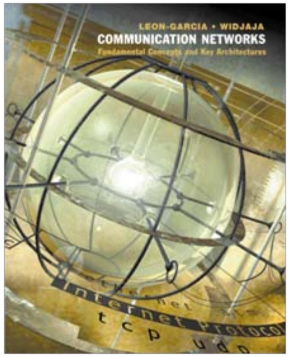
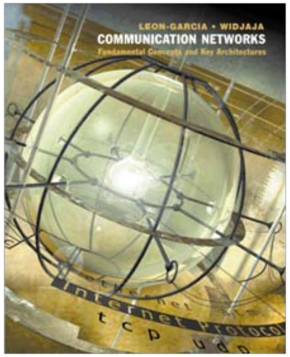
Chapter 8 Communication Networks and Services



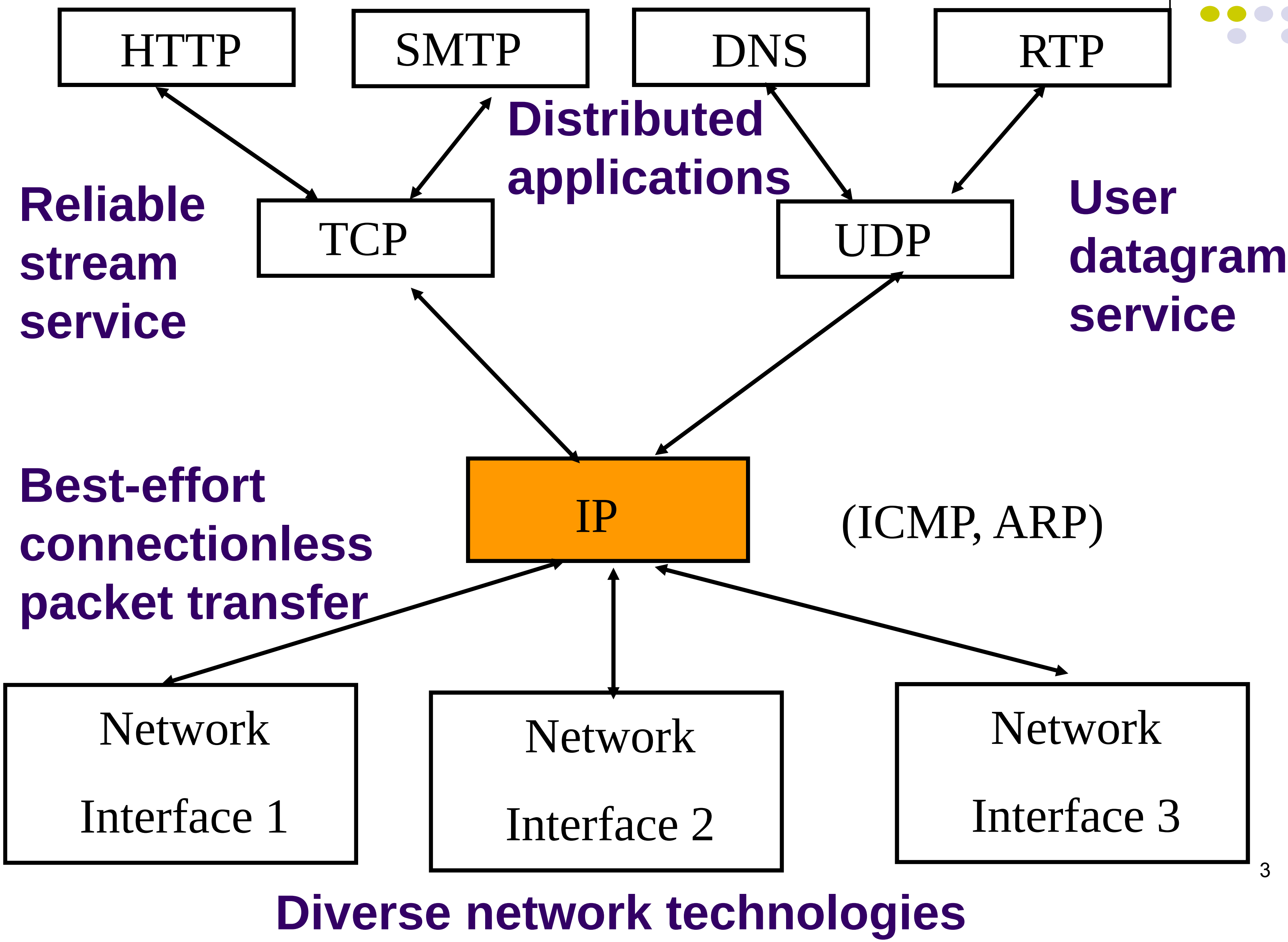
The TCP/IP Architecture
The Internet Protocol
IPv6
Transport Layer Protocols
Internet Routing Protocols
Multicast Routing
DHCP, NAT, and Mobile IP

Chapter 8
Communication Networks and Services



The TCP/IP Architecture

TCP/IP Protocol Suite



Internet Names & Addresses

Internet Names

- Each host has a unique name
  - Independent of physical location
  - Facilitate memorization by humans
  - Domain Name
  - Organization under single administrative unit
- Host Name
  - Name given to host computer
- User Name
  - Name assigned to user

leongarcia@comm.utoronto.ca

Internet Addresses

- Each host has globally unique *logical* 32 bit IP address
- Separate address for each physical connection to a network
- Routing decision is done based on destination IP address
- IP address has two parts:
  - *netid* and *hostid*
  - *netid* unique
  - *netid* facilitates routing
- Dotted Decimal Notation:

  int1.int2.int3.int4

  (intj = jth octet)

  128.100.10.13

DNS resolves IP name to IP address

Physical Addresses

- LANs (and other networks) assign physical addresses to the physical attachment to the network
- The network uses its own address to transfer packets or frames to the appropriate destination
- IP address needs to be resolved to physical address at each IP network interface
- Example: Ethernet uses 48-bit addresses
  - Each Ethernet network interface card (NIC) has globally unique Medium Access Control (MAC) or physical address
  - First 24 bits identify NIC manufacturer; second 24 bits are serial number
  - 00:90:27:96:68:07 12 hex numbers

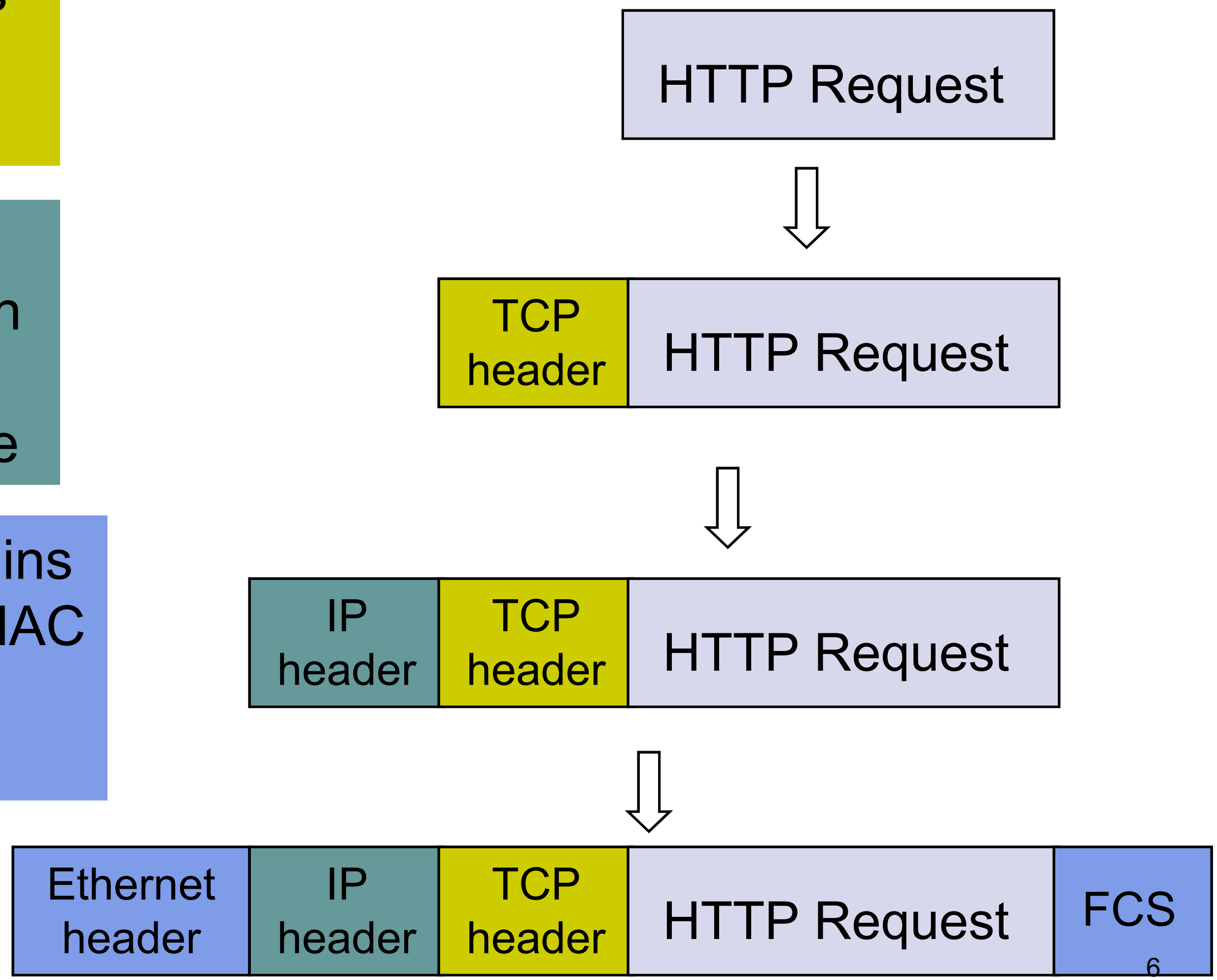
Intel (00:90:27)

Encapsulation

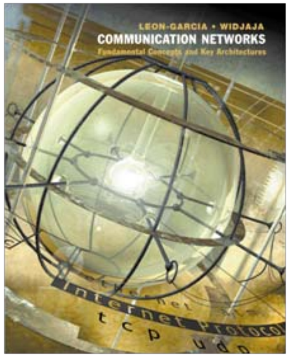
TCP Header contains source & destination port numbers

IP Header contains source and destination IP addresses; transport protocol type

Ethernet Header contains source & destination MAC addresses; network protocol type



Chapter 8
Communication Networks and Services

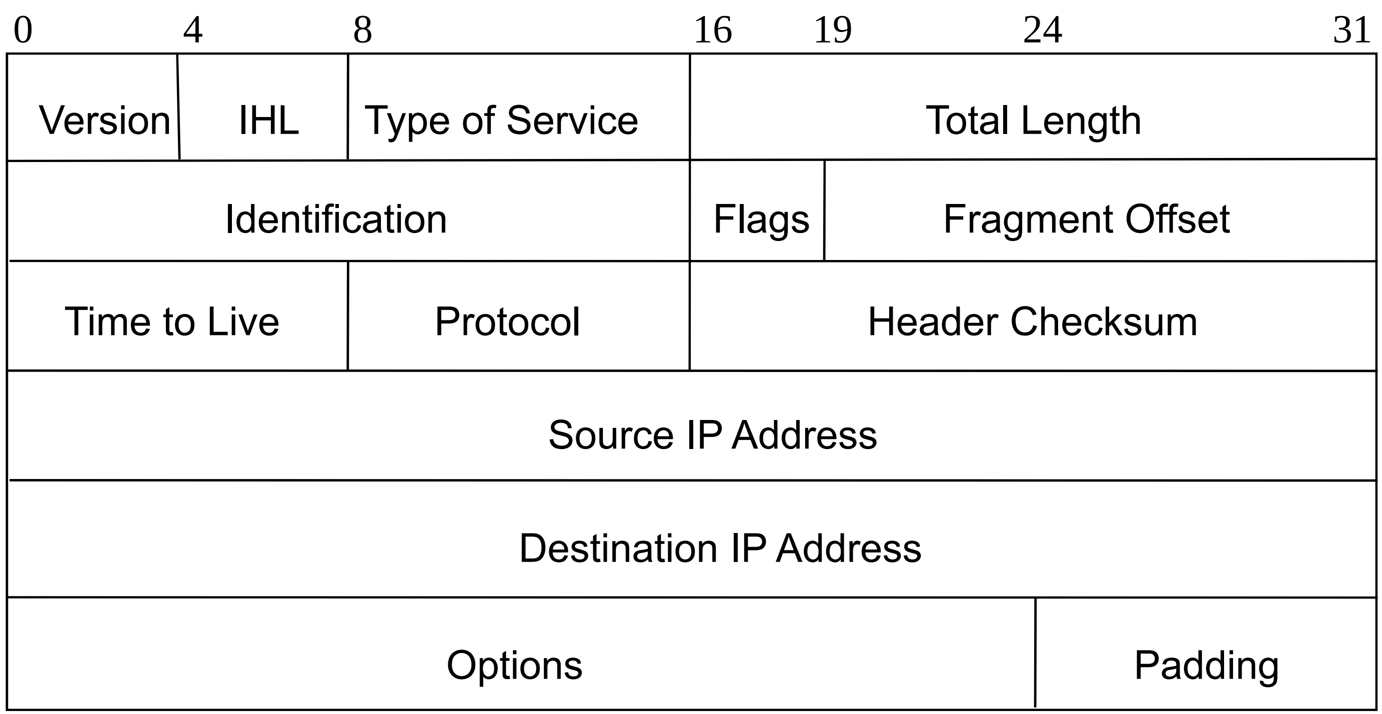


The Internet Protocol

Internet Protocol

- Provides best effort, connectionless packet delivery
  - motivated by need to keep routers simple and by adaptibility to failure of network elements
  - packets may be lost, out of order, or even duplicated
  - higher layer protocols must deal with these, if necessary
- RFCs 791, 950, 919, 922, and 2474.
- IP is part of Internet STD number 5, which also includes:
  - Internet Control Message Protocol (ICMP), RFC 792
  - Internet Group Management Protocol (IGMP), RFC 1112

IP Packet Header



- Minimum 20 bytes
- Up to 40 bytes in options fields

IP Packet Header

0	4	8	16	19	24	31
Version	IHL	Type of Service	Total Length			
Identification			Flags	Fragment Offset		
Time to Live		Protocol	Header Checksum			
Source IP Address						
Destination IP Address						
Options					Padding	

Version: current IP version is 4.

Internet header length (IHL): length of the header in 32-bit words.

Type of service (TOS): traditionally priority of packet at each router. Recent Differentiated Services redefines TOS field to include other services besides best effort.

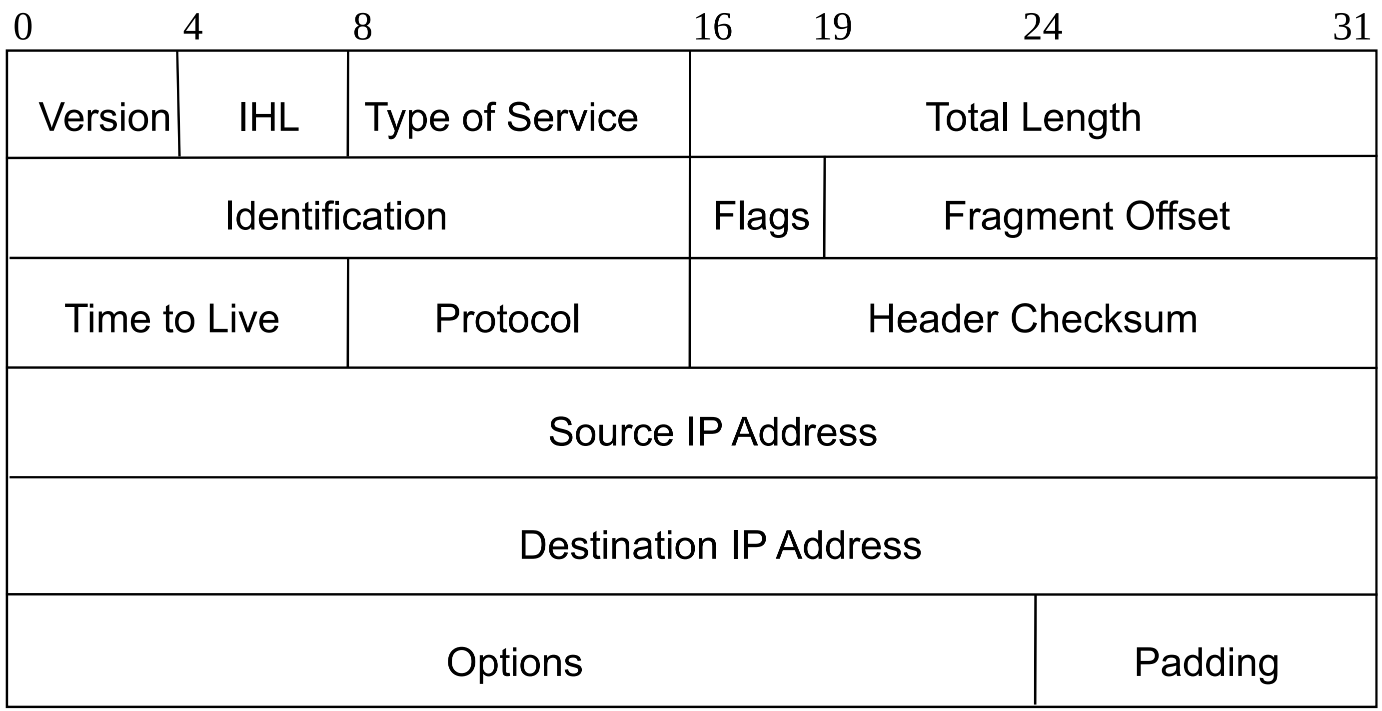
IP Packet Header

0	4	8	16	19	24	31
Version	IHL	Type of Service	Total Length			
Identification			Flags	Fragment Offset		
Time to Live		Protocol	Header Checksum			
Source IP Address						
Destination IP Address						
Options					Padding	

Total length: number of bytes of the IP packet including header and data, maximum length is 65535 bytes.

Identification, Flags, and Fragment Offset: used for fragmentation and reassembly (More on this shortly).

IP Packet Header



Time to live (TTL): number of hops packet is allowed to traverse in the network.

- Each router along the path to the destination decrements this value by one.
- If the value reaches zero before the packet reaches the destination, the router discards the packet and sends an error message back to the source.

IP Packet Header

0	4	8	16	19	24	31
Version	IHL	Type of Service	Total Length			
Identification			Flags	Fragment Offset		
Time to Live		Protocol	Header Checksum			
Source IP Address						
Destination IP Address						
Options					Padding	

Protocol: specifies upper-layer protocol that is to receive IP data at the destination. Examples include TCP (protocol = 6), UDP (protocol = 17), and ICMP (protocol = 1).

Header checksum: verifies the integrity of the IP header.

Source IP address and **destination IP address:** contain the addresses of the source and destination hosts.

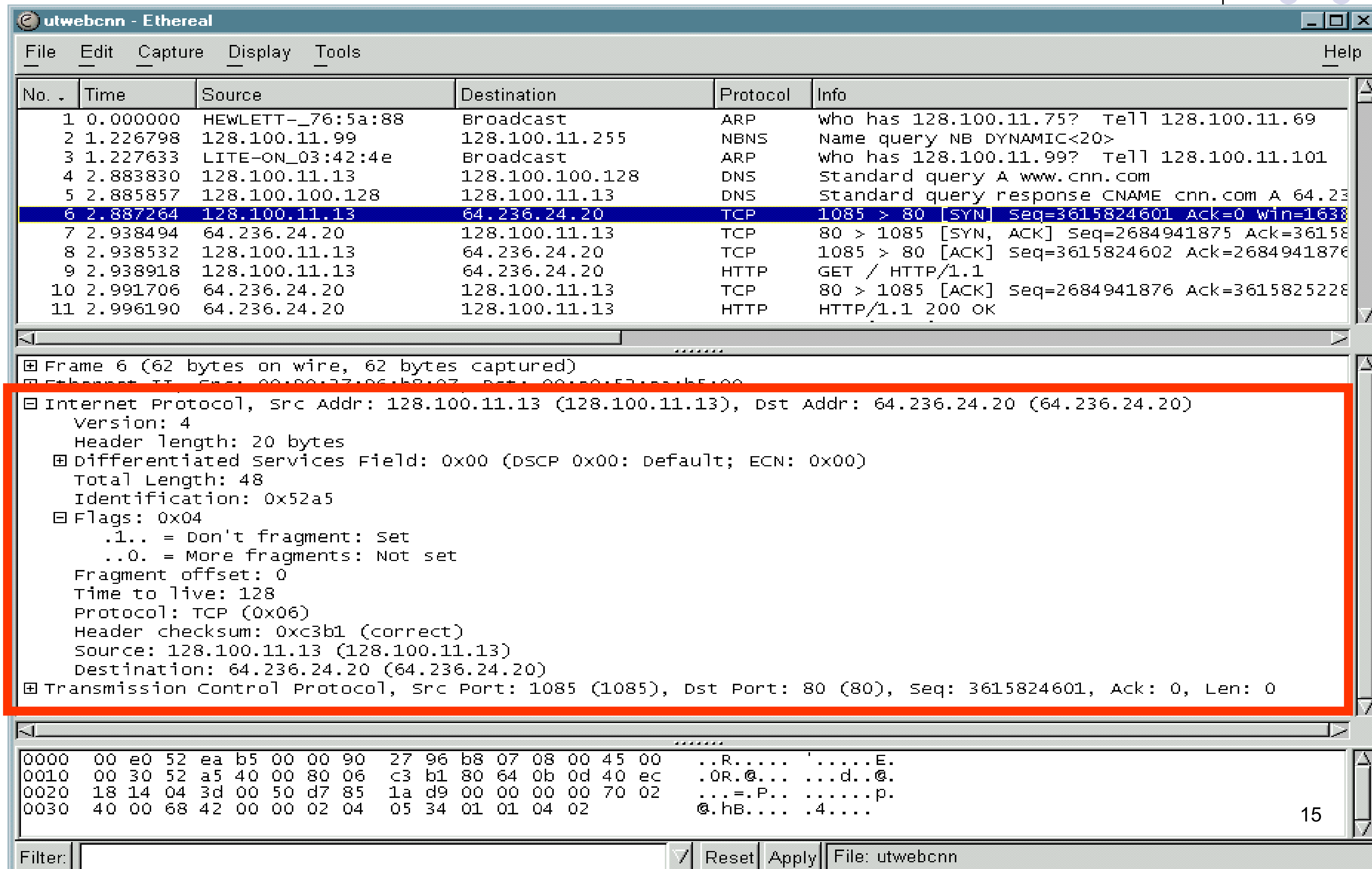
IP Packet Header

<table>
<tr><th>0</th><th>4</th><th>8</th><th>16</th><th>19</th><th>24 … 31</th></tr>
<tr><td>Version</td><td>IHL</td><td>Type of Service</td><td colspan="3">Total Length</td></tr>
<tr><td colspan="3">Identification</td><td>Flags</td><td colspan="2">Fragment Offset</td></tr>
<tr><td colspan="2">Time to Live</td><td>Protocol</td><td colspan="3">Header Checksum</td></tr>
<tr><td colspan="6">Source IP Address</td></tr>
<tr><td colspan="6">Destination IP Address</td></tr>
<tr><td colspan="5">Options</td><td>Padding</td></tr>
</table>

Options: Variable length field, allows packet to request special features such as security level, route to be taken by the packet, and timestamp at each router. Detailed descriptions of these options can be found in [RFC 791].

