Networking Overview: “Everything” you need to know, in 50 minutes

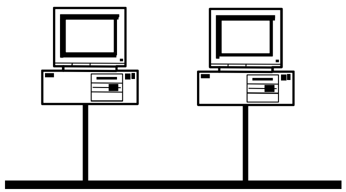
Network Security

Prof. Haojin Zhu

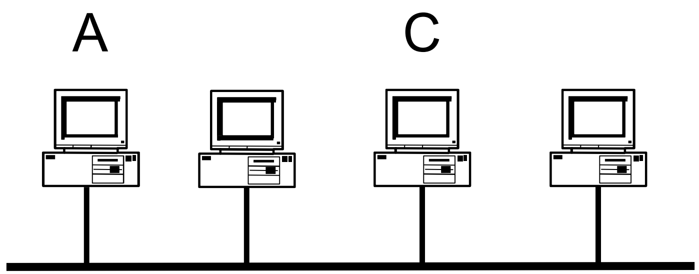
Adopted from David Wagner @ UC Berkeley

May 8, 2019

Local-Area Networks



point-to-point



shared

How does computer A send a message to computer C?

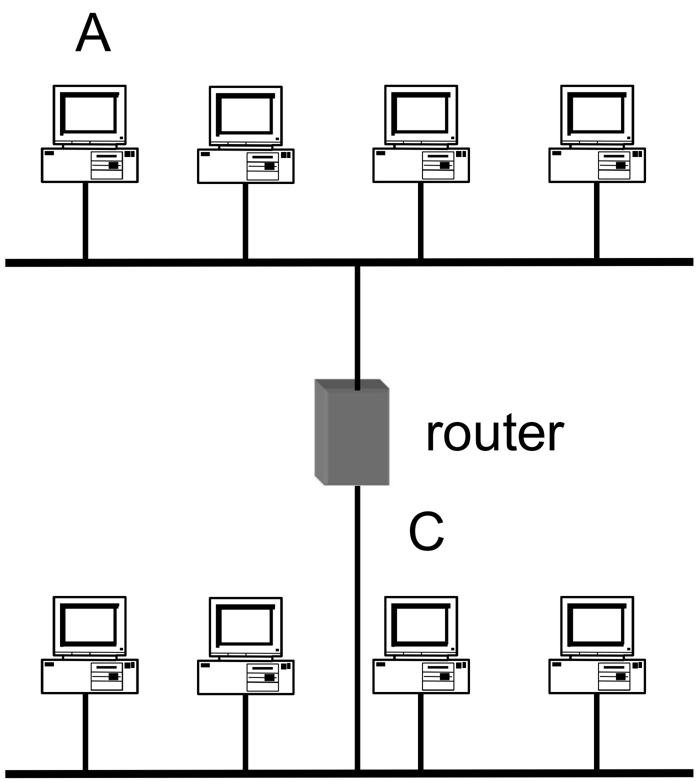
Local-Area Networks: Packets

```
From: A
To: C
Message: Hello world!
```

A	C	Hello world!

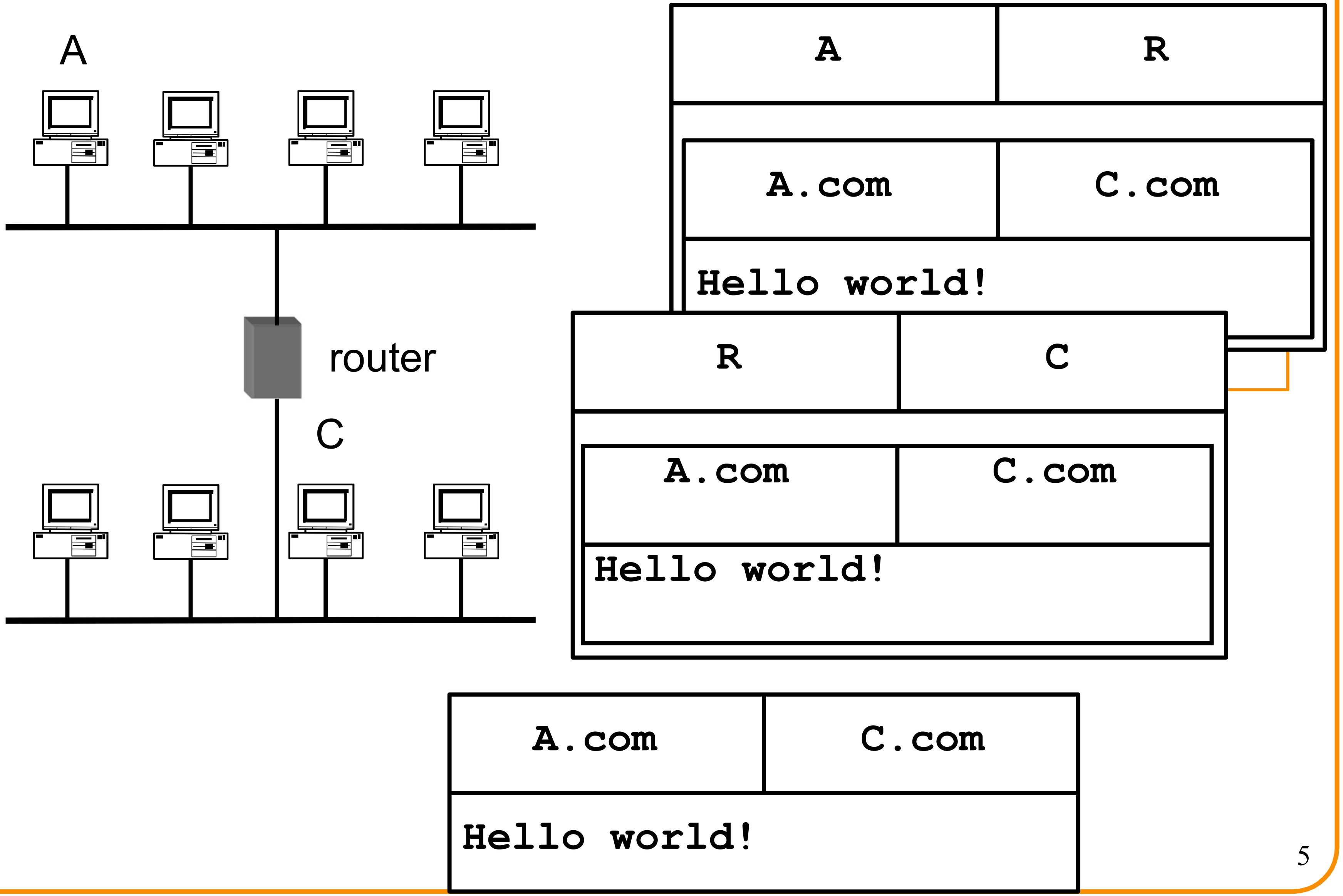
A	C
Hello world!	

Wide-Area Networks



How do we connect two LANs?

Wide-Area Networks



Key Concept #1: *Protocols*

- A protocol is an agreement on how to communicate
- Includes syntax and semantics
  - How a communication is specified & structured
    - Format, order messages are sent and received
  - What a communication means
    - Actions taken when transmitting, receiving, or timer expires
- Example: making a comment in lecture?
  1. Raise your hand.
  2. Wait to be called on.
  3. Or: wait for speaker to **pause** and vocalize
  4. If unrecognized (after timeout): say “excuse me”

Key Concept #2: *Dumb Network*

- Original Internet design: interior nodes ("routers") have <u>no</u> knowledge* of ongoing connections going through them
- **Not** how you picture the telephone system works
  - Which internally tracks all of the active voice calls
- Instead: the postal system!
  - Each Internet message ("packet") self-contained

* Today's Internet is full of hacks that violate this

Self-Contained IP Packet Format

IP = Internet *Protocol*

4-bit Version	4-bit Header Length	8-bit Type of Service (TOS)	16-bit Total Length (Bytes)
16-bit Identification		3-bit Flags	13-bit Fragment Offset
8-bit Time to Live (TTL)	8-bit Protocol	16-bit Header Checksum	
32-bit Source IP Address			
32-bit Destination IP Address			
Payload (remainder of message)			

Header **is like a letter envelope: contains all info needed for delivery**

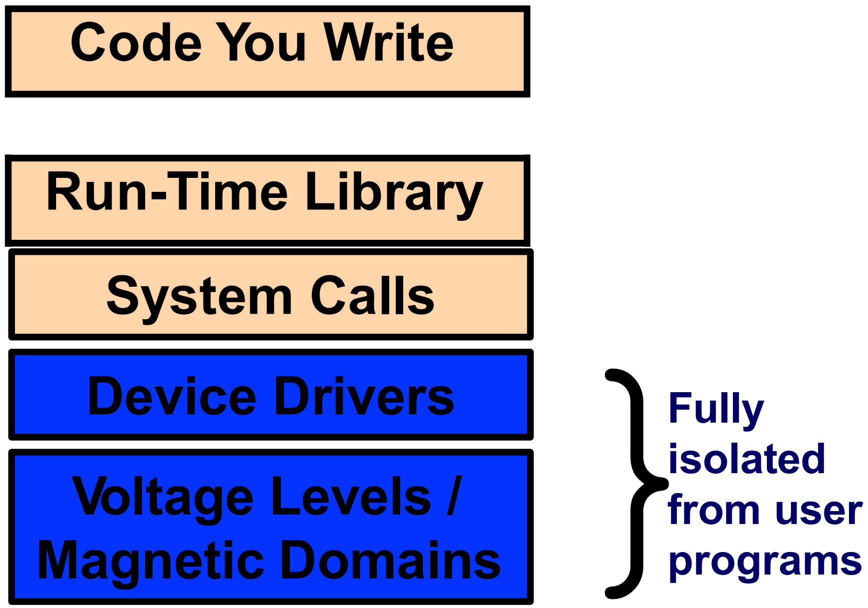
Key Concept #2: *Dumb Network*

- Original Internet design: interior nodes ("routers") have <u>no</u> knowledge* of ongoing connections going through them
- **Not**: how you picture the telephone system works
  - Which internally tracks all of the active voice calls
- Instead: the postal system!
  - Each Internet message ("packet") self-contained
  - Interior routers look at destination address to forward
  - If you want smarts, build it "*end-to-end*", not "hop-by-hop"
  - Buys simplicity & robustness at the cost of shifting complexity into end systems

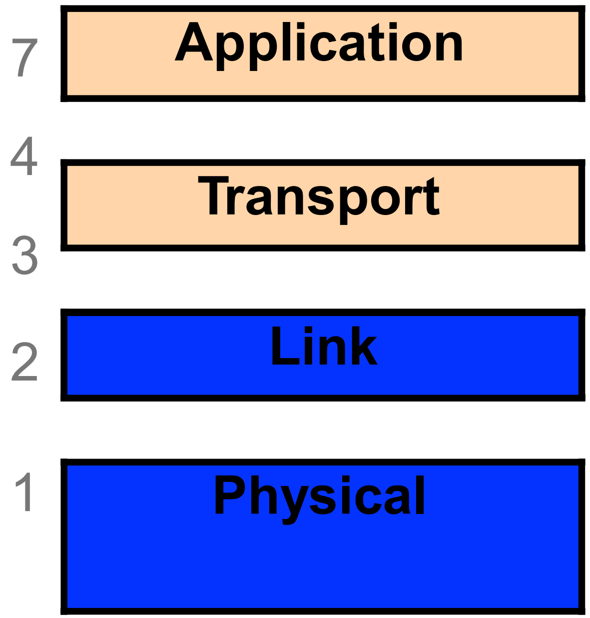
* Today's Internet is full of hacks that violate this

Key Concept #3: *Layering*

- Internet design is strongly partitioned into layers
  - Each layer relies on services provided by next layer below …
  - … and provides services to layer above it
- Analogy:
  - Consider structure of an application you've written and the "services" each layer relies on / provides



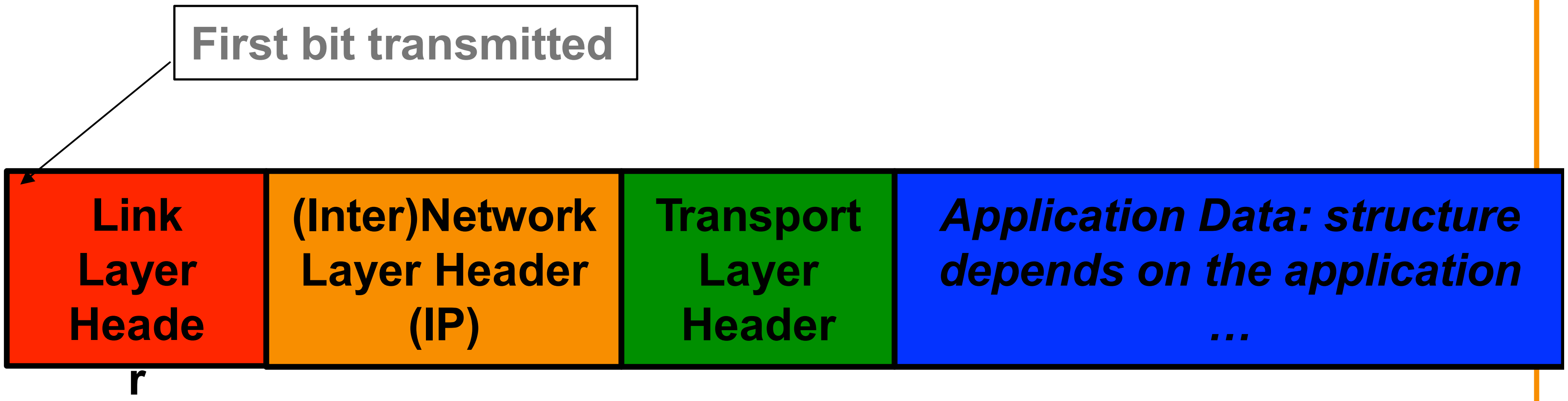
Internet Layering (“Protocol Stack”)



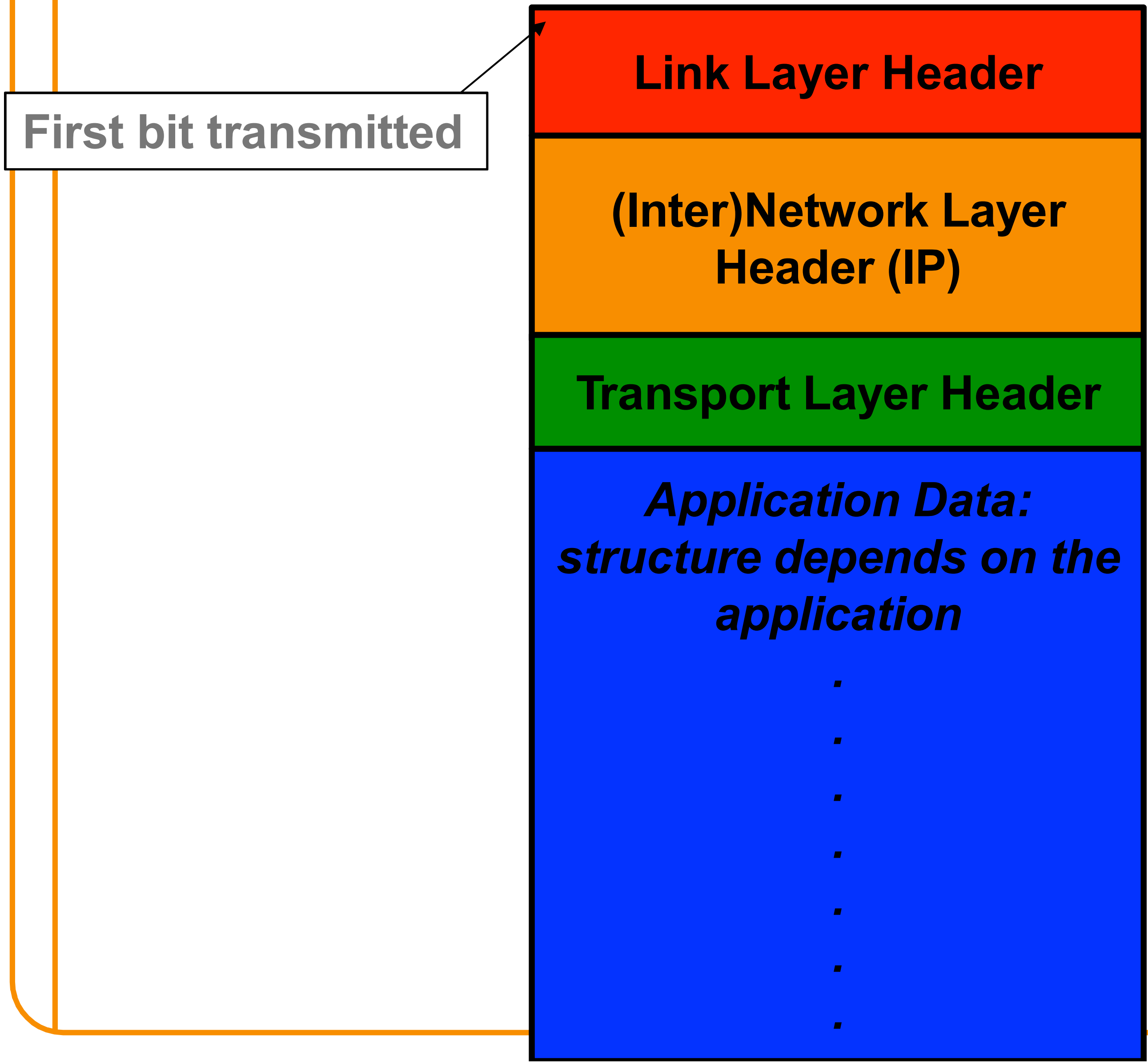
Note on a point of potential confusion: these diagrams are always drawn with lower layers **below** higher layers …

But diagrams showing the layouts of packets are often the *opposite*, with the lower layers at the **top** since their headers <u>precede</u> those for higher layers

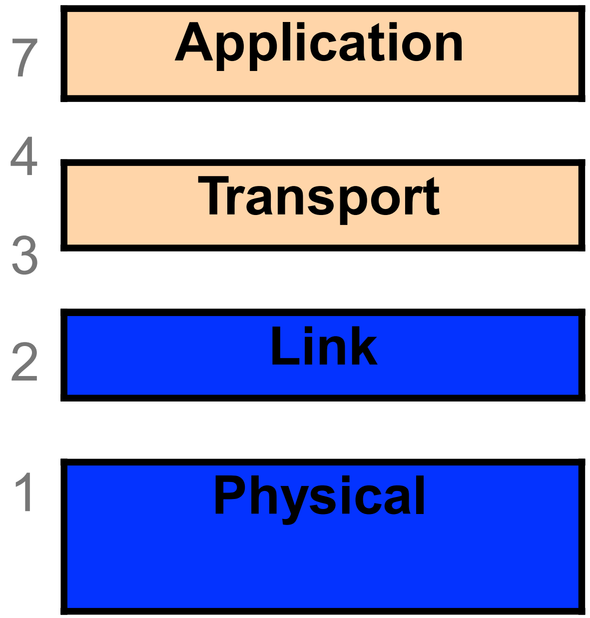
Horizontal View of a Single Packet



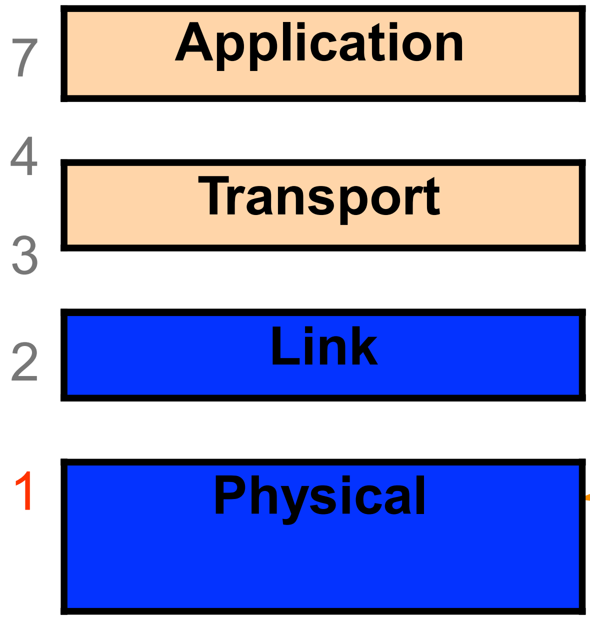
Vertical View of a Single Packet



Internet Layering ("Protocol Stack")

