University of Glasgow

Connection Establishment in a Fragmented Network

Networked Systems (H)

Lecture 2

Colin Perkins | https://csperkins.org/ | Copyright © 2020 University of Glasgow | This work is licensed under the Creative Commons Attribution-NoDerivatives 4.0 International License.
To view a copy of this license, visit http://creativecommons.org/licenses/by-nd/4.0/ or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.

Lecture Outline

- Review of TCP and client-server connection establishment
- Impact of TLS and IPv6 on connection establishment
- Peer-to-peer connections and Network Address Translation
- Problems caused by NAT
- NAT traversal and peer-to-peer connection establishment

Colin Perkins | https://csperkins.org/ | Copyright © 2020 University of Glasgow

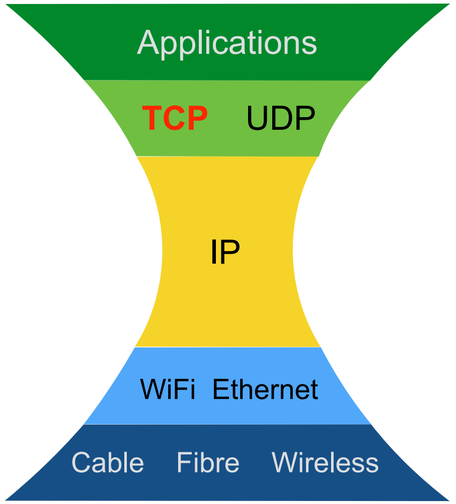
Review of TCP

- The purpose of TCP
- Segment format
- Service model
- TCP programming with BSD Sockets

Colin Perkins | https://csperkins.org/ | Copyright © 2020 University of Glasgow

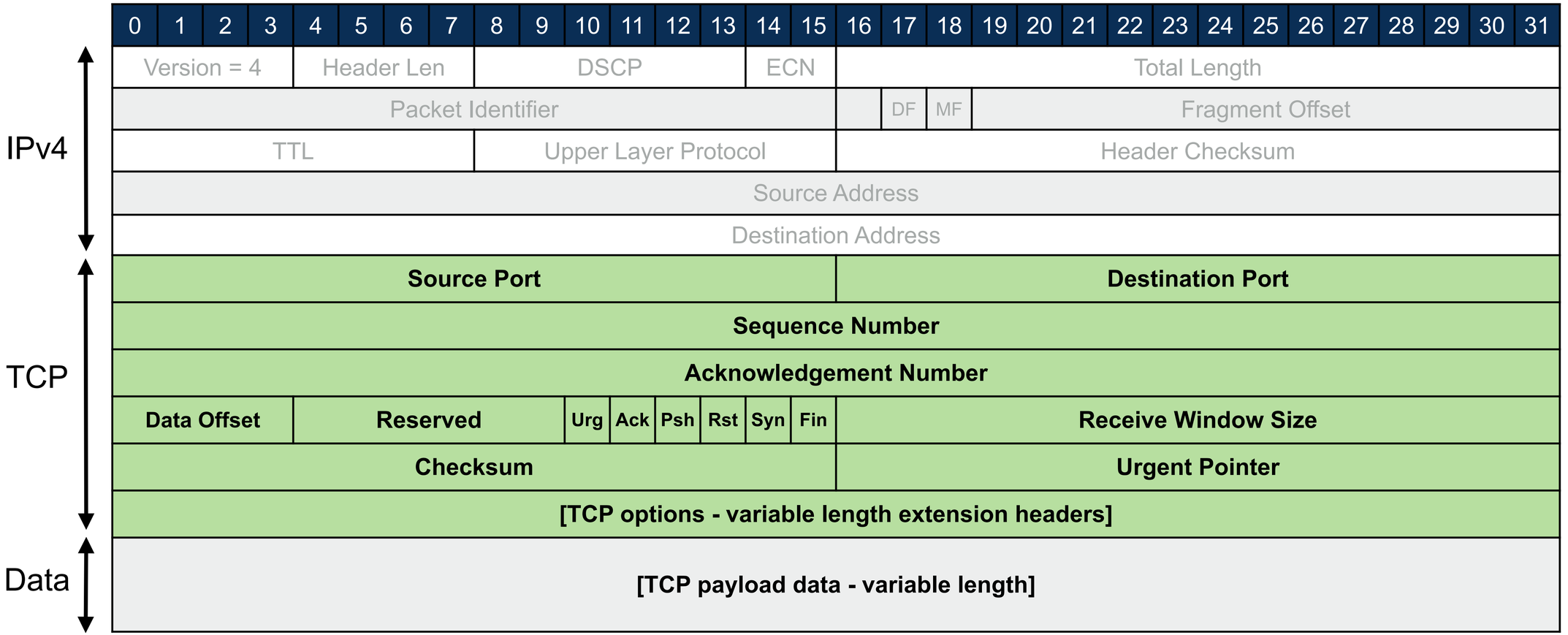
TCP: Reliable Data Transport for the Internet

