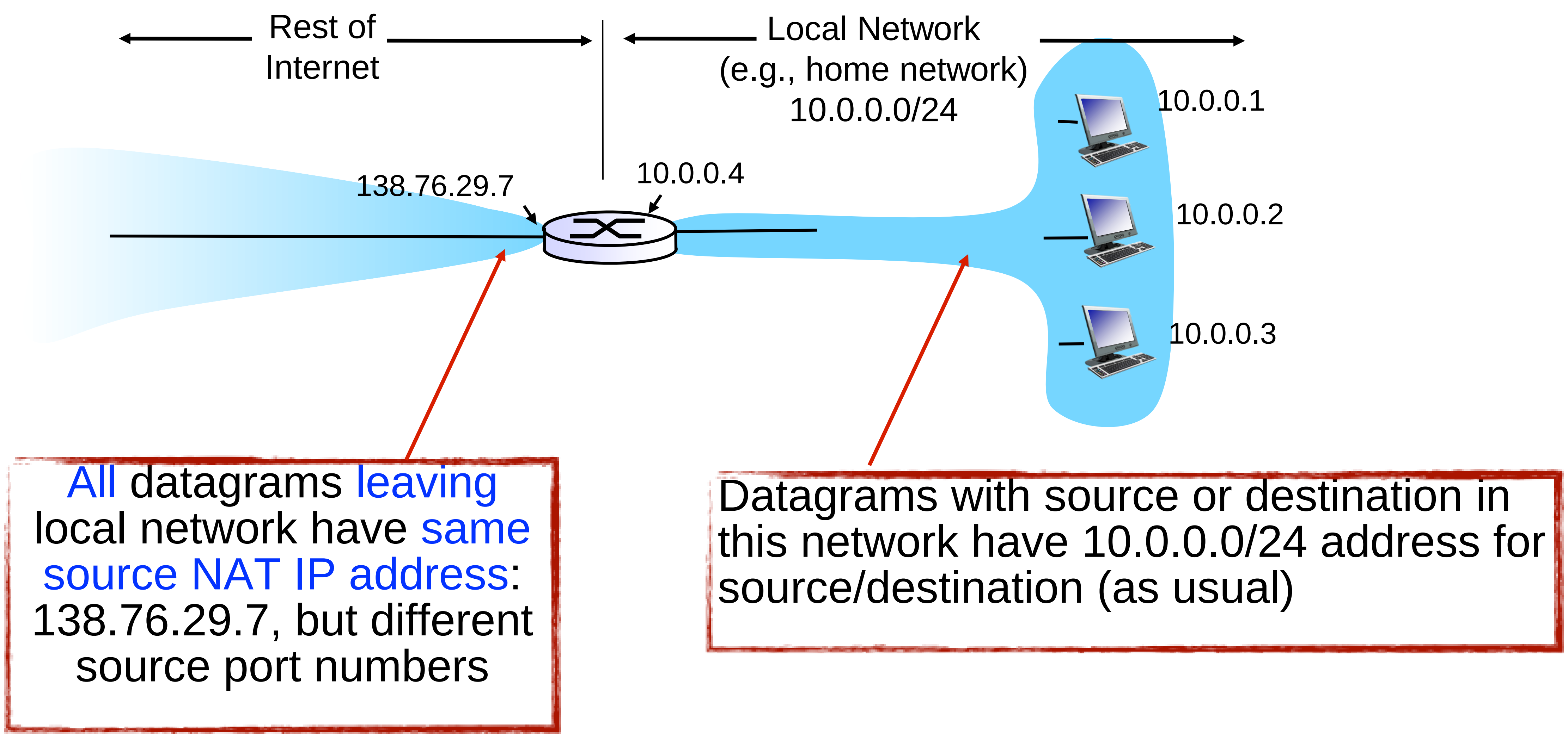
CS 330: Network Applications & Protocols

