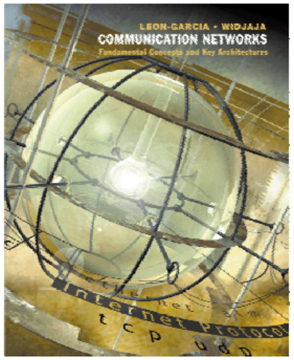
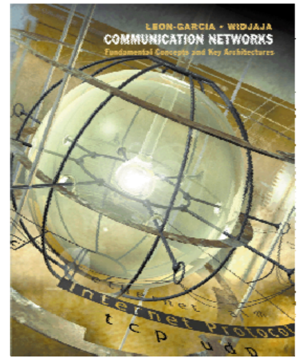
Chapter 8 Communication Networks and Services



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

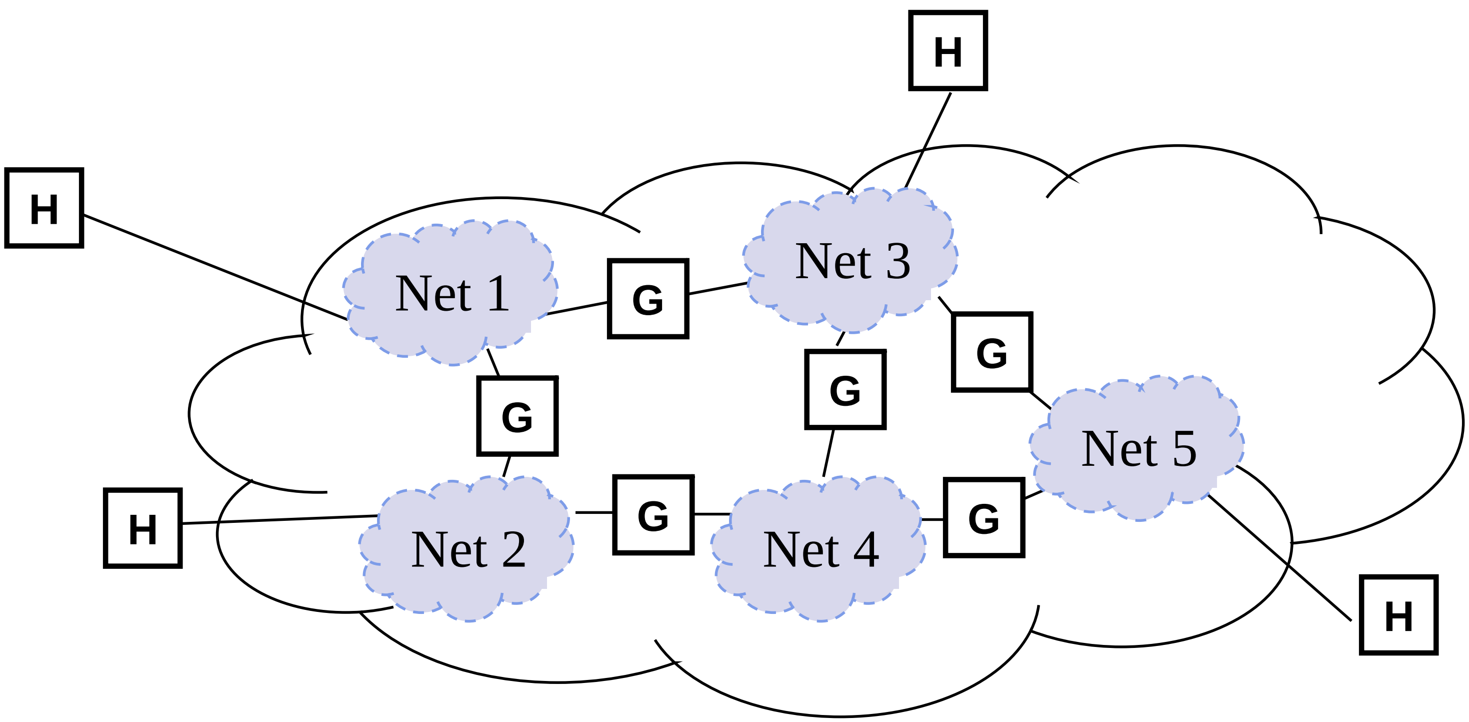
Chapter 8
Communication Networks and Services



The TCP/IP Architecture

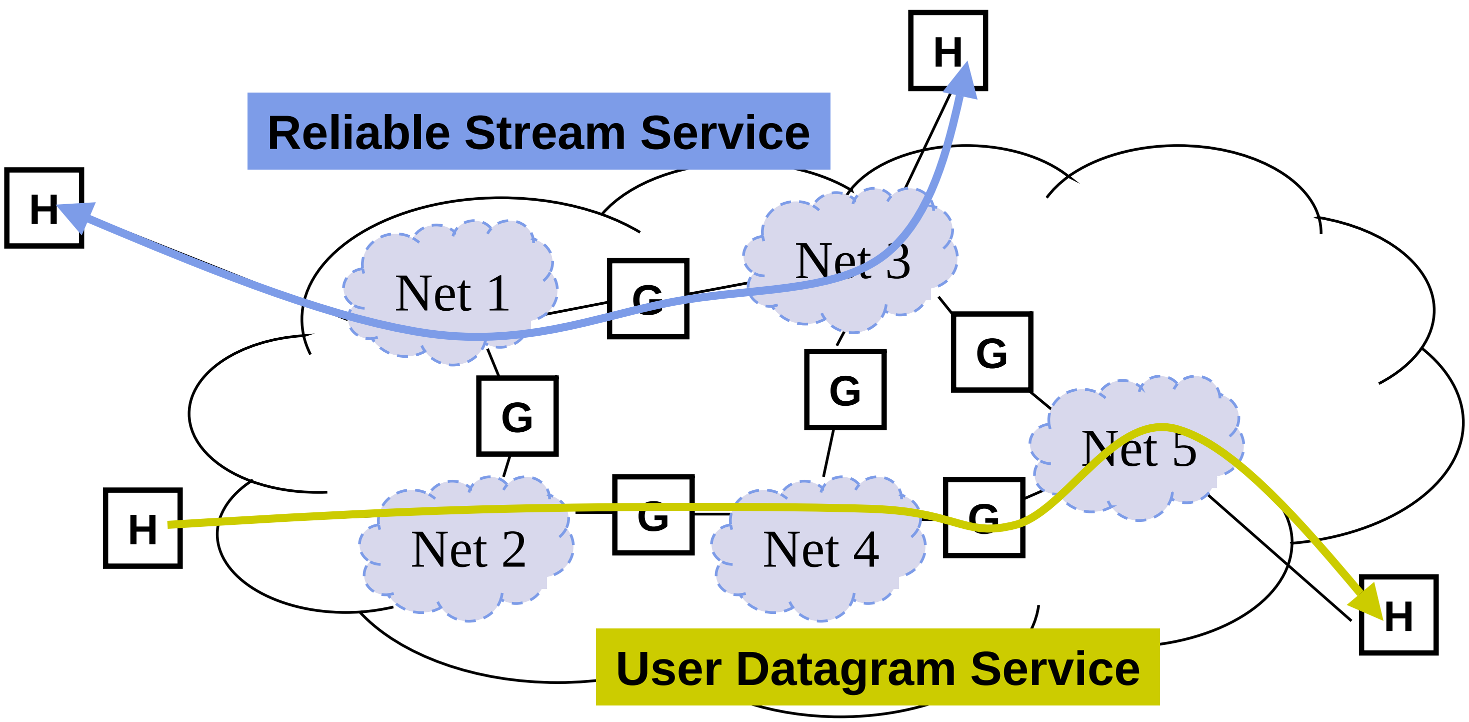
Why Internetworking?

- To build a "network of networks" or internet
  - operating over multiple, coexisting, different network technologies
  - providing ubiquitous connectivity through IP packet transfer
  - achieving huge economies of scale



Why Internetworking?

- To provide *universal communication services*
  - independent of underlying network technologies
  - providing common interface to user applications



Why Internetworking?

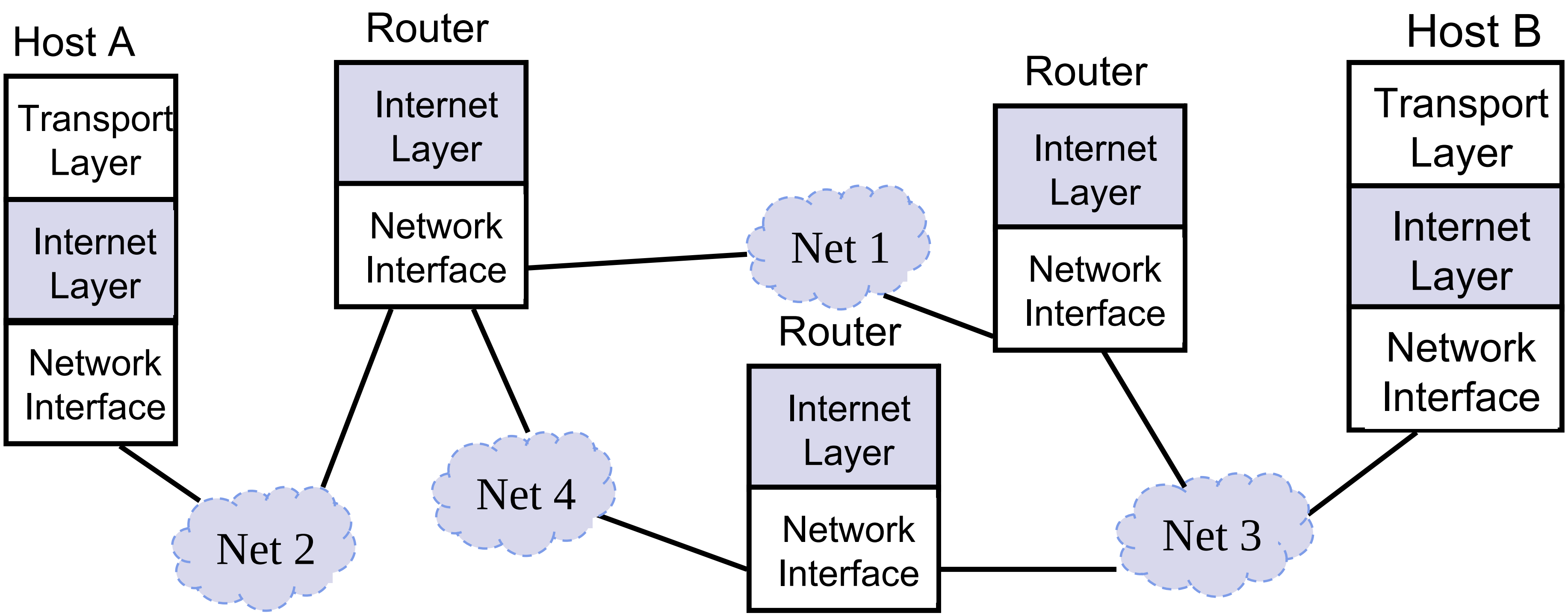
- To provide *distributed applications*
  - Any application designed to operate based on Internet communication services immediately operates across the entire Internet
  - Rapid deployment of new applications
    - Email, WWW, Peer-to-peer
  - Applications independent of network technology
    - New networks can be introduced below
    - Old network technologies can be retired

Internet Protocol Approach

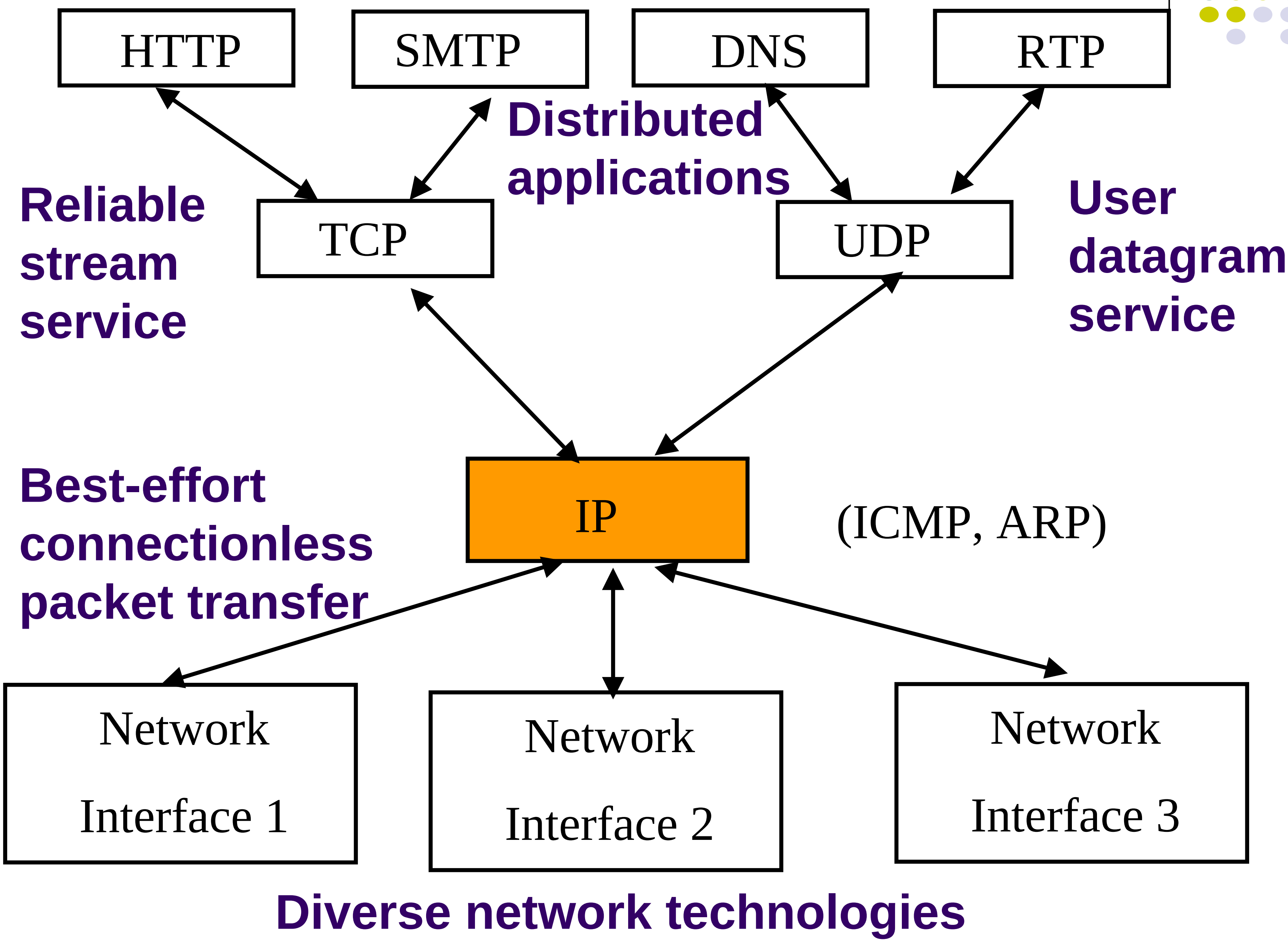
- IP packets transfer information across Internet

  Host A IP → router→ router…→ router→ Host B IP

- IP layer in each router determines next hop (router)
- Network interfaces transfer IP packets across networks



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

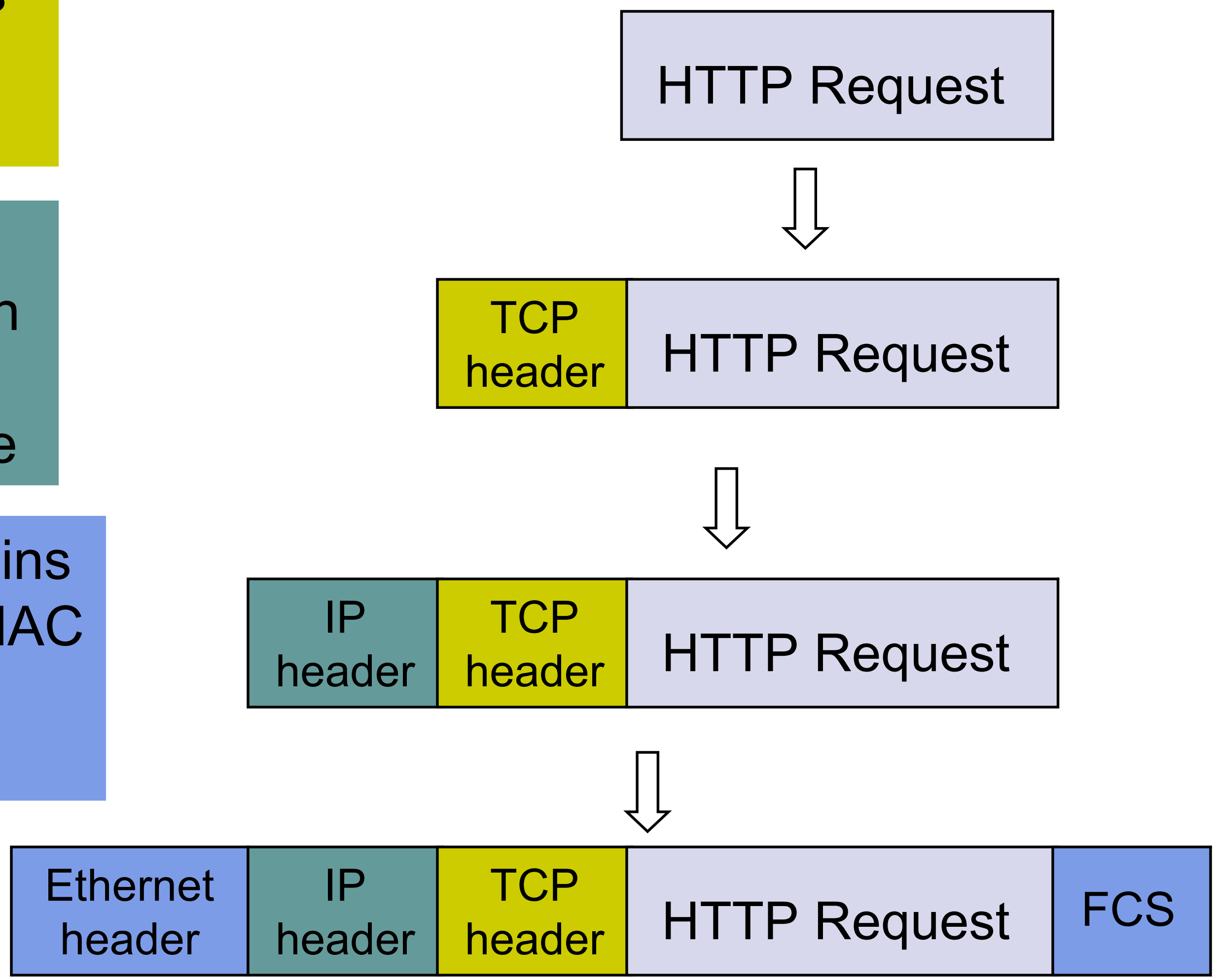
    Intel

Encapsulation

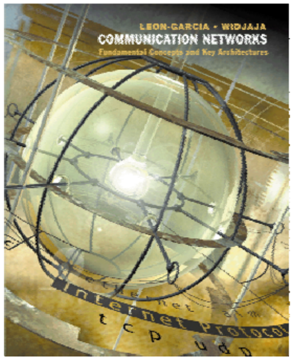
TCP Header contains source & destination port numbers

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

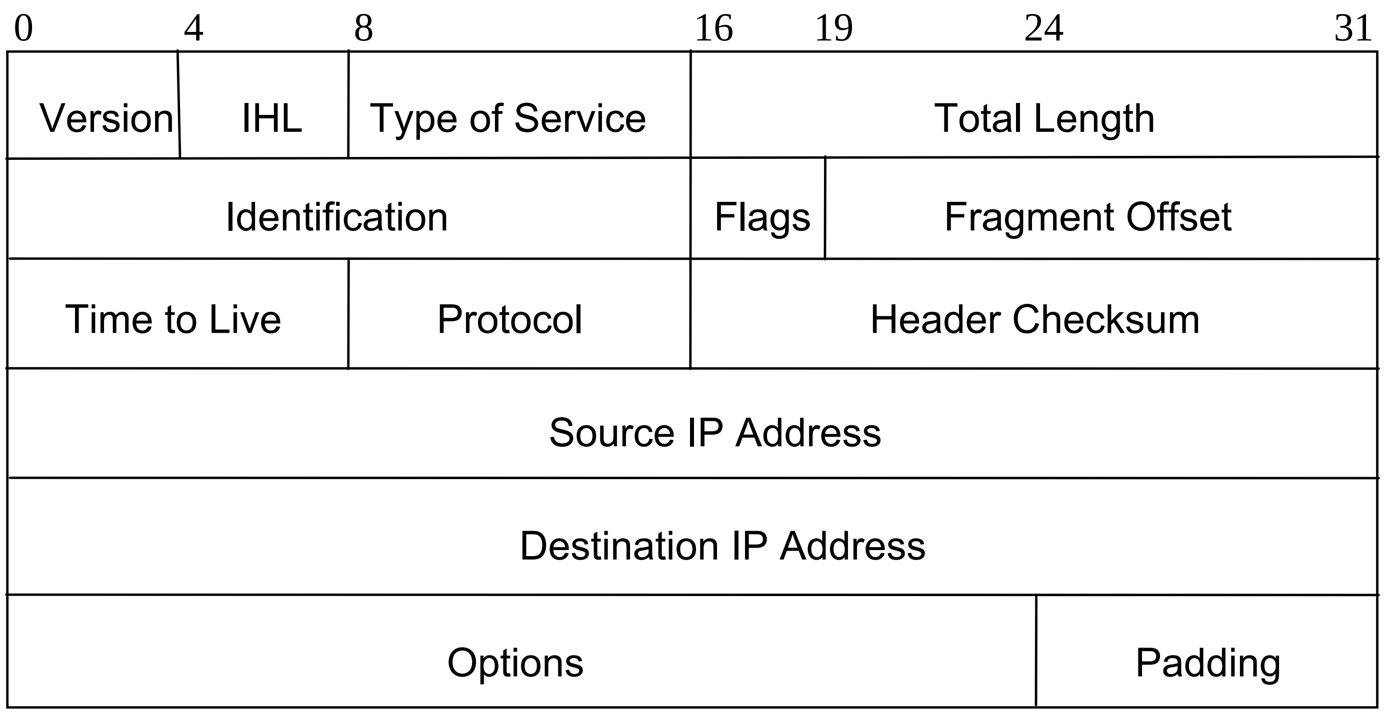


The Internet Protocol

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

<table>
<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>

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

<table>
<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>

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

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	

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

<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>

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.

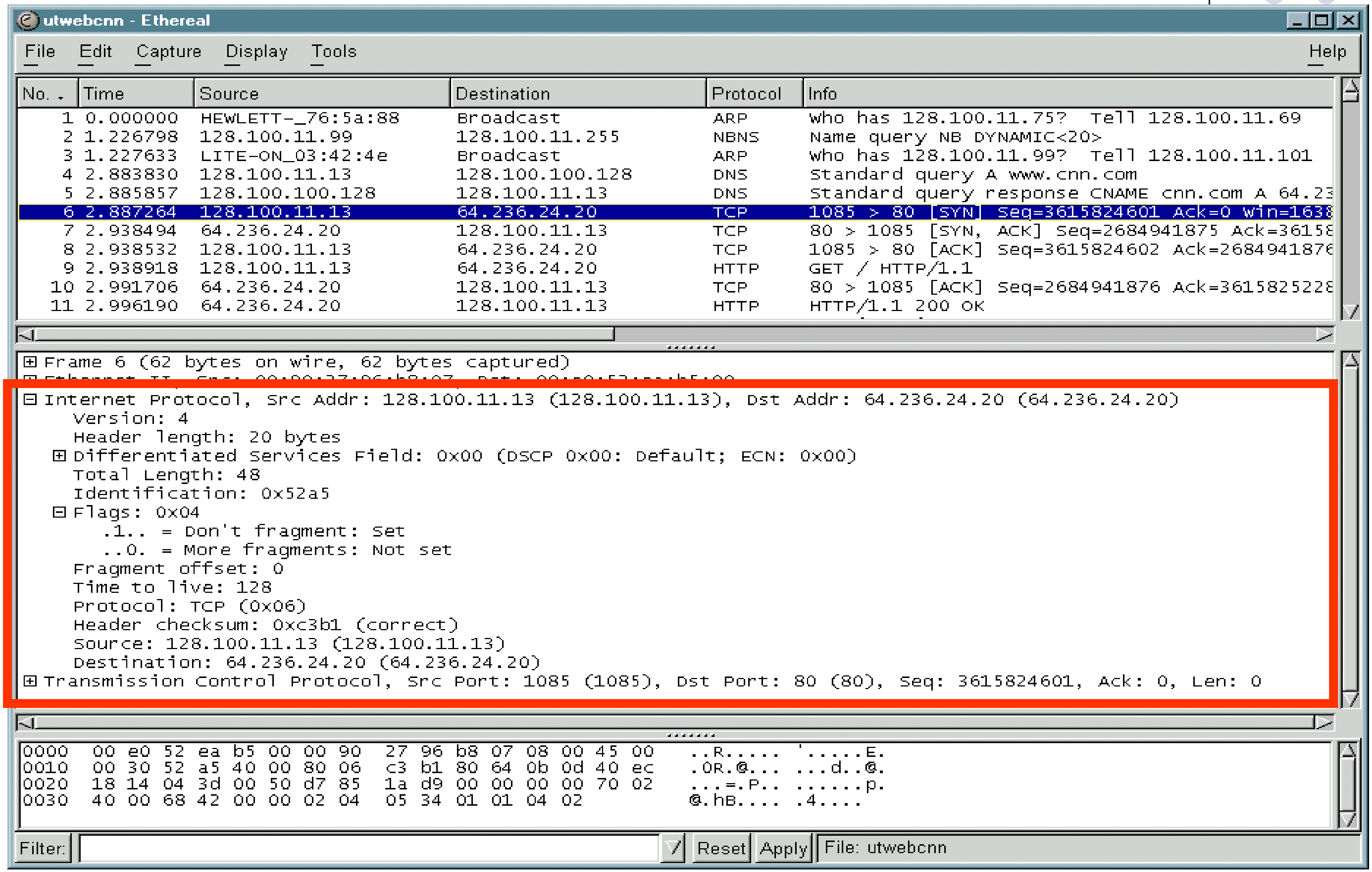
IP Packet Header

<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

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

- The checksum is then given by:

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

- 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***:

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

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

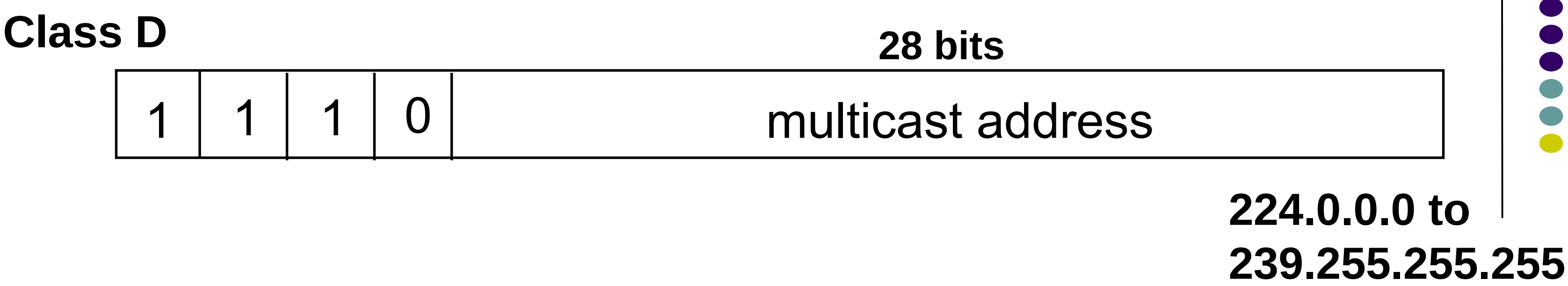
128.0.0.0 to 191.255.255.255

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



- 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)

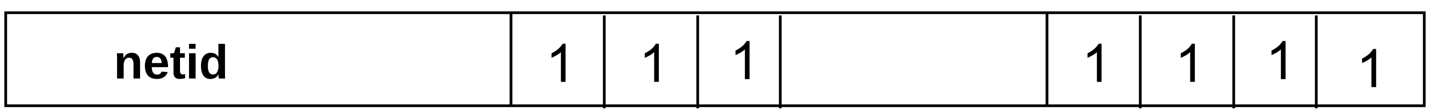
Internet address used to refer to network has hostid set to all 0s

this host
(used when
booting up)

a host
in this
network

Broadcast address has hostid set to all 1s

broadcast on
local network

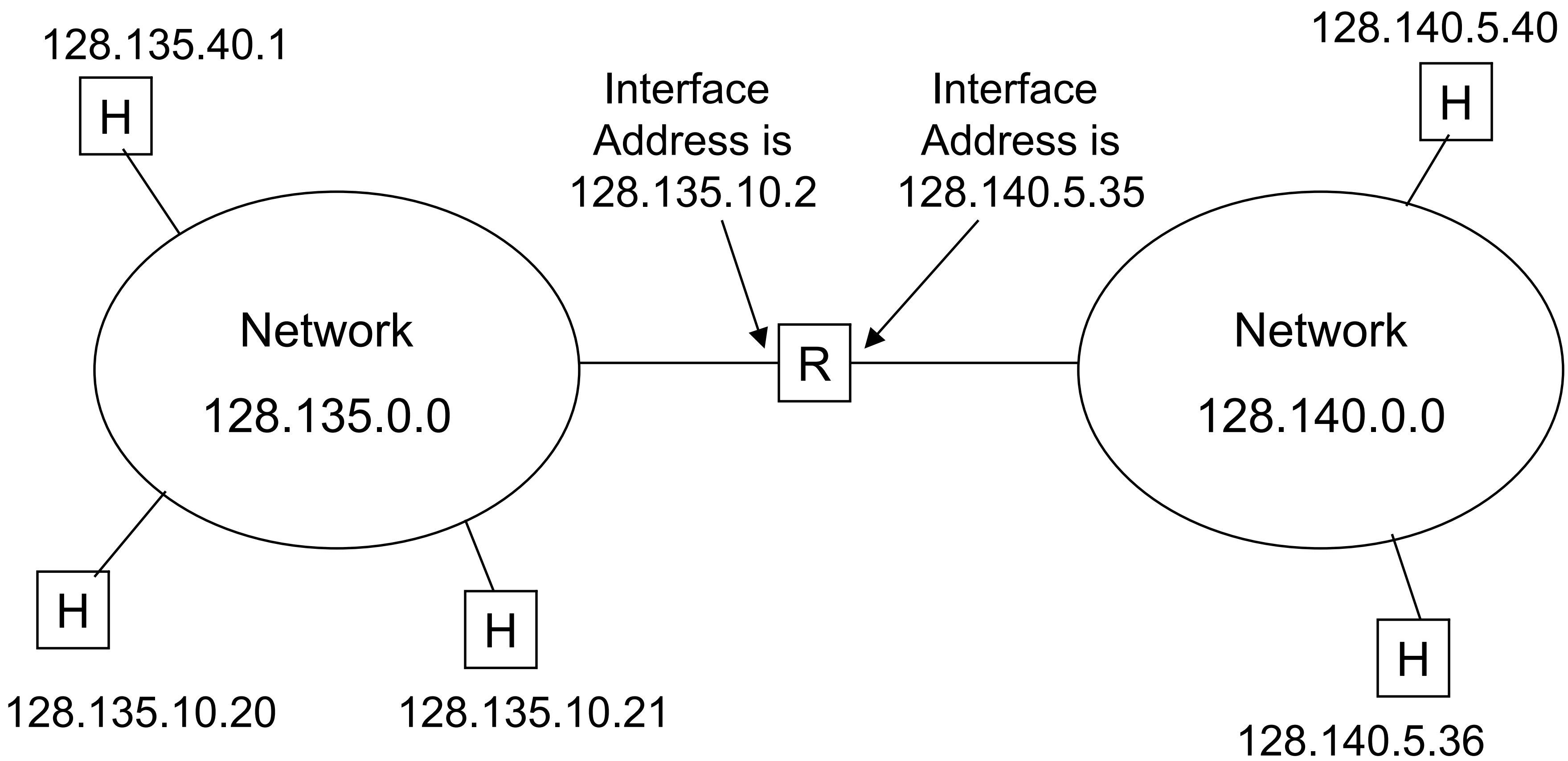


broadcast on
distant
network

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

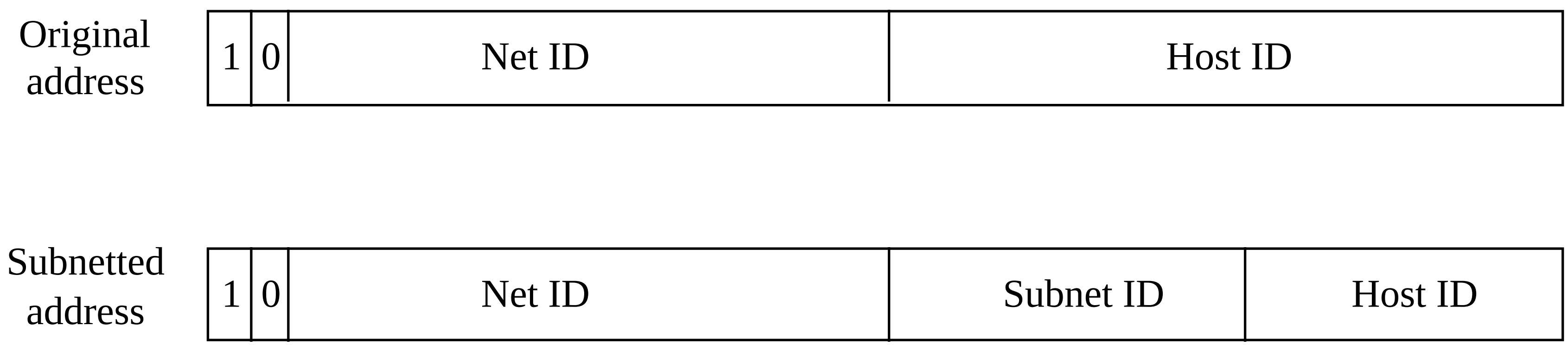
Address with host ID=all 1s refers to a broadcast packet

R = router

H = host

Subnet Addressing

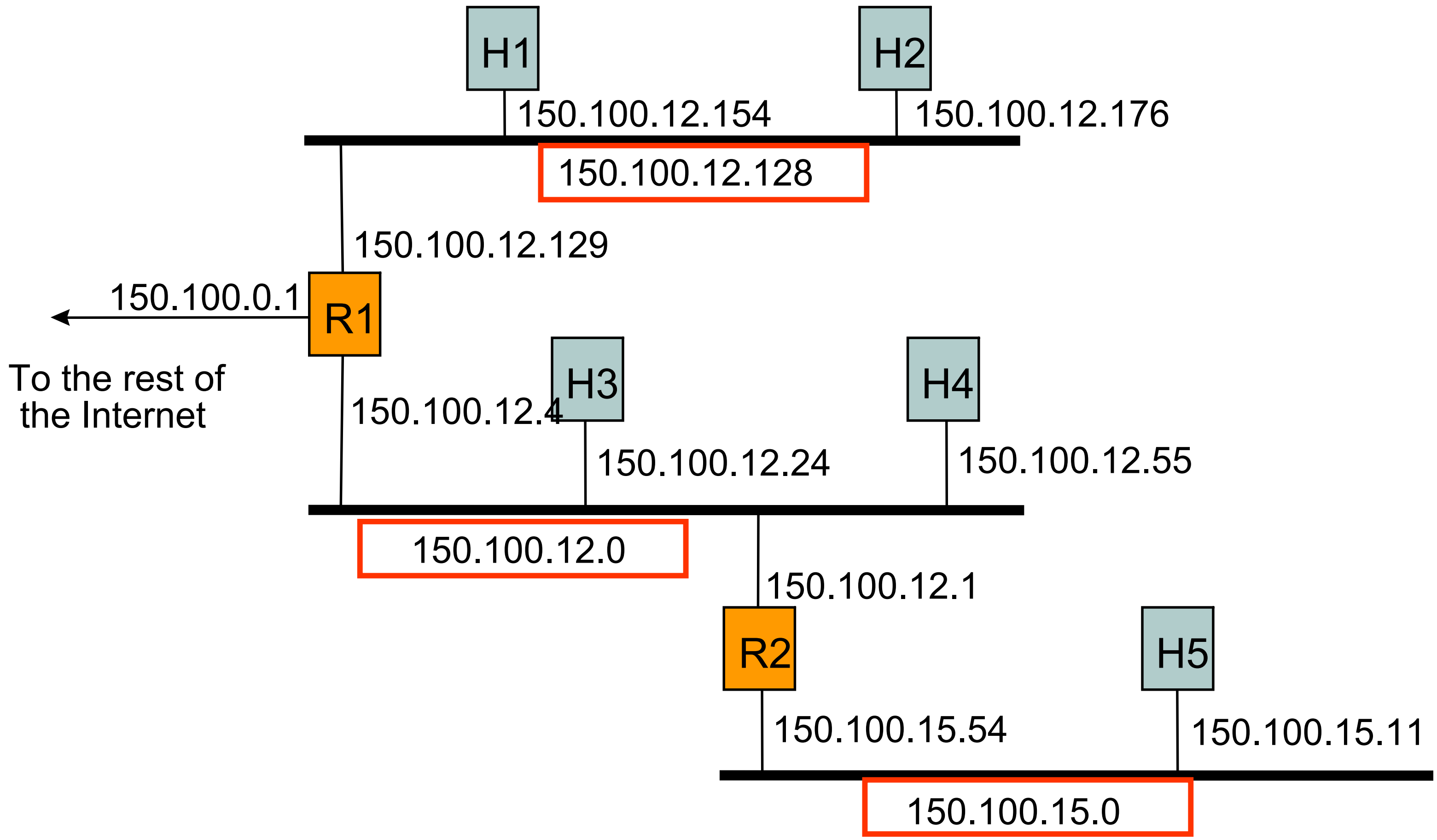
- Subnet addressing introduces another hierarchical level
- Transparent to remote networks
- Simplifies management of multiplicity of LANs
- Masking used to find subnet number



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

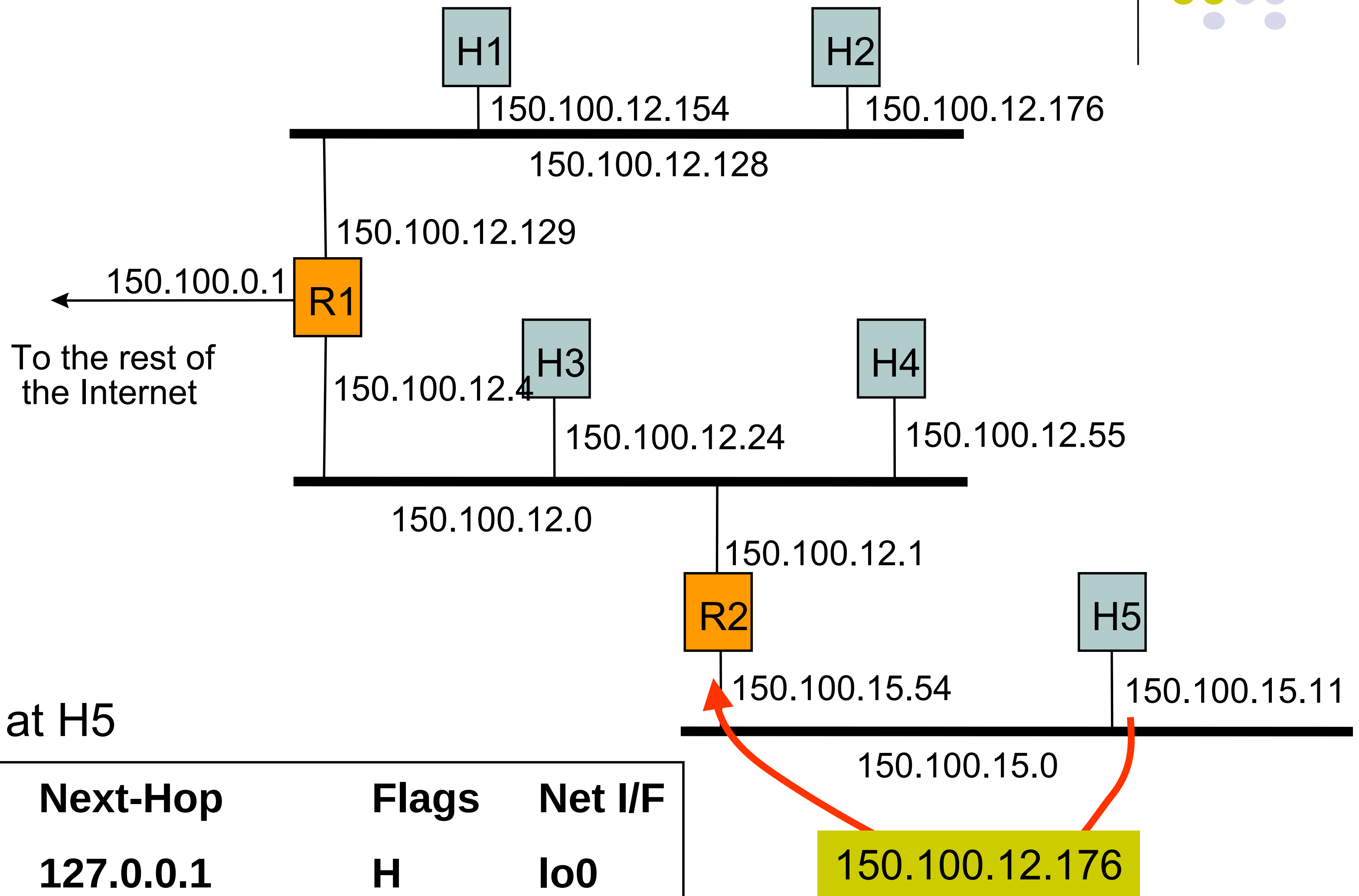
- IP layer in hosts and routers maintain a routing table
- Originating host: To send an IP packet, consult routing table
  - If destination host is in same network, send packet *directly* using appropriate network interface
  - Otherwise, send packet indirectly; typically, routing table indicates a default 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

