National University of Computer and Emerging Sciences

Assignment 02

Chapter 02

CS-3001 Computer Networks – Spring 2026

Section: BSE-6B1 **Max Marks: 50**

Instructions:

- **Show all work.** Full credit is only given for deriving the correct final expressions and values.
- Assume 1 kbps = 10^3 bps, 1 Mbps = 10^6 bps, unless otherwise stated.
- **Answer all questions.** Partial credit may be awarded if your reasoning is clear, even if the final answer is incorrect.
- **Use neat and organized work.** Clearly label all steps, diagrams, and equations.
- **Units matter.** Always include appropriate units in your answers.
- **Academic Integrity:** Students are expected to submit their own original work. Plagiarism, copying from others, or sharing solutions is strictly prohibited and may result in zero marks and disciplinary action.
- **Submission Guidelines:** Submit your assignment by the deadline in the format specified by the instructor. Late submissions may be penalized unless prior permission is granted.

Question	**Max Marks**	**Obtained**
1	20	
2	10	
3	10	
4	5	
5	5	
Total	**50**	

Question 1: Textbook Problems (20 Marks)

Part A: Review Questions (4 + 4 + 4 + 4 = 16 Marks)

R4. For a P2P file-sharing application, do you agree with the statement, "There is no notion of client and server sides of a communication session"? Why or why not?

Solution:

No, I do not agree. While P2P architectures do not have permanently dedicated servers, within any individual communication session (e.g., transferring a specific file chunk), there is a distinct definition of roles. The peer requesting/downloading the file acts as the **client**, and the peer uploading the file acts as the **server**.

R6. Suppose you wanted to do a transaction from a remote client to a server as fast as possible. Would you use UDP or TCP? Why?

Solution:

I would use **UDP**. TCP requires a connection establishment phase (3-way handshake) before data can be sent, which adds at least one Round Trip Time (RTT) of delay. UDP allows the client to send the data immediately in the first packet, making it faster for a single transaction where reliability is handled by the application layer or is not critical.

R13. Describe how Web caching can reduce the delay in receiving a requested object. Will Web caching reduce the delay for all objects requested by a user or for only some of the objects? Why?

Solution:

Web caching reduces delay in two ways:

- **For cached objects (Hits):** The object is served from the local cache (LAN), eliminating the delay of traversing the Internet and the processing time of the origin server.
- **For non-cached objects (Misses):** Although these must still be fetched from the origin, caching reduces the overall traffic intensity on the institution's access link. This reduction in traffic lowers the queuing delay for *all* traffic, including requests for non-cached objects.

Therefore, it reduces delay for **all objects**, though the reduction is most significant for cache hits.

R19. Is it possible for an organization's Web server and mail server to have exactly the same alias for a hostname (for example, foo.com)? What would be the type for the RR that contains the hostname of the mail server?

Solution:

Yes, it is possible. DNS distinguishes services based on the Resource Record (RR) type.

- The IP address of the Web server would be stored in an **A** record (or pointed to via a CNAME).
- The hostname of the mail server is stored in an **MX** (Mail Exchange) record.

Therefore, the type for the RR containing the mail server hostname is **MX**.

Part B: Problems (4 Marks)

P13. Consider sending over HTTP/2 a Web page that consists of one video clip, and five images. Suppose that the video clip is transported as 2000 frames, and each image has three frames.

(a) If all the video frames are sent first without interleaving, how many "frame times" are needed until all five images are sent?

(b) If frames are interleaved, how many frame times are needed until all five images are sent.

Solution:

(a) Without Interleaving:
The server sends the entire video first, then the images.

- Video: 2000 frames.
- 5 Images $\times$ 3 frames/image = 15 frames.

The images are sent only after the video is complete. Total time $= 2000 + 15 =$ **2015** frame times.

(b) With Interleaving:
We have 6 streams (1 Video + 5 Images). The server sends one frame from each stream in a round-robin fashion.

