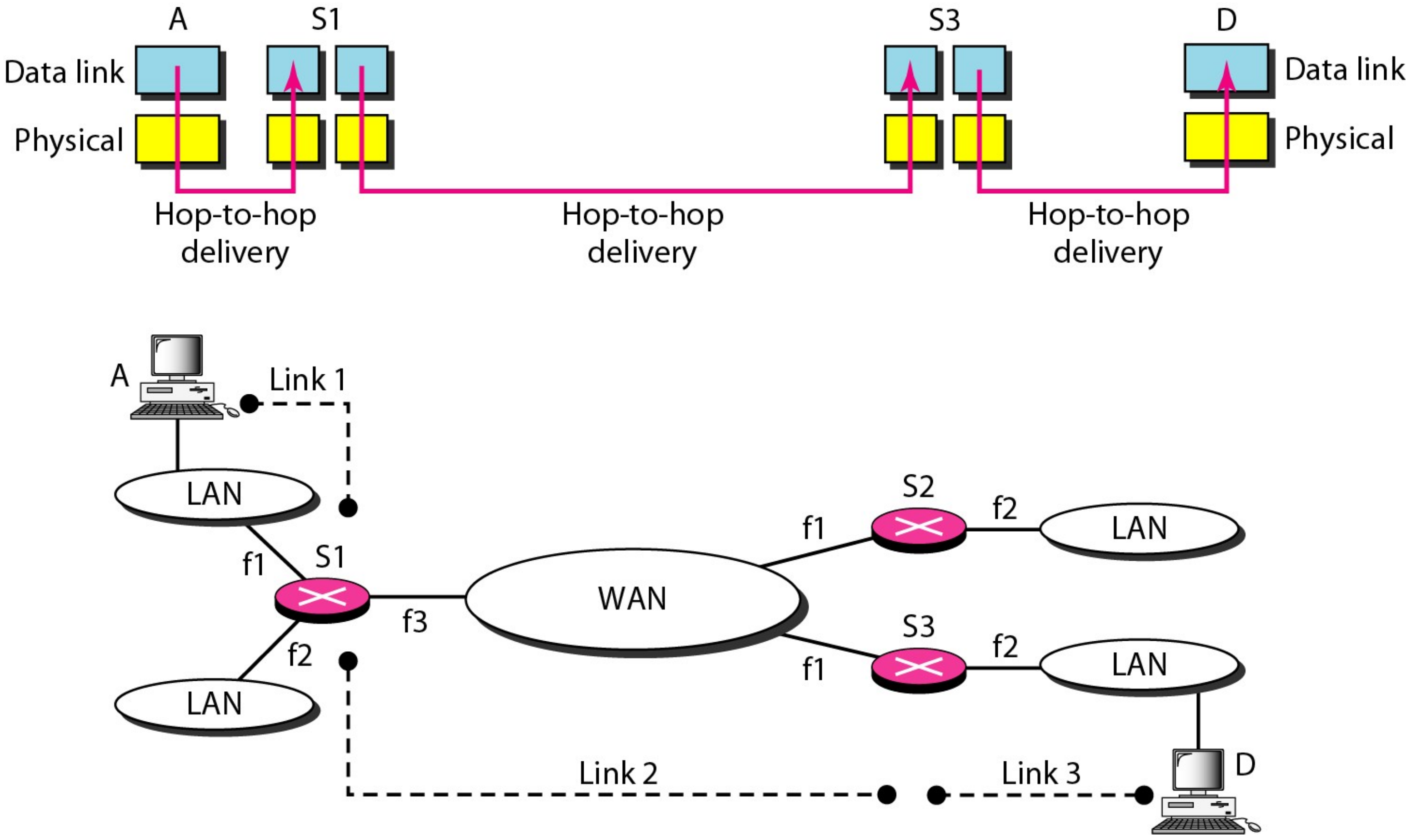
Data Communication and Computer Network

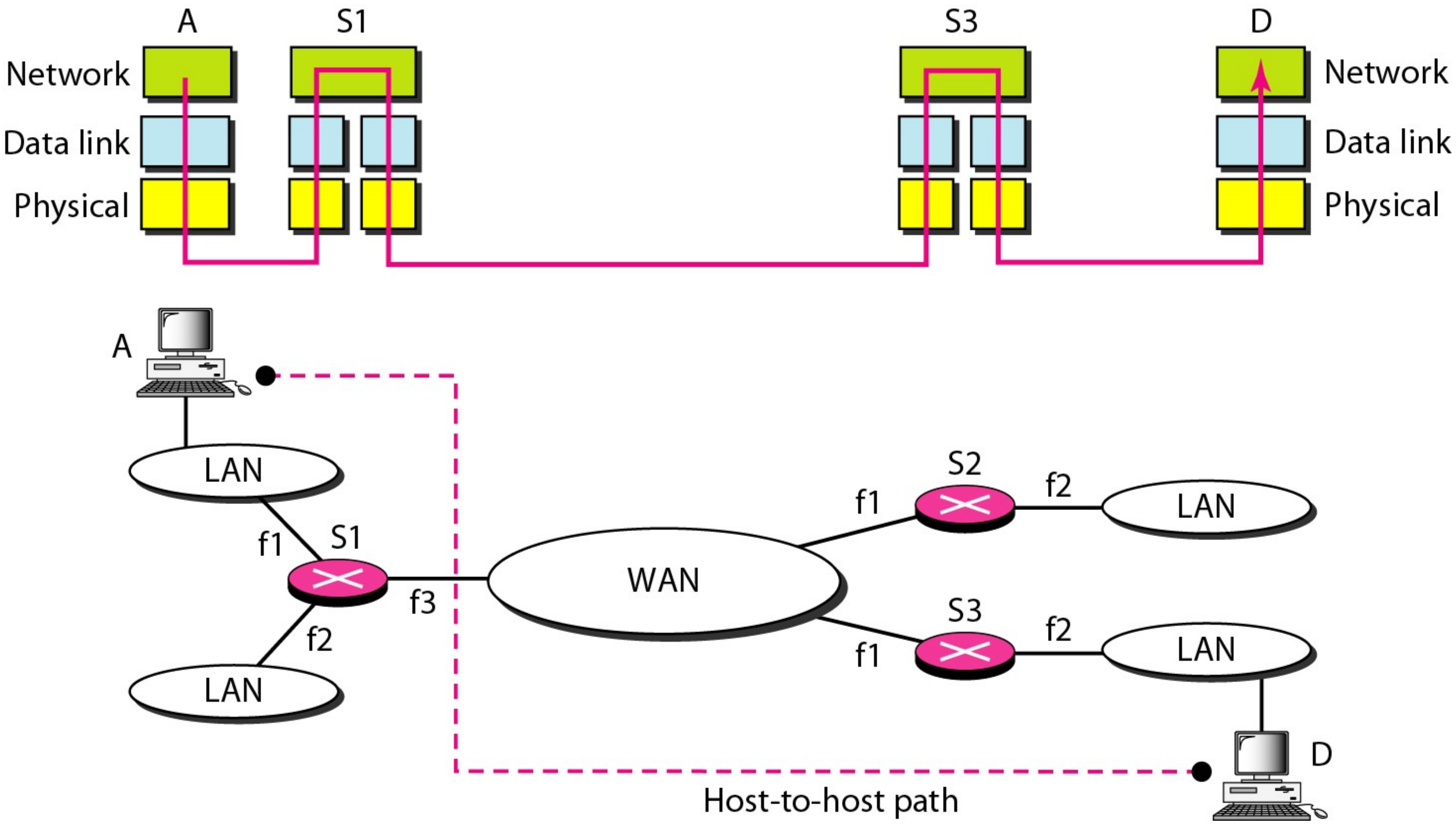
Network Layer Protocols

IPv4, ARP, ICMP

Links between two hosts

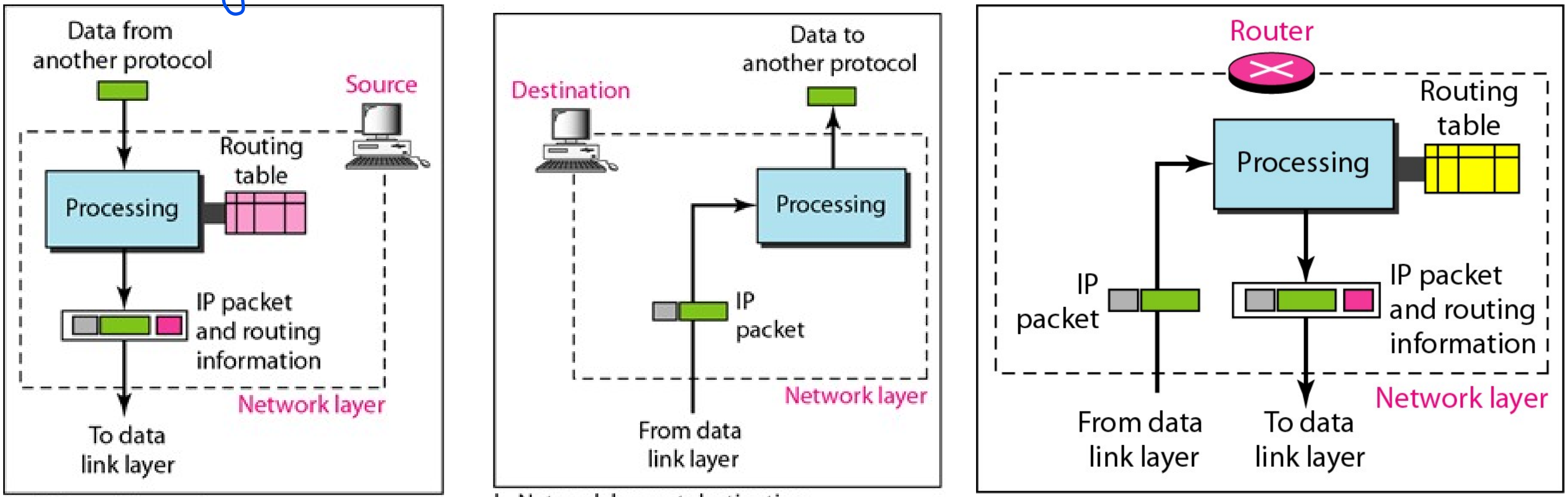


Network layer in an internetwork



Network layer at the source, router, and destination

Diagram



a. Network layer at source

b. Network layer at destination

c. Network layer at a router

Basic Data Transfer Scheme

At source node *(assume destination IP is known)*

```
Bitwise AND own IP and destination IP with netmask
if network numbers same     # destination in same network
    if destination's MAC address known
        send to it directly
    else
        broadcast ARP-Request using broadcast MAC address to get the MAC address
    end if
else        # destination node in some other network
    send to router, using router's MAC address
        # router's MAC address must be known
end if
```

Basic Data Transfer Scheme (contd.)

At any other machine

```
# at MAC layer
if dest MAC addr = own MAC addr or broadcast MAC addr
      give IP layer PDU to IP layer
else
      discard frame

# at IP layer
if dest IP addr = own IP addr or broadcast IP addr
