Internet Protocol Security

CS155 Computer and Network Security

Acknowledgments: Lecture slides are from the Computer Security course taught by Dan Boneh and Zakir Durumeric at Stanford University. When slides are obtained from other sources, a reference will be noted on the bottom of that slide. A full list of references is provided on the last slide.

Stanford University

The Internet

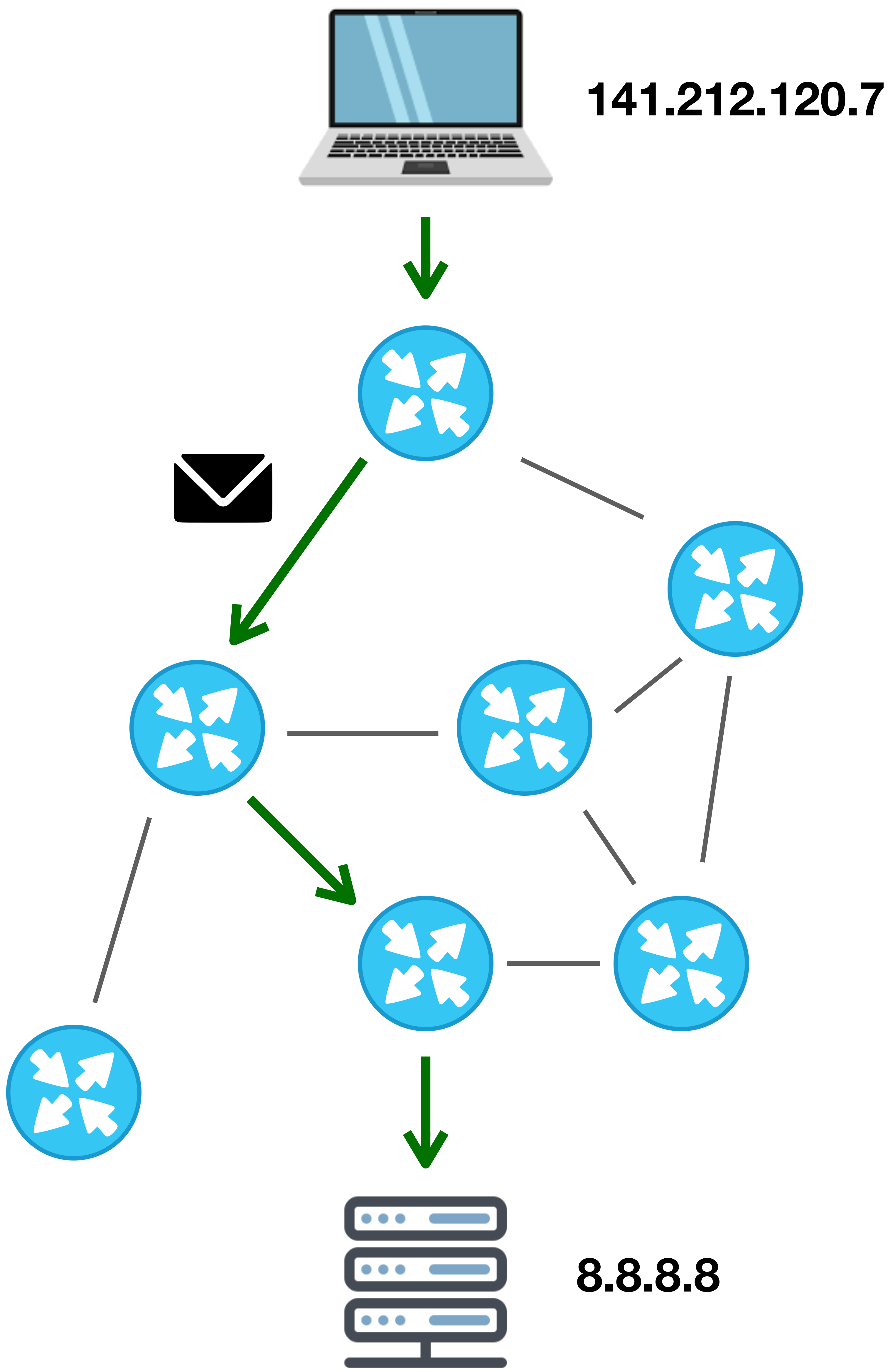
Global network that provides **best-effort** delivery of **packets** between connected hosts

Packet: a structured sequence of bytes

- **Header:** metadata used by network
- **Payload:** user data to be transported

Every host has a unique identifier — IP address

Series of routers receive packets, look at destination address on the header and send it one hop towards the destination IP address

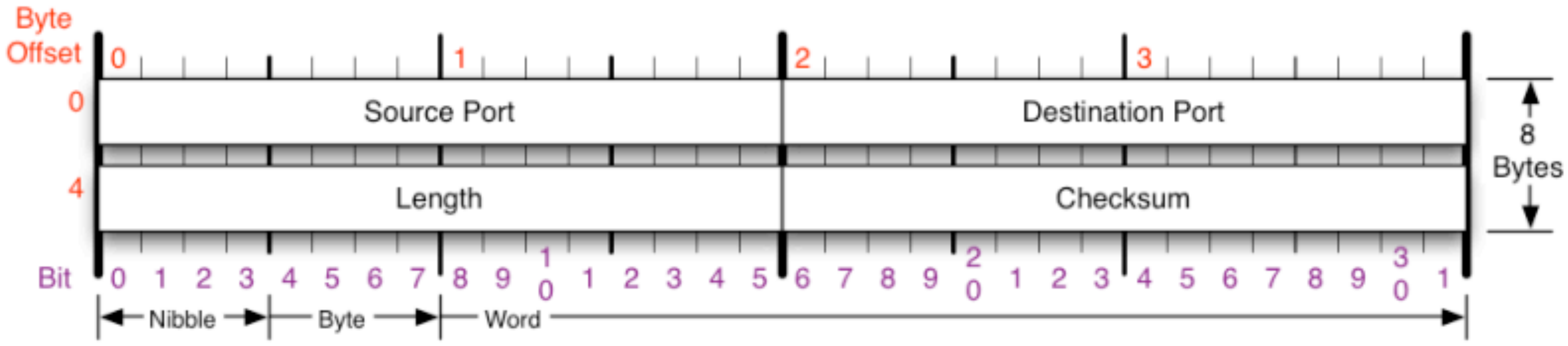


Network Protocols

We define how hosts communicate in published network protocols

Syntax: How communication is structured (e.g., format and order of messages)

Semantics: What communication means. Actions taken on transmit or receipt of message, or when a timer expires. What assumptions can be made.



Example: What bytes contain each field in a packet header

Protocol Layering

Networks use a stack of protocol layers

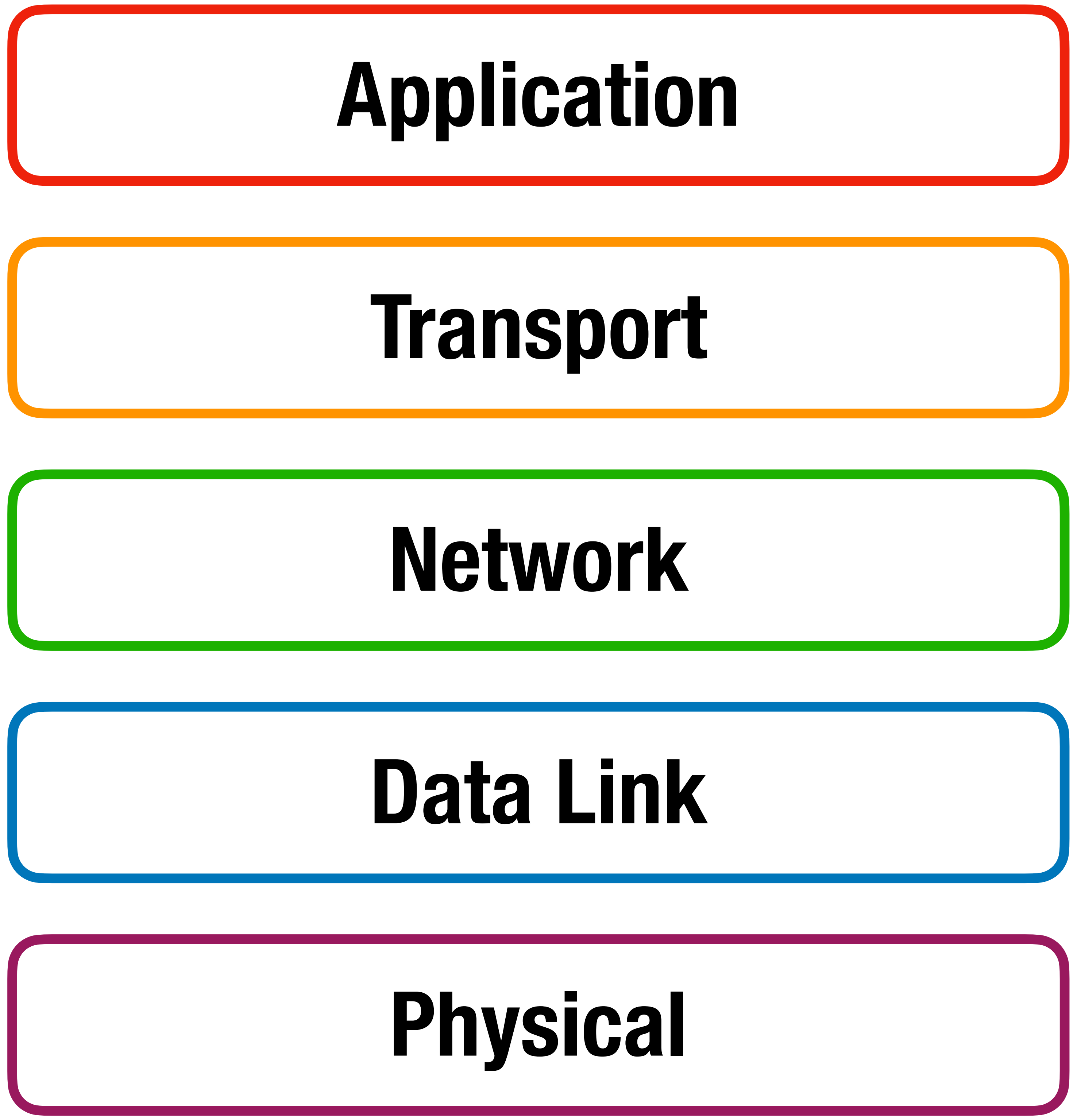
- Each layer has different responsibilities.
- Layers define abstraction boundaries

Lower layers provide services to layers above

- Don’t care what higher layers do

Higher layers use services of layers below

- Don’t worry about how it works



OSI 5 Layer Model

Physical	How do bits get translated into electrical, optical, or radio signals

OSI 5 Layer Model

Data Link

How to get packet to the next hop. Transmission of data frames between two nodes connected by a physical link.

