Spring 2024 Computer Networks (CS 3001)
Assignment No. 2 "Transport Layer Services" SOLUTION

Maximum mark: 50

Question 1: **[4 * 2.5 = 10 Marks]**

i. The transport layer provides logical rather than physical communication between application processes. Explain and illustrate how?

Solution:

The transport layer of the OSI (Open Systems Interconnection) model provides logical communication between application processes by abstracting away the underlying physical network details and providing a set of services that enable reliable and efficient data transfer between these processes. Several crucial processes enable this logical communication:

Addressing and Multiplexing:

Every application process that is operating on a host is given a unique address (port number) by the transport layer. The transport layer uses port numbers to identify various processes and route data packets to the appropriate location. A host's many application processes can share a single network connection thanks to multiplexing. Because every process has a distinct port number, the transport layer can distinguish between incoming data packets meant for various processes.

Segmentation and Reassembly:

Data from the application layer is separated into more manageable, smaller pieces called segments by the transport layer. Large data volumes are divided into smaller segments through segmentation so that they can be transferred over the network more effectively.

Before sending the data stream to the destination application process, the transport layer at the receiving end reassembles these segments into the original data stream. Reassembly guarantees that the application layer receives the data in the correct order.

Error Detection and Correction:

The transport layer offers mistake detection and correction algorithms to guarantee the dependability of data transfer. This covers acknowledgment messages, checksums, and the retransmission of faulty or lost data packets. Applications are protected from the complexity of network faults and can rely on the integrity of the network by identifying and fixing mistakes at the transport layer.

Flow Control and Congestion Control:

Flow control mechanisms govern the rate of data transmission across sender and receiver in order to avoid the receiver from becoming overwhelmed by data that arrives. By modifying the transmission rate in response to the receiver's capacity, the transport layer controls flow control.

In order to prevent network congestion, congestion control techniques dynamically modify the transmission rate in response to various network factors, including packet loss and delay. By doing this, network resources are used effectively and service quality deterioration is avoided.

Illustration:

Think of A and B, two application processes that are operating on separate hosts that are linked to a network. The transport layer is used by these activities to communicate with one another.

- Data is sent from Process A to Process B via the transport layer, where it is divided up and given source and destination addressing information (source and destination port numbers), and encapsulates it into segments.
- These segments are then transmitted over the network to the destination host.
- At the receiving end, the transport layer on host B receives the segments, performs error checking, and reassembles the segments into the original data stream.
- Finally, the transport layer delivers the data to Process B, which can then process the information without needing to be concerned with the underlying network details.

ii. Provide examples of applications where UDP is preferred over TCP.

Solution:

UDP (User Datagram Protocol) has been chosen over TCP (Transmission Control Protocol) in applications where speed, low latency, and clarity have precedence over reliability and correction of errors. Common uses of these kinds of applications are as follows:

1. Real-Time Multimedia Streaming (e.g., YouTube Live, Twitch)
2. Online Gaming
3. Voice over IP (VoIP) communication (e.g., Skype, Discord)
4. DNS (Domain Name System) Resolution
5. IoT (Internet of Things) Devices and Sensor Networks
6. Network Management and Monitoring (e.g., SNMP, syslog)
7. Broadcast and Multicast Applications (e.g., video conferencing, live broadcasting, IPTV)

iii. Define the role of the **rdt_rcv()** function in the RDT protocol. How does it differ from the **rdt_send()** function?

Solution:

The rdt_rcv() function in a reliable data transfer (RDT) protocol is responsible for receiving and processing data packets sent by the sender. It operates on the receiver side of the communication channel and ensures that data is correctly received, ordered, and delivered to the application layer. Here's how the rdt_rcv() function typically operates:

Packet Reception:

The rdt_rcv() function receives data packets from the sender over the network.

Error Detection:

It checks the received packets for errors or corruption using techniques such as checksum verification.

