User Datagram Protocol UDP

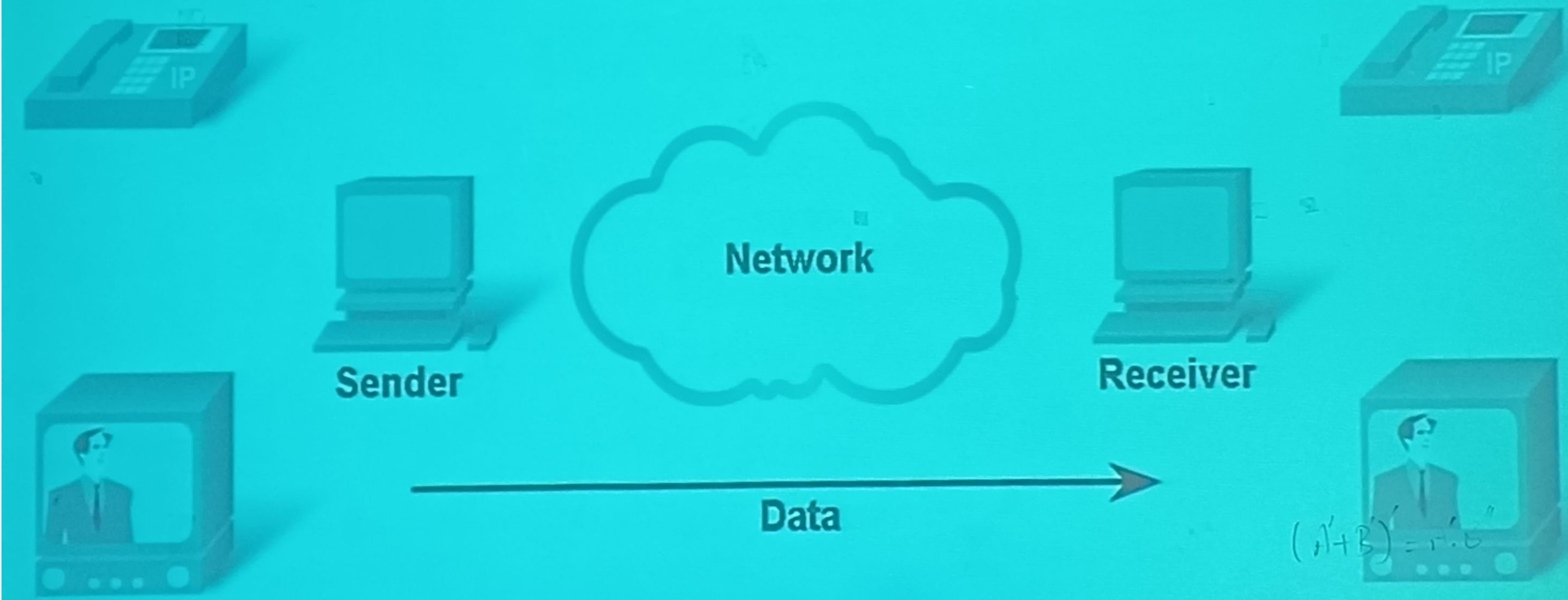
The User Datagram Protocol (UDP) is simplest Transport Layer communication protocol available of the TCP/IP protocol suite. It involves minimum amount of communication mechanism. UDP is said to be an unreliable transport protocol but it uses IP services which provides best effort delivery mechanism.

In UDP, the receiver does not generate an acknowledgement of packet received and in turn, the sender does not wait for any acknowledgement of packet sent. This shortcoming makes this protocol unreliable as well as easier on processing.

Requirement of UDP

A question may arise, why do we need an unreliable protocol to transport the data? We deploy UDP where the acknowledgement packets share significant amount of bandwidth along with the actual data. For example, in case of video streaming, thousands of packets are forwarded towards its users. Acknowledging all the packets is trouble some and may contain huge amount of bandwidth wastage. The best delivery mechanism of underlying IP protocol ensures best efforts to deliver its packets, but even if some packets in video streaming get lost, the impact is not calamitous and can be ignored easily. Loss of few packets in video and voice traffic sometimes goes unnoticed.

