EP54: Cache Systems Every Developer Should Know

ALEX XU
APR 8, 2023

This week's system design refresher:

- Cache Systems Every Developer Should Know (Youtube video)
- Serverless DB
- TCP vs. UDP
- Batch v.s. Stream Processing
- Job openings

Become a better tech leader in 10 mins a week (Sponsored)

Refactoring is a weekly newsletter about writing great software and working well together. Its articles are read every week by 30,000+ subscribers, including engineers

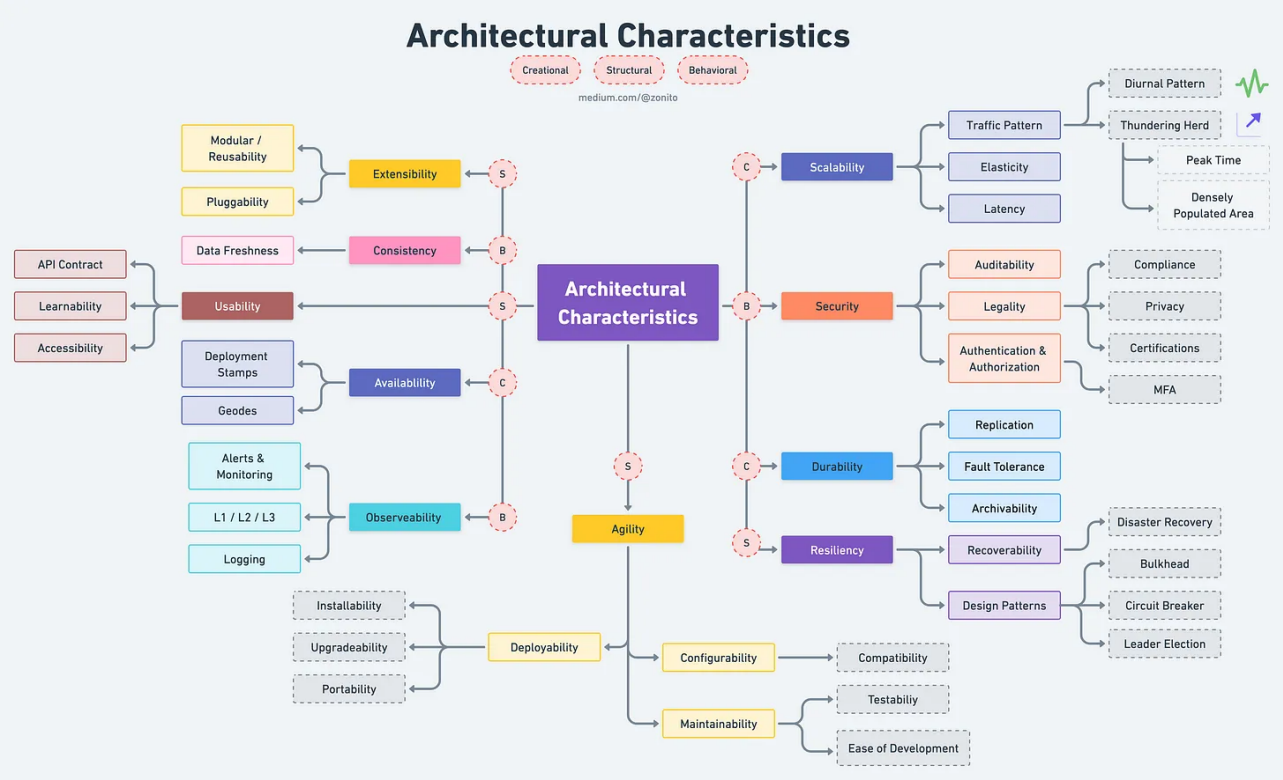
and managers at companies like Google, Meta, and Amazon.

Subscribe to:

- Receive a **new essay** every Thursday with practical advice about your work.
- Access a **curated library** of 130+ original essays and 250+ resources.
- Join a **private community** of tech leaders, founders, and engineers.
- Get **$50K+ discounts** on popular dev tools.