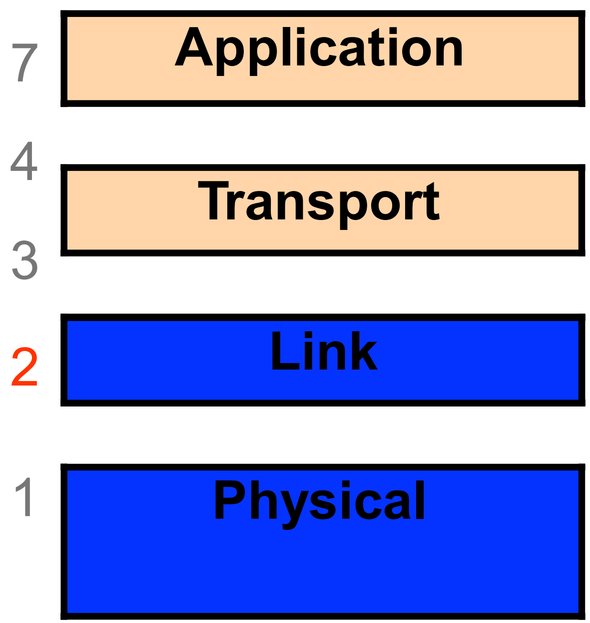


Layer 1: Physical Layer



- Encoding bits to send them over a single **physical link**
  - e.g. patterns of *voltage levels / photon intensities / RF modulation*

Layer 2: Link Layer

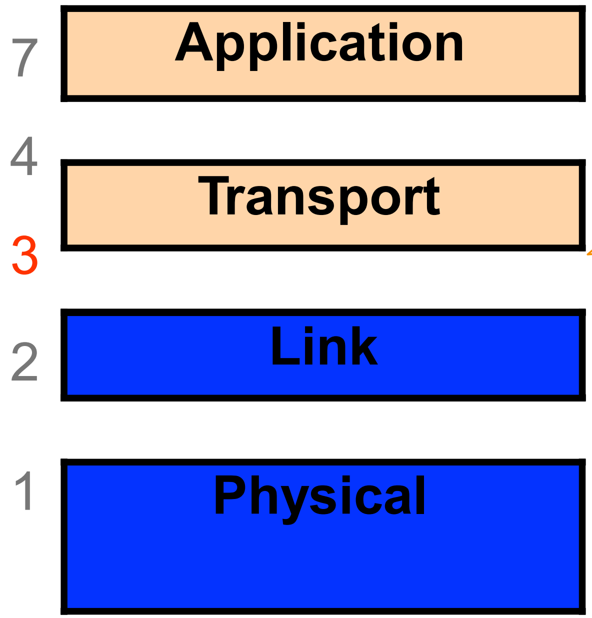


Framing and transmission of a collection of bits into individual messages sent across a single “subnetwork” (one physical technology)

Might involve multiple *physical links* (e.g., modern Ethernet)

Often technology supports broadcast transmission (**every** “node” connected to subnet receives)

Layer 3: (Inter)Network Layer *(IP)*



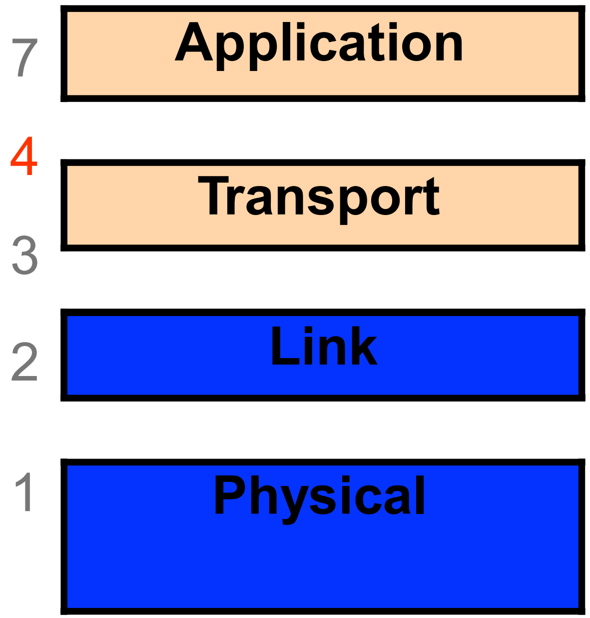
Bridges multiple "subnets" to provide *end-to-end* internet connectivity between nodes

- Provides global addressing

Works across different link technologies

Different **for each Internet "hop"**

Layer 4: Transport Layer

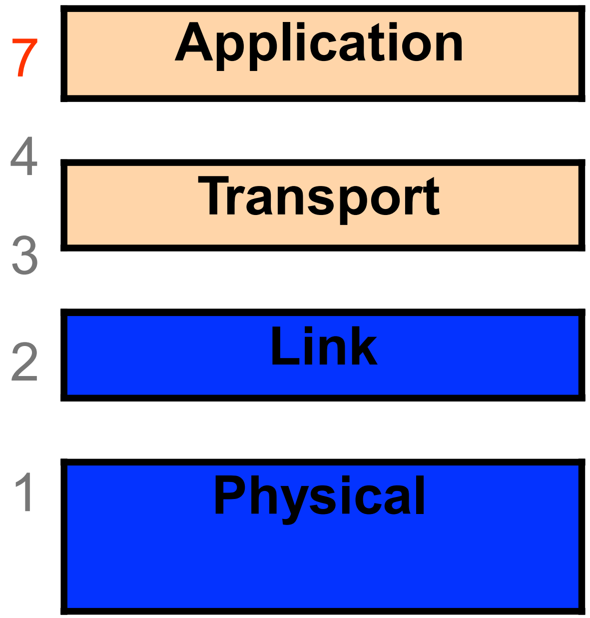


End-to-end communication between processes

Different services provided:
TCP = reliable *byte stream*
UDP = unreliable *datagrams*

(Datagram = single packet message)

Layer 7: Application Layer



Communication of whatever you wish

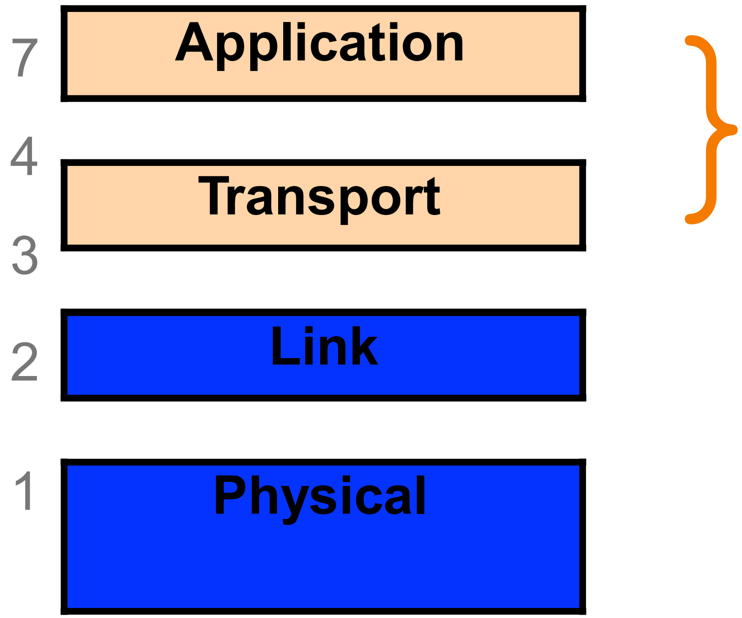
Can use whatever transport(s) is convenient

Freely structured

E.g.:

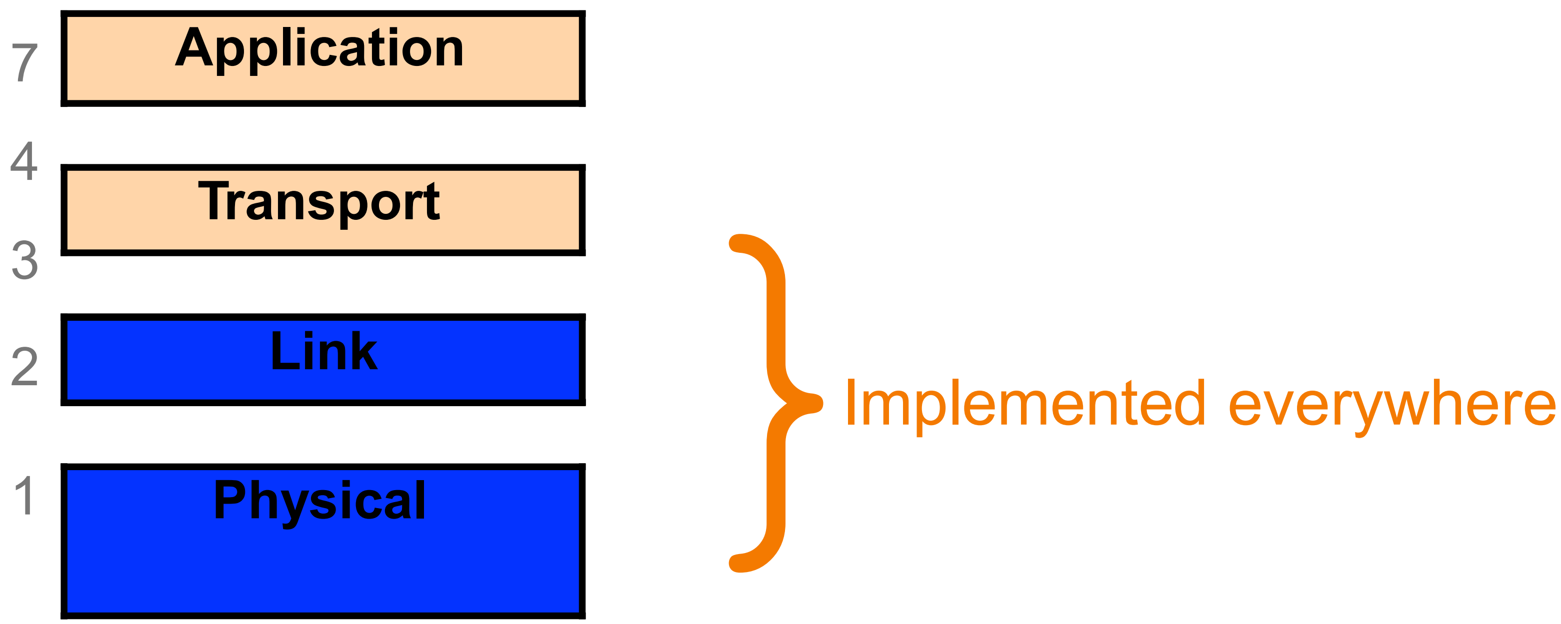
Skype, SMTP (email),
HTTP (Web), Halo, BitTorrent

Internet Layering ("Protocol Stack")

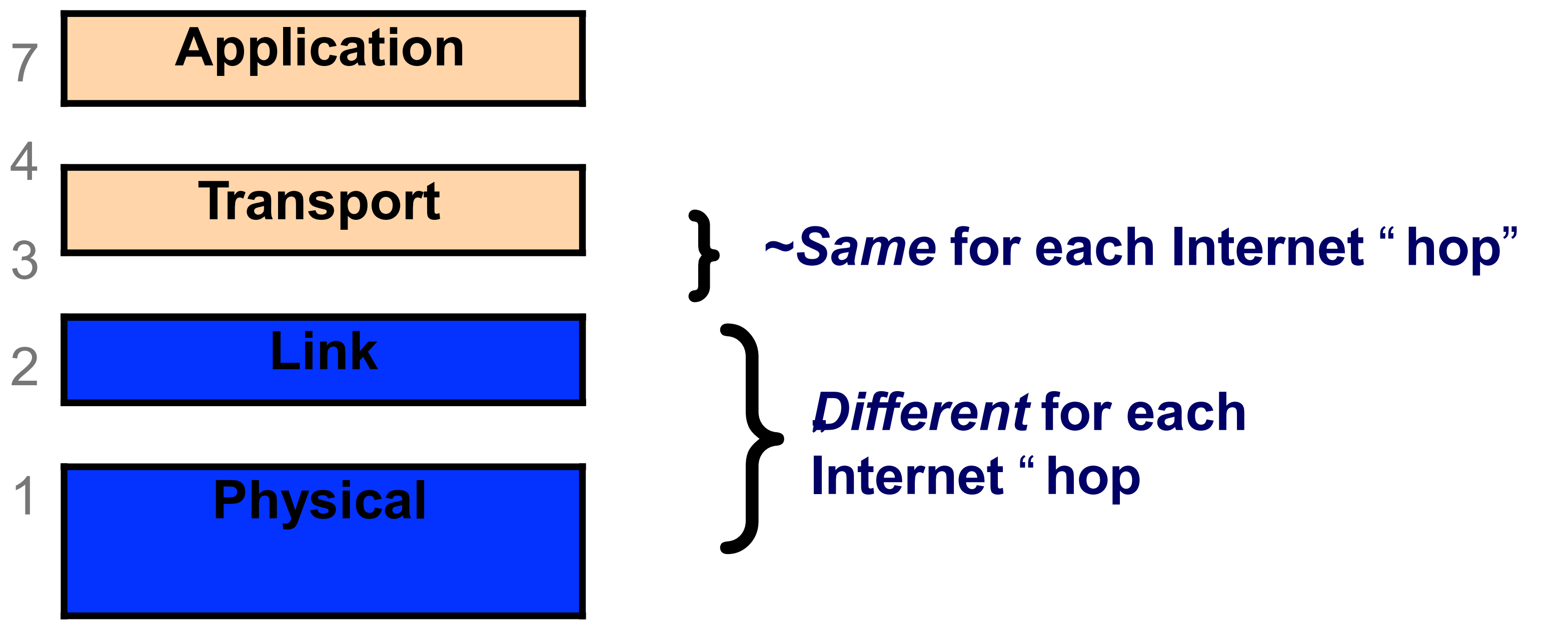


Implemented only at hosts,
not at interior router
("dumb network")

Internet Layering ("Protocol Stack")

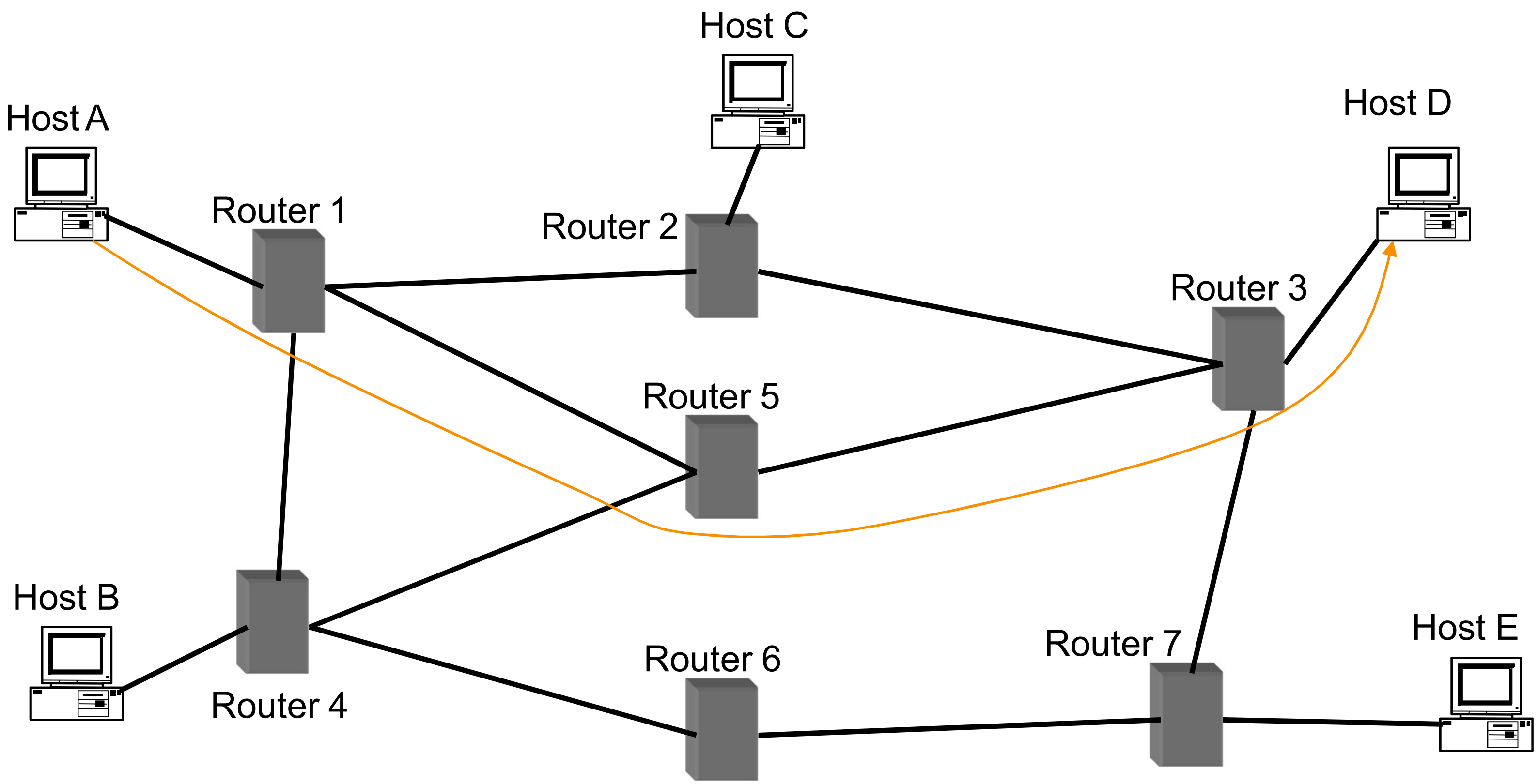


Internet Layering ("Protocol Stack")



