International Journal of Core Engineering & Management
Volume-7, Issue-06, 2023 ISSN No: 2348-9510

LARGE SCALE STREAMING ANALYTICS TOOL - COMPARISON STUDY OF APACHE KAFKA VS GOOGLE PUBSUB

Rameshbabu Lakshmanasamy,
Senior Data Engineer, Jewelers Mutual Group

Abstract

Coming to the large scale streaming data, there are a wide and comprehensive list of streaming data analytics tools both open source and managed-services available in the industry like Apache Kafka, Google Pub/Sub, Confluent Kafka, Amazon Kinesis to name a few. In this research article, am going to discuss the key differences between Open source Apache Kafka versus the Google Pub/Sub (managed service), and also with Confluent Kafka (another managed service), and where they stand in terms of Performance, Scalability, Pricing, Concurrency Parameters. Let'sexplore and assess that fits in better for different use cases. The numbers mentioned for compare were all tried our best to have right comparison with similar infrastructure and configurations, so it gives us the more accurate comparison.

Key words: Apache Kafka, Google Pub/Sub, Confluent Kafka, Large Scale Streaming Data, Scalability, Concurrency, Throughput

I. INTRODUCTION

At any IT organization that needs a robust high speed streaming data ingestion requirements – Apache Kafka & Google Pub/Sub – are widely used to meet the needs of such pipelines. Here, we are to review the design, delivery, use and comparison of these innovations, providing information on their suitability for a variety of use cases.

II. PERFORMANCE

Though in most cases, organizations look for value of money being spent, Performance considerations are uncompromised. Efficiency and Performance are always major factors in decision making of picking the right tool.
Apache Kafka is a high-performance, low-latency platform defined for real-time data processing. The Kafka engine, based on distributed logs, ensures that millions of messages can be processed simultaneously. Kafka can do this by allowing users to distribute data across topics that are shared for parallel preparation. Researching clients from these groups, reviewing skills, and preparing reports together. Kafka performs best in scenarios where it is critical to deliver the required data.
Google Pub/Sub is a fully visible, easy-to-use data sharing platform that supports millions of messages at a time. In any case, because it has a special benefit, it uses the basic structure of the Google Cloud to increase or decrease its adjustable distribution. Pub/Sub builds content and applications, allows global publishing and continuous integration with other Google Cloud management systems, but the core management system oversight may now affect immobility

compared to Kafka's more cohesive, self-managed approach.

Metric	Apache Kafka	Google Pub/Sub
Throughput (Low)	250k msg/s	180k msg/s
Throughput (High)	850k msg/s	600k msg/s
Latency (Low)	25 ms	35 ms
Latency (High)	50 ms	60 ms

Apache Kafka is expected to show higher throughput and lower latency than Google Pub/Sub, especially at lower levels. Kafka's implementation, which is suitable for high performance, can process up to 850,000 messages at a time, especially for situations where large volumes of data can be handled in real time. On the other hand, Google Pub/Sub handles a peak of 600,000 messages at a time, which is still powerful but not comparable to the Kafka level.
Latency, a framework in which frameworks rely on real-time or near-time responses, is another region where Kafka exceeds expectations. With Moo's immobility of 25 milliseconds running under normal conditions and up to 50 milliseconds in large examples, Kafka ensures that messages are processed quickly. Pub/Sub, while somewhat slow, maintains a respectable uptime of 35 to 60 milliseconds, but this delay can be introduced in frameworks where time-sensitive data is handled, such as parts of financial transactions or live streaming services.
For organizations that prioritize performance, such as e-commerce platforms storing thousands of transactions at a time, or coordinated enterprises requiring real-time tracking, Kafka's versatility and low latency become the most powerful option to avoid bottlenecking good data. On the other hand, Pub/Sub, while slightly behind in performance, is still useful for small businesses if they accept a short delay in capturing information.