     give higher layer PDU to higher layer
else if this machine is a router
     route the IP PDU (form a new MAC frame)
else
     discard the packet
```

Address Resolution Protocol (ARP)

Motivation

- A node N may have to send a frame to another node M
  - If N has to send data outside the network, N must first send the frame to router R in this network
  - N needs to know the MAC address of M (or R) for this
- Usually, IP layer of node N already knows IP address of the node to which it has to send data
  - Directly told by users (e.g., ssh to 10.2.1.97)
  - IP address of router specified while configuring nw connection
- But N needs to know the MAC address of the next hop
- ARP used to know the MAC address, given IP address

Address Resolution Protocol (ARP)

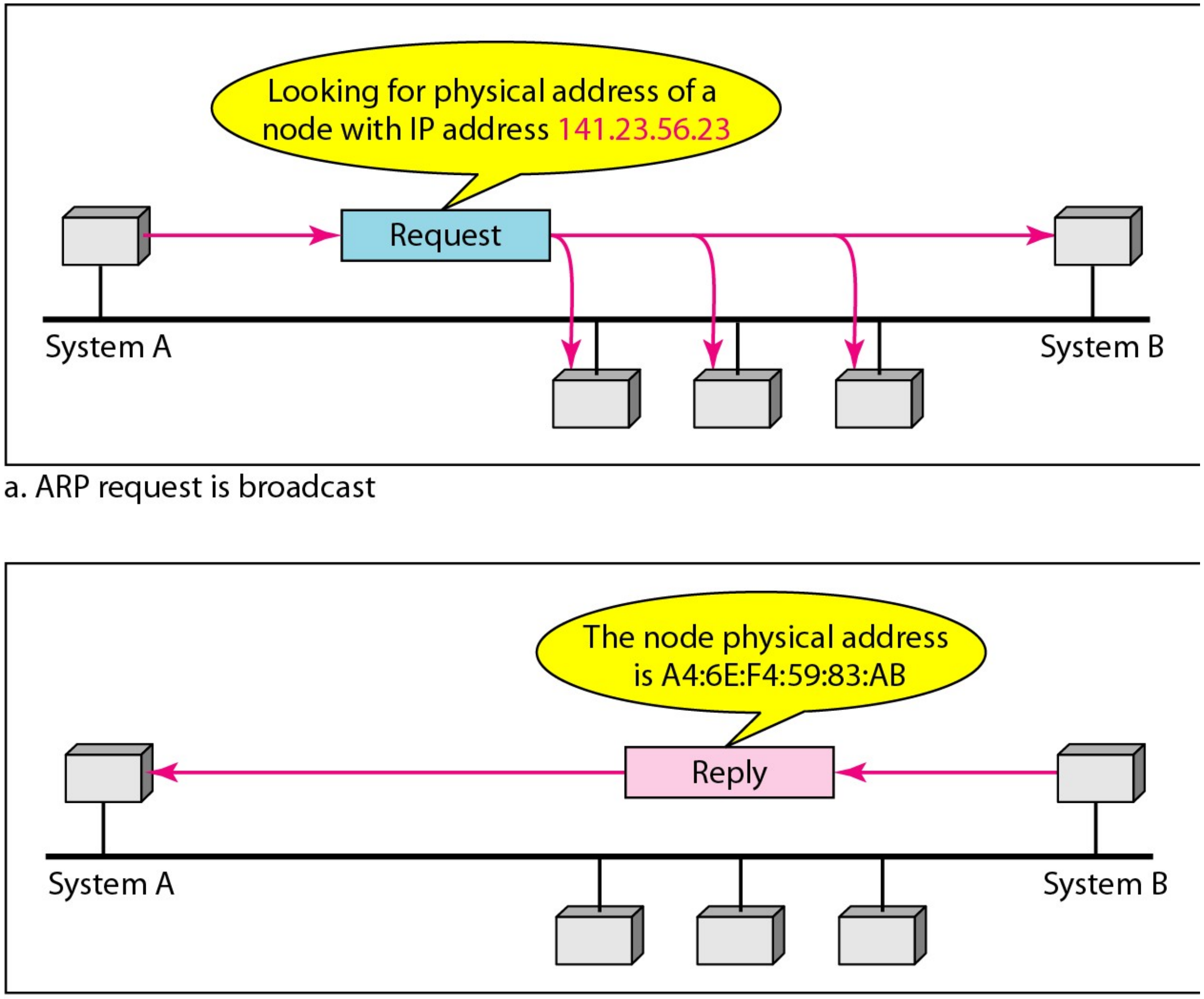
- ❑ Provides IP to Hardware (MAC) address mapping
- ❑ ARP packet (like IP packet) encapsulated in data link layer frame, e.g., Ethernet frame
- ❑ Type field in MAC frame specifies ARP packet
  - ➢ When an MAC frame is received, receiver's DLL layer sees the type field and
  - ➢ decides to which Network layer module (IP, ARP, …) the Network layer PDU will be handed