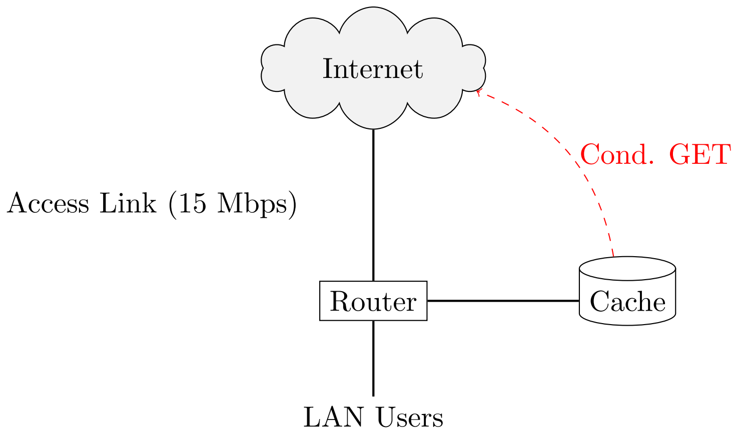
- Round 1: Video(1), Img1(1), Img2(1), Img3(1), Img4(1), Img5(1). (6 frames)
- Round 2: Video(2), Img1(2), Img2(2), Img3(2), Img4(2), Img5(2). (6 frames)
- Round 3: Video(3), Img1(3), Img2(3), Img3(3), Img4(3), Img5(3). (6 frames)

At the end of Round 3, all images (which only have 3 frames each) are fully transmitted. Total time = 3 rounds $\times$ 6 frames/round = **18** frame times.

Question 2: Web Cache & Link Utilization (10 Marks)

Scenario: An institution has a 15 Mbps access link to the Internet. The average object size is 100 Kb. The average request rate from browsers to origin servers is 100 requests/sec.

- $RTT_{internet} = 2$ seconds (average).
- $RTT_{LAN} = 10$ ms.



The institution installs a Cache Server.

- Hit rate = 0.4 (40% of requests served by cache).
- **Twist:** The cache performs a Conditional GET for every "Hit" to ensure freshness.
- If Data is fresh (304 Not Modified): Response size is small (header only) = 1 Kb.
- If Data is stale (200 OK): Response size is full object = 100 Kb.
- Assume that for the 40% "Hits", 10% of them turn out to be stale (requiring full download from origin), and 90% are fresh (small download).
- The 60% "Misses" are standard full downloads from origin.

Calculate the traffic intensity (utilization) on the Access Link.

Solution:

1. Break down the Request Rate (100 req/s):

- **Misses (60%):** $0.6 \times 100 = 60$ req/s.
- **Hits (40%):** $0.4 \times 100 = 40$ req/s.
  - Stale Hits (10% of 40): 4 req/s.
  - Fresh Hits (90% of 40): 36 req/s.

2. Calculate Traffic on Access Link: Only data entering the router from the Internet counts towards link utilization.

- **Traffic from Misses:** 60 req/s $\times$ 100 Kb/req = 6000 Kbps.
- **Traffic from Stale Hits:** 4 req/s $\times$ 100 Kb/req = 400 Kbps.
- **Traffic from Fresh Hits:** 36 req/s $\times$ 1 Kb/req = 36 Kbps.

3. Total Traffic:

$$R_{total} = 6000 + 400 + 36 = 6436 \text{ Kbps} = 6.436 \text{ Mbps}$$

4. Traffic Intensity (Utilization):

$$\text{Utilization} = \frac{R_{total}}{R_{link}} = \frac{6.436 \text{ Mbps}}{15 \text{ Mbps}}$$

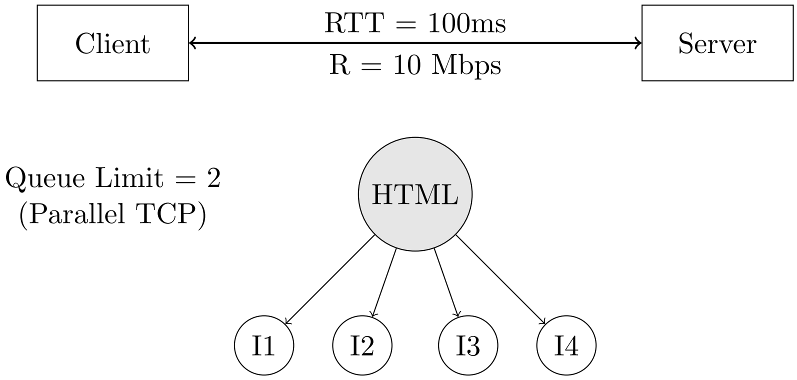
$$\text{Utilization} = 0.429 = \mathbf{42.9}\%$$

Question 3: HTTP Response Time & Parallel Connections (10 Marks)

Scenario: Consider a Web page that contains a base HTML object of size $L_{base} = 10$ KB. This HTML file references 4 distinct JPEG images, each of size $L_{img} = 100$ KB.

