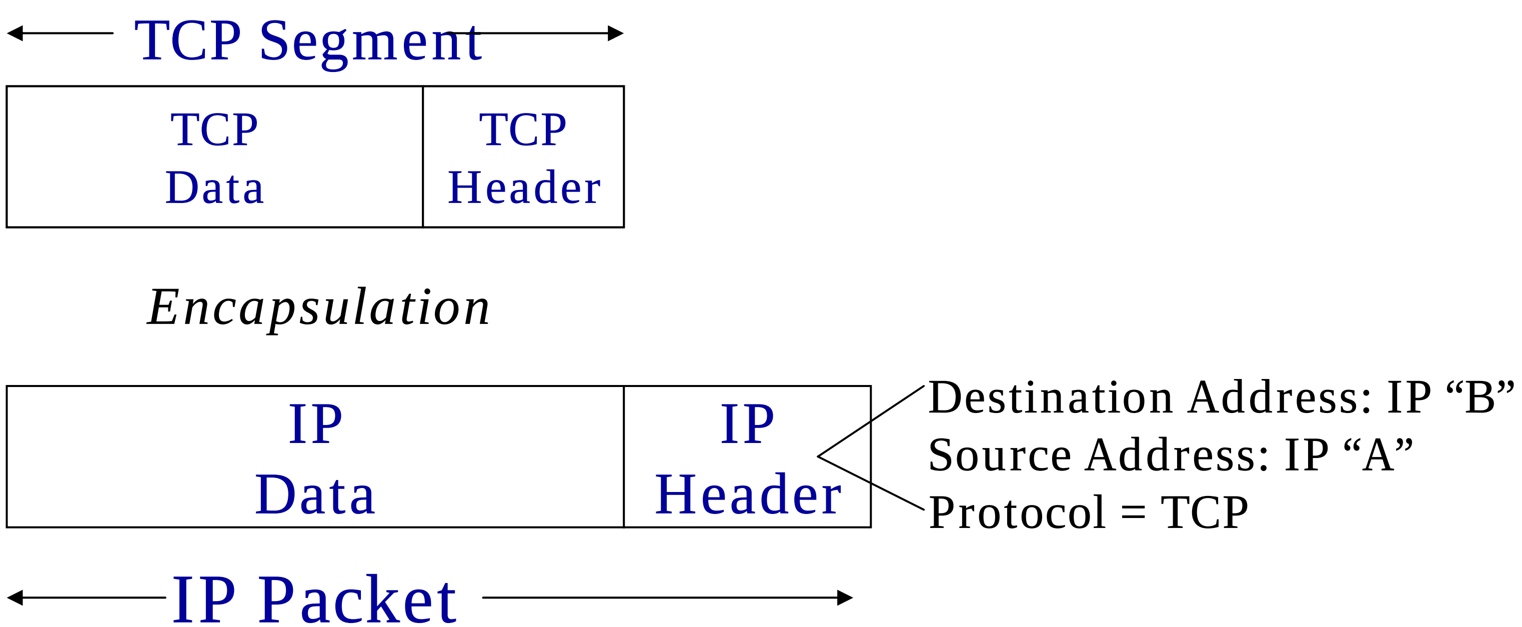


In the sending host, puma

3. Internet Protocol (IP)

Creates IP packet with correct addresses.

