Name: Anuran Chakraborty
Roll: 20

UDP

UDP or the User Datagram Protocol is a **connectionless, unreliable** transport protocol. It is only responsible to the process-to-process delivery and performs no flow control and there is only limited error checking. UDP uses the header format shown in fig. 1 for its functioning.

Source Port number (16 bits)	Destination Port number (16 bits)
Total Length (16 bits)	Checksum (16 bits)

Fig 1. User Datagram Format

The **source port number** is the port number used by the process running on the source host.
The **destination port number** is the port number used by the process running on the destination host.
Total length is the total length of the datagram header plus data.
Checksum is used to correct errors over the entire datagram header plus data.
The checksum is optional i.e. it may or may not be used.

Operation

UDP provides a connectionless service that is each datagram is sent by UDP is an independent datagram. There is no relationship between the datagrams. UDP cannot send stream data. The datagrams are not numbered. So, each request must be small enough so that it fits into a single datagram. Only those processes sending short messages should use UDP.

Flow and Error Control

UDP is a simple and unreliable transport protocol. There is no flow control, and error control is limited only to the checksum in the header. As there is no flow control the receiver may overflow with incoming messages. Since there is no error control the sender cannot know if a datagram has been lost or corrupted. The receiver just silently discards a wrong datagram.

To send a message from one process to another UDP encapsulates and decapsulates messages in an IP datagram.

Working

In UDP queues are associated with ports. Fig 2. Shows an overview of the queues and their working.

- At the client side, when a process starts, it requests a port number from the Operating System. Once granted either both an incoming queue and an outgoing queue is created or only an incoming queue is created. Thus, the process has only one outgoing queue and one incoming queue even if it has to communicate with multiple processes. Thus, separate queues are not created for every process it is communicating with. The queues function as long as the process is running after which they are destroyed.
- The client process can then send messages to the outgoing queue by using the source port number. The UDP protocol removes the messages from the queue one by one, adds the UDP header and sends it to IP for further transfer from one host to another. In case the outgoing queue overflows the operating system may ask the client process to wait.

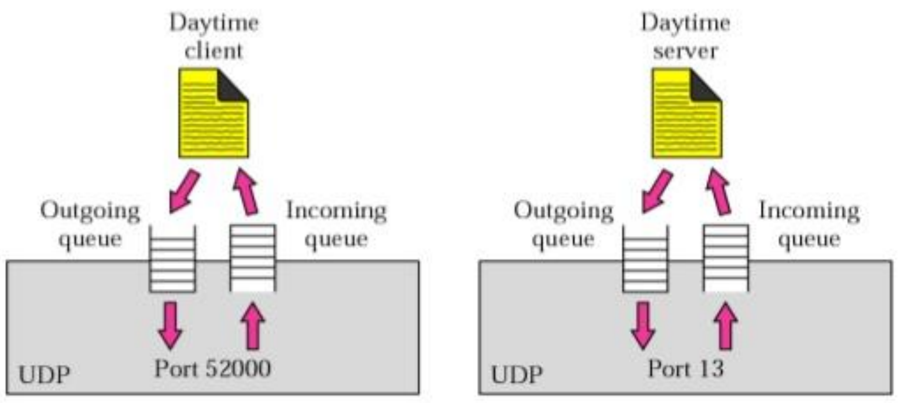

Fig 2. Queues in UDP

- When a message arrives for a client UDP checks to see if there is an incoming queue for the destination port number. If there is a queue then the datagram is pushed to the end of the queue. If there is no queue UDP discards the datagram and sends an ICMP packet to the server with message **Port Unreachable**. There is only one incoming queue for a client process even if it is receiving messages from multiple processes from the server. In case of overflow of incoming queue UDP discards the datagram and sends a **Port Unreachable** message to the server.
- At the server side, the queues are created using its well-known port and the queues remain open as long as the server is running.
- When a message arrives for the server UDP checks to see if there is an incoming queue for the destination port number. If there is a queue then the datagram is pushed to the end of the queue. If there is no such queue UDP discards the datagram and sends an ICMP packet to the client with message **Port Unreachable**. There is only one incoming queue for a server process even if it is receiving messages from multiple processes from the client. In case of overflow of incoming queue UDP discards the datagram and sends a **Port Unreachable** message to the client.
- When a server wants to respond to a client it sends its message to the outgoing queue using the **source port number** specified in the **request from the client.** Then UDP works the same way as it does for the client.

Uses

- UDP can therefore be used only for sending short messages and also for those processes where flow control is either not required or is already implemented internally in the higher-level process. One such example is the TFTP or the Trivial File Transfer Protocol. The TFTP uses UDP as it already includes the error and flow control mechanisms.
- UDP is suitable for **Multicasting**.
- UDP is used in **DNS applications**.
- UDP is used for management processes such as **SNMP**.
- UDP is used by Routing Information Protocol.