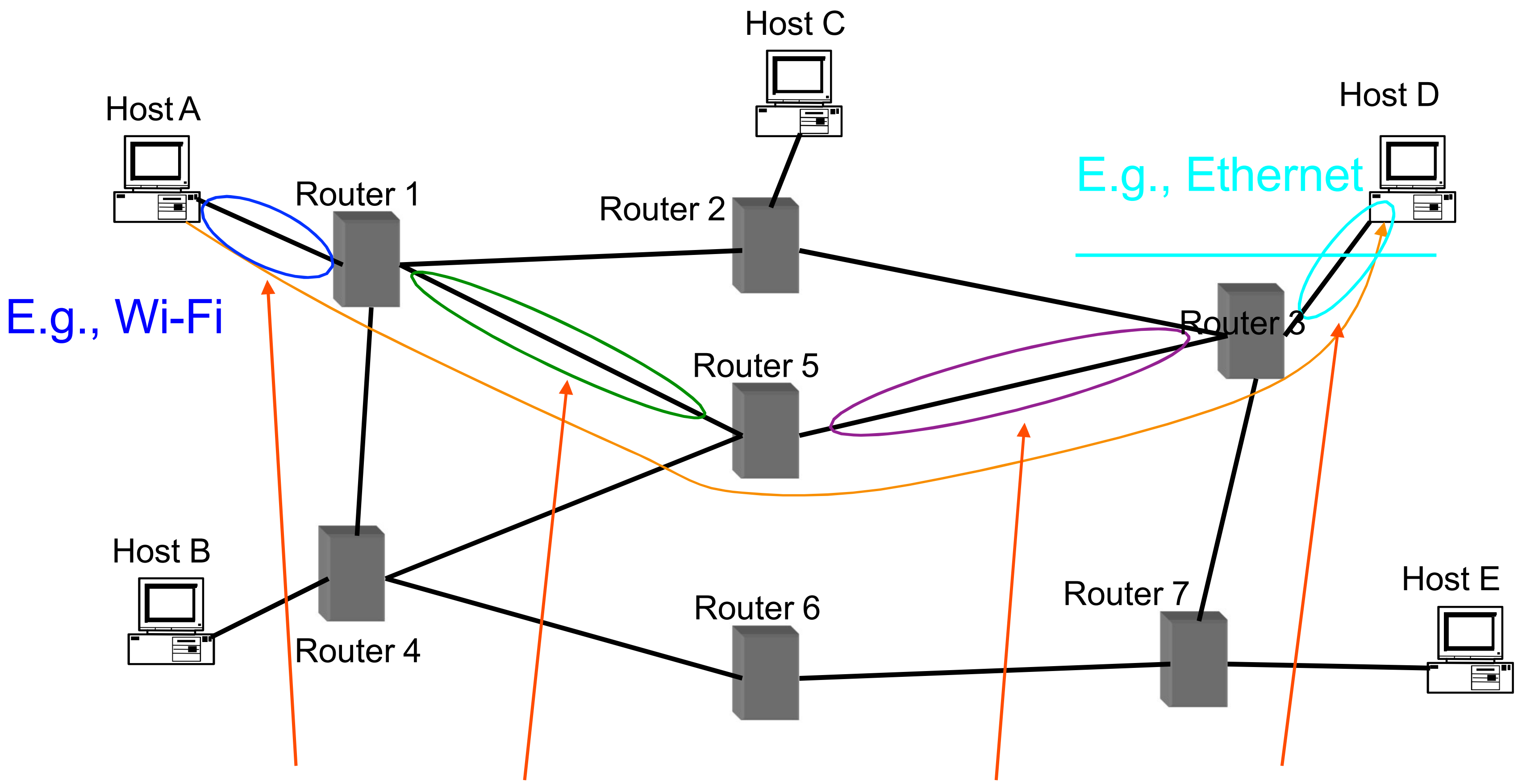
Hop-By-Hop vs. End-to-End Layers

Host A communicates with Host D



Hop-By-Hop vs. End-to-End Layers

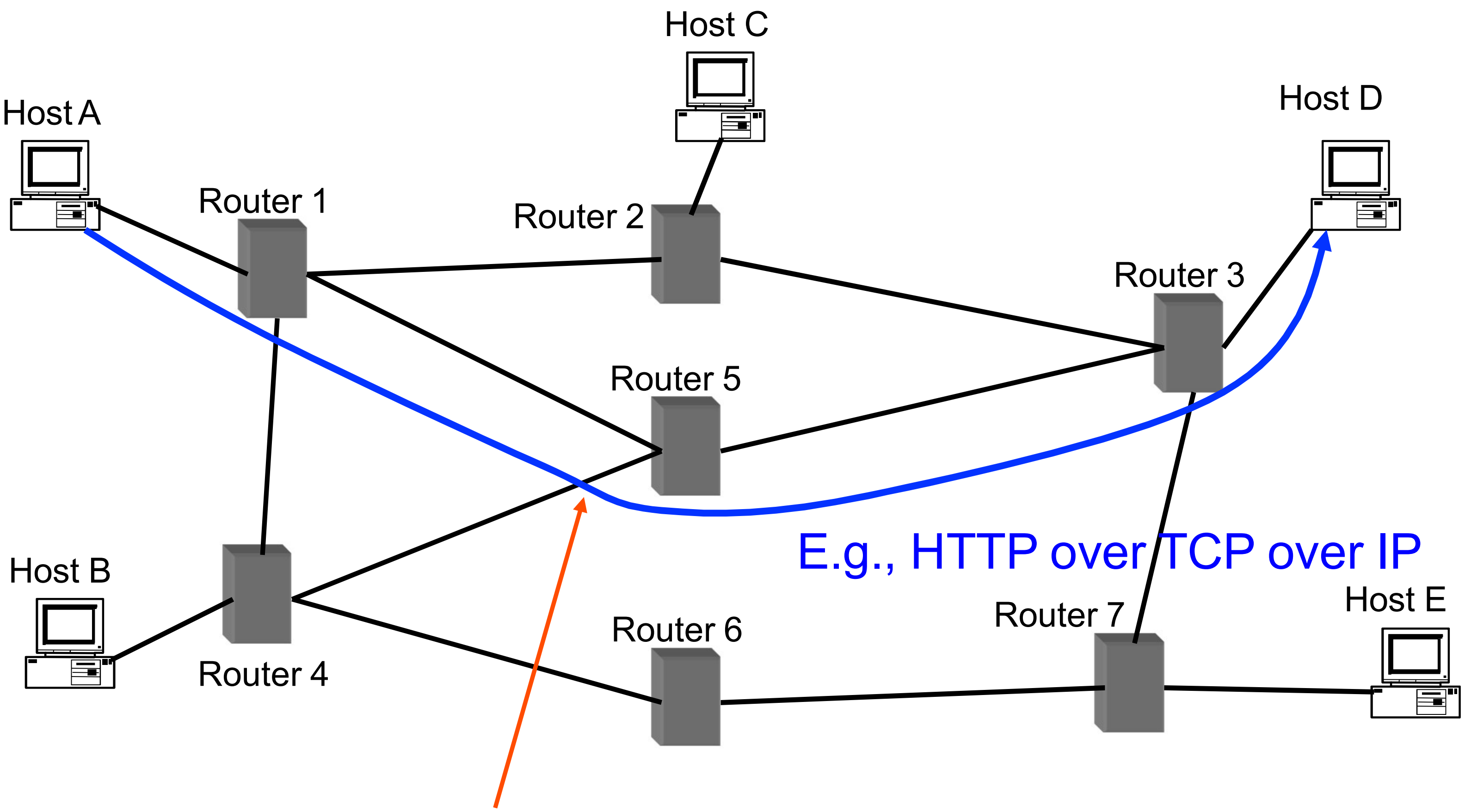
Host A communicates with Host D



***Different* Physical & Link Layers (Layers 1 & 2)**

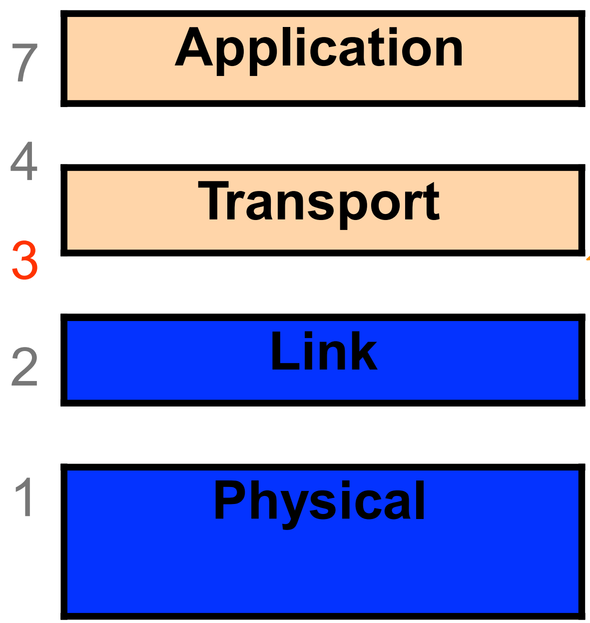
Hop-By-Hop vs. End-to-End Layers

Host A communicates with Host D



Same Network / Transport / Application Layers (3/4/7)
(Routers **ignore** Transport & Application layers)

Layer 3: (Inter)Network Layer *(IP)*

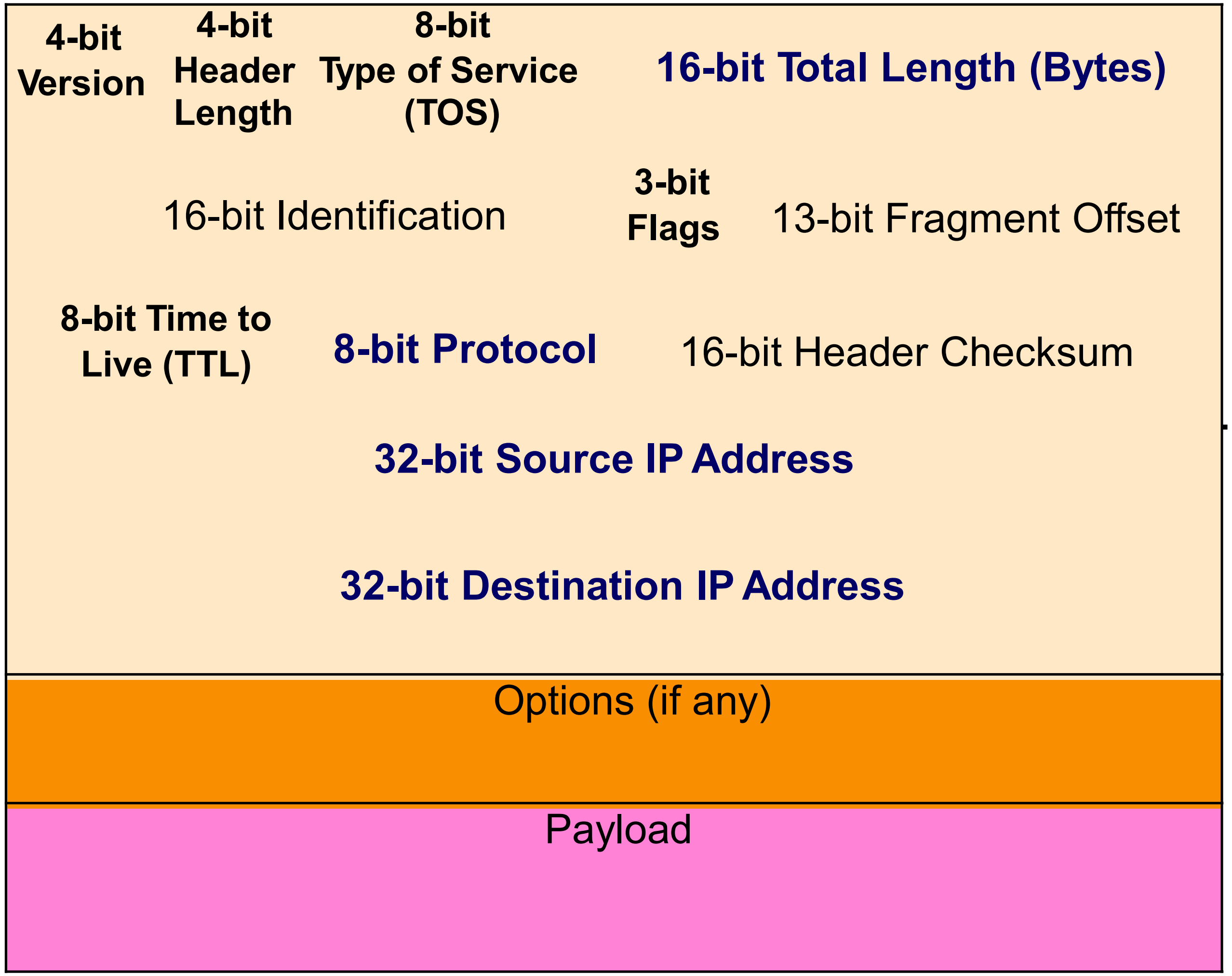


Bridges multiple "s u b n e ts" to provide *end-to-end* internet connectivity between nodes

