Arenadata™ Streaming

Версия - v1.3-ENG

Arenadata Streaming Platform Settings

2021 Arenadata™

Оглавление

1 Broker Configs 3

2 Topic-Level Configs 33

3 Producer Configs 39

4 Consumer Configs 48

5 Connect Configs 57

6 Streams Configs 58

The documentation lists the Arenadata Streaming platform settings at the broker and topic levels, as well as the configuration of Producer, Consumer, Connect, Streams.

The instruction can be useful to administrators, programmers, developers and employees of information technology departments who implement and maintain the system.

Important: Contact information support service – e-mail: info@arenadata.io

The ADS platform uses key-value pairs in the property file format for configuration. These values can be supplied either from a file or programmatically.

Глава 1

Broker Configs

All broker settings are stored in the configuration file */etc/kafka/conf/server.properties*.

The essential configurations are the following:

- *broker.id*
- *log.dirs*
- *zookeeper.connect*

The following is a list of settings with a description and an indication of their type, default values and actual values, their importance and update mode.

zookeeper.connect – Specifies the ZooKeeper connection string in the form *hostname:port* where host and port are the host and port of a ZooKeeper server. To allow connecting through other ZooKeeper nodes when that ZooKeeper machine is down you can also specify multiple hosts in the form *hostname1:port1,hostname2:port2,hostname3:port3*. The server can also have a ZooKeeper chroot path as part of its ZooKeeper connection string which puts its data under some path in the global ZooKeeper namespace. For example to give a chroot path of */chroot/path* you would give the connection string as *hostname1:port1,hostname2:port2,hostname3:port3/chroot/path*.

- TYPE – string
- DEFAULT – high
- DYNAMIC UPDATE MODE – read-only

advertised.host.name – only used when *advertised.listeners* or *listeners* are not set. Use *advertised.listeners* instead. Hostname to publish to ZooKeeper for clients to use. In IaaS environments, this may need to be different from the interface to which the broker binds. If this is not set, it will use the value for *host.name* if configured. Otherwise it will use the value returned from *java.net.InetAddress.getCanonicalHostName()*

- TYPE – string
- DEFAULT – null
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

advertised.listeners – Listeners to publish to ZooKeeper for clients to use, if different than the *listeners* config property. In IaaS environments, this may need to be different from the interface to which the broker binds. If this is not set, the value for *listeners* will be used. Unlike *listeners* it is not valid to advertise the *0.0.0.0* meta-address

- TYPE – string
- DEFAULT – null

- IMPORTANCE – high
- DYNAMIC UPDATE MODE – per-broker

advertised.port – only used when *advertised.listeners* or *listeners* are not set. Use *advertised.listeners* instead. The port to publish to ZooKeeper for clients to use. In IaaS environments, this may need to be different from the port to which the broker binds. If this is not set, it will publish the same port that the broker binds to

- TYPE – int
- DEFAULT – null
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

auto.create.topics.enable – Enable auto creation of topic on the server

- TYPE – boolean
- DEFAULT – true
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

auto.leader.rebalance.enable – Enables auto leader balancing. A background thread checks and triggers leader balance if required at regular intervals

- TYPE – boolean
- DEFAULT – true
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

background.threads – The number of threads to use for various background processing tasks

- TYPE – int
- DEFAULT – 10
- VALID VALUES – [1,...]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – cluster-wide

broker.id – The broker id for this server. If unset, a unique broker id will be generated.To avoid conflicts between zookeeper generated broker id's and user configured broker id's, generated broker ids start from *reserved.broker.max.id + 1*

- TYPE – int
- DEFAULT – - 1
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

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

- TYPE – string
- DEFAULT – producer

- IMPORTANCE – high
- DYNAMIC UPDATE MODE – cluster-wide

delete.topic.enable – Enables delete topic. Delete topic through the admin tool will have no effect if this config is turned off

- TYPE – boolean
- DEFAULT – true
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

host.name – 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

- TYPE – string
- DEFAULT – “”
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

leader.imbalance.check.interval.seconds – The frequency with which the partition rebalance check is triggered by the controller

- TYPE – long
- DEFAULT – 300
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

leader.imbalance.per.broker.percentage – The ratio of leader imbalance allowed per broker. The controller would trigger a leader balance if it goes above this value per broker. The value is specified in percentage

- TYPE – int
- DEFAULT – 10
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

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. 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*

- TYPE – string
- DEFAULT – null
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – per-broker

log.dir – The directory in which the log data is kept (supplemental for *log.dirs* property)

- TYPE – string
- DEFAULT – /tmp/kafka-logs
- IMPORTANCE – high

- DYNAMIC UPDATE MODE – read-only

log.dirs – The directories in which the log data is kept. If not set, the value in *log.dir* is used

- TYPE – string
- DEFAULT – null
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – high

log.flush.interval.messages – The number of messages accumulated on a log partition before messages are flushed to disk

- TYPE – long
- DEFAULT – 9223372036854775807
- VALID VALUES – [1,...]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – cluster-wide

log.flush.interval.ms – The maximum time in ms that a message in any topic is kept in memory before flushed to disk. If not set, the value in log.flush.scheduler.interval.ms is used

- TYPE – long
- DEFAULT – null
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – cluster-wide

log.flush.offset.checkpoint.interval.ms – The frequency with which we update the persistent record of the last flush which acts as the log recovery point

- TYPE – int
- DEFAULT – 60000
- VALID VALUES – [0,...]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

log.flush.scheduler.interval.ms – The frequency in ms that the log flusher checks whether any log needs to be flushed to disk

- TYPE – long
- DEFAULT – 9223372036854775807
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

log.flush.start.offset.checkpoint.interval.ms – The frequency with which we update the persistent record of log start offset

- TYPE – int
- DEFAULT – 60000
- VALID VALUES – [0,...]
- IMPORTANCE – high

- DYNAMIC UPDATE MODE – read-only

log.retention.bytes – The maximum size of the log before deleting it

- TYPE – long
- DEFAULT – - 1
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – cluster-wide

log.retention.hours – The number of hours to keep a log file before deleting it (in hours), tertiary to *log.retention.ms* property

- TYPE – int
- DEFAULT – 168
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

log.retention.minutes – The number of minutes to keep a log file before deleting it (in minutes), secondary to *log.retention.ms* property. If not set, the value in *log.retention.hours* is used

- TYPE – int
- DEFAULT – null
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

log.retention.ms – The number of milliseconds to keep a log file before deleting it (in milliseconds), If not set, the value in *log.retention.minutes* is used

- TYPE – long
- DEFAULT – null
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – cluster-wide

log.roll.hours – The maximum time before a new log segment is rolled out (in hours), secondary to *log.roll.ms* property

- TYPE – int
- DEFAULT – 168
- VALID VALUES – [1,...]
- IMPORTANCE – [1,...]
- DYNAMIC UPDATE MODE – read-only

log.roll.jitter.hours – The maximum jitter to subtract from *logRollTimeMillis* (in hours), secondary to *log.roll.jitter.ms* property

- TYPE – int
- DEFAULT – int
- VALID VALUES – [0,...]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

log.roll.jitter.ms – The maximum jitter to subtract from *logRollTimeMillis* (in milliseconds). If not set, the value in *log.roll.jitter.hours* is used

- TYPE – long
- DEFAULT – long
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – cluster-wide

log.roll.ms – The maximum time before a new log segment is rolled out (in milliseconds). If not set, the value in *log.roll.hours* is used

- TYPE – long
- DEFAULT – null
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – cluster-wide

log.segment.bytes – The maximum size of a single log file

- TYPE – int
- DEFAULT – 1073741824
- VALID VALUES – [14,...]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – cluster-wide

log.segment.delete.delay.ms – The amount of time to wait before deleting a file from the filesystem

- TYPE – long
- DEFAULT – 60000
- VALID VALUES – [0,...]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – cluster-wide

message.max.bytes – The largest record batch size allowed by ADS. 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

