15-440

Distributed Systems

Recitation 5

Slides By: Hend Gedawy

& Ammar Karkour

Announcements

Grades for Design Report – Today

Problem Set 2

Due: Sep. 26th (Tuesday)

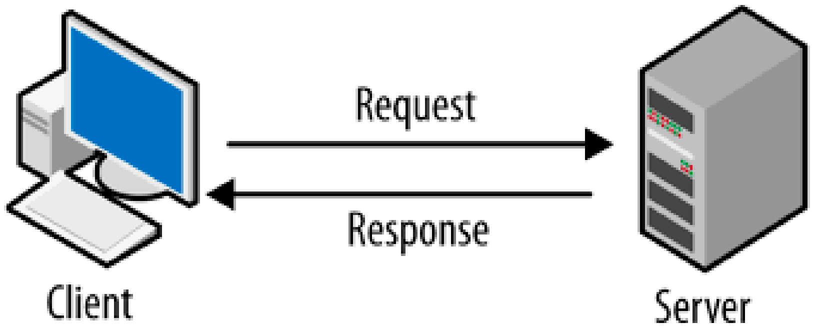
Outline

- **RPC Failure Sources**
- Handling Different Types or RPC Failures
  - Communication Failures
  - Client-Side Failures
  - Server-Side Failures
- Calculating RPC Time
- Exercises
- Feedback Reflection
- Project 1 & PS2 Questions

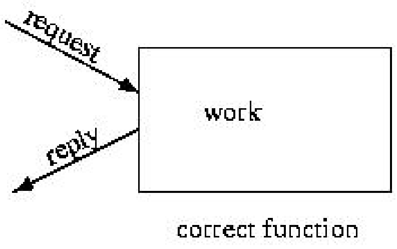
Remote Procedure Calls (RPC) Failures

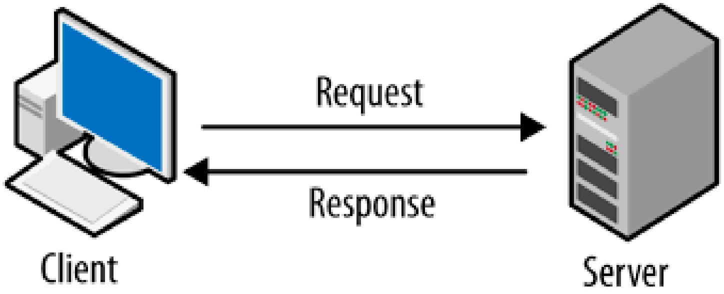
- Failures should be taken into consideration when RPC calls are made.

Where Can Failures Happen & How to Handle Them

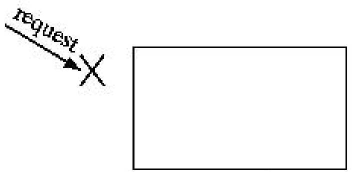


RPC Failures – Where Can Failure Happen?

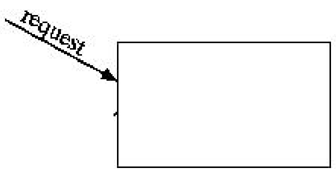




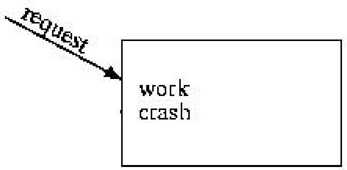
Failures



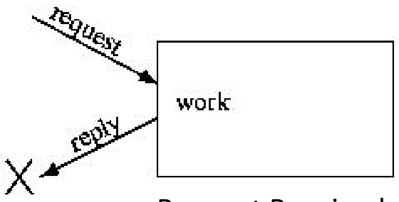
Request gets lost



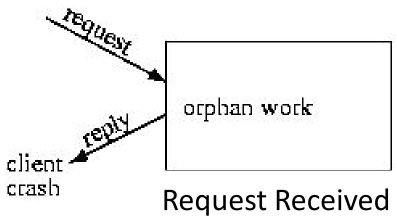
Request Received
Server Crashes



Request Received
Work Done
Server Crashes



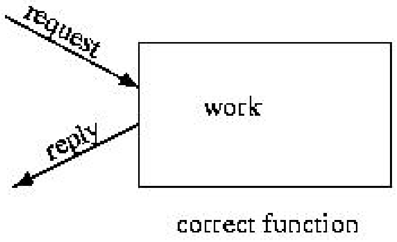
Request Received
Work Done
Reply gets Lost

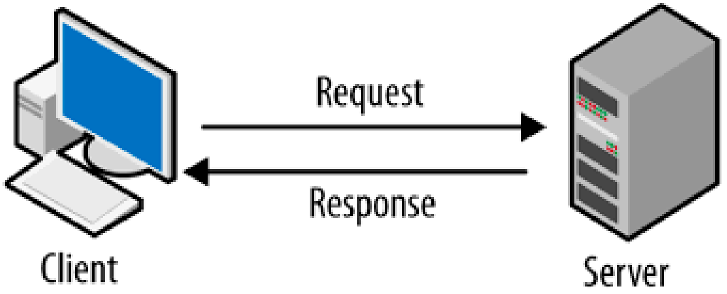


Request Received
Work Done
Reply Sent
Client Crashes Before
Receiving Result

Source: https://cseweb.ucsd.edu/classes/sp16/cse291-e/applications/ln/lecture3.html

RPC Failures – Types





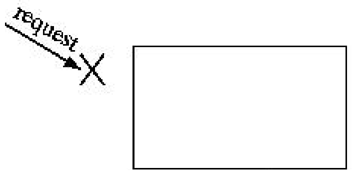
Can be Categorized Into Three Types:

Communication Failures

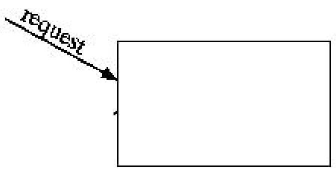
Client Failures

Server Failures

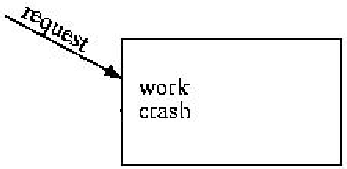
Failures



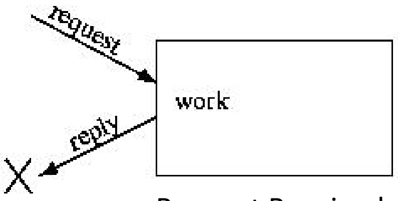
Request gets lost



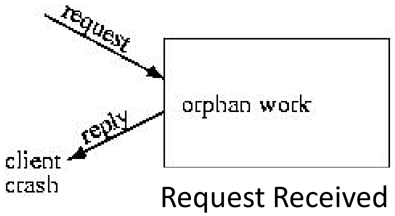
Request Received
Server Crashes



Request Received
Work Done
Server Crashes



Request Received
Work Done
Reply gets Lost



Request Received
Work Done
Reply Sent
Client Crashes Before Receiving Result

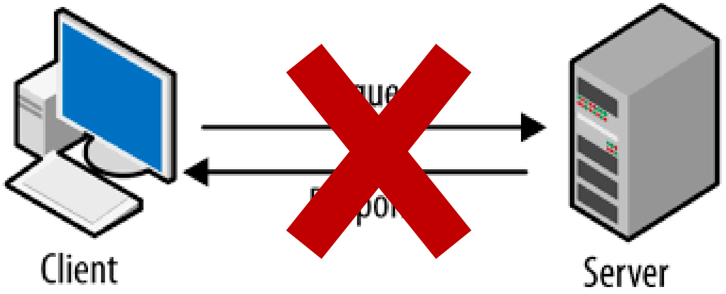
Source: https://cseweb.ucsd.edu/classes/sp16/cse291-e/applications/ln/lecture3.html

Outline

- RPC Failure Sources
- **Handling Different Types or RPC Failures**
  - Communication Failures
  - Client-Side Failures
  - Server-Side Failures
- Calculating RPC Time
- Exercises
- Feedback Reflection
- Project 1 & PS2 Questions

Outline

- RPC Failure Sources
- **Handling Different Types or RPC Failures**
  - **Communication Failures**
  - Client-Side Failures
  - Server-Side Failures
- Calculating RPC Time
- Exercises
- Feedback Reflection
- Project 1 & PS2 Questions



Handling RPC Failures - Communication

- If the client is unable to find the server,
  - The client should return an error message.

- If client and server both are running but something goes wrong on the network and….

  RPC request packet doesn't reach the server for processing or

  the server sent the reply back to client but client never received the reply.

Retransmission is Needed!

Who should handle Retransmission?

Depends on the Transport Layer Protocol that RPC is layered on top of

Handling RPC Failures - Communication

- Who should Handle Retransmission of Lost Requests/Responses?

- If the **RPC system is running over UDP,** then RPC should handle it using a timeout mechanism:
  - The client should implement a timeout mechanism. If the client doesn't receive response from server before the timeout then client should resend the RPC request.
  - Also, server in some RPC semantics (as we will see) will set a timeout waiting for client's ACK after it sends the response.

If the **RPC system is running over TCP**, then the error correction task should be offloaded to the **TCP layer**.

A packet that is sent will arrive at the other end eventually.

But this doesn't handle Failures at End Points!

Handling RPC Failures - Communication

When is it better to use TCP & UDP for RPC?

UDP: Faster but Unreliable

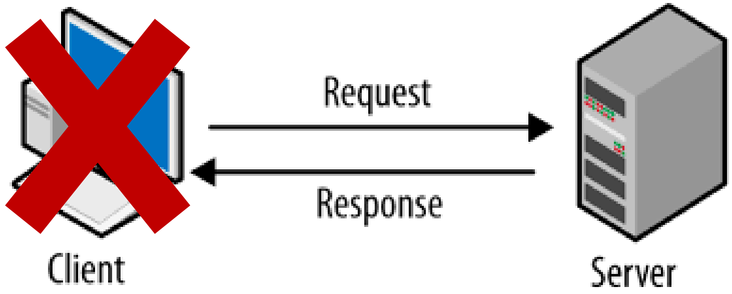
- in LAN,
- For certain kinds of Apps; e.g.:
  - Live streaming Apps
  - Apps that require fast communication (VoIP, online games, etc.)

TCP: Slower but Reliable

- in WAN,
- Apps that require full transmission of data (e.g. online banking)
- Large messages (as TCP can fragment)

Outline

- RPC Failure Sources
- **Handling Different Types or RPC Failures**
  - **Communication Failures**
  - **Client-Side Failures**
  - Server-Side Failures
- Calculating RPC Time
- Exercises
- Feedback Reflection
- Project 1 & PS2 Questions



RPC Failure – Client Side

Case 1

Server **Finished Operation** But **Client** is **not available** to receive response

Most Servers **Discard Response**

Case 2

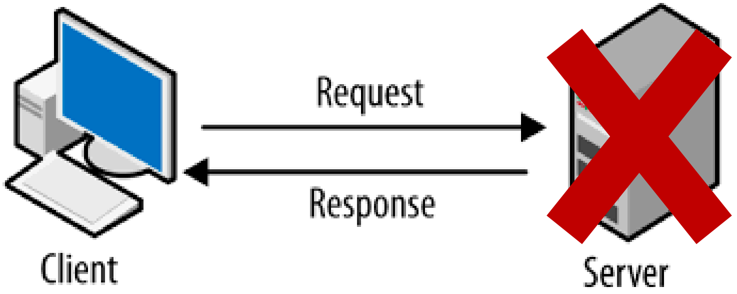
While server is **processing** log operation, it **learns** that the **client crashed**.

1) It previously **maintain**/track **clients status**. Once it knows a client **crashed, terminates** RPC **operation**
2) **Check** each RPC **process** with time **intervals**. If one takes so **long**, it **checks** on **client**, and **delete operations if** client **crashed**
3) Server **broadcasts** to **check** if **client** joined with a **different** process ID
4) Server sets a **timeout** at the start, and **checks on the client at that time**, **if** it is still **alive**, **continue** computation with a **new timeout**. Repeat until done

Source: https://lass.cs.umass.edu/~shenoy/courses/spring13/lectures/notes/677_lec09.pdf

Outline

- RPC Failure Sources
- **Handling Different Types or RPC Failures**
    - **Communication Failures**
    - **Client-Side Failures**
    - **Server-Side Failures**
- Calculating RPC Time
- Exercises
- Feedback Reflection
- Project 1 & PS2 Questions