Physical

How do bits get translated into electrical, optical, or radio signals

OSI 5 Layer Model

Network	Responsible for packet forwarding. How to get a packet to the final destination when there are many hops along the way.
Data Link	How to get packet to the next hop. Transmission of data frames between two nodes connected by a physical link.
Physical	How do bits get translated into electrical, optical, or radio signals

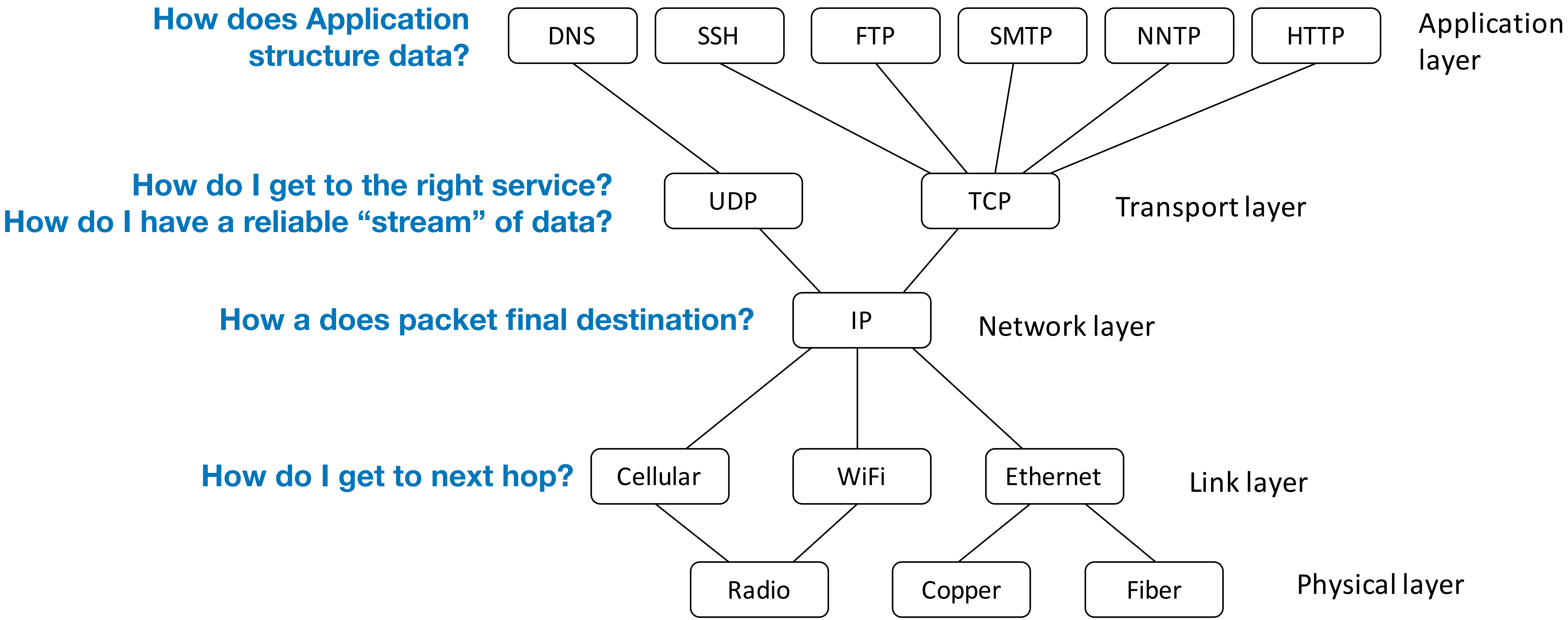
OSI 5 Layer Model

Transport	Allows a client to establish a connection to specific services (e.g., web server on port 80). Provides reliable communication.
Network	Responsible for packet forwarding. How to get a packet to the final destination when there are many hops along the way.
Data Link	How to get packet to the next hop. Transmission of data frames between two nodes connected by a physical link.
Physical	How do bits get translated into electrical, optical, or radio signals

OSI 5 Layer Model

Application	Defines how individual applications communicate. For example, **HTTP** defines how browsers send requests to web servers.
Transport	Allows a client to establish a connection to specific services (e.g., web server on port 80). Provides reliable communication.
Network	Responsible for packet forwarding. How to get a packet to the final destination when there are many hops along the way.
Data Link	How to get packet to the next hop. Transmission of data frames between two nodes connected by a physical link.
Physical	How do bits get translated into electrical, optical, or radio signals

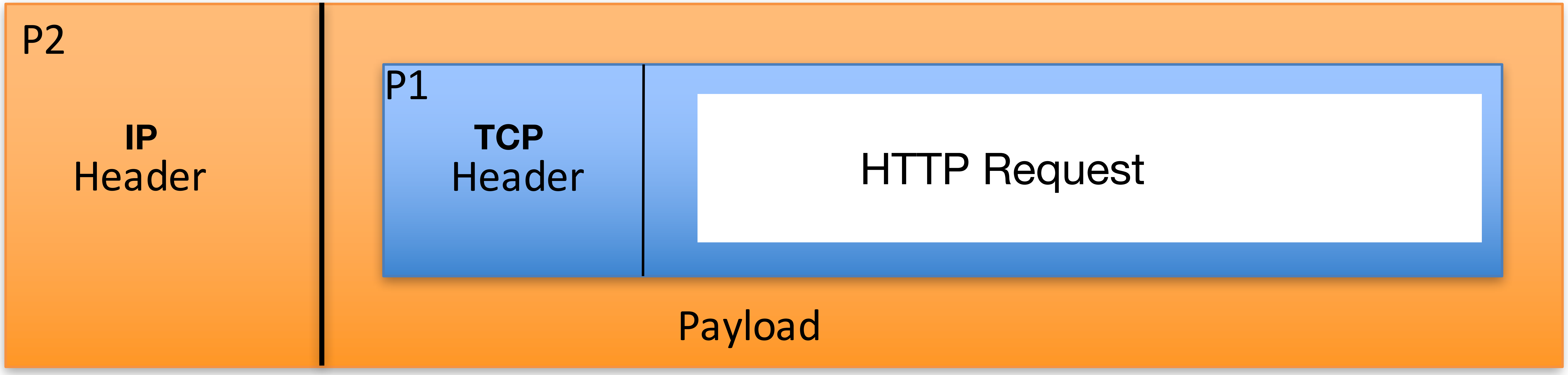
IP — The Narrow Waist



Packet Encapsulation

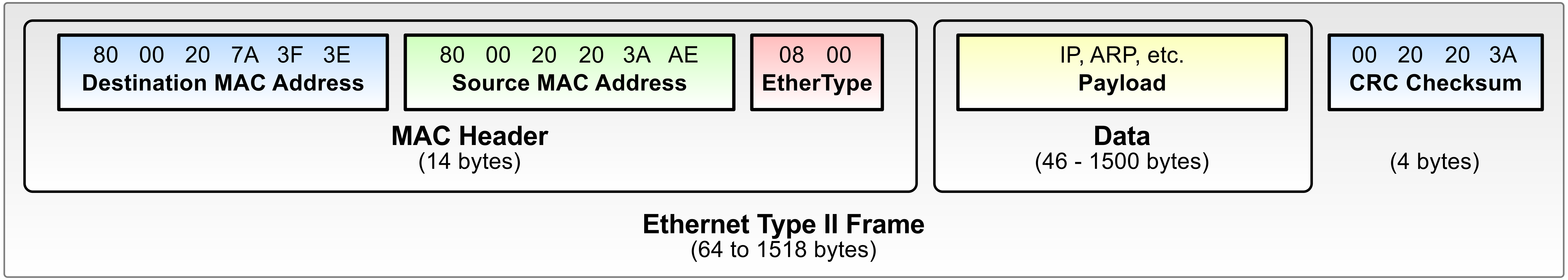
Protocol N1 can use the services of lower layer protocol N2

- A packet P1 of N1 is encapsulated into a packet P2 of N2
- The payload of p2 is p1
- The control information of p2 is derived from that of p1



Ethernet

Most common Link Layer Protocol. Let's you send packets to other local hosts.



At layer 2 (link layer) packets are called *frames*

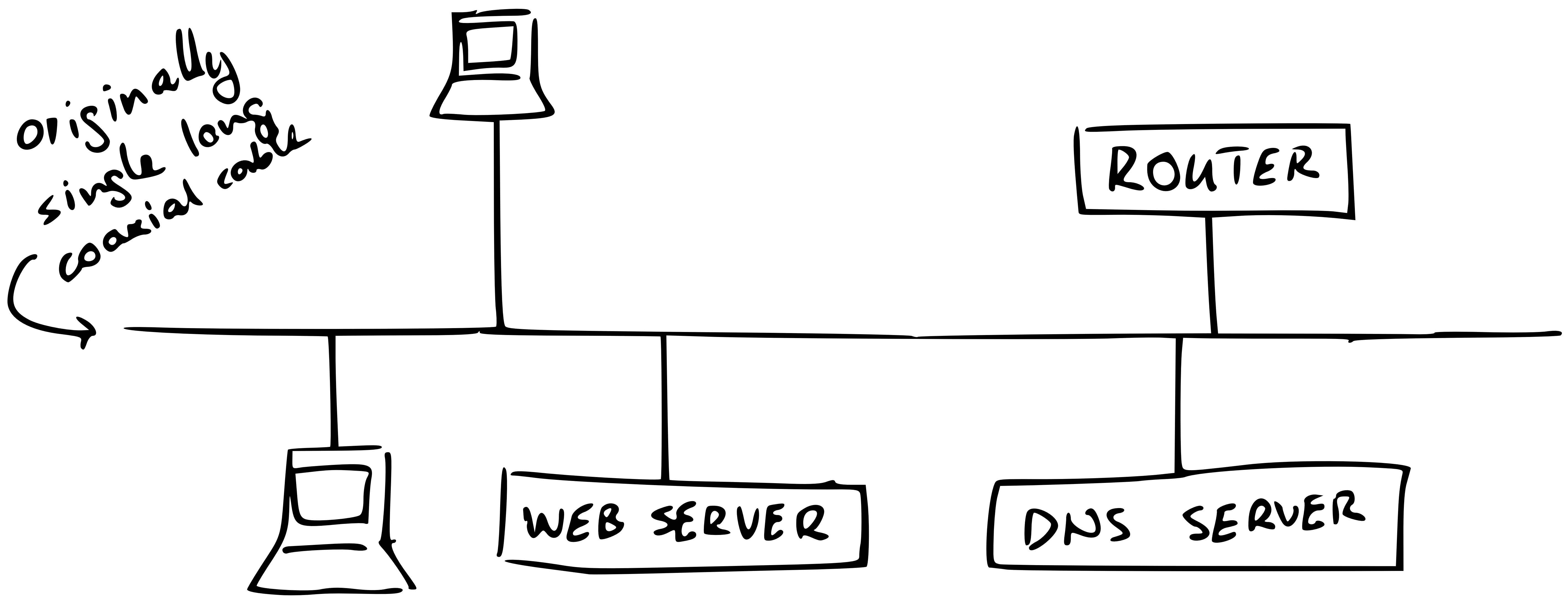
MAC addresses: 6 bytes, universally unique

EtherType gives layer 3 protocol in payload

- 0x0800: IPv4
- 0x0806: ARP
- 0x86DD: IPv6

Ethernet

Originally broadcast. Every local computer got every packet.



Switched Ethernet

With switched Ethernet, the switch *learns* at which physical port each MAC address lives based on MAC source addresses

If switch knows MAC address M is at port P, it will only send a packet for M out port P

If switch does not know which port MAC address M lives at, will broadcast to all ports