- TCP transport layer
  - Provides a reliable, ordered, byte stream service over the best-effort IP network
  - Provides congestion control, to adapt to network capacity
  - Most widely used Internet transport protocol
- Delivered within IP
  - TCP **segments** carried as data in IP packets
  - IP **packets** carried as data in link layer frames
  - Link layer **frames** delivered over physical layer



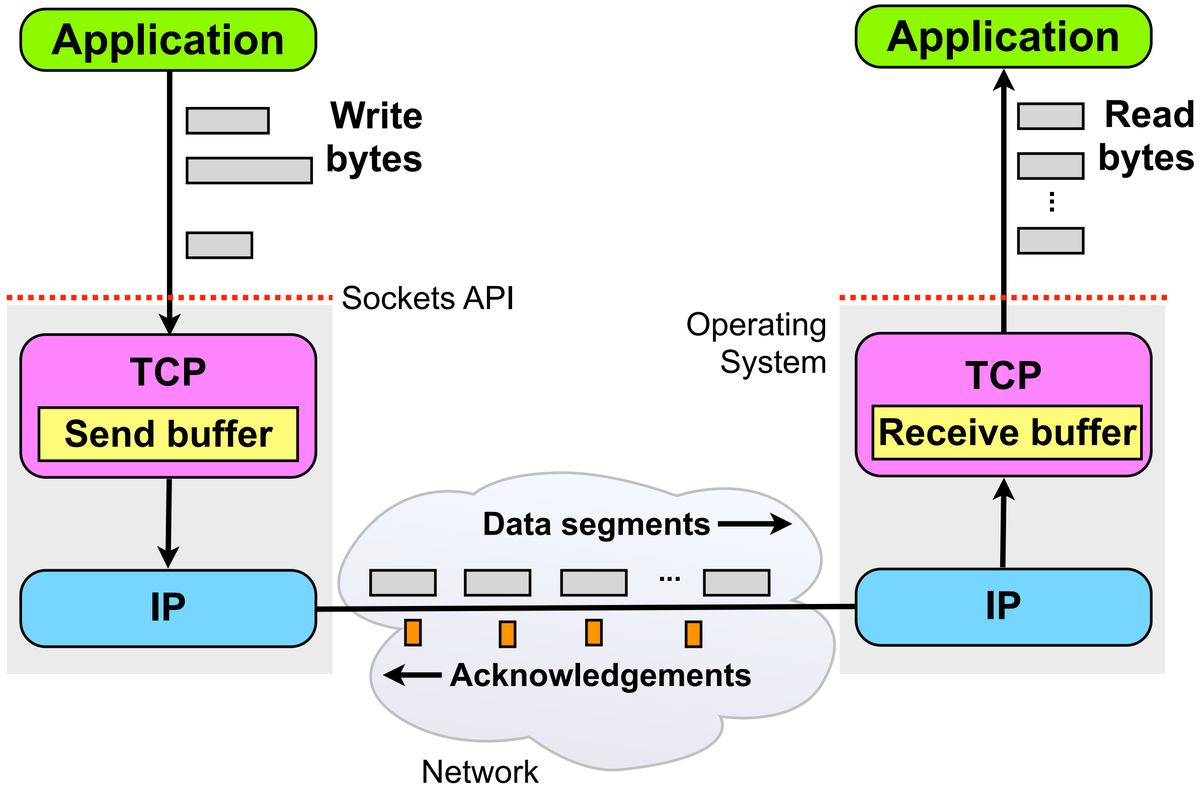
Colin Perkins | https://csperkins.org/ | Copyright © 2020 University of Glasgow

