Apache Kafka vs. Apache Pulsar: A Comparative Analysis

FA 2023 CS 511 Project Proposal

Ishita Karna
ikarna2@illinois.edu

Akshath SK
sk117@illinois.edu

Naveen Muthukumar
naveen3@illinois.edu

Joseph Jang
jyj3@illinois.edu

1 INTRODUCTION

Background. In the ever-evolving realm of data streaming, Apache Kafka [1] and Apache Pulsar [2] have risen as prominent contenders, each offering potent features for managing high-velocity, low-latency data streams. Our project's central aim is to conduct an in-depth comparison between Kafka and Pulsar, primarily focusing on benchmark-based evaluations. By subjecting these platforms to rigorous benchmarks, we seek to quantitatively assess their performance, resilience, and suitability across various data streaming scenarios.

Problem. Choosing the right streaming data platform is a critical decision for organizations. However, the abundance of options and the complexity of these platforms can make the decision-making process challenging. Understanding the nuances and trade-offs between Kafka and Pulsar is essential for organizations seeking to adopt a streaming data platform. This project seeks to address the lack of comprehensive, up-to-date, and unbiased comparative information between these two platforms.

Proposed Approach (Summary). Our proposed approach encompasses a comprehensive comparative study between Kafka and Pulsar, spanning key dimensions such as architecture, operational attributes, developer experience, ecosystems, deployment options, and security. A central focus will be the benchmark-based evaluation, where we rigorously assess performance metrics like throughput, latency, scalability, data retention, and durability. These benchmarks will be pivotal in providing empirical and actionable insights, augmenting our analysis of each platform's features and capabilities. We will leverage documentation, real-world use cases, and community insights, along with hands-on experimentation, to ensure the robustness of our findings, aiding organizations in selecting the optimal data streaming solution.

Expected Challenges (Summary). Navigating our comparative study between Kafka and Pulsar, we must acknowledge specific challenges. First, resource constraints on low-grade machines may limit our ability to generate and process substantial data volumes, potentially restricting our comparison's scalability under conditions involving large datasets. Second, aligning benchmark parameters between Kafka and Pulsar presents its own set of difficulties due to the inherent differences and unique configurations of these two platforms. Overcoming these challenges will be pivotal in ensuring a fair and comprehensive evaluation.

Difference from Existing Work. Unlike traditional research that relies on existing benchmarking tools, our project endeavors to create a custom Java tool. This tool, specifically tailored for our evaluation, promises more precise, relevant, and consistent results. While tools like JMH [5] and OpenMessaging [4] are prevalent in the industry, our unique combination of custom and industry-standard tools sets our work apart as compared to the work of Benchmarking Message Queues et al. [3]

While performance and operational metrics are crucial, the developer experience plays a pivotal role in the adoption and effective use of any platform. Our study is one of the few to offer a detailed examination of SDKs, APIs, and integrations, ensuring the evaluation is complete, not just from an operational standpoint but also from a developer's perspective.

2 PROJECT DESCRIPTION

2.1 Describe Concrete Problems

Selecting between Kafka and Pulsar is not merely a choice between two data streaming platforms, but a strategic decision influencing various aspects of data management:

- **Architecture Differences**: Both platforms have their unique architectural nuances as discussed in the comparison between Pulsar and Kafka[6]. For instance, Kafka's dependence on ZooKeeper versus Pulsar's segment-centric storage approach presents varied implications for deployment and management.
- **Operational Hurdles**: The operational traits of both systems vary. Organizations report varying experiences; some find Pulsar's multi-tenancy more appealing, while others might lean towards Kafka's mature community support.
- **Developer Experience**: Seamless integration and a developer-friendly environment are critical. The differences in SDKs, APIs, and integrations between Kafka and Pulsar might influence developer productivity and system integration timelines.
- **Ecosystem and Support**: With Pulsar being relatively newer, does its ecosystem provide the same depth and breadth of support as Kafka's mature community?

2.2 Describe Planned Approach

Our analysis aims to provide a holistic examination of both Kafka and Pulsar, intertwining empirical metrics with qualitative insights. A pivotal element of our approach is the development of a custom tool designed to streamline our analysis:

- **Custom Java-Based Tool Development**: At the heart of our evaluation lies a specialized tool, developed in Java. It's programmed to engage with the latest versions of Kafka and Pulsar. Through Docker deployments, this tool facilitates

consistent testing environments, ensuring accuracy in our results.

