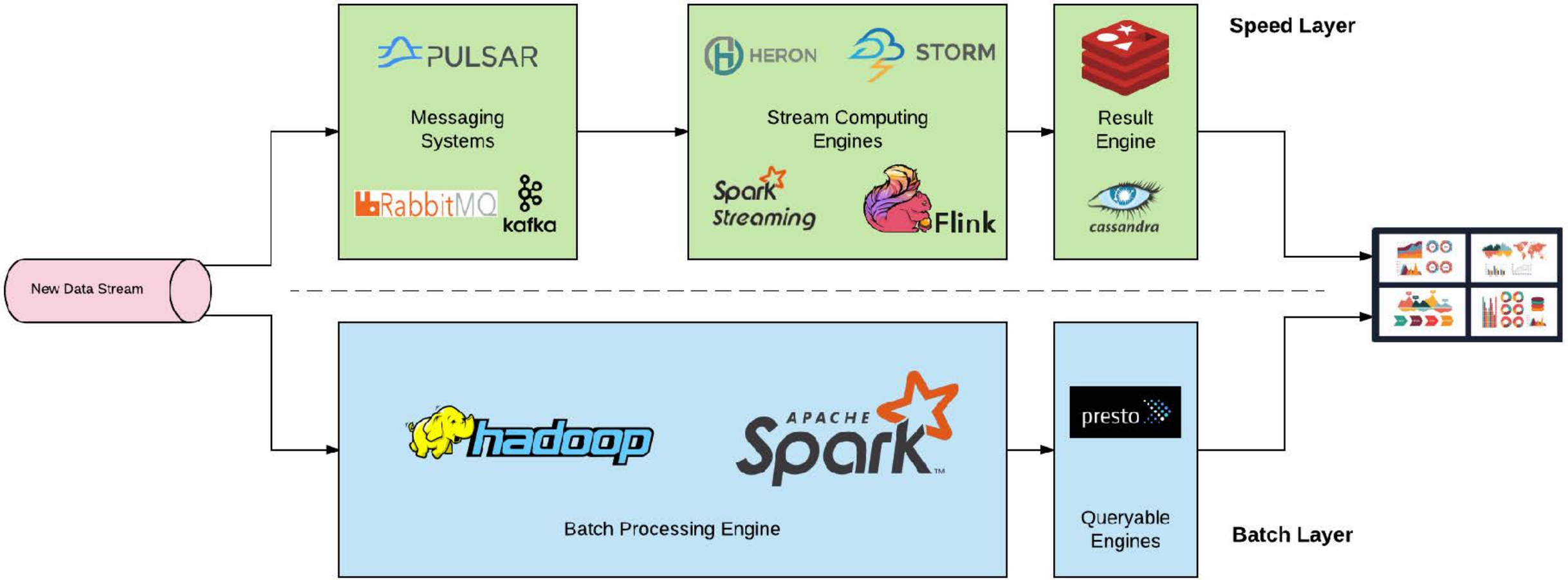
Apache Pulsar

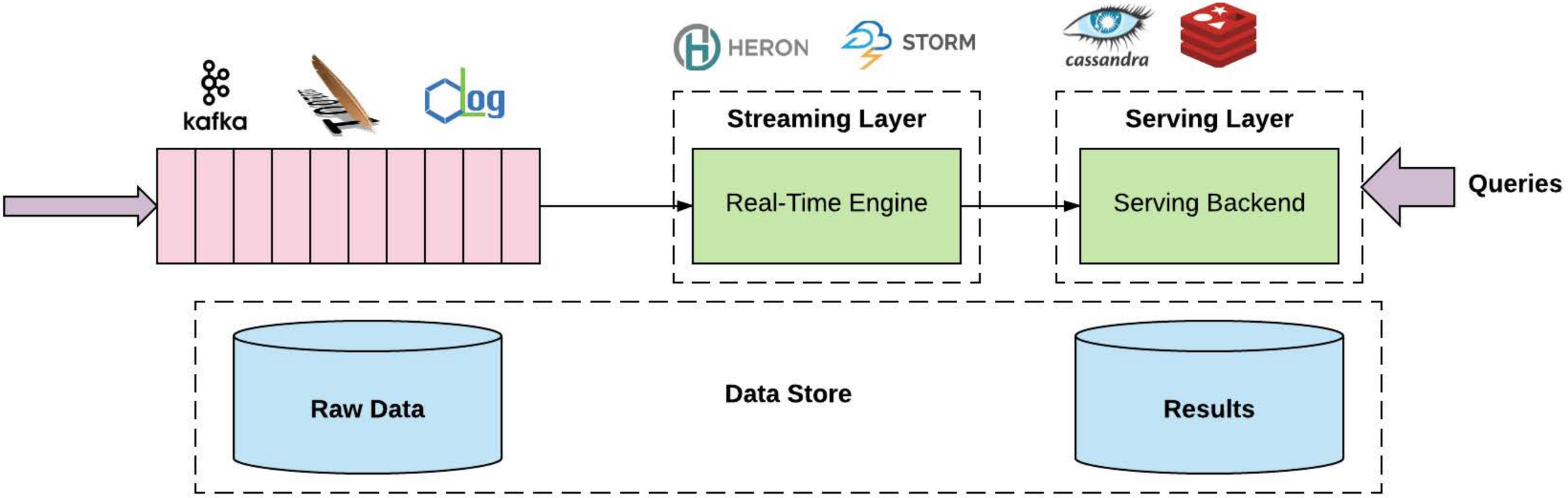
实时数据处理中
消息，计算和存储的统一

演讲者 / streamlio 翟佳

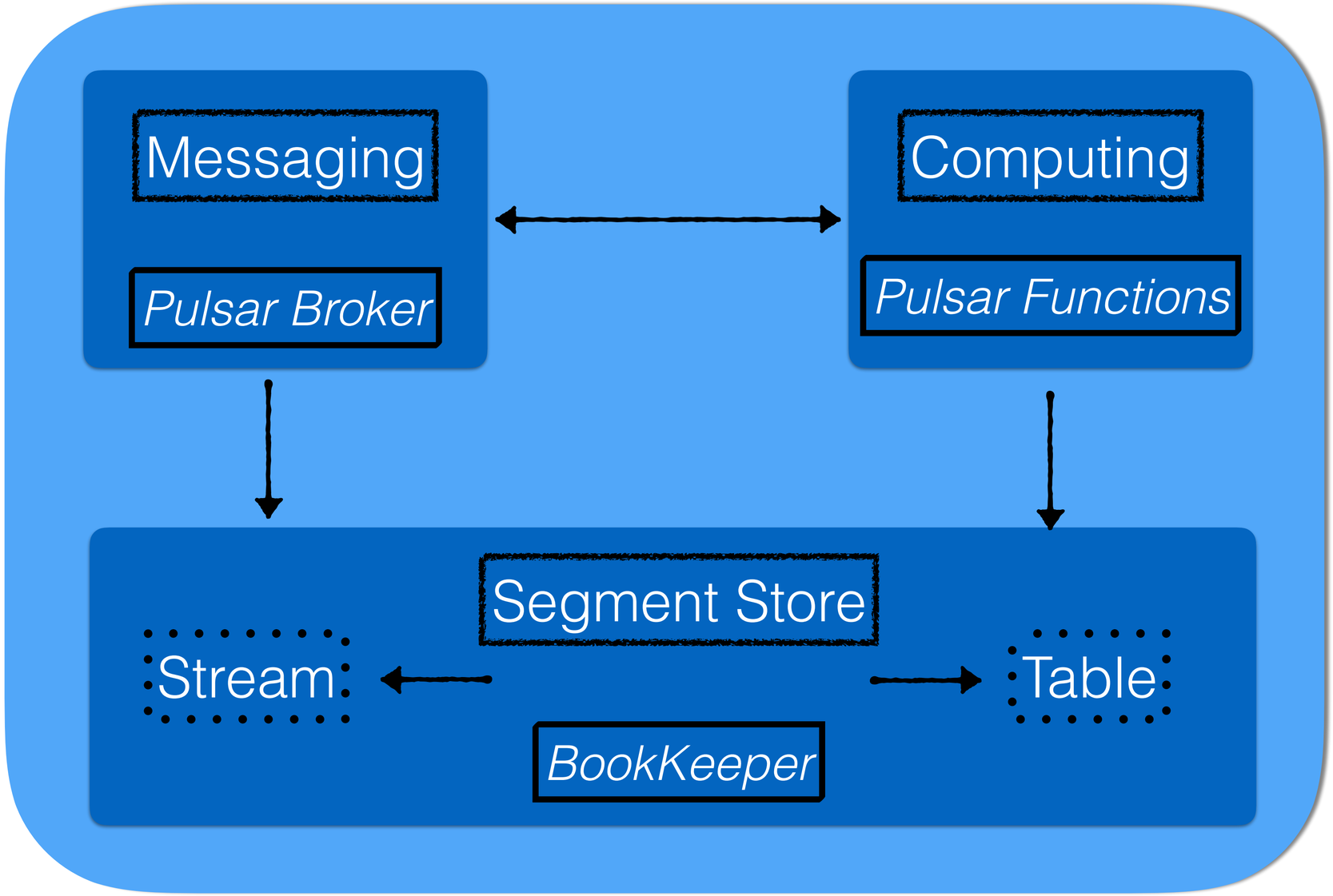
What's the state of the art



What's the state of the art



Apache Pulsar – Unify



Why Apache Pulsar?