Cache Systems Every Developer Should Know

Top 10 Architecture Characteristics / Non-Functional Requirements with Cheatsheet



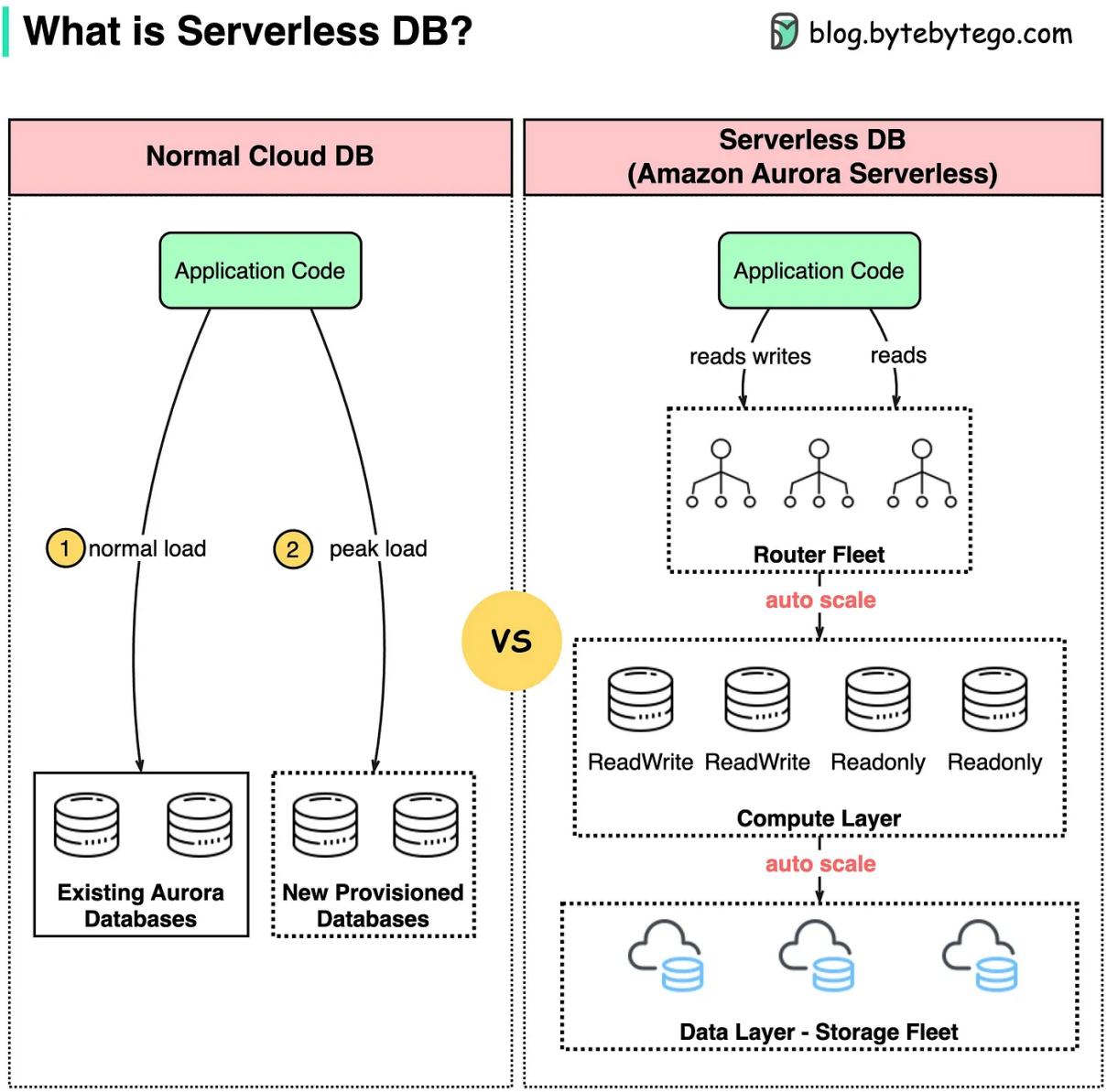
Did we miss anything? If yes, Please help to enrich this article by sharing your thoughts in the comments.

Written by Love Sharma, our guest author. We're constantly seeking valuable content, so if you'd like to contribute to our platform or have any previously published content you'd like us to share, please feel free to drop us a message.

What is Serverless DB

Are serverless databases the future? How do serverless databases differ from traditional cloud databases?

Amazon Aurora Serverless, depicted in the diagram below, is a configuration that is auto-scaling and available on-demand for Amazon Aurora.