Use of ARP

- ❑ Let node N need to know the MAC addr of node R
  - ➢ N already knows IP address of R
- ❑ N's ARP module creates and sends a MAC frame
  - ➢ TYPE field in Ethernet header set to ARP (0806_{16})
  - ➢ Destination MAC address: broadcast MAC address
  - ➢ Frame contains the IP address of R (whose MAC address required), both IP and MAC address of N

- ❑ This MAC frame reaches ARP module of all nodes, only R replies
- ❑ R sends a MAC frame to N (R already knows N's MAC address), which contains R's MAC address

ARP operation


Looking for physical address of a node with IP address 141.23.56.23
Request
System A
System B
The node physical address is A4:6E:F4:59:83:AB
Reply
System A
System B

a. ARP request is broadcast

b. ARP reply is unicast

ARP Packet Format

<table>
<tr><td colspan="2">Hardware Type</td><td>Protocol Type</td></tr>
<tr><td>Hardware length</td><td>Protocol length</td><td>Operation
Request 1, Reply 2</td></tr>
<tr><td colspan="3">Sender hardware address
(For example, 6 bytes for Ethernet)</td></tr>
<tr><td colspan="3">Sender protocol address
(For example, 4 bytes for IP)</td></tr>
<tr><td colspan="3">Target hardware address
(For example, 6 bytes for Ethernet)
(It is not filled in a request)</td></tr>
<tr><td colspan="3">Target protocol address
(For example, 4 bytes for IP)</td></tr>
</table>

ARP Packet Format (contd.)

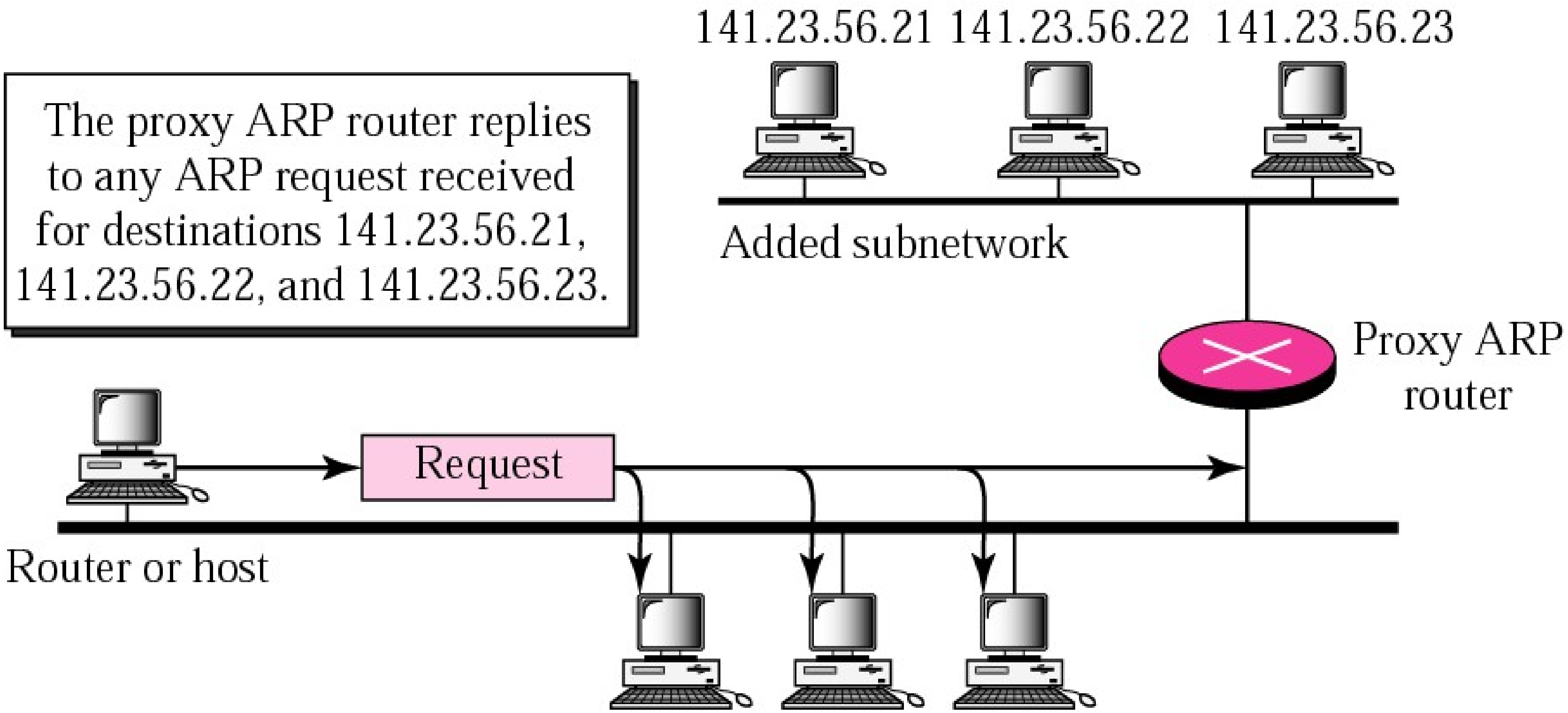
- Hardware type: a node can have multiple h/w interfaces e.g. Ethernet interface, FDDI (**Fiber Distributed Data Interface**) interface, etc
    - Enquiry for which hardware interface type (value of 1 implies Ethernet)
- Protocol type: which higher level protocol is being used, IP or some other (contains value 0800_{16} for IP)
- Target h/w address
    - Field left blank in the ARP request, filled in the ARP reply by target node
- Target node fills in missing address, swaps the target and sender address pairs, and changes operation to a reply

ARP Cache

- MAC addresses once known are cached, process not repeated always
  - When sending a packet, IP looks in its cache for a binding; if found, no broadcast required
  - Cache entries have a timeout period (typically 20 min)

- ARP can find mappings of machines only within the same (sub) network as the source node
  - Proxy ARP routers can be used for multiple connected LANs

Proxy ARP


141.23.56.21
141.23.56.22
141.23.56.23
The proxy ARP router replies to any ARP request received for destinations 141.23.56.21, 141.23.56.22, and 141.23.56.23.
Added subnetwork
Proxy ARP router
Request
Router or host