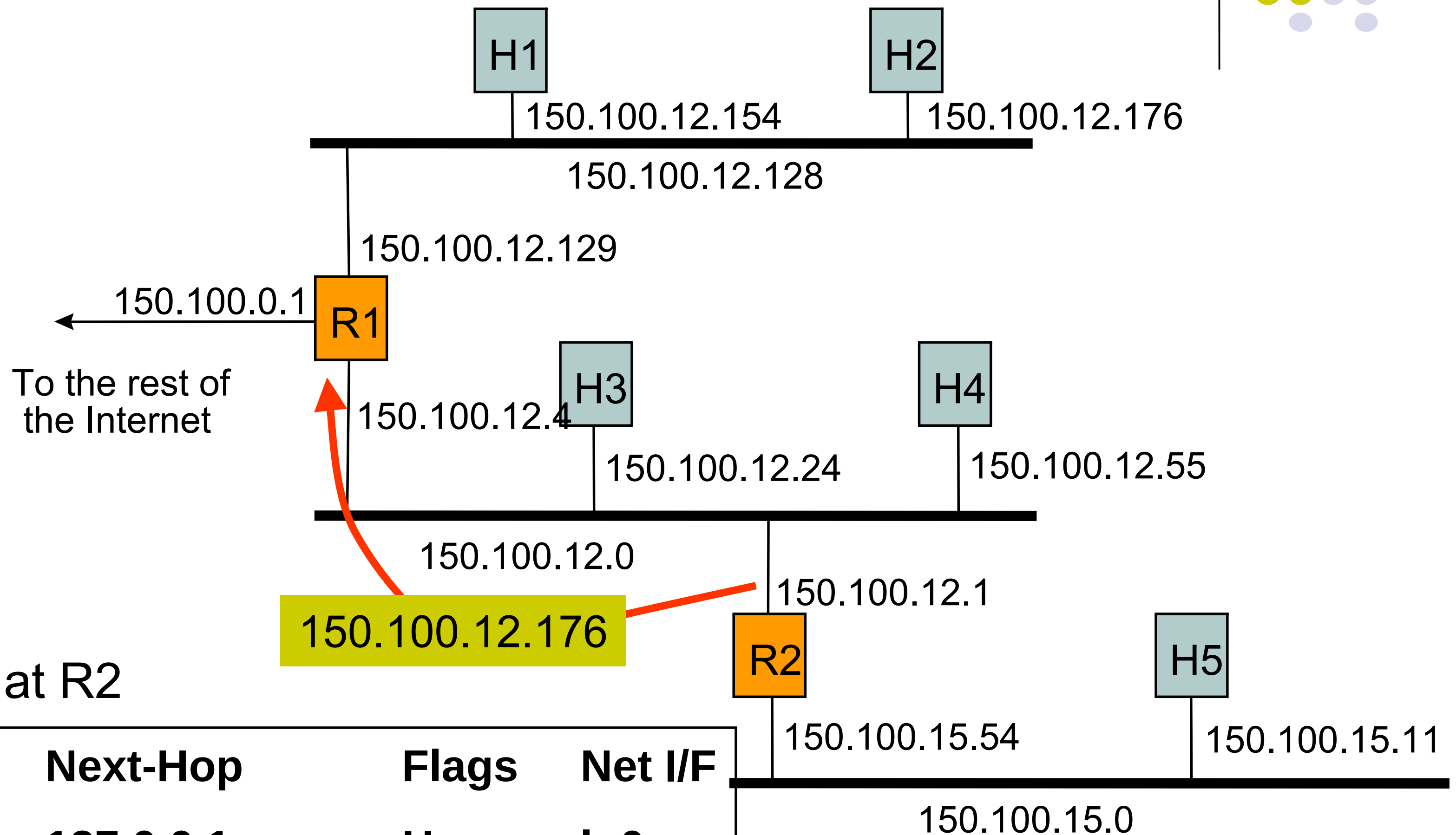
Example: Host H5 sends packet to host H2



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

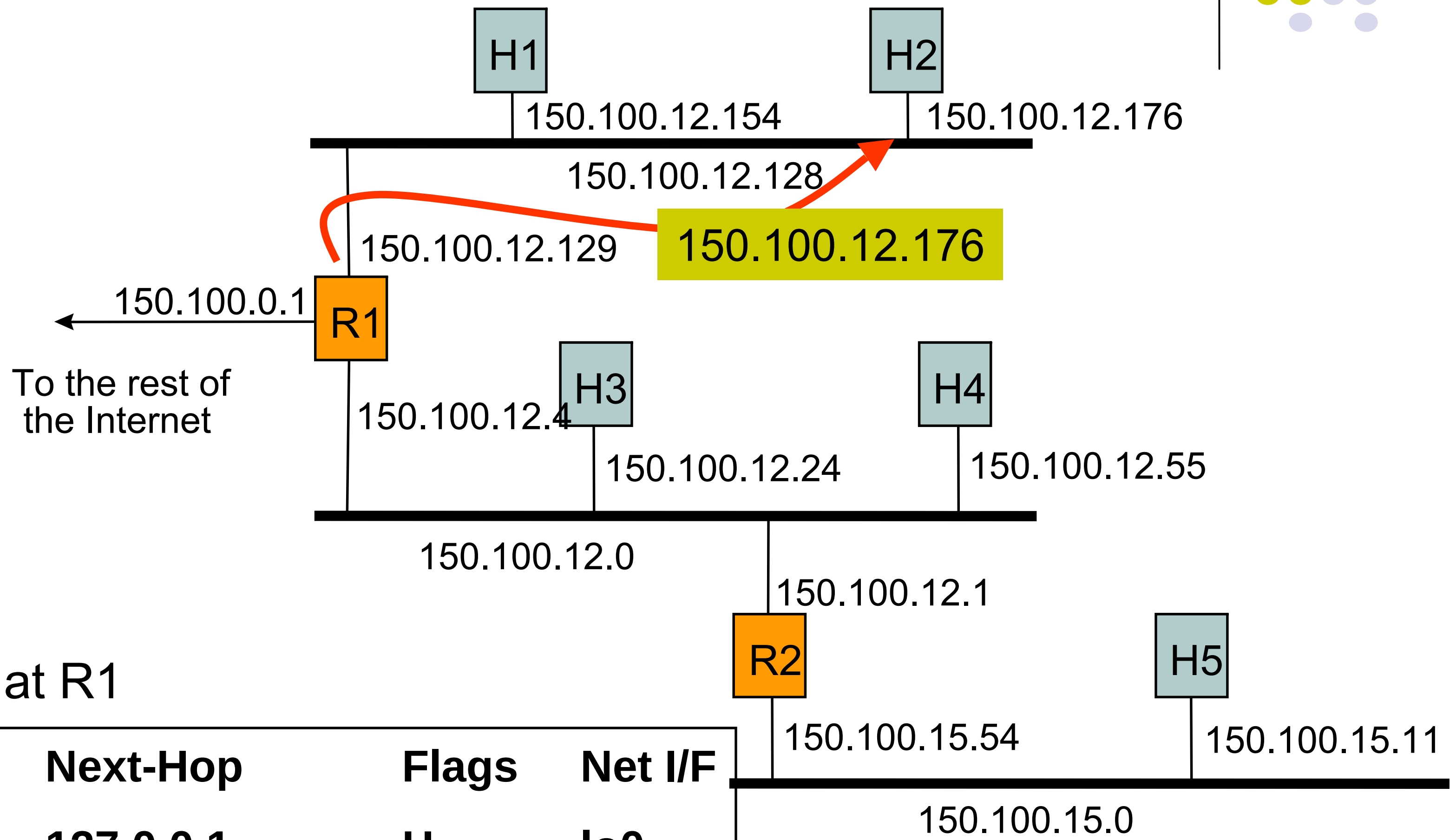
Example: Host H5 sends packet to host H2



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

New Address Allocation Policy

- Class A & B assigned only for clearly demonstrated need
- *Consecutive* blocks of class C assigned (up to 64 blocks)
  - All IP addresses in the range have a common **prefix**, and every address with that prefix is within the range
  - Arbitrary prefix length for network ID improves efficiency
- Lower half of class C space assigned to regional authorities
  - More hierarchical allocation of addresses
  - Service provider to customer

Address Requirement	Address Allocation
< 256	1 Class C
256<,<512	2 Class C
512<,<1024	4 Class C
1024<,<2048	8 Class C
2048<,<4096	16 Class C
4096<,<8192	32 Class C
8192<,<16384	64 Class C

Supernetting

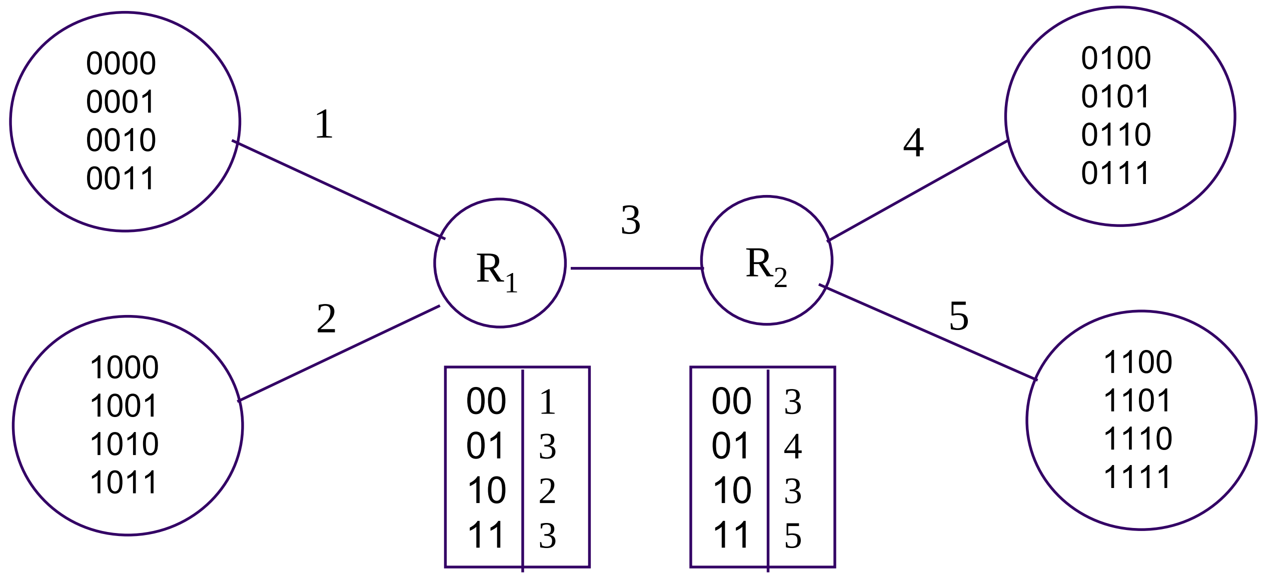
- Summarize a contiguous group of class C addresses using variable-length mask
- Example: 150.158.16.0/20
  - IP Address (150.158.16.0) & mask length (20)
  - IP add = 10010110 10011110 00010000 00000000
  - Mask = 11111111 11111111 11110000 00000000
  - Contains 16 Class C blocks:
  - From 10010110 10011110 00010000 00000000
  - i.e. 150.158.16.0
  - Up to 10010110 10011110 00011111 00000000
  - i.e. 150.158.31.0

Classless Inter-Domain Routing

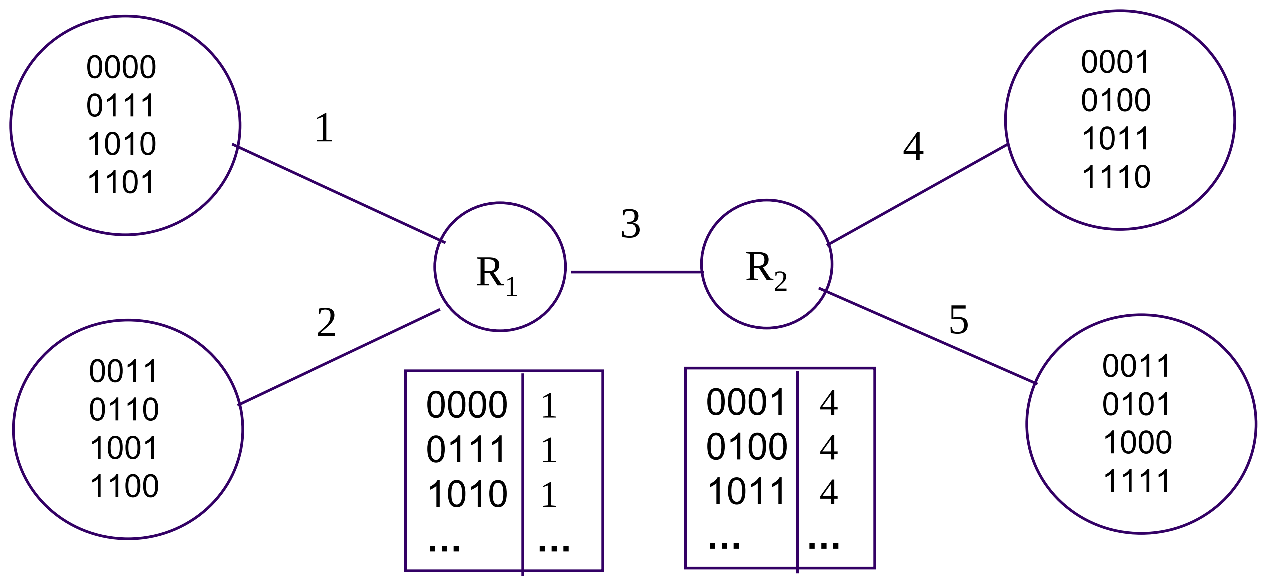
- CIDR deals with Routing Table Explosion Problem
  - Networks represented by prefix and mask
  - Pre-CIDR: Network with range of 16 contiguous class C blocks requires 16 entries
  - Post-CIDR: Network with range of 16 contiguous class C blocks requires 1 entry
- Solution: *Route according to prefix 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

Hierarchical Routing & Table Efficiency

(a)



(b)



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)

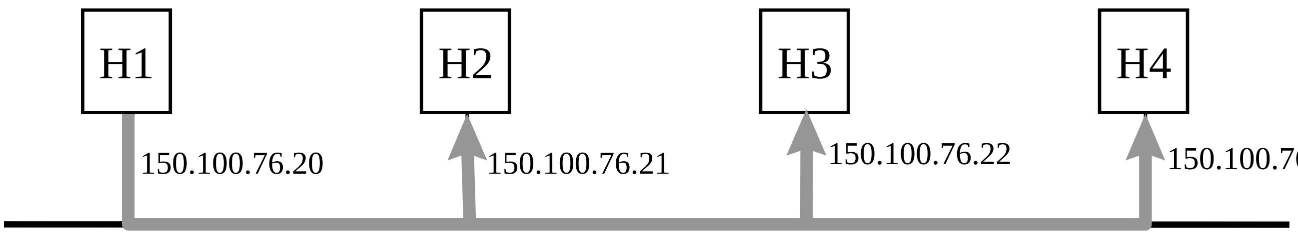
Longest Prefix Match

- 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

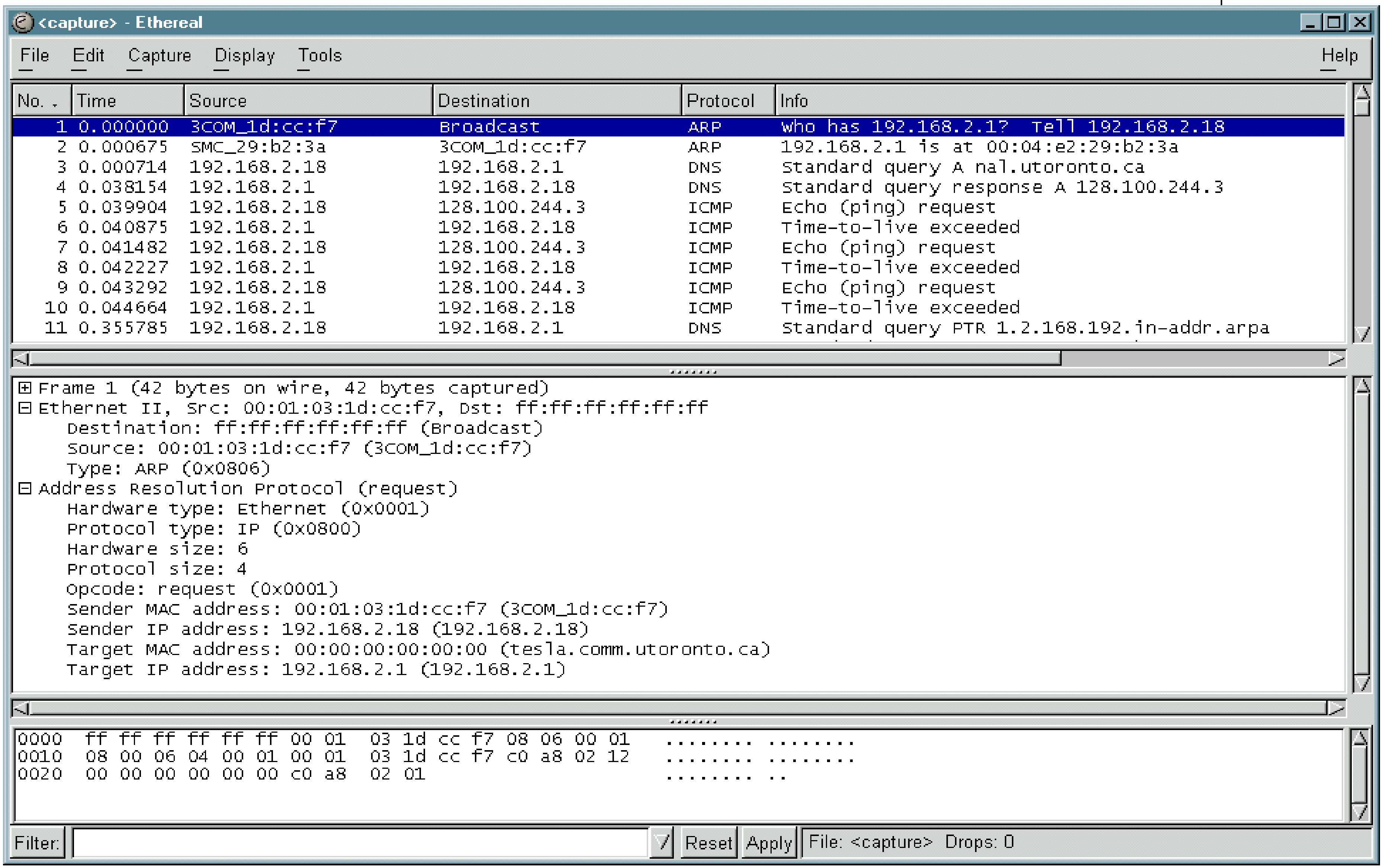


ARP request (what is the MAC address of 150.100.76.22?)

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

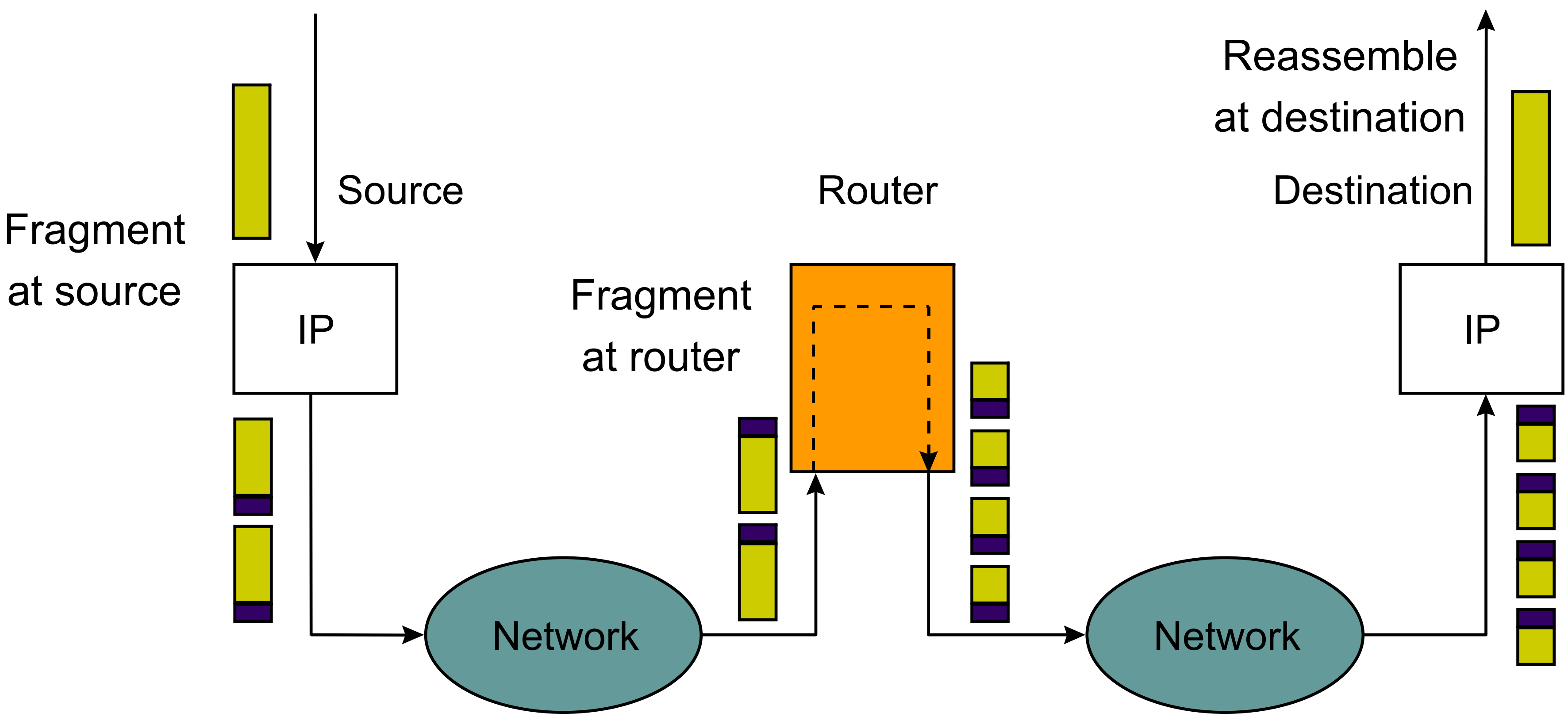
ARP response (my MAC address is 08:00:5a:3b:94)

Example of ARP



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)

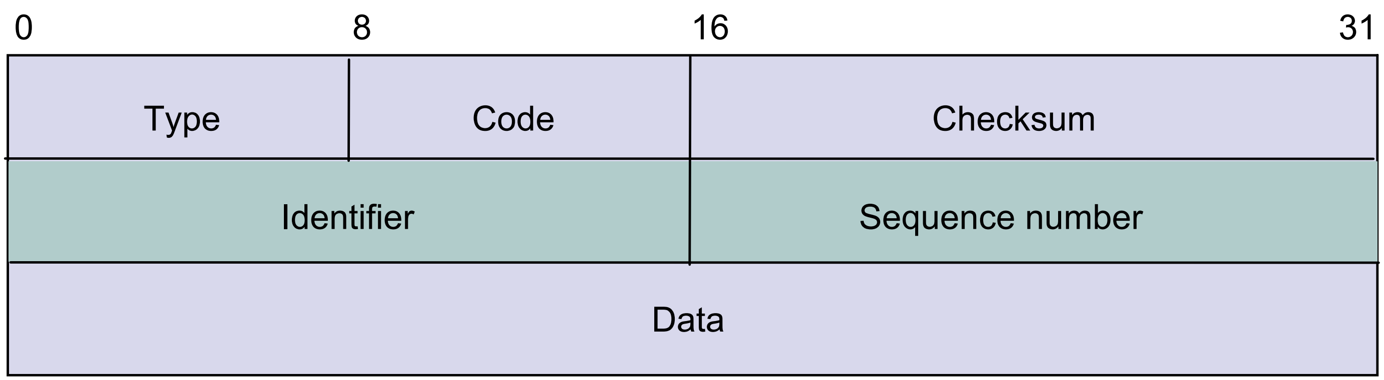
- RFC 792; Encapsulated in IP packet (protocl type = 1)
- Handles error and control messages
- If router cannot deliver or forward a packet, it sends an ICMP “host unreachable” message to the source
- If router receives packet that should have been sent to another router, it sends an ICMP “redirect” 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			

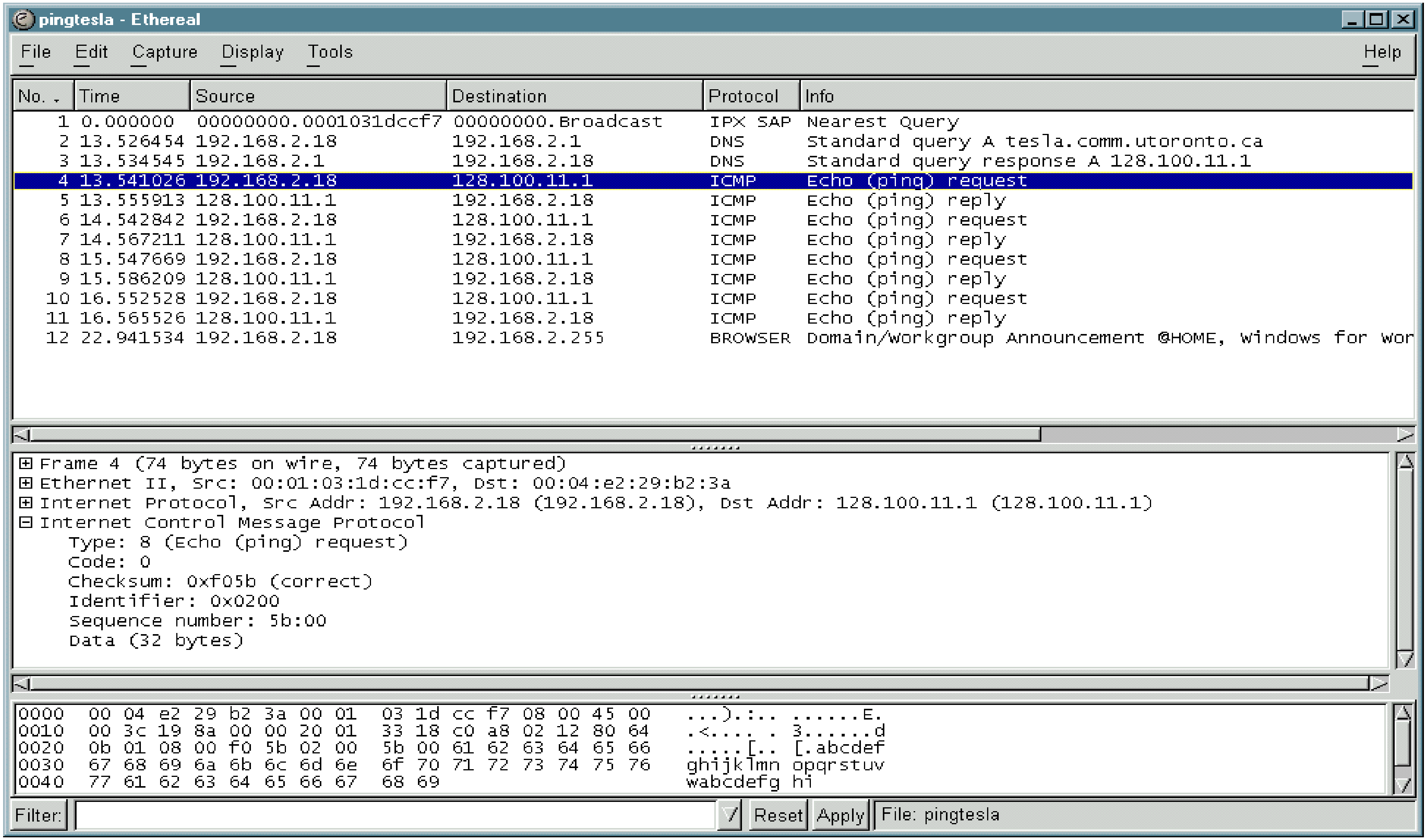
- *Type* of message: some examples
  - 0 Network Unreachable; 3 Port Unreachable
  - 1 Host Unreachable 4 Fragmentation needed
  - 2 Protocol Unreachable 5 Source route failed
  - 11 Time-exceeded, code=0 if TTL exceeded
- Code: purpose of message
- IP header & 64 bits of original datagram
  - To match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

Echo Request & Echo Reply Message Format

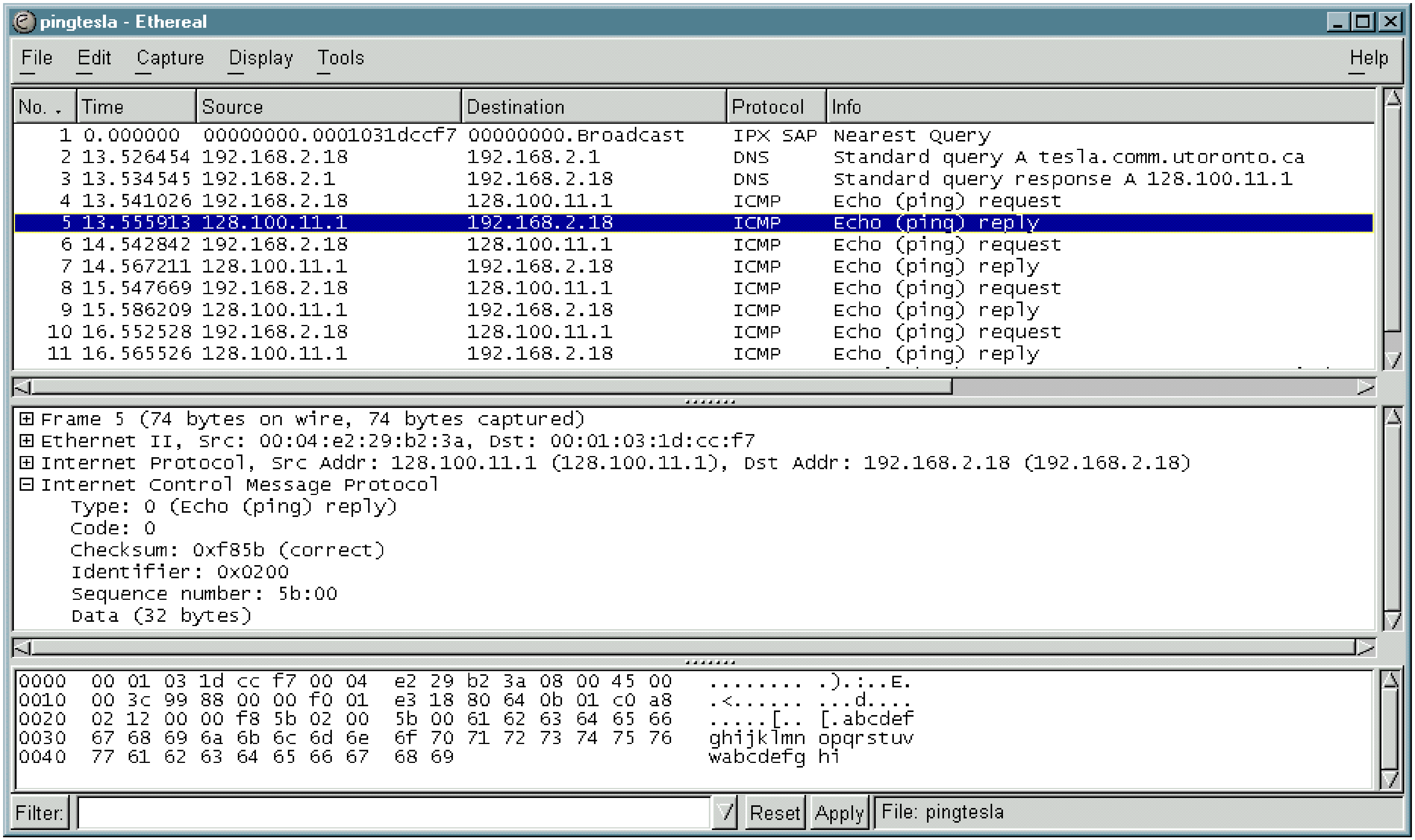


- Echo request: type=8; Echo reply: type=0
  - Destination replies with echo reply by copying data in request onto reply message
- Sequence number to match reply to request
- ID to distinguish between different sessions using echo services
- Used in PING

Example – Echo request

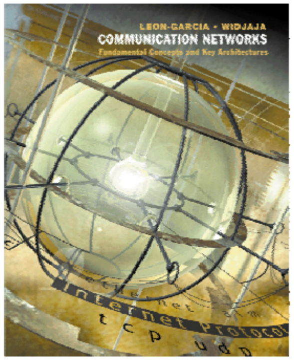


Example – Echo Reply



Chapter 8
Communication Networks and Services

IPv6



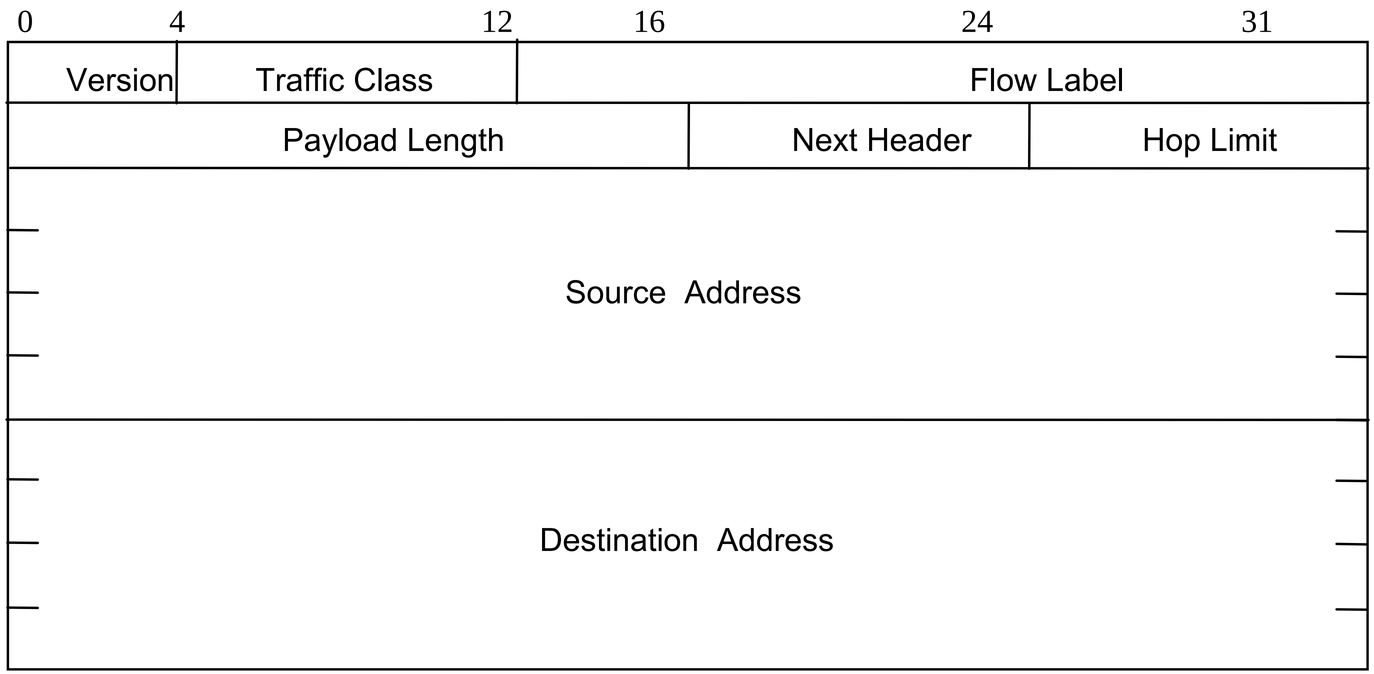
IPv6

- **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

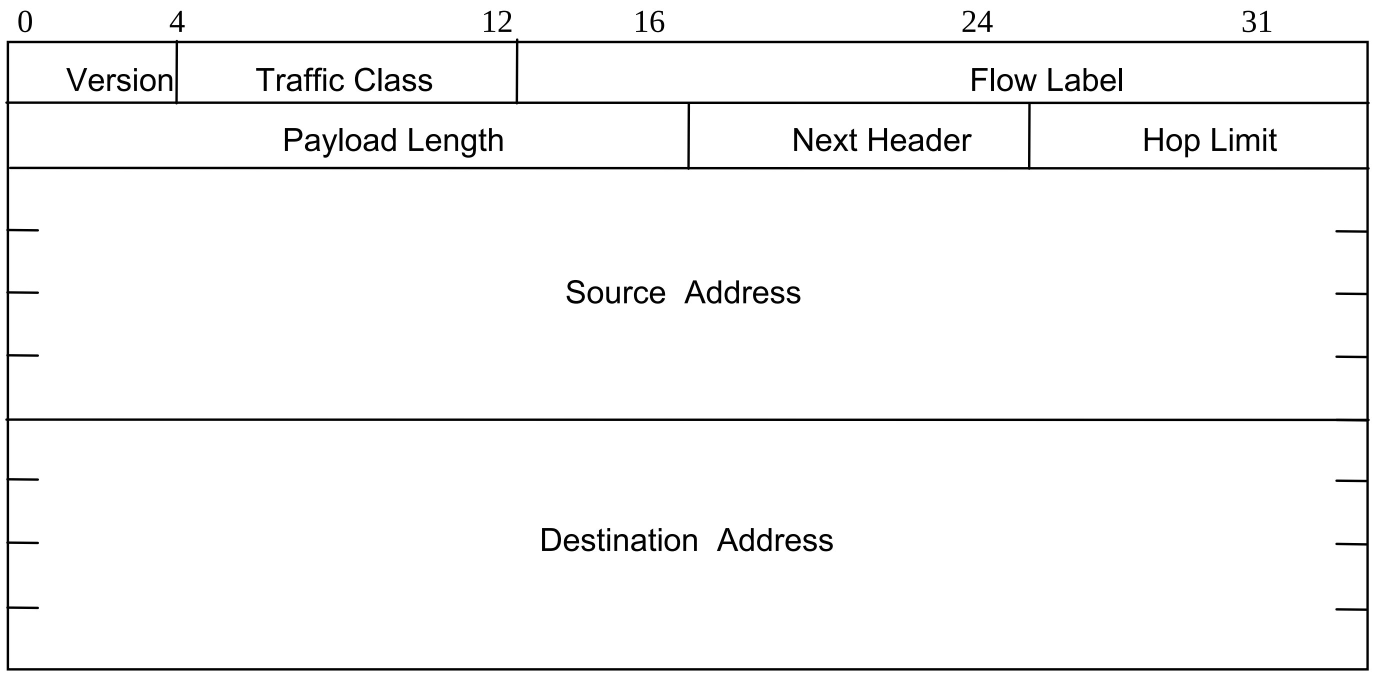
- **Flexible support for options:** more efficient and flexible options encoded in optional *extension headers*
- **Flow label capability:** “flow label” to identify a packet flow that requires a certain QoS
- **Security:** built-in authentication and confidentiality
- **Large packets:** supports payloads 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

