

Kafka

[零 参考资料 2](#_Toc67151525)

[一 概述 2](#_Toc67151526)

[1 体系结构 2](#_Toc67151527)

[1.1 概念 2](#_Toc67151528)

[2 时间轮 5](#_Toc67151529)

[3 ISR & HW & LEO 5](#_Toc67151530)

[二 Producer 7](#_Toc67151531)

[1 Synopsis 7](#_Toc67151532)

[2 配置 8](#_Toc67151533)

[3 API 18](#_Toc67151534)

[三 Consumer 18](#_Toc67151535)

[1 概述 18](#_Toc67151536)

[2 配置 18](#_Toc67151537)

[3 API 24](#_Toc67151538)

[四 Topic&Partition 24](#_Toc67151539)

[1 概述 24](#_Toc67151540)

[2 Topic配置 24](#_Toc67151541)

[五 Log Storage 29](#_Toc67151542)

[1 概述 29](#_Toc67151543)

[2 日志V0 29](#_Toc67151544)

[3 日志V1 29](#_Toc67151545)

[4 日志V3 29](#_Toc67151546)

[六 Server 29](#_Toc67151547)

[1 概述 29](#_Toc67151548)

[2 Broker配置 29](#_Toc67151549)

[七 监控 56](#_Toc67151550)

[八 其他配置 56](#_Toc67151551)

[1 Connect 56](#_Toc67151552)

[1.1 概述 56](#_Toc67151553)

[1.2 配置 56](#_Toc67151554)

[2 Streams 58](#_Toc67151555)

[2.1 概述 58](#_Toc67151556)

[2.2 配置 58](#_Toc67151557)

[3 AdminClient 58](#_Toc67151558)

[3.1 概述 58](#_Toc67151559)

[3.2 配置 58](#_Toc67151560)

[九 应用 58](#_Toc67151561)

# 零 参考资料

https://kafka.apache.org/documentation/

https://kafka.apache.org/27/javadoc/index.html

《深入理解Kafka 核心设计与实践原理》朱忠华

PS：1 文档是2.7版本，书是2.0版本，参数配置以文档为准，具体差异不在本文体现

2 红色配置为关键配置

# Synopsis

## 体系结构

### Motivation

* + - 1. We designed Kafka to be able to act as a unified platform for handling all the real-time data feeds a large company might have.
      2. To do this we had to think through a fairly broad set of use cases.
      3. It would have to have high-throughput to support high volume event streams such as real-time log aggregation.
      4. It would need to deal gracefully with large data backlogs to be able to support periodic data loads from offline systems.
      5. It also meant the system would have to handle low-latency delivery to handle more traditional messaging use-cases.
      6. We wanted to support partitioned, distributed, real-time processing of these feeds to create new, derived feeds. This motivated our partitioning and consumer model.
      7. Finally in cases where the stream is fed into other data systems for serving, we knew the system would have to be able to guarantee fault-tolerance in the presence of machine failures.

### Persistence

* + - 1. FileSystem
      2. Constant Time Suffice

### Efficiency

### 概念

* + - 1. Producer：生产者，即消息发送方，负责产生消息，投递到Kafka中
      2. Consumer：消费者，即消息接收方，连接到kafka上并接收消息，然后根据消息处理业务逻辑
      3. Broker：服务代理节点
         1. 可以简单的理解为一个单独的kafka服务节点或Kafka服务实例，
         2. 但如果一台服务器上只部署了一个kafka实例，才可以将broker看作一台Kafka服务器。
         3. 一个或多个Broker组成了kafka集群
      4. Topic：主题，逻辑上的概念，kafka消息按topic进行分类
         1. 生产者负责将消息发送到特定Topic
         2. 消费者通过订阅topic消费消息
         3. 一个Topic可以横跨多个Broker
      5. Partition: Topic的分区
         1. 一个Partition只属于单个Topic，因此也被称为Topic-Partition
         2. 同个Topic下，不同Partition包含的消息不同
         3. 在存储层面，Partition可以看作是一个可追加的Log文件
      6. Offset：偏移量，从0开始
         1. 消息被追加到Partition的log文件时，会分配到一个特定的Offset
         2. offset是消息在存储其的Partition中唯一标识。
         3. offset不跨分区，因此kafka通过offset保证消息在当前分区有序，但不保证消息在Topic中有序
      7. Replica：副本
         1. kafka基于Partition实现多Replica机制，提高容灾能力
         2. 同一个Partition的不同Replica中保存的消息最终相同，即同一时刻，同一个Partition的多Replica之间并非一定相同
         3. Replica之间是一主多从关系：

Leader 负责处理消息读写请求，follower只负责与Leader消息同步

当Leader出现故障时，从Follower中重新选举新的Leader对外服务

大部分时候follower的消息相对Leader有一定延后

* + - 1. Assigned Replica：Partition所有replica的统称
         1. ISR(In-Sync Replica)：由所有与Leader保持一定程度同步的replica组成，是AR的子集
         2. OSR(Out-of-Sync Replica)：由与Leader同步滞后过多的Replica组成，是AR的子集
         3. AR = ISR+OSR,正常情况下，AR=ISR，OSR为空
         4. Leader负责维护&跟踪所有follower的同步情况

当ISR中的follower滞后太多或失效时，leader会把其移出ISR到OSR中

当OSR中的follower同步消息追上leader时，将追上的follower从OSR移到ISR

* + - 1. HW(High Watermark)：高水位，标识了一个特定的消息offset，consumer只能获取到这个offset之前的消息
      2. LEO(Log End Offset)：标识当前log下一条待写入消息offset，相当于当前log最后条消息offset+1







## Time Sheet



## ISR & HW & LEO

* + - 1. 假设某个分区的ISR集合中有3个副本，即1个leader和两个follow
         1. 此时LEO和HW都为3
         2. 生产者发出消息3和消息4，会被先存入leader



* + - 1. 消息3&4写入Leader后，Follower会请求拉取消息3&4以进行同步



* + - 1. 同步过程中，不同Follower同步效率不同，下图情况：
         1. Leader的LEO=5，Follower1 LEO=5，Follower2 LEO=4， HW=4
         2. Consumer可以获取0-3消息



* + - 1. 当所有Replica都同步到消息4，则所有HW和LEO都为5，Consumer可以获得到消息4了



# Producer

## Synopsis

* + - 1. The producer sends data directly to the broker that is the leader for the partition without any intervening routing tier.
      2. To help the producer do this all Kafka nodes can answer a request for metadata about which servers are alive and where the leaders for the partitions of a topic are at any given time to allow the producer to appropriately direct its requests.
      3. The client controls which partition it publishes messages to.
         1. This can be done at random, implementing a kind of random load balancing, or it can be done by some semantic partitioning function.
         2. Kafka expose the interface for semantic partitioning by allowing the user to specify a key to partition by and using this to hash to a partition (there is also an option to override the partition function if need be).
         3. For example if the key chosen was a user id then all data for a given user would be sent to the same partition.
         4. This in turn will allow consumers to make locality assumptions about their consumption.
         5. This style of partitioning is explicitly designed to allow locality-sensitive processing in consumers.
      4. Batching is one of the big drivers of efficiency, and to enable batching the Kafka producer will attempt to accumulate data in memory and to send out larger batches in a single request.
         1. The batching can be configured to accumulate no more than a fixed number of messages and to wait no longer than some fixed latency bound (say 64k or 10 ms).
         2. This allows the accumulation of more bytes to send, and few larger I/O operations on the servers.
         3. This buffering is configurable and gives a mechanism to trade off a small amount of additional latency for better throughput.



