[1]Dr. Sanjay Kumar N V

[2]Dr. Anitha C L

[3]Dr. Ashwini B P

[4]Dr. Savithramma R M

[5]Mr. Subhash Kamble

Performance Benchmarking of Redpanda an event streaming platform using SBK and Open Messaging Frameworks

Abstract: - Redpanda is an event-driven, publish/subscribe based messaging system. The two main performance benchmarking frameworks, Storage Benchmark Kit (SBK) and open messaging benchmark are used here to test the performance benchmarking of redpanda's event processing system. SBK is a software framework for measuring any storage device or client's performance benchmarking and open messaging benchmark framework which supports benchmarking for a wide variety of messaging platforms. Well formulated fine tuning system for benchmarks is still an open research problem. In this paper, SBK and open messaging frameworks have been experimentally compared with respect to their messaging capabilities (Message sending and receiving throughput) on redpanda's event processing system. A number of experimental test scenarios have been conducted on data streaming workload, the performance evaluation results revealed that SBK gives maximum throughput performance compared to open messaging benchmark.

Keywords: *Benchmarking; Framework; Distributed Systems; File System; SBK; Redpanda; Events; Workloads; Producers;Consumers; Message Queues; Performance; Storage; Streaming; Throughput; Topic;*

I. INTRODUCTION

SBK: An open source, vendor neutral, software framework has been developed for assessing the performance benchmarks of various storage systems [1-2]. SBK is intended for measuring the the highest throughput and minimal latency for any storage device/client. When comparing performance Benchmark Frameworks, throughput in terms of messaging capabilities (message sending and receiving) is generally considered. In order to determine the optimal performance of storage system benchmarking is used [3]. SBK is the right tool to measure the performance of Various storage systems, including Database Systems, Persistent Key-Value storage systems, Object storage systems, Distributed messaging systems, etc., are supported by SBK for performance benchmarking, accommodating multiple writers and readers/callback (push) scenarios. It writes a large volume of data to the storage system and retrieves data from the storage system on a significant scale [4]. SBK is designed to handle high message rates and large volumes of data. It provides a

standard set of tests that measure the throughput, latency, andresource utilization of messaging platforms.

The OpenMessaging Benchmark Framework will offer benchmarking tools for a growing number of messaging platforms. It aims to develop streaming and messaging applications for various heterogeneous platforms and systems [5]. The benchmarking tool for Open Messaging facilitates the assessment of Kafka's

[1] Professor, Dept. of CSE, Kalpataru Institute of Technology, Tiptur, Karnataka, India

sanjaynv@gmail.com

[2]Professor, Dept. of CSE, Kalpataru Institute of Technology, Tiptur, Karnataka, India

clanitha@gmail.com

[3]Assistant Professor, Dept. of CSE Siddaganga Institute of Technology, Tumakuru, Karnataka, India

savirmrl@sit.ac.in

[4]Assistant Professor, Dept. of CSE Siddaganga Institute of Technology, Tumakuru, Karnataka , India

ashvinibp@sit.ac.in

[5]Assistant Professor, Global Academy of Technology, Bengaluru, India

subhashkamble@gat.ac.in

Copyright © JES 2024 on-line : journal.esrgroups.org

performance for publish/subscribe based messaging systems. Kafka serves as an event streaming platform, utilizing a persistent distributed commit log for the implementation of publish-subscribe messaging [6]. Kafka is commonly employed in the development of real-time streaming applications, wherein the input and/or output data isstored within Kafka clusters. On the client side, Kafka Streamsintegrates the ease of writing and developing conventional Java and Scala applications [7].

Redpanda is a kafka-compatible, event streaming data platform which is designed for maximum performance on any data streaming workload [8]. Redpanda can be deployed on bare-metal commodity hardware, virtual machines, and containers, supporting installations on both on-premise and cloud platforms. It is specifically crafted for cluster environments, enabling multiple clients to access it simultaneously. Redpanda and Kafka implement the Kafka API. Redpanda aims to bring operational simplicity to the existing overwhelming complexity of standing up state-of-the-art streaming systems [9]. Redpanda is built with a focus on ease of use, reliability, and scalability, and is rapidly gaining popularity as a leading event-driven platform for modern data-driven applications [10]. Redpanda is designed to be significantly faster than kafka, particularly for small messages and high message rates.

Producers and Consumers are client applications, producerssend data to Redpanda in the form of events, where as Consumers subscribe to Redpanda topics to read an events. Redpanda is designed to handle both single and multiple producer-consumer workloads. Redpanda safely stores these events in sequence and organize them into topics, which represents a replayable log of changes in the system.

Publishers and subscribers are terms used to describe message producers and subscribers in the pub/sub of redpanda's platform. Each message in pub/sub messaging is sent to a specific topic and may be delivered to one or more consumers which are registered in the topic [11]. Consumers can store, process, or reply to events after registering for the topic in order to start receiving messages.