- TYPE – int
- DEFAULT – 1000012
- VALID VALUES – [0,...]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – cluster-wide

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

- TYPE – int

- DEFAULT – 1
- VALID VALUES – [1,...]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – cluster-wide

num.io.threads – The number of threads that the server uses for processing requests, which may include disk I/O

- TYPE – int
- DEFAULT – 8
- VALID VALUES – [1,...]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – cluster-wide

num.network.threads – The number of threads that the server uses for receiving requests from the network and sending responses to the network

- TYPE – int
- DEFAULT – 3
- VALID VALUES – [1,...]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – cluster-wide

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

- TYPE – int
- DEFAULT – 1
- VALID VALUES – [1,...]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – cluster-wide

num.replica.alter.log.dirs.threads – The number of threads that can move replicas between log directories, which may include disk I/O

- TYPE – int
- DEFAULT – null
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

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

- TYPE – int
- DEFAULT – 1
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – cluster-wide

offset.metadata.max.bytes – The maximum size for a metadata entry associated with an offset commit

- TYPE – int
- DEFAULT – 4096
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

offsets.commit.required.acks – The required acks before the commit can be accepted. In general, the default ("-1") should not be overridden

- TYPE – short
- DEFAULT – - 1
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

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

- TYPE – int
- DEFAULT – 5000
- VALID VALUES – [1,. . .]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

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)

- TYPE – int
- DEFAULT – 5242880
- VALID VALUES – [1,. . .]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

offsets.retention.check.interval.ms – Frequency at which to check for stale offsets

- TYPE – long
- DEFAULT – 600000
- VALID VALUES – [1,. . .]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

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

- TYPE – int
- DEFAULT – 1440
- VALID VALUES – [1,. . .]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

offsets.topic.compression.codec – Compression codec for the offsets topic - compression may be used to achieve “atomic” commits

- TYPE – int
- DEFAULT – 0
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

offsets.topic.num.partitions – The number of partitions for the offset commit topic (should not change after deployment)

- TYPE – int
- DEFAULT – 50
- VALID VALUES – [1,...]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

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

- TYPE – short
- DEFAULT – 3
- VALID VALUES – [1,...]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

offsets.topic.segment.bytes – The offsets topic segment bytes should be kept relatively small in order to facilitate faster log compaction and cache loads

- TYPE – int
- DEFAULT – 104857600
- VALID VALUES – [1,...]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

port – only used when *listeners* is not set. Use *listeners* instead. the port to listen and accept connections on

- TYPE – int
- DEFAULT – 9092
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

queued.max.requests – The number of queued requests allowed before blocking the network threads

- TYPE – int
- DEFAULT – 500
- VALID VALUES – [1,...]
- IMPORTANCE – high

- DYNAMIC UPDATE MODE – read-only

quota.consumer.default – 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

- TYPE – long
- DEFAULT – 9223372036854775807
- VALID VALUES – [1,...]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

quota.producer.default – 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

- TYPE – long
- DEFAULT – 9223372036854775807
- VALID VALUES – [1,...]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

replica.fetch.min.bytes – Minimum bytes expected for each fetch response. If not enough bytes, wait up to *replicaMaxWaitTimeMs*

- TYPE – int
- DEFAULT – 1
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

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

- TYPE – int
- DEFAULT – 500
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

replica.high.watermark.checkpoint.interval.ms – The frequency with which the high watermark is saved out to disk

- TYPE – long
- DEFAULT – 5000
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

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

- TYPE – long
- DEFAULT – 10000

- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

replica.socket.receive.buffer.bytes – The socket receive buffer for network requests

- TYPE – int
- DEFAULT – 65536
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

replica.socket.timeout.ms – The socket timeout for network requests. Its value should be at least *replica.fetch.wait.max.ms*

- TYPE – int
- DEFAULT – 30000
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

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

- TYPE – int
- DEFAULT – 30000
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

socket.receive.buffer.bytes – The SO_RCVBUF buffer of the socket sever sockets. If the value is "-1", the OS default will be used

- TYPE – int
- DEFAULT – 102400
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

socket.request.max.bytes – The maximum number of bytes in a socket request

- TYPE – int
- DEFAULT – 104857600
- VALID VALUES – [1,...]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

socket.send.buffer.bytes – The SO_SNDBUF buffer of the socket sever sockets. If the value is "-1", the OS default will be used

- TYPE – int
- DEFAULT – 102400
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

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

- TYPE – int
- DEFAULT – 900000
- VALID VALUES – [1,...]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

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)

- TYPE – int
- DEFAULT – 5242880
- VALID VALUES – [1,...]
- IMPORTANCE – [1,...]
- DYNAMIC UPDATE MODE – read-only

transaction.state.log.min.isr – Overridden *min.insync.replicas* config for the transaction topic

- TYPE – int
- DEFAULT – 2
- VALID VALUES – [1,...]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

transaction.state.log.num.partitions – The number of partitions for the transaction topic (should not change after deployment)

- TYPE – int
- DEFAULT – 50
- VALID VALUES – [1,...]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

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

- TYPE – short
- DEFAULT – 3
- VALID VALUES – [1,...]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

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

- TYPE – int

- DEFAULT – 104857600
- VALID VALUES – [1,...]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

transactional.id.expiration.ms – The maximum amount of time in ms that the transaction coordinator will wait before proactively expire a producer's transactional id without receiving any transaction status updates from it

- TYPE – int
- DEFAULT – 604800000
- VALID VALUES – [1,...]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

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

- TYPE – boolean
- DEFAULT – false
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – cluster-wide

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

- TYPE – int
- DEFAULT – null
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

zookeeper.max.in.flight.requests – The maximum number of unacknowledged requests the client will send to Zookeeper before blocking

- TYPE – int
- DEFAULT – 10
- VALID VALUES – [1,...]
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

zookeeper.session.timeout.ms – Zookeeper session timeout

- TYPE – int
- DEFAULT – int
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

zookeeper.set.acl – Set client to use secure ACLs

- TYPE – boolean

- DEFAULT – boolean
- IMPORTANCE – high
- DYNAMIC UPDATE MODE – read-only

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

- TYPE – boolean
- DEFAULT – true
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

broker.rack – Rack of the broker. This will be used in rack aware replication assignment for fault tolerance. Examples: "RACK1", "us-east-1d"

- TYPE – string
- DEFAULT – string
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

connections.max.idle.ms – Idle connections timeout: the server socket processor threads close the connections that idle more than this

- TYPE – long
- DEFAULT – 600000
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

controlled.shutdown.enable – Enable controlled shutdown of the server

- TYPE – boolean
- DEFAULT – true
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

controlled.shutdown.max.retries – Controlled shutdown can fail for multiple reasons. This determines the number of retries when such failure happens

- TYPE – int
- DEFAULT – 3
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

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

- TYPE – long
- DEFAULT – 5000
- IMPORTANCE – medium

- DYNAMIC UPDATE MODE – read-only

controller.socket.timeout.ms – The socket timeout for controller-to-broker channels

- TYPE – int
- DEFAULT – 30000
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

default.replication.factor – default replication factors for automatically created topics

- TYPE – int
- DEFAULT – 1
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

delegation.token.expiry.time.ms – The token validity time in miliseconds before the token needs to be renewed. Default value *1 day*

- TYPE – long
- DEFAULT – 86400000
- VALID VALUES – [1,...]
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

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

- TYPE – password
- DEFAULT – null
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

delegation.token.max.lifetime.ms – The token has a maximum lifetime beyond which it cannot be renewed anymore. Default value *7 days*

- TYPE – long
- DEFAULT – 604800000
- VALID VALUES – [1,...]
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

delete.records.purgatory.purge.interval.requests – The purge interval (in number of requests) of the delete records request purgatory

- TYPE – int
- DEFAULT – 1
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

fetch.purgatory.purge.interval.requests – The purge interval (in number of requests) of the fetch request purgatory

- TYPE – int
- DEFAULT – 1000
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

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

- TYPE – int
- DEFAULT – 3000
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

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

- TYPE – int
- DEFAULT – 300000
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

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

- TYPE – int
- DEFAULT – 6000
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

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

- TYPE – string
- DEFAULT – null
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

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

- TYPE – string
- DEFAULT – 1.1-IV0
- IMPORTANCE – medium

- DYNAMIC UPDATE MODE – read-only

log.cleaner.backoff.ms – The amount of time to sleep when there are no logs to clean

- TYPE – long
- DEFAULT – 15000
- VALID VALUES – [0,...]
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – cluster-wide

log.cleaner.dedupe.buffer.size – The total memory used for log deduplication across all cleaner threads

- TYPE – long
- DEFAULT – 134217728
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – cluster-wide

log.cleaner.delete.retention.ms – How long are delete records retained?