## Configuration

|  |  |
| --- | --- |
| 配置 | 说明 |
| key.serializer | Serializer class for key that implements the **org.apache.kafka.common.serialization.Serializer interface.** |
| value.serializer | Serializer class for value that implements the **org.apache.kafka.common.serialization.Serializer interface.** |
| acks | 1 The number of acknowledgments the producer requires the leader to have received 2 2 before considering a request complete.  3 This controls the durability of records that are sent.  4 The following settings are allowed:  4.1 acks=0  If set to zero then the producer will not wait for any acknowledgment from the server at all.  The record will be immediately added to the socket buffer and considered sent. No guarantee can be made that the server has received the record in this case, and the retries configuration will not take effect (as the client won't generally know of any failures).  The offset given back for each record will always be set to -1.  4.2 acks=1  This will mean the leader will write the record to its local log but will respond without awaiting full acknowledgement from all followers.  In this case should the leader fail immediately after acknowledging the record but before the followers have replicated it then the record will be lost.  4.3 acks=all  This means the leader will wait for the full set of in-sync replicas to acknowledge the record.  This guarantees that the record will not be lost as long as at least one in-sync replica remains alive. This is the strongest available guarantee.  This is equivalent to the acks=-1 setting.  Default: 1  Valid Values: [all, -1, 0, 1] |
| bootstrap.servers | 1 A list of host/port pairs to use for establishing the initial connection to the Kafka cluster.  2 The client will make use of all servers irrespective of which servers are specified here for bootstrapping—this list only impacts the initial hosts used to discover the full set of servers.  3 This list should be in the form host1:port1,host2:port2,.... Since these servers are just used for the initial connection to discover the full cluster membership (which may change dynamically), this list need not contain the full set of servers (you may want more than one, though, in case a server is down).  Default: ""  Valid Values: non-null string |
| buffer.memory | The total bytes of memory the producer can use to buffer records waiting to be sent to the server. If records are sent faster than they can be delivered to the server the producer will block for max.block.ms after which it will throw an exception.  This setting should correspond roughly to the total memory the producer will use, but is not a hard bound since not all memory the producer uses is used for buffering. Some additional memory will be used for compression (if compression is enabled) as well as for maintaining in-flight requests.  Default: 33554432  Valid Values: [0,...] |
| compression.type | The compression type for all data generated by the producer. The default is none (i.e. no compression). Valid values are none, gzip, snappy, lz4, or zstd. Compression is of full batches of data, so the efficacy of batching will also impact the compression ratio (more batching means better compression).  Default: none |
| retries | Setting a value greater than zero will cause the client to resend any record whose send fails with a potentially transient error. Note that this retry is no different than if the client resent the record upon receiving the error. Allowing retries without setting max.in.flight.requests.per.connection to 1 will potentially change the ordering of records because if two batches are sent to a single partition, and the first fails and is retried but the second succeeds, then the records in the second batch may appear first. Note additionally that produce requests will be failed before the number of retries has been exhausted if the timeout configured by delivery.timeout.ms expires first before successful acknowledgement. Users should generally prefer to leave this config unset and instead use delivery.timeout.ms to control retry behavior.  Default: 2147483647  Valid Values: [0,...,2147483647] |
| batch.size | The producer will attempt to batch records together into fewer requests whenever multiple records are being sent to the same partition. This helps performance on both the client and the server. This configuration controls the default batch size in bytes.  No attempt will be made to batch records larger than this size.  Requests sent to brokers will contain multiple batches, one for each partition with data available to be sent.  A small batch size will make batching less common and may reduce throughput (a batch size of zero will disable batching entirely). A very large batch size may use memory a bit more wastefully as we will always allocate a buffer of the specified batch size in anticipation of additional records.  Default: 16384  Valid Values: [0,...] |
| client.dns.lookup | Controls how the client uses DNS lookups. If set to use\_all\_dns\_ips, connect to each returned IP address in sequence until a successful connection is established. After a disconnection, the next IP is used. Once all IPs have been used once, the client resolves the IP(s) from the hostname again (both the JVM and the OS cache DNS name lookups, however). If set to resolve\_canonical\_bootstrap\_servers\_only, resolve each bootstrap address into a list of canonical names. After the bootstrap phase, this behaves the same as use\_all\_dns\_ips. If set to default (deprecated), attempt to connect to the first IP address returned by the lookup, even if the lookup returns multiple IP addresses.  Default: use\_all\_dns\_ips  Valid Values: [default, use\_all\_dns\_ips, resolve\_canonical\_bootstrap\_servers\_only] |
| client.id | An id string to pass to the server when making requests. The purpose of this is to be able to track the source of requests beyond just ip/port by allowing a logical application name to be included in server-side request logging.  Default: "" |
| connections.max.idle.ms | Close idle connections after the number of milliseconds specified by this config.  Default: 540000 (9 minutes) |
| delivery.timeout.ms | An upper bound on the time to report success or failure after a call to send() returns. This limits the total time that a record will be delayed prior to sending, the time to await acknowledgement from the broker (if expected), and the time allowed for retriable send failures. The producer may report failure to send a record earlier than this config if either an unrecoverable error is encountered, the retries have been exhausted, or the record is added to a batch which reached an earlier delivery expiration deadline. The value of this config should be greater than or equal to the sum of request.timeout.ms and linger.ms.  Default: 120000 (2 minutes)  Valid Values: [0,...] |
| linger.ms | The producer groups together any records that arrive in between request transmissions into a single batched request. Normally this occurs only under load when records arrive faster than they can be sent out. However in some circumstances the client may want to reduce the number of requests even under moderate load. This setting accomplishes this by adding a small amount of artificial delay—that is, rather than immediately sending out a record the producer will wait for up to the given delay to allow other records to be sent so that the sends can be batched together. This can be thought of as analogous to Nagle's algorithm in TCP. This setting gives the upper bound on the delay for batching: once we get batch.size worth of records for a partition it will be sent immediately regardless of this setting, however if we have fewer than this many bytes accumulated for this partition we will 'linger' for the specified time waiting for more records to show up. This setting defaults to 0 (i.e. no delay). Setting linger.ms=5, for example, would have the effect of reducing the number of requests sent but would add up to 5ms of latency to records sent in the absence of load.  Default: 0  Valid Values: [0,...] |
| max.block.ms | The configuration controls how long the KafkaProducer's send(), partitionsFor(), initTransactions(), sendOffsetsToTransaction(), commitTransaction() and abortTransaction() methods will block. For send() this timeout bounds the total time waiting for both metadata fetch and buffer allocation (blocking in the user-supplied serializers or partitioner is not counted against this timeout). For partitionsFor() this timeout bounds the time spent waiting for metadata if it is unavailable. The transaction-related methods always block, but may timeout if the transaction coordinator could not be discovered or did not respond within the timeout.  Default: 60000 (1 minute)  Valid Values: [0,...] |
| max.request.size | The maximum size of a request in bytes. This setting will limit the number of record batches the producer will send in a single request to avoid sending huge requests. This is also effectively a cap on the maximum uncompressed record batch size. Note that the server has its own cap on the record batch size (after compression if compression is enabled) which may be different from this.  Default: 1048576  Valid Values: [0,...] |
| partitioner.class | Partitioner class that implements the org.apache.kafka.clients.producer.Partitioner interface.  Default: org.apache.kafka.clients.producer.internals.DefaultPartitioner |
| receive.buffer.bytes | The size of the TCP receive buffer (SO\_RCVBUF) to use when reading data. If the value is -1, the OS default will be used.  Default: 32768 (32 kibibytes)  Valid Values: [-1,...] |
| request.timeout.ms | The configuration controls the maximum amount of time the client will wait for the response of a request. If the response is not received before the timeout elapses the client will resend the request if necessary or fail the request if retries are exhausted. This should be larger than replica.lag.time.max.ms (a broker configuration) to reduce the possibility of message duplication due to unnecessary producer retries.  Default: 30000 (30 seconds)  Valid Values: [0,...] |
| security.protocol | Protocol used to communicate with brokers.  Default: PLAINTEXT  Valid Values: PLAINTEXT, SSL, SASL\_PLAINTEXT, SASL\_SSL. |
| send.buffer.bytes | The size of the TCP send buffer (SO\_SNDBUF) to use when sending data. If the value is -1, the OS default will be used.  Default: 131072 (128 kibibytes)  Valid Values: [-1,...] |
| socket.connection.setup.timeout.max.ms | The maximum amount of time the client will wait for the socket connection to be established. The connection setup timeout will increase exponentially for each consecutive connection failure up to this maximum. To avoid connection storms, a randomization factor of 0.2 will be applied to the timeout resulting in a random range between 20% below and 20% above the computed value.  Default: 127000 (127 seconds) |
| socket.connection.setup.timeout.ms | The amount of time the client will wait for the socket connection to be established. If the connection is not built before the timeout elapses, clients will close the socket channel.  Default: 10000 (10 seconds) |
| enable.idempotence | When set to 'true', the producer will ensure that exactly one copy of each message is written in the stream. If 'false', producer retries due to broker failures, etc., may write duplicates of the retried message in the stream. Note that enabling idempotence requires max.in.flight.requests.per.connection to be less than or equal to 5, retries to be greater than 0 and acks must be 'all'. If these values are not explicitly set by the user, suitable values will be chosen. If incompatible values are set, a ConfigException will be thrown.  Default: false |
| interceptor.classes | A list of classes to use as interceptors. Implementing the org.apache.kafka.clients.producer.ProducerInterceptor interface allows you to intercept (and possibly mutate) the records received by the producer before they are published to the Kafka cluster. By default, there are no interceptors.  Default: ""  Valid Values: non-null string |
| max.in.flight.requests.per.connection | The maximum number of unacknowledged requests the client will send on a single connection before blocking. Note that if this setting is set to be greater than 1 and there are failed sends, there is a risk of message re-ordering due to retries (i.e., if retries are enabled).  Default: 5  Valid Values: [1,...] |
| metadata.max.age.ms | The period of time in milliseconds after which we force a refresh of metadata even if we haven't seen any partition leadership changes to proactively discover any new brokers or partitions.  Default: 300000 (5 minutes)  Valid Values: [0,...] |
| metadata.max.idle.ms | Controls how long the producer will cache metadata for a topic that's idle. If the elapsed time since a topic was last produced to exceeds the metadata idle duration, then the topic's metadata is forgotten and the next access to it will force a metadata fetch request.  Default: 300000 (5 minutes)  Valid Values: [5000,...] |
| metric.reporters | A list of classes to use as metrics reporters. Implementing the org.apache.kafka.common.metrics.MetricsReporter interface allows plugging in classes that will be notified of new metric creation. The JmxReporter is always included to register JMX statistics.  Default: ""  Valid Values: non-null string |
| metrics.num.samples | The number of samples maintained to compute metrics.  Default: 2  Valid Values: [1,...] |
| metrics.recording.level | The highest recording level for metrics.  Default: INFO  Valid Values: [INFO, DEBUG, TRACE] |
| metrics.sample.window.ms | The window of time a metrics sample is computed over.  Default: 30000 (30 seconds)  Valid Values: [0,...] |
| reconnect.backoff.max.ms | The maximum amount of time in milliseconds to wait when reconnecting to a broker that has repeatedly failed to connect. If provided, the backoff per host will increase exponentially for each consecutive connection failure, up to this maximum. After calculating the backoff increase, 20% random jitter is added to avoid connection storms.  Default: 1000 (1 second)  Valid Values: [0,...] |
| reconnect.backoff.ms | The base amount of time to wait before attempting to reconnect to a given host. This avoids repeatedly connecting to a host in a tight loop. This backoff applies to all connection attempts by the client to a broker.  Default: 50  Valid Values: [0,...] |
| retry.backoff.ms | The amount of time to wait before attempting to retry a failed request to a given topic partition. This avoids repeatedly sending requests in a tight loop under some failure scenarios.  Default: 100  Valid Values: [0,...] |
| security.providers | A list of configurable creator classes each returning a provider implementing security algorithms. These classes should implement the org.apache.kafka.common.security.auth.SecurityProviderCreator interface.  Default: null |
| transaction.timeout.ms | The maximum amount of time in ms that the transaction coordinator will wait for a transaction status update from the producer before proactively aborting the ongoing transaction.If this value is larger than the transaction.max.timeout.ms setting in the broker, the request will fail with a InvalidTxnTimeoutException error.  Default: 60000 (1 minute) |
| transactional.id | The TransactionalId to use for transactional delivery. This enables reliability semantics which span multiple producer sessions since it allows the client to guarantee that transactions using the same TransactionalId have been completed prior to starting any new transactions. If no TransactionalId is provided, then the producer is limited to idempotent delivery. If a TransactionalId is configured, enable.idempotence is implied. By default the TransactionId is not configured, which means transactions cannot be used. Note that, by default, transactions require a cluster of at least three brokers which is the recommended setting for production; for development you can change this, by adjusting broker setting transaction.state.log.replication.factor.  Default: null  Valid Values: non-empty string |
| ssl.key.password | The password of the private key in the key store file orthe PEM key specified in `ssl.keystore.key'. This is required for clients only if two-way authentication is configured.  Default: null |
| ssl.keystore.certificate.chain | Certificate chain in the format specified by 'ssl.keystore.type'. Default SSL engine factory supports only PEM format with a list of X.509 certificates  Default: null |
| ssl.keystore.key | Private key in the format specified by 'ssl.keystore.type'. Default SSL engine factory supports only PEM format with PKCS#8 keys. If the key is encrypted, key password must be specified using 'ssl.key.password'  Default: null |
| ssl.keystore.location | The location of the key store file. This is optional for client and can be used for two-way authentication for client.  Default: null |
| ssl.keystore.password | The store password for the key store file. This is optional for client and only needed if 'ssl.keystore.location' is configured. Key store password is not supported for PEM format.  Default: null |
| ssl.truststore.certificates | Trusted certificates in the format specified by 'ssl.truststore.type'. Default SSL engine factory supports only PEM format with X.509 certificates.  Default: null |
| ssl.truststore.location | The location of the trust store file.  Default: null |
| ssl.truststore.password | The password for the trust store file. If a password is not set, trust store file configured will still be used, but integrity checking is disabled. Trust store password is not supported for PEM format.  Default: null |
| ssl.enabled.protocols | The list of protocols enabled for SSL connections. The default is 'TLSv1.2,TLSv1.3' when running with Java 11 or newer, 'TLSv1.2' otherwise. With the default value for Java 11, clients and servers will prefer TLSv1.3 if both support it and fallback to TLSv1.2 otherwise (assuming both support at least TLSv1.2). This default should be fine for most cases. Also see the config documentation for `ssl.protocol`.  Default: TLSv1.2 |
| ssl.keystore.type | The file format of the key store file. This is optional for client.  Default: JKS |
| ssl.protocol | The SSL protocol used to generate the SSLContext. The default is 'TLSv1.3' when running with Java 11 or newer, 'TLSv1.2' otherwise. This value should be fine for most use cases. Allowed values in recent JVMs are 'TLSv1.2' and 'TLSv1.3'. 'TLS', 'TLSv1.1', 'SSL', 'SSLv2' and 'SSLv3' may be supported in older JVMs, but their usage is discouraged due to known security vulnerabilities. With the default value for this config and 'ssl.enabled.protocols', clients will downgrade to 'TLSv1.2' if the server does not support 'TLSv1.3'. If this config is set to 'TLSv1.2', clients will not use 'TLSv1.3' even if it is one of the values in ssl.enabled.protocols and the server only supports 'TLSv1.3'.  Default: TLSv1.2 |
| ssl.provider | The name of the security provider used for SSL connections. Default value is the default security provider of the JVM.  Default: null |
| ssl.truststore.type | The file format of the trust store file.  Default: JKS |
| ssl.cipher.suites | A list of cipher suites. This is a named combination of authentication, encryption, MAC and key exchange algorithm used to negotiate the security settings for a network connection using TLS or SSL network protocol. By default all the available cipher suites are supported.  Default: null |
| ssl.endpoint.identification.algorithm | The endpoint identification algorithm to validate server hostname using server certificate.  Default: https |
| ssl.engine.factory.class | The class of type org.apache.kafka.common.security.auth.SslEngineFactory to provide SSLEngine objects. Default value is org.apache.kafka.common.security.ssl.DefaultSslEngineFactory  Default: null |
| ssl.keymanager.algorithm | The algorithm used by key manager factory for SSL connections. Default value is the key manager factory algorithm configured for the Java Virtual Machine.  Default: SunX509 |
| ssl.secure.random.implementation | The SecureRandom PRNG implementation to use for SSL cryptography operations.  Default: null |
| ssl.trustmanager.algorithm | The algorithm used by trust manager factory for SSL connections. Default value is the trust manager factory algorithm configured for the Java Virtual Machine.  Default: PKIX |
| sasl.kerberos.kinit.cmd | Kerberos kinit command path.  Default: /usr/bin/kinit |
| sasl.kerberos.min.time.before.relogin | Login thread sleep time between refresh attempts.  Default: 60000 |
| sasl.kerberos.ticket.renew.jitter | Percentage of random jitter added to the renewal time.  Default: 0.05 |
| sasl.kerberos.ticket.renew.window.factor | Login thread will sleep until the specified window factor of time from last refresh to ticket's expiry has been reached, at which time it will try to renew the ticket.  Default: 0.8 |
| sasl.login.refresh.buffer.seconds | The amount of buffer time before credential expiration to maintain when refreshing a credential, in seconds. If a refresh would otherwise occur closer to expiration than the number of buffer seconds then the refresh will be moved up to maintain as much of the buffer time as possible. Legal values are between 0 and 3600 (1 hour); a default value of 300 (5 minutes) is used if no value is specified. This value and sasl.login.refresh.min.period.seconds are both ignored if their sum exceeds the remaining lifetime of a credential. Currently applies only to OAUTHBEARER.  Default: 300  Valid Values: [0,...,3600] |
| sasl.login.refresh.min.period.seconds | The desired minimum time for the login refresh thread to wait before refreshing a credential, in seconds. Legal values are between 0 and 900 (15 minutes); a default value of 60 (1 minute) is used if no value is specified. This value and sasl.login.refresh.buffer.seconds are both ignored if their sum exceeds the remaining lifetime of a credential. Currently applies only to OAUTHBEARER.  Default: 60  Valid Values: [0,...,900] |
| sasl.login.refresh.window.factor | Login refresh thread will sleep until the specified window factor relative to the credential's lifetime has been reached, at which time it will try to refresh the credential. Legal values are between 0.5 (50%) and 1.0 (100%) inclusive; a default value of 0.8 (80%) is used if no value is specified. Currently applies only to OAUTHBEARER.  Default: 0.8  Valid Values: [0.5,...,1.0] |
| sasl.login.refresh.window.jitter | The maximum amount of random jitter relative to the credential's lifetime that is added to the login refresh thread's sleep time. Legal values are between 0 and 0.25 (25%) inclusive; a default value of 0.05 (5%) is used if no value is specified. Currently applies only to OAUTHBEARER.  Default: 0.05  Valid Values: [0.0,...,0.25] |
| sasl.client.callback.handler.class | The fully qualified name of a SASL client callback handler class that implements the AuthenticateCallbackHandler interface.  Default: null |
| sasl.jaas.config | JAAS login context parameters for SASL connections in the format used by JAAS configuration files. JAAS configuration file format is described here. The format for the value is: 'loginModuleClass controlFlag (optionName=optionValue)\*;'. For brokers, the config must be prefixed with listener prefix and SASL mechanism name in lower-case. For example, listener.name.sasl\_ssl.scram-sha-256.sasl.jaas.config=com.example.ScramLoginModule required;  Default: null |
| sasl.kerberos.service.name | The Kerberos principal name that Kafka runs as. This can be defined either in Kafka's JAAS config or in Kafka's config.  Default: null |
| sasl.login.callback.handler.class | The fully qualified name of a SASL login callback handler class that implements the AuthenticateCallbackHandler interface. For brokers, login callback handler config must be prefixed with listener prefix and SASL mechanism name in lower-case. For example, listener.name.sasl\_ssl.scram-sha-256.sasl.login.callback.handler.class=com.example.CustomScramLoginCallbackHandler  Default: null |
| sasl.login.class | The fully qualified name of a class that implements the Login interface. For brokers, login config must be prefixed with listener prefix and SASL mechanism name in lower-case. For example, listener.name.sasl\_ssl.scram-sha-256.sasl.login.class=com.example.CustomScramLogin  Default: null |
| sasl.mechanism | SASL mechanism used for client connections. This may be any mechanism for which a security provider is available. GSSAPI is the default mechanism.  Default: GSSAPI |