Storage benchmarking has primarily been carried out to assess performance [12]. In this paper the two open source benchmarking frameworks, SBK and Open messaging have been experimentally compared with each other to measure the throughput performance of a redpanda's event processing system. when making a decision based on performance and data accessibility we tested with single partition for a workloadof 1-topic-1-producer-1-consumer-100bytes and 5 partitions, 1- topic-5-producer-5-consumer-100bytes, in all testing scenarios SBK performs well with the maximum number of messages published to the topic as well as maximum number of messages consumption from the topic per second. To determine the actual throughput for a specific use case, it is recommended to conduct performance benchmarking, which can help to optimize and fine-tune the system for the workload.

II. RELATED WORK

This section gives the most relevant work that evaluate the performance of two different benchmark tool. In the paper titled 'Distributed Streaming Storage Performance Benchmarking: Kafka and Pravega' Authors: Dr. Keshava Munegowda & Sanjay Kumar N V, have assessed the messaging capabilities of SBK and the open messaging benchmark framework suggested that The throughput achieved by the open messaging benchmark tool, when evaluating the performance of 10 Kafka producers alongside the Pravega benchmark tool, is notably lower in comparison tothe Pravega benchmark tool [13]. The open messaging benchmark tool does not pump or flush the data to the Kafka client at the maximum rate. Presently, distributed messaging and streaming platforms are being benchmarked in the development of the open messaging benchmark framework [14].

A streaming data platform Redpanda which is a kafkacompatible provides 10 times faster and 6 times more hardware efficient. While the Kafka performance benchmarking is made easier by the open messaging benchmark tool, SBK streamlines benchmarking processes across streaming and distributed messaging platforms, along with both generic persistent and non-persistent storage platforms. Finding the optimal data partitioning strategy is particularly challenging in Big Data applications [15].

III. TEST METHODOLOGY

The performance and architecture of storage are crucial factors that significantly impact the overall system

performance [16]. Redpanda is comptaible with the kafka API,No Zookeeper, No JVM is required, so it works with the full ecosystem of tools and integration built on kafka [17]. This section gives the most relevant work that evaluate the performance of two different benchmark tool. In our experiment redpanda is deployed on bare metal, here we can benchmark read/write events to an events processing system (redpanda). In Redpanda we can use the kafka protocol to implement a producer and consumer for a topic [18]. We are currently assessing the maximum message produced

/consumed are achievable with both the Open messaging and SBK frameworks for a given redpanda configuration. The two different combinations of workloads have undergone test using their default redpanda settings and configurations. In each test case, the throughput is measured for every test application i.e to send/receive messages to/from broker, and results have been recorded.

A. Single producer and consumer model

In a model with a single producer and consumer (Max- rate-1-topic-1-partition-1p-1c-100b), A Single topic with single partition, a single producer and a single consumer process messages to publish and subscribe the data across with one broker. In a single producer consumer workload, there is only one producer and one consumer interacting with Redpanda. This is typically used for low-volume workloads or for testing and development purposes.

B. Five producer and consumer model

In a Five-Producer-Consumer Model (Max-rate -1-topic-5-partition-5p-5c-100b), In order to publish/subscribe the message, The broker serves as a load-balancing node, distributing each message to all five consumers subscribed to the same topic. In this model of Five-producer consumer workload, Five producers and consumers interacting with Redpanda simultaneously. Multiple producers and consumers are used for high-volume workloads where data is being produced and consumed at a high rate.

TABLE I. DESCRIPTION OF THE HARDWARE AND SOFTWARE CONFIGURATIONS UTILIZED IN THE TEST SETUP

Components	Remarks
Number of compute nodes	2 Nodes 1 for Redpanda broker 1for OpenMessaging & SBK Each node is installed with Ubuntu(22.04.1 LTS)
CPU	4 CPUs : Each of 64 Bit, 3.6 GHz
RAM	8 GB
Hard Disk	SSD (Solid State Drive) of size1TB (Tera Bytes)
Ethernet	100Mbps Network
Software Versions	SBK : 3.0 Kafka : 3.3.2 Redpanda : 22.3.11

Table 1. shows, the hardware and software specifications that we have used for performance evaluation. Configuring the workload in Redpanda requires changing a number of factors in order to tailor the platform to our particular use case.

A few of the key parameters that can be configured include the number of partitions, replication factor, and message size. The number of partitions determines how many logical units of data are stored in the distributed log. The throughput of the system can be increased by increasing the number of partitions, but can also increase the complexity of

managing the system. The number of copies that each partition is stored in the system depends on the replication factor. As Replication factor increases can enhance data durability and fault tolerance, but it can also increase the overall resource requirements of the system [19].

The message size determines the size of each record in the distributed log. Reducing the message size can improve the overall throughput of the system, but can also increase the overhead of processing individual messages. Redpanda's workload configuration requires modifying the configuration file or using the command-line interface. It's important to carefully evaluate the impact of each configuration parameter on the overall performance of the system, and to test thesystem under realistic workloads to ensure that it meets an intended requirements [20].

IV. IMPLEMENTATION DETAILS

The initial version of SBK was implemented in Java 8 and the latest release of SBK 3.0 is incorporated in Java version 17 and it can be found on the Git repository hub [2] with the latest release of 3.0 SBK and Open messaging is used to check performance benchmarking of redpanda's events processing platform. In open messaging benchmark workloads are specified in YAML configuration files which describes the number of topics, size of the messages being produced and consumed, the number of producers and subscriptions per topic, the time duration etc,[21]. In SBK running performance benchmarking with write/read a specific amount of events/record and write/read the events/record to the specified amount of time. At the end of the benchmarking sessions both SBK and open messaging outputs the total written/read, average throughput and latency. Benchmarking Redpanda with open messaging using a latest version [22] provides a standard set of test that measure the throughput and latency.