Ethernet

```
zakir@scratch-01:~$ ifconfig
ens160: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.216.2.64  netmask 255.255.192.0  broadcast 10.216.63.255
        inet6 fe80::250:56ff:fe86:b203  prefixlen 64  scopeid 0x20<link>
        ether 00:50:56:86:b2:03  txqueuelen 1000  (Ethernet)
        RX packets 1404151714  bytes 1784388363701 (1.7 TB)
        RX errors 0  dropped 73  overruns 0  frame 0
        TX packets 1155689210  bytes 6010503085464 (6.0 TB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Two Problems

Local: How does a host know what MAC address their destination has?

Internet: How does each router know where to send each packet next?

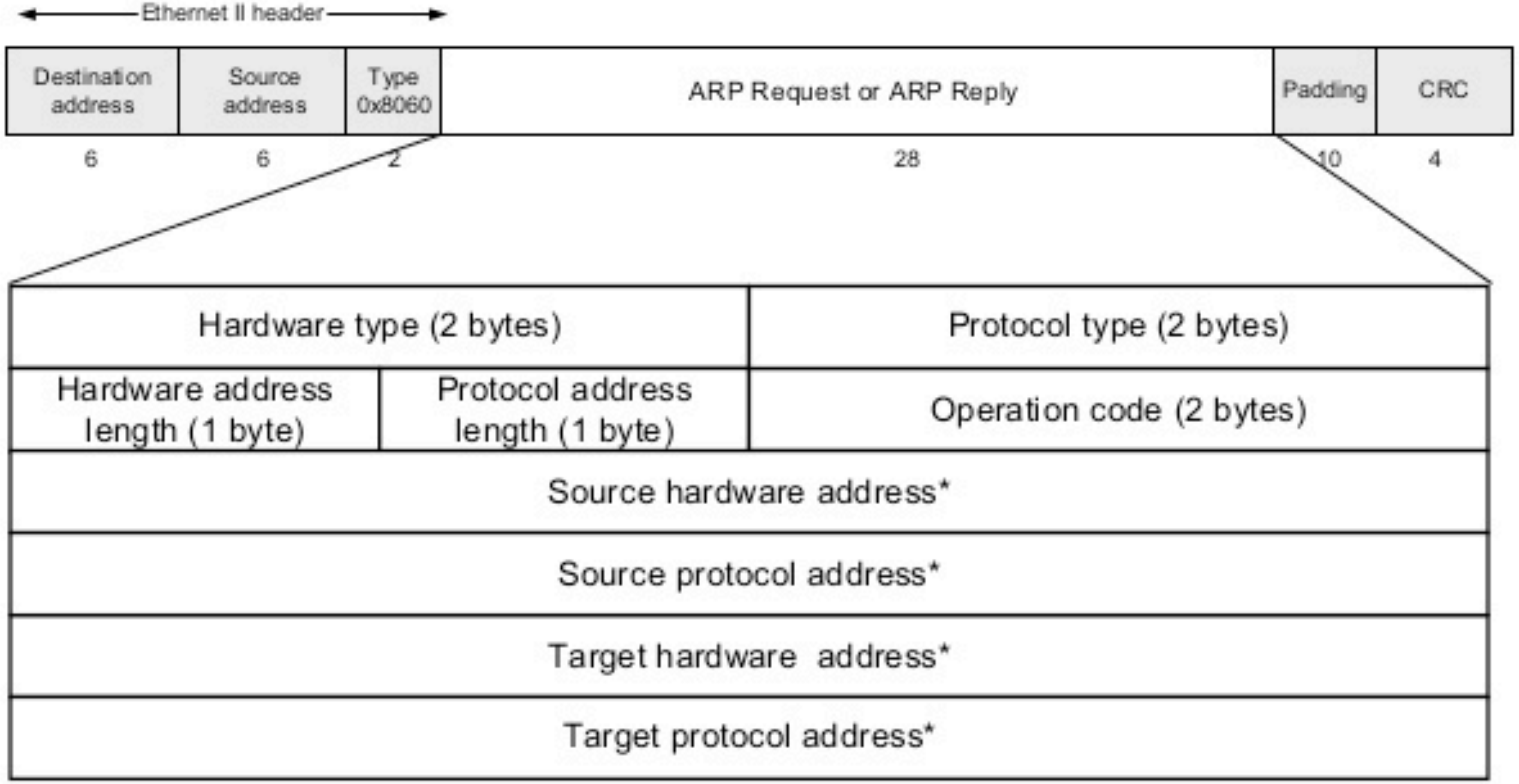
ARP: Address Resolution Protocol

ARP is a Network protocol that lets hosts map IP addresses to MAC addresses

Host who needs MAC address M corresponding to IP address N broadcasts an ARP packet to LAN asking, “who has IP address N?”

Host that has IP address N will reply, “IP *N* is at MAC address *M*.”

ARP Packet



* Note: The length of the address fields is determined by the corresponding address length fields

ARP Security

Any host on the LAN can send ARP requests and replies: *any host can claim to be another host on the local network!*

- This is called *ARP spoofing*

This allows any host X to force IP traffic between any two other hosts A and B to flow through X *(MitM!)*

- Claim $N_\mathbf{A}$ is at attacker's MAC address $M_\mathbf{X}$
- Claim $N_\mathbf{B}$ is at attacker's MAC address $M_\mathbf{X}$
- Re-send traffic addressed to $N_\mathbf{A}$ to $M_\mathbf{A}$, and vice versa

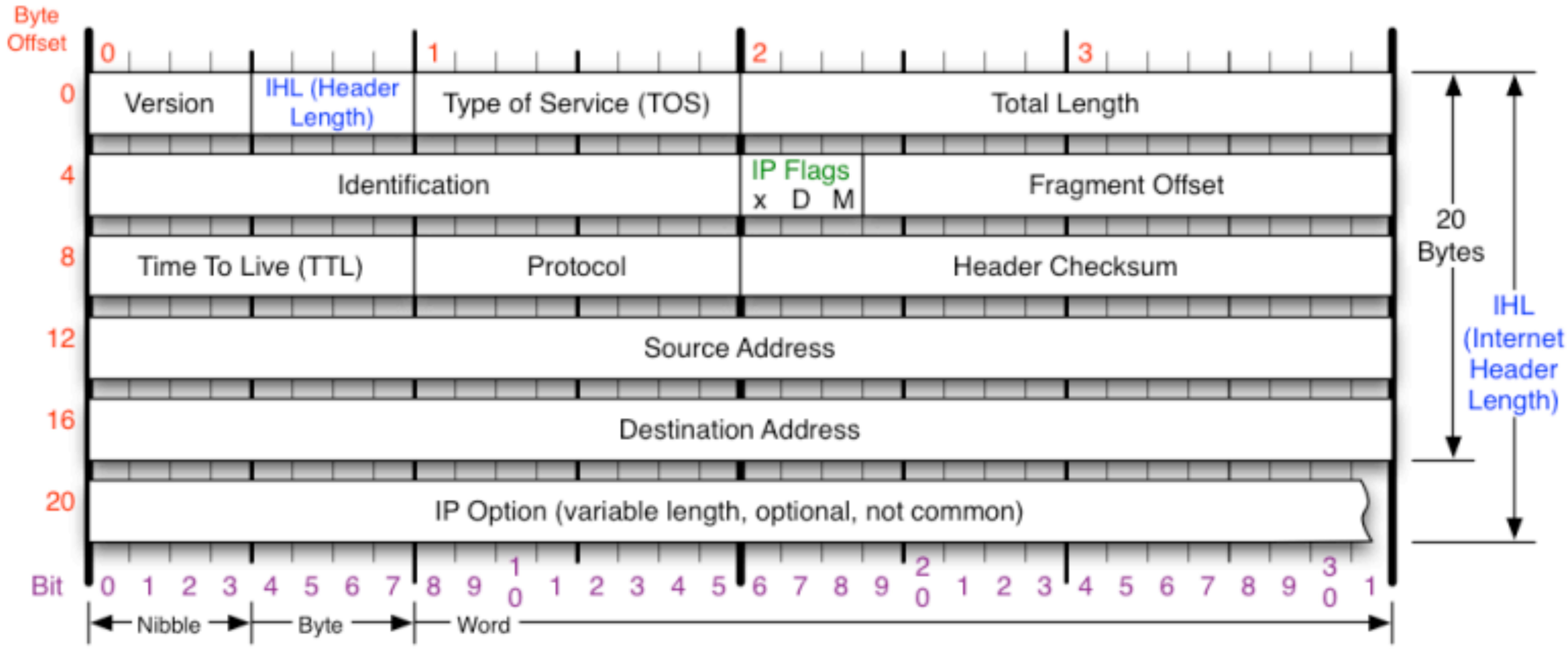
IP Addresses

IPv4: 32-bit host addresses
Written as 4 bytes in form A.B.C.D
where A,...,D are 8 bit integers in decimal