UDP Low Overhead Data Transport

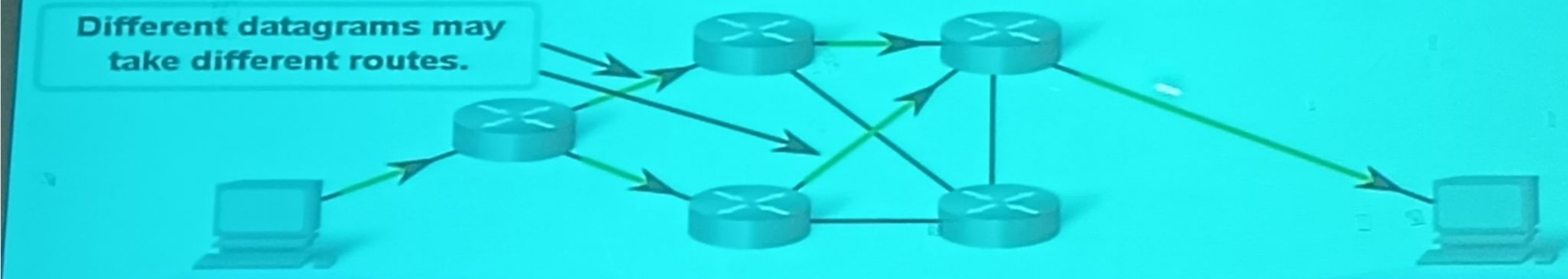


$(A'+B')' = A''\cdot B''$

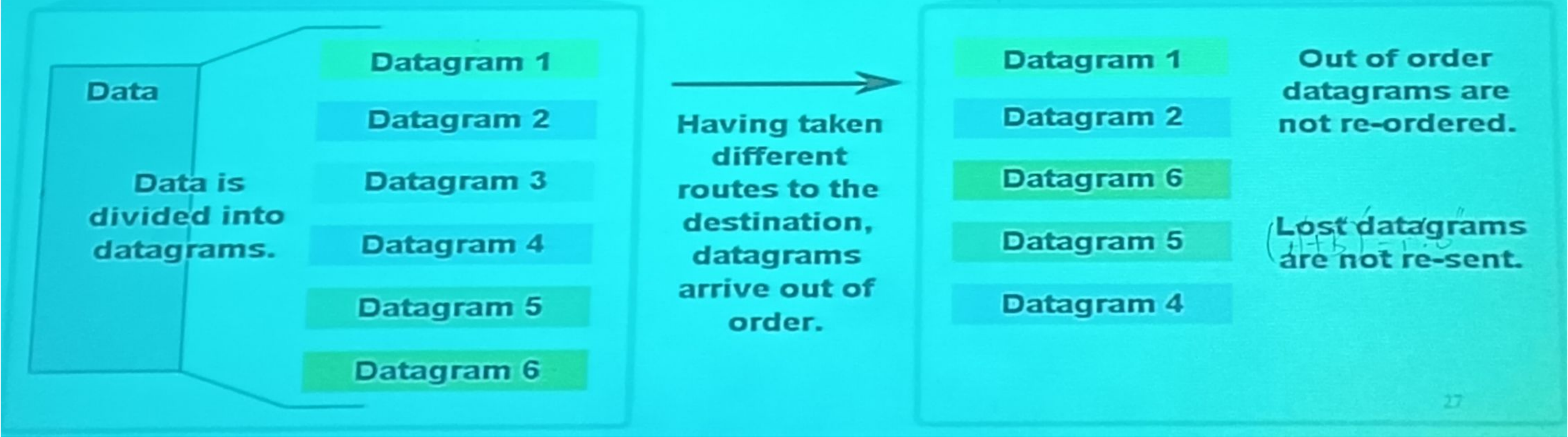
UDP does not establish a connection before sending data.

UDP Data Transfer

UDP: Connectionless and Unreliable



Different datagrams may take different routes.



Data

Data is divided into datagrams.

Datagram 1

Datagram 2

Datagram 3

Datagram 4

Datagram 5

Datagram 6

Having taken different routes to the destination, datagrams arrive out of order.

Datagram 1

Datagram 2

Datagram 6

Datagram 5

Datagram 4

Out of order datagrams are not re-ordered.

Lost datagrams are not re-sent.