In-order Delivery:

rdt_rcv() ensures that data packets are delivered to the application layer in the correct order, even if they arrive out of order over the network.

Duplicate Packet Handling:

It discards duplicate packets received from the sender to prevent duplicate data delivery to the application layer.

Acknowledgment Generation:

The rdt_rcv() function generates acknowledgments (ACKs) for successfully received packets and sends them back to the sender to acknowledge receipt.

Data Extraction:

Once the received packets are error-free and in-order, rdt_rcv() extracts the payload or application data from the packets and delivers it to the application layer for further processing.

Buffering:

rdt_rcv() may buffer out-of-order packets until all preceding packets are received, ensuring in-order delivery to the application layer.

Difference:

The rdt_rcv() function operates on the receiver side of the communication channel in a reliable data transfer (RDT) protocol, responsible for receiving, processing, and delivering data packets to the application layer. It handles duties like data extraction, acknowledgment creation, duplicate packet handling, in-order delivery, and error detection. On the other handm, the rdt_send() function functions on the sender side and is in charge of transmitting data packets to the recipient. It also handles timeout management, packet transmission, sequence number assignment, error detection, and packet resend in the event that a packet is lost. Rdt_send() focuses on reliably delivering data from the sender to the recipient, whereas rdt_rcv() guarantees that data is correctly received, sorted, and provided to the application layer.

iv. Describe the purpose of the **udt_send()** function in the RDT protocol.

Solution:

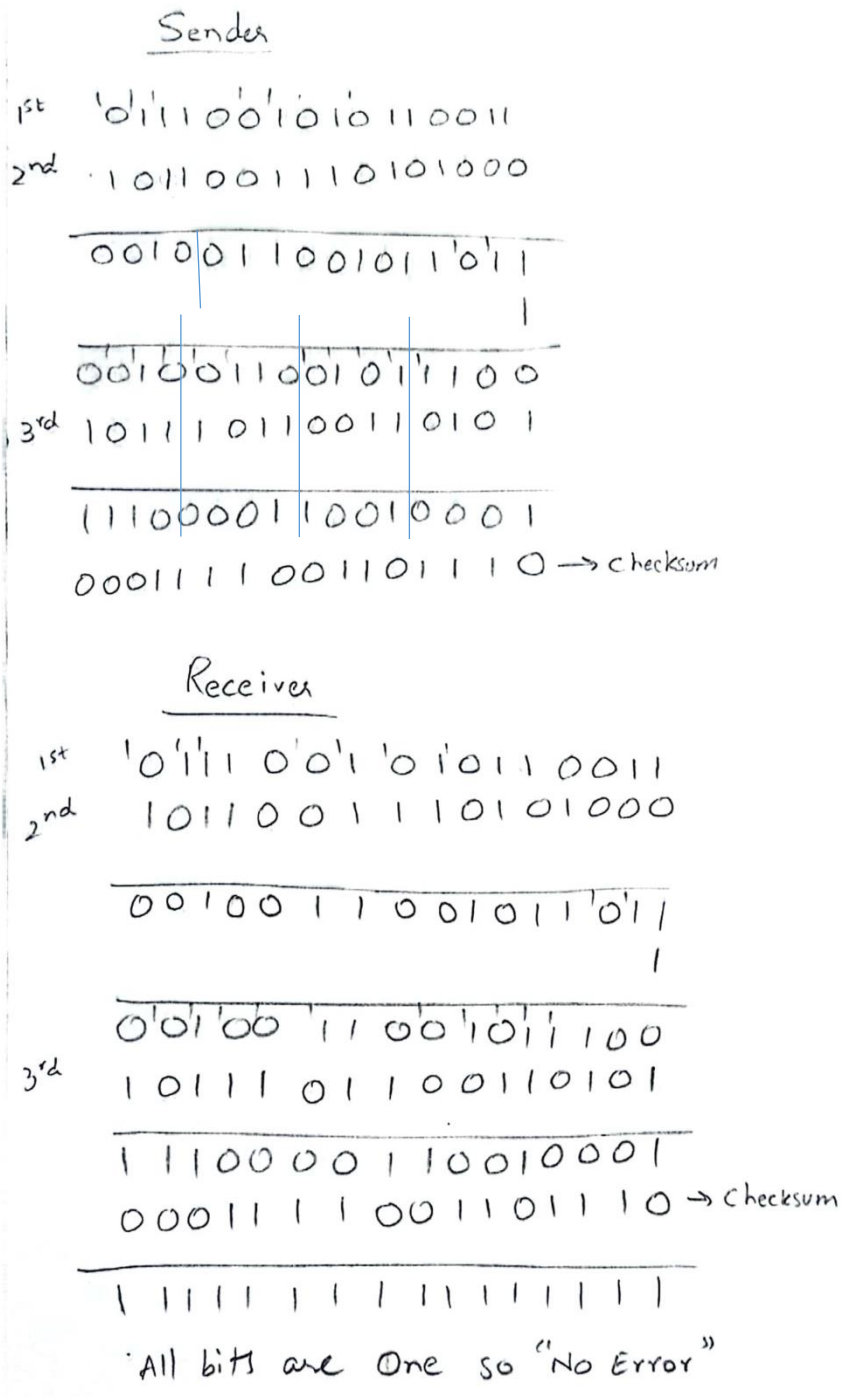
Sending data packets via an underlying network protocol, like TCP (Transmission Control Protocol) or UDP (User Datagram Protocol), is handled by the udt_send() function in a reliable data transfer (RDT) protocol. It transfers the data to be sent to the location indicated by the network address after encapsulating it in network packets. By abstracting the specifics of the underlying network communication from the reliable data transfer protocol, the udt_send() function enables the protocol to concentrate on the orderly and dependable delivery of data between sender and recipient, free from the complexities of network transmission.