TCP Segment Format



Colin Perkins | https://csperkins.org/ | Copyright © 2020 University of Glasgow

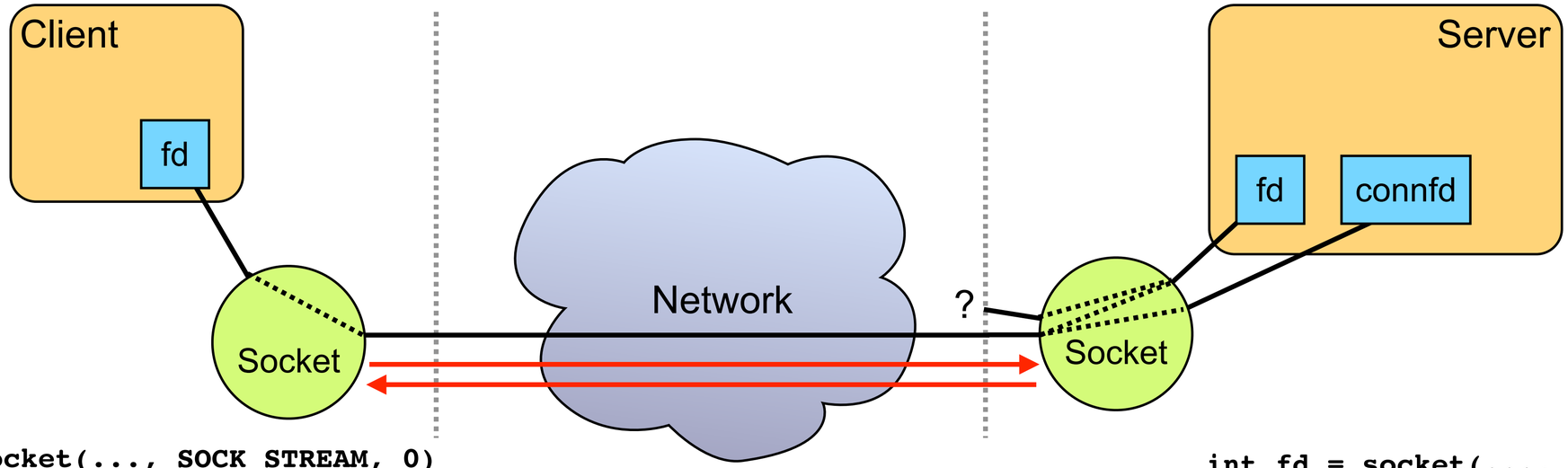
TCP



- Reliable, ordered, byte stream delivery service running over IP
  - Lost packets are retransmitted; ordering is preserved; message boundaries **are not** preserved
  - Adapts sending rate to match network capacity → congestion control
  - Adds port number to identify services
- Used by applications needing reliability → default choice for most applications

Colin Perkins | https://csperkins.org/ | Copyright © 2020 University of Glasgow

TCP Programming Model




```
int fd = socket(..., SOCK_STREAM, 0)

connect(fd, ..., ...)

send(fd, data, datalen, flags)

recv(fd, buffer, buflen, flags)

close(fd)
```

```
int fd = socket(..., SOCK_STREAM, 0)
bind(fd, ..., ...)
listen(fd, ...)
connfd = accept(fd, ...)

recv(connfd, buffer, buflen, flags)

send(connfd, data, datalen, flags)

close(connfd)
```

Colin Perkins | https://csperkins.org/ | Copyright © 2020 University of Glasgow

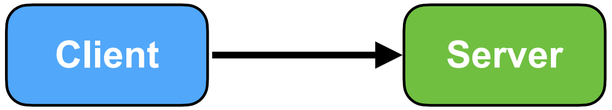
Client-server Connections

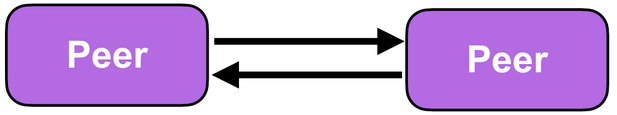
- Connection Establishment and the Impact of the RTT on performance

Colin Perkins | https://csperkins.org/ | Copyright © 2020 University of Glasgow

TCP Connections

- Typical TCP-based applications are client-server
  - Server listens for connections on a well-known port
  - Clients connects to the server
  - Client sends requests; server responds
- Can be used in a peer-to-peer manner
  - The two peers simultaneously **`connect()`** to each other, then send and receiver data
  - Uncommon, but possible – requires both peers to have known and accessible IP addresses, and to **`bind()`** to known ports