(called *dotted quad*) *e.g.* 192.168.1.1

IPv6: 128 bit host addresses
Written as 16 bytes in form AA:BB::XX:YY:ZZ
where AA,...,ZZ are 16 bit integers in hexadecimal
and :: implies zero bytes
e.g. 2620:0:e00:b::53 = 2620:0:e00:b:0:0:0:53

IPv4 Header

Instruct routers and hosts what to do with a packet
All values are filled in by the sending host

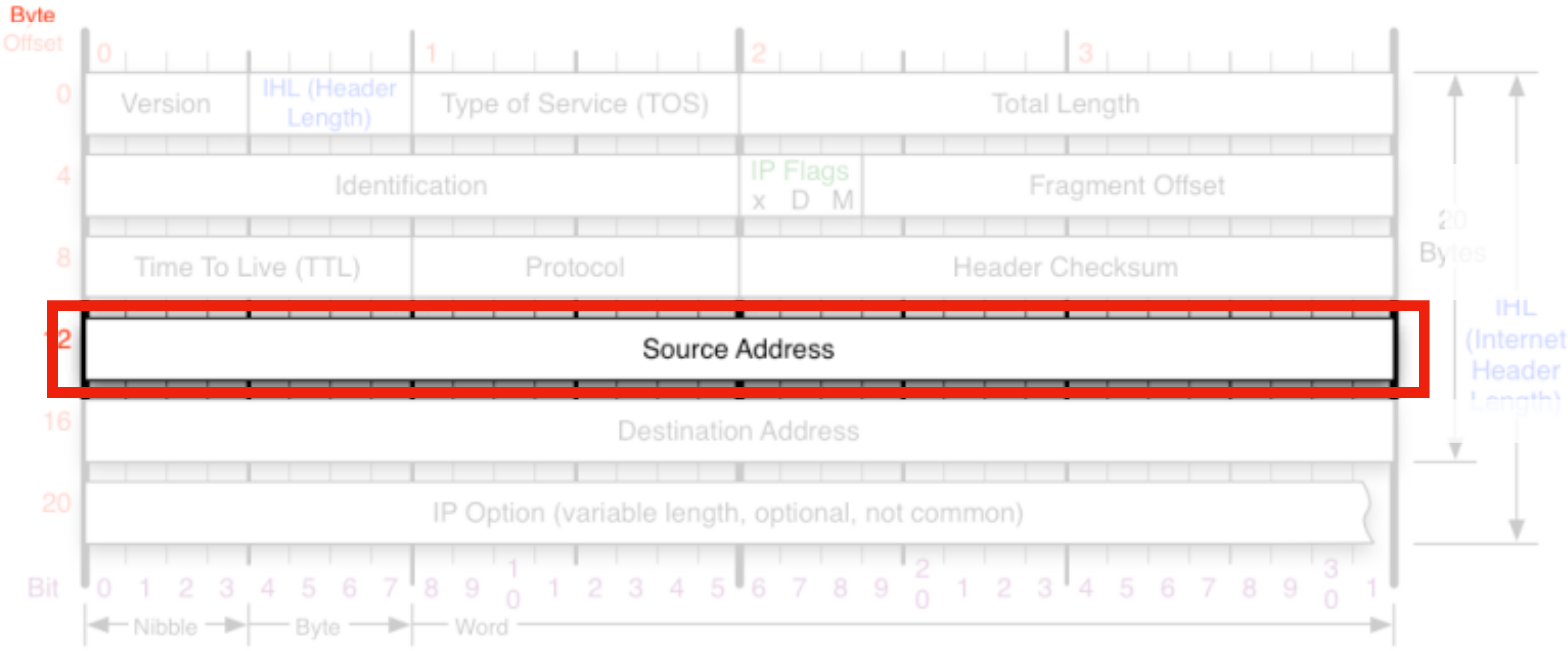


Destination Address

Sender sets destination address
Routers try to forward packet to that address

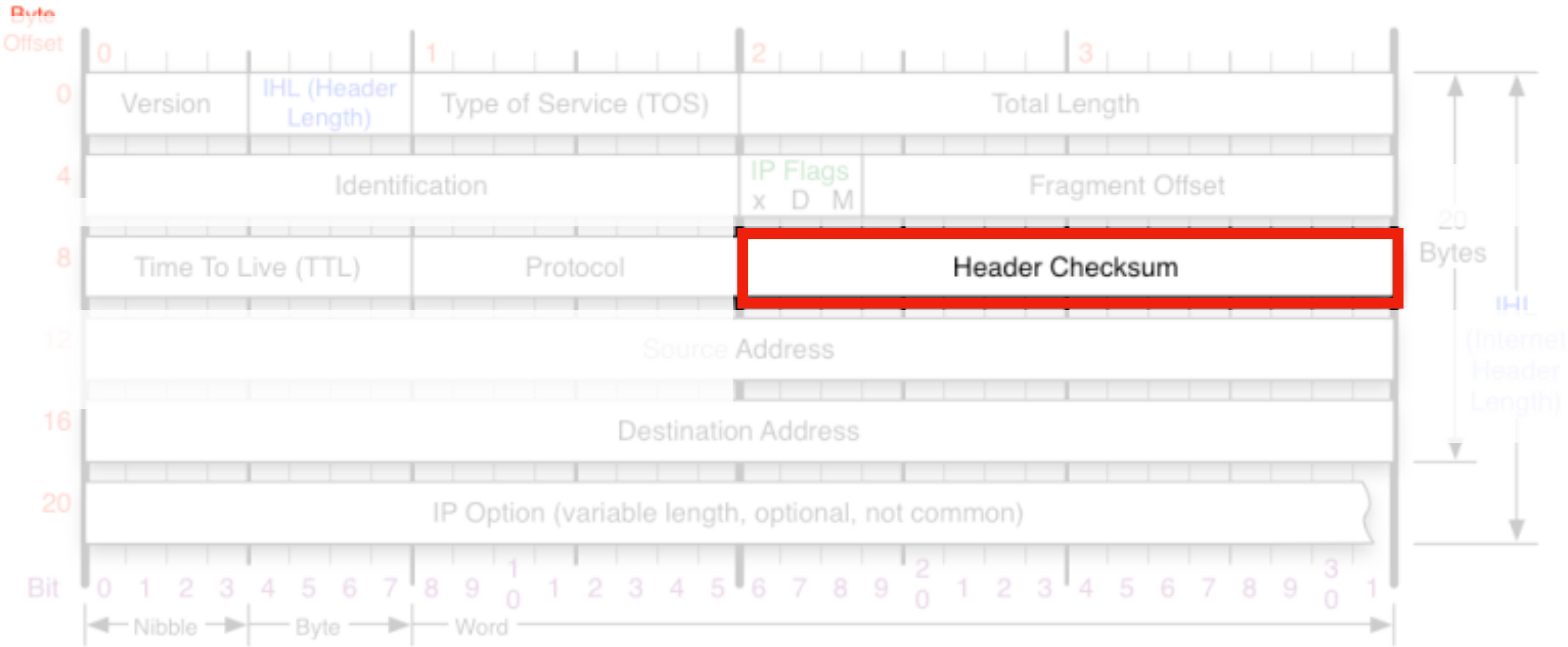
Source Address

Source Address (sender)
Sender fills in. Routers due not verify.



Checksum

16-bit Simple Header checksum (filled in by sender)



IP Security

Client is trusted to embed correct source IP

- Easy to override using lower level network sockets
- **Libnet:** a library for formatting raw packets with arbitrary IP headers

Anyone who owns their machine can send packets with arbitrary source IP

- Denial of Service Attacks
- Anonymous infection (if one packet)

Internet Protocol (IP)

Yes:

Routing. If host knows IP of destination host, route packet to it.

Fragmentation and reassembly: Split data into packets and reassemble

Error Reporting: (maybe, if you're lucky) tell source it dropped your packet

No:

Everything else. No ordering. No retransmission. No (real) error checking. No acknowledgement of receipt. No "connections". No security. Just packets.

Routing (BGP)

BGP (Border Gateway Protocol): protocol that allows routers to exchange information about their routing tables

Each router announces what it can route to all of its neighbors.

Every router maintains a global table of routes

Pakistan hijacks YouTube

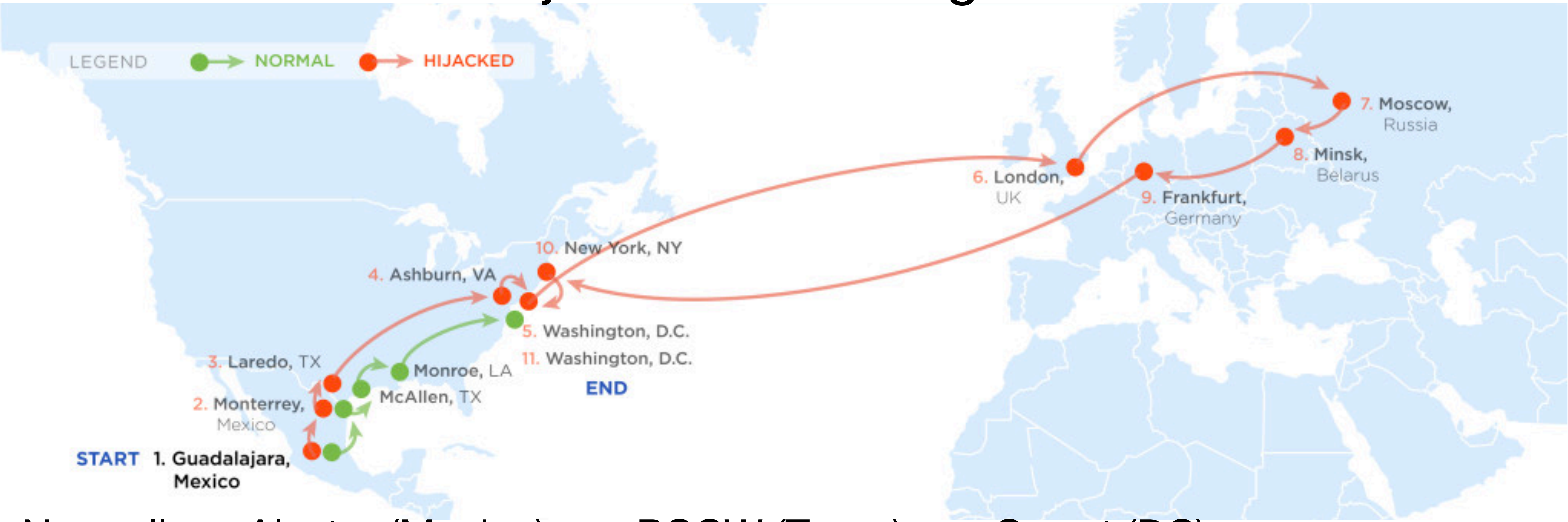
On 24 February 2008, Pakistan Telecom (AS 17557) began advertising a small part of YouTube’s (AS 36561) assigned network