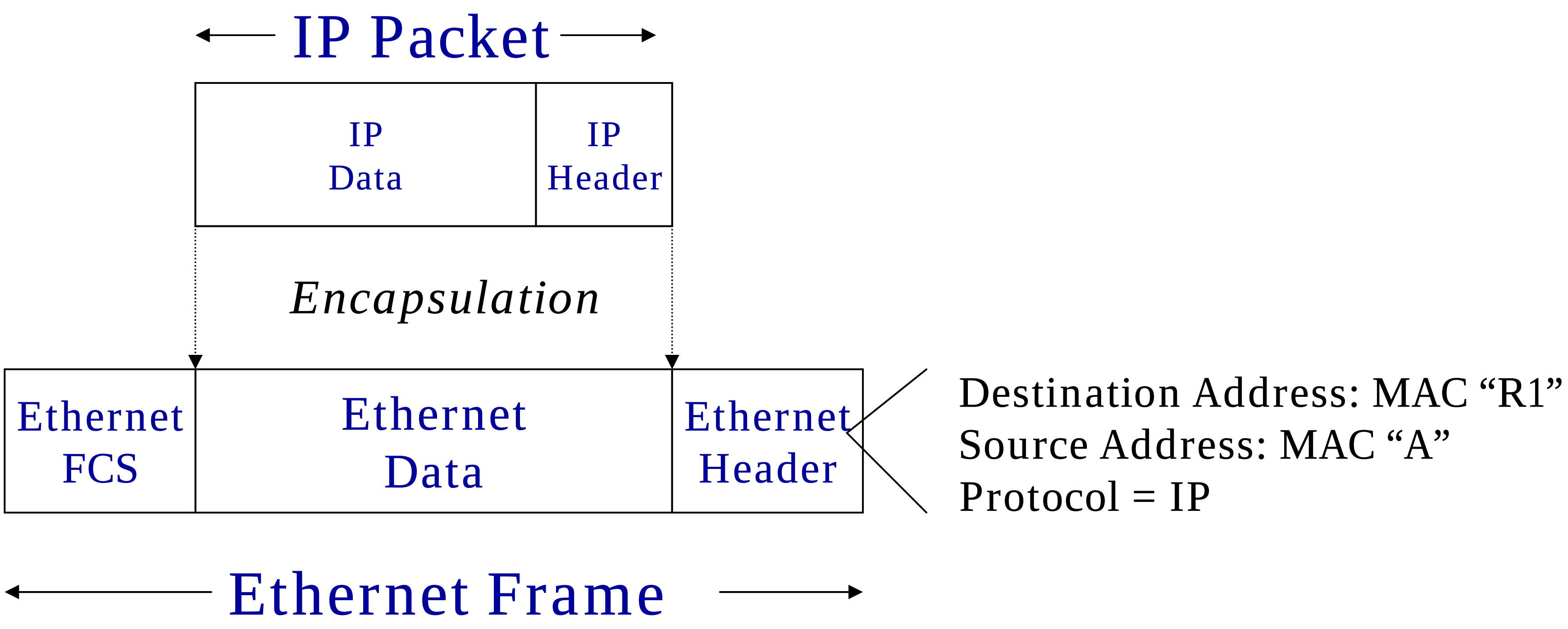
IP requests packet to be sent to router.



In the sending host, puma

4. Link ("MAC" or Ethernet) Protocol

Creates MAC frame with Frame Check Sequence (FCS).
Wait for Access to the line.
MAC requests PHY to send each bit of the frame.