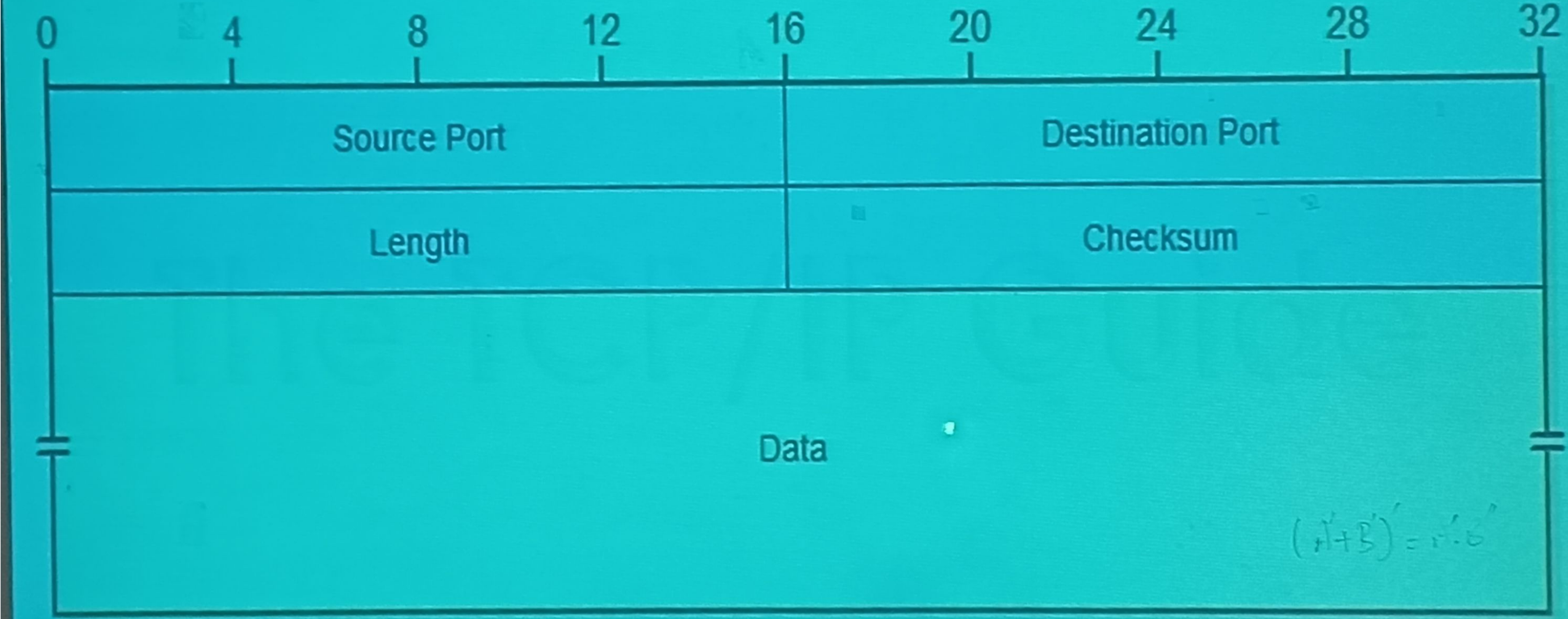
Features of UDP

- 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 stateless (it doesn't acknowledge that the packets being sent have been received..)
- UDP is suitable protocol for streaming applications such as VoIP, multimedia streaming.

$(A+B) = r.b$

UDP Header

UDP header is as simple as its function.



0 4 8 12 16 20 24 28 32

Source Port

Destination Port

Length

Checksum

Data

$(A'+B')' = A''\cdot B''$

UDP header contains four main parameters:

- **Source Port:** This 16 bits information is used to identify the source port of the packet.
- **Destination Port:** This 16 bits information is used identify application level service on destination machine.
- **Length:** Length field specifies the entire length of UDP packet (including header). It is 16-bits field and minimum value is 8-byte, i.e. the size of UDP header itself.
- **Checksum:** This field stores the checksum value generated by the sender before sending. IPv4 has this field as optional so when checksum field does not contain any value, it is made 0 and all its bits are set to zero.

UDP application

Here are few applications where UDP is used to transmit data:

- Domain Name Services
- Simple Network Management Protocol
- Trivial File Transfer Protocol
- Routing Information Protocol
- Kerberos

UDP checksum example:

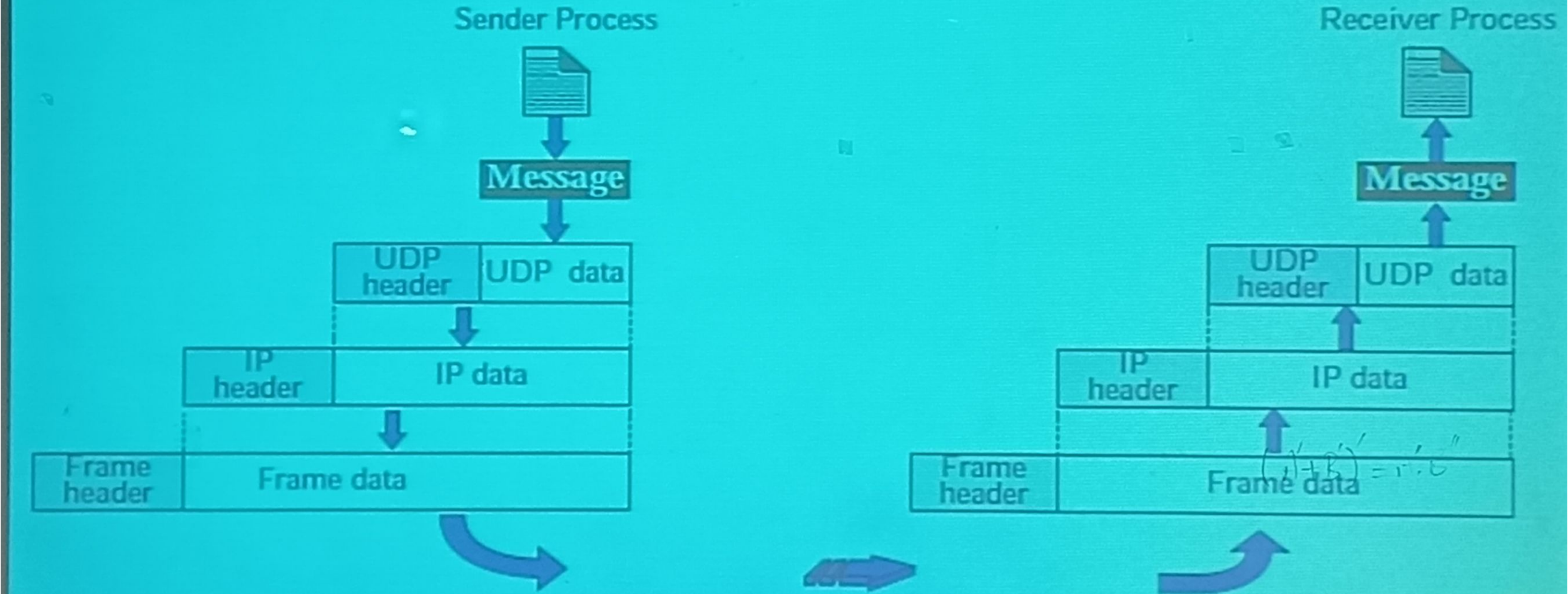
- Three packets of 16 bits each
  - 0110011001100110
  - 0101010101010101
  - 0000111100001111
- adding the three, calling it 'r':
  - 1100101011001010
- Send the four packets, the original three and 1's complement of 'r' to destination
- The 1's complement of 'r' is:
  - 0011010100110101
- at destination, the sum of four packets should be:
  - 1111111111111111
- **If the packet is damaged:**
  - 1111**0**11111111111 **(zeros!!)**

$(A'+B')' = A''.B''$

Why provide for error checking? ***No guarantee that it is provided in all of the links between source and destination***

UDP Encapsulation and Decapsulation

When a process has a message to send through UDP, it passes the message to UDP along with a pair of socket addresses and the length of data.



a. Encapsulation

b. Decapsulation

Classification of Application Layer Protocols

Protocols	TCP	UDP	Function
FTP (20/21)	√		Transfer data between client & Server
Telnet (23)	√		Remote Control
HTTP (80)	√		Transfer of web pages
HTTPS (443)	√		Secure Transfer of web pages
SMTP (25)	√		Transfer E-Mail to Server
POP (110)	√		Receive E-Mail from Server
TFTP (69)		√	Transfer data without check delivery
DHCP (67/68)		√	Automatic assigning of IP to PCs
SMB		√	File sharing in MS operating systems
DNS (53)		√	Translating hostname to IP address

$(A'+B)=1.0$

TCP (connection) & UDP (connectionless) are protocols of Transport Layer

Comparison between TCP and UDP

TCP	UDP
Transmission Control Protocol	User Datagram Protocol
Connection Oriented	Connection Less
Slow	Fast
Highly Reliable	Unreliable
20 Bytes	8 Bytes
It takes acknowledgement of data and has the ability to retransmit if the user requests.	It neither takes acknowledgement, nor it retransmits the lost data.
TCP is heavy-weight.	UDP is lightweight.

$(A'+B')' = A'\cdot B''$

TCP	UDP
Stream-based	Message-based
Delivery of all data is managed	Not performed
Flow control using sliding window protocol	None
TCP doesn't supports Broadcasting.	UDP supports Broadcasting.
Small to moderate amounts of data	Small to enormous amounts of the data
Applications where reliable transmission of data matters.	Application where data delivery speed matters.
FTP, Telnet, SMTP, IMAP.	DNS, BOOTP, DHCP, TFTP.

$(A'+B')' = A''\cdot B''$