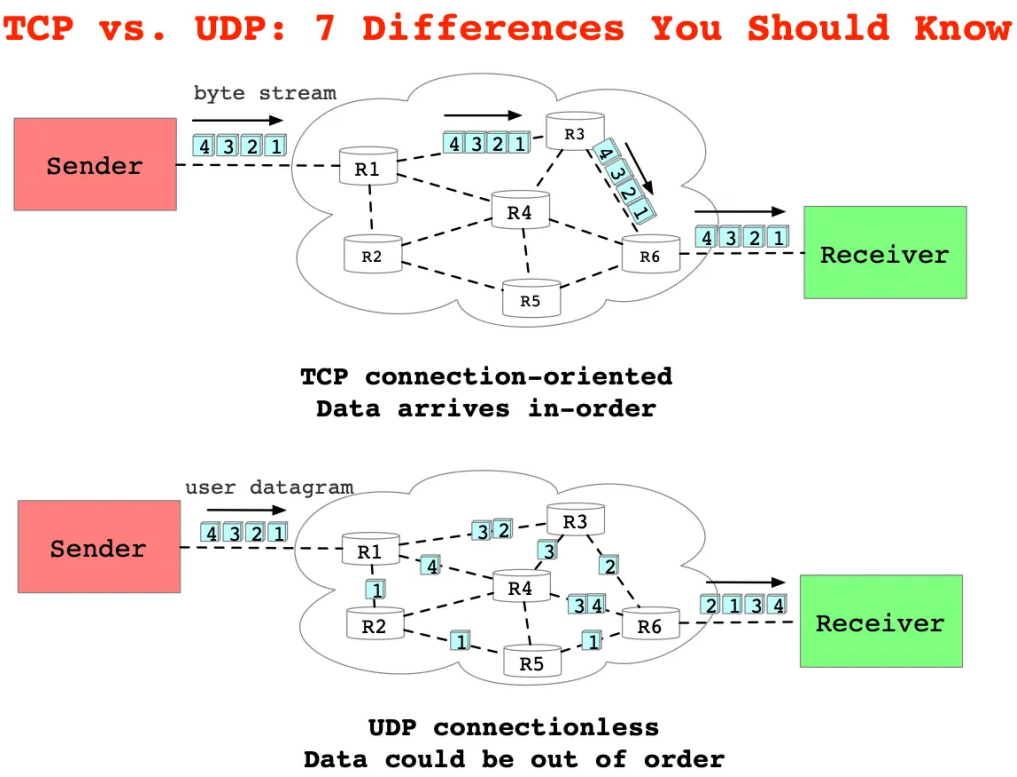


- Aurora Serverless has the ability to scale capacity automatically up or down as per business requirements. For example, an eCommerce website preparing for a major promotion can scale the load to multiple databases within a few milliseconds. In comparison to regular cloud databases, which necessitate the provision and administration of database instances, Aurora Serverless can automatically start up and shut down.
- By decoupling the compute layer from the data storage layer, Aurora Serverless is able to charge fees in a more precise manner. Additionally, Aurora Serverless can be a combination of provisioned and serverless instances, enabling existing provisioned databases to become a part of the serverless pool.

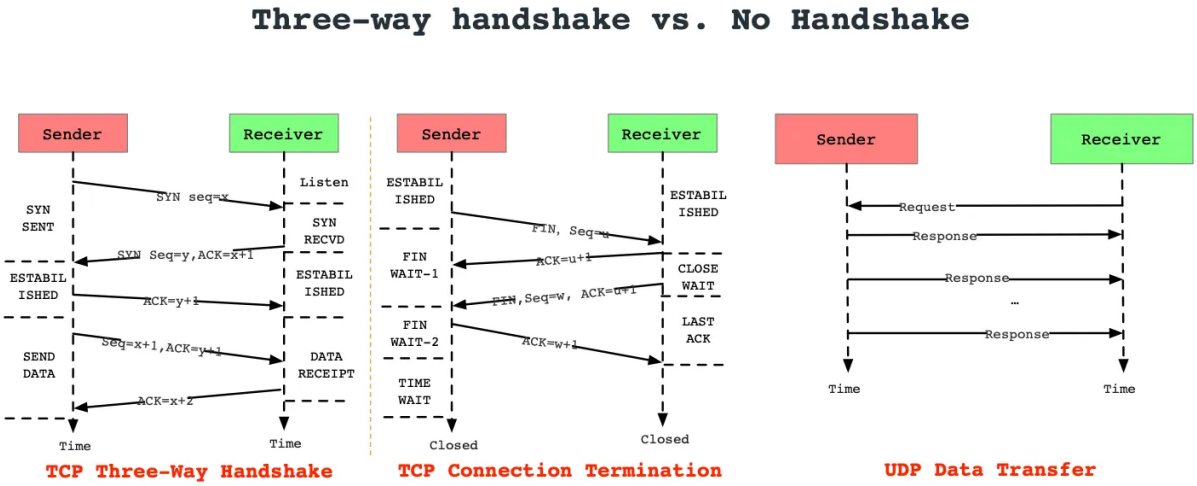
Over to you: Have you used a serverless DB? Does it save cost?

