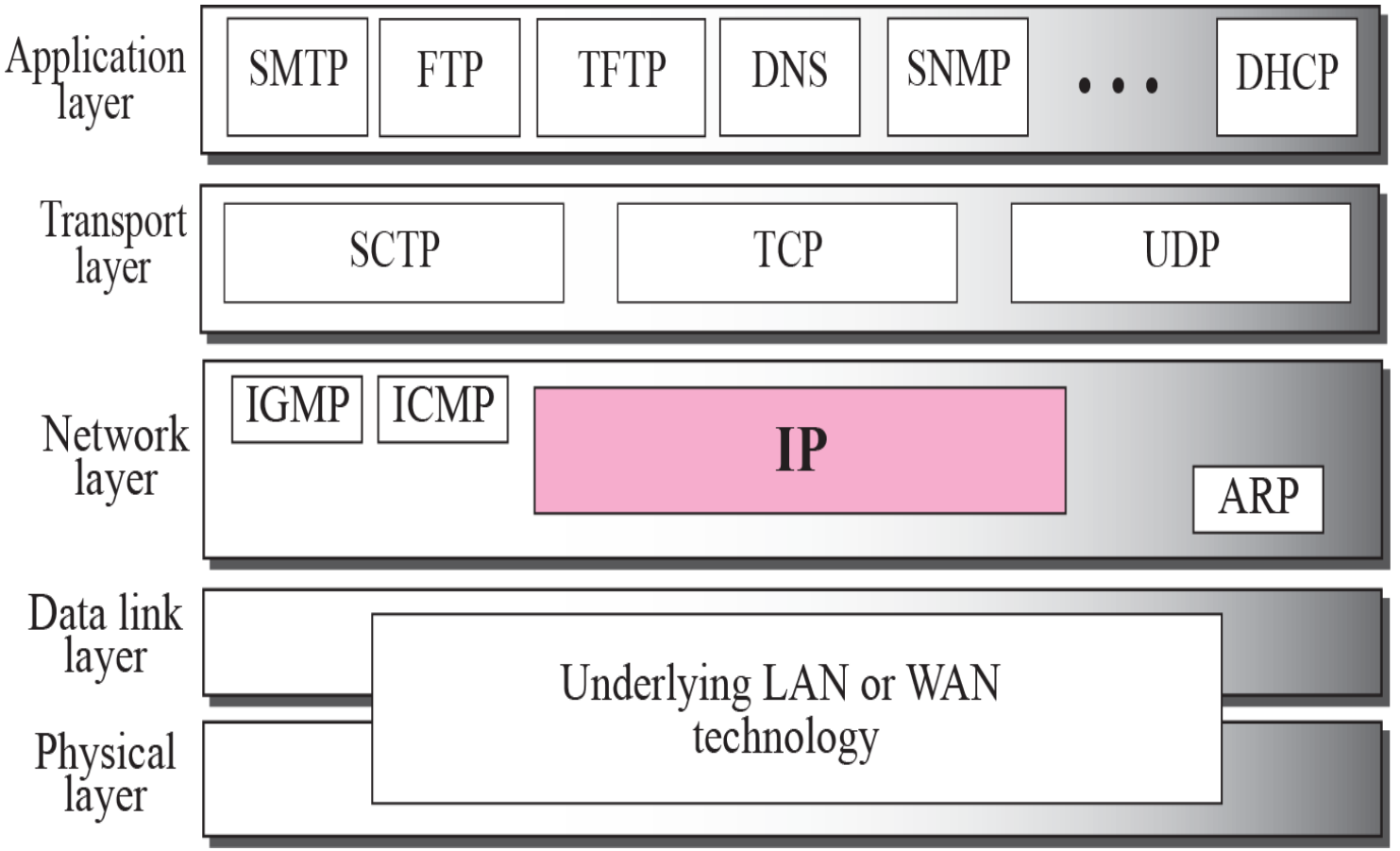
TCP/IP Protocol Suite

INTERNET PROTOCOL (IPV4)

Position of IP in TCP/IP protocol suite



IP Protocol – Needed Functions

- Logical addressing
- Packet format
  - Fragmentation & Reassembly
- Routing
- Forwarding
- Error Reporting

The Internet Protocol (IP)

- Provides a packet delivery service source to destination which is:
  - Unreliable
  - Best-effort
  - Connectionless
- Defines the basic unit of da
- Performs the routing function
- Includes a set of rules that embody the idea of unreliable packet delivery in packet switched networks
- The basic unit of data transfer is datagram
- It supports unicast, broadcast and multicast
- No Congestion Control
- Two versions IPV4 and IPV6

- Unreliable : IP doesn't make an attempt to recover lost packets
- Connectionless : Each packet is handled independently
- Best Effort : IP doesn't make guarantees on the service (No through output , No delay guarantee…)

IP Protocol

Internet Protocol (IP) – host-to-host network-layer delivery protocol for the Internet with following properties

- connectionless service – each packet is handled independently (possibly along different path)
- best-effort delivery service
  1) does its best to deliver packet to its destination, but with no guarantees
  2) limited error control – only error detection, corrupted packets are discarded
  3) no flow control
- must be paired with a reliable transport- (TCP) and/or application- layer protocol to ensure reliability