- TYPE – long
- DEFAULT – 86400000
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – cluster-wide

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

- TYPE – boolean
- DEFAULT – true
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

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

- TYPE – double
- DEFAULT – 0.9
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – cluster-wide

log.cleaner.io.buffer.size – The total memory used for log cleaner I/O buffers across all cleaner threads

- TYPE – int
- DEFAULT – 524288
- VALID VALUES – [0,...]
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – cluster-wide

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

- TYPE – double
- DEFAULT – 1.7976931348623157E308
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – cluster-wide

log.cleaner.min.cleanable.ratio – The minimum ratio of dirty log to total log for a log to eligible for cleaning

- TYPE – double
- DEFAULT – 0.5
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – cluster-wide

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

- TYPE – long
- DEFAULT – 0
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – cluster-wide

log.cleaner.threads – The number of background threads to use for log cleaning

- TYPE – int
- DEFAULT – 1
- VALID VALUES – [0,...]
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – cluster-wide

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"

- TYPE – list
- DEFAULT – delete
- VALID VALUES – [compact, delete]
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – cluster-wide

log.index.interval.bytes – The interval with which we add an entry to the offset index

- TYPE – int
- DEFAULT – 4096
- VALID VALUES – [0,...]
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – cluster-wide

log.index.size.max.bytes – The maximum size in bytes of the offset index

- TYPE – int
- DEFAULT – 10485760
- VALID VALUES – [4,...]
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – cluster-wide

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

- TYPE – string
- DEFAULT – 1.1-IV0
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

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

- TYPE – long
- DEFAULT – 9223372036854775807
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – cluster-wide

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"

- TYPE – string
- DEFAULT – CreateTime
- VALID VALUES – [CreateTime, LogAppendTime]
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – cluster-wide

log.preallocate – Should pre allocate file when create new segment? If you are using ADS on Windows, you probably need to set it to "true"

- TYPE – boolean
- DEFAULT – false
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – cluster-wide

log.retention.check.interval.ms – The frequency in milliseconds that the log cleaner checks whether any log is eligible for deletion

- TYPE – long
- DEFAULT – 300000

- VALID VALUES – [1,...]
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

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*

- TYPE – int
- DEFAULT – 2147483647
- VALID VALUES – [1,...]
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

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"

- TYPE – string
- DEFAULT – ""
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

max.incremental.fetch.session.cache.slots – The maximum number of incremental fetch sessions that we will maintain

- TYPE – int
- DEFAULT – 1000
- VALID VALUES – [0,...]
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

num.partitions – The default number of log partitions per topic

- TYPE – int
- DEFAULT – 1
- VALID VALUES – [1,...]
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

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

- TYPE – password
- DEFAULT – null
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

password.encoder.secret – The secret used for encoding dynamically configured passwords for this broker

- TYPE – password

- DEFAULT – null
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

principal.builder.class – The fully qualified name of a class that implements the ADSPrincipalBuilder interface, which is used to build the ADSPrincipal 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 name will be 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"

- TYPE – class
- DEFAULT – null
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

producer.purgatory.purge.interval.requests – The purge interval (in number of requests) of the producer request purgatory

- TYPE – int
- DEFAULT – 1000
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

queued.max.request.bytes – The number of queued bytes allowed before no more requests are read

- TYPE – long
- DEFAULT – - 1
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

replica.fetch.backoff.ms – The amount of time to sleep when fetch partition error occurs

- TYPE – int
- DEFAULT – 1000
- VALID VALUES – [0,...]
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

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)

- TYPE – int
- DEFAULT – 1048576
- VALID VALUES – [0,...]
- IMPORTANCE – medium

- DYNAMIC UPDATE MODE – read-only

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)

- TYPE – int
- DEFAULT – 10485760
- VALID VALUES – [0,...]
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

reserved.broker.max.id – Max number that can be used for a *broker.id*

- TYPE – int
- DEFAULT – 1000
- VALID VALUES – [0,...]
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

sasl.client.callback.handler.class – The fully qualified name of a SASL client callback handler class that implements the AuthenticateCallbackHandler interface

- TYPE – class
- DEFAULT – null
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

sasl.enabled.mechanisms – The list of SASL mechanisms enabled in the ADS server. The list may contain any mechanism for which a security provider is available. Only "GSSAPI" is enabled by default

- TYPE – list
- DEFAULT – GSSAPI
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

sasl.jaas.config – AAS 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;*

- TYPE – password
- DEFAULT – null
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

sasl.kerberos.kinit.cmd – Kerberos kinit command path

- TYPE – string
- DEFAULT – /usr/bin/kinit
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

sasl.kerberos.min.time.before.relogin – Login thread sleep time between refresh attempts

- TYPE – long
- DEFAULT – 60000
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

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}*. Note that this configuration is ignored if an extension of ADSPrincipalBuilder is provided by the *principal.builder.class* configuration

- TYPE – list
- DEFAULT – DEFAULT
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

sasl.kerberos.service.name – The Kerberos principal name that ADS runs as. This can be defined either in ADS's JAAS config or in ADS's config

- TYPE – string
- DEFAULT – null
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

sasl.kerberos.ticket.renew.jitter – Percentage of random jitter added to the renewal time

- TYPE – double
- DEFAULT – 0.05
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

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

- TYPE – double
- DEFAULT – 0.8
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

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*

- TYPE – class
- DEFAULT – null
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

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*

- TYPE – class
- DEFAULT – null
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

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"

- TYPE – short
- DEFAULT – 300
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

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"

- TYPE – short
- DEFAULT – 60
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

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"

- TYPE – double
- DEFAULT – 0.8
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

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"

- TYPE – double

- DEFAULT – 0.05
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

sasl.mechanism.inter.broker.protocol – SASL mechanism used for inter-broker communication. Default is "GSSAPI"

- TYPE – string
- DEFAULT – GSSAPI
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

sasl.server.callback.handler.class – SASL mechanism used for inter-broker communication. Default is "GSSAPI"

- TYPE – string
- DEFAULT – GSSAPI
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

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

- TYPE – string
- DEFAULT – PLAINTEXT
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – read-only

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

- TYPE – list
- DEFAULT – null
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

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=request* – This means client authentication is optional. unlike requested , 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
- TYPE – string
- DEFAULT – none
- VALID VALUES – [required, requested, none]
- IMPORTANCE – medium

- DYNAMIC UPDATE MODE – per-broker

ssl.enabled.protocols – The list of protocols enabled for SSL connections

- TYPE – list
- DEFAULT – TLSv1.2,TLSv1.1,TLSv1
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

ssl.key.password – The password of the private key in the key store file. This is optional for client

- TYPE – password
- DEFAULT – null
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

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

- TYPE – string
- DEFAULT – SunX509
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

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

- TYPE – string
- DEFAULT – null
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

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*

- TYPE – password
- DEFAULT – null
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

ssl.keystore.type – The file format of the key store file. This is optional for client