Network Layer

Galin Zhelezov
Department of Physical Sciences
York College of Pennsylvania

NAT: Network Address Translation



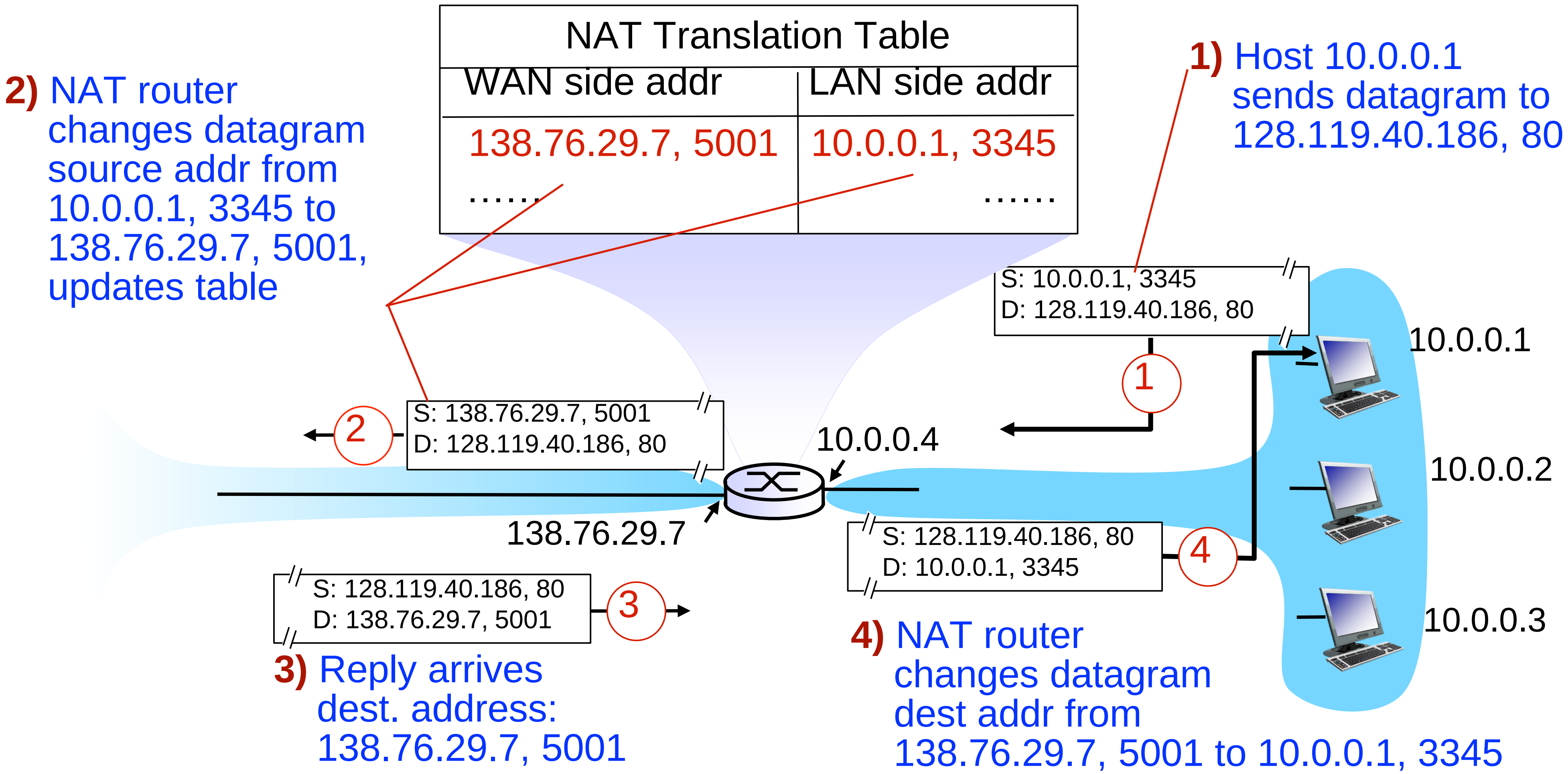
NAT: Network Address Translation

- **Local network uses just one IP address as far as outside world is concerned**
  - Don’t need a whole range of addresses from ISP, just one IP address is used for all devices on local network
  - Can change addresses of devices in local network without notifying outside world
  - Can change ISP without changing addresses of devices in local network
  - Devices inside local network are not explicitly addressable or visible by the outside world (a security plus)

NAT: Network Address Translation

- **Implementation of NAT in a router**
  - Outgoing Datagrams - replace **`(source IP address,port #)`** of every outgoing datagram to **`(NAT IP address,new port #)`**
    - Remote clients/servers will respond using **`(NAT IP address,new port #)`** as destination address
  - Router must remember/maintain a NAT Translation Table
    - Maps every **`(source IP address, port #)`** to **`(NAT IP address, new port #)`**
  - Incoming Datagrams - replace **`(NAT IP address,new port #)`** in destination fields of *every incoming* datagram with corresponding **`(source IP address,port #)`** stored in NAT table

NAT: Network Address Translation



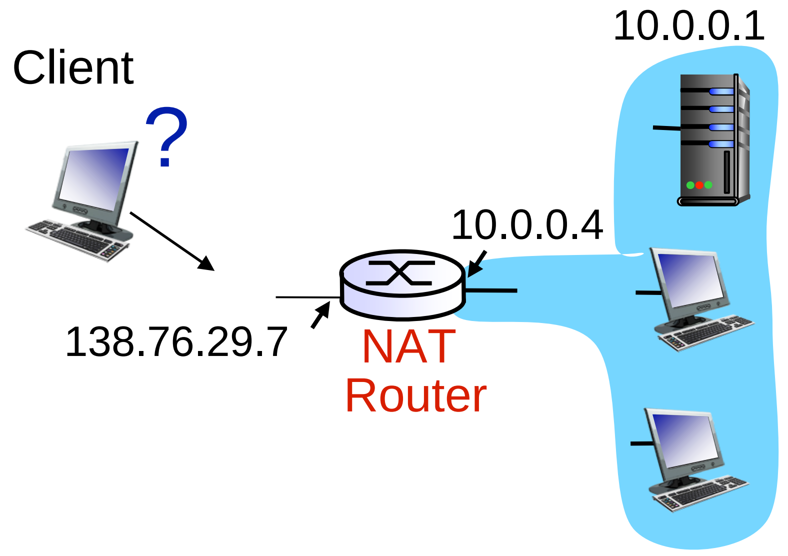
NAT: Network Address Translation

- **Transport Layer provides 16-bit port-number field**
  - With NAT, can support ~60,000 simultaneous connections with a single IP address!

