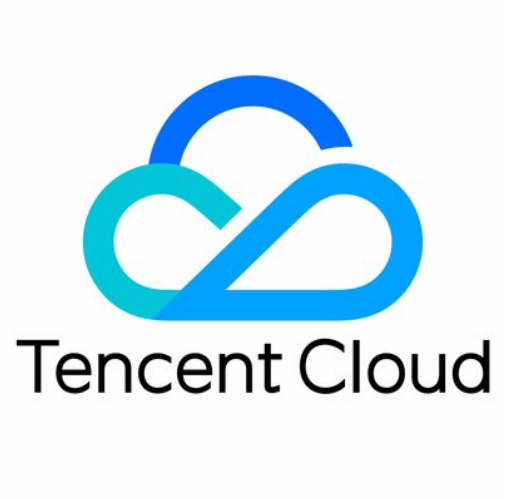
TDMQ for CKafka

Product Introduction

Product Documentation



Contents

Product Introduction
Overview

Last updated：2024-07-19 12:28:39

TDMQ for CKafka Overview

Based on the open-source Apache Kafka message queuing engine, TDMQ for CKafka provides high-throughput and highly scalable message queuing services. It is perfectly compatible with the APIs of Apache Kafka v0.9, v0.10, v1.1, v2.4, v2.8, and 3.2 and has greater advantages in terms of performance, scalability, business security, and OPS, allowing you to enjoy powerful features at low costs while eliminating tedious Ops work.