V. RESULTS AND DISCUSSIONS

In this paper the total number of messages produced or consumed per second is the measure of throughput. Test performance metrics are used in benchmarking to assess storage performance. The most fundamental measure of input/output (IO) performance is throughput. To determine optimum storage performance, a model workload or appropriate benchmark must be used in order to evaluate and compare the benchmarking results on the specified system [23].

The experimental messaging performance study of the broker is presented in this section using five test scenarios to illustrate the results. Redpanda is designed to handle bothsingle and multiple producer-consumer workloads. In a single producer consumer workload, there is only one producer and one consumer interacting with Redpanda. In a multiple producer-consumer workload, there are multiple producers and consumers interacting with Redpanda simultaneously. Redpanda's setup options, such as the number of partitions, replication factor, and buffer size for network I/O, may need tobe changed in order to adapt to a multi-producer consumer workload.

A. In a Single topic-1producer-1consumer model

The redpanda performance benchmarking is conducted with data size of 100 bytes. In a single topic, one producer publish the message to the topic at the other end one consumer read themessages from the same topic, i.e only one producer and one consumer interacting with the broker.

Redpanda's performance is benchmarked using CLI, for Singletopic-1producer-1consumer model.

- **Through Open Messaging**

~/openmessaging-benchmark $ sudo bin/benchmark --drivers driver-kafka/redpanda-throughput.yaml workloads/max-rate- 1-topic-1-partition-1p-1c-100b.yaml. This command initiates a benchmarking process aimed at assessing the throughput capabilities of the system. The --drivers driver- kafka/redpanda-throughput.yaml segment specifies the driver or configuration file required to interact with either Kafka or Redpanda, indicating the specific system under evaluation. Meanwhile, the workloads/max-rate-1-topic-1-partition-1p-1c- 100b.yaml portion defines the workload parameters, including the maximum rate, number of topics, partitions, producers(1p), consumers (1c), and message size (100 bytes).

- **Through SBK**

~/SBK$ ./build/install/sbk/bin/sbk -class redpanda -topic 1 - partitions 1 -replica 1 -insync 1 -writers 1 -readers 1 -size 100

-seconds 180 -broker 10.0.100.134:9092 -throughput 30 - create true. This command is used to run a benchmarking test on a Redpanda instance with specific configurations, including the number of topics, partitions, replication factor, message size, duration, throughput, and broker address, among others. It will assess the performance of Redpanda under the defined conditions.

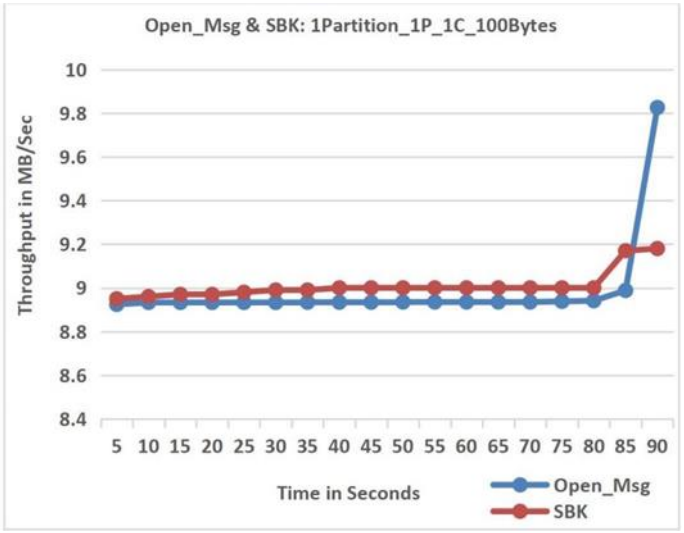

Fig. 1. Iteration1 Results Chart.

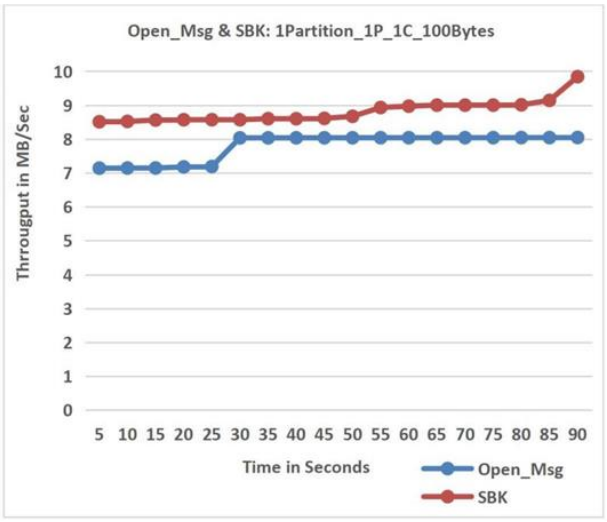

Fig. 2. Iteration2 Results Chart.