- **NAT is controversial**
  - Routers should only process up to Layer 3 (i.e. they shouldn’t even be looking at port numbers)
  - Violates end-to-end argument, hosts should communicate directly, without other devices changing addresses and port numbers
    - NAT must be taken into account by application designers (e.g. P2P applications)
  - Address shortage should instead be solved by IPv6

NAT Traversal Problem

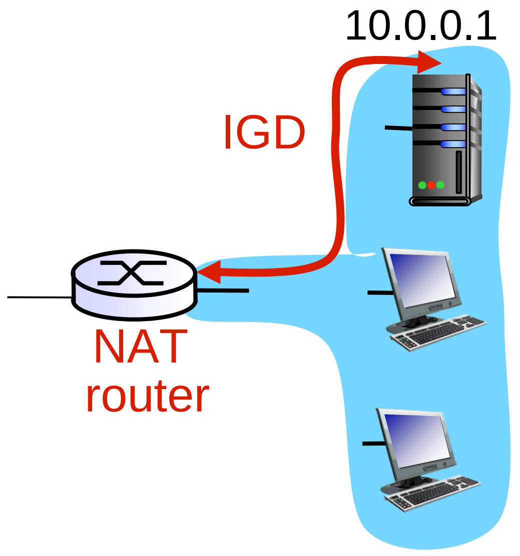
- **Client wants to connect to server with address 10.0.0.1**
  - Server address 10.0.0.1 is local to LAN only (client can't use it as a destination address)
  - Only one externally visible NATed address: 138.76.29.7



- **Solution #1 - statically configure NAT to forward incoming connection requests at given port to server**
  - e.g. Always forward **`(138.76.29.7, port 2500)`** to **`(10.0.0.1, port 25000)`**

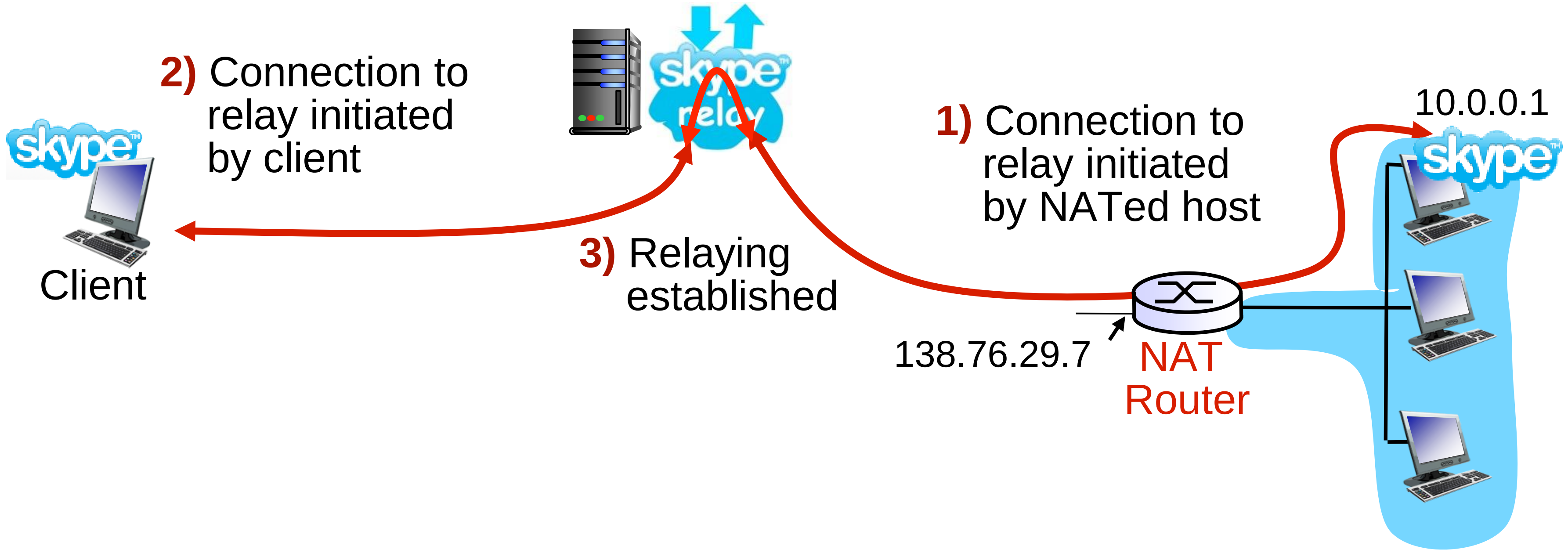
NAT Traversal Problem

- **Solution #2 - Universal Plug and Play (UPnP) Internet Gateway Device (IGD) protocol**
  - Allows NATed host to:
    - Learn public IP address (138.76.29.7)
    - Add/remove port mappings (with lease times)
    - Essentially automating the static NAT port map configuration from solution #1
      - This is done by the application that wants to advertise an open port to the world



NAT Traversal Problem

- **Solution #3 - relaying (used in Skype)**
  - NATed client establishes connection to relay
  - External client connects to relay
  - Relay bridges packets between to connections