Padding: This field is used to make the header a multiple of 32-bit words.

Example of IP Header



Header Checksum

- IP header uses check bits to detect errors in the ***header***
- A checksum is calculated for header contents
- Checksum recalculated at every router, so algorithm selected for ease of implementation in software
- Let header consist of L, 16-bit words,

  $\mathbf{b}_0, \mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_{L-1}$
- The algorithm appends a 16-bit *checksum* $\mathbf{b}_L$

Checksum Calculation

The checksum $\mathbf{b}_L$ is calculated as follows:

- Treating each 16-bit word as an integer, find

  $$\mathbf{x} = \mathbf{b}_0 + \mathbf{b}_1 + \mathbf{b}_2 + \ldots + \mathbf{b}_{L-1} \text{ modulo } 2^{15}\text{-}1$$

- The checksum is then given by:

  $$\mathbf{b}_L = -\mathbf{x} \quad \text{modulo } 2^{15}\text{-}1$$

- This is the 16-bit 1's complement sum of the **b**'s
- If checksum is 0, use all 1's representation (all zeros reserved to indicate checksum was not calculated)
- *Thus, the headers must satisfy the following* ***pattern***:

  $$\mathbf{0} = \mathbf{b}_0 + \mathbf{b}_1 + \mathbf{b}_2 + \ldots + \mathbf{b}_{L-1} + \mathbf{b}_L \text{ modulo } 2^{15}\text{-}1$$

IP Header Processing

1. Compute header checksum for correctness and check that fields in header (e.g. version and total length) contain valid values
2. Consult routing table to determine next hop
3. Change fields that require updating (TTL, header checksum)

IP Addressing

- RFC 1166
- Each host on Internet has unique 32 bit IP address
- Each address has two parts: *netid* and *hostid*
- *netid* unique & administered by
  - American Registry for Internet Numbers (ARIN)
  - Reseaux IP Europeens (RIPE)
  - Asia Pacific Network Information Centre (APNIC)
- Facilitates routing
- A separate address is required for each physical connection of a host to a network; "multi-homed" hosts
- Dotted-Decimal Notation:

  int1.int2.int3.int4 where intj = integer value of jth octet

  IP address of 10000000 10000111 01000100 00000101

  is 128.135.68.5 in dotted-decimal notation

Classful Addresses

Class A

	7 bits	24 bits
0	netid	hostid

- 126 networks with up to 16 million hosts

1.0.0.0 to 127.255.255.255

Class B

		14 bits	16 bits
1	0	netid	hostid

- 16,382 networks with up to 64,000 hosts

128.0.0.0 to 191.255.255.255

Class C

			22 bits	8 bits
1	1	0	netid	hostid

- 2 million networks with up to 254 hosts

192.0.0.0 to 223.255.255.255

Class D

28 bits

1	1	1	0	multicast address

224.0.0.0 to 239.255.255.255

- Up to 250 million multicast groups at the same time
- Permanent group addresses
  - All systems in LAN; All routers in LAN;
  - All OSPF routers on LAN; All designated OSPF routers on a LAN, etc.
- Temporary groups addresses created as needed
- Special multicast routers

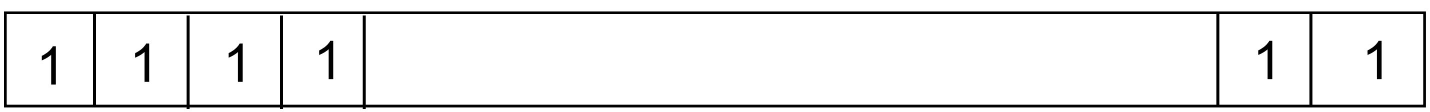
Reserved Host IDs (all 0s & 1s)

Internet address used to refer to network has hostid set to all 0s

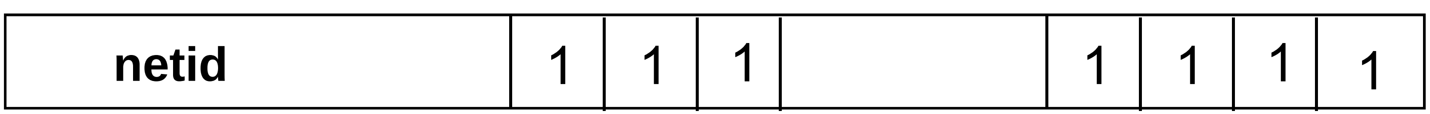
this host (used when booting up)

a host in this network

Broadcast address has hostid set to all 1s



broadcast on local network

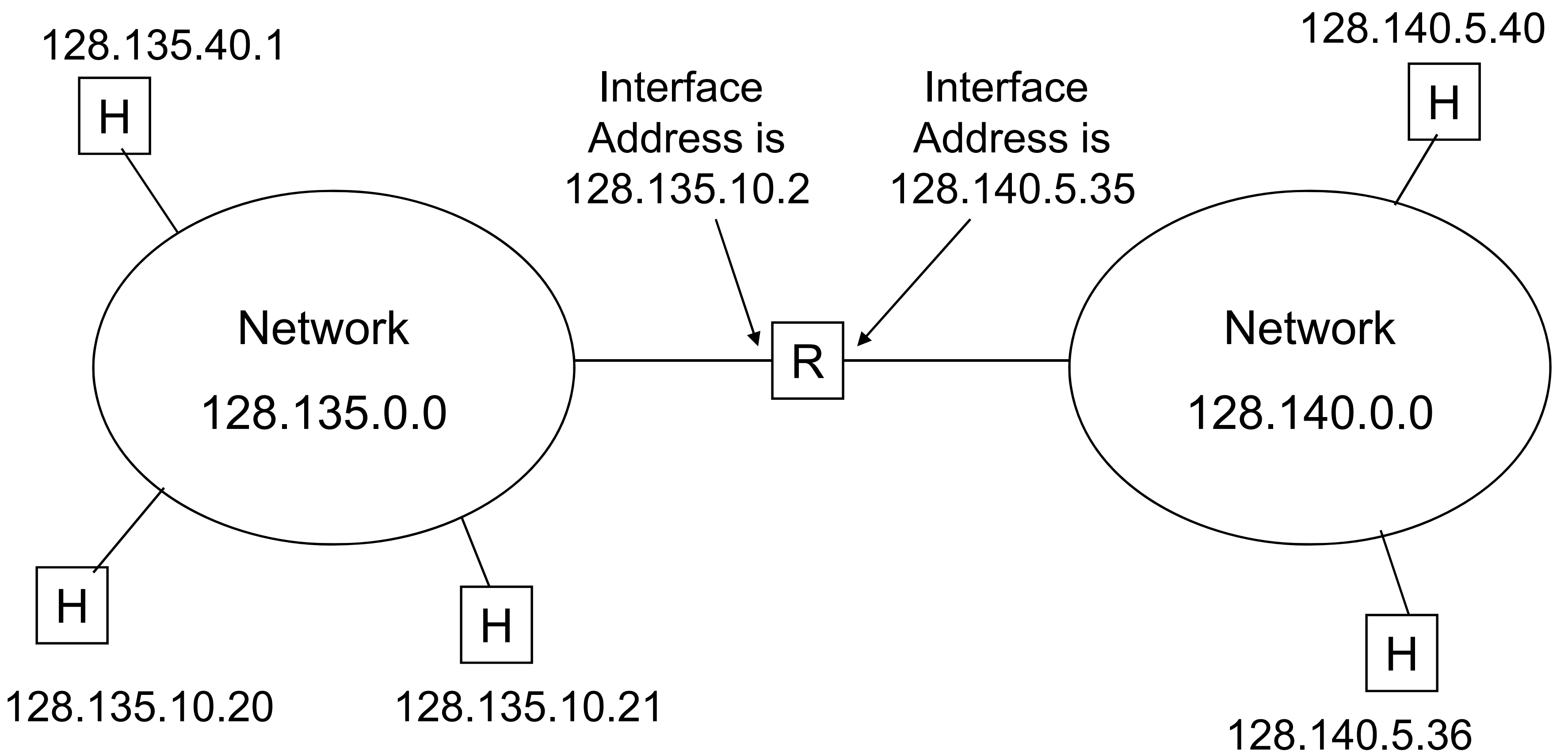


broadcast on distant network

Private IP Addresses

- Specific ranges of IP addresses set aside for use in private networks (RFC 1918)
- Use restricted to private internets; routers in public Internet discard packets with these addresses
- Range 1: 10.0.0.0 to 10.255.255.255
- Range 2: 172.16.0.0 to 172.31.255.255
- Range 3: 192.168.0.0 to 192.168.255.255
- Network Address Translation (NAT) used to convert between private & global IP addresses

Example of IP Addressing



Address with host ID=all 0s refers to the network

Address with host ID=all 1s refers to a broadcast packet

R = router

H = host

Subnet Addressing

- Subnet addressing introduces another hierarchical level
- Transparent to remote networks
- Simplifies management of multiplicity of LANs
- Masking used to find subnet number

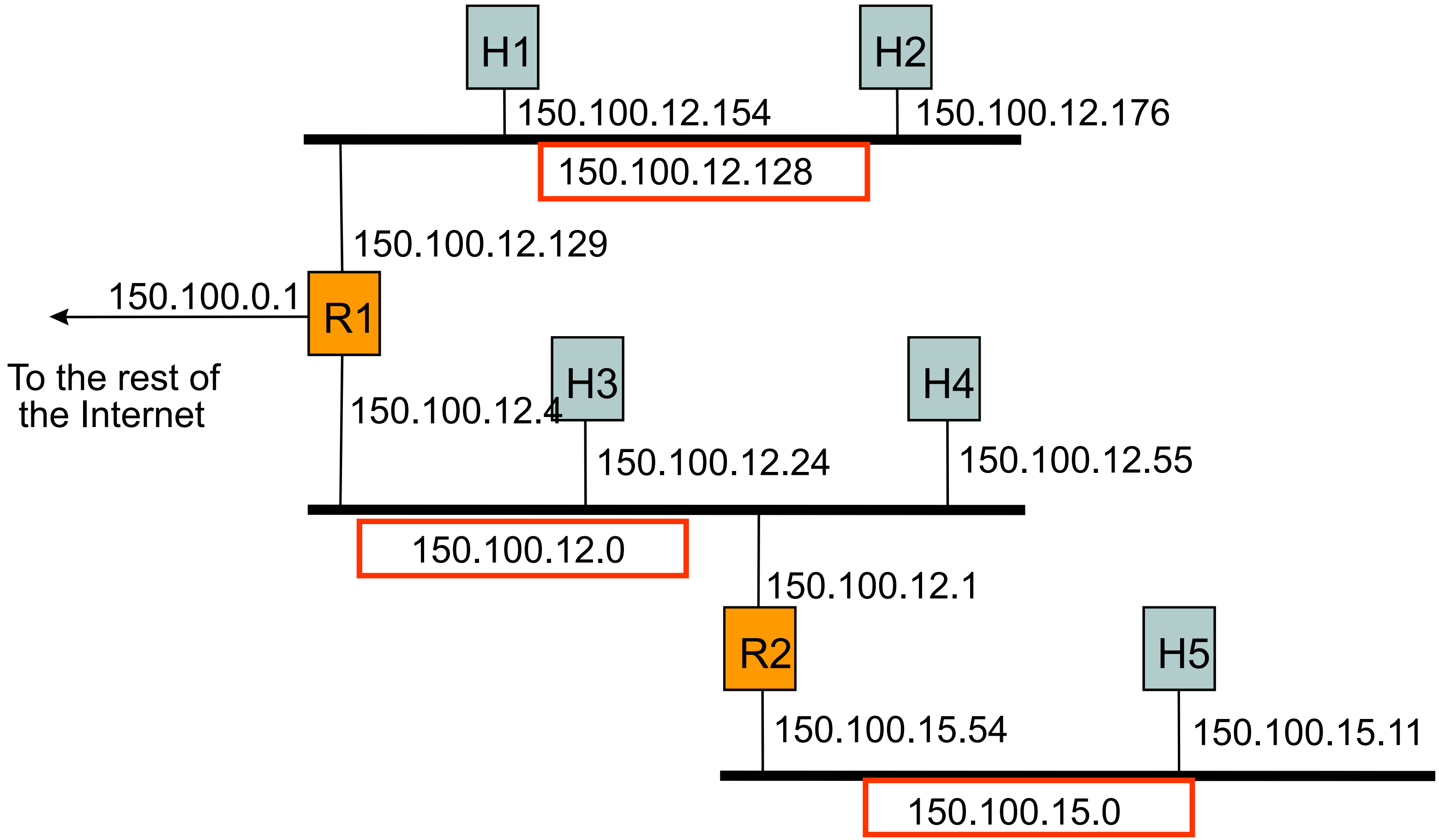
Original address	1	0	Net ID	Host ID

Subnetted address	1	0	Net ID	Subnet ID	Host ID

Subnetting Example

- Organization has Class B address (16 host ID bits) with network ID: 150.100.0.0
- Create subnets with up to 100 hosts each
  - 7 bits sufficient for each subnet
  - 16-7=9 bits for subnet ID
- Apply subnet mask to IP addresses to find corresponding subnet
  - Example: Find subnet for 150.100.12.176
  - IP add = 10010110 01100100 00001100 10110000
  - Mask = 11111111 11111111 11111111 10000000
  - AND = 10010110 01100100 00001100 10000000
  - Subnet = 150.100.12.128
  - Subnet address used by routers within organization

Subnet Example



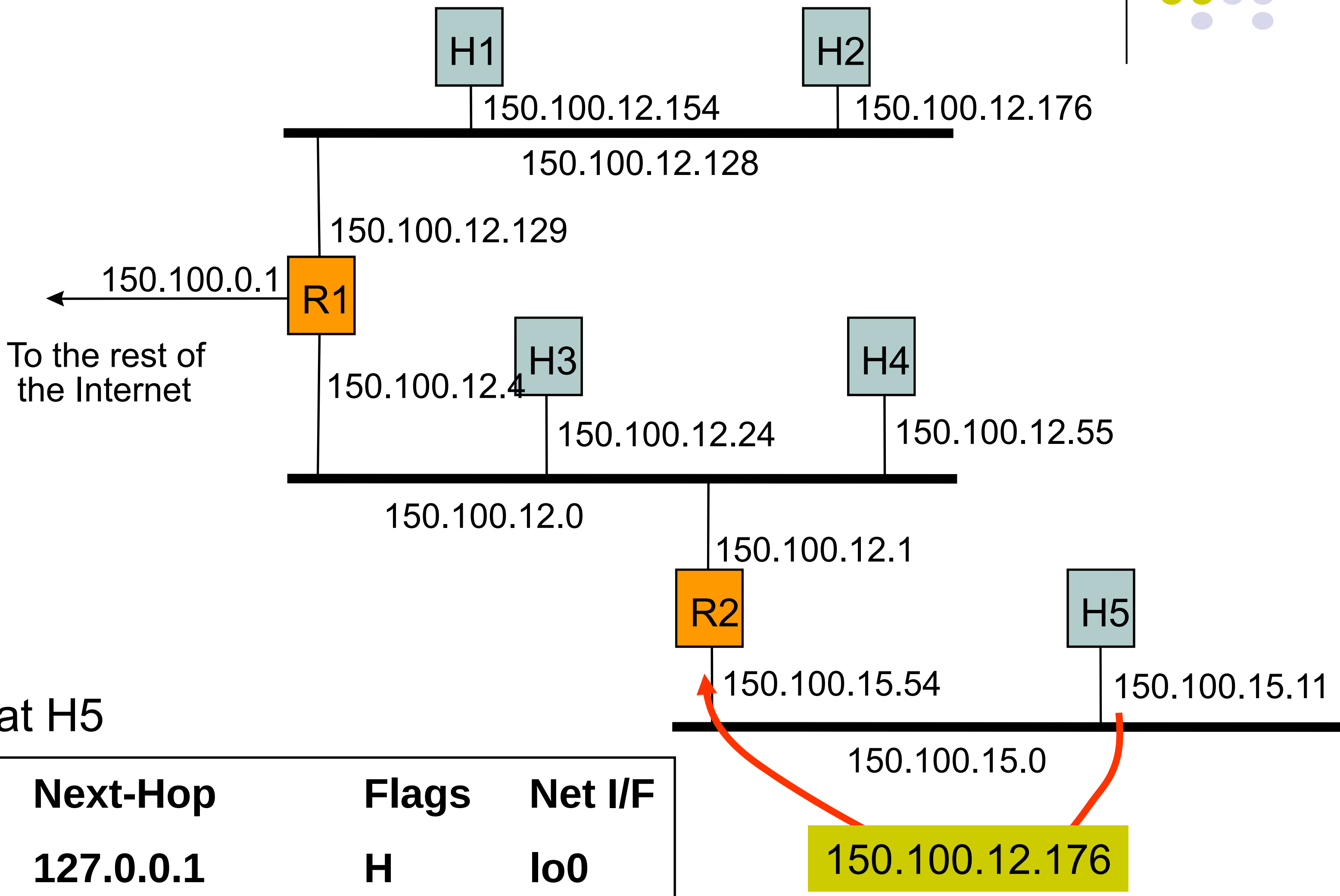
Routing with Subnetworks

- IP layer in hosts and routers maintain a routing table
- Originating host: To send an IP packet, consult routing table
  - If destination host is in same network, send packet *directly* using appropriate network interface
  - Otherwise, send packet indirectly; typically, routing table indicates a default router
- Router: Examine IP destination address in arriving packet
  - If dest IP address not own, router consults routing table to determine next-hop and associated network interface & forwards packet

Routing Table

- Each row in routing table contains:
  - Destination IP address
  - IP address of next-hop router
  - Physical address
  - Statistics information
  - Flags
    - H=1 (0) indicates route is to a host (network)
    - G=1 (0) indicates route is to a router (directly connected destination)
- Routing table search order & action
  - Complete destination address; send as per next-hop & G flag
  - Destination network ID; send as per next-hop & G flag
  - Default router entry; send as per next-hop
  - Declare packet undeliverable; send ICMP "host unreachable error" packet to originating host