- The link bandwidth between the client and server is $R = 10$ Mbps.
- The Round Trip Time (RTT) is 100 ms.
- Control packets (ACKs, SYNs, HTTP Requests) are negligible in size.

- The server requires 20 ms of processing time for any request before starting transmission.



Calculate the total time elapsed from the moment the user clicks the link until the browser finishes receiving all objects assuming:

1. Non-persistent HTTP with **Parallel TCP connections** (Max 2 simultaneous connections).
2. The browser cannot parse the references to the images until the entire HTML file is received.
3. The 10 Mbps client-server link is the only bottleneck and is shared by all simultaneous connections. If two objects are downloaded simultaneously, each receives 5 Mbps.

Solution:

1. Transmission Times:

$$T_{base} = \frac{10 \text{ KB} \times 8}{10 \text{ Mbps}} = \frac{80 \text{ kb}}{10000 \text{ kbps}} = 0.008 \text{ s} = 8 \text{ ms}$$

$$T_{img} = \frac{100 \text{ KB} \times 8}{10 \text{ Mbps}} = \frac{800 \text{ kb}}{10000 \text{ kbps}} = 0.080 \text{ s} = 80 \text{ ms}$$

2. Step 1: Base HTML Object

- TCP Handshake: 1 RTT = 100 ms.
- Request + Processing + Transmission: $RTT + Proc + T_{base} = 100 + 20 + 8 = 128$ ms.
- Total Step 1 = $100 + 128 = 228$ ms.

HTML is parsed. Browser finds 4 images. Max 2 connections allowed.

3. Step 2: Images 1 & 2 (Parallel Batch 1)

- Non-persistent means new connections.
- Handshake: 1 RTT = 100 ms.
- Request + Processing + Trans:
- Note: Since 2 images download in parallel over the bottleneck link R, the effective rate is shared, or simply sum the transmission times.
- Time for data = $RTT + Proc + (2 \times T_{img}) = 100 + 20 + 160 = 280$ ms.
- Total Step 2 = $100 + 280 = 380$ ms.

4. Step 3: Images 3 & 4 (Parallel Batch 2)

- Identical to Step 2.

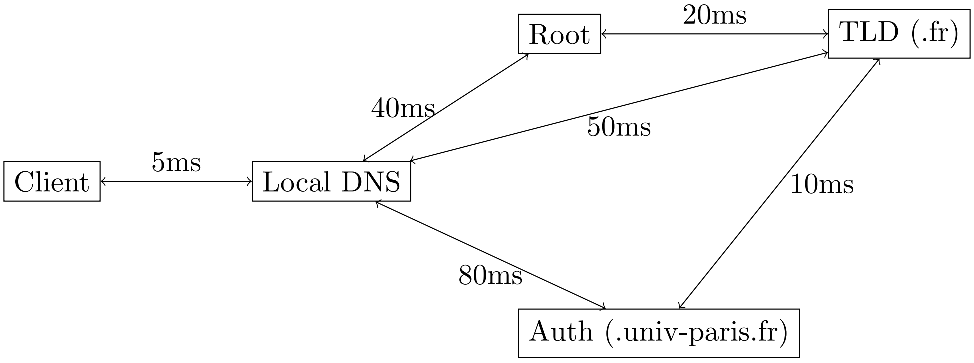
- Total Step 3 = 380 ms.

Total Response Time:

$$T_{total} = 228 + 380 + 380 = \mathbf{988} \text{ ms}$$

Question 4: Recursive DNS & Latency (5 Marks)

Scenario: A client in New York requests the IP for `www.univ-paris.fr`. The local DNS server is empty. The hierarchy and One-Way delays (OWD) are modeled below. Assume processing time is negligible.



Determine the total resolution time experienced by the client under the following mixed-mode scenario:

1. The Client communicates with Local DNS.
2. Local DNS queries Root Iteratively.
3. Root returns the address of TLD.
4. Local DNS queries TLD **Recursively** (i.e., TLD acts on behalf of Local to contact Auth).
5. TLD returns the final Answer to Local.
6. All listed delays are one-way (OWD). In particular, the link Local DNS to TLD (.fr) has an OWD of 50 ms (not 20 ms)

Solution:

We sum the delays for each leg of the query path:

1. **Client ↔ Local:** Query goes up, Answer comes down. Time = 5 ms (up) + 5 ms (down) = 10 ms.
2. **Local ↔ Root (Iterative):** Local asks Root, Root replies with TLD address. Time = 40 ms (up) + 40 ms (down) = 80 ms.
3. **Recursive Chain (Local → TLD → Auth → TLD → Local):**
   - Local → TLD (Query): 50 ms.
   - TLD → Auth (Query): 10 ms.