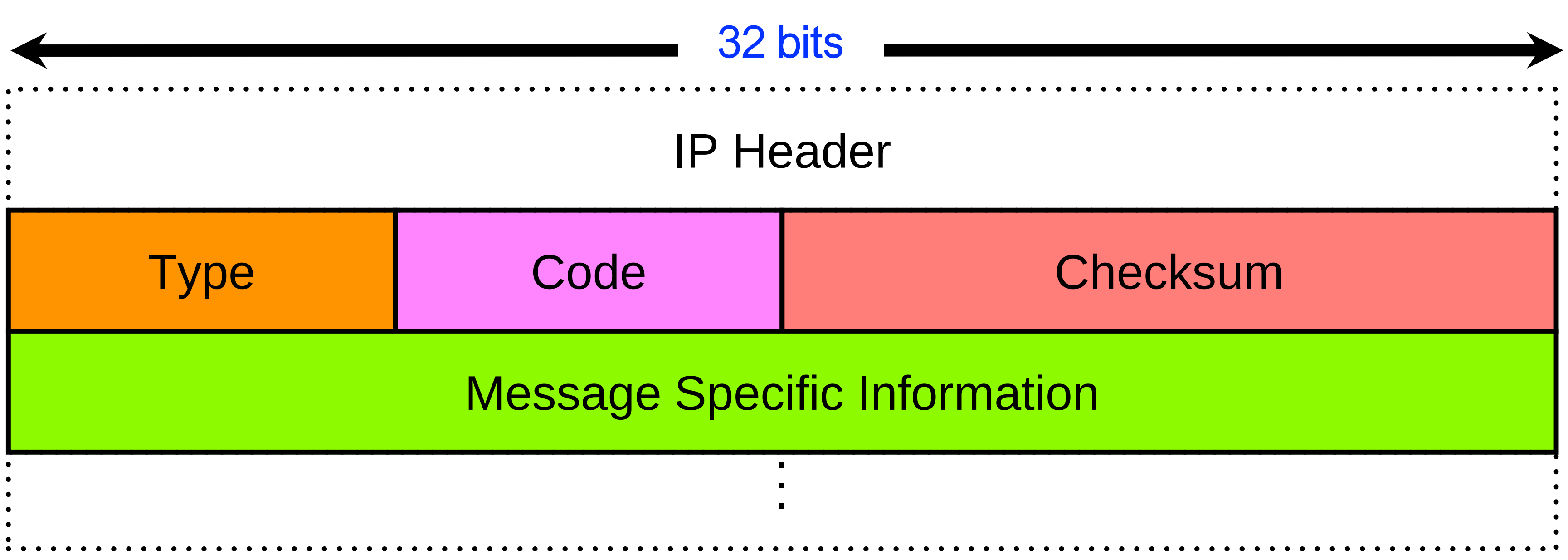


Overview of Network Layer

- **Virtual Circuit and Datagram Networks**
- **Router Architectures**
- **IP: Internet Protocol**
  - Datagram Format
  - IPv4 Addressing
  - ICMP
  - IPv6
- **Routing algorithms**
- **Routing in the Internet**
- **Broadcast and multicast routing**

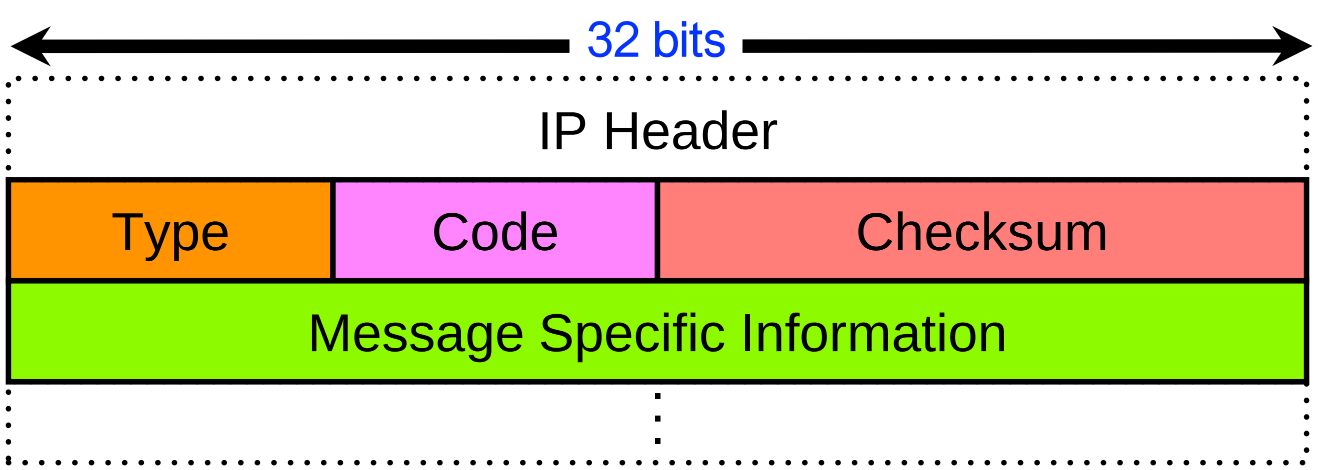
ICMP: Internet Control Message Protocol

- **Used by hosts and routers to communicate network-level information such as:**
  - Error reporting (e.g. destination host unreachable, TTL expired in transit)
  - Echo request/reply (used by ping)
- **ICMP messages are carried in IP datagrams**



ICMP Format

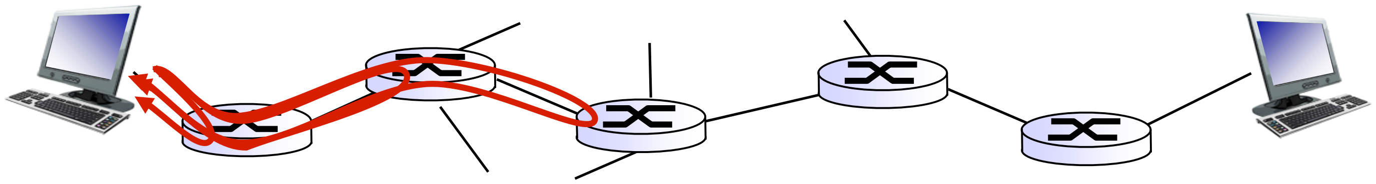
Type	Code	Description
0	0	Echo reply (used in ping)
3	0	Destination network unreachable
3	1	Destination host unreachable
3	2	Destination protocol unreachable
3	3	Destination port unreachable
3	4	Fragmentation required
3	6	Destination network unknown
3	7	Destination host unknown
8	0	Echo request (used in ping)
9	0	Router advertisement
10	0	Router discovery/selection/solicitation
11	0	TTL expired in transit



