# Distributed Message Service for Kafka

# Service Overview

**Issue** 01

**Date** 2024-11-11

HUAWEI

HUAWEI CLOUD COMPUTING TECHNOLOGIES CO., LTD.

## Trademarks and Permissions


and other Huawei trademarks are the property of Huawei Technologies Co., Ltd.
All other trademarks and trade names mentioned in this document are the property of their respective holders.

## Notice

The purchased products, services and features are stipulated by the contract made between Huawei Cloud and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

# Contents



# 1 What Is DMS for Kafka?

Apache Kafka is distributed message middleware that features high throughput, data persistence, horizontal scalability, and stream data processing. It adopts the publish-subscribe pattern and is widely used for log collection, data streaming, online/offline system analytics, and real-time monitoring.

Distributed Message Service (DMS) for Kafka is a message queuing service based on Apache Kafka. This service provides Kafka premium instances. The computing, storage, and bandwidth resources used by an instance are exclusively occupied by the user. You can apply for instances as required. The instances can be used right out of the box, taking off the deployment and O&M pressure for you so that you can focus on developing your services.

## Readers' Guide

This documentation introduces DMS for Kafka and its differences from Apache Kafka. You will learn about the detailed information about the specifications, console operations, and client access to instances of Huawei Cloud DMS for Kafka.

For more information about the basic knowledge of Kafka or technical details about creating and retrieving messages, please go to the **official Apache Kafka website**.

# 2 Product Advantages

Huawei Cloud DMS for Kafka provides easy-to-use message queuing based on Apache Kafka. Services can be quickly migrated to the cloud without any change, reducing maintenance and usage costs.

- Rapid deployment

  Simply set instance information on the DMS for Kafka console, submit your order, and a complete Kafka instance will be automatically created and deployed.

- Service migration without modifications

  DMS for Kafka is compatible with open-source Kafka APIs and supports all message processing functions of open-source Kafka.

  If your application services are developed based on open-source Kafka, you can easily migrate them to DMS for Kafka after specifying a few authentication configurations.

  NOTE

  Kafka instances are compatible with Apache Kafka v1.1.0, v2.7, and v3.x. Keep the client and server versions the same.

- Security

  Operations on Kafka instances are recorded and can be audited. Messages can be encrypted before transmission.

  In addition to Simple Authentication and Security Layer (SASL) authentication, Virtual Private Clouds (VPCs) and security groups also provide security controls on network access.

- Data reliability

  Kafka instances support data persistence and replication. Messages can be synchronously or asynchronously replicated between replicas and flushed to disk.

- High availability

  Kafka runs in clusters, enabling failover and fault tolerance so that services can run smoothly.

  Kafka instance brokers can be deployed across AZs to enhance service availability. Data is synchronized between different AZs based on Kafka's in-sync replica (ISR) mechanism. A topic must have multiple data copies and

distribute them across ISRs. When ISR replication is normal, the recovery point objective (RPO) is close to 0.

- Simple O&M

  Huawei Cloud provides a whole set of monitoring and alarm services, eliminating the need for 24/7 attendance. Kafka instance metrics are monitored and reported, including the number of partitions, topics, and accumulated messages. You can configure alarm rules and receive SMS or email notifications on how your services are running in real time.

- Massive accumulation and scaling

  Kafka features high scalability because it runs in a distributed system, or cluster. Users can configure up to 200 partitions for a topic. The storage space, broker quantity and flavor can be also expanded. This means that billions of messages can be accumulated, suitable for scenarios requiring high concurrency, high performance, and large-scale access.

- Flexible specifications

  You can customize the bandwidth and storage space for the instance and the number of partitions and replicas for topics in the instance.

# 3 Application Scenarios

Kafka is popular message-oriented middleware that features highly reliable, asynchronous message delivery. It is widely used for transmitting data between different systems in many industries, including enterprise application, payment, telecommunications, e-commerce, social networking, instant messaging, video, Internet of Things, and Internet of Vehicle.

## Asynchronous Communication

Non-core or less important messages are sent asynchronously to receiving systems, so that the main service process is not kept waiting for the results of other systems, allowing for faster responses.

For example, Kafka can be used to send a notification email and SMS message after a user has registered with a website, providing fast responses throughout the registration process.

**Figure 3-1** Serial registration and notification


**Figure 3-2** Asynchronous registration and notification using message queues


## Traffic Control

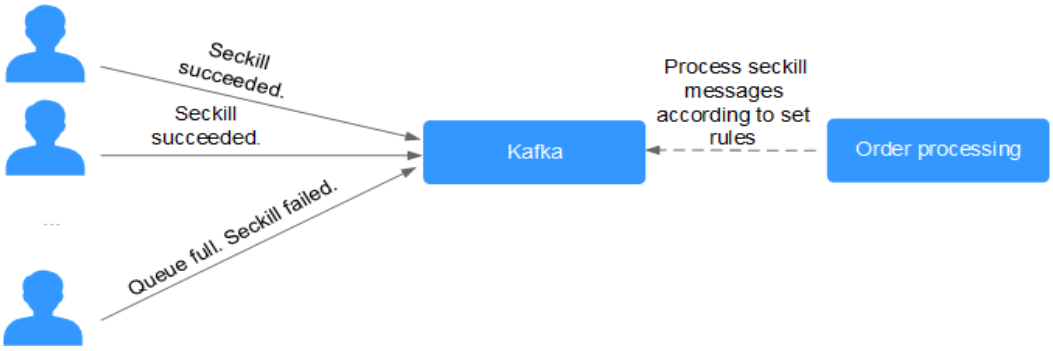
In e-commerce systems or large-scale websites, there is a processing capability gap between upstream and downstream systems. Traffic bursts from upstream systems with high processing capabilities may have a large impact on downstream systems with lower processing capabilities. For example, online sales promotions involve a huge amount of traffic flooding into e-commerce systems. Kafka

provides a three-day buffer by default for hundreds of millions of messages, such as orders and other information. In this way, message consumption systems can process the messages during off-peak periods.

In addition, flash sale traffic bursts originating from frontend systems can be handled with Kafka, keeping the backend systems from crashing.

**Figure 3-3** Traffic burst handling using Kafka




## Log Synchronization

In large-scale service systems, logs of different applications are collected for quick troubleshooting, full-link tracing, and real-time monitoring.

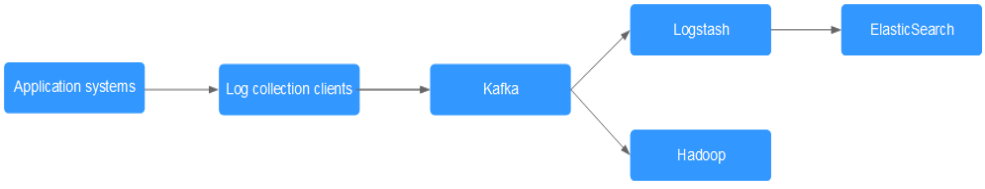
Kafka is originally designed for this scenario. Applications asynchronously send log messages to message queues over reliable transmission channels. Other components can read the log messages from message queues for further analysis, either in real time or offline. In addition, Kafka can collect key log information to monitor applications.

Log synchronization involves three major components: log collection clients, Kafka, and backend log processing applications.

1. The log collection clients collect log data from a user application service and asynchronously send the log data in batches to Kafka clients.

   Kafka clients receive and compress messages in batches. This only has a minor impact on the service performance.
2. Kafka persists logs.
3. Log processing applications, such as Logstash, subscribe to messages in Kafka and retrieve log messages from Kafka. Then, the messages are searched for by file search services or delivered to big data applications such as Hadoop for storage and analysis.

**Figure 3-4** Log synchronization process



> **NOTE**
>
> Logstash is for log analytics, Elasticsearch is for log search, and Hadoop is for big data analytics. They are all open-source tools.