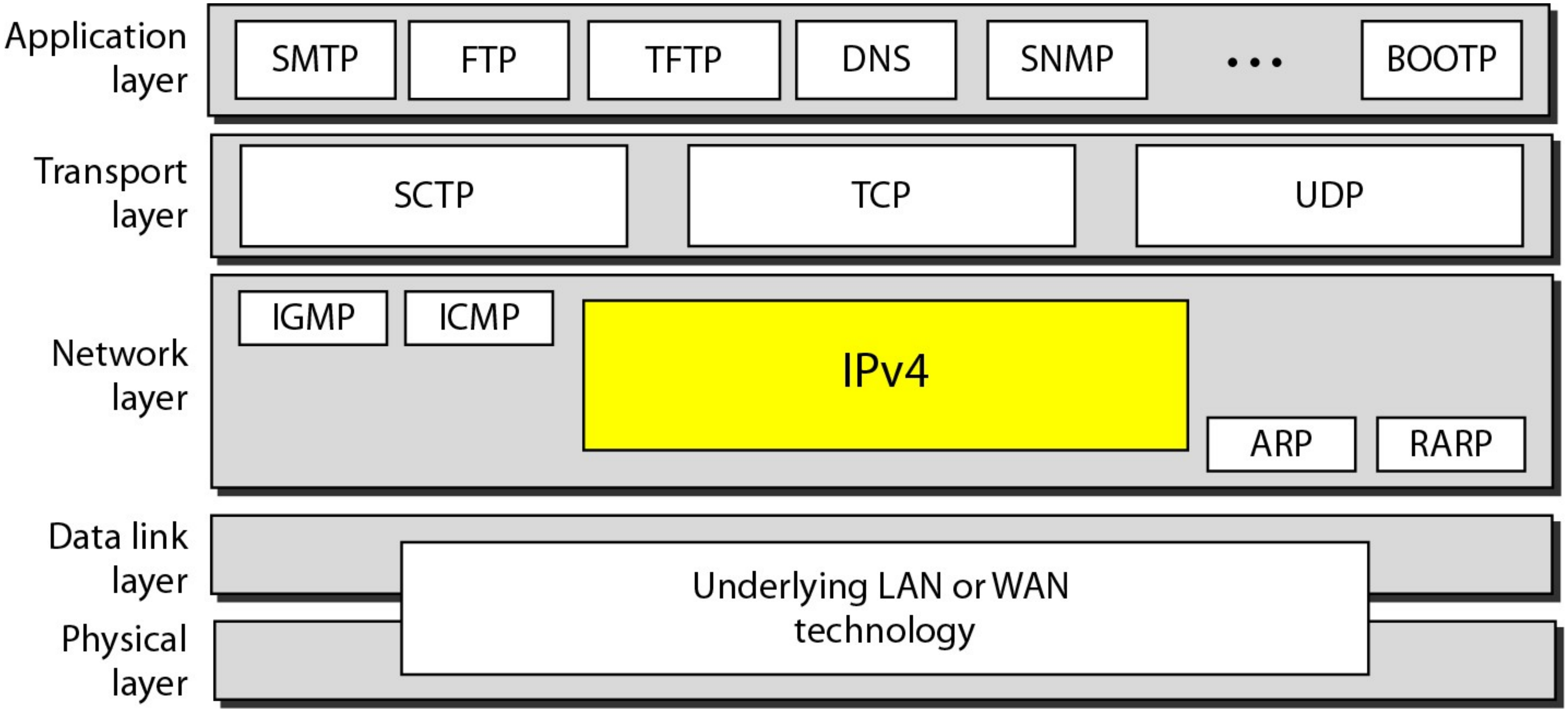
The proxy router is willing to accept packets to be sent to any host on the other subnet.

RARP

- Reverse of ARP: gets IP address, given hardware address
  - Can be used by diskless machines / small embedded systems which need to transfer files from some remote server, to obtain their initial boot image
  - Already has a hardware address, but needs an IP address for file transfer
- RARP uses same packet format as ARP
- Requesting node broadcasts RARP request
- RARP servers (one or more in a network) reply, giving the IP address

IPv4 – Internet Protocol

Layering of Protocols



Application layer

SMTP FTP TFTP DNS SNMP • • • BOOTP

Transport layer

SCTP TCP UDP

Network layer

IGMP ICMP IPv4 ARP RARP

Data link layer

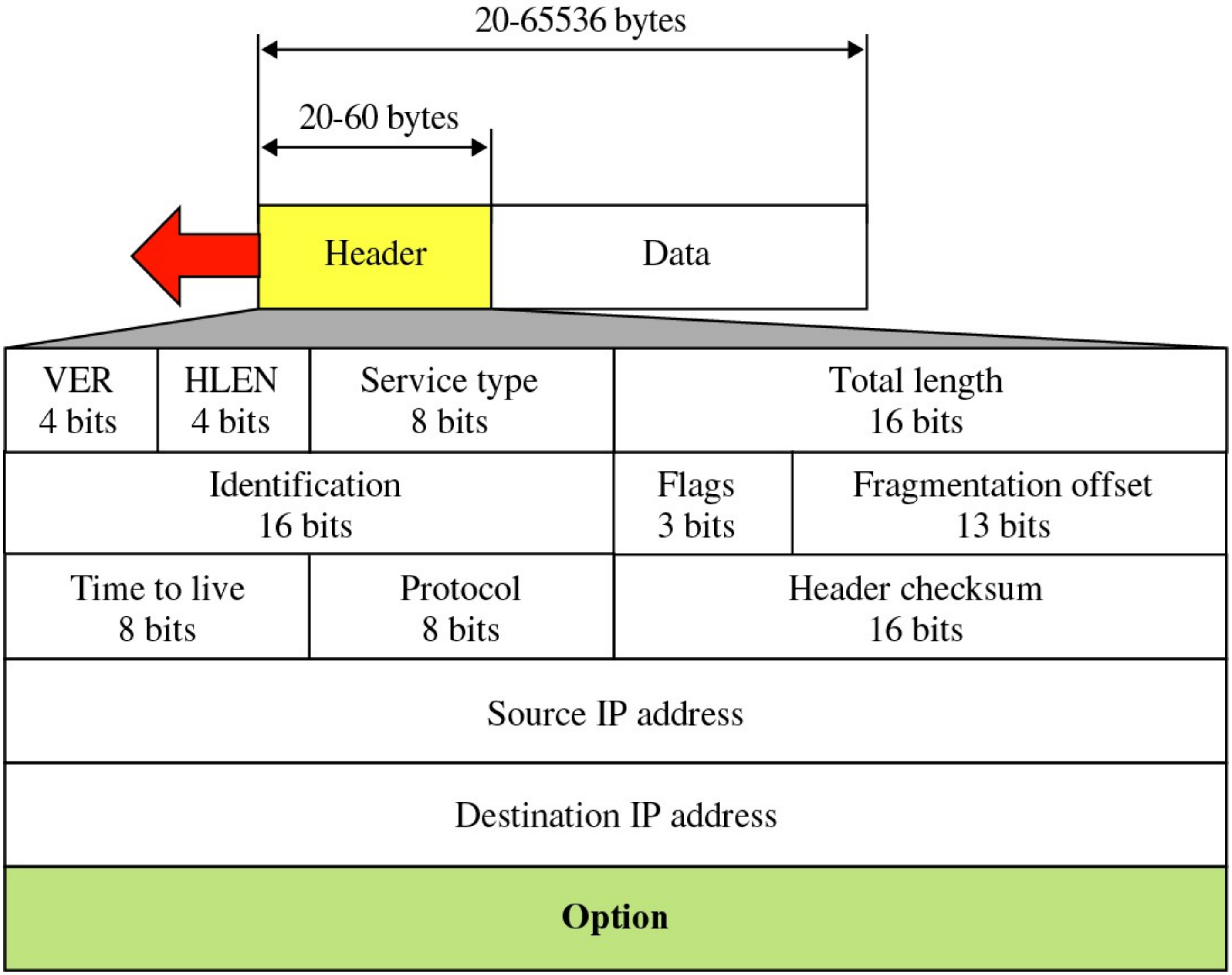
Physical layer

Underlying LAN or WAN technology

IPv4

- Most widely used network layer protocol in TCP/IP suite, IP defined originally in RFC 791
  - Connectionless: no explicit connection setup / termination phase before / after data transfer
  - Message broken up into packets, each packet switched independently between routers
  - IP header attached to each packet
  - Flexible, robust to failures, no unnecessary overhead
  - Unreliable, best-effort service: Packets can be lost, duplicated, come out-of-sequence
- Main issues handled at network layer: routing and fragmentation / reassembly

IP datagram



20-65536 bytes

20-60 bytes

Header

Data

VER
4 bits

HLEN
4 bits

Service type
8 bits

Total length
16 bits

Identification
16 bits

Flags
3 bits

Fragmentation offset
13 bits

Time to live
8 bits

Protocol
8 bits

Header checksum
16 bits

Source IP address

Destination IP address

Option

Fields in IP Header

- ❑ Version number (4 bits)
  - ➢ Indicates the version of the IP protocol
  - ➢ Typically “4” (for IPv4), sometimes “6” (for IPv6)
- ❑ Header length (4 bits)
  - ➢ Number of 32-bit words in the header
  - ➢ Typically “5” (IPv4 header is at least 20 bytes)
- ❑ Total length (16 bits)
  - ➢ Number of bytes in the entire packet (including header and data)
  - ➢ Maximum size is 63,535 bytes (2^{16} -1)

Fields in IP Header (contd.)