PCCW (3491) did not validate Pakistan Telecom’s (17557) advertisement for 208.65.153.0/24

Youtube offline.

Example path hijack (source: Renesys 2013)

Feb 2013: Guadalajara ⟶ Washington DC via Belarus

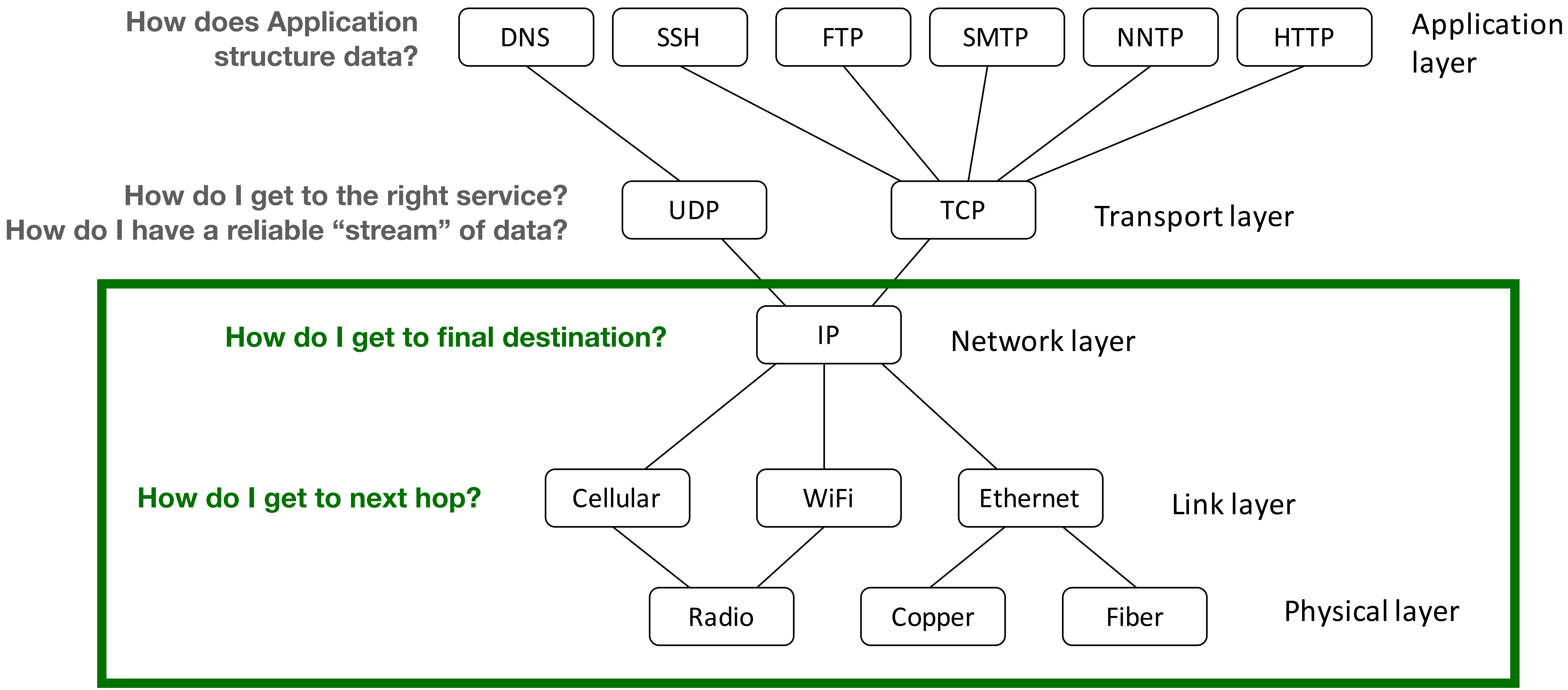


Normally: Alestra (Mexico) ⟶ PCCW (Texas) ⟶ Qwest (DC)

Reverse route (DC ⟶ Guadalajara) is unaffected:

Person browsing the Web in DC cannot tell by traceroute that HTTP responses are routed through Moscow

Protocol Layering

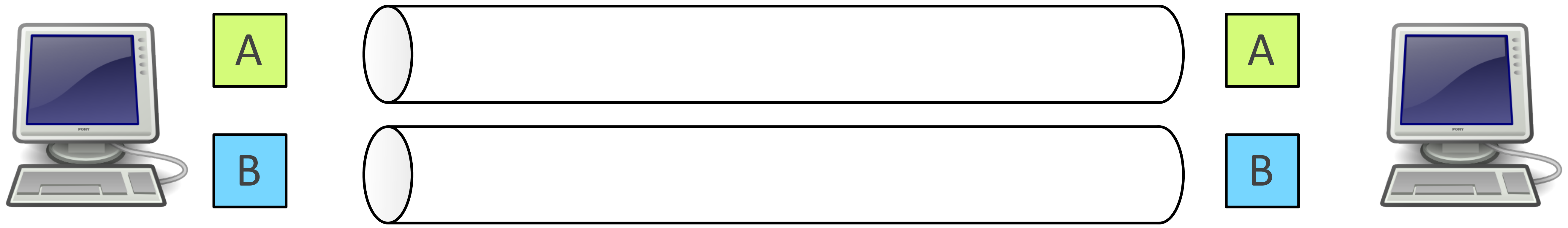


Ports

Each application on a host is identified by a *port number*

TCP connection established between port *A* on host *X* to port *B* on host *Y*
- Ports are 1–65535 (16 bits)

Some destination port numbers used for specific applications by convention



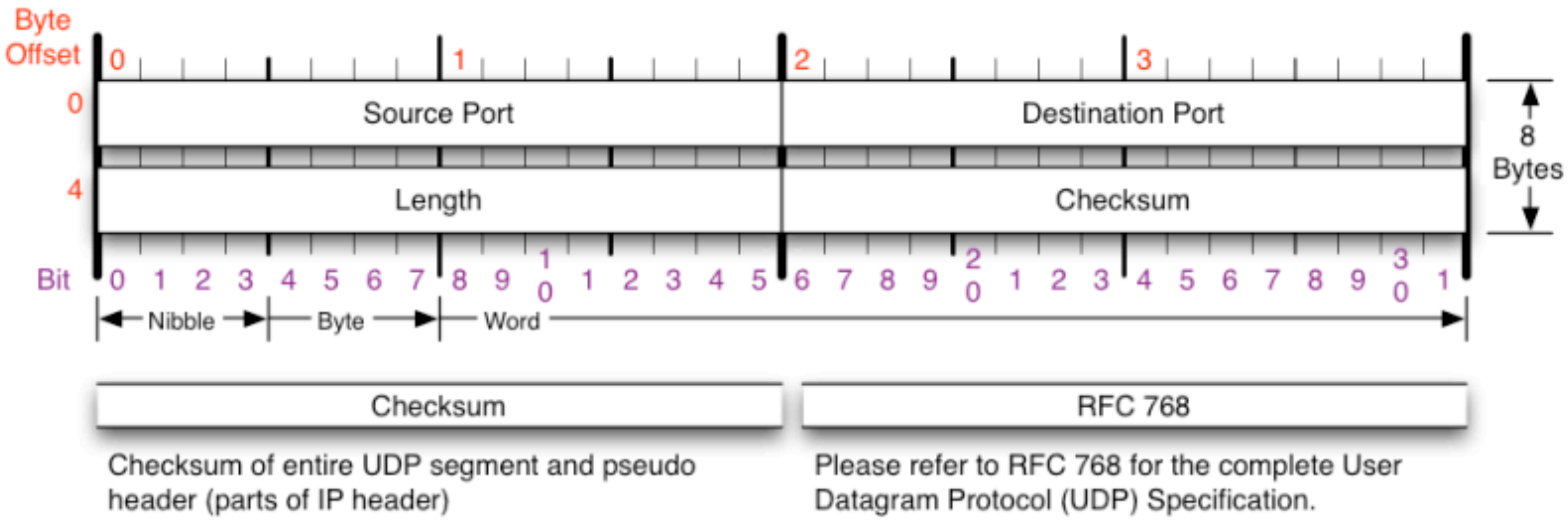
Common Ports

Port	Application
80	HTTP (Web)
443	HTTPS (Web)
25	SMTP (mail)
67	DHCP (host config)
22	SSH (secure shell)
23	Telnet

UDP (User Datagram Protocol)

User Datagram Protocol (UDP) is a transport layer protocol that is essentially a wrapper around IP

Adds ports to demultiplex traffic by application



From Packets to Streams

Most applications want a stream of bytes delivered reliably and in-order between applications on different hosts

Transmission Control Protocol (TCP) provides…

- Connection-oriented protocol with explicit setup/teardown
- Reliable in-order byte stream
- Congestion control

Despite IP packets being dropped, re-ordered, and duplicated

TCP Sequence Numbers

Two data streams in a TCP session, one in each direction

Bytes in data stream numbered with a 32-bit *sequence number*

Every packet has sequence number that indicates where data belongs

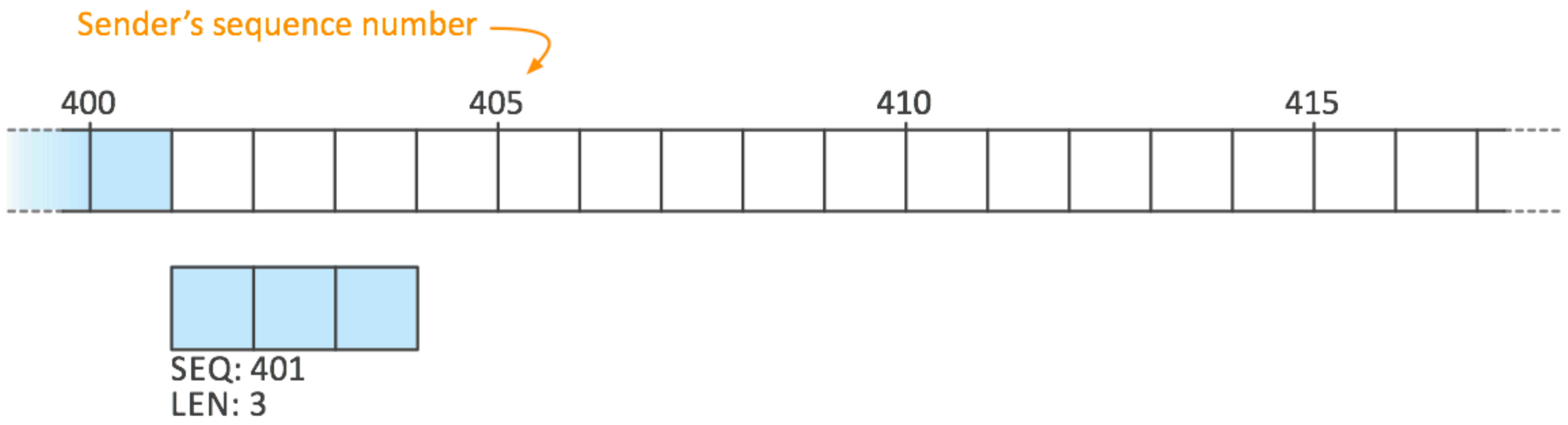