Fig. 3. Iteration3 Results Chart.

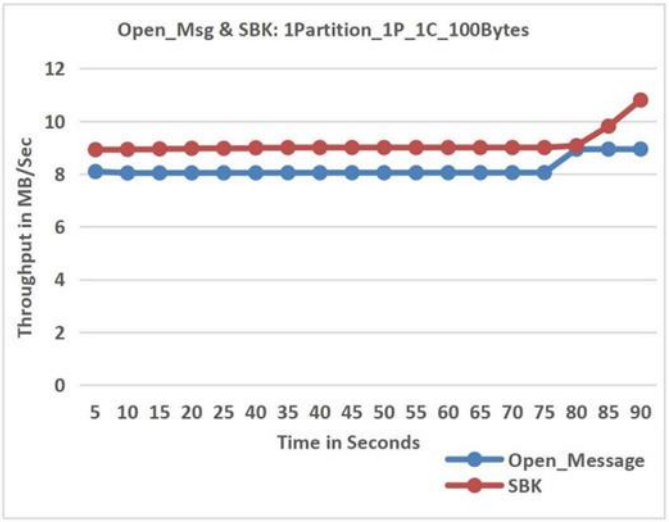

Fig. 4. Iteration4 Results Chart.

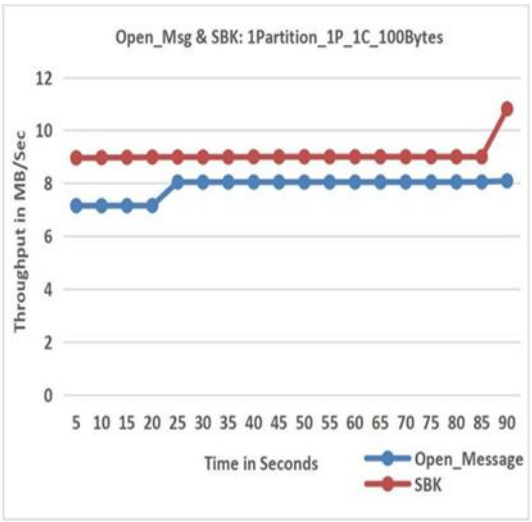

Fig. 5. Iteration5 Results Chart.

Figure 1-5 shows that SBK and Open Messaging benchmark performance are compared for a workload of '1-topic-1-partition-1p-1c-100b', here the analyzed metric is throughput, SBK benchmarking results shows the

approximatemaximum throughput of 11MB/sec and open messaging gives the maximum throughput result of 8MB/sec. The test scenario was run five times, lasting three minutes each time, and the results were graphed, when compare to open messaging at the completion of each testing period, SBK performs better to write/read events of size 100 bytes to/from the topic.

B. In a Single topic-5producer-5consumer model

The benchmarking of Redpanda performance is carried out using 100-byte data size. Single topic with 5producers are writing events to the topic (publish the message through the queue) five consumers at the opposite end subscribe to these events (subscribe the messages from the queue). In this model,a single topic is created in Redpanda, and five producers areset up to write messages to this topic. These producers publish messages to the topic, which is essentially a queue. On the other end of the queue, there are five consumers who subscribe to the topic and retrieve the messages and each message is delivered to only one of the consumers.

Redpanda's performance is benchmarked using CLI for Singletopic-5producer-5consumer model.

- **Through Open Messaging**

To initiate a benchmarking process aimed at evaluating the throughput capabilities of the system, the workload parametersdefined in the 'max-rate-1-topic-1-partition-5p-5c-100b.yaml' file play a crucial role. This workload specification encompasses various parameters, including the maximum rate, number of topics, partitions, as well as the configuration for producers (5p) and consumers (5c).

- **Through SBK**

SBK initiating a performance evaluation of a Redpanda instance with specific configurations for 5writers, 5readersnumber of topic is 1, 5partitions, 1replication factor, 100 bytesmessage size, duration, throughput, and broker address.

Fig. 6. Iteration1 Results Chart.

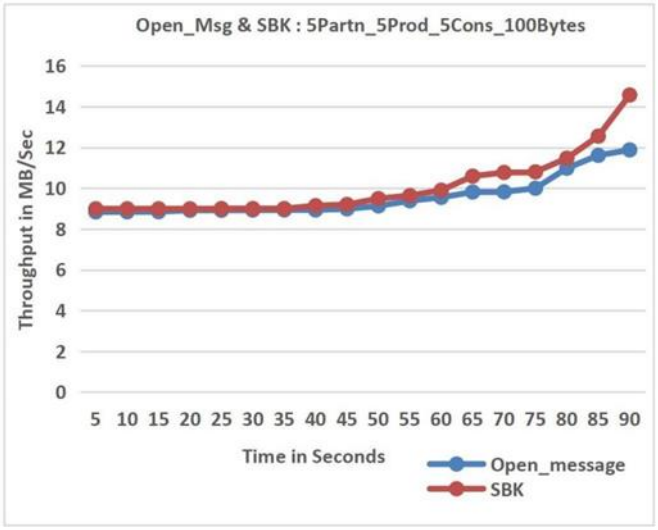

Fig. 7. Iteration2 Results Chart.

