Dispatch Roadmap

Completed Releases

- 0.1 - 1/13/2014
  - Initial Release
  - Jiras Resolved
- 0.2 - 4/21/2014
  - Added configurable address-forwarding semantics
  - Jiras Resolved
- 0.3 - 1/16/2015
  - Theme: Management and Stability
  - Jiras Resolved
  - Management and Configuration Features
  - Stability Features and Fixes
- 0.4 - 4/9/2015
  - Theme: Link Routing and Broker Integration
  - Jiras Resolved
  - Features
    - Link-Attach routing - Remote broker access
    - Major improvements to the routing protocol
- 0.5 - 9/14/2015
  - Jiras
  - Features
    - Compatibility with Proton 0.10
    - SASL Integration for Connectors and Listeners
    - Per-Connection management access for
      - SSL/TLS/en-clair
      - Endpoint certificates for SSL/TLS
      - SASL mechanism in-effect
      - Authenticated identity of clients

Planned Releases

- 0.6 - March, 2016
  - What's New in 0.6.0
  - Jiras
  - Planned Features
    - Router core update
      - Remove lock contention issue around routing data structures
      - Improved end-to-end flow control, protects from filling up output FIFOs
    - Adaptive balancing of routed anycast messages
    - Improved waypoint implementation
- 0.7 - June, 2016
  - Jiras
- 0.8 - September, 2016
  - Jiras

Feature Details

Link Cost in Route Computation

Allow each inter-router connection to be configured with a cost metric. The metric is a natural number (non-zero positive integer) and represents relative cost where higher numbers denote higher cost. This value shall then be added to the link-state data and used in the computation of least-cost paths across the router topology. This involves the following changes:

- Add link-cost value to the configuration of inter-router connectors and listeners.
- Enhance the HELLO protocol to negotiate cost, where each side provides requested cost and negotiated cost. Requested cost is the locally configured cost metric and negotiated cost is the greater of the local configured cost and the peer's requested cost.
- Enhance the Link State Update messages to contain the negotiated link cost per link-state record.
- Enhance the Dijkstra lowest-cost-path computation to use the cost metric instead of the hard-coded value of 1.

Possible extensions and enhancements to consider:

- Asymmetric connection cost.

Link-Attach Routing (0.4)

Currently Dispatch performs routing on delivered messages based on the address in the message's metadata. There is a second mode of routing in which routing is performed on the link-attach command. In this case, an attached link causes another link to be established along the next-hop connection to the destination. The process is repeated until a serially connected set of links is established from node-to-node across the network. The result is like a Virtual Circuit for AMQP.

In each router, the links are paired, incoming to outgoing. The forwarding of performatives between the links is trivially simple. Everything that comes out of one is propagated out the other in both directions. There is no need to decode or interpret any of the transfers or controls flowing between the links.

Address-forwarding semantics apply to link-attach routing in a way similar to how they apply to message routing. Link-attaches can be routed to the nearest destination, or balanced across the full set of destinations. The only forwarding semantic that does not apply to link-attach is multicast. All link-routing is ultimately point-to-point.

The following work items and considerations apply:

- There must be a determination as to when a link-attach should be routed and when it should be simply terminated in the router.
- There needs to be a way to identify an endpoint connection as being the path to a particular node (the destination). This could possibly be done by creating a "pilot" subscription (receiving link) that's sole purpose is to associate a node destination address with a connection/container. It could also be done through provisioning (like a waypoint).
- Forwarding logic must be added to propagate flow commands from consumer to producer.
- Logic must be added that propagates link-loss/detach outward in both directions from the site of a failure.

Roadmap Items Requiring Description

- Input buffer for producer-endpoints and waypoints
- Adaptive/balanced delivery of messages to anycast destinations
- Websocket encapsulation option for listeners
- SASL integration for listeners and connectors
- Expose AMQP Session in the container.h API
- Establish a separate inter-router session for router-control messages
- Establish a single, separate inter-router session for routed links
    - Consider a policy for inter-router sessions (i.e. QoS for routed links)
- Architecture for destination lookup (for high scale)
    - Router plugin for address lookup
    - Distributed service for address lookup
- Multi-tennancy
    - Partition management entity space by application (open.hostname)
    - Address translation based on application
- Access Control
    - Policy enforcement based on (application, authentic id, address, action)
    - Policy storage and distribution
- Plugin for Exchange/Binding-style routing