# 4 Kafka Instance Specifications

## 4.1 Single-node Kafka Instances

### Instance Specifications

A single-node Kafka instance has one broker, is compatible with open-source Kafka 2.7, and is applicable to test scenarios. Do not use it for production services.

NOTE

For Kafka instances, the number of transactions per second (TPS) is the maximum number of messages that can be written per second. In the following table, transactions per second (TPS) are calculated assuming that the size of a message is 1 KB. The test scenario is private access in plaintext. The disk type is ultra-high I/O.

**Table 4-1** Single-node Kafka instance specifications

| Flavor | Brokers | Max. TPS per Broker | Max. Partitions per Broker | Recommended Consumer Groups per Broker | Max. Client Connections per Broker | Storage Space (GB) | Traffic Limit per Broker (MB/s) |
|---|---|---|---|---|---|---|---|
| kafka.2u4g.single.small | 1 | 20,000 | 100 | 15 | 2000 | 100–10,000 | 40 |
| kafka.2u4g.single | 1 | 30,000 | 250 | 20 | 2000 | 100–10,000 | 100 |

### Storage Space Estimation

Kafka instances can store messages in multiple replicas. The storage space is consumed by message replicas, logs, and metadata. When creating an instance, specify its storage space based on the expected service message size, the number of replicas, and reserved disk space. Each Kafka broker reserves 33 GB disk space for storing logs and metadata.

For example, if the expected service message size is 100 GB, the number of replicas is 2, and the number of brokers is 1, the disk size should be at least 233 GB (100 GB × 2 + 33 GB × 1).

The storage space can be expanded as your service grows.

### Topic Quantity Calculation

There are limits on the topic quantity and the aggregate number of partitions in the topics. When the partition quantity limit is reached, you can no longer create topics.

The number of topics is related to the maximum number of partitions allowed and the specified number of partitions in each topic (see **Table 4-1**).

**The maximum number of partitions allowed for an instance with kafka.2u4g.single is 250.**

- If the number of partitions of each topic in the instance is 2, the maximum number of topics is 250/2 = 125.
- If the number of partitions of each topic in the instance is 1, the maximum number of topics is 250/1 = 250.

## 4.2 Cluster Kafka Instances

### Instance Specifications

A cluster Kafka instance has three or more brokers, and is compatible with open-source Kafka 1.1.0, 2.7, and 3.x.

NOTE

For Kafka instances, the number of transactions per second (TPS) is the maximum number of messages that can be written per second. In the following table, transactions per second (TPS) are calculated assuming that the size of a message is 1 KB. The test scenario is private access in plaintext. The disk type is ultra-high I/O. For more information about TPS performance, see **Kafka Instance TPS**.

**Table 4-2** Cluster Kafka instance specifications

| Flavor | Brokers | Maximum TPS per Broker | Maximum Partitions per Broker | Recommended Consumer Groups per Broker | Maximum Client Connections per Broker | Storage Space (GB) | Traffic per Broker (MB/s) |
|---|---|---|---|---|---|---|---|
| kafka.2u4g.cluster.small | 3–30 | 20,000 | 100 | 15 | 2000 | 300–300,000 | 40 |
| kafka.2u4g.cluster | 3–30 | 30,000 | 250 | 20 | 2000 | 300–300,000 | 100 |
| kafka.4u8g.cluster | 3–30 | 100,000 | 500 | 100 | 4000 | 300–600,000 | 200 |
| kafka.8u16g.cluster | 3–50 | 150,000 | 1000 | 150 | 4000 | 300–1,500,000 | 375 |
| kafka.12u24g.cluster | 3–50 | 200,000 | 1500 | 200 | 4000 | 300–1,500,000 | 625 |
| kafka.16u32g.cluster | 3–50 | 250,000 | 2000 | 200 | 4000 | 300–1,500,000 | 750 |

## Instance Specifications and Network Bandwidth

The network bandwidth of a Kafka instance consists of the following:

1. Network bandwidth used by the instance brokers
2. Bandwidth of the disk used by the instance brokers. For details, see **Disk Types and Performance**.

Note:

- By default, Kafka tests are performed in the tail read scenario (that is, only the latest production data is consumed) instead of the cold read scenario (that is, historical data is consumed from the beginning).
- The bandwidth of an instance with an old flavor (such as 100 MB/s) is the total network bandwidth of the instance's all brokers.

**Traffic calculation of instances with new flavors (such as kafka.2u4g.cluster) is described as follows:**

- The read/write ratio is 1:1.
- The default number of topic replicas is 3.
- Total network traffic = Traffic per broker x Broker quantity
- Total instance traffic = Service traffic + Data replication traffic between brokers

Assume that the current flavor is kafka.2u4g.cluster, the traffic per broker is 100 MB/s, and the number of brokers is 3. What are the total network traffic, maximum read traffic, and maximum write traffic of the instance?

1. Total network traffic = Traffic per broker x Broker quantity = 100 MB/s x 3 = 300 MB/s
2. Maximum read traffic = Total instance network traffic/Default number of replicas/2 = 300 MB/s/3/2= 50 MB/s
3. Maximum write traffic = Total instance network traffic/Default number of replicas/2 = 300 MB/s/3/2 = 50 MB/s

## Mapping Between Old and New Flavors

Table 4-3 compares the old and new Kafka instance flavors.

**Table 4-3** Mapping between old and new Kafka instance flavors

| Old Flavor | | New Flavor | |
|---|---|---|---|
| Flavor | Total Instance Network Traffic | Flavor | Total Instance Network Traffic |
| 100 MB/s | 100 MB/s | kafka.2u4g.cluster.small * 3 | 120 MB/s |
| 300 MB/s | 300 MB/s | kafka.2u4g.cluster * 3 | 300 MB/s |
| 600 MB/s | 600 MB/s | kafka.4u8g.cluster * 3 | 600 MB/s |
| 1200 MB/s | 1200 MB/s | kafka.4u8g.cluster * 6 | 1250 MB/s |

Instances with new flavors have the following features:

- Better performance and cost effectiveness: They use exclusive resources (except for kafka.2u4g.cluster.small). By contrast, old flavors use non-exclusive resources. If the load is heavy, resources conflicts will occur.
- Latest functions, for example, reassigning partitions, changing the SSL setting, and viewing rebalancing logs.
- Flexible flavor changes: For example, you can increase or decrease the broker flavor.
- Flexible disk capacity: Only related to the broker quantity, and not to the flavor.
- More specification options: A wider range of combinations of broker flavor (over 10,000 MB/s) and quantity are available.

- More disk type options: General Purpose SSD and Extreme SSD are now available, in addition to the original disk types.

## Flavor Selection

- kafka.2u4g.cluster.small with 3 brokers

  Recommended for up to 6000 client connections, 45 consumer groups, and 60,000 TPS

- kafka.2u4g.cluster with 3 brokers

  Recommended for up to 6000 client connections, 60 consumer groups, and 90,000 TPS

- kafka.4u8g.cluster with 3 brokers

  Recommended for up to 12,000 client connections, 300 consumer groups, and 300,000 TPS

- kafka.8u16g.cluster with 3 brokers

  Recommended for up to 12,000 client connections, 450 consumer groups, and 450,000 TPS

- kafka.12u24g.cluster with 3 brokers

  Recommended for up to 12,000 client connections, 600 consumer groups, and 600,000 TPS

- kafka.16u32g.cluster with 3 brokers

  Recommended for up to 12,000 client connections, 600 consumer groups, and 750,000 TPS

## Storage Space Selection

Kafka instances can store messages in multiple replicas. The storage space is consumed by message replicas, logs, and metadata. When creating an instance, specify its storage space based on the expected service message size, the number of replicas, and reserved disk space. Each Kafka broker reserves 33 GB disk space for storing logs and metadata.

For example, if the expected service message size is 100 GB, the number of replicas is 2, and the number of brokers is 3, the disk size should be at least 299 GB (100 GB x 2 + 33 GB x 3).

The storage space can be expanded as your service grows.