Durability
Data replicated and synced to disk

Ordering
Guaranteed ordering

Delivery Guarantees
At least once, at most once and effectively once

Geo-replication
Out of box support for geographically distributed applications

Multi-tenancy
A single cluster can support many tenants and use cases

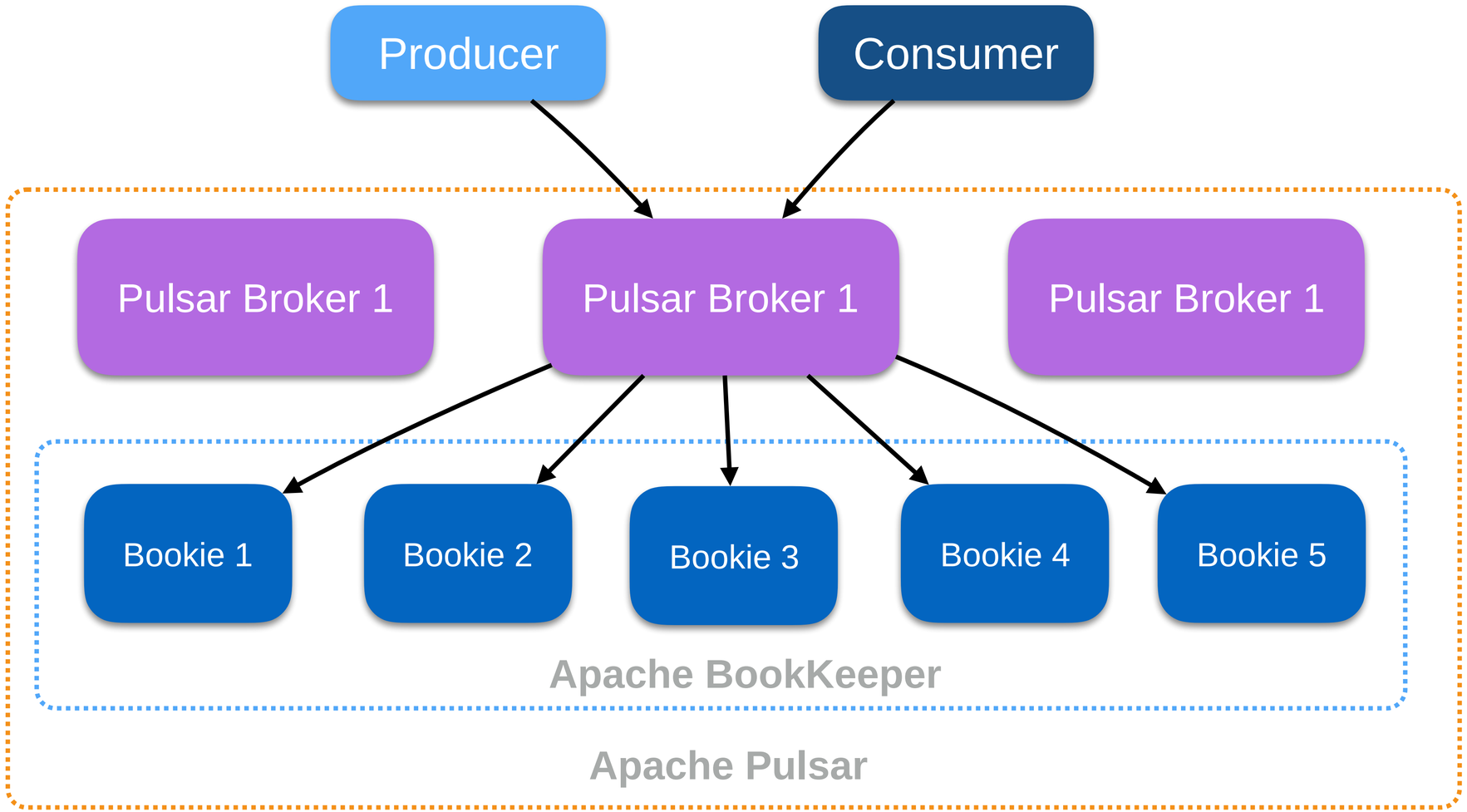
Low Latency
Low publish latency of 5ms at 99pct

Unified messaging model
Support both Streaming and Queuing in a single model

High throughput
Can reach 1.8 M messages/s in a single partition

Highly scalable
Can support millions of topics

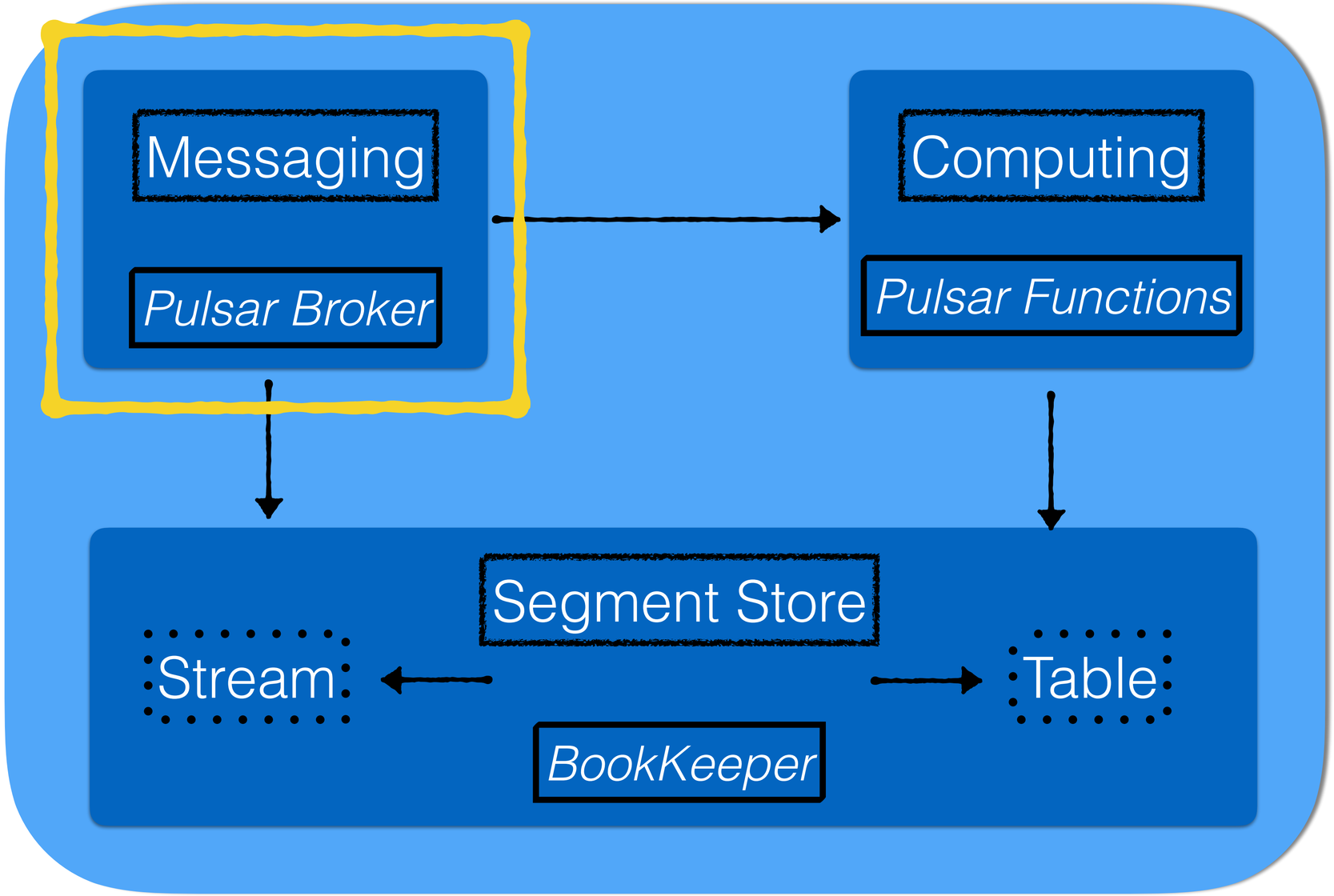
Pulsar Architecture



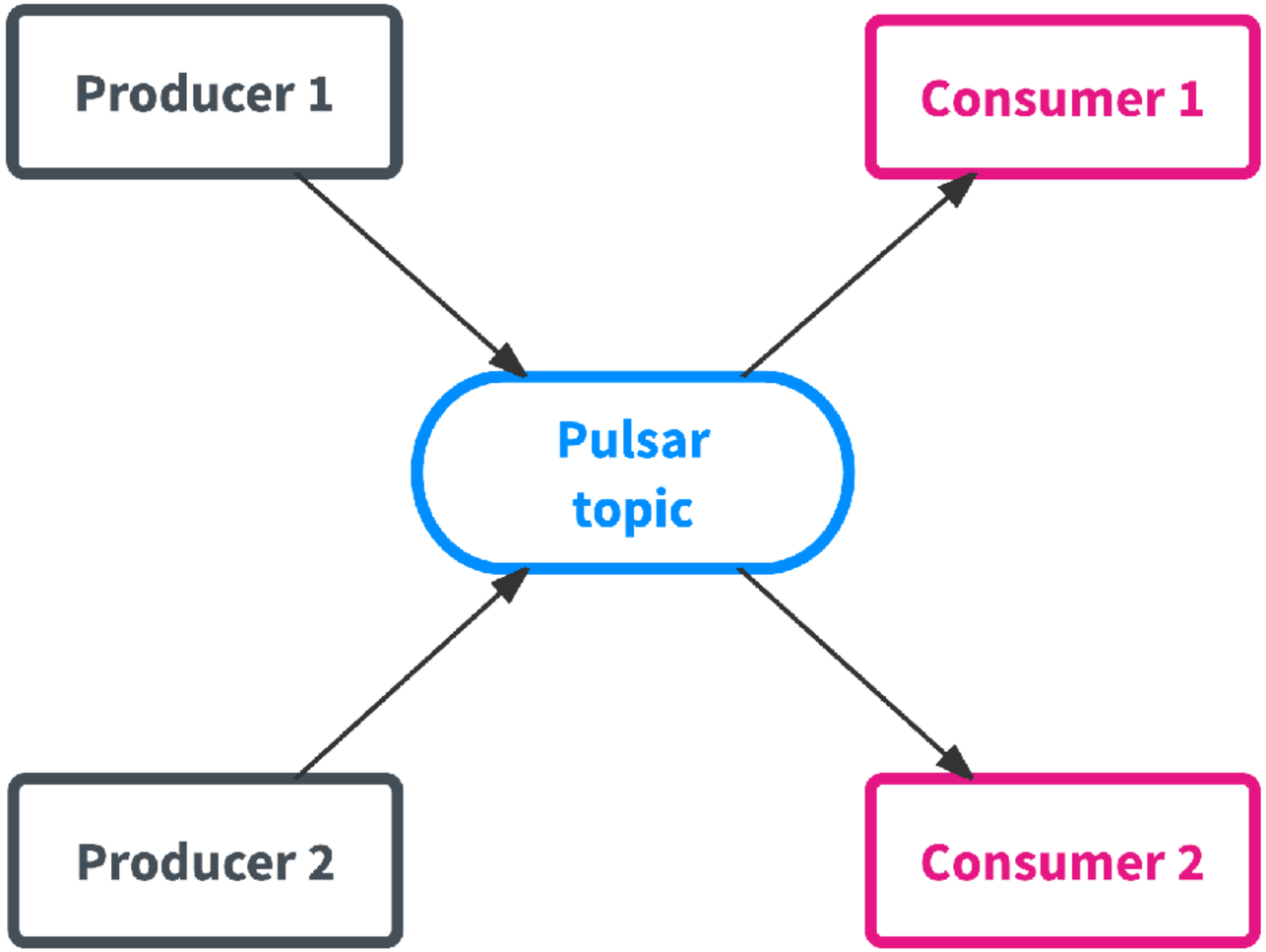
Separate layers between brokers bookies

- Broker and bookies can be added independently
- Traffic can be shifted very quickly across brokers
- New bookies will ramp up on traffic quickly