## API

# Consumer

## Synopsis

## Configuration

|  |  |
| --- | --- |
| 配置项 | 说明 |
| key.deserializer | 指定反序列化key值的反序列化工具  继承org.apache.kafka.common.serialization.Deserializer |
| value.deserializer | 指定反序列化消息value值的反序列化工具  继承org.apache.kafka.common.serialization.Deserializer |
| bootstrap.servers | 指定broker服务地址，用于消费者获取集群信息  可以只指定集群一个broker，或全部broker  多个地址用逗号隔开host1:port1,host2:port2,... |
| group.id | 设置消费者所属组  如果消费者采用subscribe(topic) 或 the Kafka-based offset管理策略作为组管理方法，则必须配置，且必须唯一 |
| fetch.max.bytes | 每次批量拉取消息最大数据大小  为保障消费者正常处理，如果第一次拉取消息数据大小大于此值，仍继续正常拉取  值受message.max.bytes (broker config) or max.message.bytes (topic config) |
| fetch.min.bytes | 指定消费者抓取数据最小大小，如果没有足够数据，服务器将等待积累数据，直到数据足够时才返回数据给消费者  默认1byte，即服务器及时响应请求  适当设置高点，可以提高服务器吞吐量 |
| fetch.max.wait.ms | 拉取数据没达到fetch.min.bytes指定数据量时，最大等待时间 |
| heartbeat.interval.ms | 心跳检查时间间隔  值必须小于session.timeout.ms配置，推荐不超过1/3 session.timeout.ms  默认3000ms |
| max.partition.fetch.bytes | 每个分区批量获取消息最大数据量  为保障消费者正常执行，第一次批量拉取消息时超过了限制，则也会拉取消息  此配置受message.max.bytes (broker config) 或max.message.bytes(topic config)影响  默认1048576(1mb) |
| session.timeout.ms | 心跳检测超时时间  如果超时，broker会移除当前消费者，并重新负载  默认10000  值必须在group.min.session.timeout.ms配置和group.max.session.timeout.ms配置之间 |
| allow.auto.create.topics | 是否自动创建订阅的topic  仅当broker配置auto.create.topics.enable为tru时有效  链接0.11.0之前版本的broker，必须为false  默认为true |
| auto.offset.reset | 当消费者没有指定初始offset或指定的offset不存在时，指定定位offset策略  取值[latest, earliest, none]  earliest：定位到当前分区第一条记录  latest：定位到当前分区最后一条记录  none：如果没有指定初始offset或指定的offset不存在时，抛异常  其他值都抛异常 |
| client.dns.lookup | 设置DNS查找域名对应的ip列表缓存规则  默认use\_all\_dns\_ips  取值[default, use\_all\_dns\_ips, resolve\_canonical\_bootstrap\_servers\_only]  use\_all\_dns\_ips：依次访问DNS返回的IP 列表，直到ip可访问；如果断开链接，则尝试下个ip；如果所有ip都无法连接，则重新请求DNS获取ip列表  resolve\_canonical\_bootstrap\_servers\_only：解析bootstrap address为规范名称列表，然后按照use\_all\_dns\_ips策略访问  default：仅尝试请求第一个ip地址，即使配置了多个broker地址 |
| connections.max.idle.ms | 链接空闲最大时间  默认540000s (9m) |
| default.api.timeout.ms | api请求超时时间，每个超时相关配置没配置时，会取此配置值  默认60000ms(1m) |
| enable.auto.commit | 是否自动提交消费事务  默认true |
| auto.commit.interval.ms | 定时提交offset，当enable.auto.commit为true时  默认5000 ms |
| exclude.internal.topics | 当使用正则表达式订阅topic时，是否排除内部topic  默认为true |
| group.instance.id | 消费者在群组中唯一标识  默认为不设置  如果不设置，则作为一个动态成员加入组  如果设置，则作为一个静态成员加入组，可以优化长时间session |
| isolation.level | 配置读取写事务中消息的策略  默认值read\_uncommitted  取值[read\_committed, read\_uncommitted]  read\_committed：只读取事务提交后的消息，  read\_uncommitted：读取所有消息 |
| max.poll.interval.ms | customer每次poll等待最大时间  当customer没有在此时间内拉取数据，broker会认为此customer失效，会将此customer负责的分区重新分配给组内其他成员  如果此customer指定了group.instance.id，则不会立即重新分配分区，直到customer的session断开时间超过session.timeout.ms配置，才会在组内重新分配分区  默认300000ms |
| max.poll.records | 每次poll最大数据量  默认500 |
| partition.assignment.strategy | 分区获取策略，默认是依次轮询  默认org.apache.kafka.clients.consumer.RoundRobinAssignor  可以通过实现org.apache.kafka.clients.consumer.ConsumerPartitionAssignor接口自定义策略 |
| send.buffer.bytes | 配置发送套接字缓存  如果设置为-1，则以OS配置为准  默认131072 B(128KB) |
| receive.buffer.bytes | 套接字(SO\_RCVBUF)缓冲大小  如果设置为-1，则以OS配置为准  默认65536 B(64KB) |
| request.timeout.ms | customer等待broker响应时间  默认30000ms |
| security.protocol | 配置安全协议  默认PLAINTEXT  取值PLAINTEXT, SSL, SASL\_PLAINTEXT, SASL\_SSL. |
| security.providers | A list of configurable creator classes each returning a provider implementing security algorithms. These classes should implement the org.apache.kafka.common.security.auth.SecurityProviderCreator interface.  Default: null |
| socket.connection.setup.timeout.max.ms | 配置套接字链接超时最大时间  每次套接字链接超时，都会以0.2为因子，随机比例增加或减少超时时间,但最大超时时间不会超过此值 |
| socket.connection.setup.timeout.ms | 配置套接字链接超时时间 |
| check.crcs | 是否检查消息的CRC32值，保证消息完整  默认为true  会有一定的性能消耗 |
| client.id | 配置客户端ID，用于broker日志中追踪请求源  默认为空字符串 |
| client.rack | 主机所在位置标识，用于标识customer物理位置  默认为空字符串 |
| interceptor.classes | 配置拦截器  通过org.apache.kafka.clients.consumer.ConsumerInterceptor接口实现自定义拦截器，  默认空 |
| metadata.max.age.ms | 定时强制更新元数据信息，保证发现broker或分区变动  默认300000s(5min) |
| metric.reporters | A list of classes to use as metrics reporters. Implementing the org.apache.kafka.common.metrics.MetricsReporter interface allows plugging in classes that will be notified of new metric creation. The JmxReporter is always included to register JMX statistics.  Default: ""  Valid Values: non-null string |
| metrics.num.samples | The number of samples maintained to compute metrics.  Default: 2  Valid Values: [1,...] |
| metrics.recording.level | The highest recording level for metrics.  Default: INFO  Valid Values: [INFO, DEBUG, TRACE] |
| metrics.sample.window.ms | The window of time a metrics sample is computed over.  Default: 30000 (30 seconds)  Valid Values: [0,...] |
| reconnect.backoff.max.ms | 重新链接前最大等待时间  默认1000ms |
| reconnect.backoff.ms | 重新链接前等待时间  默认50ms |
| retry.backoff.ms | 重试间隔等时间  默认100ms |
| ssl.key.password | The password of the private key in the key store file orthe PEM key specified in `ssl.keystore.key'. This is required for clients only if two-way authentication is configured.  Default: null |
| ssl.keystore.certificate.chain | Certificate chain in the format specified by 'ssl.keystore.type'. Default SSL engine factory supports only PEM format with a list of X.509 certificates  Default: null |
| ssl.keystore.location | The location of the key store file. This is optional for client and can be used for two-way authentication for client.  Default: null |
| ssl.keystore.key | Private key in the format specified by 'ssl.keystore.type'. Default SSL engine factory supports only PEM format with PKCS#8 keys. If the key is encrypted, key password must be specified using 'ssl.key.password'  Default: null |
| ssl.keystore.password | The store password for the key store file. This is optional for client and only needed if 'ssl.keystore.location' is configured. Key store password is not supported for PEM format.  Default: null |
| ssl.truststore.location | Trusted certificates in the format specified by 'ssl.truststore.type'. Default SSL engine factory supports only PEM format with X.509 certificates.  Default: null |
| ssl.truststore.password | The password for the trust store file. If a password is not set, trust store file configured will still be used, but integrity checking is disabled. Trust store password is not supported for PEM format.  Default: null |
| ssl.truststore.certificates | Trusted certificates in the format specified by 'ssl.truststore.type'. Default SSL engine factory supports only PEM format with X.509 certificates.  Default: null |
| ssl.enabled.protocols | The endpoint identification algorithm to validate server hostname using server certificate.  Default: https |
| ssl.keystore.type | The file format of the key store file. This is optional for client.  Default: JKS |
| ssl.protocol | The SSL protocol used to generate the SSLContext. The default is 'TLSv1.3' when running with Java 11 or newer, 'TLSv1.2' otherwise. This value should be fine for most use cases. Allowed values in recent JVMs are 'TLSv1.2' and 'TLSv1.3'. 'TLS', 'TLSv1.1', 'SSL', 'SSLv2' and 'SSLv3' may be supported in older JVMs, but their usage is discouraged due to known security vulnerabilities. With the default value for this config and 'ssl.enabled.protocols', clients will downgrade to 'TLSv1.2' if the server does not support 'TLSv1.3'. If this config is set to 'TLSv1.2', clients will not use 'TLSv1.3' even if it is one of the values in ssl.enabled.protocols and the server only supports 'TLSv1.3'.  Default: TLSv1.2 |
| ssl.provider | The name of the security provider used for SSL connections. Default value is the default security provider of the JVM.  Default: null |
| ssl.truststore.type | The file format of the trust store file.  Default: JKS |
| ssl.cipher.suites | A list of cipher suites. This is a named combination of authentication, encryption, MAC and key exchange algorithm used to negotiate the security settings for a network connection using TLS or SSL network protocol. By default all the available cipher suites are supported.  Default: null |
| ssl.endpoint.identification.algorithm | The endpoint identification algorithm to validate server hostname using server certificate.  Default: https |
| ssl.engine.factory.class | The class of type org.apache.kafka.common.security.auth.SslEngineFactory to provide SSLEngine objects. Default value is org.apache.kafka.common.security.ssl.DefaultSslEngineFactory  Default: null |
| ssl.keymanager.algorithm | The algorithm used by key manager factory for SSL connections. Default value is the key manager factory algorithm configured for the Java Virtual Machine.  Default: SunX509 |
| ssl.secure.random.implementation | The SecureRandom PRNG implementation to use for SSL cryptography operations.  Default: null |
| ssl.trustmanager.algorithm | The algorithm used by trust manager factory for SSL connections. Default value is the trust manager factory algorithm configured for the Java Virtual Machine.  Default: PKIX |
| sasl.client.callback.handler.class | The fully qualified name of a SASL client callback handler class that implements the AuthenticateCallbackHandler interface.  Default: null |
| sasl.jaas.config | JAAS login context parameters for SASL connections in the format used by JAAS configuration files. JAAS configuration file format is described here. The format for the value is: 'loginModuleClass controlFlag (optionName=optionValue)\*;'. For brokers, the config must be prefixed with listener prefix and SASL mechanism name in lower-case. For example, listener.name.sasl\_ssl.scram-sha-256.sasl.jaas.config=com.example.ScramLoginModule required;  Default: null |
| sasl.kerberos.service.name | The Kerberos principal name that Kafka runs as. This can be defined either in Kafka's JAAS config or in Kafka's config.  Default: null |
| sasl.login.callback.handler.class | The fully qualified name of a SASL login callback handler class that implements the AuthenticateCallbackHandler interface. For brokers, login callback handler config must be prefixed with listener prefix and SASL mechanism name in lower-case. For example, listener.name.sasl\_ssl.scram-sha-256.sasl.login.callback.handler.class=com.example.CustomScramLoginCallbackHandler  Default: null |
| sasl.login.class | The fully qualified name of a class that implements the Login interface. For brokers, login config must be prefixed with listener prefix and SASL mechanism name in lower-case. For example, listener.name.sasl\_ssl.scram-sha-256.sasl.login.class=com.example.CustomScramLogin  Default: null |
| sasl.mechanism | SASL mechanism used for client connections. This may be any mechanism for which a security provider is available. GSSAPI is the default mechanism.  Default: GSSAPI |
| sasl.kerberos.kinit.cmd | Kerberos kinit command path.  Default: /usr/bin/kinit |
| sasl.kerberos.min.time.before.relogin | Login thread sleep time between refresh attempts.  Default: 60000 |
| sasl.kerberos.ticket.renew.jitter | Percentage of random jitter added to the renewal time.  Default: 0.05 |
| sasl.kerberos.ticket.renew.window.factor | Login thread will sleep until the specified window factor of time from last refresh to ticket's expiry has been reached, at which time it will try to renew the ticket.  Default: 0.8 |
| sasl.login.refresh.buffer.seconds | The amount of buffer time before credential expiration to maintain when refreshing a credential, in seconds. If a refresh would otherwise occur closer to expiration than the number of buffer seconds then the refresh will be moved up to maintain as much of the buffer time as possible. Legal values are between 0 and 3600 (1 hour); a default value of 300 (5 minutes) is used if no value is specified. This value and sasl.login.refresh.min.period.seconds are both ignored if their sum exceeds the remaining lifetime of a credential. Currently applies only to OAUTHBEARER.  Default: 300  Valid Values: [0,...,3600] |
| sasl.login.refresh.min.period.seconds | The desired minimum time for the login refresh thread to wait before refreshing a credential, in seconds. Legal values are between 0 and 900 (15 minutes); a default value of 60 (1 minute) is used if no value is specified. This value and sasl.login.refresh.buffer.seconds are both ignored if their sum exceeds the remaining lifetime of a credential. Currently applies only to OAUTHBEARER.  Default: 60  Valid Values: [0,...,900] |
| sasl.login.refresh.window.factor | Login refresh thread will sleep until the specified window factor relative to the credential's lifetime has been reached, at which time it will try to refresh the credential. Legal values are between 0.5 (50%) and 1.0 (100%) inclusive; a default value of 0.8 (80%) is used if no value is specified. Currently applies only to OAUTHBEARER.  Default: 0.8  Valid Values: [0.5,...,1.0] |
| sasl.login.refresh.window.jitter | The maximum amount of random jitter relative to the credential's lifetime that is added to the login refresh thread's sleep time. Legal values are between 0 and 0.25 (25%) inclusive; a default value of 0.05 (5%) is used if no value is specified. Currently applies only to OAUTHBEARER.  Default: 0.05  Valid Values: [0.0,...,0.25] |