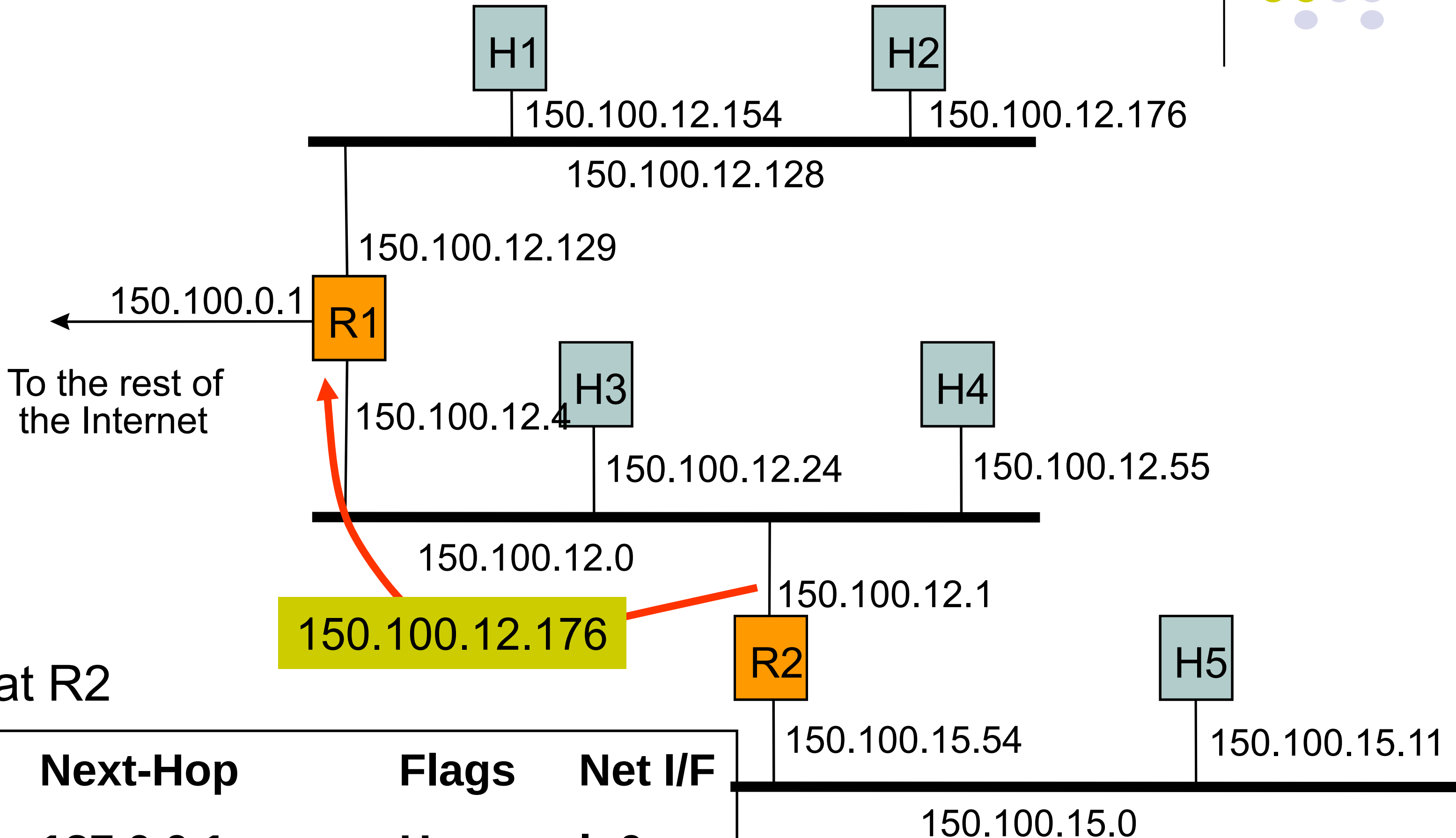
Example: Host H5 sends packet to host H2



Routing Table at H5

Destination	Next-Hop	Flags	Net I/F
127.0.0.1	127.0.0.1	H	lo0
default	150.100.15.54	G	emd0
150.100.15.0	150.100.15.11		emd0

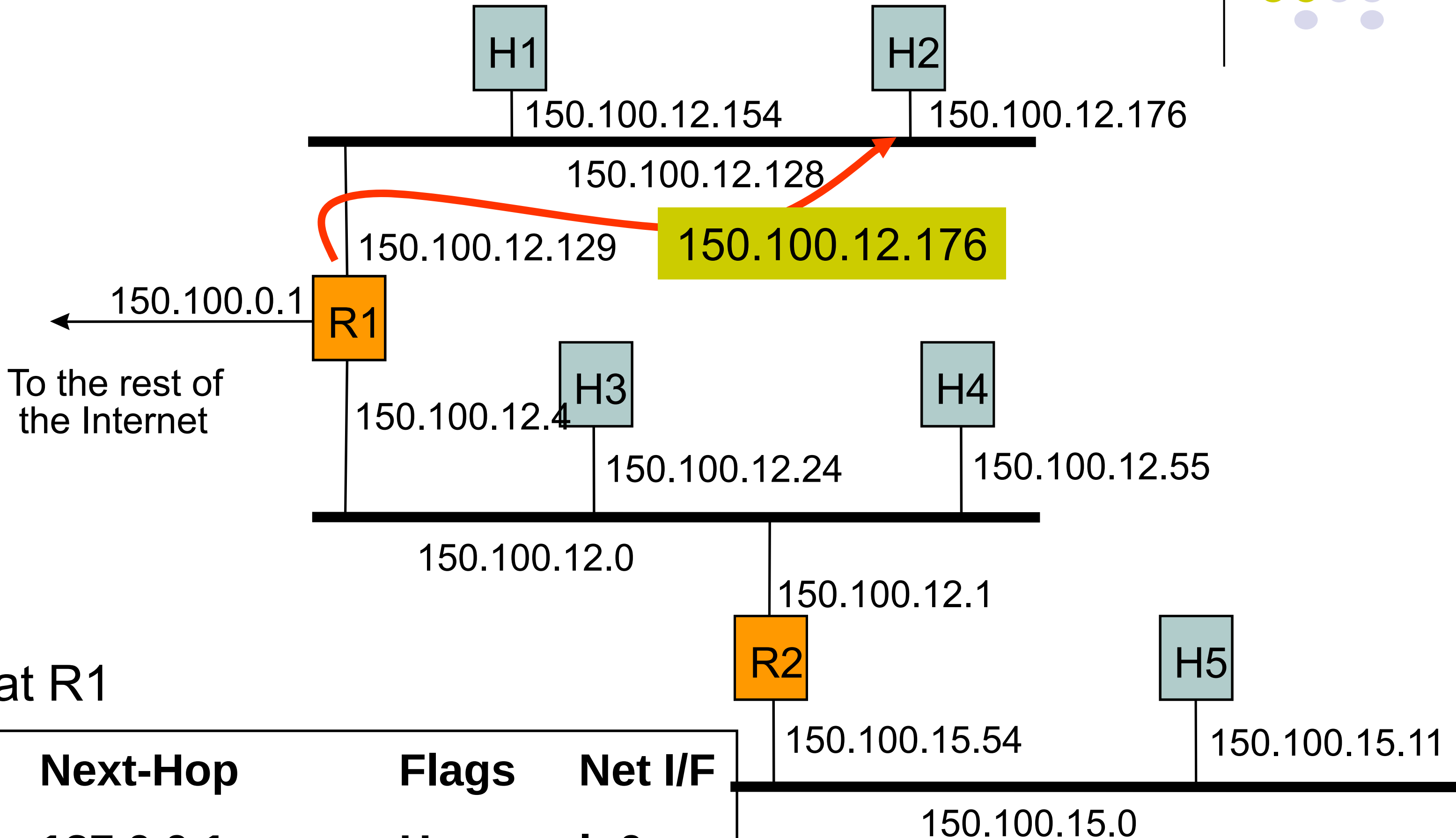
Example: Host H5 sends packet to host H2



Routing Table at R2

Destination	Next-Hop	Flags	Net I/F
127.0.0.1	127.0.0.1	H	lo0
default	150.100.12.4	G	emd0
150.100.15.0	150.100.15.54		emd1
150.100.12.0	150.100.12.1		emd0

Example: Host H5 sends packet to host H2



Routing Table at R1

Destination	Next-Hop	Flags	Net I/F
127.0.0.1	**127.0.0.1**	**H**	**lo0**
150.100.12.176	**150.100.12.176**		**emd0**
150.100.12.0	**150.100.12.4**		**emd1**
150.100.15.0	**150.100.12.1**	**G**	**emd1**

IP Address Problems

- In the 1990, two problems became apparent
  - IP addresses were being exhausted
  - IP routing tables were growing very large
- IP Address Exhaustion
  - Class A, B, and C address structure inefficient
    - Class B too large for most organizations, but future proof
    - Class C too small
    - Rate of class B allocation implied exhaustion by 1994
- IP routing table size
  - Growth in number of networks in Internet reflected in # of table entries
    - From 1991 to 1995, routing tables doubled in size every 10 months
    - Stress on router processing power and memory allocation
- Short-term solution:
- Classless Interdomain Routing (CIDR), RFC 1518
- New allocation policy (RFC 2050)
- Private IP Addresses set aside for intranets
- Long-term solution: IPv6 with much bigger address space

Classless Inter-Domain Routing

- CIDR deals with Routing Table Explosion Problem
  - Networks represented by prefix and mask
  - Pre-CIDR: Network with range of 16 contiguous class C blocks requires 16 entries
  - Post-CIDR: Network with range of 16 contiguous class C blocks requires 1 entry
- Solution: *Route according to prefix of address*, not class
  - Routing table entry has <IP address, network mask>
  - Example: 192.32.136.0/21
  - `11000000 00100000 10001000 00000001` min address
  - `11111111 11111111 11111--- --------` mask
  - `11000000 00100000 10001--- --------` IP prefix
  - `11000000 00100000 10001111 11111110` max address
  - `11111111 11111111 11111--- --------` mask
  - `11000000 00100000 10001--- --------` same IP prefix

CIDR Allocation Principles
(RFC 1518-1520)

- IP address assignment reflects physical topology of network
- Network topology follows continental/national boundaries
  - IP addresses should be assigned on this basis
- Transit routing domains (TRDs) have unique IP prefix
  - carry traffic between routing domains
  - interconnected non-hierarchically, cross national boundaries
  - Most routing domains single-homed: attached to a single TRD
  - Such domains assigned addresses with TRD's IP prefix
  - All of the addresses attached to a TRD aggregated into 1table entry
- Implementation primarily through BGPv4 (RFC 1520)

Longest Prefix Match

- CIDR impacts routing & forwarding
- Routing tables and routing protocols must carry IP address and mask
- Multiple entries may match a given IP destination address
- Example: Routing table may contain
  - 205.100.0.0/22 which corresponds to a given supernet
  - 205.100.0.0/20 which results from aggregation of a larger number of destinations into a supernet
  - Packet must be routed using the *more specific route*, that is, the longest prefix match
- Several fast longest-prefix matching algorithms are available

Address Resolution Protocol

Although IP address identifies a host, the packet is physically delivered by an underlying network (e.g., Ethernet) which uses its own *physical address* (MAC address in Ethernet). How to map an IP address to a physical address?

H1 wants to learn physical address of H3 -> broadcasts an ARP request

ARP request (what is the MAC address of 150.100.76.22?)

Every host receives the request, but only H3 reply with its physical address

ARP response (my MAC address is 08:00:5a:3b:94)

Example of ARP

<capture> - Ethereal

File Edit Capture Display Tools Help

No.	Time	Source	Destination	Protocol	Info
1	0.000000	3COM_1d:cc:f7	Broadcast	ARP	Who has 192.168.2.1? Tell 192.168.2.18
2	0.000675	SMC_29:b2:3a	3COM_1d:cc:f7	ARP	192.168.2.1 is at 00:04:e2:29:b2:3a
3	0.000714	192.168.2.18	192.168.2.1	DNS	Standard query A nal.utoronto.ca
4	0.038154	192.168.2.1	192.168.2.18	DNS	Standard query response A 128.100.244.3
5	0.039904	192.168.2.18	128.100.244.3	ICMP	Echo (ping) request
6	0.040875	192.168.2.1	192.168.2.18	ICMP	Time-to-live exceeded
7	0.041482	192.168.2.18	128.100.244.3	ICMP	Echo (ping) request
8	0.042227	192.168.2.1	192.168.2.18	ICMP	Time-to-live exceeded
9	0.043292	192.168.2.18	128.100.244.3	ICMP	Echo (ping) request
10	0.044664	192.168.2.1	192.168.2.18	ICMP	Time-to-live exceeded
11	0.355785	192.168.2.18	192.168.2.1	DNS	Standard query PTR 1.2.168.192.in-addr.arpa

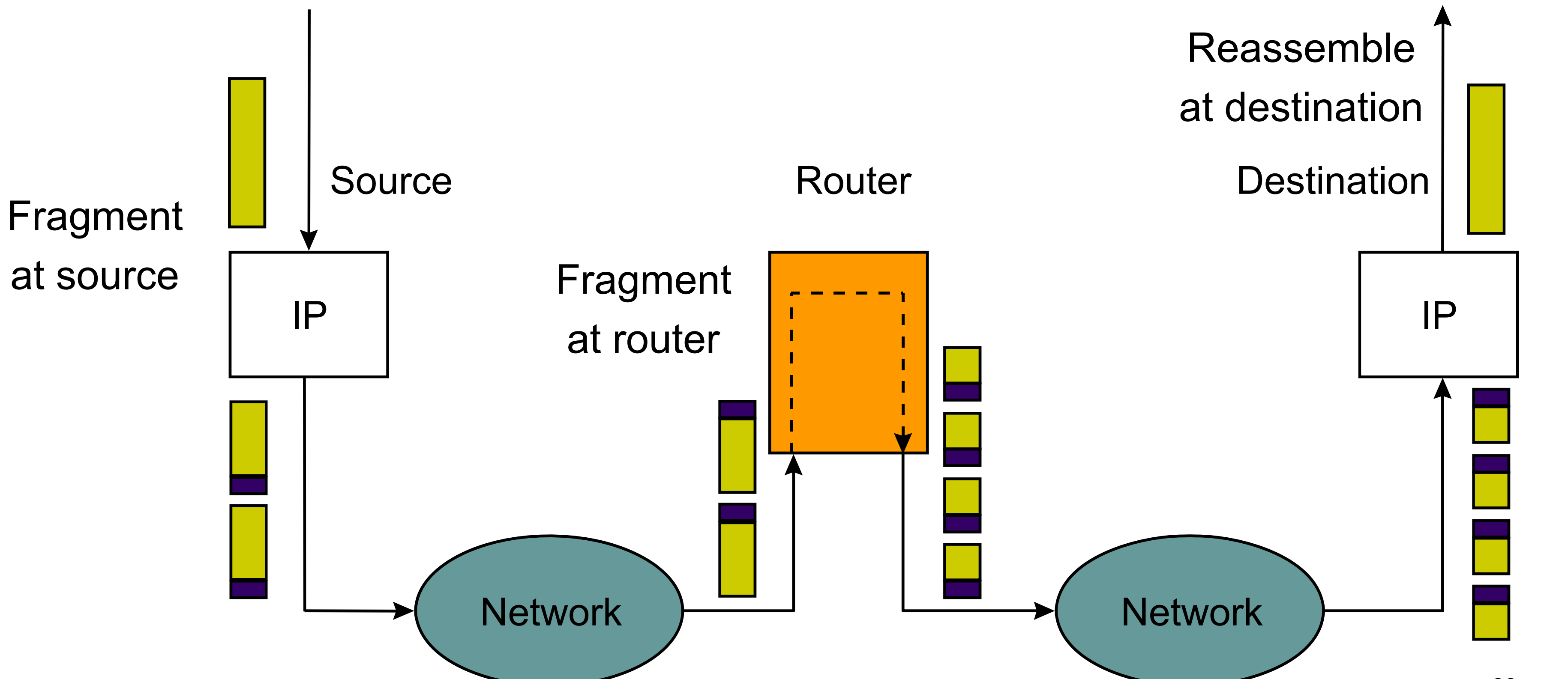
```
⊞ Frame 1 (42 bytes on wire, 42 bytes captured)
⊟ Ethernet II, Src: 00:01:03:1d:cc:f7, Dst: ff:ff:ff:ff:ff:ff
    Destination: ff:ff:ff:ff:ff:ff (Broadcast)
    Source: 00:01:03:1d:cc:f7 (3COM_1d:cc:f7)
    Type: ARP (0x0806)
⊟ Address Resolution Protocol (request)
    Hardware type: Ethernet (0x0001)
    Protocol type: IP (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (0x0001)
    Sender MAC address: 00:01:03:1d:cc:f7 (3COM_1d:cc:f7)
    Sender IP address: 192.168.2.18 (192.168.2.18)
    Target MAC address: 00:00:00:00:00:00 (tesla.comm.utoronto.ca)
    Target IP address: 192.168.2.1 (192.168.2.1)
```

```
0000  ff ff ff ff ff ff 00 01  03 1d cc f7 08 06 00 01   ........ ........
0010  08 00 06 04 00 01 00 01  03 1d cc f7 c0 a8 02 12   ........ ........
0020  00 00 00 00 00 00 c0 a8  02 01                     ........ ..
```

Filter: Reset Apply File: <capture> Drops: 0

Fragmentation and Reassembly

- Identification identifies a particular packet
- Flags = (unused, don't fragment/DF, more fragment/MF)
- Fragment offset identifies the location of a fragment within a packet



Example: Fragmenting a Packet

- A packet is to be forwarded to a network with MTU of 576 bytes. The packet has an IP header of 20 bytes and a data part of 1484 bytes. and of each fragment.
- Maximum data length per fragment = 576 - 20 = 556 bytes.
- We set maximum data length to 552 bytes to get multiple of 8.

	Total Length	Id	MF	Fragment Offset
Original packet	1504	x	0	0
Fragment 1	572	x	1	0
Fragment 2	572	x	1	69
Fragment 3	400	x	0	138

Internet Control Message Protocol (ICMP)

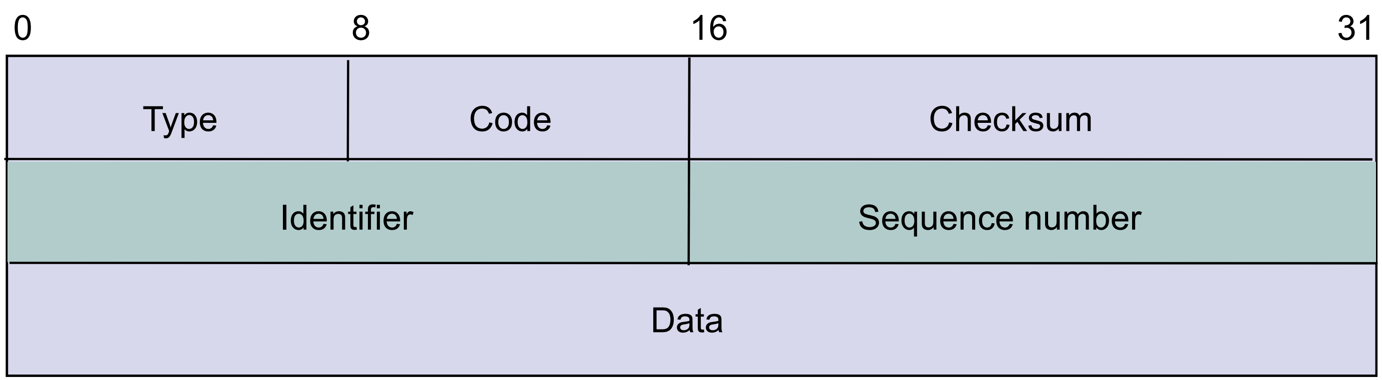
- RFC 792; Encapsulated in IP packet (protocl type = 1)
- Handles error and control messages
- If router cannot deliver or forward a packet, it sends an ICMP “host unreachable” message to the source
- If router receives packet that should have been sent to another router, it sends an ICMP “redirect” message to the sender; Sender modifies its routing table
- ICMP “router discovery” messages allow host to learn about routers in its network and to initialize and update its routing tables
- ICMP echo request and reply facilitate diagnostic and used in “ping”

ICMP Basic Error Message Format

0	8	16	31
Type	Code	Checksum	
Unused			
IP header and 64 bits of original datagram			

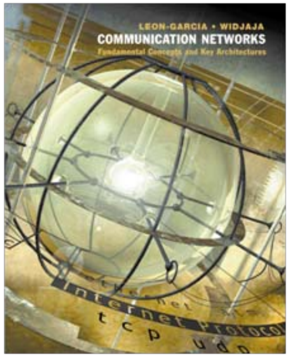
- *Type* of message: some examples
  - 0 Network Unreachable; 3 Port Unreachable
  - 1 Host Unreachable 4 Fragmentation needed
  - 2 Protocol Unreachable 5 Source route failed
  - 11 Time-exceeded, code=0 if TTL exceeded
- Code: purpose of message
- IP header & 64 bits of original datagram
  - To match ICMP message with original data in IP packet

Echo Request & Echo Reply Message Format



- Echo request: type=8; Echo reply: type=0
  - Destination replies with echo reply by copying data in request onto reply message
- Sequence number to match reply to request
- ID to distinguish between different sessions using echo services
- Used in PING

