Computer Network(CSC 503)

Shilpa Ingoley

Lecture 34

User Datagram Protocol (UDP)

- The User Datagram Protocol (UDP) is a **connectionless, unreliable transport protocol.**
- It does not add anything to the services of IP except for providing process-to-process communication instead of host-to-host communication.
- It is not usually used for a process such as FTP that needs to send bulk data
- If UDP is so powerless, why would a process want to use it?
- UDP is a very **simple protocol using** and has **minimum overhead**.
- If a process wants to send a small message and does not care much about reliability, it can use UDP.
- Sending a small message using UDP takes much less interaction between the sender and receiver than using TCP.

UDP Applications

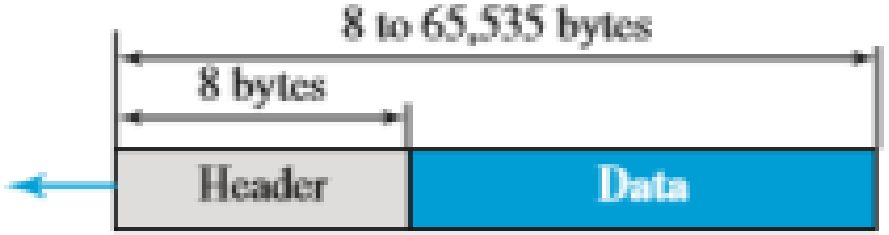
- It is suitable for a process that requires simple request-response communication with **little concern for flow and error control.**
- UDP is suitable for a process with internal flow- and error-control mechanisms. For example, the **Trivial File Transfer Protocol** (TFTP) process includes flow and error control.
- UDP is a suitable transport protocol for **Multicasting**. Multicasting capability is embedded in the UDP software but not in the TCP software.

Contd…

- UDP is used for management processes such as **Simple Network Management Protocol** (SNMP)
- UDP is used for some route updating protocols such **as Routing Information Protocol** (RIP)
- UDP is normally used for **interactive real-time applications** that cannot tolerate uneven delay between sections of a received
- Streaming media applications such as movies.
  - Online multiplayer games.
  - Voice over IP (VoIP)
- UDP can be used for application which has inbuilt mechanism for flow and error control.
  - **Example : TFTP (Trivial File Transfer protocol)**

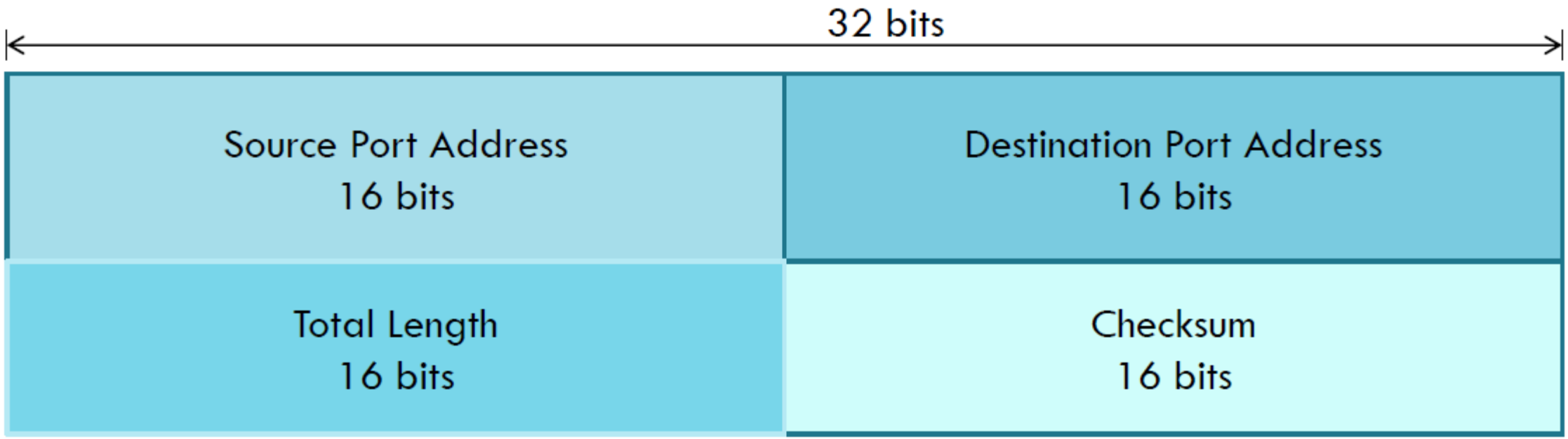
User Datagram Protocol (UDP)