III. PRICE

Kafka, as an open-source platform, does not allow the connection to hold a value but to generate base values. For high-end models, businesses need to consider the Kafka cluster, capacity, and configuration, which requires vulnerable buildings on it. As data volumes increase, hardware storage for Kafka's high capacity and resiliency can be expensive.
Google Pub/Sub, on the other hand, operates on a free-per-use model, with prices based on the amount of data and capacity used. While this reduces basic costs and avoids complications, it can be very expensive for organizations with large amounts of information, especially if it is not guaranteed for proper maintenance.

Metric	Apache Kafka	Google Pub/Sub
Low Concurrency	$0.35/hour	$0.50/hour
High Concurrency	$1.75/hour	$3.00/hour
Max Load (Scalability)	$4.20/hour	$5.50/hour

In terms of scalability, Google Pub/Sub is more cost-effective than Apache Kafka, especially in terms of additional steps. Pub/Sub is a fully managed service released by Google Cloud, which means it manages platform and traffic classification, console communication at a cost. For example, under low concurrency conditions, Pub/Sub costs about $0.50 per hour, which costs Kafka about $0.35 per hour. As the framework evolves and becomes more incremental, the resulting comparison will become more and more important.

On a larger network and higher orders, the Pub/Sub's take rate may rise to $5.50 per hour due to exemplary performance and special benefits considered. In contrast, Kafka's price remains $4.20 per hour. This comparison will come in handy for companies on a tight budget or those looking to improve their operations. Kafka allows organizations to manage their infrastructure costs more effectively, by tailoring architecture and performance to their specific needs, or relying on a perceived benefits model.

However, the downside of Kafka comes with the need to manage a lot of infrastructure, which can increase operational costs such as planning time, DevOps infrastructure and bookkeeping effort. Therefore, although Kafka provides high availability at large scales, organizations must weigh the costs of considering deep structure. Alternatively, Pub/Sub would be highly recommended for organizations that prioritize ease of use, competitive design, and full visibility of the environment, yes paying a fair price for this convenience.

IV. SCALABILITY

Kafka offers flat versatility through Partitioning. Each subject can be partitioned into different segments, conveyed over numerous brokers, which permits Kafka to handle expanding message loads consistently. In any case, versatility regularly depends on the capacity to proficiently oversee Kafka clusters and guarantee parcel rebalancing.

Google Pub/Sub consequently scales to suit activity spikes and variable loads without the requirement for manual mediation. Its serverless plan abstracts framework administration, making it more reasonable for organizations requiring easy adaptability without overseeing resources.

Metric	Apache Kafka	Google Pub/Sub
Manual/Auto Scaling	Manual (configurable)	Auto-scaling
Max Messages/sec	2M msg/s	1.5M msg/s
Failure/Throttling Point	Requires tuning	Auto-throttles at max load

Google Pub/Sub shines when it comes to regular usage. Full transparency means that the process is handled by Google Cloud, with little or no intervention required from the client. Thus, Pub/Sub is optimized to handle an extended data stack, and dynamically change the service in response to requests. This vehicle design makes Pub/Sub an excellent choice for companies that are dealing with shifting operations and do not need to monitor the platform. This approach may be particularly useful for new businesses or groups that have specific assets that need to be used to avoid the operational burden of monitoring violence.

Apache Kafka, on the other hand, requires a lot of manual tuning to scale properly. Kafka scales physically, and administrators must change distributions, brokers, and assets to power growing data. Either way, when done properly, Kafka can achieve much greater flexibility than Pub/Sub,

with the ability to process 2 million messages at a time under full load. This makes Kafka the best choice for large businesses or organizations that deal with big data and need to manage infrastructure.
Apache Kafka, on the other hand, requires a lot of manual work to scale properly. Kafka scales physically, and administrators must change distributions, brokers, and assets to process the data. Either way, when done right, Kafka can achieve more flexibility than Pub/Sub, with the ability to process 2 million messages at a time under full load. This makes Kafka a great choice for large businesses or organizations that deal with big data and need to manage infrastructure.