Traceroute and ICMP

- **Source sends series of UDP segments to destination on a port unlikely to be open**
  - First set has TTL=1
  - Second set has TTL=2, etc.
- **When n^{th} set of datagrams arrives at the n^{th} router**
  - Router discards datagrams because TTL = 0
  - Sends source host an ICMP messages (type 11, code 0)
  - ICMP messages include name and IP address of router
- **When ICMP messages arrive at source host, source host records RTTs**

- **Stopping criteria**
  - UDP segment eventually arrives at destination host
    - TTL was sufficiently large to get all the way to the destination
  - Destination returns ICMP "port unreachable" message (type 3, code 3)
  - Source stops sending UDP segments



Overview of Network Layer

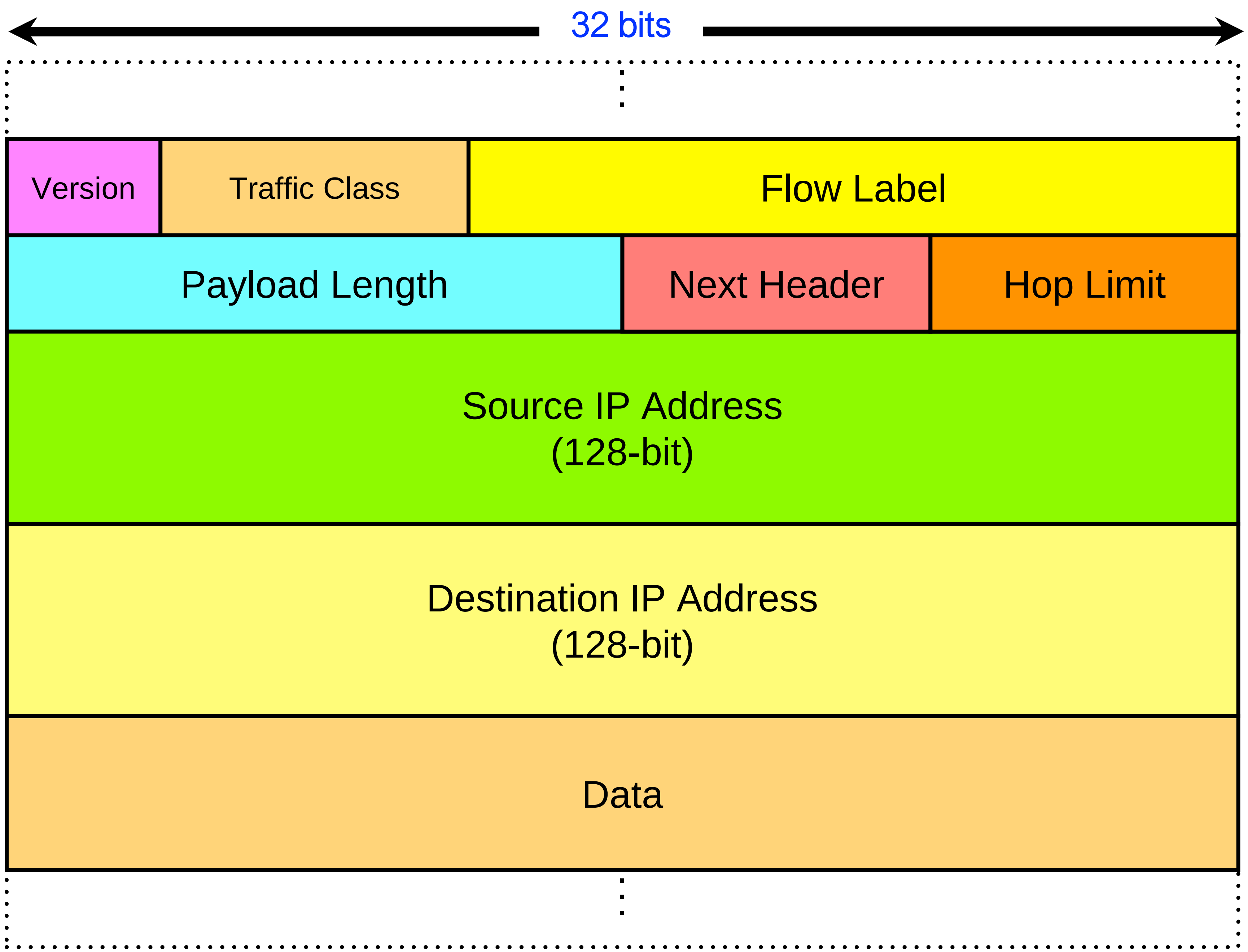
- **Virtual Circuit and Datagram Networks**
- **Router Architectures**
- **IP: Internet Protocol**
  - Datagram Format
  - IPv4 Addressing
  - ICMP
  - IPv6
- **Routing algorithms**
- **Routing in the Internet**
- **Broadcast and multicast routing**