- UDP packets, called **user datagrams**, have a fixed-size header of 8 bytes made of four fields, each of 2 bytes (16 bits).



a. UDP user datagram

UDP Header


32 bits
Source Port Address
16 bits
Destination Port Address
16 bits
Total Length
16 bits
Checksum
16 bits

UDP Header Format

- The first two fields define **the source and destination port numbers.**
- The third field defines the **total length** of the user datagram, header plus data.
  - The 16 bits can define a **total length** of 0 to 65,535 bytes.
  - However, the total length needs to be **less** because a UDP user datagram is stored in an IP datagram with the total length of 65,535 bytes.
- The last field can carry the optional **checksum**

Contd…

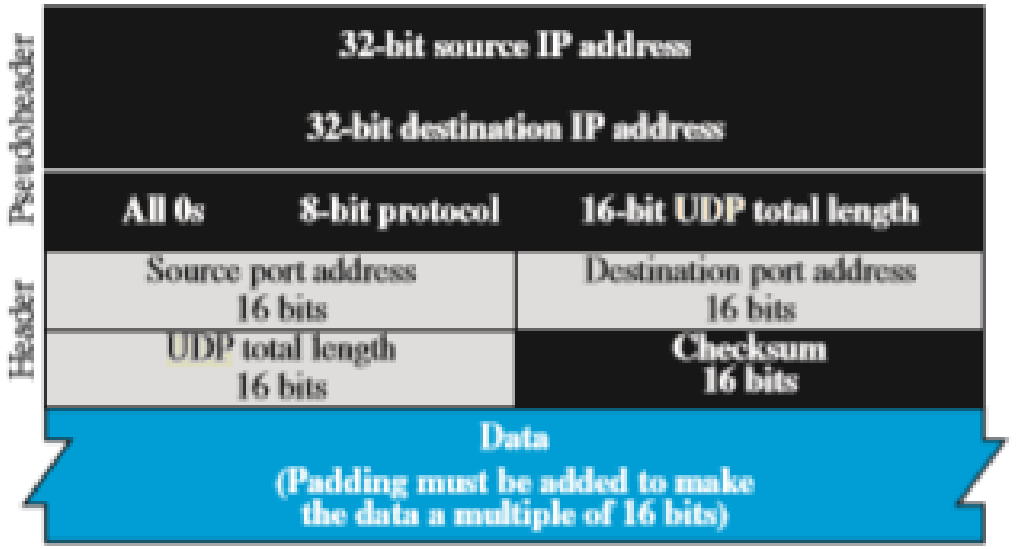
Checksum :

UDP checksum calculation includes **three** sections:

- **A pseudoheader:**
- **The UDP header,**
- **The data coming from the application layer.**

Contd…

- **A pseudoheader**: The pseudoheader is the part of the header of the IP packet in which the user datagram is to be encapsulated with some fields filled with 0s



UDP and TCP Port Numbers

Port	*Protocol*	*UDP*	*TCP*	*SCTP*	*Description*
7	Echo	√	√	√	Echoes back a received datagram
9	Discard	√	√	√	Discards any datagram that is received
11	Users	√	√	√	Active users
13	Daytime	√	√	√	Returns the date and the time
17	Quote	√	√	√	Returns a quote of the day
19	Chargen	√	√	√	Returns a string of characters
20	FTP-data		√	√	File Transfer Protocol
21	FTP-21		√	√	File Transfer Protocol
23	TELNET		√	√	Terminal Network
25	SMTP		√	√	Simple Mail Transfer Protocol
53	DNS	√	√	√	Domain Name Service
67	DHCP	√	√	√	Dynamic Host Configuration Protocol
69	TFTP	√	√	√	Trivial File Transfer Protocol
80	HTTP		√	√	HyperText Transfer Protocol
111	RPC	√	√	√	Remote Procedure Call
123	NTP	√	√	√	Network Time Protocol
161	SNMP-server	√			Simple Network Management Protocol
162	SNMP-client	√			Simple Network Management Protocol

Contd…

- **Optional Inclusion of Checksum :**The sender of a UDP packet can choose not to calculate the checksum.
- In this case, the checksum field is filled with **all 0s** before being sent.
- In the situation where the sender decides to calculate the checksum, but it happens that the result is all 0s, the checksum is changed to all 1s before the packet is sent that means the sender complements the sum two times.
- Note that this does not create confusion because the value of the checksum is never all 1s in a normal situation

UDP Services

- **Process to Process Communication:** UDP provides **process-to-process** communication using socket addresses, a combination of IP addresses and port numbers.
- **Connectionless Service**: UDP provides a connectionless service.
  - This means that each user datagram sent by UDP is an **independent datagram**.
  - There is no relationship between the different user datagrams even if they are coming from the same source process and going to the same destination program.
  - The user datagrams are **not numbered**.
  - Also, unlike TCP, there is **no connection establishment and no connection termination.**
  - This means that each user **datagram can travel on a different path**.