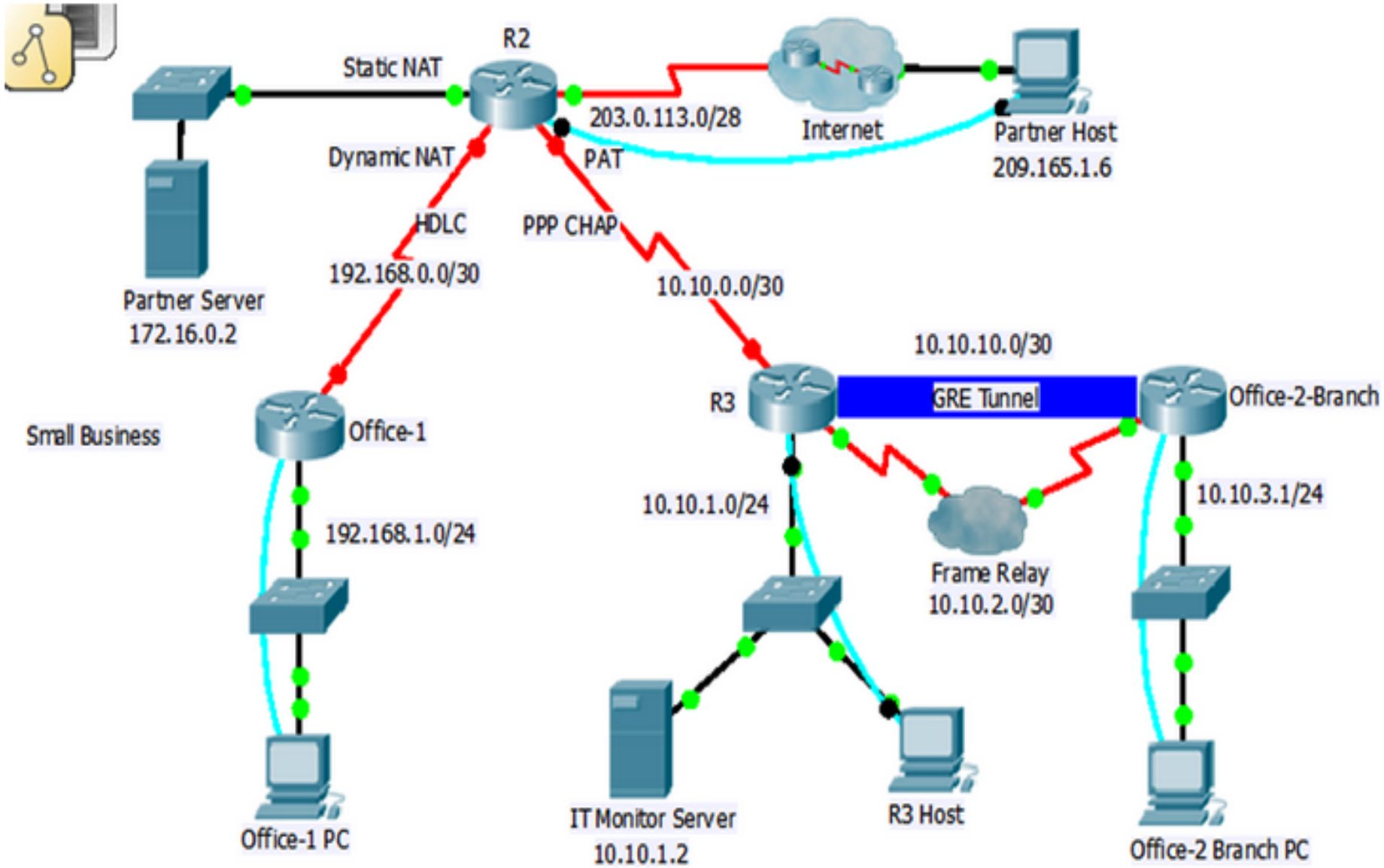
IP Protocol Versions

- IPv4 – version currently in wide use (1981)
- IPv6 – new version of IP protocol created to correct some of significant problems of IPv4 such as exhaustion of address space (1996)
- Mobile IP – enhanced version of IPv4 – supports IP in mobile environments (1996)

Goal of Internet Protocol

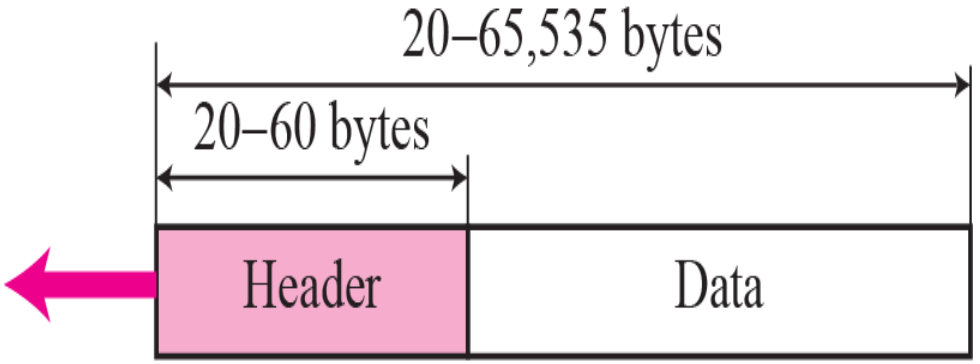
- To **interconnect heterogeneous networks**
  - LAN x to LAN y
  - WAN a to WAN b
  - LAN p to WAN q

Sample Network – Routers connecting to a various types of network – Physical Layer/Data link layer technologies - different



DATAGRAMS

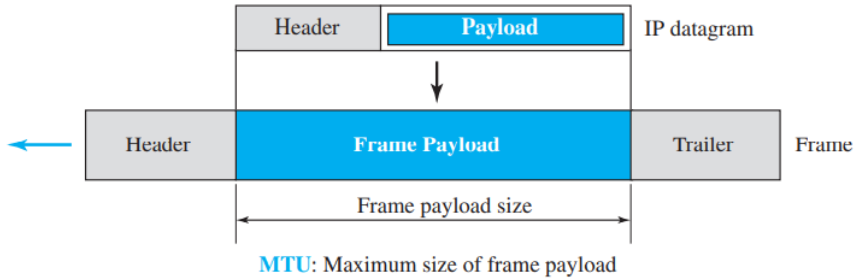
Packets in the network (internet) layer are called *datagrams*. A datagram is a variable-length packet consisting of two parts: header and data. The header is 20 to 60 bytes in length and contains information essential to routing and delivery. It is customary in TCP/IP to show the header in 4-byte sections.



a. IP datagram

Problem Faced

- Various Physical Layer technologies have **DIFFERENT**

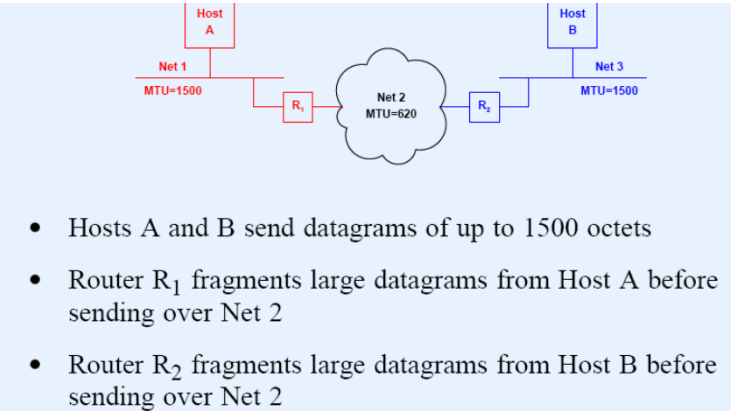


  - Frame formats

  - maximum amount of data that link-layer frame can carry
  - If size of MTU outweighs the capacity of router, it is re-transmitted again causing delay.

Maximum Transfer Unit (MTU)

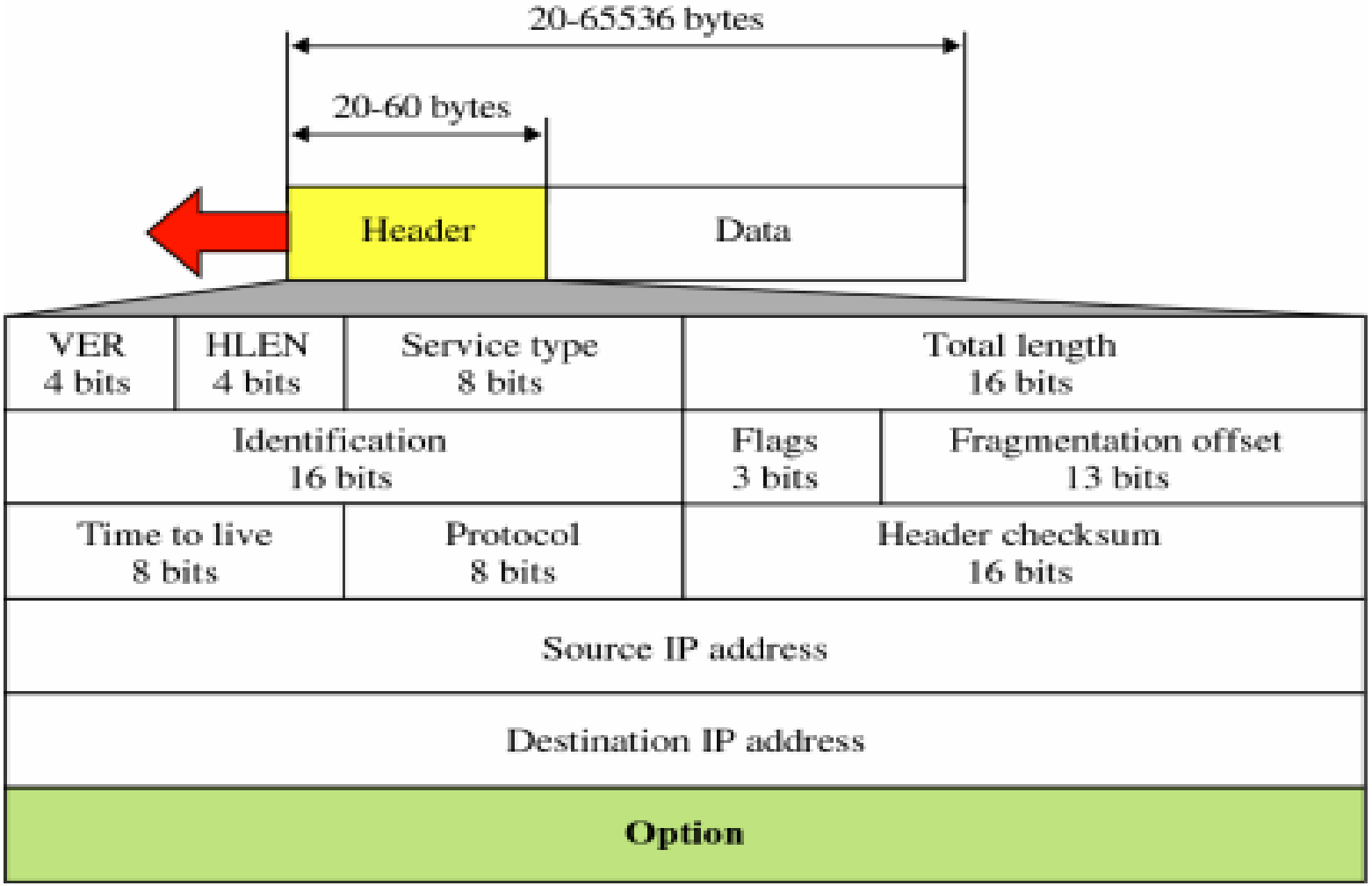
- Each physical networking technology limits the amount of data that can fit in a frame
  - Ethernet: 1500 octets
  - FDDI: 4470 octets
  - PPP: 296 bytes
  - WiFi: 7981 bytes