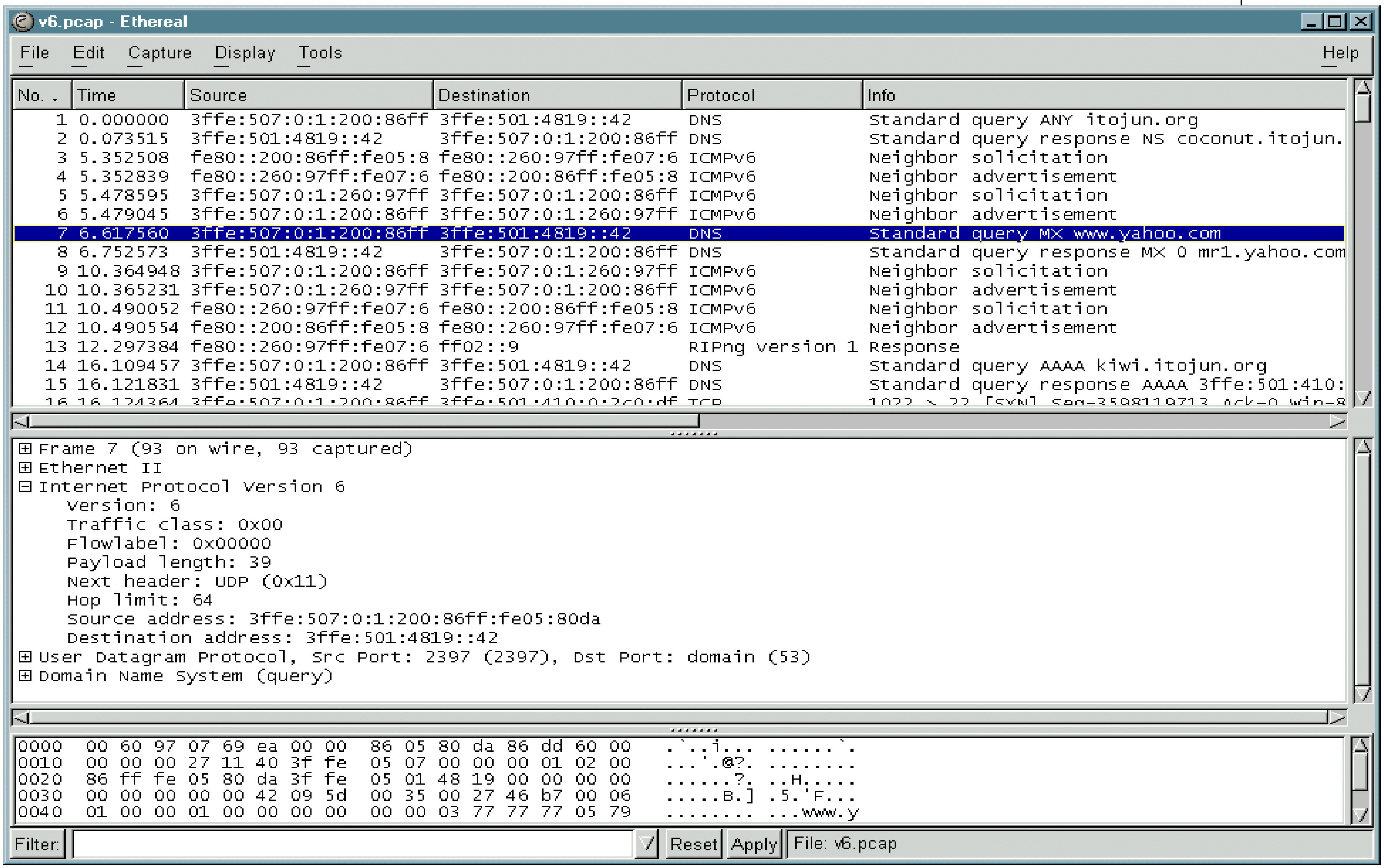


- Payload length: length of data excluding header, up to 65535 B
- Next header: type of extension header that follows basic header
- Hop limit: # hops packet can travel before being dropped by a router

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

Example


v6.pcap - Ethereal
File Edit Capture Display Tools Help
No. Time Source Destination Protocol Info
1 0.000000 3ffe:507:0:1:200:86ff 3ffe:501:4819::42 DNS Standard query ANY itojun.org
2 0.073515 3ffe:501:4819::42 3ffe:507:0:1:200:86ff DNS Standard query response NS coconut.itojun.
3 5.352508 fe80::200:86ff:fe05:8 fe80::260:97ff:fe07:6 ICMPv6 Neighbor solicitation
4 5.352839 fe80::260:97ff:fe07:6 fe80::200:86ff:fe05:8 ICMPv6 Neighbor advertisement
5 5.478595 3ffe:507:0:1:260:97ff 3ffe:507:0:1:200:86ff ICMPv6 Neighbor solicitation
6 5.479045 3ffe:507:0:1:200:86ff 3ffe:507:0:1:260:97ff ICMPv6 Neighbor advertisement
7 6.617560 3ffe:507:0:1:200:86ff 3ffe:501:4819::42 DNS Standard query MX www.yahoo.com
8 6.752573 3ffe:501:4819::42 3ffe:507:0:1:200:86ff DNS Standard query response MX 0 mr1.yahoo.com
9 10.364948 3ffe:507:0:1:200:86ff 3ffe:507:0:1:260:97ff ICMPv6 Neighbor solicitation
10 10.365231 3ffe:507:0:1:260:97ff 3ffe:507:0:1:200:86ff ICMPv6 Neighbor advertisement
11 10.490052 fe80::260:97ff:fe07:6 fe80::200:86ff:fe05:8 ICMPv6 Neighbor solicitation
12 10.490554 fe80::200:86ff:fe05:8 fe80::260:97ff:fe07:6 ICMPv6 Neighbor advertisement
13 12.297384 fe80::260:97ff:fe07:6 ff02::9 RIPng version 1 Response
14 16.109457 3ffe:507:0:1:200:86ff 3ffe:501:4819::42 DNS Standard query AAAA kiwi.itojun.org
15 16.121831 3ffe:501:4819::42 3ffe:507:0:1:200:86ff DNS Standard query response AAAA 3ffe:501:410:
Frame 7 (93 on wire, 93 captured)
Ethernet II
Internet Protocol Version 6
Version: 6
Traffic class: 0x00
Flowlabel: 0x00000
Payload length: 39
Next header: UDP (0x11)
Hop limit: 64
Source address: 3ffe:507:0:1:200:86ff:fe05:80da
Destination address: 3ffe:501:4819::42
User Datagram Protocol, Src Port: 2397 (2397), Dst Port: domain (53)
Domain Name System (query)
0000 00 60 97 07 69 ea 00 00 86 05 80 da 86 dd 60 00
0010 00 00 00 27 11 40 3f fe 05 07 00 00 00 01 02 00
0020 86 ff fe 05 80 da 3f fe 05 01 48 19 00 00 00 00
0030 00 00 00 00 00 42 09 5d 00 35 00 27 46 b7 00 06
0040 01 00 00 01 00 00 00 00 00 00 03 77 77 77 05 79
Filter:
Reset
Apply
File: v6.pcap

Address Types based on Prefixes

Binary prefix	Types	Percentage of address space
0000 0000	Reserved	0.39
0000 0001	Unassigned	0.39
0000 001	ISO network addresses	0.78
0000 010	IPX network addresses	0.78
0000 011	Unassigned	0.78
0000 1	Unassigned	3.12
0001	Unassigned	6.25
001	Unassigned	12.5
010	Provider-based unicast addresses	12.5
011	Unassigned	12.5
100	Geographic-based unicast addresses	12.5
101	Unassigned	12.5
110	Unassigned	12.5
1110	Unassigned	6.25
1111 0	Unassigned	3.12
1111 10	Unassigned	1.56
1111 110	Unassigned	0.78
1111 1110 0	Unassigned	0.2
1111 1110 10	Link local use addresses	0.098
1111 1110 11	Site local use addresses	0.098
1111 1111	Multicast addresses	0.39

Special Purpose Addresses

	n bits	m bits	o bits	p bits	(125-m-n-o-p) bits
010	Registry ID	Provider ID	Subscriber ID	Subnet ID	Interface ID

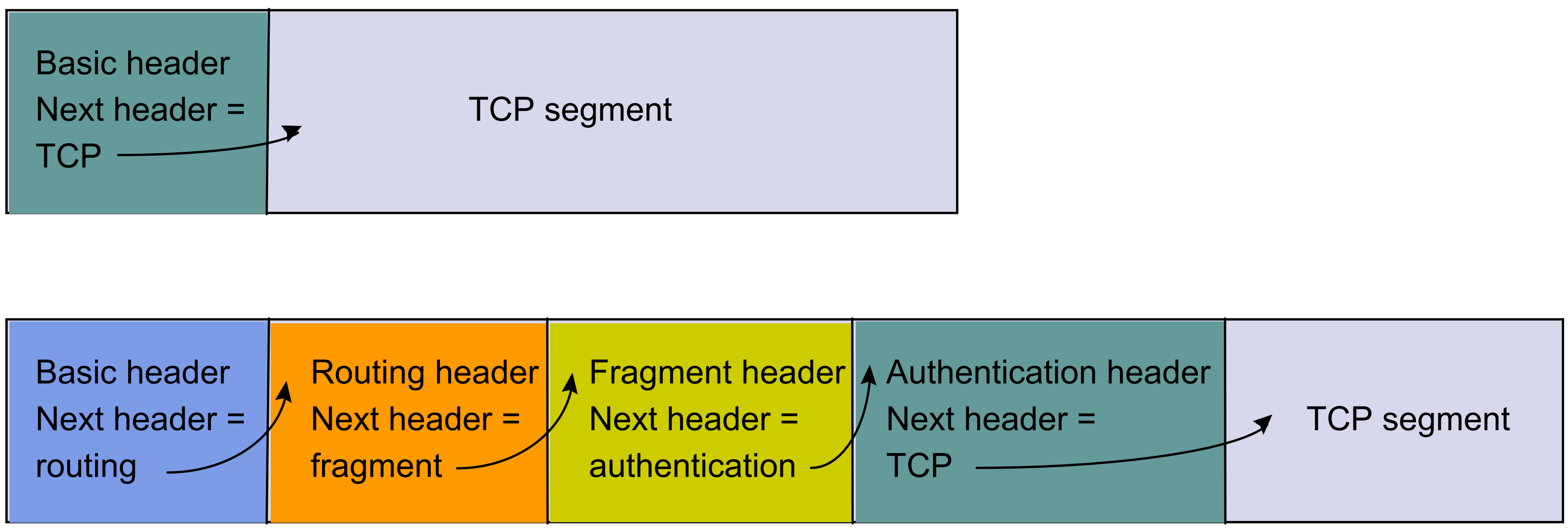
- *Provider-based Addresses*: 010 prefix
  - Assigned by providers to their customers
  - Hierarchical structure promotes aggregation
    - Registry ID: ARIN, RIPE, APNIC
    - ISP
    - Subscriber ID: subnet ID & interface ID
- *Local Addresses*: do not connect to global Internet
  - Link-local: for single link
  - Site-local: for single site
  - Designed to facilitate transition to connection to Internet

Special Purpose Addresses

- *Unspecified Address*: 0::0
  - Used by source station to learn own address
- *Loopback Address*: ::1
- *IPv4-compatible addresses*: 96 0's + IPv4
  - For tunneling by IPv6 routers connected to IPv4 networks
  - ::135.150.10.247
- *IP-mapped addresses*: 80 0's + 16 1's + IPv4
  - Denote IPv4 hosts & routers that do not support IPv6

Extension Headers

Daisy chains of extension headers



- Extension headers processed in order of appearance

Six Extension Headers

Header code	Header type
0	Hop-by-hop options header
43	Routing header
44	Fragment header
51	Authentication header
52	Encapsulating security payload header
60	Destination options header

Extension Headers

- Large Packet: payload>64K

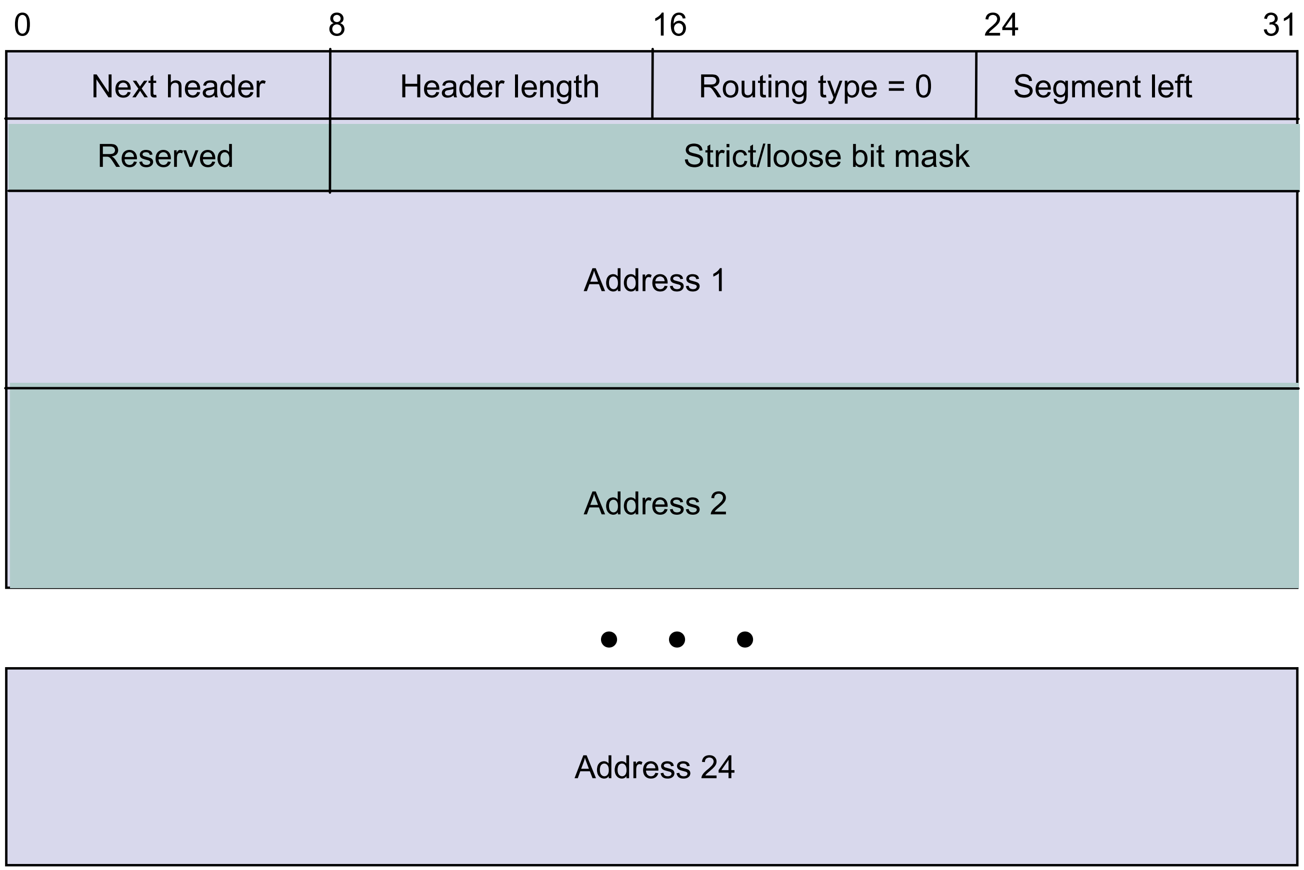
0	8	16	24	31
Next header	0	194	Opt len = 4	
Jumbo payload length				

- Fragmentation: At source only

0	8	16	29	31
Next header	Reserved	Fragment offset	Res	M
Identification				

Extension Headers

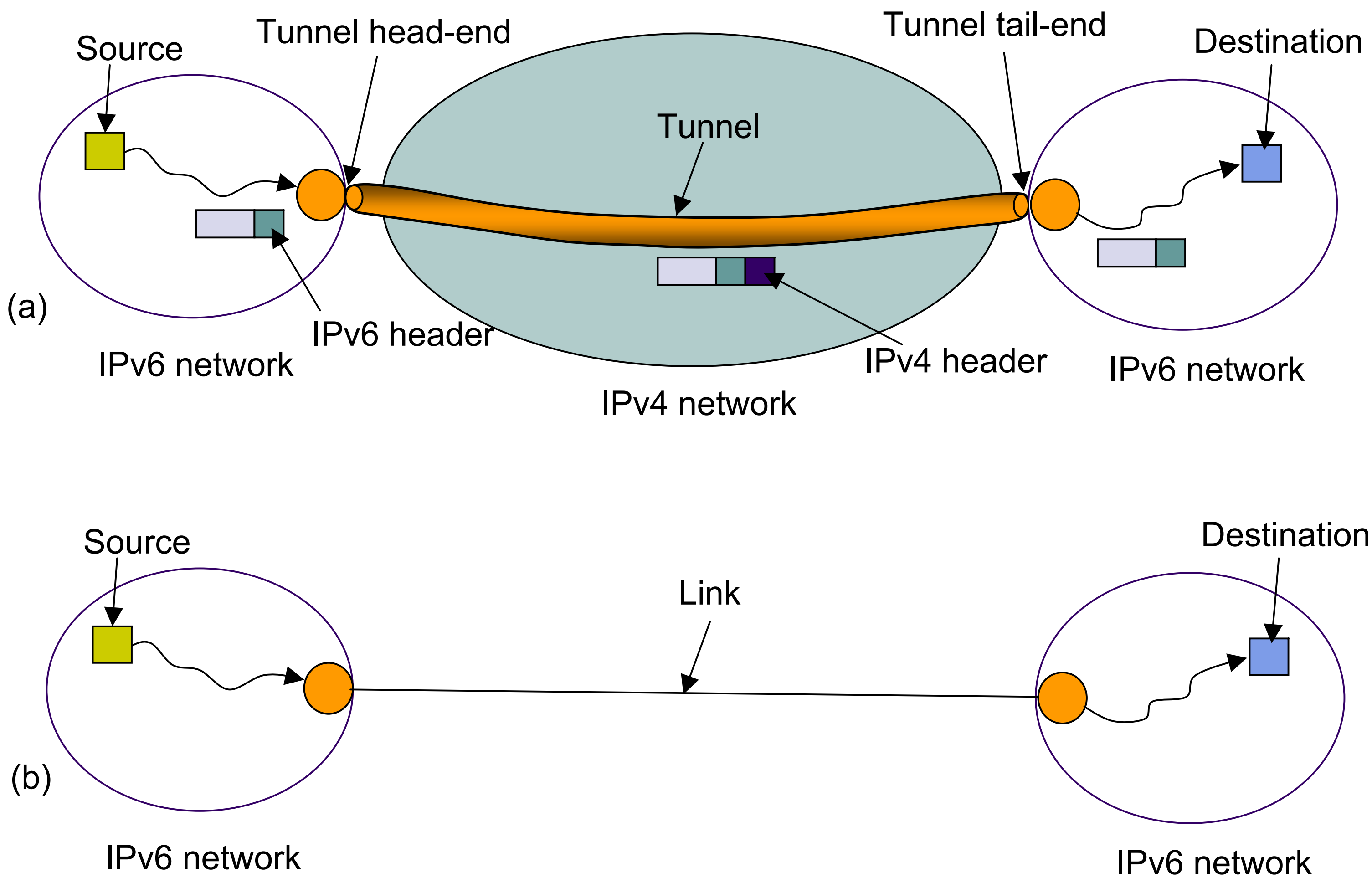
- Source Routing: strict/loose routes



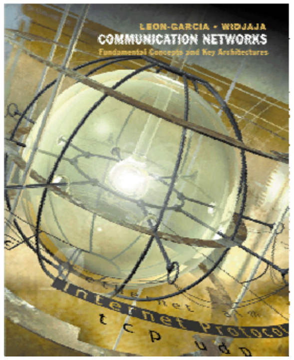
Migration from IPv4 to IPv6

- 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

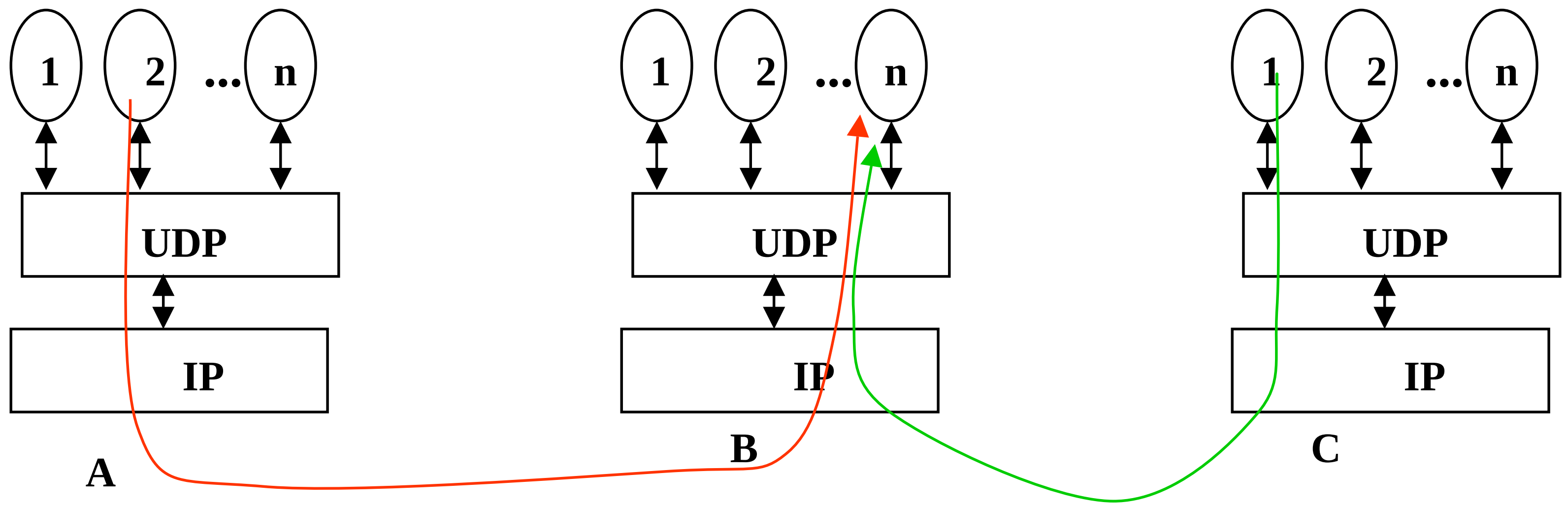
1024-65536

- 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

0	8	16	31
Source IP Address			
Destination IP Address			
0 0 0 0 0 0 0 0	Protocol = 17	UDP Length	

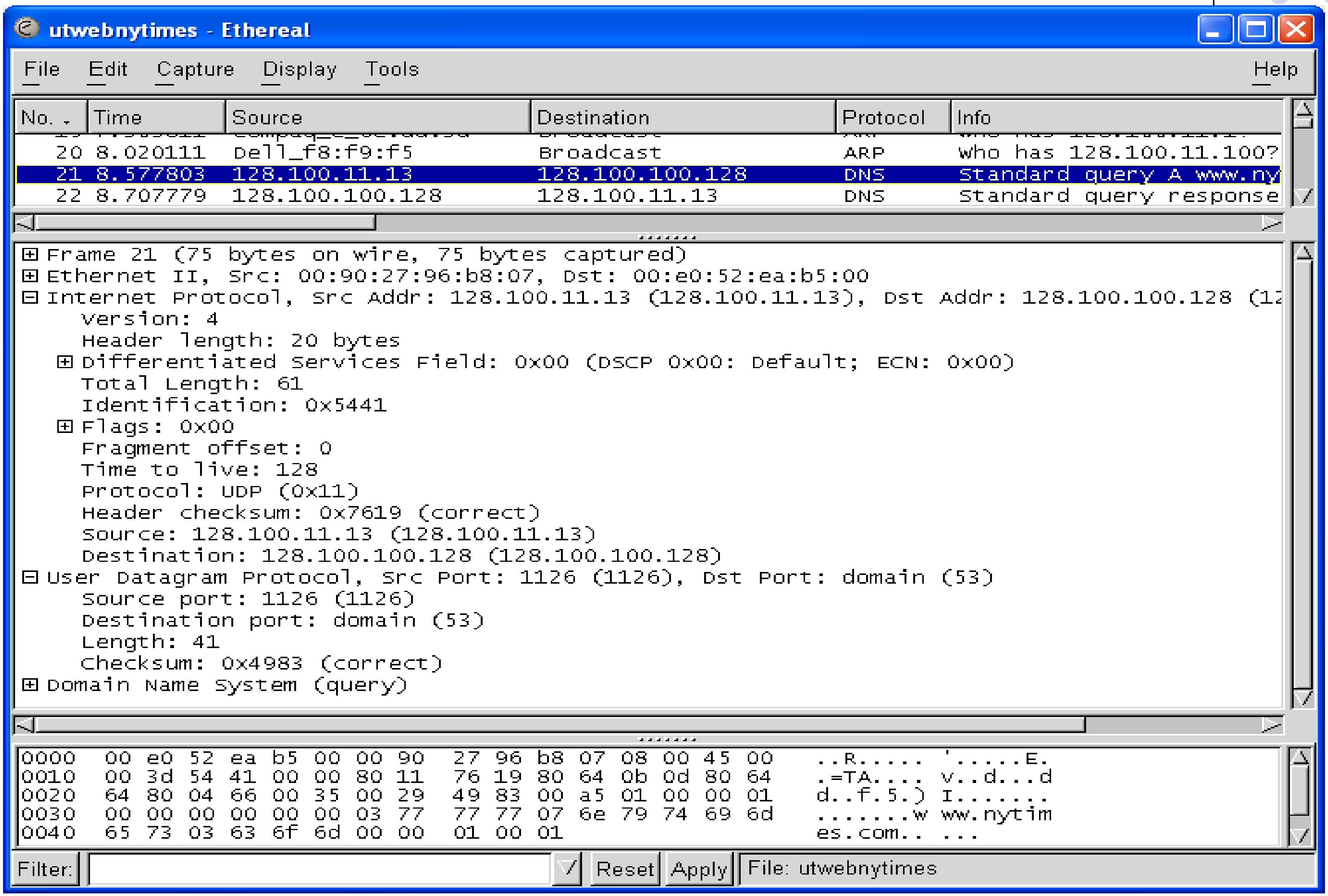
UDP pseudo-header

- 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

Example


utwebnytimes - Ethereal
File Edit Capture Display Tools Help
No. Time Source Destination Protocol Info
20 8.020111 Dell_f8:f9:f5 Broadcast ARP Who has 128.100.11.100?
21 8.577803 128.100.11.13 128.100.100.128 DNS Standard query A www.ny
22 8.707779 128.100.100.128 128.100.11.13 DNS Standard query response
Frame 21 (75 bytes on wire, 75 bytes captured)
Ethernet II, Src: 00:90:27:96:b8:07, Dst: 00:e0:52:ea:b5:00
Internet Protocol, Src Addr: 128.100.11.13 (128.100.11.13), Dst Addr: 128.100.100.128 (12
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
Total Length: 61
Identification: 0x5441
Flags: 0x00
Fragment offset: 0
Time to live: 128
Protocol: UDP (0x11)
Header checksum: 0x7619 (correct)
Source: 128.100.11.13 (128.100.11.13)
Destination: 128.100.100.128 (128.100.100.128)
User Datagram Protocol, Src Port: 1126 (1126), Dst Port: domain (53)
Source port: 1126 (1126)
Destination port: domain (53)
Length: 41
Checksum: 0x4983 (correct)
Domain Name System (query)
0000 00 e0 52 ea b5 00 00 90 27 96 b8 07 08 00 45 00 ..R..... '.....E.
0010 00 3d 54 41 00 00 80 11 76 19 80 64 0b 0d 80 64 .=TA.... v..d...d
0020 64 80 04 66 00 35 00 29 49 83 00 a5 01 00 00 01 d..f.5.) I.......
0030 00 00 00 00 00 00 03 77 77 77 07 6e 79 74 69 6d ....w ww.nytim
0040 65 73 03 63 6f 6d 00 00 01 00 01 es.com.. ...
Filter: Reset Apply File: utwebnytimes

Outline

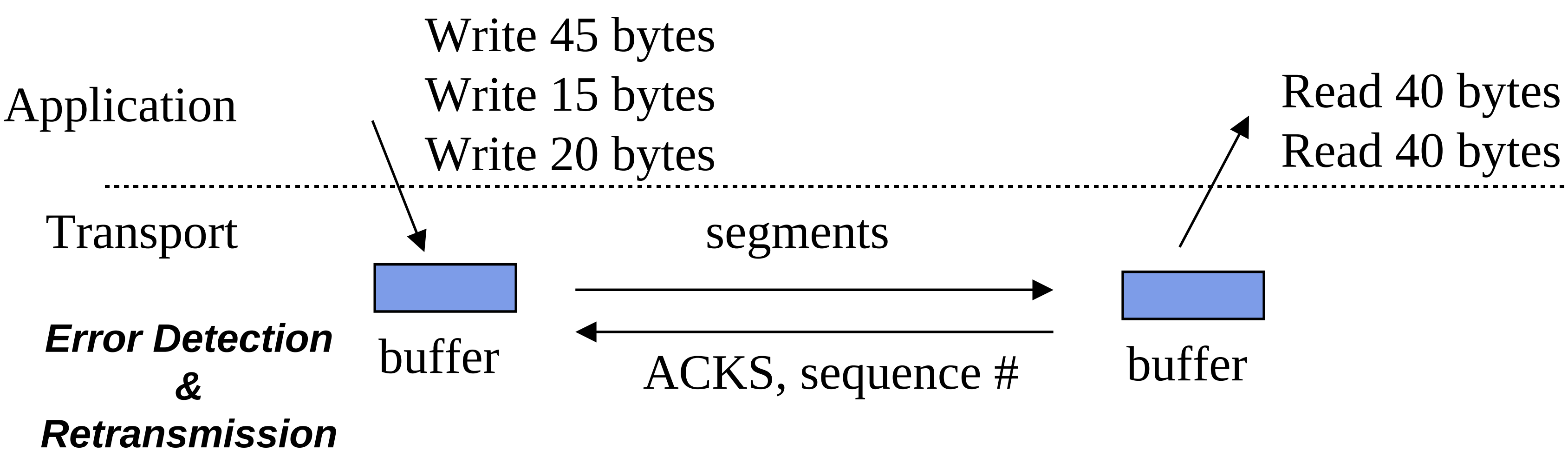
- UDP Protocol
- TCP Reliable Stream Service
- TCP Protocol
- TCP Connection Management
- TCP Congestion Control

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, …

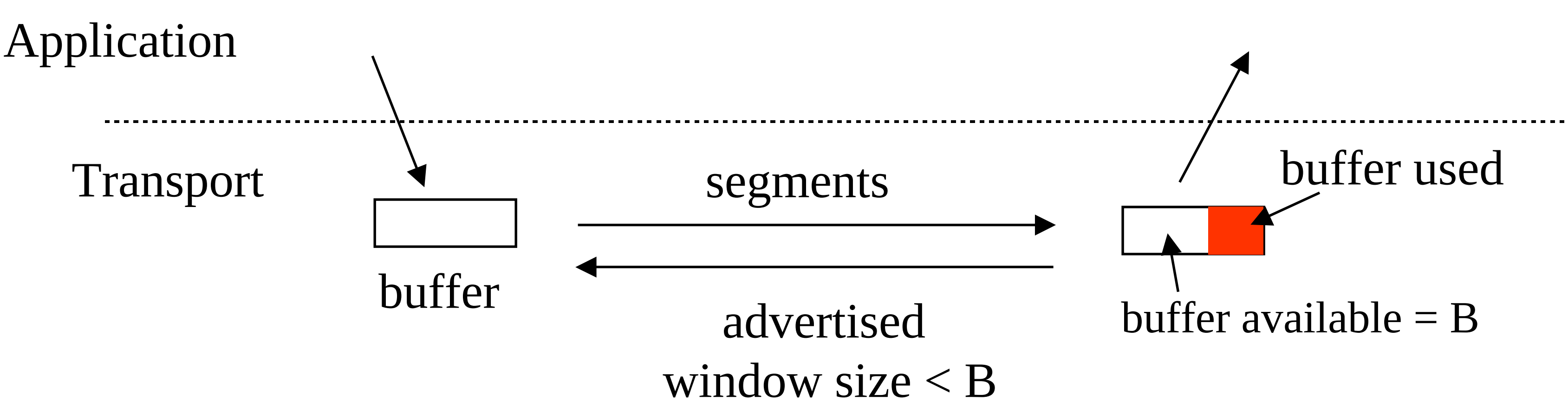
Reliable Byte-Stream Service

- 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



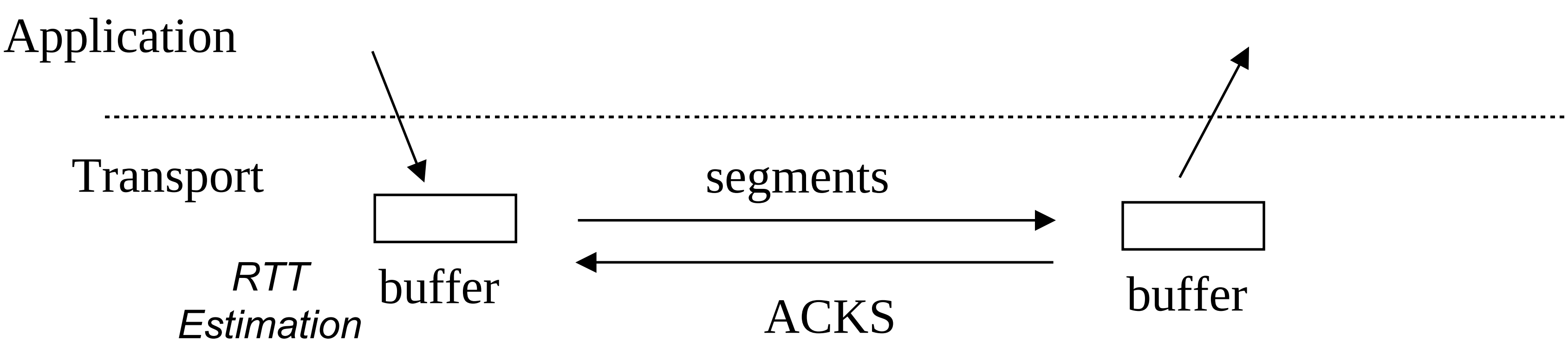
Flow Control

- 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



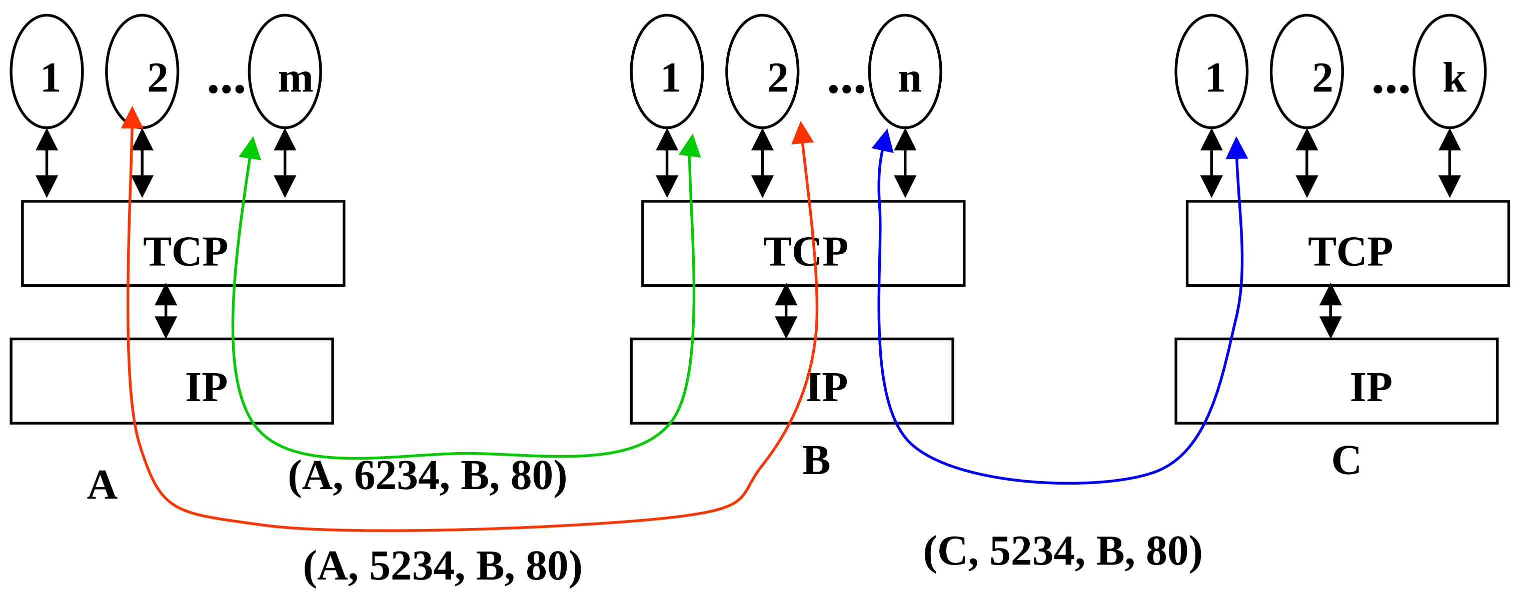
Congestion Control

- 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

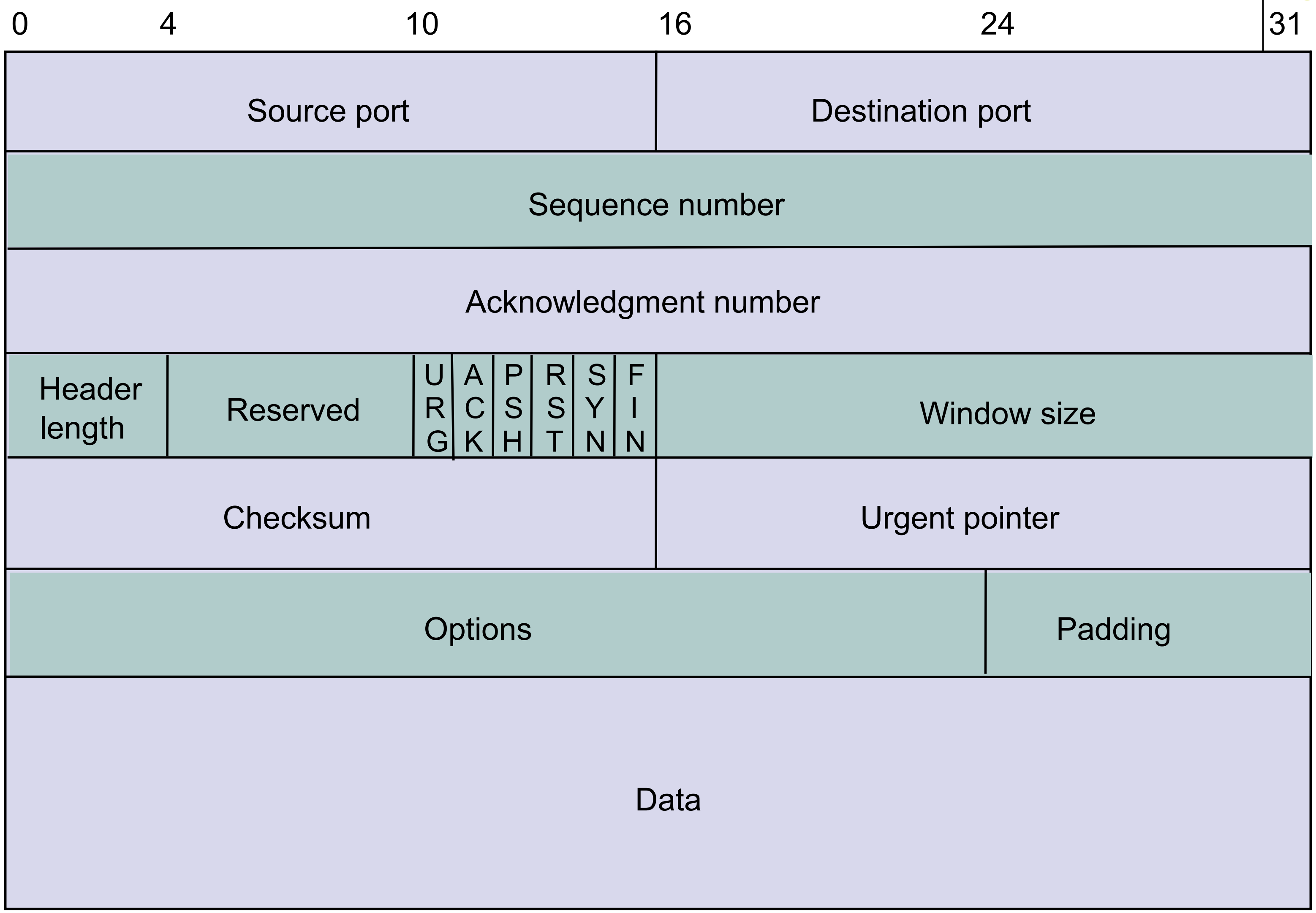
- A *TCP connection* is specified by a *4-tuple*
  - (source IP address, source port, destination IP address, destination port)
- TCP allows multiplexing of multiple connections between end systems to support multiple applications simultaneously
- Arriving segment directed according to connection 4-tuple