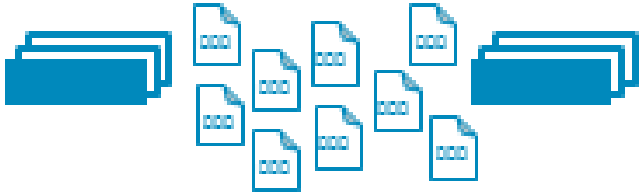


RPC Failures – Server Side

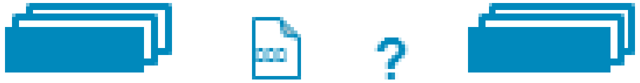
Server Could crash before or after work is done

We studied **3 semantics** to determine **how frequently the remote operation can run** in the event of failure

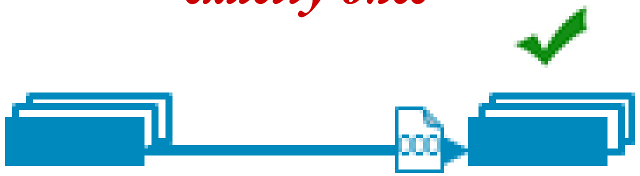
at-least-once



at-most-once



exactly-once



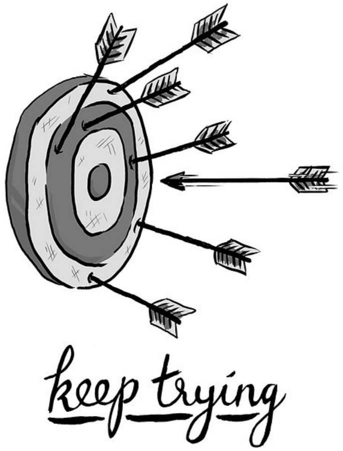
Photos Credit: https://www.datanami.com/2017/02/09/exactly-big-deal-apache-kafka/

RPC Server Failure Semantics *at-least-once*

- If the **client received a reply** from the server, it means that **call has been executed at least once** on the server.
- Doesn't apply **duplicate request filtering**
  Whenever **client timesout,** it will **resend request**
- If **server crashed** during execution or after execution but before sending reply
  Server **comes up and** then **again executes** the call and **sends the reply** to the client.
  That is why it is Suitable For **Idempotent** Operations
- How does **it make sure client received the reply**?
  It waits for client **ACK. If lost**, and server **times out**, it will **retransmit reply**
- **Use Cases:** Applications where receiving every message is important & duplicity is acceptable (Queries)

Worked

Didn't Work



RPC Server Failure Semantics *at-most-once*

- If the **client gets a reply** from the server, it means that the RPC call have been **executed at most once** on the server.
- If **timeout doesn't happen** at client, it means server must have **succeeded** in the first try
- Applies **duplicate requests filtering**
- Suitable for **non-idempotent operations**
- When **operation request** is **received**
  - If **executed before,** it **looks it up** from non-volatile memory & sends it
  - If **not executed before,** process it
- If **server crashes** during execution?
  - If a failure happens before all actions are executed, roll back (ensures *atomicity*)
  - If a failure happens while results are in volatie memory, Redo (ensures *durability*)
- **It doesn't worry about client receiving the reply**
- **Use Cases:** Applications that need high throughput and low latency (collecting sensor measurements)

Worked

Didn't Work (timeout)

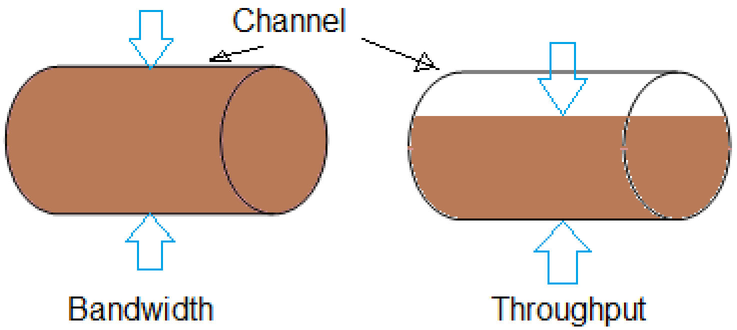
RPC Server Failure Semantics *exactly-once*

- This is the **most desirable** scenario, no matter what happened on the server side (crashes / restarts)
- If the client gets back reply, then it is confirmed that the RPC call was **executed exactly once** on the server
- Applies **duplicate requests filtering**
- Server **keeps results** in cache **until client ACKs**
- Most **like the local-procedure calls** BUT **very expensive** to implement!
    - *That's why systems implement weaker semantics and errors need to be handled explicitely*
- Use Cases: **Financial Operations**

Outline

- RPC Failure Sources
- **Handling Different Types or RPC Failures**
  - **Communication Failures**
  - **Client-Side Failures**
  - **Server-Side Failures**
- **Calculating RPC Time**
- Exercises
- Feedback Reflection
- Project 1 & PS2 Questions

Bandwidth v.s. Throughput



Bandwidth: The **maximum** speed of bit transmission in a channel or a link, measured in **bits/sec**.

Throughput: The **actual** speed of bit transmission in a channel or a link, measured in **bits/sec**.

Why are they different?

- Because networks often experience issues that hinder performance (e.g., overloading).

RPC Time- Communication Time

Transmission time: The time it takes to upload your message from your machine to the transferring medium (link)

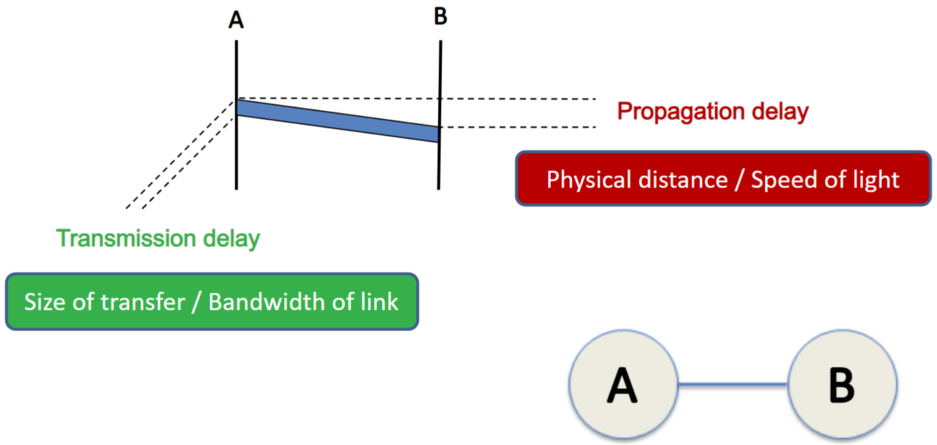
- Measured from when first bit of data is pushed ionto the link until last bit of data is pushed onto the link

Transmission time = Message Size / Throughput
(P.S: Throughput might be equal to Bandwidth in some situations)

Propagation time: The time it takes for a bit in your message to travel across the transferring medium from source to destination.