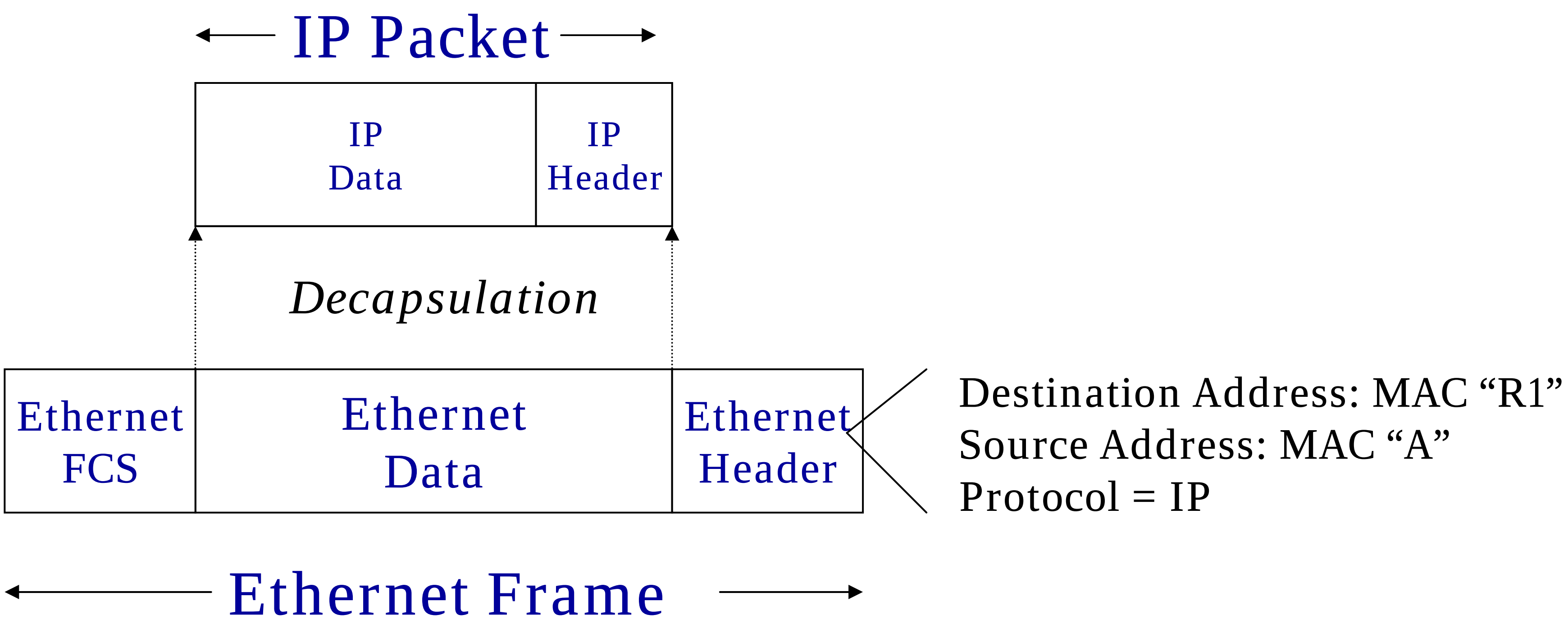


In Router R1

5. Link ("MAC" or Ethernet) Protocol

Accept MAC frame, check address and Frame Check Sequence (FCS) to ensure no bit errors.

Pass data to IP Protocol.