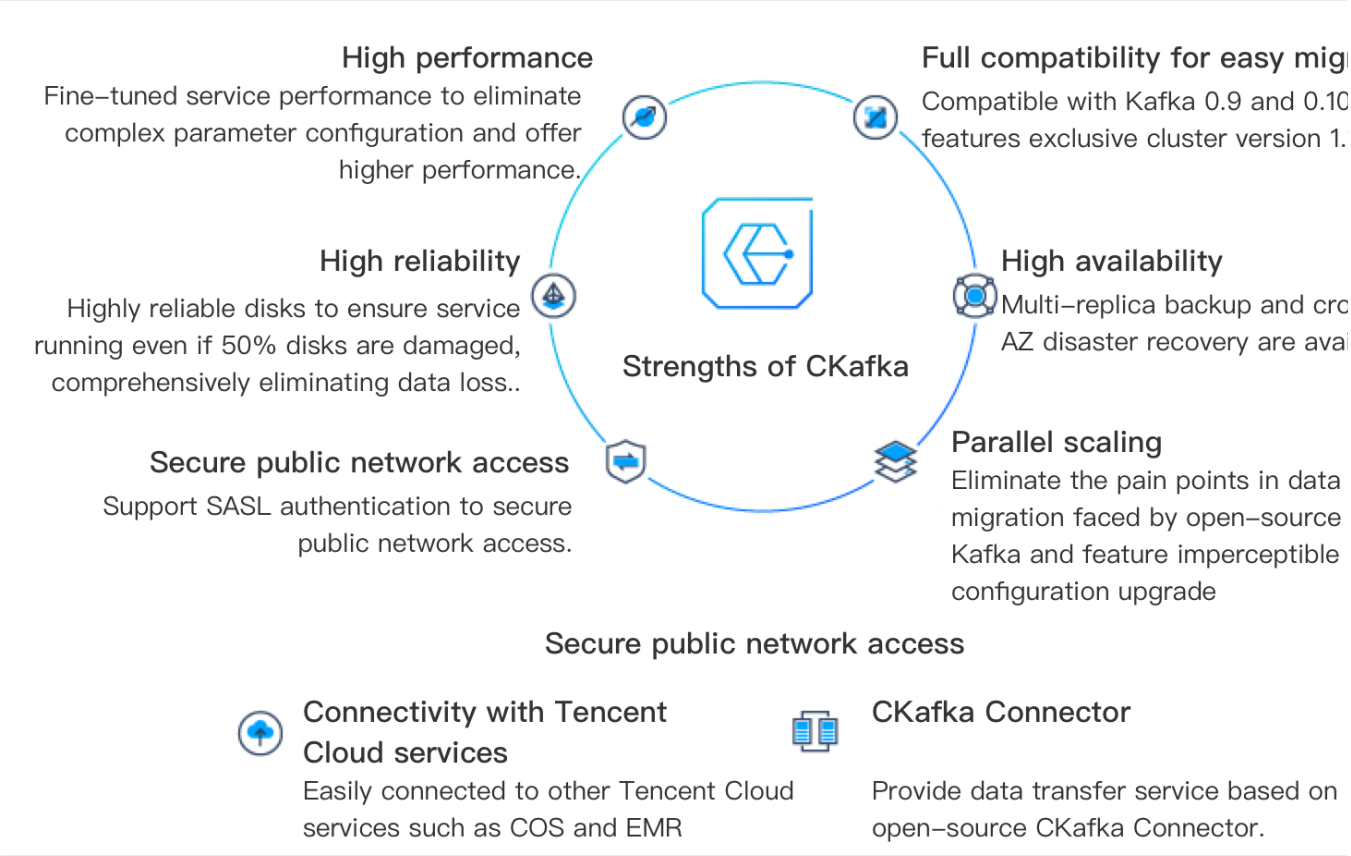
Features

Decoupling producers and consumers
TDMQ for CKafka effectively decouples the relationship between producers and consumers, allowing for independent scaling or modification of the processing flow between producers and consumers while ensuring the same API constraints.
Peak shifting and valley filling
TDMQ for CKafka can withstand access traffic surges instead of completely crashing due to sudden overwhelming requests, which effectively boosts system robustness.
Sequential read/write
TDMQ for CKafka ensures the orderliness of messages within a partition. Like most message queues, it guarantees that data is processed in sequence, greatly improving disk efficiency.
Async communication
In scenarios where immediate message processing is not required, TDMQ for CKafka provides an asynchronous message processing mechanism. During high traffic periods, messages are simply placed in the queue and processed later when the traffic decreases, alleviating system pressure.
Note:
TDMQ for CKafka supports private deployment. You can submit a ticket for consultation.

Strengths

Last updated：2024-07-19 12:30:20

This document describes the strengths of TDMQ for CKafka compared to Apache Kafka.



Strengths

Full compatibility with Apache Kafka and easy migration

TDMQ for CKafka is compatible with Kafka versions 0.9, 0.10, 1.1, 2.4, and 3.2.
TDMQ for CKafka is based on the existing code of the open-source Apace Kafka ecosystem. Without any changes to your existing project, you can migrate data to the cloud and enjoy the high-performance message queue services provided by Tencent Cloud

High performance

TDMQ for CKafka team has further fine-tuned the service performance to eliminate complex parameter configuration and offer higher performance.

TDMQ for CKafka offers UI-based upgrade and scaling, as well as a powerful IaaS layer that provides 10% to 20% more production capabilities than Apache Kafka.

High availability

Leveraging Tencent's years of experience in monitoring technologies, TDMQ for CKafka provides comprehensive monitoring on clusters. Additionally, our professional operations team is available 24/7 to handle alerts, ensuring the high availability
Custom multi-AZ deployments in the same region is supported to improve disaster recovery ability.

High reliability

The disks used are highly reliable, and the service will not be affected even if 50% disks are damaged.
There are 2 replicas by default, and 3 replicas can be supported. The more replicas, the higher the reliability.

Parallel scaling

The backend system automatically scales resources when cluster traffic and disk usage hit the alarm thresholds. The client side is unaffected during the process.
TDMQ for CKafka eliminates the pain points in data migration faced by open-source Kafka and features imperceptible configuration upgrade.

Data security

TDMQ for CKafka provides various security features such as authentication, authorization, root account, and sub-account, enabling enterprise-level security protection.
VPC: Access from more secure VPC is supported.
SASL authentication is supported to secure public network access.
Root account and sub-account: CAM features such as root account, sub-account, and collaborator are fully supported, enabling authorization between root account and sub-account as well as across organizational accounts.

Connectivity with Related Services

Integration with cloud services

TDMQ forCKafka can be easily connected to other Tencent Cloud services such as COS and EMR.

TDMQ for CKafka Connector

CKafka supports data transfer with open-source Kafka Connect, so that data can be exchanged between two Kafka clusters.