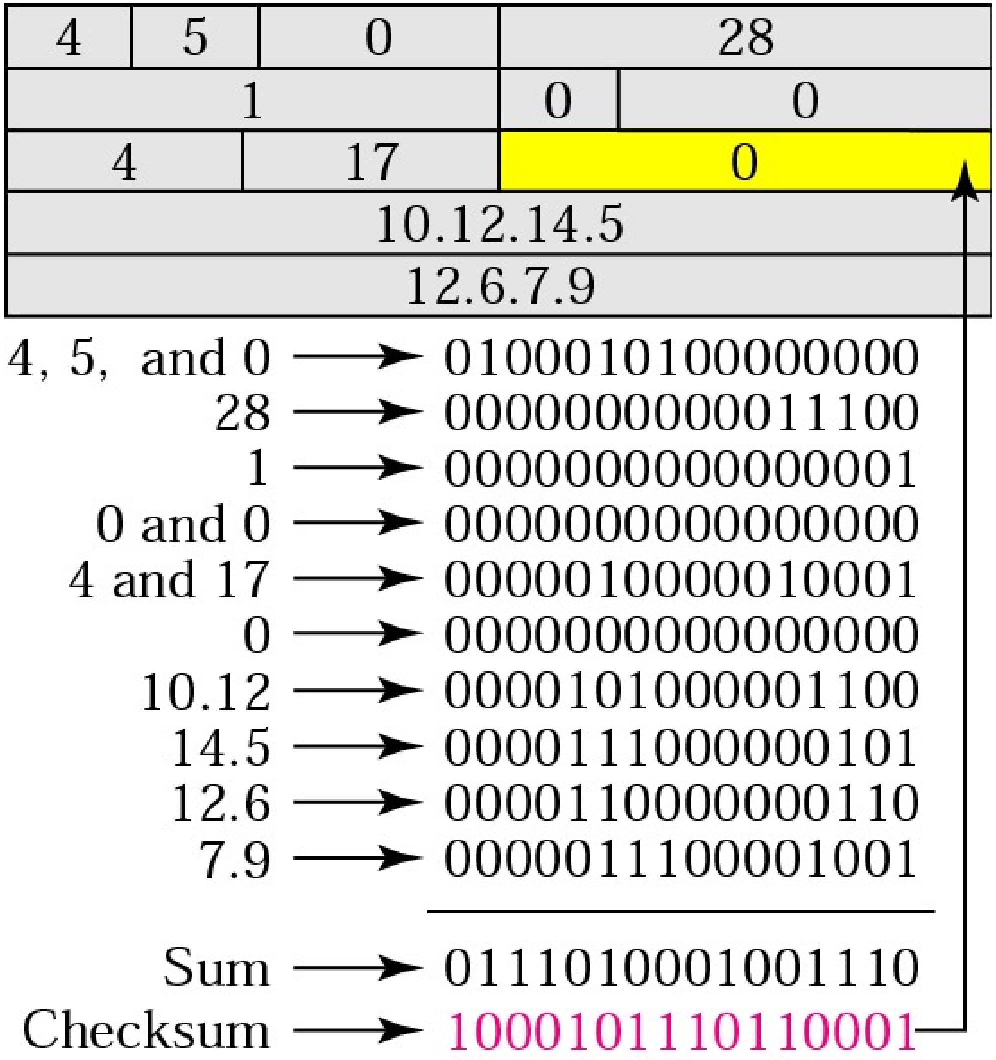
- ❑ Time-To-Live (8 bits)
  - ➢ Used to identify packets stuck in forwarding loops and eventually discard them from the network (prevents a data packet from circulating indefinitely)
  - ➢ Used to control the max number of hops(router) a datagram can be visited. Source hosts put a number approx. 2 times than the number of router, each router decrements by 1. Once TTL is 0, it is discarded.

- ❑ Protocol (8 bits)
  - ➢ Identifies the higher-level protocol for which this IP packet is meant
    - ✓ E.g. "6" for the TCP, "17" for UDP

Checksum on the IP Header

- Checksum (16 bits)
  - Break IP Header into 16-bit units (checksum field = 0)
  - Sum these units, using 1’s complement arithmetic
  - Take 1’s complement of the sum
- Checksum verified and re-computed at each router and at the final destination
  - If mismatch, discard corrupted packets
  - Sending host will retransmit the packet, if needed
- IP checksum computed only on IP header, NOT on the data in the packet

Example of checksum calculation

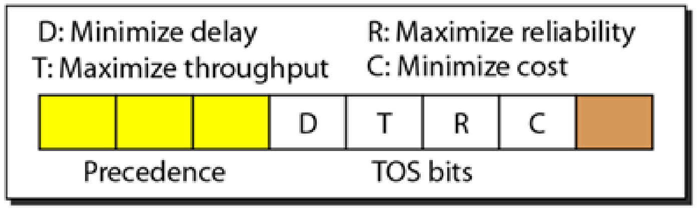


4	5	0	28
1		0	0
4	17	0	
10.12.14.5			
12.6.7.9			

4, 5, and 0	→	0100010100000000
28	→	0000000000011100
1	→	0000000000000001
0 and 0	→	0000000000000000
4 and 17	→	0000010000010001
0	→	0000000000000000
10.12	→	0000101000001100
14.5	→	0000111000000101
12.6	→	0000110000000110
7.9	→	0000011100001001
Sum	→	0111010001001110
Checksum	→	1000101110110001

Type of service field (8 bits)

- 3-bit precedence field: datagram precedence with values 0 (normal data) – 7 (network control)
  - Routers may give more precedence to control information than to normal data. However, it never used.
- Three 1-bit fields specifying desired service qualities (as desired by higher layer protocols)
  - D bit: request to minimize delay
  - T bit: request to maximize throughput
  - R bit: request to maximize reliability
  - Only one bit can be set, none set implies ‘normal service’
  - Last 2 (C and other) bits unused and must be 0

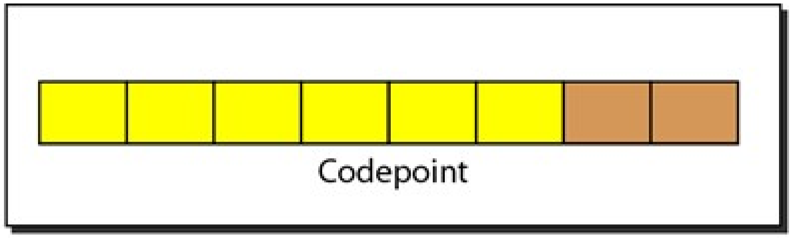


Service type

"throughput" in a networking context is a measure of how much data can be successfully transmitted from one point to another in a network within a specific timeframe.

