IPv6 header format:

<table>
<tr><td colspan="2">version
4</td><td colspan="2">traffic class
8</td><td colspan="3">flow label
20</td></tr>
<tr><td colspan="3">payload length
16</td><td colspan="2">next header
8</td><td colspan="2">hop limit
8</td></tr>
<tr><td colspan="7">source address
128</td></tr>
<tr><td colspan="7">destination address
128</td></tr>
</table>

- traffic class: similar role as TOS field in IPv4
- flow label: flow label + source address

  $\rightarrow$ per-flow traffic management

  $\rightarrow$ significant extra bits
- next header: similar to IPv4 protocol field

  $\rightarrow$ plus double duty for option headers
- note missing fields

- Dynamically assigned IP addresses

→ share an IP address pool

→ reusable

→ e.g., DHCP (dynamic host configuration protocol)

→ UDP-based client/server protocol (ports 67/68)

→ note: process/daemon based service

→ used by access ISPs, enterprises, etc.

→ where are the IP address savings?

Old days of non-permanent dial-up modems . . .

- Network address translation (NAT)

→ dynamically assigned + address translation

→ private vs. public IP address

→ private: Internet routers discard them

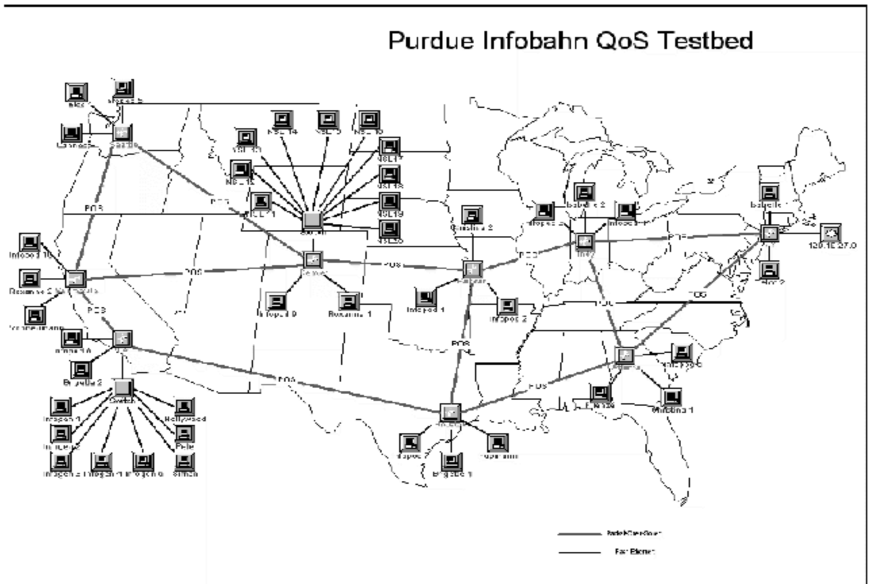
→ e.g., 192.168.0.0 is private

→ 10.x.x.x are also private

→ useful for home networks, small businesses

→ also industry and university research labs

Example: private network testbed (HAAS G50)



- routers have 10.0.0.0/8 addresses

  → each interface: a separate subnet

- only one of the routers connected to Internet

  → 128.10.27.0/24 address

- PCs connected to routers are dual-homed

  → 10.0.0.0/8 address & 128.10.27.0/24 address

  → dual-homed forwarding

- NAPT (NAT + port)