Chapter 8
Communication Networks and Services



IPv6

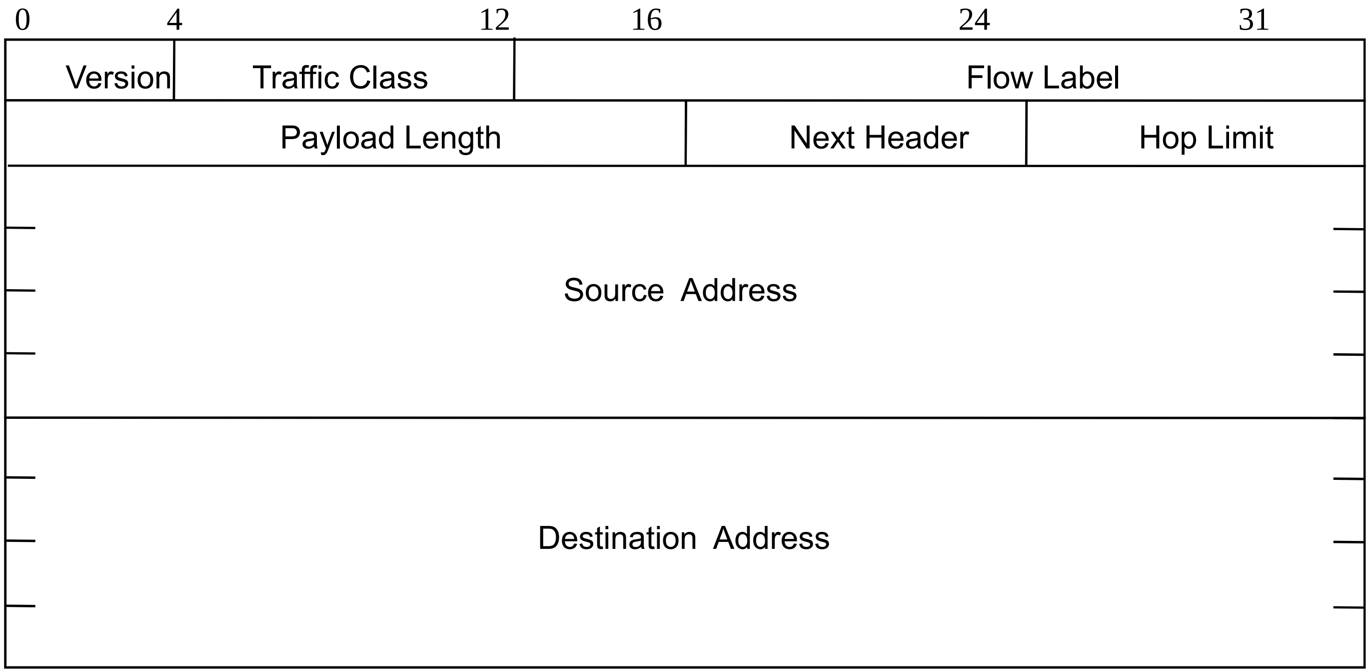
IPv6

- **Longer address field:**
  - 128 bits can support up to 3.4 x 10^{38} hosts
- **Simplified header format:**
  - Simpler format to speed up processing of each header
  - All fields are of fixed size
  - IPv4 vs IPv6 fields:
    - Same: Version
    - Dropped: Header length, ID/flags/frag offset, header checksum
    - Replaced:
      - Datagram length by Payload length
      - Protocol type by Next header
      - TTL by Hop limit
      - TOS by traffic class
    - New: Flow label

Other IPv6 Features

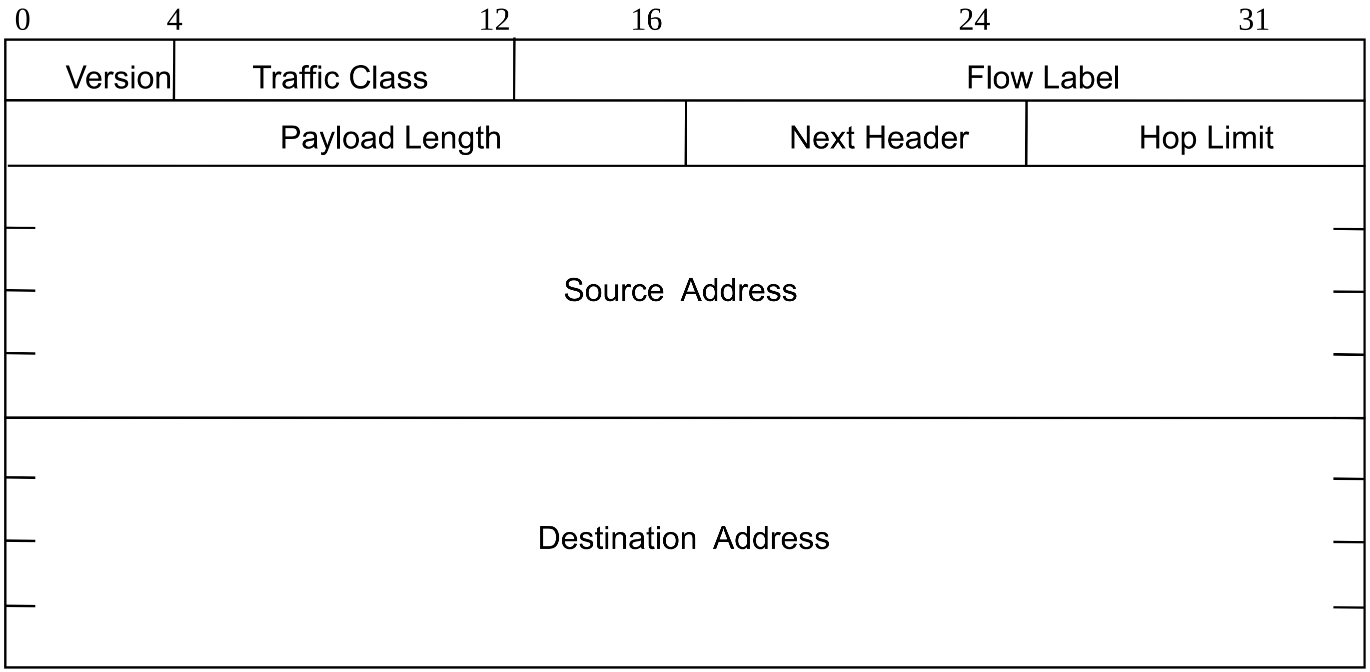
- **Flexible support for options:** more efficient and flexible options encoded in optional *extension headers*
- **Flow label capability:** “flow label” to identify a packet flow that requires a certain QoS
- **Security:** built-in authentication and confidentiality
- **Large packets:** supports payloads that are longer than 64 K bytes, called *jumbo* payloads.
- **Fragmentation at source only:** source should check the minimum MTU along the path
- **No checksum field:** removed to reduce packet processing time in a router

IPv6 Header Format



- Version field same size, same location
- Traffic class to support differentiated services
- Flow: sequence of packets from particular source to particular destination for which source requires special handling

IPv6 Header Format



- Payload length: length of data excluding header, up to 65535 B
- Next header: type of extension header that follows basic header
- Hop limit: # hops packet can travel before being dropped by a router

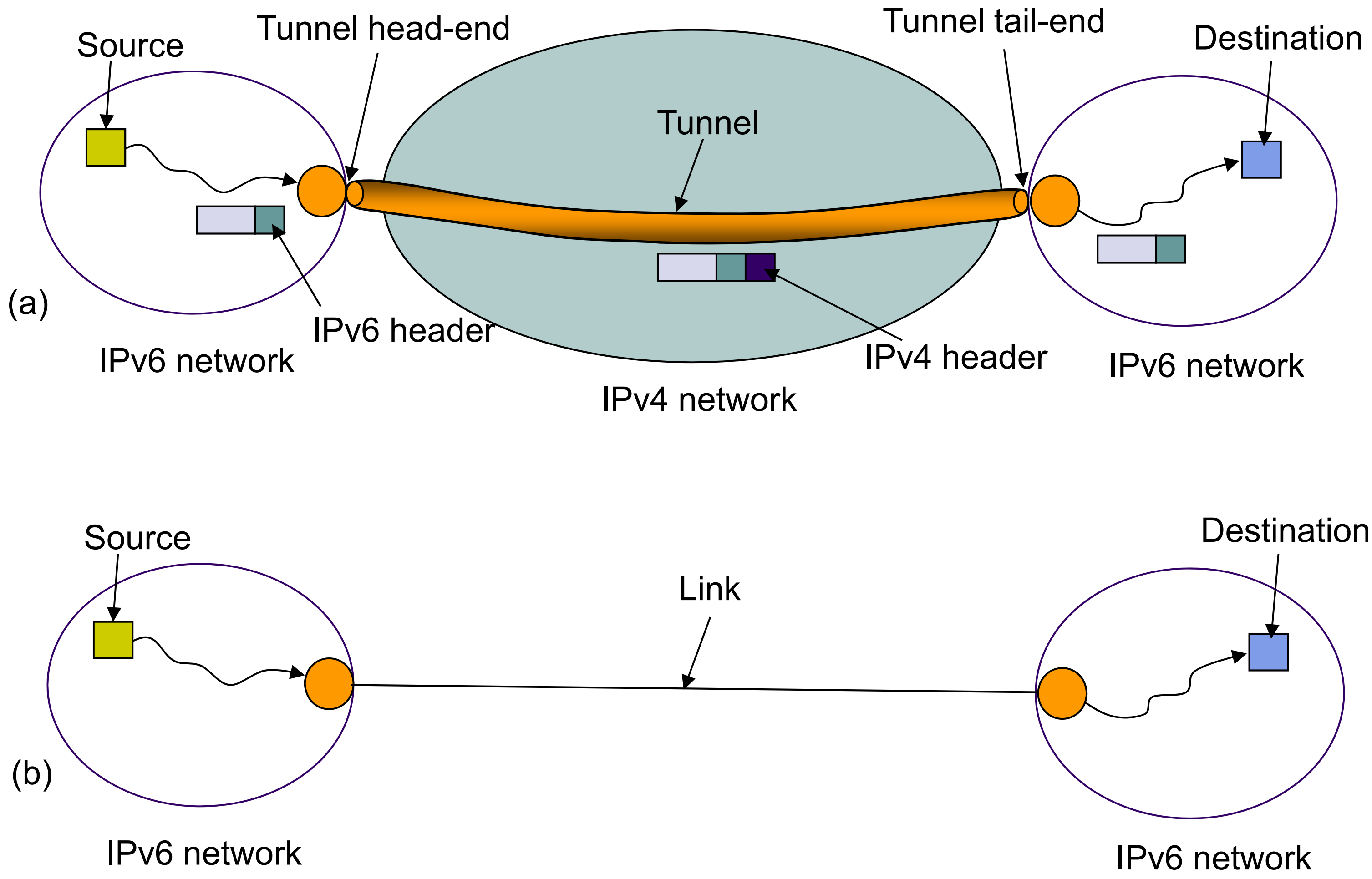
IPv6 Addressing

- Address Categories
  - Unicast: single network interface
  - Multicast: group of network interfaces, typically at different locations. Packet sent to all.
  - Anycast: group of network interfaces. Packet sent to only one interface in group, e.g. nearest.
- Hexadecimal notation
  - Groups of 16 bits represented by 4 hex digits
  - Separated by colons
    - 4BF5:AA12:0216:FEBC:BA5F:039A:BE9A:2176
  - Shortened forms:
    - 4BF5:0000:0000:0000:BA5F:039A:000A:2176
    - To 4BF5:0:0:0:BA5F:39A:A:2176
    - To 4BF5::BA5F:39A:A:2176
  - Mixed notation:
    - ::FFFF:128.155.12.198

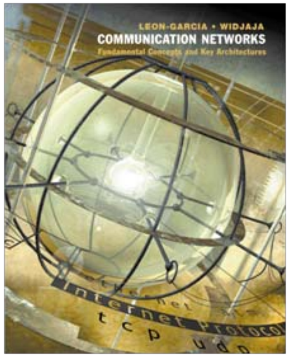
Migration from IPv4 to IPv6

- Gradual transition from IPv4 to IPv6
- Dual IP stacks: routers run IPv4 & IPv6
  - Type field used to direct packet to IP version
- IPv6 islands can tunnel across IPv4 networks
  - Encapsulate user packet insider IPv4 packet
  - Tunnel endpoint at source host, intermediate router, or destination host
  - Tunneling can be recursive

Migration from IPv4 to IPv6



Chapter 8
Communication Networks and Services



Transport Layer Protocols: UDP and TCP

Outline

- UDP Protocol
- TCP Reliable Stream Service
- TCP Protocol
- TCP Connection Management
- TCP Flow Control
- TCP Congestion Control

UDP

- Best effort datagram service
- Multiplexing enables sharing of IP datagram service
- Simple transmitter & receiver
  - Connectionless: no handshaking & no connection state
  - Low header overhead
  - No flow control, no error control, no congestion control
  - UDP datagrams can be lost or out-of-order
- Applications
  - multimedia (e.g. RTP)
  - network services (e.g. DNS, RIP, SNMP)

UDP Datagram

0	16	31
Source Port	Destination Port	
UDP Length	UDP Checksum	
Data		

0-255
- Well-known ports

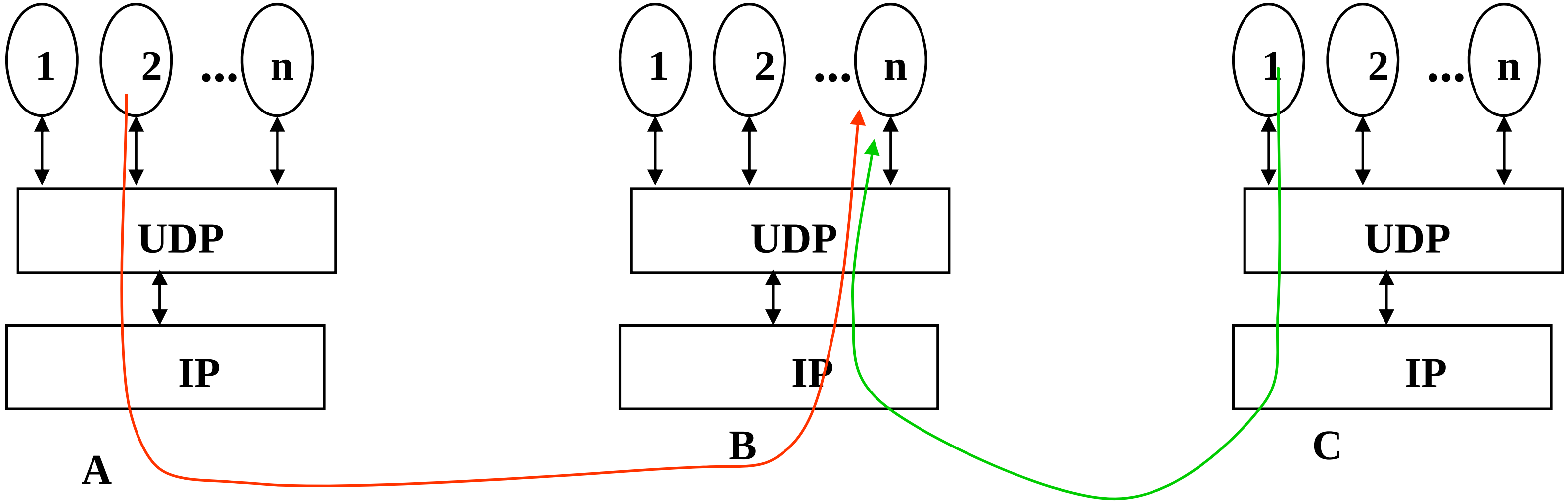
256-1023
- Less well-known ports

1024-65536
- Ephemeral client ports

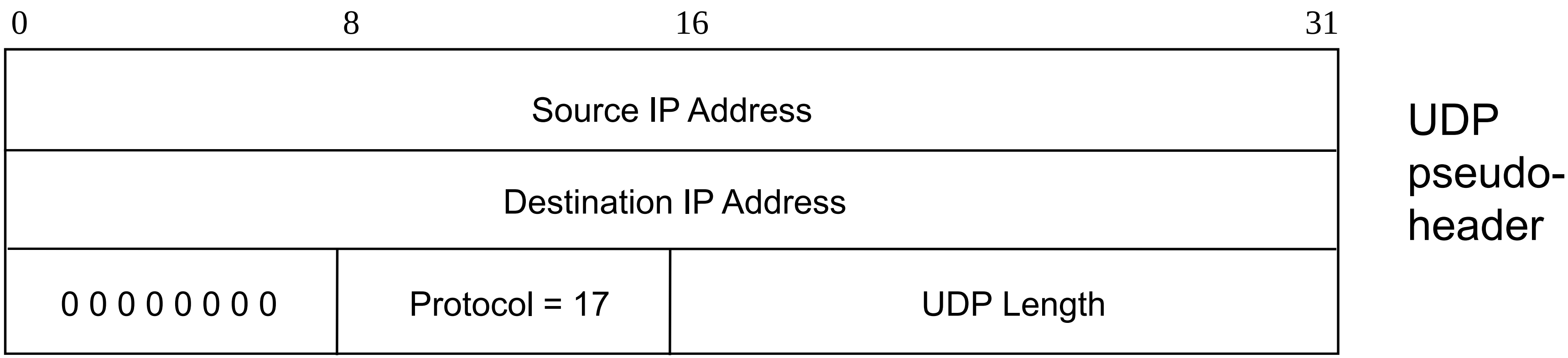
- Source and destination port numbers
  - Client ports are ephemeral
  - Server ports are well-known
  - Max number is 65,535
- UDP length
  - Total number of bytes in datagram (including header)
  - 8 bytes ≤ length ≤ 65,535
- UDP Checksum
  - Optionally detects errors in UDP datagram

UDP Multiplexing

- All UDP datagrams arriving to IP address B and destination port number *n* are delivered to the same process
- Source port number is not used in multiplexing



UDP Checksum Calculation



- UDP checksum detects for end-to-end errors
- Covers pseudoheader followed by UDP datagram
- IP addresses included to detect against misdelivery
- IP & UDP checksums set to zero during calculation
- Pad with 1 byte of zeros if UDP length is odd

UDP Receiver Checksum

- UDP receiver recalculates the checksum and silently discards the datagram if errors detected
  - “silently” means no error message is generated
- The use of UDP checksums is optional
- But hosts are required to have checksums enabled

Outline

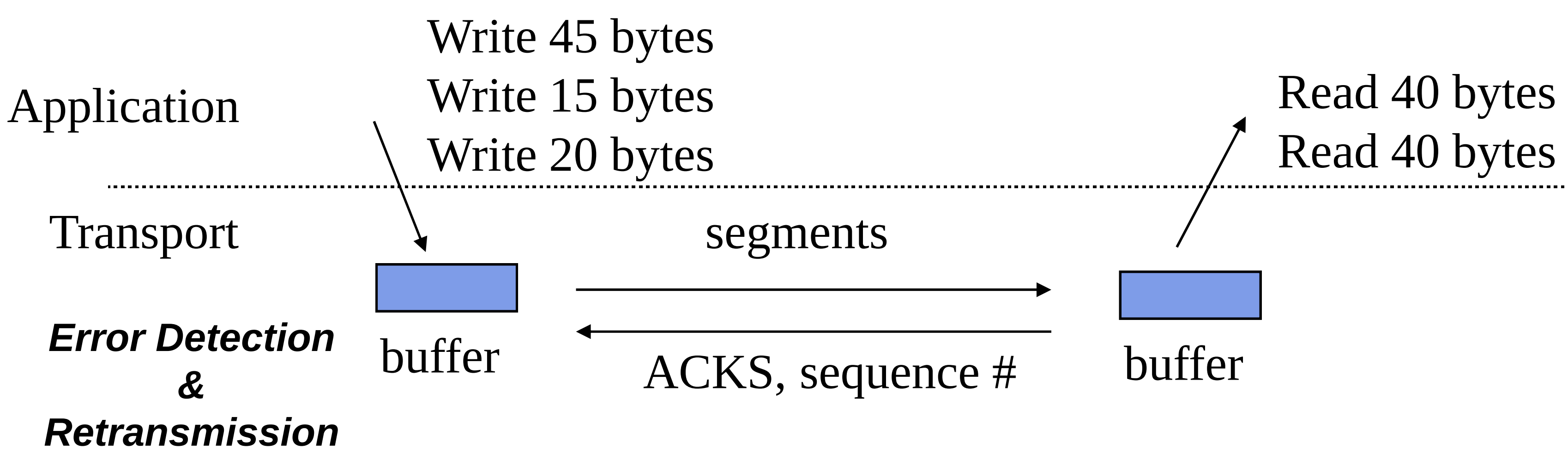
- UDP Protocol
- TCP Reliable Stream Service
- TCP Protocol
- TCP Connection Management
- TCP Congestion Control

TCP

- Reliable byte-stream service
- More complex transmitter & receiver
  - Connection-oriented: full-duplex unicast connection between client & server processes
  - Connection setup, connection state, connection release
  - Higher header overhead
  - Error control, flow control, and congestion control
  - Higher delay than UDP
- Most applications use TCP
  - HTTP, SMTP, FTP, TELNET, POP3, …

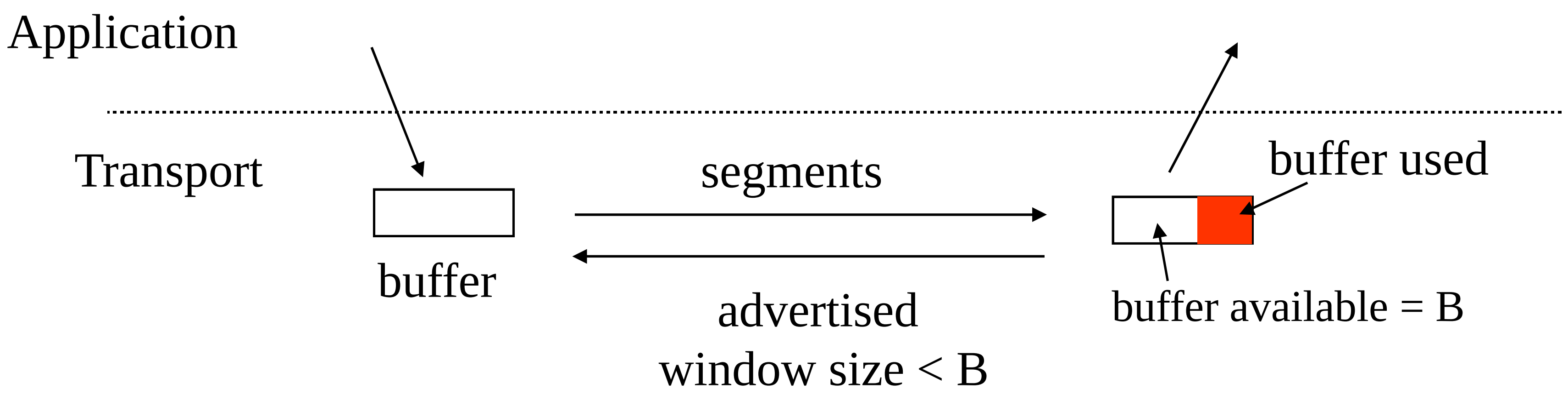
Reliable Byte-Stream Service

- Stream Data Transfer
  - transfers a contiguous stream of bytes across the network, with no indication of boundaries
  - groups bytes into segments
  - transmits segments as convenient (Push function defined)
- Reliability
  - error control mechanism to deal with IP transfer impairments