- This is called the network's MTU
- Limiting datagrams to fit in the smallest possible MTU would make travelling across networks with a larger MTU inefficient
- Allowing datagrams to be larger than a network's MTU means that datagrams will not always fit in a single frame

What is a problem is that each of the links along the route between sender and receiver can use different link-layer protocols, and each of these protocols can have different MTUs.

IP datagram - Header



Datagram – **IP packet = variable length packet consisting of header & data**

- **header – 20 to 60 bytes in length, contains information essential to routing and delivery**
- **data – length determined by Maximum Transmission Unit (MTU) of link layer protocol** (theoretically between 20 to 65536 bytes)

Version Number – 4-bit field – specifies IP protocol version of the datagram (IPv4 or IPv6)

- different version of IP use different datagram formats
- by looking at version number router can determine how to interpret remainder of datagram

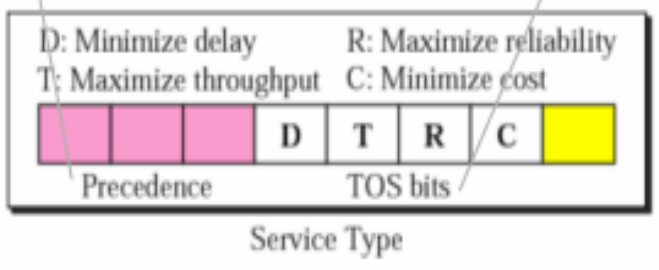
Header Length – 4-bit field – defines total length of datagram header in 4-byte words

- when there are no options header length is 20 ⇒ HLEN = 5

Differentiated Service (formerly Service Type) – 8-bit field – allows different types of datagrams to be distinguished from each other based on their associated / requested QoS

- e.g. datagrams particularly requiring low delay, high throughput, or reliability

Precedence defines the priority of datagram in case of congestion. If a router is congested and needs to discard some datagrams, those datagrams with lowest precedence are discarded first. **Network management datagrams have the highest precedence!**

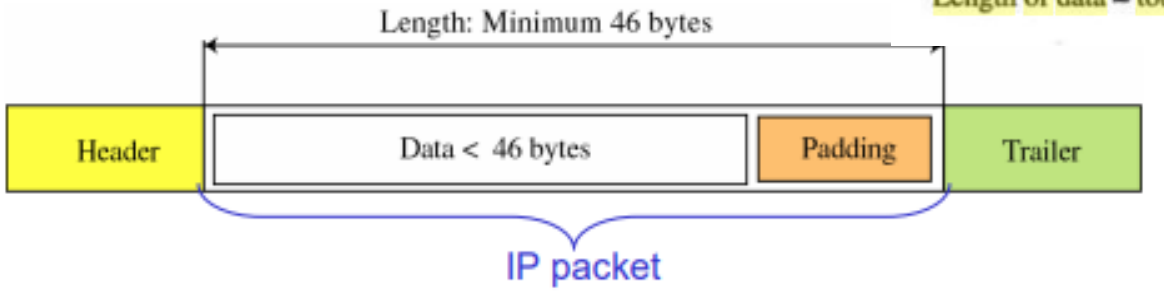


Although each TOS bit has a special meaning, only one bit can be set to 1 in each datagram.
0000 – normal type of service
0001 – minimize cost
0010 – maximize reliability
0100 – maximize throughput
1000 – minimize delay

Total Length – 16-bit field – defines total datagram length in bytes, including header

- 16 bits ⇒ maximum size = 65,535 bytes
- some physical networks are not able to encapsulate a datagram of 65,535 bytes, so datagram must be fragmented to be able to pass through those networks
- some physical networks have restriction on minimum size of data that can be encapsulated in a frame, so datagram must be padded (e.g. Ethernet min size of data – 46 bytes)

Length of data = total length – header length



Identifier, Flags, Fragmentation Offset – 3 fields used in fragmentation

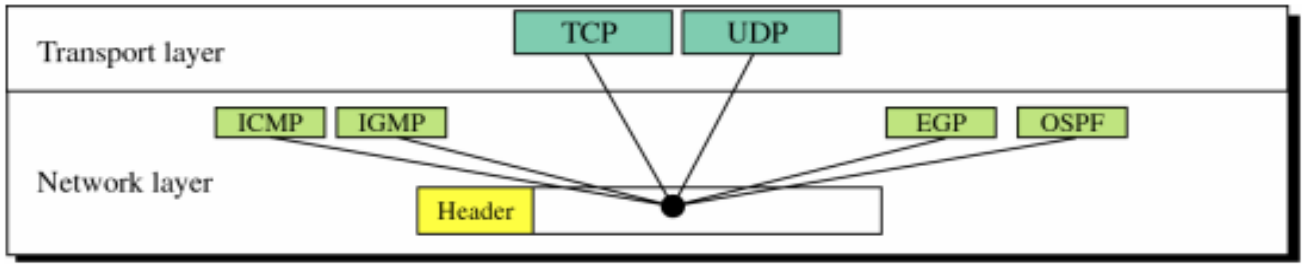
- IPv6 does not allow fragmentation at routers since it is time consuming operation – if an IPv6 packet is too big, it is simply dropped and an ICMP message is sent back to the source

Time-To-Live (TTL) – **8-bit field – controls max number of hops visited by datagram and/or time spend in the network**

- **field is decremented by one each time datagram is processed by a router – when TTL reaches 0, datagram must be dropped**
- **ensures that**
  1) **datagram does not circulate/loop forever, or**
  2) **to limit its journey** (e.g. LAN only: TTL = 1)

Protocol – **8-bit field – indicates specific transport-layer protocol to which data portion of this IP datagram should be passed**

- **used only at final destination to facilitate demultiplexing process**
- protocol number is glue that binds network & transport layer, while port number is glue that binds transport & application layer
- **values: 1 – ICMP, 2 – IGMP, 6 – TCP, 17 – UDP, 89 – OSPF**



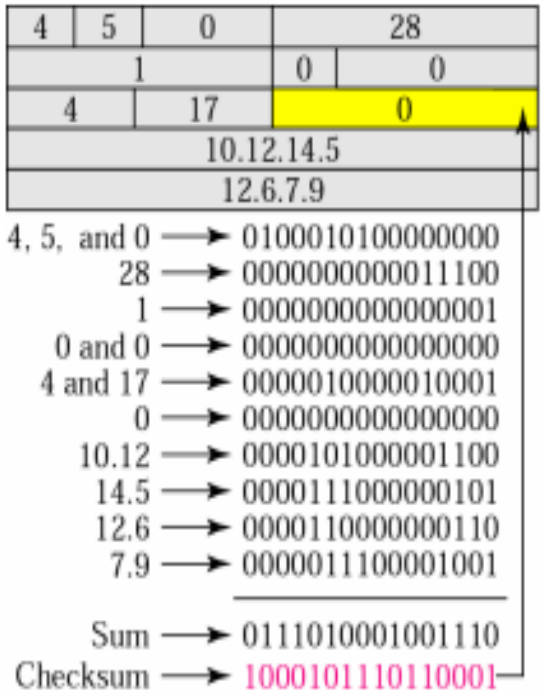
Header Checksum – 16-bit field – aids in detecting errors in header only!