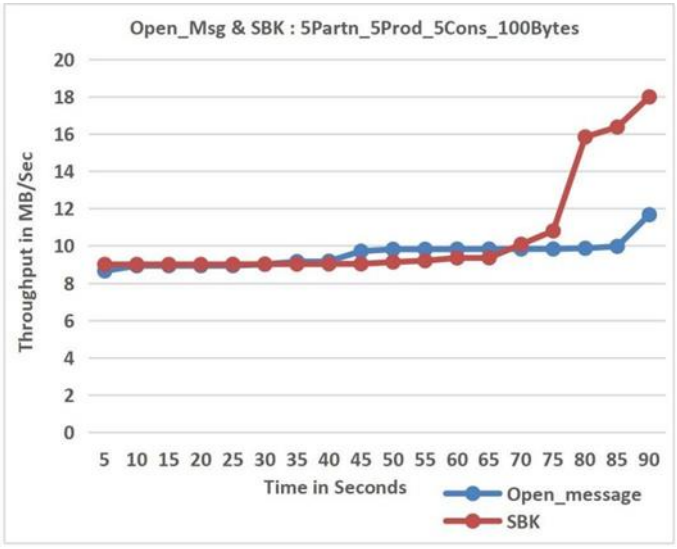

Fig. 8. Iteration3 Results Chart.

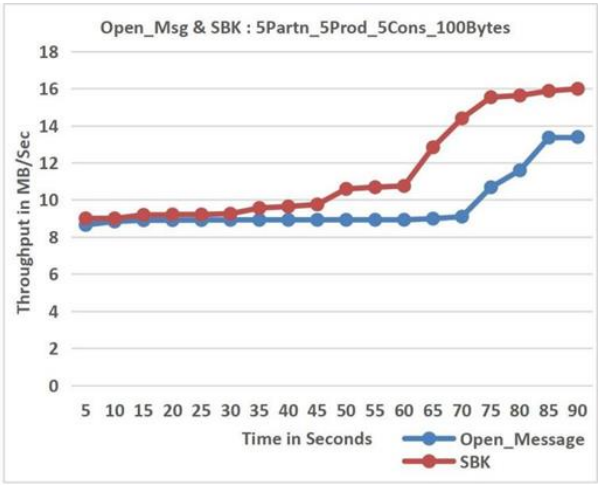

Fig. 9. Iteratio4 Results Chart.

Fig. 10. Iteration5 Results Chart.

Fig. 5-10 shows Multiple Producers Consumers performance Benchmarking, the performance benchmarking of redpanda for 5 producers and 5consumers for a workload 'max-rate-1-topic-5-partition-5p-5c-100b' On a single broker, a topic is divided into 5 partitions Every message has an offsetassigned to it that is with in the partitions which allows the consumers to read from all the 5 partitions in the same order. SBK Benchmarking results shows that the approximate maximum throughput of 18MB/sec, whereas open messaging gives the maximum throughput result of 16MB/sec hence SBK achieves the maximum throughput for record (data) size of 100Bytes compare to open messaging.

C. In a Single topic-1producer-1consumer model for eventsprocessing.

The actual numbers of events that can be processed vary depending on parameters like the size of message, the number of producers and consumers, the network bandwidth, the disk throughput, and the CPU resources available [24]. Whenever an event gets published to a topic, it's directly appended to the specific partition created for that topic. Subsequently, any consumer assigned to that particular topic-partition will consistently retrieve events in the exact sequence they were originally written. One producer write event to the topic by publishing messages into the queue. On the receiving end, oneconsumer subscribe to these events and retrieve the messages from the queue.

Event processing is typically done in real-time, single topic is a string which identifies the stream of events that the record belongs to. In a distributed streaming platform redpanda allows to publish and subscribe to stream of records [25]. Once the record is published, redpanda stores it in a partition within the topic. Here Single Producer publish an event by sending a record, by sending a key, value and a topic.Once it is published it stores it in a partition within the topic. Single Consumer can then subscribe to a topic and read eventsfrom a partition, and receive key, value and meta data associated with the event.

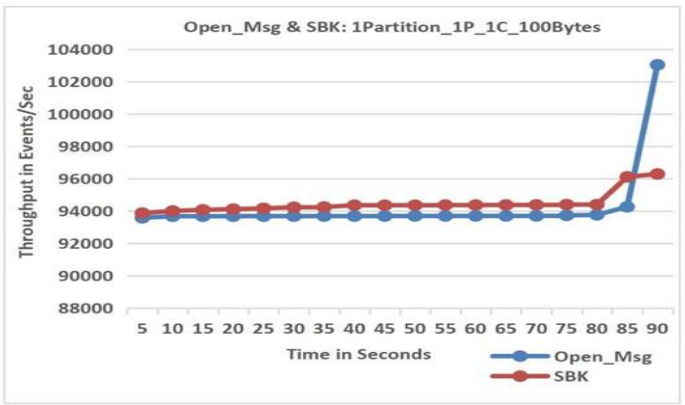

Fig. 11. Iteration1 Results Chart.