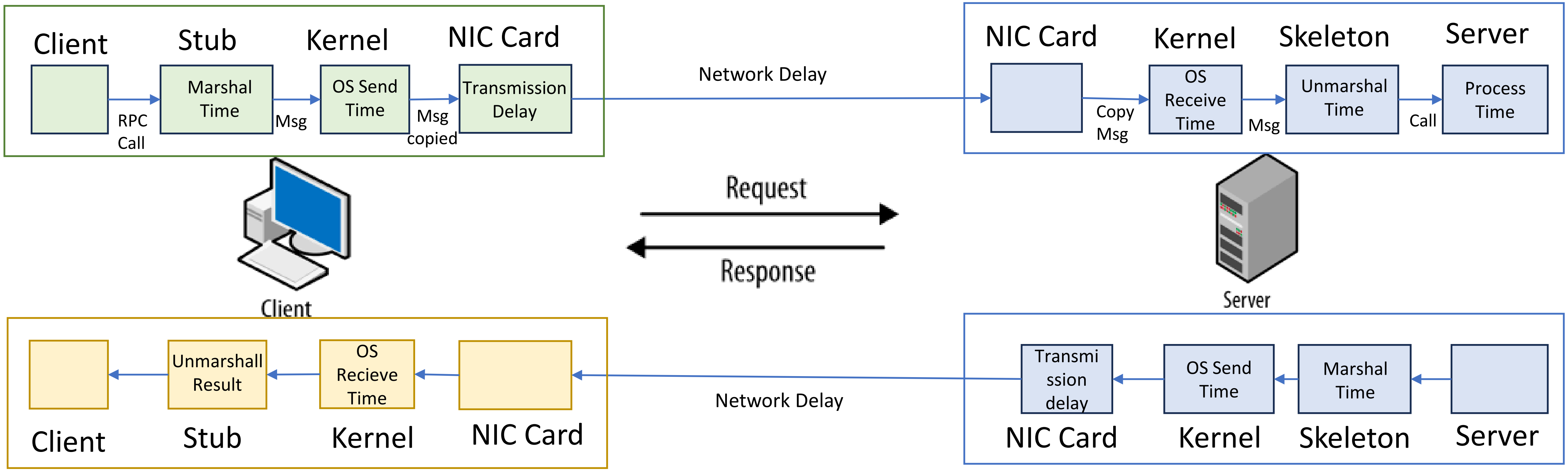
Propagation time= Distance/ Speed of Light

Communication time = Transmission time + Propagation time


A
B
Propagation delay
Physical distance / Speed of light
Transmission delay
Size of transfer / Bandwidth of link
A
B

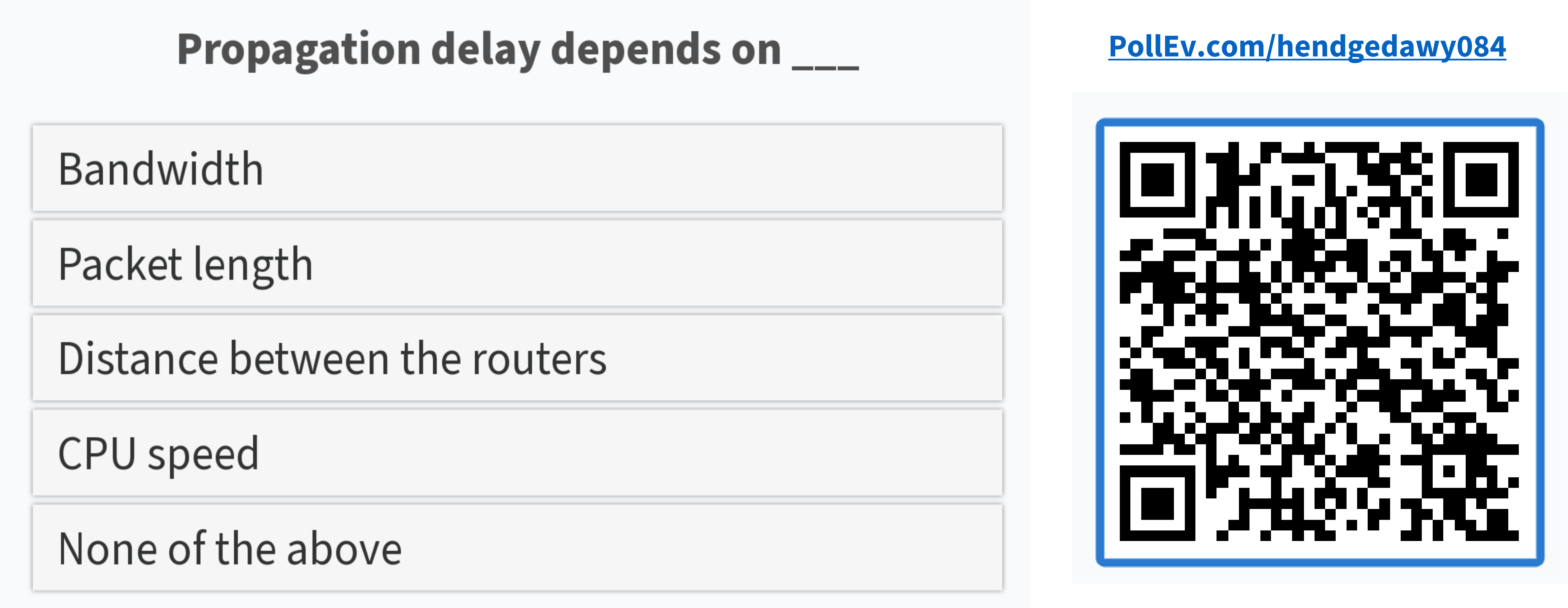
RPC Total Time

RPC Total Time= TimeToSendRequest+ TimeToReceiveRequest+TimeToProcess+TimeToSendResults+ TimeToReceiveResult



Outline

- RPC Failure Sources
- **Handling Different Types or RPC Failures**
  - **Communication Failures**
  - **Client-Side Failures**
  - **Server-Side Failures**
- **Calculating RPC Time**
- **Exercises**
- Feedback Reflection
- Project 1 & PS2 Questions



Propagation delay depends on ___

- Bandwidth
- Packet length
- Distance between the routers
- CPU speed
- None of the above

PollEv.com/hendgedawy084

Transmission delay does not depend on ___________

PollEv.com/hendgedawy084

Packet length

Distance between the routers

Bandwidth / Throughput of medium

None of the above

As Bandwidth × Length increases, uncertainty increases

PollEv.com/hendgedawy084

True

False

PollEv.com/hendgedawy084

A network with bandwidth of 10 Mbps can pass 12,000 frames per minute with each frame carrying 10,000 bits. What is the throughput of this network as a percentage of the bandwidth?