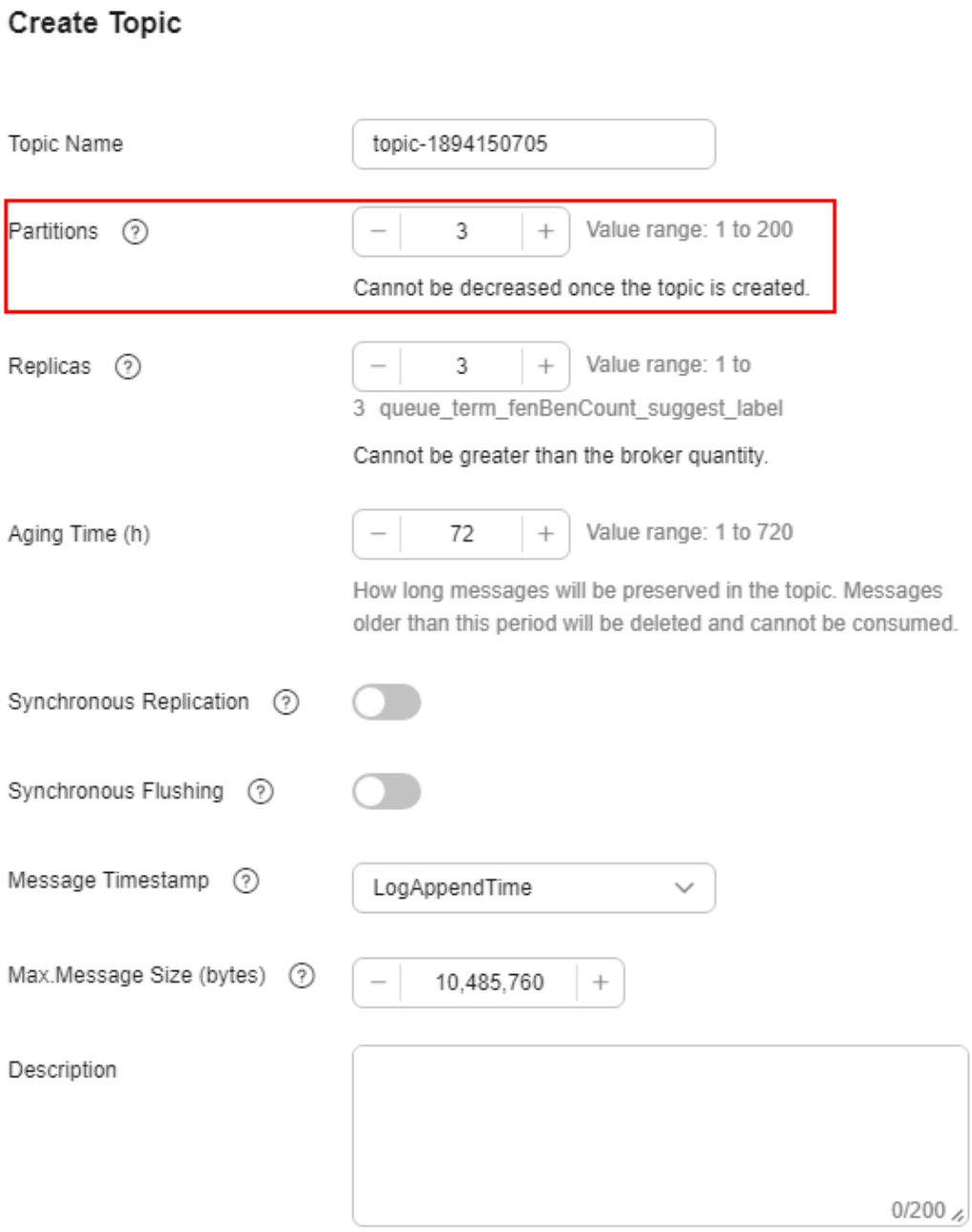
## Topic Quantity

There are limits on the topic quantity and the aggregate number of partitions in the topics. When the partition quantity limit is reached, you can no longer create topics.

The number of topics is related to the maximum number of partitions allowed (see **Figure 4-1**) and the specified number of partitions in each topic (see **Table 4-2**).

**Figure 4-1** Setting the number of partitions




**The maximum number of partitions allowed for an instance with kafka.2u4g.cluster and 3 brokers is 750.**

- If the number of partitions of each topic in the instance is 3, the maximum number of topics is 750/3 = 250.
- If the number of partitions of each topic in the instance is 1, the maximum number of topics is 750/1 = 750.

# 5 Comparing Single-node and Cluster Kafka Instances

A single-node Kafka instance has only one broker. These instances do not guarantee performance or reliability and are for trial use or testing only. In the production environment, use cluster instances.

Table 5-1 compares single-node and cluster instances by features and functions.

**Table 5-1** Comparing single-node and cluster instances

| Item | Single-node | Cluster |
| --- | --- | --- |
| Version | 2.7 | 1.1.0, 2.7, and 3.x |
| AZ | Single | 1, or 3 or more |
| Brokers | 1 | 3 and more |
| Access mode | Plaintext access | Plaintext and ciphertext access |
| Modifying instance specifications | × | √ |
| Resetting Kafka password | × | √ |
| Viewing disk usage | × | √ |
| Reassigning partitions | × | √ |
| Configuring topic permissions | × | √ |
| Managing users | × | √ |
| Viewing rebalancing logs | × | √ |
| Smart Connect | × | √ |
| Managing Kafka quotas | × | √ |

| Item | Single-node | Cluster |
| --- | --- | --- |
| Modifying configuration parameters | × | √ |

# 6 Comparing Kafka, RabbitMQ, and RocketMQ

**Table 6-1** Functions

| Feature | RocketMQ | Kafka | RabbitMQ |
| --- | --- | --- | --- |
| Priority queue | Not supported | Not supported | • V3.8.35: Supported. It is recommended that the priority be set to 0–10.<br>• AMQP-0-9-1: Supported. Set the priority to 1–9. |
| Delayed queue | Supported | Not supported | • V3.8.35: Not supported.<br>• AMQP-0-9-1: Supported. |
| Dead letter queue | Supported | Not supported | Supported |
| Message retry | Supported | Not supported | • V3.8.35: Not supported.<br>• AMQP-0-9-1: Supported. |
| Retrieval mode | Pull-based and push-based | Pull-based | Pull-based and push-based |
| Message broadcasting | Supported | Supported | Supported |

| Feature | RocketMQ | Kafka | RabbitMQ |
| --- | --- | --- | --- |
| Message tracking | Supported | Supports offset and timestamp tracking. | • V3.8.35: Not supported. Once a message retrieval has been acknowledged, RabbitMQ will be notified that the message can be deleted.<br>• AMQP-0-9-1: Supported. |
| Message accumulation | Supported | Supports higher accumulation performance than RabbitMQ thanks to high throughput. | Supported |
| Persistence | Supported | Supported | Supported |
| Message tracing | Supported | Not supported | • V3.8.35: Not supported.<br>• AMQP-0-9-1: Supported. |
| Message filtering | Supported | Supported | • V3.8.35: Not supported, but can be encapsulated.<br>• AMQP-0-9-1: Supported. |
| Multi-tenancy | Supported | Supported | Supported |
| Multi-protocol | Compatible with RocketMQ. | Only supports Apache Kafka. | RabbitMQ is based on AMQP. |
| Multi-language | Supports clients in multiple programming languages. | Kafka is written in Scala and Java and supports clients in multiple programming languages. | Supports clients in multiple programming languages. |
| Throttling | RocketMQ 5.x supports traffic control based on instance specifications. | Supports throttling on producer or consumer clients, users, and topics. | Supports credit-based throttling on producers, a mechanism that triggers protection from within. |

| Feature | RocketMQ | Kafka | RabbitMQ |
| --- | --- | --- | --- |
| Ordered message delivery | Message order is maintained within a queue. | Supports partition-level FIFO. | Supports FIFO only for single-threaded message queuing without advanced features such as delayed queues or priority queues. |
| Security | Supports SSL authentication. | Supports SSL and SASL authentication and read/write permissions control. | • 3.8.35: SSL authentication is supported.<br>• AMQP-0-9-1: ACL is supported. |
| Transactional messages | Supported | Supported | Supported |

# 7 Comparing DMS for Kafka and Open-Source Kafka

DMS for Kafka is compatible with open-source Kafka and has customized and enhanced Kafka features. In addition to the advantages of open-source Kafka, DMS for Kafka provides more reliable and useful features.