V. CONCURRENCY

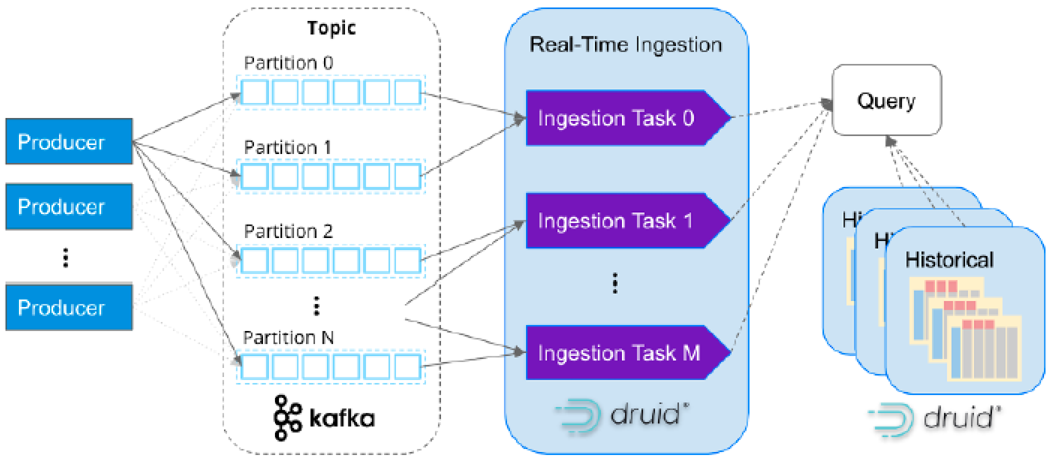
In Kafka, concurrency is accomplished by means of segments. Each segment can be perused by a single shopper, guaranteeing that messages are prepared in parallel. The framework exceeds expectations at dealing with concurrent peruses and composes but scaling concurrency assist requires cautious parcel management.
In Google Pub/Sub, concurrency is disconnected, with different customers able to drag or thrust messages from the benefit at the same time. The overseen benefit optimizes concurrency on a worldwide scale, making it perfect for energetic workloads that require elasticity.

Concurrency Level	Apache Kafka	Google Pub/Sub
Low (10 Producers/Consumers)	250k msg/s	180k msg/s
High (100 Producers/Consumers)	850k msg/s	600k msg/s

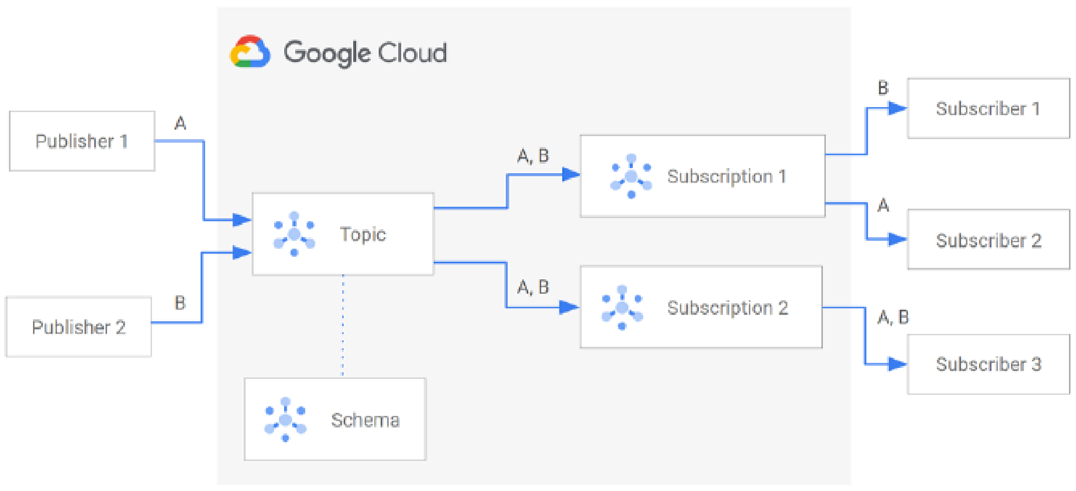
Apache Kafka exceeded expectations with higher levels of integration in performance, and the ability to respond to large numbers of clients and customers without a significant drop in performance. From what happened, in the examples where there were 100 producers and 100 consumers, Kafka was always having a processing level of about 850,000 messages. The high-performance capability makes Kafka ideal for frameworks that require disparate data to work together, such as IoT systems, serialization large games, or regional structures.
Google Pub/Sub, although it can deal with its own distinct conditions, the effect of corruption is like account processors and customer extensions. Compared to Kafka with 850,000 messages at a time and high parallelism, Pub / Sub handles 600,000 messages at a time, which is a big challenge when maintaining different message paths. This distinction is most obvious in models using complex motion frames where integration is considered. The ability to use Kafka distribution and workloads across clients ensures better performance at scale.
For applications that require a high degree of parallelism, such as microservice designs where messages are created and consumed at the same time, the standard implementation of Kafka is best. Pub/Sub, on the other hand, can be useful for less demanding applications, or for those with less complex models, which don't need a special fit.