- TYPE – string
- DEFAULT – JKS
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

ssl.protocol – The SSL protocol used to generate the SSLContext. Default setting is "TLS", which is fine for most cases. Allowed values in recent JVMs are "TLS", "TLSv1.1" and "TLSv1.2". "SSL", "SSLv2" and "SSLv3" may be supported in older JVMs, but their usage is discouraged due to known security vulnerabilities

- TYPE – string

- DEFAULT – TLS
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

ssl.provider – The name of the security provider used for SSL connections. Default value is the default security provider of the JVM

- TYPE – string
- DEFAULT – null
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

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

- TYPE – string
- DEFAULT – PKIX
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

ssl.truststore.location – The location of the trust store file

- TYPE – string
- DEFAULT – null
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

ssl.truststore.password – The password for the trust store file. If a password is not set access to the truststore is still available, but integrity checking is disabled

- TYPE – password
- DEFAULT – null
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

ssl.truststore.type – The file format of the trust store file

- TYPE – string
- DEFAULT – JKS
- IMPORTANCE – medium
- DYNAMIC UPDATE MODE – per-broker

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

- TYPE – class
- DEFAULT – null
- IMPORTANCE – low
- DYNAMIC UPDATE MODE – read-only

authorizer.class.name – The authorizer class that should be used for authorization

- TYPE – string
- DEFAULT – “”
- IMPORTANCE – low
- DYNAMIC UPDATE MODE – read-only

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

- TYPE – class
- DEFAULT – null
- IMPORTANCE – low
- DYNAMIC UPDATE MODE – read-only

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*)

- TYPE – string
- DEFAULT – PLAINTEXT:PLAINTEXT,SSL:SSL,SASL_PLAINTEXT:SASL_PLAINTEXT,SASL_SSL:SASL_SSL
- IMPORTANCE – low
- DYNAMIC UPDATE MODE – per-broker

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

- TYPE – list
- DEFAULT – “”
- IMPORTANCE – low
- DYNAMIC UPDATE MODE – cluster-wide

metrics.num.samples – The number of samples maintained to compute metrics

- TYPE – int
- DEFAULT – 2
- VALID VALUES – [1,...]
- IMPORTANCE – low
- DYNAMIC UPDATE MODE – read-only

metrics.recording.level – The highest recording level for metrics

- TYPE – string
- DEFAULT – INFO

- IMPORTANCE – low
- DYNAMIC UPDATE MODE – read-only

metrics.sample.window.ms – The window of time a metrics sample is computed over

- TYPE – long
- DEFAULT – 30000
- VALID VALUES – [1,...]
- IMPORTANCE – low
- DYNAMIC UPDATE MODE – read-only

quota.window.num – The number of samples to retain in memory for client quotas

- TYPE – int
- DEFAULT – 11
- VALID VALUES – [1,...]
- IMPORTANCE – low
- DYNAMIC UPDATE MODE – read-only

quota.window.size.seconds – The time span of each sample for client quotas

- TYPE – int
- DEFAULT – 1
- VALID VALUES – [1,...]
- IMPORTANCE – low
- DYNAMIC UPDATE MODE – read-only

replication.quota.window.num – The number of samples to retain in memory for replication quotas

- TYPE – int
- DEFAULT – 11
- VALID VALUES – [1,...]
- IMPORTANCE – low
- DYNAMIC UPDATE MODE – read-only

replication.quota.window.size.seconds – The time span of each sample for replication quotas

- TYPE – int
- DEFAULT – 1
- VALID VALUES – [1,...]
- IMPORTANCE – low
- DYNAMIC UPDATE MODE – read-only

ssl.endpoint.identification.algorithm – The endpoint identification algorithm to validate server hostname using server certificate

- TYPE – string
- DEFAULT – null

- IMPORTANCE – low
- DYNAMIC UPDATE MODE – per-broker

ssl.secure.random.implementation – The SecureRandom PRNG implementation to use for SSL cryptography operations

- TYPE – string
- DEFAULT – null
- IMPORTANCE – low
- DYNAMIC UPDATE MODE – per-broker

transaction.abort.timed.out.transaction.cleanup.interval.ms – The interval at which to rollback transactions that have timed out

- TYPE – int
- DEFAULT – 60000
- VALID VALUES – [1,...]
- IMPORTANCE – low
- DYNAMIC UPDATE MODE – read-only

transaction.remove.expired.transaction.cleanup.interval.ms – The interval at which to remove transactions that have expired due to *transactional.id.expiration.ms passing*

- TYPE – int
- DEFAULT – 3600000
- VALID VALUES – [1,...]
- IMPORTANCE – low
- DYNAMIC UPDATE MODE – read-only

zookeeper.sync.time.ms – How far a ZK follower can be behind a ZK leader

- TYPE – int
- DEFAULT – 2000
- IMPORTANCE – low
- DYNAMIC UPDATE MODE – read-only

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

Глава 2

Topic-Level Configs

Configurations pertinent to topics have both a server default as well an optional per-topic override. If no per-topic configuration is given the server default is used. The override can be set at topic creation time by giving one or more *–config* options.

The following are the topic-level configurations. The server's default configuration for this property is given under the Server Default Property heading. A given server default config value only applies to a topic if it does not have an explicit topic config override.

cleanup.policy – A string that is either "delete" or "compact". 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

- TYPE – list
- DEFAULT – delete
- VALID VALUES – [compact, delete]
- SERVER DEFAULT PROPERTY –log.cleanup.policy
- IMPORTANCE – medium

compression.type – Specify the final compression type for a given topic. This configuration accepts the standard compression codecs ("gzip", "snappy", "lz4"). It additionally accepts "uncompressed" which is equivalent to no compression; and "producer" which means retain the original compression codec set by the producer

- TYPE – string
- DEFAULT – producer
- VALID VALUES – [uncompressed, snappy, lz4, gzip, producer]
- SERVER DEFAULT PROPERTY – compression.type
- IMPORTANCE – medium

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)

- TYPE – long
- DEFAULT – 86400000
- VALID VALUES – [0,. . .]

- SERVER DEFAULT PROPERTY – log.cleaner.delete.retention.ms
- IMPORTANCE – medium

file.delete.delay.ms – The time to wait before deleting a file from the filesystem

- TYPE – long
- DEFAULT – 60000
- VALID VALUES – [0,...]
- SERVER DEFAULT PROPERTY – log.segment.delete.delay.ms
- IMPORTANCE – medium

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

- TYPE – long
- DEFAULT – 9223372036854775807
- VALID VALUES – [0,...]
- SERVER DEFAULT PROPERTY – log.flush.interval.messages
- IMPORTANCE – medium

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

- TYPE – long
- DEFAULT – 9223372036854775807
- VALID VALUES – [0,...]
- SERVER DEFAULT PROPERTY – log.flush.interval.ms
- IMPORTANCE – medium

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

- TYPE – list
- DEFAULT – ""
- VALID VALUES – [partitionId],[brokerId]:[partitionId],[brokerId]:...
- SERVER DEFAULT PROPERTY – follower.replication.throttled.replicas
- IMPORTANCE – medium

index.interval.bytes – This setting controls how frequently Kafka adds an index entry to it's 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

- TYPE – int
- DEFAULT – 4096

- VALID VALUES – [0,...]
- SERVER DEFAULT PROPERTY – log.index.interval.bytes
- IMPORTANCE – medium

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

- TYPE – list
- DEFAULT – ""
- VALID VALUES – [partitionId],[brokerId]:[partitionId],[brokerId]:...
- SERVER DEFAULT PROPERTY – leader.replication.throttled.replicas
- IMPORTANCE – medium

max.message.bytes – The largest record batch size allowed by **ADS**. If this is increased the consumers' fetch size must also be increased so that the they can fetch record batches this large

- TYPE – int
- DEFAULT – 1000012
- VALID VALUES – [0,...]
- SERVER DEFAULT PROPERTY – message.max.bytes
- IMPORTANCE – medium