- checksum must be recomputed & stored again at each router as TTL and some options fields may change
- routers discard datagrams for which an error is detected
- checksum calculation:
  1) divide header into 16-bit (2-byte) sections – checksum field itself is set to 0
  2) sum all sections using 1s complement arithmetic

Each intermediate router must:

1) verify / recompute checksum on every incoming packet

2) compute checksum for every outgoing packet



Checksum in IP covers only the header, not the data.

OPTIONS in IP Datagram

- The header of the IP datagram is made of two parts: a fixed part and a variable part.
- The fixed part is 20 bytes long.
- The variable part comprises the options, which can be a maximum of 40 bytes.
- Options, as the name implies, are not required for a datagram.
- They can be used for network testing and debugging.

Option format

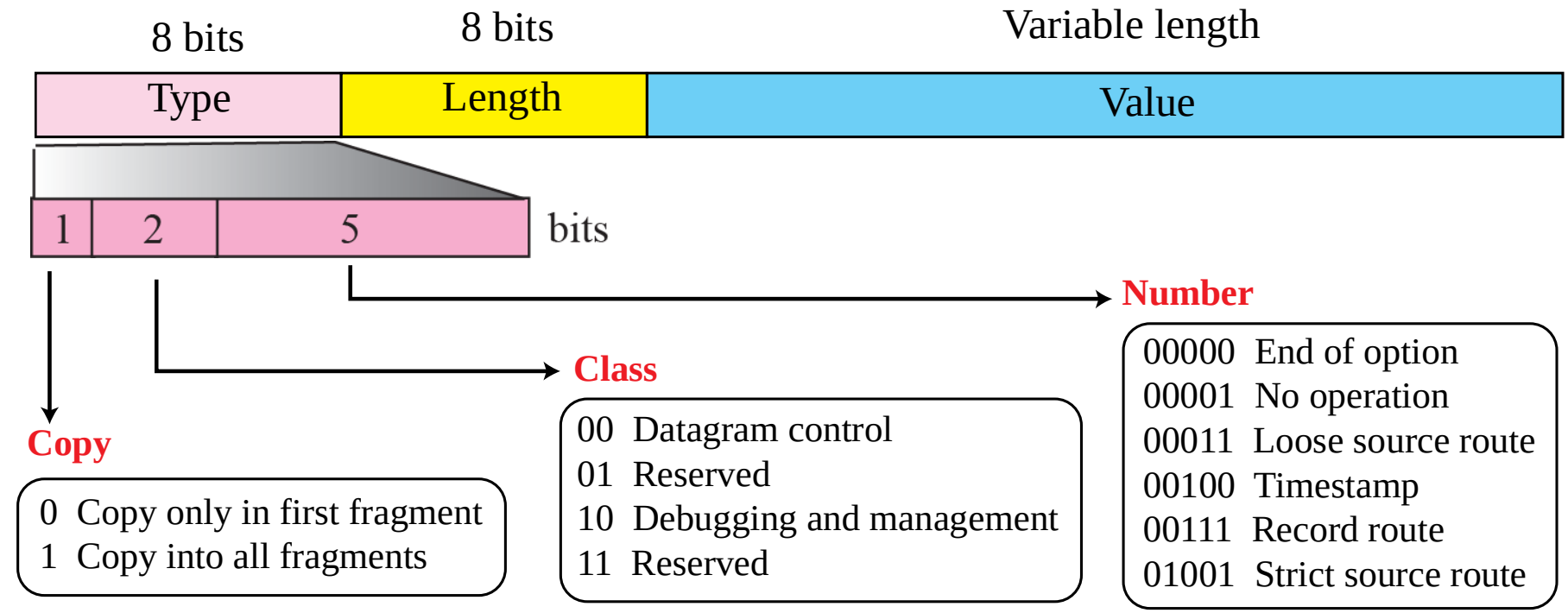
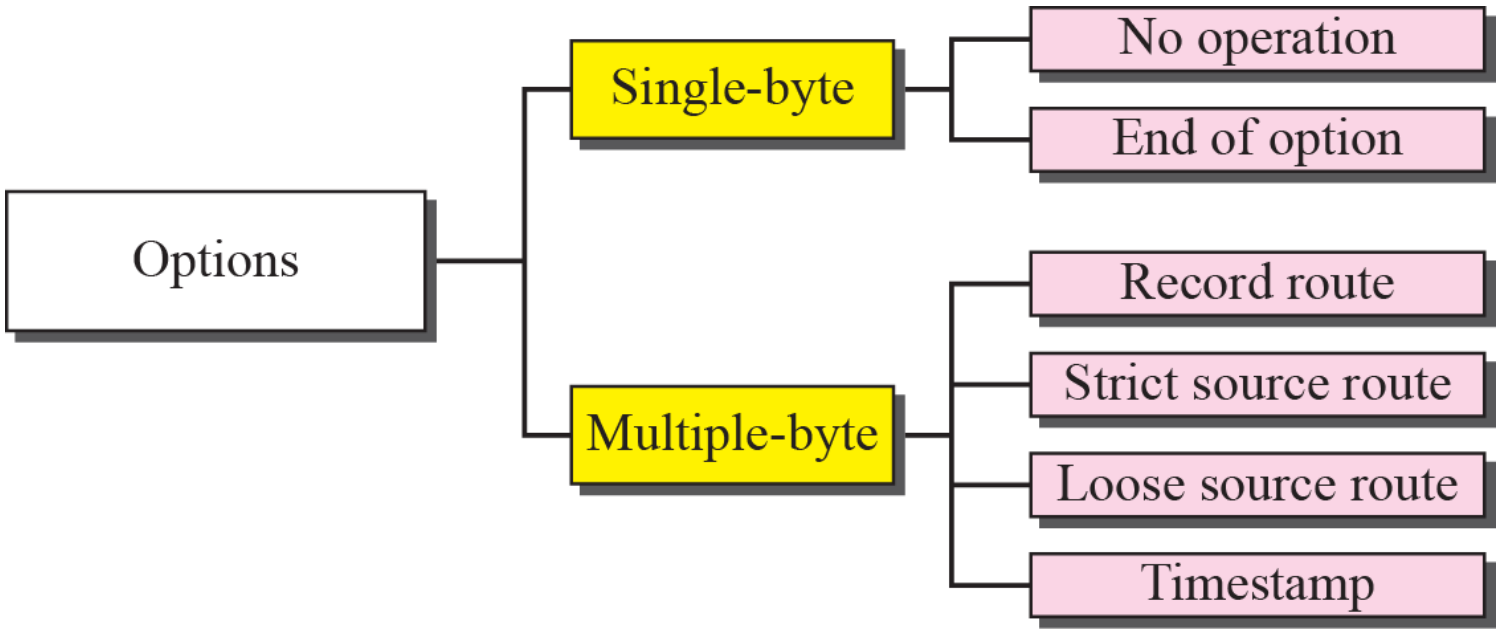


