Confluent KAFKA Administration

Rajesh Pasham

Configuring threads and performance

- With your favorite text editor, open your server.properties file copy.
- Adjust the following parameters:
  - message.max.bytes=1000000
  - num.network.threads=3
  - num.io.threads=8
  - background.threads=10
  - queued.max.requests=500
  - socket.send.buffer.bytes=102400
  - socket.receive.buffer.bytes=102400
  - socket.request.max.bytes=104857600
  - num.partitions=1

Configuring threads and performance

- Here is an explanation of every parameter:
  - message.max.bytes:
    - Default value: 10 00 000.
    - This is the maximum size, in bytes, for each message.
    - This is designed to prevent any producer from sending extra large messages and saturating the consumers.
  - num.network.threads:
    - Default value: 3.
    - This is the number of simultaneous threads running to handle a network's request.
    - If the system has too many simultaneous requests, consider increasing this value.

Configuring threads and performance

- Here is an explanation of every parameter:
  - num.io.threads:
    - Default value: 8.
    - This is the number of threads for Input Output operations.
    - This value should be at least the number of present processors.
  - background.threads:
    - Default value: 10.
    - This is the number of threads for background jobs.
    - For example, old log files deletion.

Configuring threads and performance

- Here is an explanation of every parameter:
  - queued.max.requests:
    - Default value: 500.
    - This is the number of messages queued while the other messages are processed by the I/O threads.
    - Remember, when the queue is full, the network threads will not accept more requests.
    - If your application has erratic loads, set this to a value at which it will not throttle.
  - socket.send.buffer.bytes:
    - Default value: 102 400.
    - This is SO_SNDBUFF buffer size, used for socket connections.

Configuring threads and performance

- Here is an explanation of every parameter:
  - socket.receive.buffer.bytes:
    - Default value: 102 400.
    - This is SO_RCVBUFF buffer size, also used for socket connections.
  - socket.request.max.bytes:
    - Default value: 104 857 600.
    - This is the maximum request size, in bytes, that the server can accept.
    - It should always be smaller than the Java heap size.

Configuring threads and performance

- Here is an explanation of every parameter:
  - num.partitions:
    - Default value: 1.
    - This is the number of default partitions of a topic, without giving any partition size.

Configuring the log settings

- The log settings are fundamental, so it is the way the messages are persisted in the broker machine.
- With your favorite text editor, open your server.properties file copy.
- Adjust the following parameters:
  - log.segment.bytes=1073741824
  - log.roll.hours=168
  - log.cleanup.policy=delete
  - log.retention.hours=168

Configuring the log settings

- Adjust the following parameters:
  - log.retention.bytes=-1
  - log.retention.check.interval.ms=30000
  - log.cleaner.enable=false
  - log.cleaner.threads=1
  - log.cleaner.backoff.ms=15000
  - log.index.size.max.bytes=10485760
  - log.index.interval.bytes=4096
  - log.flush.interval.messages=Long.MaxValue
  - log.flush.interval.ms=Long.MaxValue

Configuring the log settings

- Here is an explanation of every parameter:
  - log.segment.bytes:
    - Default value: 1 GB.
    - This defines the maximum segment size in bytes (the concept of segment will be explained later).
    - Once a segment file reaches that size, a new segment file is created.
    - Topics are stored as a bunch of segment files in the log directory.
    - This property can also be set per topic.

Configuring the log settings

- Here is an explanation of every parameter:
  - log.roll.{ms,hours}:
    - Default value: 7 days.
    - This defines the time period after a new segment file is created, even if it has not reached the size limit.
    - This property can also be set per topic.

Configuring the log settings

- Here is an explanation of every parameter:
  - log.cleanup.policy:
    - Default value: delete.
    - Possible options are delete or compact.
    - If the delete option is set, the log segments will be deleted periodically when it reaches its time threshold or size limit.
    - If the compact option is set, log compaction is used to clean up obsolete records.
    - This property can also be set per topic.

Configuring the log settings

- Here is an explanation of every parameter:
  - log.retention.{ms,minutes,hours}:
    - Default value: 7 days.
    - This defines the time to retain the log segments.
    - This property can also be set per topic.
  - log.retention.bytes:
    - Default value: -1.
    - This defines the number of logs per partition to retain before deletion.
    - This property can also be set per topic.
    - The segments are deleted when the log time or size limits are reached.