**Table 7-1** Differences between DMS for Kafka and open-source Kafka

| Category | Item | DMS for Kafka | Open-source Kafka |
| --- | --- | --- | --- |
| Ease of use | Readily available | Instances can be created intuitively within minutes and used right out of the box with visualized operations and real-time monitoring. | Preparing server resources and installing and configuring the software is time-consuming and prone to mistakes. |
| | APIs | Instances can be managed easily by calling RESTful APIs. | N/A |
| Costs | On-demand use | Multiple specifications are available to suit different needs. The instance broker quantity, broker flavor, and disk space can be increased with a few clicks. | Expenses are incurred for setting up a message service and occupying underlying resources. |
| | Fully managed | Services are readily available without requiring additional hardware resources or expenses. | Users must prepare hardware resources and set up the service by themselves, and bear high usage and maintenance costs. |

| Category | Item | DMS for Kafka | Open-source Kafka |
| --- | --- | --- | --- |
| Proven success | Mature | DMS has been deployed in many Huawei Cloud products and proven successful in large e-commerce events. It is also used in the clouds of carrier-grade customers across the world, and meets strict carrier-grade reliability standards. DMS closely follows up with community updates to continuously fix known open-source vulnerabilities and add support for new features. | Using open-source software requires lengthy self-development and verification and has had few successful cases. |
| | Feature-rich | While maintaining 100% open-source compatibility, DMS further optimizes open-source code to improve performance and reliability, and provides message querying, migration, and many other features. | Functionality is limited and requires self-development. |
| Reliability | Highly available | DMS supports cross-AZ deployment to improve reliability. In addition, automatic fault detection and alarms ensure reliable operations of key services. | High availability requires self-development or open-source code implementation, which are costly and cannot guarantee reliability. |
| | Simple O&M | O&M is entirely transparent to tenants with a full set of monitoring and alarm functions. O&M personnel will be informed of any exceptions, eliminating the need for 24/7 attending. | Users need to develop and optimize O&M functions, especially alarm notification functions. Otherwise, manual attendance is required. |
| | Secure | DMS uses VPC isolation and SSL channel encryption. | Security must be hardened by users themselves. |

# 8 Security

## 8.1 Shared Responsibilities

Huawei guarantees that its commitment to cyber security will never be outweighed by the consideration of commercial interests. To cope with emerging cloud security challenges and pervasive cloud security threats and attacks, Huawei Cloud builds a comprehensive cloud service security assurance system for different regions and industries based on Huawei's unique software and hardware advantages, laws, regulations, industry standards, and security ecosystem.

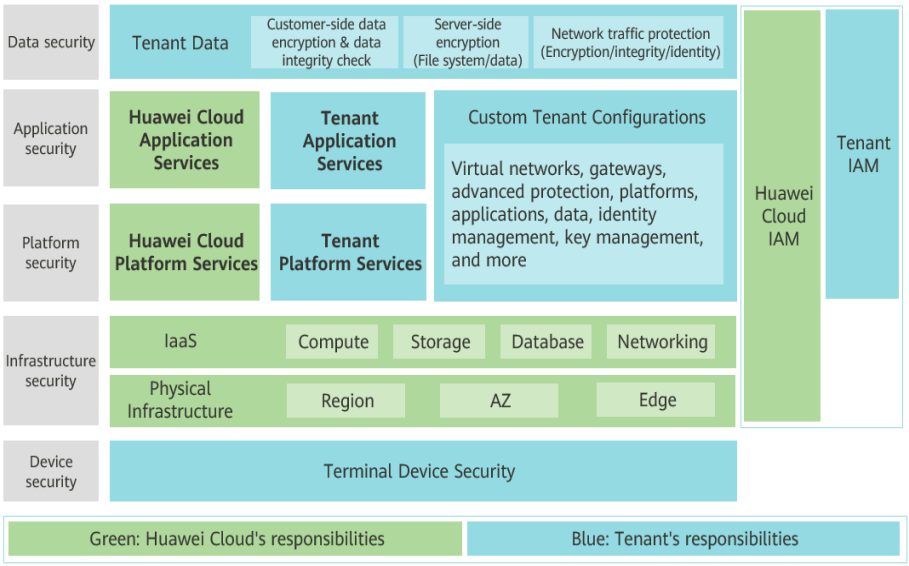
**Figure 8-1** illustrates the responsibilities shared by Huawei Cloud and users.

- **Huawei Cloud**: Ensure the security of cloud services and provide secure clouds. Huawei Cloud's security responsibilities include ensuring the security of our IaaS, PaaS, and SaaS services, as well as the physical environments of the Huawei Cloud data centers where our IaaS, PaaS, and SaaS services operate. Huawei Cloud is responsible for not only the security functions and performance of our infrastructure, cloud services, and technologies, but also for the overall cloud O&M security and, in the broader sense, the security and compliance of our infrastructure and services.
- **Tenant**: Use the cloud securely. Tenants of Huawei Cloud are responsible for the secure and effective management of the tenant-customized configurations of cloud services including IaaS, PaaS, and SaaS. This includes but is not limited to virtual networks, the OS of virtual machine hosts and guests, virtual firewalls, API Gateway, advanced security services, all types of cloud services, tenant data, identity accounts, and key management.

**Huawei Cloud Security White Paper** elaborates on the ideas and measures for building Huawei Cloud security, including cloud security strategies, the shared responsibility model, compliance and privacy, security organizations and personnel, infrastructure security, tenant service and security, engineering security, O&M security, and ecosystem security.

**Figure 8-1** Huawei Cloud shared security responsibility model




# 8.2 Identity Authentication and Access Control

## Identity Authentication

No matter whether you access DMS for Kafka through the console or APIs, you are required to provide the identity credential and verify the identity validity. In addition, login and login authentication policies are provided to harden identity authentication security.

DMS for Kafka uses IAM to provide three identity authentication modes: **passwords**, **access keys**, and **temporary access keys**. **Login protection** and **login authentication policies** are also provided.

## Access Control

You can assign different permissions for DMS for Kafka to employees in your organization for fine-grained permissions management. IAM provides identity authentication, permissions management, and access control, helping you secure access to your Huawei Cloud resources. For details about DMS for Kafka permissions, see **Permissions**.

# 8.3 Data Protection

DMS for Kafka takes different measures to keep data secure and reliable.

**Table 8-1** DMS for Kafka data protection methods and features

| Measure | Description | Reference |
| --- | --- | --- |
| DR and multi-active | To meet reliability requirements of your data and services, you can deploy a Kafka instance in a single AZ (single equipment room) or across AZs (intra-city DR), and use the data synchronization tool MirrorMaker to synchronize data across instances. | • **Buying an Instance Within an AZ or Across AZs**<br>• **Using MirrorMaker to Synchronize Data Across Instances** |
| Data replication | Data can be synchronized between replicas for data consistency. When a network exception or node fault occurs, a failover is automatically performed using the replicas. After the fault is rectified, data is synchronized from the leader replica to ensure data consistency. | **Creating a Topic with Multiple Replicas** |
| Data persistence | Exceptions may occur during daily running of the service system. Some service systems require high reliability, including high availability of instances, data security, and recoverability, so that backup data can be used to restore instances if an exception occurs, ensuring service running. | - |

# 8.4 Audit and Logs

Cloud Trace Service (CTS) records operations on the cloud resources in your account. You can use the logs generated by CTS to perform security analysis, trace resource changes, audit compliance, and locate faults.