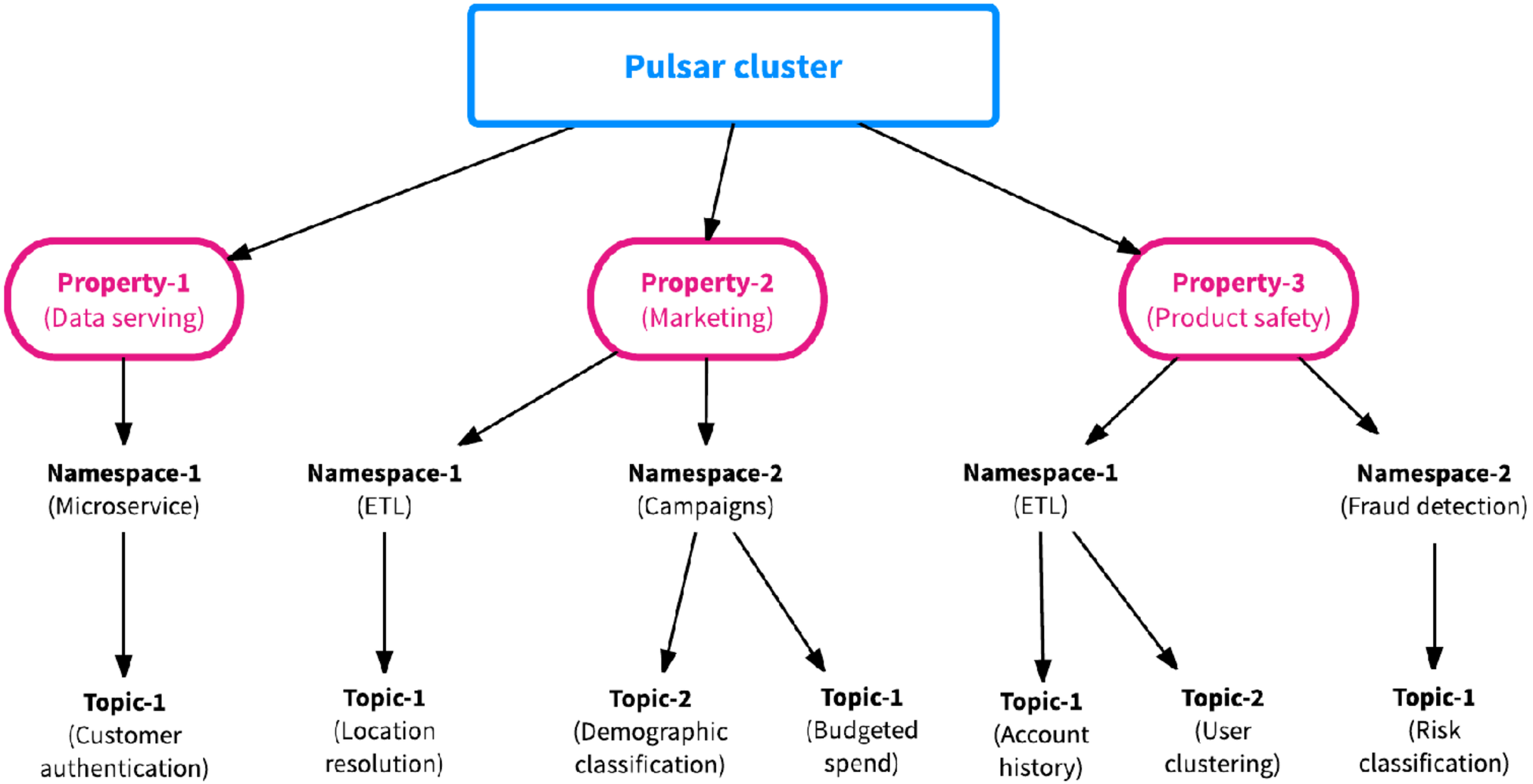
Messaging


Messaging
Pulsar Broker
Computing
Pulsar Functions
Segment Store
Stream
Table
BookKeeper