Configuring the log settings

- Here is an explanation of every parameter:
  - log.retention.check.interval.ms:
    - Default value is five minutes.
    - This defines the time periodicity at which the logs are checked for deletion to meet retention policies.
  - log.cleaner.enable:
    - To enable log compaction, set this to true.
  - log.cleaner.threads:
    - Indicates the number of threads working on clean logs for compaction.

Configuring the log settings

- Here is an explanation of every parameter:
  - log.cleaner.backoff.ms:
    - Periodicity at which the logs will check whether any log needs cleaning.
  - log.index.size.max.bytes:
    - Default value: 1 GB.
    - This sets the maximum size, in bytes, of the offset index.
    - This property can also be set per topic.

Configuring the log settings

- Here is an explanation of every parameter:
  - log.index.interval.bytes:
    - Default value: 4096.
    - The interval at which a new entry is added to the offset index (the offset concept will be explained later).
    - In each fetch request, the broker does a linear scan for this number of bytes to find the correct position in the log to begin and end the fetch.
    - Setting this value too high may mean larger index files and more memory used, but less scanning.

Configuring the log settings

- Here is an explanation of every parameter:
  - log.flush.interval.messages:
    - Default value: 9 223 372 036 854 775 807.
    - The number of messages kept in memory before flushed to disk.
    - It does not guarantee durability, but gives finer control.
  - log.flush.interval.ms:
    - Sets maximum time in ms that a message in any topic is kept in memory before it is flushed to disk.
    - If not set, it is used the value in log.flush.scheduler.interval.ms.

Configuring the replica settings

- The replication is configured for reliability purposes.
- Replication can also be tuned.
- With your favorite text editor, open your server.properties file copy.
- Adjust the following parameters:
  - default.replication.factor=1
  - replica.lag.time.max.ms=10000
  - replica.fetch.max.bytes=1048576
  - replica.fetch.wait.max.ms=500

Configuring the replica settings

- Adjust the following parameters:
  - num.replica.fetchers=1
  - replica.high.watermark.checkpoint.interval.ms=5000
  - fetch.purgatory.purge.interval.requests=1000
  - producer.purgatory.purge.interval.requests=1000
  - replica.socket.timeout.ms=30000
  - replica.socket.receive.buffer.bytes=65536

Configuring the replica settings

- Here is an explanation of these settings:
  - default.replication.factor:
    - Default value: 1.
    - For an automatically created topic, this sets how many replicas it has.
  - replica.lag.time.max.ms:
    - Default value: 10 000.
    - There are leaders and followers; if a follower has not sent any fetch request or is not consumed up in at least this time, the leader will remove the follower from the ISR list and consider the follower dead.

Configuring the replica settings

- Here is an explanation of these settings:
  - replica.fetch.max.bytes:
    - Default value: 1 048 576.
    - In each request, for each partition, this value sets the maximum number of bytes fetched by a request from its leader.
    - Remember that the maximum message size accepted by the broker is defined by message.max.bytes (broker configuration) or max.message.bytes (topic configuration).

Configuring the replica settings

- Here is an explanation of these settings:
  - replica.fetch.wait.max.ms:
    - Default value: 500.
    - This is the maximum amount of time for the leader to respond to a replica's fetch request.
    - Remember that this value should always be smaller than the replica.lag.time.max.ms.
  - num.replica.fetchers:
    - Default value: 1.
    - The number of fetcher threads used to replicate messages from a source broker.
    - Increasing this number increases the I/O rate in the following broker.

Configuring the replica settings

- Here is an explanation of these settings:
  - replica.high.watermark.checkpoint.interval.ms:
    - Default value: 500.
    - The **high watermark** (**HW**) is the offset of the last committed message.
    - This value is the frequency at which each replica saves its high watermark to the disk for recovery.
  - fetch.purgatory.purge.interval.requests:
    - Default value: 1000.
    - Purgatory is the place where the fetch requests are kept on hold till they can be serviced (great name, isn't?).
    - The purge interval is specified in number of requests (not in time) of the fetch request purgatory.