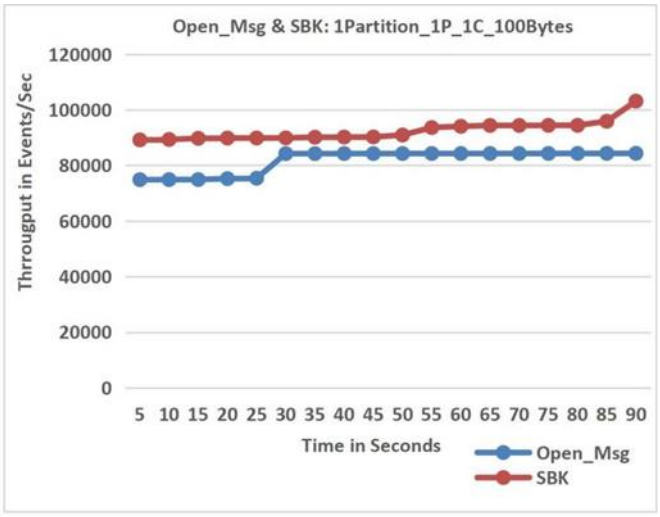

Fig. 12. Iteration2 Results Chart.

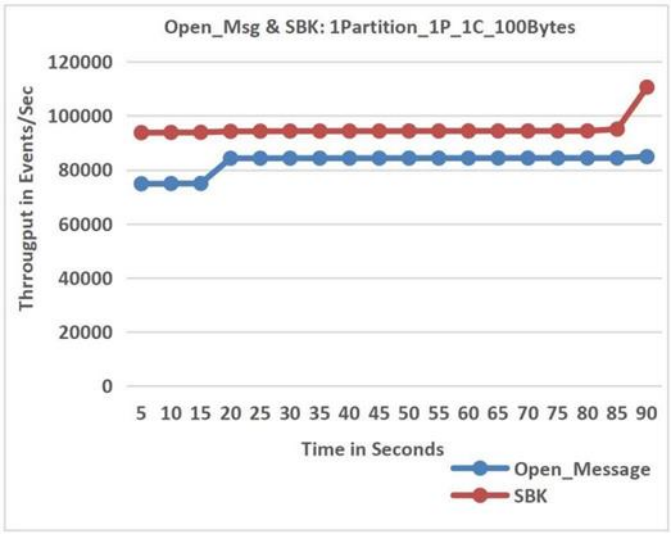

Fig. 13. Iteration3 Results Chart.

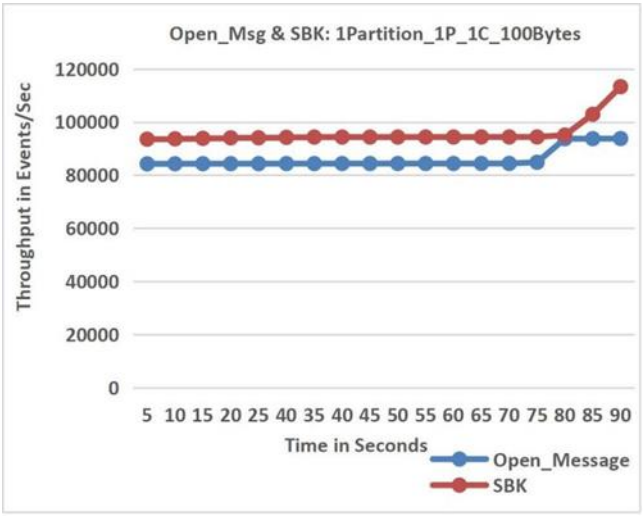

Fig. 14. Iteration4 Results Chart.

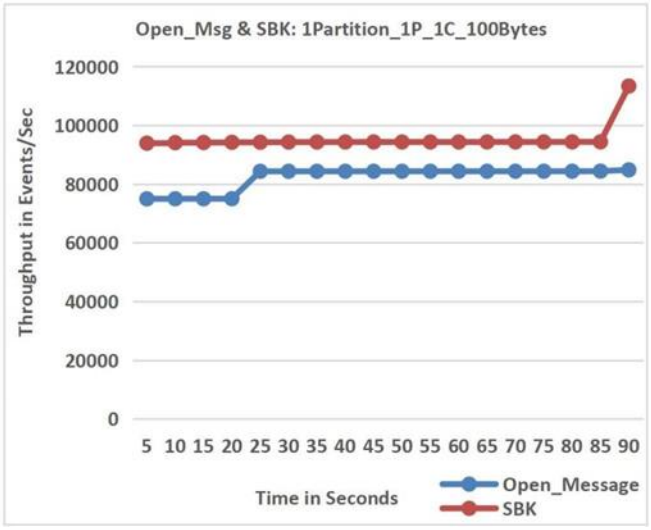

Fig. 15. Iteration5 Results Chart.

In a Single Producer-Consumer model for events processing, Figure 10 - 15 shows that the throughput for total number of events for read/write for the record size of 100 bytes for a given workload of 'max-rate-1-topic-1-partition-1p-1c-100b'. The graph shows that, in all the 5 iterations for a period of 3 minutes, SBK records more events than open messaging.