Use Cases

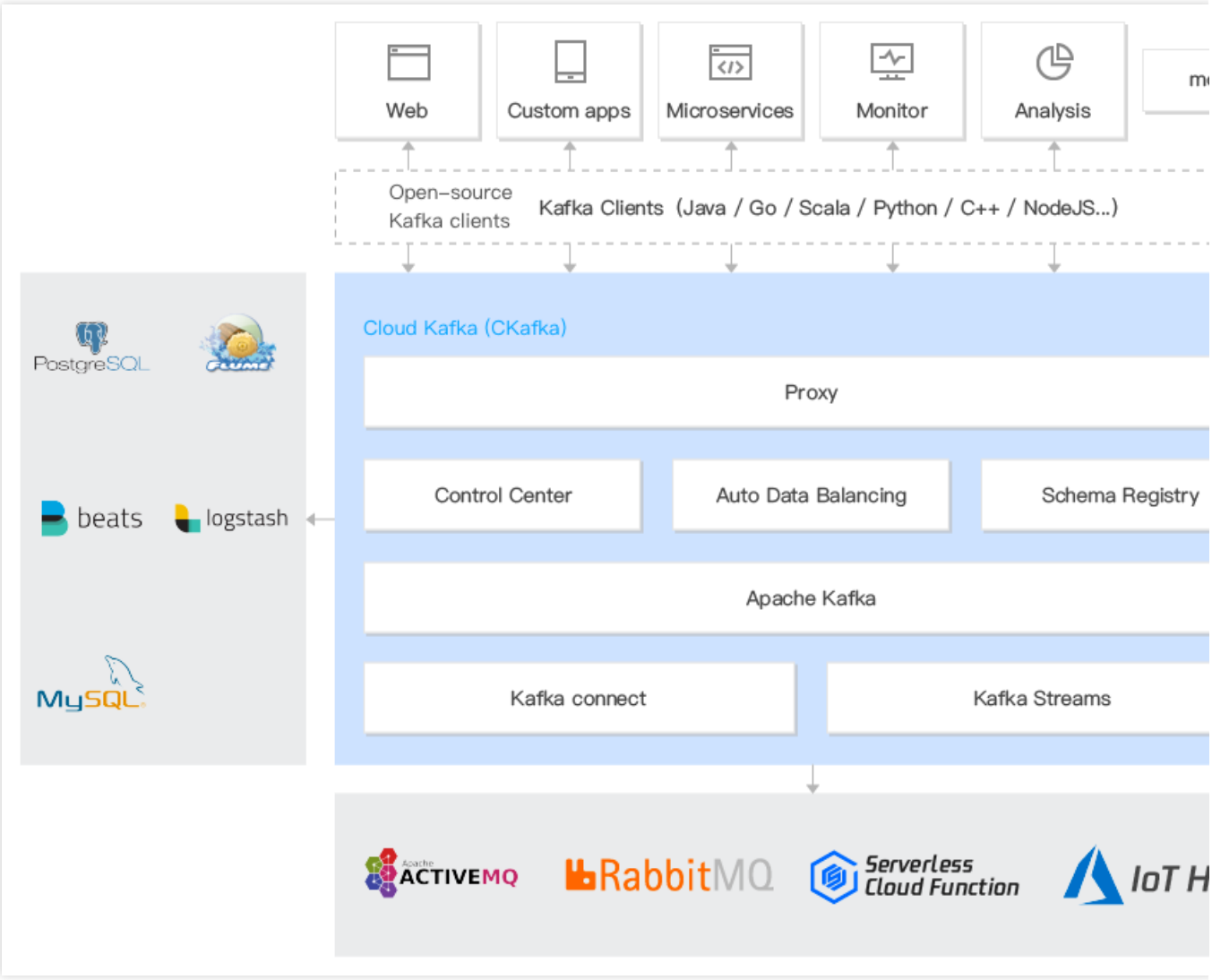
Last updated：2024-07-19 12:26:34

TDMQ for CKafka is widely used in big data scenarios, such as webpage tracking, behavior analysis, log aggregation, monitoring, streaming data processing, and online and offline data analysis.

You can simplify data integration in the following ways:

Import messages from TDMQ for CKafka into COS, Stream Compute Service, and other data warehouses.

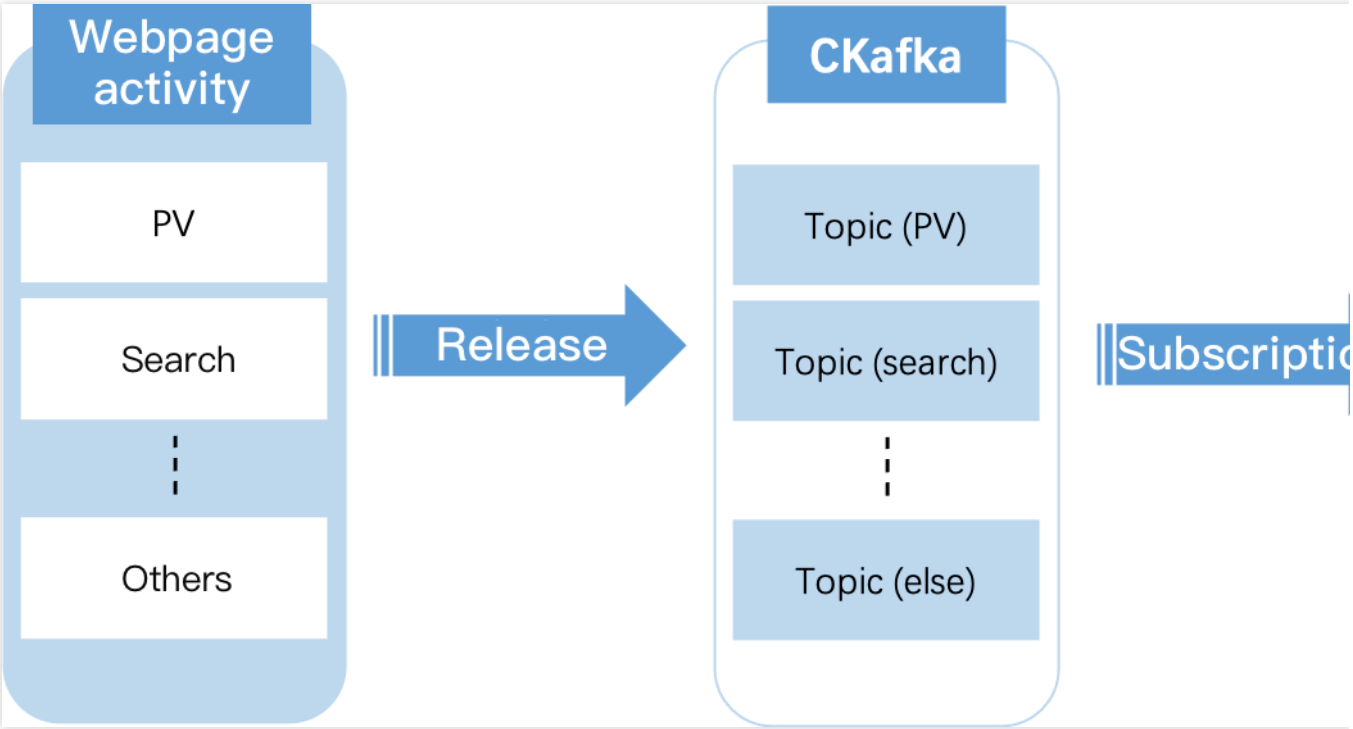
Connect with other Tencent cloud products using Serverless Cloud Functions triggers.



