Solutions

Marking: **M1** a correct method, **A1** the correct answer, **B1** an independent mark.

Each answer shows the steps, then the **result**.

2.1

file download: TCP, every byte must arrive in order

video call: UDP, a late frame is useless, so do not wait for it

DNS query: UDP, one small question and answer, no set-up time

2.2

SYN-ACK: seq 5000, ack 1001

ACK: seq 1001, ack 5001

2.3

$65\,536 \times 8$ bits per RTT

$\div 0.05$ s

$\approx$ **10.5 Mbit/s**

2.4

bytes 2000 to 3499 have arrived

ACK **3500**

3.1

(a) 1, 2, 4, 8, 16, 32: **5** RTTs

(b) ssthresh $= 40/2 =$ **20**; cwnd = **20**

(c) 1 MSS

(d) $20 \times 1460 \times 8 / 0.1 \approx$ **2.3 Mbit/s**

3.2

(a) 1, 2 and 3: the receiver keeps only in-order packets, so it discarded 2 and 3

(b) only 1

(c) with a larger window, a resent old packet and a new packet can have the same number; the receiver could not tell which it is and might accept old data as new

3.3

(a) $60 \times 100 \times 8 =$ **48 kbit/s**

(b) TCP holds all later data until the lost segment is resent, at least one RTT: about $80 \times 60/1000 \approx 5$ updates arrive late together, so the game stutters

(c) number each update; the client uses the newest and ignores older or missing ones; each update holds the full state, so a lost one is replaced by the next

exercises.lol exercises.com.cn

(d) UDP for position updates and a reliable channel for purchases, e.g. QUIC streams or app-level acknowledgements and resends: movement stays smooth and important events are still guaranteed