Receiver sends acknowledgement number that indicates data received

TCP Packet

Byte Offset
0
1
2
3
0
4
8
12
16
20
Source Port
Destination Port
Sequence Number
Acknowledgment Number
Offset
Reserved
TCP Flags
C E U A P R S F
Window
Checksum
Urgent Pointer
TCP Options (variable length, optional)
20 Bytes
Offse
Bit
0 1 2 3 4 5 6 7 8 9 10 1 2 3 4 5 6 7 8 9 20 1 2 3 4 5 6 7 8 9 30 1
Nibble
Byte
Word

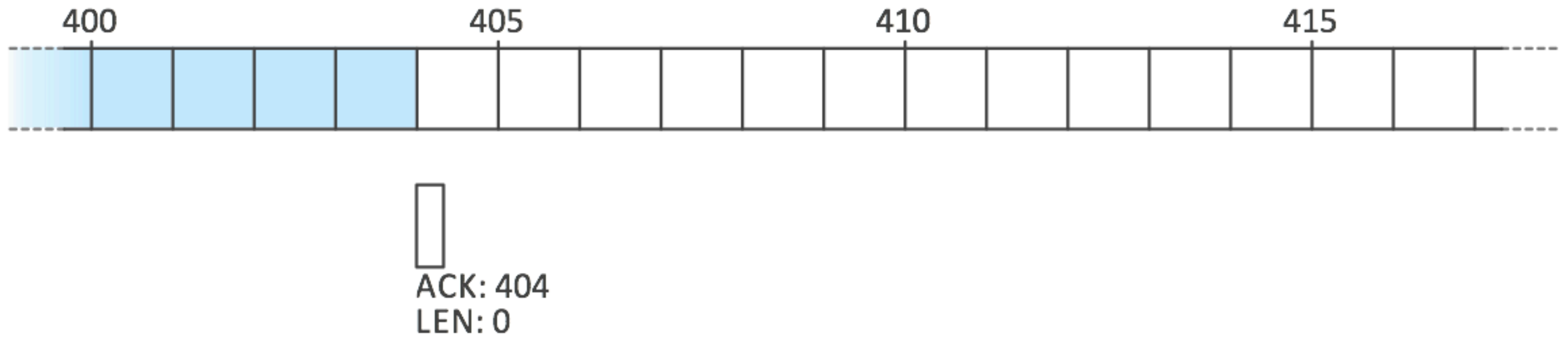
Transmission Control Protocol

- Sender sends 3 byte segment
- Sequence number indicates where data belongs in byte sequence (at byte 401)
  - *Note:* Wireshark shows *relative* sequence numbers



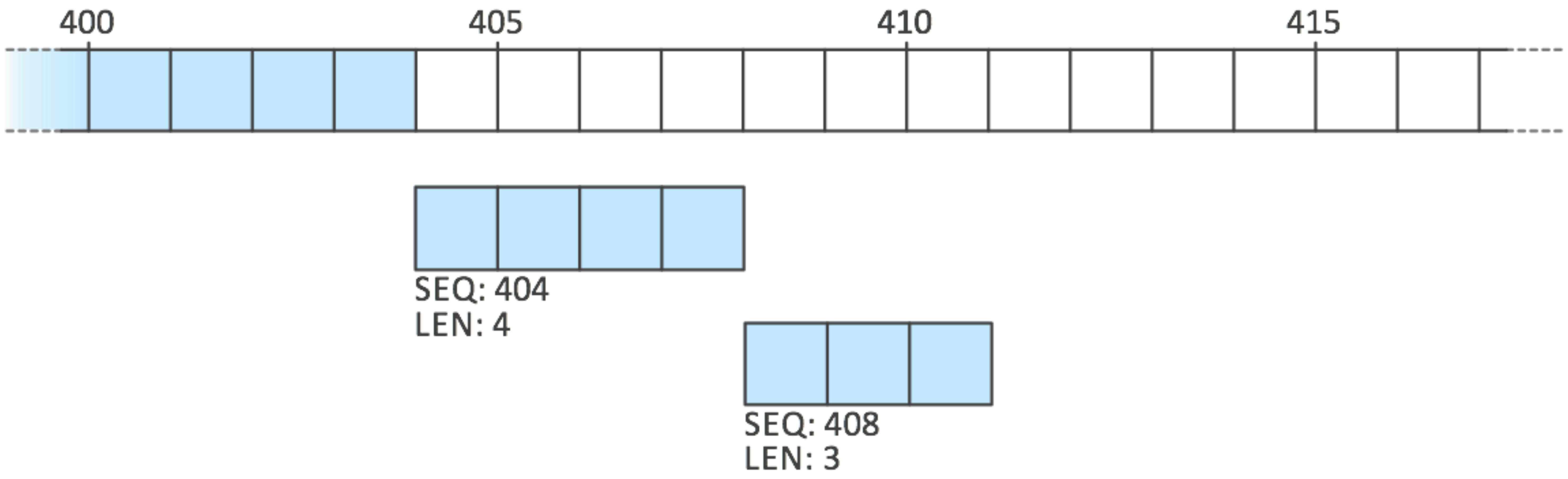
TCP Acknowledgement Numbers

- Receiver acknowledges received data
  - Sets ACK flag in TCP header
  - Sets acknowledgement number to indicate next expected byte in sequence



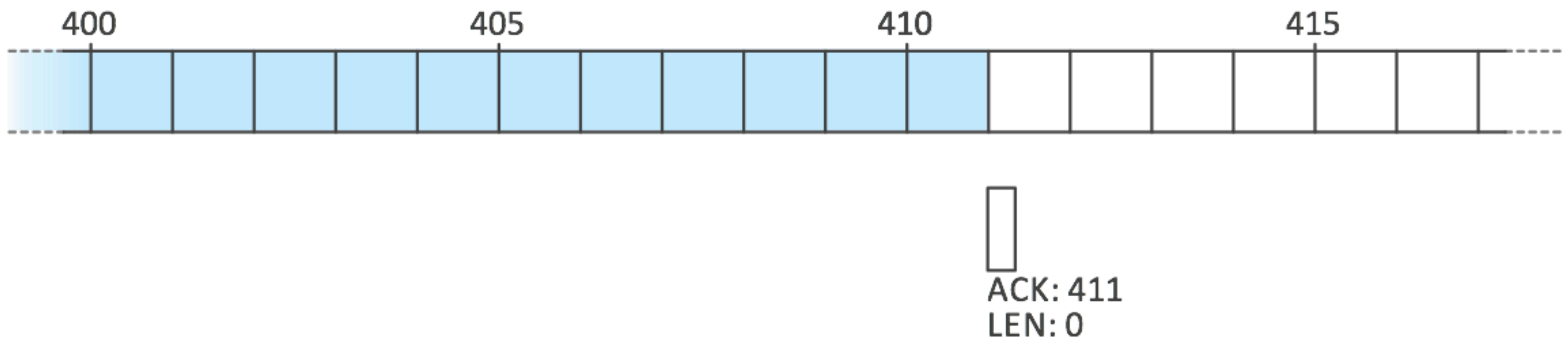
ACKing Multiple Segments

- Sender may send several segments before receiving acknowledgement



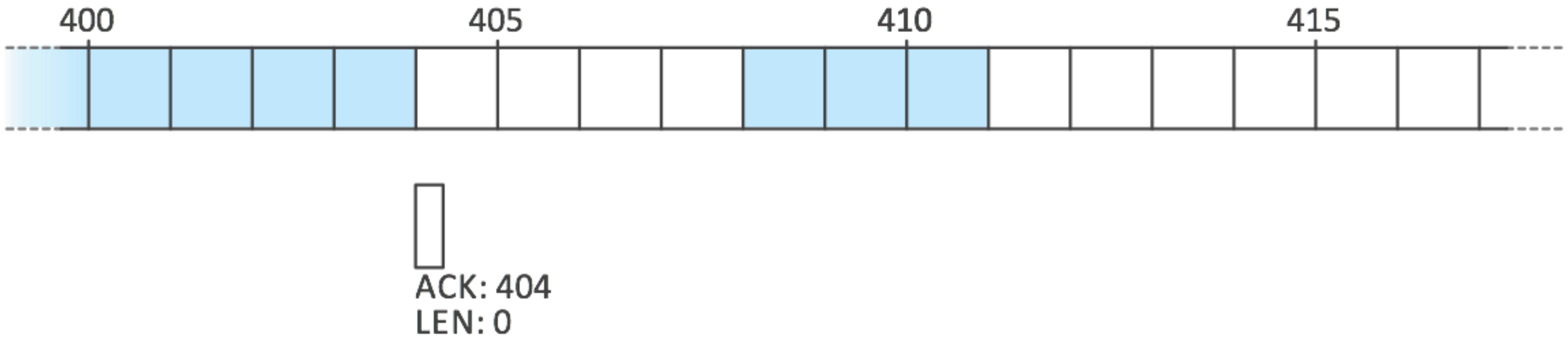
ACKing Multiple Segments

- Sender may send several segments before receiving acknowledgement
- Receiver always acknowledges with seq. no. of next expected byte



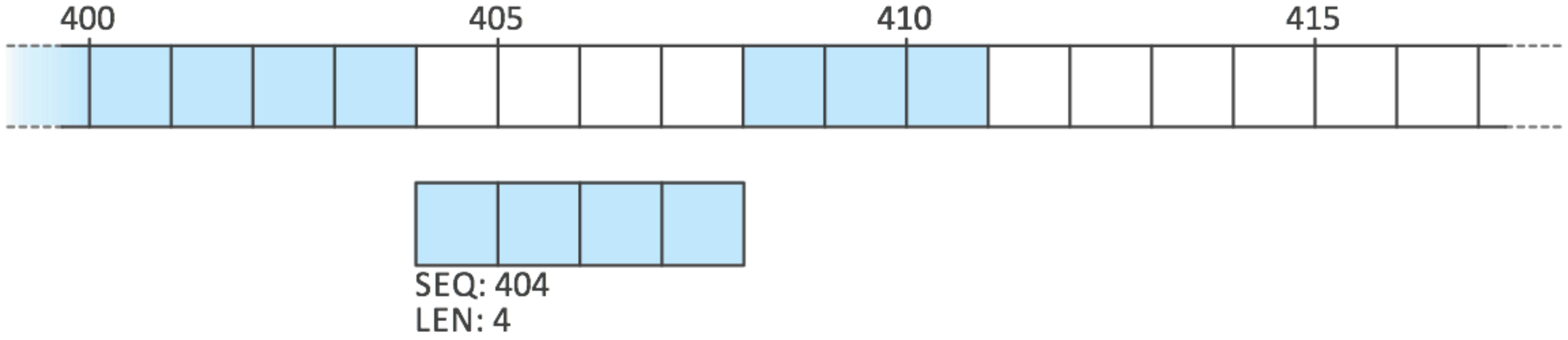
Transmission Control Protocol

- *What if the first packet is dropped in network?*
- Receiver always acknowledges with seq. no. of next expected byte



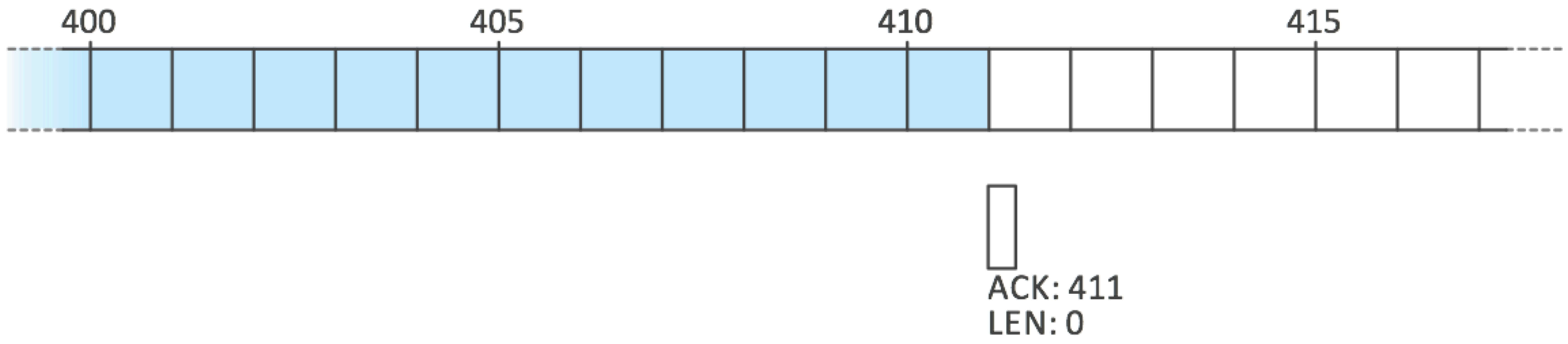
Transmission Control Protocol

- *What if the first packet is dropped in network?*
- Receiver always acknowledges with seq. no. of next expected byte
- Sender retransmits lost segment

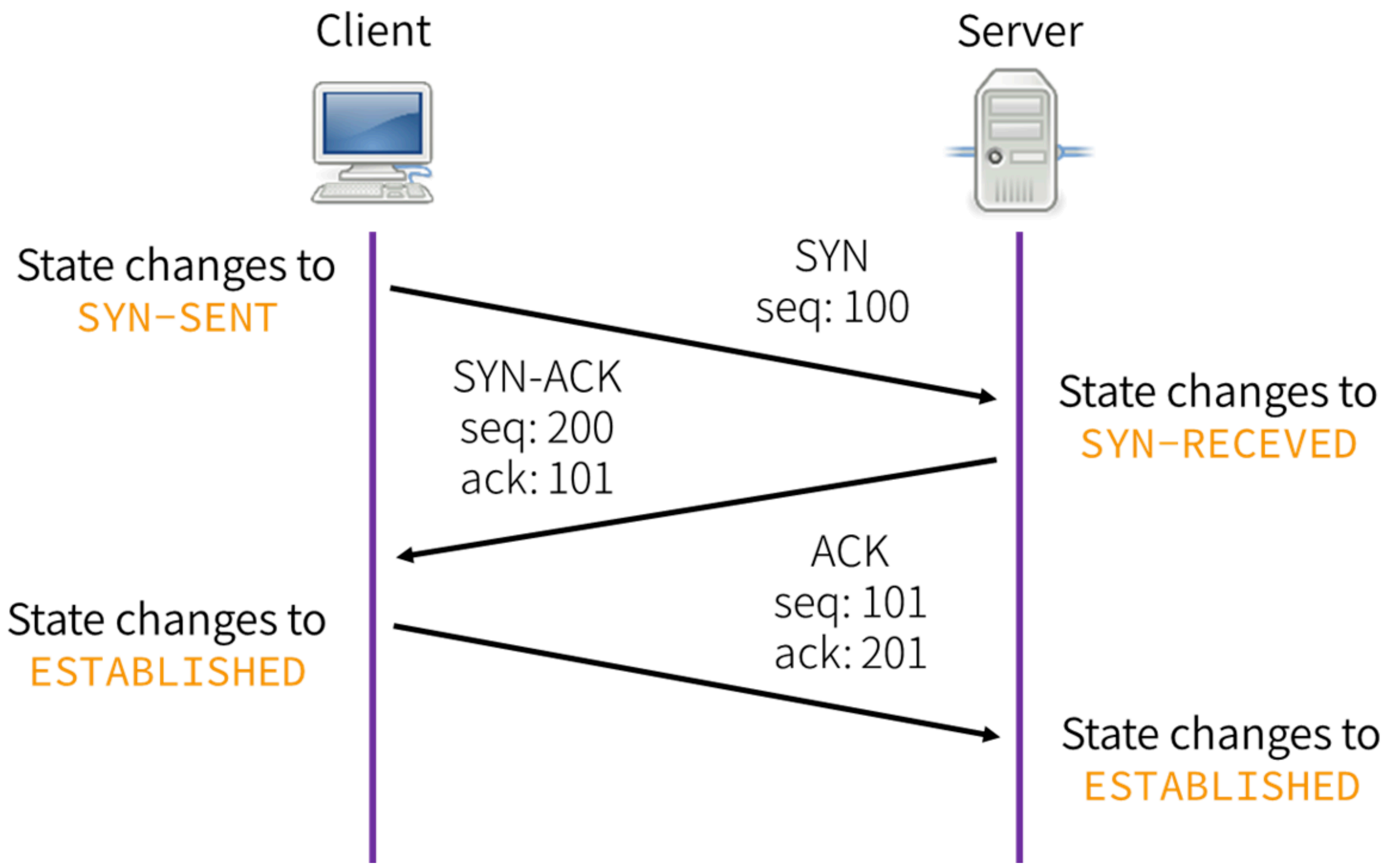


Transmission Control Protocol

- *What if the first packet is dropped in network?*
- Sender retransmits lost segment