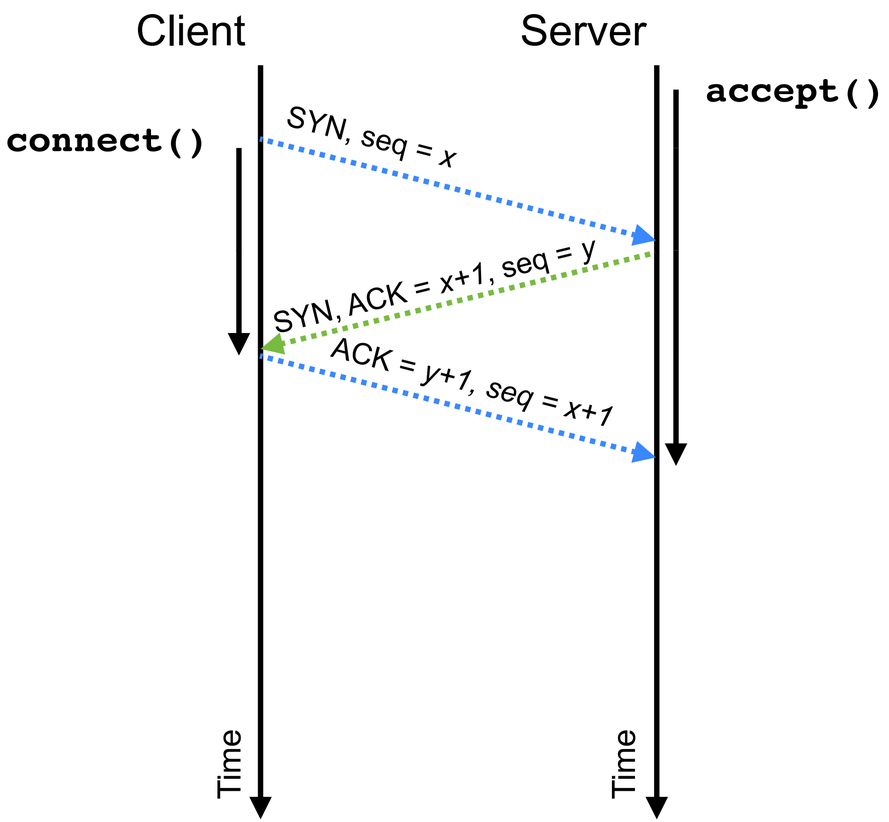




Colin Perkins | https://csperkins.org/ | Copyright © 2020 University of Glasgow

TCP Client-Server Connection Establishment

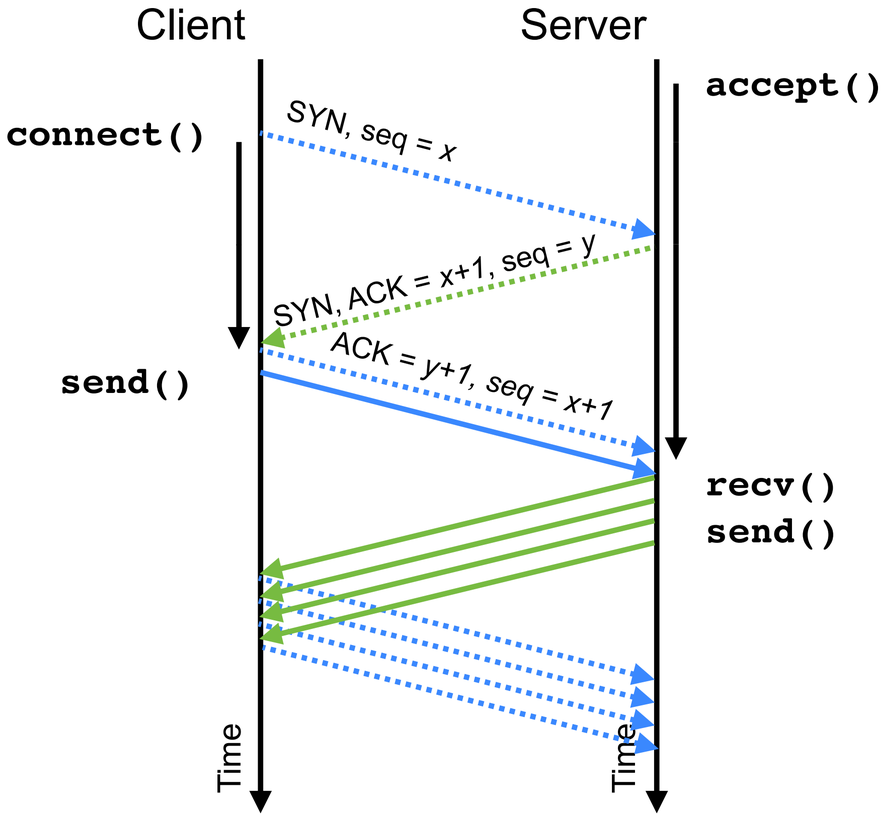
- The server calls **`accept()`** and waits
- The **`connect()`** call triggers the three-way handshake:
  - **Client → Server:**
    - SYN (“synchronise”) bit set in TCP header
    - Client’s initial sequence number chosen at random
  - **Server → Client:**
    - SYN
    - ACK for client’s initial sequence number
    - Server’s initial sequence number chosen at random
  - **Client → Server:**
    - ACK for server’s initial sequence number
- When handshake completes, connection is established



