EP13: TCP or UDP for Zoom? Also...

ALEX XU
JUL 2, 2022

In this newsletter, we'll cover the following topics:

- TCP or UDP for Zoom?
- Why is Kafka fast? (YouTube videos)
- Distributed SQL
- Blocking vs non-blocking queue

TCP or UDP for Zoom?

Which protocol does Zoom use for video streaming, TCP or UDP?

Let's review the differences first.

🔹 The primary difference between TCP and UDP is that TCP is connection-based whereas UDP is connectionless.

Connection-based: It implies that all messages will arrive and arrive in the correct order.

Connectionless: It does not guarantee order or completeness.

- The second difference is that UDP is faster than TCP.

- UDP does not require ACK message back

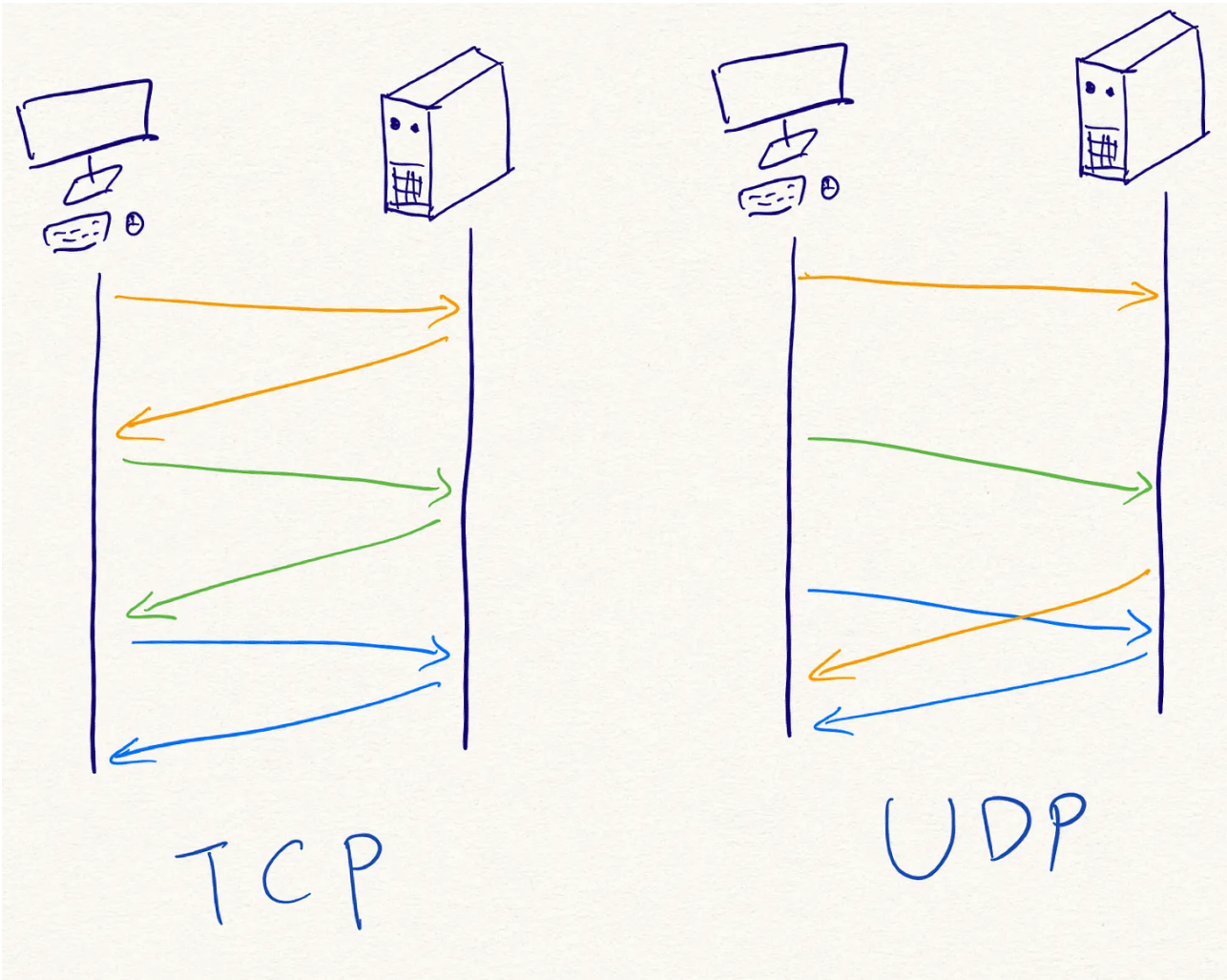
- UDP has no flow control

- No duplication verification at the receiving end

- Shorter header

UDP sacrifices correctness for speed. Users generally prefer smooth video calls and UDP is indeed the default choice for Zoom.