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

- TYPE – string
- DEFAULT – 1.1-IV0
- SERVER DEFAULT PROPERTY – log.message.format.version
- IMPORTANCE – medium

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*

- TYPE – long
- DEFAULT – 9223372036854775807
- VALID VALUES – [0,...]
- SERVER DEFAULT PROPERTY – log.message.timestamp.difference.max.ms
- IMPORTANCE – medium

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"

- TYPE – string
- DEFAULT – CreateTime

- SERVER DEFAULT PROPERTY – log.message.timestamp.type
- IMPORTANCE – medium

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

- TYPE – double
- DEFAULT – 0.5
- VALID VALUES – [0,. . . ,1]
- SERVER DEFAULT PROPERTY – log.cleaner.min.cleanable.ratio
- IMPORTANCE – medium

min.compaction.lag.ms – The minimum time a message will remain uncompacted in the log. Only applicable for logs that are being compacted

- TYPE – long
- DEFAULT – 0
- VALID VALUES – [0,. . .]
- SERVER DEFAULT PROPERTY – log.cleaner.min.compaction.lag.ms
- IMPORTANCE – medium

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

- TYPE – int
- DEFAULT – 1
- VALID VALUES – [1,. . .]
- SERVER DEFAULT PROPERTY – min.insync.replicas
- IMPORTANCE – medium

preallocate – True if we should preallocate the file on disk when creating a new log segment

- TYPE – boolean
- DEFAULT – false
- SERVER DEFAULT PROPERTY – log.preallocate
- IMPORTANCE – medium

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

- TYPE – long

- DEFAULT – - 1
- SERVER DEFAULT PROPERTY – log.retention.bytes
- IMPORTANCE – medium

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

- TYPE – long
- DEFAULT – 604800000
- SERVER DEFAULT PROPERTY – log.retention.ms
- IMPORTANCE – medium

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

- TYPE – int
- DEFAULT – 1073741824
- VALID VALUES – [14,...]
- SERVER DEFAULT PROPERTY – log.segment.bytes
- IMPORTANCE – medium

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

- TYPE – int
- DEFAULT – 10485760
- VALID VALUES – [0,...]
- SERVER DEFAULT PROPERTY – log.index.size.max.bytes
- IMPORTANCE – medium

segment.jitter.ms – The maximum random jitter subtracted from the scheduled segment roll time to avoid thundering herds of segment rolling

- TYPE – long
- DEFAULT – 0
- VALID VALUES – [0,...]
- SERVER DEFAULT PROPERTY – log.roll.jitter.ms
- IMPORTANCE – medium

segment.ms – This configuration controls the period of time after which **ADS** will force the log to roll even if the segment file isn't full to ensure that retention can delete or compact old data

- TYPE – long
- DEFAULT – 604800000
- VALID VALUES – [0,...]
- SERVER DEFAULT PROPERTY – log.roll.ms
- IMPORTANCE – medium

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

- TYPE – boolean
- DEFAULT – false
- SERVER DEFAULT PROPERTY – unclean.leader.election.enable
- IMPORTANCE – medium

Глава 3

Producer Configs

Below is the configuration of the Java producer.

key.serializer – Serializer class for key that implements the *org.apache.kafka.common.serialization.Serializer* interface

- TYPE – class
- IMPORTANCE – high

value.serializer – Serializer class for value that implements the *org.apache.kafka.common.serialization.Serializer* interface

- TYPE – class
- IMPORTANCE – high

acks – The number of acknowledgments the producer requires the leader to have received before considering a request complete. This controls the durability of records that are sent. The following settings are allowed:

- *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"
- *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
- *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
- TYPE – string
- DEFAULT – 1
- VALID VALUES – [all, - 1, 0, 1]
- IMPORTANCE – high

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

- TYPE – long
- DEFAULT – 33554432
- VALID VALUES – [0,...]
- IMPORTANCE – high

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", or "lz4". Compression is of full batches of data, so the efficacy of batching will also impact the compression ratio (more batching means better compression)

- TYPE – string
- DEFAULT – none
- IMPORTANCE – high

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

- TYPE – int
- DEFAULT – 0
- VALID VALUES – [0,...,2147483647]
- IMPORTANCE – high

ssl.key.password – The password of the private key in the key store file. This is optional for client

- TYPE – password
- DEFAULT – null
- IMPORTANCE – high

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

- TYPE – string
- DEFAULT – null
- IMPORTANCE – high

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

- TYPE – password
- DEFAULT – null
- IMPORTANCE – high

ssl.truststore.location – The location of the trust store file

- TYPE – string
- DEFAULT – null
- IMPORTANCE – high

ssl.truststore.password – The password for the trust store file. If a password is not set access to the truststore is still available, but integrity checking is disabled

- TYPE – password
- DEFAULT – null
- IMPORTANCE – high

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

- TYPE – int
- DEFAULT – 16384
- VALID VALUES – [0,...]
- IMPORTANCE – medium

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

- TYPE – string
- DEFAULT – “”
- IMPORTANCE – medium

connections.max.idle.ms – Close idle connections after the number of milliseconds specified by this config

- TYPE – long
- DEFAULT – 540000
- IMPORTANCE – medium

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

- TYPE – long
- DEFAULT – 0
- VALID VALUES – [0,...]
- IMPORTANCE – medium

max.block.ms – The configuration controls how long *KafkaProducer.send()* and *KafkaProducer.partitionsFor()* will block. These methods can be blocked either because the buffer is full or metadata *unavailable.Blocking* in the user-supplied serializers or partitioner will not be counted against this timeout

- TYPE – long

- DEFAULT – 60000
- VALID VALUES – [0,...]
- IMPORTANCE – medium

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 record batch size. Note that the server has its own cap on record batch size which may be different from this

- TYPE – int
- DEFAULT – 1048576
- VALID VALUES – [0,...]
- IMPORTANCE – medium

partitioner.class – Partitioner class that implements the *org.apache.kafka.clients.producer.Partitioner* interface

- TYPE – class
- DEFAULT – org.apache.kafka.clients.producer.internals.DefaultPartitioner
- IMPORTANCE – medium

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

- TYPE – int
- DEFAULT – 32768
- VALID VALUES – [-1,...]
- IMPORTANCE – medium

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

- TYPE – int
- DEFAULT – 30000
- VALID VALUES – [0,...]
- IMPORTANCE – medium

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: "(=)*;"

- TYPE – password
- DEFAULT – null
- IMPORTANCE – medium

sasl.kerberos.service.name – The Kerberos principal name that ADS runs as. This can be defined either in ADS's JAAS config or in ADS's config

- TYPE – string
- DEFAULT – null
- IMPORTANCE – medium

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

- TYPE – string
- DEFAULT – GSSAPI
- IMPORTANCE – medium

security.protocol – Protocol used to communicate with brokers. Valid values are: “PLAINTEXT”, “SSL”, “SASL_PLAINTEXT”, “SASL_SSL”

- TYPE – string
- DEFAULT – PLAINTEXT
- IMPORTANCE – medium

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

- TYPE – int
- DEFAULT – 131072
- VALID VALUES – [-1,...]
- IMPORTANCE – medium

ssl.enabled.protocols – The list of protocols enabled for SSL connections

- TYPE – list
- DEFAULT – TLSv1.2,TLSv1.1,TLSv1
- IMPORTANCE – medium

ssl.keystore.type – The file format of the key store file. This is optional for client

- TYPE – string
- DEFAULT – JKS
- IMPORTANCE – medium

ssl.protocol – The SSL protocol used to generate the SSLContext. Default setting is “TLS”, which is fine for most cases. Allowed values in recent JVMs are “TLS”, “TLSv1.1” and “TLSv1.2”. “SSL”, “SSLv2” and “SSLv3” may be supported in older JVMs, but their usage is discouraged due to known security vulnerabilities