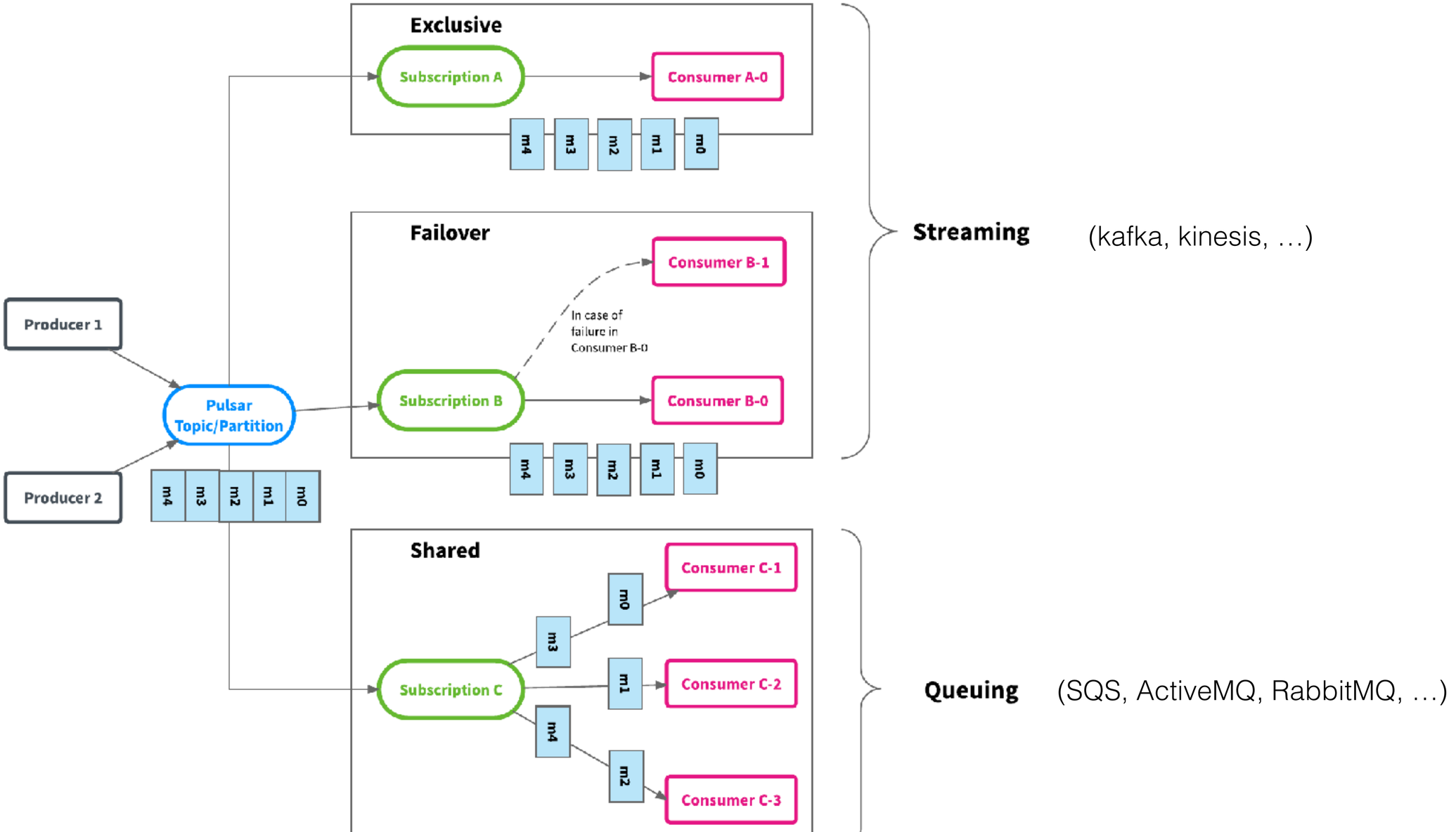
Messaging - Concepts



Messaging - Namespace



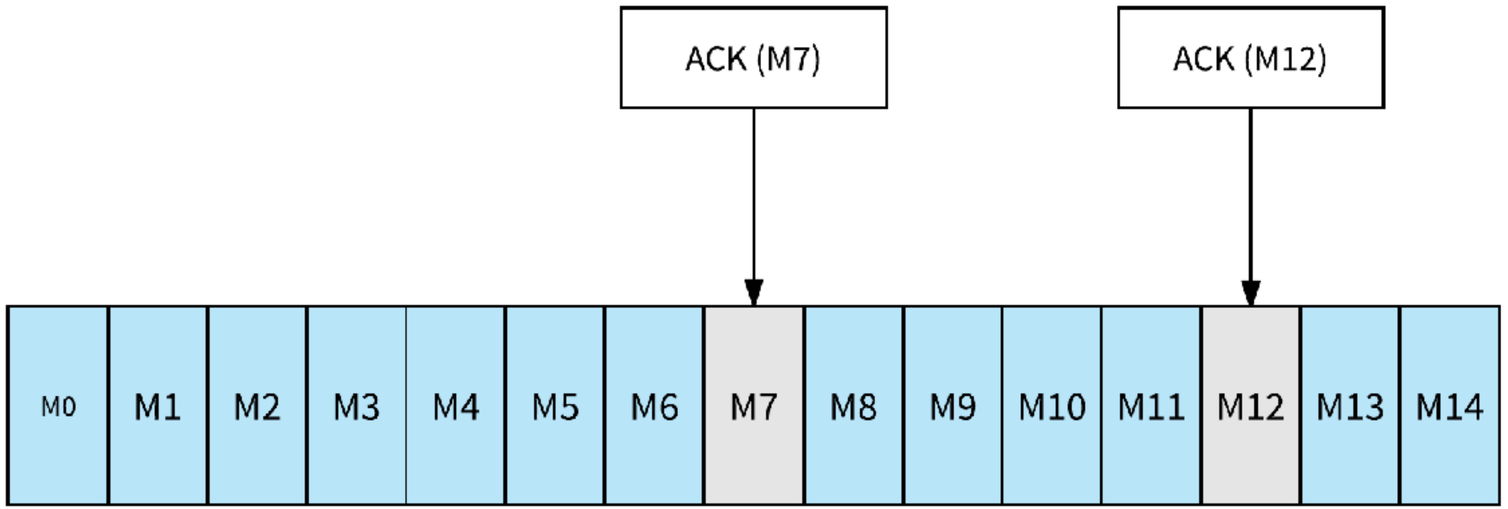
Messaging - Queuing & Streaming



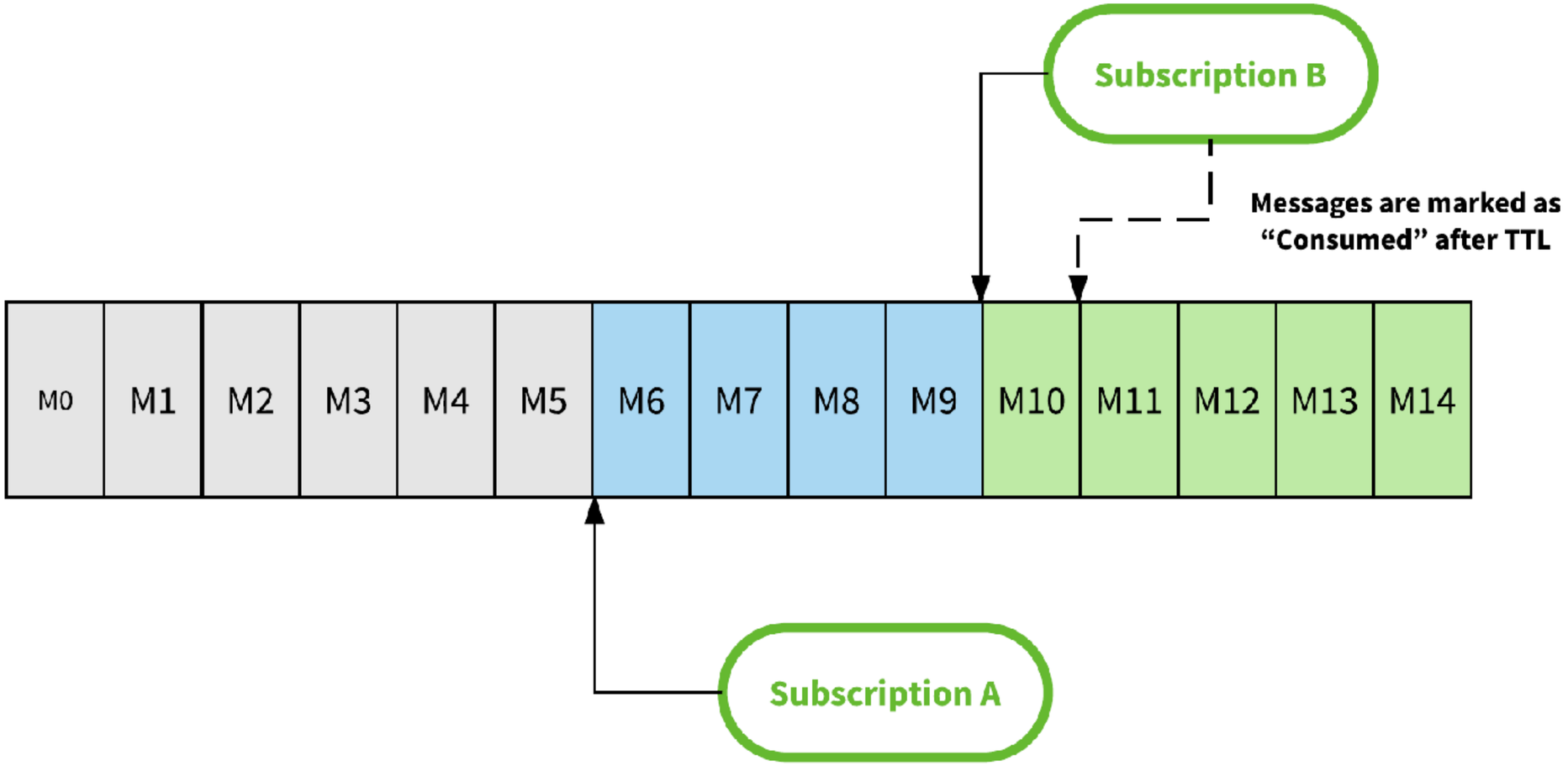
Messaging - ACK

Cumulative

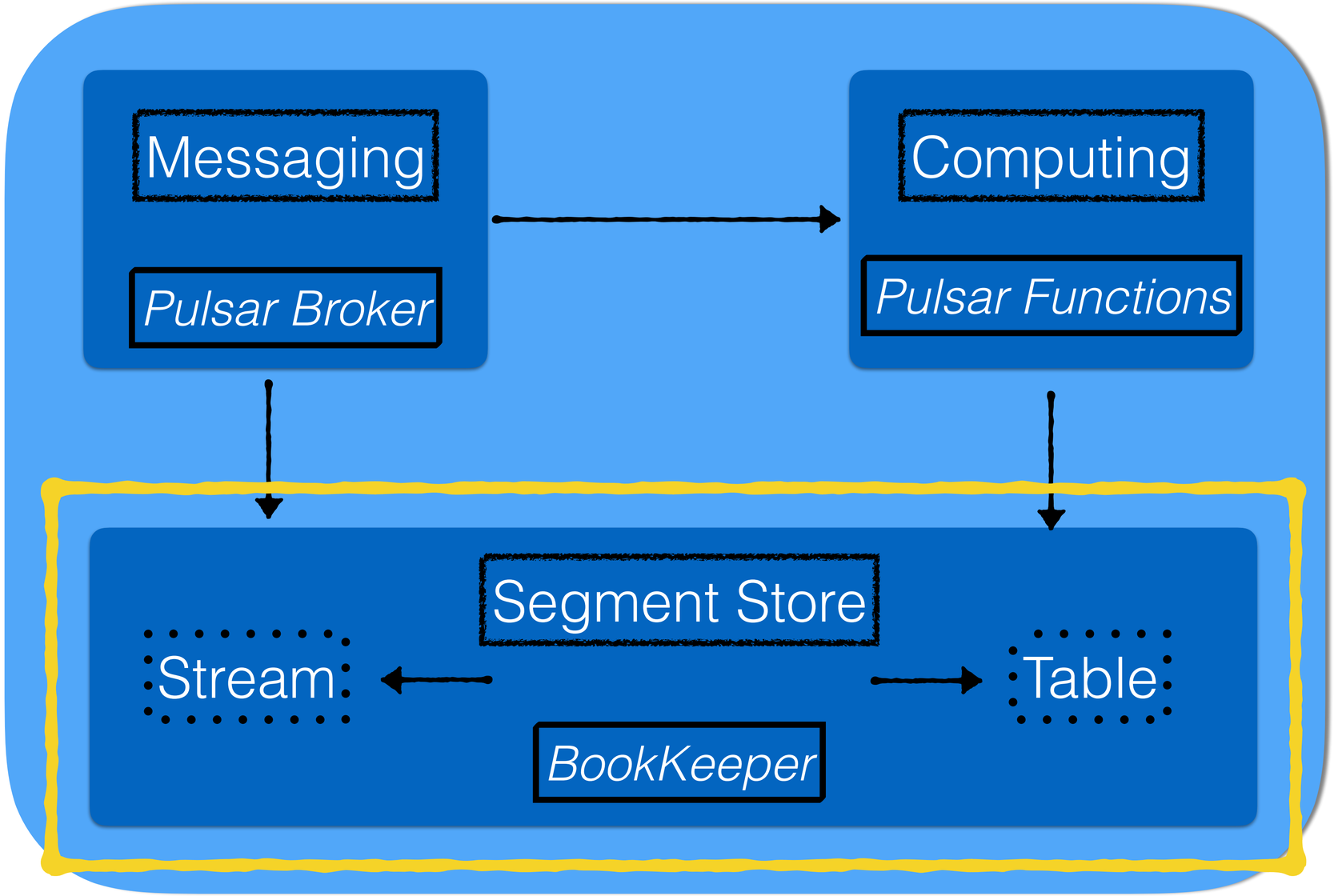
Individual



Messaging - Retention

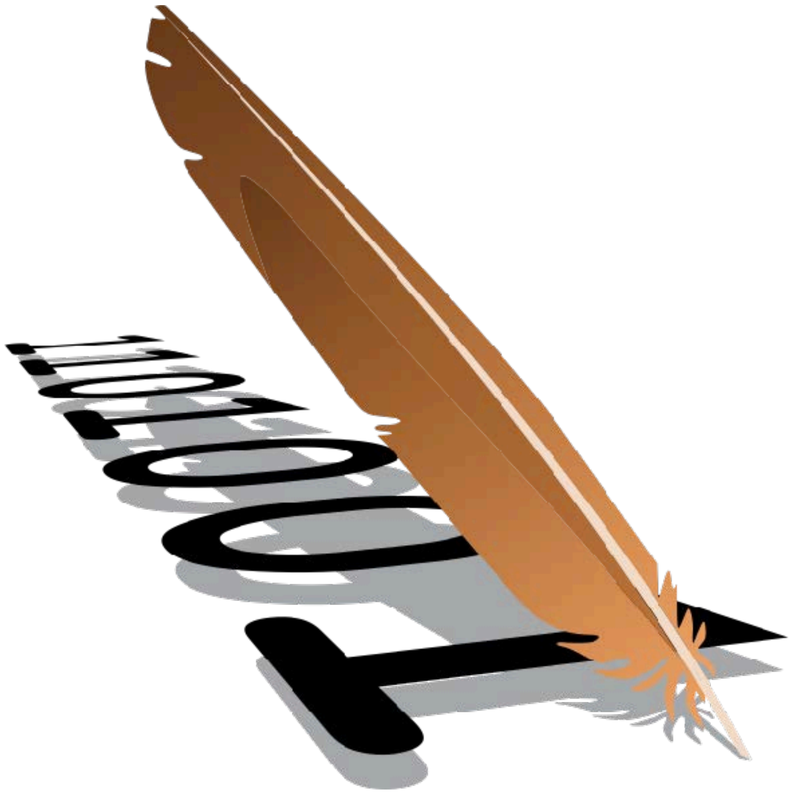