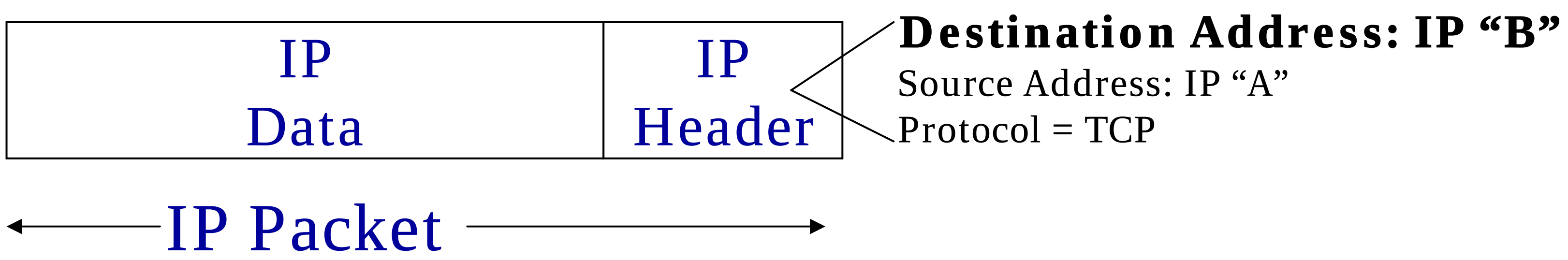


In Router R1

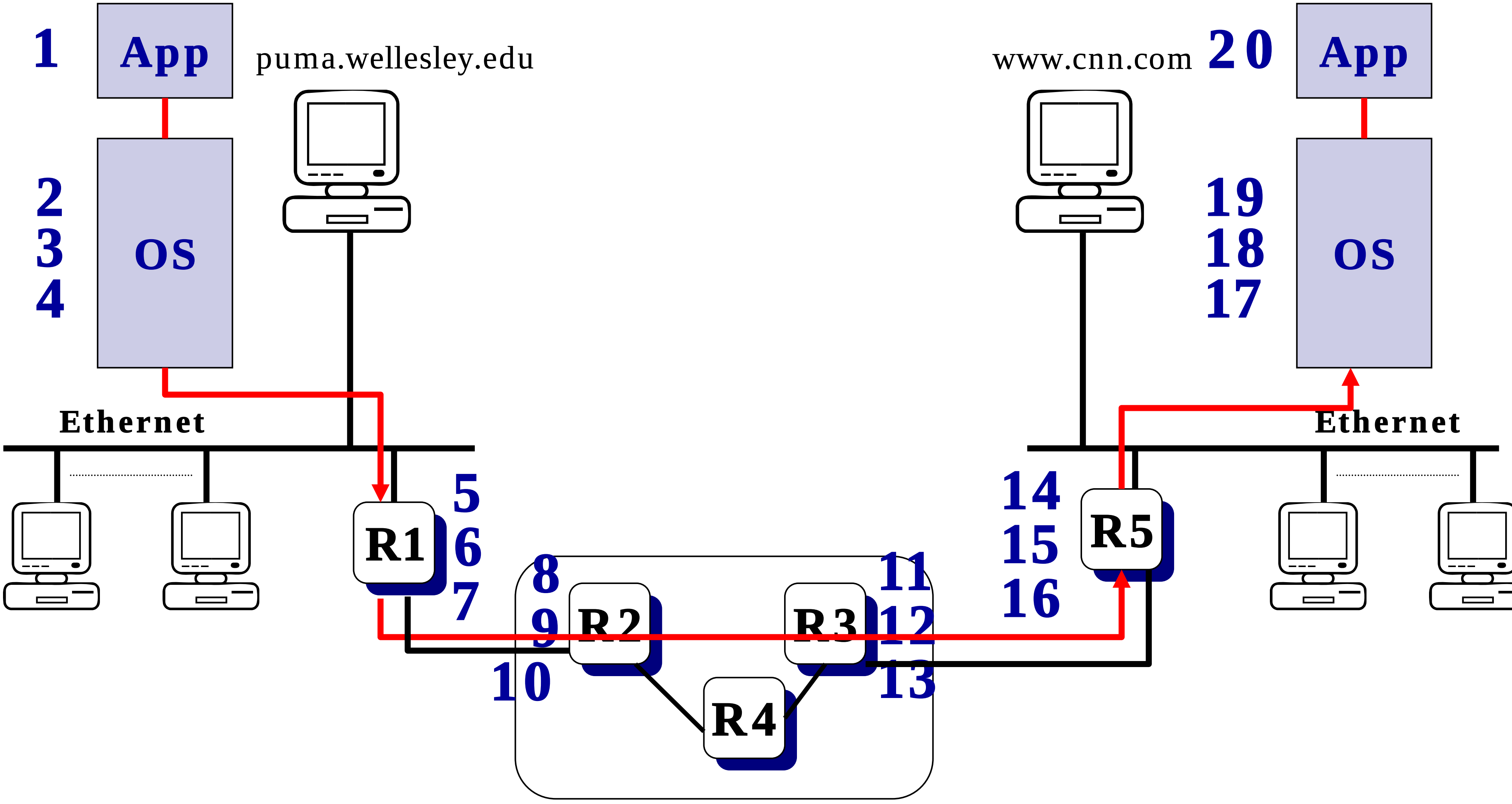
6. Internet Protocol (IP)

Use IP destination address to decide where to send packet next (“next-hop routing”).

Request Link Protocol to transmit packet.



Example Connection: HTTP

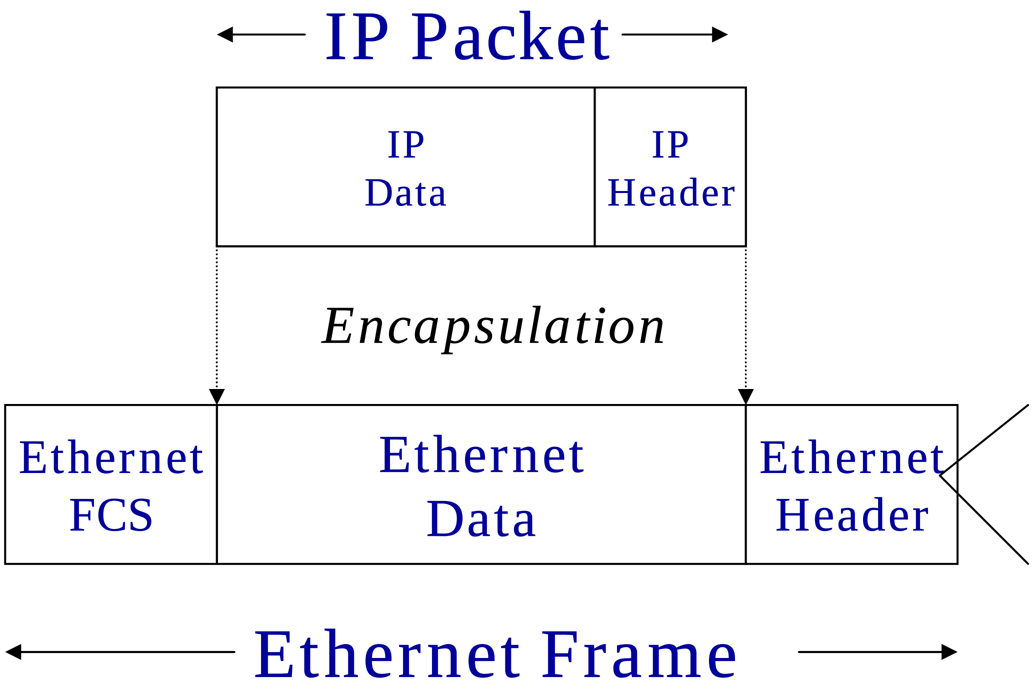


At puma we are firing up a web browser and going to www.cnn.com

In Router R1

7. Link (“MAC” or Ethernet) Protocol

Creates MAC frame with Frame Check Sequence (FCS).
Wait for Access to the line.
MAC requests PHY to send each bit of the frame.