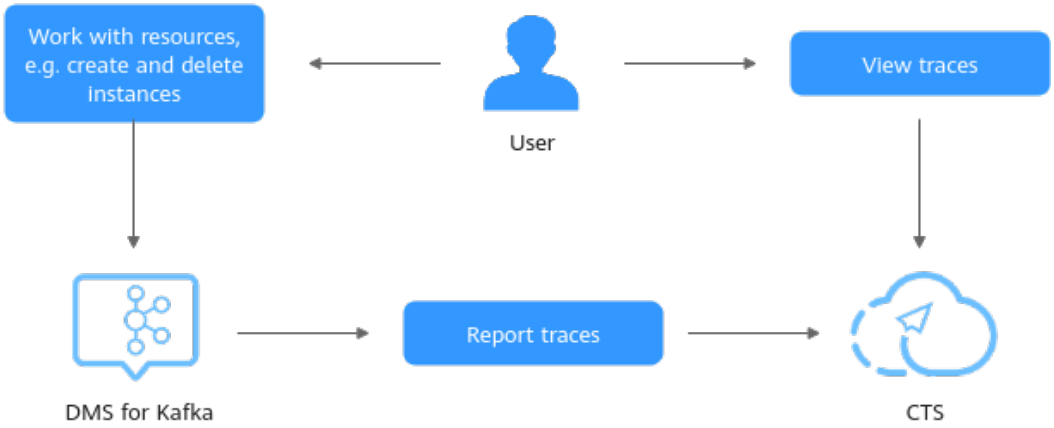
After you enable CTS and configure a tracker, CTS records management traces of DMS for Kafka for auditing.

For details about how to enable and configure CTS, see **Enabling CTS**.

For details about the DMS for Kafka management traces that can be recorded, see **Operations Logged by CTS**.

**Figure 8-2** CTS




# 8.5 Resilience

DMS for Kafka provides a three-level reliability architecture and uses cross-AZ DR, intra-AZ instance DR, and instance data replication to ensure service durability and reliability.

**Table 8-2** Reliability architecture of DMS for Kafka

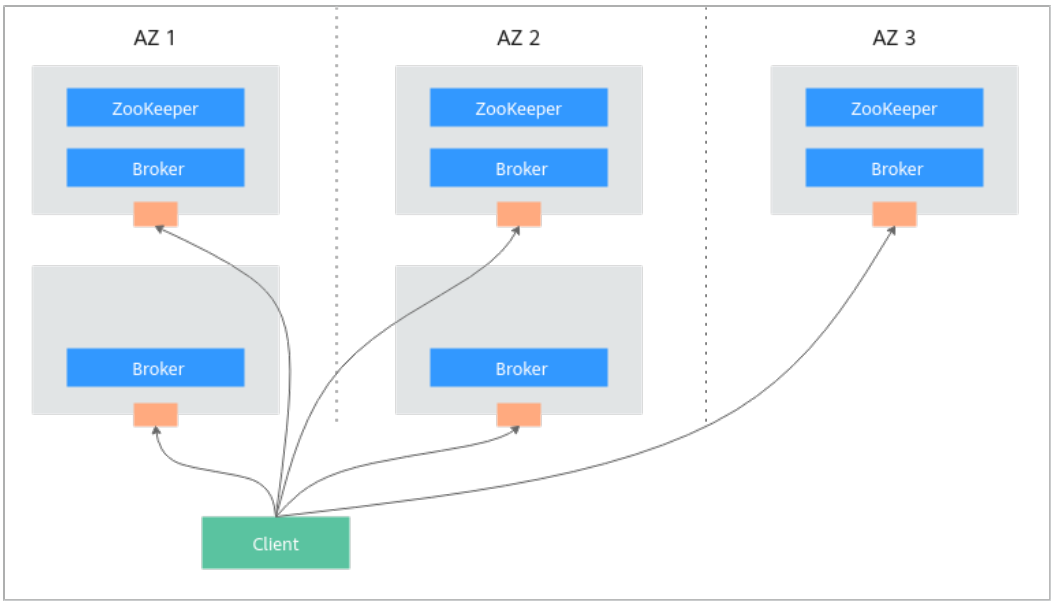
| Reliability Solution | Description |
| --- | --- |
| Cross-AZ DR | DMS for Kafka provides cross-AZ instances that support cross-AZ DR. When an AZ is abnormal, the instances can still provide services. |
| Intra-AZ instance DR | A Kafka instance has multiple replicas for instance-level DR within an AZ. If the leader replica is faulty, another leader is quickly elected to ensure uninterrupted Kafka services. |
| Data DR | Data DR is implemented through data replication. |

## Cross-AZ DR Deployment Architecture

DR can be achieved across AZs when DMS for Kafka is deployed in three or more AZs.

**Figure 8-3** Cross-AZ deployment architecture




# 8.6 Security Risk Monitoring

DMS for Kafka uses Cloud Eye to help you monitor your Kafka instances and receive alarms and notifications in real time. You can obtain key information about instances in real time, such as service requests, resource usage, traffic, number of connections, and number of accumulated messages.

For details about DMS for Kafka metrics and how to create alarm rules, see **Kafka Metrics**.

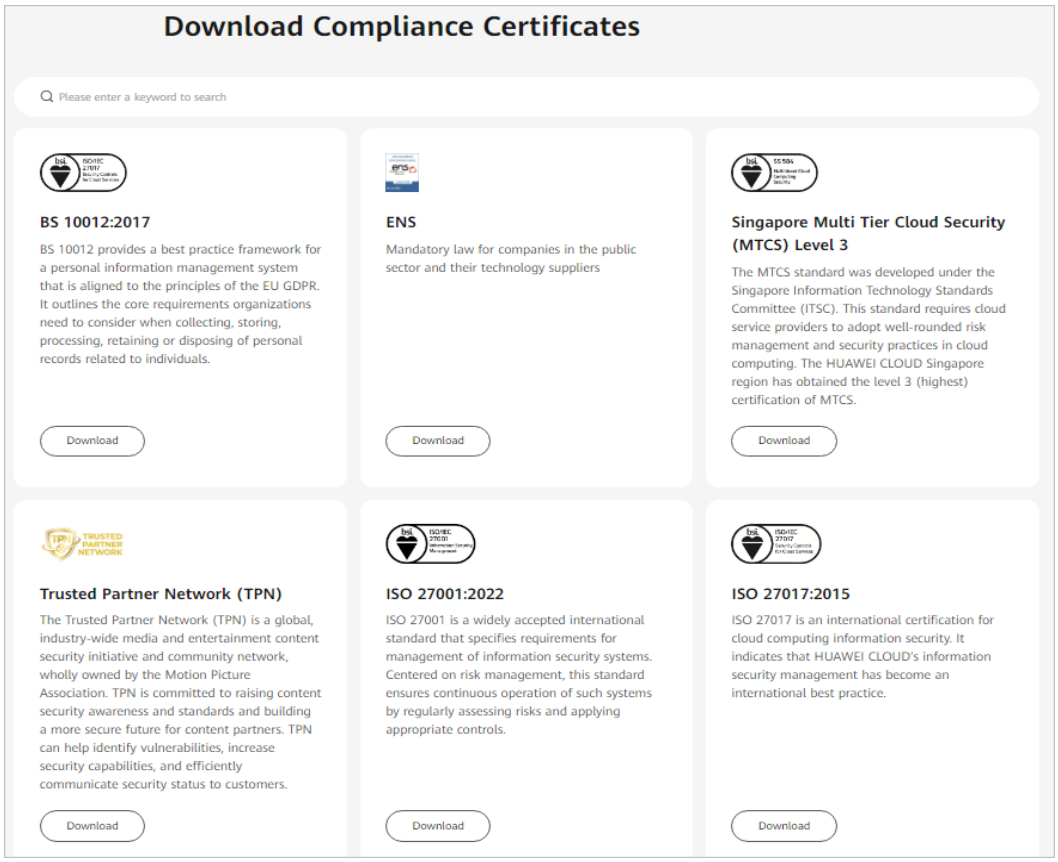
# 8.7 Certificates

## Compliance Certificates

Huawei Cloud services and platforms have obtained various security and compliance certifications from authoritative organizations, such as International Organization for Standardization (ISO). You can **download** them from the console.