## API

# Topic&Partition

## Synopsis

## Topic Configuration

|  |  |
| --- | --- |
| 配置项 | 说明 |
| cleanup.policy | A string that is either "delete" or "compact" or both. This string designates the retention policy to use on old log segments. The default policy ("delete") will discard old segments when their retention time or size limit has been reached. The "compact" setting will enable log compaction on the topic.  Default: delete  [compact, delete]  Server Default Property: log.cleanup.policy |
| compression.type | Specify the final compression type for a given topic. This configuration accepts the standard compression codecs ('gzip', 'snappy', 'lz4', 'zstd'). It additionally accepts 'uncompressed' which is equivalent to no compression; and 'producer' which means retain the original compression codec set by the producer.  Default: producer  Valid Values: [uncompressed, zstd, lz4, snappy, gzip, producer]  Server Default Property: compression.type |
| delete.retention.ms | The amount of time to retain delete tombstone markers for log compacted topics. This setting also gives a bound on the time in which a consumer must complete a read if they begin from offset 0 to ensure that they get a valid snapshot of the final stage (otherwise delete tombstones may be collected before they complete their scan).  Default: 86400000 (1 day)  Valid Values: [0,...]  Server Default Property: log.cleaner.delete.retention.ms |
| file.delete.delay.ms | The time to wait before deleting a file from the filesystem  Default: 60000 (1 minute)  Valid Values: [0,...]  Server Default Property: log.segment.delete.delay.ms |
| flush.messages | This setting allows specifying an interval at which we will force an fsync of data written to the log. For example if this was set to 1 we would fsync after every message; if it were 5 we would fsync after every five messages. In general we recommend you not set this and use replication for durability and allow the operating system's background flush capabilities as it is more efficient. This setting can be overridden on a per-topic basis (see the per-topic configuration section).  Default: 9223372036854775807  Valid Values: [0,...]  Server Default Property: log.flush.interval.messages |
| flush.ms | This setting allows specifying a time interval at which we will force an fsync of data written to the log. For example if this was set to 1000 we would fsync after 1000 ms had passed. In general we recommend you not set this and use replication for durability and allow the operating system's background flush capabilities as it is more efficient.  Default: 9223372036854775807  Valid Values: [0,...]  Server Default Property: log.flush.interval.ms |
| follower.replication.throttled.replicas | A list of replicas for which log replication should be throttled on the follower side. The list should describe a set of replicas in the form [PartitionId]:[BrokerId],[PartitionId]:[BrokerId]:... or alternatively the wildcard '\*' can be used to throttle all replicas for this topic.  Default: ""  Valid Values: [partitionId]:[brokerId],[partitionId]:[brokerId],...  Server Default Property: follower.replication.throttled.replicas |
| index.interval.bytes | This setting controls how frequently Kafka adds an index entry to its offset index. The default setting ensures that we index a message roughly every 4096 bytes. More indexing allows reads to jump closer to the exact position in the log but makes the index larger. You probably don't need to change this.  Default: 4096 (4 kibibytes)  Valid Values: [0,...]  Server Default Property: log.index.interval.bytes |
| leader.replication.throttled.replicas | A list of replicas for which log replication should be throttled on the leader side. The list should describe a set of replicas in the form [PartitionId]:[BrokerId],[PartitionId]:[BrokerId]:... or alternatively the wildcard '\*' can be used to throttle all replicas for this topic.  Default: ""  Valid Values: [partitionId]:[brokerId],[partitionId]:[brokerId],...  Server Default Property: leader.replication.throttled.replicas |
| max.compaction.lag.ms | The maximum time a message will remain ineligible for compaction in the log. Only applicable for logs that are being compacted.  Default: 9223372036854775807  Valid Values: [1,...]  Server Default Property: log.cleaner.max.compaction.lag.ms |
| max.message.bytes | The largest record batch size allowed by Kafka (after compression if compression is enabled). If this is increased and there are consumers older than 0.10.2, the consumers' fetch size must also be increased so that they can fetch record batches this large. In the latest message format version, records are always grouped into batches for efficiency. In previous message format versions, uncompressed records are not grouped into batches and this limit only applies to a single record in that case.  Default: 1048588  Valid Values: [0,...]  Server Default Property: message.max.bytes |
| message.format.version | Specify the message format version the broker will use to append messages to the logs. The value should be a valid ApiVersion. Some examples are: 0.8.2, 0.9.0.0, 0.10.0, check ApiVersion for more details. By setting a particular message format version, the user is certifying that all the existing messages on disk are smaller or equal than the specified version. Setting this value incorrectly will cause consumers with older versions to break as they will receive messages with a format that they don't understand.  Default: 2.7-IV2  Valid Values: [0.8.0, 0.8.1, 0.8.2, 0.9.0, 0.10.0-IV0, 0.10.0-IV1, 0.10.1-IV0, 0.10.1-IV1, 0.10.1-IV2, 0.10.2-IV0, 0.11.0-IV0, 0.11.0-IV1, 0.11.0-IV2, 1.0-IV0, 1.1-IV0, 2.0-IV0, 2.0-IV1, 2.1-IV0, 2.1-IV1, 2.1-IV2, 2.2-IV0, 2.2-IV1, 2.3-IV0, 2.3-IV1, 2.4-IV0, 2.4-IV1, 2.5-IV0, 2.6-IV0, 2.7-IV0, 2.7-IV1, 2.7-IV2]  Server Default Property: log.message.format.version |
| message.timestamp.difference.max.ms | The maximum difference allowed between the timestamp when a broker receives a message and the timestamp specified in the message. If message.timestamp.type=CreateTime, a message will be rejected if the difference in timestamp exceeds this threshold. This configuration is ignored if message.timestamp.type=LogAppendTime.  Default: 9223372036854775807  Valid Values: [0,...]  Server Default Property: log.message.timestamp.difference.max.ms |
| message.timestamp.type | Define whether the timestamp in the message is message create time or log append time. The value should be either `CreateTime` or `LogAppendTime`  Default: CreateTime  Valid Values: [CreateTime, LogAppendTime]  Server Default Property: log.message.timestamp.type |
| min.cleanable.dirty.ratio | This configuration controls how frequently the log compactor will attempt to clean the log (assuming log compaction is enabled). By default we will avoid cleaning a log where more than 50% of the log has been compacted. This ratio bounds the maximum space wasted in the log by duplicates (at 50% at most 50% of the log could be duplicates). A higher ratio will mean fewer, more efficient cleanings but will mean more wasted space in the log. If the max.compaction.lag.ms or the min.compaction.lag.ms configurations are also specified, then the log compactor considers the log to be eligible for compaction as soon as either: (i) the dirty ratio threshold has been met and the log has had dirty (uncompacted) records for at least the min.compaction.lag.ms duration, or (ii) if the log has had dirty (uncompacted) records for at most the max.compaction.lag.ms period.  Default: 0.5  Valid Values: [0,...,1]  Server Default Property: log.cleaner.min.cleanable.ratio |
| min.compaction.lag.ms | The minimum time a message will remain uncompacted in the log. Only applicable for logs that are being compacted.  Default: 0  Valid Values: [0,...]  Server Default Property: log.cleaner.min.compaction.lag.ms |
| min.insync.replicas | When a producer sets acks to "all" (or "-1"), this configuration specifies the minimum number of replicas that must acknowledge a write for the write to be considered successful. If this minimum cannot be met, then the producer will raise an exception (either NotEnoughReplicas or NotEnoughReplicasAfterAppend).  When used together, min.insync.replicas and acks allow you to enforce greater durability guarantees. A typical scenario would be to create a topic with a replication factor of 3, set min.insync.replicas to 2, and produce with acks of "all". This will ensure that the producer raises an exception if a majority of replicas do not receive a write.  Default: 1  Valid Values: [1,...]  Server Default Property: min.insync.replicas |
| preallocate | True if we should preallocate the file on disk when creating a new log segment.  Default: false  Server Default Property: log.preallocate |
| retention.bytes | This configuration controls the maximum size a partition (which consists of log segments) can grow to before we will discard old log segments to free up space if we are using the "delete" retention policy. By default there is no size limit only a time limit. Since this limit is enforced at the partition level, multiply it by the number of partitions to compute the topic retention in bytes.  Default: -1  Server Default Property: log.retention.bytes |
| retention.ms | This configuration controls the maximum time we will retain a log before we will discard old log segments to free up space if we are using the "delete" retention policy. This represents an SLA on how soon consumers must read their data. If set to -1, no time limit is applied.  Default: 604800000 (7 days)  Valid Values: [-1,...]  Server Default Property: log.retention.ms |
| segment.bytes | This configuration controls the segment file size for the log. Retention and cleaning is always done a file at a time so a larger segment size means fewer files but less granular control over retention.  Default: 1073741824 (1 gibibyte)  Valid Values: [14,...]  Server Default Property: log.segment.bytes |
| segment.index.bytes | This configuration controls the size of the index that maps offsets to file positions. We preallocate this index file and shrink it only after log rolls. You generally should not need to change this setting.  Default: 10485760 (10 mebibytes)  Valid Values: [0,...]  Server Default Property: log.index.size.max.bytes |
| segment.jitter.ms | The maximum random jitter subtracted from the scheduled segment roll time to avoid thundering herds of segment rolling  Default: 0  Valid Values: [0,...]  Server Default Property: log.roll.jitter.ms |
| segment.ms | This configuration controls the period of time after which Kafka will force the log to roll even if the segment file isn't full to ensure that retention can delete or compact old data.  Default: 604800000 (7 days)  Valid Values: [1,...]  Server Default Property: log.roll.ms |
| unclean.leader.election.enable | Indicates whether to enable replicas not in the ISR set to be elected as leader as a last resort, even though doing so may result in data loss.  Default: false  Server Default Property: unclean.leader.election.enable |
| message.downconversion.enable | This configuration controls whether down-conversion of message formats is enabled to satisfy consume requests. When set to false, broker will not perform down-conversion for consumers expecting an older message format. The broker responds with UNSUPPORTED\_VERSION error for consume requests from such older clients. This configurationdoes not apply to any message format conversion that might be required for replication to followers.  Default: true  Server Default Property: log.message.downconversion.enable |