D. In a Single topic-5producer-5consumer model for eventsprocessing.

Single broker redpanda performance benchmarking is conducted to evaluate event processing for the record size of 100 bytes on a single topic with 5 partitions, 5-producer and 5 consumer setup. if the message size is small, say 100 bytes, and the system is configured for high throughput, Redpanda can handle up to 5 million events per second in the Single topic-5producer-5consumer model. However, the actual throughput will depend on various factors such as the number of partitions in the topic, the number of threads used by each producer and consumer, and the batch size of messages.

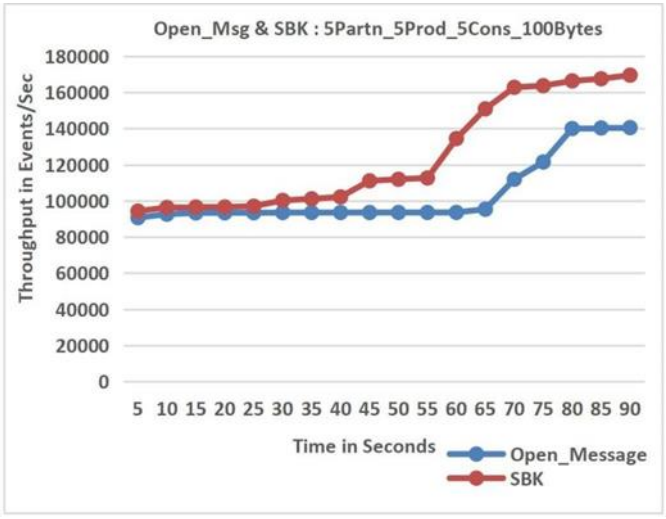

Fig. 16. Iteration1 Results Chart

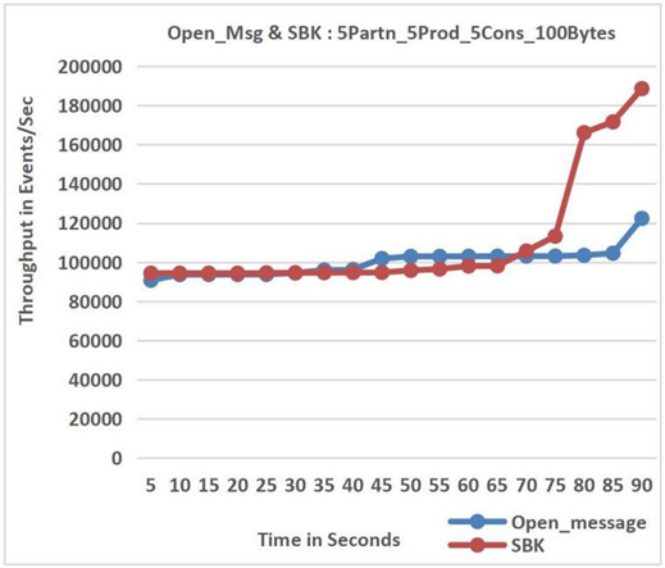

Fig. 18. Iteration3 Results Chart

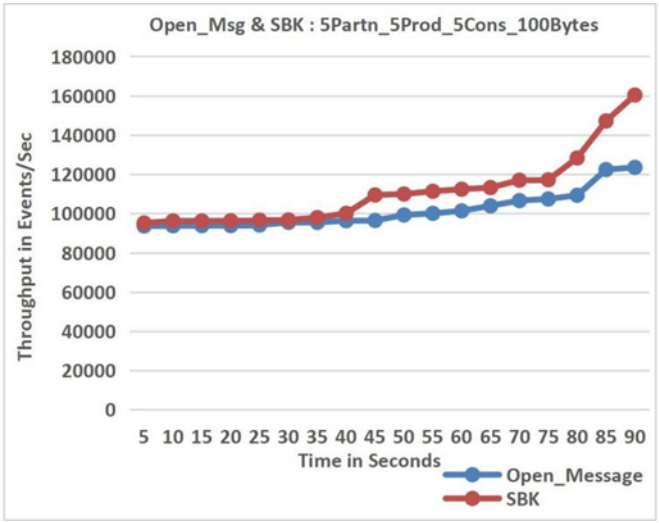

Fig. 19. Iteration4 Results Chart

Fig. 17. Iteration2 Results Chart

Fig. 20. Iteration5 Results Chart

Figure 15-20 illustrates the performance benchmark of five producers and five consumers. in the above 5 iterations for a period of 3 minutes, total number of events for read/write for the record size of 100 bytes for a given workload 'max-rate-1- topic-5-partition-5p-5c-100b', SBK shows maximum performance compared to open messaging, SBK achieves the maximum write/read events throughput than open messaging. Redpanda provides various tools for monitoring and measuring the performance of the system, which can help to determine the optimal configuration and performance metrics.

VI. CONCLUSION

This paper gives an experimental evaluation and analysis of two leading open source benchmarking framework to conduct performance benchmarking on redpanda's event processing system. The design of redpanda is to provide maximum performance on any data streaming workload, benchmarking redpanda with Open Messaging and SBK provides valuable insights into the performance and scalability of the data streaming platform. In our work, Redpanda is setup on bare metal with a commodity hardware, so that we may test its publish/subscribe communications. In our experiment For a specific Redpanda configuration, the maximum number of messages that can be produced/consumed using the Open messaging and SBK frameworks is being assessed. The performance benchmarking has been determined by 5 iterations of 4 experimental test scenarios. In each benchmarking results SBK performs well.