Over to you: the HTTP protocol used to be based on TCP, but the latest protocol HTTP/3 is based on UDP. Do you know why?



Why is Kafka fast? (YouTube videos)

Mainly talked about two techniques: Sequential I/O and Zero-copy

Distributed SQL

What is Distributed SQL? Why do we need it?

Google's Spanner popularized the term “distributed SQL” database in 2012. Distributed SQL databases automatically replicate data across multiple nodes and are strongly consistent. Paxos or Raft algorithms are commonly used to achieve consensus across multiple nodes.

Examples of Distributed SQL databases: Google Spanner, Amazon Aurora, CockroachDB, YugabyteDB, TiDB, etc.

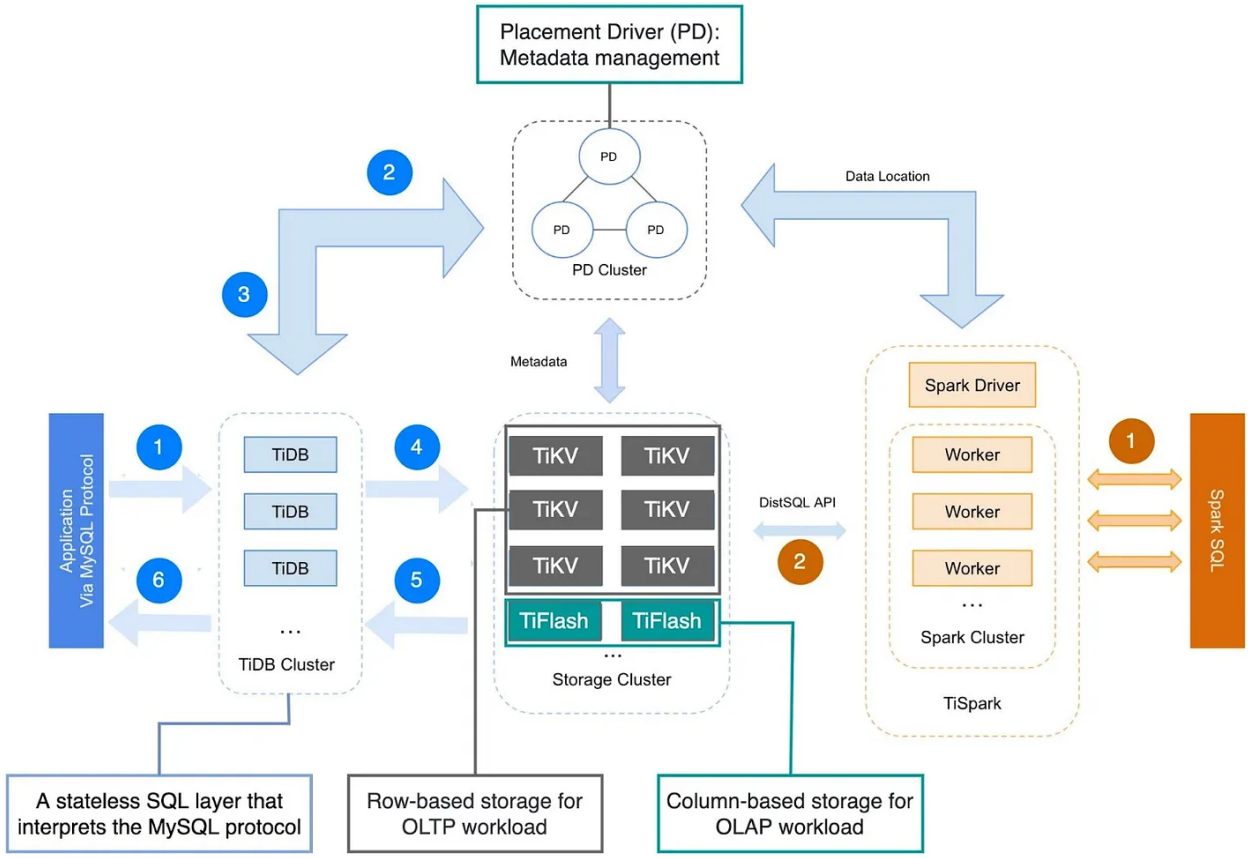
A few weeks ago, I met the CTO of TiDB ED Huang and a few other team members. We discussed how Distributed SQL databases evolved and how TiDB developed its own open-sourced database. After that conversation, I read more documentation/source code about it and I found it to be an interesting case study. I want to share my learning here.