TCP vs. UDP: 7 Differences You Should Know

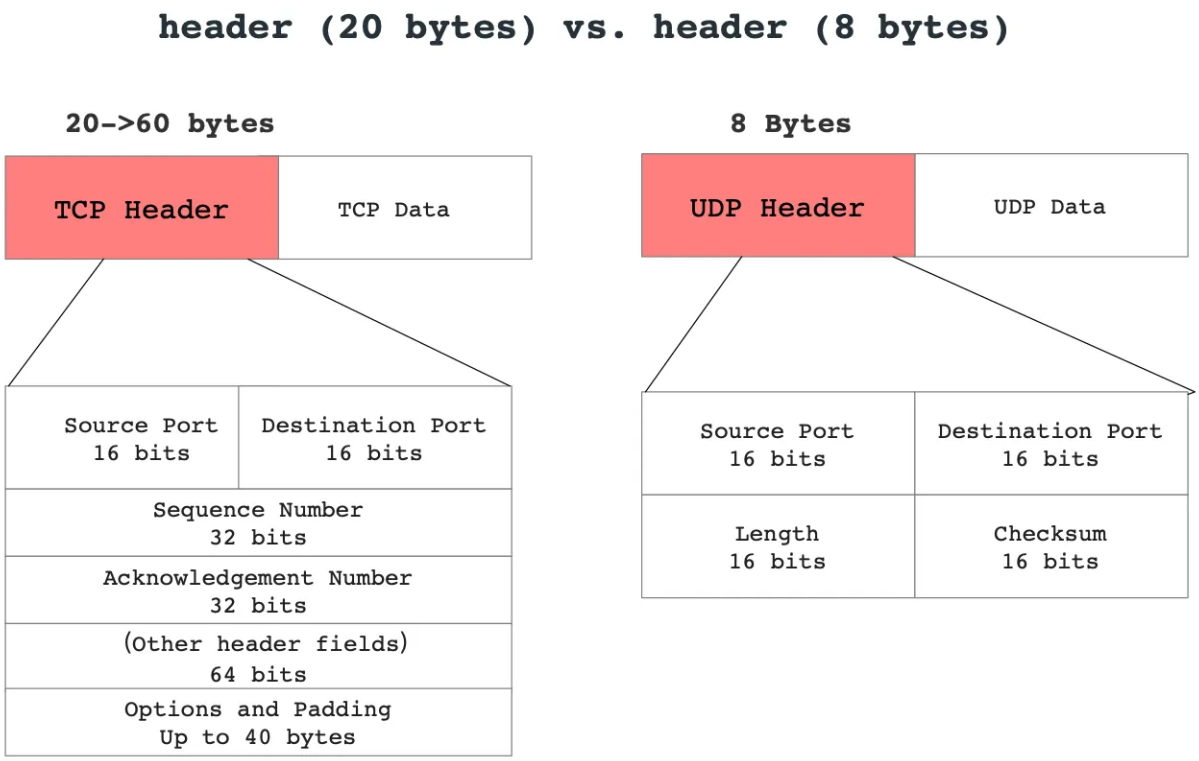
1. Connection-oriented vs. connectionless