IPv6 Motivation

- **32-bit address space of IPv4 soon to be completely allocated**
  - In some regions, all IPv4 addresses have already been allocated
  - 2^{128} addresses available in IPv6 (~ 3.4×10^{38})
- **New header format helps speed up processing/forwarding**
  - Fewer header fields
  - No checksum to validate/recompute
  - No Fragmentation allowed
    - Drop packets where fragmentation would be required
    - Send ICMPv6 message back to source with “Packet Too Big” error

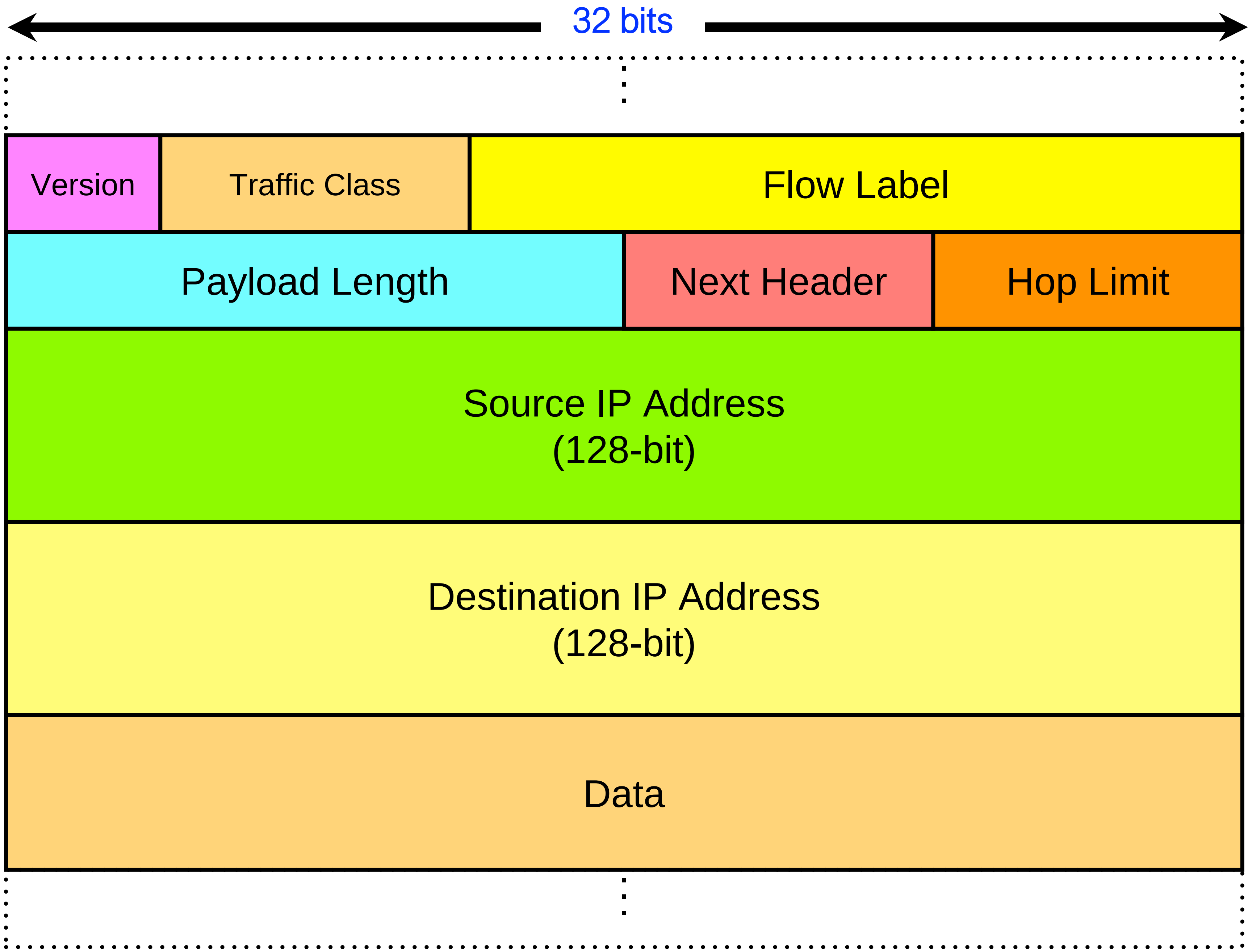
IPv6 Datagram Format

- **Version - 4-bit value that specifies the IP protocol version of the datagram (e.g. 0x6 for IPv6)**
- **Traffic Class - 8-bit value similar to the Type of Service from IPv4**
  - Used for Differentiated Services
  - Used for Explicit Congestion Notification
- **Flow Label - 20-bit value used to identify a flow of datagrams**
  - Can be used by routers to send datagrams in same flow along same path to prevent re-ordering of datagrams
- **Payload Length - 16-bit value that specifies the number of bytes of *data* in the datagram**
  - Does NOT include the 40-byte header



IPv6 Datagram Format

- **Next Header - 8-bit value that specifies the type of the next header**
  - Similar to the Protocol field from IPv4
  - Typically specifies the Transport Layer protocol within the IPv6 datagram
  - Values are the same as those from IPv4 (e.g. 6 indicates TCP, 17 indicates UDP)
- **Hop Limit - 8-bit value that limits the number of hops for this datagram**
  - Just like the TTL from IPv4
  - Decremented by 1 at each router, if value reaches 0 packet is dropped
- **Source IP Address - 128-bit IPv6 address of the machine sending the datagram**
- **Destination IP Address - 128-bit IPv6 address of the intended recipient**

