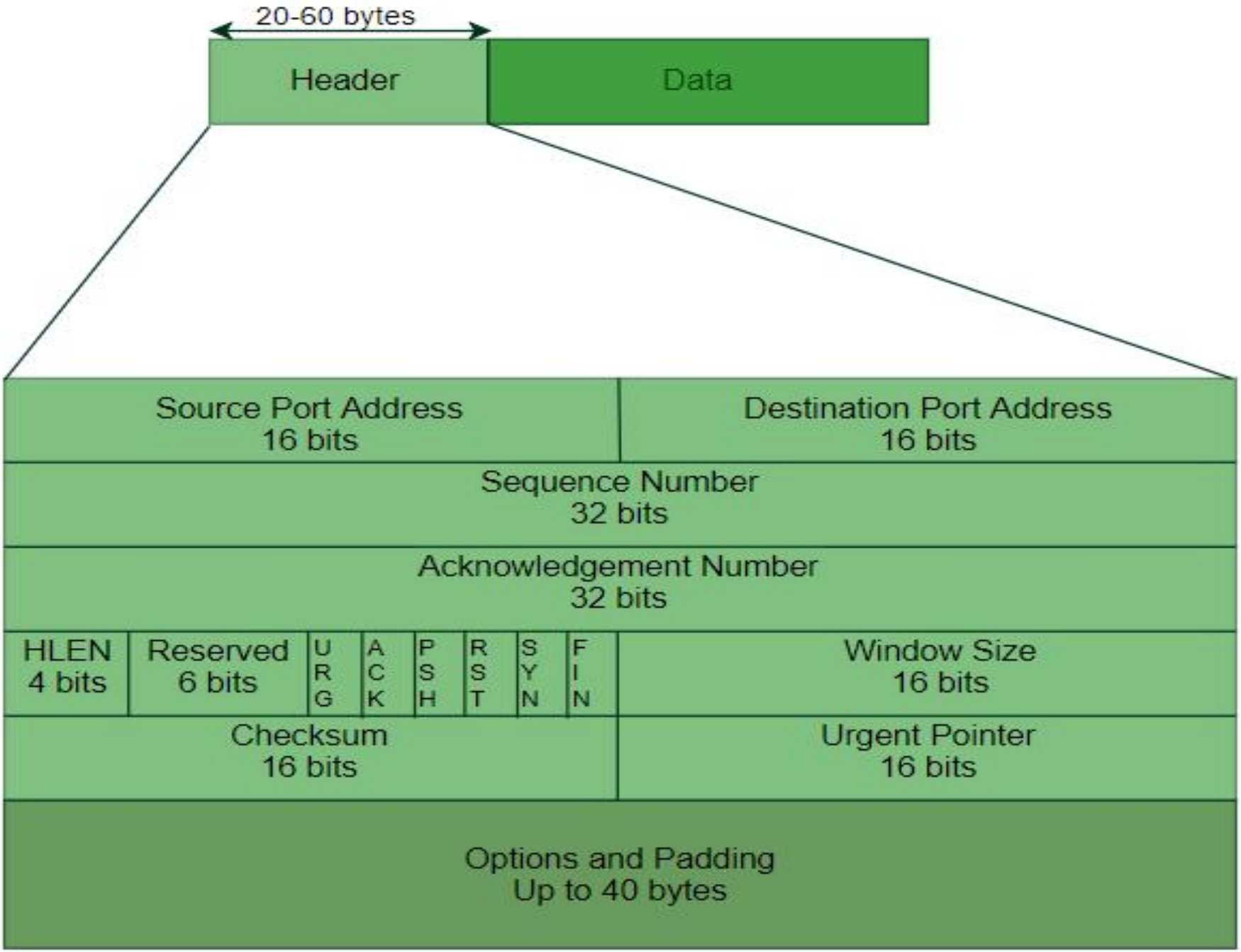
TCP and UDP protocols

Presented by

Mariena A. A.

Asst. Professor and Head, Department of Computer Science

Little Flower College, Guruvayoor.