Figure 7.11 *Categories of options*

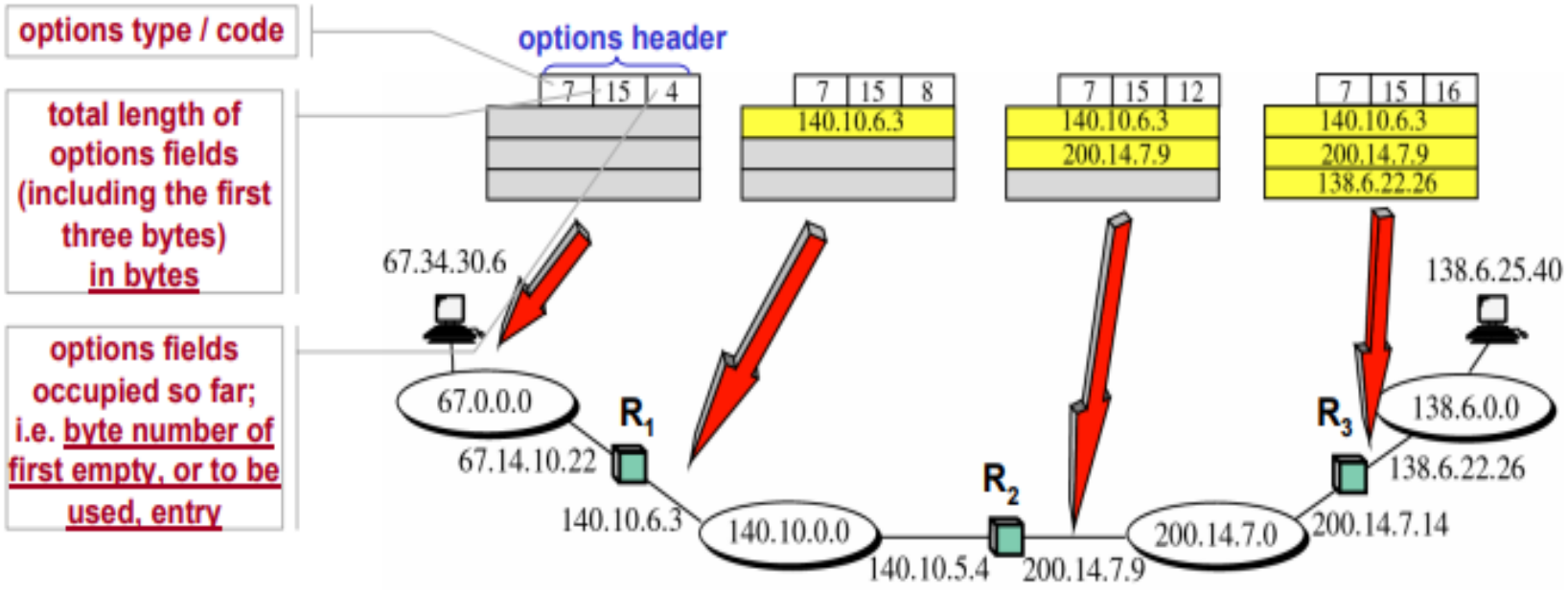


Source and Destination IP Addresses – 32-bit fields – must remain unchanged until IP datagram reaches its final destination

Options – 32-bit field(s) – not required for every datagram! – allows expansion of IP header for special purposes

(a) Record Route option – used to trace route that datagram takes

- source creates empty fields for IP addresses – up to 9
  (40 bytes options – 4 bytes option header) / 4 bytes for IP address
- each router that processes datagram inserts its outgoing IP address



Example

We can also use the ping utility with the -R option to implement the record route option. The result shows the interfaces and IP addresses.

```
$ ping -R fhda.edu
PING fhda.edu (153.18.8.1) 56(124) bytes of data.
64 bytes from tiptoe.fhda.edu
(153.18.8.1): icmp_seq=0 ttl=62 time=2.70 ms
RR:  voyager.deanza.fhda.edu (153.18.17.11)
     Dcore_G0_3-69.fhda.edu (153.18.251.3)
     Dbackup_V13.fhda.edu (153.18.191.249)
     tiptoe.fhda.edu (153.18.8.1)
     Dbackup_V62.fhda.edu (153.18.251.34)
     Dcore_G0_1-6.fhda.edu (153.18.31.254)
     voyager.deanza.fhda.edu (153.18.17.11)
```

Example

The traceroute utility can also be used to keep track of the route of a packet. The result shows the three routers visited.

```
$ traceroute fhda.edu
traceroute to fhda.edu (153.18.8.1), 30 hops max, 38 byte packets
 1 Dcore_G0_1-6.fhda.edu (153.18.31.254)  0.972 ms  0.902 ms
   0.881 ms
 2 Dbackup_V69.fhda.edu (153.18.251.4)  2.113 ms  1.996 ms
   2.059 ms
 3 tiptoe.fhda.edu (153.18.8.1)  1.791 ms  1.741 ms  1.751 ms
```

Example

The traceroute program can be used to implement loose source routing. The -g option allows us to define the routers to be visited, from the source to destination. The following shows how we can send a packet to the fhda.edu server with the requirement that the packet visit the router 153.18.251.4.

```
$ traceroute -g  153.18.251.4 fhda.edu.
traceroute to fhda.edu (153.18.8.1), 30 hops max, 46 byte packets
 1  Dcore_G0_1-6.fhda.edu (153.18.31.254)  0.976 ms  0.906 ms
    0.889 ms
 2  Dbackup_V69.fhda.edu (153.18.251.4)  2.168 ms  2.148 ms
    2.037 ms
```

Example

The traceroute program can also be used to implement strict source routing. The -G option forces the packet to visit the routers defined in the command line. The following shows how we can send a packet to the fhda.edu server and force the packet to visit only the router 153.18.251.4.

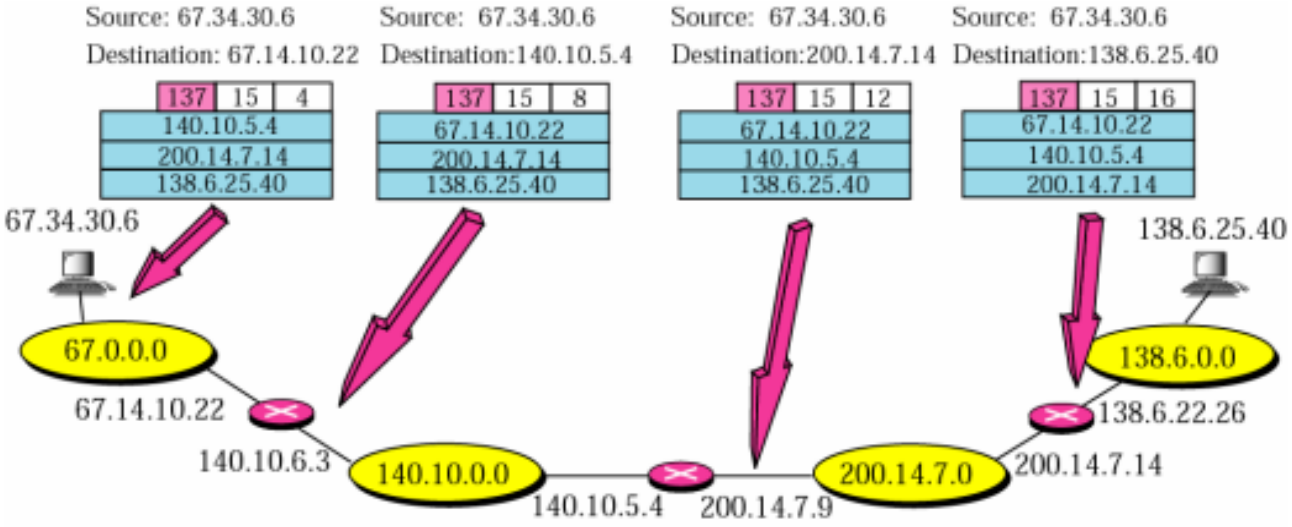
```
$ traceroute -G  153.18.251.4 fhda.edu.
traceroute to fhda.edu (153.18.8.1), 30 hops max, 46 byte packets
  1  Dbackup_V69.fhda.edu (153.18.251.4)  2.168 ms  2.148 ms
     2.037 ms
```

Options (cont.)

(b) **Timestamp option** – similar to (a), plus records datagram end-processing time by each router, in milliseconds

(c) **Strict Source Route option** – used by source to predetermine route for datagram

- source provides a list of IP addresses (sequence of routers) that datagram must (is allowed) to visit on its way to destination



(d) **Loose Source Route option** – similar to (c), but it is more relaxed – each router in the list must be visited, though datagram can visit other routers as well

Example 7.1

An IP packet has arrived with the first 8 bits as shown:

```
01000010
```

The receiver discards the packet. Why?