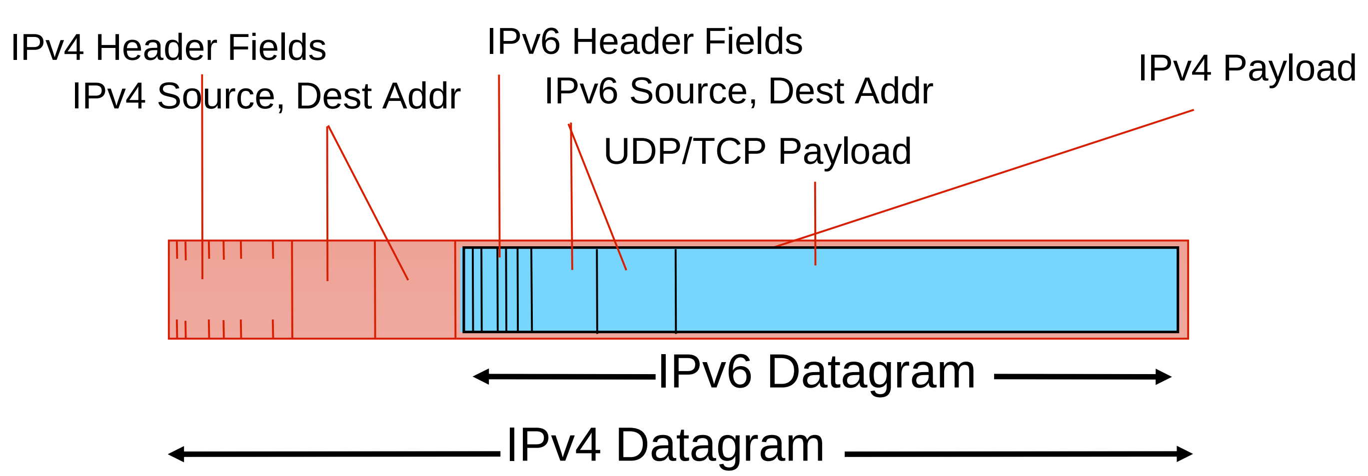


Notable Changes from IPv4

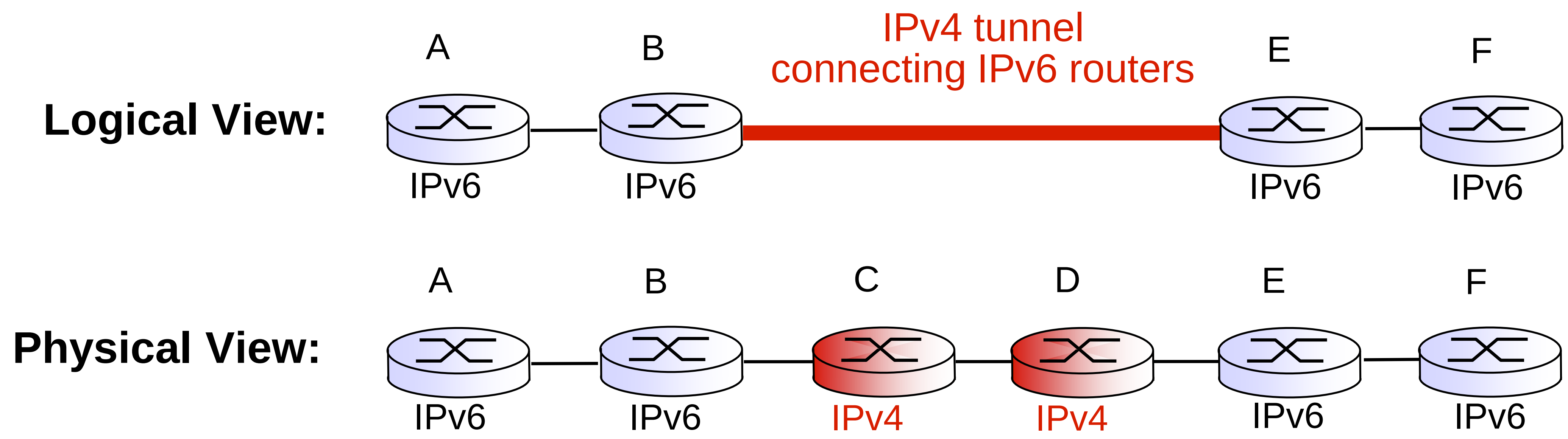
- **IPv6 header no longer contains a checksum**
  - Reduces processing time at each node
- **Options are still allowed, but are no longer part of the header**
  - Can indicate existence of options using the *Next Header* field
- **Utilizes ICMPv6**
  - Includes additional message types (e.g. “Packet Too Big”)
  - Supports multicast group management functions
    - Send datagrams to groups of hosts

Transition from IPv4 to IPv6

- **Option #1 - Upgrade ALL routers/hosts on the Internet simultaneously**
  - BAH ... impossible to coordinate a single-day transition from IPv4 to IPv6 where all routers are upgraded simultaneously
- **Option #2 - Implement a dual stack at each network node**
  - Each node supports both IPv6 and IPv4
  - IPv6 is used only if the source, the destination, and ALL routers along a path support IPv6
- **Option #3 - Tunneling**
  - The last IPv6 router along a path puts the *entire* IPv6 datagram into a new IPv4 datagram that is addressed to the next IPv6 router
    - IPv6 datagram carried as payload in IPv4 datagram among IPv4 routers

