Brief summary of TCP, UDP and QUIC

IERG5090

Outline of today’s topic

- Review main functions of transport protocol
- Briefly explain Internet’s transport protocol: TCP and UDP
- How TCP and UDP are used in the Internet
- HTTP, HTTP2, and QUIC

- More details can be found in Ross and Kerose’s PPT, Chapter 3, also posted on course web site.

Main functions for transport protocol

- Multiplexing / de-multiplexing
- Error control
- Flow control
- Congestion control

- Multiplexing / de-multiplexing is the basic function supported by both UDP and TCP
- There are different ways to do error control and flow control
  - Not supported in UDP, other than providing a way to check error
- Congestion control concerns network-wide resource allocation – will be discussed in next lecture
  - Not supported in UDP

Multiplexing / de-multiplexing

- IP delivers packets from host to host
- Transport delivers packet(s) from a process at source node to a process at destination node
- When creating a socket, the process gets a port number
- Servers can create sockets with well-known port:
  - E.g. 80 = web, 443 = SSL
- Transport delivers packet from source (IP, port) to destination (IP, port)

- Socket interface

  https://en.wikipedia.org/wiki/Network_socket

Error control

- ARQ = Automatic Repeat Request

Based on:

  - Error (or loss) detection
  - Receiver feedback
  - Retransmission

- Given these ability, you can always program source and destination to figure out if you need a transmission to correct error (even if sometimes you have duplicates)

- But ARQ protocol working on one packet at a time has poor performance
  - One round trip time per packet in steady state, even with no error/ loss
  - For high bandwidth but large RRT, throughput limited by RRT

Go-back-N (or sliding window)

- ARQ protocol with pipelining, allowing N packets to be in transit at a time
- The window of N slides forward, as ACKs are received
- The windowing mechanism is also used for flow control and congestion control

- A performance issue:
  - When a packet is lost, all subsequent packets are retransmitted

Selective Repeat

Sender:

- Given data from application:

if next available seq # in window, send packet

- Timeout (n)

resend packet n, restart timer

- Receive ACK(n)

If in [sendbase,sendbase+N]:

mark packet n as received

if n smallest unACKed pkt, advance window base to next unACKed seq #

Receiver:

- packet n in [rcvbase, rcvbase+N-1]

  send ACK(n)

  out-of-order: buffer

  in-order: deliver (also deliver buffered, in-order packets), advance window to next not-yet-received packet

- packet n in [rcvbase-N,rcvbase-1]

  ACK(n)

- otherwise:

  ignore

TCP SACK

- Known as Selective Acknowledgement option of TCP
- It is an option, not used often

The use of TCP and UDP

- UDP used for:
  - DNS
  - RIP (routing protocol)
  - SNMP (network management)
  - Multimedia (but not any more)
- Most videos are either streamed over HTTP, or transferred as files
- Why?
  - Firewalls restrict UDP, for (a) avoiding DDOS attacks, and (b) using TCP for congestion control

- TCP used for:
  - File transfer (FTP)
  - E-mail
  - Remote terminal
  - HTTP/WWW (but QUIC is proposed to use UDP)
- What is QUIC - later

HTTP

- Perhaps the most important Internet protocol for applications
- HTTP 1.0 (1999) is a simple protocol, with each request-response creating a separate TCP connection
- HTTP 1.1 allows the TCP connection to be shared by multiple requests
  - Known as persistent connection
  - Furthermore, requests can be pipelined, i.e. multiple outstanding requests

- Each web page typically contains lots of objects (pictures, video etc), sometimes up to 100 objects
- The idea here is similar to the pipelining in Go-Back-N
- HTTP 1.1 is more efficient than opening multiple TCP connections in parallel
- But the results still come back in sequence, can suffer head-of-line blocking to some extent

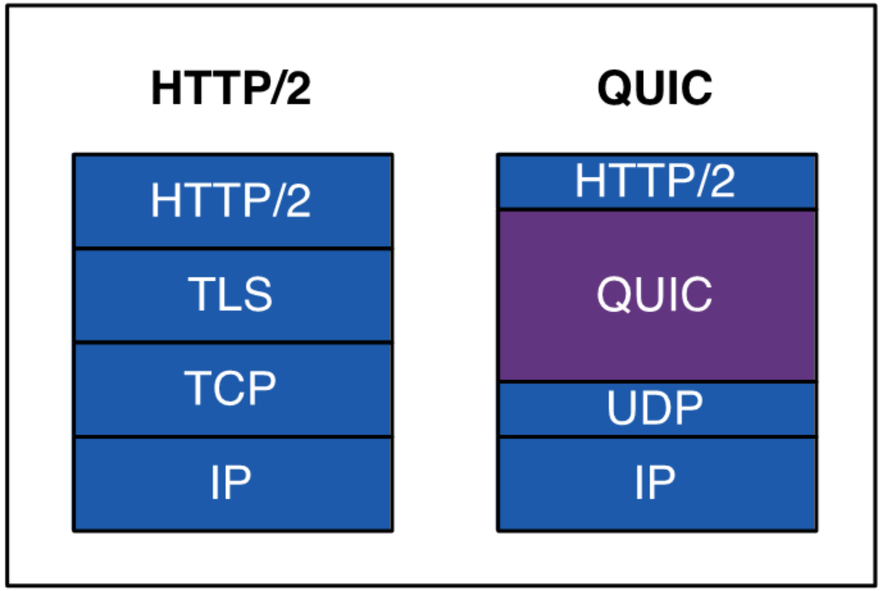
HTTP 2.0

- HTTP 2.0 became standard in 2015, 16 years after version 1.1
  - Google's SPDY was used as a starting point
- Google was able to experiment because it has large user base for browser and server

- **Multiplexing and concurrency**: Several requests can be sent in rapid succession on the same TCP connection, and responses can be received out of order - eliminating the need for multiple connections between the client and the server
- **Stream dependencies:** the client can indicate to the server which of the resources are more important than the others
- **Header compression:** HTTP header size is drastically reduced
- **Server push:** The server can send resources the client has not yet requested

QUIC – Quick UDP Internet Connection

- According to our visitor Anthony Chan (from Huawei/US) last week, the hot topic in IETF now is QUIC
- It is another effort from Google, to make HTTP 2.0 faster
- It is layered on top of UDP, redesigning the way TCP's connection setup, multiplexing of multiple request streams, some error control based on FEC, modified congestion control(?)



Can it pass through firewalls?

- Since HTTP/QUIC uses UDP as transport, will it have problem getting through firewalls?
- Preliminary trials by Google in their browser-server connections have 93% success
- Reason:
  - Server side under Google's control
  - Most firewalls allow UDP at client side
- Google thinks this is easier than changing TCP
  - Existing TCP and UDP are implemented in kernels, and will take a long time to evolve

Summary

- We give a brief review of transport protocol functions
- Reviewed the use of UDP and TCP by internet applications
- Discussed what QUIC is – a new transport protocol in development?

- Next lecture: congestion control and network resource allocation