Question 2: **[10 Marks]**

Checksum is an error-detecting code used in many Internet standard protocols, including IP, TCP, and UDP. You have to generate Checksum for the following data blocks, which are transmitted using UDP.

0111001010110011	**1011001110101000**	**1011101100110101**

Solution:



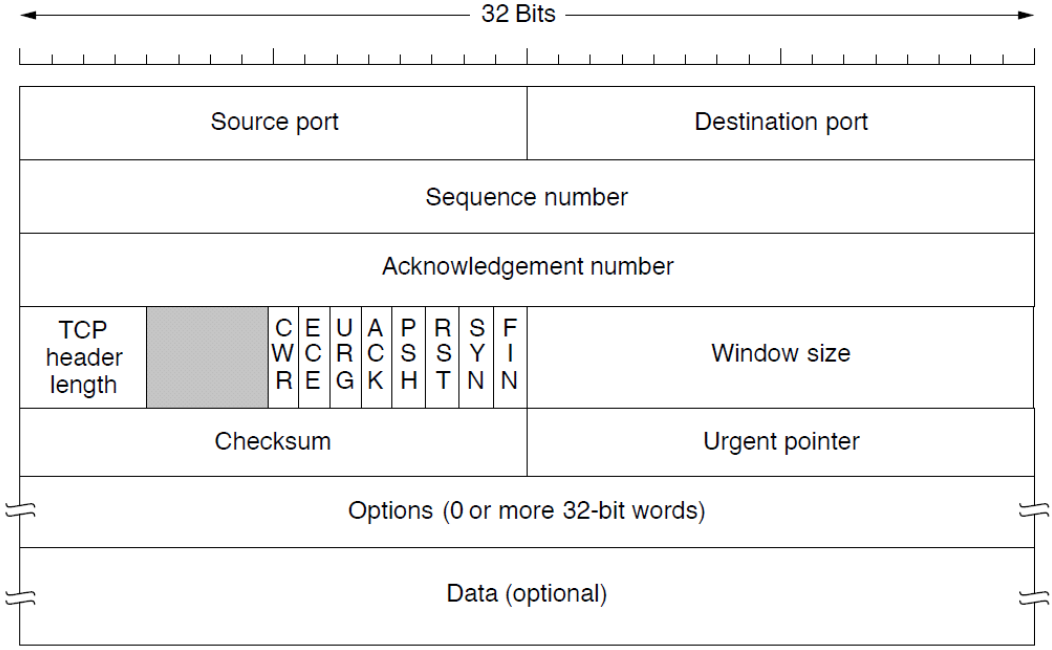
Question 3: **[10 * 2 = 20 Marks]**

Write the values of the following fields of the TCP header, in 'decimal' notation format. The stream, from which the values are to be extracted, is written below in hexadecimal notation.

000**Y** 00**Z**0 0000 0001 0000 0000 3FF0 0000 000**X** FFFF 0100 0000 0010 0000

(1) Source Port number, (2) Destination Port, (3) Sequence Number, (4) Acknowledgement number, (5) TCP header length, (6) ACK bit, (7) FIN bit, (8) SYN bit, (9) Window size, and (10) Checksum.

Note!
(a) The given stream is of total 224 bits.
(b) In the given stream, 'Option' field uses 32-bits to show its value.
(c) There is also some contents of the 'data' field, and is shown by the last 32-bit values in the given stream.
(d) X, Y, and Z in the given stream are the parts of student ID number. The most significant digit of the ID will be assigned to variable X, second-most-significant to Y and Z for third-most-significant (i.e. the 2nd last) digit in ID. That is, if the student ID is "20k-1165", then X=1, Y=1 and Z=6. Every student must have to calculate values of X,Y and Z, as per their IDs to attempt this question.



Solution:
By using same IDs as given above, X=1, Y=1 and Z=6.
So, the byte stream after replacing the variable values are:
000**1** 00**6**0 0000 0001 0000 0000 3FF0 0000 000**1** FFFF 0100 0000 0010 0000

In binary, with 32 bit on each line:
0000 0000 0000 0001 0000 0000 0110 0000
0000 0000 0000 0000 0000 0000 0000 0001
0000 0000 0000 0000 0000 0000 0000 0000
0011 1111 1111 0000 0000 0000 0000 0000
0000 0000 0000 0001 1111 1111 1111 1111
0000 0001 0000 0000 0000 0000 0000 0000
0000 0000 0001 0000 0000 0000 0000 0000

The value of the required fields are given below:
(1) Source Port number: 1
(2) Destination Port: 96
(3) Sequence Number: 1
(4) Acknowledgement number: 0
(5) TCP header length: 3
(6) ACK bit: 1
(7) FIN bit: 0
(8) SYN bit = 0
(9) Window Size = 0
(10) Checksum: 1

Question 4: **[5 * 2 = 10 Marks]**

A computer with IP Address: 202.28.33.21 sends a request to another computer. The request contains following UDP header:

0019D36A001C001C

Categorize content of above UDP header by answering the following questions

a) What is the source port number?
b) What is the socket address of the sender end?
c) What is the total length of the user datagram?
d) What is the length of data
e) Is the packet directed from a client to a server or vice versa?

Q4) Solution

0019 D36A 001C 001C
SP# DP# Length Checksum

(a) Source Port no = 0019
$= 0\times16^3+0\times16^2+1\times16^1+9\times16^0$
$= 16+9$
$= 25$

(b) Socket Address = 202.28.33.21 | 25

(c) Total Length = 001C
$= 0\times16^3+0\times16^2+1\times16^1+12\times16^0$
$= 0+0+16+12$
$= 28$ bytes

(d) Length of Data = 28 − 8 = 20 bytes (Total length − Header)

(e) Since Source Port no is 25 (well known port)
The packet is directed from Server to Client