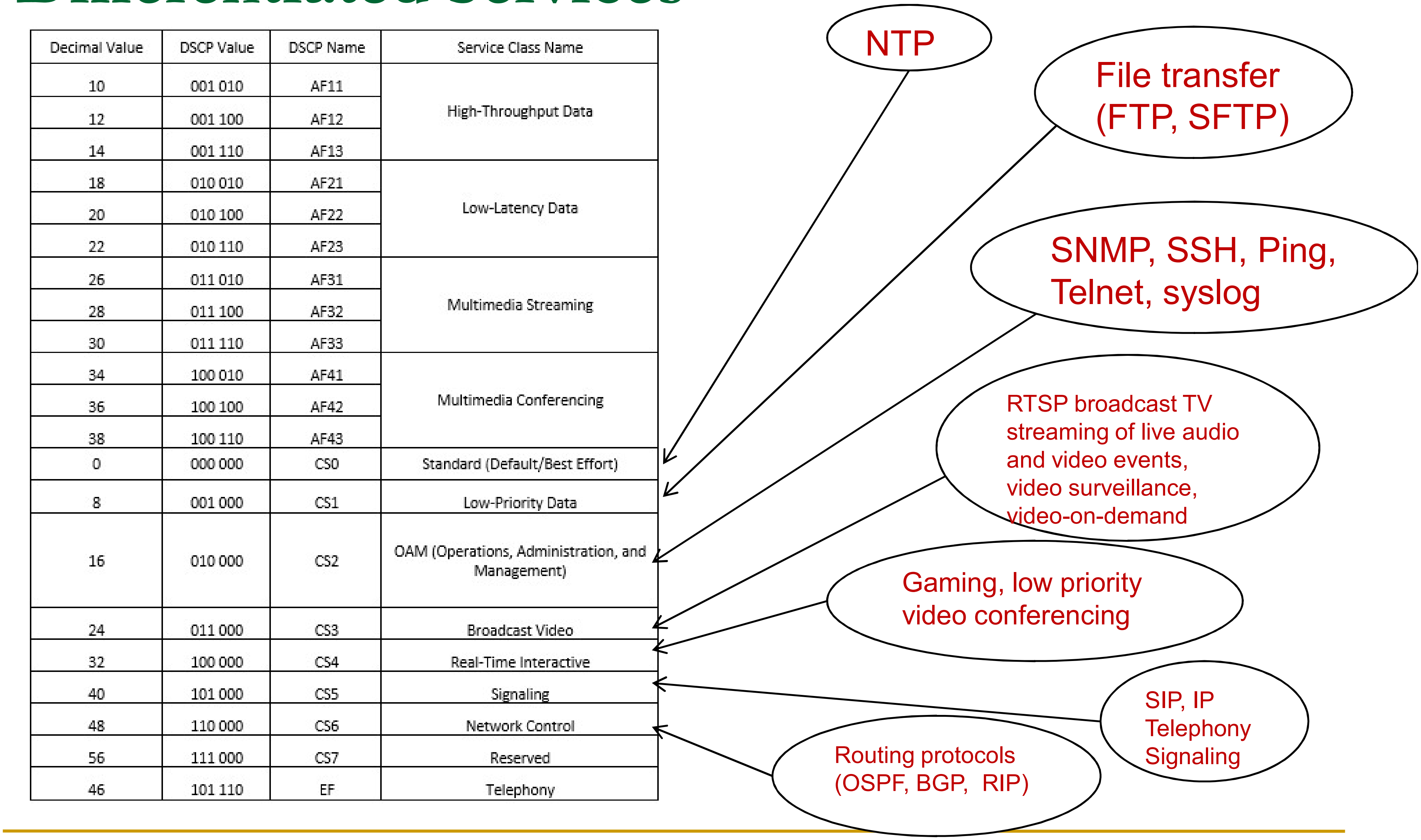
Differentiated Services

- Modern data networks carry many different types of services, including voice, video, streaming music, web pages and email. Many of the proposed QoS mechanisms that allowed these services to co-exist were both complex and failed to scale to meet the demands of the public Internet.
- In late 1990, IETF redefined the field to provide Differentiated service (DiffServ) use Quality of Service (QoS)
- DiffService Capable Node uses differentiated services code point (DSCP) 6 bits as an index to a table defining packet handling mechanism for the current packet being processed



Differentiated services

Differentiated Services



Decimal Value	DSCP Value	DSCP Name	Service Class Name
10	001 010	AF11	High-Throughput Data
12	001 100	AF12	
14	001 110	AF13	
18	010 010	AF21	Low-Latency Data
20	010 100	AF22	
22	010 110	AF23	
26	011 010	AF31	Multimedia Streaming
28	011 100	AF32	
30	011 110	AF33	
34	100 010	AF41	Multimedia Conferencing
36	100 100	AF42	
38	100 110	AF43	
0	000 000	CS0	Standard (Default/Best Effort)
8	001 000	CS1	Low-Priority Data
16	010 000	CS2	OAM (Operations, Administration, and Management)
24	011 000	CS3	Broadcast Video
32	100 000	CS4	Real-Time Interactive
40	101 000	CS5	Signaling
48	110 000	CS6	Network Control
56	111 000	CS7	Reserved
46	101 110	EF	Telephony

NTP

File transfer (FTP, SFTP)

SNMP, SSH, Ping, Telnet, syslog

RTSP broadcast TV streaming of live audio and video events, video surveillance, video-on-demand

Gaming, low priority video conferencing

SIP, IP Telephony Signaling

Routing protocols (OSPF, BGP, RIP)

Data in an IP datagram

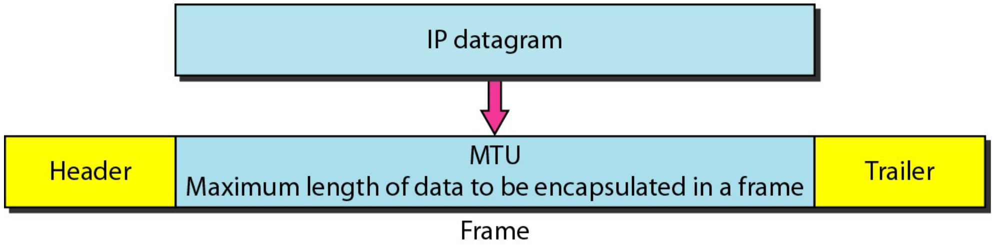
- Carries user data from higher transport layer
- Length: in units of bytes (octet)
- Maximum total length of datagram (header plus data): 65,535 bytes (2^16 – 1)
  - Total length is a 16-bit field

- However, such a large datagram is not usually allowed at lower layers

  E.g. Ethernet allows MAC frames of up to 1518 bytes

Fragmentation of IP datagram

- Maximum transfer unit (MTU)
  - Any network technology has a MTU (e.g. for Ethernet, higher layer PDU can be at most 1500 bytes)



Protocol	*MTU*
Hyperchannel	65,535
Token Ring (16 Mbps)	17,914
Token Ring (4 Mbps)	4,464
FDDI	4,352
Ethernet	1,500
X.25	576
PPP	296

Fragmentation of IP datagram

- When a router has to transmit a datagram too large for the MTU of the outgoing link, datagram is fragmented

- A single IP datagram can arrive at the destination as multiple fragments
  - Fragments re-assembled at the destination node
  - Intermediate routers do NOT re-assemble fragments

Two terms: Packet vs Datagram

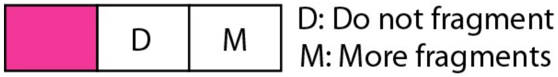
- An IP datagram is the unit of end-to-end transmission at the IP layer (before fragmentation & after reassembly)