REFERENCES

[1] [Online].Available: Storage Benchmark Kit (SBK) :https://github.com/kmgowda/SBK , 2023.

[2] [Online].Available: SBK Releases: https://github.com/kmgowda/SBK/releases , 2023.

[3] Peter M. Chen and David A. Patterson, fellow, ieee, "Storage Performance-Metrics and Benchmarks", Proceedings of the IEEE, Vol. 81. No. 8, August 1993.

[4] Munegowda, K., Sanjay Kumar, N.V. (2022). "Design and Implementation of Storage Benchmark Kit". Emerging Researchi n Computing, Information, Communication and Applications. Lecture Notes in Electrical Engineering, vol 790. Springer,

Singapore. https://doi.org/10.1007/978-981-16-1342-5_5

[5] [Online].Available: Open Messaging Benchmark Website : https://openmessaging.cloud/docs/benchmark.

[6] [Online].Available: Apache Kafka website : https://kafka.apache.org/ , 2023.

[7] Neha Narkhede, Gwen Shapira and Todd Palino, "Kafka, The Definitive guide", O'reilly series, 1st edition, july 2017.

[8] [Online].Available: Redpanda website :https://docs.redpanda.com/docs/get-started/ 2023.

[9] [Online].Available: Redpanda website : https://redpanda.com/blog/redpanda-vs-kafka-

faster-safer/ 2023.

[10] T. Clark and B. S. Barn, "Event driven architecture modelling and simulation," Proceedings of 2011 IEEE 6th International Symposium on Service Oriented System (SOSE), Irvine, CA, USA, 2011, pp. 43-54, doi: 10.1109/SOSE.2011.6139091.

[11] S. Arnautov, P. Felber, C. Fetzer and B. Trach, "FFQ: A Fast Single-Producer/Multiple-Consumer Concurrent FIFO Queue," 2017 IEEE International Parallel and Distributed Processing Symposium (IPDPS), Orlando, FL, USA, 2017, pp. 907-916, doi: 10.1109/IPDPS.2017.41.

[12] David Bermbach , J¨orn Kuhlenkamp , Akon Dey , ArunmoezhiRamachandran , Alan Fekete , and Stefan Tai1, "BenchFoundry:A Benchmarking Framework for Cloud Storage Services" https://dbermbach.github.io/publications/2017-icsos- benchfoundry.pdf

[13] [Online].Available: Pravega website : Exploring Pravega

[14] G. van Dongen and D. Van den Poel, "Evaluation of Stream Processing Frameworks," in IEEE Transactions on Parallel and Distributed Systems, vol. 31, no. 8, pp. 1845-1858, 1 Aug. 2020, doi: 10.1109/TPDS.2020.2978480.

[15] H. Wang, J. Zhang, D. Zhang, S. Pumma and W. -c. Feng, "PaPar: A Parallel Data Partitioning Framework for Big Data Applications," 2017 IEEE International Parallel and Distributed Processing Symposium (IPDPS), Orlando, FL, USA,2017, pp. 605-614, doi: 10.1109/IPDPS.2017.119.

[16] Peter M. Chen and David A. Patterson, fellow, IEEE, "Storage Performance-Metrics and Benchmarks", Proceedings of the IEEE, Vol. 81. No. 8, August 1993.

[17] J. Kreps et al., "Kafka: A distributed messaging system for log processing", Proc. NetDB, pp. 1-7, 2011.

[18] [Online].Available: Kafka version 3.4.0 release :https://kafka.apache.org/downloads , 2023.

[19] Aaron B. Brown and Joseph L. Hellerstein, "An Approach to Benchmarking Configuration Complexity", Conference: Proceedings of the 11st ACM SIGOPS EuropeanWorkshop, Leuven, Belgium, September 19-22, 2004.

[20] Funke, F. et al. (2012). "Metrics for Measuring the Performance of the Mixed Workload CH-benCHmark". In: Nambiar, R., Poess, M. (eds) Topics in Performance Evaluation, Measurement and Characterization. TPCTC 2011. Lecture Notesin Computer Science, vol 7144. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-32627-1_2.

[21] T. P. Raptis and A. Passarella, "On Efficiently Partitioning a Topic in Apache Kafka," 2022 International Conference on Computer, Information and Telecommunication Systems (CITS), Piraeus,Greece,2022,pp.1-8, doi: 10.1109/CITS55221.2022.9832981

[22] [Online].Available: Open messaging source code : https://github.com/openmessaging/openmessaging-benchmark , 2023.

[23] [http://www.codecentric.de/de/m/kompetenzen/publikationen/stu dien/

[24] M. Astekin, S. Özcan and H. Sözer, "Incremental Analysis of Large-Scale System Logs for Anomaly Detection," 2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, 2019, pp. 2119-2127, doi: 10.1109/BigData47090.2019.9006593.

[25] T. P. Raptis and A. Passarella, "On Efficiently Partitioning a Topic in Apache Kafka," 2022 International Conference on Computer, Information and Telecommunication Systems (CITS), Piraeus, Greece, 2022, pp. 1-8, doi: 10.1109/CITS55221.2022.9832981.