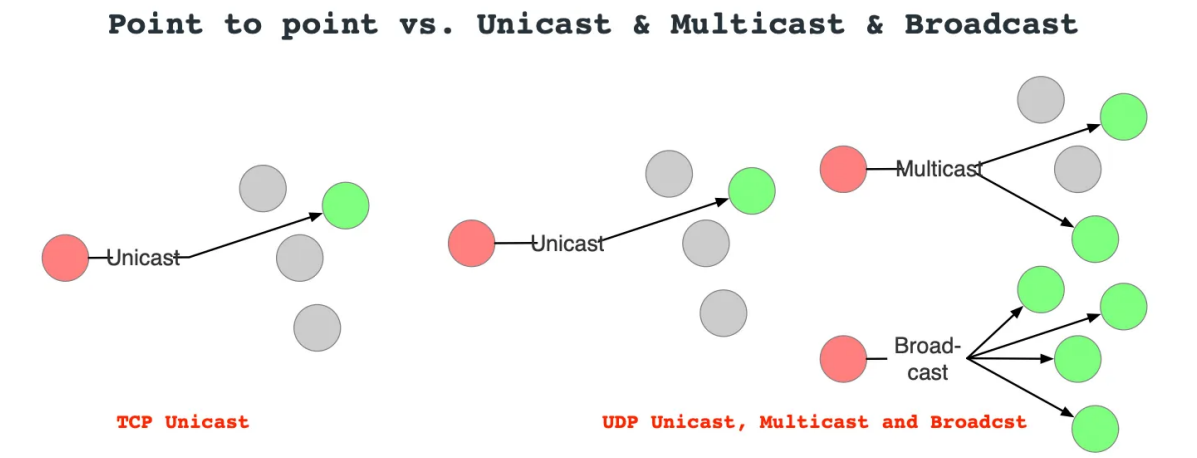
2. Three-way handshake vs. No handshake



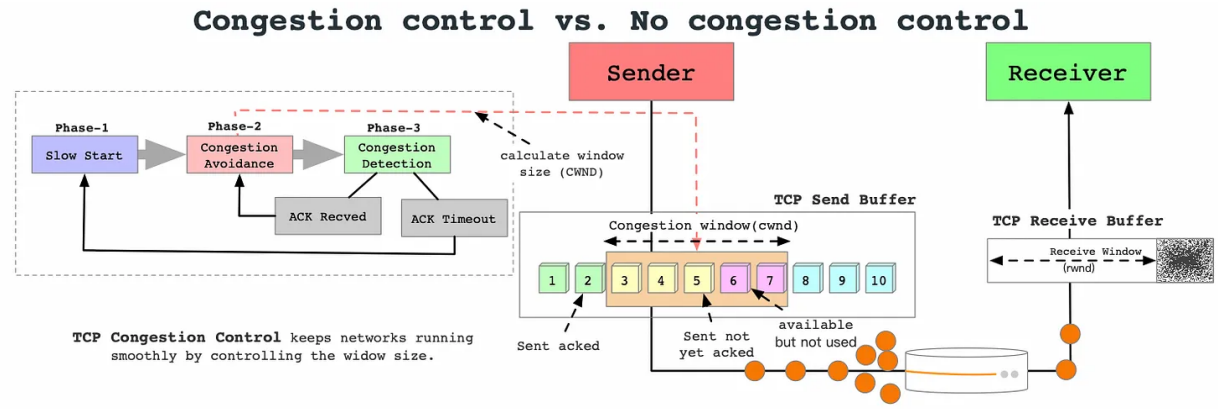
3. Header (20 bytes) vs. (8 bytes)