Website activity tracking

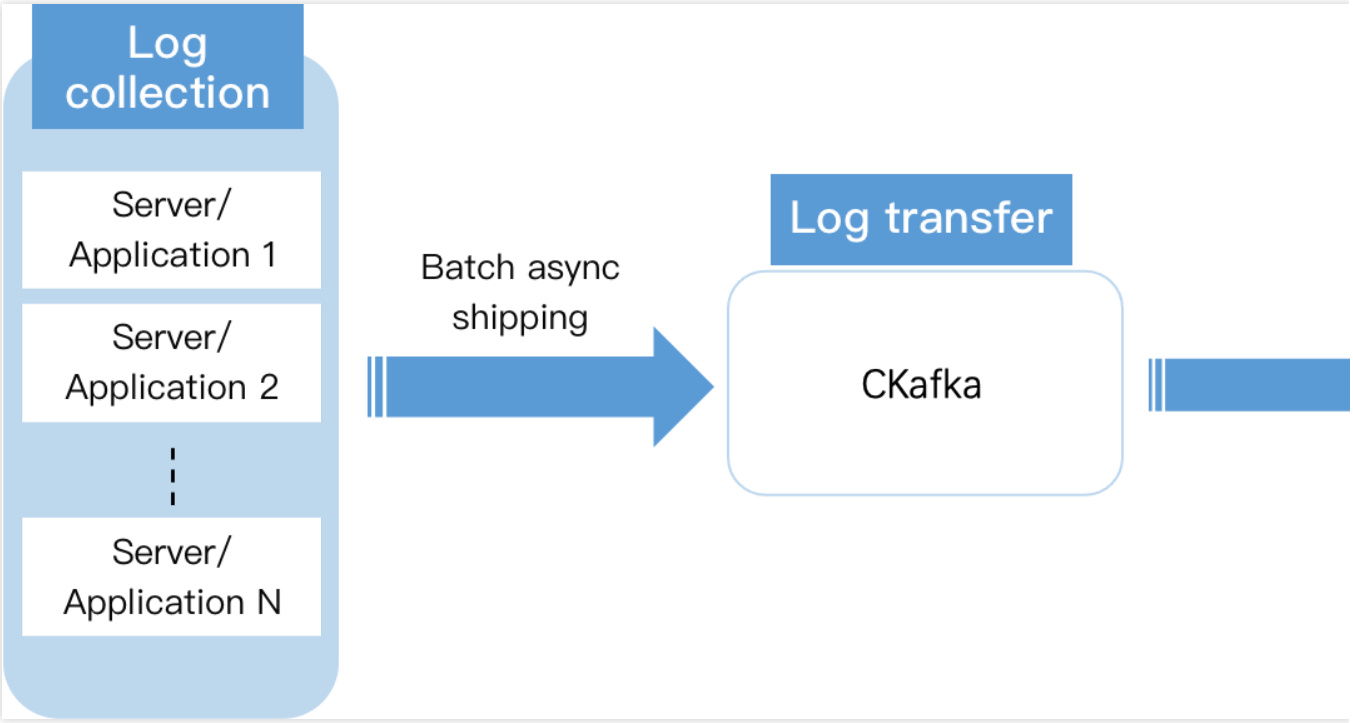
TDMQ for CKafka can process website activities such as PV, search, and other user behaviors in real time and then publishes them to topics by type. These information flows can be used for real-time monitoring or offline statistical analysis.

As a large amount of activity information is generated in each user's page views, website activity tracking requires a very high throughput. TDMQ for CKafka can perfectly meet the requirements of high throughput and offline processing.



Log aggregation

The low-latency processing capability of TDMQ for CKafka makes it easier to process (consume) distributed data from multiple data sources. Under the same performance conditions, TDMQ for CKafka provides more durable persistent storage and lower end-to-end latency than a centralized data aggregation system.
The above features make TDMQ for CKafka clusters an ideal log collection center. Multiple servers and applications can asynchronously send operation logs in batches to TDMQ for CKafka clusters instead of saving them locally or in a database. TDMQ for CKafka clusters can submit and compress messages in batches, making the performance overhead almost negligible for producers. Uers can use systematic storage and analysis systems such as Hadoop to analyze the pulled logs.