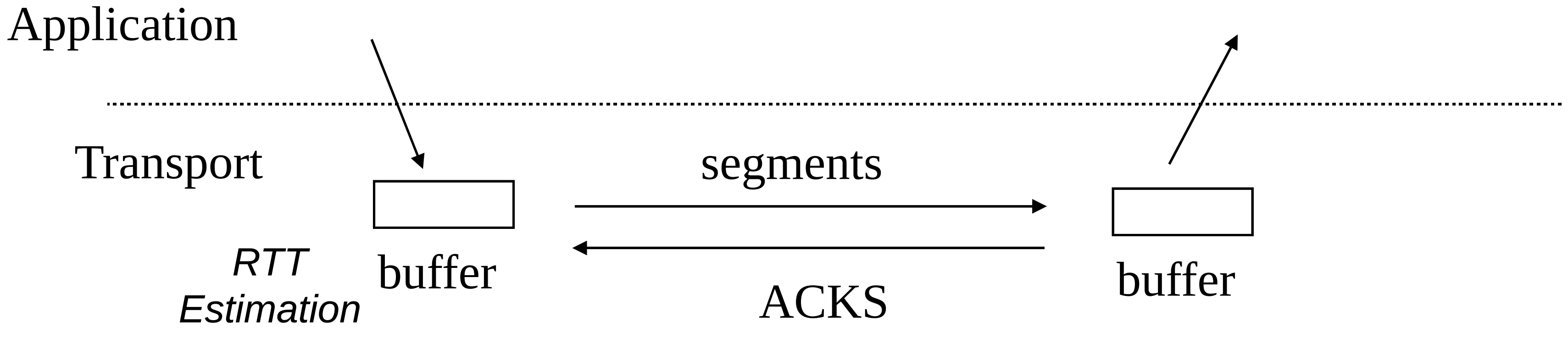
Flow Control

- Buffer limitations & speed mismatch can result in loss of data that arrives at destination
- Receiver controls rate at which sender transmits to prevent buffer overflow



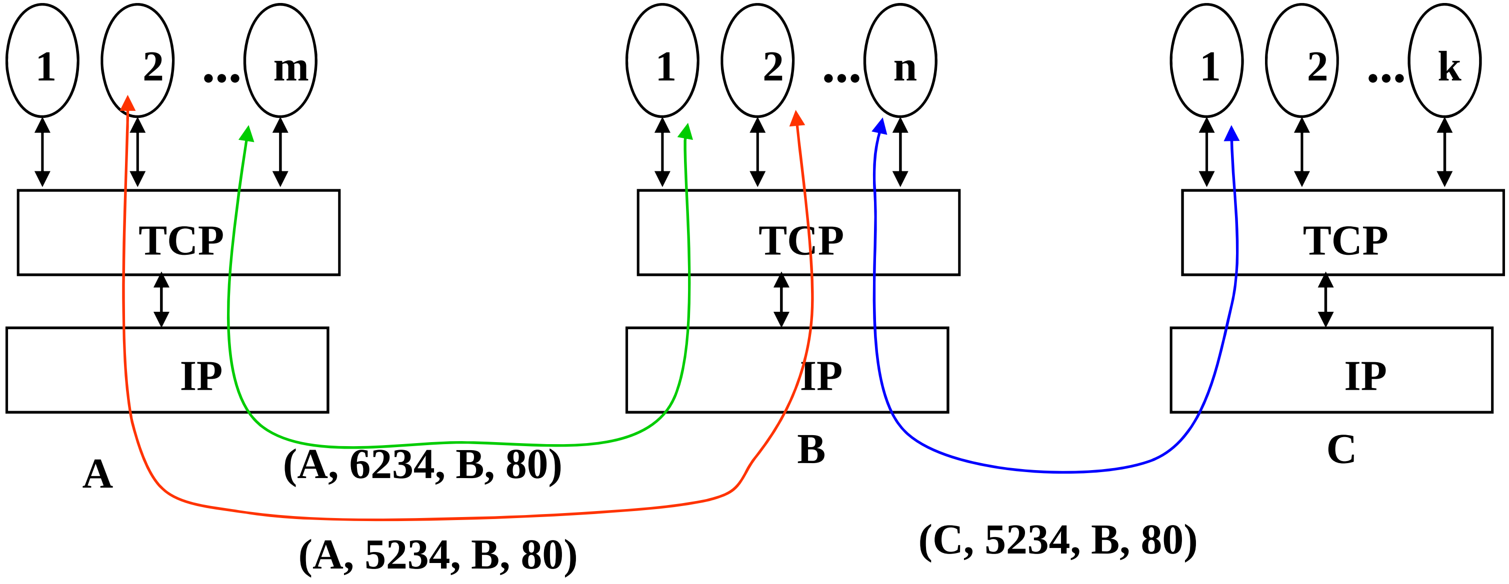
Congestion Control

- Available bandwidth to destination varies with activity of other users
- Transmitter dynamically adjusts transmission rate according to network congestion as indicated by RTT (round trip time) & ACKs
- Elastic utilization of network bandwidth



TCP Multiplexing

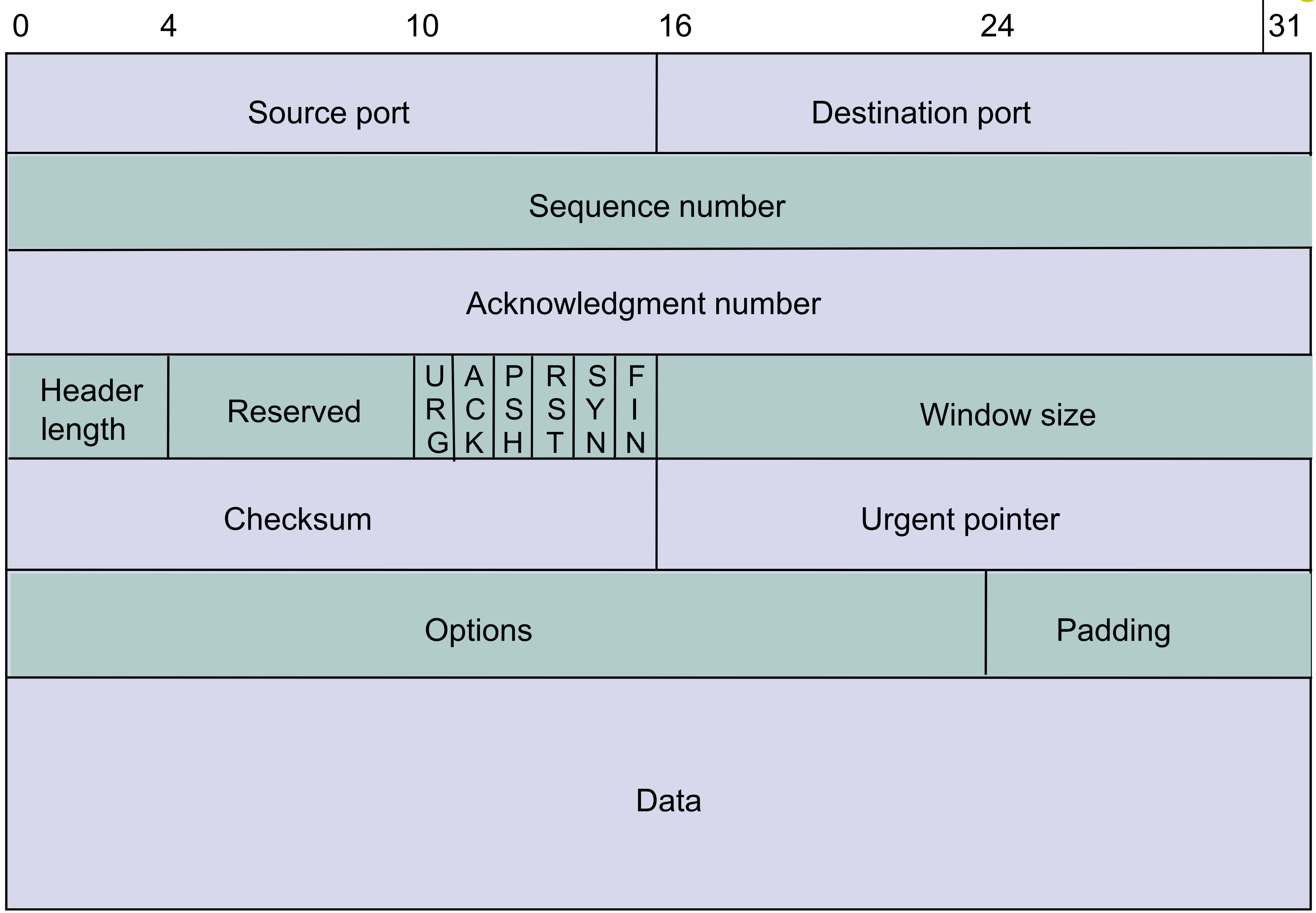
- A *TCP connection* is specified by a *4-tuple*
  - (source IP address, source port, destination IP address, destination port)
- TCP allows multiplexing of multiple connections between end systems to support multiple applications simultaneously
- Arriving segment directed according to connection 4-tuple



Outline

- UDP Protocol
- TCP Reliable Stream Service
- TCP Protocol
- TCP Connection Management
- TCP Congestion Control

TCP Segment Format



- Each TCP segment has header of 20 or more bytes + 0 or more bytes of data

TCP Header

Port Numbers

- A socket identifies a connection endpoint
  - IP address + port
- A connection specified by a *socket pair*
- Well-known ports
  - FTP 20
  - Telnet 23
  - DNS 53
  - HTTP 80

Sequence Number

- Byte count
- First byte in segment
- 32 bits long
- $0 \leq SN \leq 2^{32}-1$
- Initial sequence number selected during connection setup

TCP Header

Acknowledgement Number

- SN of next byte expected by receiver
- Acknowledges that all prior bytes in stream have been received correctly
- Valid if ACK flag is set

Header length

- 4 bits
- Length of header in multiples of 32-bit words
- Minimum header length is 20 bytes
- Maximum header length is 60 bytes

TCP Header

Reserved

- 6 bits

Control

- 6 bits
- URG: urgent pointer flag
  - Urgent message end = SN + **urgent pointer**
- ACK: ACK packet flag
- PSH: override TCP buffering
- RST: reset connection
  - Upon receipt of RST, connection is terminated and application layer notified
- SYN: establish connection
- FIN: close connection

TCP Header

Window Size

- 16 bits to advertise window size
- Used for flow control
- Sender will accept bytes with SN from ACK to ACK + window
- Maximum window size is 65535 bytes

TCP Checksum

- Internet checksum method
- TCP pseudoheader + TCP segment

TCP Checksum Calculation

0	8	16	31
Source IP address			
Destination IP address			
0 0 0 0 0 0 0 0	Protocol = 6	TCP segment length	

TCP pseudo-header

- TCP error detection uses same procedure as UDP

TCP Header

Options

- Variable length
- NOP (No Operation) option is used to pad TCP header to multiple of 32 bits
- Time stamp option is used for round trip measurements

Options

- Maximum Segment Size (MSS) option specifices largest segment a receiver wants to receive
- Window Scale option increases TCP window from 16 to 32 bits

Outline

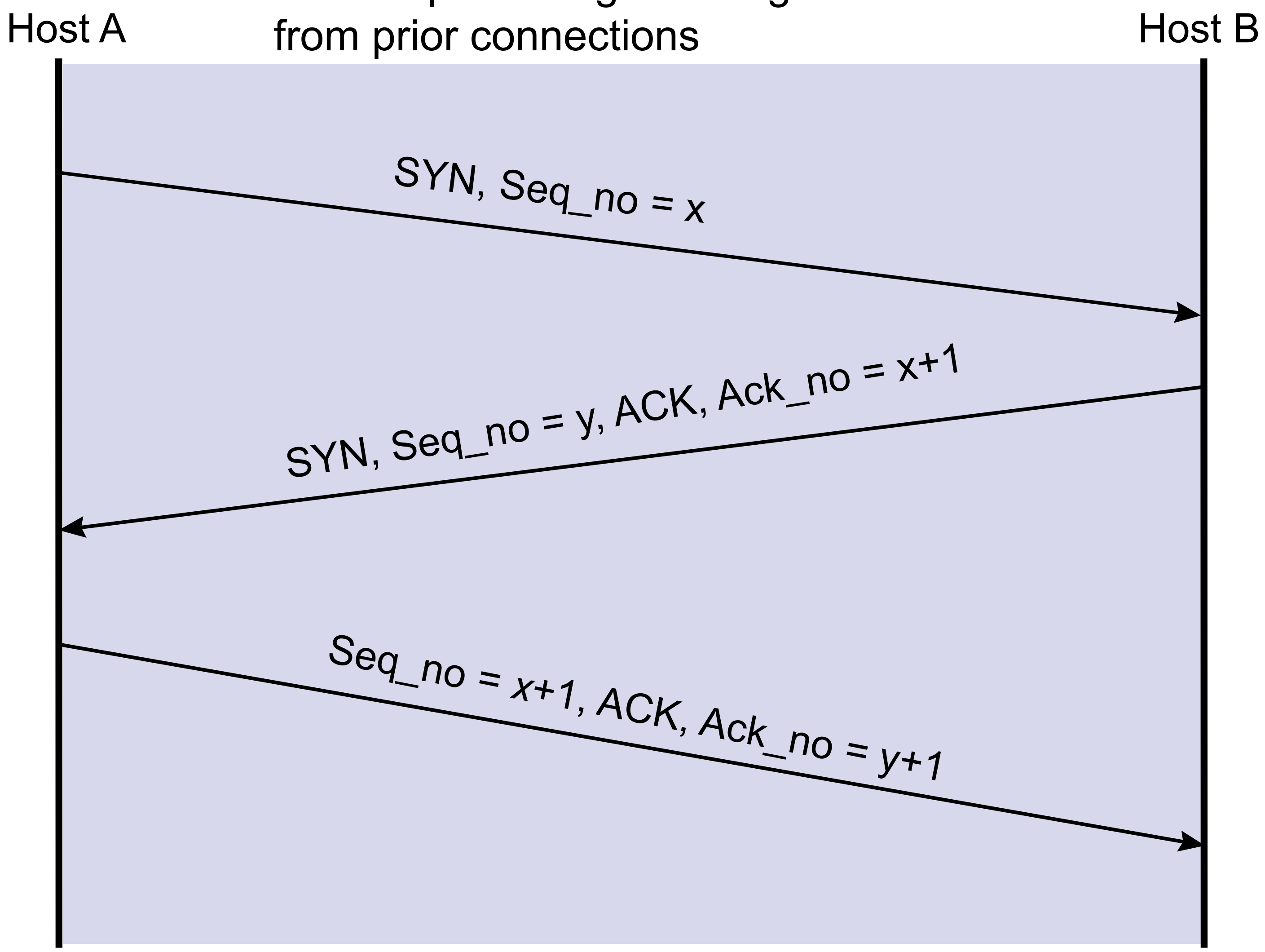
- UDP Protocol
- TCP Reliable Stream Service
- TCP Protocol
- TCP Connection Management
- TCP Congestion Control

Initial Sequence Number

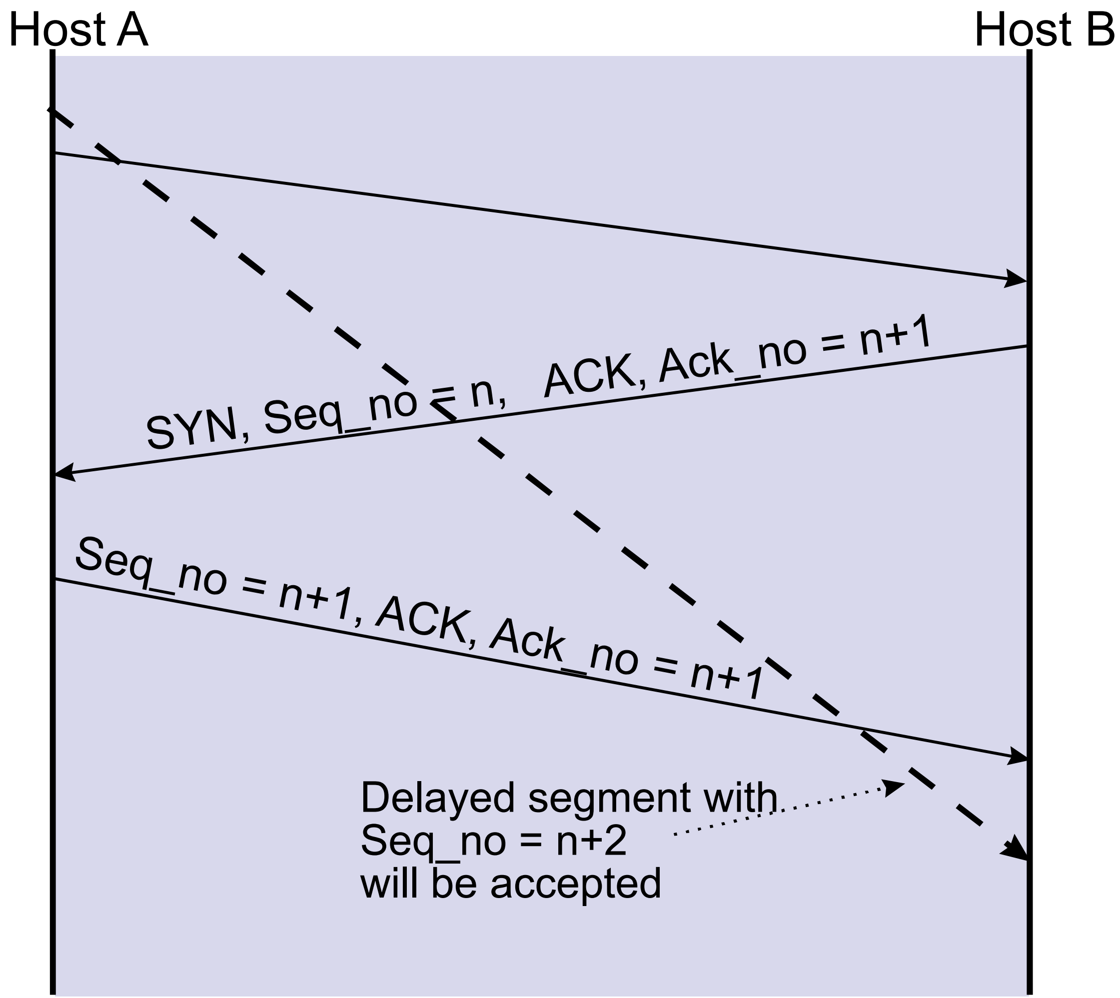
- Select initial sequence numbers (ISN) to protect against segments from prior connections (that may circulate in the network and arrive at a much later time)
- Select ISN to avoid overlap with sequence numbers of prior connections
- Use local clock to select ISN sequence number
- Time for clock to go through a full cycle should be greater than the maximum lifetime of a segment (MSL); Typically MSL=120 seconds
- High bandwidth connections pose a problem
- 2^n > 2 * max packet life * R bytes/second