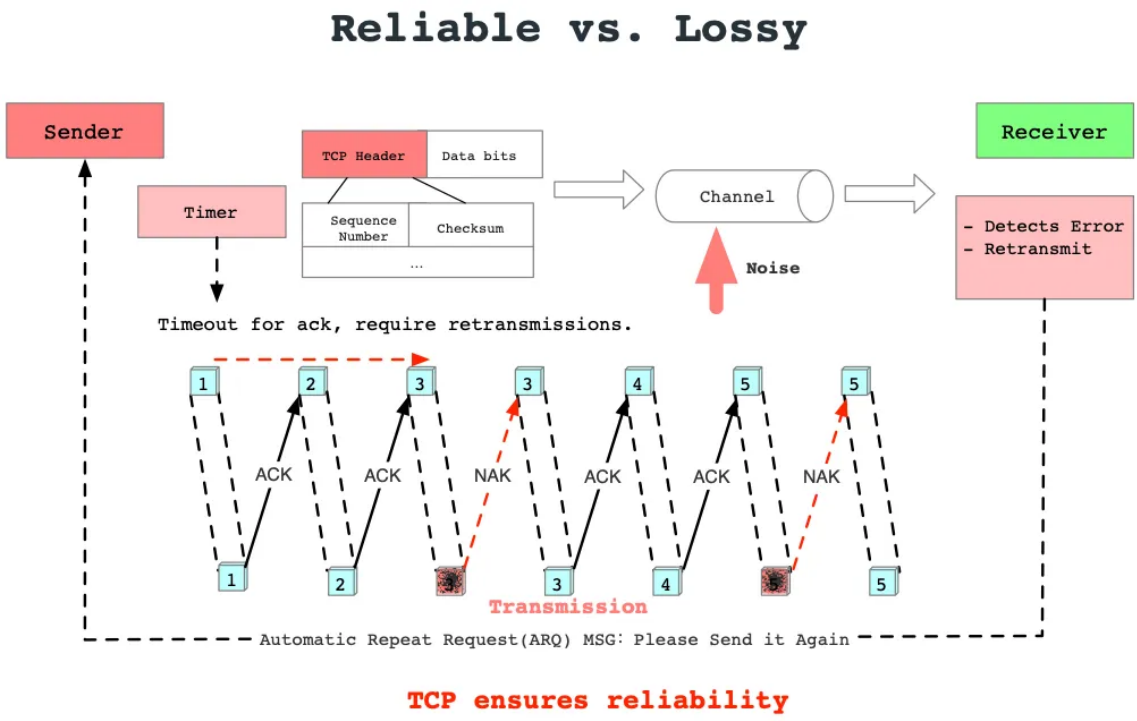
4. Point-to-point vs. Unicast & Multicast & Broadcast



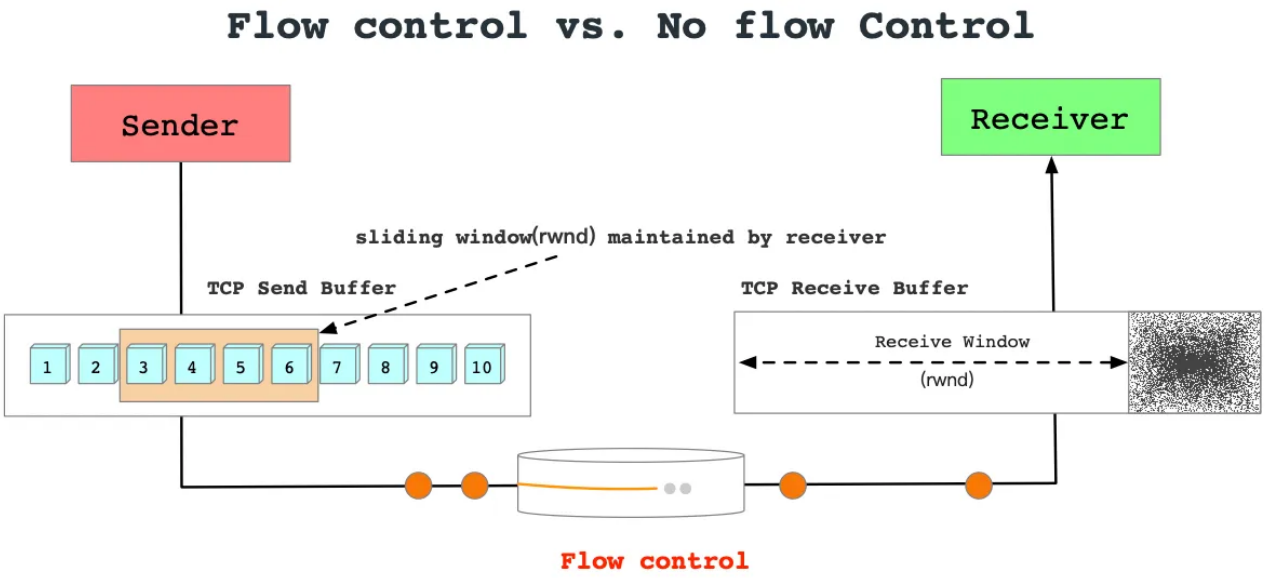
5. Congestion control vs. no congestion control