20-60 bytes
Header
Data
Source Port Address
16 bits
Destination Port Address
16 bits
Sequence Number
32 bits
Acknowledgement Number
32 bits
HLEN
4 bits
Reserved
6 bits
URG
ACK
PSH
RST
SYN
FIN
Window Size
16 bits
Checksum
16 bits
Urgent Pointer
16 bits
Options and Padding
Up to 40 bytes

The header of a TCP segment can range from 20-60 bytes. 40 bytes are for options. If there are no options, header is of 20 bytes else it can be of upmost 60 bytes. Header fields:

Source Port Address –

16 bit field that holds the port address of the application that is sending the data segment.

Destination Port Address –

16 bit field that holds the port address of the application in the host that is receiving the data segment.

Sequence Number –

32 bit field that holds the sequence number, i.e, the byte number of the first byte that is sent in that particular segment. It is used to reassemble the message at the receiving end if the segments are received out of order.

Acknowledgement Number –
32 bit field that holds the acknowledgement number, i.e, the byte number that the receiver expects to receive next. It is an acknowledgment for the previous bytes being received successfully.
Header Length (HLEN) –
This is a 4 bit field that indicates the length of the TCP header by number of 4-byte words in the header, i.e, if the header is of 20 bytes(min length of TCP header), then this field will hold 5 (because 5 x 4 = 20) and the maximum length: 60 bytes, then it'll hold the value 15(because 15 x 4 = 60). Hence, the value of this field is always between 5 and 15.
Window size –
This field tells the window size of the sending TCP in bytes.

- **Control flags –**
  These are 6 1-bit control bits that control connection establishment, connection termination, connection abortion, flow control, mode of transfer etc. Their function is:
  - URG: Urgent pointer is valid
  - ACK: Acknowledgement number is valid(used in case of cumulative acknowledgement)
  - PSH: Request for push
  - RST: Reset the connection
  - SYN: Synchronize sequence numbers
  - FIN: Terminate the connection

The **flag field** contains 6 bits. The **ACK bit** is used to indicate that the value carried in the acknowledgement for a segment that has been successfully received.
The **RST**, **SYN**, and **FIN** bits are used for connection setup and teardown. Setting the **PSH** bit indicates that the receiver should pass the data to the upper layer immediately.

Finally, the **URG** bit is used to indicate that there is data in this segment that the sending-side upper-layer entity has marked as “urgent”. The location of the last byte of this urgent data is indicated by the 16-bit **urgent data pointer field**.

TCP must inform the receiving-side upper-layer entity when urgent data exists and pass it to a pointer to the end of the urgent data.

Checksum –

This field holds the checksum for error control. It is mandatory in TCP as opposed to UDP.

Urgent pointer –

This field (valid only if the URG control flag is set) is used to point to data that is urgently required that needs to reach the receiving process at the earliest. The value of this field is added to the sequence number to get the byte number of the last urgent byte.

TCP services

Process-to-Process Communication –

TCP provides process to process communication, i.e, the transfer of data takes place between individual processes executing on end systems. This is done using port numbers or port addresses. Port numbers are 16 bit long that help identify which process is sending or receiving data on a host.

Stream oriented –

This means that the data is sent and received as a stream of bytes(unlike UDP or IP that divides the bits into datagrams or packets). However, the network layer, that provides service for the TCP, sends packets of information not streams of bytes. Hence, TCP groups a number of bytes together into a *segment* and adds a header to each of these segments and then delivers these segments to the network layer. At the network layer, each of these segments are encapsulated in an IP packet for transmission. The TCP header has information that is required for control purpose which will be discussed along with the segment structure.

Full duplex service –

This means that the communication can take place in both directions at the same time

Connection oriented service –

Unlike UDP, TCP provides connection oriented service. It defines 3 different phases:Connection establishment