- A packet is the unit of data passed between the IP layer and the link layer

- A packet can be a complete IP datagram or a fragment

Fields used for fragmentation

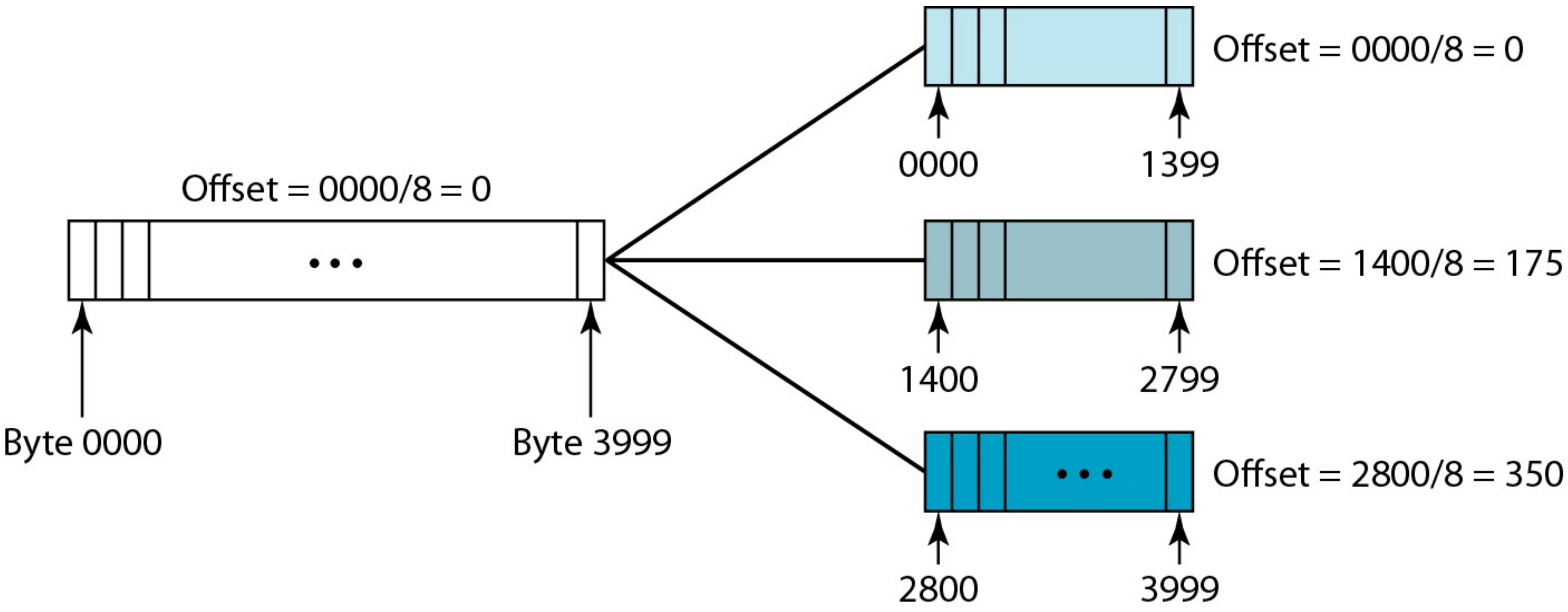
- Identification
  - Identifies each datagram uniquely originated from a host – managed by a counter at IP layer
  - Destination node uses the <source IP, identification> to identify which arriving fragment belongs to which datagram
- Flag: 3-bit field
  - DF: do not fragment. If source sets to 1
    - routers send this datagram un-fragmented if possible, otherwise
    - discard and may send an ICMP message which indicates the condition "*Packet too Big*"
  - MF: are there more fragments (of this datagram) after this one?
  - Third bit is reserved



- Fragment offset
  - Offset of the data contained in this fragment, in the data contained in the actual IP datagram sent by source
  - Given in units of 8-byte blocks

Fragmentation – an example

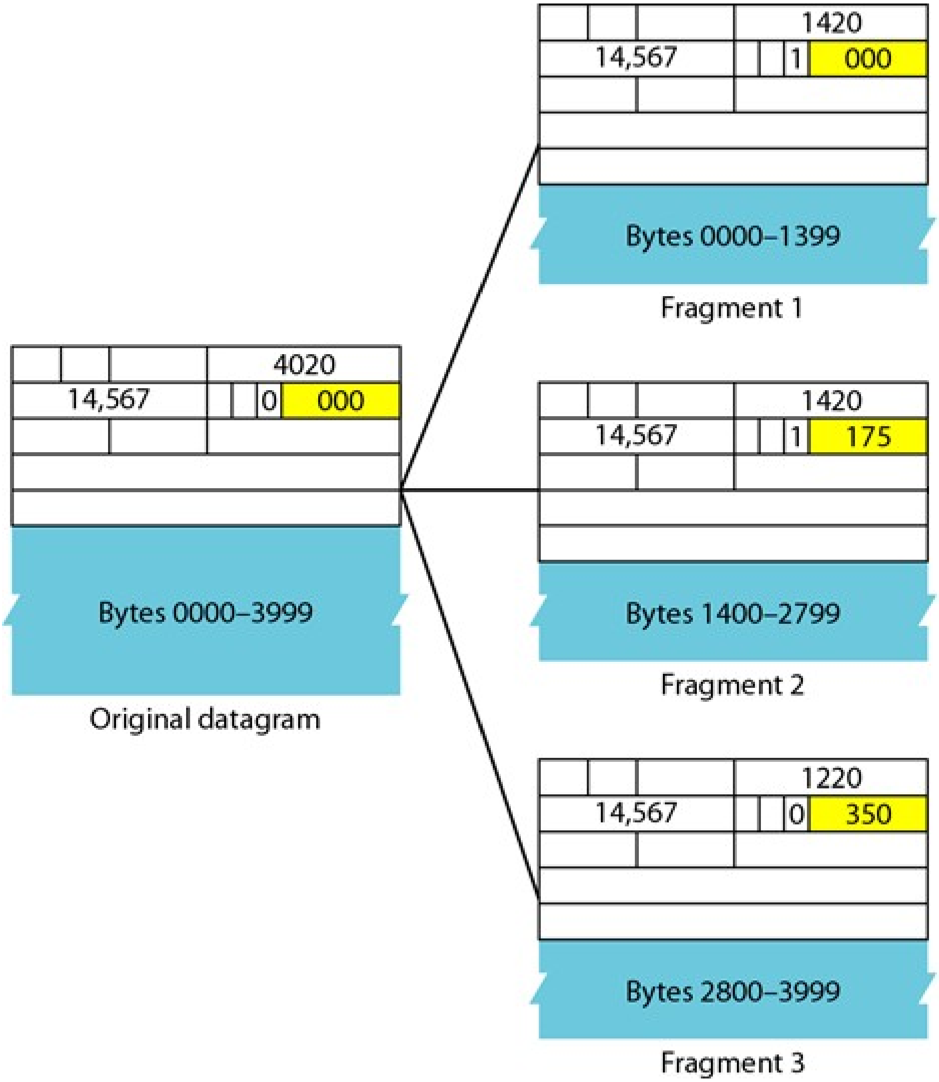
- ❑ 4020 byte datagram including 20 byte IP header
- ❑ Length of data in datagram: 4000 bytes
- ❑ At some intermediate router next hop's MTU is 1420 byte



Fragmentation – an example (contd…)

- 4020 byte datagram including 20 byte IP header. Length of data in datagram: 4000 bytes