Terms:

OLTP = Online transactional processing

OLAP = Online analytical processing

HTAP = Hybrid transaction/analytical processing



- The life of an **OLTP** query (marked with **blue** sequence numbers):

Step 1. A client sends a MySQL query and the query is interpreted by TiDB, a stateless SQL layer that interprets the MySQL protocol.

Step 2: TiDB requests data mapping/placement information from the placement driver (PD).

Step 3: PD responds with data mapping/ placement instructions & timestamp.

Step 4: TiDB executes queries on TiKV servers (row-based storage).

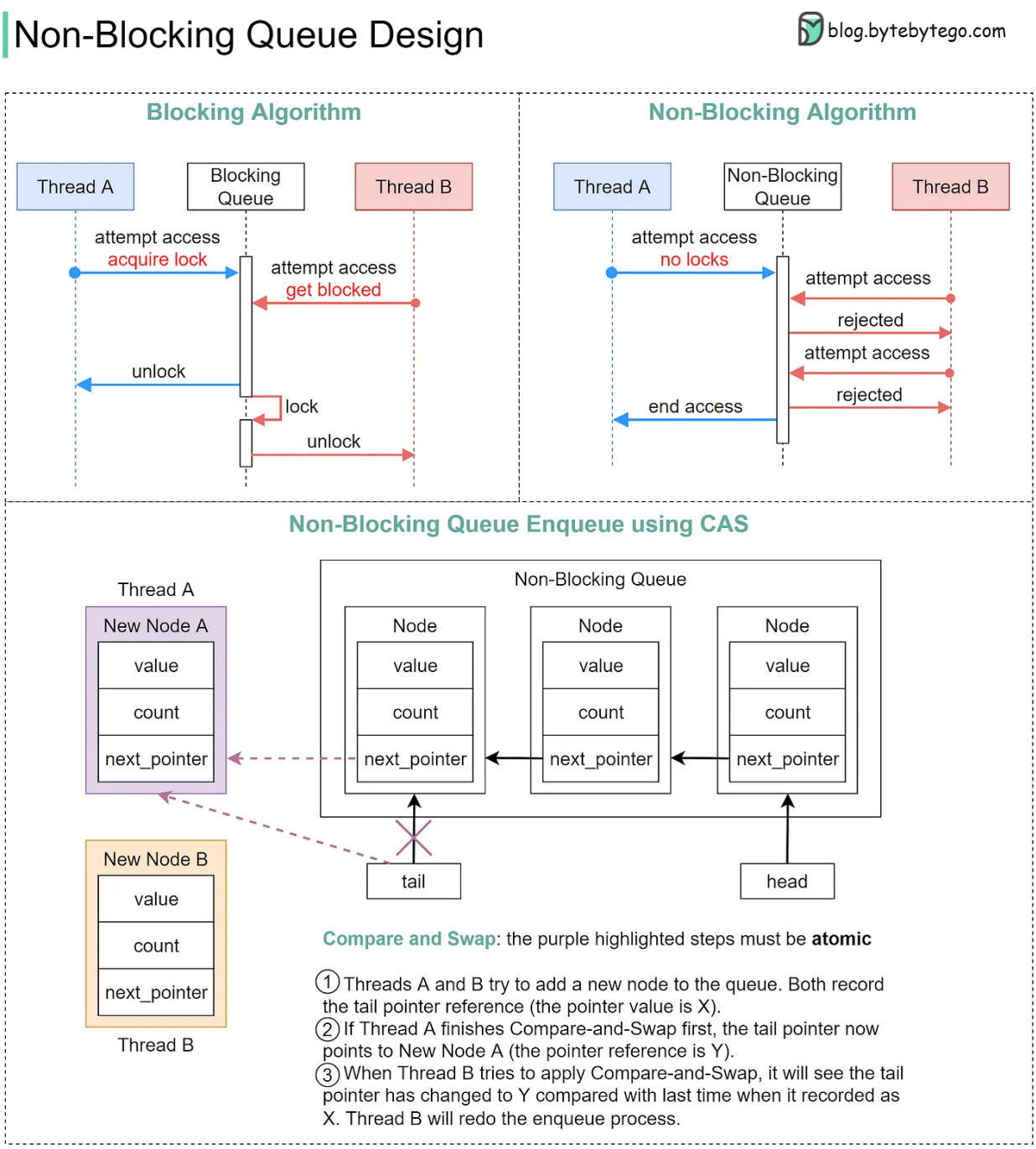
Step 5, 6: Query results are sent back to the client.