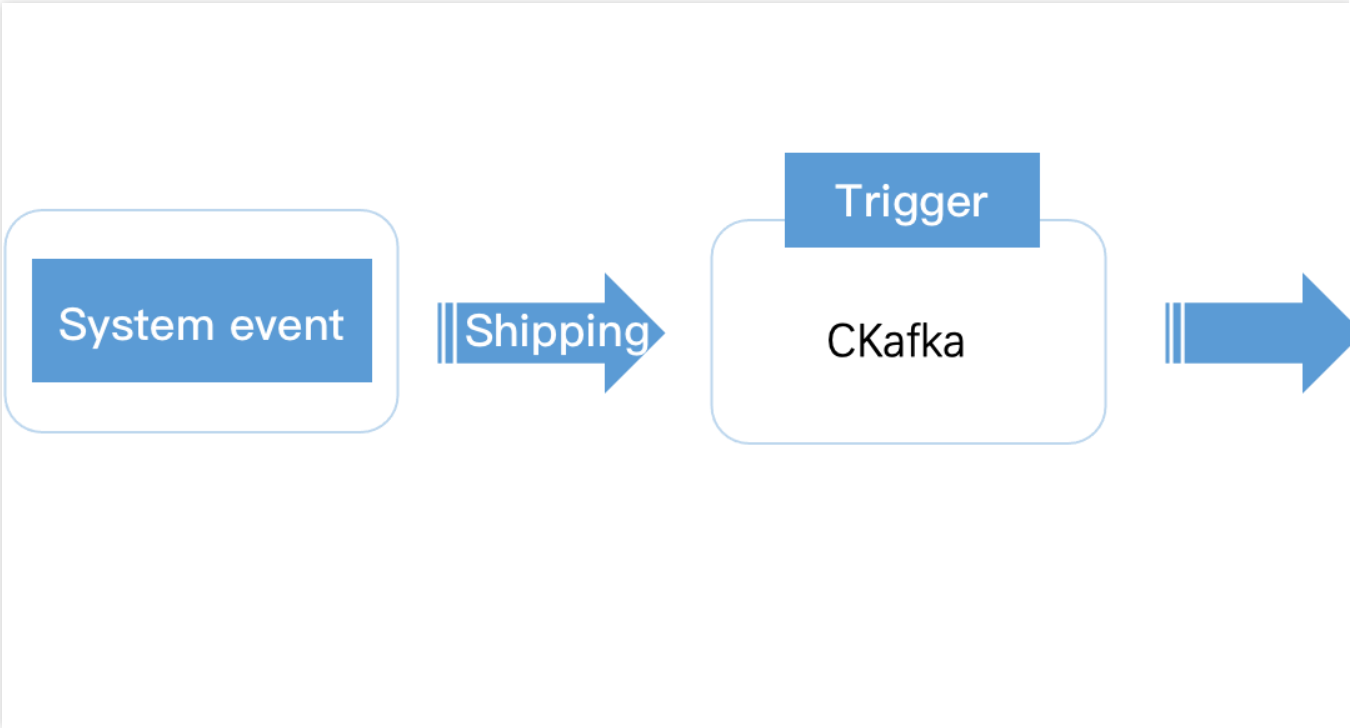


Big data

In some big data scenarios, a large amount of concurrent data needs to be processed and aggregated. This requires clusters to have excellent processing performance and high scalability. Moreover, TDMQ for CKafka clusters' data distribution mechanism, in terms of disk space allocation, message format processing, server selection, and data compression, also makes them suitable for handling high numbers of real-time messages and aggregating distributed application data, which facilitates system OPS.
TDMQ for CKafka clusters can better aggregate, process, and analyze offline and streaming data.