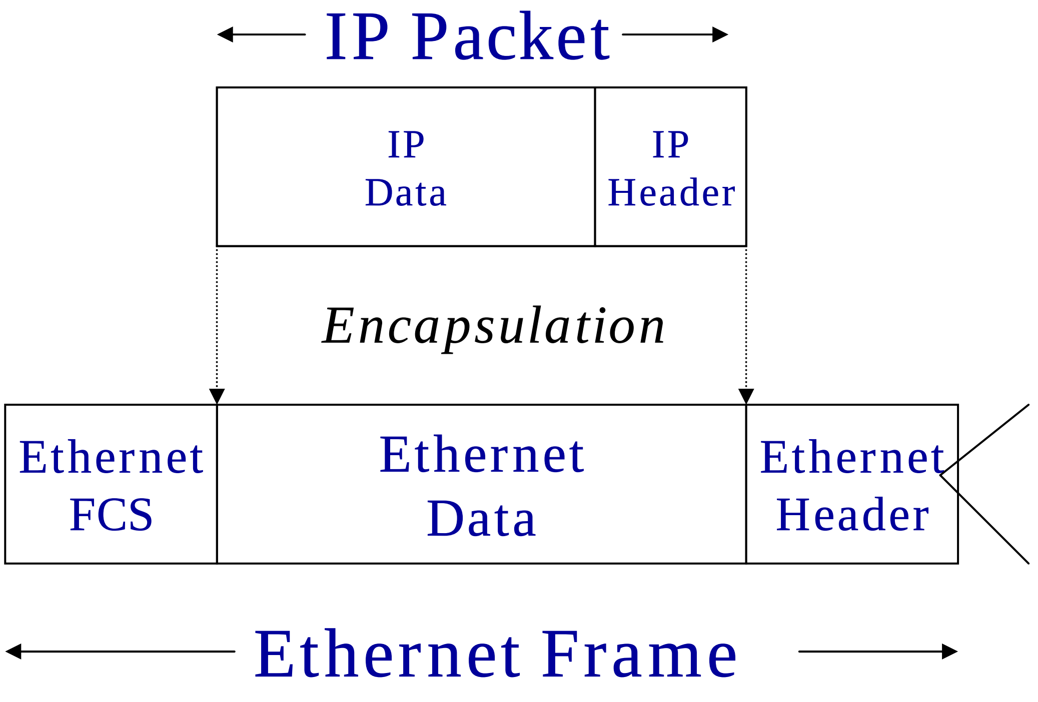
Destination Address: **MAC “R2”**
Source Address: **MAC “R1”**
Protocol = IP

Steps 8-15 are the same as before ...

In Router R5

16. Link ("MAC" or Ethernet) Protocol

Creates MAC frame with Frame Check Sequence (FCS)
Wait for access to the line.
MAC requests PHY to send each bit of the frame