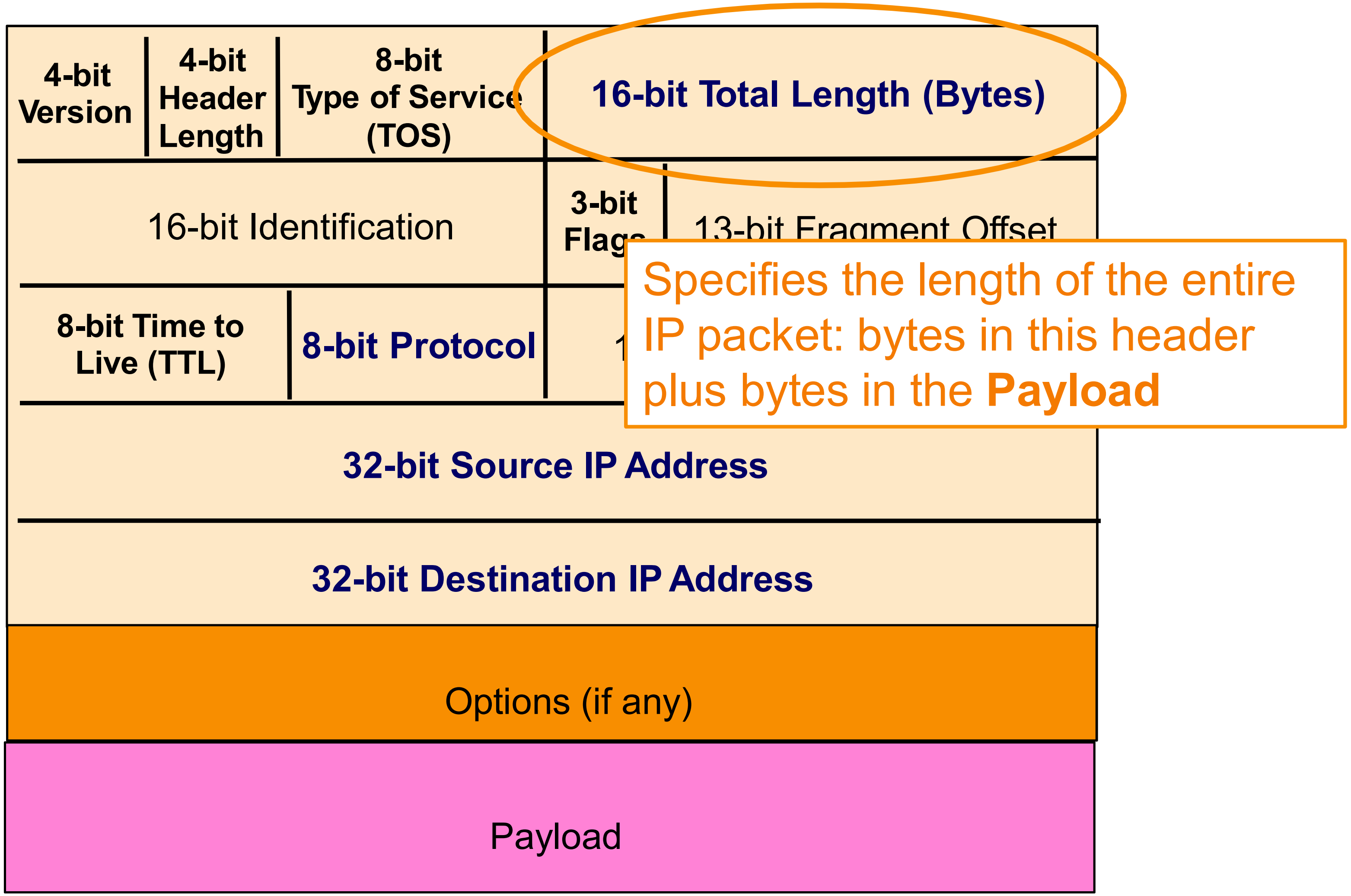
- Provides global addressing

Works across different link technologies

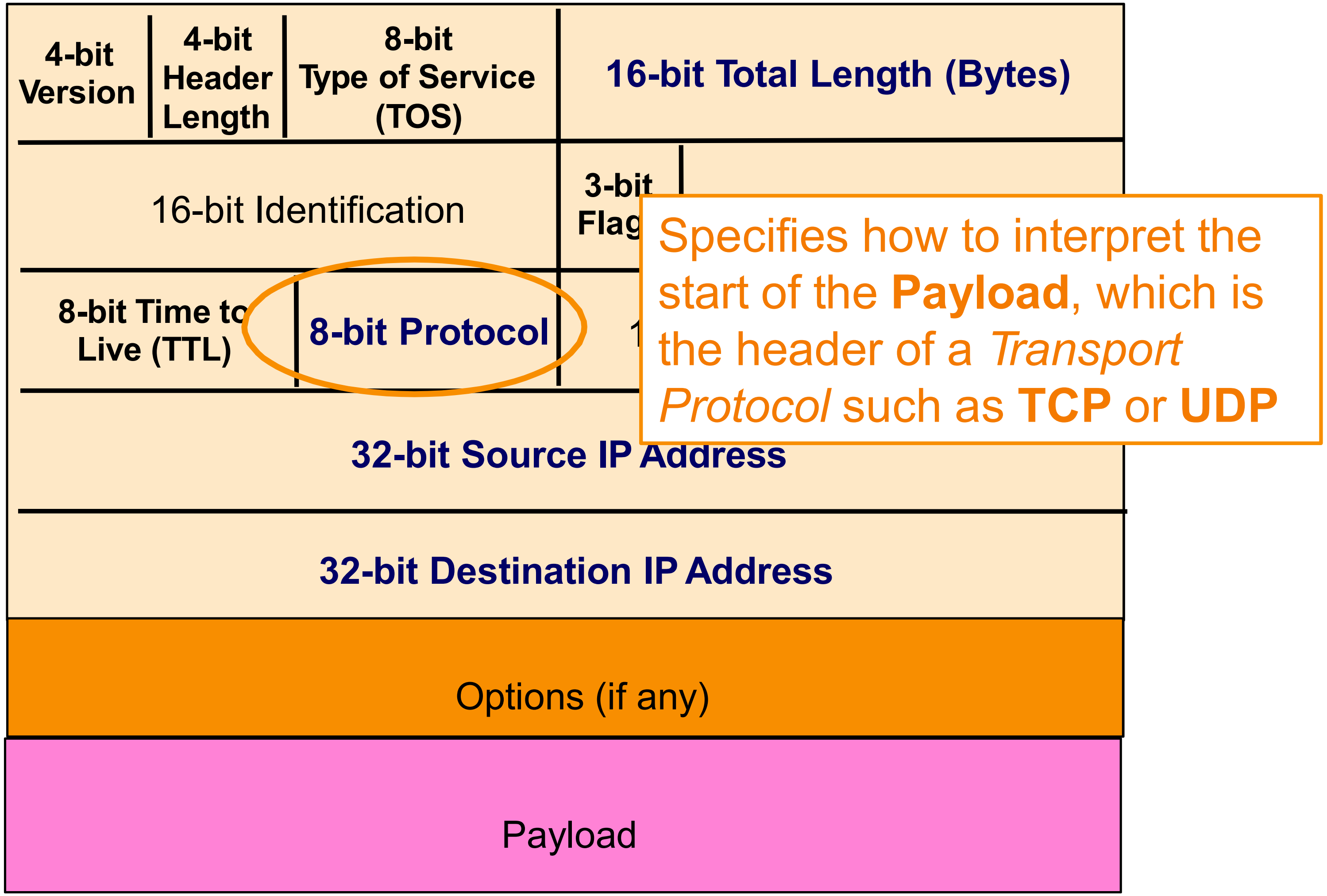
IP Packet Structure



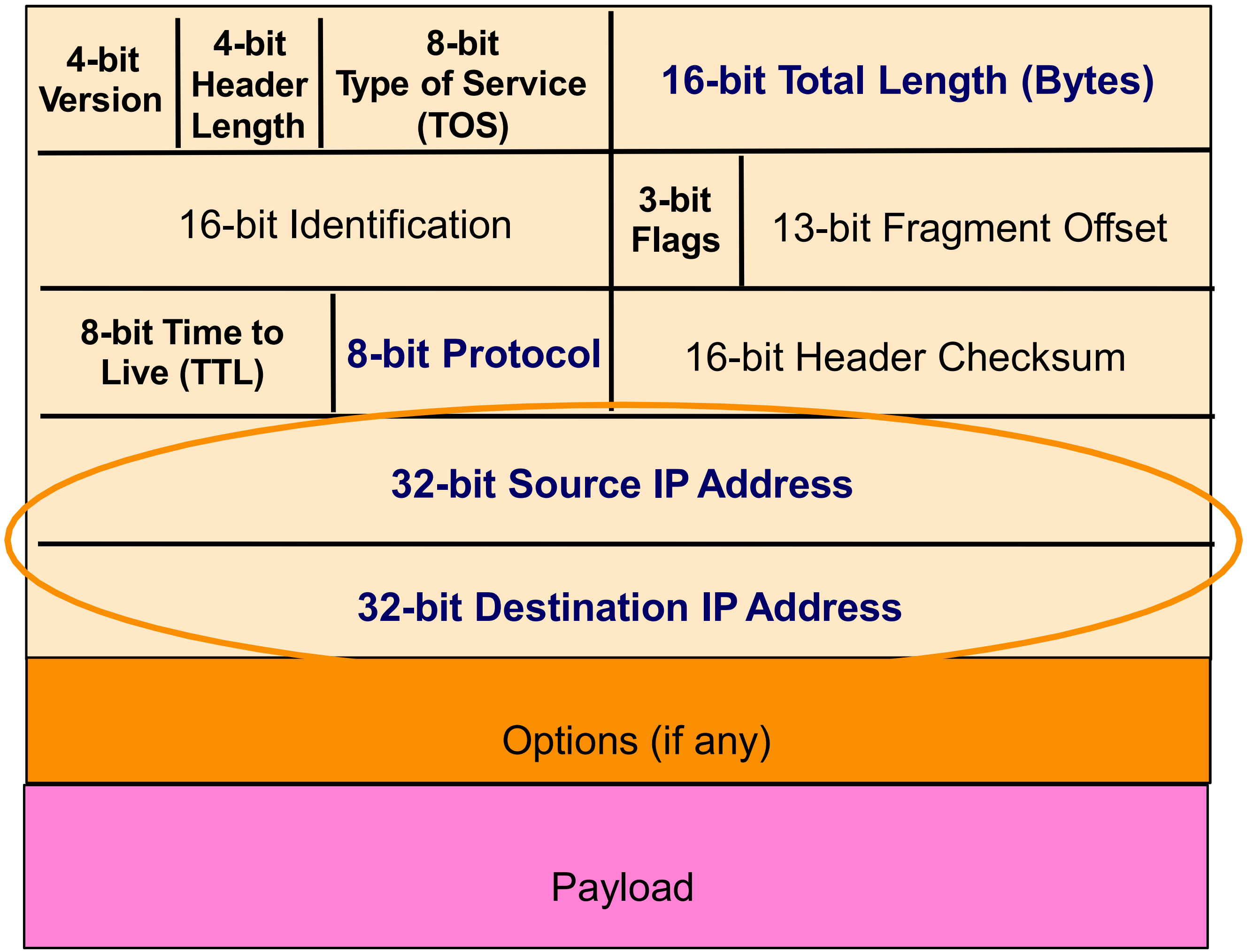
IP Packet Structure



IP Packet Structure



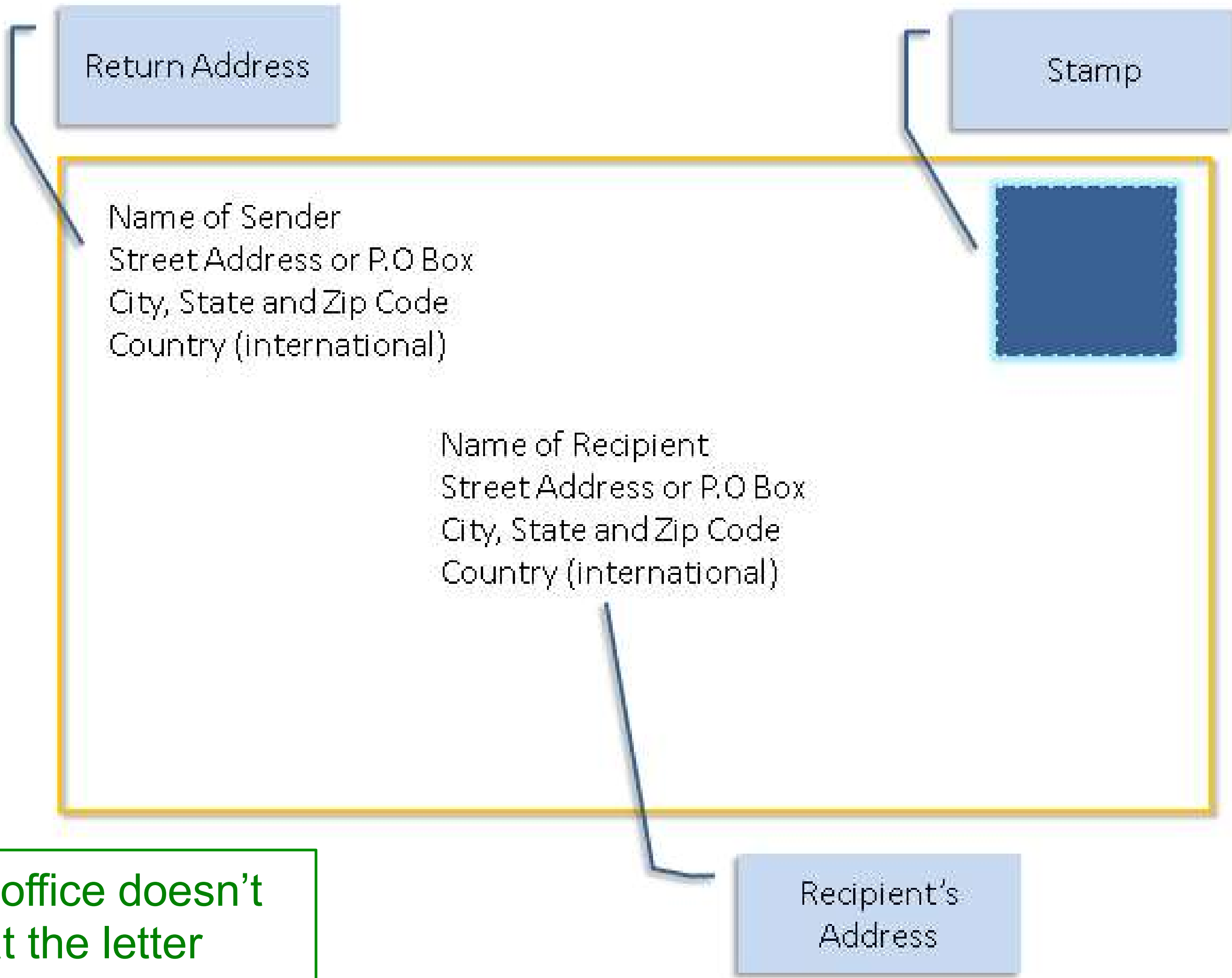
IP Packet Structure



IP Packet Header (Continued)

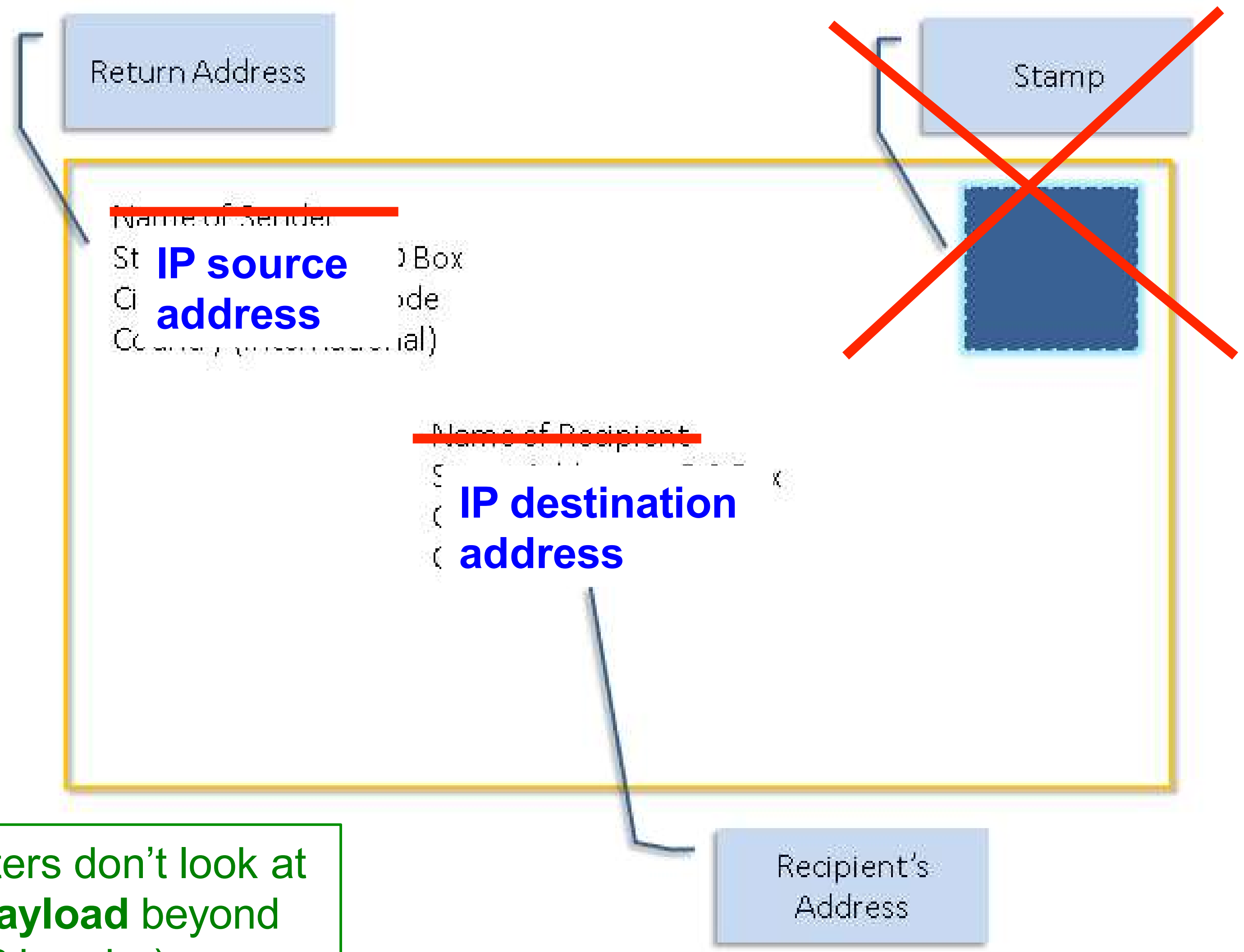
- Two IP addresses
  - Source IP address (32 bits)
  - Destination IP address (32 bits)
- Destination address
  - Unique identifier/locator for the receiving host
  - Allows each node to make forwarding decisions
- Source address
  - Unique identifier/locator for the sending host
  - Recipient can decide whether to accept packet
  - Enables recipient to send a reply back to source

Postal Envelopes:



(Post office doesn't look at the letter inside the envelope)

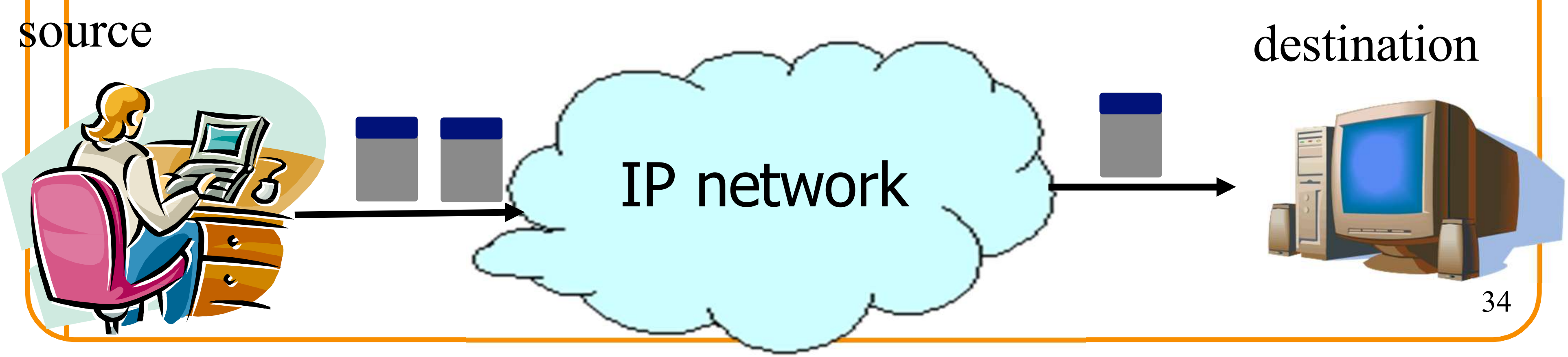
Analogy of IP to Postal Envelopes:



(Routers don't look at the **payload** beyond the IP header)

IP: "*Best Effort*" Packet Delivery

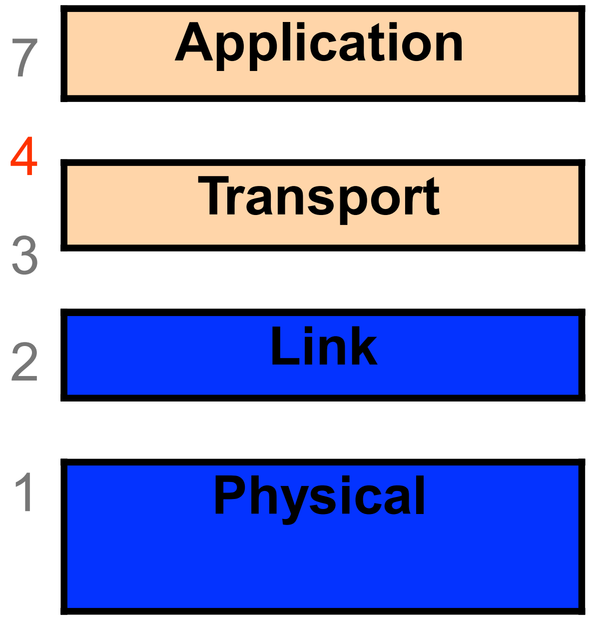
- Routers inspect destination address, locate "next hop" in forwarding table
  - Address = ~unique identifier/locator for the receiving host
- Only provides a "*I'll give it a try*" delivery service:
  - Packets may be lost
  - Packets may be corrupted
  - Packets may be delivered out of order



“*Best Effort*” is Lame! What to do?

- It’s the job of our Transport (layer 4) protocols to build services our apps need out of IP’s modest layer-3 service

Layer 4: Transport Layer



End-to-end communication between processes

Different services provided:

TCP = reliable *byte stream*

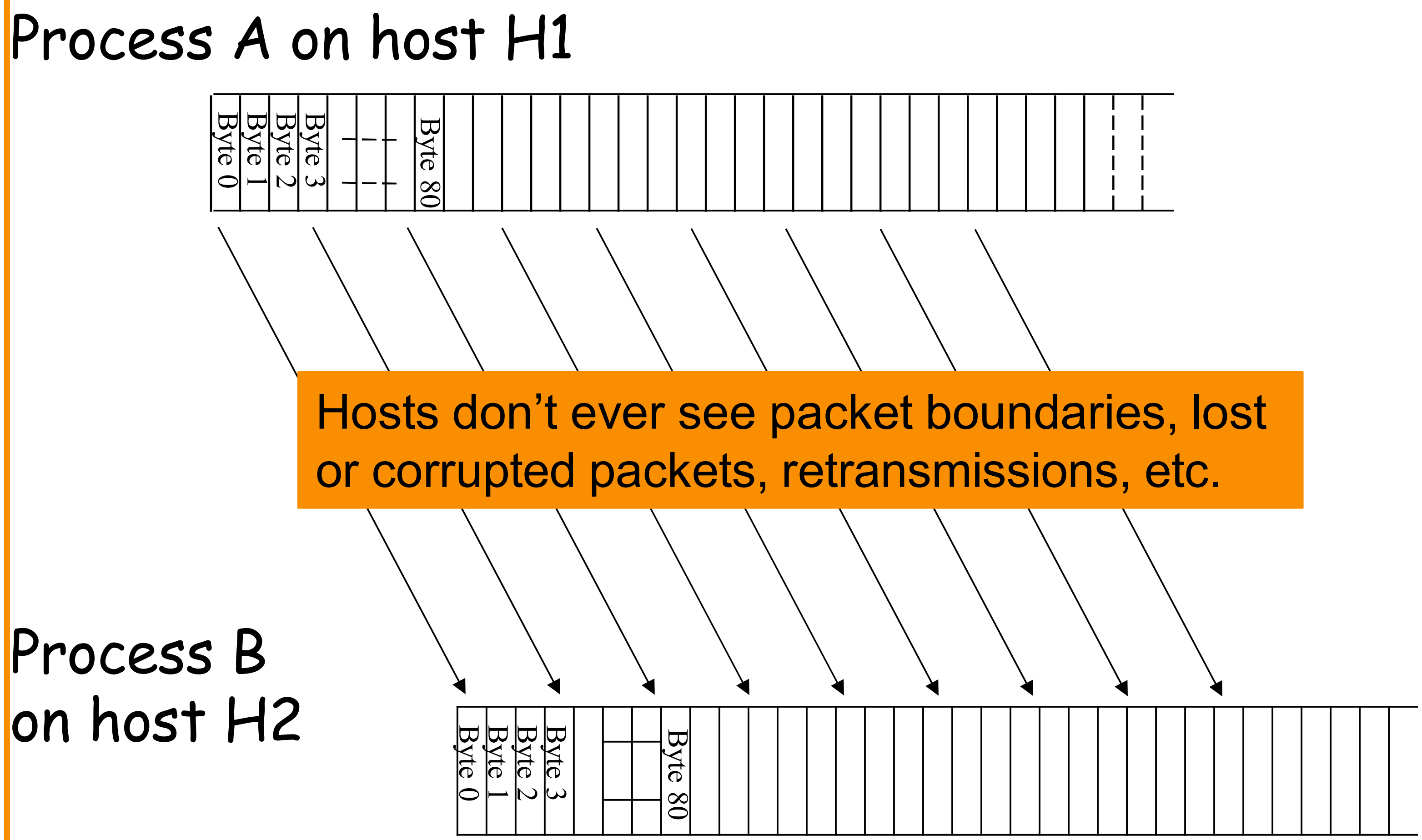
UDP = unreliable *datagrams*

(Datagram = single packet message)

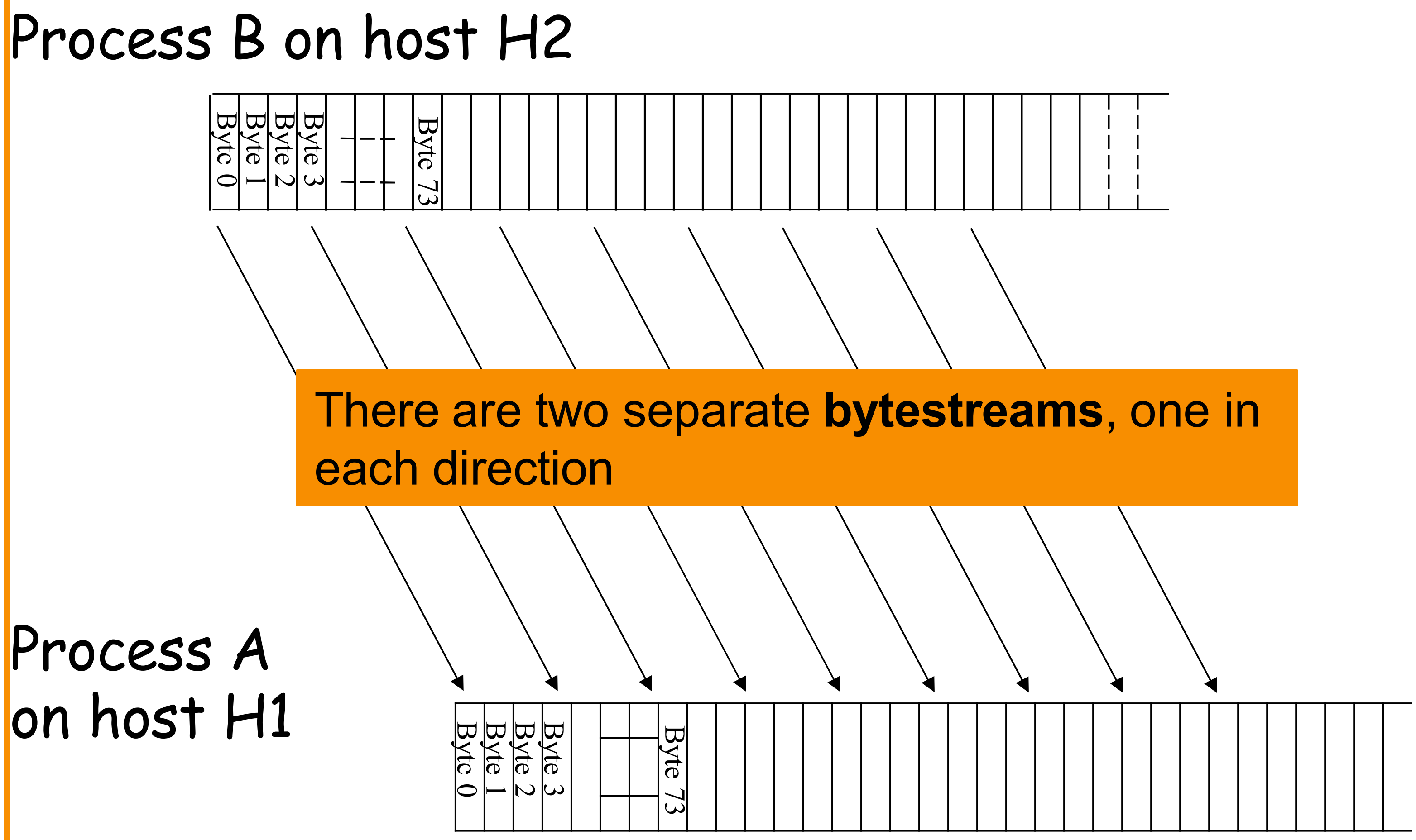
"Best Effort" is Lame! What to do?