Data transfer

Connection termination

Connection oriented service –

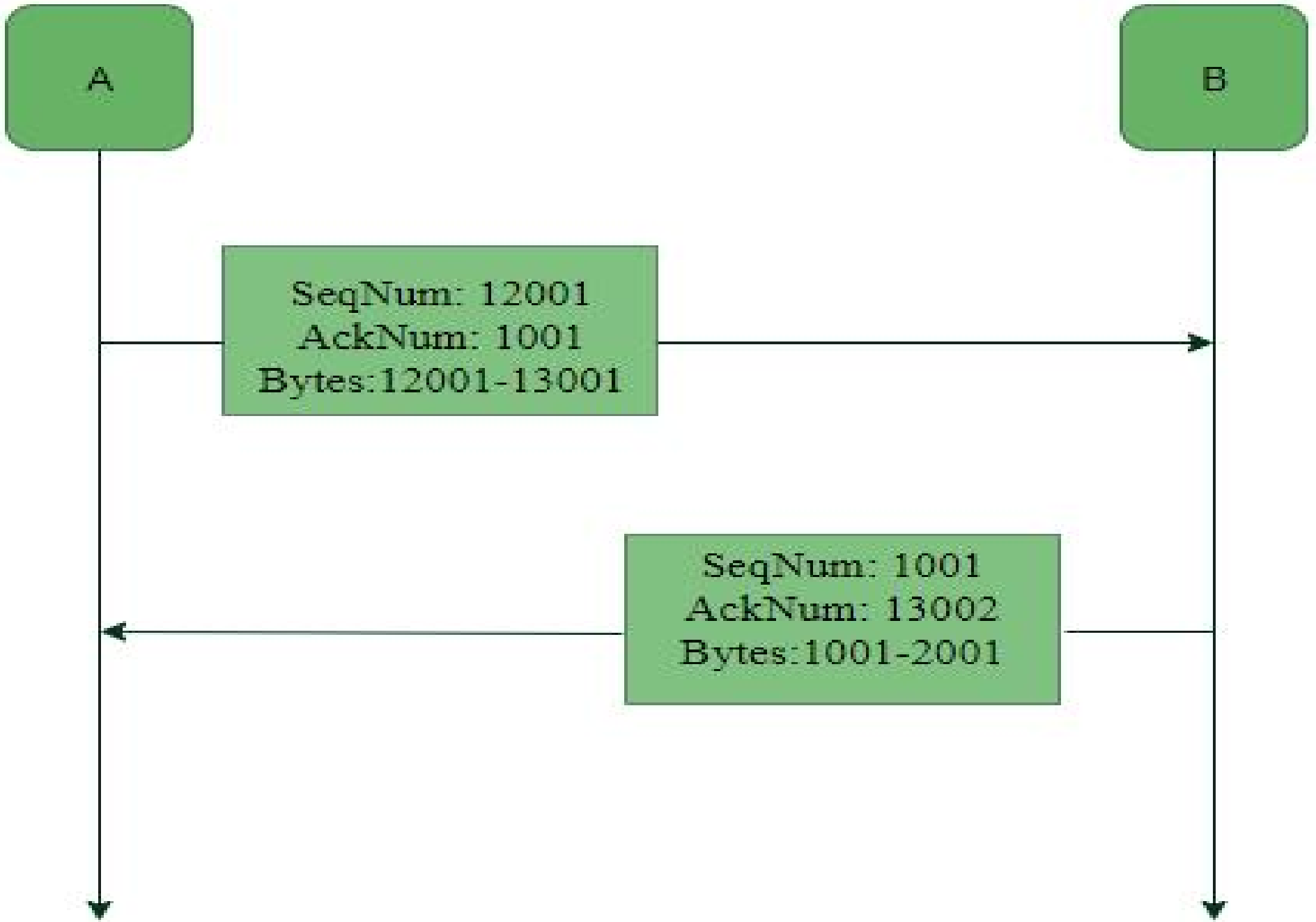
Unlike UDP, TCP provides connection oriented service. It defines 3 different phases:Connection establishment

Data transfer

Connection termination

Byte number, Sequence number and Acknowledgement number:

All the data bytes that are to be transmitted are numbered and the beginning of this numbering is arbitrary. Sequence numbers are given to the segments so as to reassemble the bytes at the receiver end even if they arrive in a different order. Sequence number of a segment is the byte number of the first byte that is being sent. Acknowledgement number is required since TCP provides full duplex service. Acknowledgement number is the next byte number that the receiver expects to receive which also provides acknowledgement for receiving the previous bytes.


A
B
SeqNum: 12001
AckNum: 1001
Bytes:12001-13001
SeqNum: 1001
AckNum: 13002
Bytes:1001-2001

n this example we see that, A sends acknowledgement number1001, which means that it has received data bytes till byte number 1000 and expects to receive 1001 next, hence B next sends data bytes starting from 1001. Similarly, since B has received data bytes till byte number 13001 after the first data transfer from A to B, therefore B sends acknowledgement number 13002, the byte number that it expects to receive from A next.