- 10%
- 12%
- 20%
- 25%

Calculate the propagation time needed for a bit to travel between two points with a connecting cable length of 12,000 m. Assuming that the cable's propagation speed is 2.4 * 10^8 m/s

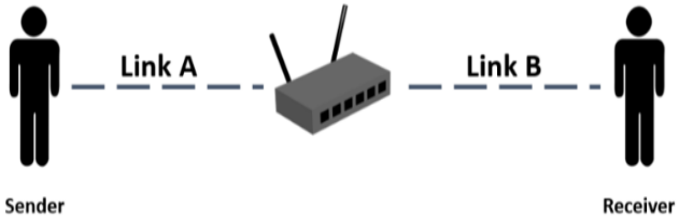
PollEv.com/hendgedawy084

- 50 ms
- 5 ms
- 0.5 ms
- 0.05 ms

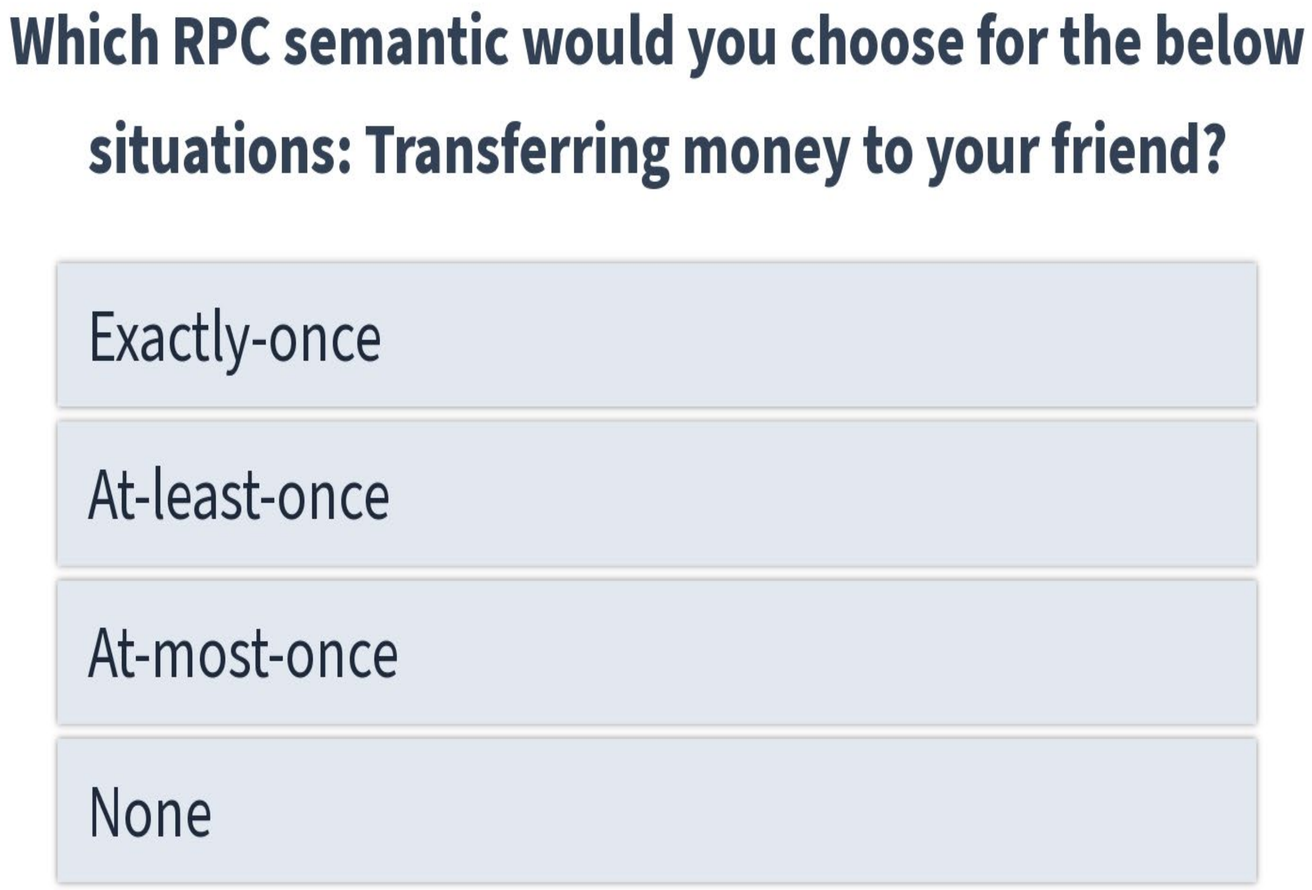
Sender "ping"s a receiver in a different country. Ping would send a packet of size 64 bytes. What is the minimum time that it will take the ping packet to return to the sender?

PollEv.com/hendgedawy084

Link	One-way propagation delay	Capacity
Link A	10ms	1Gbit/second
Link B	20ms	1Gbit/second