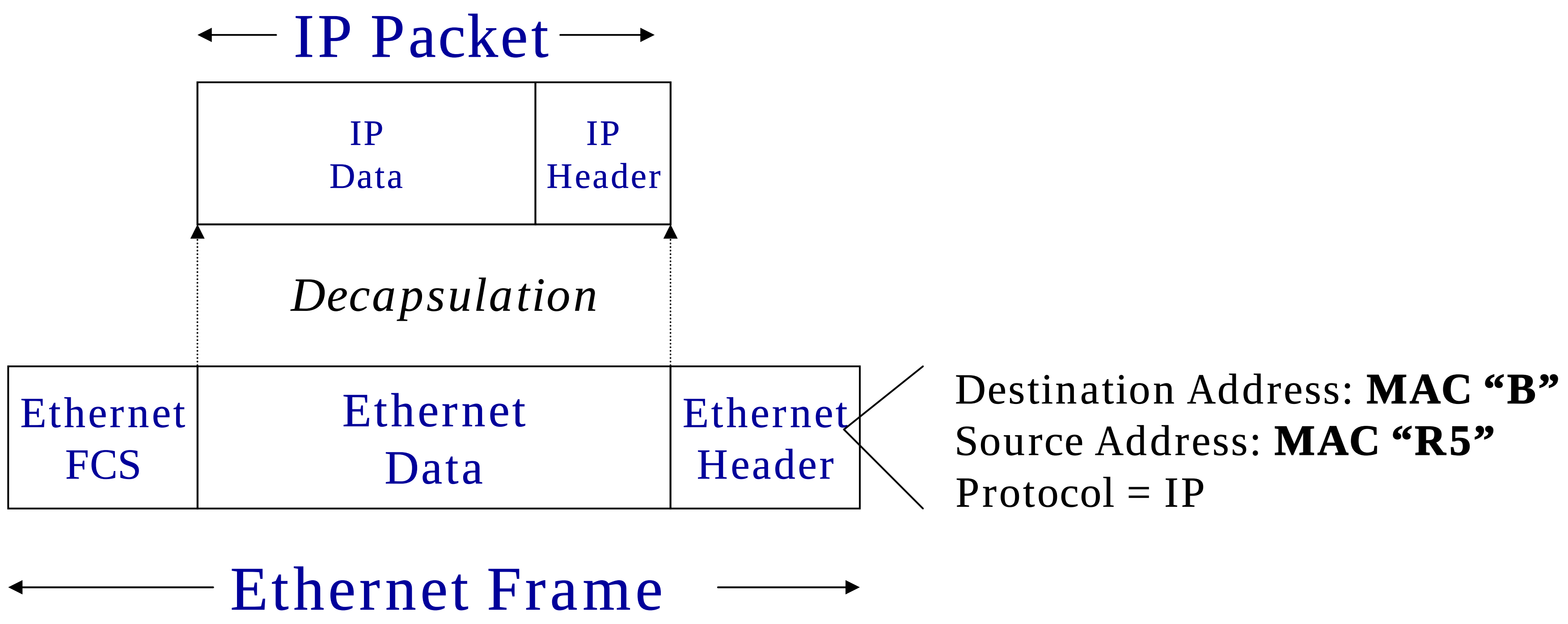
Destination Address: **MAC "B"**
Source Address: **MAC "R5"**
Protocol = IP

Receiving host: Traverse the stack

17. Link ("MAC" or Ethernet) Protocol

Accept MAC frame, check address and Frame Check Sequence (FCS)