TCP is a connection oriented protocol and every connection oriented protocol needs to establish connection in order to reserve resources at both the communicating ends.

1. Connection Establishment –

Sender starts the process with following:

Sequence number (Seq=521): contains the random initial sequence number which generated at sender side.

Syn flag (Syn=1): request receiver to synchronize its sequence number with the above provided sequence number.

Maximum segment size (MSS=1460 B): sender tells its maximum segment size, so that receiver sends datagram which won't require any fragmentation. MSS field is present inside **Option** field in TCP header.

Window size (window=14600 B): sender tells about his buffer capacity in which he has to store messages from

2. TCP is a full duplex protocol so both sender and receiver require a window for receiving messages from one another.

Sequence number (Seq=2000): contains the random initial sequence number which generated at receiver side.

Syn flag (Syn=1): request sender to synchronize its sequence number with the above provided sequence number.

Maximum segment size (MSS=500 B): sender tells its maximum segment size, so that receiver sends datagram which won't require any fragmentation. MSS field is present inside **Option** field in TCP header. Since $MSS_{receiver} < MSS_{sender}$, both parties agree for minimum MSS i.e., 500 B to avoid fragmentation of packets at both ends

Window size (window=10000 B): receiver tells about his buffer capacity in which he has to store messages from sender.

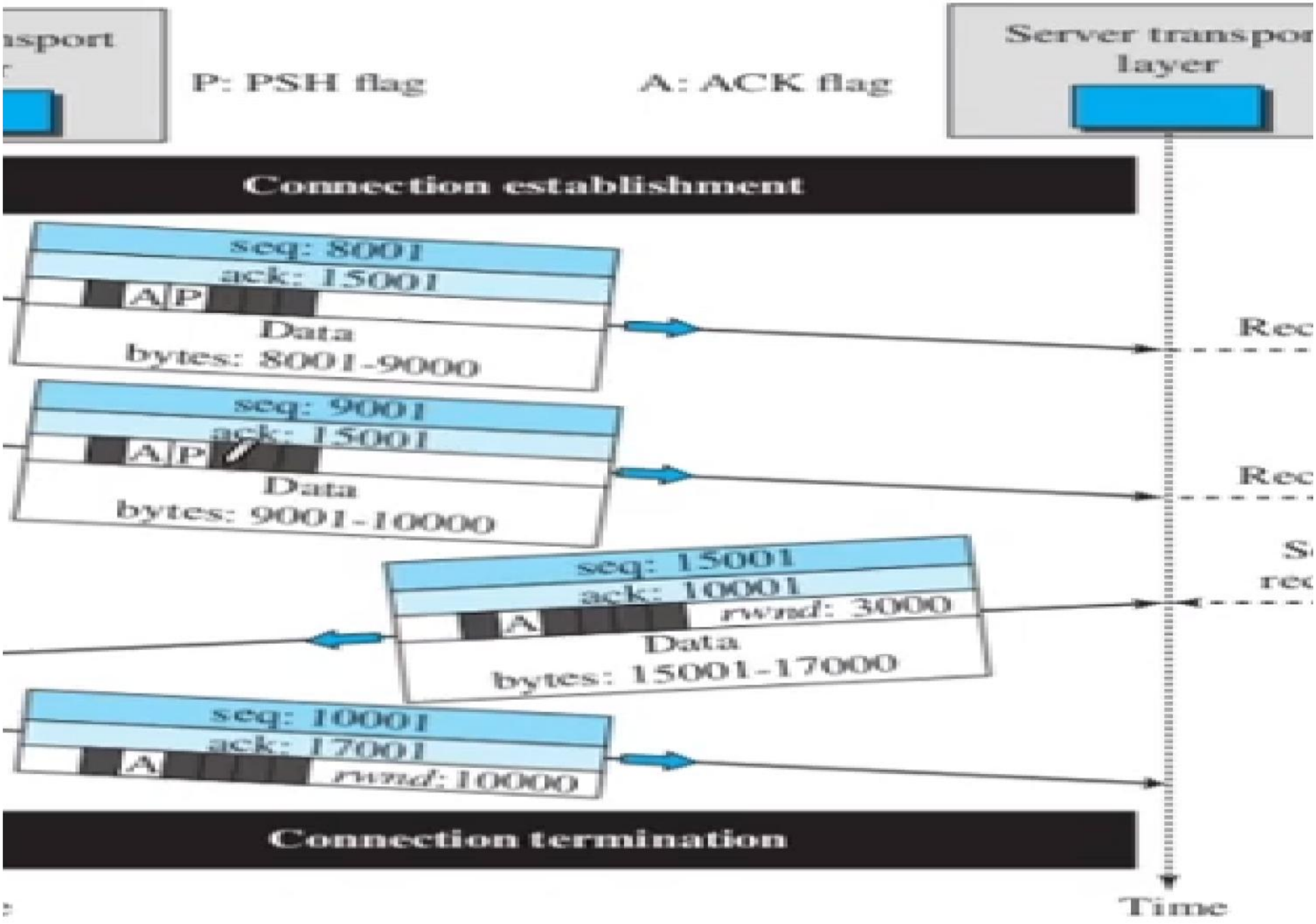
Acknowledgement Number (Ack no.=522): Since sequence number 521 is received by receiver so, it makes a request of next sequence number with Ack no.=522 which is the next packet expected by receiver since Syn flag consumes 1 sequence no.