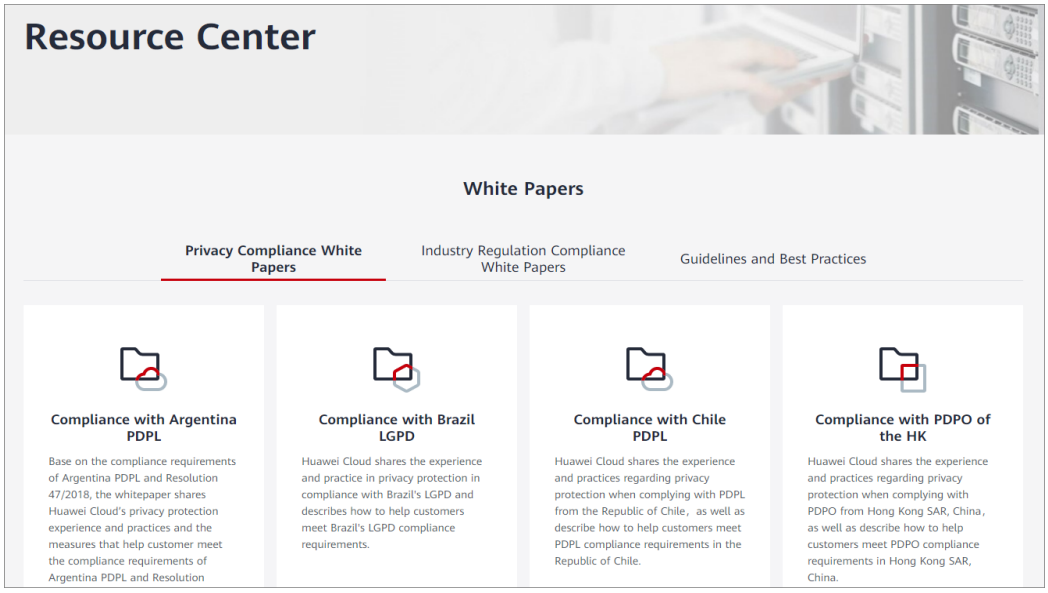
**Figure 8-4** Downloading compliance certificates




## Resource Center

Huawei Cloud also provides the following resources to help users meet compliance requirements. For details, see **Resource Center**.

**Figure 8-5** Resource center



# 9 Notes and Constraints

This section describes the notes and constraints on Distributed Message Service (DMS) for Kafka. Use your Kafka instances as prescribed to avoid program exceptions.

**NOTICE**

Any instability caused by ignorance of the notes and constraints is not covered by the SLA.

## Instance

**Table 9-1** Instance notes and constraints

| Item | Notes and Constraints |
| --- | --- |
| Kafka ZooKeeper | Kafka clusters are managed using ZooKeeper. Opening ZooKeeper may cause misoperations and service losses. Currently, ZooKeeper is used only within Kafka clusters and does not provide services externally. |
| Version | • The service version can be 1.1.0, 2.7, or 3.x. Kafka instances cannot be upgraded once they are created.<br>• Clients later than version 0.10 are supported. Use a version that is consistent with the service version. |
| Logging in to the VM where the Kafka brokers reside | Not supported |
| Storage | • The storage space of cluster instances can be expanded but cannot be reduced.<br>• You can expand the storage space up to 20 times.<br>• The storage space of single-node instances cannot be changed. |

| Item | Notes and Constraints |
| --- | --- |
| Bandwidth or broker quantity | • The bandwidth and broker quantity of cluster instances can be increased but cannot be decreased.<br>• The broker quantity cannot be changed for single-node instances. |
| Broker flavor | • The broker flavor of cluster instances can be increased or decreased.<br>• Single-replica topics do not support message creation and retrieval during this period. Services will be interrupted.<br>• If a topic has multiple replicas, scaling up or down the broker flavor does not interrupt services, but may cause disorder of partition messages. Evaluate this impact and avoid peak hours.<br>• Broker rolling restarts will cause partition leader changes, interrupting connections for less than a minute when the network is stable. For multi-replica topics, configure the retry mechanism on the producer client.<br>• If the total number of partitions created for a cluster instance is greater than the upper limit allowed by a new flavor, scale-down cannot be performed.<br>• The broker flavor of single-node instances cannot be changed. |
| VPC, subnet, and AZ | After an instance is created, its VPC, subnet, and AZ cannot be modified. |
| Kerberos authentication | Not supported |
| Client connections from each IP address | For instances purchased in July 2020 and later, each Kafka broker allows a maximum of 1000 connections from each IP address by default. For instances purchased before July 2020, each Kafka broker allows a maximum of 200 connections from each IP address by default. Excess connections will be rejected. |

## Topic

**Table 9-2** Topic notes and constraints

| Item | Notes and Constraints |
| --- | --- |
| Total number of topic partitions | The total number of topic partitions is related to the instance specifications. For details, see **Cluster Kafka Instances**.<br>Kafka manages messages by partition. If there are too many partitions, message creation, storage, and retrieval will be fragmented, affecting the performance and stability. If the total number of partitions of topics reaches the upper limit, you cannot create more topics. |
| Number of partitions in a topic | • Based on the open-source Kafka constraints, the number of partitions in a topic can be increased but cannot be decreased.<br>• To ensure performance, a partition number within 200 is recommended for each topic. |
| Topic quantity | The topic quantity is related to the total number of topic partitions and number of partitions in each topic. For details, see **Cluster Kafka Instances**. |

| Item | Notes and Constraints |
| --- | --- |
| Automatic topic creation | Supported. If this option is enabled, a topic will be automatically created when a message is produced in or consumed from a topic that does not exist. By default, the topic has the following parameters:<br>• Partitions: 1 for single-node instances and 3 for cluster instances<br>• Replicas: 1 for single-node instances and 3 for cluster instances<br>• **Aging Time**: 72<br>• **Synchronous Replication** and **Synchronous Flushing** disabled<br>• **Message Timestamp**: **CreateTime**<br>• **Max. Message Size (bytes)**: 10,485,760<br>For cluster instances, after you change the value of the **log.retention.hours** (retention period), **default.replication.factor** (replica quantity), or **num.partitions** (partition quantity) parameter, the value will be used in later topics that are automatically created. These parameters cannot be changed for single-node instances.<br>For example, assume that **num.partitions** is changed to 5, an automatically created topic has the following parameters:<br>• **Partitions**: 5<br>• **Replicas**: 3<br>• **Aging Time**: 72<br>• **Synchronous Replication** and **Synchronous Flushing** disabled<br>• **Message Timestamp**: **CreateTime**<br>• **Max.Message Size (bytes)**: 10,485,760 |
| Synchronous replication | If a topic has only one replica, synchronous replication cannot be enabled. |
| Replica quantity | Single-replica topics are not recommended for cluster instances. If an instance node is faulty, an internal service error may be reported when you query messages in a topic with only one replica. Therefore, you are not advised to use a topic with only one replica. |
| Aging time | The value of the **log.retention.hours** parameter takes effect only if the aging time has not been set for the topic.<br>For example, if the aging time of Topic01 is set to 60 hours and **log.retention.hours** is set to 72 hours, the actual aging time of Topic01 is 60 hours. |

| Item | Notes and Constraints |
| --- | --- |
| Batch importing and exporting topics | Batch export is supported, but batch import is not supported. |
| Topic name | If a topic name starts with a special character, for example, a number sign (#), monitoring data cannot be displayed. |
| Delay queues | Not supported |
| Broker faults | When some brokers of an instance are faulty, topics cannot be created, modified, or deleted, but can be queried. |