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

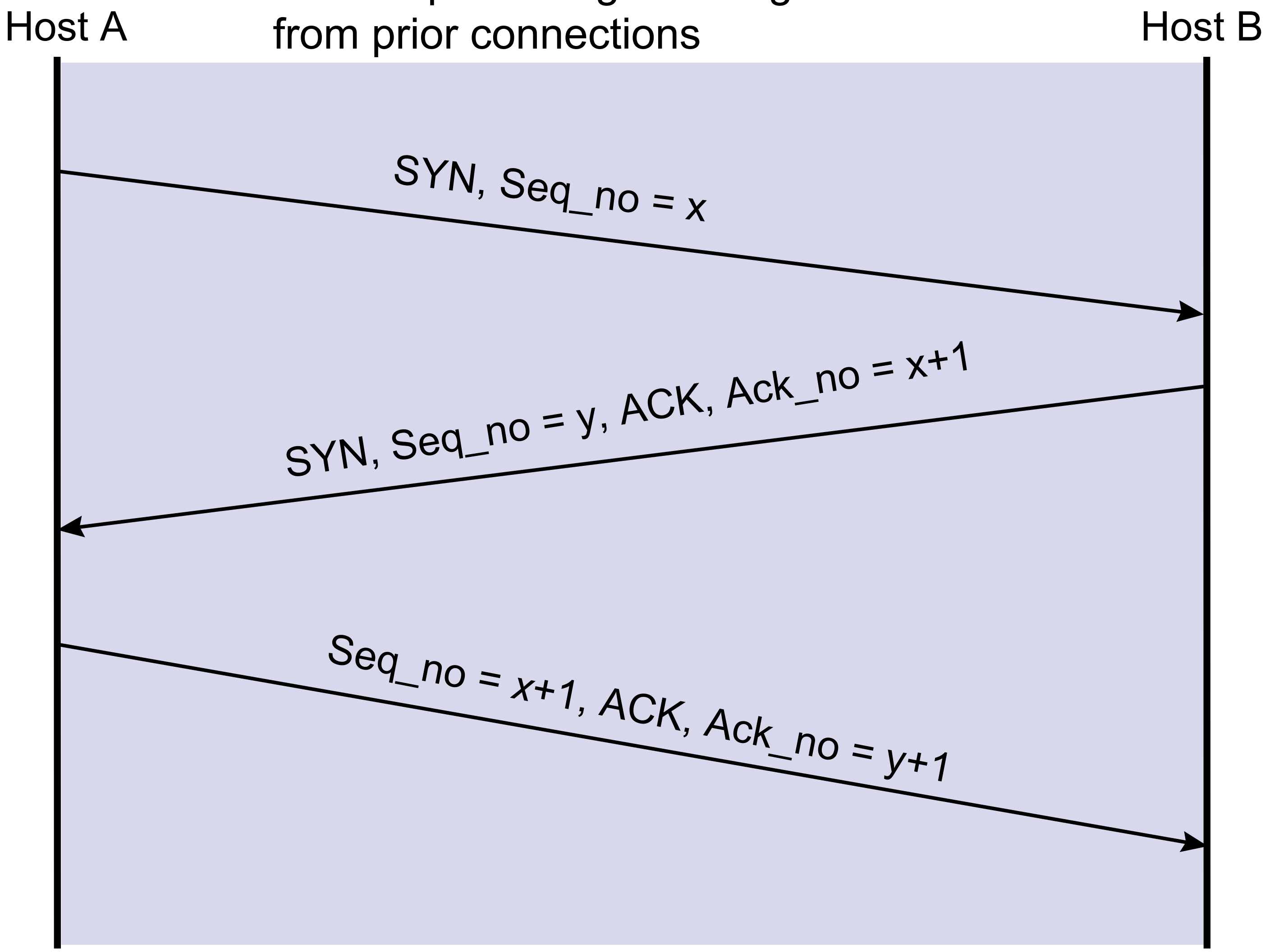
- UDP Protocol
- TCP Reliable Stream Service
- TCP Protocol
- TCP Connection Management
- TCP Congestion Control

Initial Sequence Number

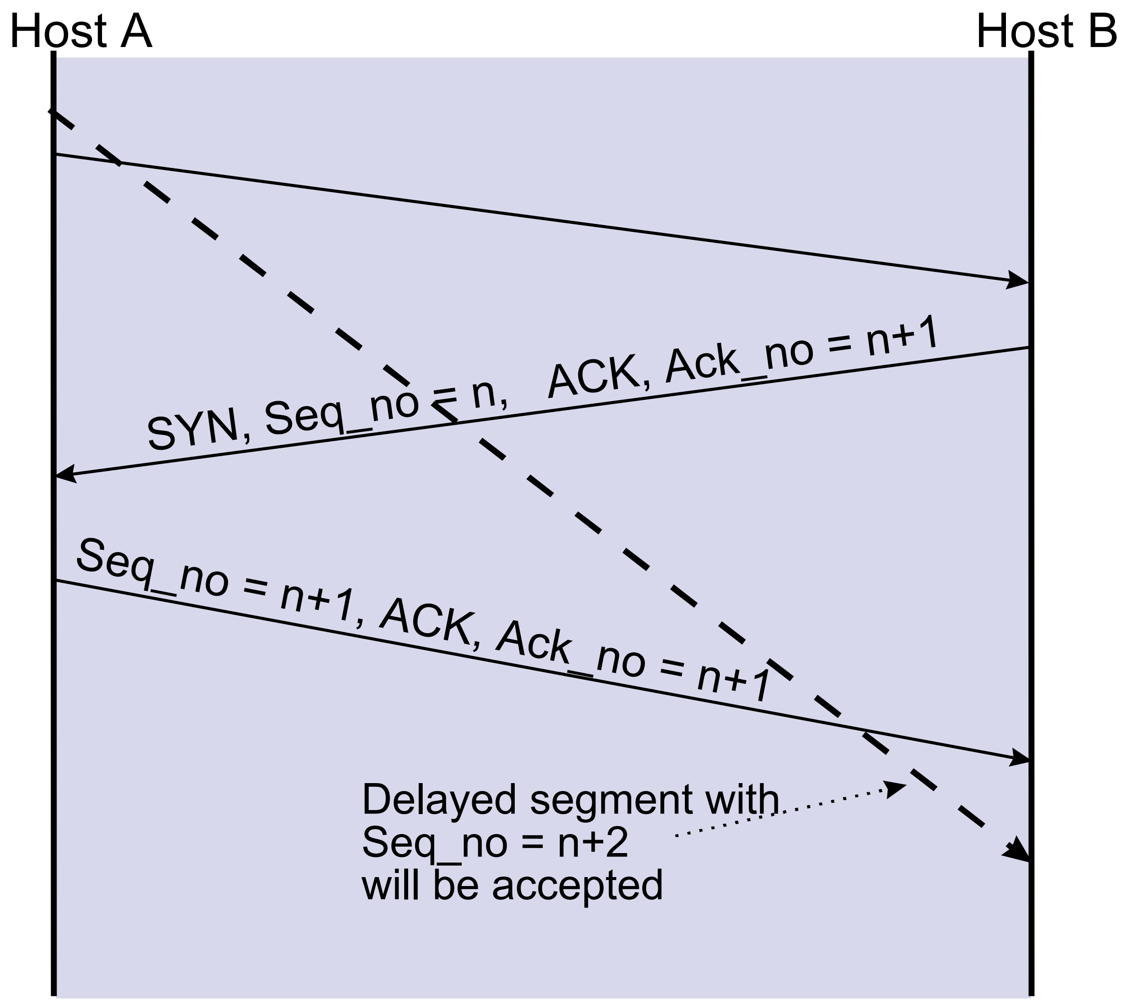
- Select initial sequence numbers (ISN) to protect against segments from prior connections (that may circulate in the network and arrive at a much later time)
- Select ISN to avoid overlap with sequence numbers of prior connections
- Use local clock to select ISN sequence number
- Time for clock to go 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



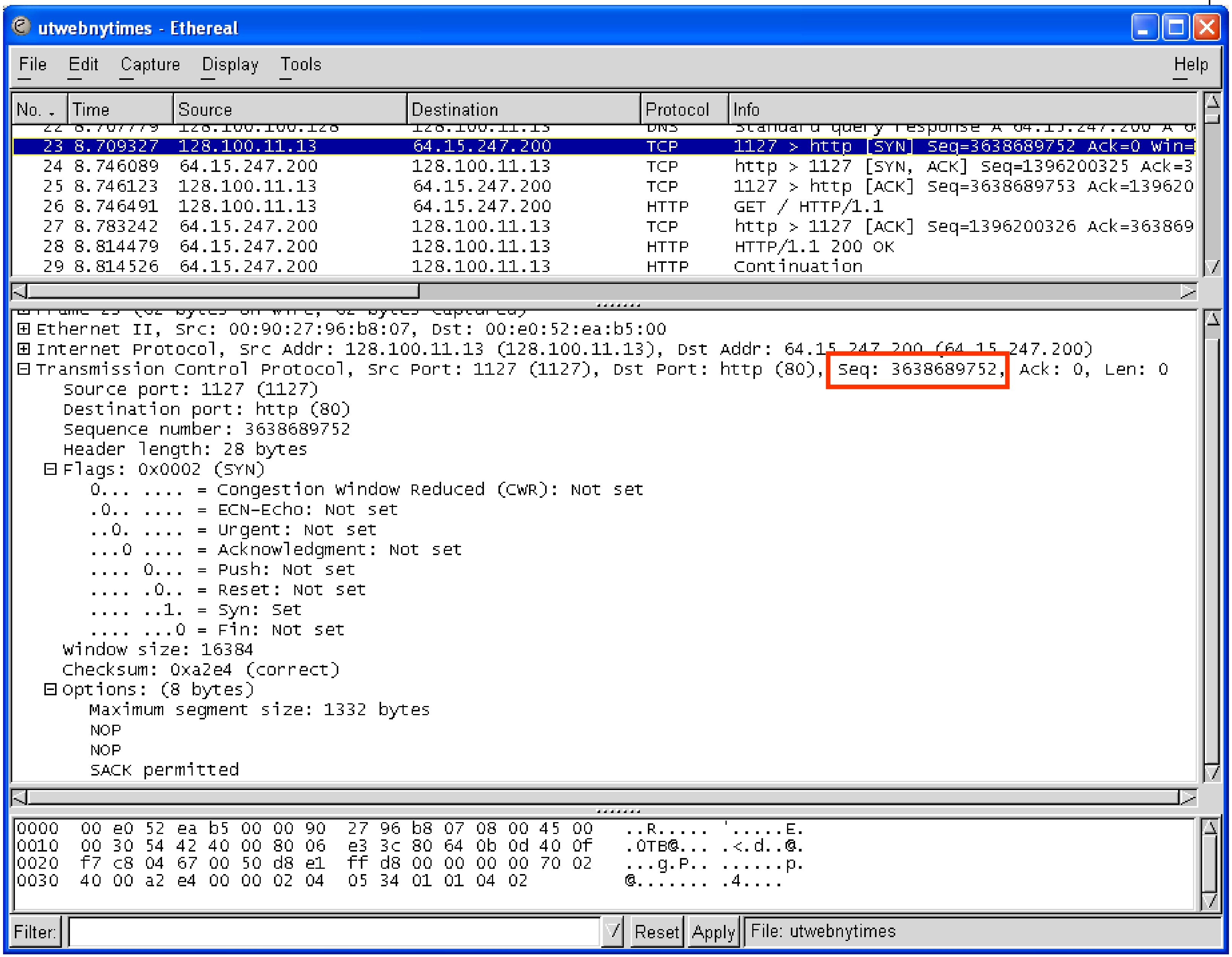
If host always uses the same ISN



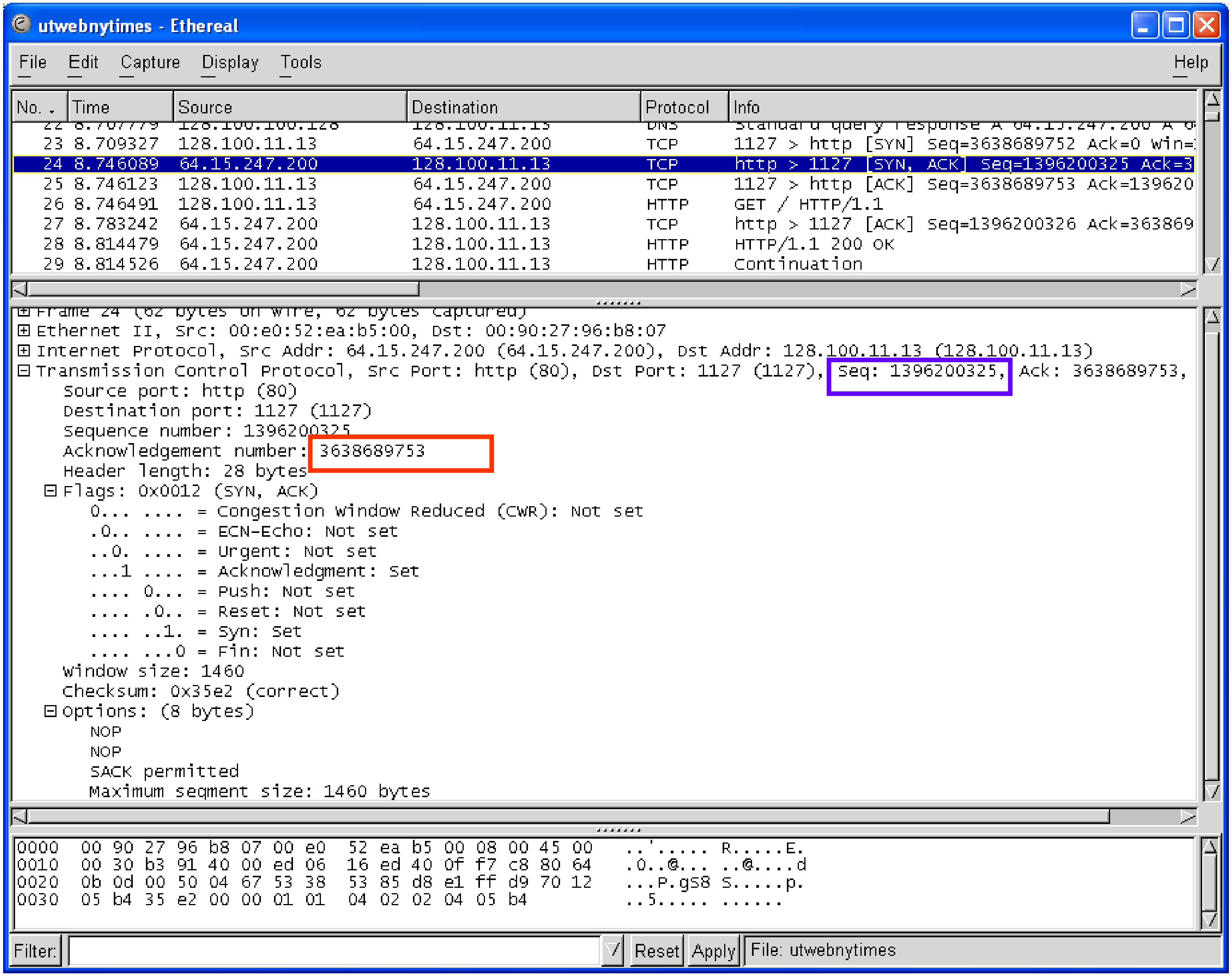
Maximum Segment Size

- 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

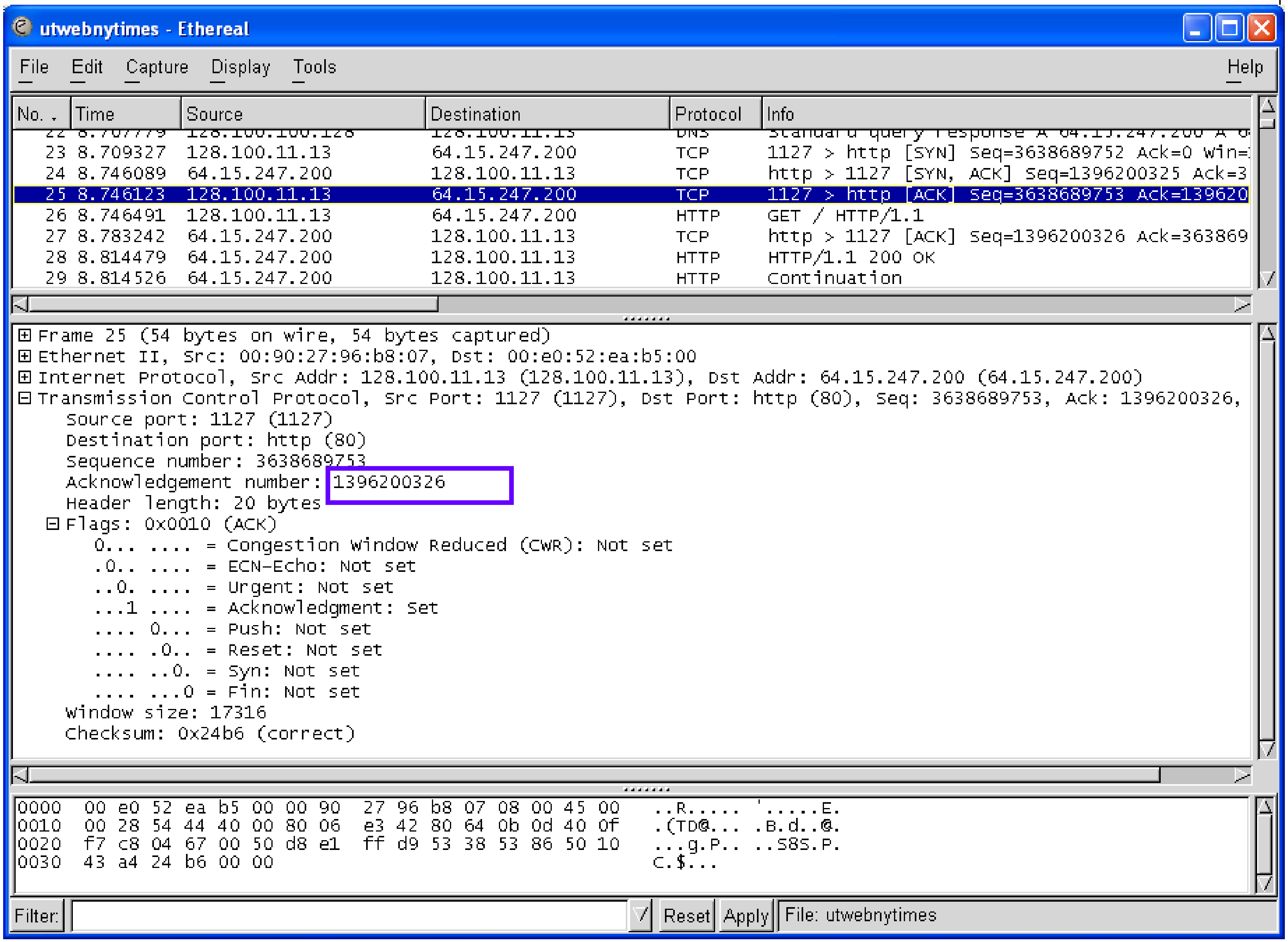
Near End: Connection Request



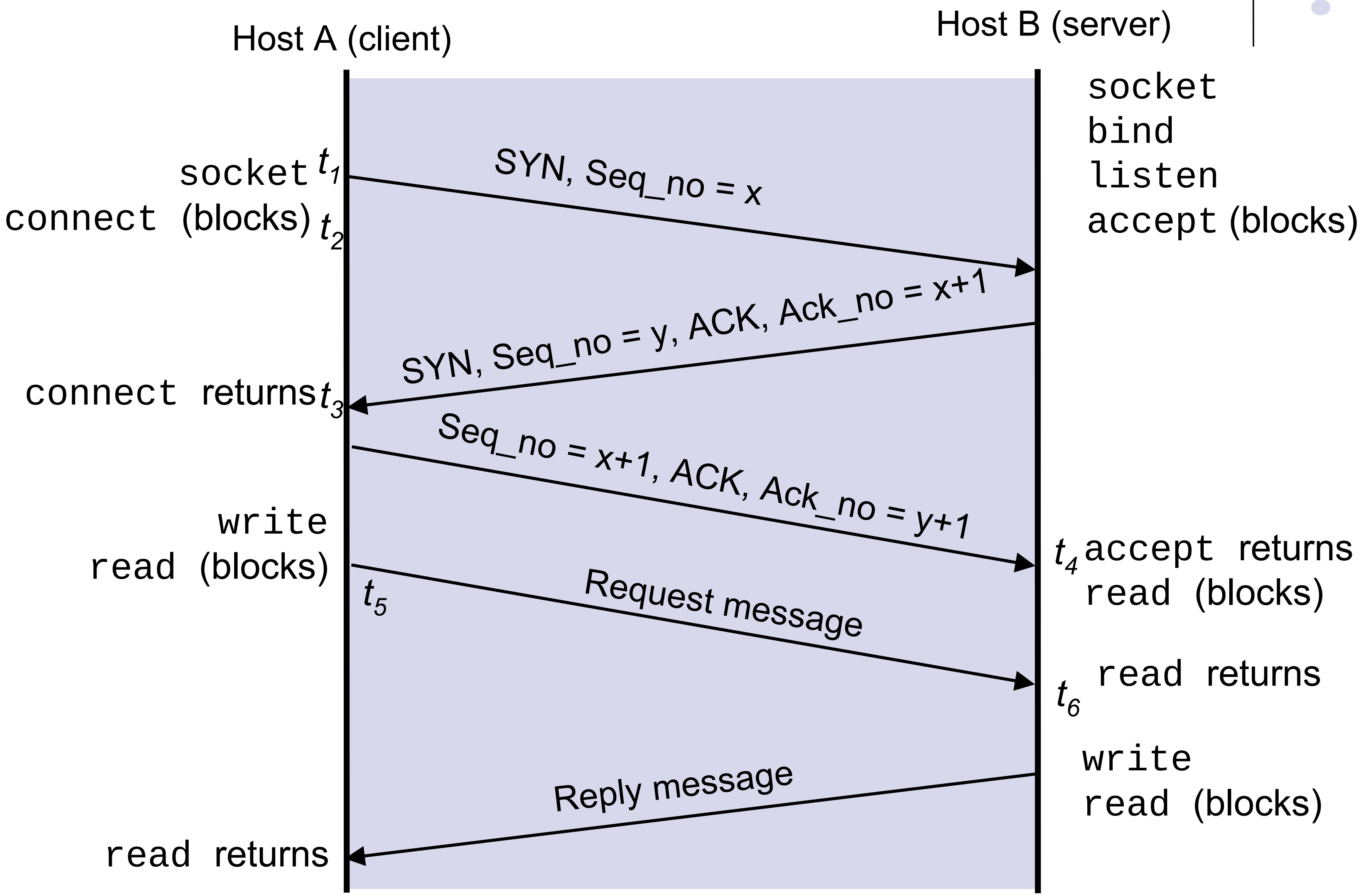
Far End: Ack and Request



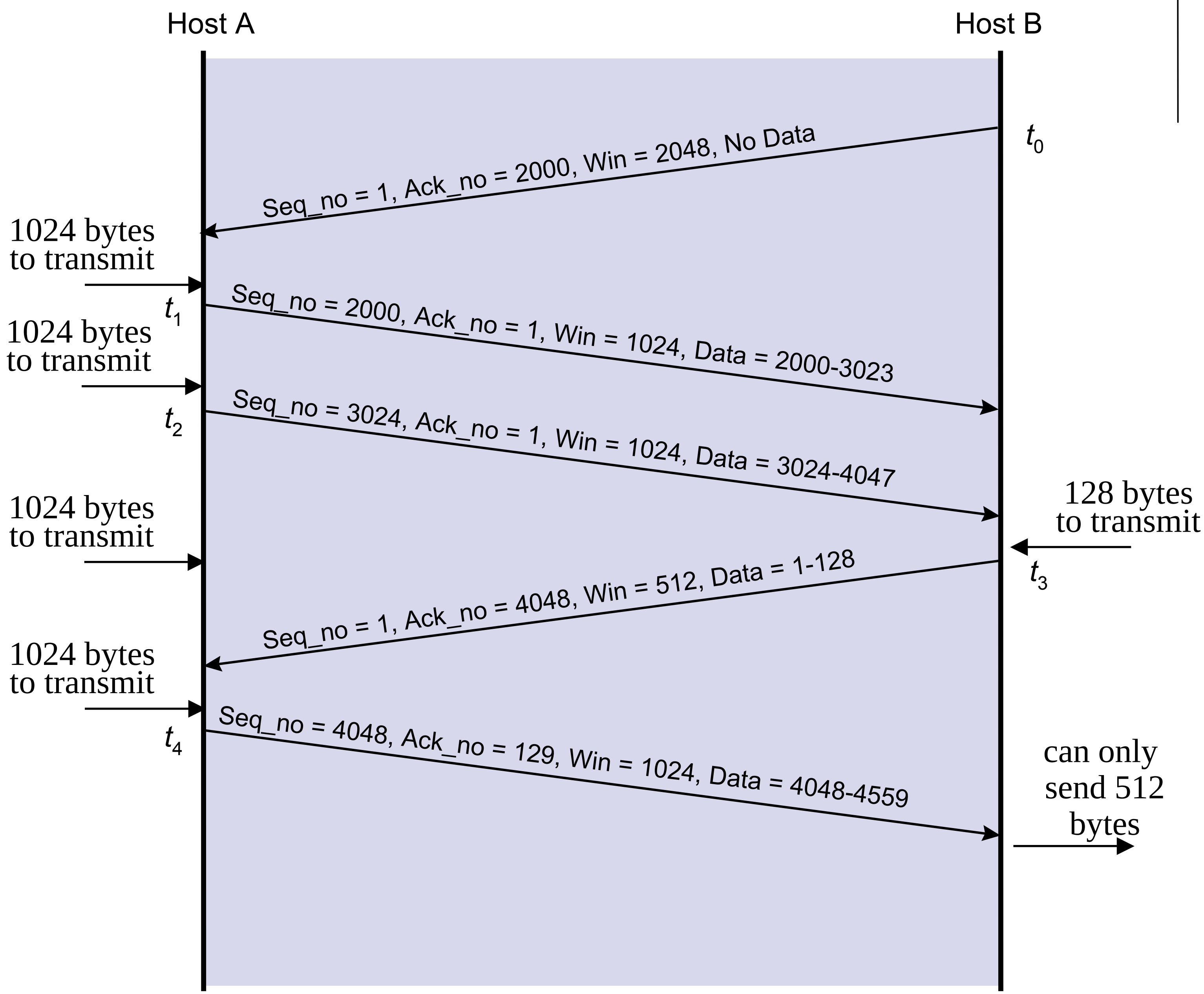
Near End: Ack


utwebnytimes - Ethereal
File Edit Capture Display Tools Help
No. Time Source Destination Protocol Info
23 8.709327 128.100.11.13 64.15.247.200 TCP 1127 > http [SYN] Seq=3638689752 Ack=0
24 8.746089 64.15.247.200 128.100.11.13 TCP http > 1127 [SYN, ACK] Seq=1396200325 Ack=3
25 8.746123 128.100.11.13 64.15.247.200 TCP 1127 > http [ACK] Seq=3638689753 Ack=139620
26 8.746491 128.100.11.13 64.15.247.200 HTTP GET / HTTP/1.1
27 8.783242 64.15.247.200 128.100.11.13 TCP http > 1127 [ACK] Seq=1396200326 Ack=363869
28 8.814479 64.15.247.200 128.100.11.13 HTTP HTTP/1.1 200 OK
29 8.814526 64.15.247.200 128.100.11.13 HTTP Continuation
Frame 25 (54 bytes on wire, 54 bytes captured)
Ethernet II, Src: 00:90:27:96:b8:07, Dst: 00:e0:52:ea:b5:00
Internet Protocol, Src Addr: 128.100.11.13 (128.100.11.13), Dst Addr: 64.15.247.200 (64.15.247.200)
Transmission Control Protocol, Src Port: 1127 (1127), Dst Port: http (80), Seq: 3638689753, Ack: 1396200326,
Source port: 1127 (1127)
Destination port: http (80)
Sequence number: 3638689753
Acknowledgement number: 1396200326
Header length: 20 bytes
Flags: 0x0010 (ACK)
0... .... = Congestion Window Reduced (CWR): Not set
.0.. .... = ECN-Echo: Not set
..0. .... = Urgent: Not set
...1 .... = Acknowledgment: Set
.... 0... = Push: Not set
.... .0.. = Reset: Not set
.... ..0. = Syn: Not set
....0 = Fin: Not set
Window size: 17316
Checksum: 0x24b6 (correct)
0000 00 e0 52 ea b5 00 00 90 27 96 b8 07 08 00 45 00
0010 00 28 54 44 40 00 80 06 e3 42 80 64 0b 0d 40 0f
0020 f7 c8 04 67 00 50 d8 e1 ff d9 53 38 53 86 50 10
0030 43 a4 24 b6 00 00
Filter:
Reset Apply
File: utwebnytimes

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

Sequence Number Wraparound

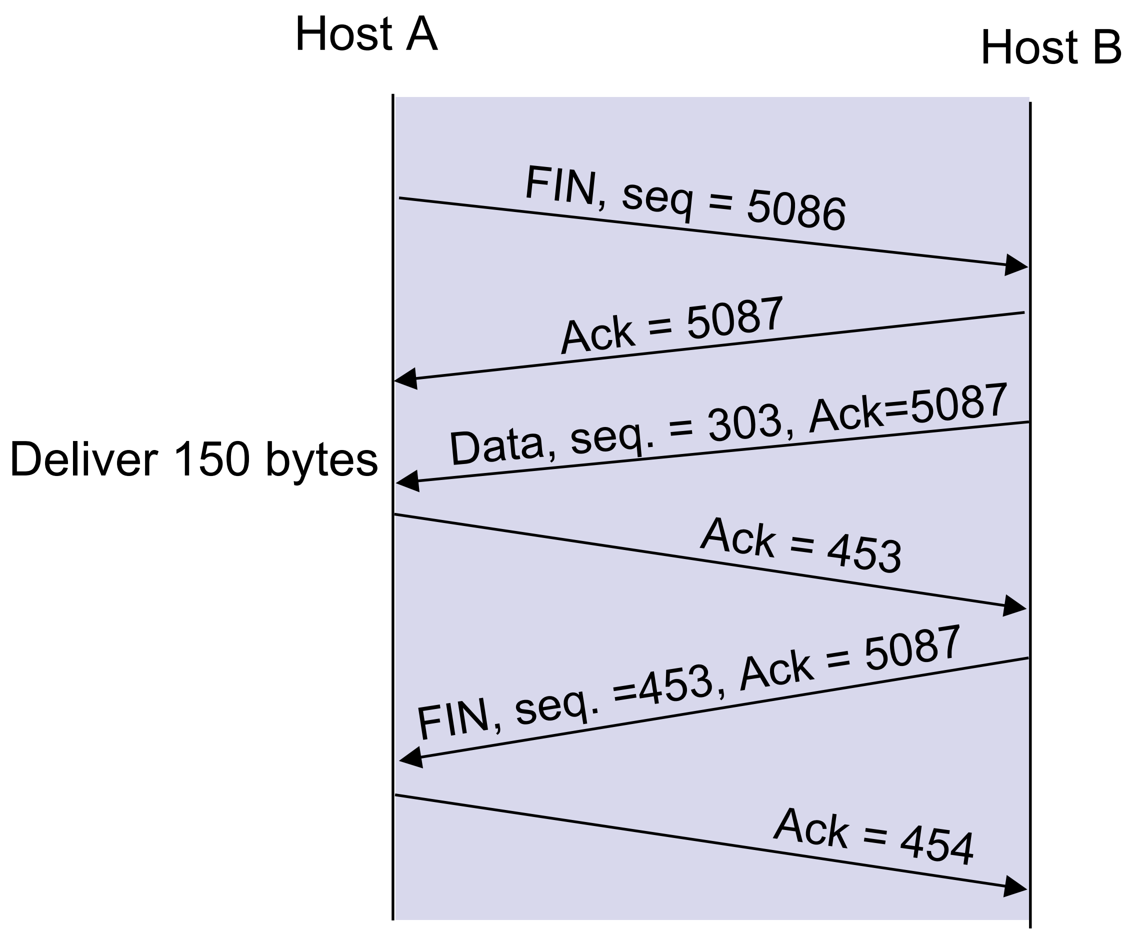
- $2^{32} = 4.29 \times 10^9$ bytes $= 34.3 \times 10^9$ bits
  - At 1 Gbps, sequence number wraparound in 34.3 seconds.
- Timestamp option: Insert 32 bit timestamp in header of each segment
  - Timestamp + sequence no → 64-bit seq. no
  - Timestamp clock must:
    - tick forward at least once every 2^{31} bits
    - Not complete cycle in less than one MSL
    - Example: clock tick every 1 ms @ 8 Tbps wraps around in 25 days

Delay-BW Product & Advertised Window Size

- Suppose RTT=100 ms, R=2.4 Gbps
  - # bits in pipe = 3 Mbytes
- If single TCP process occupies pipe, then required advertised window size is
  - RTT x Bit rate = 3 Mbytes
  - Normal maximum window size is 65535 bytes
- Solution: Window Scale Option
  - Window size up to 65535 x 2^{14} = 1 Gbyte allowed
  - Requested in SYN segment

TCP Connection Closing

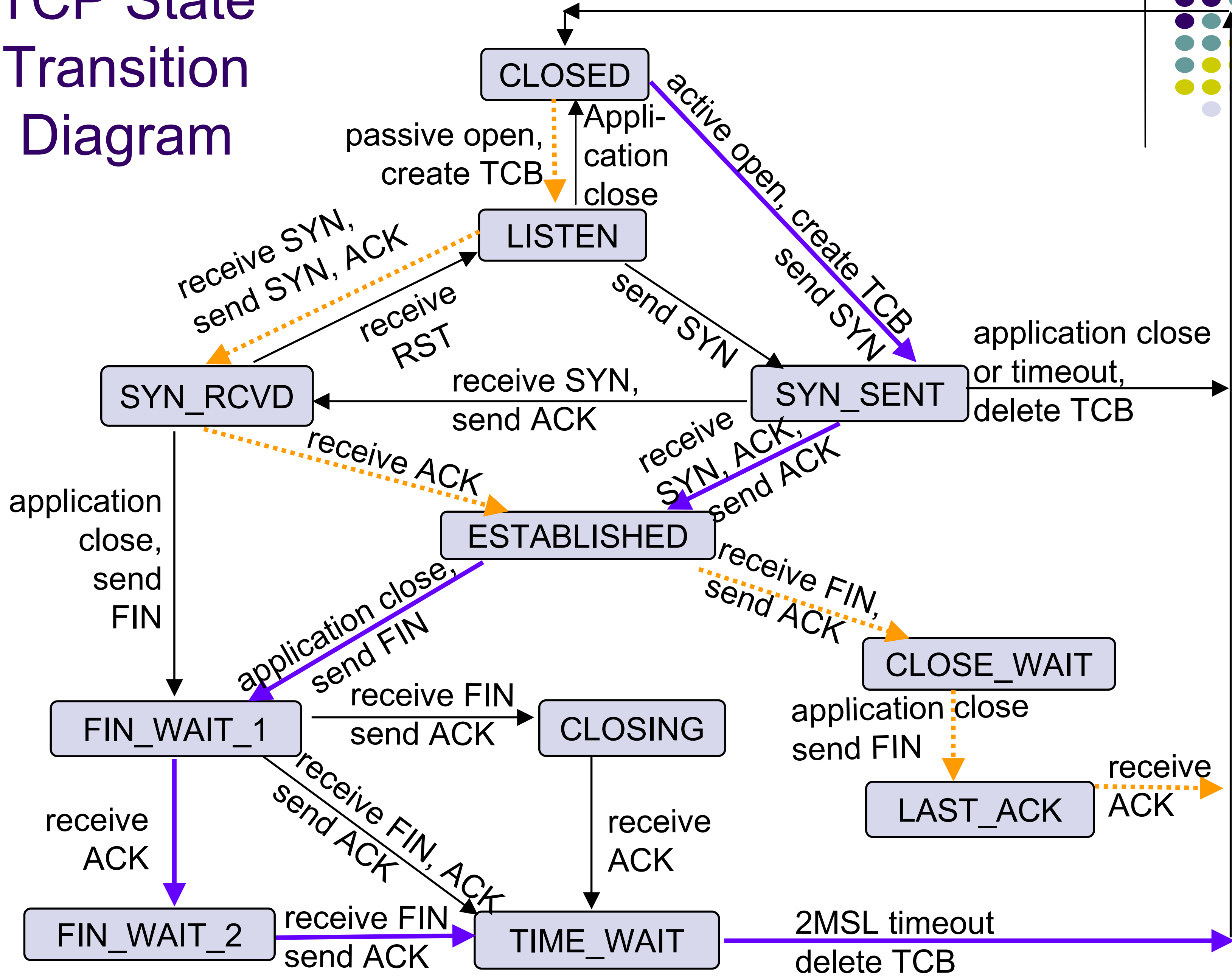
"Graceful Close"



TIME_WAIT state

- When TCP receives ACK to last FIN, TCP enters TIME_WAIT state
  - Protects future incarnations of connection from delayed segments
  - TIME_WAIT = 2 x MSL
  - Only valid segment that can arrive while in TIME_WAIT state is FIN retransmission
    - If such segment arrives, resent ACK & restart TIME_WAIT timer
  - When timer expires, close TCP connection & delete connection record