Contd…

- **Flow Control:** There is **no flow control**, and hence no window mechanism. The receiver may overflow with incoming messages.
- **Congestion Control:** Since UDP is a connectionless protocol, it **does not provide congestion control**. UDP assumes that the packets sent are small and cannot create congestion in the network.
- **Error Control**: There is **no error control** mechanism in UDP except for the checksum.
  - This means that the **sender does not know if a message has been lost or duplicated.**
  - When the **receiver** detects an error through the checksum, the user datagram is silently **discarded.**

Contd…

- If the checksum does not include the pseudoheader, a user datagram may arrive safe and sound.
- However, if the **IP header is corrupted**, it may be delivered to the **wrong host**.
- The **protocol field** is added to ensure that the packet belongs to **UDP**, and not to TCP.
- A process can use either **UDP or TCP**, the **destination port number** can be the same.
- The value of the protocol field for **UDP is 17**.
- If this value is changed during transmission, the checksum calculation at the receiver will detect it and UDP drops the packet.
- It is not delivered to the wrong protocol

Example

Ex: Below is an hexadecimal dump of an UDP datagram captured.

e2 a7 00 0D 00 20 74 9e 0e ff 00 00 00 01 00 00 00 00 00 00 06 69 73 61 74 61 70 00 00 01 00 01

(i) What is source port number ?

(ii) What is destination port number ?

(iii) What is total length of the user datagram ?

(iv) What is total length of data ?

(v) Is packet directed from client to server or vice versa ?

Soln. :

Following is the UDP datagram structure. The UDP Header is of size 8 bytes and remaining bytes are of data.

- 1. Source port number – $(E2\ A7)_{16}$
- 2. Destination port number - $(000D)_{16}$
- 3. Total length of the user datagram - $(0020)_{16} = (32)_{10}$ bytes
- 4. Total length of data = Total length – header length = 32-8 = 24 bytes
- 5. Packet directed from **client to server**.

TCP Vs UDP

Parameter	TCP	UDP
Acronym	Transmission Control Protocol	User Datagram Protocol
Connection	TCP is a connection-oriented protocol.	UDP is a connectionless protocol.
Usage	TCP is suitable for applications which need high **reliability**, and where the transmission time is relatively less critical.	UDP is suitable for applications which need **fast**, efficient transmission, such as games.
Use by other protocols	HTTP, HTTPs, FTP, SMTP, Telnet	DNS, DHCP, TFTP, SNMP, RIP, VoIP.
Ordering of data packets	Ordering of packets is maintained by TCP.	UDP has no inherent order since all of the packets are independent of each other.
Speed of transfer	The speed for TCP is **slower** compared to UDP.	UDP is **faster** compared to TCP.
Reliability	There is **complete assurance** that the data transferred will be reached at destination	There is **no guarantee** that the data packets sent would reach at all.
Header Size	**20** bytes	**8** bytes.
Weight	TCP is **heavy-weight**.	UDP is **light-weight**.
Data Flow Control	TCP does **Flow Control**.	UDP does **not** have an option for **flow control**
Error Checking	TCP does error checking and error recovery.	UDP does error checking but simply discards erroneous packets. Error recovery is not attempted.
Acknowledgement	Acknowledgement segments	No Acknowledgment

	TCP	UDP
Streaming of data	Data is read as a byte stream, no distinguishing indications are transmitted to signal message (segment) boundaries.	Packets are sent individually and are checked for integrity only if they arrive.
Weight	TCP is heavy-weight. TCP requires three packets to set up a socket connection, before any user data can be sent. TCP handles reliability and congestion control.	UDP is lightweight. There is no ordering of messages, no tracking connections, etc. It is a small transport layer designed on top of IP.
Data Flow Control	TCP does Flow Control. TCP requires three packets to set up a socket connection, before any user data can be sent. TCP handles reliability and congestion control.	UDP does not have an option for flow control