## Consumer Group

**Table 9-3** Consumer group notes and constraints

| Item | Notes and Constraints |
| --- | --- |
| Creating consumer groups, consumers, and producers | • When parameter **auto.create.groups.enable** is set to **true**, you do not need to create a consumer group, producer, or consumer because they are generated automatically when you use the instance.<br>• When parameter **auto.create.groups.enable** is set to **false**, you need to create a consumer group, but do not need to create a producer or consumer. |
| Resetting the consumer offset | Messages may be retrieved more than once after the offset is reset. |
| Consumer group name | If a consumer group name starts with a special character, for example, a number sign (#), monitoring data cannot be displayed. |
| Broker faults | When some instance brokers are faulty, consumer groups cannot be created, modified, or deleted, or consumption progress cannot be reset, but consumer groups can be queried. |

## Message

**Table 9-4** Message notes and constraints

| Item | Notes and Constraints |
| --- | --- |
| Message size | The maximum length of a message is 10 MB. If the length exceeds 10 MB, the production fails. |

## User

**Table 9-5** User notes and constraints

| Item | Notes and Constraints |
| --- | --- |
| Number of users | The maximum users that can be created for a Kafka instance is 20 or 500. Check the console for the actual limit. |
| Broker faults | When some instance brokers are faulty, users cannot be created, modified, or deleted, or password cannot be reset, but users can be queried. |

# 10 Related Services

- Cloud Trace Service (CTS)

  CTS generates traces to provide you with a history of operations performed on cloud service resources. The traces include operation requests sent using the management console or open APIs, as well as the operation results. You can view all generated traces to query, audit, and backtrack performed operations.

  For details about the operations recorded by CTS, see **Operations Logged by CTS**.

- Virtual Private Cloud (VPC)

  Kafka instances run in VPCs and use the IP addresses and bandwidth of VPC. Security groups of VPCs enhance the security of network access to the Kafka instances.

- Elastic Cloud Server (ECS)

  An ECS is a basic computing unit that consists of vCPUs, memory, OS, and EVS disks. Kafka instances run on ECSs. A broker corresponds to an ECS.

- Elastic Volume Service (EVS)

  EVS provides block storage services for ECSs. All Kafka data, such as messages, metadata, and logs, is stored in EVS disks.

- Identity and Access Management (IAM)

  IAM enables you to easily manage users and control their access to cloud services and resources. Grant different users different Kafka permissions required to perform a given task based on their job responsibilities.

- Cloud Eye (CES)

  Cloud Eye is an open platform that provides monitoring, alarm reporting, and alarm notification for your resources in real time.

  NOTE

  The values of all Kafka instance metrics are reported to Cloud Eye every minute.

- Elastic IP (EIP)

  The EIP service provides independent public IP addresses and bandwidth for Internet access. Kafka instances bound with EIPs can be accessed over public networks.

- Tag Management Service (TMS)

TMS is a visualized service for fast and unified cross-region tagging and categorization of cloud services.

Tags facilitate Kafka instance identification and management.

- VPC Endpoint

  A client can access a Kafka instance over a private network: When a client and a Kafka instance are deployed across VPCs in one region, connect the client and the Kafka instance across VPCs using a VPC endpoint.

- NAT Gateway

  A Kafka instance can communicate with a client over a public network: Configure port mapping from EIPs to specified instance ports using destination NAT (DNAT) of NAT Gateway.

# 11 Basic Concepts

DMS for Kafka of Huawei Cloud uses Kafka as the message engine. This chapter presents explanations of basic concepts of Kafka.

## Topic

A topic is a category for messages. Messages are created, retrieved, and managed in the form of topics.

Topics adopt the publish-subscribe pattern. Producers publish messages into topics. One or more consumers subscribe to the messages in the topics. The producers and consumers are not directly linked to each other.

## Producer

A producer publishes messages into topics. The messages are then delivered to other systems or modules for processing as agreed.

## Consumer

A consumer subscribes to messages in topics and processes the messages. For example, a monitoring and alarm platform (a consumer) subscribing to log messages in certain topics can identify alarm logs and then send SMS or email alarm notifications.

## Broker

A broker is a Kafka process in a Kafka cluster. Each process runs on a server, so a broker includes the storage, bandwidth, and other server resources.

## Partition

A topic is divided into partitions. Messages are distributed to multiple partitions to achieve scalability and fault tolerance.

## Replica

A replica is a redundant copy of a partition in a topic. Each partition can have one or more replicas, enabling message reliability.

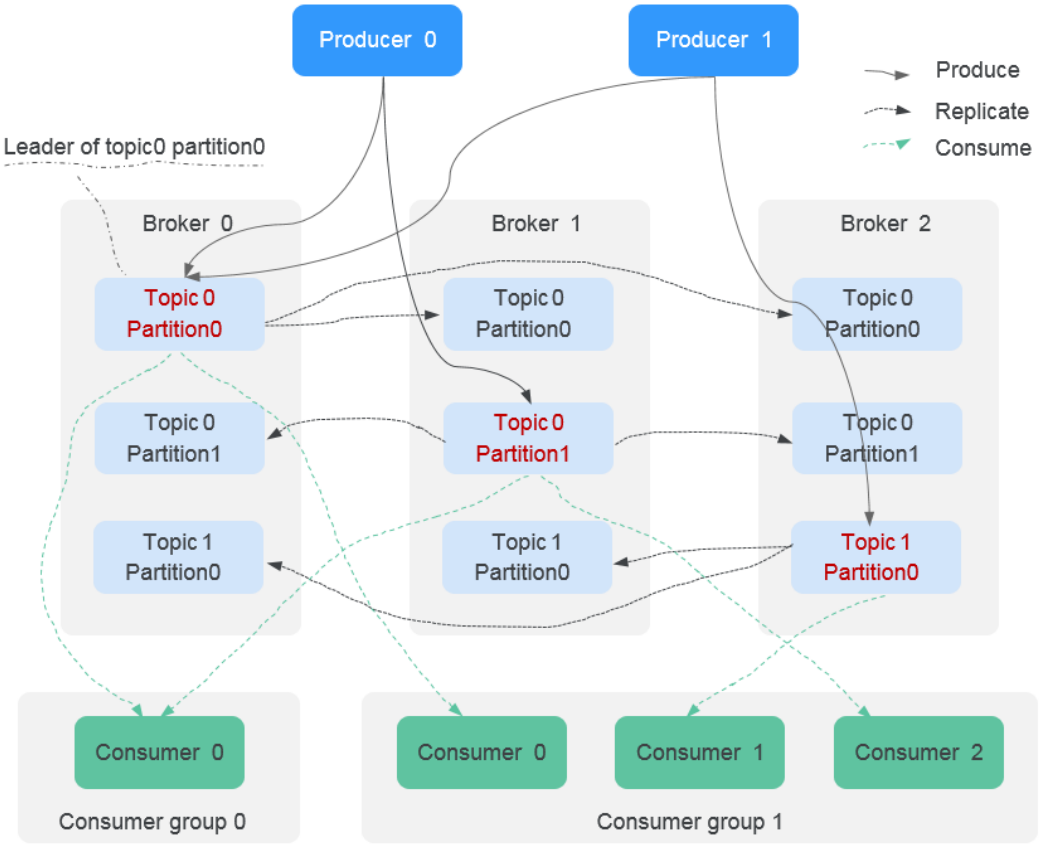
  

Messages in each partition are fully replicated and synchronized, preventing data loss if one replica fails.

Each partition has one replica as the leader which handles the creation and retrievals of all messages. The rest replicas are followers which replicate the leader.

Topics and partitions are logical concepts, while replicas and brokers are physical concepts. The following diagram shows the relationships between partitions, brokers, and topics in messages streaming.

