User Datagram Protocol (UDP)

- Described in [RFC 768, 1980]
- Simple
- Connectionless (no long term relationship between UDP client and server)
  - Create a socket → send a datagram to a server, then immediately send another datagram on the same socket to a different server
- Unreliable (no guarantees, no order, and can deliver duplicates)
- Each UDP datagram has a length (limited to IP MTU)
- Full duplex → concurrent transfers can take place in both directions

UDP Length: length of datagram in bytes, including header and data, max is 65,535 bytes
Checksum: optional → 16-bit checksum over header and data, or zero
Why is there a UDP?

- no connection establishment (which can add delay)
- simple: no connection state at sender, receiver
- small segment header
- no congestion control: UDP can blast away as fast as desired
- often used for streaming multimedia apps
  - loss tolerant
  - rate sensitive
- other UDP uses → DNS and SNMP
- Suitable for multicasting
- reliable transfer over UDP: add reliability at application layer
  → application-specific error recovery!

UDP versus TCP

1/4

Choice of UDP versus TCP is based on
- Functionality
- Performance

Performance
- TCP’s window-based flow control scheme leads to bursty bulk transfers (not rate based)
- TCP’s “slow start” algorithm can reduce throughput
- TCP has extra overhead per segment
- UDP can send small, inefficient datagrams

2/4

Reliability
- TCP provides reliable, in-order transfers
- UDP provides unreliable service – application must accept or deal with
  - Packet loss due to overflows and errors
  - Out-of-order datagrams

Multicast and broadcast
- Supported only by UDP
- TCP’s error control scheme does not lend itself to reliable multicast

3/4

Application complexity
- Application-level framing (ALF) can be difficult using TCP because of the Nagle algorithm [RFC 896, 1984]
  - ALF implies that data should be organized in units that make the most sense for the application
  
- What makes sense for a video application or an audio application?
  - Nagle algorithm controls when TCP segments are sent to use IP datagrams efficiently
  - But, data may be received and read by applications in different units than how it was sent

4/4

Nagle Algorithm
- Tinygrams (small datagrams) can cause congestion in WANs
  - “Small” means less than the segment size
  - Think what is the datagram size for 1 byte of data?
- A TCP connection can have only one outstanding small segment that has not yet been acknowledged
- No additional small segments can be sent until the acknowledgment is received.
- Small amounts of data are collected by TCP and sent in a single segment when the acknowledgment arrives