6. Reliable vs. lossy

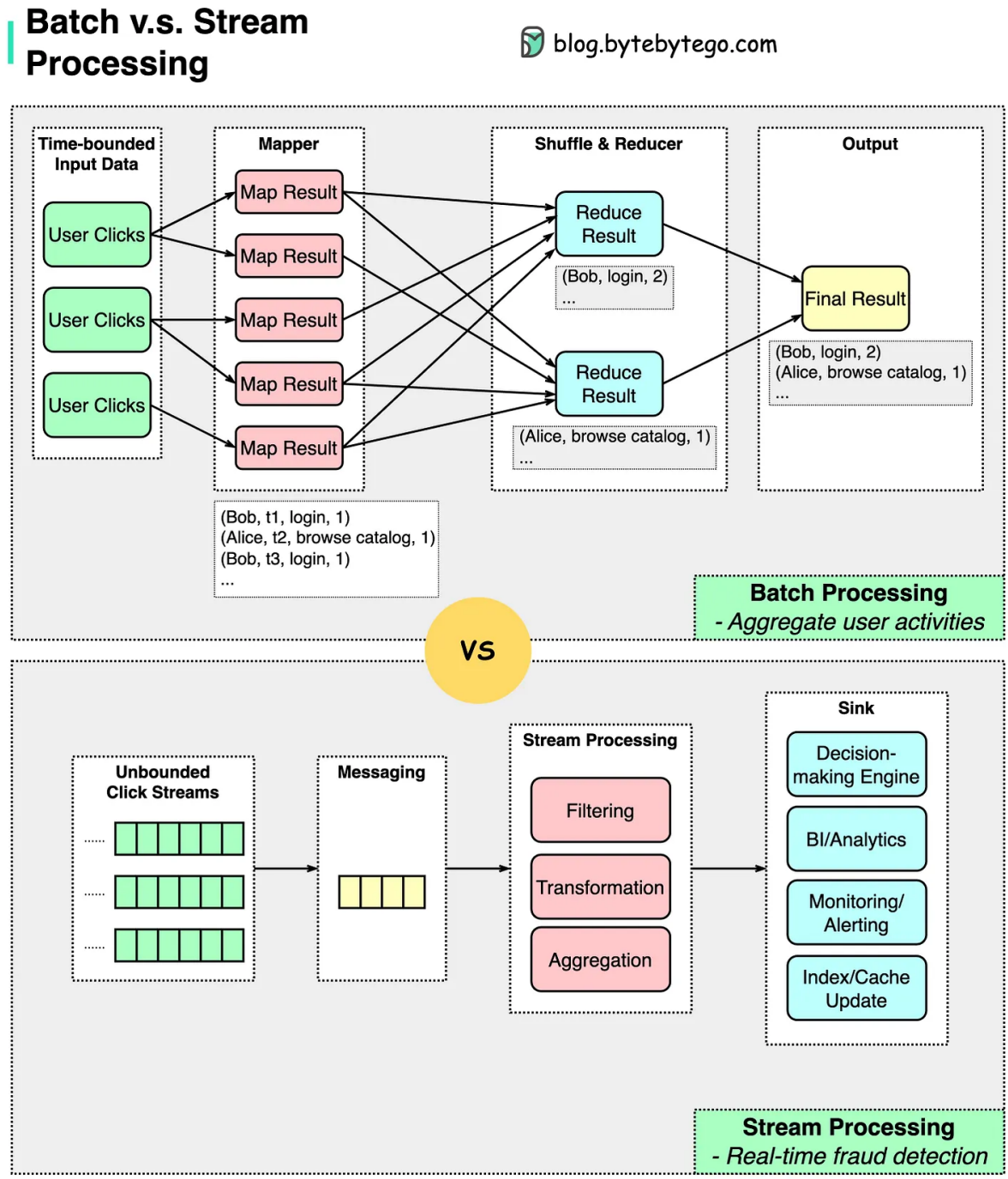


7. Flow control vs. no flow control



Batch v.s. Stream Processing

- Batch Processing: We aggregate user click activities at end of the day.
- Stream Processing: We detect potential frauds with the user click streams in real-time.



Both processing models are used in big data processing. The major differences are:

1. Input

   Batch processing works on time-bounded data, which means there is an end to the input data.

   Stream processing works on data streams, which doesn't have a boundary.

2. Timelineness

   Batch processing is used in scenarios where the data doesn't need to be

processed in real-time.
Stream processing can produce processing results as the data is generated.