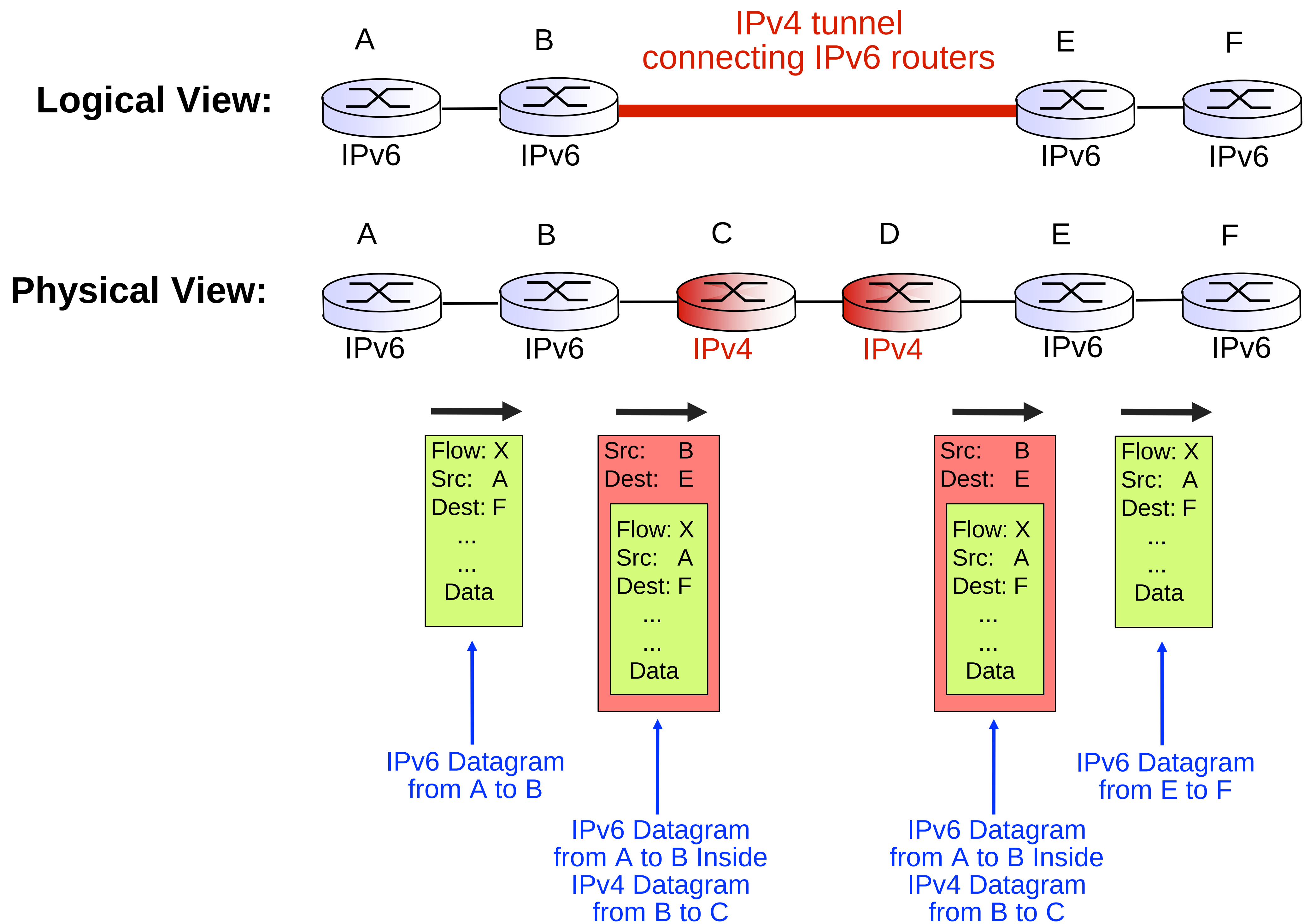


Tunneling



- **Sending IPv6 datagrams through a network that contains IPv4 routers**
  - Both source A and Destination F support IPv6
  - Need to be able to get IPv6 datagrams from nodes B to E
  - Tunnel IPv6 datagrams through IPv4 routers

Tunneling



Logical View:
A
B
IPv4 tunnel
connecting IPv6 routers
E
F
IPv6
IPv6
IPv6
IPv6
Physical View:
A
B
C
D
E
F
IPv6
IPv6
IPv4
IPv4
IPv6
IPv6
Flow: X
Src: A
Dest: F
...
...
Data
Src: B
Dest: E
Flow: X
Src: A
Dest: F
...
...
Data
Src: B
Dest: E
Flow: X
Src: A
Dest: F
...
...
Data
Flow: X
Src: A
Dest: F
...
...
Data
IPv6 Datagram
from A to B
IPv6 Datagram
from A to B Inside
IPv4 Datagram
from B to C
IPv6 Datagram
from A to B Inside
IPv4 Datagram
from B to C
IPv6 Datagram
from E to F

IPv6: adoption

- **Google: 8% of clients access services via IPv6**
- **NIST: 1/3 of all US government domains are IPv6 capable**

- ***Long (long!) time for deployment, use***

  -20 years and counting!

  -think of application-level changes in last 20 years: WWW, Facebook, streaming media, Skype, …

  -*Why?*