- The life of a complex **OLAP** query: marked with **yellow** sequence numbers.

Over to you: do you think the terms OLTP and OLAP have become obsolete or are they still very relevant? When should we use distributed SQL databases vs traditional relational databases?

Blocking vs non-blocking queue

How do we implement a **non-blocking** queue? What are the differences between blocking and non-blocking algorithms?



The terms we use when discussing blocking and non-blocking algorithms can be confusing, so let's start by reviewing the terminology in the concurrency area with a diagram.

- blocking

The blocking algorithm uses locks. Thread A acquires the lock first, and Thread B might wait for arbitrary lengthy periods if Thread A gets suspended while holding the lock. This algorithm may cause Thread B to starve.

- non-blocking:

The non-blocking algorithm allows Thread A to access the queue, but Thread A must complete a task in a certain number of steps. Other threads like Thread B may still starve due to the rejections.

This is the main **difference** between blocking and non-blocking algorithms: The blocking algorithm blocks Thread B until the lock is released. A non-blocking algorithm notifies Thread B that access is rejected.

- starvation-free:

Thread Starvation means a thread cannot acquire access to certain shared resources and cannot proceed. Starvation-free means this situation does not occur.

- wait-free:

All threads can complete the tasks within a finite number of steps.

Wait-free = Non-Blocking + Starvation-free

➡️ Non-Blocking Queue **implementation**

We can use Compare and Swap (CAS) to implement a non-blocking queue. The diagram below illustrates the algorithm.

➡️ **Benefits**

1. No thread suspension. Thread B can get a response immediately and then decide what to do next. In this way, the thread latency is greatly reduced.

2. No deadlocks. Threads A and B do not wait for the lock to release, meaning that there is no possibility of a deadlock occurring.

Other things we made:

Our bestselling book “System Design Interview - An Insider’s Guide” is available in both paperback and digital format.

Paperback edition: https://geni.us/XxCd

Digital edition: https://bit.ly/3lg41jK

New System Design YouTube channel: https://bit.ly/ByteByteGoVideos

Thanks for reading ByteByteGo Newsletter!
Subscribe for free to receive new posts and
support my work.

93 Likes

2 Comments

Write a comment...

Abhisar Raj Feb 12

You didn't explain which protocol TPC or UDP is used by zoom ?

LIKE REPLY SHARE

1 reply

1 more comment...