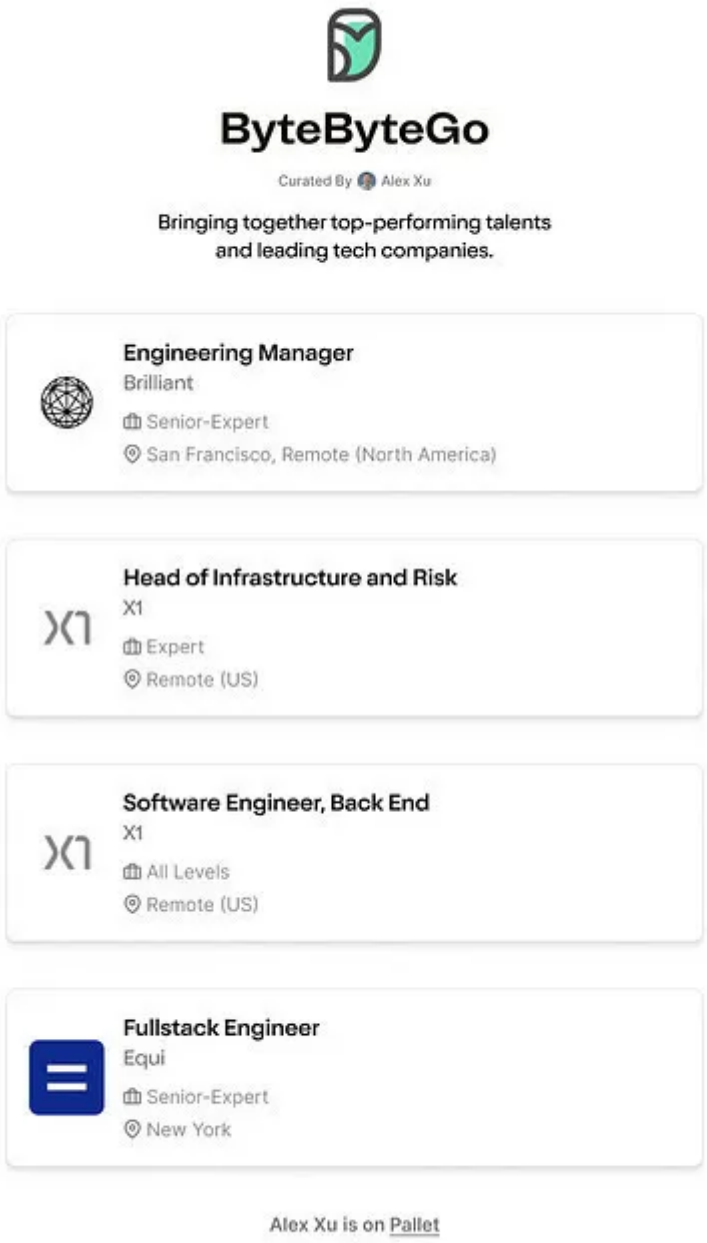
3. Output
   Batch processing usually generates one-off results, for example, reports.
   Stream processing's outputs can pipe into fraud decision-making engines, monitoring tools, analytics tools, or index/cache updaters.
4. Fault tolerance
   Batch processing tolerates faults better as the batch can be replayed on a fixed set of input data.
   Stream processing is more challenging as the input data keeps flowing in. There are some approaches to solve this:

   a) Microbatching which splits the data stream into smaller blocks (used in Spark);
   b) Checkpoint which generates a mark every few seconds to roll back to (used in Flink).

Over to you: Have you worked on stream processing systems?

Join the ByteByteGo Talent Collective

If you're looking for a new gig, join the collective for customized job offerings from selected companies. Public or anonymous options are available. Leave anytime.



If you're **hiring**, join the ByteByteGo Talent Collective to start getting bi-monthly drops of world-class hand-curated engineers who are open to new opportunities.

Featured job openings

X1 Card: Engineering Leader - Card Platform (United States, remote)

143 Likes · 4 Restacks

4 Comments

Write a comment...

Jakub Jablonski Writes Jakub's Substack Apr 10

Regarding architecture characteristics, I have started collecting a library of those for myself, see https://jakubjablonski2-tomtom.github.io/architecture-characteristics/.

LIKE (2) REPLY SHARE

John Stanhope Writes John's Substack Apr 15

I would also consider performance (resource efficiency) and latency as an architectural characteristics.

LIKE REPLY SHARE

2 more comments...