ACK flag (ACk=1): tells that acknowledgement number field contains

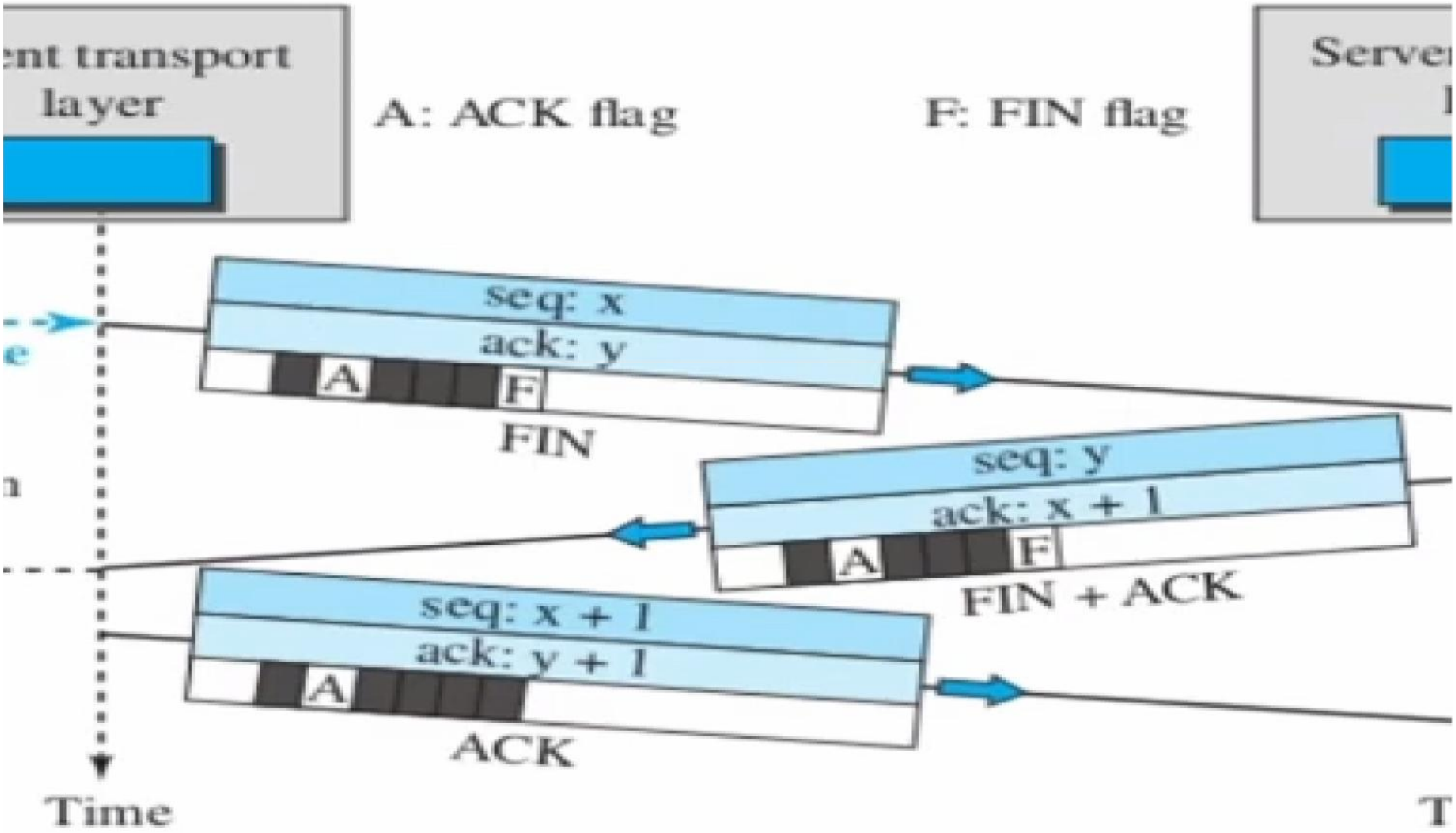
3. Sender makes the final reply for connection establishment in following way:**Sequence number (Seq=522):** since sequence number = 521 in 1st step and SYN flag consumes one sequence number hence, next sequence number will be 522.