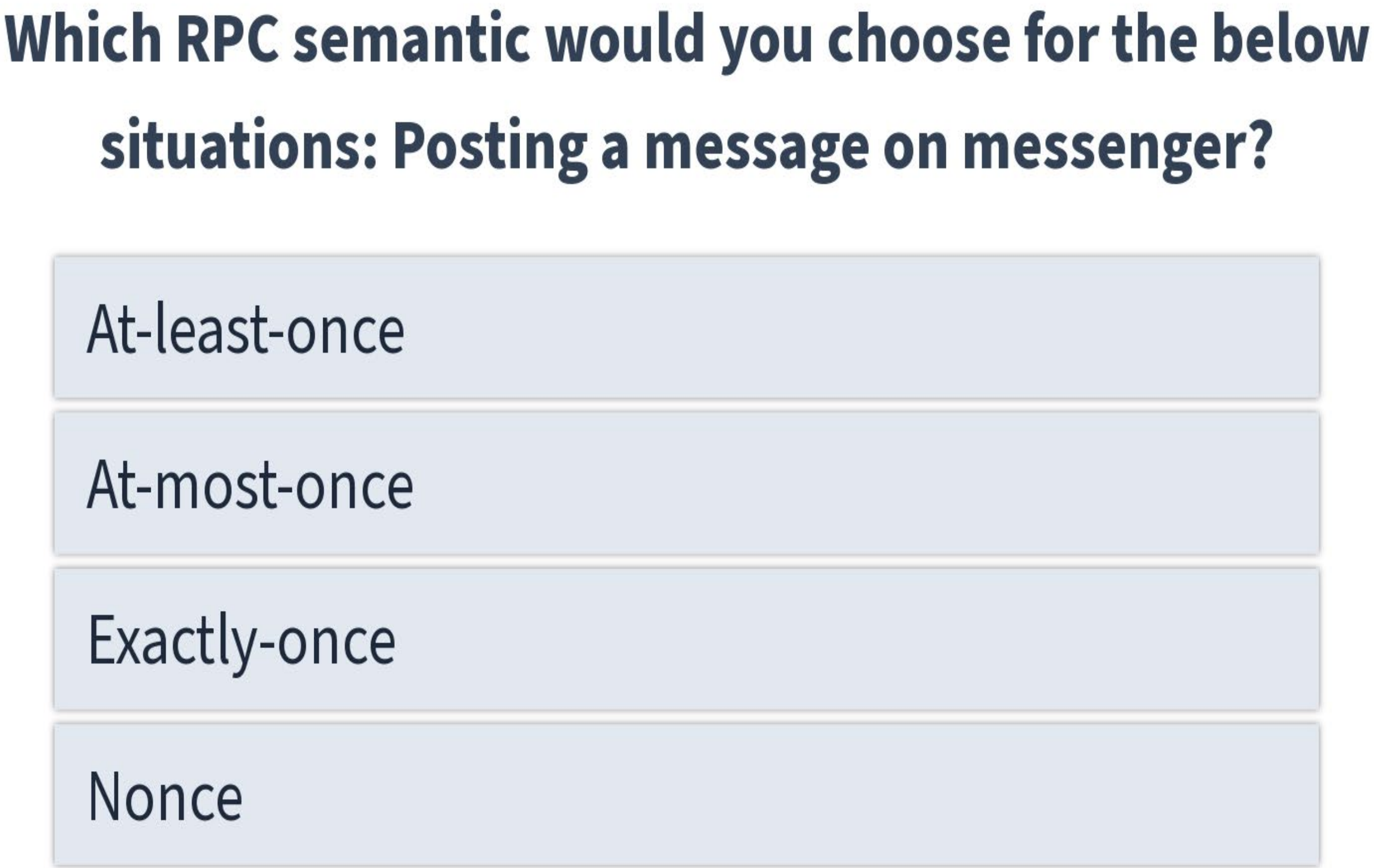
- (2 * 64) / 1073741824 + 30 ms
- (2 * 64 * 8) / 1073741824 + 30 ms
- (4 * 64) / 1073741824 + 60 ms
- (4 * 64 * 8) / 1073741824 + 60 ms



Which RPC semantic would you choose for the below situations: Transferring money to your friend?