# Log Storage

## 概述

## 日志V0

## 日志V1

## 日志V3

# Server

## 概述

## Broker配置

|  |  |
| --- | --- |
| 配置项 | 说明 |
| zookeeper.connect | 指定zk地址hostname:port  多个用英文逗号隔开  也可以指定zk路径  hostname1:port1,hostname2:port2,hostname3:port3/chroot/path |
| advertised.host.name | 不推荐使用  仅当advertised.listeners或listeners配置为空时有效  建议用advertised.listeners替代  将指定外网地址推送到zk  如果没有配置则取host.name值  如果也没有配置host.name则取java.net.InetAddress.getCanonicalHostName()值  在IaaS等外网地址需要特指的环境中使用 |
| advertised.listeners | 将指定外网地址推送到zk  如果没有配置，则推送listeners配置  不能配置为0.0.0.0 |
| advertised.port | 不推荐使用  仅当advertised.listeners或listeners配置为空时有效  建议用advertised.listeners替代  将指定端口推送到zk  如果没有配置，则推送broker绑定端口 |
| auto.create.topics.enable | 是否允许自动创建topic  默认true  如果为true，则producer发送消息到一个未存在的topic，broker会自动创建一个topic |
| auto.leader.rebalance.enable | 是否允许leader重新负载  默认为true 当为true时，后台线程会根据leader.imbalance.check.interval.seconds定期检查当前leader，如果当前leader负载超过了leader.imbalance.per.broker.percentage百分比，则自动重新负载分配leader |
| background.threads | 配置后台处理辅助任务的线程数  默认10，取值为大于0的正整数 |
| broker.id | 设置broker唯一标识  默认值-1， 取值int型  如果不设置，则自动生成一个  为保证同个zk下broker.id唯一，自动生成的id从reserved.broker.max.id+1开始 |
| compression.type | 指定topic消息默认压缩方式  默认值 producer  取值gzip, snappy, lz4, zstd，producer，uncompressed  如果设置为uncompressed，则为不压缩  如果设置为producer，则根据producer配置决定压缩方式 |
| control.plane.listener.name | Name of listener used for communication between controller and brokers. Broker will use the control.plane.listener.name to locate the endpoint in listeners list, to listen for connections from the controller. For example, if a broker's config is :  listeners = INTERNAL://192.1.1.8:9092, EXTERNAL://10.1.1.5:9093, CONTROLLER://192.1.1.8:9094  listener.security.protocol.map = INTERNAL:PLAINTEXT, EXTERNAL:SSL, CONTROLLER:SSL  control.plane.listener.name = CONTROLLER  On startup, the broker will start listening on "192.1.1.8:9094" with security protocol "SSL".  On controller side, when it discovers a broker's published endpoints through zookeeper, it will use the control.plane.listener.name to find the endpoint, which it will use to establish connection to the broker.  For example, if the broker's published endpoints on zookeeper are :  "endpoints" : ["INTERNAL://broker1.example.com:9092","EXTERNAL://broker1.example.com:9093","CONTROLLER://broker1.example.com:9094"]  and the controller's config is :  listener.security.protocol.map = INTERNAL:PLAINTEXT, EXTERNAL:SSL, CONTROLLER:SSL  control.plane.listener.name = CONTROLLER  then controller will use "broker1.example.com:9094" with security protocol "SSL" to connect to the broker.  If not explicitly configured, the default value will be null and there will be no dedicated endpoints for controller connections.  Default: null |
| delete.topic.enable | 是否允许非admin权限删除topic  默认为true  如果设置为false，不影响admin tool删除topic |
| host.name | DEPRECATED: only used when listeners is not set. Use listeners instead.  hostname of broker. If this is set, it will only bind to this address. If this is not set, it will bind to all interfaces  Default: "" |
| leader.imbalance.check.interval.seconds | 定期检查leader负载情况  默认300s |
| leader.imbalance.per.broker.percentage | 每个leader最高负载率  默认10 |
| listeners | Listener List - Comma-separated list of URIs we will listen on and the listener names. If the listener name is not a security protocol, listener.security.protocol.map must also be set.  Listener names and port numbers must be unique.  Specify hostname as 0.0.0.0 to bind to all interfaces.  Leave hostname empty to bind to default interface.  Examples of legal listener lists:  PLAINTEXT://myhost:9092,SSL://:9091  CLIENT://0.0.0.0:9092,REPLICATION://localhost:9093  Default: null |
| log.dir | 指定log目录  当log.dirs未设置时，生效 |
| log.dirs | 指定多个log目录  如果未设置则取log.dir配置 |
| log.flush.interval.messages | log partation积累的消息flush到disk的阈值  默认9223372036854775807  取值为大于0的整数 |
| log.flush.interval.ms | 定时将消息从memory flush to disk  默认不设置  如果不设置，则取log.flush.scheduler.interval.ms值 |
| log.flush.offset.checkpoint.interval.ms | 设置更新最后flush to disk的记录信息的频率，相当于定时更新恢复点  默认60000ms  取值大于等于0整数 |
| log.flush.scheduler.interval.ms | 定时检查日志是否flush to disk  默认9223372036854775807ms  取值大于等于0整数 |
| log.flush.start.offset.checkpoint.interval.ms | 定时更新当前检查点记录对应的起始offset  默认60000ms (1分钟) |
| log.retention.bytes | 日志保留最大大小  默认 -1 |
| log.retention.hours | 消息日志保留时间小时，  默认168 h（7d） |
| log.retention.minutes | 消息日志保留时间分钟，  如果不设置则按log.retention.hours配置处理  默认null |
| log.retention.ms | 消息日志保留时间毫秒  如果不设置则按log.retention.minutes配置处理  默认null  如果设置为-1，则永久保留消息 |
| log.roll.hours | 当前日志文件创建时间距离当前时间的小时差大于指定值，则切分日志文件  默认168 h（7d）  仅当log.roll.ms没配置时生效 |
| log.roll.jitter.hours |  |
| log.roll.jitter.ms |  |
| log.roll.ms | 当前日志文件创建时间距离当前时间的毫秒差大于指定值，则切分日志文件  默认不设置，采用log.roll.hours配置 |
| log.segment.bytes | The maximum size of a single log file  Default: 1073741824 (1 gibibyte)  Valid Values: [14,...] |
| log.segment.delete.delay.ms | The amount of time to wait before deleting a file from the filesystem  Default: 60000 (1 minute)  Valid Values: [0,...] |
| message.max.bytes | The largest record batch size allowed by Kafka (after compression if compression is enabled). If this is increased and there are consumers older than 0.10.2, the consumers' fetch size must also be increased so that they can fetch record batches this large. In the latest message format version, records are always grouped into batches for efficiency. In previous message format versions, uncompressed records are not grouped into batches and this limit only applies to a single record in that case.This can be set per topic with the topic level max.message.bytes config.  Default: 1048588  Valid Values: [0,...] |
| min.insync.replicas | When a producer sets acks to "all" (or "-1"), min.insync.replicas specifies the minimum number of replicas that must acknowledge a write for the write to be considered successful. If this minimum cannot be met, then the producer will raise an exception (either NotEnoughReplicas or NotEnoughReplicasAfterAppend).  When used together, min.insync.replicas and acks allow you to enforce greater durability guarantees. A typical scenario would be to create a topic with a replication factor of 3, set min.insync.replicas to 2, and produce with acks of "all". This will ensure that the producer raises an exception if a majority of replicas do not receive a write.  Default: 1  Valid Values: [1,...] |
| num.io.threads | The number of threads that the server uses for processing requests, which may include disk I/O  Default: 8  Valid Values: [1,...] |
| num.network.threads | The number of threads that the server uses for receiving requests from the network and sending responses to the network  Default: 3  Valid Values: [1,...] |
| num.recovery.threads.per.data.dir | The number of threads per data directory to be used for log recovery at startup and flushing at shutdown  Default: 1  Valid Values: [1,...] |
| num.replica.alter.log.dirs.threads | The number of threads that can move replicas between log directories, which may include disk I/O  Default: null |
| num.replica.fetchers | Number of fetcher threads used to replicate messages from a source broker. Increasing this value can increase the degree of I/O parallelism in the follower broker.  Default: 1 |
| offset.metadata.max.bytes | The maximum size for a metadata entry associated with an offset commit  Default: 4096 (4 kibibytes) |
| offsets.commit.required.acks | The required acks before the commit can be accepted. In general, the default (-1) should not be overridden  Default: -1 |
| offsets.commit.timeout.ms | Offset commit will be delayed until all replicas for the offsets topic receive the commit or this timeout is reached. This is similar to the producer request timeout.  Default: 5000 (5 seconds)  Valid Values: [1,...] |
| offsets.load.buffer.size | Batch size for reading from the offsets segments when loading offsets into the cache (soft-limit, overridden if records are too large).  Default: 5242880  Valid Values: [1,...] |
| offsets.retention.check.interval.ms | Frequency at which to check for stale offsets  Default: 600000 (10 minutes)  Valid Values: [1,...] |
| offsets.retention.minutes | After a consumer group loses all its consumers (i.e. becomes empty) its offsets will be kept for this retention period before getting discarded. For standalone consumers (using manual assignment), offsets will be expired after the time of last commit plus this retention period.  Default: 10080  Valid Values: [1,...] |
| offsets.topic.compression.codec | Compression codec for the offsets topic - compression may be used to achieve "atomic" commits  Default: 0 |
| offsets.topic.num.partitions | The number of partitions for the offset commit topic (should not change after deployment)  Default: 50  Valid Values: [1,...] |
| offsets.topic.replication.factor | The replication factor for the offsets topic (set higher to ensure availability). Internal topic creation will fail until the cluster size meets this replication factor requirement.  Default: 3  Valid Values: [1,...] |
| offsets.topic.segment.bytes | The offsets topic segment bytes should be kept relatively small in order to facilitate faster log compaction and cache loads  Default: 104857600 (100 mebibytes)  Valid Values: [1,...] |
| port | DEPRECATED: only used when listeners is not set. Use listeners instead.  the port to listen and accept connections on  Default: 9092 |
| queued.max.requests | The number of queued requests allowed for data-plane, before blocking the network threads  Default: 500  Valid Values: [1,...] |
| quota.consumer.default | DEPRECATED: Used only when dynamic default quotas are not configured for or in Zookeeper. Any consumer distinguished by clientId/consumer group will get throttled if it fetches more bytes than this value per-second  Default: 9223372036854775807  Valid Values: [1,...] |
| quota.producer.default | DEPRECATED: Used only when dynamic default quotas are not configured for , or in Zookeeper. Any producer distinguished by clientId will get throttled if it produces more bytes than this value per-second  Default: 9223372036854775807  Valid Values: [1,...] |
| replica.fetch.min.bytes | Minimum bytes expected for each fetch response. If not enough bytes, wait up to replica.fetch.wait.max.ms (broker config).  Default: 1 |
| replica.fetch.wait.max.ms | max wait time for each fetcher request issued by follower replicas. This value should always be less than the replica.lag.time.max.ms at all times to prevent frequent shrinking of ISR for low throughput topics  Default: 500 |
| replica.high.watermark.checkpoint.interval.ms | The frequency with which the high watermark is saved out to disk  Default: 5000 (5 seconds) |
| replica.lag.time.max.ms | If a follower hasn't sent any fetch requests or hasn't consumed up to the leaders log end offset for at least this time, the leader will remove the follower from isr  Default: 30000 (30 seconds) |
| replica.socket.receive.buffer.bytes | The socket receive buffer for network requests  Default: 65536 (64 kibibytes) |
| replica.socket.timeout.ms | The socket timeout for network requests. Its value should be at least replica.fetch.wait.max.ms  Default: 30000 (30 seconds) |
| request.timeout.ms | The configuration controls the maximum amount of time the client will wait for the response of a request. If the response is not received before the timeout elapses the client will resend the request if necessary or fail the request if retries are exhausted.  Default: 30000 (30 seconds) |
| socket.receive.buffer.bytes | The SO\_RCVBUF buffer of the socket server sockets. If the value is -1, the OS default will be used.  Default: 102400 (100 kibibytes) |
| socket.request.max.bytes | The maximum number of bytes in a socket request  Default: 104857600 (100 mebibytes)  Valid Values: [1,...] |
| socket.send.buffer.bytes | The SO\_SNDBUF buffer of the socket server sockets. If the value is -1, the OS default will be used.  Default: 102400 (100 kibibytes) |
| transaction.max.timeout.ms | The maximum allowed timeout for transactions. If a client’s requested transaction time exceed this, then the broker will return an error in InitProducerIdRequest. This prevents a client from too large of a timeout, which can stall consumers reading from topics included in the transaction.  Default: 900000 (15 minutes)  Valid Values: [1,...] |
| transaction.state.log.load.buffer.size | Batch size for reading from the transaction log segments when loading producer ids and transactions into the cache (soft-limit, overridden if records are too large).  Default: 5242880  Valid Values: [1,...] |
| transaction.state.log.min.isr | Overridden min.insync.replicas config for the transaction topic.  Default: 2  Valid Values: [1,...] |
| transaction.state.log.num.partitions | The number of partitions for the transaction topic (should not change after deployment).  Default: 50  Valid Values: [1,...] |
| transaction.state.log.replication.factor | The replication factor for the transaction topic (set higher to ensure availability). Internal topic creation will fail until the cluster size meets this replication factor requirement.  Default: 3  Valid Values: [1,...] |
| transaction.state.log.segment.bytes | The transaction topic segment bytes should be kept relatively small in order to facilitate faster log compaction and cache loads  Default: 104857600 (100 mebibytes)  Valid Values: [1,...] |