- Auth → TLD (Answer): 10 ms.
- TLD → Local (Answer): 50 ms.

Total Chain $= 50 + 10 + 10 + 50 = 120$ ms.

Total Resolution Time:

$$T_{total} = 10 + 80 + 120 = \mathbf{210} \text{ ms}$$

Question 5: Multiplexing and Latency (5 Marks)

Scenario: Consider an HTTP/2 connection downloading 4 distinct objects:

- **Object A:** 4 frames (A1, A2, A3, A4)
- **Object B:** 2 frames (B1, B2)
- **Object C:** 1 frame (C1)
- **Object D:** 3 frames (D1, D2, D3)

Object A (4)	A1	A2	A3	A4
Object B (2)	B1	B2		
Object C (1)	C1			
Object D (3)	D1	D2	D3	

Each frame takes **8 ms** to transmit. Assume all objects become ready at the same time and HTTP/2 uses **Round Robin scheduling** among active streams to prevent Head-of-Line (HOL) blocking.

1. List the exact sequence of frame arrivals at the client and the time interval for each frame transmission under HTTP/2 Round Robin scheduling.
2. Assume HTTP/1.1 without pipelining, where objects are downloaded strictly in the order $A \rightarrow B \rightarrow C \rightarrow D$.
   (a) Calculate the completion time of Object D in HTTP/1.1 and compare it with its completion time in HTTP/2.
   (b) How much earlier (in ms) does Object C finish in HTTP/2 compared to HTTP/1.1? Show calculations.

Solution:

We sum the transmission times for each object/frame in the HTTP/2 Round Robin schedule:

1. **HTTP/2 Frame Transmission (Round Robin):** Frames are sent in cycles across objects A, B, C, D. Each frame takes 8 ms. Sequence of frames:
   - **Cycle 1:** A1, B1, C1, D1 (C finishes)
   - **Cycle 2:** A2, B2, D2 (B finishes)

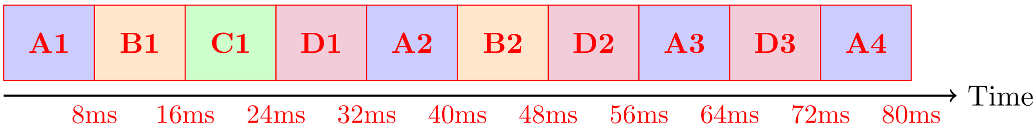


Figure 1: HTTP/2 Frame Transmission Timeline

- **Cycle 3:** A3, D3 (D finishes)
- **Cycle 4:** A4 (A finishes)

2. **Sequence Table (Frame Completion Times):**

Seq	Frame	Time (ms)	Note
1	A1	8	
2	B1	16	
3	C1	24	**Object C Finished**
4	D1	32	
5	A2	40	
6	B2	48	**Object B Finished**
7	D2	56	
8	A3	64	
9	D3	72	**Object D Finished**
10	A4	80	**Object A Finished**

3. **HTTP/1.1 Sequential Transmission:** Each object is sent fully before the next starts. Each frame = 8 ms:
   - Object A (4 frames): $4 \times 8 = 32$ ms (Finish: 32 ms)
   - Object B (2 frames): $32 + 2 \times 8 = 48$ ms (Finish: 48 ms)
   - Object C (1 frame): $48 + 1 \times 8 = 56$ ms (Finish: 56 ms)
   - Object D (3 frames): $56 + 3 \times 8 = 80$ ms (Finish: 80 ms)

4. **Comparison of Completion Times:**
   - **Object D:** HTTP/1.1 = 80 ms, HTTP/2 = 72 ms $\Rightarrow$ finishes 8 ms earlier.
   - **Object C:** HTTP/1.1 = 56 ms, HTTP/2 = 24 ms $\Rightarrow$ finishes 32 ms earlier.

Total Insight: HTTP/2 Round Robin interleaves frames from multiple objects, reducing the completion time of smaller objects compared to HTTP/1.1 sequential transmission.