→ variant of NAT: borrow src port numbers as address bits

→ e.g.: 192.168.10.10 and 192.168.10.11 map to 128.10.27.10

→ but 192.168.10.10 maps to 128.10.26.10:6001

→ and 192.168.10.11 maps to 128.10.26.10:6002

→ huge increase in address space

→ popular technique used by ISPs

→ issues: e.g., NAT traversal

⟶ proxies/relays

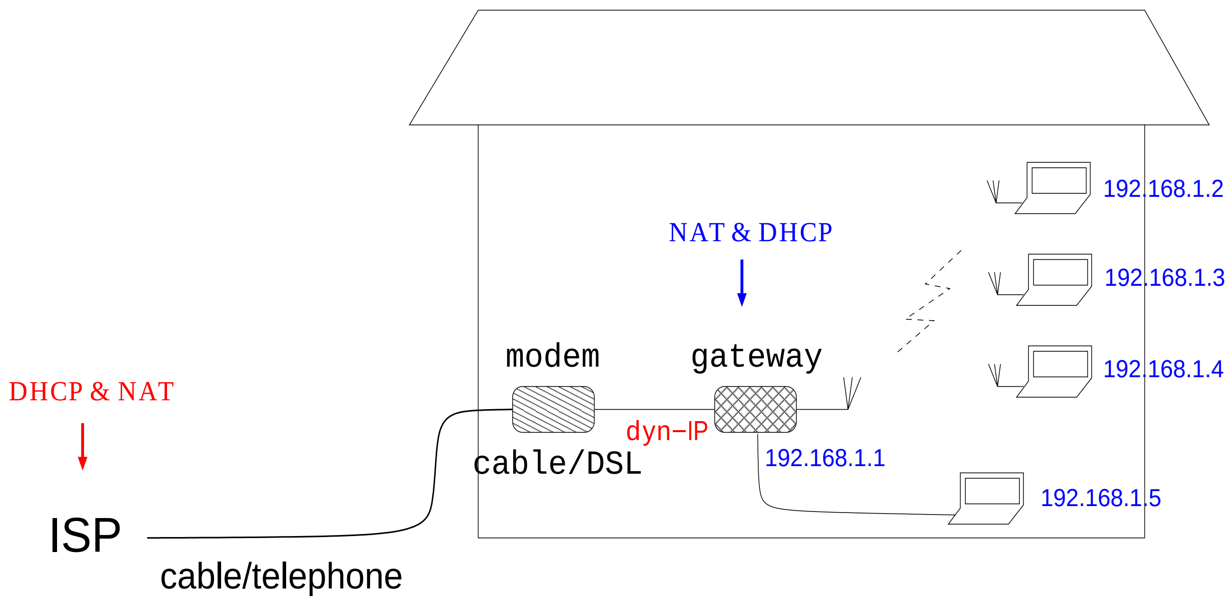
⟶ e.g., UDP/TCP hole punching (incl. VoIP apps)

⟶ enhanced gateway capabilities

⟶ e.g., IGD (Internet Gateway Device) in UPnP

Ex.: SOHO (small office/home office)

$\longrightarrow$ now: home networking



- dynamic IP address provided by ISP is shared through NAT
- IANA (Internet Assigned Numbers Authority)

  $\rightarrow$ non-routable: e.g., 192.168.0.0/16, 10.0.0.0/8

CIDR and dynamically assigned IP addresses with NAPT

→ significant increase of Internet's effective address space

→ saved the day

→ but last block allocated by IANA early 2011

→ back to address space crunch?

→ a number of technical and performance issues

→ 40-byte header

→ not backward compatible with IPv4

→ must use separate compatibility mechanisms (e.g., tunneling, hybrid sockets)

→ some companies have not-so-pleasant memories

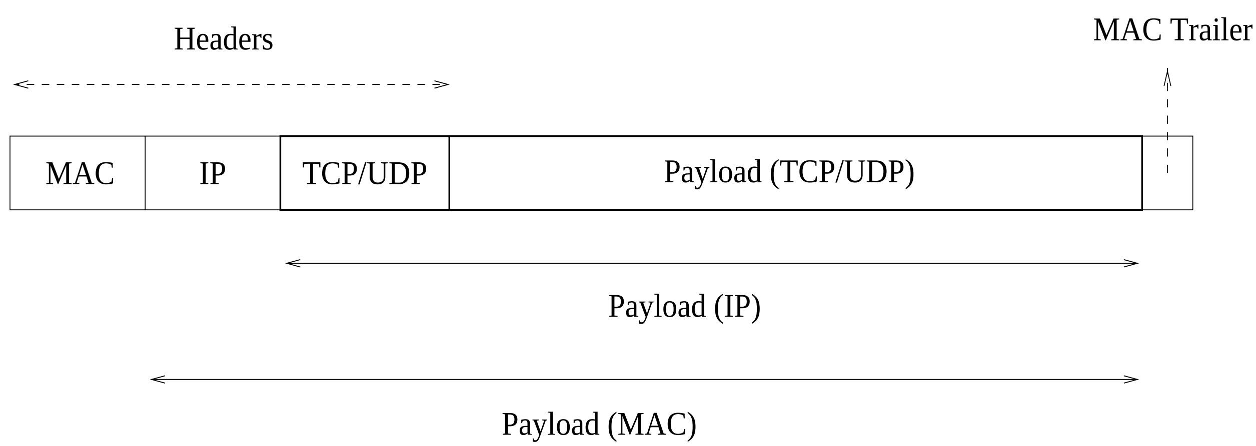
Transplort Protocols: TCP and UDP

→ end-to-end protocol

→ runs on top of network layer protocols

→ treat network layer & below as black box

Three-level encapsulation:



→ meaning of protocol "stack": push/pop headers

→ common TCP payload: HTTP

Network layer (IP) assumptions:

- unreliable
- out-of-order delivery
- absence of QoS guarantees (delay, throughput, etc.)
- insecure (IPv4)

  → IPsec (native in IPv6)

Additional performance properties:

- works "ok"
- can break down under high load conditions

  → flash crowds

  → DoS and worm attack
- wide behavioral range

  → sometimes good, so so, or bad

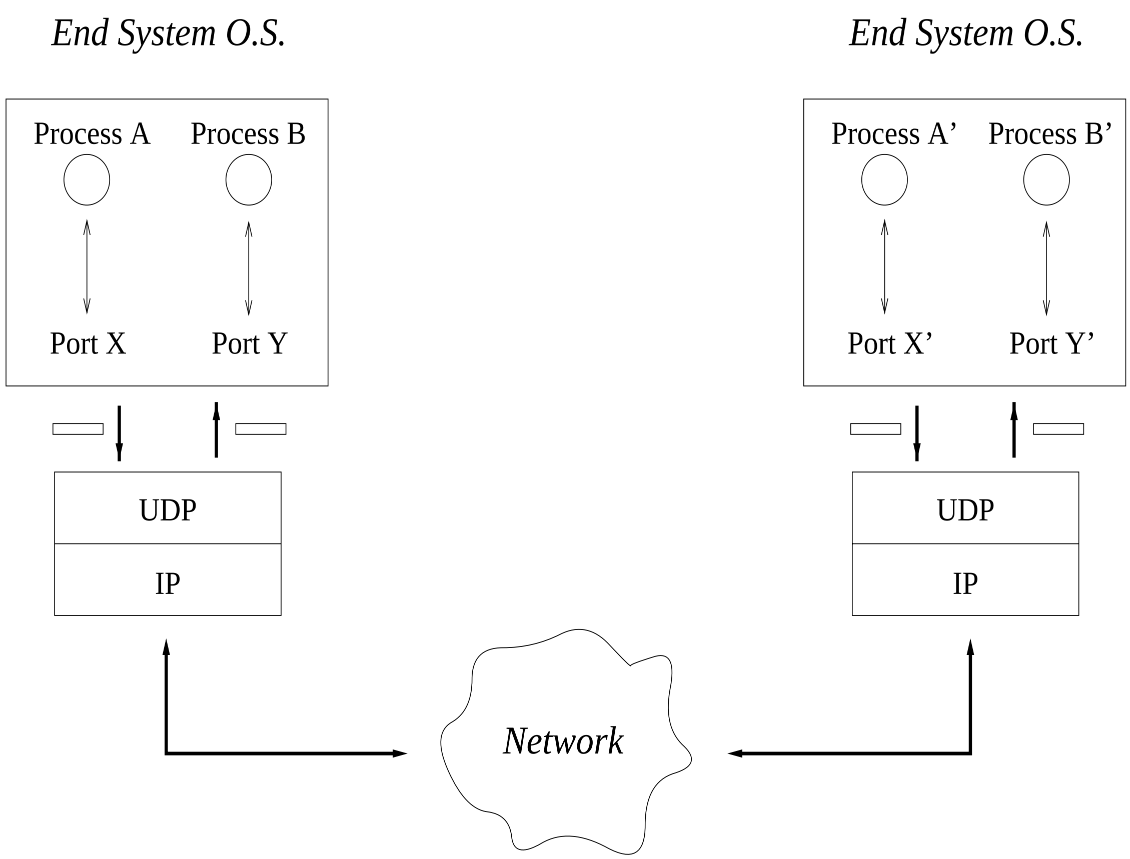
  → multitude of causes (e.g., end systems)

Goal of UDP (User Datagram Protocol):

$\longrightarrow$ process identification

$\longrightarrow$ port number as demux key

$\longrightarrow$ minimal support beyond IP



UDP packet format:

2	2
Source Port	Destination Port
Length	Checksum
Payload	

Checksum calculation (pseudo header):

<table>
<tr><th colspan="3">4</th></tr>
<tr><td colspan="3">Source Address</td></tr>
<tr><td colspan="3">Destination Address</td></tr>
<tr><td>00 ··· 0</td><td>Protocol</td><td>UDP Length</td></tr>
</table>

$\longrightarrow$ pseudo header, UDP header and payload

UDP usage:

- multimedia streamining

  → lean and nimble

  → at minimum requires process identification

  → congestion control carried out above UDP

- stateless client/server applications

  → persistent state a hinderance

  → lightweight

Goals of TCP (Transmission Control Protocol):

- process identification
- reliable communication: ARQ
- speedy communication: congestion control
- segmentation

$\longrightarrow$ connection-oriented, i.e., stateful

$\longrightarrow$ complex mixture of functionalities

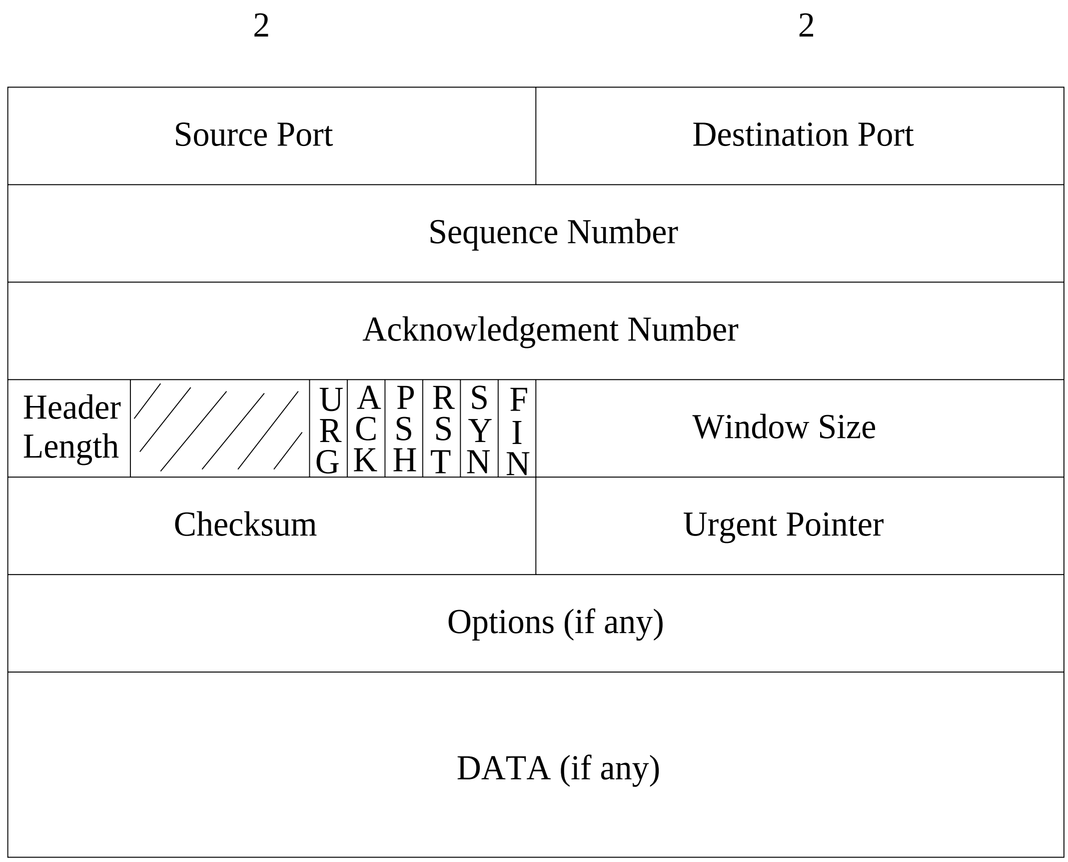
Segmentation task: provide "stream" interface to higher level protocols

$\longrightarrow$ exported semantics: contiguous byte stream

$\longrightarrow$ recall ARQ

- segment stream of bytes into blocks of fixed size
- segment size determined by TCP MTU (Maximum Transmission Unit)
- actual unit of transmission in ARQ

TCP packet format:



- Sequence Number: position of first byte of payload
- Acknowledgement: next byte of data expected (receiver)
- Header Length (4 bits): 4 B units
- URG: urgent pointer flag
- ACK: ACK packet flag
- PSH: override TCP buffering
- RST: reset connection
- SYN: establish connection
- FIN: close connection
- Window Size: receiver's advertised window size
- Checksum: prepend pseudo-header
- Urgent Pointer: byte offset in current payload where urgent data begins
- Options: MTU; take min of sender & receiver (default 556 B)

Checksum calculation (pseudo header):

4

Source Address		
Destination Address		
00 · · · 0	Protocol	TCP Segment Length

$\longrightarrow$ pseudo header, TCP header and payload