- It's the job of our Transport (layer 4) protocols to build services our apps need out of IP's modest layer-3 service
- #1 workhorse: TCP (Transmission Control Protocol)
- Service provided by TCP:
  - Connection oriented (explicit set-up / tear-down)
    - End hosts (processes) can have multiple concurrent long-lived communication
  - **Reliable**, in-order, *byte-stream* delivery
    - Robust detection & retransmission of lost data

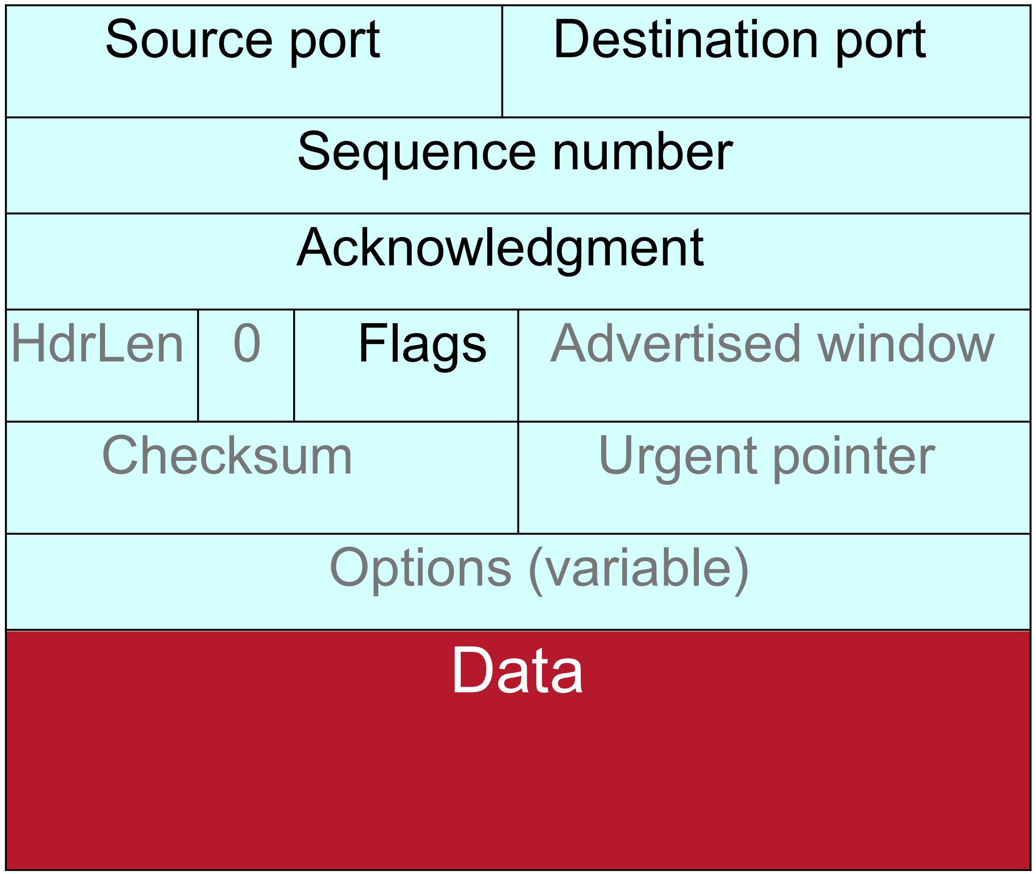
TCP "Bytestream" Service



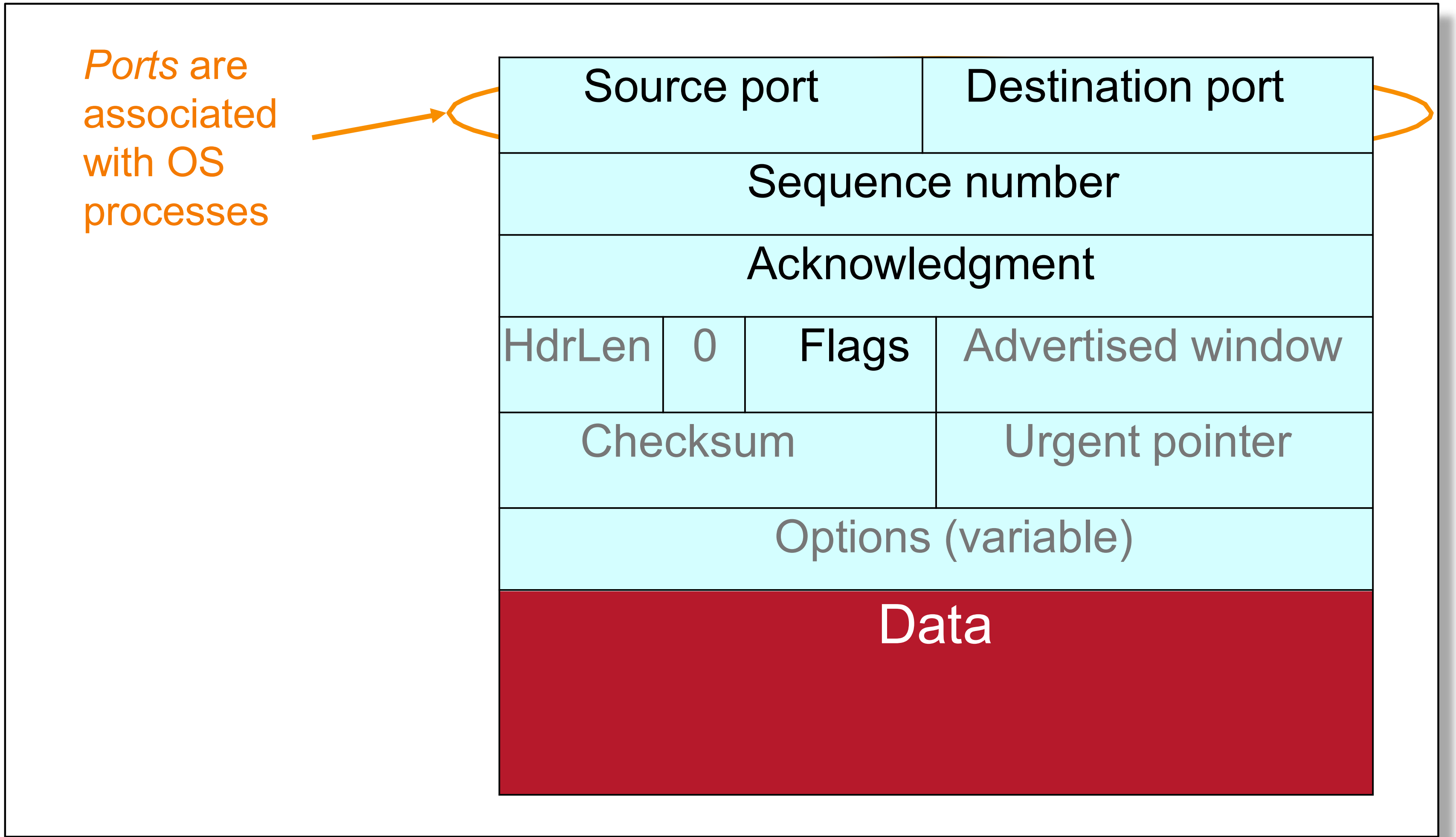
Bidirectional communication:



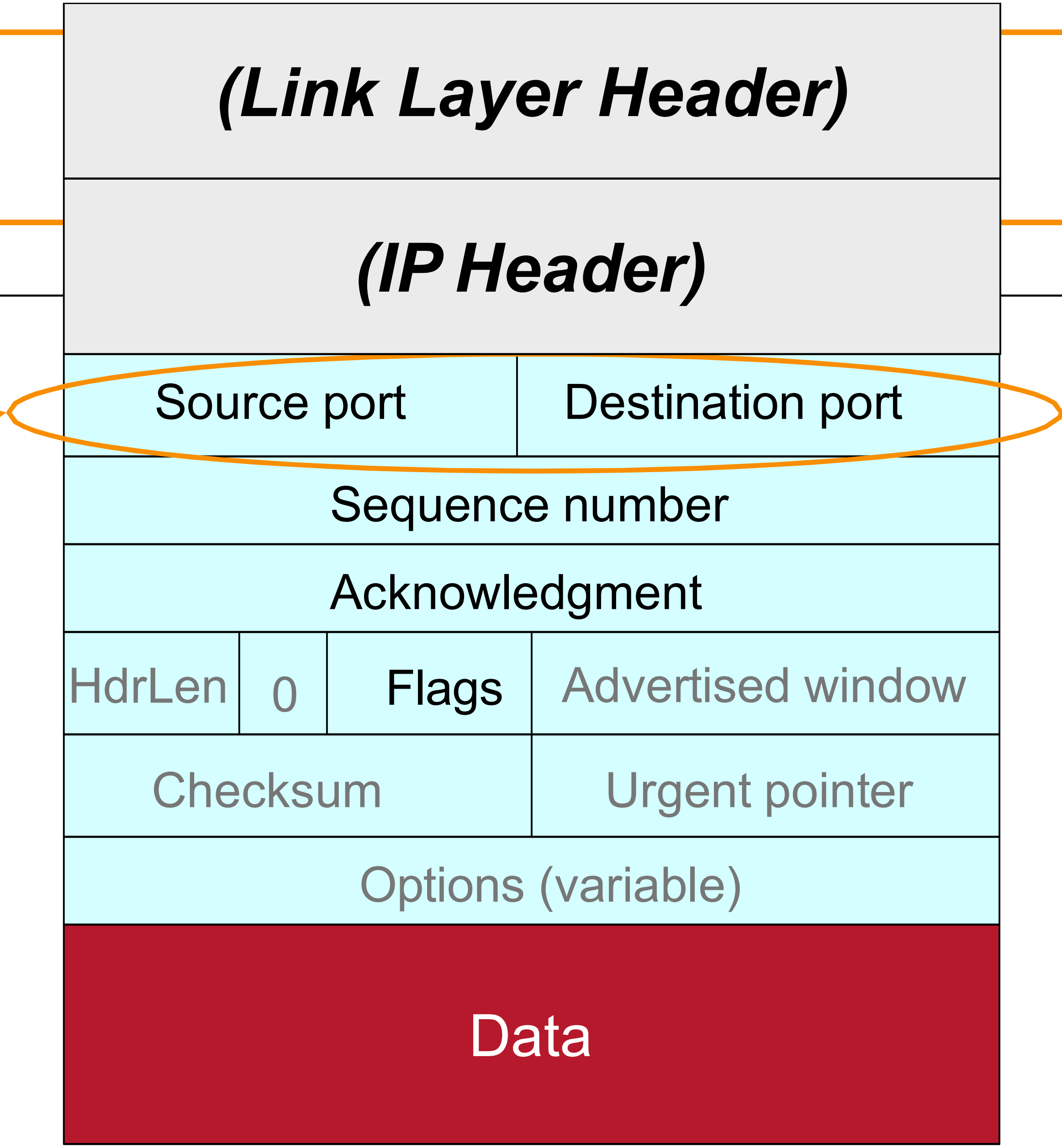
TCP Header


Source port
Destination port
Sequence number
Acknowledgment
HdrLen
0
Flags
Advertised window
Checksum
Urgent pointer
Options (variable)
Data

TCP Header


Ports are associated with OS processes
Source port
Destination port
Sequence number
Acknowledgment
HdrLen
0
Flags
Advertised window
Checksum
Urgent pointer
Options (variable)
Data

TCP Header



Ports are associated with OS processes

IP source & destination addresses plus TCP source and destination ports uniquely identifies a TCP connection

TCP Header

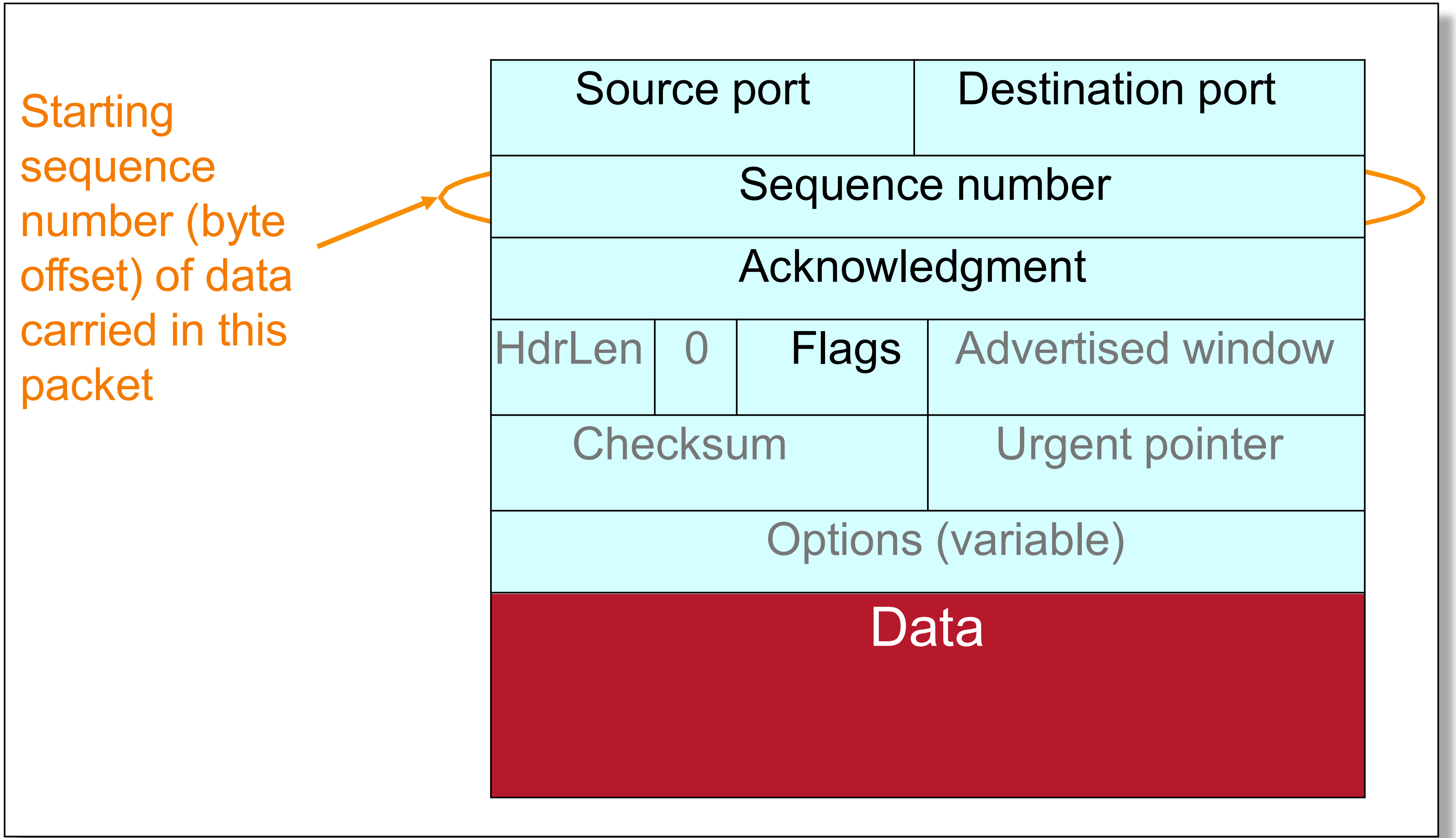
Ports are associated with OS processes

IP source & destination addresses plus TCP source and destination ports uniquely identifies a TCP connection

Some port numbers are "well known" / reserved
e.g. port 80 = HTTP

Source port		Destination port	
Sequence number			
Acknowledgment			
HdrLen	0	Flags	Advertised window
Checksum		Urgent pointer	
Options (variable)			
Data			

