Quorum Queues

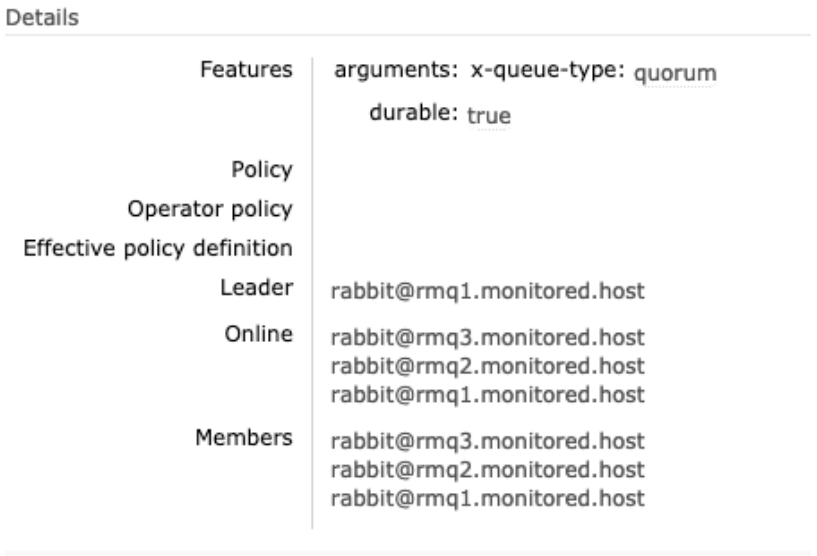
Introduction to Quorum Queues

- The most significant new features -> Quorum Queues
- Important when RabbitMQ is used in a clustered installation, it provides less network intensive message replication using the Raft protocol underneath
- Quorum queues are created differently, not by policies.

```
Queue.declare(publisher_chan, "my-quorum-queue", durable: true, arguments: [ "x-queue-type": "quorum" ])
```

- All communication is routed to the Queue Leader, which means the queue leader locality has an effect on the latency and bandwidth requirement of the messages, however the effect should be lower than it was in Classic Queues.
- Consuming from a Quorum Queue is done in the same fashion as other types of queues.

Quorum Queues



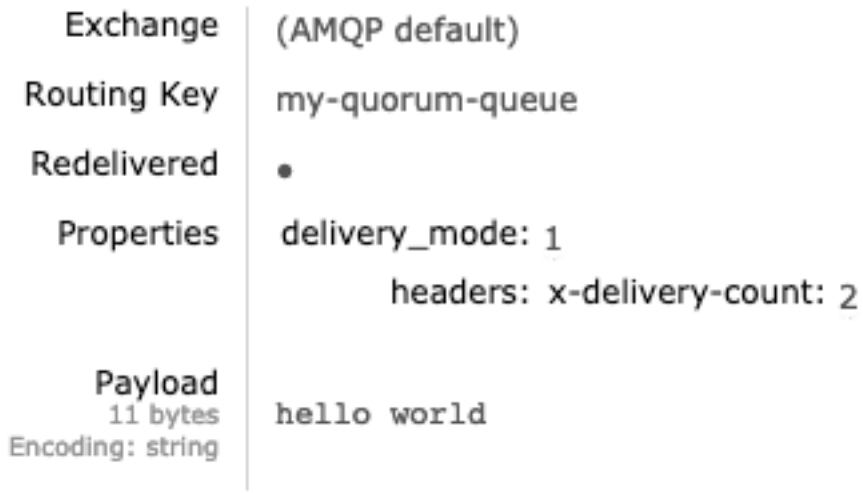
- They can not be non-durable, because the Raft log is always written to disk, so they can never be declared as transient.
- They also do not support, as of 3.8.2, message TTLs and message priorities.
- They also cannot be declared as exclusive, which would mean they get deleted as soon as the consumer disconnects.
- Since all messages in Quorum Queues are persistent, the AMQP “delivery-mode” option has no effect on their operation

Single Active Consumer

- Single Active Consumer enables you to attach multiple consumers to a queue, while only one of the consumers is active.
- Queue.declare(publisher_chan, "single-active-queue", durable: true, arguments: ["x-queue-type": "quorum", "x-single-active-consumer": true])
- Basic.get, or inspecting the message on the Management Interface, can not be done with Single Active Consumer queues.

Poison Messages

- QQ lets you handle the so-called poison-messages more effectively than before, as previous implementations often suffered from the inability to give up retrying in the case of a stuck message, or had to keep track of how many times a message was delivered in an external database.
- redelivered flag, helps you to check if the message is a duplicate or not. The same flag exists, however it was extended with the x-delivery-count header, which keeps track of how often this requeueing has occurred.



Controlling the members of the quorum

- Quorum queues do not automatically change the group of followers / leaders
- Adding a new member to the quorum can be achieved using the grow command

  rabbitmq-queues grow rabbit@$NEW_HOST all
- Removing a now stale, for example deleted, host from the members can be done through the shrink command.

  rabbitmq-queues shrink rabbit@$OLD_HOST
- We can also rebalance the queue masters so that the load is equal on the nodes:

  rabbitmq-queues rebalance all
- Which (in bash) will display a nice table with statistics on the number of masters on nodes. On Windows, use the --formatter json flag to get a readable output.

Lab : Quorum Queues