**Figure 11-1** Kafka message streaming




## Aging Time

The period that messages are retained for. Consumers must retrieve messages before this period ends. Otherwise, the messages will be deleted and can no longer be retrieved.

# 12 Permissions

If you need to grant your enterprise personnel permission to access your DMS for Kafka resources, use Identity and Access Management (IAM). IAM provides identity authentication, fine-grained permissions management, and access control. IAM helps you secure access to your Huawei Cloud resources.

You can create IAM users for your employees, and assign permissions to these users on a principle of least privilege (PoLP) basis to control their access to specific resource types. For example, you can create IAM users for software developers and assign specific permissions to allow them to use Kafka instance resources but prevent them from being able to delete resources or perform any high-risk operations.

If your HUAWEI ID does not require individual IAM users for permissions management, skip this section.

IAM is a free service. You only pay for the resources in your account.

For more information, see **IAM Service Overview**.

NOTE

Permissions policies of DMS for Kafka are based on DMS. Therefore, when assigning permissions, select DMS permissions policies.

## DMS for Kafka Permissions

By default, new IAM users do not have any permissions assigned. To assign permissions to these new users, add them to one or more groups, and attach permissions policies or roles to these groups.

DMS for Kafka is a project-level service deployed and accessed in specific physical regions. When assigning DMS for Kafka permissions to a user group, specify region-specific projects where the permissions will take effect. If you select **All projects**, the permissions will be granted for all region-specific projects. When accessing DMS for Kafka, the users need to switch to a region where they have been authorized to use this service.

You can grant permissions by using roles and policies.

- Roles: A type of coarse-grained authorization mechanism that provides only a limited number of service-level roles. When using roles to grant permissions,

you also need to assign dependency roles. However, roles are not an ideal choice for fine-grained authorization and secure access control.

- Policies: A fine-grained authorization strategy that defines permissions required to perform operations on specific cloud resources under certain conditions. This mechanism allows for more flexible policy-based authorization for more secure access control. For example, you can grant DMS for Kafka users only the permissions for managing instances. Most policies define permissions based on APIs. For the API actions supported by DMS for Kafka, see **Permissions Policies and Supported Actions**.

**Table 12-1** lists all the system-defined roles and policies supported by DMS for Kafka.

**Table 12-1** System-defined roles and policies supported by DMS for Kafka

| Role/Policy Name | Description | Type | Dependency |
| --- | --- | --- | --- |
| DMS FullAccess | Administrator permissions for DMS. Users granted these permissions can perform all operations on DMS. | System-defined policy | None |
| DMS UserAccess | Common user permissions for DMS, excluding permissions for creating, modifying, deleting, dumping, and scaling up instances. | System-defined policy | None |
| DMS ReadOnlyAccess | Read-only permissions for DMS. Users granted these permissions can only view DMS data. | System-defined policy | None |
| DMS VPCAccess | VPC operation permissions to assign to DMS agencies. | System-defined policy | None |
| DMS KMSAccess | KMS operation permissions to assign to DMS agencies. | System-defined policy | None |
| DMS ELBAccess | ELB operation permissions to assign to DMS agencies. | System-defined policy | None |
| DMS VPCEndpointAccess | VPC endpoint operation permissions to assign to DMS agencies. | System-defined policy | None |
| DMSAgencyCheckAccessPolicy | IAM operation permissions to assign to DMS agencies. | System-defined policy | None |

| Role/Policy Name | Description | Type | Dependency |
| --- | --- | --- | --- |
| DMS Administrator | Administrator permissions for DMS. | System-defined role | This role depends on the **Tenant Guest** and **VPC Administrator** roles. |

NOTE

System-defined policies contain OBS actions. Due to data caching, the policies take effect five minutes after they are attached to a user, user group, or enterprise project.

**Table 12-2** lists the common operations supported by each DMS for Kafka system policy. Select the policies as required.

**Table 12-2** Common operations supported by system-defined policies

| Operation | DMS FullAccess | DMS UserAccess | DMS ReadOnlyAccess | DMS VPCAccess | DMS KMSAccess | DMS ELBAccess | DMS VPCEndpoint Access | DMSAgency CheckAccessPolicy |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Creating an instance | √ | × | × | × | × | × | × | × |
| Modifying instances | √ | × | × | × | × | × | × | × |
| Deleting instances | √ | × | × | × | × | × | × | × |
| Modifying instance specifications | √ | × | × | × | × | × | × | × |

| Operation | DMS FullAccess | DMS UserAccess | DMS ReadOnlyAccess | DMS VPCAccess | DMS KMSAccess | DMS ELBAccess | DMS VPCEndpoint Access | DMSAgency Check Access Policy |
|---|---|---|---|---|---|---|---|---|
| Enabling Smart Connect | √ | × | × | × | × | × | × | × |
| Creating a Smart Connect task | √ | √ | × | × | × | × | × | × |
| Restarting instances | √ | √ | × | × | × | × | × | × |
| Querying instance information | √ | √ | √ | × | × | × | × | × |

## Fine-grained Authorization

To use a custom fine-grained policy, log in to the IAM console as an administrator and select the desired fine-grained permissions for DMS. **Table 12-3** describes fine-grained permission dependencies of DMS for Kafka.

**Table 12-3** Fine-grained permission dependencies of DMS for Kafka

| Permission | Description | Dependency |
|---|---|---|
| dms:instance:list | Viewing the instance list | None |
| dms:instance:get | Viewing instance details | None |

| Permission | Description | Dependency |
| --- | --- | --- |
| dms:instance:create | Creating an instance | • vpc:vpcs:get<br>• vpc:ports:create<br>• vpc:securityGroups:get<br>• vpc:ports:get<br>• vpc:subnets:get<br>• vpc:vpcs:list<br>• vpc:publicIps:get<br>• vpc:publicIps:list<br>• vpc:ports:update<br>• vpc:publicIps:update<br>• vpc:ports:delete |
| dms:instance:getBackgroundTask | Viewing background task details | None |
| dms:instance:deleteBackgroundTask | Deleting a background task | None |
| dms:instance:modifyStatus | Restarting an instance | None |
| dms:instance:resetAuthInfo | Resetting an instance password | None |
| dms:instance:modifyAuthInfo | Changing an instance password | None |
| dms:instance:modify | Modifying an instance | • vpc:vpcs:get<br>• vpc:ports:create<br>• vpc:securityGroups:get<br>• vpc:ports:get<br>• vpc:subnets:get<br>• vpc:vpcs:list<br>• vpc:publicIps:get<br>• vpc:publicIps:list<br>• vpc:ports:update<br>• vpc:publicIps:update<br>• vpc:ports:delete |

| Permission | Description | Dependency |
| --- | --- | --- |
| dms:instance:scale | Scaling up an instance | • vpc:vpcs:get<br>• vpc:ports:create<br>• vpc:securityGroups:get<br>• vpc:ports:get<br>• vpc:subnets:get<br>• vpc:vpcs:list<br>• vpc:publicIps:get<br>• vpc:publicIps:list<br>• vpc:ports:update<br>• vpc:publicIps:update |
| dms:instance:delete | Deleting an instance | None |
| dms:instance:connector | Enabling dumping | • vpc:vpcs:get<br>• vpc:ports:create<br>• vpc:securityGroups:get<br>• vpc:ports:get<br>• vpc:subnets:get<br>• vpc:vpcs:list<br>• vpc:publicIps:get<br>• vpc:publicIps:list<br>• vpc:ports:update<br>• vpc:publicIps:update |
| dms:instance:createConnectorSinkTask | Creating a dumping task | None |
| dms:instance:getConnectorSinkTask | Viewing dumping task details | None |
| dms:instance:listConnectorSinkTask | Viewing the dumping task list | None |
| dms:instance:deleteConnectorSinkTask | Deleting a dumping task | None |

## Helpful Links

- **What Is IAM?**
- **Creating User Groups, Users, and Granting DMS for Kafka permissions**
- **Permissions Policies and Supported Actions**