Storage

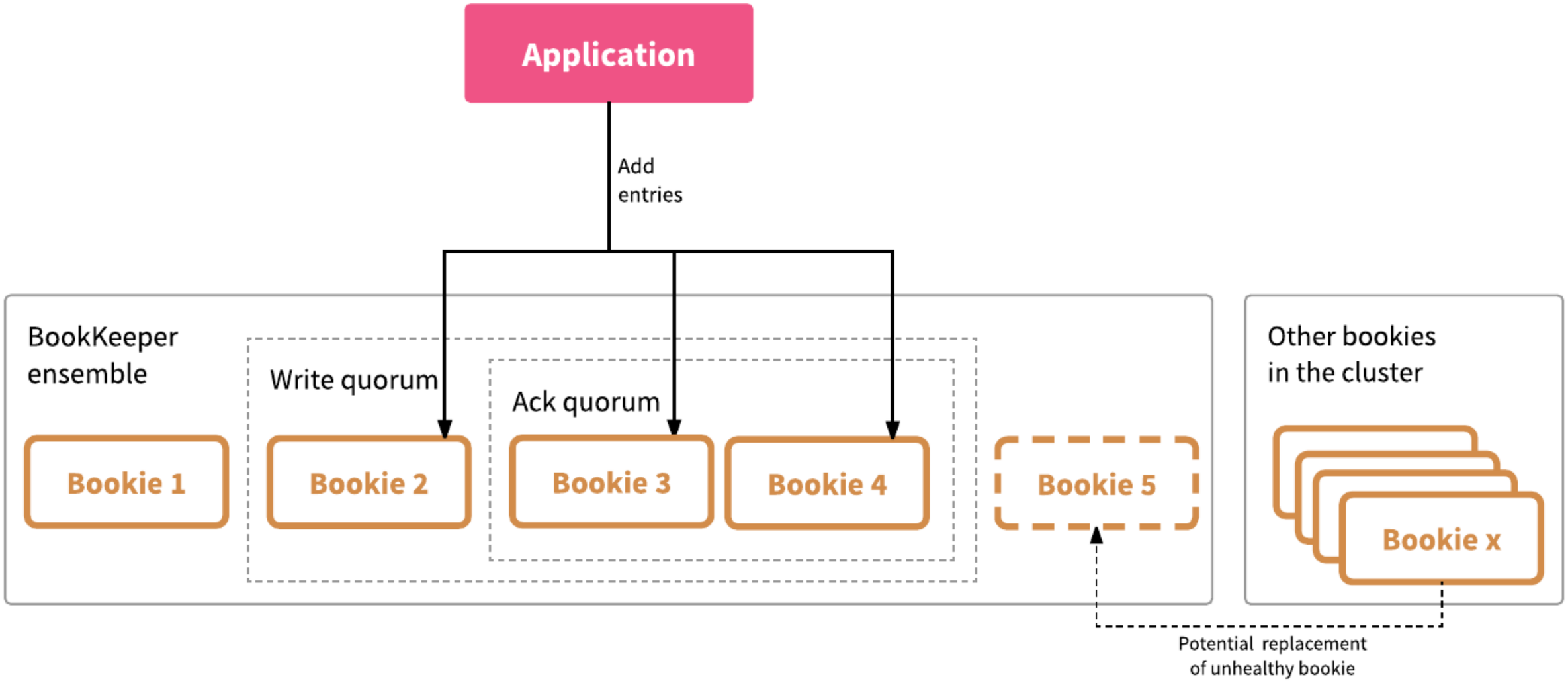

Messaging
Pulsar Broker
Computing
Pulsar Functions
Segment Store
Stream
Table
BookKeeper

Storage - Apache BookKeeper

- A replicated log storage
  - Low-latency durable writes
  - Simple repeatable read consistency
  - Highly available
  - Store many logs per node
  - I/O Isolation