Serverless Cloud Functions triggers

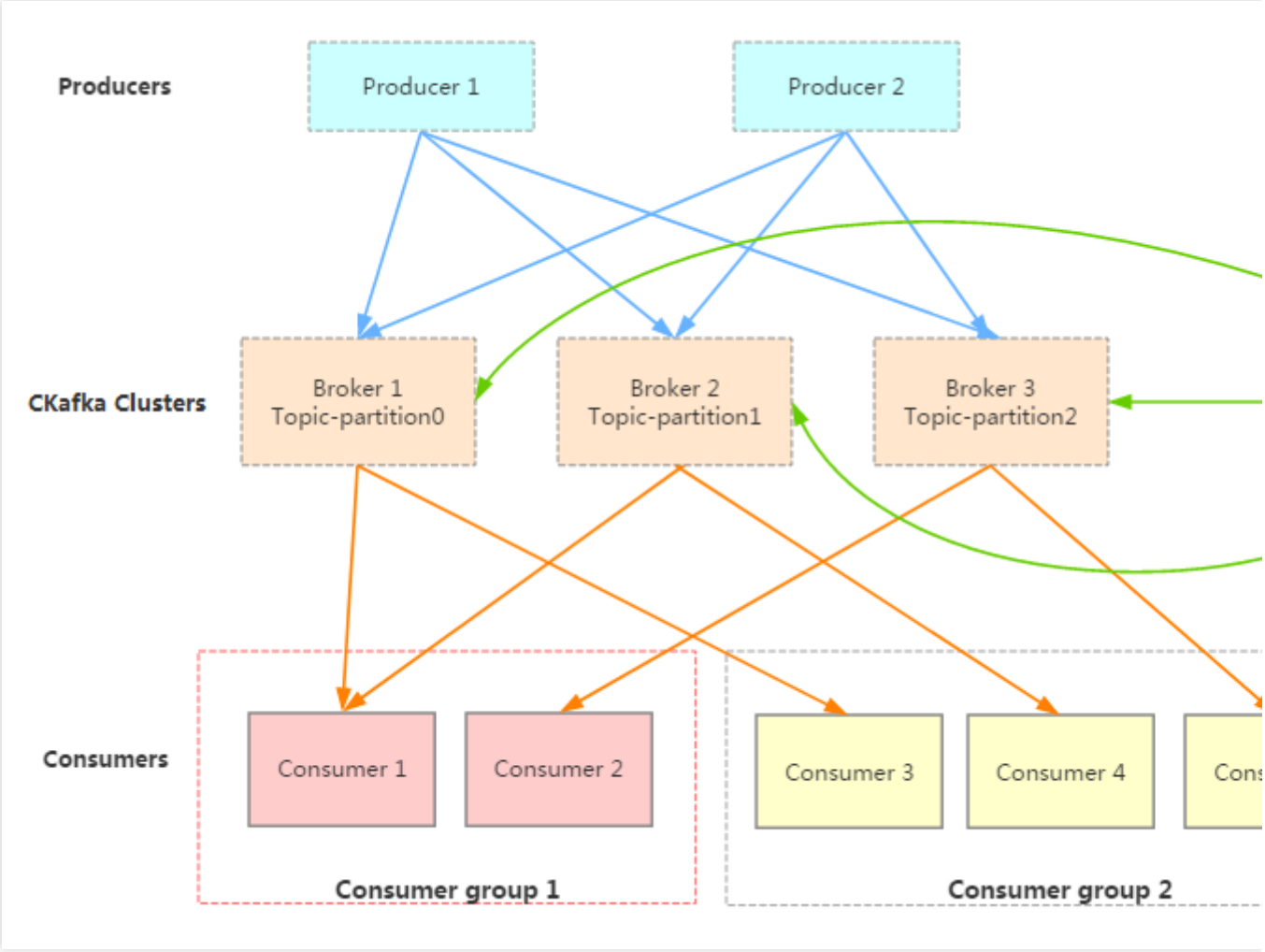
TDMQ for CKafka can be used as SCF triggers, and when a message is received, a function can be triggered and the message will be passed to the function as event content. For example, when CKafka triggers a function, the function can transform the message structure, filter the message contents, or deliver the message to Elasticsearch Service (ES).
Note:
For more information on the availability of SCF, see Service Level Agreement for SCF.



How It Works

Last updated：2024-07-19 12:22:21

The architecture of TDMQ for CKafka is as follows:



A producer could be a source of messages generated from web activities, service logs, and other similar information. It publishes messages to TDMQ CKafka's broker cluster using a push model.

The cluster uses Zookeeper to manage cluster configurations, perform leader elections, and handle fault tolerance.

Consumers are divided into several consumer groups. A consumer consumes messages from the broker using a pull mode.

For the advantages of TDMQ for CKafka over Apache Kafka, see Strengths.

High throughput

In TDMQ for CKafka, a huge amount of network data is permanently stored in disks and high numbers of disk files are sent over the network. The performance of this process directly impacts TDMQ for CKafka's overall throughput and is achieved through the following methods:

Efficient disk usage: Data is read and written sequentially on the disk to improve disk utilization.

Writing messages: Messages are written to the page cache and then flushed to disk by asynchronous threads.

Reading messages: Messages are sent directly from the page cache to the socket.

When the required data is not found in the page cache, disk I/O occurs, loading the messages from the disk into the page cache and then sending them directly from the socket.

Broker's zero copy mechanism: the sendfile system is called to send data directly from the page cache to the network.

Reducing network overhead

Data compression reduces the network load.

Batch processing mechanism: The producer writes data to the broker in batch, and the consumer pulls data from the broker in batch.

Data persistence

Data persistence is mainly implemented in TDMQ for CKafka through the following principles:

Partition storage distribution in topics

In TDMQ for CKafka's file storage system, a single topic can have multiple different partitions. Each partition corresponds to a physical folder that stores the messages and index files for that partition. For example, if two topics are created where topic 1 has 5 partitions and topic 2 has 10 partitions, then a total of 5 + 10 = 15 folders will be generated in the cluster.