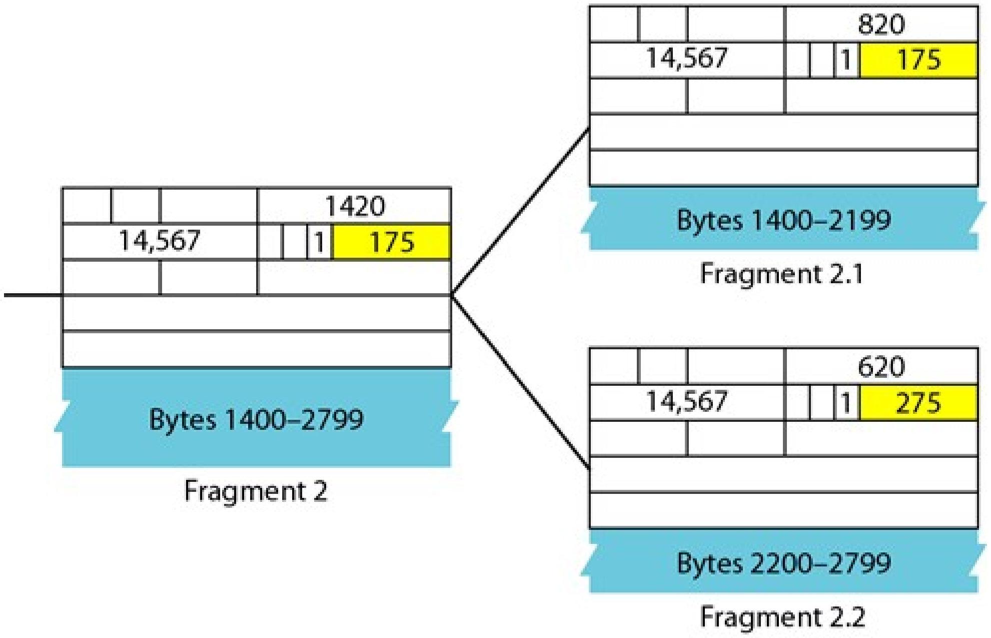
- At some intermediate router next hop's MTU is 1420 byte.



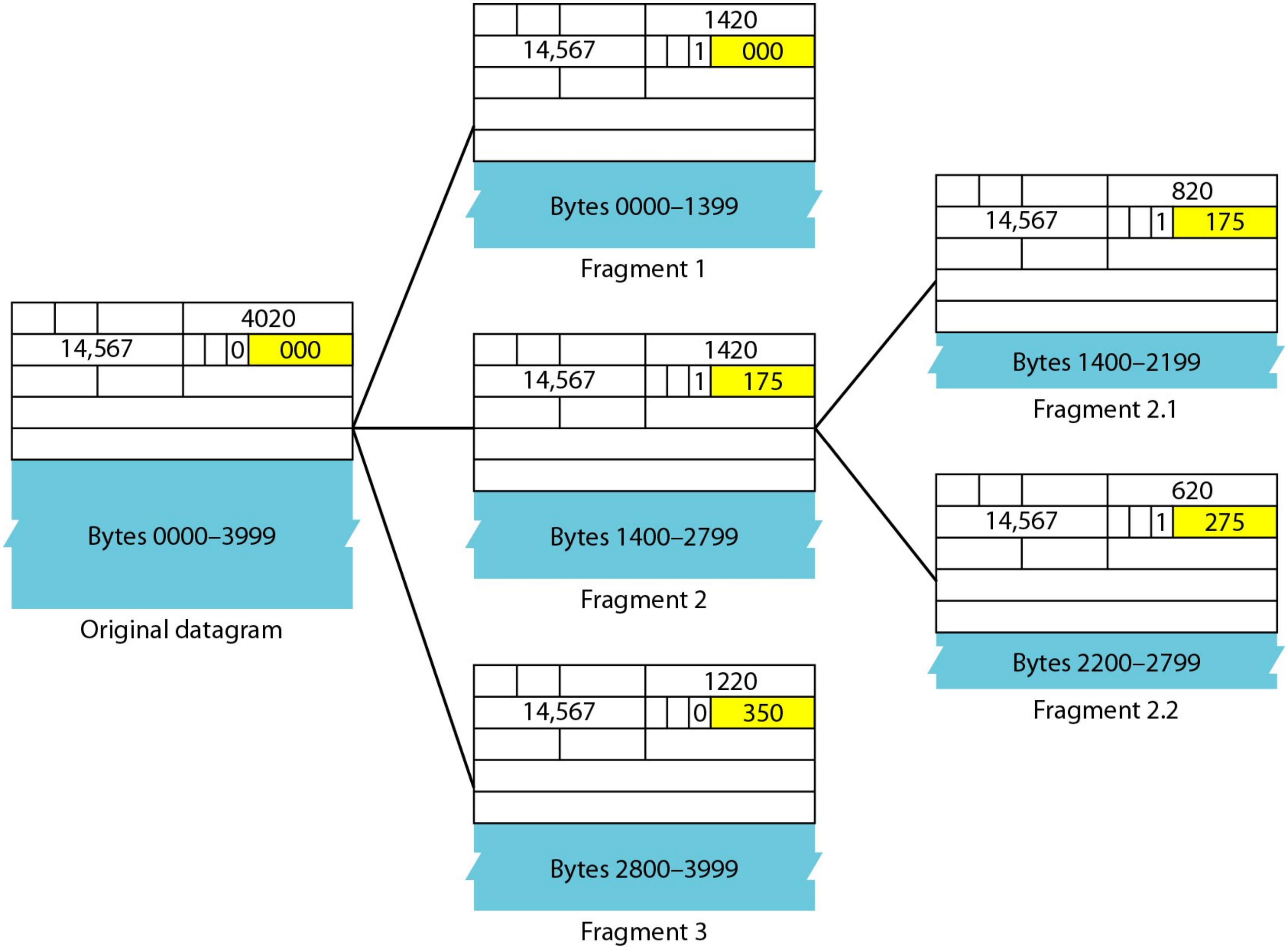
Fragmentation – an example (contd…)

❑ What happens if some more fragmentation needed at another intermediate router ?

✓ For Example if another intermediate router (where Fragment-2 reached and to be forwarded) next hop's MTU of is 820



Fragmentation – an example (contd…)



Dealing with failure

- Receiver starts reassembly timer when first fragment of a datagram is obtained
  - If timeout before all fragments arrive, discard all fragments of this datagram
  - Until the IP layer of the receiver has received all fragments of a datagram, it cannot hand the entire datagram to the higher layer
- IP does not guarantee delivery
  - Responsibility of higher layer to re-transmit packet
  - Routers attempt to inform source if packet discarded

Options field in IP header

- ❑ Options included primarily for network testing or debugging

- ✓ Examples
  - Source routing
  - Record route

How a router handles an IP datagram

- ❑ When a router gets an IP datagram
  - ➢ Extract data part, by stripping off IP header
  - ➢ Find outgoing interface using dest. IP and routing table
  - ➢ If data part > MTU of outgoing link to next hop
    - ✓ Fragment data part, put each fragment into a separate IP datagram
    - ✓ Put an IP header within each IP datagram
    - ✓ Copy fields: version, Type of Service, identification, protocol, source address, destination address, some options
    - ✓ Compute length and header checksum individually for each fragment
    - ✓ Put suitable flags and frame offset in each fragment
    - ✓ Put 1 less than TTL of original datagram as TTL in each fragment

Internet Control Message Protocol

ICMP

ICMP

- Every Network layer implementation must implement ICMP, along with IP and ARP
  - A required support protocol at the IP layer
- Used for reporting errors back to the source of an IP packet or for monitoring / measurement / feedback
  - When a node detects an error, an ICMP packet sent back to the source
  - Only error *reporting*, no error correction; correction is left to the source node

RFC 792 and RFC 1122

ICMP (contd.)

- ICMP packet
  - Contains ICMP header and may contain other information depending on type of message
  - Carried in data portion of an IP packet
  - The IP packet contains a IP header and is routed normally back to the source

- Examples of use of ICMP
  - Echo reply (to see if a host is up)
  - Subnet mask request and reply (among routers)
  - Router informs source about packet drop (may be due to unreachable destination, TTL exceeded, congestion)

References

- *Data Communications & Networking, 5th Edition, Behrouz A. Forouzan*
- *Computer Networks, Andrew S. Tanenbaum and David J. Wetherall*
- *Wikipedia*