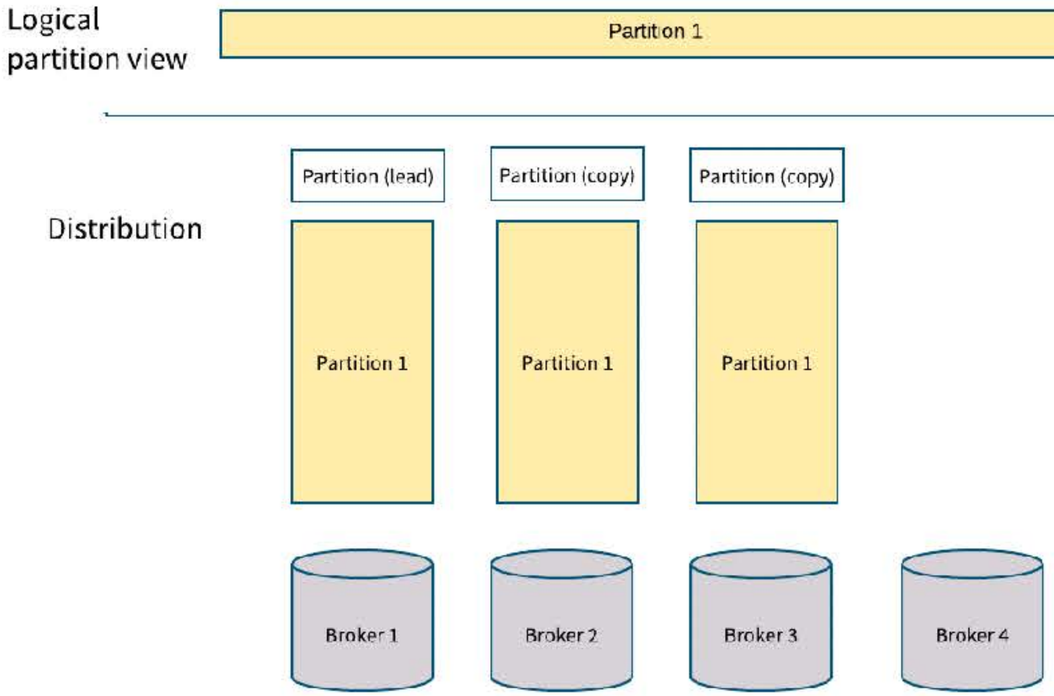


Storage - Apache BookKeeper



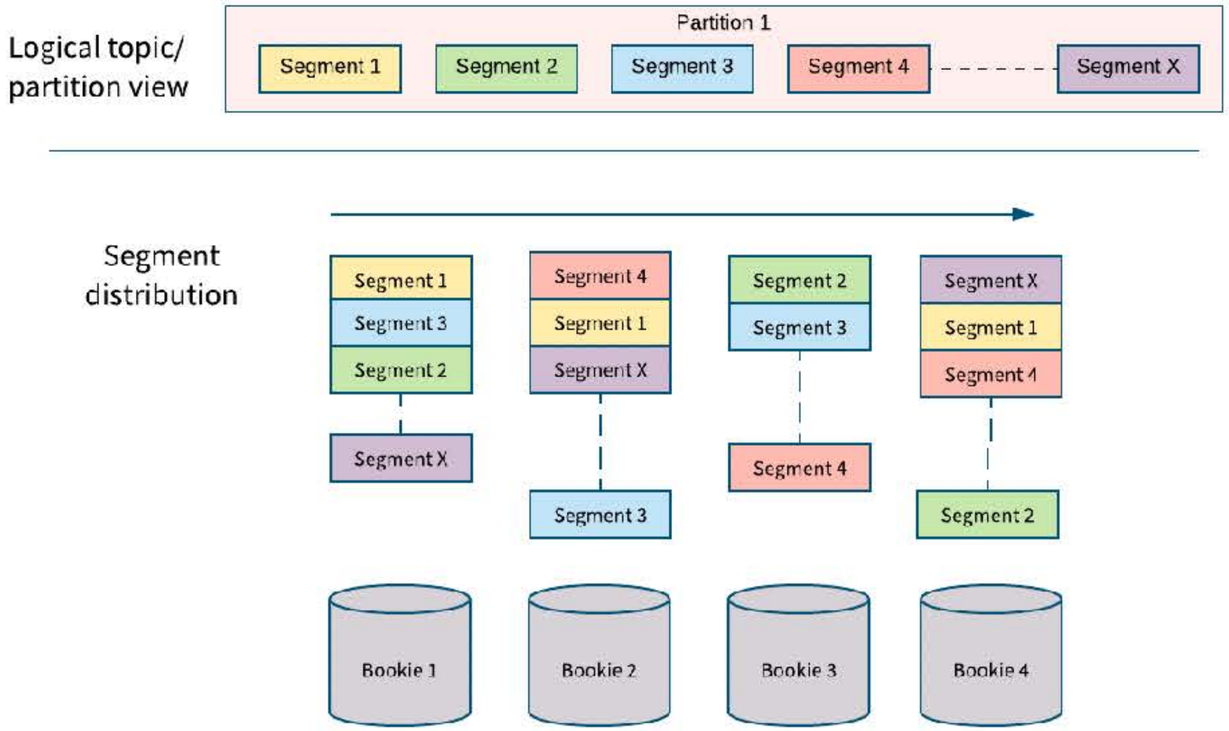
Storage - Segment Centric

Apache Kafka



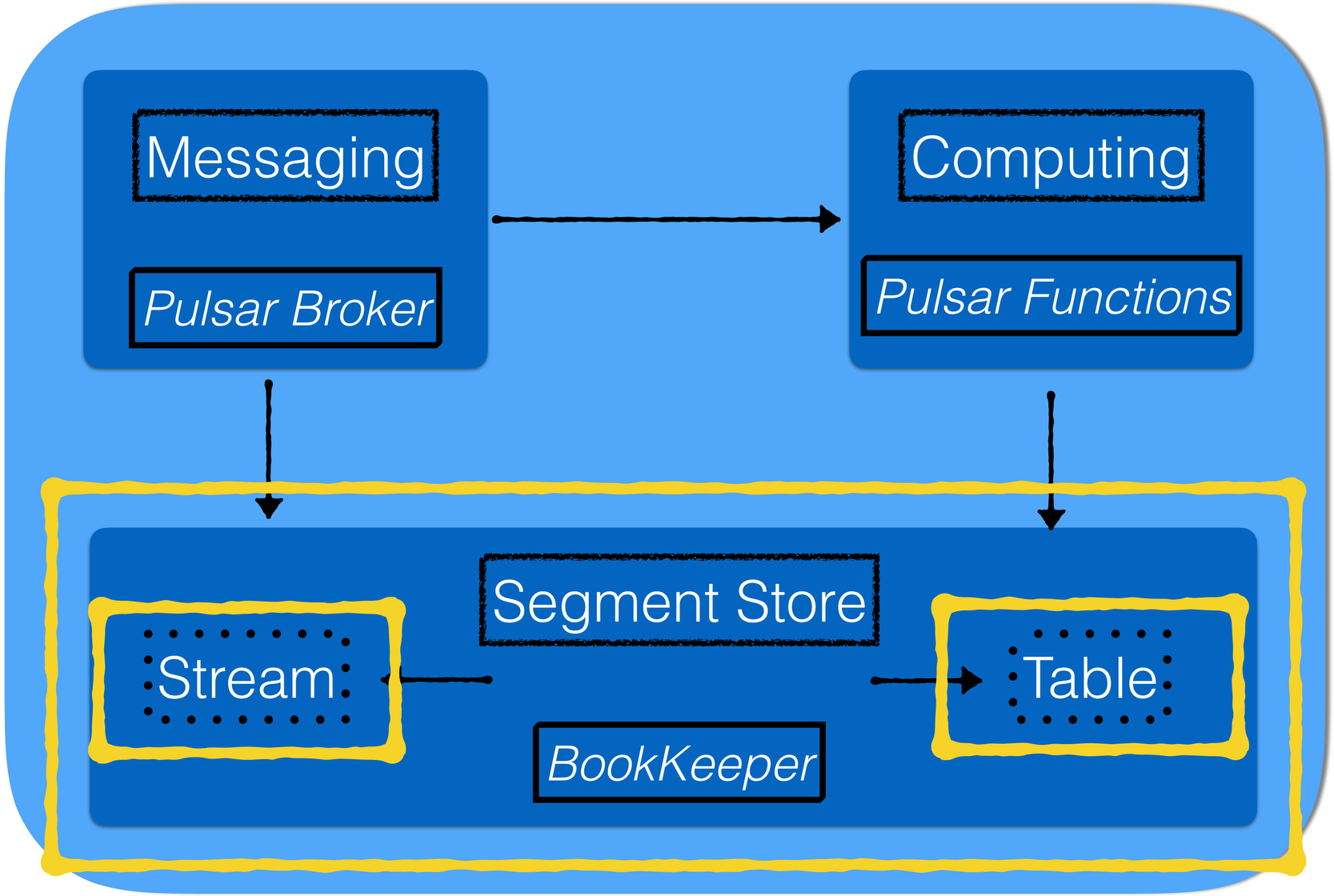
Kafka Partitions — All log segments are replicated in order across brokers (replication = 3 here).

Apache Pulsar/BookKeeper

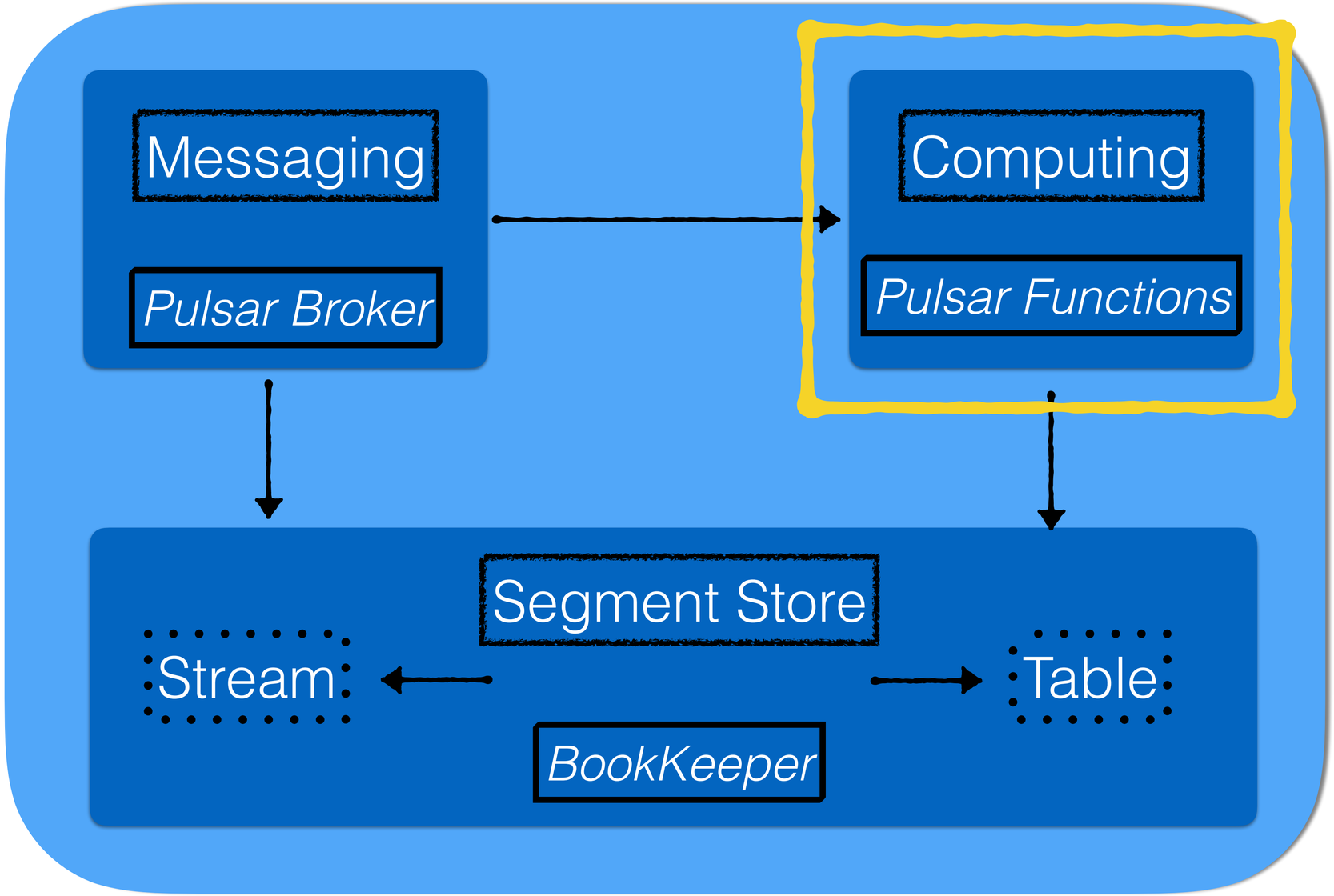


Pulsar/BookKeeper Stream — All log segment are replicated to a configurable number of bookies (replication = 3 here) across N possible bookies (N = 4 here). Log segments are evenly distributed to achive horizontal scalability with no rebalancing.

Storage - Segment/Stream/Table

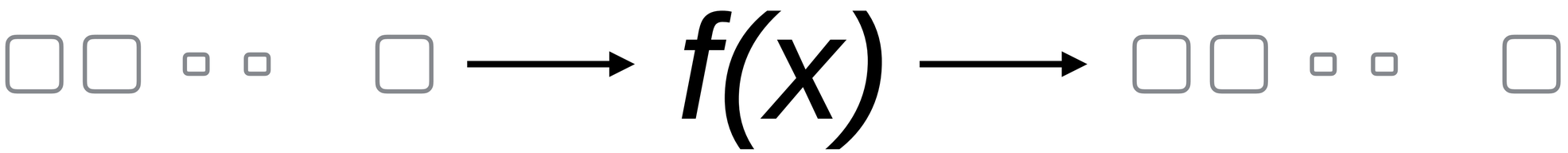


Compute


Messaging
Pulsar Broker
Computing
Pulsar Functions
Segment Store
Stream
Table
BookKeeper

Compute Representation

Abstract View



Incoming Messages

Output Messages

Lessons learnt

- A significant percentage of transformations are simple
  - ETL/Reactive Services/Classification/Real-time Aggregation
  - Event Routing/Microservices
- The emergence of Serverless
  - Simple Function API
  - Run per event
  - Composition APIs to do complex things
  - Wildly popular

Whats needed: Stream-Native Compute

Insight gained from serverless

- Simplest possible API
  - Method/Procedure/Function
  - Multi Language API
  - Scale developers
- Stream native concepts
  - Input/Output/Log as topics
- Flexible runtime
  - Simple standalone applications vs system managed applications

Pulsar Functions — API

- SDK less API

```
import java.util.function.Function;
public class ExclamationFunction implements Function<String, String> {
    @Override
    public String apply(String input) {
        return input + "!";
    }
}
```

- SDK API

```
import org.apache.pulsar.functions.api.PulsarFunction;
import org.apache.pulsar.functions.api.Context;
public class ExclamationFunction implements PulsarFunction<String, String> {
    @Override
    public String process(String input, Context context) {
        return input + "!";
    }
}
```

Pulsar Functions

Running as a standalone application

```
bin/pulsar-admin functions localrun \
    --input persistent://sample/standalone/ns1/test_input \
    --output persistent://sample/standalone/ns1/test_result \
    --className org.mycompany.ExclamationFunction \
    --jar myjar.jar
```

- Runs as a standalone process
- Run as many instances as you want. Framework automatically balances data
- Run and manage via Mesos/K8/Nomad/your favorite tool

Pulsar Functions: Use Cases

Edge Computing

- Sensor devices generate tons of data
- Lot of local actions
  - Simple filtering, threshold detection, regex matching, etc
- Resource Constrained
  - Limited scope for Full blown schedulers/Job Managers