- TYPE – string
- DEFAULT – TLS
- IMPORTANCE – medium

ssl.provider –The name of the security provider used for SSL connections. Default value is the default security provider of the JVM

- TYPE – string
- DEFAULT – null
- IMPORTANCE – medium

ssl.truststore.type – The file format of the trust store file

- TYPE – string
- DEFAULT – JKS

- IMPORTANCE – medium

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

- TYPE – boolean
- DEFAULT – false
- IMPORTANCE – low

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 ADS cluster. By default, there are no interceptors

- TYPE – list
- DEFAULT – ""
- VALID VALUES – org.apache.kafka.common.config.ConfigDef$NonNullValidator@6a41eaa2
- IMPORTANCE – low

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)

- TYPE – int
- DEFAULT – 5
- VALID VALUES – [1,...]
- IMPORTANCE – low

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

- TYPE – long
- DEFAULT – 300000
- VALID VALUES – [0,...]
- IMPORTANCE – low

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

- TYPE – list
- DEFAULT – ""
- VALID VALUES – org.apache.kafka.common.config.ConfigDef$NonNullValidator@7cd62f43
- IMPORTANCE – low

metrics.num.samples – The number of samples maintained to compute metrics

- TYPE – int
- DEFAULT – 2

- VALID VALUES – [1,...]
- IMPORTANCE – low

metrics.recording.level – The highest recording level for metrics

- TYPE – string
- DEFAULT – INFO
- VALID VALUES – [INFO, DEBUG]
- IMPORTANCE – low

metrics.sample.window.ms – The window of time a metrics sample is computed over

- TYPE – long
- DEFAULT – 30000
- VALID VALUES – [0,...]
- IMPORTANCE – low

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

- TYPE – long
- DEFAULT – 1000
- VALID VALUES – [0,...]
- IMPORTANCE – low

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

- TYPE – long
- DEFAULT – 50
- VALID VALUES – [0,...]
- IMPORTANCE – low

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

- TYPE – long
- DEFAULT – 100
- VALID VALUES – [0,...]
- IMPORTANCE – low

sasl.kerberos.kinit.cmd – Kerberos kinit command path

- TYPE – string
- DEFAULT – /usr/bin/kinit
- IMPORTANCE – low

sasl.kerberos.min.time.before.relogin – Login thread sleep time between refresh attempts

- TYPE – long
- DEFAULT – 60000
- IMPORTANCE – low

sasl.kerberos.ticket.renew.jitter – Percentage of random jitter added to the renewal time

- TYPE – double
- DEFAULT – 0.05
- IMPORTANCE – low

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

- TYPE – double
- DEFAULT – 0.8
- IMPORTANCE – low

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

- TYPE – list
- DEFAULT – null
- IMPORTANCE – low

ssl.endpoint.identification.algorithm – The endpoint identification algorithm to validate server hostname using server certificate

- TYPE – string
- DEFAULT – null
- IMPORTANCE – low

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

- TYPE – string
- DEFAULT – SunX509
- IMPORTANCE – low

ssl.secure.random.implementation – The SecureRandom PRNG implementation to use for SSL cryptography operations

- TYPE – string
- DEFAULT – null
- IMPORTANCE – low

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

- TYPE – string
- DEFAULT – PKIX
- IMPORTANCE – low

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 *InvalidTransactionTimeout* error

- TYPE – int
- DEFAULT – 60000
- IMPORTANCE – low

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. Note that *enable.idempotence* must be enabled if a *TransactionalId* is configured. The default is "null", which means transactions cannot be used. Note that transactions requires a cluster of at least three brokers by default what is the recommended setting for production; for development you can change this, by adjusting broker setting *transaction.state.log.replication.factor*

- TYPE – string
- DEFAULT – null
- VALID VALUES – non-empty string
- IMPORTANCE – low

Глава 4

Consumer Configs

Below is the configuration for the new consumer.

key.deserializer – Deserializer class for key that implements the *org.apache.kafka.common.serialization.Deserializer* interface

- TYPE – class
- IMPORTANCE – high

value.deserializer – Deserializer class for value that implements the *org.apache.kafka.common.serialization.Deserializer* interface

- TYPE – class
- IMPORTANCE – high

fetch.min.bytes – The minimum amount of data the server should return for a fetch request. If insufficient data is available the request will wait for that much data to accumulate before answering the request. The default setting of "1" byte means that fetch requests are answered as soon as a single byte of data is available or the fetch request times out waiting for data to arrive. Setting this to something greater than "1" will cause the server to wait for larger amounts of data to accumulate which can improve server throughput a bit at the cost of some additional latency

- TYPE – int
- DEFAULT – 1
- VALID VALUES – [0,...]
- IMPORTANCE – high

group.id – A unique string that identifies the consumer group this consumer belongs to. This property is required if the consumer uses either the group management functionality by using *subscribe(topic)* or the ADS-based offset management strategy

- TYPE – string
- DEFAULT – ""
- IMPORTANCE – high

heartbeat.interval.ms – The expected time between heartbeats to the consumer coordinator when using ADS's group management facilities. Heartbeats are used to ensure that the consumer's session stays active and to facilitate rebalancing when new consumers join or leave the group. The value must be set lower than *session.timeout.ms*, but typically should be set no higher than 1/3 of that value. It can be adjusted even lower to control the expected time for normal rebalances

- TYPE – int
- DEFAULT – 3000
- IMPORTANCE – high

max.partition.fetch.bytes – The maximum amount of data per-partition the server will return. Records are fetched in batches by the consumer. If the first record batch in the first non-empty partition of the fetch is larger than this limit, the batch will still be returned to ensure that the consumer can make progress. The maximum record batch size accepted by the broker is defined via *message.max.bytes* (broker config) or *max.message.bytes* (topic config). See *fetch.max.bytes* for limiting the consumer request size

- TYPE – int
- DEFAULT – 1048576
- VALID VALUES – [0,...]
- IMPORTANCE – high

session.timeout.ms – The timeout used to detect consumer failures when using ADS's group management facility. The consumer sends periodic heartbeats to indicate its liveness to the broker. If no heartbeats are received by the broker before the expiration of this session timeout, then the broker will remove this consumer from the group and initiate a rebalance. Note that the value must be in the allowable range as configured in the broker configuration by *group.min.session.timeout.ms* and *group.max.session.timeout.ms*

- TYPE – int
- DEFAULT – 10000
- IMPORTANCE – high

ssl.key.password – The password of the private key in the key store file. This is optional for client

- TYPE – password
- DEFAULT – null
- IMPORTANCE – high

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

- TYPE – string
- DEFAULT – null
- IMPORTANCE – high

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

- TYPE – password
- DEFAULT – null
- IMPORTANCE – high

ssl.truststore.location – The location of the trust store file

- TYPE – string
- DEFAULT – null
- IMPORTANCE – high

ssl.truststore.password – The password for the trust store file. If a password is not set access to the truststore is still available, but integrity checking is disabled

- TYPE – password
- DEFAULT – null
- IMPORTANCE – null

auto.offset.reset – What to do when there is no initial offset in ADS or if the current offset does not exist any more on the server (e.g. because that data has been deleted):

- *earliest*: automatically reset the offset to the earliest offset;
- *latest*: automatically reset the offset to the latest offset;
- *none*: throw exception to the consumer if no previous offset is found for the consumer's group;
- *anything else*: throw exception to the consumer.
- TYPE – string
- DEFAULT – latest
- VALID VALUES – [latest, earliest, none]
- IMPORTANCE – medium

connections.max.idle.ms – Close idle connections after the number of milliseconds specified by this config

- TYPE – long
- DEFAULT – 540000
- IMPORTANCE – medium

enable.auto.commit – If "true" the consumer's offset will be periodically committed in the background

- TYPE – boolean
- DEFAULT – true
- IMPORTANCE – medium

exclude.internal.topics – Whether records from internal topics (such as offsets) should be exposed to the consumer. If set to "true" the only way to receive records from an internal topic is subscribing to it