TCP Header

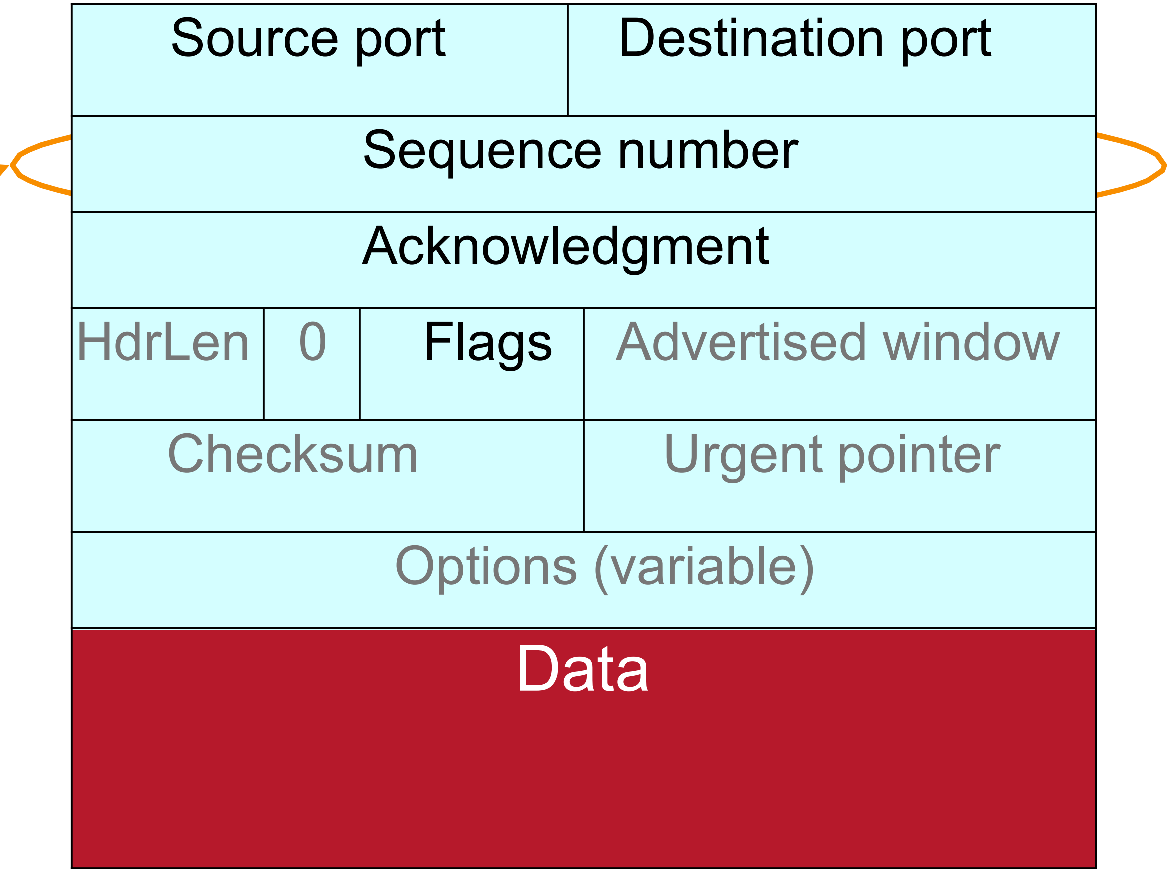


Starting sequence number (byte offset) of data carried in this packet
Source port
Destination port
Sequence number
Acknowledgment
HdrLen
0
Flags
Advertised window
Checksum
Urgent pointer
Options (variable)
Data

TCP Header

Starting sequence number (byte offset) of data carried in this packet

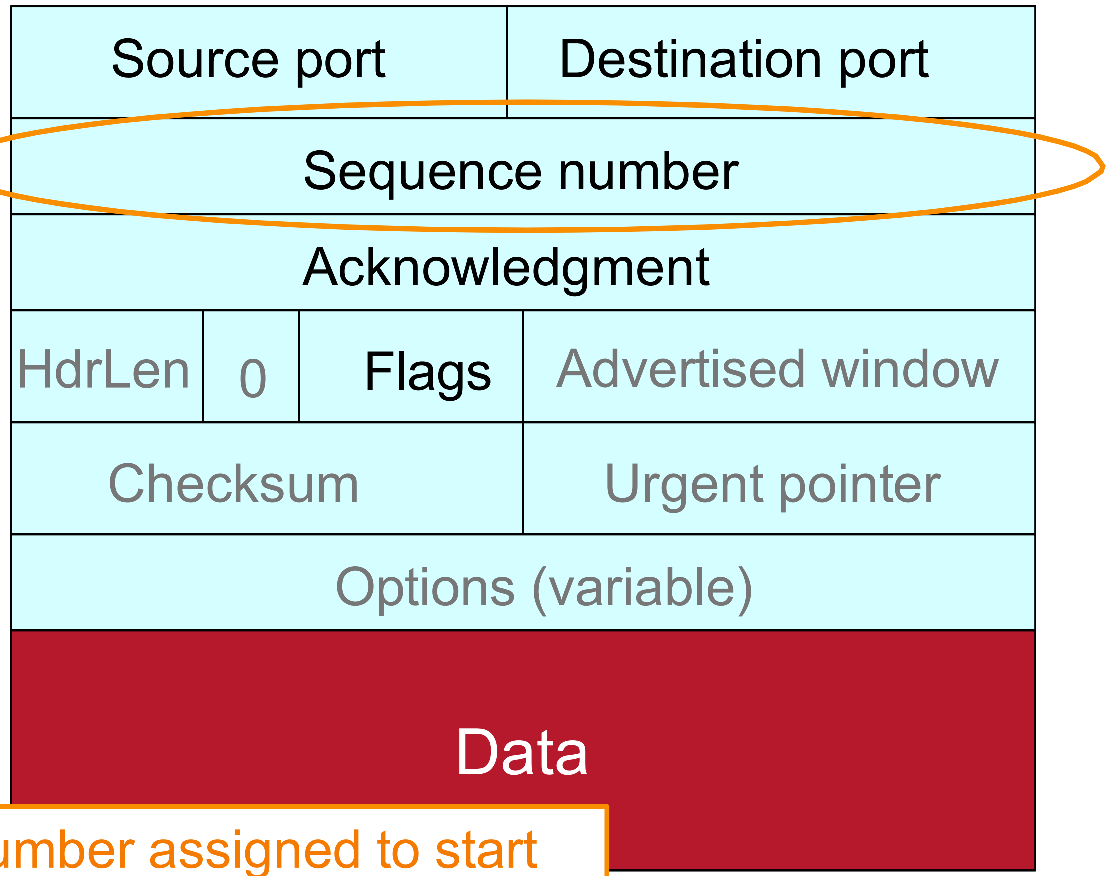
Byte streams numbered independently in each direction



TCP Header

Starting sequence number (byte offset) of data carried in this packet

Byte stream numbered independently in each direction

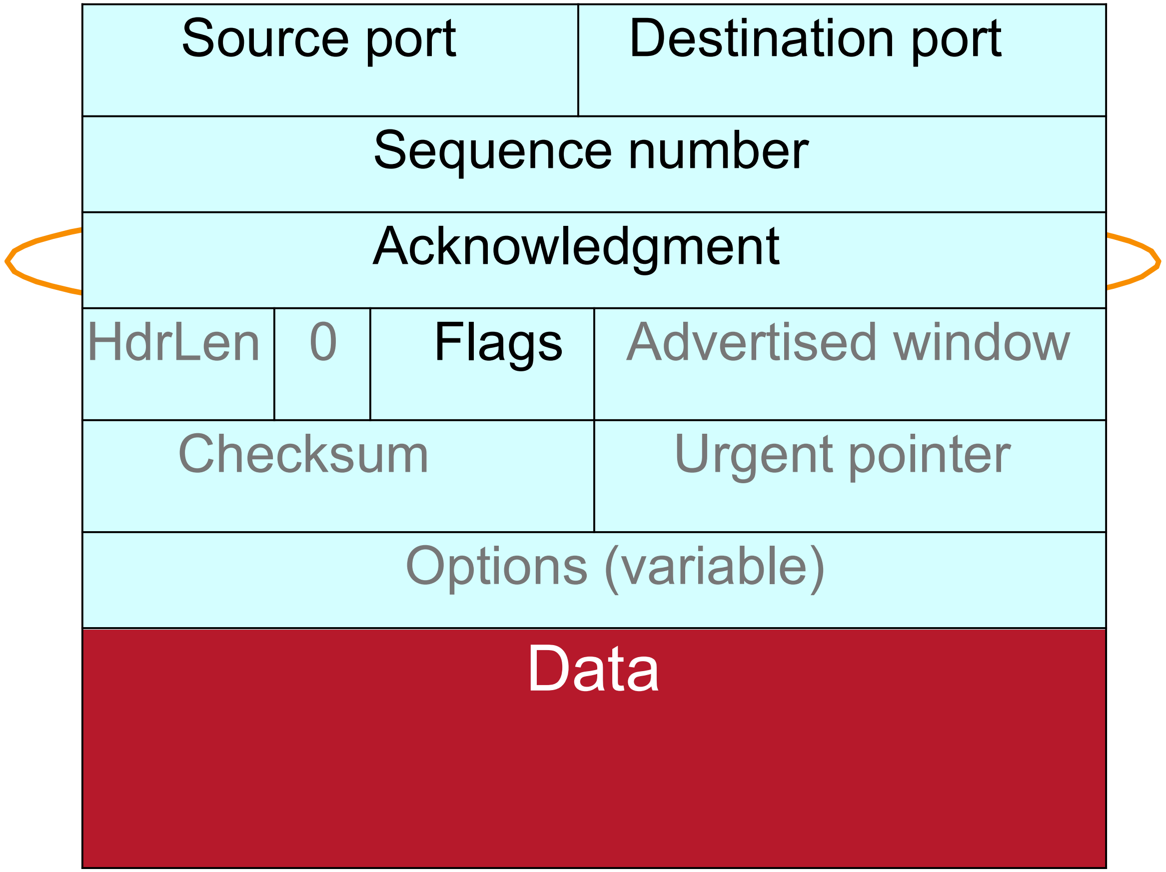


Sequence number assigned to start of byte stream is picked when connection begins; **doesn't** start at 0

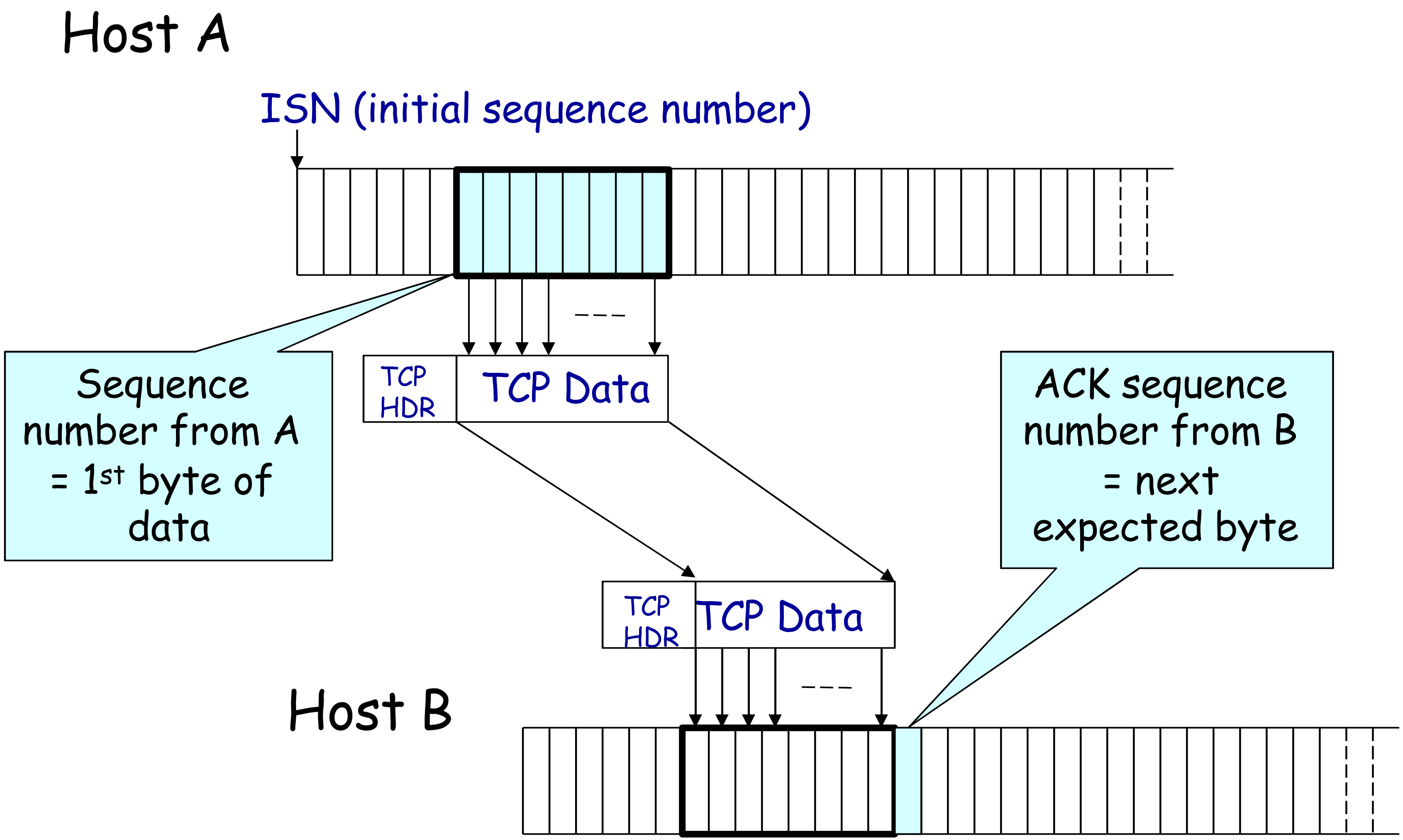
TCP Header

Acknowledgment gives seq # **just beyond** highest seq. received **in order**.

If sender sends **N** bytestream bytes starting at seq **S** then "ack" for it will be **S+N**.



Sequence Numbers

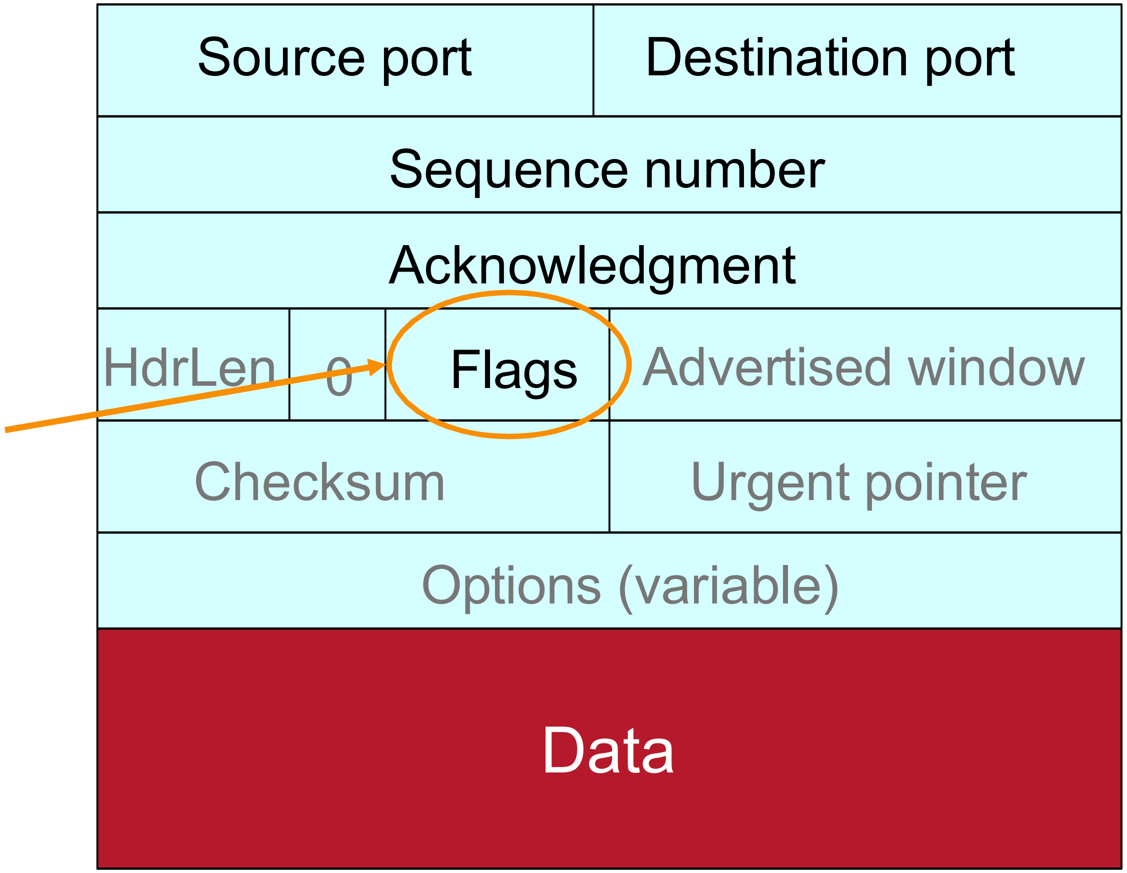


TCP Header

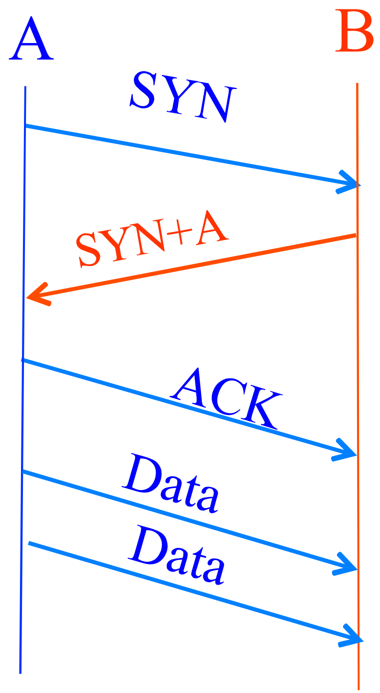
Uses include:

acknowledging data ("**ACK**")

setting up ("**SYN**") and closing connections ("**FIN**" and "**RST**")



Establishing a TCP Connection

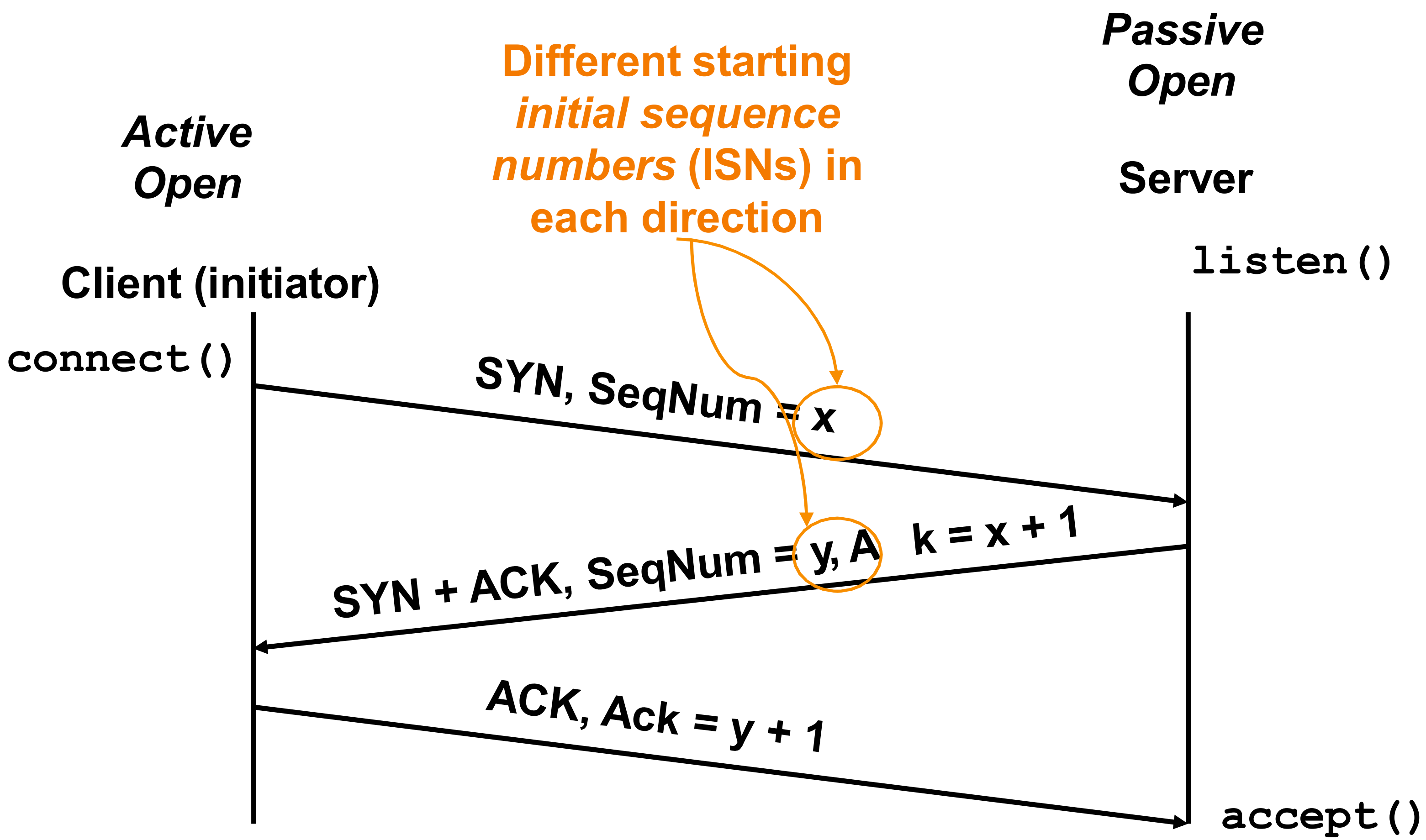


Each host tells its *Initial Sequence Number* (ISN) to the other host.