TCP State Transition Diagram

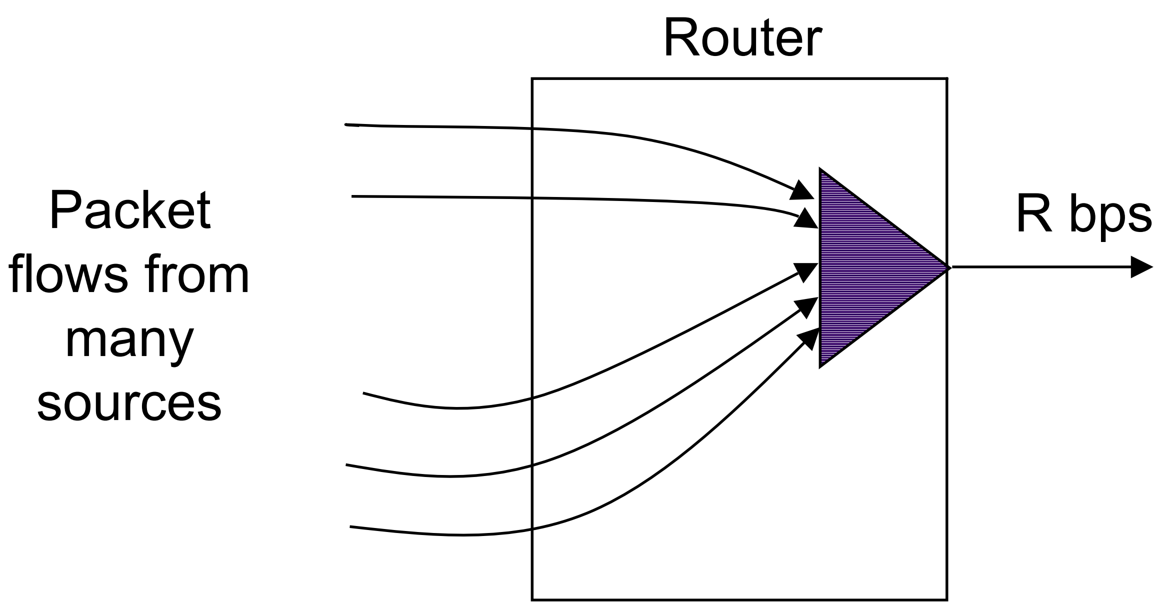


Outline

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

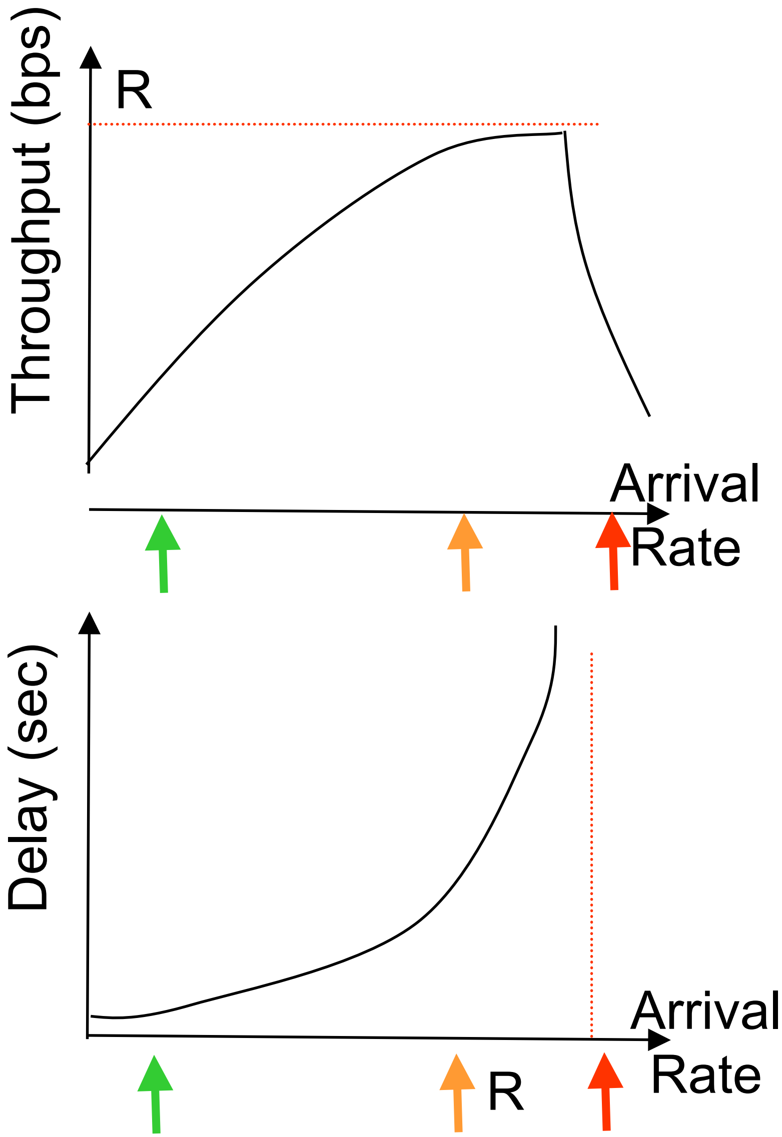
TCP Congestion Control

- *Advertised window* size is used to ensure that receiver's buffer will not overflow
- However, buffers at intermediate routers between source and destination may overflow



- Congestion occurs when total arrival rate from all packet flows exceeds R over a sustained period of time
- Buffers at multiplexer will fill and packets will be lost

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

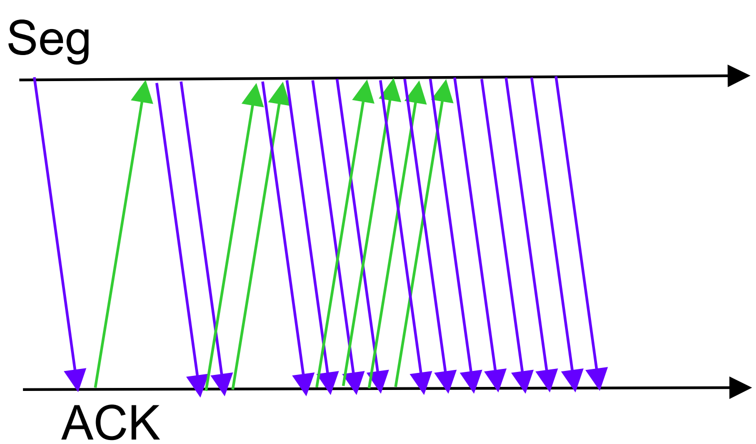
- Desired operating point: just before knee
  - Sources must control their sending rates so that aggregate arrival rate is just before knee
- TCP sender maintains a *congestion window* cwnd to control congestion at intermediate routers
- Effective window is minimum of congestion window and advertised 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

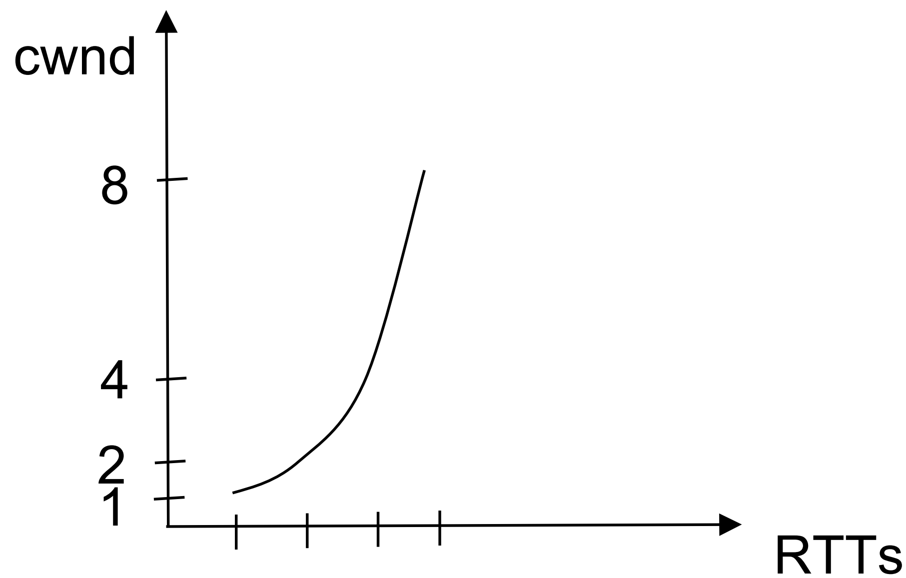
Congestion Window

- 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

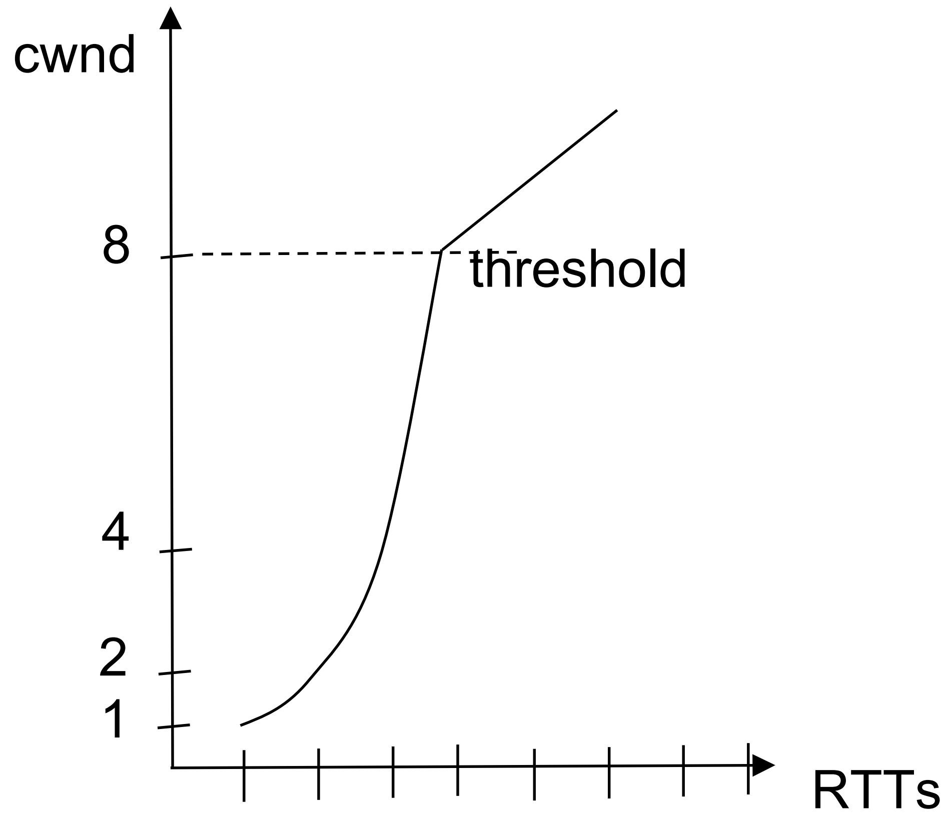
- **Slow start**: increase congestion window size by one segment upon receiving an ACK from receiver
  - initialized at ≤ 2 segments
  - used at (re)start of data transfer
  - congestion window increases exponentially



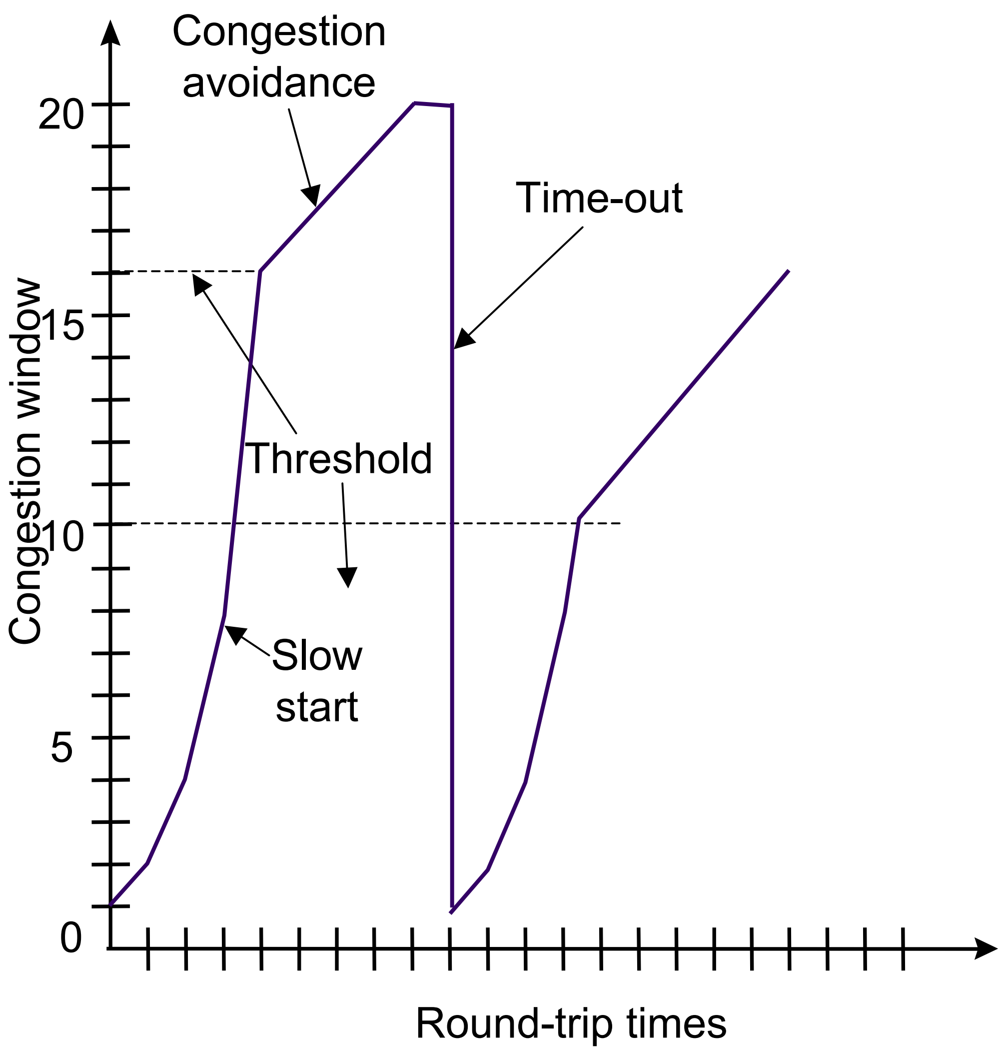


TCP Congestion Control: Congestion Avoidance

- 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



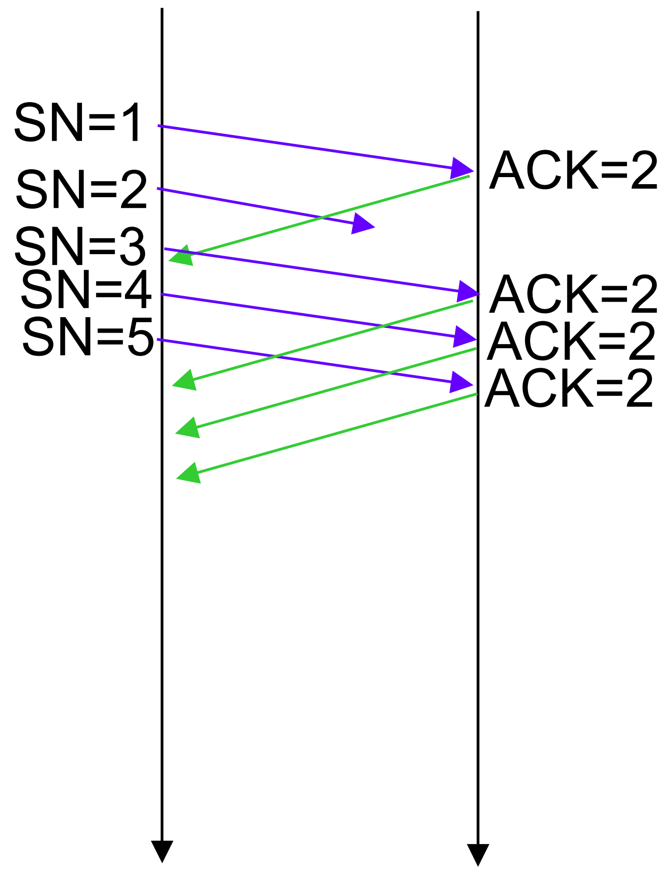
TCP Congestion Control: Congestion



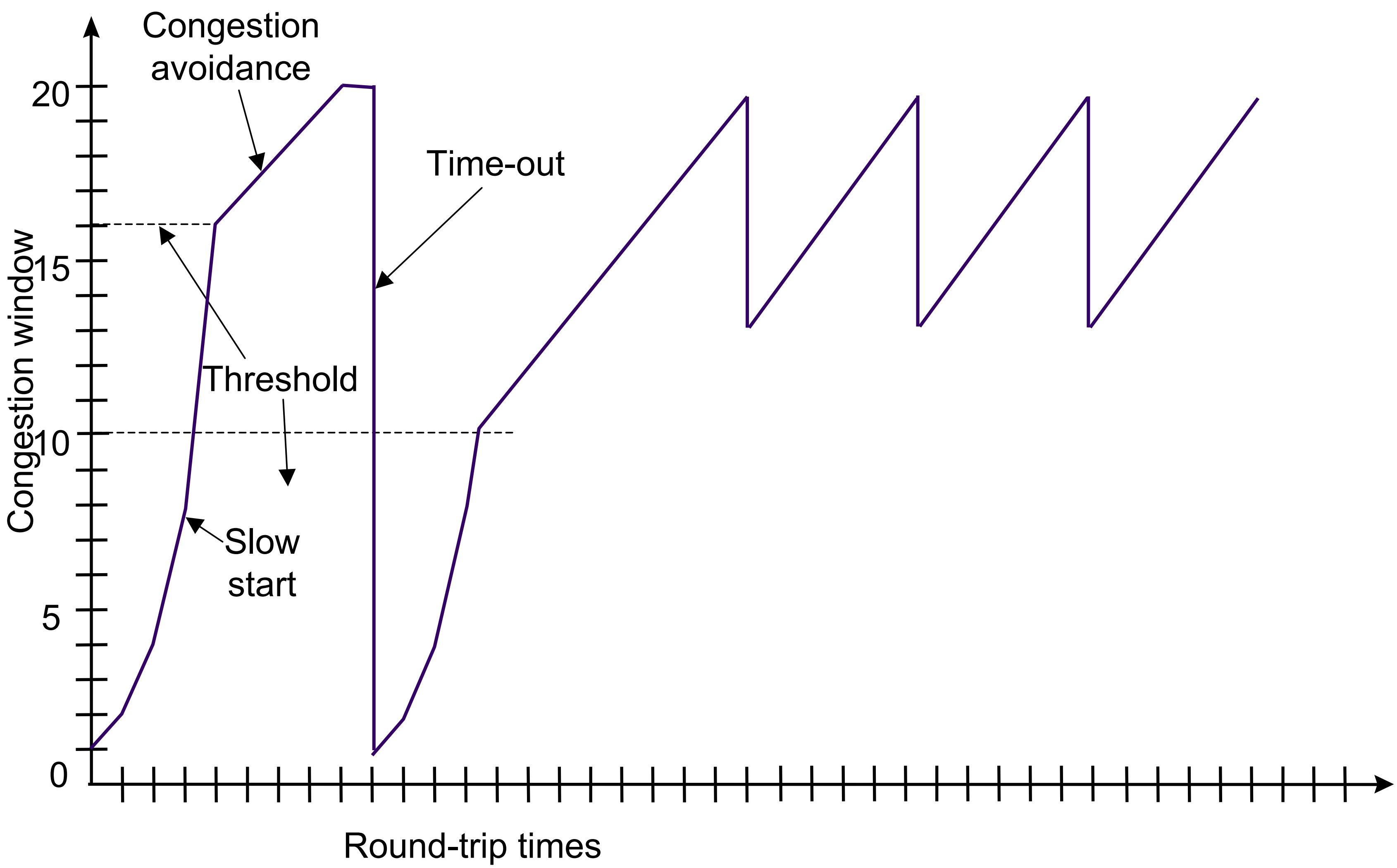
- Congestion is detected upon timeout or receipt of duplicate ACKs
- Assume current cwnd corresponds to available bandwidth
- Adjust congestion threshold = ½ x current cwnd
- Reset cwnd to 1
- Go back to slow-start
- Over several cycles expect to converge to congestion threshold equal to about ½ the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

Fast Retransmit & Fast Recovery

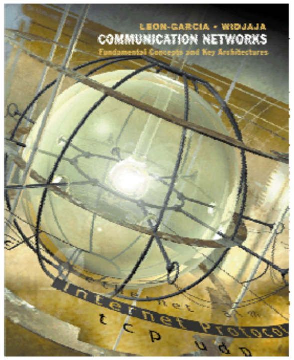
- Congestion causes many segments to be dropped
- If only a single segment is dropped, then subsequent segments trigger duplicate ACKs before timeout
- Can avoid large decrease in cwnd as follows:
  - When three duplicate ACKs arrive, retransmit lost segment immediately
  - Reset congestion threshold 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

Host Behavior

- Every host must do IP forwarding
- For datagram generated by own higher layers
  - if destination connected through point-to-point link or on shared network, send datagram directly to destination
  - Else, send datagram to a default router
- For datagrams received on network interface
  - if destination address, own address, pass to higher layer
  - if destination address, not own, discard “silently”

Router Behavior

Router's IP layer

- can receive datagrams from own higher layers
- can receive datagram from a network interface
  - if destination IP address own or broadcast address, pass to layer above
  - else, forward the datagram to next hop
- routing table determines handling of datagram

Routing Table Entries

- Destination IP Address:
  - complete host address or network address
- IP address of
  - next-hop router or directly connected network
- Flags
  - Is destination IP address a net address or host address?
  - Is next hop, a router or directly connected?
- Network interface on which to send packet

Static routing

- Used on hosts or on very small networks
- Manually tell the machine where to send the packets for each prefix

```
% netstat -nr

Routing Table:
  Destination Gateway       Flags Ref   Use
   Interface
------------- ------------ ----- ---- ----- --------
   -
127.0.0.1     127.0.0.1      UH    0      0   lo0
128.100.10.0  128.100.10.9   U     3    548   le0
224.0.0.0     128.100.10.9   U     3      0   le0
default       128.100.10.2   UG    0  35792
```

U-Route is up   H-route is to host (else route is to network)

G-route to gateway (else direct connection)

Forwarding Procedure

- Does routing table have entry that matches complete destination IP address? If so, use this entry to forward
- Else, does routing table have entry that matches the longest prefix of the destination IP address? If so, use this entry to forward
- Else, does the routing table have a default entry? If so, use this entry.
- Else, packet is undeliverable

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

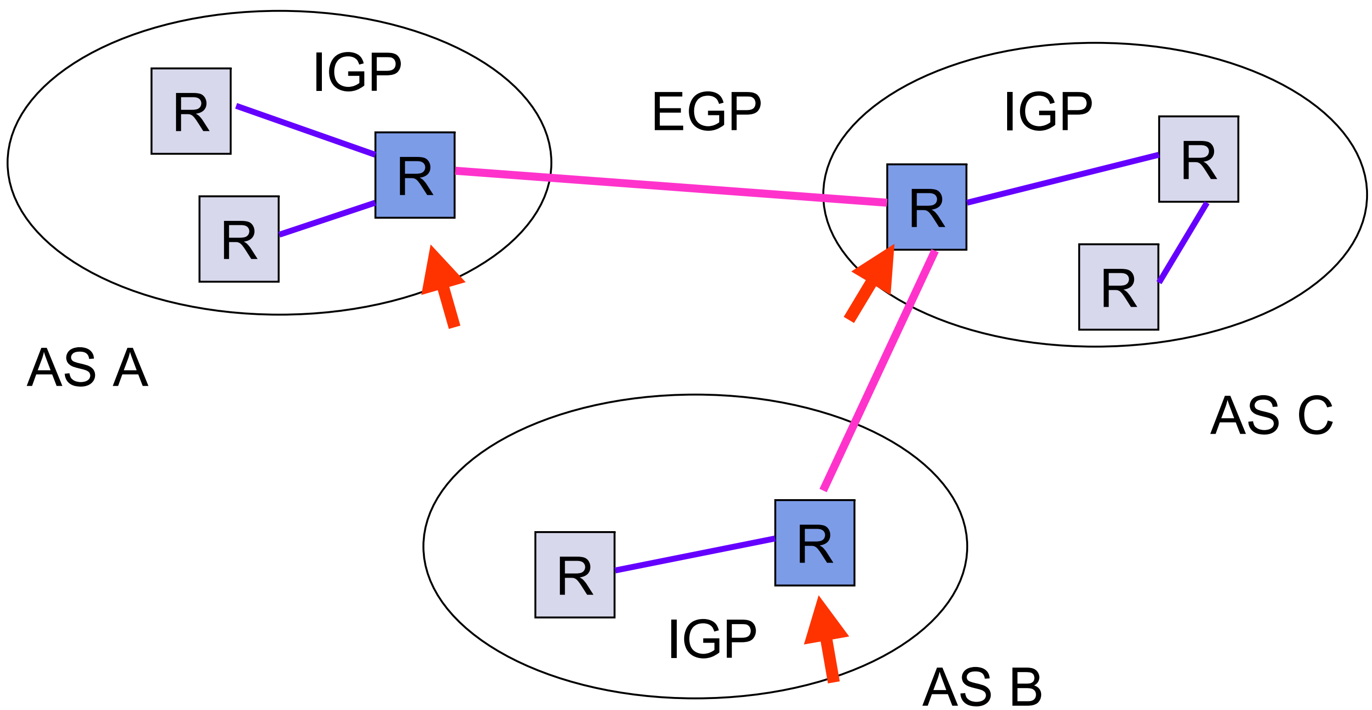
Interior Gateway Protocol (IGP): routing within AS

- 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

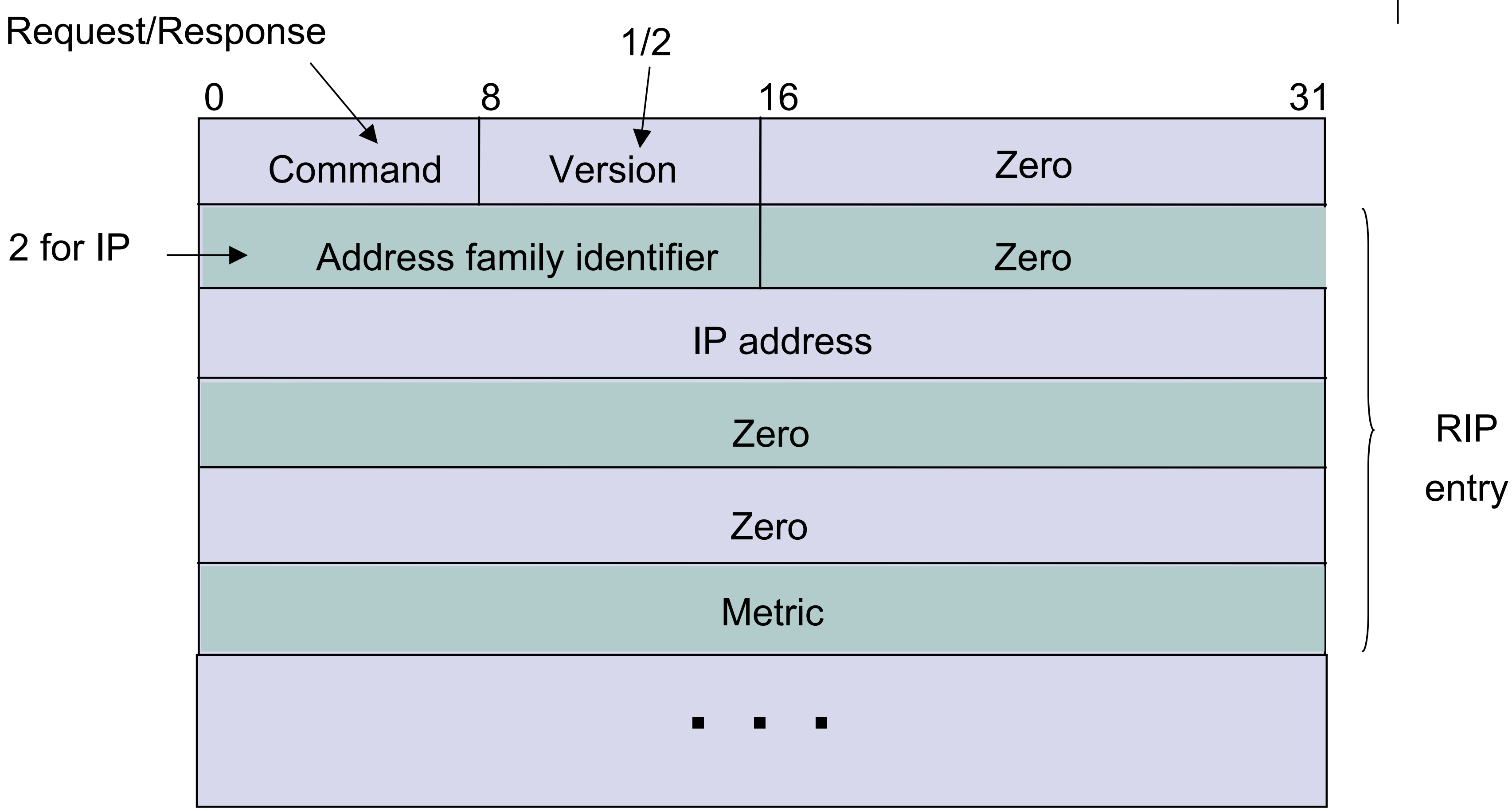
RIP Operation

- 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

RIP Protocol

- Routers run RIP in active mode (advertise distance vector tables)
- Hosts can run RIP in passive mode (update distance vector tables, but do not advertise)
- RIP datagrams broadcast over LANs & specifically addressed on pt-pt or multi-access non-broadcast nets
- Two RIP packet types:
  - *request* to ask neighbor for distance vector table
  - *response* to advertise distance vector table
    - periodically; in response to request; triggered

RIP Message Format



Up to 25 RIP entries per message

RIP Message Format

- Command: request or response
- Version: v1 or v2
- One or more of:
  - Address Family: 2 for IP
  - IP Address: network or host destination
  - Metric: number of hops to destination
- Does not have access to subnet mask information
- Cannot work with variable-length subnet masks

RIP v2 (RFC 2453):

- Subnet mask, next hop, routing domain
- can work with CIDR
- still uses max cost of 16

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

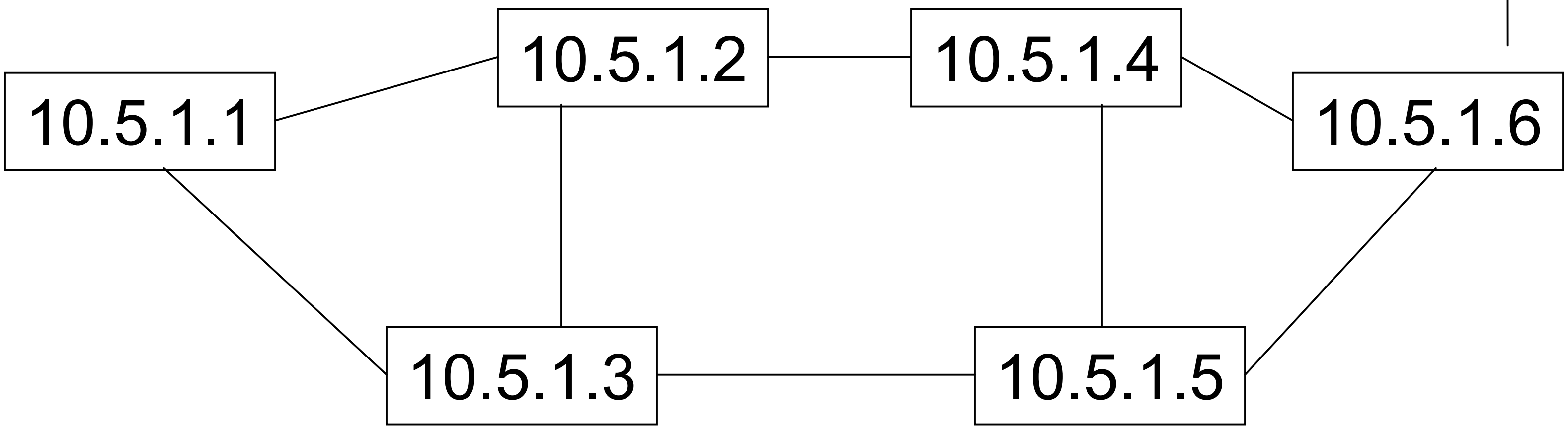
OSPF Features

- *Multiple routes* to a given destination, one per type of service
- Support for *variable-length subnetting* by including the subnet mask in the routing message
- More *flexible link cost* which can range from 1 to 65,535
- Distribution of traffic over *multiple paths* of equal cost
- *Authentication* to ensure routers exchange information with trusted neighbors
- Uses *notion of area* to partition sites into subsets
- Support *host-specific routes* as well as net-specific routes
- *Designated router* to minimize table maintenance overhead

Flooding

- Used in OSPF to distribute link state (LS) information
- Forward incoming packet to all ports except where packet came in
- Packet eventually reaches destination as long as there is a path between the source and destination
- Generates exponential number of packet transmissions
- Approaches to limit # of transmissions:
  - Use a TTL at each packet; won't flood if TTL is reached
  - Each router adds its identifier to header of packet before it floods the packet; won't flood if its identifier is detected
  - Each packet from a given source is identified with a unique sequence number; won't flood if sequence number is same

Example OSPF Topology



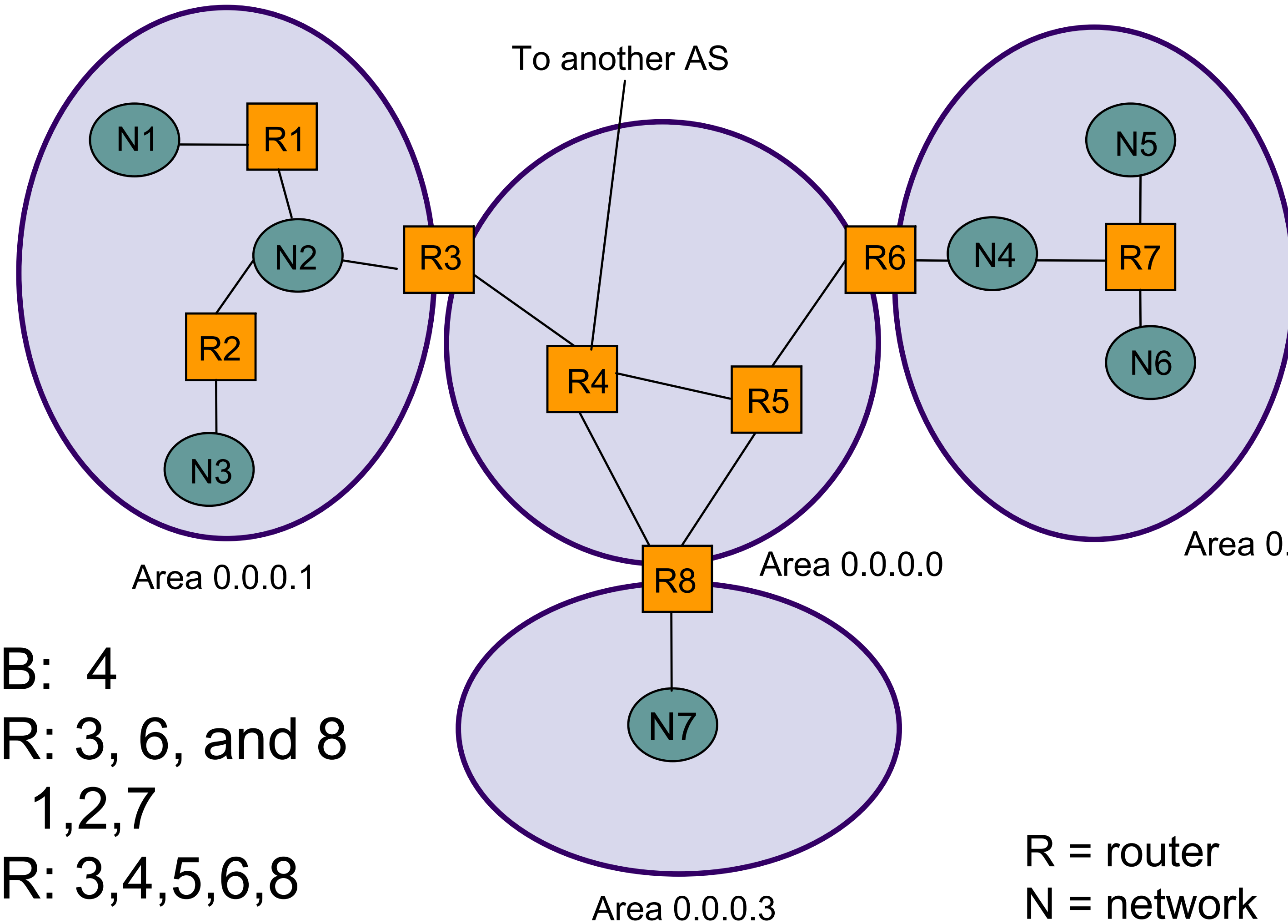
At steady state:

- All routers have same LS database
- Know how many routers in network
- Interfaces & links between routers
- Cost of each link
- Occasional Hello messages (10 sec) & LS updates sent (30 min)

OSPF Network

- To improve scalability, AS may be partitioned into *areas*
  - Area is identified by 32-bit Area ID
  - Router in area only knows complete topology inside area & limits the flooding of link-state information to area
  - *Area border routers* summarize info from other areas
- Each area must be connected to *backbone area* (0.0.0.0)
  - Distributes routing info between areas
- *Internal router* has all links to nets within the same area
- *Area border router* has links to more than one area
- *backbone router* has links connected to the backbone
- *Autonomous system boundary (ASB) router* has links to another autonomous system.