- Receiver always acknowledges with seq. no. of next expected byte



TCP Three Way Handshake



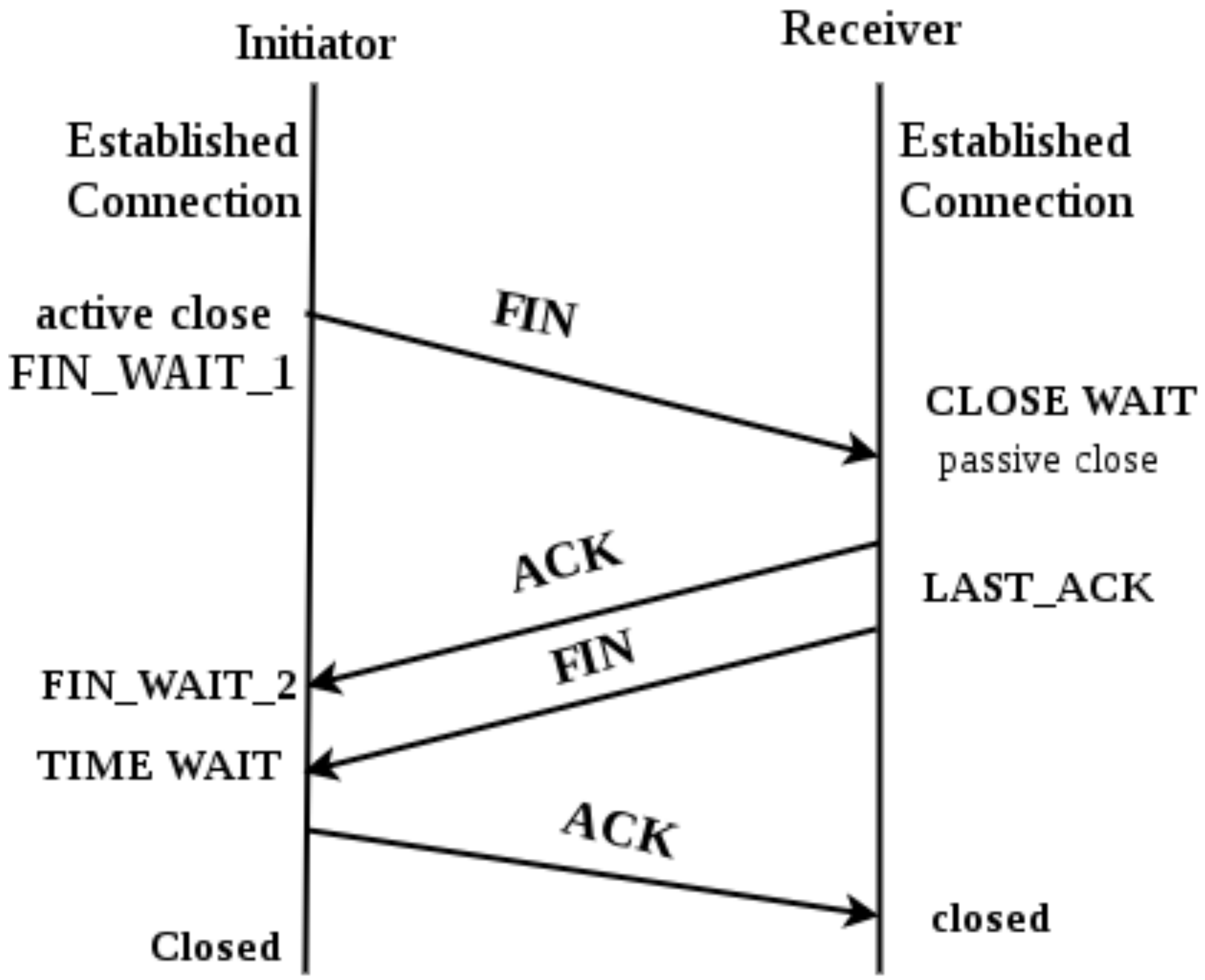
Ending a Connection

Sends packet with FIN flag set
  Must have ACK flag with valid seqnum

Peer receiving FIN packet acknowledges
  receipt of FIN packet with ACK

FIN “consumes” one byte of seq. number

Eventually other side sends packet with
  FIN flag set — terminates session



TCP Connection Reset

TCP designed to handle possibility of spurious TCP packets (e.g. from previous connections)

Packets that are invalid given current state of session generate a reset

- If a connection exists, it is torn down
- Packet with RST flag sent in response

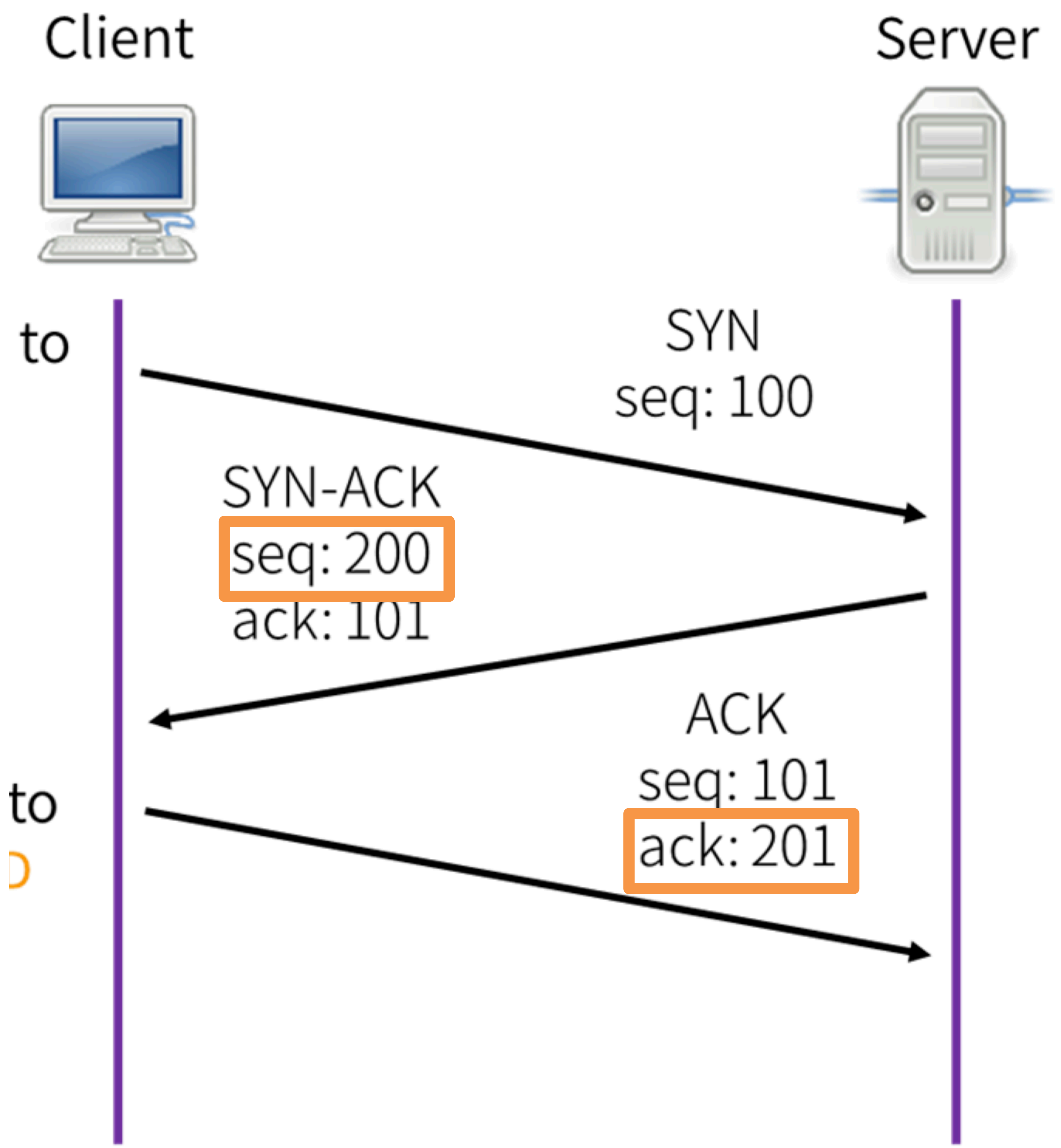
If a host receives a TCP packet with RST flag, it tears down the connection

TCP Connection Spoofing

Can we impersonate another host when *initiating* a connection?

Off-path attacker can send initial SYN to server …
… but cannot complete three-way handshake without seeing the server's sequence number

1 in 2^{32} chance to guess right if initial sequence number chosen uniformly at random



TCP Reset Attack

Can we reset an *existing* TCP connection?

Need to know port numbers (16 bits)
  Initiator's port number usually chosen random by OS
  Responder's port number may be well-known port of service

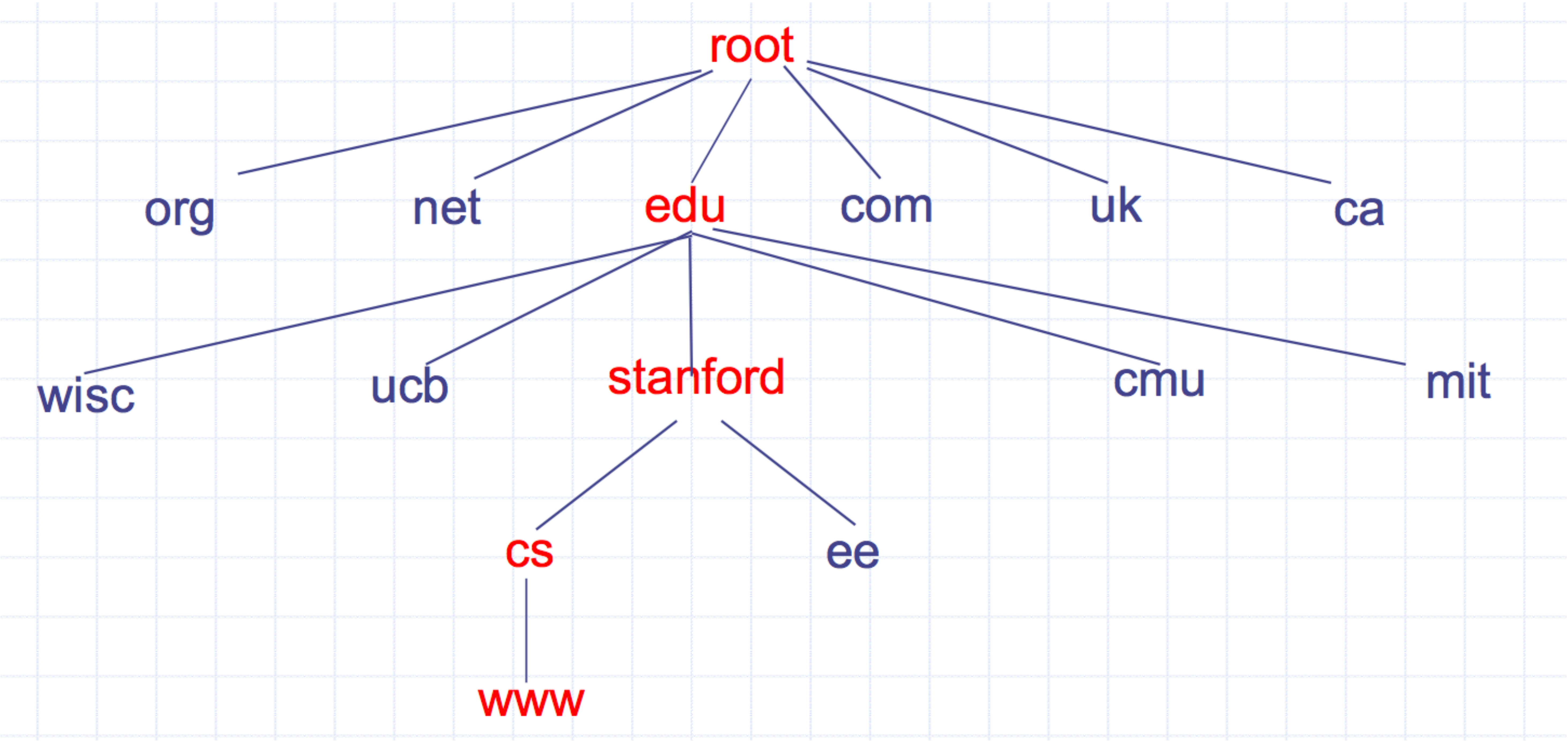
There is leeway in sequence numbers B will accept
  Must be within window size (32-64K on most modern OSes)

1 in $2^{16+32/W}$(where W is window size) chance to guess right

DNS — Domain Name Service

Application-layer protocols (and people) usually refer to Internet host by host name (e.g., google.com)

DNS is a delegatable, hierarchical name space



DNS Record

A DNS server has a set of records it authoritatively knows about