- **Benchmark-Based Evaluation**: Our custom Java tool, complemented by industry-acknowledged tools like JMH and OpenMessaging, will spearhead our systematic assessment of the following key performance metrics:
  - Throughput:
    - Gauge the maximum rate of message processing under ideal conditions.
    - Analyze the influence of parameters like message size, batching, and serialization on throughput.
    - Investigate behavior under high-stress conditions to identify potential system bottlenecks.
  - Latency:
    - Measure the average time between message dispatch and acknowledgment.
    - Delve into the intricacies of end-to-end latency, from message production to consumption.
    - Evaluate the impact of external disruptions, such as network issues and system overloads, on latency.
  - Scalability:
    - Examine system behavior in response to increasing workloads, especially with the addition of more nodes.
    - Study the efficiency of horizontal scalability and data distribution across brokers.
    - Monitor performance consistency as the system's scale varies.
  - Data Retention:
    - Explore various retention policies, like time-based or volume-based data purging.
    - Evaluate system proficiency in retrieving archived data, essential for scenarios like data replay or deep analytics.
    - Probe into the performance implications of long-term data retention.
  - Durability:
    - Test system robustness against adversities, including node failures or network partitioning.
    - Examine the efficiency of data recovery protocols, focusing on replication strategies.
    - Delve into the efficacy of acknowledgment protocols in ensuring no data loss.
- **Architectural and Operational Analysis**: Through hands-on experimentation and an exhaustive study of official documentation, we will unravel the architectural nuances and operational strengths unique to each platform.
- **Developer-Centric Assessment**: Moving beyond mere metrics, our evaluation will pivot towards understanding the developer's journey on both platforms. This involves an exhaustive study of SDKs, APIs, and potential integration challenges, painting a complete picture of the developer experience.
- **Ecosystem Exploration**: Our engagement won't stop at platform mechanics. Direct interactions with both Kafka's and Pulsar's communities will offer insights into ecosystem dynamism, community support, and prevailing sentiments.
- **Deployment and Security Analysis**: Our bespoke tool, along with simulated real-world scenarios, will shed light on deployment challenges, scalability considerations, and intrinsic security measures.

  Through our multifaceted methodology, blending innovative tool-based assessments with broader evaluations, we are poised to deliver an exhaustive, incisive, and highly pertinent comparative study between Kafka and Pulsar.

2.3 Evaluation Plan

The success and effectiveness of our comparative study between Kafka and Pulsar hinge on a well-structured evaluation plan:

- **Metrics and Indicators**:
  - **Quantitative Metrics**: Primarily extracted using our custom Java tool, these will include raw data on throughput, latency, scalability, data retention, and durability.
  - **Qualitative Indicators**: Insights drawn from architectural and operational analyses, developer-centric assessments, ecosystem explorations, and deployment and security analyses.
- **Data Collection**: Our custom Java tool, deployed on Docker, will automate the majority of our quantitative data collection, ensuring accuracy and consistency.
- **Analysis Tools and Methods**: The JMH and OpenMessaging tools, along with our custom tool, will be pivotal in analyzing quantitative metrics.
- **Evaluation Timeline**:
  - **Benchmark Evaluations**: Scheduled at regular intervals, we project to conclude our quantitative analysis within the initial half of our project timeline.
  - **Final Analysis and Report Compilation**: Reserved for the final phase, post all data collections, ensuring a comprehensive consolidation of findings.
- **Mid-Course Corrections**: Based on preliminary findings, especially from our quantitative benchmarks, potential refinements to our approach might be necessitated. Regular internal reviews will allow timely detection of such requirements, ensuring our study remains on track and relevant.

  To ensure our project's effectiveness:

We'll quantify the mean, median, and 95th percentile values for each of the core performance metrics during our benchmark tests similar to as the comparison between Kafka and Pulsar[7]

2.4 Collaboration Plan

Project Schedule:

Week	Tasks & Focus Areas
Week 1	Understanding of Kafka and Pulsar; Docker setup
Week 2	Working Java-based tool; Initial benchmark results.
Week 3	Deep-Dive into Architectural & Operational Analysis
Week 4	SDKs, APIs analysis and Developer Experience
Week 5	Focused Benchmarking - Intensive benchmark results
Week 6	Consolidated data points; Draft of the report.
Week 7	Review, Finalization & Submission

Role Assignment:

Member	Role
ikarna2	Project Coordinator & Data Analyst
sk117	Java Tool Developer
naveen3	Research Analyst
jyj3	Benchmarking Specialist

REFERENCES

[1] Apache Kafka. *Apache Kafka.* https://kafka.apache.org/. Accessed: 17 October 2023.

[2] Apache Pulsar. *Apache Pulsar.* https://pulsar.apache.org/. Accessed: [Insert Date of Access, e.g., 17 October 2023].

[3] Maharjan, Rokin; Chy, Md Showkat Hossain; Arju, Muhammad Ashfakur; Cerny, Tomas. *Benchmarking Message Queues.* Telecom, 2023; 4(2): 298-312. https://doi.org/10.3390/telecom4020018.

[4] OpenMessaging Contributors. *Benchmark.* GitHub. 2023. https://github.com/openmessaging/benchmark.

[5] OpenJDK. *JMH - Java Microbenchmarking Harness.* https://openjdk.org/projects/code-tools/jmh/. Accessed: 17 October 2023.

[6] Hevo Data. *Pulsar vs Kafka: A Detailed Comparison - Hevo Data.* https://hevodata.com/learn/pulsar-vs-kafka/#35. Accessed: 17 October 2023.

[7] Quix. *Kafka vs Pulsar: A Detailed Comparison.* https://quix.io/blog/kafka-vs-pulsar-comparison. Accessed: 17 October 2023.