Acknowledgement Number (Ack no.=2001): since sender is acknowledging SYN=1 packet from the receiver with sequence number 2000 so, the next sequence number expected is 2001.

ACK flag (ACK=1): tells that acknowledgement number field contains the next sequence expected by sender.


sport
P: PSH flag
A: ACK flag
Server transpor
layer
Connection establishment
seq: 8001
ack: 15001
A P
Data
bytes: 8001-9000
seq: 9001
ack: 15001
A P
Data
bytes: 9001-10000
seq: 15001
ack: 10001
rwnd: 3000
A
Data
bytes: 15001-17000
seq: 10001
ack: 17001
A
rwnd: 10000
Connection termination
Time

ination


ent transport layer
A: ACK flag
F: FIN flag
Server
seq: x
ack: y
A
F
FIN
seq: y
ack: x + 1
A
F
FIN + ACK
seq: x + 1
ack: y + 1
A
ACK
Time

Features

UDP is used when acknowledgement of data does not hold any significance.

UDP is good protocol for data flowing in one direction.

UDP is simple and suitable for query based communications.

UDP is not connection oriented.

UDP does not provide congestion control mechanism.

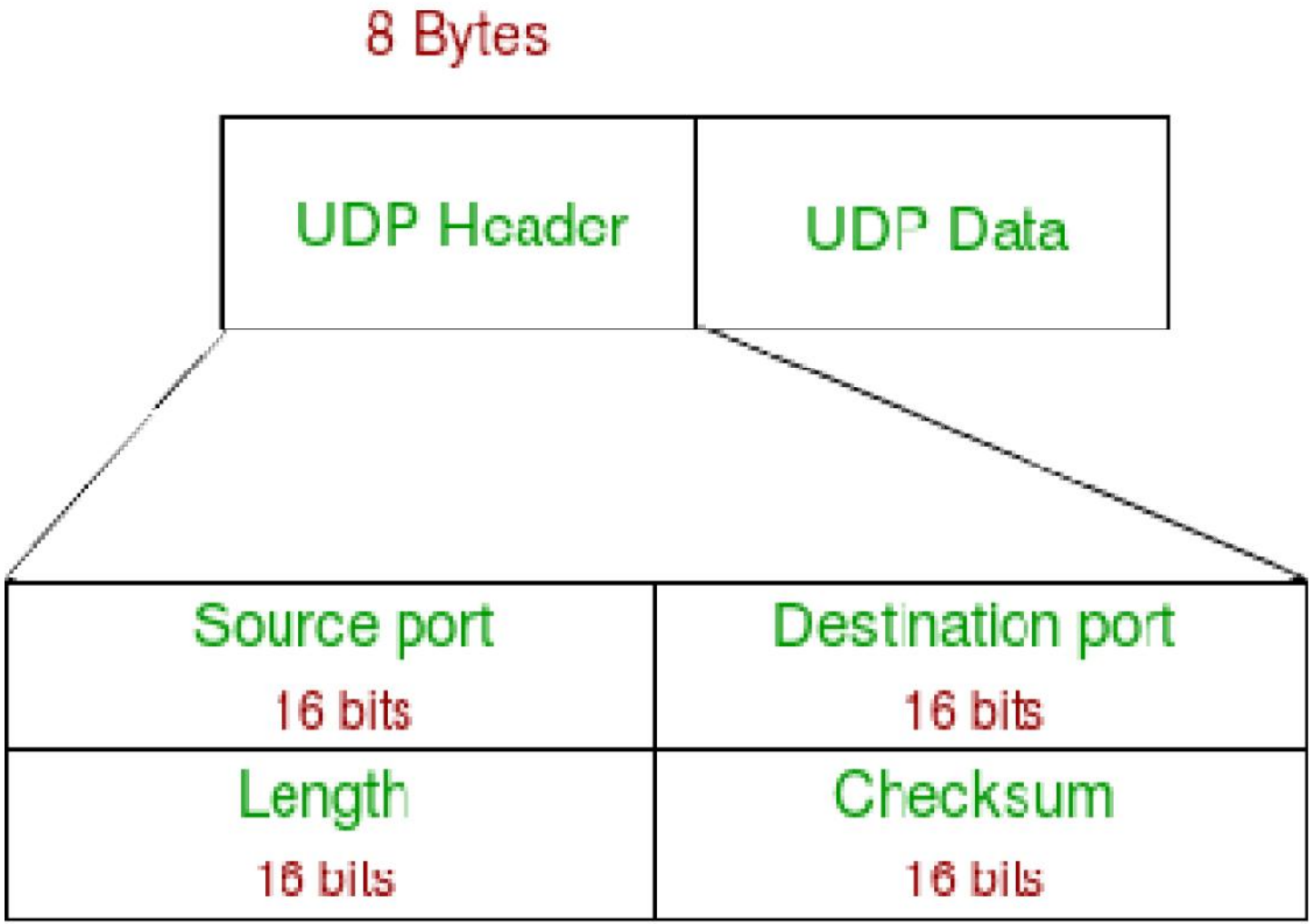
UDP does not guarantee ordered delivery of data.

UDP is used for RIP protocols.

UDP is suitable protocol for streaming applications such as VoIP, multimedia streaming.

UDP is used for multicasting

UDP is suitable for process with internal flow and error control mechanisms. TFTP process includes flow and


8 Bytes
UDP Header
UDP Data
Source port
16 bits
Destination port
16 bits
Length
16 bits
Checksum
16 bits

UDP Header –

UDP header is **8-bytes** fixed and simple header, while for TCP it may vary from 20 bytes to 60 bytes. First 8 Bytes contains all necessary header information and remaining part consist of data. UDP port number fields are each 16 bits long, therefore range for port numbers defined from 0 to 65535; port number 0 is reserved. Port numbers help to distinguish different user requests or process.

Source Port : Source Port is 2 Byte long field used to identify port number of source.

Destination Port : It is 2 Byte long field, used to identify the port of destined packet.

Length : Length is the length of UDP including header and the data. It is 16-bits field.

Checksum : Checksum is 2 Bytes long field. It is the 16-bit one's complement of the one's complement sum of the UDP header, pseudo header of information from the IP header and the data, padded with zero octets at the end (if necessary) to make a multiple of two octets.

- Used for simple request response communication when size of data is less and hence there is lesser concern about flow and error control.

- It is suitable protocol for multicasting as UDP supports packet switching.

- UDP is used for some routing update protocols like RIP(Routing Information Protocol).

- Normally used for real time applications which can not tolerate uneven delays between sections of a received message.