Colin Perkins | https://csperkins.org/ | Copyright © 2020 University of Glasgow

TCP Client-Server Connection Establishment

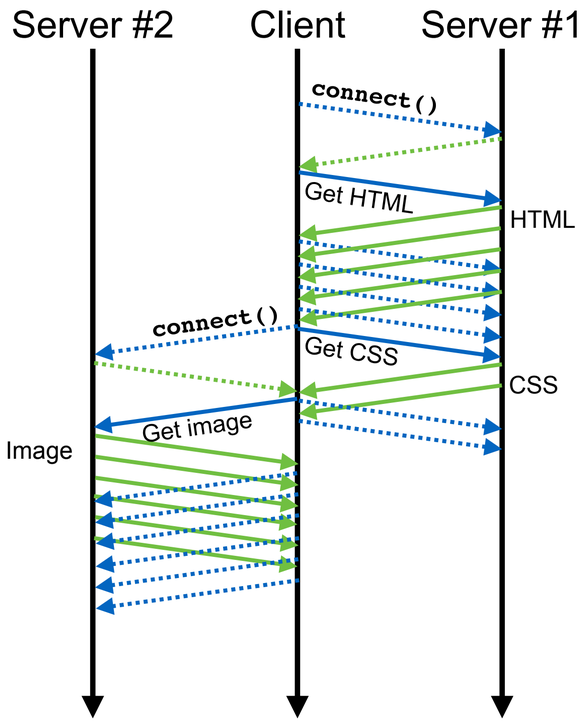
- Calls **`send()`**/**`recv()`** transmit data
- For short flows, initial three-way handshake can take a significant fraction of connection lifetime



Colin Perkins | https://csperkins.org/ | Copyright © 2020 University of Glasgow

Impact of Latency on Performance (1/3)

- Assume a very simple web page:
  - 1x HTML file, 1x CSS file, and 1x image
  - HTML and CSS files hosted on one server, image on a different server
  - Everything retrieved via HTTP over TCP/IP
- How long does it take to retrieve that page?
  - 1x RTT handshake to connect to server 1
  - 1x RTT + serialisation time to fetch HTML file
  - Longest of:
    - 1x RTT + serialisation time to fetch CSS file
  - and
    - 1x RTT handshake to connect to server 2
    - 1x RTT + serialisation time to fetch image



Serialisation time is time taken to transmit data. Example:

- 1Mbyte file = 1,048,576 bytes = 8,388,608 bits
- 2Mbps link = 2,097,152 bits per second
- 8,388,608 / 2,097,152 = 4 seconds to transmit

Colin Perkins | https://csperkins.org/ | Copyright © 2020 University of Glasgow

Impact of Latency on Performance (2/3)

- Performance depends on RTT and available bandwidth
- What is a typical RTT?
  - Depends on distance
  - Depends on network congestion
- What is typical available bandwidth?
  - ADSL2+ → 25Mbps
  - VDSL → 50Mbps
  - 4G wireless → 15-30Mbps

  (all assuming otherwise idle network)

https://www.opensignal.com/reports/2019/04/uk/mobile-network-experience

Destination	Min RTT (ms)	Avg RTT (ms)	Max RTT (ms)
WiFi Router	1.1	32.6	431.7
BBC	33.4	74.6	432.3
Google	33.9	72.5	264.3
Columbia University, New York	103.8	153.3	344.1
University of California, Los Angeles	165.1	221.2	529.8
University of Technology Sydney, Australia	307.0	381.2	685.9