- TYPE – boolean
- DEFAULT – true
- IMPORTANCE – medium

fetch.max.bytes – The maximum amount of data the server should return for a fetch request. Records are fetched in batches by the consumer, 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 the consumer can make progress. As such, this is not a 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). Note that the consumer performs multiple fetches in parallel

- TYPE – int
- DEFAULT – 52428800
- VALID VALUES – [0,...]
- IMPORTANCE – medium

isolation.level – Controls how to read messages written transactionally. If set to "read_committed", *consumer.poll()* will only return transactional messages which have been committed. If set to "read_uncommitted" (the default), *consumer.poll()* will return all messages, even transactional messages which have been aborted. Non-transactional messages will be returned unconditionally in either mode. Messages will always be returned in

offset order. Hence, in "read_committed" mode, *consumer.poll()* will only return messages up to the last stable offset (LSO), which is the one less than the offset of the first open transaction. In particular any messages appearing after messages belonging to ongoing transactions will be withheld until the relevant transaction has been completed. As a result, "read_committed" consumers will not be able to read up to the high watermark when there are in flight transactions. Further, when in "read_committed" the *seekToEnd* method will return the LSO

- TYPE – string
- DEFAULT – read_uncommitted
- VALID VALUES – [read_committed, read_uncommitted]
- IMPORTANCE – medium

max.poll.interval.ms – The maximum delay between invocations of *poll()* when using consumer group management. This places an upper bound on the amount of time that the consumer can be idle before fetching more records. If *poll()* is not called before expiration of this timeout, then the consumer is considered failed and the group will rebalance in order to reassign the partitions to another member

- TYPE – int
- DEFAULT – 300000
- VALID VALUES – [1,...]
- IMPORTANCE – medium

max.poll.records – The maximum number of records returned in a single call to *poll()*

- TYPE – int
- DEFAULT – 500
- VALID VALUES – [1,...]
- IMPORTANCE – medium

partition.assignment.strategy – The class name of the partition assignment strategy that the client will use to distribute partition ownership amongst consumer instances when group management is used

- TYPE – list
- DEFAULT – class org.apache.kafka.clients.consumer.RangeAssignor
- VALID VALUES – org.apache.kafka.common.config.ConfigDef$NonNullValidator@6d4b1c02
- IMPORTANCE – medium

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

- TYPE – int
- DEFAULT – 65536
- VALID VALUES – [-1,...]
- IMPORTANCE – medium

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

- TYPE – int
- DEFAULT – 305000
- VALID VALUES – [0,...]

- IMPORTANCE – medium

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: "(=)*;"

- TYPE – password
- DEFAULT – null
- IMPORTANCE – medium

sasl.kerberos.service.name – The Kerberos principal name that ADS runs as. This can be defined either in ADS's JAAS config or in ADS's config

- TYPE – string
- DEFAULT – null
- IMPORTANCE – medium

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

- TYPE – string
- DEFAULT – GSSAPI
- IMPORTANCE – medium

security.protocol – Protocol used to communicate with brokers. Valid values are: "PLAINTEXT", "SSL", "SASL_PLAINTEXT", "SASL_SSL"

- TYPE – string
- DEFAULT – PLAINTEXT
- IMPORTANCE – medium

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

- TYPE – int
- DEFAULT – 131072
- VALID VALUES – [-1,...]
- IMPORTANCE – medium

ssl.enabled.protocols – The list of protocols enabled for SSL connections

- TYPE – list
- DEFAULT – TLSv1.2,TLSv1.1,TLSv1
- IMPORTANCE – medium

ssl.keystore.type – The file format of the key store file. This is optional for client

- TYPE – string
- DEFAULT – JKS
- IMPORTANCE – medium

ssl.protocol – The SSL protocol used to generate the SSLContext. Default setting is "TLS", which is fine for most cases. Allowed values in recent JVMs are "TLS", "TLSv1.1" and "TLSv1.2". "SSL", "SSLv2" and "SSLv3" may be supported in older JVMs, but their usage is discouraged due to known security vulnerabilities

- TYPE – string
- DEFAULT – TLS
- IMPORTANCE – medium

ssl.provider – The name of the security provider used for SSL connections. Default value is the default security provider of the JVM

- TYPE – string
- DEFAULT – null
- IMPORTANCE – medium

ssl.truststore.type – The file format of the trust store file

- TYPE – string
- DEFAULT – JKS
- IMPORTANCE – medium

auto.commit.interval.ms – The frequency in milliseconds that the consumer offsets are auto-committed to ADS if *enable.auto.commit* is set to "true"

- TYPE – int
- DEFAULT – 5000
- VALID VALUES – [0,...]
- IMPORTANCE – low

check.crcs – Automatically check the CRC32 of the records consumed. This ensures no on-the-wire or on-disk corruption to the messages occurred. This check adds some overhead, so it may be disabled in cases seeking extreme performance

- TYPE – boolean
- DEFAULT – true
- IMPORTANCE – low

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

- TYPE – string
- DEFAULT – ""
- IMPORTANCE – low

fetch.max.wait.ms – The maximum amount of time the server will block before answering the fetch request if there isn't sufficient data to immediately satisfy the requirement given by *fetch.min.bytes*

- TYPE – int
- DEFAULT – 500
- VALID VALUES – [0,...]
- IMPORTANCE – low

interceptor.classes – A list of classes to use as interceptors. Implementing the *org.apache.kafka.clients.consumer.ConsumerInterceptor* interface allows you to intercept (and possibly mutate) records received by the consumer. By default, there are no interceptors

- TYPE – list
- DEFAULT – “”
- VALID VALUES – org.apache.kafka.common.config.ConfigDef$NonNullValidator@6093dd95
- IMPORTANCE – low

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

- TYPE – long
- DEFAULT – 300000
- VALID VALUES – [0,...]
- IMPORTANCE – low

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

- TYPE – list
- DEFAULT – “”
- VALID VALUES – org.apache.kafka.common.config.ConfigDef$NonNullValidator@5622fdf
- IMPORTANCE – low

metrics.num.samples – The number of samples maintained to compute metrics

- TYPE – int
- DEFAULT – 2
- VALID VALUES – [1,...]
- IMPORTANCE – low

metrics.recording.level – The highest recording level for metrics

- TYPE – string
- DEFAULT – INFO
- VALID VALUES – [INFO, DEBUG]
- IMPORTANCE – low

metrics.sample.window.ms – The window of time a metrics sample is computed over

- TYPE – long
- DEFAULT – 30000
- VALID VALUES – [0,...]
- IMPORTANCE – low

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

- TYPE – long
- DEFAULT – 1000

- VALID VALUES – [0,...]
- IMPORTANCE – low

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

- TYPE – long
- DEFAULT – 50
- VALID VALUES – [0,...]
- IMPORTANCE – low

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

- TYPE – long
- DEFAULT – 100
- VALID VALUES – [0,...]
- IMPORTANCE – low

sasl.kerberos.kinit.cmd – Kerberos kinit command path

- TYPE – string
- DEFAULT – /usr/bin/kinit
- IMPORTANCE – low

sasl.kerberos.min.time.before.relogin – Login thread sleep time between refresh attempts

- TYPE – long
- DEFAULT – 60000
- IMPORTANCE – low

sasl.kerberos.ticket.renew.jitter – Percentage of random jitter added to the renewal time

- TYPE – double
- DEFAULT – 0.05
- IMPORTANCE – low

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

- TYPE – double
- DEFAULT – 0.8
- IMPORTANCE – low

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

- TYPE – list
- DEFAULT – null
- IMPORTANCE – low

ssl.endpoint.identification.algorithm – The endpoint identification algorithm to validate server hostname using server certificate