- Exactly-once
- At-least-once
- At-most-once
- None

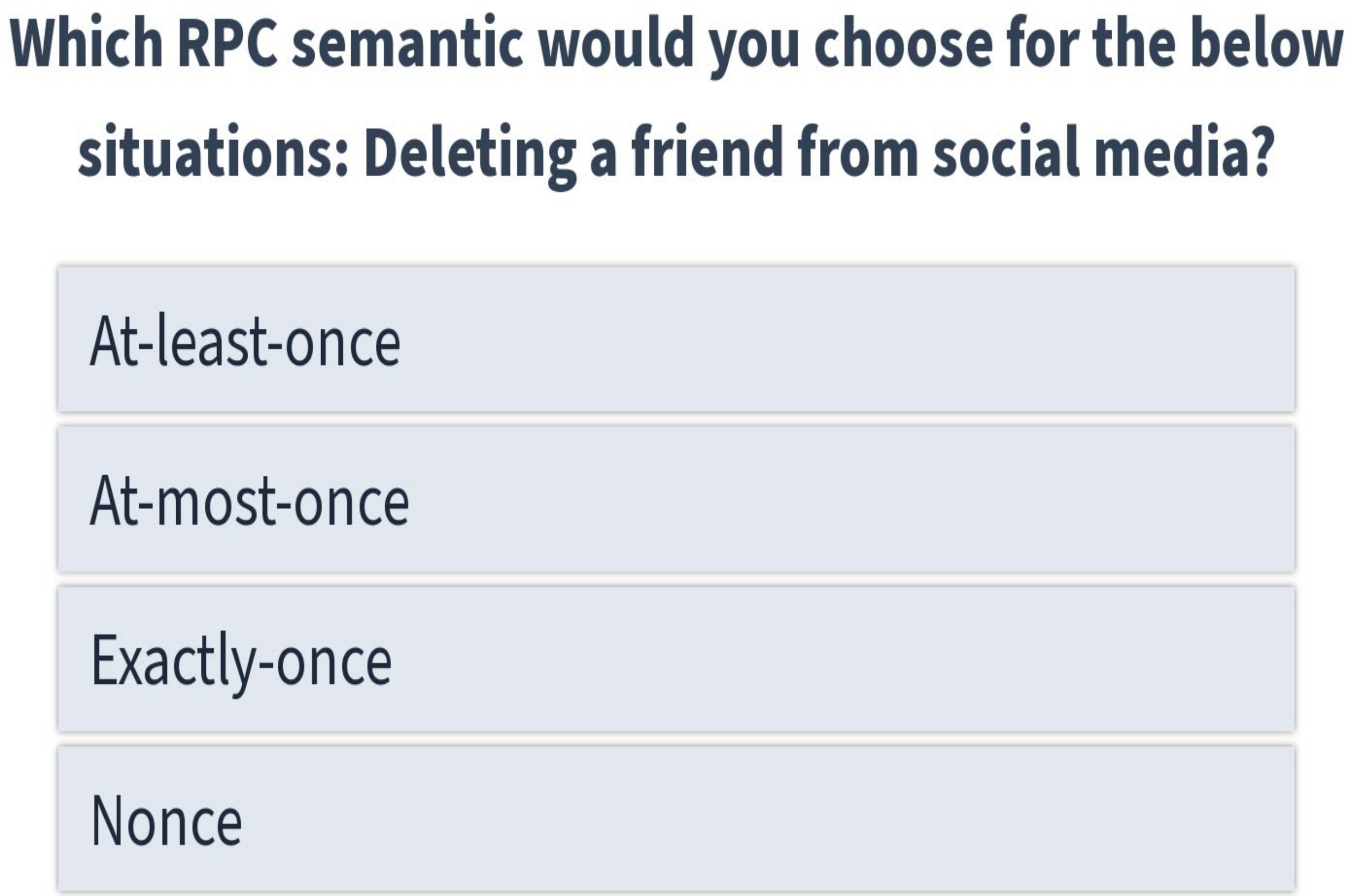
PollEv.com/hendgedawy084



Which RPC semantic would you choose for the below situations: Posting a message on messenger?

- At-least-once
- At-most-once
- Exactly-once
- Nonce

PollEv.com/hendgedawy084



Which RPC semantic would you choose for the below situations: Deleting a friend from social media?

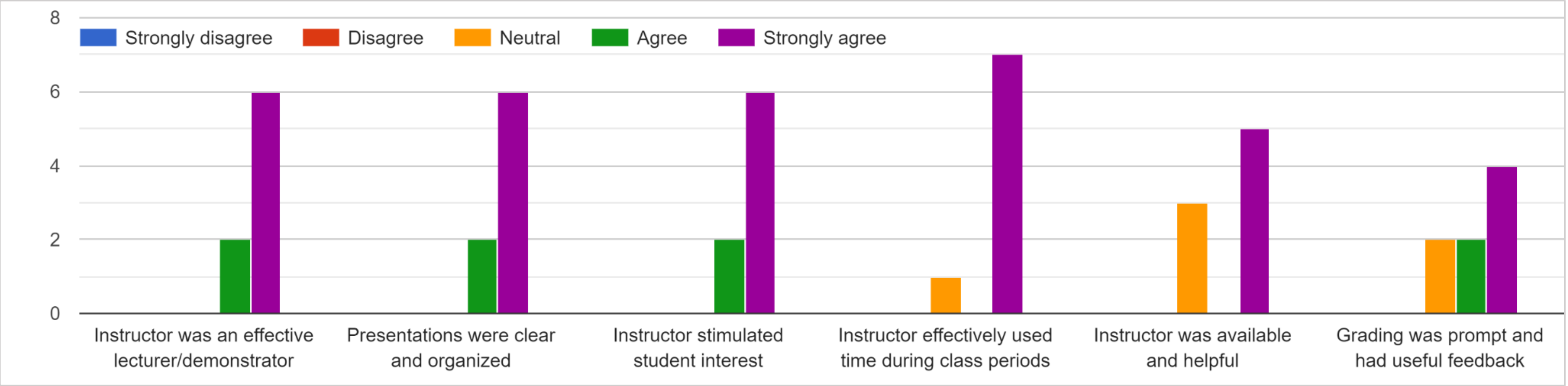
- At-least-once
- At-most-once
- Exactly-once
- Nonce

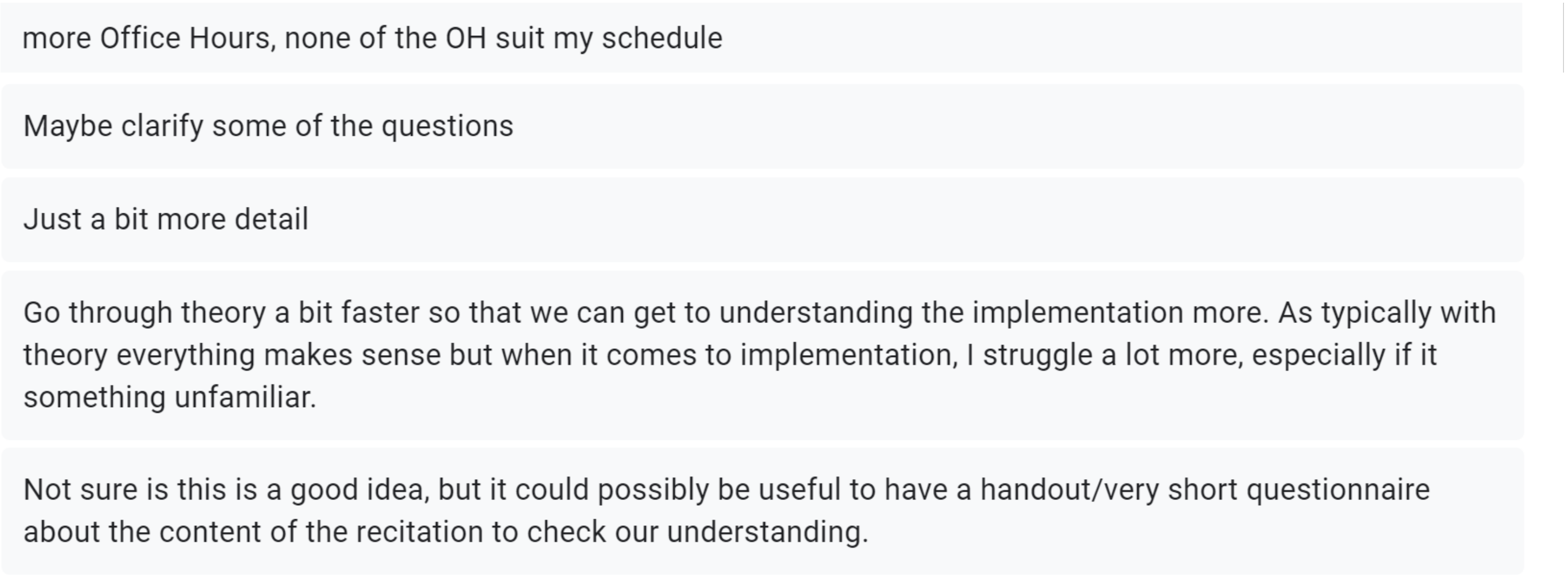
PollEv.com/hendgedawy084

Outline

- RPC Failure Sources
- **Handling Different Types or RPC Failures**
  - **Communication Failures**
  - **Client-Side Failures**
  - **Server-Side Failures**
- **Calculating RPC Time**
- **Exercises**
- **Feedback Reflection**
- Project 1 & PS2 Questions

Your Feedback & Suggestions





more Office Hours, none of the OH suit my schedule

Maybe clarify some of the questions

Just a bit more detail

Go through theory a bit faster so that we can get to understanding the implementation more. As typically with theory everything makes sense but when it comes to implementation, I struggle a lot more, especially if it something unfamiliar.