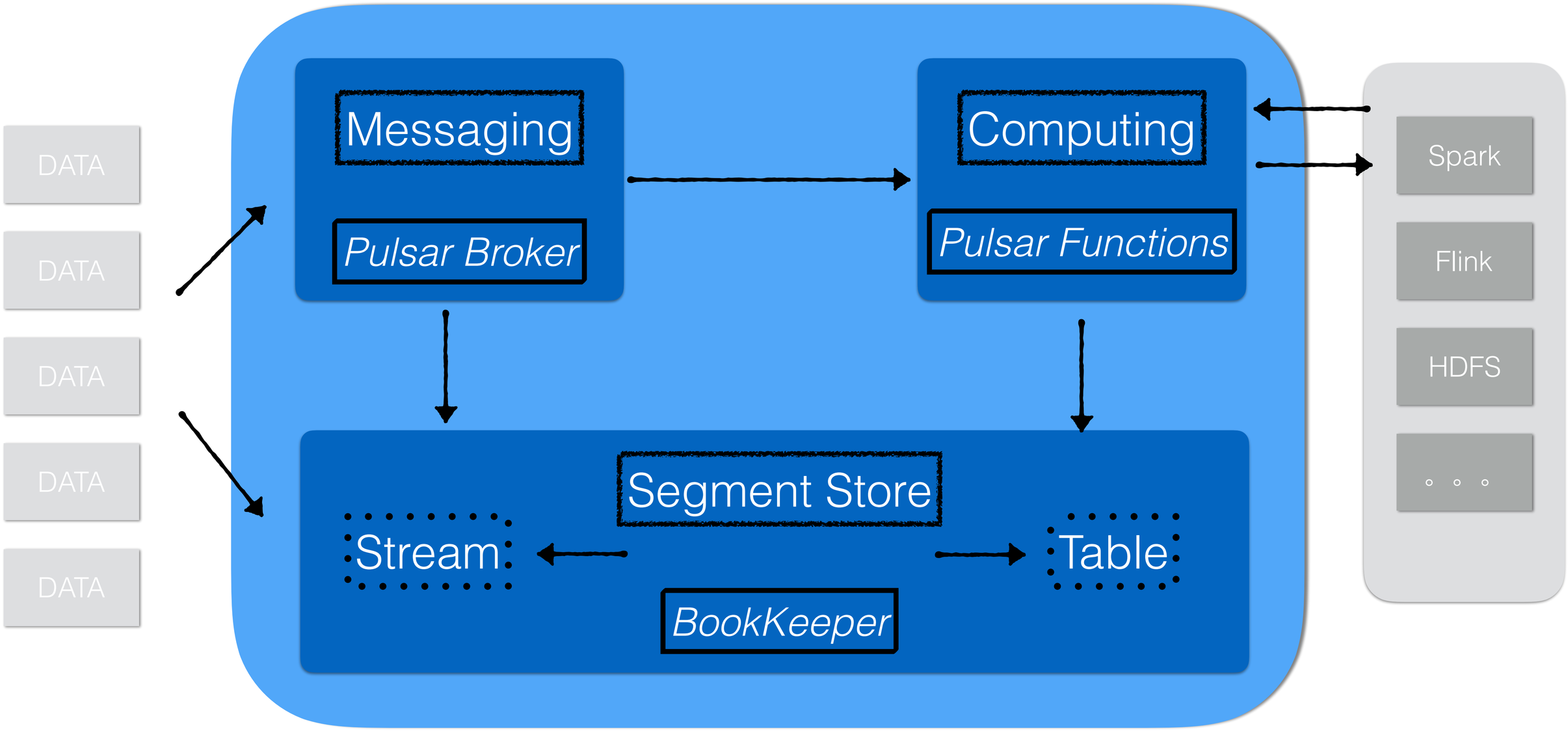
Model Serving

- Models computed via offline analysis
- Incoming requests should be classified using the model
- Function is a natural representation for the classification action
- Model itself can be stored in Bookkeeper

Pulsar Functions

- Unify Messaging and Compute cluster into one
- Function executed for every message of input topic
- Supports multiple topics as inputs
- Runtime User Controlled Guarantees:
  - ATMOST_ONCE / ATLEAST_ONCE / EFFECTIVE_ONCE
- Built-in State Management:
  - Unified Stream & State Store with BookKeeper.
- Simplified application development
  - No need to standup an extra system to develop/test/integrate/operate

Unified Streaming Solution



Messaging Benchmark

https://github.com/openmessaging/openmessaging-benchmark

Benchmark

- Testing goals
  - Throughput & latency under different conditions
  - Min 2 guaranteed copies
  - Running on 3 EC2 VMs with local SSDs

Kafka settings

- Topic settings

```
replicationFactor=3

min.insync.replicas=2

log.flush.interval.ms= # Using default: means no fsyncs
```

- Kafka producer config

```
acks=all

linger.ms=1

batch.size=131072
```

Pulsar/BookKeeper settings

- Use ensemble=3 write=3 ack=2
- Data synced on disk before ack
- Pulsar publisher settings:

```
batchingEnabled : true

batchingMaxPublishDelayMs : 1

blockIfQueueFull : true
```