Configuring the replica settings

- Here is an explanation of these settings:
  - producer.purgatory.purge.interval.requests:
    - Default value: 1000.
    - It sets the purge interval in number of requests (not in time) of the producer request purgatory (do you catch the difference to the previous parameter?).

Configuring the ZooKeeper settings

- Apache Zookeeper is a centralized service for maintaining configuration information providing distributed synchronization.
- ZooKeeper is used in Kafka for cluster management and to maintain the topics information synchronized.
- With your favorite text editor, open your server.properties file copy.

Configuring the ZooKeeper settings

- Adjust the following parameters:
  - zookeeper.connect=127.0.0.1:2181,192.168.0.32:2181
  - zookeeper.session.timeout.ms=6000
  - zookeeper.connection.timeout.ms=6000
  - zookeeper.sync.time.ms=2000
- Here is an explanation of these settings:
  - zookeeper.connect:
    - Default value: null.
    - This is a comma-separated value in the form of the hostname:port string, indicating the Zookeeper connection.

Configuring the ZooKeeper settings

- Here is an explanation of these settings:
  - zookeeper.connect:
    - Specifying several connections ensures the Kafka cluster reliability and continuity.
    - When one node fails, Zookeeper uses the chroot path (/chroot/path) to make the data available under that particular path.
    - This enables having the Zookeeper cluster available for multiple Kafka clusters.
    - This path must be created before starting the Kafka cluster, and consumers must use the same string.

Configuring the ZooKeeper settings

- Here is an explanation of these settings:
  - zookeeper.session.timeout.ms:
    - Default value: 6000.
    - Session timeout means that if in this time period a heartbeat from the server is not received, it is considered dead.
    - This parameter is fundamental, since if it is long and if the server is dead the whole system will experience problems.
    - If it is small, a living server could be considered dead.
  - zookeeper.connection.timeout.ms:
    - Default value: 6000.
    - This is the maximum time that the client will wait while establishing a connection to Zookeeper.

Configuring the ZooKeeper settings

- Here is an explanation of these settings:
  - zookeeper.sync.time.ms:
    - Default value: 2000.
    - This is the time a Zookeeper follower can be behind its Zookeeper leader.

Configuring other miscellaneous parameters

- No parameter should be left at default when optimal behavior is desired.
- These parameters should be taken into consideration to achieve the best performance.
- With your favorite text editor, open your server.properties file copy.
- Adjust the following parameters:
  - auto.create.topics.enable=true
  - controlled.shutdown.enable=true
  - controlled.shutdown.max.retries=3

Configuring other miscellaneous parameters

- Adjust the following parameters:
  - controlled.shutdown.retry.backoff.ms=5000
  - auto.leader.rebalance.enable=true
  - leader.imbalance.per.broker.percentage=10
  - leader.imbalance.check.interval.seconds=300
  - offset.metadata.max.bytes=4096
  - max.connections.per.ip=Int.MaxValue
  - connections.max.idle.ms=600000
  - unclean.leader.election.enable=true
  - offsets.topic.num.partitions=50
  - offsets.topic.retention.minutes=1440

Configuring other miscellaneous parameters

- Adjust the following parameters:
  - offsets.retention.check.interval.ms=600000
  - offsets.topic.replication.factor=3
  - offsets.topic.segment.bytes=104857600
  - offsets.load.buffer.size=5242880
  - offsets.commit.required.acks=-1
  - offsets.commit.timeout.ms=5000

Configuring other miscellaneous parameters

- Here is an explanation of these settings:
  - auto.create.topics.enable:
    - Default value: true.
    - Suppose that metadata is fetched or a message is produced for a nonexistent topic; if this value is true, the topic will automatically be created.
    - In production environments, this value should be false.
  - controlled.shutdown.enable:
    - Default value: true.
    - If this value is true, when a shutdown is called on the broker, the leader will gracefully move all the leaders to a different broker.
    - When it is true, the availability is increased.

Configuring other miscellaneous parameters

- Here is an explanation of these settings:
  - controlled.shutdown.max.retries:
    - Default value: 3.
    - This is the maximum number of retries the broker tries a controlled shutdown before making a forced and unclean shutdown.
  - controlled.shutdown.retry.backoff.ms:
    - Default value: 5000.
    - Suppose that a failure happens (controller fail over, replica lag, and so on); this value determines the time to wait before recovery from the state that caused the failure.