OSPF Areas



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

Neighbor, Adjacent & Designated Routers

- *Neighbor routers*: two routers that have interfaces to a common network
  - Neighbors are discovered dynamically by *Hello protocol*
- Each neighbor of a router described by a state
  - down, attempt, init, 2-way, Ex-Start, Exchange, Loading, Full
- *Adjacent router*: neighbor routers become adjacent when they synchronize topology databases by exchange of link state information
  - Neighbors on point-to-point links become adjacent
  - Routers on multiaccess nets become adjacent only to *designated & backup designated routers*
    - Reduces size of topological database & routing traffic

Designated Routers

- Reduces number of adjacencies
- Elected by each multiaccess network after neighbor discovery by hello protocol
- Election based on priority & id fields
- Generates link advertisements that list routers attached to a multi-access network
- Forms adjacencies with routers on multi-access network
- Backup prepared to take over if designated router fails

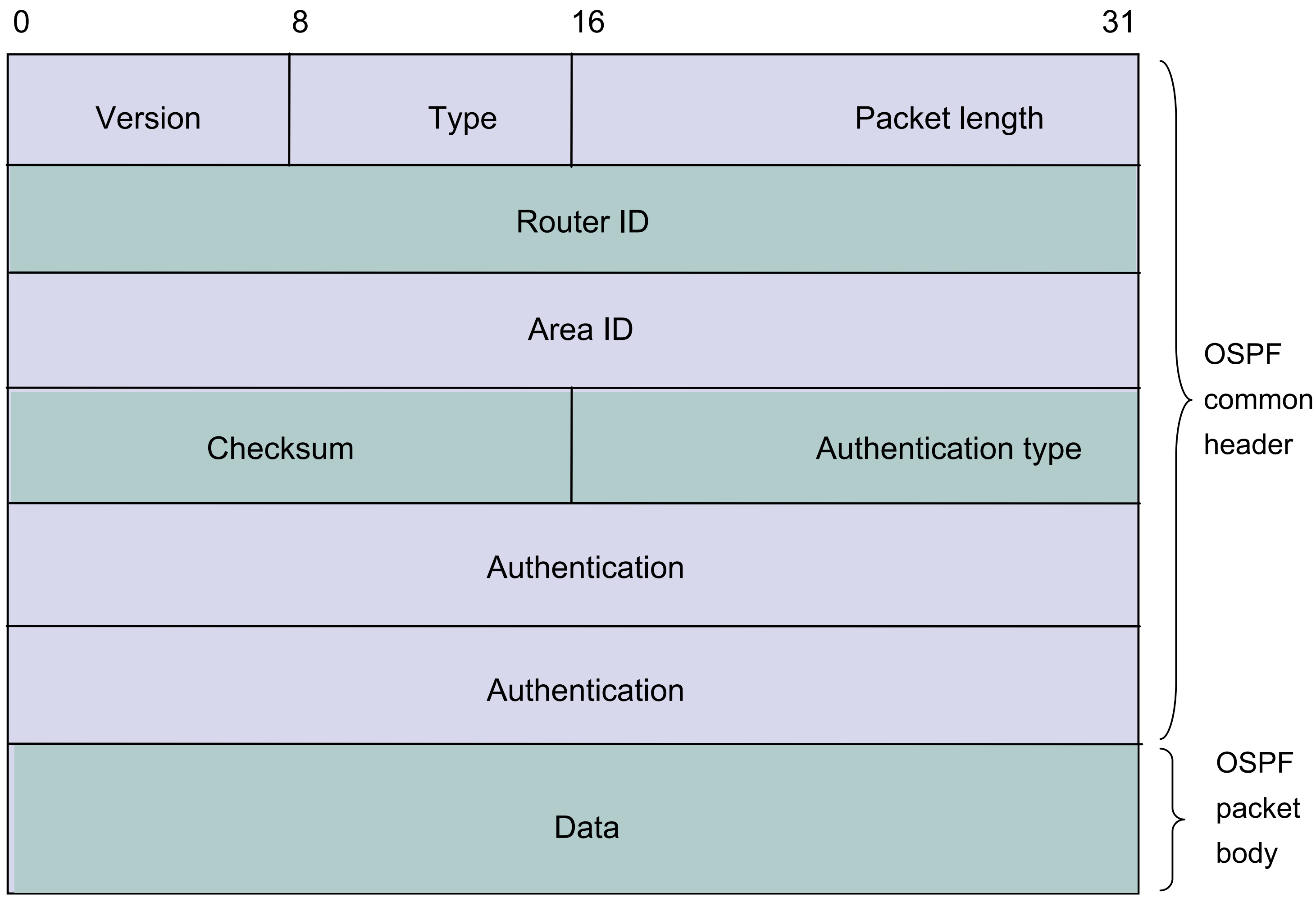
Link State Advertisements

- Link state info exchanged by adjacent routers to allow
  - area topology databases to be maintained
  - inter-area & inter-AS routes to be advertised
- *Router link ad*: generated by all OSPF routers
  - state of router links within area; flooded within area only
- *Net link ad*: generated by the designated router
  - lists routers connected to net: flooded within area only
- *Summary link ad*: generated by area border routers
  - 1. routes to dest in other areas; 2. routes to ASB routers
- *AS external link ad*: generated by ASB routers
  - describes routes to destinations outside the OSPF net
  - flooded in all areas in the OSPF net

OSPF Protocol

- OSPF packets transmitted directly on IP datagrams; Protocol ID 89
- TOS 0, IP precedence field set to internetwork control to get precedence over normal traffic
- OSPF packets sent to multicast address 224.0.0.5 (allSPFRouters on pt-2-pt and broadcast nets)
- OSPF packets sent on specific IP addresses on non-broadcast nets
- Five OSPF packet types:
  - *Hello*
  - *Database description*
  - *Link state request; Link state update; Link state ack*

OSPF Header



- Type: Hello, Database description, Link state request, Link state update, Link state acknowledgements

OSPF Stages

1. Discover neighbors by sending Hello packets (every 10 sec) and designated router elected in multiaccess networks
2. Adjacencies are established & link state databases are synchronized
3. Link state information is propagated & routing tables are calculated

We elaborate on OSPF stages in following

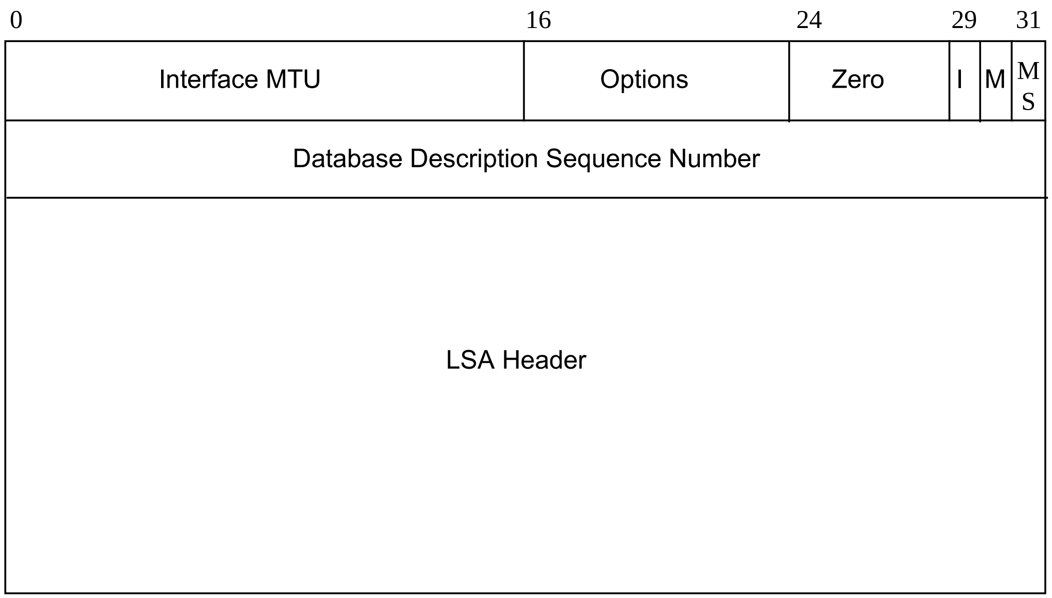
Stage 1: OSPF Hello Packet

0	16	24 31
Network mask		
Hello interval	Options	Priority
Dead interval		
Designated router		
Backup designated router		
Neighbor 1		
. . .		
Neighbor n		

- Send Hello packets to establish & maintain neighbor relationship

- Hello interval: number of seconds between Hello packets
- Priority: used to elect designated router & backup
- Dead interval: # sec before declaring a non-responding neighbor down.
- Neighbor: the Router ID of each neighbor from whom Hello packets have recently been received

Stage 2: OSPF Database Description



- Once neighbor routers become adjacent, they exchange database description packets to synchronize their link-state databases.

- Init bit 1 if pkt is first in sequence of database description packets
- More bit 1 if there are more database description packets to follow
- Master/Slave bit indicates that the router is the master.
- Link state ad (LSA) header describes state of router or network; contains info to uniquely identify entry in LSA (type, ID, and advertising router).
- Can have multiple LSA headers

LSA Header

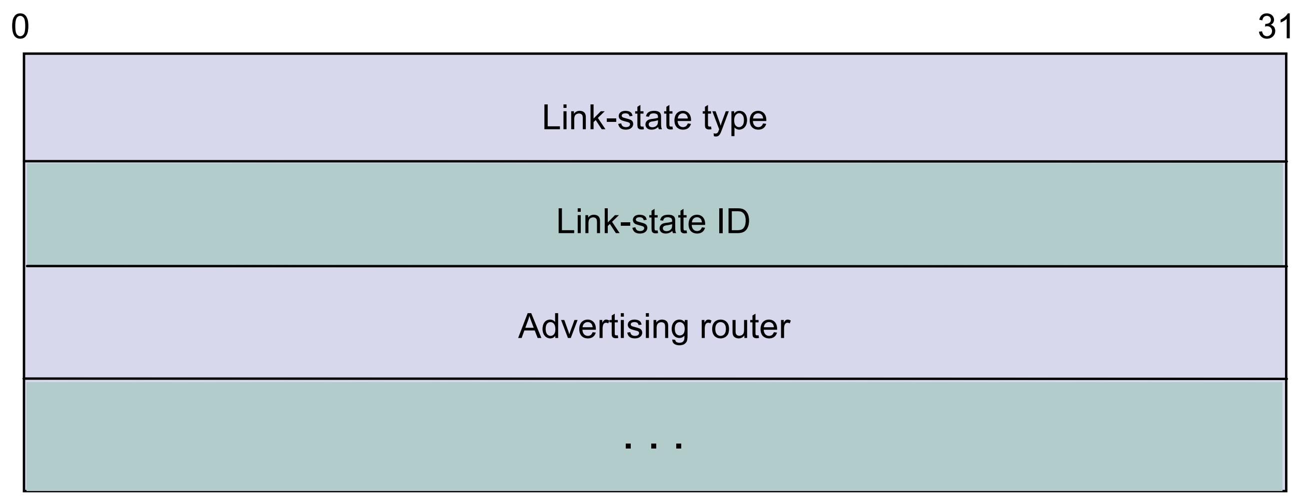
0	16	24	31
Link-state age	Options	Link-state type	
Link-state ID			
Advertising router			
Link-state sequence number			
Link-state checksum	Length		

- LS type: Router LSAs generated by all OSPF routers; Network LSAs generated by designated routers; Summary LSAs by area border routers; AS-external LSAs by ASBRs
- LS id: identifies piece of routing domain being described by LSA
- LS Seq. Number: numbers LSAs to detect old/duplicate LSAs
- LS checksum: covers contents of LSA except link state age

Database Synchronization

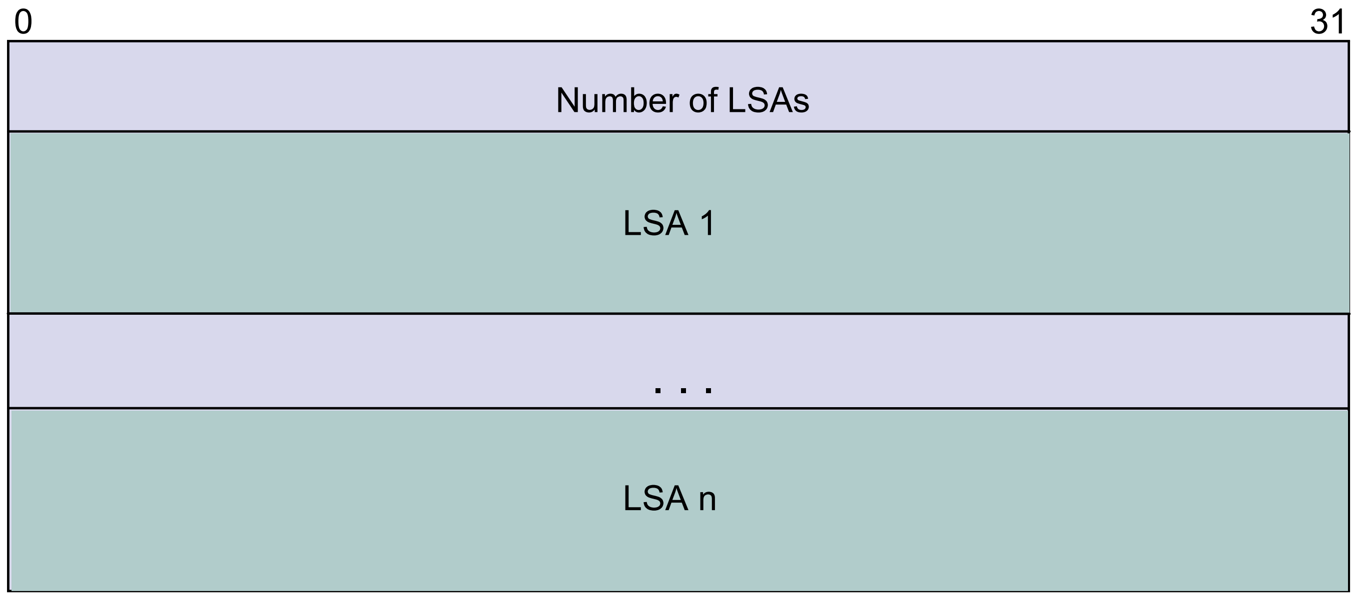
- LS Database (LSDB): collection of the Link State Advertisements (LSAs) accepted at a node.
  - This is the "map" for Dijkstra algorithm
- When the connection between two neighbors comes up, the routers must wait for their LSDBs to be synchronized.
  - Else routing loops and black holes due to inconsistency
- OSPF technique:
  - Source sends only LSA headers, then
  - Neighbor requests LSAs that are more recent
  - Those LSAs are sent over
  - After sync, the neighbors are said to be "fully adjacent"

Stage 3: OSPF Link State Request

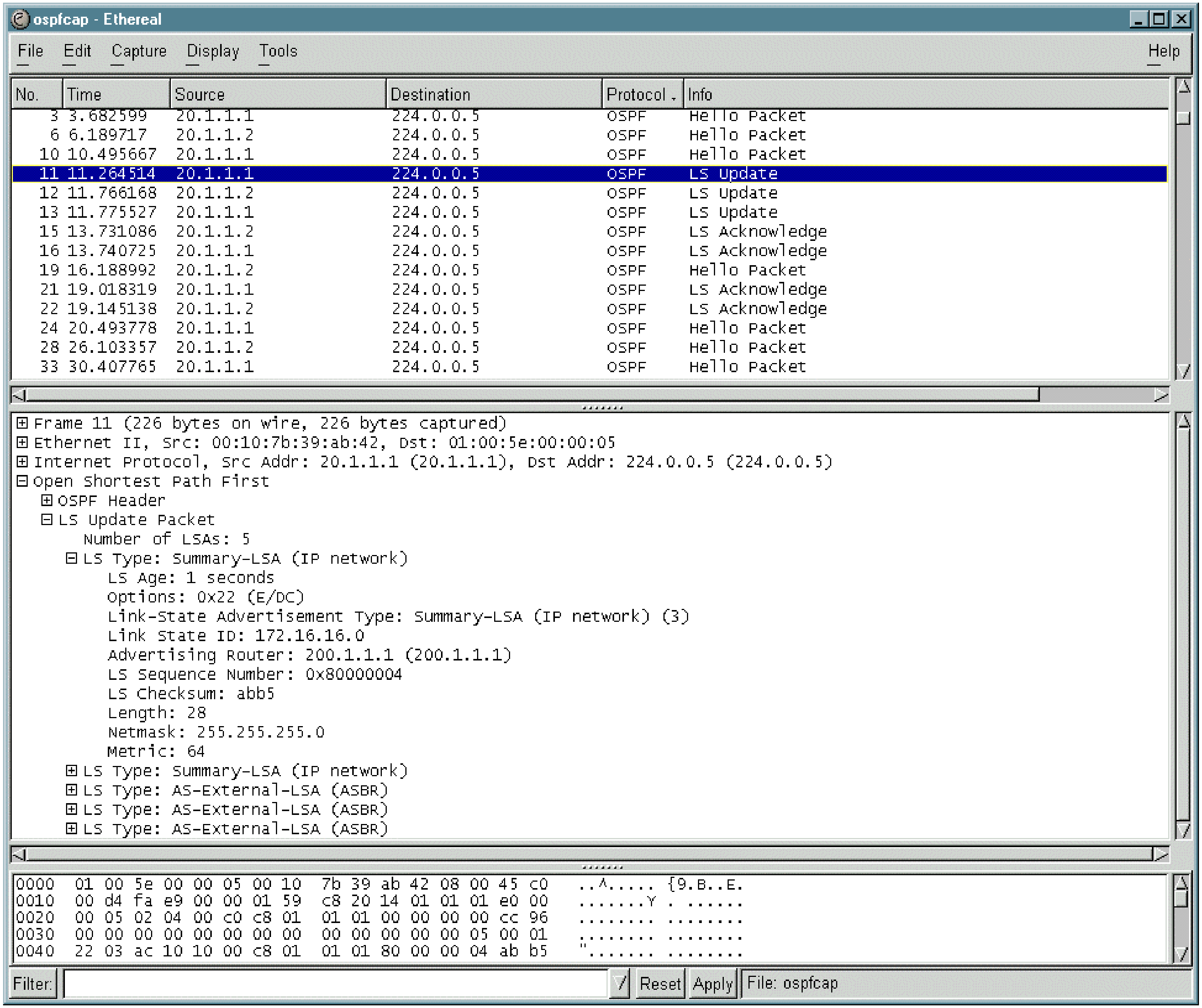


- Router sends a LS request packet to neighbor to update part of its link-state database
- Each LSA request is specified by the link state type, link state ID, and the advertising router.

OSPF Link State Update



- In response to LS request or trigger, router will send new LS info using the LS update message
- Contents are composed of link state advertisements (LSAs)
- LS update message is acknowledged using LS ack pkt to ensure that the flooding algorithm is reliable; Link state acknowledgement packets consist of a list of LSA headers.


ospfcap - Ethereal
File Edit Capture Display Tools Help
No. Time Source Destination Protocol Info
3 3.682599 20.1.1.1 224.0.0.5 OSPF Hello Packet
6 6.189717 20.1.1.2 224.0.0.5 OSPF Hello Packet
10 10.495667 20.1.1.1 224.0.0.5 OSPF Hello Packet
11 11.264514 20.1.1.1 224.0.0.5 OSPF LS Update
12 11.766168 20.1.1.2 224.0.0.5 OSPF LS Update
13 11.775527 20.1.1.1 224.0.0.5 OSPF LS Update
15 13.731086 20.1.1.2 224.0.0.5 OSPF LS Acknowledge
16 13.740725 20.1.1.1 224.0.0.5 OSPF LS Acknowledge
19 16.188992 20.1.1.2 224.0.0.5 OSPF Hello Packet
21 19.018319 20.1.1.1 224.0.0.5 OSPF LS Acknowledge
22 19.145138 20.1.1.2 224.0.0.5 OSPF LS Acknowledge
24 20.493778 20.1.1.1 224.0.0.5 OSPF Hello Packet
28 26.103357 20.1.1.2 224.0.0.5 OSPF Hello Packet
33 30.407765 20.1.1.1 224.0.0.5 OSPF Hello Packet
Frame 11 (226 bytes on wire, 226 bytes captured)
Ethernet II, Src: 00:10:7b:39:ab:42, Dst: 01:00:5e:00:00:05
Internet Protocol, Src Addr: 20.1.1.1 (20.1.1.1), Dst Addr: 224.0.0.5 (224.0.0.5)
Open Shortest Path First
OSPF Header
LS Update Packet
Number of LSAs: 5
LS Type: Summary-LSA (IP network)
LS Age: 1 seconds
Options: 0x22 (E/DC)
Link-State Advertisement Type: Summary-LSA (IP network) (3)
Link State ID: 172.16.16.0
Advertising Router: 200.1.1.1 (200.1.1.1)
LS Sequence Number: 0x80000004
LS Checksum: abb5
Length: 28
Netmask: 255.255.255.0
Metric: 64
LS Type: Summary-LSA (IP network)
LS Type: AS-External-LSA (ASBR)
LS Type: AS-External-LSA (ASBR)
LS Type: AS-External-LSA (ASBR)
0000 01 00 5e 00 00 05 00 10 7b 39 ab 42 08 00 45 c0 ..^..... {9.B..E.
0010 00 d4 fa e9 00 00 01 59 c8 20 14 01 01 01 e0 00 ....Y . ...
0020 00 05 02 04 00 c0 c8 01 01 01 00 00 00 00 cc 96 ....
0030 00 00 00 00 00 00 00 00 00 00 00 00 00 05 00 01 ....
0040 22 03 ac 10 10 00 c8 01 01 01 80 00 00 04 ab b5 "....... ....
Filter: Reset Apply File: ospfcap

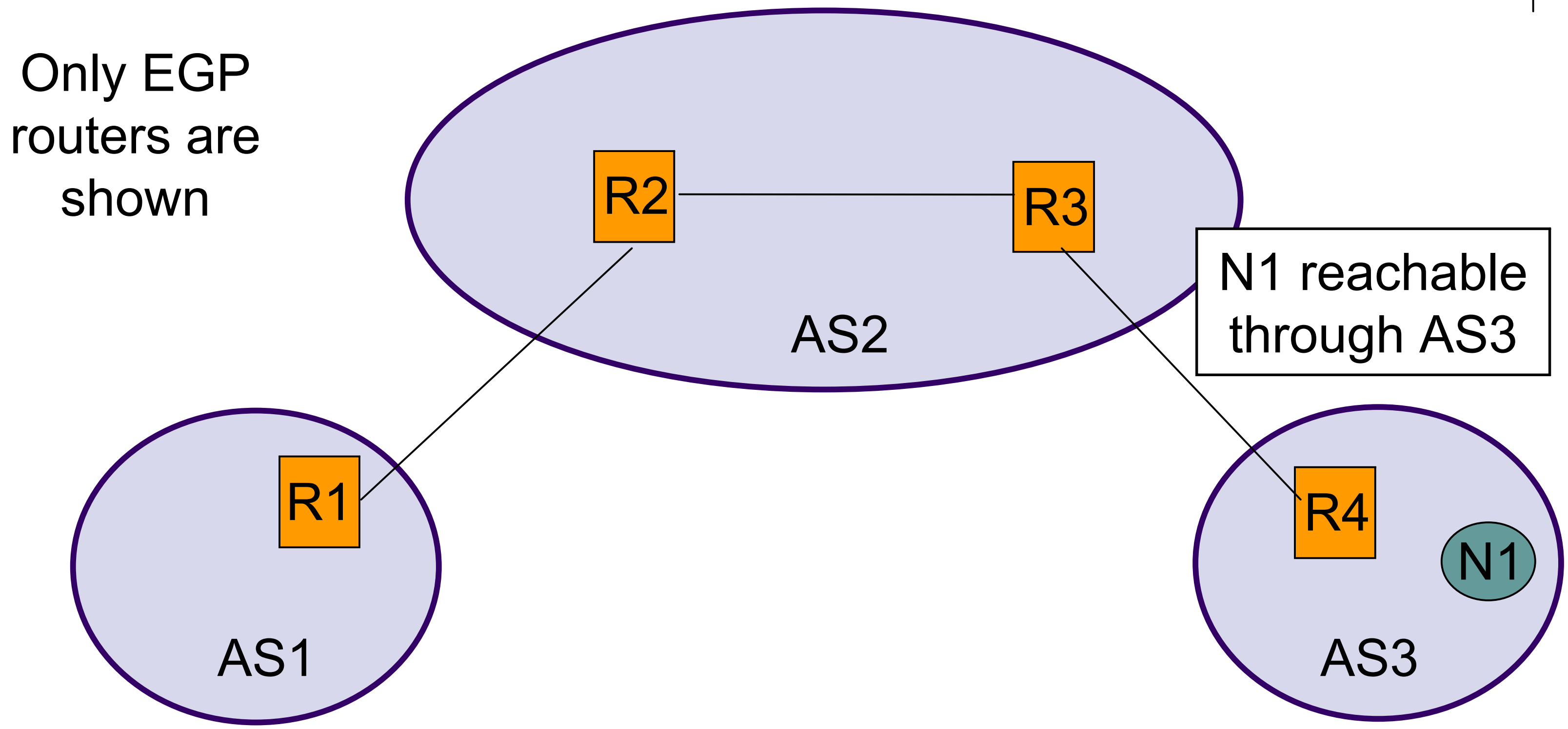
Outline

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

Exterior Gateway Protocols

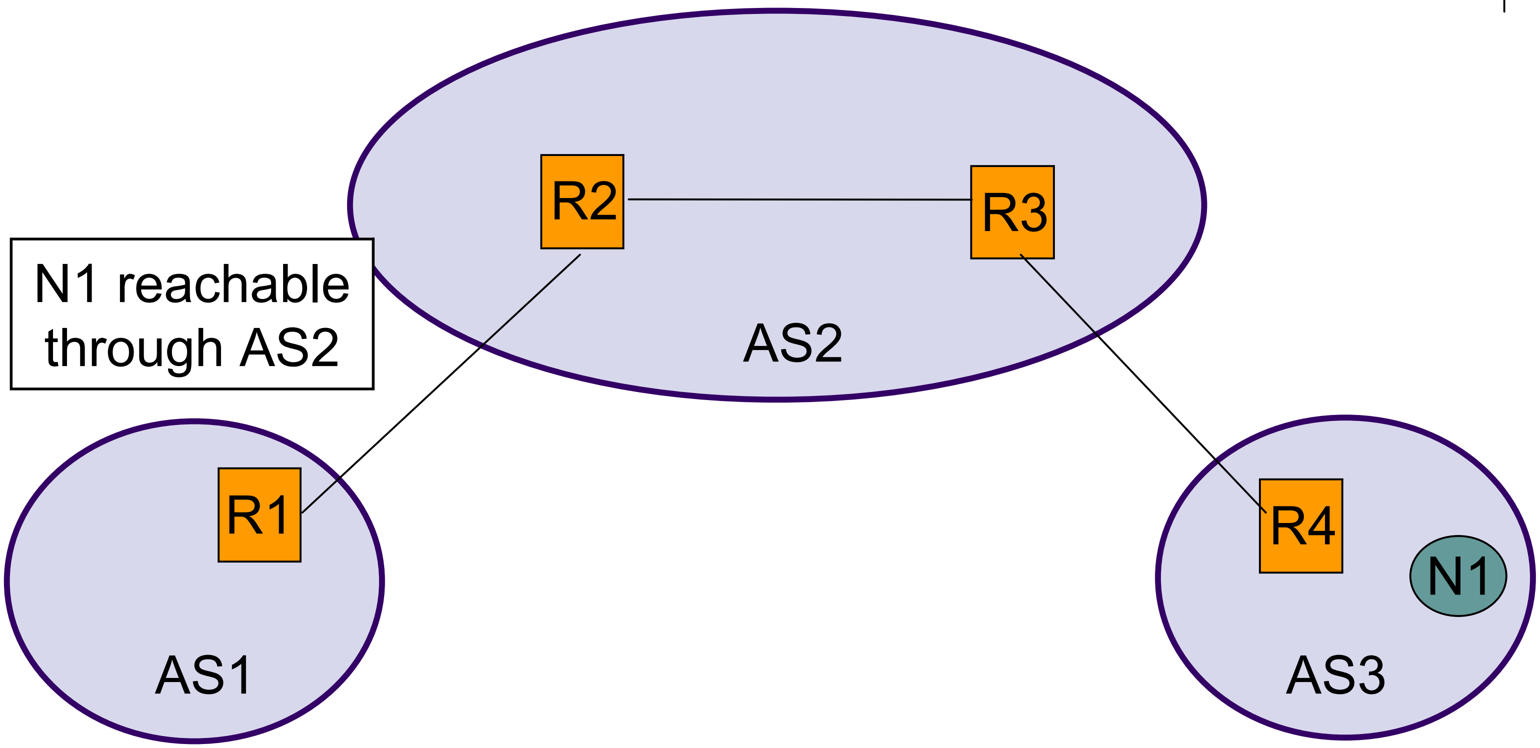
- Within each AS, there is a consistent set of routes connecting the constituent networks
- The Internet is woven into a coherent whole by *Exterior Gateway Protocols (EGPs)* that operate between AS's
- EGP enables two AS's to exchange routing information about:
  - The networks that are contained within 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



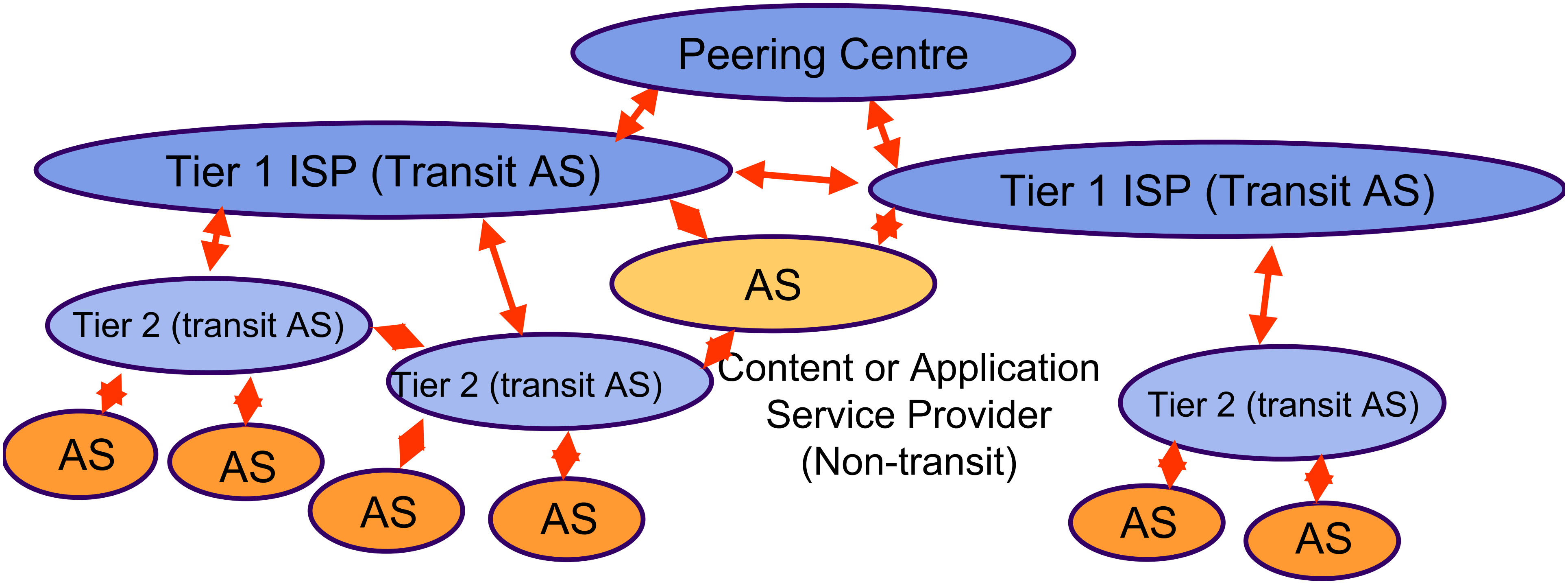
- R4 advertises that network N1 can be reached through AS3
- R3 examines announcement & applies *policy* to decide whether it will forward packets to N1 through R4
- If yes, routing table updated in R3 to indicate R4 as next hop to N1
- IGP propagates N1 reachability information through AS2

EGP Example



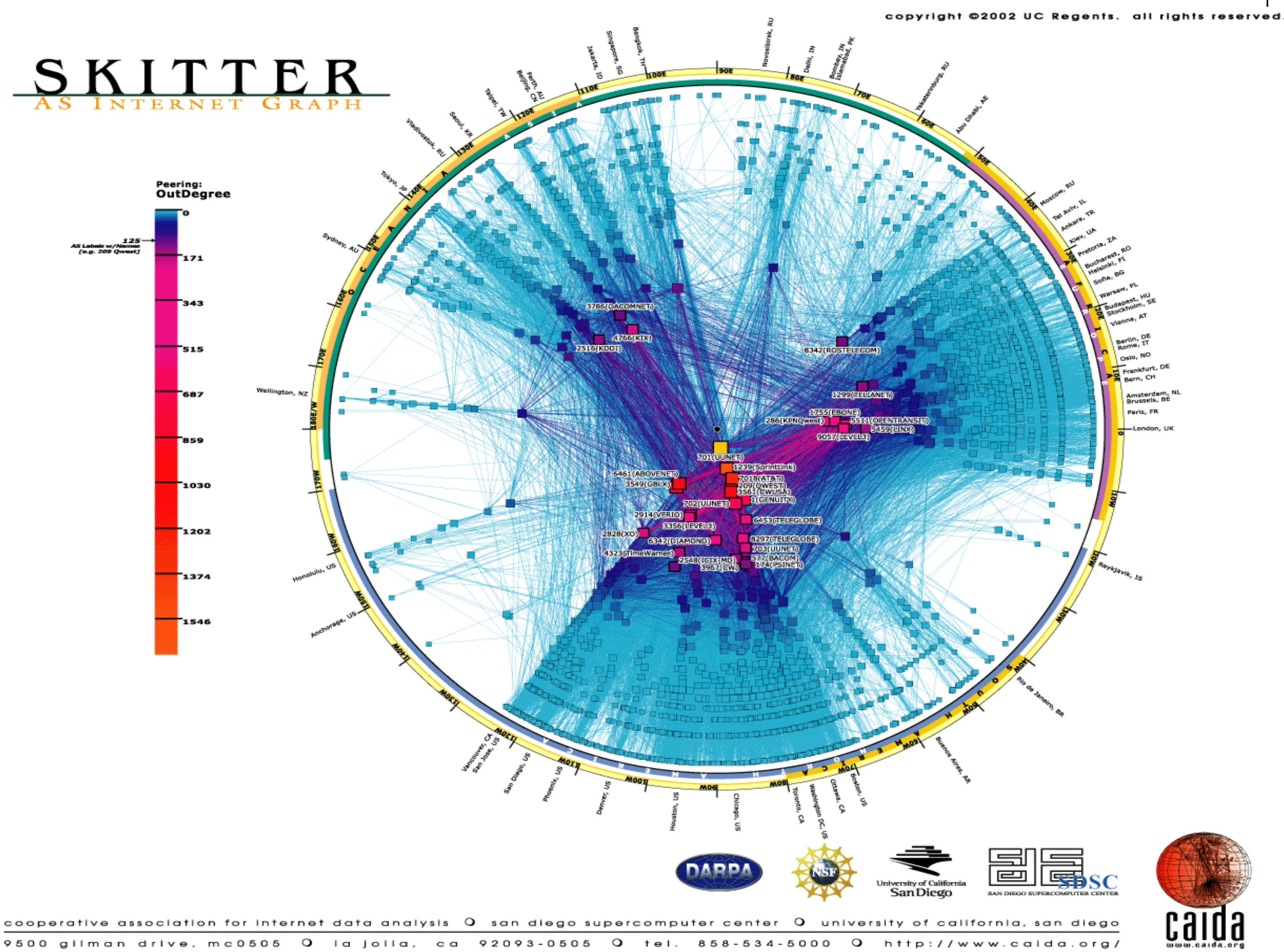
- EGP routers within an AS, e.g. R3 and R2, are kept consistent
- Suppose AS2 willing to handle *transit* packets from AS1 to N1
- R2 advertises to AS1 the reachability of N1 through AS2
- R1 applies its policy to decide whether to send to N1 via AS2

Peering and Inter-AS connectivity



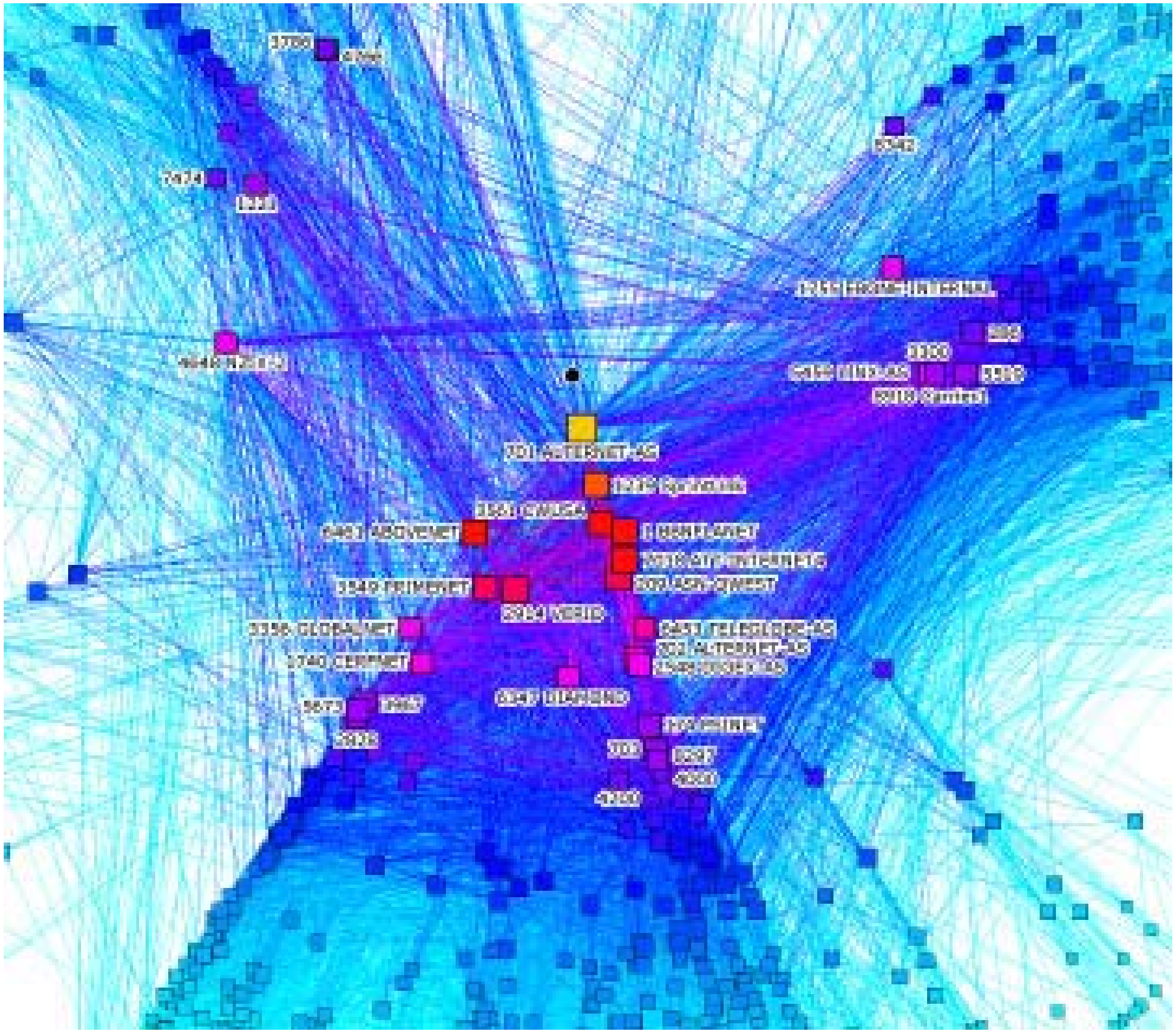
- Non-transit AS's (stub & multihomed) do not carry transit traffic
- Tier 1 ISPs peer with each other, privately & peering centers
- Tier 2 ISPs peer with each other & obtain transit services from Tier 1s; Tier 1's carry transit traffic between their Tier 2 customers
- Client AS's obtain service from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

CAIDA Internet AS Map



CAIDA is a program of the University of California's San Diego Supercomputer Center (UCSD/SDSC)
skitter is supported by DARPA NGI Cooperative Agreement N66001-98-2-8922, NSF ANIR Grant NCR-9711092 and CAIDA members