Example 7.1

An IP packet has arrived with the first 8 bits as shown:

```
01000010
```

The receiver discards the packet. Why?

Solution

There is an error in this packet. The 4 left-most bits (0100) show the version, which is correct. The next 4 bits (0010) show the wrong header length (2 × 4 = 8). The minimum number of bytes in the header must be 20. The packet has been corrupted in transmission.

Example 7.2

In an IP packet, the value of HLEN is 1000 in binary. How many bytes of options are being carried by this packet?

Solution

The HLEN value is 8, which means the total number of bytes in the header is 8 × 4 or 32 bytes. The first 20 bytes are the base header, the next 12 bytes are the options.

Example 7.3

In an IP packet, the value of HLEN is 5_{16} and the value of the total length field is 0028_{16}. How many bytes of data are being carried by this packet?

Solution

The HLEN value is 5, which means the total number of bytes in the header is 5 × 4 or 20 bytes (no options).

The total length is 40 bytes, which means the packet is carrying 20 bytes of data (40 − 20).

Example 7.4

An IP packet has arrived with the first few hexadecimal digits as shown below:

```
45000028000100000102 . . .
```

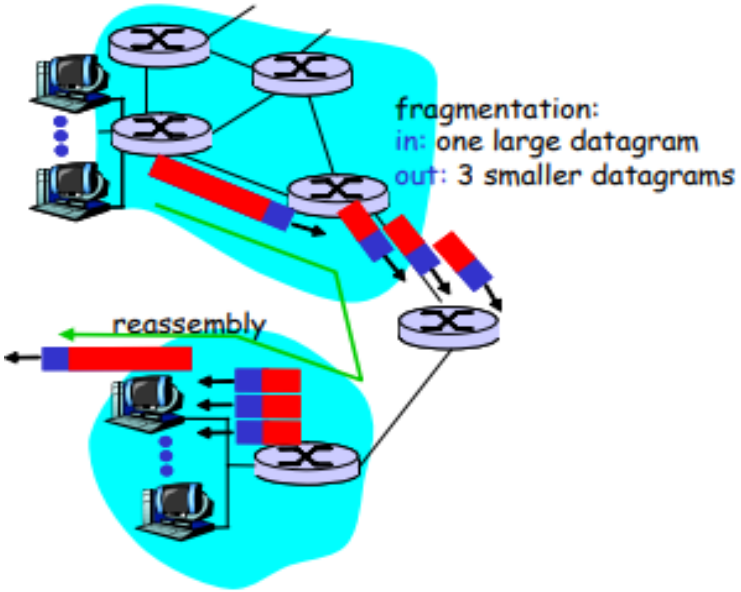
How many hops can this packet travel before being dropped? The data belong to what upper layer protocol?

Solution

To find the time-to-live field, we skip 8 bytes (16 hexadecimal digits). The time-to-live field is the ninth byte, which is 01. This means the packet can travel only one hop. The protocol field is the next byte (02), which means that the upper layer protocol is IGMP

FRAGMENTATION

IP Datagram Fragmentation – process of dividing datagram into smaller fragments that meet MTU requirements of underlying data-link layer protocol



- datagram can be fragmented by source-host or any other router in the path; however reassembly of datagram is done only by destination host! – parts of a fragmented datagram may take different routes !!!
- once fragmented datagram may be further fragmented if it encounters network with even smaller MTU
- when a datagram is fragmented, each fragment gets its own header with most fields repeated, but some changed
  - host or router that fragments datagram must change values of three fields: *flags*, *fragmentation offset* and *total length*

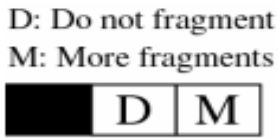
IP Fragmentation

Identification – 16-bit field – uniquely identifies datagram originating from source host

- to guarantee uniqueness, IP uses counter to label each datagram
- when IP sends a datagram, it copies current counter value to identification field, and increments counter by one
- when datagram is fragmented, identification field is copied into all fragments
- identification number helps destination in reassembling datagram – all fragments with same identification value should be assembled into one datagram

Flags – 3-bit field

- 1st bit is reserved
- 2nd bit is called "do not fragment" bit
  - if its value is 1, machine must NOT fragment datagram
  - if fragment cannot pass through physical network router discards packet and sends ICMP error message back to source host
- 3rd bit is called "more fragment" bit
  - if its value is 1, datagram is not last fragment – there are more fragments after this one
  - if its value is 0, this is last or only fragment



IP Fragmentation

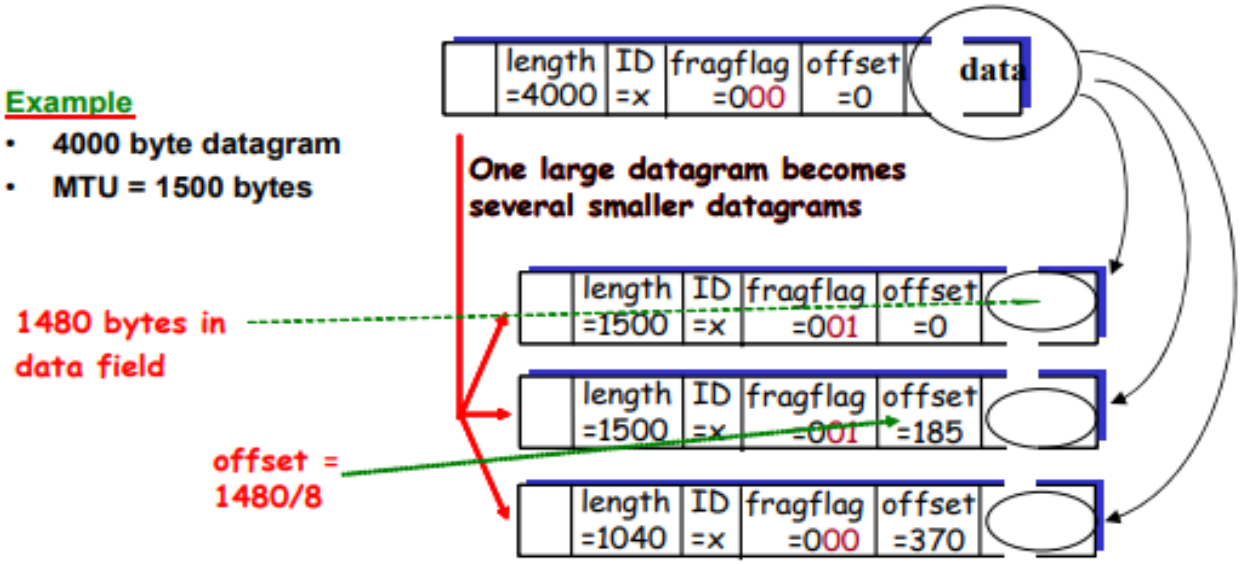
Fragmentation Offset – **13-bit field – shows relative position of fragment's data with respect to whole datagram**

- **the offset is measured in units of 8 bytes – this is done because offset field is only 13 bits long and otherwise could not represent sequences greater than 8191**
- **this forces hosts and routers to choose fragment sizes divisible by 8**

Example [fragmentation]

Example

- **4000 byte datagram**
- **MTU = 1500 bytes**



Fragmentation Process

Case-01:

- Size of the datagram is found to be smaller than or equal to MTU.
- In this case, router transmits the datagram without any fragmentation.