Configuring other miscellaneous parameters

- Here is an explanation of these settings:
  - auto.leader.rebalance.enable:
    - Default value: true.
    - If this value is true, the broker will automatically try to balance the partition leadership among the brokers.
    - At regular intervals, a background thread checks and triggers leader balance if required, setting the leadership to the preferred replica of each partition if available.
  - leader.imbalance.per.broker.percentage:
    - Default value: 10.
    - This value is specified in percentages and is the leader imbalance allowed per broker (the leader imbalance will be explained later).
    - The cluster will rebalance the leadership if this percentage goes above the set value.

Configuring other miscellaneous parameters

- Here is an explanation of these settings:
  - leader.imbalance.check.interval.seconds:
    - Default value: 300.
    - This value is the frequency at which to check the leader imbalance by the controller.
  - offset.metadata.max.bytes:
    - Default value: 4096.
    - This is the maximum size allowed to the client for a metadata to be stored with an offset commit.
  - max.connections.per.ip:
    - Default value: 2 147 483 647.
    - This is the maximum number of connections that the broker accepts from each IP address.

Configuring other miscellaneous parameters

- Here is an explanation of these settings:
  - connections.max.idle.ms:
    - Default value: 600 000.
    - This is the idle connection's timeout.
    - The server socket processor threads close the connections that idle more than this value.
  - unclean.leader.election.enable:
    - Default value: true.
    - If this value is true, the replicas that are not ISR can become leaders.
    - Doing so may result in data loss.

Configuring other miscellaneous parameters

- Here is an explanation of these settings:
  - offsets.topic.num.partitions:
    - Default value: 50.
    - This is the number of partitions for the offset commit topic.
    - This value cannot be changed post deployment.
  - offsets.retention.minutes:
    - Default value: 1440.
    - This is the log retention window for the offsets topic.
    - This is the time to keep the offsets.
    - Passed this, the offsets will be marked for deletion.

Configuring other miscellaneous parameters

- Here is an explanation of these settings:
  - offsets.retention.check.interval.ms:
    - Default value: 60 000.
    - This is the frequency at which to check for stale offsets
  - offsets.topic.replication.factor:
    - Default value: 3.
    - This is the number of replicas for the offset commit topic.
    - The higher this value, the higher the availability.
    - If the number of brokers is lower than the replication factor, the number of replicas will be equal to the number of brokers.

Configuring other miscellaneous parameters

- Here is an explanation of these settings:
  - offsets.retention.check.interval.ms:
    - Default value: 60 000.
    - This is the frequency at which to check for stale offsets
  - offsets.topic.replication.factor:
    - Default value: 3.
    - This is the number of replicas for the offset commit topic.
    - The higher this value, the higher the availability.
    - If the number of brokers is lower than the replication factor, the number of replicas will be equal to the number of brokers.

Configuring other miscellaneous parameters

- Here is an explanation of these settings:
  - offsets.topic.segment.bytes:
    - Default value: 104 857 600.
    - This is the segment size for the offsets topic.
    - The lower this value, the faster the log compaction and cache loading are.
  - offsets.load.buffer.size:
    - Default value: 5 242 880.
    - This is the batch size to be used for reading offset segments when loading offsets into the cache.

Configuring other miscellaneous parameters

- Here is an explanation of these settings:
  - offsets.commit.required.acks:
    - Default value: -1.
    - This is the number of acknowledgements required before the offset commit can be accepted.
    - It is recommended to not override the default value of -1, meaning no acknowledgements required.
  - offsets.commit.timeout.ms:
    - Default value: 5000.
    - This is the time that an offset commit will be delayed until all replicas for the offsets topic receive the commit or this time value is reached.
    - This value is similar to the producer request.timeout.ms.

Configuring other miscellaneous parameters

- Here is an explanation of these settings:
  - offsets.commit.required.acks:
    - Default value: -1.
    - This is the number of acknowledgements required before the offset commit can be accepted.
    - It is recommended to not override the default value of -1, meaning no acknowledgements required.
  - offsets.commit.timeout.ms:
    - Default value: 5000.
    - This is the time that an offset commit will be delayed until all replicas for the offsets topic receive the commit or this time value is reached.
    - This value is similar to the producer request.timeout.ms.