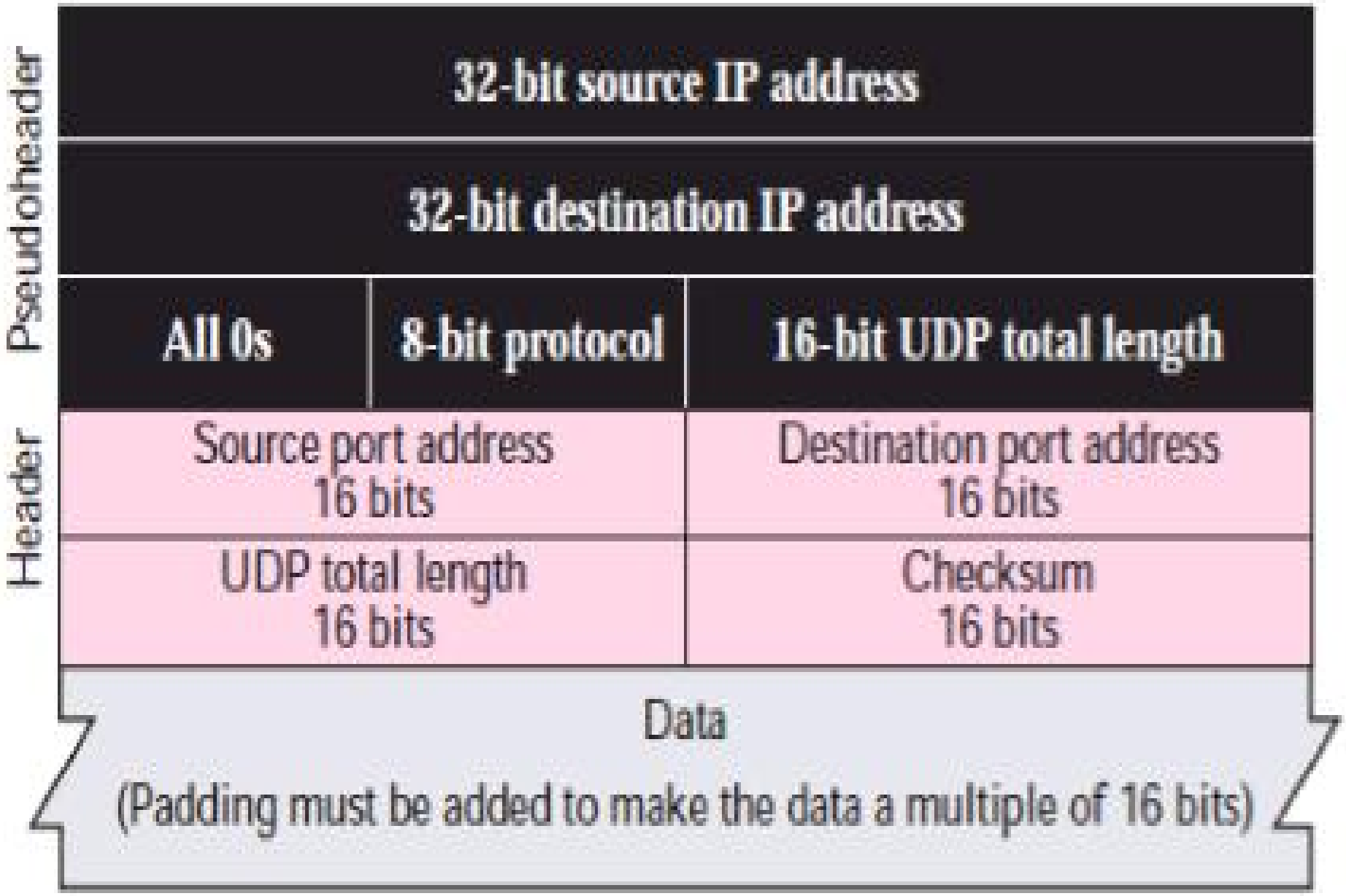


Fig1: Pseudo header for checksum calculation

demo header that basically helps in calculating the CheckSum of TCP UDP Packets.

From the TCP or UDP point of view, the TCP packet does not contain IP addresses.

Thus, to do a proper checksum, a "pseudo-header" is included. It's "pseudo", because it is not actually part of the TCP/UDP datagram.

It contains the most important parts of the IP header, that is, source and destination address, protocol number and data length.

Checksum :

Here the checksum includes three sections: a pseudo header, the UDP header, and the data coming from the application layer.

UDP services

Process to process communication
Connectionless service
Connection oriented service
Flow control
Error control
Congestion control
Encapsulation and deencapsulation
Queuing
Multiplexing and demultiplexing