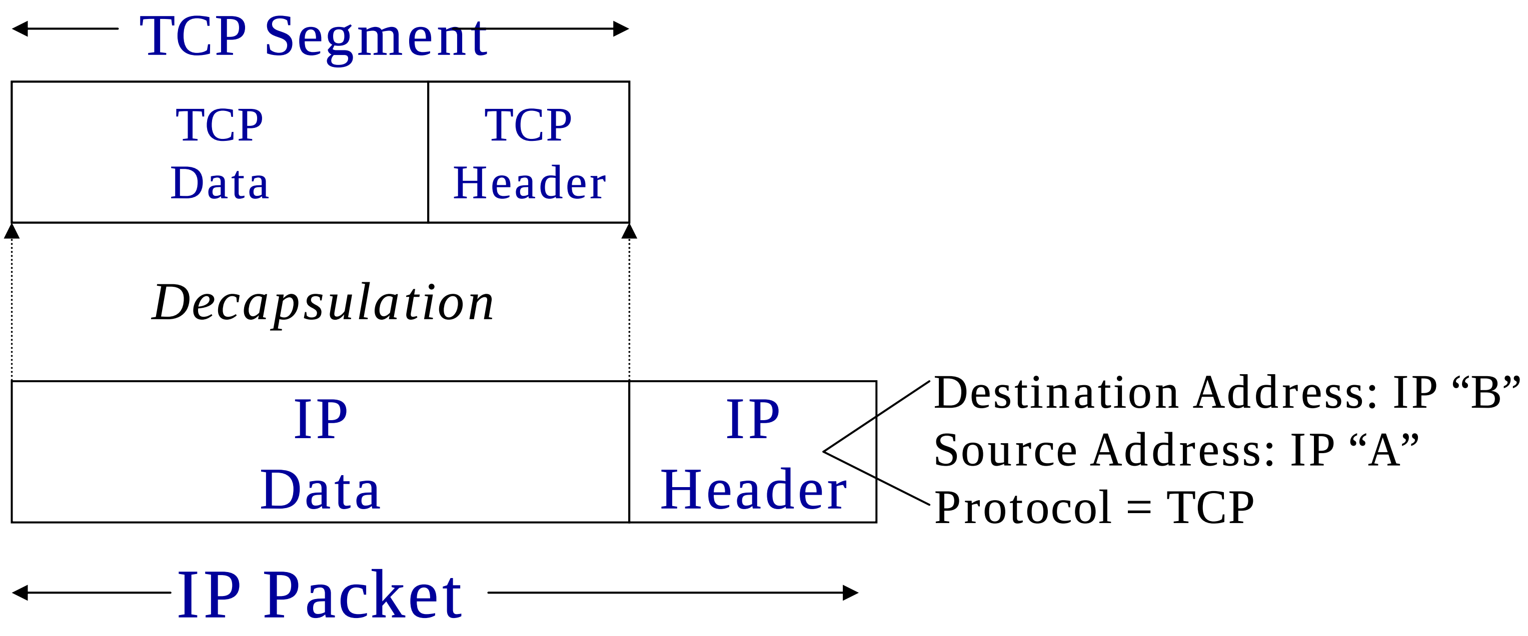
Pass data to IP Protocol



Receiving host: Traverse the Stack

18. Internet Protocol (IP)

Verify IP address.
Extract/decapsulate TCP packet from IP packet.
Pass TCP packet to TCP Protocol



Receiving host: Traverse the Stack

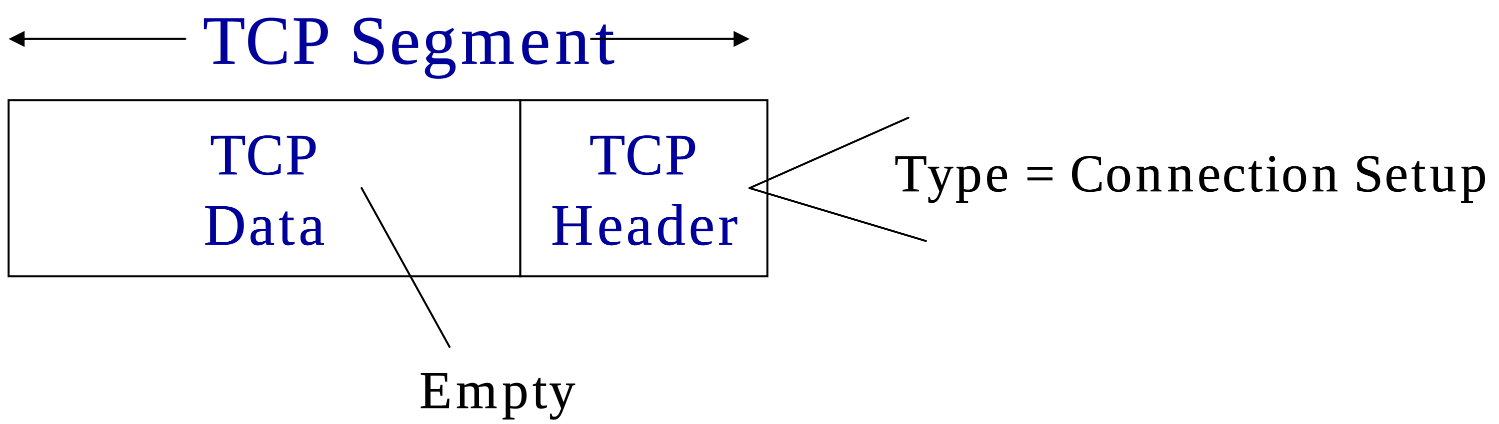
19. Transmission Control Protocol (TCP)

Accepts TCP “Connection setup” packet

Establishes connection by sending “Ack”.

20. Application-Programming Interface (API)

Application receives request for TCP connection with puma.wellesley.edu



Summary

- IP is the protocol that every participant on the Internet has to understand
- Each layer of the TCP/IP protocol stack has its defined responsibilities
  - Provides a service to the layer above
  - Communicates with same layer on peer host

For next time

- Read Tannenbaum's intro to networks for review (on website)
- Try ethereal or wireshark to get a taste for traffic on a network, e.g.
  - Observe what you see when you don't do anything – try to identify or guess what that traffic is about
  - Go to a web page (HTTP, HTTPS)
  - Check your mail with FirstClass

Additional material

- CIDR addressing
- TCP Flow control
- UDP,TCP,IP header layout

Classless Interdomain Routing

- CIDR, pronounced “cider”
- Arbitrary Split Between Network & Host IDs

  Specify either by mask or prefix length

  1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

network	**host**

  E.g., Wellesley can be specified as

  149.130.0.0 with netmask 255.255.0.0

  149.130.0.0/16
- Used to aggregate and split network entries as needed

Aggregation with CIDR

- Original Use: Aggregate Class C Addresses
- One organization assigned contiguous range of class C's
  - e.g., Microsoft given all addresses 207.46.192.X -- 207.46.255.X
  - Specify as CIDR address 207.46.192.0/18