Not sure is this is a good idea, but it could possibly be useful to have a handout/very short questionnaire about the content of the recitation to check our understanding.

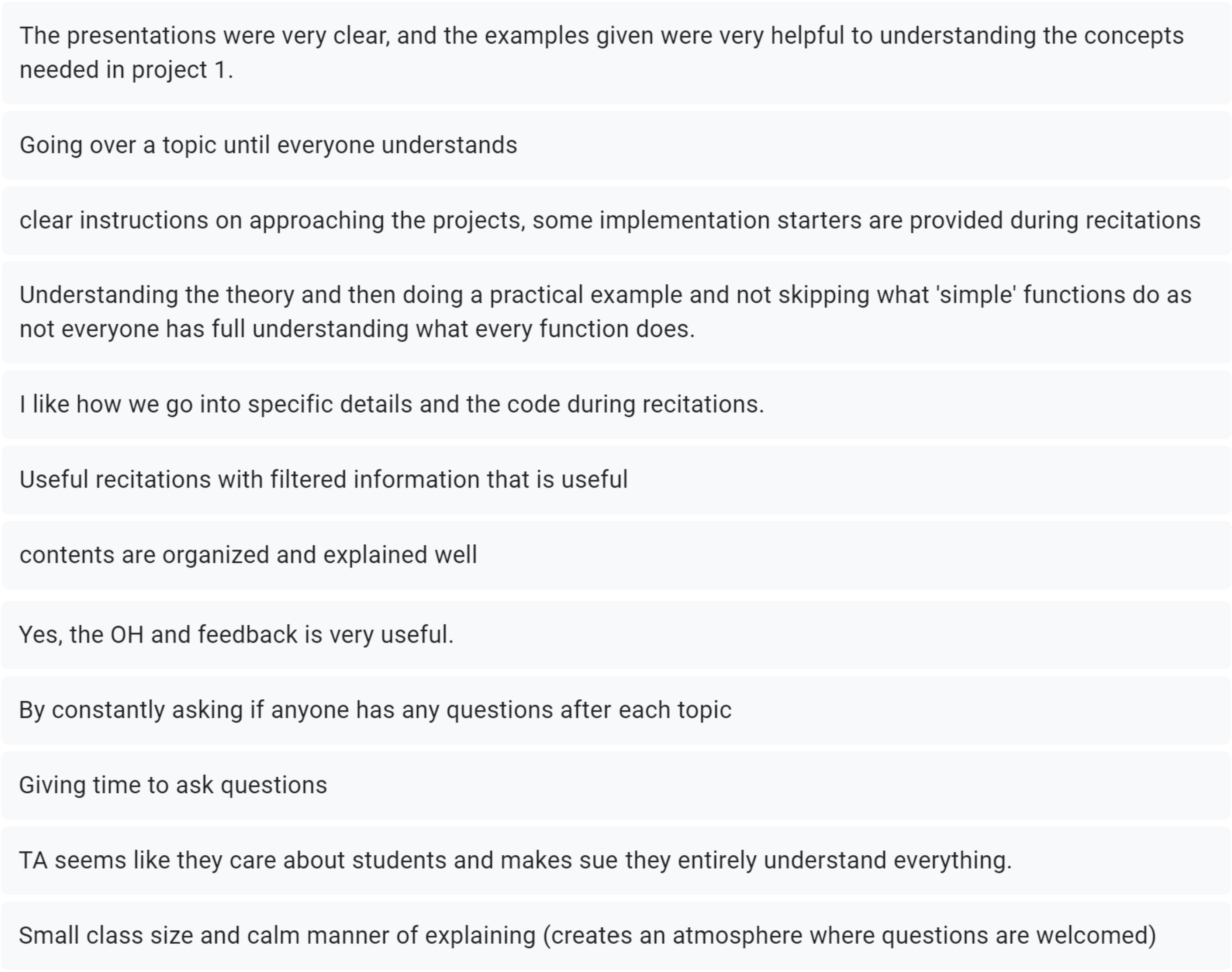
Reflection & Plan

- **Office Hours & Availability**
  - Added More Office Hours (Open Door): every day!
    - Sunday & Thursday : 10-12,
    - Tuesday 12-1
    - Monday & Wednesday 9-10
  - Besides, I am daily in the office 7:30-3:30
    - If door is slightly closed (it helps me focus):
      - Knock OR Send me on Google Chat
      - I will meet if time permits
    - Otherwise, we can always schedule a meeting
  - I am always available to answer questions on Piazza

Reflection & Plan

- **Suggestion: More on the implementation than theory**
  - Depending on the assignment we do some implementation (e.g. Multi-threaded TCP, RMI, …)
  - During recitations, please point out where more implementation information is needed, I can accommodate if possible
  - We must leave some assignment implementation details for you to figure out!
- **Suggestions: Questionnaire to check understanding of concepts at the end of each recitation**
  - Will make sure always to include some exercises
- **Grading & Feedback**
  - Will give more feedback on grades
  - Always happy to explain missed questions
  - Always happy to reassess

Things You Liked ☺



The presentations were very clear, and the examples given were very helpful to understanding the concepts needed in project 1.

Going over a topic until everyone understands

clear instructions on approaching the projects, some implementation starters are provided during recitations

Understanding the theory and then doing a practical example and not skipping what 'simple' functions do as not everyone has full understanding what every function does.

I like how we go into specific details and the code during recitations.

Useful recitations with filtered information that is useful

contents are organized and explained well

Yes, the OH and feedback is very useful.

By constantly asking if anyone has any questions after each topic

Giving time to ask questions

TA seems like they care about students and makes sue they entirely understand everything.

Small class size and calm manner of explaining (creates an atmosphere where questions are welcomed)

Outline

- RPC Failure Sources
- **Handling Different Types or RPC Failures**
  - **Communication Failures**
  - **Client-Side Failures**
  - **Server-Side Failures**
- **Calculating RPC Time**
- **Exercises**
- **Feedback Reflection**
- **Project 1 & PS2 Questions**