Max throughput

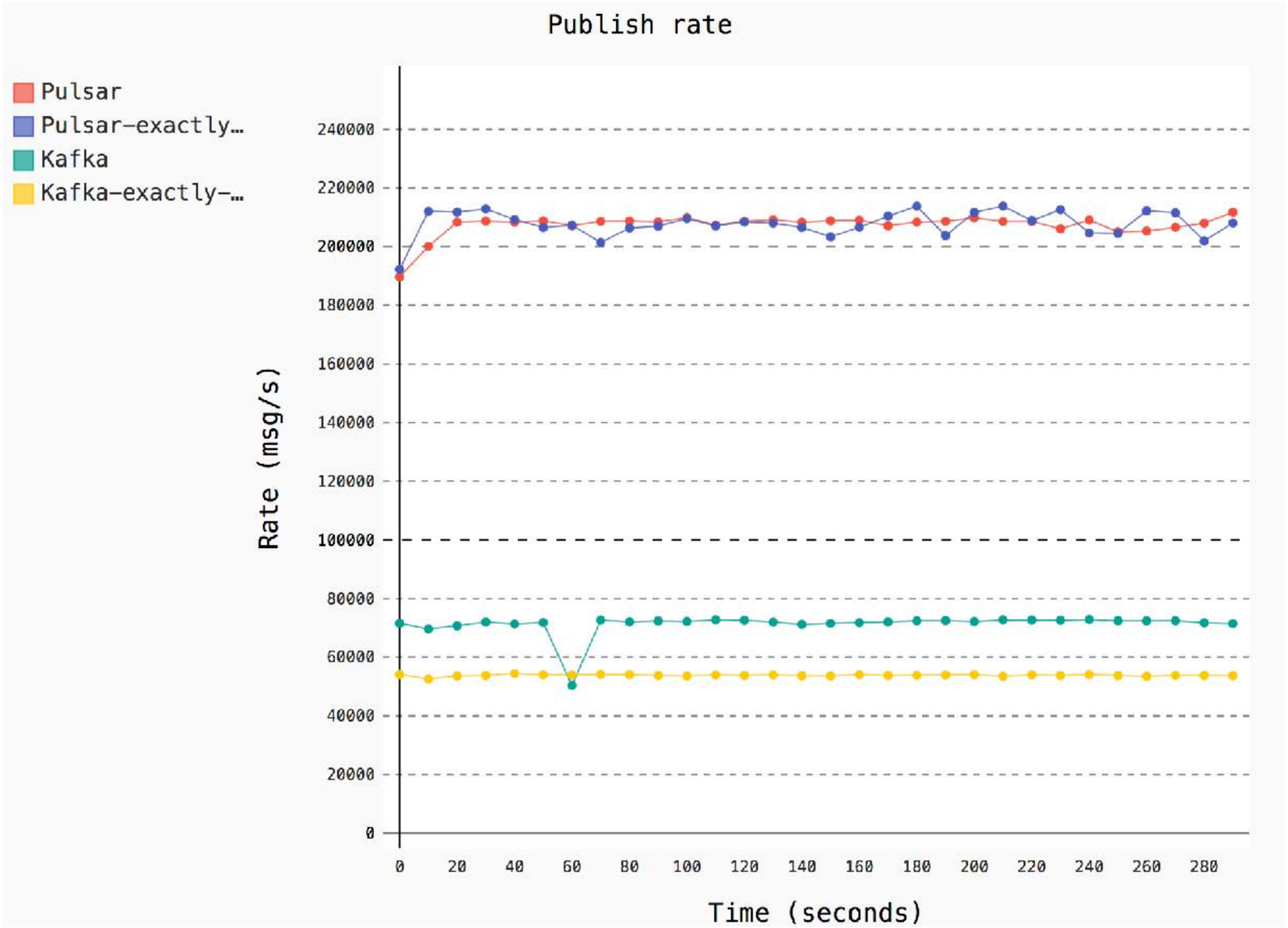
1 Topic

1 Partition

1 Producer

1 Consumer

1Kb msg



Latency at fixed rate - 50K msg/s

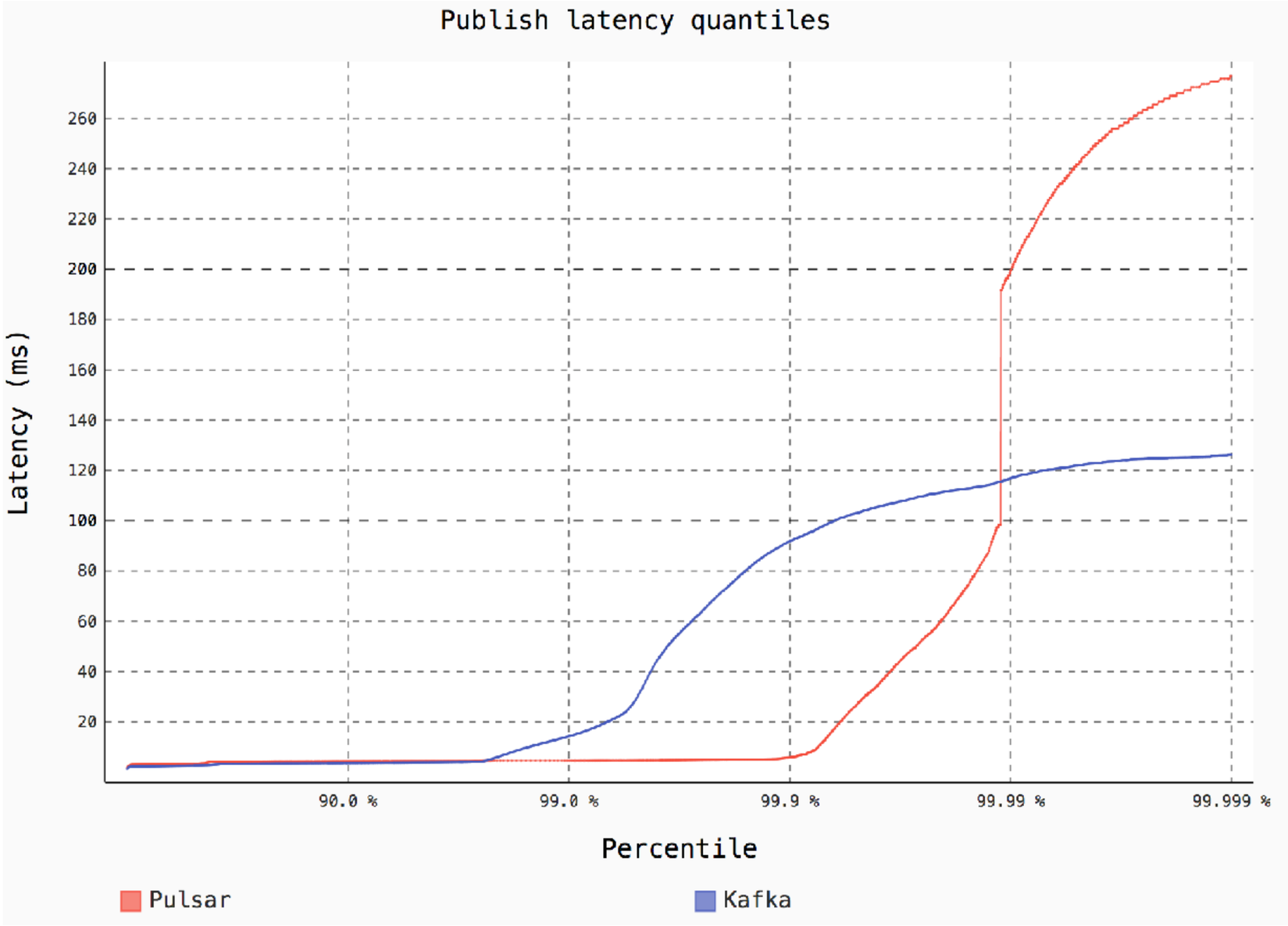
1 Topic

1 Partition

1 Producer

1 Consumer

1Kb msg



Latency at fixed rate - 50K msg/s

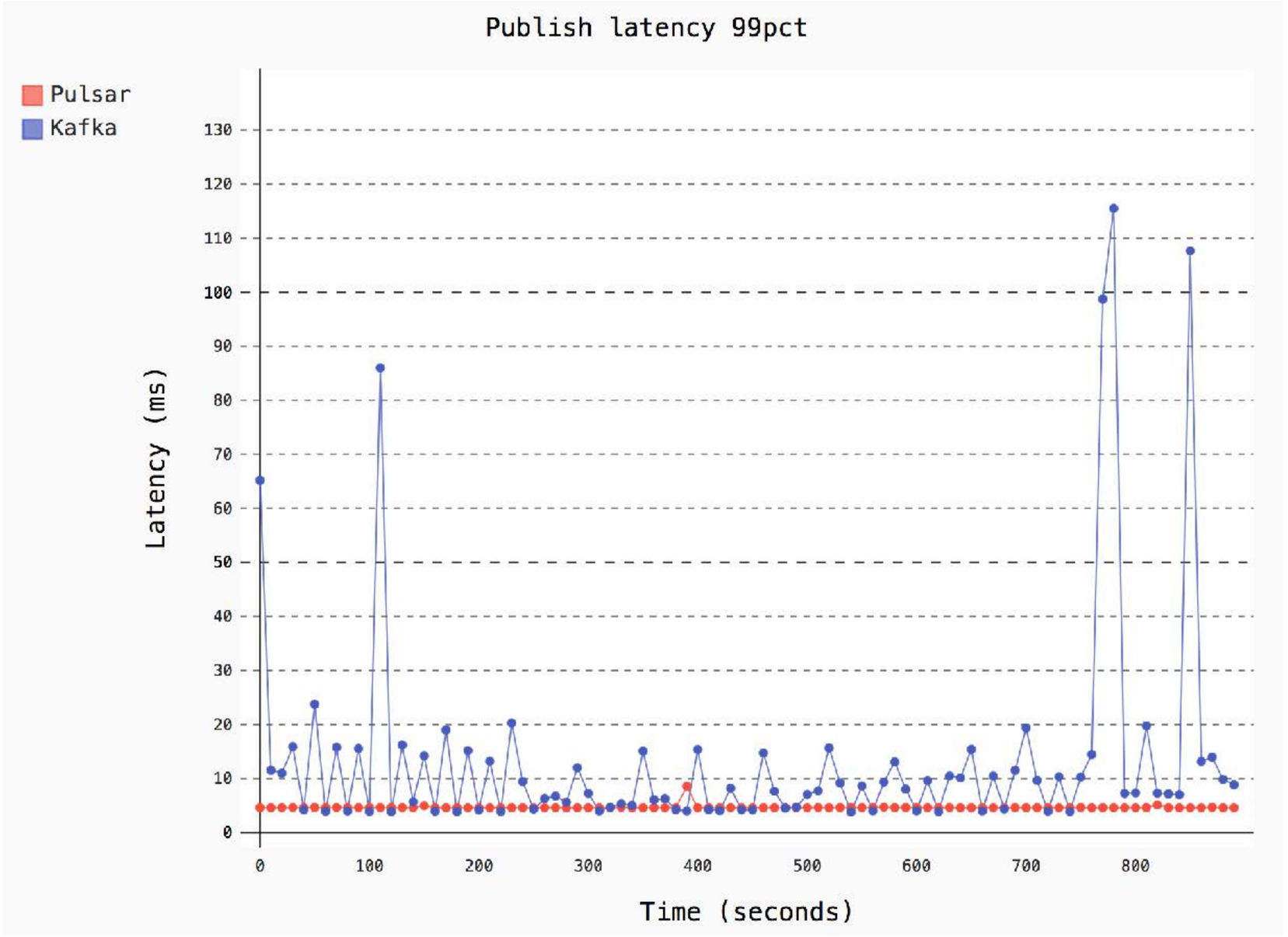
1 Topic

1 Partition

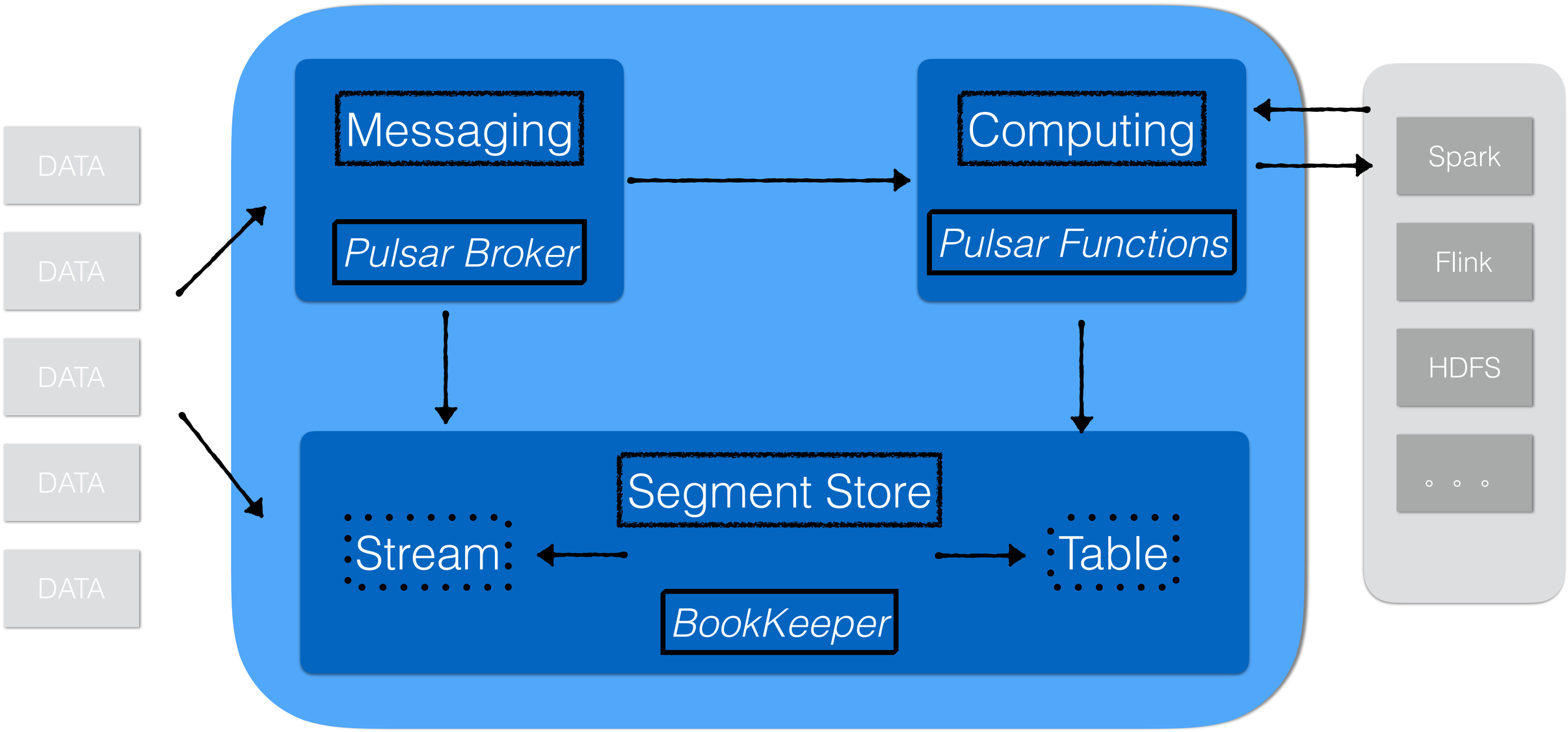
1 Producer

1 Consumer

1Kb msg



Unified Streaming Solution



Curious to Learn More?

- Apache Pulsar : http://pulsar.incubator.apache.org
- Apache BookKeeper : http://bookkeeper.apache.org
- Technical Blog : https://streaml.io/blog/
- Twitter: @apache_pulsar @asfbookkeeper
- slack:
  - https://apache-pulsar.herokuapp.com/
  - https://apachebookkeeper.herokuapp.com/

关注QCon微信公众号，
获得更多干货！

Thanks!

INTERNATIONAL SOFTWARE DEVELOPMENT CONFERENCE

主办方 Geekbang. 极客邦科技 InfoQ