RTT measurements (ping times) from a residential site in Glasgow to various locations around the world

Colin Perkins | https://csperkins.org/ | Copyright © 2020 University of Glasgow

Impact of Latency on Performance (3/3)

- Latency hurts performance
  - Connection establishment is slower
  - Retrieving data takes longer
- As RTT increases, benefits of increasing bandwidth reduce
  - For this example, with 300ms RTT, increasing bandwidth from 15Mbps to 1Gbps gives only 22% reduction in page download time
  - Connection setup time – the 3-way handshake of TCP – dominates in many scenarios

RTT \ BW	1 Mbps	15 Mbps	25 Mbps	50 Mbps	100 Mbps	1 Gbps
1ms	43.9	2.9	1.7	0.9	0.4	0.04 → -90%
50ms	44.1	3.1	1.9	1.1	0.6	0.24
100ms	44.3	3.3	2.1	1.3	0.8	0.44 → -45%
150ms	44.5	3.5	2.4	1.5	1.0	0.64
300ms	45.1	4.1	2.9	2.1	1.6	1.24 → -22%

Download times, in seconds, for the simple web page described earlier, for a range of RTTs and bandwidths

HTML = 500 kbytes, CSS = 100 kbytes, IMG = 5 Mbytes
Assuming no impact due to congestion control

- Is it still worth paying extra for a faster Internet connection?

Colin Perkins | https://csperkins.org/ | Copyright © 2020 University of Glasgow

Example: HTTP/1.1

```
C: GET /index.html HTTP/1.1
C: Accept-Language: en-gb
C: Accept-Encoding: gzip, deflate
C: Accept: text/html, text/plain
C: User-Agent: Safari/523.12.2
C: Cache-Control: max-age=0
C: Connection: keep-alive
C: Host: www.dcs.gla.ac.uk
C:

S: HTTP/1.1 200 OK
S: Date: Wed, 27 Feb 2008 22:44:25 GMT
S: Server: Apache/2.0.46 (Red Hat)
S: Last-Modified: Mon, 17 Nov 2003 08:06:50 GMT
S: Accept-Ranges: bytes
S: Content-Length: 3646
S: Connection: close
S: Content-Type: text/html; charset=UTF-8
S:
S: <HTML>
S: <HEAD>
S: <TITLE>Computing Science, University of Glasgow</TITLE>
S: ...remainder of page elided...
```

- Example: fetch web page using HTTP/1.1
  - Once a connection is open, each object requires only a single request-response exchange
  - Connections are reused for different objects, when possible

Colin Perkins | https://csperkins.org/ | Copyright © 2020 University of Glasgow

Example: SMTP

```
S: 220 mr1.dcs.gla.ac.uk ESMTP Exim 4.42 Wed, 27 Feb 2008 10:31:18 +0000
C: HELO bo720-1-01.dcs.gla.ac.uk
S: 250 mr1.dcs.gla.ac.uk Hello bo720-1-01.dcs.gla.ac.uk [130.209.250.151]
C: MAIL FROM:csp@dcs.gla.ac.uk
S: 250 OK
C: RCPT TO:csp@csperkins.org
S: 250 Accepted
C: DATA
S: 354 Enter message, ending with "." on a line by itself
C: From: Colin Perkins <csp@dcs.gla.ac.uk>
C: To:   Colin Perkins <csp@csperkins.org>
C: Date: Wed 27 Feb 2008 10:32:45
C: Subject: Test
C:
C: This is a test
C: .
S: 250 OK id=1JUJa1-00073j-22
C: QUIT
S: 221 mr1.dcs.gla.ac.uk closing connection
```

- Example: sending email using SMTP
  - Line-by-line request and response:
  - Many unnecessary round trips to server
  - Compare to HTTP/1.1, that bundles all parameters into a single request
  - Poor performance

Colin Perkins | https://csperkins.org/ | Copyright © 2020 University of Glasgow

Client-server connections using TCP

- TCP establishes connections using a three-way handshake
- Performance of a connection depends on latency and bandwidth
- Latency is often dominant factor

Colin Perkins | https://csperkins.org/ | Copyright © 2020 University of Glasgow