TCP Connection Establishment

- "Three-way Handshake"
- ISN's protect against segments from prior connections



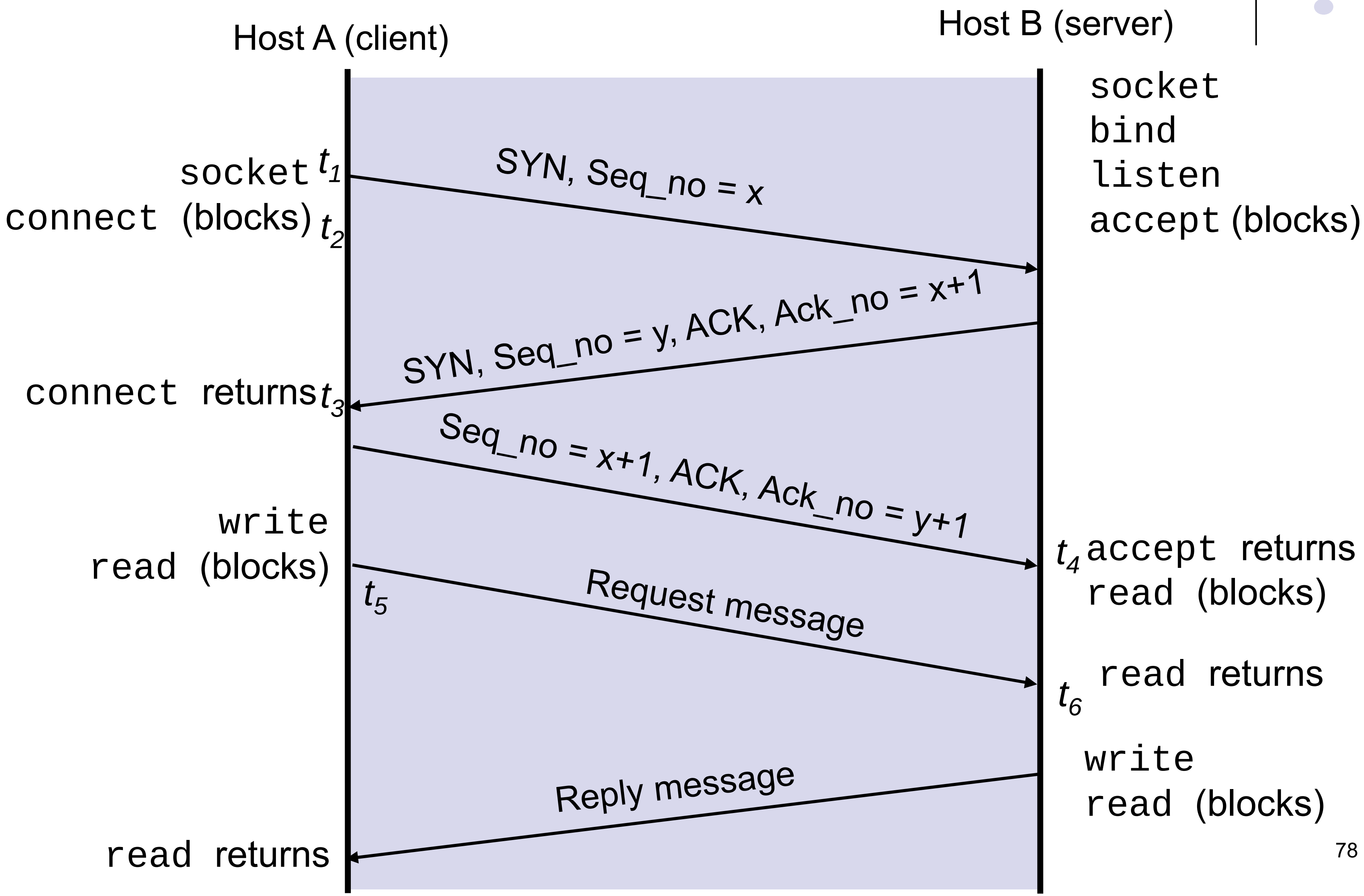
If host always uses the same ISN



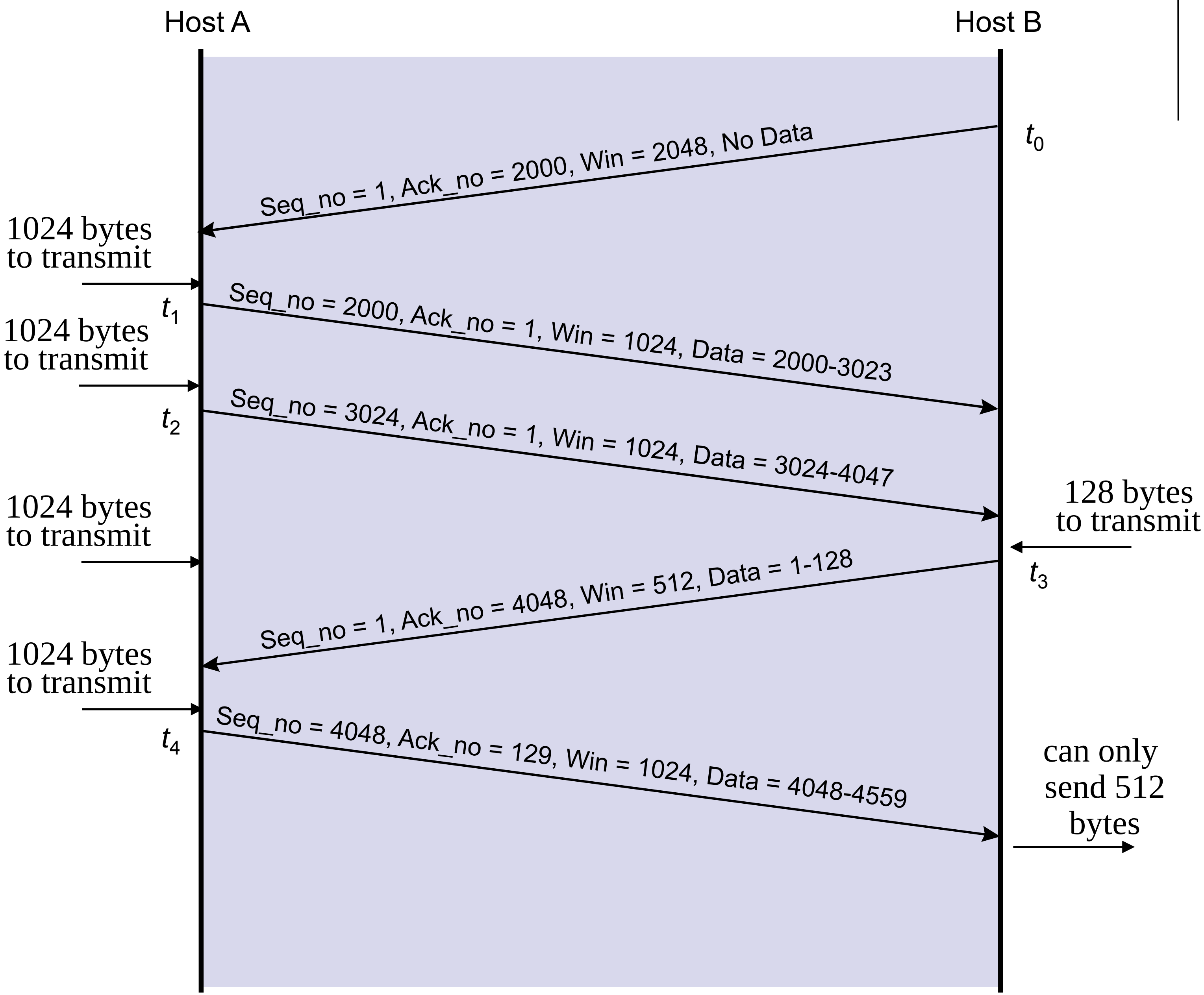
Maximum Segment Size

- Maximum Segment Size
  - largest block of data that TCP sends to other end
- Each end can announce its MSS during connection establishment
- Default is 576 bytes including 20 bytes for IP header and 20 bytes for TCP header
- Ethernet implies MSS of 1460 bytes
- IEEE 802.3 implies 1452

Client-Server Application



TCP Window Flow Control



Nagle Algorithm

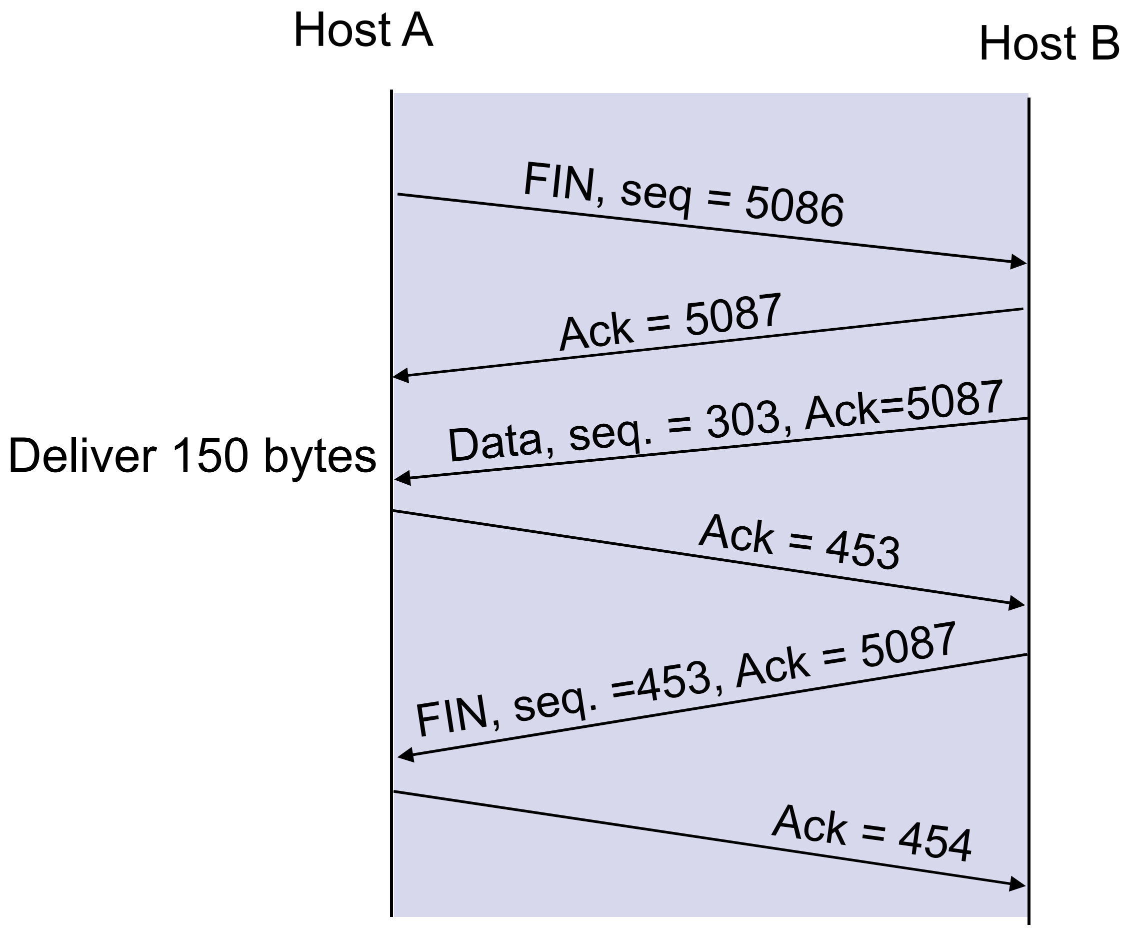
- Situation: user types 1 character at a time
  - Transmitter sends TCP segment per character (41B)
  - Receiver sends ACK (40B)
  - Receiver echoes received character (41B)
  - Transmitter ACKs echo (40 B)
  - 162 bytes transmitted to transfer 1 character!
- Solution:
  - TCP sends data & waits for ACK
  - New characters buffered
  - Send new characters when ACK arrives
  - Algorithm adjusts to RTT
    - Short RTT send frequently at low efficiency
    - Long RTT send less frequently at greater efficiency

Silly Window Syndrome

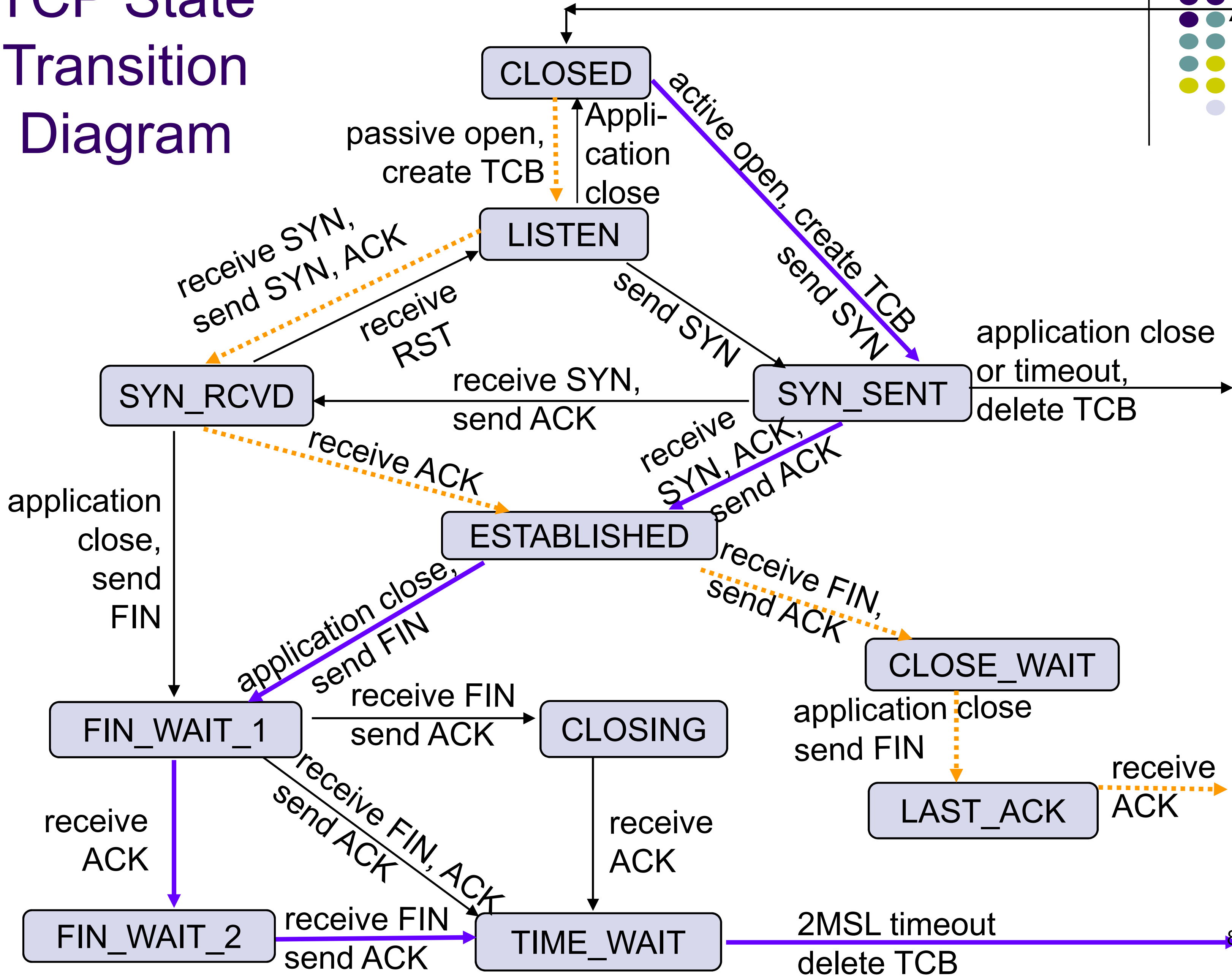
- Situation:
  - Transmitter sends large amount of data
  - Receiver buffer depleted slowly, so buffer fills
  - Every time a few bytes read from buffer, a new advertisement to transmitter is generated
  - Sender immediately sends data & fills buffer
  - Many small, inefficient segments are transmitted
- Solution:
  - Receiver does not advertize window until window is at least ½ of receiver buffer or maximum segment size
  - Transmitter refrains from sending small segments

TCP Connection Closing

"Graceful Close"



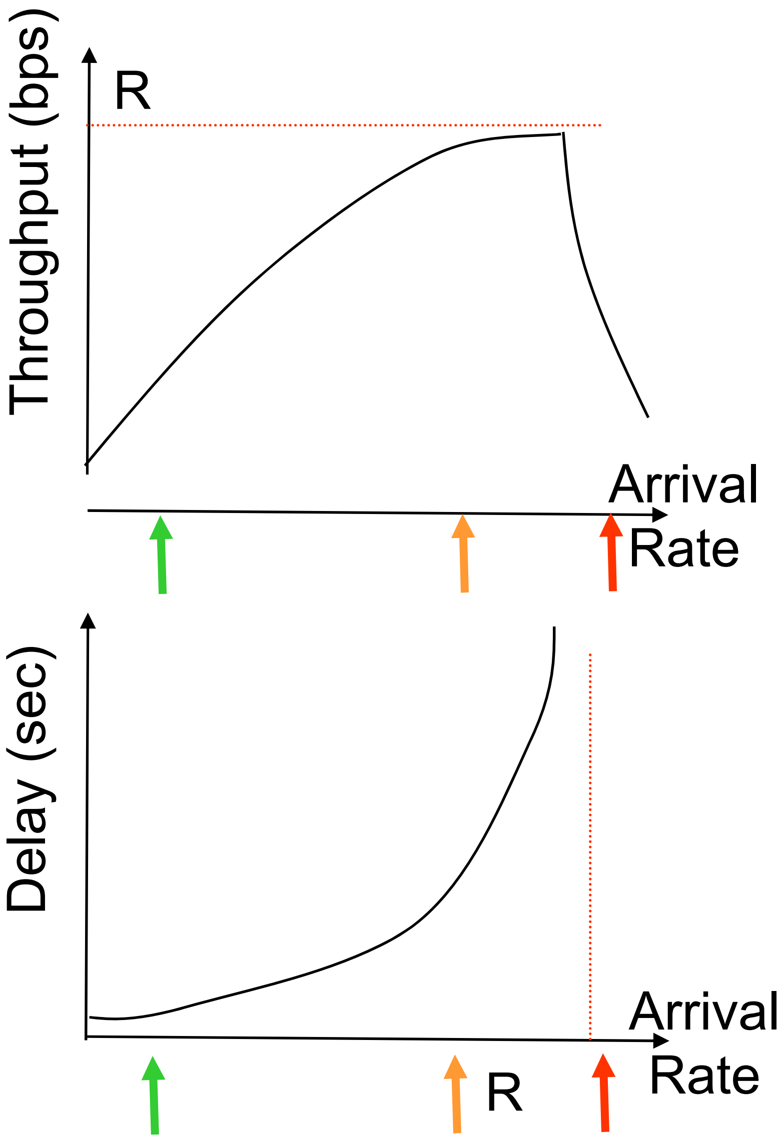
TCP State Transition Diagram



Outline

- UDP Protocol
- TCP Reliable Stream Service
- TCP Protocol
- TCP Connection Management
- TCP Congestion Control

Phases of Congestion Behavior



1. Light traffic
   - Arrival Rate << R
   - Low delay
   - Can accommodate more
2. Knee (congestion onset)
   - Arrival rate approaches R
   - Delay increases rapidly
   - Throughput begins to saturate
3. Congestion collapse
   - Arrival rate > R
   - Large delays, packet loss
   - Useful application throughput drops

Window Congestion Control

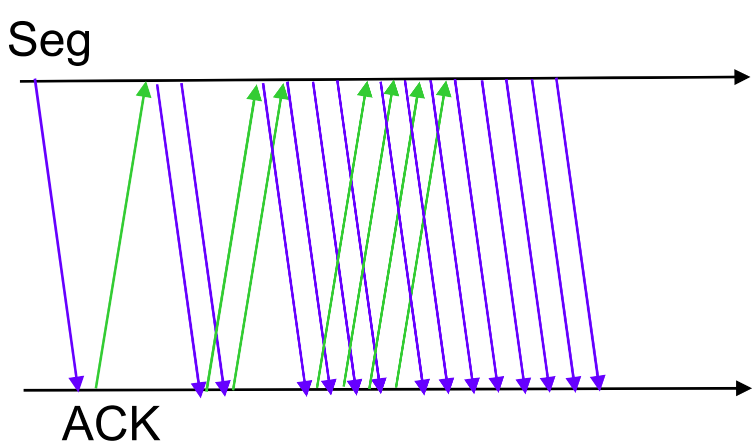
- Desired operating point: just before knee
  - Sources must control their sending rates so that aggregate arrival rate is just before knee
- TCP sender maintains a *congestion window* cwnd to control congestion at intermediate routers
- Effective window is minimum of congestion window and advertised window
- Problem: source does not know what its “fair” share of available bandwidth should be
- Solution: adapt dynami ally to available BW
  - Sources probe the network by increasing cwnd
  - When congestion detected, sources reduce rate
  - Ideally, sources sending rate stabilizes near ideal point

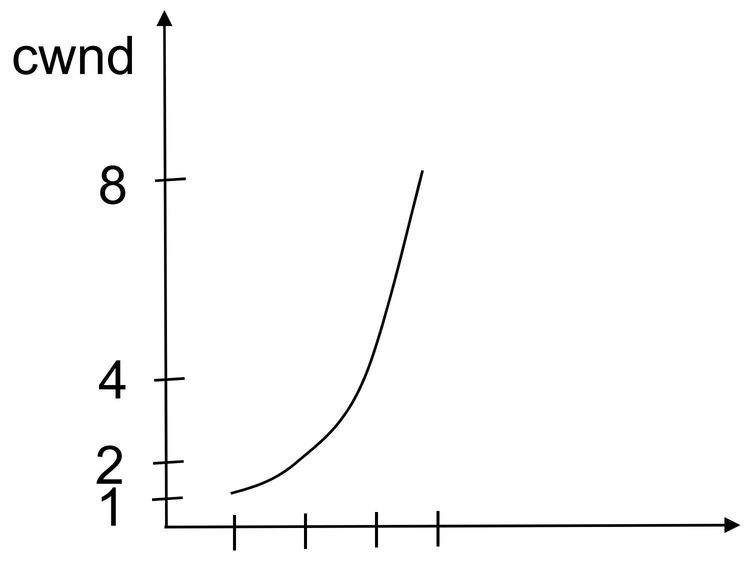
Congestion Window

- How does the TCP congestion algorithm change congestion window dynamically according to the most up-to-date state of the network?
- At light traffic: each segment is ACKed quickly
  - Increase cwnd aggresively
- At knee: segment ACKs arrive, but more slowly
  - Slow down increase in cwnd
- At congestion: segments encounter large delays (so retransmission timeouts occur); segments are dropped in router buffers (resulting in duplicate ACKs)
  - Reduce transmission rate, then probe again