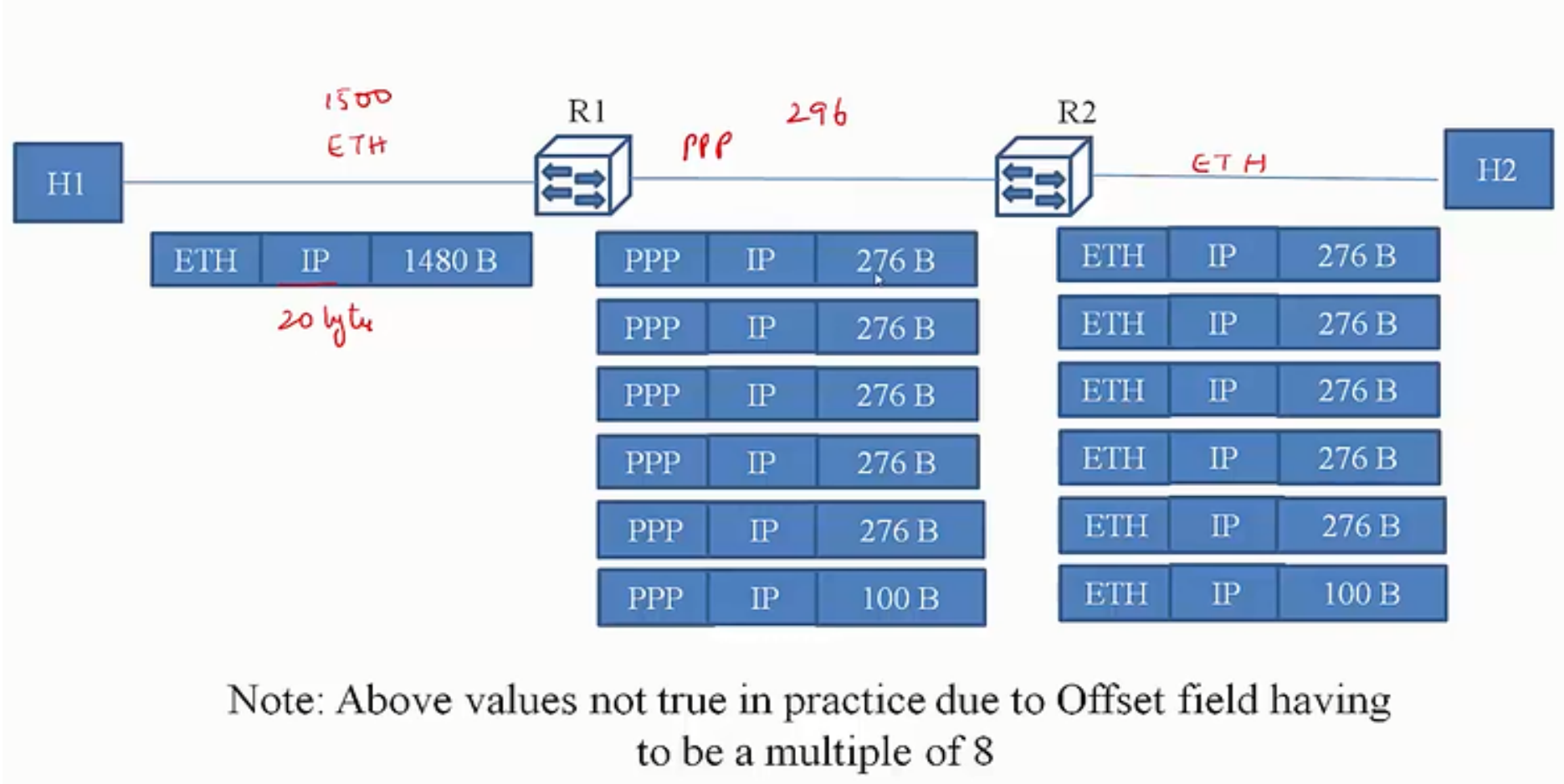
Case-02:

- Size of the datagram is found to be greater than MTU and DF bit set to 1.
- In this case, router discards the datagram.

Case-03:

- Size of the datagram is found to be greater than MTU and DF bit set to 0.
- In this case, router divides the datagram into fragments of size less than or equal to MTU.
- Router attaches an IP header with each fragment making the following changes in it.
- Then, router transmits all the fragments of the datagram.

Need for Fragmentation (Concept)



Note: Above values not true in practice due to Offset field having to be a multiple of 8

IP Fragmentation Example

An IP Datagram of length 5140 bytes is routed thru Ethernet with MTU 1500 bytes

Original IP Datagram

Sequence	Identifier	Total Length	DF May / Don't	MF Last / More	Fragment Offset
0	345	5140	0	0	0

Example 2:

An IP Datagram of length 5140 bytes is routed thru Ethernet with MTU 1500 bytes

Original IP Datagram

Sequence	Identifier	Total Length	DF May / Don't	MF Last / More	Fragment Offset
0	345	5140	0	0	0

IP Fragments (Ethernet)

Sequence	Identifier	Total Length	DF May / Don't	MF Last / More	Fragment Offset
0-0	345	1500	0	1	0
0-1	345	1500	0	1	185
0-2	345	1500	0	1	370
0-3	345	700	0	0	555

Datagram Reassembly from Fragments

- Fragments from multiple datagrams can arrive out-of-order
  - Individual fragments can be lost or arrive out-of-order
- How does it reassemble fragments that arrive out-of-order?
- A sender places a unique identification number in the IDENTIFICATION field of each outgoing datagram
- When a router fragments a datagram
  - the router copies the identification number into each fragment
- A receiver uses the identification number and IP source address in an incoming fragment
  - to determine the datagram to which the fragment belongs
- The FRAGMENT OFFSET field tells a receiver where data in the fragment belongs in the original datagram

Why Reassembly at End Host?

- Fragments travel independent of each other in the intermediate networks. Sometimes, the fragments of the same datagram might take different paths.
- If an intermediate router has to take care of reassembly, there will be some state information required to maintained for the fragments of the datagram. But, IP is a stateless, connectionless protocol.

Consequence of Fragment Loss

- A datagram cannot be reassembled until all fragments arrive
- A problem arises when one or more fragments from a datagram arrive and other fragments are delayed or lost
- The receiver must save (buffer) the fragments
  - that have arrived in case missing fragments are only delayed
- A receiver cannot hold fragments an arbitrarily long time
  - because fragments occupy space in memory
- IP specifies a maximum time to hold fragments
- When the first fragment arrives from a given datagram
  - the receiver starts a reassembly timer
- If all fragments of a datagram arrive before the timer expires
  - the receiver cancels the timer and reassembles the datagram
- If the timer expires before all fragments arrive
  - the receiver discards the fragments that have arrived