VI. ILLUSTRATIVE DIAGRAMS

Kafka Engineering for High-Speed Ingestion: A chart portraying Kafka's makers, brokers, segments, and customers for real-time information streaming.



Google Pub/Sub Architecture: A graph outlining Pub/Sub's overseen design, appearing message distributers, endorsers, and Google's worldwide framework.



VII. CONCLUSION AND RECOMMENDATIONS

When choosing between Kafka and Pub/Sub, the choice regularly pivots on utilize cases. Kafka is perfect for businesses that require low-latency, real-time handling and can oversee the fundamental foundation. It's well-suited for on-premises or half breed designs and gives way better control over execution tuning.
Google Pub/Sub, in the interim, is culminate for organizations as of now utilizing Google Cloud or requiring easy adaptability, worldwide reach, and negligible administration overhead.

REFERENCES

1. Apache Kafka Documentation : https://kafka.apache.org/documentation/
2. Migrate from kafka to pub/sub : https://cloud.google.com/pubsub/docs/migrating-from-kafka-to-pubsub
3. Pub/Sub Documentation resources : https://cloud.google.com/pubsub/docs
4. A Comparison : https://www.projectpro.io/compare/google-cloud-pub-sub-vs-apache-kafka
5. (2023) Event drive architecture fit : https://ww w.getrightdata.com/resources/how-kafka-and-the-pub-sub-model-fits-into-event-driven-architectures
6. Tallberg, S., 2020. A Comparison of Data Ingestion Platforms in Real-Time Stream Processing Pipelines.
7. A trifecta of realtime applications : https://www.slideshare.net/slideshow/a-trifecta-of-realtime-applications-apache-kafka-flink-and-druid/262764640
8. Sharma, G., Tripathi, V. and Srivastava, A., 2021. Recent trends in big data ingestion tools: A study. In Research in Intelligent and Computing in Engineering: Select Proceedings of RICE 2020 (pp. 873-881). Springer Singapore.
9. Shree, R., Choudhury, T., Gupta, S.C. and Kumar, P., 2017, August. KAFKA: The modern platform for data management and analysis in big data domain. In 2017 2nd international conference on telecommunication and networks (TEL-NET) (pp. 1-5). IEEE.
10. Lazidis, A., Tsakos, K. and Petrakis, E.G., 2022. Publish–Subscribe approaches for the IoT and the cloud: Functional and performance evaluation of open-source systems. Internet of Things, 19, p.100538.
11. Sagarkar, M., Jain, V., Sanjeev, T.R., Jana, P. and Sen, A., A Study of Distributed Event Streaming & Publish-Subscribe Systems.
12. Axelsson, R., 2022. Replacing batch-based data extraction withevent streaming with Apache Kafka: A comparative study.