Setup / Concepts

- Same hw setup for all tests (constant)
- Same samples by num. of packets and packets sizes (constant)
- Measures sampled from the client side (constant - black box testing, what does the messaging user really see?)
- Default settings for everything (constant - should work out of the box right? Otherwise fix your defaults)
- No I didn't test “option foo”, if it's necessary it should be in the defaults (constant)
- All setups and how-to's are documented. It's easy for anybody to replicate the tests
- Split into many graphs to make it easier to ready
References

- https://github.com/fabbione/rhos-ha-deploy
- https://github.com/fabbione/oslo-messaging-clients
- (modified version of kgiusti tree, to print more info related to performances and adds info on setup and how to run tests)
What did we measure?

Every graph contains the transactions/sec relation with packet size. The higher the transation number the better. All tests have been repeated with/without LB to grasp overhead/benefits of a LB managed queue.

The matrix:

<table>
<thead>
<tr>
<th></th>
<th>+LB</th>
<th>Hot-Standby</th>
<th>-LB</th>
<th>Hot-Standby (*)</th>
<th>-LB</th>
<th>A/A clustered (not supported) (*)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Qpid</td>
<td></td>
<td></td>
<td>Qpid</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+LB</td>
<td></td>
<td>Hot-Standby</td>
<td>-LB</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-LB</td>
<td></td>
<td>Hot-Standby (*)</td>
<td>-LB</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+LB</td>
<td></td>
<td>A/A clustered (not supported)</td>
<td>-LB</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-LB</td>
<td></td>
<td>A/A clustered (not supported) (*)</td>
<td>-LB</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RabbitMQ</td>
<td>+LB</td>
<td>Mirrored queues</td>
<td>-LB</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-LB</td>
<td></td>
<td>Mirrored queues (*)</td>
<td>-LB</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+LB</td>
<td></td>
<td>No mirrored queues (*)</td>
<td>-LB</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-LB</td>
<td></td>
<td>No mirrored queues (*)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(*) both client and server connects to the same machine. Only useful to measure LB overhead or performance when using qpid_hosts/rabbitmq_hosts.
QPID via olso.messaging

Payload size

Transactions/sec

0 1 2 4 8 16 32 64 128

+LB - Hot Standby
-LB - Hot Standby
+LB - Cluster mode
-LB - Cluster mode
RabbitMQ via oslo.messaging

- LB - mirrored queues
- LB - Hot Standby
+ LB - mirrored queues
+ LB - Hot Standby

Transactions/sec vs Payload size

0 1 2 4 8 16 32 64 128
RabbitMQ - raw (java test client)
RabbitMQ vs Qpid
Did you really think it was that simple?!?!
RabbitMQ vs Qpid raw vs oslo.messaging

- Rabbit raw +LB - mirrored queues
- Qpid raw +LB - Hot Standby
- Rabbit oslo +LB - mirrored queues
- Qpid oslo +LB - Hot Standby

Transactions/sec vs Payload size
Did you forget to add numbers/data to the previous slide?
RabbitMQ vs Qpid raw

Payload size

Transactions/sec

Rabbit +LB - mirrored queues

Qpid +LB - Hot Standby
RabbitMQ vs Qpid via oslo.messaging

- Rabbit + LB - mirrored queues
- Qpid + LB - Hot Standby

The graph shows the comparison between RabbitMQ and Qpid using oslo.messaging. The x-axis represents payload size, while the y-axis shows transactions per second.
Conclusions with supported vs what customers think they want

- Raw performance shows that qpid is better for small packet sizes, rabbit for big packet size. Basically, it means absolutely nothing unless there is an analysis of OpenStack traffic patterns.
- Oslo.messaging provides a “natural QoS” for any messaging queue.
- The only driver for picking up a message queue (and LB or non-LB) access is driven only by how-fast you want to recover (and if you care about message persistence or not)