TCP Congestion Control: Slow Start

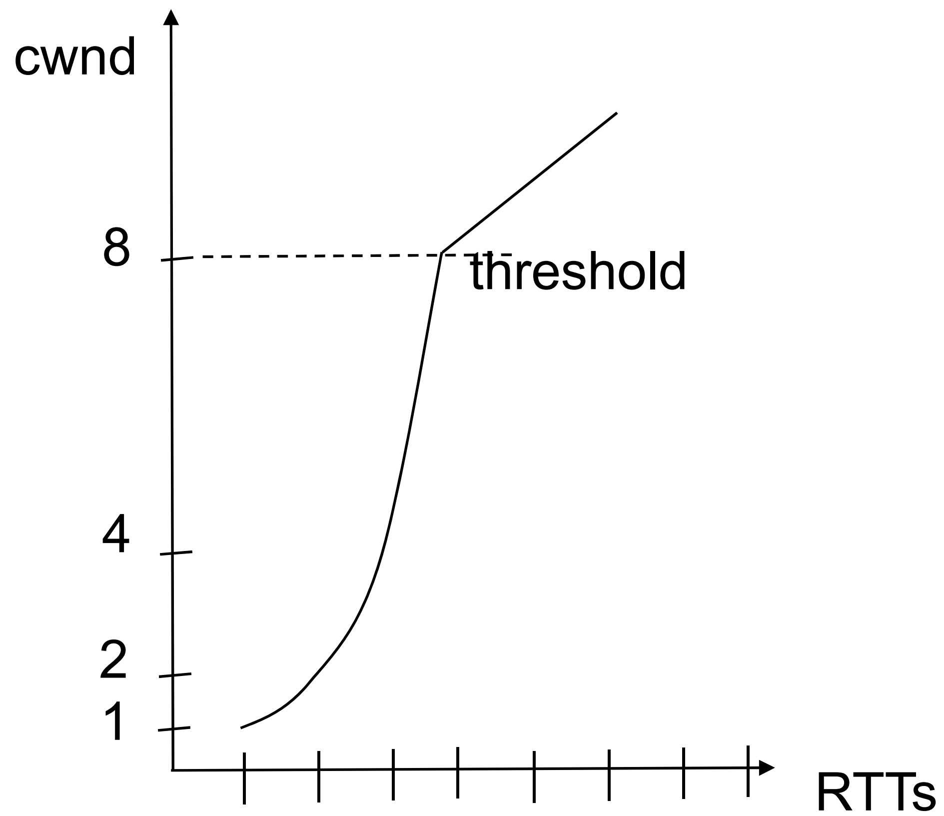
- **Slow start**: increase congestion window size by one segment upon receiving an ACK from receiver
  - initialized at ≤ 2 segments
  - used at (re)start of data transfer
  - congestion window increases exponentially



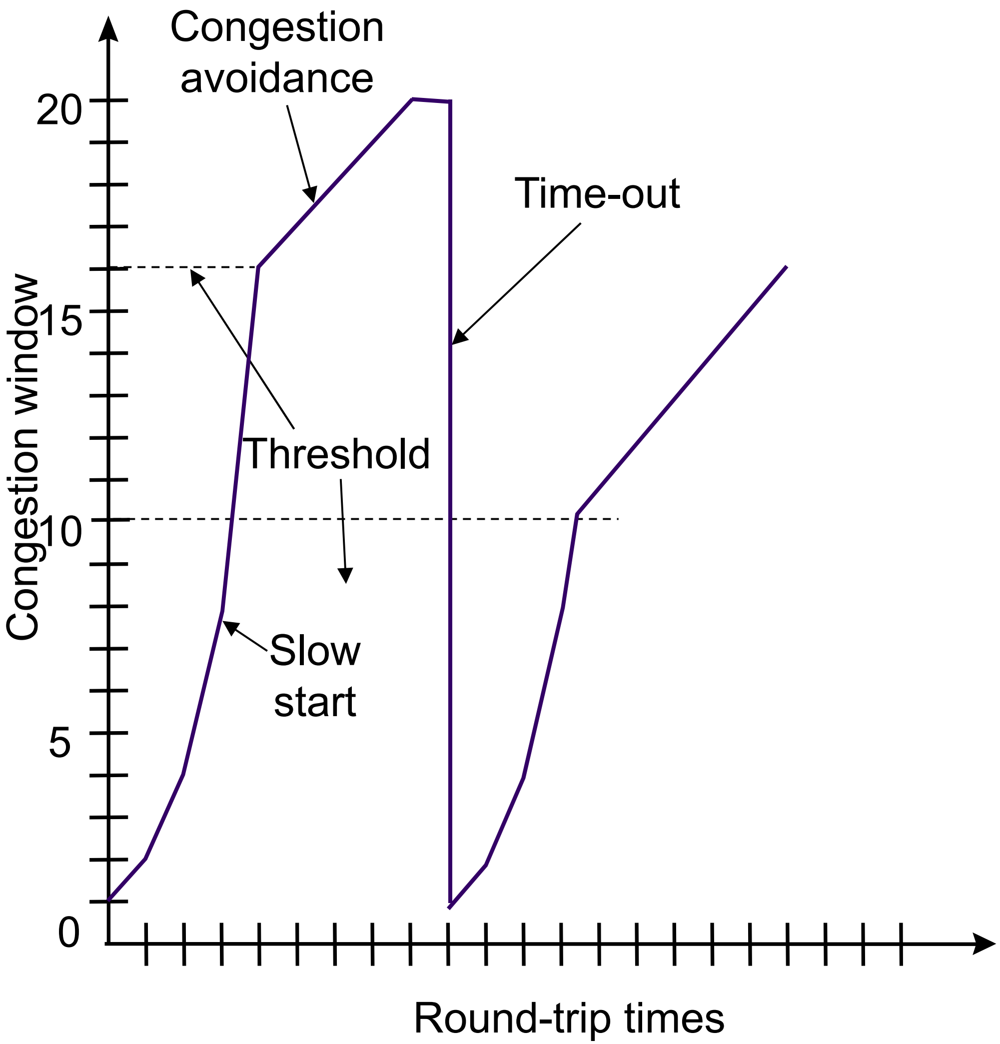


TCP Congestion Control: Congestion Avoidance

- Algorithm progressively sets a *congestion threshold*
  - When cwnd > threshold, slow down rate at which cwnd is increased
- Increase congestion window size by one segment per round-trip-time (RTT)
  - Each time an ACK arrives, cwnd is increased by 1/cwnd
  - In one RTT, cwnd segments are sent, so total increase in cwnd is cwnd x 1/cwnd = 1
  - cwnd grows linearly with time



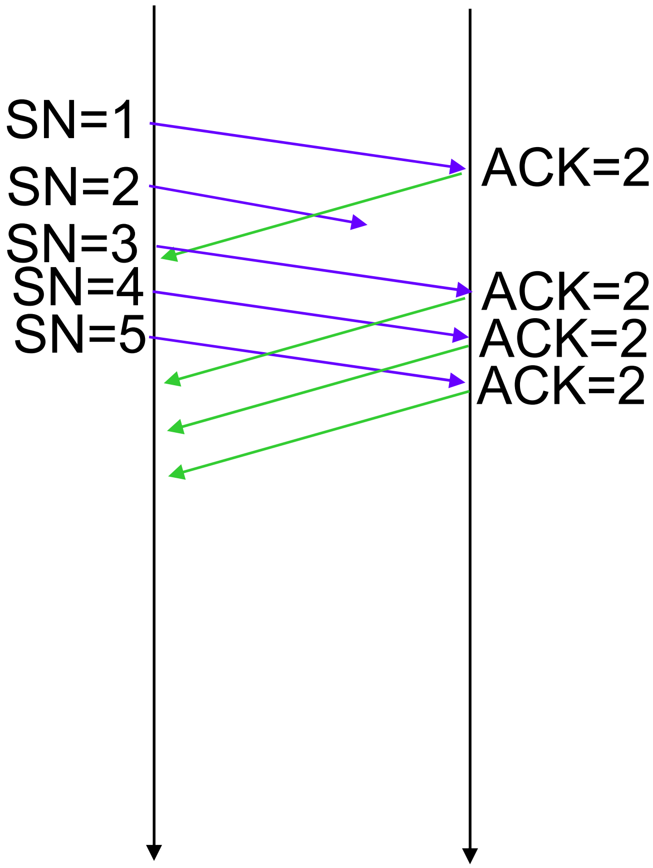
TCP Congestion Control: Congestion



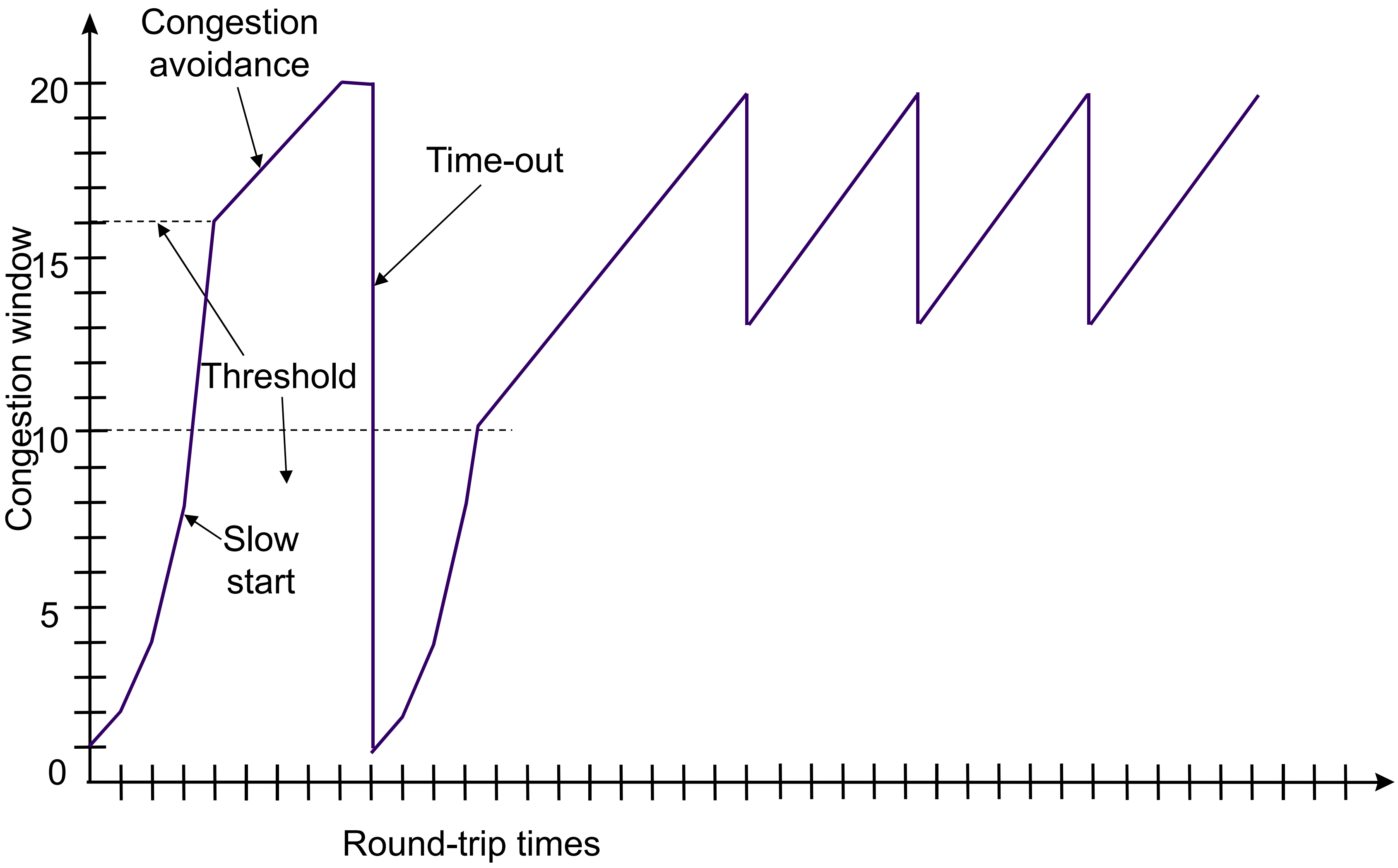
- Congestion is detected upon timeout or receipt of duplicate ACKs
- Assume current cwnd corresponds to available bandwidth
- Adjust congestion threshold = ½ x current cwnd
- Reset cwnd to 1
- Go back to slow-start
- Over several cycles expect to converge to congestion threshold equal to about ½ the available bandwidth

Fast Retransmit & Fast Recovery

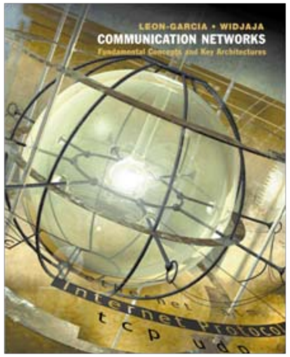
- Congestion causes many segments to be dropped
- If only a single segment is dropped, then subsequent segments trigger duplicate ACKs before timeout
- Can avoid large decrease in cwnd as follows:
  - When three duplicate ACKs arrive, retransmit lost segment immediately
  - Reset congestion threshold to ½ cwnd
  - Reset cwnd to congestion threshold + 3 to account for the three segments that triggered duplicate ACKs
  - Remain in congestion avoidance phase
  - However if timeout expires, reset cwnd to 1
  - In absence of timeouts, cwnd will oscillate around optimal value



TCP Congestion Control:
Fast Retransmit & Fast Recovery



Chapter 8
Communication Networks and Services



Internet Routing Protocols

Outline

- Basic Routing
- Routing Information Protocol (RIP)
- Open Shortest Path First (OSPF)
- Border Gateway Protocol (BGP)

Routing and Forwarding

- Routing
  - How to determine the routing table entries
    - carried out by routing daemon
- Forwarding
  - Look up routing table & forward packet from input to output port
    - carried out by IP layer

Routers exchange information using routing protocols to develop the routing tables

Autonomous Systems

- Global Internet viewed as collection of autonomous systems.
- **Autonomous system (AS)** is a set of routers or networks administered by a single organization
- Same routing protocol need not be run within the AS
- But, to the outside world, an AS should present a *consistent picture of what ASs are reachable* through it
- **Stub AS:** has only a single connection to the outside world.
- **Multihomed AS:** has multiple connections to the outside world, but refuses to carry transit traffic
- **Transit AS:** has multiple connections to the outside world, and can carry transit and local traffic.

AS Number

- For exterior routing, an AS needs a globally unique AS 16-bit integer number
- Currently, there are about 11,000 registered ASs in Internet (and growing)
- *Stub AS*, which is the most common type, does not need an AS number since the prefixes are placed at the provider's routing table
- *Transit AS* needs an AS number
- Request an AS number from the ARIN, RIPE and APNIC

Inter and Intra Domain Routing

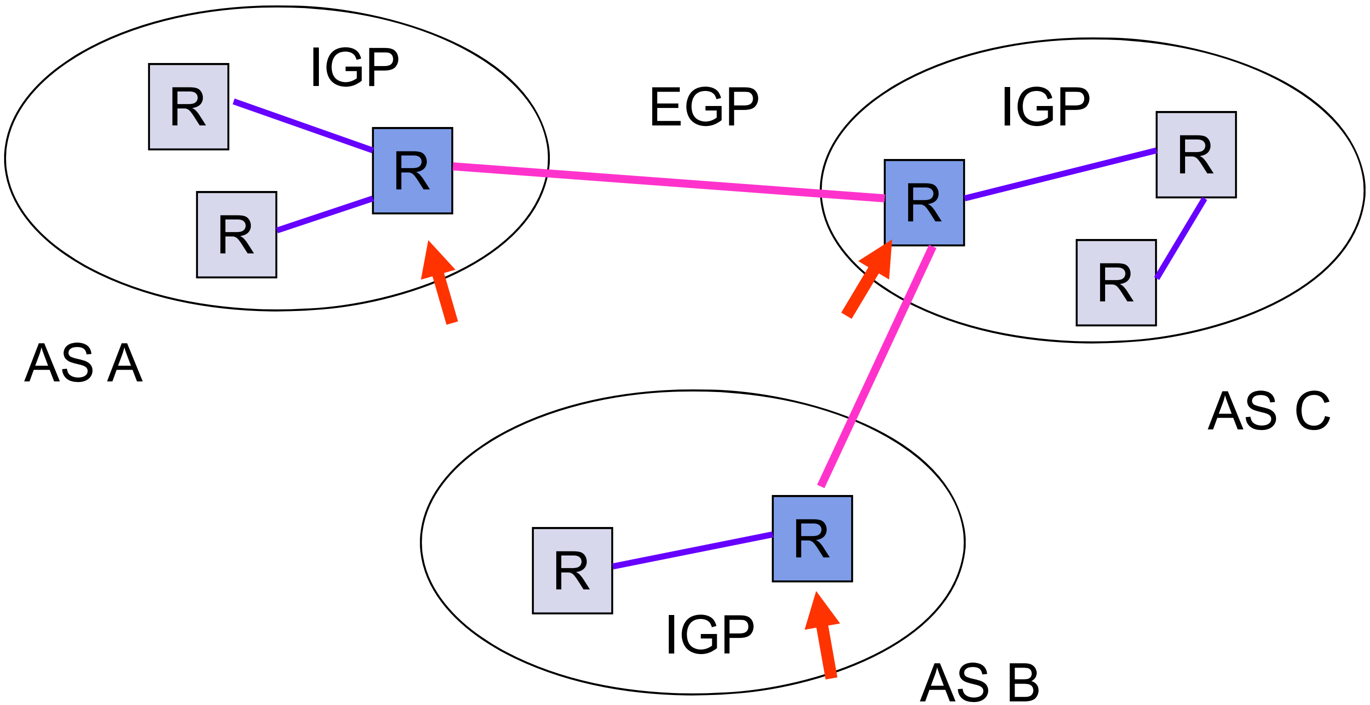
Interior Gateway Protocol (IGP): routing within AS

- RIP, OSPF

Exterior Gateway Protocol (EGP): routing between AS's

- BGPv4

Border Gateways perform IGP & EGP routing



Outline

- Basic Routing
- Routing Information Protocol (RIP)
- Open Shortest Path First (OSPF)
- Border Gateway Protocol (BGP)

Routing Information Protocol (RIP)

- RFC 1058
- **RIP** based on routed, “route d”, distributed in BSD UNIX
- Uses the **distance-vector algorithm**
- Runs on top of UDP, port number 520
- Metric: number of hops
- Max limited to 15
  - suitable for small networks (local area environments)
  - value of 16 is reserved to represent infinity
  - small number limits the *count-to-infinity* problem

RIP Operation

- Router sends update message to neighbors every 30 sec
- A router expects to receive an update message from each of its neighbors within 180 seconds in the worst case
- If router does not receive update message from neighbor X within this limit, it assumes the link to X has failed and sets the corresponding minimum cost to 16 (infinity)
- Uses ***split horizon with poisoned reverse***
- Convergence speeded up by triggered updates
  - neighbors notified immediately of changes in distance vector table

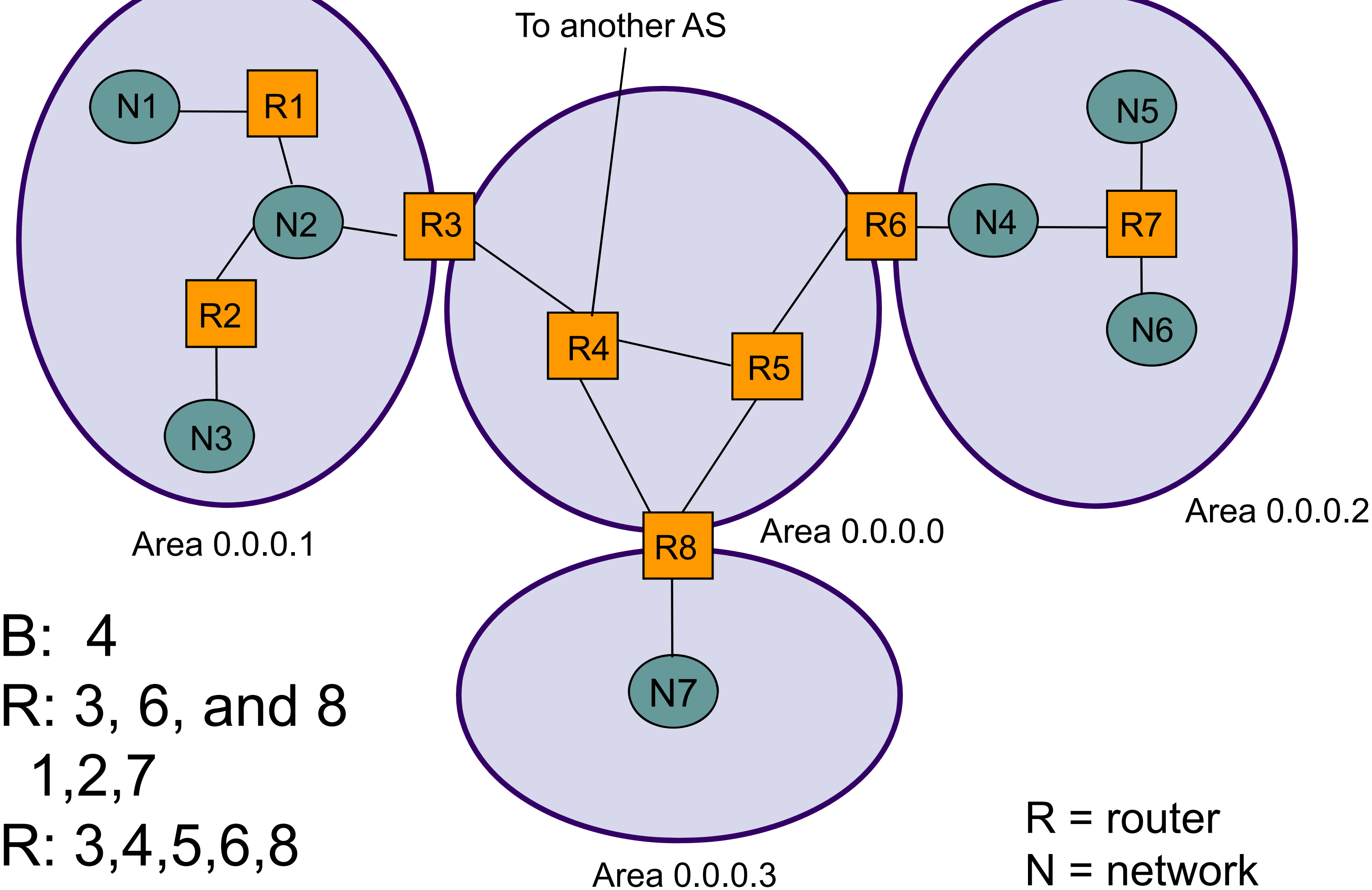
Outline

- Basic Routing
- Routing Information Protocol (RIP)
- Open Shortest Path First (OSPF)
- Border Gateway Protocol (BGP)