| transactional.id.expiration.ms | The time in ms that the transaction coordinator will wait without receiving any transaction status updates for the current transaction before expiring its transactional id. This setting also influences producer id expiration - producer ids are expired once this time has elapsed after the last write with the given producer id. Note that producer ids may expire sooner if the last write from the producer id is deleted due to the topic's retention settings.  Default: 604800000 (7 days)  Valid Values: [1,...] |
| unclean.leader.election.enable | Indicates whether to enable replicas not in the ISR set to be elected as leader as a last resort, even though doing so may result in data loss  Default: false |
| zookeeper.connection.timeout.ms | The max time that the client waits to establish a connection to zookeeper. If not set, the value in zookeeper.session.timeout.ms is used  Default: null |
| zookeeper.max.in.flight.requests | The maximum number of unacknowledged requests the client will send to Zookeeper before blocking.  Default: 10  Valid Values: [1,...] |
| zookeeper.session.timeout.ms | Zookeeper session timeout  Default: 18000 (18 seconds) |
| zookeeper.set.acl | Set client to use secure ACLs  Default: false |
| broker.id.generation.enable | Enable automatic broker id generation on the server. When enabled the value configured for reserved.broker.max.id should be reviewed.  Default: true |
| broker.rack | Rack of the broker. This will be used in rack aware replication assignment for fault tolerance. Examples: `RACK1`, `us-east-1d`  Default: null |
| connections.max.idle.ms | Idle connections timeout: the server socket processor threads close the connections that idle more than this  Default: 600000 (10 minutes) |
| connections.max.reauth.ms | When explicitly set to a positive number (the default is 0, not a positive number), a session lifetime that will not exceed the configured value will be communicated to v2.2.0 or later clients when they authenticate. The broker will disconnect any such connection that is not re-authenticated within the session lifetime and that is then subsequently used for any purpose other than re-authentication. Configuration names can optionally be prefixed with listener prefix and SASL mechanism name in lower-case. For example, listener.name.sasl\_ssl.oauthbearer.connections.max.reauth.ms=3600000  Default: 0 |
| controlled.shutdown.enable | Enable controlled shutdown of the server  Default: true |
| controlled.shutdown.max.retries | Controlled shutdown can fail for multiple reasons. This determines the number of retries when such failure happens  Default: 3 |
| controlled.shutdown.retry.backoff.ms | Before each retry, the system needs time to recover from the state that caused the previous failure (Controller fail over, replica lag etc). This config determines the amount of time to wait before retrying.  Default: 5000 (5 seconds) |
| controller.socket.timeout.ms | The socket timeout for controller-to-broker channels  Default: 30000 (30 seconds) |
| default.replication.factor | default replication factors for automatically created topics  Default: 1 |
| delegation.token.expiry.time.ms | The token validity time in miliseconds before the token needs to be renewed. Default value 1 day.  Default: 86400000 (1 day)  Valid Values: [1,...] |
| delegation.token.master.key | Master/secret key to generate and verify delegation tokens. Same key must be configured across all the brokers. If the key is not set or set to empty string, brokers will disable the delegation token support.  Default: null |
| delegation.token.max.lifetime.ms | The token has a maximum lifetime beyond which it cannot be renewed anymore. Default value 7 days.  Default: 604800000 (7 days)  Valid Values: [1,...] |
| delete.records.purgatory.purge.interval.requests | The purge interval (in number of requests) of the delete records request purgatory  Default: 1 |
| fetch.max.bytes | The maximum number of bytes we will return for a fetch request. Must be at least 1024.  Default: 57671680 (55 mebibytes)  Valid Values: [1024,...] |
| fetch.purgatory.purge.interval.requests | The purge interval (in number of requests) of the fetch request purgatory  Default: 1000 |
| group.initial.rebalance.delay.ms | The amount of time the group coordinator will wait for more consumers to join a new group before performing the first rebalance. A longer delay means potentially fewer rebalances, but increases the time until processing begins.  Default: 3000 (3 seconds) |
| group.max.session.timeout.ms | The maximum allowed session timeout for registered consumers. Longer timeouts give consumers more time to process messages in between heartbeats at the cost of a longer time to detect failures.  Default: 1800000 (30 minutes) |
| group.max.size | The maximum number of consumers that a single consumer group can accommodate.  Default: 2147483647  Valid Values: [1,...] |
| group.min.session.timeout.ms | The minimum allowed session timeout for registered consumers. Shorter timeouts result in quicker failure detection at the cost of more frequent consumer heartbeating, which can overwhelm broker resources.  Default: 6000 (6 seconds |
| inter.broker.listener.name | Name of listener used for communication between brokers. If this is unset, the listener name is defined by security.inter.broker.protocol. It is an error to set this and security.inter.broker.protocol properties at the same time.  Default: null |
| inter.broker.protocol.version | Specify which version of the inter-broker protocol will be used.  This is typically bumped after all brokers were upgraded to a new version.  Example of some valid values are: 0.8.0, 0.8.1, 0.8.1.1, 0.8.2, 0.8.2.0, 0.8.2.1, 0.9.0.0, 0.9.0.1 Check ApiVersion for the full list.  Default: 2.7-IV2  Valid Values: [0.8.0, 0.8.1, 0.8.2, 0.9.0, 0.10.0-IV0, 0.10.0-IV1, 0.10.1-IV0, 0.10.1-IV1, 0.10.1-IV2, 0.10.2-IV0, 0.11.0-IV0, 0.11.0-IV1, 0.11.0-IV2, 1.0-IV0, 1.1-IV0, 2.0-IV0, 2.0-IV1, 2.1-IV0, 2.1-IV1, 2.1-IV2, 2.2-IV0, 2.2-IV1, 2.3-IV0, 2.3-IV1, 2.4-IV0, 2.4-IV1, 2.5-IV0, 2.6-IV0, 2.7-IV0, 2.7-IV1, 2.7-IV2] |
| log.cleaner.backoff.ms | The amount of time to sleep when there are no logs to clean  Default: 15000 (15 seconds)  Valid Values: [0,...] |
| log.cleaner.dedupe.buffer.size | The total memory used for log deduplication across all cleaner threads  Default: 134217728 |
| log.cleaner.delete.retention.ms | How long are delete records retained?  Default: 86400000 (1 day) |
| log.cleaner.enable | Enable the log cleaner process to run on the server. Should be enabled if using any topics with a cleanup.policy=compact including the internal offsets topic. If disabled those topics will not be compacted and continually grow in size.  Default: true |
| log.cleaner.io.buffer.load.factor | Log cleaner dedupe buffer load factor. The percentage full the dedupe buffer can become. A higher value will allow more log to be cleaned at once but will lead to more hash collisions  Default: 0.9 |
| log.cleaner.io.buffer.size | The total memory used for log cleaner I/O buffers across all cleaner threads  Default: 524288  Valid Values: [0,...] |
| log.cleaner.io.max.bytes.per.second | The log cleaner will be throttled so that the sum of its read and write i/o will be less than this value on average  Default: 1.7976931348623157E308 |
| log.cleaner.max.compaction.lag.ms | The maximum time a message will remain ineligible for compaction in the log. Only applicable for logs that are being compacted.  Default: 9223372036854775807 |
| log.cleaner.min.cleanable.ratio | The minimum ratio of dirty log to total log for a log to eligible for cleaning. If the log.cleaner.max.compaction.lag.ms or the log.cleaner.min.compaction.lag.ms configurations are also specified, then the log compactor considers the log eligible for compaction as soon as either: (i) the dirty ratio threshold has been met and the log has had dirty (uncompacted) records for at least the log.cleaner.min.compaction.lag.ms duration, or (ii) if the log has had dirty (uncompacted) records for at most the log.cleaner.max.compaction.lag.ms period.  Default: 0.5 |
| log.cleaner.min.compaction.lag.ms | The minimum time a message will remain uncompacted in the log. Only applicable for logs that are being compacted.  Default: 0 |
| log.cleaner.threads | The number of background threads to use for log cleaning  Default: 1  Valid Values: [0,...] |
| log.cleanup.policy | The default cleanup policy for segments beyond the retention window. A comma separated list of valid policies. Valid policies are: "delete" and "compact"  Default: delete  Valid Values: [compact, delete] |
| log.index.interval.bytes | The interval with which we add an entry to the offset index  Default: 4096 (4 kibibytes)  Valid Values: [0,...] |
| log.index.size.max.bytes | The maximum size in bytes of the offset index  Default: 10485760 (10 mebibytes)  Valid Values: [4,...] |
| log.message.format.version | Specify the message format version the broker will use to append messages to the logs. The value should be a valid ApiVersion. Some examples are: 0.8.2, 0.9.0.0, 0.10.0, check ApiVersion for more details. By setting a particular message format version, the user is certifying that all the existing messages on disk are smaller or equal than the specified version. Setting this value incorrectly will cause consumers with older versions to break as they will receive messages with a format that they don't understand.  Default: 2.7-IV2  [0.8.0, 0.8.1, 0.8.2, 0.9.0, 0.10.0-IV0, 0.10.0-IV1, 0.10.1-IV0, 0.10.1-IV1, 0.10.1-IV2, 0.10.2-IV0, 0.11.0-IV0, 0.11.0-IV1, 0.11.0-IV2, 1.0-IV0, 1.1-IV0, 2.0-IV0, 2.0-IV1, 2.1-IV0, 2.1-IV1, 2.1-IV2, 2.2-IV0, 2.2-IV1, 2.3-IV0, 2.3-IV1, 2.4-IV0, 2.4-IV1, 2.5-IV0, 2.6-IV0, 2.7-IV0, 2.7-IV1, 2.7-IV2] |
| log.message.timestamp.difference.max.ms | The maximum difference allowed between the timestamp when a broker receives a message and the timestamp specified in the message. If log.message.timestamp.type=CreateTime, a message will be rejected if the difference in timestamp exceeds this threshold. This configuration is ignored if log.message.timestamp.type=LogAppendTime.The maximum timestamp difference allowed should be no greater than log.retention.ms to avoid unnecessarily frequent log rolling.  Default: 9223372036854775807 |
| log.message.timestamp.type | Define whether the timestamp in the message is message create time or log append time. The value should be either `CreateTime` or `LogAppendTime`  Default: CreateTime  Valid Values: [CreateTime, LogAppendTime] |
| log.preallocate | Should pre allocate file when create new segment? If you are using Kafka on Windows, you probably need to set it to true.  Default: false |
| log.retention.check.interval.ms | The frequency in milliseconds that the log cleaner checks whether any log is eligible for deletion  Default: 300000 (5 minutes)  Valid Values: [1,...] |
| max.connection.creation.rate | The maximum connection creation rate we allow in the broker at any time. Listener-level limits may also be configured by prefixing the config name with the listener prefix, for example, listener.name.internal.max.connection.creation.rate.Broker-wide connection rate limit should be configured based on broker capacity while listener limits should be configured based on application requirements. New connections will be throttled if either the listener or the broker limit is reached, with the exception of inter-broker listener. Connections on the inter-broker listener will be throttled only when the listener-level rate limit is reached.  Default: 2147483647  Valid Values: [0,...] |
| max.connections | The maximum number of connections we allow in the broker at any time. This limit is applied in addition to any per-ip limits configured using max.connections.per.ip. Listener-level limits may also be configured by prefixing the config name with the listener prefix, for example, listener.name.internal.max.connections. Broker-wide limit should be configured based on broker capacity while listener limits should be configured based on application requirements. New connections are blocked if either the listener or broker limit is reached. Connections on the inter-broker listener are permitted even if broker-wide limit is reached. The least recently used connection on another listener will be closed in this case.  Default: 2147483647  Valid Values: [0,...] |
| max.connections.per.ip | The maximum number of connections we allow from each ip address. This can be set to 0 if there are overrides configured using max.connections.per.ip.overrides property. New connections from the ip address are dropped if the limit is reached.  Default: 2147483647  Valid Values: [0,...] |
| max.connections.per.ip.overrides | A comma-separated list of per-ip or hostname overrides to the default maximum number of connections. An example value is "hostName:100,127.0.0.1:200"  Default: "" |
| max.incremental.fetch.session.cache.slots | The maximum number of incremental fetch sessions that we will maintain.  Default: 1000  Valid Values: [0,...] |
| num.partitions | The default number of log partitions per topic  Default: 1  Valid Values: [1,...] |
| password.encoder.old.secret | The old secret that was used for encoding dynamically configured passwords. This is required only when the secret is updated. If specified, all dynamically encoded passwords are decoded using this old secret and re-encoded using password.encoder.secret when broker starts up.  Default: null |