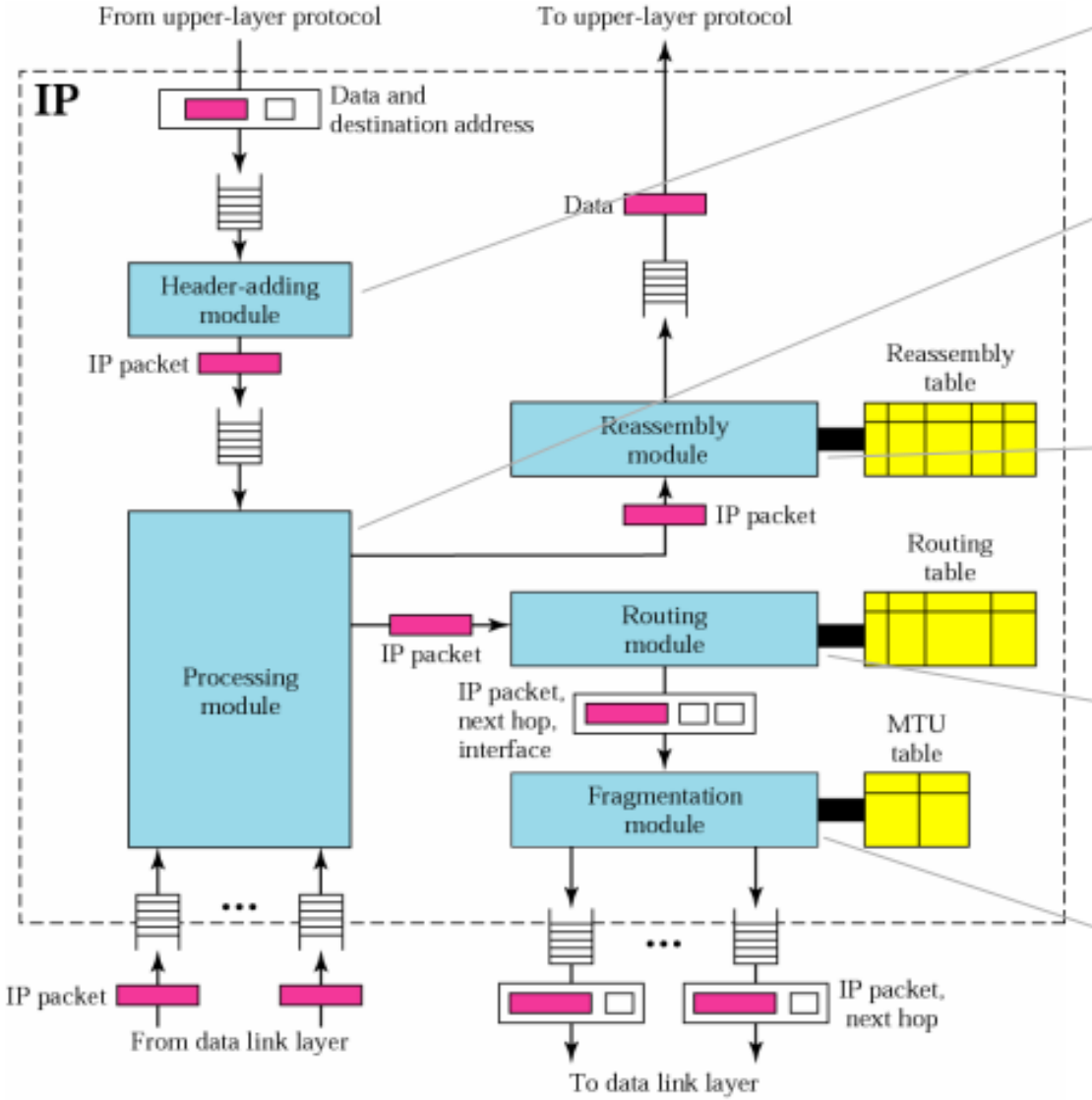
Consequence of Fragment Loss

- The result of IP's reassembly timer is all-or-nothing:
  - either all fragments arrive and IP reassembles the datagram,
  - If not then IP discards the incomplete datagram
- There is no mechanism for a receiver to tell the sender which fragments have arrived
  - The sender does not know about fragmentation
- If a sender retransmits, the datagram routes may be different
  - a retransmission would not necessarily traverse the same routers
    - also, there is no guarantee that a retransmitted datagram would be fragmented in the same way as the original

Fragmenting a Fragment

- What happens if a fragment eventually reaches a network that has a smaller MTU?
- It is possible to fragment a fragment when needed
  - A router along the path divides the fragment into smaller fragments
- If networks are arranged in a sequence of decreasing MTUs
  - each router along the path must further fragment each fragment
- Designers work carefully to ensure that such situations do not occur in the Internet

IP Datagram Processing



Header-adding module
encapsulates the data from the transport-layer protocol by adding the IP header

Processing module
1) checks if destination IP address = host IP address
2) decrements TTL by 1
(done only at router-nodes)

Reassembly module
orders and reassembles the fragments belonging to the same datagram. Reassembly table keeps track of received fragments

Routing module
finds the IP address of the next station along with the interface number to which the packet should be sent.

Fragmentation module
consults MTU table to find the MUT for the specific interface number.

Example 7.5

A packet has arrived with an M bit value of 0. Is this the first fragment, the last fragment, or a middle fragment? Do we know if the packet was fragmented?

Example 7.5

A packet has arrived with an M bit value of 0. Is this the first fragment, the last fragment, or a middle fragment? Do we know if the packet was fragmented?

Solution

If the M bit is 0, it means that there are no more fragments; the fragment is the last one. However, we cannot say if the original packet was fragmented or not. A nonfragmented packet is considered the last fragment.

Example 7.6

A packet has arrived with an M bit value of 1. Is this the first fragment, the last fragment, or a middle fragment? Do we know if the packet was fragmented?

Solution

If the M bit is 1, it means that there is at least one more fragment. This fragment can be the first one or a middle one, but not the last one. We don't know if it is the first one or a middle one; we need more information (the value of the fragmentation offset).

Example 7.7

A packet has arrived with an M bit value of 1 and a fragmentation offset value of zero. Is this the first fragment, the last fragment, or a middle fragment?

Solution

Because the M bit is 1, it is either the first fragment or a middle one. Because the offset value is 0, it is the first fragment.

Example 7.8

A packet has arrived in which the offset value is 100. What is the number of the first byte? Do we know the number of the last byte?

Solution

To find the number of the first byte, we multiply the offset value by 8. This means that the first byte number is 800. We cannot determine the number of the last byte unless we know the length of the data.

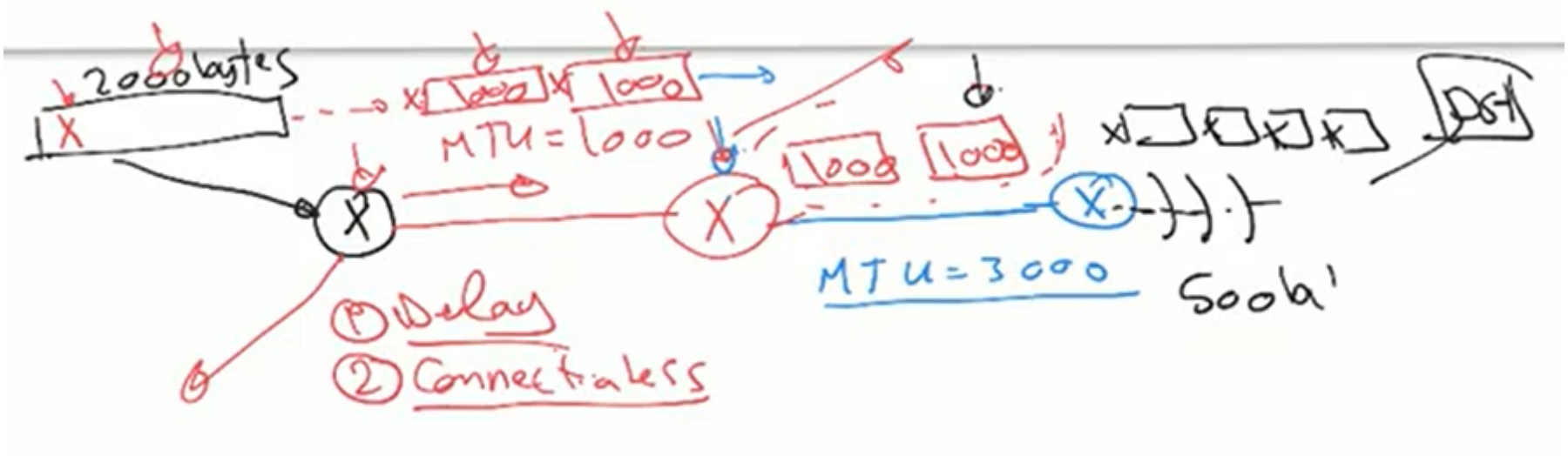
Example

A packet has arrived in which the offset value is 100, the value of HLEN is 5 and the value of the total length field is 100. What is the number of the first byte and the last byte?

Solution

The first byte number is 100 × 8 = 800. The total length is 100 bytes and the header length is 20 bytes (5 × 4), which means that there are 80 bytes in this datagram. If the first byte number is 800, the last byte number must be 879.

IP Datagram Forwarding


2000 bytes
X
1000
1000
MTU = 1000
1000
1000
MTU = 3000
Sooh!
① Delay
② Connectionless

Fragmentation process

- Routers split the packets that are too larger
- It breaks the larger packets into smaller packets
- Copy the IP header into those smaller packets
- Sets the fragmentation offset to indicate position
- Sets MF (more fragments) on the smaller packets except the last packet
- Receiving host reassembles the smaller packets
- Identification field links the smaller packets together
- MF bit tells the receiver when it has all the packets that got fragmented