Contd…

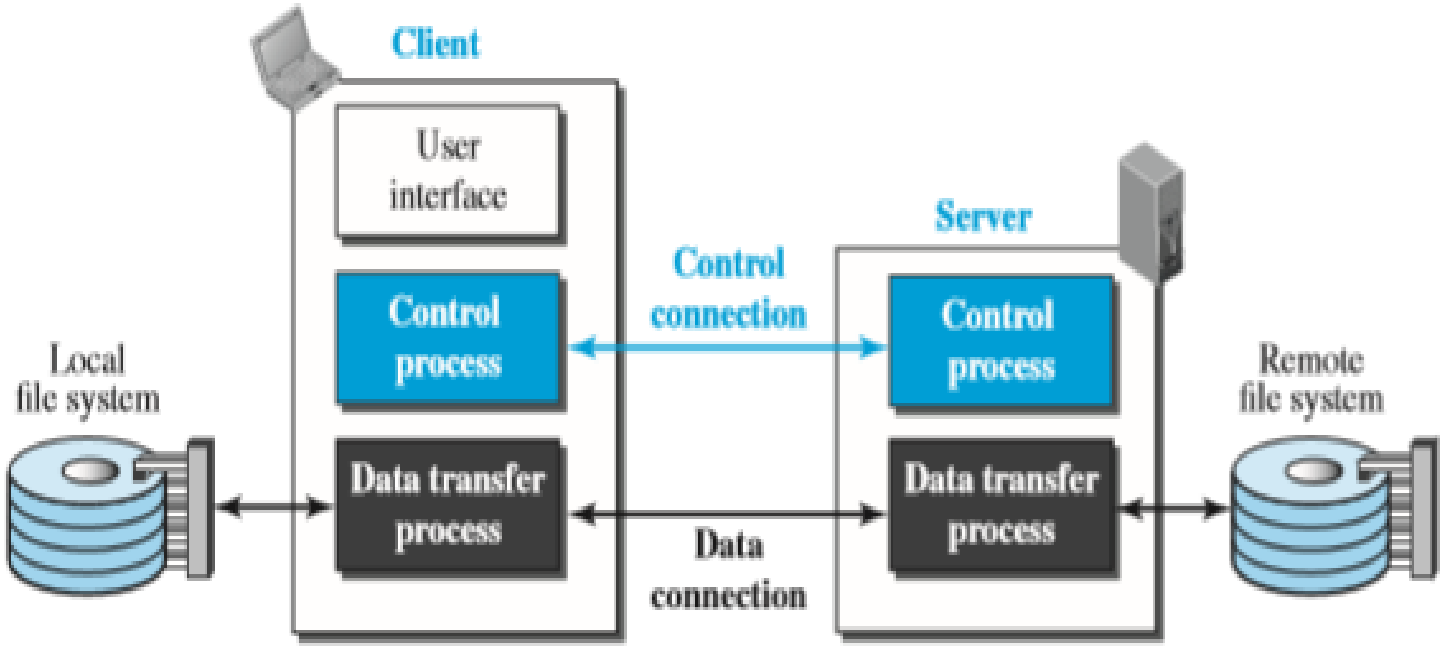
	TCP	UDP
Acknowledgement	Acknowledgement segments	No Acknowledgment
Handshake	SYN, SYN-ACK, ACK	No handshake (connectionless protocol)
Error Checking	TCP does error checking and error recovery. Erroneous packets are retransmitted from the source to the destination.	UDP does error checking but simply discards erroneous packets. Error recovery is not attempted.

Chap 6: File Transfer Protocol (FTP)

- File Transfer Protocol (FTP) →(20,21)
- Secured File Transfer Protocol (SFTP/FTPS) →(22)
- Trivial File transfer Protocol (TFTP) →(69)

File Transfer Protocol (FTP)

- File Transfer Protocol (FTP) is the standard protocol provided by TCP/IP for copying a file from **one host to another**
- Problems: Two systems may use different file name conventions, two systems may have different ways to represent data, two systems may have different directory structures.



Contd…

- The client has **three** components:
  - The user interface
  - The client control process
  - The client data transfer process
- The server has **two** components:
  - The server control process
  - The server data transfer process
- The control connection is made between the control processes. The data connection is made between the data transfer processes.
- The two connections in FTP have different lifetimes.
- The **control connection** remains connected during the **entire interactive FTP session**
- The **data connection** is **opened and then closed** for each file transfer activity that means it opens each time commands that involve transferring files are used, and it closes when the file is transferred.