| password.encoder.secret | The secret used for encoding dynamically configured passwords for this broker.  Default: null |
| principal.builder.class | The fully qualified name of a class that implements the KafkaPrincipalBuilder interface, which is used to build the KafkaPrincipal object used during authorization. This config also supports the deprecated PrincipalBuilder interface which was previously used for client authentication over SSL. If no principal builder is defined, the default behavior depends on the security protocol in use. For SSL authentication, the principal will be derived using the rules defined by ssl.principal.mapping.rules applied on the distinguished name from the client certificate if one is provided; otherwise, if client authentication is not required, the principal name will be ANONYMOUS. For SASL authentication, the principal will be derived using the rules defined by sasl.kerberos.principal.to.local.rules if GSSAPI is in use, and the SASL authentication ID for other mechanisms. For PLAINTEXT, the principal will be ANONYMOUS.  Default: null |
| producer.purgatory.purge.interval.requests | The purge interval (in number of requests) of the producer request purgatory  Default: 1000 |
| queued.max.request.bytes | The number of queued bytes allowed before no more requests are read  Default: -1 |
| replica.fetch.backoff.ms | The amount of time to sleep when fetch partition error occurs.  Default: 1000 (1 second)  Valid Values: [0,...] |
| replica.fetch.max.bytes | The number of bytes of messages to attempt to fetch for each partition. This is not an absolute maximum, if the first record batch in the first non-empty partition of the fetch is larger than this value, the record batch will still be returned to ensure that progress can be made. The maximum record batch size accepted by the broker is defined via message.max.bytes (broker config) or max.message.bytes (topic config).  Default: 1048576 (1 mebibyte)  Valid Values: [0,...] |
| replica.fetch.response.max.bytes | Maximum bytes expected for the entire fetch response. Records are fetched in batches, and if the first record batch in the first non-empty partition of the fetch is larger than this value, the record batch will still be returned to ensure that progress can be made. As such, this is not an absolute maximum. The maximum record batch size accepted by the broker is defined via message.max.bytes (broker config) or max.message.bytes (topic config).  Default: 10485760 (10 mebibytes)  Valid Values: [0,...] |
| replica.selector.class | The fully qualified class name that implements ReplicaSelector. This is used by the broker to find the preferred read replica. By default, we use an implementation that returns the leader.  Default: null |
| reserved.broker.max.id | Max number that can be used for a broker.id  Default: 1000  Valid Values: [0,...] |
| sasl.client.callback.handler.class | The fully qualified name of a SASL client callback handler class that implements the AuthenticateCallbackHandler interface.  Default: null |
| sasl.enabled.mechanisms | The list of SASL mechanisms enabled in the Kafka server. The list may contain any mechanism for which a security provider is available. Only GSSAPI is enabled by default.  Default: GSSAPI |
| sasl.jaas.config | JAAS login context parameters for SASL connections in the format used by JAAS configuration files. JAAS configuration file format is described here. The format for the value is: 'loginModuleClass controlFlag (optionName=optionValue)\*;'. For brokers, the config must be prefixed with listener prefix and SASL mechanism name in lower-case. For example, listener.name.sasl\_ssl.scram-sha-256.sasl.jaas.config=com.example.ScramLoginModule required;  Default: null |
| sasl.kerberos.kinit.cmd | Kerberos kinit command path.  Default: /usr/bin/kinit |
| sasl.kerberos.min.time.before.relogin | Login thread sleep time between refresh attempts.  Default: 60000 |
| sasl.kerberos.principal.to.local.rules | A list of rules for mapping from principal names to short names (typically operating system usernames). The rules are evaluated in order and the first rule that matches a principal name is used to map it to a short name. Any later rules in the list are ignored. By default, principal names of the form {username}/{hostname}@{REALM} are mapped to {username}. For more details on the format please see security authorization and acls. Note that this configuration is ignored if an extension of KafkaPrincipalBuilder is provided by the principal.builder.class configuration.  Default: DEFAULT |
| sasl.kerberos.service.name | The Kerberos principal name that Kafka runs as. This can be defined either in Kafka's JAAS config or in Kafka's config.  Default: null |
| sasl.kerberos.ticket.renew.jitter | Percentage of random jitter added to the renewal time.  Default: 0.05 |
| sasl.kerberos.ticket.renew.window.factor | Login thread will sleep until the specified window factor of time from last refresh to ticket's expiry has been reached, at which time it will try to renew the ticket.  Default: 0.8 |
| sasl.login.callback.handler.class | The fully qualified name of a SASL login callback handler class that implements the AuthenticateCallbackHandler interface. For brokers, login callback handler config must be prefixed with listener prefix and SASL mechanism name in lower-case. For example, listener.name.sasl\_ssl.scram-sha-256.sasl.login.callback.handler.class=com.example.CustomScramLoginCallbackHandler  Default: null |
| sasl.login.class | The fully qualified name of a class that implements the Login interface. For brokers, login config must be prefixed with listener prefix and SASL mechanism name in lower-case. For example, listener.name.sasl\_ssl.scram-sha-256.sasl.login.class=com.example.CustomScramLogin  Default: null |
| sasl.login.refresh.buffer.seconds | The amount of buffer time before credential expiration to maintain when refreshing a credential, in seconds. If a refresh would otherwise occur closer to expiration than the number of buffer seconds then the refresh will be moved up to maintain as much of the buffer time as possible. Legal values are between 0 and 3600 (1 hour); a default value of 300 (5 minutes) is used if no value is specified. This value and sasl.login.refresh.min.period.seconds are both ignored if their sum exceeds the remaining lifetime of a credential. Currently applies only to OAUTHBEARER.  Default: 300 |
| sasl.login.refresh.min.period.seconds | The desired minimum time for the login refresh thread to wait before refreshing a credential, in seconds. Legal values are between 0 and 900 (15 minutes); a default value of 60 (1 minute) is used if no value is specified. This value and sasl.login.refresh.buffer.seconds are both ignored if their sum exceeds the remaining lifetime of a credential. Currently applies only to OAUTHBEARER.  Default: 60 |
| sasl.login.refresh.window.factor | Login refresh thread will sleep until the specified window factor relative to the credential's lifetime has been reached, at which time it will try to refresh the credential. Legal values are between 0.5 (50%) and 1.0 (100%) inclusive; a default value of 0.8 (80%) is used if no value is specified. Currently applies only to OAUTHBEARER.  Default: 0.8 |
| sasl.login.refresh.window.jitter | The maximum amount of random jitter relative to the credential's lifetime that is added to the login refresh thread's sleep time. Legal values are between 0 and 0.25 (25%) inclusive; a default value of 0.05 (5%) is used if no value is specified. Currently applies only to OAUTHBEARER.  Default: 0.05 |
| sasl.mechanism.inter.broker.protocol | SASL mechanism used for inter-broker communication. Default is GSSAPI.  Default: GSSAPI |
| sasl.server.callback.handler.class | The fully qualified name of a SASL server callback handler class that implements the AuthenticateCallbackHandler interface. Server callback handlers must be prefixed with listener prefix and SASL mechanism name in lower-case. For example, listener.name.sasl\_ssl.plain.sasl.server.callback.handler.class=com.example.CustomPlainCallbackHandler.  Default: null |
| security.inter.broker.protocol | Security protocol used to communicate between brokers. Valid values are: PLAINTEXT, SSL, SASL\_PLAINTEXT, SASL\_SSL. It is an error to set this and inter.broker.listener.name properties at the same time.  Default: PLAINTEXT |
| socket.connection.setup.timeout.max.ms | The maximum amount of time the client will wait for the socket connection to be established. The connection setup timeout will increase exponentially for each consecutive connection failure up to this maximum. To avoid connection storms, a randomization factor of 0.2 will be applied to the timeout resulting in a random range between 20% below and 20% above the computed value.  Default: 127000 (127 seconds) |
| socket.connection.setup.timeout.ms | The amount of time the client will wait for the socket connection to be established. If the connection is not built before the timeout elapses, clients will close the socket channel.  Default: 10000 (10 seconds) |
| ssl.cipher.suites | A list of cipher suites. This is a named combination of authentication, encryption, MAC and key exchange algorithm used to negotiate the security settings for a network connection using TLS or SSL network protocol. By default all the available cipher suites are supported.  Default: "" |
| ssl.client.auth | Configures kafka broker to request client authentication. The following settings are common:  ssl.client.auth=required If set to required client authentication is required.  ssl.client.auth=requested This means client authentication is optional. unlike required, if this option is set client can choose not to provide authentication information about itself  ssl.client.auth=none This means client authentication is not needed.  Default: none  Valid Values: [required, requested, none] |
| ssl.enabled.protocols | The list of protocols enabled for SSL connections. The default is 'TLSv1.2,TLSv1.3' when running with Java 11 or newer, 'TLSv1.2' otherwise. With the default value for Java 11, clients and servers will prefer TLSv1.3 if both support it and fallback to TLSv1.2 otherwise (assuming both support at least TLSv1.2). This default should be fine for most cases. Also see the config documentation for `ssl.protocol`.  Default: TLSv1.2 |
| ssl.key.password | The password of the private key in the key store file orthe PEM key specified in `ssl.keystore.key'. This is required for clients only if two-way authentication is configured.  Default: null |
| ssl.keymanager.algorithm | The algorithm used by key manager factory for SSL connections. Default value is the key manager factory algorithm configured for the Java Virtual Machine.  Default: SunX509 |
| ssl.keystore.certificate.chain | Certificate chain in the format specified by 'ssl.keystore.type'. Default SSL engine factory supports only PEM format with a list of X.509 certificates  Default: null |
| ssl.keystore.key | Private key in the format specified by 'ssl.keystore.type'. Default SSL engine factory supports only PEM format with PKCS#8 keys. If the key is encrypted, key password must be specified using 'ssl.key.password'  Default: null |
| ssl.keystore.location | The location of the key store file. This is optional for client and can be used for two-way authentication for client.  Default: null |
| ssl.keystore.password | The store password for the key store file. This is optional for client and only needed if 'ssl.keystore.location' is configured. Key store password is not supported for PEM format.  Default: null |
| ssl.keystore.type | The file format of the key store file. This is optional for client.  Default: JKS |
| ssl.protocol | The SSL protocol used to generate the SSLContext. The default is 'TLSv1.3' when running with Java 11 or newer, 'TLSv1.2' otherwise. This value should be fine for most use cases. Allowed values in recent JVMs are 'TLSv1.2' and 'TLSv1.3'. 'TLS', 'TLSv1.1', 'SSL', 'SSLv2' and 'SSLv3' may be supported in older JVMs, but their usage is discouraged due to known security vulnerabilities. With the default value for this config and 'ssl.enabled.protocols', clients will downgrade to 'TLSv1.2' if the server does not support 'TLSv1.3'. If this config is set to 'TLSv1.2', clients will not use 'TLSv1.3' even if it is one of the values in ssl.enabled.protocols and the server only supports 'TLSv1.3'.  Default: TLSv1.2 |
| ssl.provider | The name of the security provider used for SSL connections. Default value is the default security provider of the JVM.  Default: null |
| ssl.trustmanager.algorithm | The algorithm used by trust manager factory for SSL connections. Default value is the trust manager factory algorithm configured for the Java Virtual Machine.  Default: PKIX |
| ssl.truststore.certificates | Trusted certificates in the format specified by 'ssl.truststore.type'. Default SSL engine factory supports only PEM format with X.509 certificates.  Default: null |
| ssl.truststore.location | The location of the trust store file.  Default: null |
| ssl.truststore.password | The password for the trust store file. If a password is not set, trust store file configured will still be used, but integrity checking is disabled. Trust store password is not supported for PEM format.  Default: null |