- TYPE – string
- DEFAULT – null
- IMPORTANCE – low

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

- TYPE – string
- DEFAULT – SunX509
- IMPORTANCE – low

ssl.secure.random.implementation – The SecureRandom PRNG implementation to use for SSL cryptography operations

- TYPE – string
- DEFAULT – null
- IMPORTANCE – low

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

- TYPE – string
- DEFAULT – PKIX
- IMPORTANCE – low

Глава 5

Connect Configs

Глава 6

Streams Configs

Below is the configuration of the Kafka Streams client library.

application.id – An identifier for the stream processing application. Must be unique within the Kafka cluster. It is used as 1) the default client-id prefix, 2) the group-id for membership management, 3) the changelog topic prefix

- TYPE – string
- IMPORTANCE – high

bootstrap.servers – A list of host/port pairs to use for establishing the initial connection to the Kafka cluster. 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. 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)

- TYPE – list
- IMPORTANCE – high

replication.factor – The replication factor for change log topics and repartition topics created by the stream processing application

- TYPE – int
- DEFAULT – 1
- IMPORTANCE – high

state.dir – Directory location for state store

- TYPE – string
- DEFAULT – /tmp/kafka-streams
- IMPORTANCE – high

cache.max.bytes.buffering – Maximum number of memory bytes to be used for buffering across all threads

- TYPE – long
- DEFAULT – 10485760
- VALID VALUES – [0,...]
- IMPORTANCE – medium

client.id – An ID prefix string used for the client IDs of internal consumer, producer and restore-consumer, with pattern "-StreamThread–"

- TYPE – string
- DEFAULT – ""
- IMPORTANCE – medium

default.deserialization.exception.handler – Exception handling class that implements the *org.apache.kafka.streams.errors.DeserializationExceptionHandler* interface

- TYPE – class
- DEFAULT – org.apache.kafka.streams.errors.LogAndFailExceptionHandler
- IMPORTANCE – medium

default.key.serde – Default serializer / deserializer class for key that implements the *org.apache.kafka.common.serialization.Serde* interface

- TYPE – class
- DEFAULT – org.apache.kafka.common.serialization.Serdes$ByteArraySerde
- IMPORTANCE – medium

default.timestamp.extractor – Default timestamp extractor class that implements the *org.apache.kafka.streams.processor.TimestampExtractor* interface

- TYPE – class
- DEFAULT – org.apache.kafka.streams.processor.FailOnInvalidTimestamp
- IMPORTANCE – medium

default.value.serde – Default serializer/deserializer class for value that implements the *org.apache.kafka.common.serialization.Serde* interface

- TYPE – class
- DEFAULT – org.apache.kafka.common.serialization.Serdes$ByteArraySerde
- IMPORTANCE – medium

num.standby.replicas – The number of standby replicas for each task

- TYPE – int
- DEFAULT – 0
- IMPORTANCE – medium

num.stream.threads – The number of threads to execute stream processing

- TYPE – int
- DEFAULT – 1
- IMPORTANCE – medium

processing.guarantee – The processing guarantee that should be used. Possible values are "at_least_once" (default) and "exactly_once". Note that "exactly-once" processing requires a cluster of at least three brokers by default what is the recommended setting for production; for development you can change this, by adjusting broker setting *transaction.state.log.replication.factor*

- TYPE – string
- DEFAULT – at_least_once

- VALID VALUES – [at_least_once, exactly_once]
- IMPORTANCE – medium

security.protocol – Protocol used to communicate with brokers. Valid values are: "PLAINTEXT", "SSL", "SASL_PLAINTEXT", "SASL_SSL"

- TYPE – string
- DEFAULT – PLAINTEXT
- IMPORTANCE – medium

application.server – A host:port pair pointing to an embedded user defined endpoint that can be used for discovering the locations of state stores within a single KafkaStreams application

- TYPE – string
- DEFAULT – ""
- IMPORTANCE – low

buffered.records.per.partition – The maximum number of records to buffer per partition

- TYPE – int
- DEFAULT – 1000
- IMPORTANCE – low

commit.interval.ms – The frequency with which to save the position of the processor. Note, if *processing.guarantee* is set to "exactly_once", the default value is "100", otherwise the default value is "30000"

- TYPE – long
- DEFAULT – 30000
- IMPORTANCE – low

connections.max.idle.ms – Close idle connections after the number of milliseconds specified by this config

- TYPE – long
- DEFAULT – 540000
- IMPORTANCE – low

key.serde – Serializer/deserializer class for key that implements the *org.apache.kafka.common.serialization.Serde* interface. This config is deprecated, use *default.key.serde* instead

- TYPE – class
- DEFAULT – null
- IMPORTANCE – low

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

- TYPE – long
- DEFAULT – 300000
- VALID VALUES – [0,...]
- IMPORTANCE – low

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

- TYPE – list
- DEFAULT – “”
- IMPORTANCE – low

metrics.num.samples – The number of samples maintained to compute metrics

- TYPE – int
- DEFAULT – 2
- VALID VALUES – [1,...]
- IMPORTANCE – low

metrics.recording.level – The highest recording level for metrics

- TYPE – string
- DEFAULT – INFO
- VALID VALUES – [INFO, DEBUG]
- IMPORTANCE – low

metrics.sample.window.ms – The window of time a metrics sample is computed over

- TYPE – long
- DEFAULT – 30000
- VALID VALUES – [0,...]
- IMPORTANCE – low

partition.grouper – Partition grouper class that implements the *org.apache.kafka.streams.processor.PartitionGrouper* interface

- TYPE – class
- DEFAULT – org.apache.kafka.streams.processor.DefaultPartitionGrouper
- IMPORTANCE – low

poll.ms – The amount of time in milliseconds to block waiting for input

- TYPE – long
- DEFAULT – 100
- IMPORTANCE – low

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

- TYPE – int
- DEFAULT – 32768
- VALID VALUES – [0,...]
- IMPORTANCE – low

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

- TYPE – long
- DEFAULT – 1000
- VALID VALUES – [0,...]
- IMPORTANCE – low

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

- TYPE – long
- DEFAULT – 50
- VALID VALUES – [0,...]
- IMPORTANCE – low

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

- TYPE – int
- DEFAULT – 40000
- VALID VALUES – [0,...]
- IMPORTANCE – low

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

- TYPE – long
- DEFAULT – 100
- VALID VALUES – [0,...]
- IMPORTANCE – low

rocksdb.config.setter – A Rocks DB config setter class or class name that implements the *org.apache.kafka.streams.state.RocksDBConfigSetter* interface

- TYPE – class
- DEFAULT – null
- IMPORTANCE – low

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

- TYPE – int
- DEFAULT – 131072
- VALID VALUES – [0,...]
- IMPORTANCE – low

state.cleanup.delay.ms – The amount of time in milliseconds to wait before deleting state when a partition has migrated. Only state directories that have not been modified for at least *state.cleanup.delay.ms* will be removed

- TYPE – long
- DEFAULT – 600000
- IMPORTANCE – low

timestamp.extractor – Timestamp extractor class that implements the *org.apache.kafka.streams.processor.TimestampExtractor* interface. This config is deprecated, use *default.timestamp.extractor* instead

- TYPE – class
- DEFAULT – null
- IMPORTANCE – low

value.serde – Serializer/deserializer class for value that implements the *org.apache.kafka.common.serialization.Serde* interface. This config is deprecated, use *default.value.serde* instead

- TYPE – class
- DEFAULT – null
- IMPORTANCE – low

windowstore.changelog.additional.retention.ms – Added to a windows *maintainMs* to ensure data is not deleted from the log prematurely. Allows for clock drift. Default is *1 day*

- TYPE – long
- DEFAULT – 86400000
- IMPORTANCE – low

zookeeper.connect – Zookeeper connect string for ADS topics management. This config is deprecated and will be ignored as Streams API does not use Zookeeper anymore

- TYPE – string
- DEFAULT – “”
- IMPORTANCE – low