Contd…

- When a user starts an FTP session, the control connection opens.
- While the control connection is open, **the data connection can be opened and closed multiple times if several files are transferred**.
- FTP uses two well-known TCP ports:
  - Port **21** is used for the **control connection**
  - Port **20** is used for the **data connection**

Contd…

Control connection

- During this control connection, commands are sent from the client to the server and responses are sent from the server to the client.
- Commands, which are sent from the FTP client control process, are in the form of ASCII uppercase, which may or may not be followed by an argument.
- Every FTP command generates at least one response.
- A response has two parts: a **three-digit** number followed by text.
- The numeric part defines the code
- The text part defines needed parameters or further explanations.
- The **first digit** defines the **status of the command**
- The **second digit** defines **the area in which the status applies**
- The **third digit** provides **additional information**

Contd…

- **Data connection**
  - The data connection uses the well-known port 20 at the server site

- Steps:
  - The client issues a passive open using an ephemeral port. This must be done by the client because it is the client that issues the commands for transferring files.
  - Using the PORT command the client sends this port number to the server
  - The server receives the port number and issues an active open using the well known port 20 and the received ephemeral port number

Contd...

- File transfer in FTP means one of **three** things:
  - **Retrieving a file (server to client)**
  - **Storing a file (client to server)**
  - **Directory listing (server to client)**
- The client must define the type of file to be transferred, the structure of the data, and the transmission mode
- Before sending the file through the data connection, we prepare for transmission through the control connection.

Three attributes of communication:

- **File type**
- **Data structure**
- **Transmission mode**

- **File Type:** FTP can transfer one of the following file types across the data connection: ASCII file, EBCDIC file, or image file

Contd...

- **Data structure**:
  - FTP can transfer a file across the data connection using one of the following interpretations of the structure of the data:
    - **File structure**
    - **Record structure**
    - **Page structure**

  - **File structure:** The file structure format (used by default) has **no structure**. It is a continuous stream of bytes.
  - **Record structure:** In the record structure, the **file is divided into records**. This can be used only with **text files**.
  - **Page structure:** In the page structure, the file is divided into **pages**, with each page having a **page number and a page header**. The pages can be stored and accessed randomly or sequentially.

Contd…

- **Transmission Mode**
  - FTP can transfer a file across the data connection using one of the following transmission modes:
    - **Stream mode**
    - **Block mode**
    - **Compressed mode**

  - **Stream Mode:** The stream mode is the default mode; data are delivered from FTP to TCP as a **continuous stream of bytes**.
  - **Block Mode:** In the block mode, data can be delivered from FTP to TCP in blocks. In this case, each block is preceded by a 3-byte header. The first byte is called the **block descriptor**; the next two bytes define the **size** of the block in bytes
  - **Compressed mode:** Data is compressed using a simple algorithm (usually run-length encoding).

FTP Commands

- Following are some important FTP commands :

Commands	Description
?	To request help or information about the FTP commands
Ascii	To set the mode of file transfer to ASCII
Binary	To set the mode of file transfer to binary
Bye	To exit the FTP environment (same as quit)
Close	To terminate a connection with another computer
Delete	To delete (remove) a file in the current remote directory (same as rm in UNIX)
Get	To copy one file from the remote machine to the local machine
Open	To open a connection with another computer
Put	To copy one file from the local machine to the remote machine
Pwd	To find out the pathname of the current directory on the remote machine
Quit	To exit the FTP environment (same as bye)

Secure File Transfer Protocol

- SFTP (**Secure File Transfer Protocol**) is a file transfer protocol that leverages a set of utilities that provide secure access to a remote computer to deliver secure communications.
- Secure File Transfer Protocol (SFTP) works over the **Secure Shell (SSH)** data stream to establish **a secure connection** and provide organizations with a higher level of file transfer protection.
- SFTP only needs a single port number (port 22) to establish a server connection
- SFTP runs over an SSH session, usually on **TCP port 22**

Trivial File Transfer Protocol (TFTP)

- **Trivial File Transfer Protocol** (TFTP) is a simple protocol that provides basic file transfer function with no user authentication.
- TFTP is intended for applications that do not need the sophisticated interactions that File Transfer Protocol (FTP) provides.
- TFTP was designed as a lightweight file transfer mechanism primarily used for **transferring short configuration files to routers and other devices**, typically over a short dedicated link or at least within a LAN environment.
- TFTP **uses UDP as its transport protocol**.
- A transfer request is always initiated targeting port 69

Example:

1) FTP uses how many parallel TCP connections to transfer a file.

- (a) 4
- (b) 3
- (c) 2
- (d) 1

2) If 4 files are transferred from server A to client B in the same session. The number of TCP connections between A and B are ____

(a) 5
(b) 8
(c) 2
(d) 6