Open Shortest Path First

- RFC 2328 (v2)
- Fixes some of the deficiencies in RIP
- Enables each router to learn complete network topology
- Each router monitors the *link state* to each neighbor and floods the link-state information to other routers
- Each router builds an identical *link-state database*
- Allows router to build shortest path tree with router as root
- OSPF typically converges faster than RIP when there is a failure in the network

OSPF Areas



ASB: 4
ABR: 3, 6, and 8
IR: 1,2,7
BBR: 3,4,5,6,8

R = router
N = network

Outline

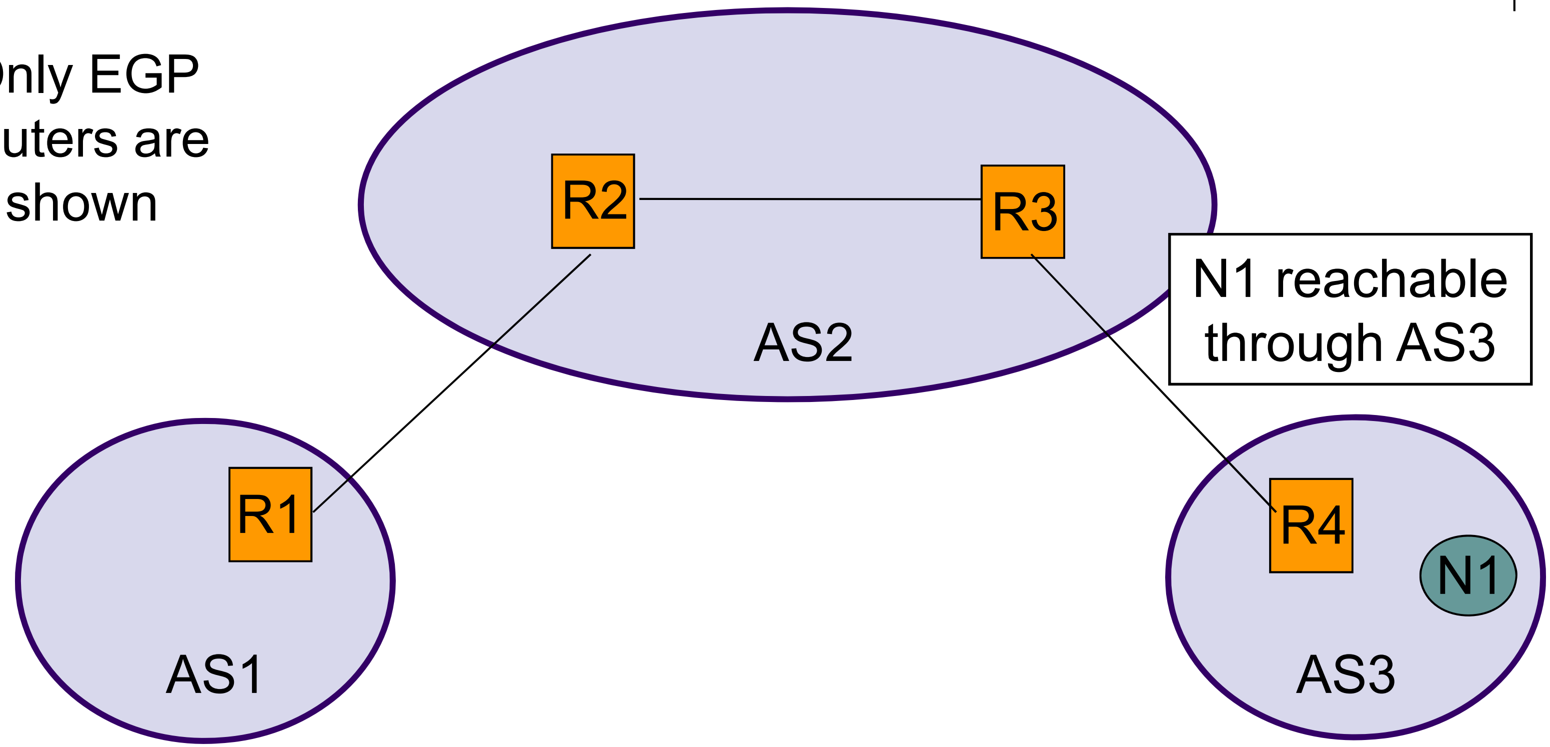
- Basic Routing
- Routing Information Protocol (RIP)
- Open Shortest Path First (OSPF)
- Border Gateway Protocol (BGP)

Exterior Gateway Protocols

- Within each AS, there is a consistent set of routes connecting the constituent networks
- The Internet is woven into a coherent whole by *Exterior Gateway Protocols (EGPs)* that operate between AS's
- EGP enables two AS's to exchange routing information about:
  - The networks that are contained within each AS
  - The AS's that can be reached through each AS
- EGP path selection guided by policy rather than path optimality
  - Trust, peering arrangements, etc

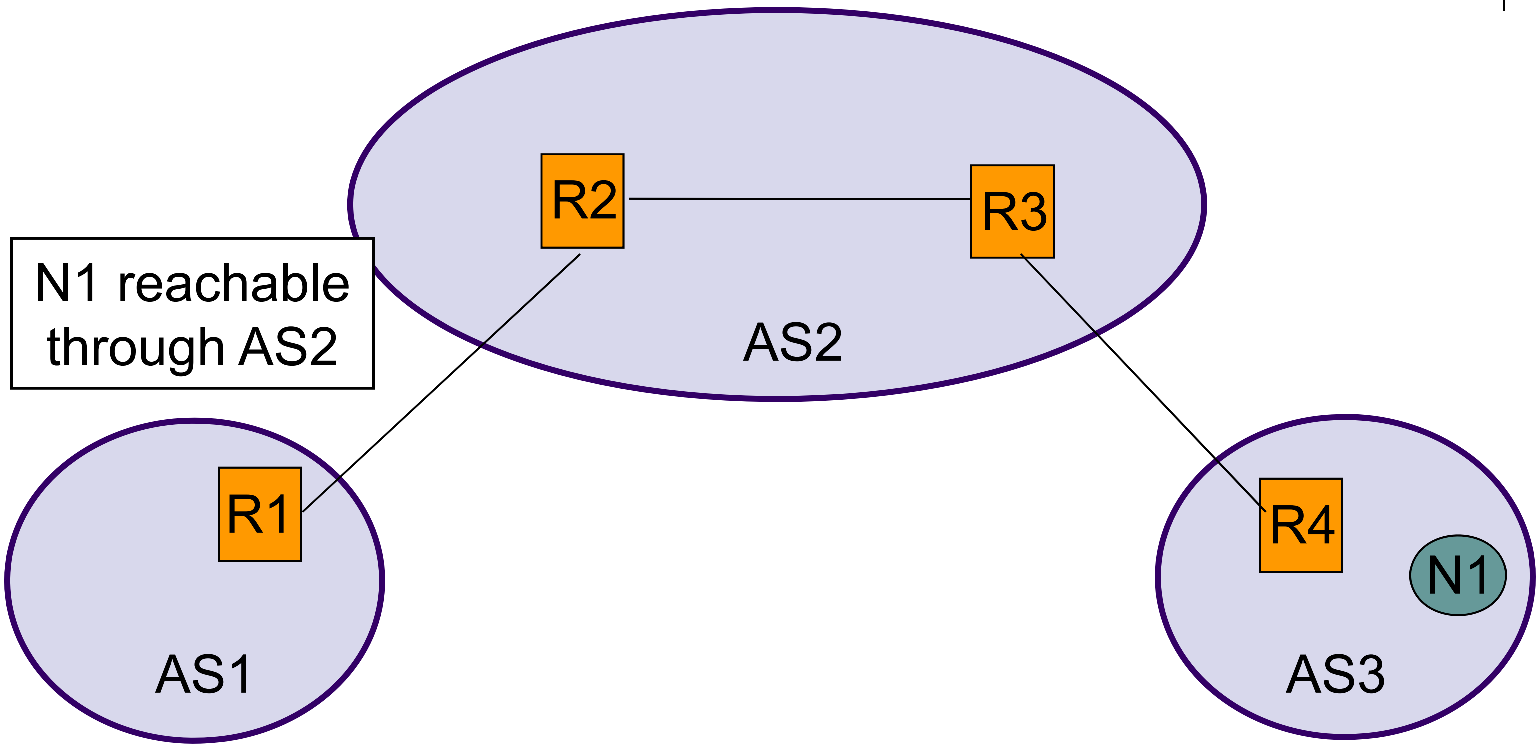
EGP Example

Only EGP routers are shown



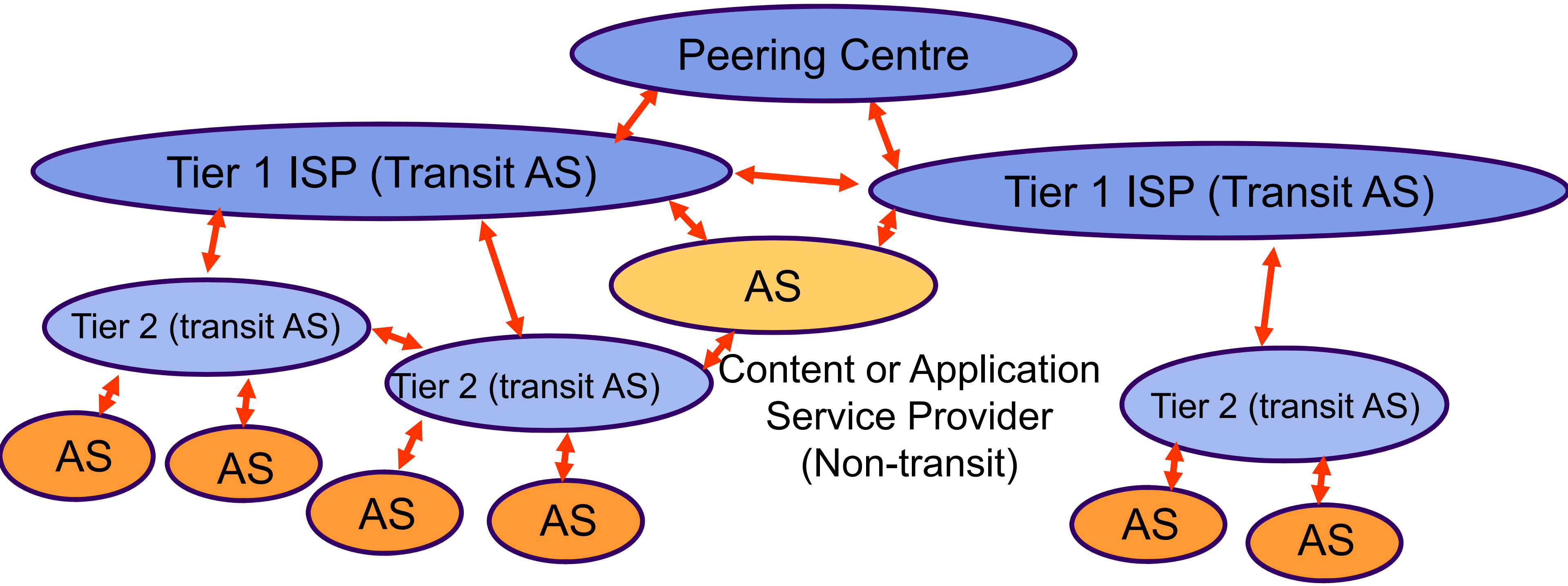
- R4 advertises that network N1 can be reached through AS3
- R3 examines announcement & applies *policy* to decide whether it will forward packets to N1 through R4
- If yes, routing table updated in R3 to indicate R4 as next hop to N1
- IGP propagates N1 reachability information through AS2

EGP Example



- EGP routers within an AS, e.g. R3 and R2, are kept consistent
- Suppose AS2 willing to handle *transit* packets from AS1 to N1
- R2 advertises to AS1 the reachability of N1 through AS2
- R1 applies its policy to decide whether to send to N1 via AS2

Peering and Inter-AS connectivity

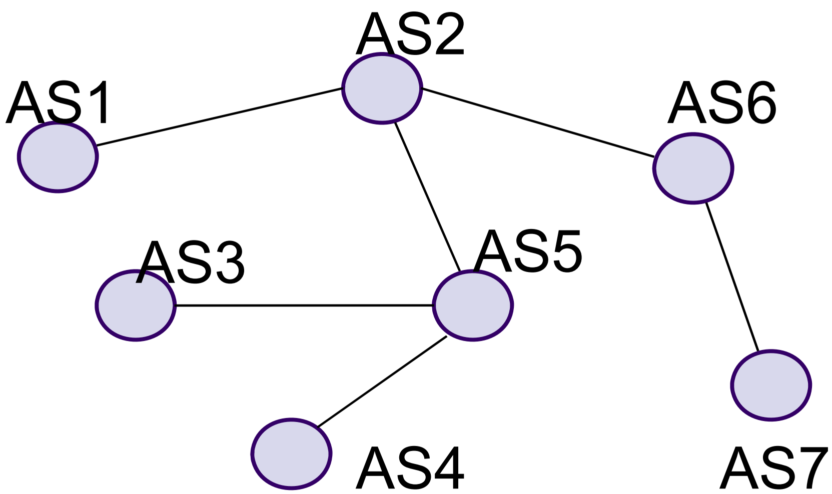


- Non-transit AS's (stub & multihomed) do not carry transit traffic
- Tier 1 ISPs peer with each other, privately & peering centers
- Tier 2 ISPs peer with each other & obtain transit services from Tier 1s; Tier 1's carry transit traffic between their Tier 2 customers
- Client AS's obtain service from Tier 2 ISPs

EGP Requirements

- Scalability to global Internet
  - Provide connectivity at global scale
  - Link-state does not scale
  - Should promote address aggregation
  - Fully distributed
- EGP path selection guided by policy rather than path optimality
  - Trust, peering arrangements, etc
  - EGP should allow flexibility in choice of paths

Border Gateway Protocol v4



- BGP (RFC 1771) an EGP routing protocol to exchange network reachability information among BGP routers (also called *BGP speakers*)
- Network reachability info contains sequence of ASs that packets traverse to reach a destination network
- Info exchanged between BGP speakers allows a router to construct a graph of AS connectivity
  - Routing loops can be pruned
  - Routing policy at AS level can be applied

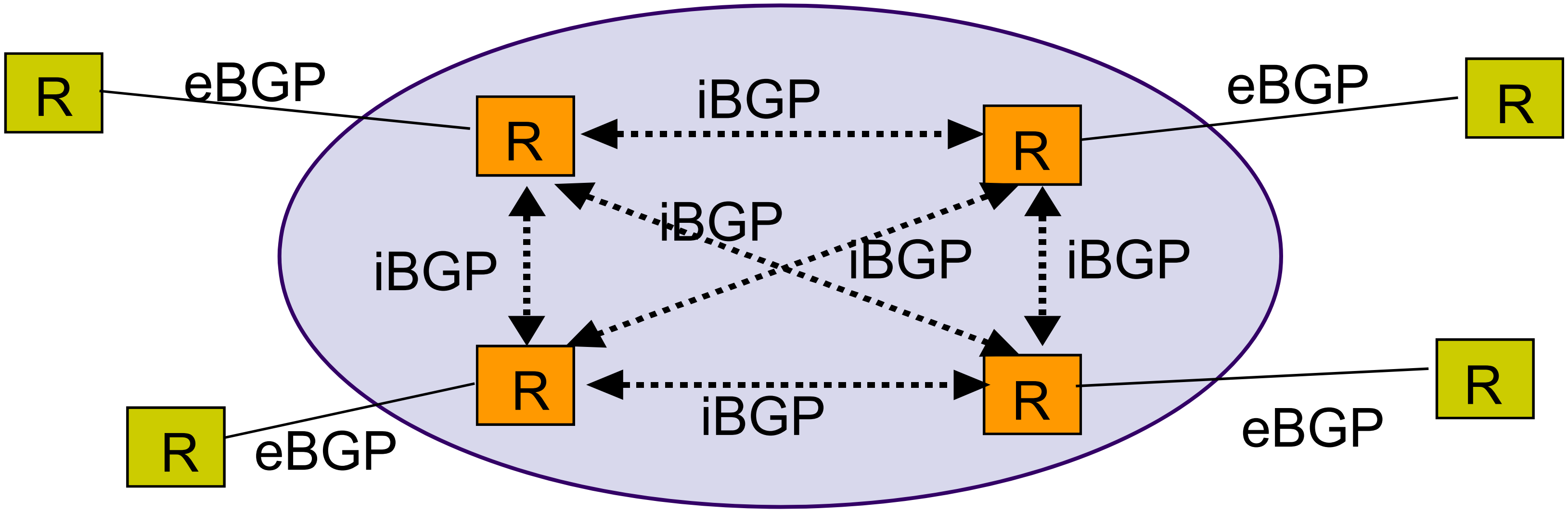
BGP Features

- BGP is *path vector protocol*: advertises sequence of AS numbers to the destination network
- Path vector info used to prevent routing loops
- BGP enforces policy through selection of different paths to a destination and by control of redistribution of routing information
- Uses CIDR to support aggregation & reduction of routing information

BGP Speaker & AS Relationship

- *BGP speaker*: a router running BGP
- *Peers or neighbors*: two speakers exchanging information on a connection
- BGP peers use TCP (port 179) to exchange messages
- Initially, BGP peers exchange entire BGP routing table
  - Incremental updates sent subsequently
  - Reduces bandwidth usage and processing overhead
  - Keepalive messages sent periodically (30 seconds)
- *Internal BGP* (iBPG) between BGP routers in same AS
- *External BGP* (eBGP) connections across AS borders

iBGP & eBGP



- eBGP to exchange reachability information in different AS's
  - eBGP peers directly connected
- iBGP to ensure net reachability info is consistent among the BGP speakers in the same AS
  - usually not directly connected
  - iBGP speakers exchange info learned from other iBGP speakers, and thus fully meshed

Path Selection

- Each BGP speaker
  - Evaluates paths to a destination from an AS border router
  - Selects the best that complies with policies
  - Advertises that route to all BGP neighbors
- BGP assigns a preference order to each path & selects path with highest value; BGP does not keep a cost metric to any path
- When multiple paths to a destination exist, BGP maintains all of the paths, but only advertises the one with highest preference value

BGP Policy

- Examples of policy:
  - Never use AS X
  - Never use AS X to get to a destination in AS Y
  - Never use AS X and AS Y in the same path
- *Import policies* to accept, deny, or set preferences on route advertisements from neighbors
- *Export policies* to determine which routes should be advertised to which neighbors
  - A route is advertised only if AS is willing to carry traffic on that route

BGP Protocol

- Opening & confirming of a BGP connection with a neighbor router
- Maintaining the BGP connection
- Sending reachability information
- Notification of error conditions

Attributes

Attribute Type	Attribute Length	Attribute Value

O	T	P	E	0	Attribute Type Code

Attribute Codes

ORIGIN: defines origin of NLRI

AS_PATH lists sequence of ASs that route has traversed to reach the destination

NEXT_HOP defines IP address of border router that should be used as the next hop to the destinations listed in the NLRI.

MULTI_EXIT_DISC: used to discriminate among multiple entry/exit points to neighboring AS and to hint about the preferred path.

LOCAL_PREF: informs other BGP speakers within the same AS of its degree of preference for an advertised route

ATOMIC_AGGREGATE: informs other BGP speakers that it selected a less specific route without selecting a more specific one which is included in it.

AGGREGATOR: specifies last AS number that formed the aggregate route followed by the IP address of the BGP speaker that formed the aggregate route