File storage method in partition

A partition is physically composed of multiple segments of equal size. These segments are read and written sequentially, allowing for the quick deletion of expired segments and improving disk utilization.

Scale out

One topic can include multiple partitions distributed in one or more brokers.

One consumer can subscribe to one or more partitions.

The producer is responsible for evenly distributing messages to the corresponding partitions.

Messages in partitions are sequential.

Consumer group

TDMQ for CKafka does not delete consumed messages.
Any consumer must belong to a group.
Multiple consumers within the same consumer group do not consume the same partition simultaneously.
Different groups can consume the same message simultaneously, supporting both queue and publish-subscribe models.

Multiple replicas

The multi-replica design can enhance the system availability and reliability. Replicas are evenly distributed across the entire cluster using the following algorithm:
1. Sort all brokers (assuming there are n brokers) and the partitions to be allocated.
2. Assign the i-th partition to the (i mod n)-th broker.
3. Assign the j-th replica of the i-th partition to the ((i + j) mod n)-th broker.

Leader election mechanism

TDMQ for CKafka dynamically maintains a set of in-sync replicas (ISR) in ZooKeeper. All replicas in the ISR have caught up with the leader. Only members of the ISR can be elected as leaders.
If there are f + 1 replicas in ISR, a partition can tolerate the failure of f replicas while ensuring that committed messages are not lost.
If there are a total of 2f + 1 replicas (including the leader and followers), it is necessary to ensure that at least f + 1 replicas have successfully replicated the messages before the commit operation. To ensure the correct election of a new leader, no more than f replicas can fail.

Comparison with Apache Kafka

Last updated：2024-07-19 12:22:42

The performance comparison between TDMQ for CKafka and open-source Apache Kafka is as follows:

Feature	TDMQ for CKafka	Apache Kafka
Basic features	Supports adjusting topic parameter configurations and the number of topic partitions, sending messages, and resetting the consumption offset in the console; supports visual management for clusters, topics, and consumer groups.	Configuring Topic parameters and changing the number of partitions require command-line operations, making it difficult for business personnel to customize. It does not support sending messages or manually resetting consumption progress on the backend, which can be cumbersome and error-prone. Additionally, it needs to be paired with an open-source management system, which reduces ease of use.
Smart O&M	Supports automatic disk capacity scaling, quick diagnosis, smart inspection (with solutions provided), and smart disk configuration.	Not supported.
Isolation	Supports creating multiple instances based on one physical cluster; with usage limit based on bandwidth and disk capacity; topic-level traffic throttling; the configuration of multiple access points; separation of control and data planes; and logical data isolation between primary accounts.	Not supported.
Monitoring and alarms	Offers out-of-the-box deployment and monitoring solutions that are mature and standardized; supports multidimensional monitoring and alarming, metric sorting, and viewing the details of logs (such as exception event logs) in the console for troubleshooting.	Not supported.
High availability	Supports high-availability and multi-AZ deployment and offers mature failure recovery solutions; supports upgrading from single-AZ deployment to multi-AZ deployment as well as cross-region disaster recovery.	Supported, but the process is complicated.

Security compliance	Supports topic-level ACL access control and allows you to configure SASL password authentication and SSL certification in the console. It also provides control over operational permissions, integrates with CloudAudit for management operations, and ensures traceability.	Parameters need to be configured with command lines, which is complicated and error-prone; and it does not support traceable operations.
Others	TDMQ for CKafka provides version upgrade capabilities, making it convenient to seamlessly upgrade to a new version in case of bugs or security vulnerabilities in Apache Kafka.	Not supported.

Use Limits

Last updated：2024-07-19 12:12:26

This document outlines some limitations on metrics and performance in TDMQ for CKafka. Be mindful not to exceed these limits during use to avoid any issues.

Item	Description
Number of topics	The topic limit depends on product specifications. For details, see Billing Overview | Tencent Cloud.
Number of partitions	One topic can support up to 3,000 partitions. The instance-level partition limit includes replicas, which are typically 2 or 3. Reducing the number of partitions is not supported.
Partition throughput	In the case of `ack = 1`, the throughput of a single TDMQ for CKafka partition is between 30 and 60 MB/sec due to factors such as TDMQ for CKafka's partition architecture, business data size, and request frequency. When using `ack = -1` (strong consistency), **we recommend maintaining the throughput of a single TDMQ for CKafka partition between 10 and 20 MB/sec**. This helps ensure that the stability of request duration is not impacted by factors such as TDMQ for CKafka's partition architecture, business data size, and request frequency.
Duration	TDMQ for CKafka is a high-throughput, high-volume message queue and cannot guarantee low latency for every request. It is recommended to set the timeout as follows: For the producer, when ACK = 1, the timeout period defaults to **30s.** For the producer, when ACK = -1, the timeout period defaults to **60s.** The timeout period for the consumer is set to **60s**.
Number of consumer groups	For Standard Edition, the recommended number of instance-level consumer groups is up to 50. For Pro Edition, the recommended number of instance-level consumer groups is up to 200. You can submit a ticket to apply for more.
Instance	The region attribute of instances cannot be changed. The maximum number of client connections to a Standard Edition instance is 5,000, and to a Pro Edition instance, 10,000. When the limit is reached, the client cannot create more connections. If this maximum value is unreasonable for your actual business, you can submit a ticket to request an increase.
Version	Standard Edition: It is compatible with open-source versions 0.9, 0.10, and 1.1. v1.1 is installed by default. Customized versions are not supported. Pro Edition: It is compatible with open-source versions 0.9, 0.10, 1.1, 2.4, and 2.8.
Routes	An instance can create up to five routes, with only one of them being a public network