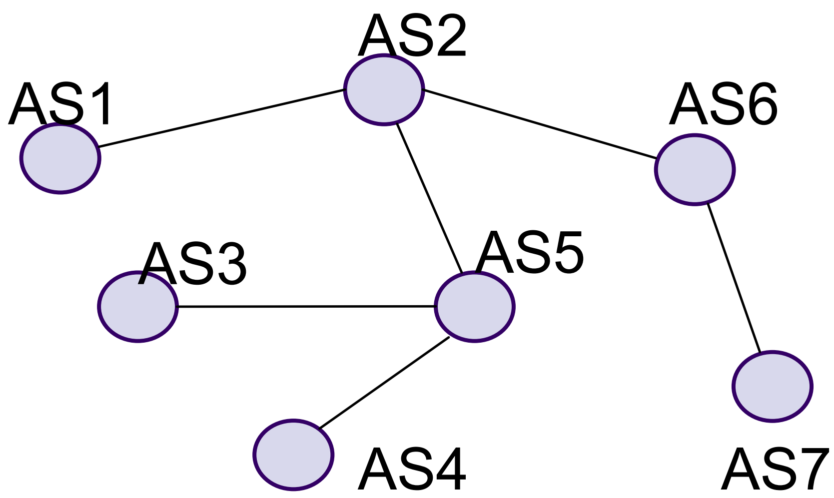
Zoom into Internet AS Map



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

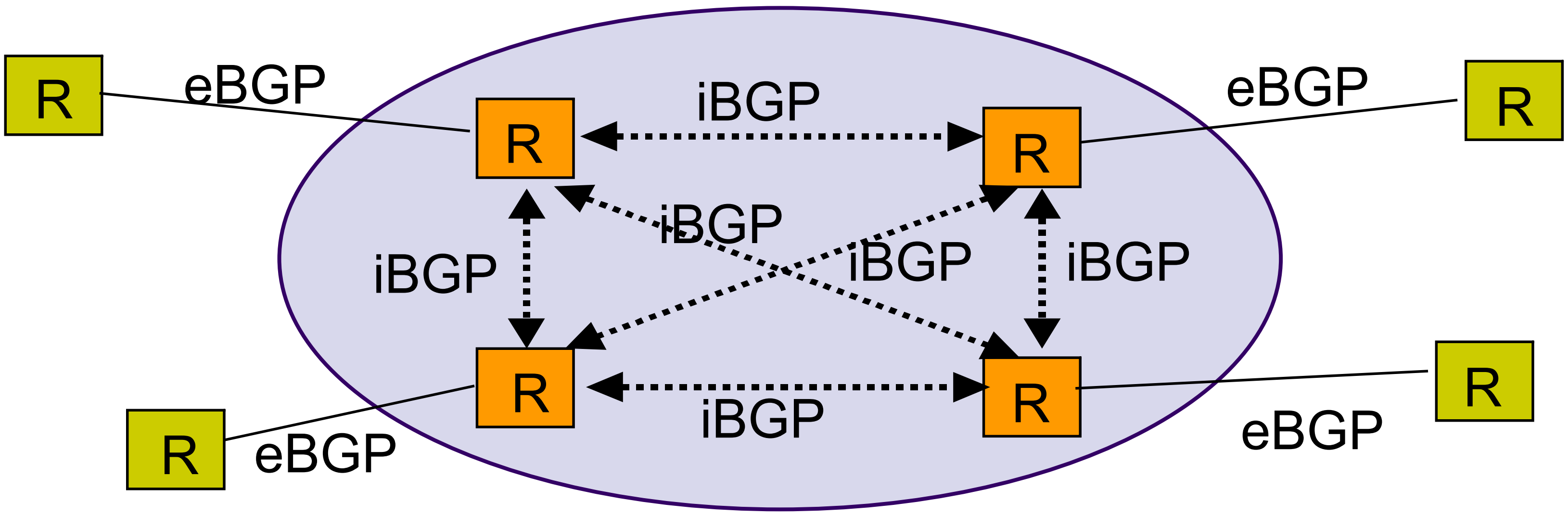
BGP Features

- 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

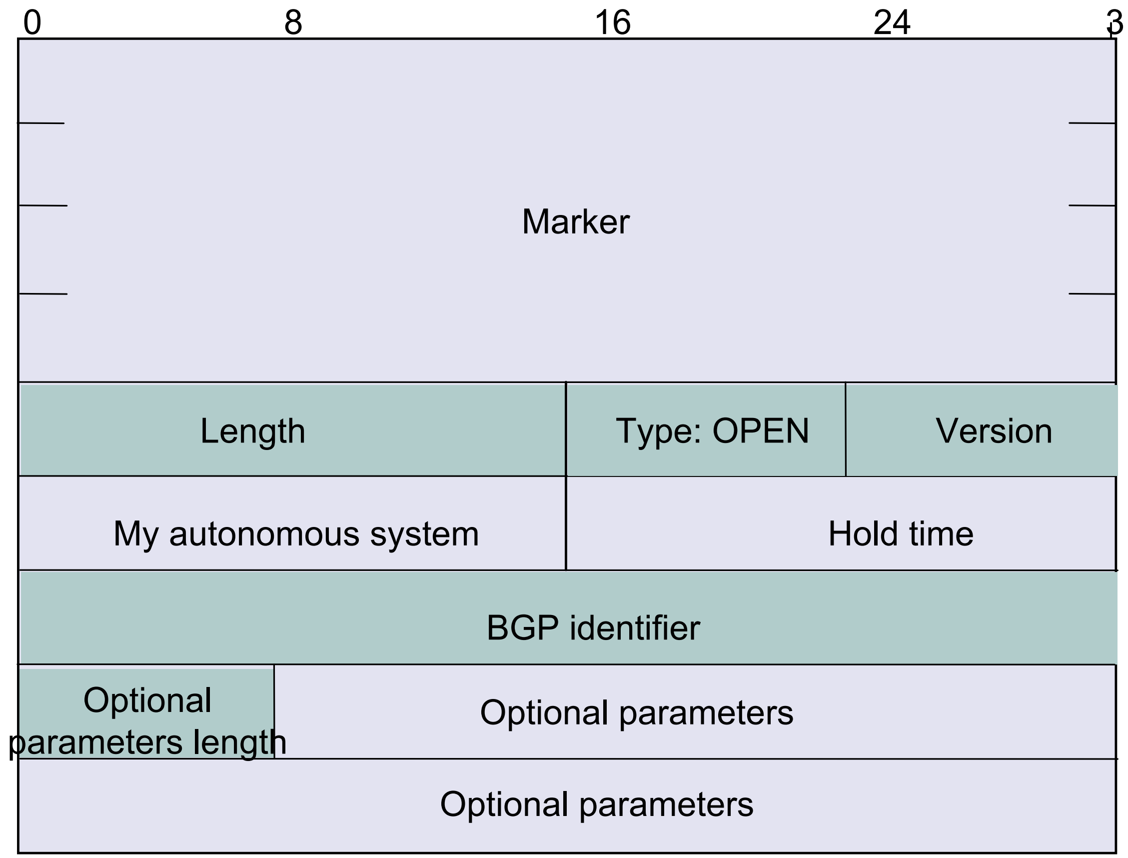
BGP Policy

- 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

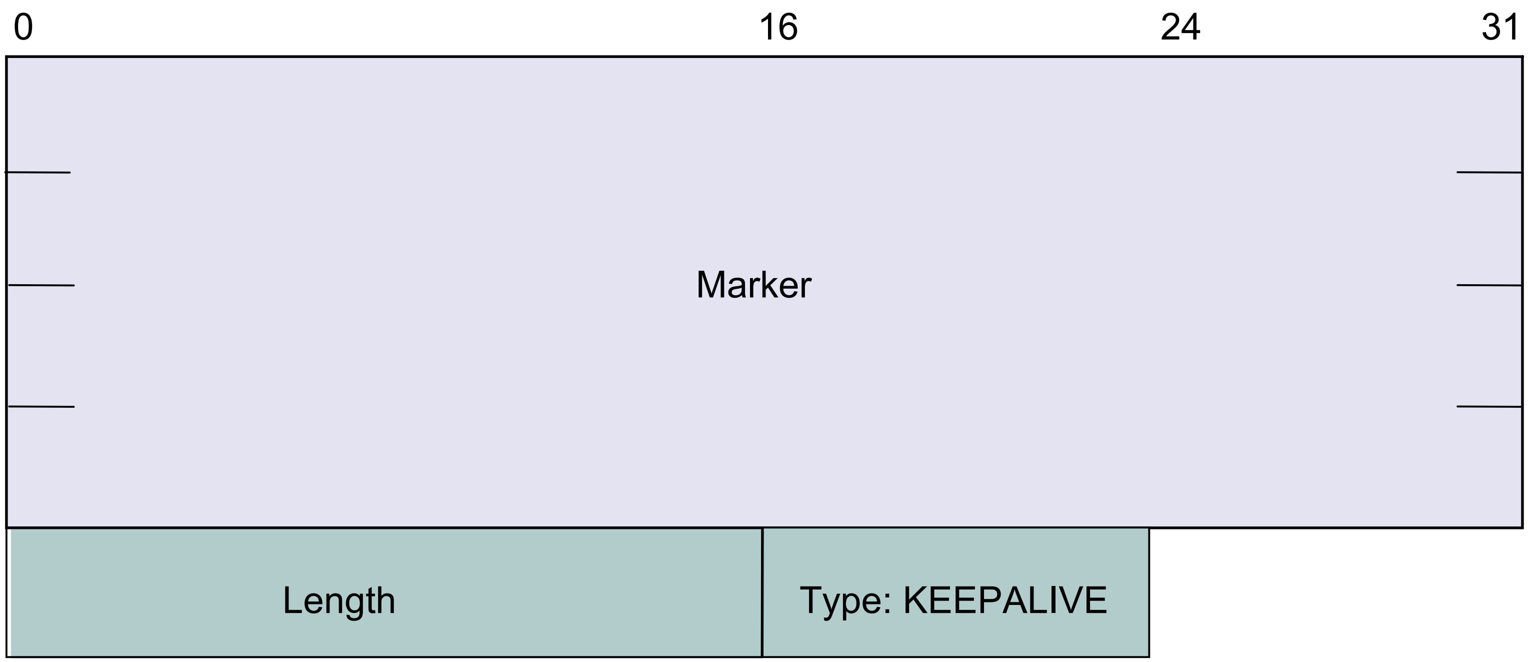
- Opening & confirming of a BGP connection with a neighbor router
- Maintaining the BGP connection
- Sending reachability information
- Notification of error conditions

BGP Open Message



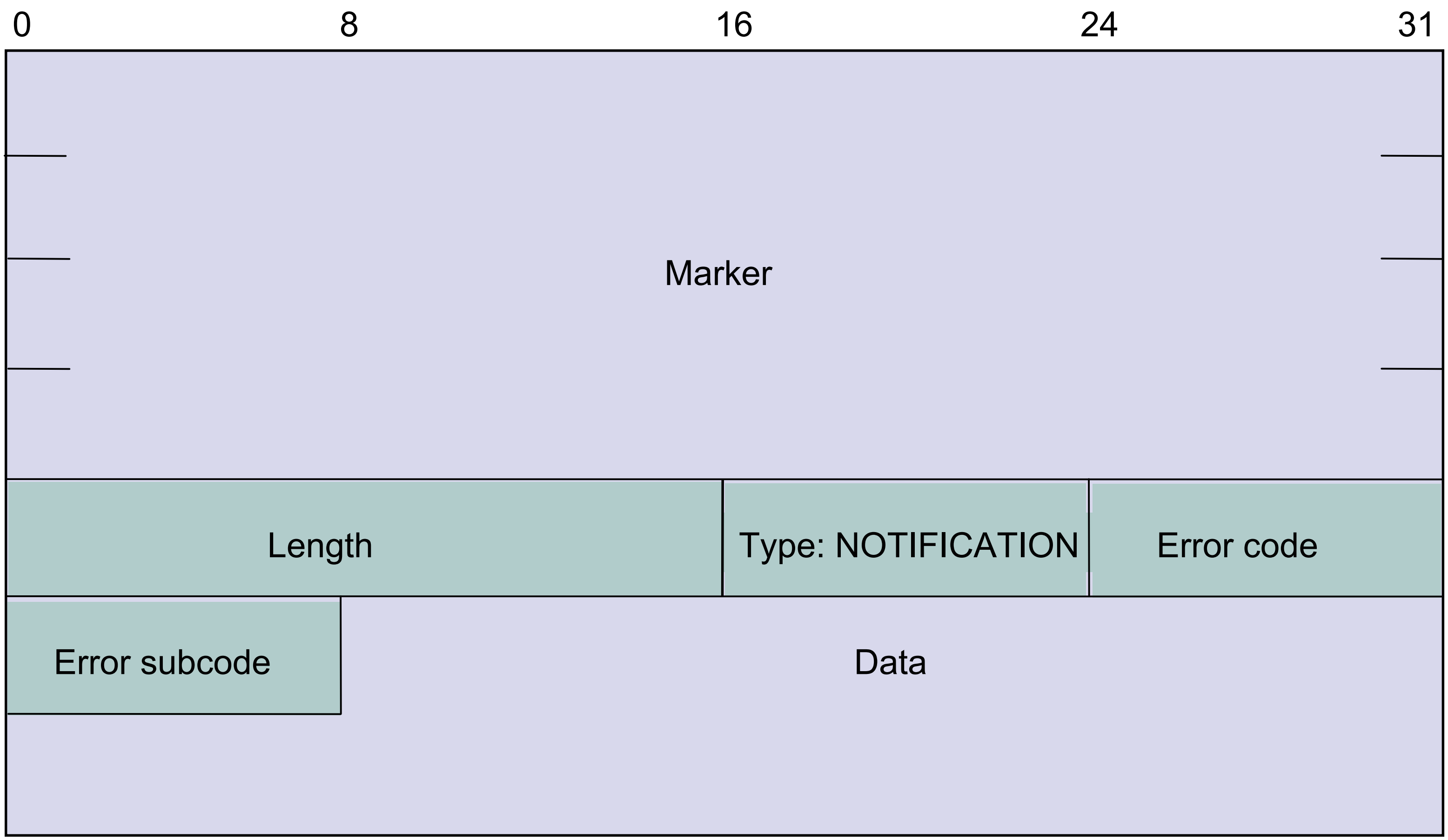
- Marker: authenticates incoming BGP messages or detects loss of synchronization between a pair of BGP peers.
- Hold time: to propose number of seconds between transmission of successive KEEPALIVE messages
- BGP ID: identifies sending BGP router; value is determined by one of the IP local interface addresses of the BGP router

BGP Keep-Alive Message



- BGP speakers continuously monitor the reachability of the peers by exchanging the KEEPALIVE messages periodically
- KEEPALIVE message is BGP header with the type field set to 4

BGP Notification Message



- When BGP speaker detects error or exception, it sends NOTIFICATION message and then closes TCP connection
- Error Code indicates the type of error condition

Update Message

Unfeasible routes length (two octets)
Withdrawn routes (variable)
Total path attribute length (two octets)
Path attributes (variable)
Network layer reachability information (variable)

- After connection established, BGP peers exchange routing information using the UPDATE messages
- UPDATE messages used to construct a graph of AS connectivity
- Info: Unfeasible Routes, Path Attributes; Network Layer Reachability Information (NLRI).
- UPDATE message can advertise single route &/or withdraw list of routes.

Route Advertisement

- BGP router uses NLRI, Total Path Attributes Length, and Path Attributes, to advertise a route
- *NLRI* contains list of IP address prefixes that can be reached by the route
- *Path Attributes* describe characteristics of the route and is used to affect routing behavior
- UPDATE message has a variable length sequence of path attributes. Each path attribute is a triple

<Attribute Type, Attribute Length, Attribute Value>

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.

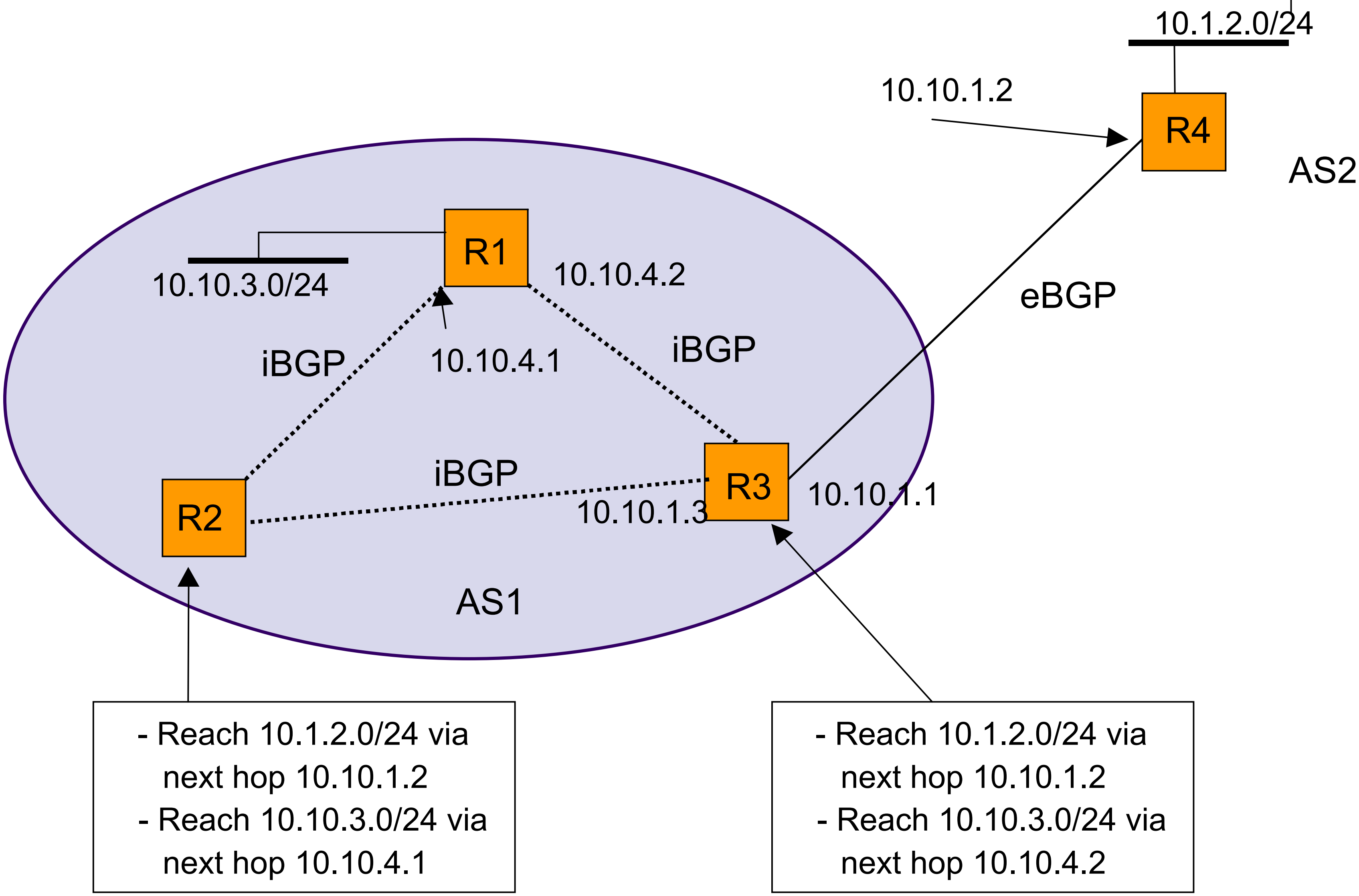
MULTI_EXIT_DISC: used to discriminate among multiple entry/exit points to neighboring AS and to hint about the preferred path.

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

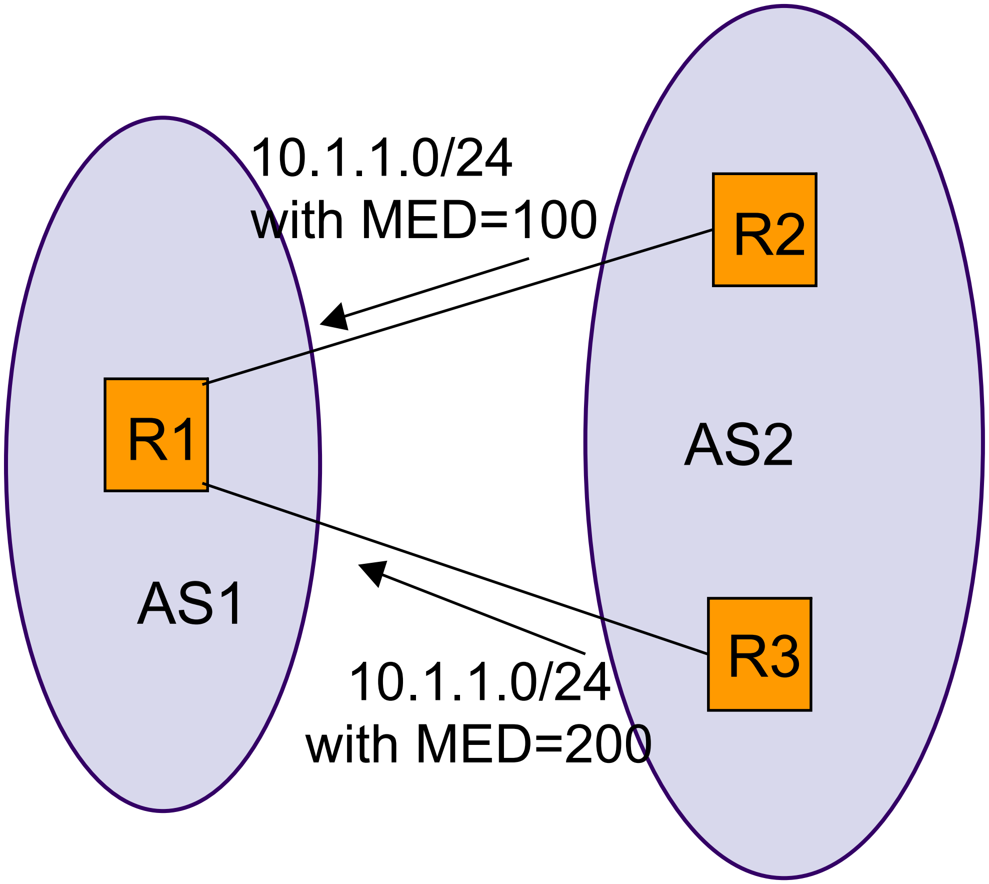
ATOMIC_AGGREGATE: informs other BGP speakers that it selected a less specific route without selecting a more specific one which is included in it.

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

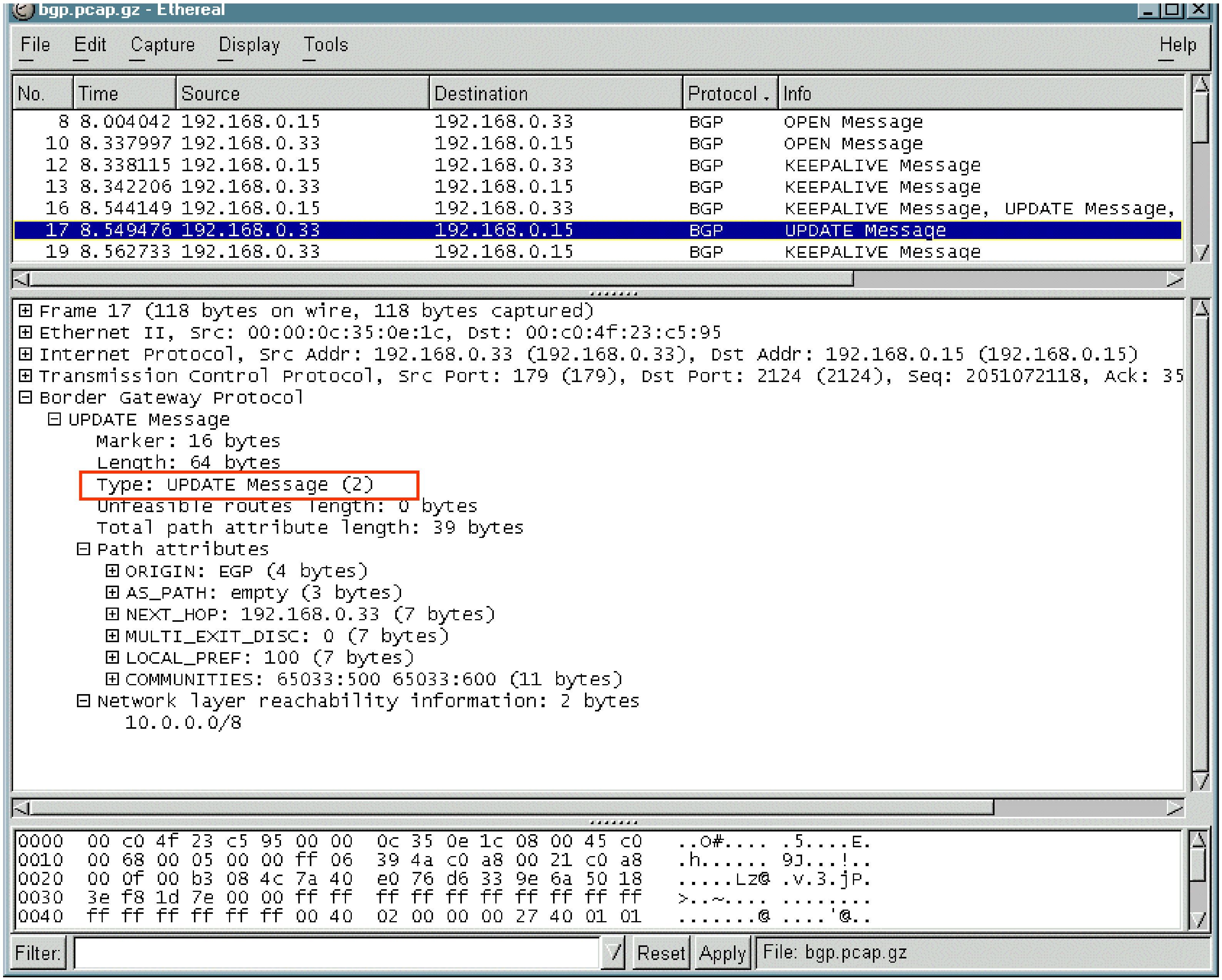
BGP NEXT_HOP


10.1.2.0/24
10.10.1.2
R4
AS2
eBGP
R1
10.10.3.0/24
10.10.4.2
iBGP
10.10.4.1
iBGP
iBGP
R2
R3
10.10.1.3
10.10.1.1
AS1
- Reach 10.1.2.0/24 via next hop 10.10.1.2
- Reach 10.10.3.0/24 via next hop 10.10.4.1
- Reach 10.1.2.0/24 via next hop 10.10.1.2
- Reach 10.10.3.0/24 via next hop 10.10.4.2

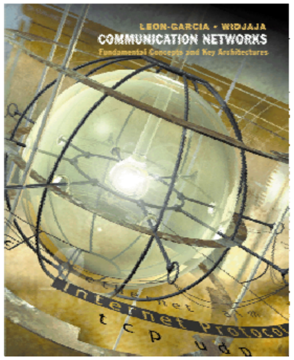
BGP MULTI_EXIT_DISC



- Multi-Exist Discriminator used to give another AS a hint as to which route is preferred
- In example, AS2 uses weight values to indicate to AS1 that it prefers that route via R2 be used

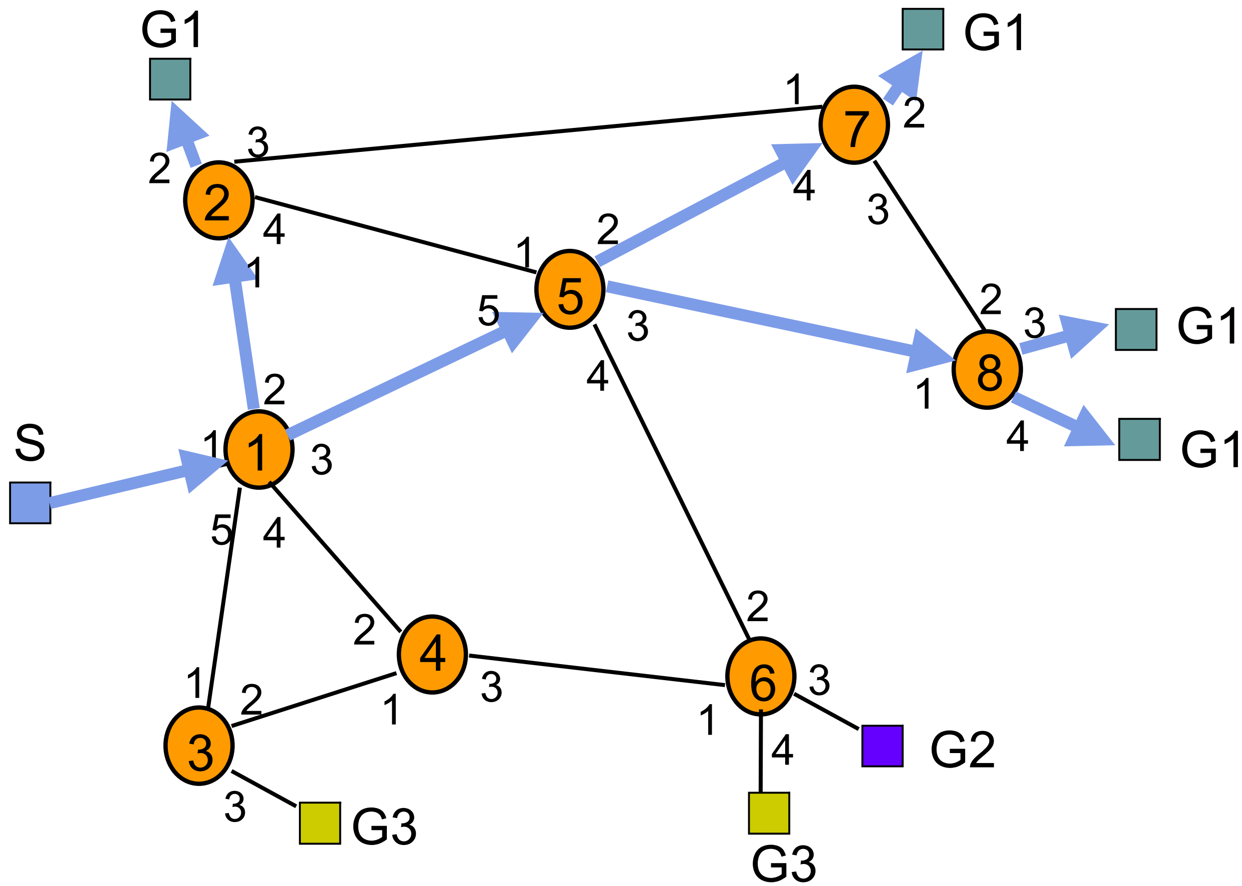

bgp.pcap.gz - Ethereal
File Edit Capture Display Tools Help
No. Time Source Destination Protocol Info
8 8.004042 192.168.0.15 192.168.0.33 BGP OPEN Message
10 8.337997 192.168.0.33 192.168.0.15 BGP OPEN Message
12 8.338115 192.168.0.15 192.168.0.33 BGP KEEPALIVE Message
13 8.342206 192.168.0.33 192.168.0.15 BGP KEEPALIVE Message
16 8.544149 192.168.0.15 192.168.0.33 BGP KEEPALIVE Message, UPDATE Message,
17 8.549476 192.168.0.33 192.168.0.15 BGP UPDATE Message
19 8.562733 192.168.0.33 192.168.0.15 BGP KEEPALIVE Message
Frame 17 (118 bytes on wire, 118 bytes captured)
Ethernet II, Src: 00:00:0c:35:0e:1c, Dst: 00:c0:4f:23:c5:95
Internet Protocol, Src Addr: 192.168.0.33 (192.168.0.33), Dst Addr: 192.168.0.15 (192.168.0.15)
Transmission Control Protocol, Src Port: 179 (179), Dst Port: 2124 (2124), Seq: 2051072118, Ack: 35
Border Gateway Protocol
UPDATE Message
Marker: 16 bytes
Length: 64 bytes
Type: UPDATE Message (2)
Unfeasible routes length: 0 bytes
Total path attribute length: 39 bytes
Path attributes
ORIGIN: EGP (4 bytes)
AS_PATH: empty (3 bytes)
NEXT_HOP: 192.168.0.33 (7 bytes)
MULTI_EXIT_DISC: 0 (7 bytes)
LOCAL_PREF: 100 (7 bytes)
COMMUNITIES: 65033:500 65033:600 (11 bytes)
Network layer reachability information: 2 bytes
10.0.0.0/8
0000 00 c0 4f 23 c5 95 00 00 0c 35 0e 1c 08 00 45 c0 ..O#.... .5....E.
0010 00 68 00 05 00 00 ff 06 39 4a c0 a8 00 21 c0 a8 .h...... 9J...!..
0020 00 0f 00 b3 08 4c 7a 40 e0 76 d6 33 9e 6a 50 18 .....Lz@ .v.3.jP.
0030 3e f8 1d 7e 00 00 ff ff ff ff ff ff ff ff ff ff >..~.... ....
0040 ff ff ff ff ff ff 00 40 02 00 00 00 27 40 01 01 ....@ ....'@..
Filter: Reset Apply File: bgp.pcap.gz

Chapter 8
Communication Networks and Services



Multicast Routing

Multicasting



- Source S sends packets to multicast group G1

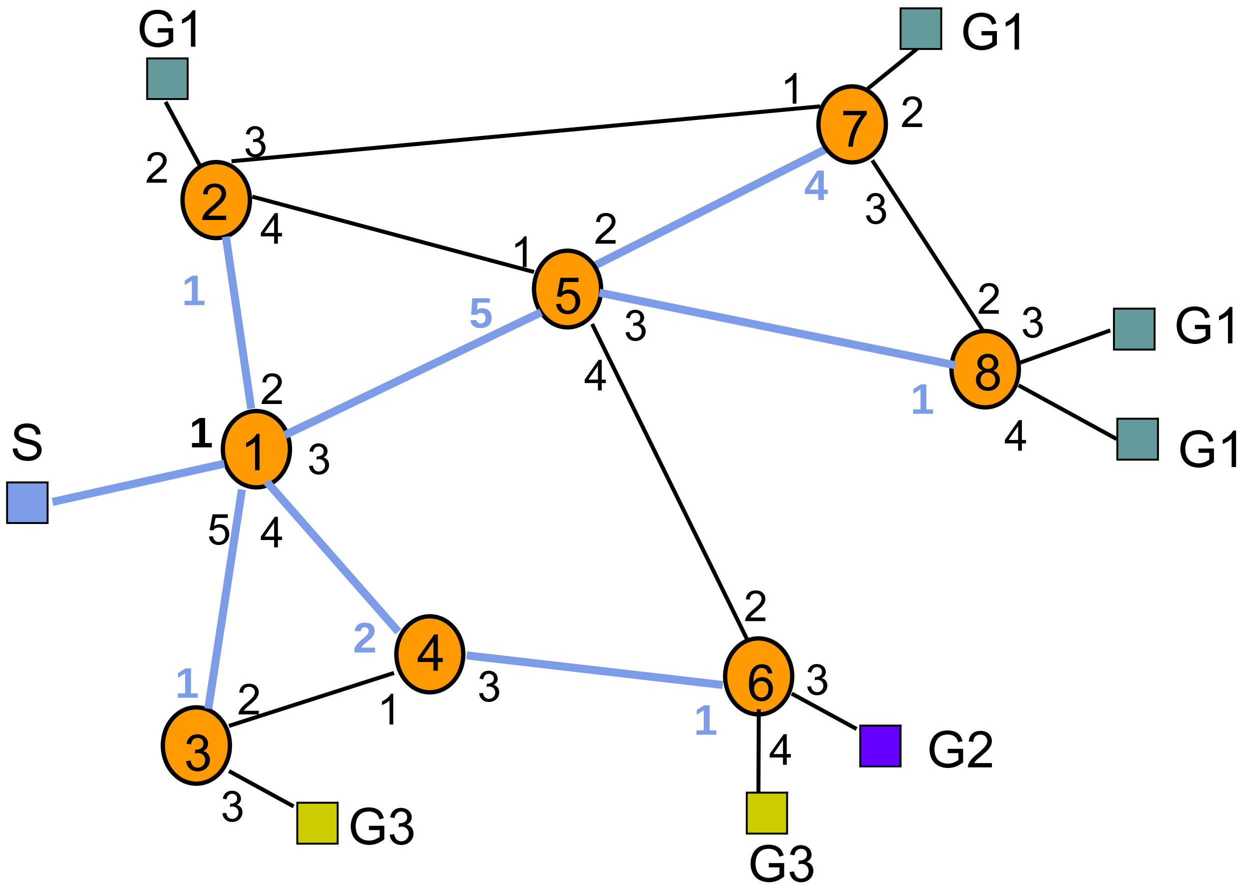
Multicast Routing

- Multicast routing useful when a source wants to transmits its packets to several destinations simultaneously
- Relying on unicast routing by transmitting each copy of packet separately works, but can be very inefficient if number of destination is large
- Typical applications is multi-party conferencing over the Internet
- Example: Multicast Backbone (MBONE) uses *reverse path multicasting*

Reverse-Path Broadcasting (RPB)

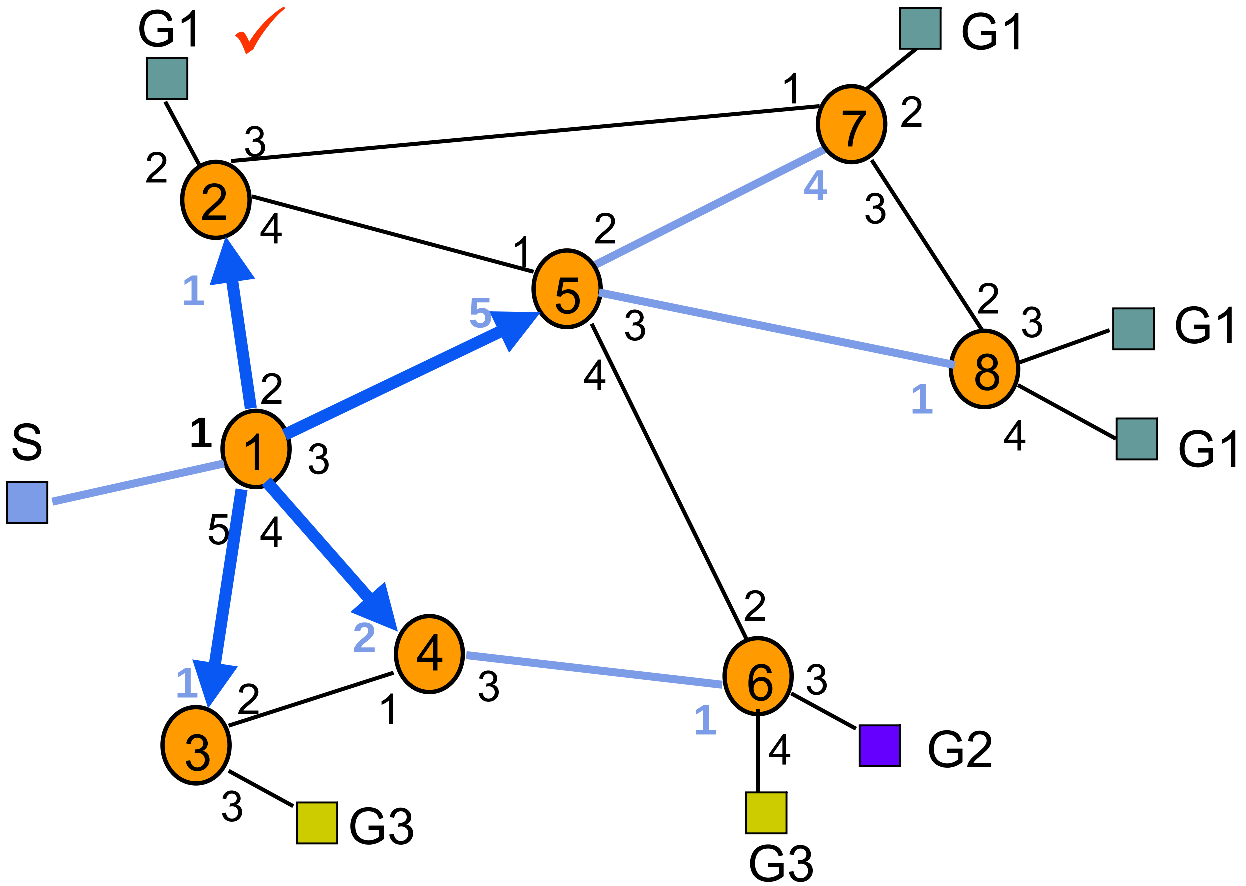
- Fact: Set of shortest paths to the source node S forms a tree that spans the network
  - Approach: Follow paths in *reverse* direction
- Assume each router knows current shortest path to S
  - Upon receipt of a multicast packet, router records the packet's source address and the port it arrives on
  - If shortest path to source is through same port ("parent port"), router forwards the packet to all other ports
  - Else, drops the packet
- Loops are suppressed; each packet forwarded a router exactly once
- Implicitly assume shortest path to source S is same as shortest path from source
  - If paths asymmetric, need to use link state info to compute shortest paths from S

Example: Shortest Paths from S



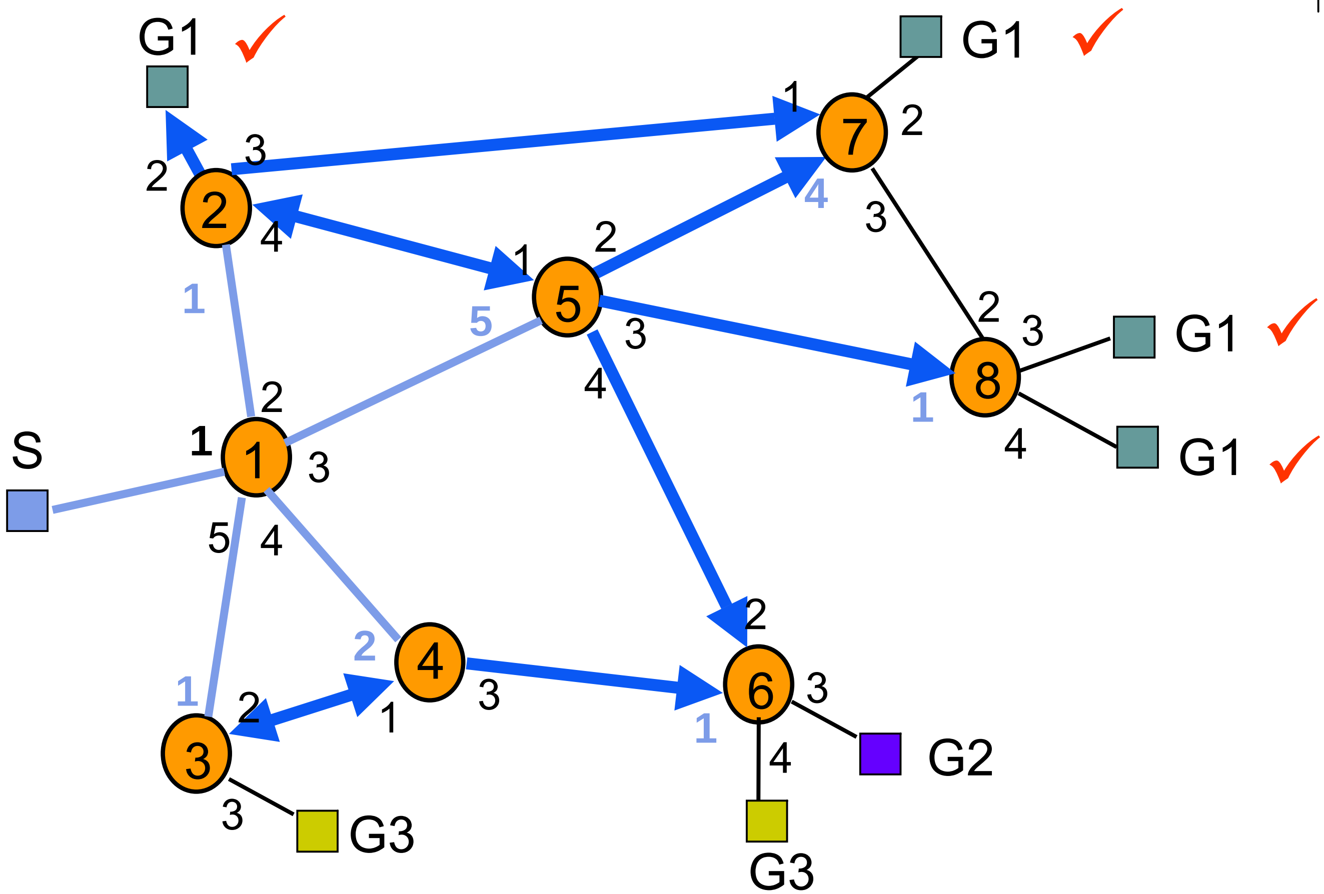
- Spanning tree of shortest paths to node S and parent ports are shown in blue