(Spec says to pick based on local clock)

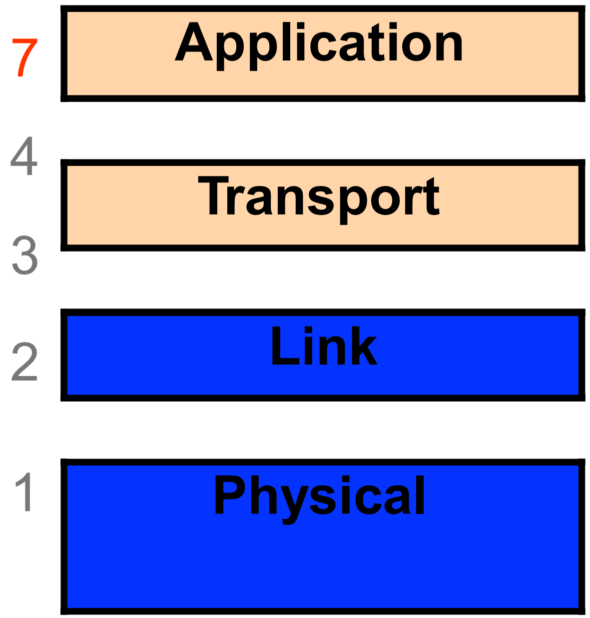
- Three-way handshake to establish connection
  - Host A sends a **SYN** (open; "synchronize sequence numbers") to host B
  - Host B returns a SYN acknowledgment (**SYN+ACK**)
  - Host A sends an **ACK** to acknowledge the SYN+ACK

Timing Diagram: 3-Way Handshaking



Extra Material

Layer 7: Application Layer



Communication of whatever you wish

Can use whatever transport(s) is convenient

Freely structured

E.g.:

Skype, SMTP (email),
HTTP (Web), Halo, BitTorrent

Web (HTTP) Request

Method — Resource — HTTP version — Headers

```
GET /index.html HTTP/1.1
Accept: image/gif, image/x-bitmap, image/jpeg, */*
Accept-Language: en
Connection: Keep-Alive
User-Agent: Mozilla/1.22 (compatible; MSIE 2.0; Windows 95)
Host: www.example.com
Referer: http://www.google.com?q=dingbats

```

Blank line

Data (if POST; none for GET)

GET: download data. POST: upload data.

Web (HTTP) Response

HTTP version **Status code** **Reason phrase** **Headers** **Data**

```
HTTP/1.0 200 OK
Date: Sun, 19 Apr 2009 02:20:42 GMT
Server: Microsoft-Internet-Information-Server/5.0
Connection: keep-alive
Content-Type: text/html
Last-Modified: Sat, 18 Apr 2009 17:39:05 GMT
Set-Cookie: session=44eb; path=/servlets
Content-Length: 2543

<HTML> Some data... blah, blah, blah </HTML>
```

Host Names vs. IP addresses

- Host names
  - Examples: www.cnn.com and bbc.co.uk
  - Mnemonic name appreciated by humans
  - Variable length, full alphabet of characters
  - Provide little (if any) information about location
- IP addresses
  - Examples: 64.236.16.20 and 212.58.224.131
  - Numerical address appreciated by routers
  - Fixed length, binary number
  - Hierarchical, related to host location

IP Packet Structure

4-bit Version	4-bit Header Length	8-bit Type of Service (TOS)	16-bit Total Length (Bytes)	
16-bit Identification			3-bit Flags	13-bit Fragment Offset
8-bit Time to Live (TTL)		8-bit Protocol	16-bit Header Checksum	
32-bit Source IP Address				
32-bit Destination IP Address				
Options (if any)				
Payload				

IP Packet Header Fields (Continued)

- Total length (16 bits)
  - Number of bytes in the packet
  - Maximum size is 65,535 bytes (2^{16} -1)
  - … though underlying links may impose smaller limits
- Fragmentation: when forwarding a packet, an Internet router can split it into multiple pieces ("fragments") if too big for next hop link
- End host reassembles to recover original packet
- Fragmentation information (32 bits)
  - Packet identifier, flags, and fragment offset
  - Supports dividing a large IP packet into fragments
  - … in case a link cannot handle a large IP packet

Example: E-Mail Message Using MIME

```
From: jrex@cs.princeton.edu
To: feamster@cc.gatech.edu
Subject: picture of my cat
MIME-Version: 1.0
Content-Transfer-Encoding: base64
Content-Type: image/jpeg

Base64 encoded data ....
JVBERi0xLjMNJeLjz9MNMSAwI
.........................
......base64 encoded data
```

MIME version

method used to encode data

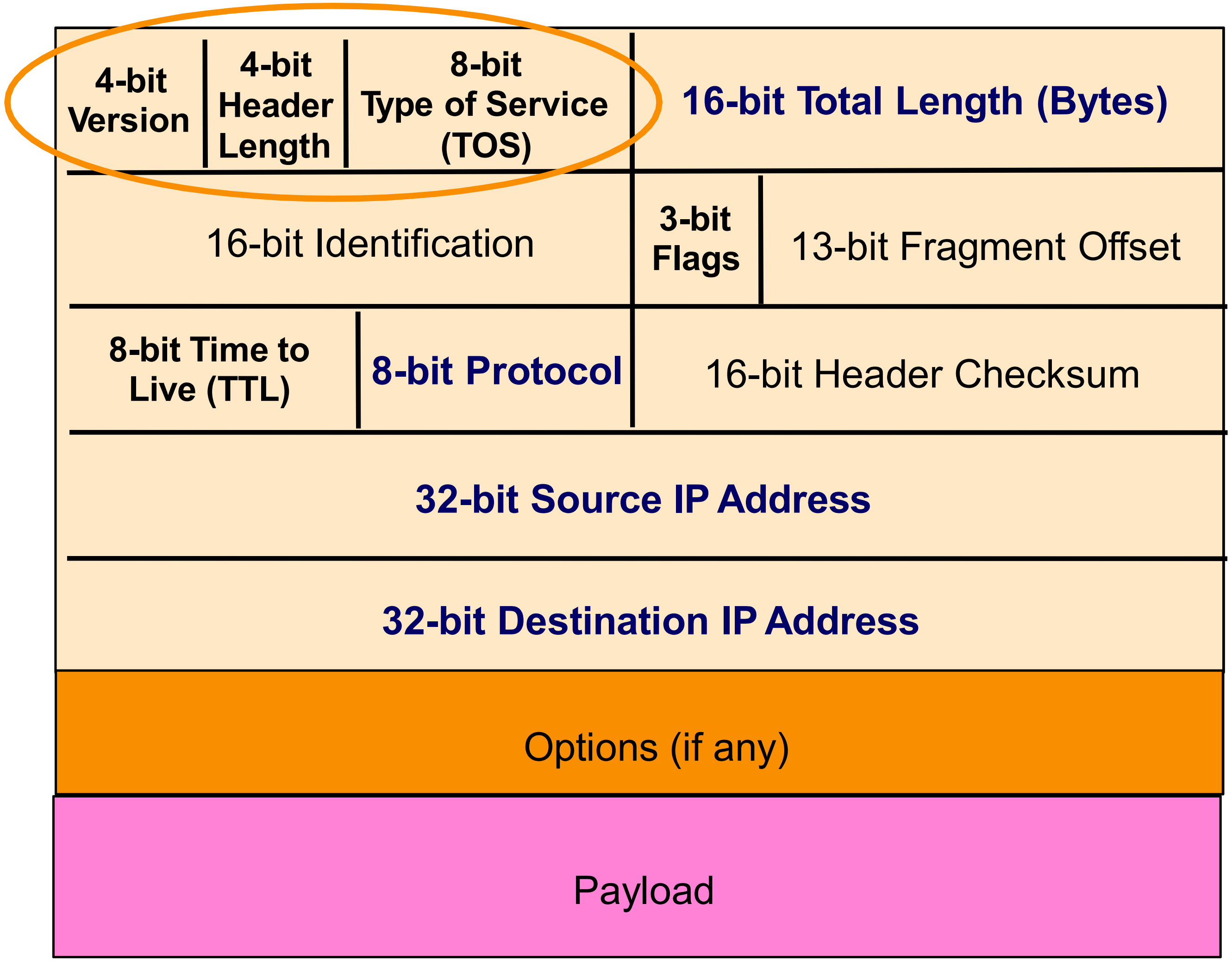
type and subtype

encoded data

Example With Received Header

Return-Path: <casado@cs.stanford.edu>
Received: from ribavirin.CS.Princeton.EDU (ribavirin.CS.Princeton.EDU [128.112.136.44])
by newark.CS.Princeton.EDU (8.12.11/8.12.11) with SMTP id k04M5R7Y023164
for <jrex@newark.CS.Princeton.EDU>; Wed, 4 Jan 2006 17:05:37 -0500 (EST)
Received: from bluebox.CS.Princeton.EDU ([128.112.136.38])
by ribavirin.CS.Princeton.EDU (SMSSMTP 4.1.0.19) with SMTP id M20060104170536O7946
for <jrex@newark.CS.Princeton.EDU>; Wed, 04 Jan 2006 17:05:36 -0500
Received: from smtp-roam.Stanford.EDU (smtp-roam.Stanford.EDU [171.64.10.152])
by bluebox.CS.Princeton.EDU (8.12.11/8.12.11) with ESMTP id k04M5XNQ005204
for <jrex@cs.princeton.edu>; Wed, 4 Jan 2006 17:05:35 -0500 (EST)
Received: from [192.168.1.101] (adsl-69-107-78-147.dsl.pltn13.pacbell.net [69.107.78.147])
(authenticated bits=0)
by smtp-roam.Stanford.EDU (8.12.11/8.12.11) with ESMTP id k04M5W92018875
(version=TLSv1/SSLv3 cipher=DHE-RSA-AES256-SHA bits=256 verify=NOT);
Wed, 4 Jan 2006 14:05:32 -0800
Message-ID: <43BC46AF.3030306@cs.stanford.edu>
Date: Wed, 04 Jan 2006 14:05:35 -0800
From: Martin Casado <casado@cs.stanford.edu>
User-Agent: Mozilla Thunderbird 1.0 (Windows/20041206)
MIME-Version: 1.0
To: jrex@CS.Princeton.EDU
CC: Martin Casado <casado@cs.stanford.edu>
Subject: Using VNS in Class
Content-Type: text/plain; charset=ISO-8859-1; format=flowed
Content-Transfer-Encoding: 7bit

IP Packet Structure



IP Packet Header Fields

- Version number (4 bits)
  - Indicates the version of the IP protocol
  - Necessary to know what other fields to expect
  - Typically "4" (for IPv4), and sometimes "6" (for IPv6)
- Header length (4 bits)
  - Number of 32-bit words in the header
  - Typically "5" (for a 20-byte IPv4 header)
  - Can be more when IP options are used
- Type-of-Service (8 bits)
  - Allow packets to be treated differently based on needs
  - E.g., low delay for audio, high bandwidth for bulk transfer

Sample Email (SMTP) interaction

```
S: 220 hamburger.edu
C: HELO crepes.fr
S: 250  Hello crepes.fr, pleased to meet you
C: MAIL FROM: <alice@crepes.fr>
S: 250 alice@crepes.fr... Sender ok
C: RCPT TO: <bob@hamburger.edu>
S: 250 bob@hamburger.edu ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: F
C  To: hamburger-list@burger-king.com
C:
C:
C
C:
S: 250 Message accepted for delivery
C: QUIT
S: 221 hamburger.edu closing connection
```

Email header

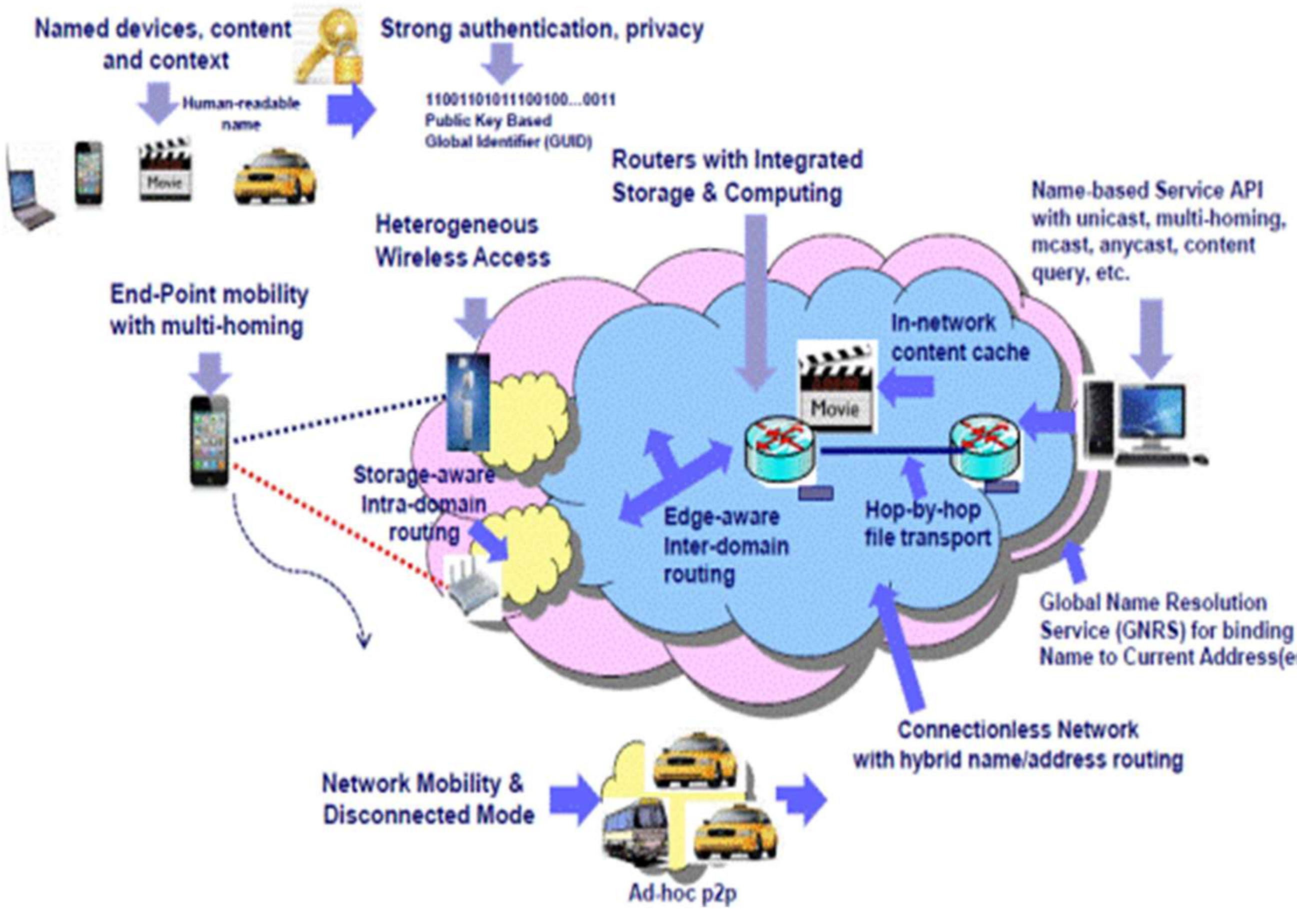
Email body

Lone period marks end of message

Future Internet Architecture (FIA) Awards - 2010

- Follow on to FIND – Future Internet Design, 50 awards over 5 years
- Inform the development of future trustworthy, robust networks
- $8M/3 year grant for each of these projects:
- Mobility First
- NEBULA
- eXpressive Internet Architecture (XIA)
- Named Data Networking (NDN)

Mobility First



Members

Rutgers:
D. Raychaudhuri*+
Wade Trappe

UMass - Amherst
Arun Venkataramani#+
Jim Kurose

Umichigan:Z. Morley Mao+

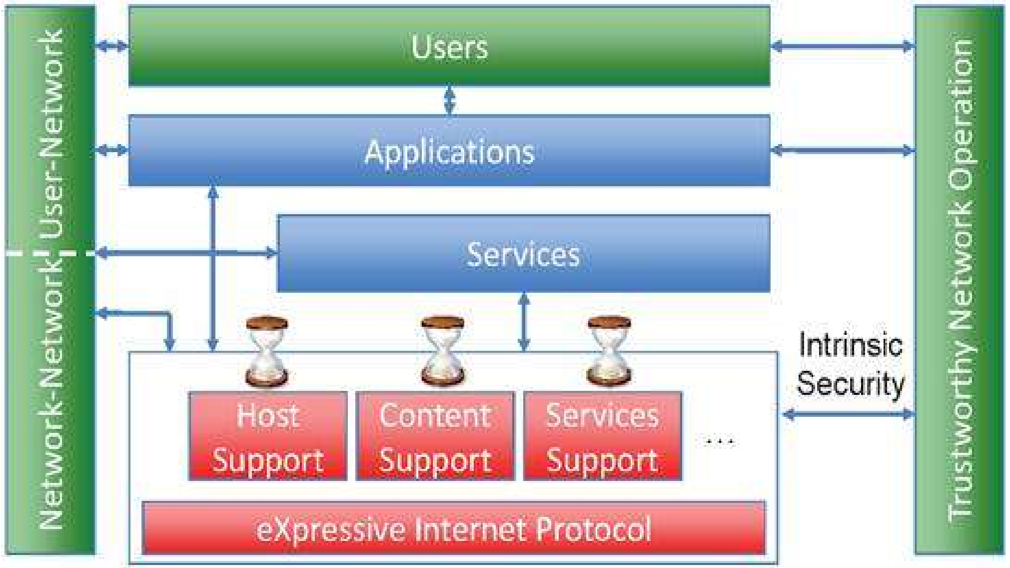
Duke University: Xaiowei Yang+

MIT: Bill Lehr+

U Wisconsin

U Nebraska

eXpressive Internet Architecture (XIA)