	route.
Public network bandwidth	TDMQ for CKafka provides a public network bandwidth of 3 Mbps for free by default. Pro Edition instances can upgrade public network bandwidth to 198 Mbps additionally.
Exposing ZooKeeper	Not supported.
Exposing underlying resources	It is not supported so as to avoid risks caused by user's operation.
Message size	It cannot exceed 12 MB; otherwise, messages will fail to be sent.
Tag	Each Tencent Cloud resource can have up to 50 tags.
Concurrent operations in the console	To prevent consistency issues from high concurrency in instance metadata operations, some interfaces are locked to limit requests. Currently, only 20 concurrent console requests per instance are allowed to ensure stability and success (including SDK API calls).

Note:

Due to TDMQ for CKafka's partition architecture, business data size, request frequency, and the stability of the physical layer, TDMQ for CKafka cannot guarantee that each request has low latency.

However, we strive to ensure the following two requirements:

The percentage of low-latency production for a single instance each month aligns with the instance's SLA.

The percentage of low-latency consumption for a single instance each month aligns with the instance's SLA.

The calculation formulas are as follows:

The percentage of low-latency production for a month = (Number of minutes in the month when 99.9th percentile latency production time per minute is 30 seconds or less / Total service minutes in the month) × 100%.

The percentage of low-latency consumption for a month = (Number of minutes in the month when 99.9th percentile latency consumption is 60 seconds or less / Total service minutes in the month) × 100%.

Regions and AZs

Last updated：2024-07-19 12:03:25

A region is the physical location of an IDC. Availability zone (AZ) refers to the physical IDC of Tencent Cloud in the same region with independent power supplies and network resources. For more information, see CVM-Regions and AZs

Supported regions

China

Region		Code
South China	Guangzhou	ap-guangzhou
	Shenzhen	ap-shenzhen
East China	Shanghai	ap-shanghai
	Nanjing	ap-nanjing
	Hangzhou	ap-hangzhou
North China	Beijing	ap-beijing
	Tianjin	ap-tianjin
Central China	Changsha	ap-changsha
Southwest China	Chengdu	ap-chengdu
	Chongqing	ap-chongqing
Hong Kong/Macao/Taiwan (China)	Taipei (China)	ap-taipei
	Hong Kong (China)	ap-hongkong

Other countries and regions

Region		Code
Southeast Asia	Singapore	ap-singapore
	Bangkok	ap-bangkok

	Jakarta	ap-jakarta
South Asia	Mumbai	ap-mumbai
Northeast Asia	Seoul	ap-seoul
	Tokyo	ap-tokyo
US West	Silicon Valley	na-siliconvalley
US East	Virginia	na-ashburn
North America	Toronto	na-toronto
South America	São Paulo	sa-saopaulo
Europe	Frankfurt	eu-frankfurt

Selection of Regions and AZs

When selecting a region and AZ, take the following into consideration:
The geographic locations of TDMQ for CKafka instances, your business, and your target users: We recommend that you choose the region closest to your customers when purchasing a TDMQ for CKafka instance to reduce latency and improve access speed.
Relationship between CKafka and other Tencent Cloud services:
When you select other Tencent Cloud services, we recommend that you try to locate them all in the same region and AZ to allow them to communicate with each other through the private network, reducing access latency and increasing access speed.
High availability and disaster recovery.
Even if you have just one VPC, we still recommend that you deploy your businesses in different AZs to prevent a single point of failure and enable cross-AZ disaster recovery.
There may be network latency among different AZs. We recommend that you assess your business requirements and find the optimal balance between high availability and low latency.
If you need access to TDMQ for CKafka instances in other countries or regions, we recommend that you select an instance in those other countries or regions. If you use a TDMQ for CKafka instance in China to access servers in other countries or regions, you may encounter much higher network latency.

Resource availability

The following table describes global, regional, and AZ-specific TDMQ for CKafka resources.

Resource	Resource ID Format (8-Digit String of Numbers and Letters)	Type	Description
User account	Unlimited	Globally unique	Users can use the same account to access Tencent Cloud resources from around the world.
TDMQ for CKafka instance	ckafka-xxxxxxxx	TDMQ for CKafka instances can only be used in a single AZ of a single region.	TDMQ for CKafka instances can only be created in a specific AZ.

Related Operations

Migrating instance to another AZ

CKafka supports instance migration across AZs in the same region. If you need this, submit a ticket for assistance.