Example: S sends a packet



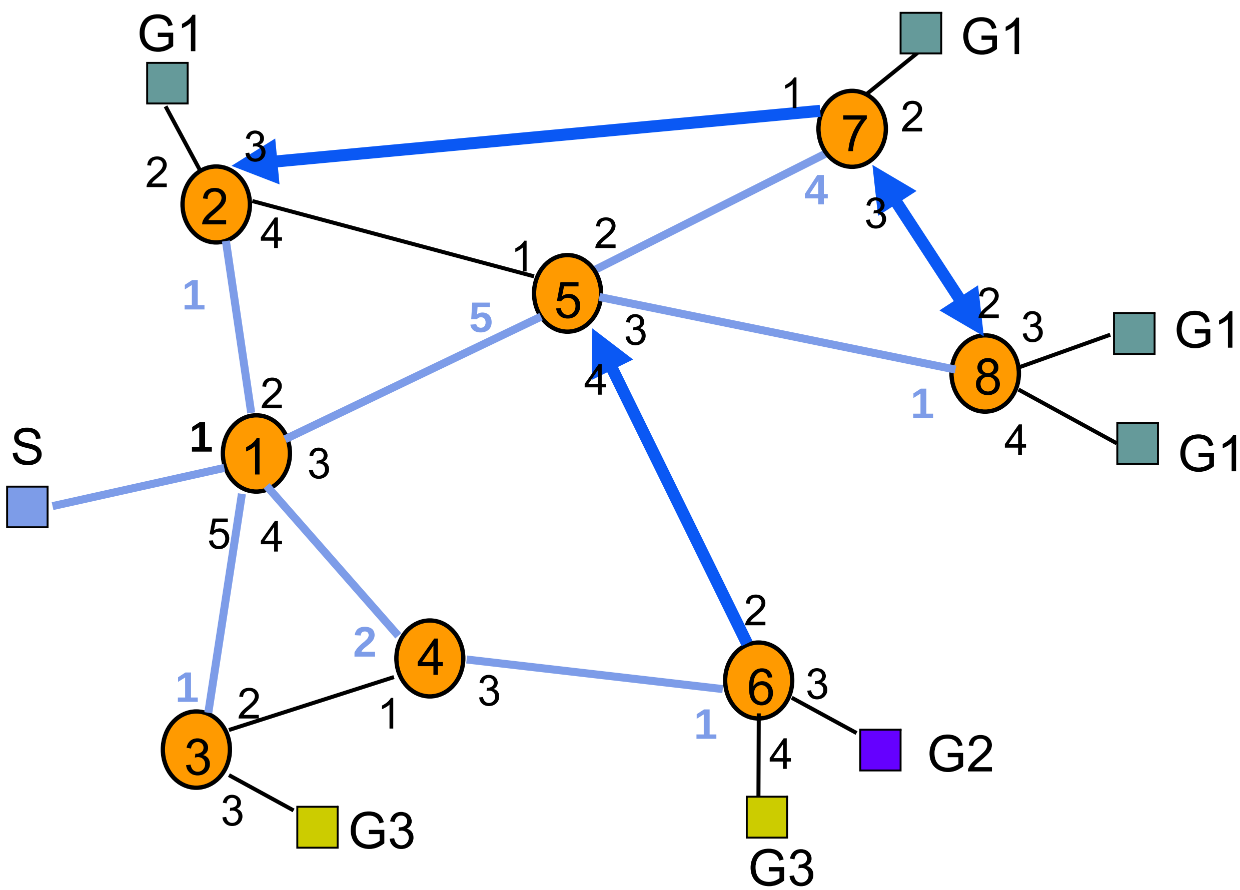
- S sends a packet to node 1
- Node 1 forwards to all ports, except parent port

Example: Hop 1 nodes broadcast



- Nodes 2, 3, 4, and 5 broadcast, except on parent ports
- All nodes, not only G1, receive packets

Example: Broadcast continues



- *Truncated RPB (TRPB)*: Leaf routers do not broadcast if none of its attached hosts belong to packet’s multicast group

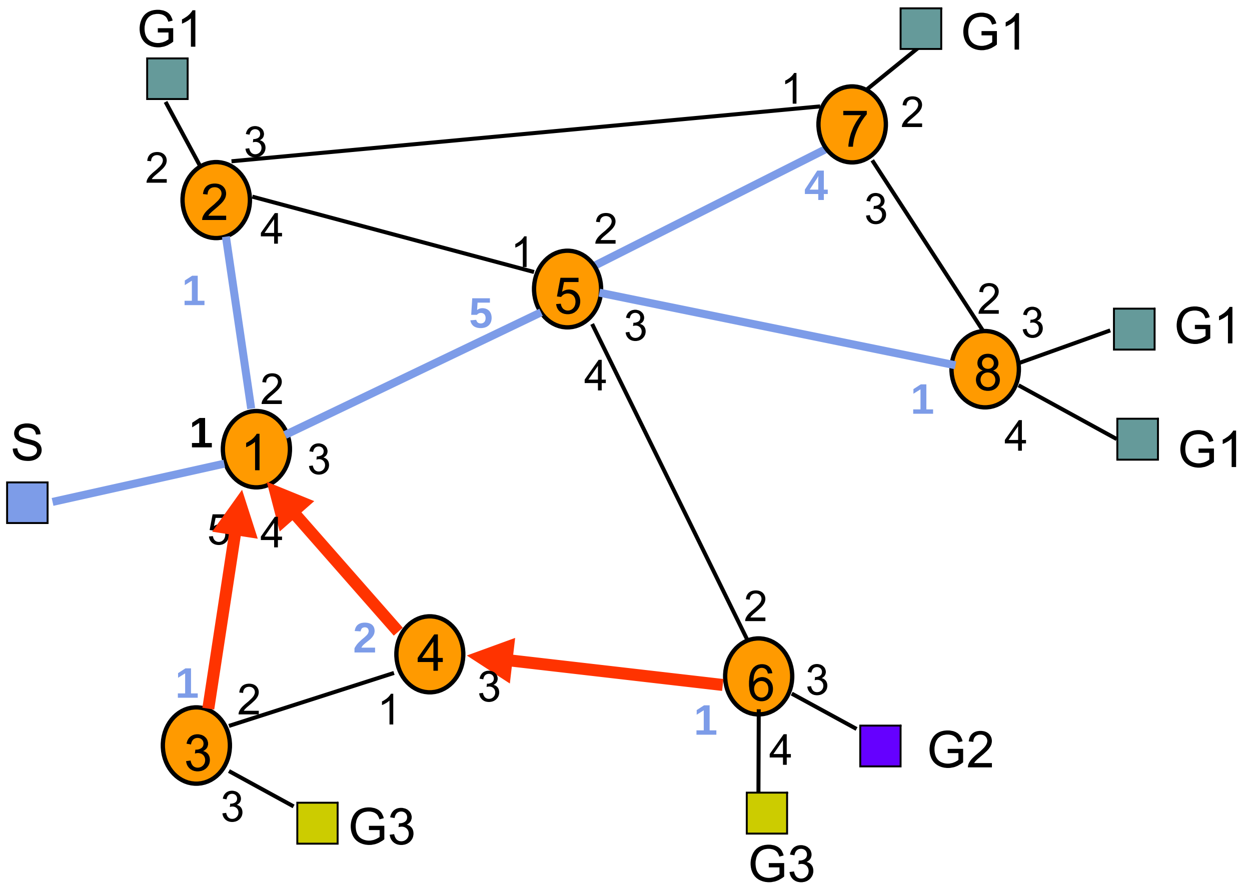
Internet Group Management Protocol (IGMP)

- *Internet Group Management Protocol:*
  - Host can join a multicast group by sending an IGMP message to its router
- Each multicast router periodically sends an IGMP query message to check whether there are hosts belonging to multicast groups
  - Hosts respond with list of multicast groups they belong to
  - Hosts randomize response time; cancel response if other hosts reply with same membership
- Routers determine which multicast groups are associated with a certain port
- Routers only forward packets on ports that have hosts belonging to the multicast group

Reverse-Path Multicasting (RPM)

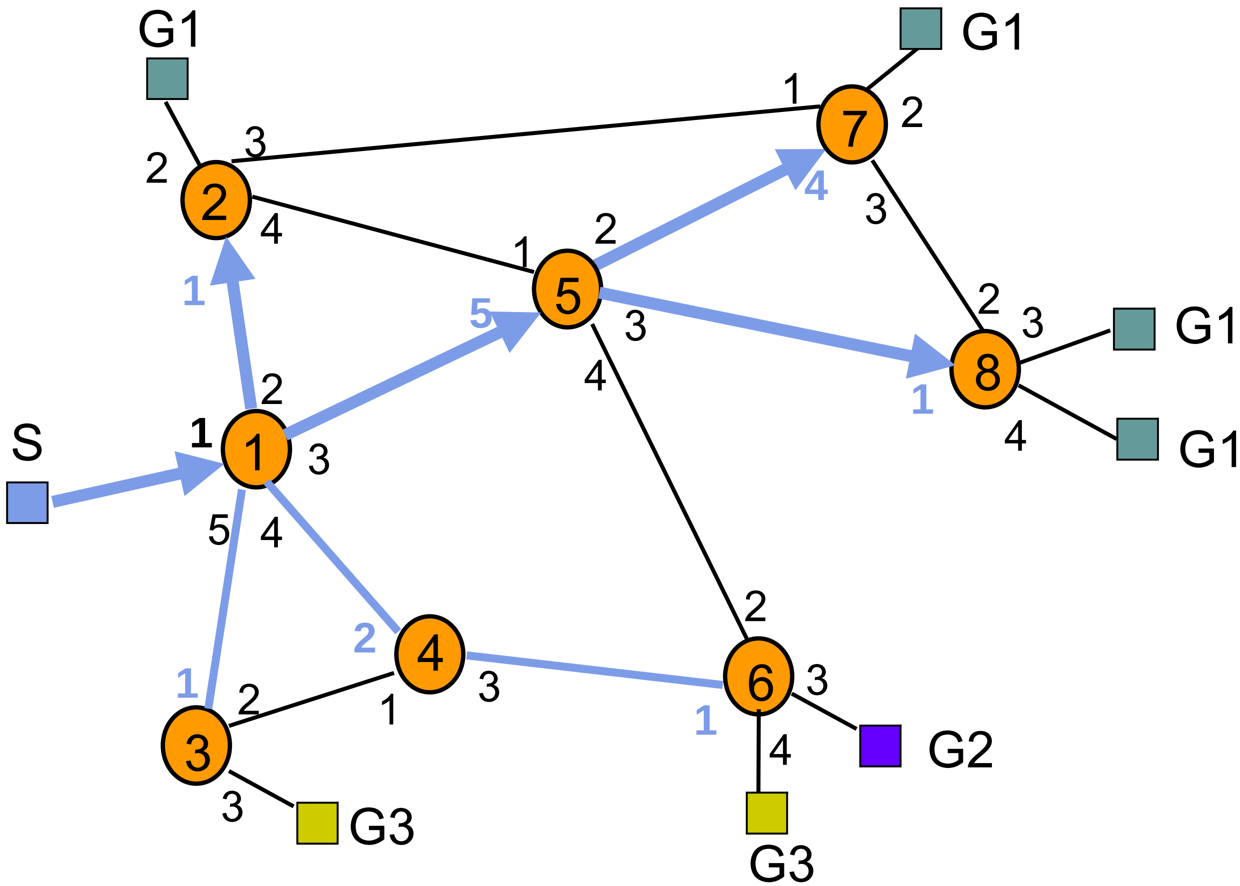
- *Reverse Path Multicasting (RPM)* relies on IGMP to identify multicast group membership
- The first packet to a given (source, group), i.e. (S,G) is transmitted to all leaf routers using TRPB
- Each leaf router that has no hosts that belong to this group on any of its ports, sends a ***prune*** message to its upstream router to stop sending packets to (S, G)
- Upstream routers that receive prune messages from all their downstream routers, send prune messages upstream
- Prune entries in each router have finite lifetime
- If a host requests to join a group, routers can cancel previous pruning with a ***graft*** message

Example: Pruning for G1



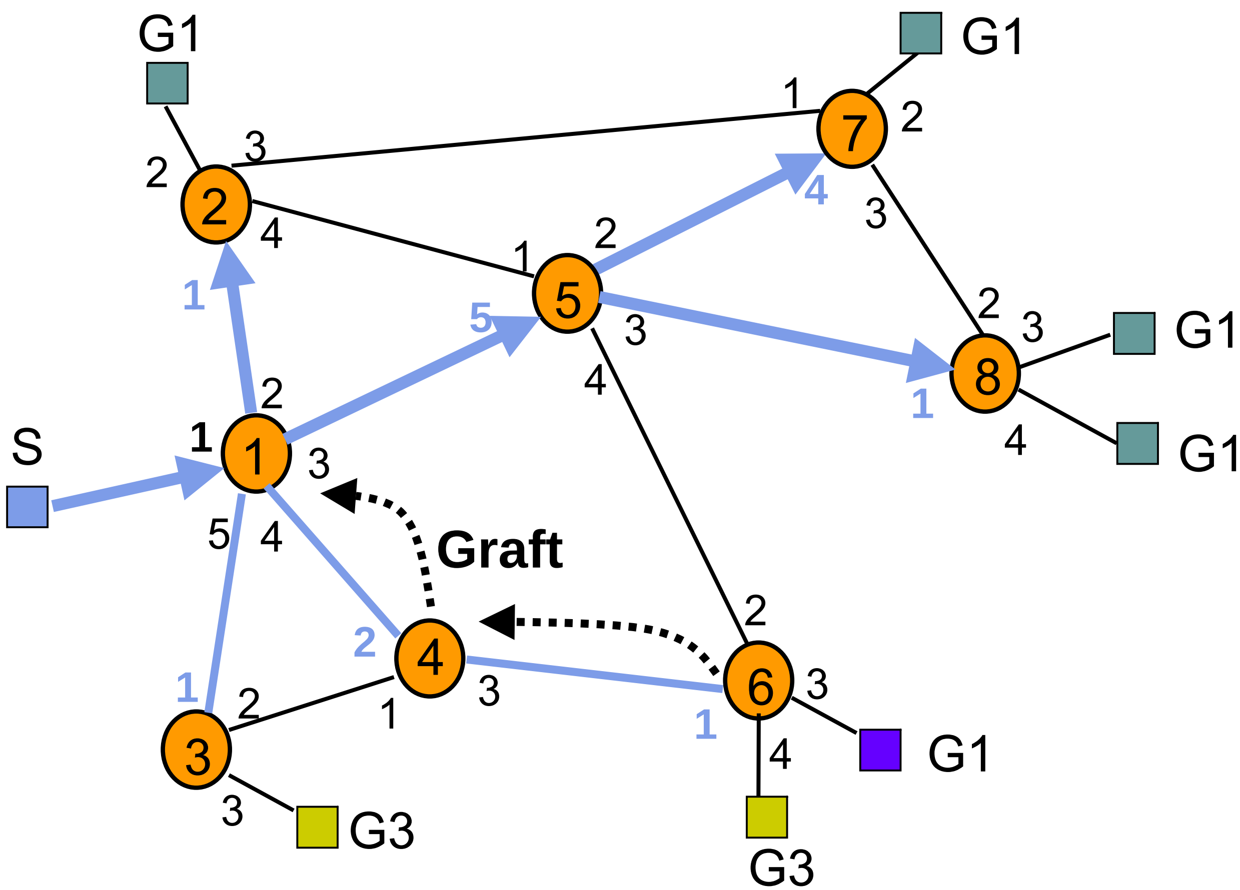
- Routers 3, 4, and 6 send prune messages upstream

Example: RPM Multicast Tree



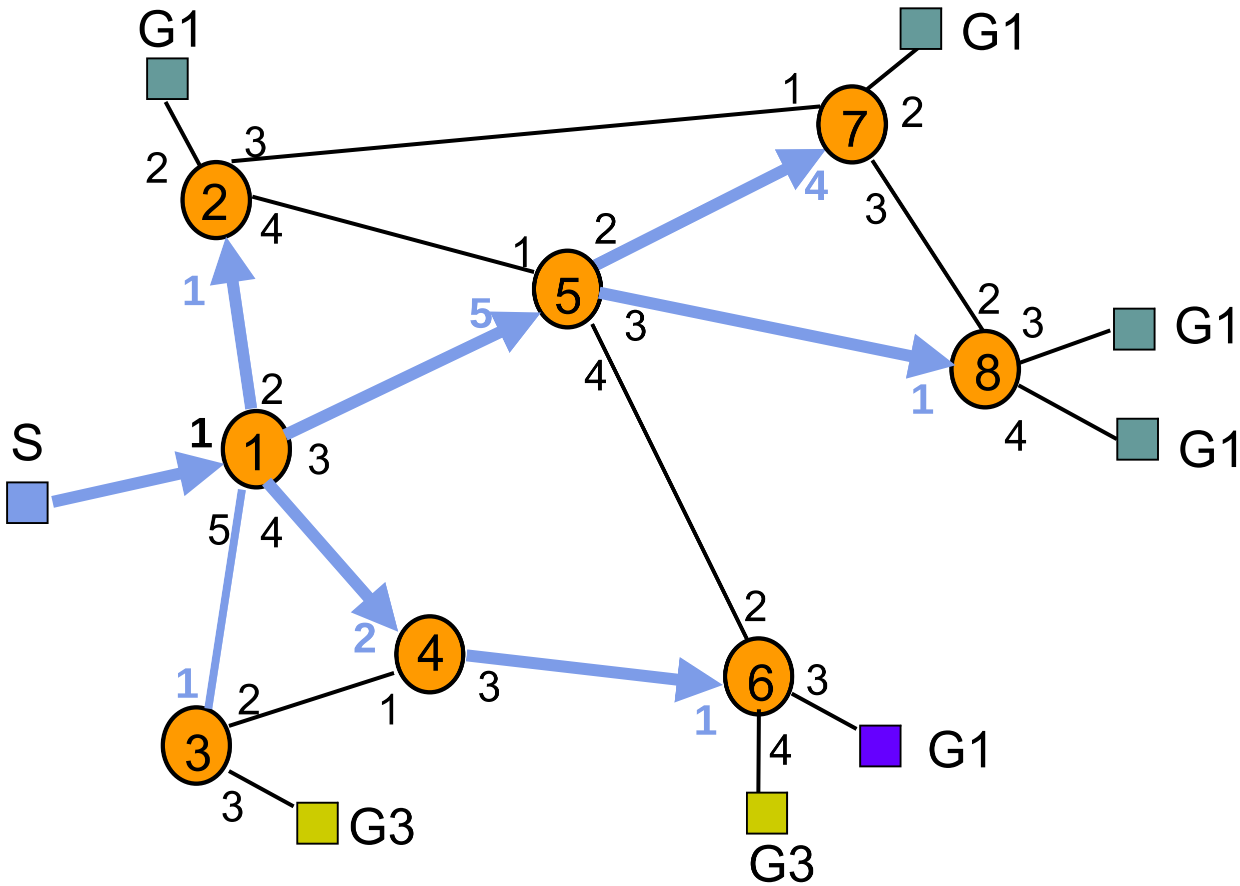
- RPM multicast tree after pruning

Example: Graft from Router 6



- Graft message flows upstream to router 1

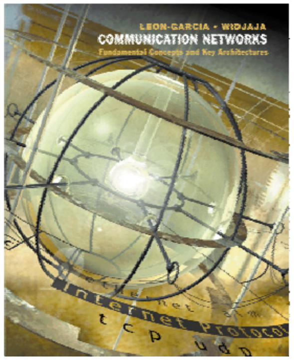
Example: RPM Tree after Graft



DVMRP

- Distance Vector Multicast Routing Protocol
  - Uses variation of RIP to determine next hop towards the source
  - Uses RPM and pruning to obtain multicast tree
  - Uses tunneling to traverse non-multicast networks
- Used in MBONE
- Not scalable
  - 15 hop limit
  - Flooding all leaf routers inefficient

Chapter 8
Communication Networks and Services



DHCP, NAT, and Mobile IP

DHCP

- Dynamic Host Configuration Protocol (RFC 2131)
- BOOTP (RFC 951, 1542) allows a diskless workstation to be remotely booted up in a network
  - UDP port 67 (server) & port 68 (client)
- DHCP builds on BOOTP to allow servers to deliver configuration information to a host
  - Used extensively to assign temporary IP addresses to hosts
  - Allows ISP to maximize usage of their limited IP addresses

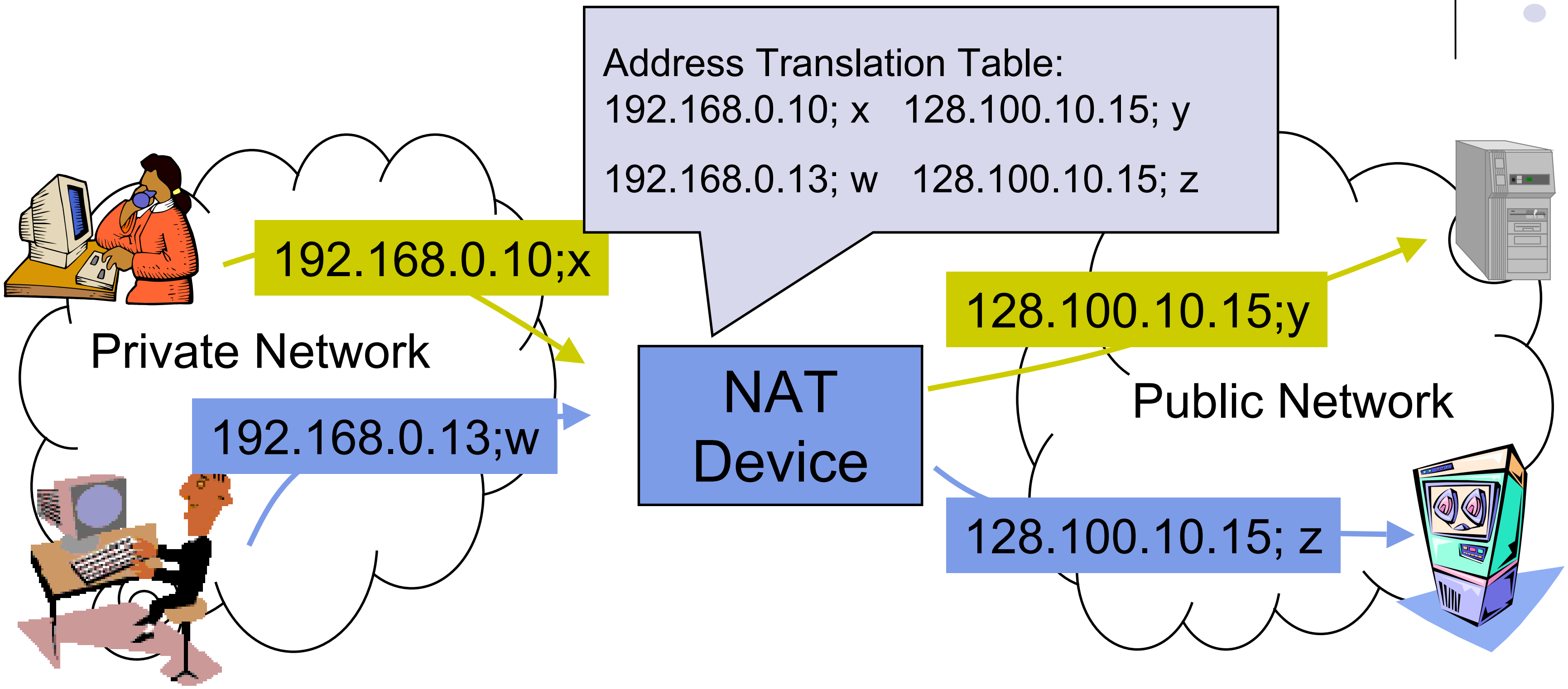
DHCP Operation

- Host broadcasts DHCP *Discover* message on its physical network
- Server replies with *Offer* message (IP address + configuration information)
- Host selects one offer and broadcasts *DHCP Request* message
- Server allocates IP address for lease time T
  - Sends DHCP ACK message with T, and threshold times T1 (=1/2 T) and T2 (=.875T)
- At T1, host attempts to renew lease by sending DHCP Request message to original server
- If no reply by T2, host broadcasts DHCP Request to *any* server
- If no reply by T, host must relinquich IP address and start from the beginning

Network Address Translation (NAT)

- Class A, B, and C addresses have been set aside for use within private internets
  - Packets with private ("unregistered") addresses are discarded by routers in the global Internet
- NAT (RFC 1631): method for mapping packets from hosts in private internets into packets that can traverse the Internet
  - A device (computer, router, firewall) acts as an agent between a private network and a public network
  - A number of hosts can share a limited number of registered IP addresses
    - Static/Dynamic NAT: map unregistered addresses to registered addresses
    - Overloading: maps multiple unregistered addresses into a single registered address (e.g. Home LAN)

NAT Operation (Overloading)

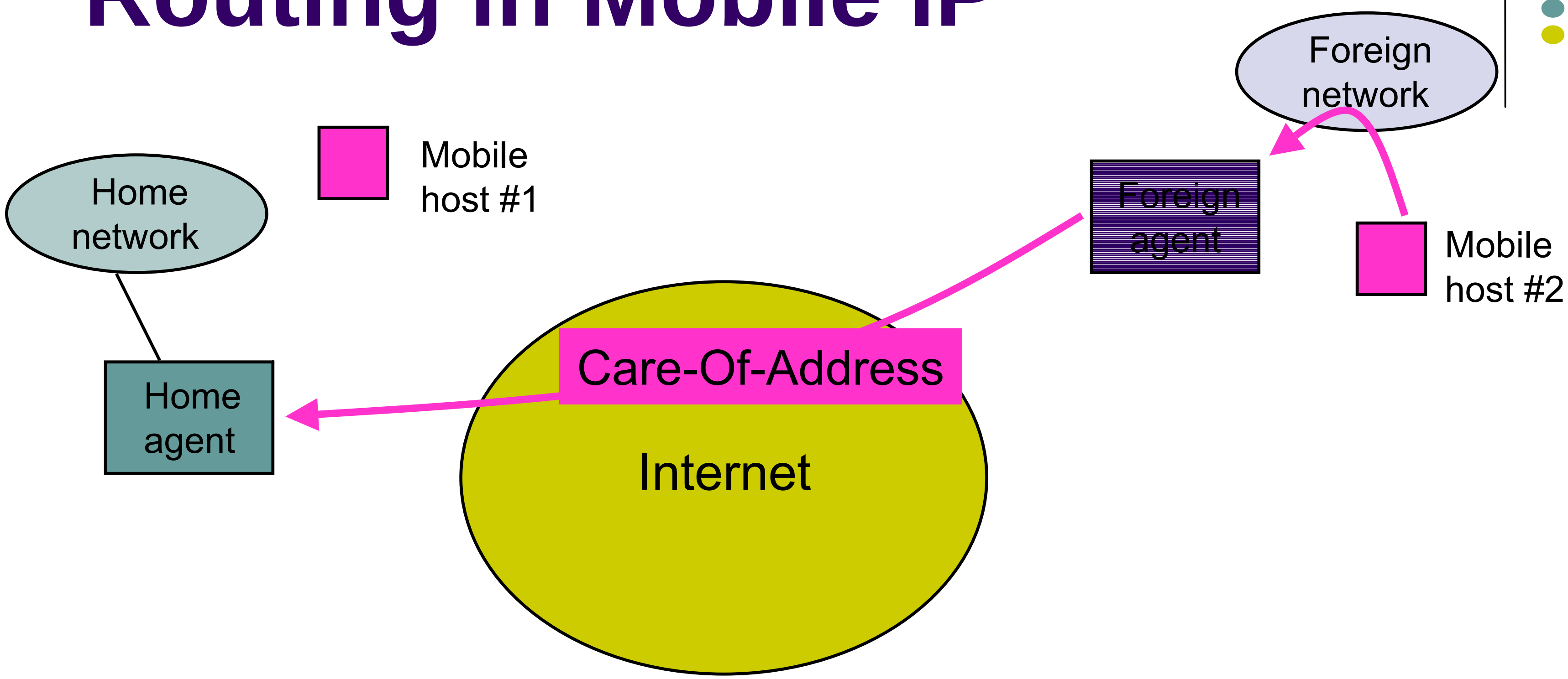


- Hosts inside private networks generate packets with private IP address & TCP/UDP port #s
- NAT maps each private IP address & port # into shared global IP address & available port #
- Translation table allows packets to be routed unambiguously

Mobile IP

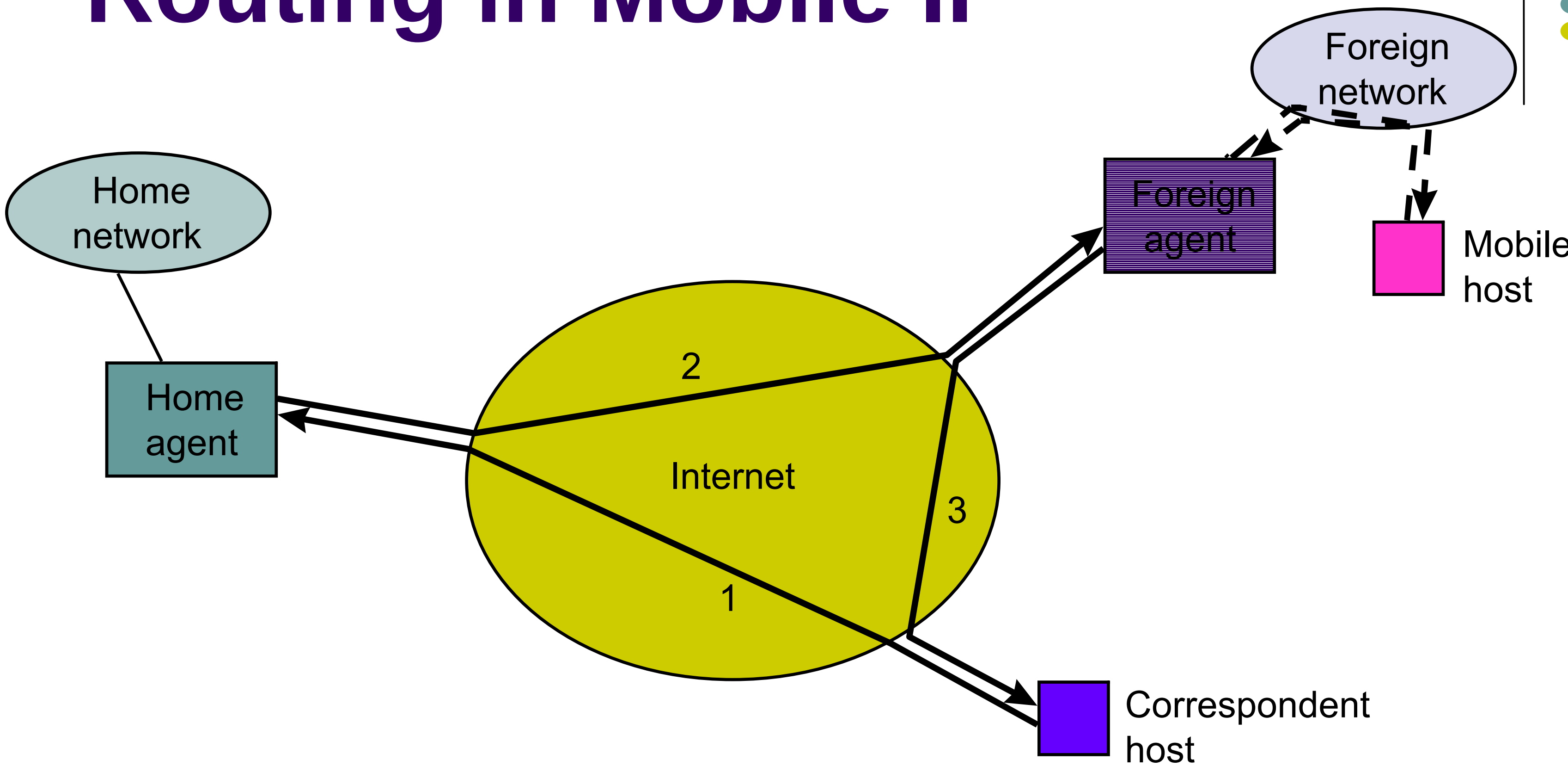
- Proliferation of mobile devices: PDAs, laptops, cellphones, …
- As user moves, point-of-attachment to network necessarily changes
- Problem: IP address specifies point-of-attachment to Internet
  - Changing IP address involves terminating all connections & sessions
- *Mobile IP (RFC 2002)*: device can change point-of-attachment while retaining IP address and maintaining communications

Routing in Mobile IP



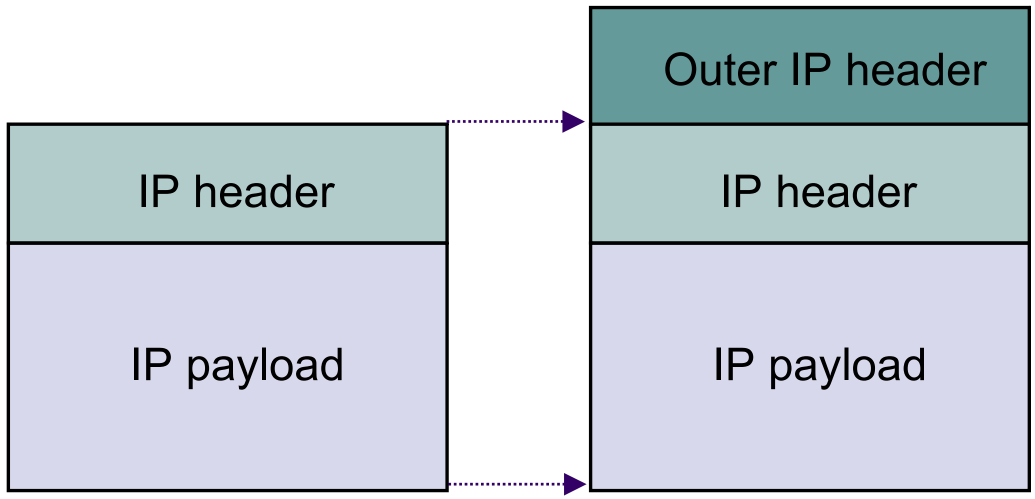
- Home Agent (HA) keeps track of location of each Mobile Host (MH) in its network; HA periodically announces its presence
- If an MH is in home network, e.g. MH#1, HA forwards packets directly to MH
- When an MH moves to a Foreign network, e.g. MH#2, MH obtains a care-of-address from foreign agent (FA) and registers this new address with its HA

Routing in Mobile IP



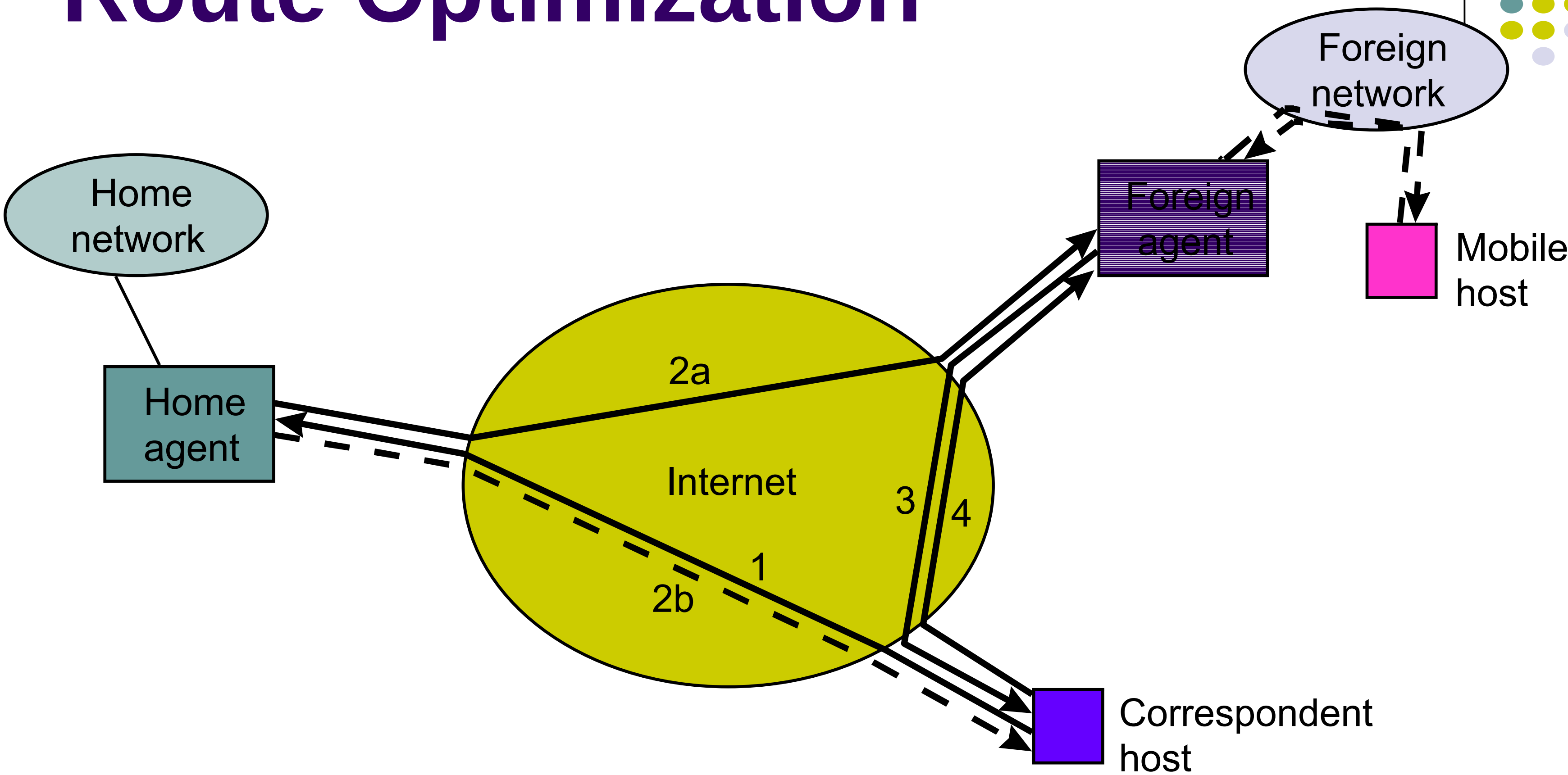
- Correspondent Host (CH) sends packets as usual (1)
- Packets are intercepted by HA which then forwards to Foreign Agent (FA) (2)
- FA forwards packets to the MH
- MH sends packet to CH as usual (3)
- How does HA send packets to MH in foreign network?

IP-to-IP Encapsulation



- HA uses IP-to-IP encapsulation
- IP packet has MH IP address
- Outer IP header has HA's address as source address and care-of-address as destination address
- FA recovers IP packet and delivers to MH

Route Optimization



- Going to HA inefficient if CH and MH are in same foreign network
- When HA receives pkt from CH (1), it tunnels using care-of-address (2a); HA also sends care-of-address to CH (2b)
- CH can then send packets directly to care-of-address (4)