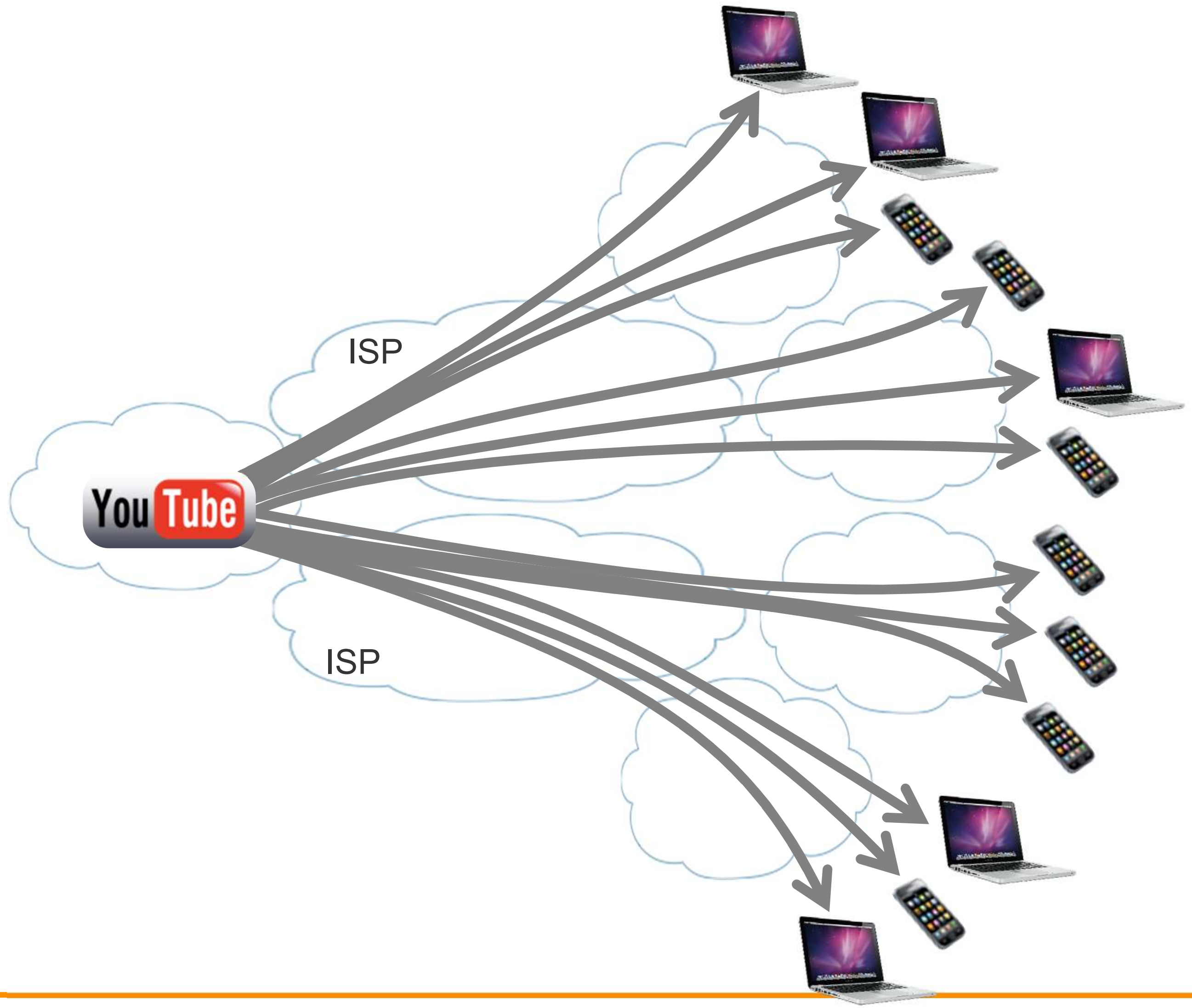


Aditya Akella - CS&S, UW-Madison
David Andersen - CS, CMU
Dan Barrett - CS, CMU (Senior Research Programmer)
John Byers - CS, BU
Laura Dabbish - HCII/Heinz, CMU
David Eckhardt - CS, CMU
Nitin Gupta - CS, CMU
Ruogu Kang - HCI, CMU
Sara Kiesler - HCII, CMU
Yanlin Li - CMU (Research Scientist)
Bruce Maggs - CS, Duke
Duyen Mary Nguyen, HCI, CMU (Post Doc
Jon Peha - EPP/ECE, CMU
Adrian Perrig - EPP/ECE/CS, CMU
Raja Sambasivan - CS, CMU (Post Doc)
Marvin Sirbu - EPP/ECE, CMU
Chuck Song (Senior Scientist)
Srinivasan Seshan - CS, CMU
Peter Steenkiste (PI) - CS/ECE, CMU
Hui Zhang - CS, CMU

Named Data Networking (NDN)

Lixia Zhang, Deborah Estrin, and Jeffrey Burke
University of California, Los Angeles
Van Jacobson, James D. Thornton, and Diana K. Smetters
Palo Alto Research Center (PARC)
Beichuan Zhang
University of Arizona
Gene Tsudik
University of California, Irvine
kc claffy and Dmitri Krioukov
University of California, San Diego
Dan Massey and Christos Papadopoulos
Colorado State University
Tarek Abdelzaher
University of Illinois at Urbana-Champaign
Lan Wang
University of Memphis
Patrick Crowley
Washington University
Edmund Yeh
Yale University

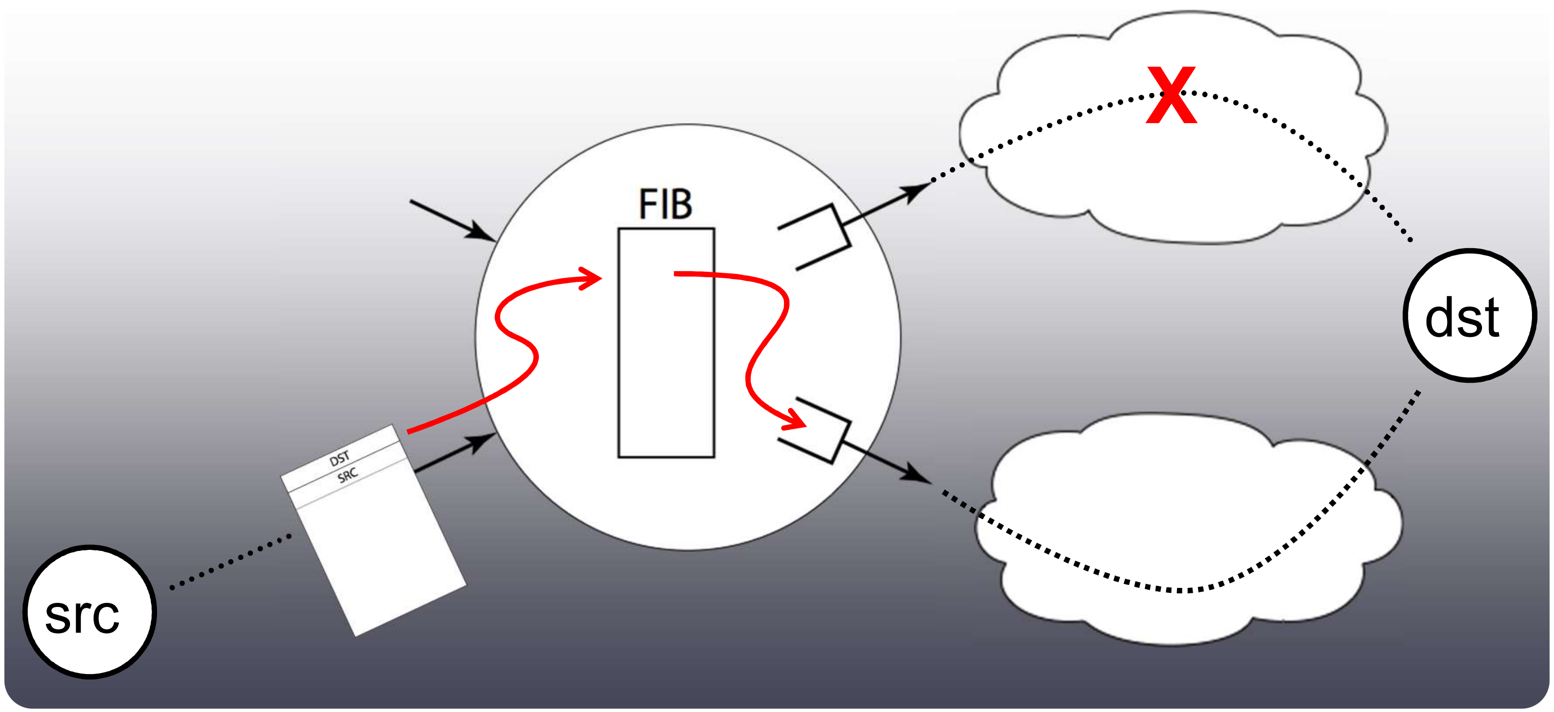
The problem



Communication v. Distribution

	Communication	Distribution
Naming	Endpoints	Content
Security	Secure Process	Secure Content

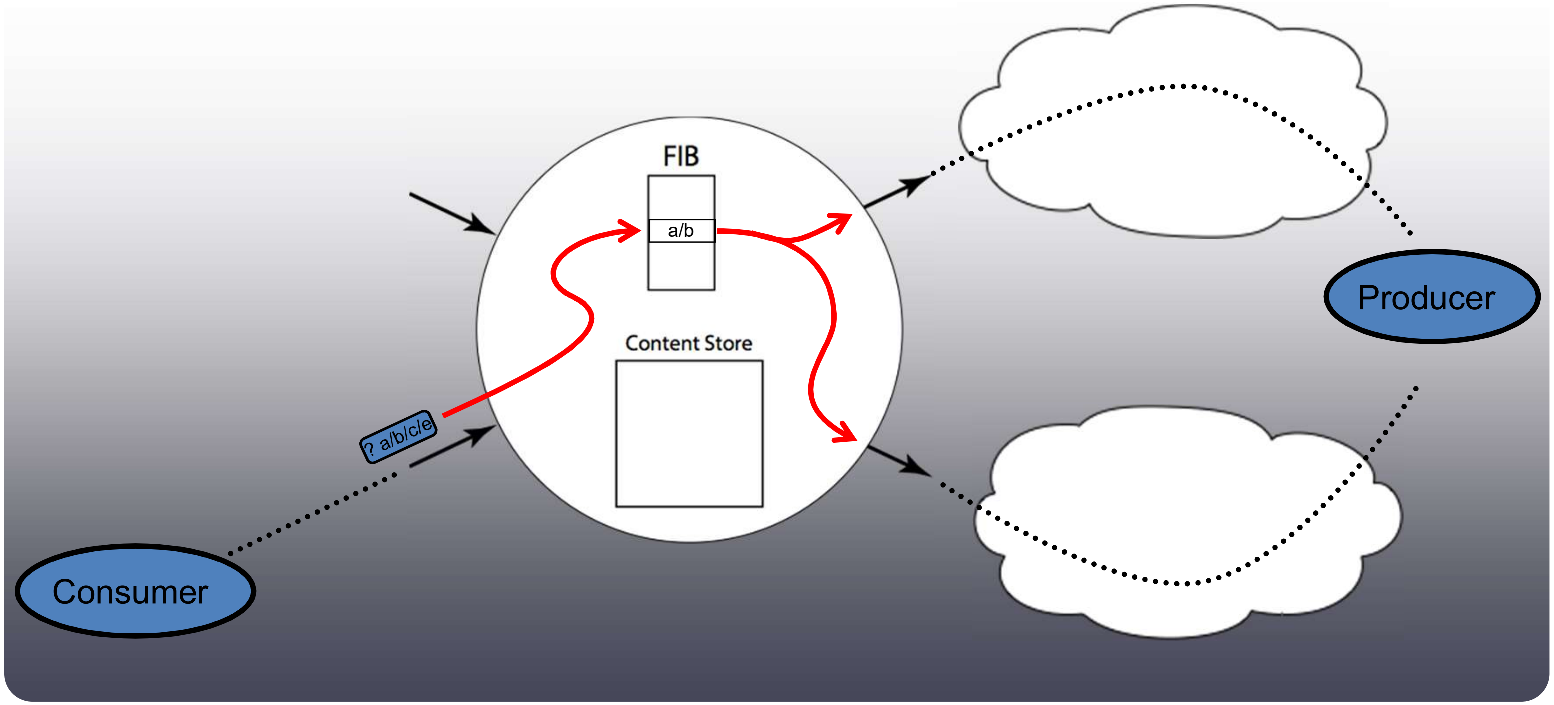
Today



Path determined by global routing, not local choice

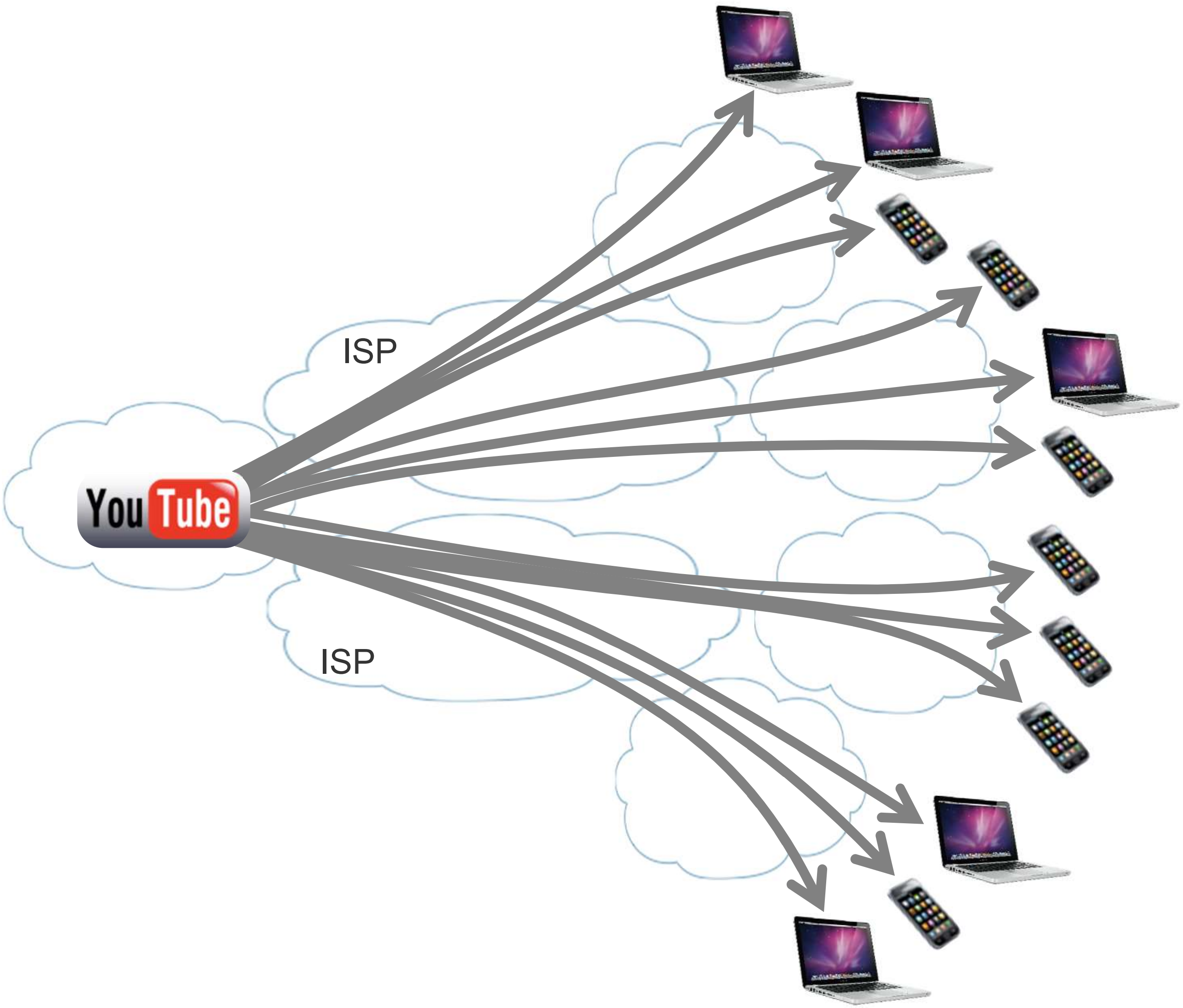
Structural asymmetry precludes market mechanisms and encourages monopoly formation

NDN approach

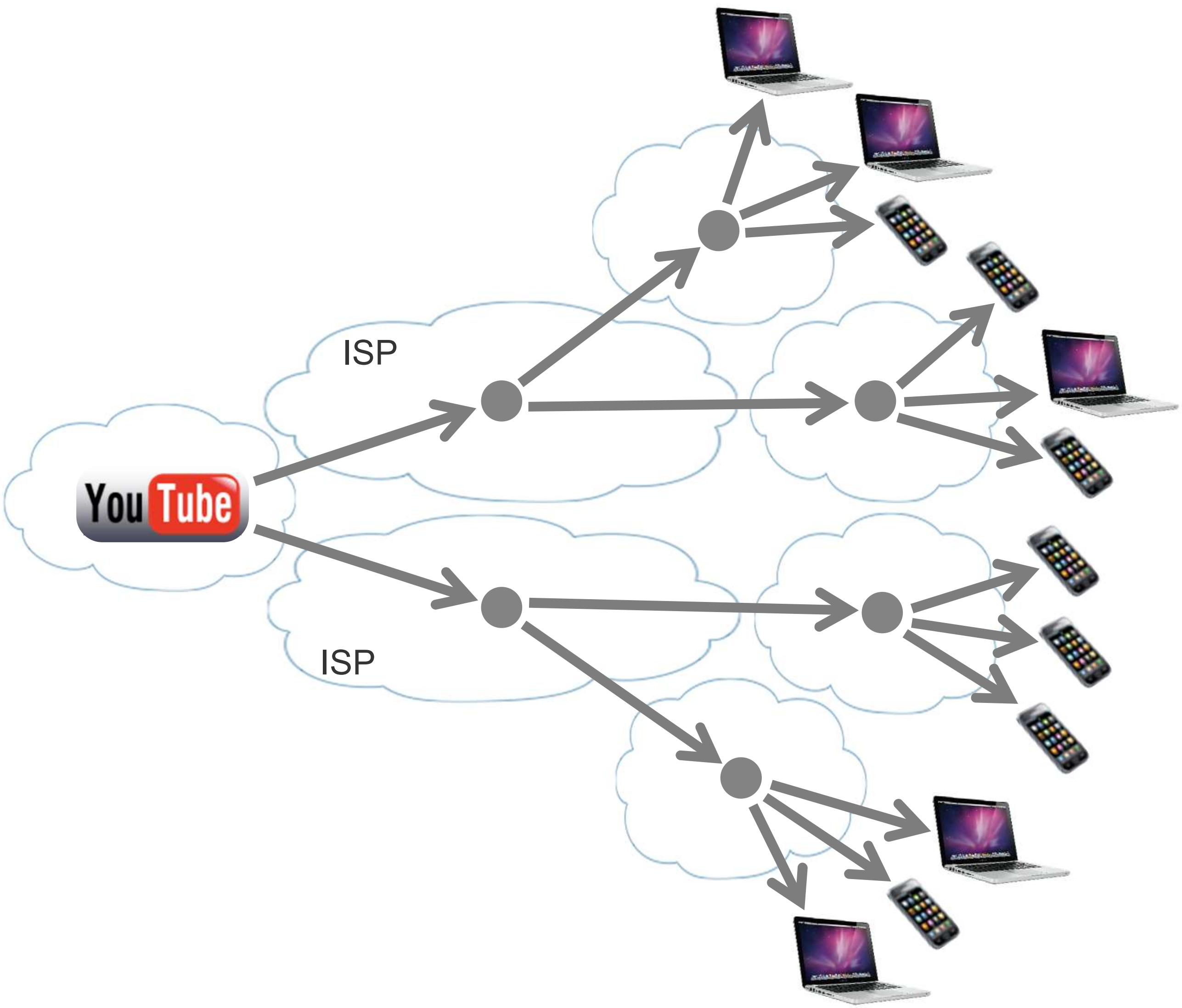


- Packets say 'what' not 'where' (no src or dst)
- Forwarding decision is local
- Upstream performance is measurable

We envision replacing this:



With THIS:



Future Internet Architecture Next Phase (FIA-NP) Awards - 2014

- Built upon success of FIA. Move from design to piloted deployments in real world setting.
- No one architecture will replace Internet wholesale.
- $15M total to 3 projects:
  - XIA-NP
    - Tests in vehicle network and large scale video delivery
  - NDN-NP – established industry consortium
    - Open mHealth and UCLA Facilities Management
  - Mobility First-NP
    - Wireless provider, content delivery network, weather emergency notification system

Group Discussions

- Pls discuss the potential attacks towards different layers (physical/link layers, IP layers, transportation layers).