```
$ dig bob.ucsd.edu

;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 30439
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 6

;; QUESTION SECTION:
;bob.ucsd.edu.          IN  A

;; ANSWER SECTION:
bob.ucsd.edu.     3600  IN  A   132.239.80.176

;; AUTHORITY SECTION:
ucsd.edu.         3600  IN  NS  ns0.ucsd.edu.
ucsd.edu.         3600  IN  NS  ns1.ucsd.edu.
ucsd.edu.         3600  IN  NS  ns2.ucsd.edu.
```

DNS Root Name Servers

In total, there are 13 main **DNS root servers**, each of which is named with the letters 'A' to 'M'.

HOSTNAME	IP ADDRESSES	MANAGER
a.root-servers.net	198.41.0.4, 2001:503:ba3e::2:30	VeriSign, Inc.
b.root-servers.net	199.9.14.201, 2001:500:200::b	University of Southern California (ISI)
c.root-servers.net	192.33.4.12, 2001:500:2::c	Cogent Communications
d.root-servers.net	199.7.91.13, 2001:500:2d::d	University of Maryland
e.root-servers.net	192.203.230.10, 2001:500:a8::e	NASA (Ames Research Center)
f.root-servers.net	192.5.5.241, 2001:500:2f::f	Internet Systems Consortium, Inc.
g.root-servers.net	192.112.36.4, 2001:500:12::d0d	US Department of Defense (NIC)
h.root-servers.net	198.97.190.53, 2001:500:1::53	US Army (Research Lab)
i.root-servers.net	192.36.148.17, 2001:7fe::53	Netnod
j.root-servers.net	192.58.128.30, 2001:503:c27::2:30	VeriSign, Inc.
k.root-servers.net	193.0.14.129, 2001:7fd::1	RIPE NCC
l.root-servers.net	199.7.83.42, 2001:500:9f::42	ICANN
m.root-servers.net	202.12.27.33, 2001:dc3::35	WIDE Project

Caching

DNS responses are cached

- Quick response for repeated translations
- NS records for domains also cached

DNS negative queries are cached

- Save time for nonexistent sites, e.g. misspelling

Cached data periodically times out

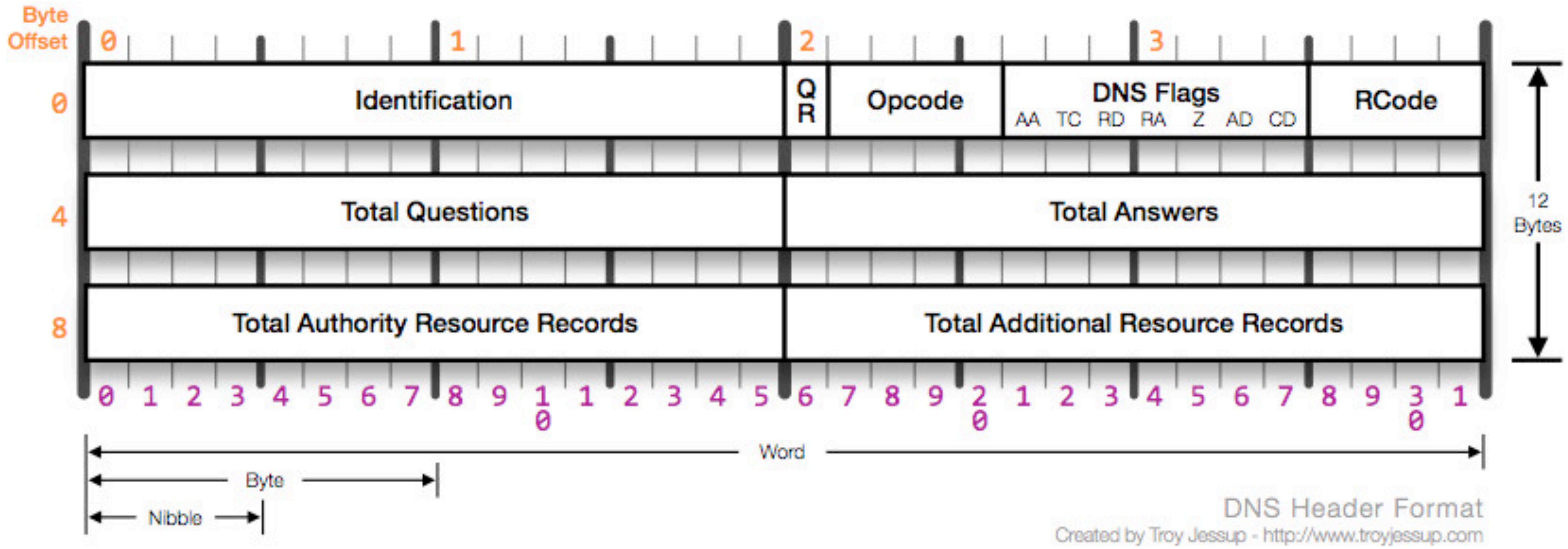
- Lifetime (TTL) of data controlled by owner of data
- TTL passed with every record

DNS Packet

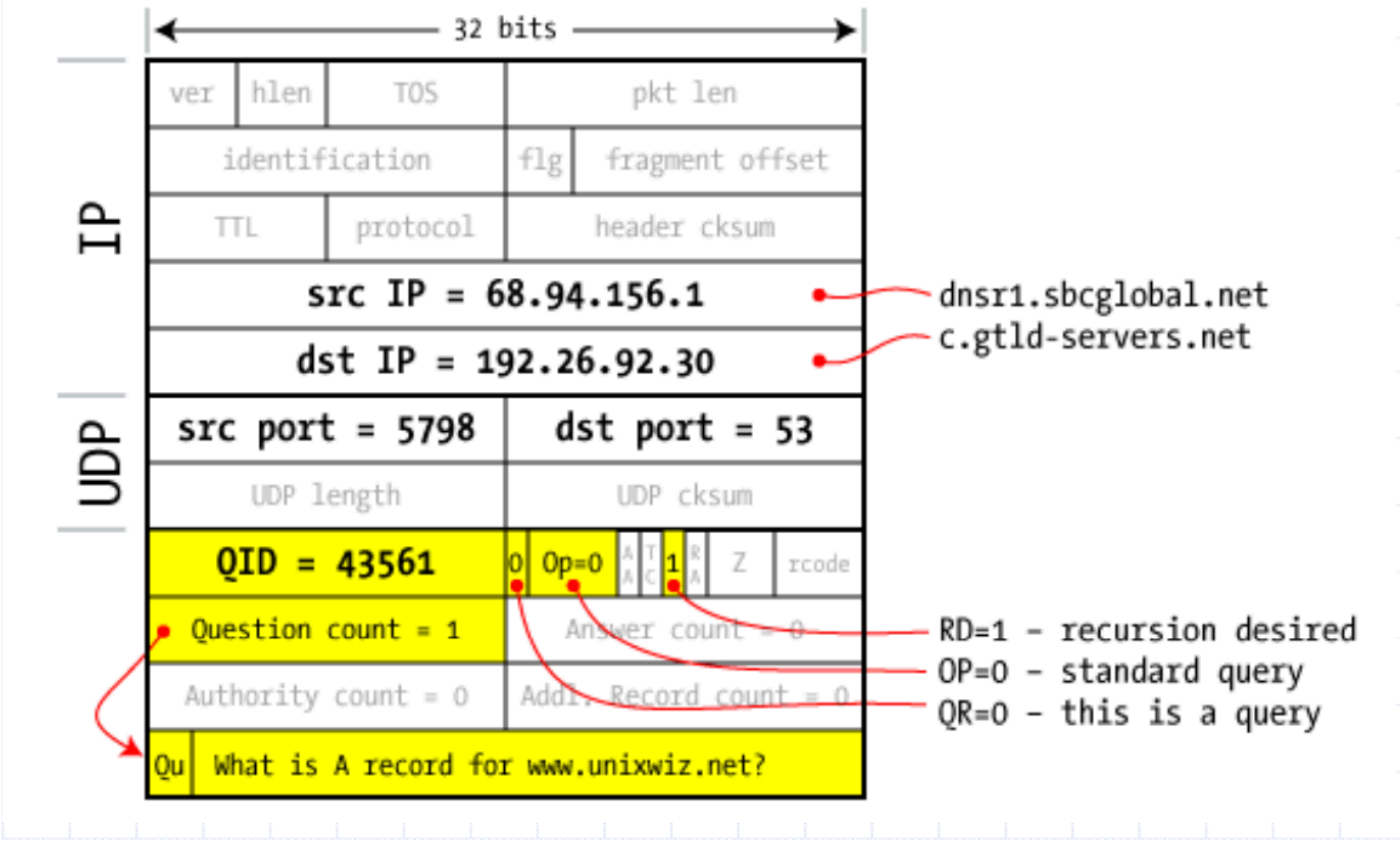
DNS requests sent over UDP

Four sections: questions, answers, authority, additional records

Query ID:
16 bit random value
Links response to query

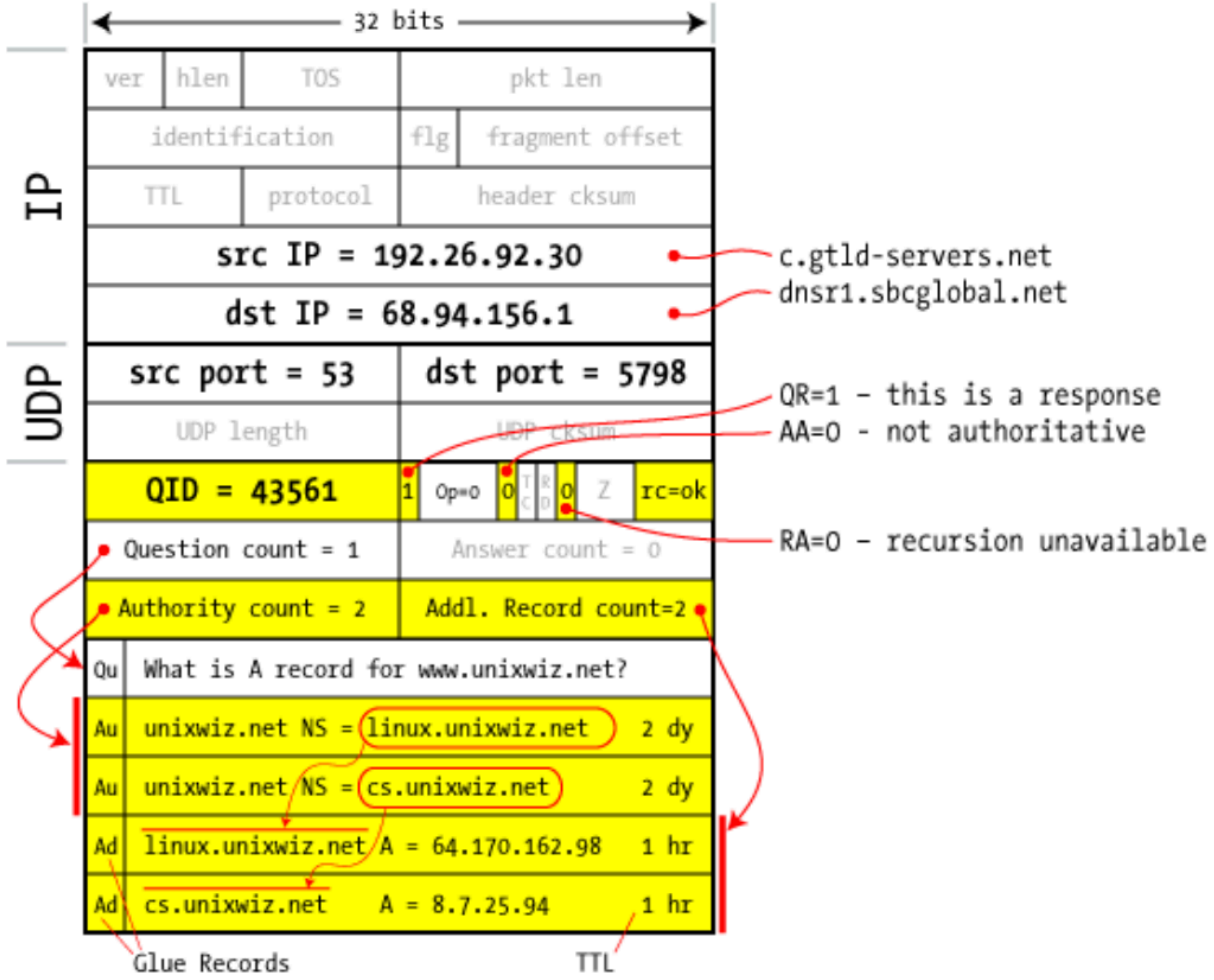


Request



32 bits
IP
UDP
ver
hlen
TOS
pkt len
identification
flg
fragment offset
TTL
protocol
header cksum
src IP = 68.94.156.1
dst IP = 192.26.92.30
dnsr1.sbcglobal.net
c.gtld-servers.net
src port = 5798
dst port = 53
UDP length
UDP cksum
QID = 43561
0
Op=0
AA
TC
1
RA
Z
rcode
Question count = 1
Answer count = 0
Authority count = 0
Addl. Record count = 0
Qu
What is A record for www.unixwiz.net?
RD=1 - recursion desired
OP=0 - standard query
QR=0 - this is a query

Response


32 bits
IP
UDP
ver
hlen
TOS
pkt len
identification
flg
fragment offset
TTL
protocol
header cksum
src IP = 192.26.92.30
dst IP = 68.94.156.1
src port = 53
dst port = 5798
UDP length
UDP cksum
QID = 43561
1
Op=0
0
TC
RD
0
Z
rc=ok
Question count = 1
Answer count = 0
Authority count = 2
Addl. Record count=2
Qu
What is A record for www.unixwiz.net?
Au
unixwiz.net NS = linux.unixwiz.net
2 dy
Au
unixwiz.net NS = cs.unixwiz.net
2 dy
Ad
linux.unixwiz.net A = 64.170.162.98
1 hr
Ad
cs.unixwiz.net A = 8.7.25.94
1 hr
Glue Records
TTL
c.gtld-servers.net
dnsr1.sbcglobal.net
QR=1 - this is a response
AA=0 - not authoritative
RA=0 - recursion unavailable

Authoritative Response