0	8	16	24 31	
207	**46**	**192**	**0**	**Decimal**
cf	**2e**	**c0**	**00**	**Hexadecimal**
1100 1111	**0010 1110**	**11xx xxxx**	**xxxx xxxx**	**Binary**

Upper 18 bits frozen **Lower 14 bits arbitrary**

  - Represents $2^6 = 64$ class C networks
- Use single entry in routing table
  - Just as if were single network address

Important Concepts

- Datagram service model
- Hierarchical addressing critical for scalable system
  - Don’t require everyone to know everyone else
  - Reduces amount of updating when something changes
- Non-uniform hierarchy useful for heterogeneous networks
  - Class-based addressing too coarse
  - CIDR helps
  - Move to IPv6 due to limited number of 32-bit addresses
- Implementation Challenge
  - Longest prefix matching much more difficult than when no ambiguity

TCP Flow Control

- Sliding window protocol
  - For window size *n*, can send up to *n* bytes without receiving an acknowledgement
  - When the data are acknowledged then the window slides forward
- Window size determines
  - How much unacknowledged data can the sender sends

TCP Flow control: Complication

- TCP receiver can delete acknowledged data only after the data has been delivered to the application

- So, depending on how fast the application is reading the data, the receiver’s window size may change!!!

TCP Flow control: Solution

- Receiver tells sender what is the current window size in every packet it transmits to the sender
- Sender uses this current window size instead of a fixed value

- Window size (also called Advertised window) is continuously changing
- Can go to zero!

  Sender not allowed to send anything!

IP Header

```
 0               1               3               4
 0 1 2 3 4 5 6 7 8 9 A B C D E F 0 1 2 3 4 5 6 7 8 9 A B C D E F
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version|  HL   |Type of Service|          Total Length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Identification        |Flags|      Fragment Offset    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Time to Live |    Protocol   |         Header Checksum       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Source Address                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Destination Address                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Options                    |    Padding    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IP Header

- Version: 4 bits
  - helps smooth the transition to future version of IP
- Header length: 4 bits
  - limits the header to 15 * 32bits = 60 bytes
- Type of Service: 4 bits
  - Specify a tradeoff between fast service and reliable service, not commonly used
- Total length: 16 bits
  - Limits each packet to 64K bytes
- Time-To-Live (TTL): 8 bits
  - limit the life of the packet on the network
    - Initialized to thirty
    - Decremented each time the packet arrives at a routing step
    - Discarded when it is equal to 0
- Identification (16 bits), Flags (3 bits), and Fragment Offset (13 bits)
  - Partition a datagram into packet if it is too large
    - Each packet must be no larger than 64K
    - The maximum number of fragments per datagram is 8192

UDP Header

```
 0      7 8     15 16    23 24    31
+--------+--------+--------+--------+
|     Source      |   Destination   |
|      Port       |      Port       |
+--------+--------+--------+--------+
|                 |                 |
|     Length      |    Checksum     |
+--------+--------+--------+--------+
```

TCP Header

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 a b c d e f 0 1 2 3 4 5 6 7 8 9 a b c d e f
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Source Port          |       Destination Port        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sequence Number                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Acknowledgment Number                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Data |           |U|A|E|R|S|F|                               |
| Offset| Reserved  |R|C|O|S|Y|I|            Window             |
|       |           |G|K|L|T|N|N|                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Checksum            |         Urgent Pointer        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Options                    |    Padding    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             data                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Transport Layer (DOD)

- **Provides End to End connections between two devices by performing sequencing, acknowledgements, checksums and flow control**
- TCP

  Connection-oriented and reliable communications
- UDP

  Non-connection-oriented and unreliable

Transmission Control Protocol (TCP)

- **Segments Application-layer data stream**
  - Provides acknowledgment timers
  - Enables sequence number checking
  - Provides buffer management
  - Initiates connection with three-way handshake
  - Performs error and duplication checking

TCP Header Layout

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Source Port          |       Destination Port        |  0 - 3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sequence Number                        |  4 - 7
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Acknowledgment Number                      |  8 - 11
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Data |           |U|A|P|R|S|F|                               |
| Offset| Reserved  |R|C|S|S|Y|I|            Window             |  12 - 15
|       |           |G|K|H|T|N|N|                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Checksum            |         Urgent Pointer        |  16 - 19
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Options                    |    Padding    |  20 --->
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

RFC 793