| ssl.truststore.type | The file format of the trust store file.  Default: JKS |
| zookeeper.clientCnxnSocket | Typically set to org.apache.zookeeper.ClientCnxnSocketNetty when using TLS connectivity to ZooKeeper. Overrides any explicit value set via the same-named zookeeper.clientCnxnSocket system property.  Default: null |
| zookeeper.ssl.client.enable | Set client to use TLS when connecting to ZooKeeper. An explicit value overrides any value set via the zookeeper.client.secure system property (note the different name). Defaults to false if neither is set; when true, zookeeper.clientCnxnSocket must be set (typically to org.apache.zookeeper.ClientCnxnSocketNetty); other values to set may include zookeeper.ssl.cipher.suites, zookeeper.ssl.crl.enable, zookeeper.ssl.enabled.protocols, zookeeper.ssl.endpoint.identification.algorithm, zookeeper.ssl.keystore.location, zookeeper.ssl.keystore.password, zookeeper.ssl.keystore.type, zookeeper.ssl.ocsp.enable, zookeeper.ssl.protocol, zookeeper.ssl.truststore.location, zookeeper.ssl.truststore.password, zookeeper.ssl.truststore.type  Default: false |
| zookeeper.ssl.keystore.location | Keystore location when using a client-side certificate with TLS connectivity to ZooKeeper. Overrides any explicit value set via the zookeeper.ssl.keyStore.location system property (note the camelCase).  Default: null |
| zookeeper.ssl.keystore.password | Keystore password when using a client-side certificate with TLS connectivity to ZooKeeper. Overrides any explicit value set via the zookeeper.ssl.keyStore.password system property (note the camelCase). Note that ZooKeeper does not support a key password different from the keystore password, so be sure to set the key password in the keystore to be identical to the keystore password; otherwise the connection attempt to Zookeeper will fail.  Default: null |
| zookeeper.ssl.keystore.type | Keystore type when using a client-side certificate with TLS connectivity to ZooKeeper. Overrides any explicit value set via the zookeeper.ssl.keyStore.type system property (note the camelCase). The default value of null means the type will be auto-detected based on the filename extension of the keystore.  Default: null |
| zookeeper.ssl.truststore.location | Truststore location when using TLS connectivity to ZooKeeper. Overrides any explicit value set via the zookeeper.ssl.trustStore.location system property (note the camelCase).  Default: null |
| zookeeper.ssl.truststore.password | Truststore password when using TLS connectivity to ZooKeeper. Overrides any explicit value set via the zookeeper.ssl.trustStore.password system property (note the camelCase).  Default: null |
| zookeeper.ssl.truststore.type | Truststore type when using TLS connectivity to ZooKeeper. Overrides any explicit value set via the zookeeper.ssl.trustStore.type system property (note the camelCase). The default value of null means the type will be auto-detected based on the filename extension of the truststore.  Default: null |
| alter.config.policy.class.name | The alter configs policy class that should be used for validation. The class should implement the org.apache.kafka.server.policy.AlterConfigPolicy interface.  Default: null |
| alter.log.dirs.replication.quota.window.num | The number of samples to retain in memory for alter log dirs replication quotas  Default: 11  Valid Values: [1,...] |
| alter.log.dirs.replication.quota.window.size.seconds | The time span of each sample for alter log dirs replication quotas  Default: 1  Valid Values: [1,...] |
| authorizer.class.name | The fully qualified name of a class that implements sorg.apache.kafka.server.authorizer.Authorizer interface, which is used by the broker for authorization. This config also supports authorizers that implement the deprecated kafka.security.auth.Authorizer trait which was previously used for authorization.  Default: "" |
| client.quota.callback.class | The fully qualified name of a class that implements the ClientQuotaCallback interface, which is used to determine quota limits applied to client requests. By default, , or quotas stored in ZooKeeper are applied. For any given request, the most specific quota that matches the user principal of the session and the client-id of the request is applied.  Default: null |
| connection.failed.authentication.delay.ms | Connection close delay on failed authentication: this is the time (in milliseconds) by which connection close will be delayed on authentication failure. This must be configured to be less than connections.max.idle.ms to prevent connection timeout.  Default: 100  Valid Values: [0,...] |
| controller.quota.window.num | The number of samples to retain in memory for controller mutation quotas  Default: 11  Valid Values: [1,...] |
| controller.quota.window.size.seconds | The time span of each sample for controller mutations quotas  Default: 1  Valid Values: [1,...] |
| create.topic.policy.class.name | The create topic policy class that should be used for validation. The class should implement the org.apache.kafka.server.policy.CreateTopicPolicy interface.  Default: null |
| delegation.token.expiry.check.interval.ms | Scan interval to remove expired delegation tokens.  Default: 3600000 (1 hour)  Valid Values: [1,...] |
| kafka.metrics.polling.interval.secs | The metrics polling interval (in seconds) which can be used in kafka.metrics.reporters implementations.  Default: 10  Valid Values: [1,...] |
| kafka.metrics.reporters | A list of classes to use as Yammer metrics custom reporters. The reporters should implement kafka.metrics.KafkaMetricsReporter trait. If a client wants to expose JMX operations on a custom reporter, the custom reporter needs to additionally implement an MBean trait that extends kafka.metrics.KafkaMetricsReporterMBean trait so that the registered MBean is compliant with the standard MBean convention.  Default: "" |
| listener.security.protocol.map | Map between listener names and security protocols. This must be defined for the same security protocol to be usable in more than one port or IP. For example, internal and external traffic can be separated even if SSL is required for both. Concretely, the user could define listeners with names INTERNAL and EXTERNAL and this property as: `INTERNAL:SSL,EXTERNAL:SSL`. As shown, key and value are separated by a colon and map entries are separated by commas. Each listener name should only appear once in the map. Different security (SSL and SASL) settings can be configured for each listener by adding a normalised prefix (the listener name is lowercased) to the config name. For example, to set a different keystore for the INTERNAL listener, a config with name listener.name.internal.ssl.keystore.location would be set. If the config for the listener name is not set, the config will fallback to the generic config (i.e. ssl.keystore.location).  Default: PLAINTEXT:PLAINTEXT,SSL:SSL,SASL\_PLAINTEXT:SASL\_PLAINTEXT,SASL\_SSL:SASL\_SSL |
| log.message.downconversion.enable | This configuration controls whether down-conversion of message formats is enabled to satisfy consume requests. When set to false, broker will not perform down-conversion for consumers expecting an older message format. The broker responds with UNSUPPORTED\_VERSION error for consume requests from such older clients. This configurationdoes not apply to any message format conversion that might be required for replication to followers.  Default: true |
| metric.reporters | A list of classes to use as metrics reporters. Implementing the org.apache.kafka.common.metrics.MetricsReporter interface allows plugging in classes that will be notified of new metric creation. The JmxReporter is always included to register JMX statistics.  Default: "" |
| metrics.num.samples | The number of samples maintained to compute metrics.  Default: 2  Valid Values: [1,...] |
| metrics.recording.level | The highest recording level for metrics.  Default: INFO |
| metrics.sample.window.ms | The window of time a metrics sample is computed over.  Default: 30000 (30 seconds)  Valid Values: [1,...] |
| password.encoder.cipher.algorithm | The Cipher algorithm used for encoding dynamically configured passwords.  Default: AES/CBC/PKCS5Padding |
| password.encoder.iterations | The iteration count used for encoding dynamically configured passwords.  Default: 4096  Valid Values: [1024,...] |
| password.encoder.key.length | The key length used for encoding dynamically configured passwords.  Default: 128  Valid Values: [8,...] |
| password.encoder.keyfactory.algorithm | The SecretKeyFactory algorithm used for encoding dynamically configured passwords. Default is PBKDF2WithHmacSHA512 if available and PBKDF2WithHmacSHA1 otherwise.  Default: null |
| quota.window.num | The number of samples to retain in memory for client quotas  Default: 11  Valid Values: [1,...] |
| quota.window.size.seconds | The time span of each sample for client quotas  Default: 1  Valid Values: [1,...] |
| replication.quota.window.num | The number of samples to retain in memory for replication quotas  Default: 11  Valid Values: [1,...] |
| replication.quota.window.size.seconds | The time span of each sample for replication quotas  Default: 1  Valid Values: [1,...] |
| security.providers | A list of configurable creator classes each returning a provider implementing security algorithms. These classes should implement the org.apache.kafka.common.security.auth.SecurityProviderCreator interface.  Default: null |
| ssl.endpoint.identification.algorithm | The endpoint identification algorithm to validate server hostname using server certificate.  Default: http |
| ssl.engine.factory.class | The class of type org.apache.kafka.common.security.auth.SslEngineFactory to provide SSLEngine objects. Default value is org.apache.kafka.common.security.ssl.DefaultSslEngineFactory  Default: nul |
| ssl.principal.mapping.rules | A list of rules for mapping from distinguished name from the client certificate to short name. The rules are evaluated in order and the first rule that matches a principal name is used to map it to a short name. Any later rules in the list are ignored. By default, distinguished name of the X.500 certificate will be the principal. For more details on the format please see security authorization and acls. Note that this configuration is ignored if an extension of KafkaPrincipalBuilder is provided by the principal.builder.class configuration.  Default: DEFAULT |
| ssl.secure.random.implementation | The SecureRandom PRNG implementation to use for SSL cryptography operations.  Default: null |
| transaction.abort.timed.out.transaction.cleanup.interval.ms | The interval at which to rollback transactions that have timed out  Default: 10000 (10 seconds)  Valid Values: [1,...] |
| transaction.remove.expired.transaction.cleanup.interval.ms | The interval at which to remove transactions that have expired due to transactional.id.expiration.ms passing  Default: 3600000 (1 hour)  Valid Values: [1,...] |
| zookeeper.ssl.cipher.suites | Specifies the enabled cipher suites to be used in ZooKeeper TLS negotiation (csv). Overrides any explicit value set via the zookeeper.ssl.ciphersuites system property (note the single word "ciphersuites"). The default value of null means the list of enabled cipher suites is determined by the Java runtime being used.  Default: null |
| zookeeper.ssl.crl.enable | Specifies whether to enable Certificate Revocation List in the ZooKeeper TLS protocols. Overrides any explicit value set via the zookeeper.ssl.crl system property (note the shorter name).  Default: false |
| zookeeper.ssl.enabled.protocols | Specifies the enabled protocol(s) in ZooKeeper TLS negotiation (csv). Overrides any explicit value set via the zookeeper.ssl.enabledProtocols system property (note the camelCase). The default value of null means the enabled protocol will be the value of the zookeeper.ssl.protocol configuration property.  Default: null |
| zookeeper.ssl.endpoint.identification.algorithm | Specifies whether to enable hostname verification in the ZooKeeper TLS negotiation process, with (case-insensitively) "https" meaning ZooKeeper hostname verification is enabled and an explicit blank value meaning it is disabled (disabling it is only recommended for testing purposes). An explicit value overrides any "true" or "false" value set via the zookeeper.ssl.hostnameVerification system property (note the different name and values; true implies https and false implies blank).  Default: HTTP |
| zookeeper.ssl.ocsp.enable | Specifies whether to enable Online Certificate Status Protocol in the ZooKeeper TLS protocols. Overrides any explicit value set via the zookeeper.ssl.ocsp system property (note the shorter name).  Default: false |
| zookeeper.ssl.protocol | Specifies the protocol to be used in ZooKeeper TLS negotiation. An explicit value overrides any value set via the same-named zookeeper.ssl.protocol system property.  Default: TLSv1.2 |
| zookeeper.sync.time.ms | How far a ZK follower can be behind a ZK leader  Default: 2000 (2 seconds) |

More details about broker configuration can be found in the scala class kafka.server.KafkaConfig.

# 监控

# 其他配置

## Connect

### 概述

### 配置

|  |  |
| --- | --- |
| 配置项 | 说明 |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

## Streams

### 概述

### 配置

## AdminClient

### 概述

### 配置

# 应用