General Enhancements Added

- `src/iterator.c`
  - `qd_iterator_equal_n`
    An iterator string-compare was needed that does not use null-termination to determine length.
- `src/router_core/modules/mobile_sync/mobile.c`
  - Consolidated pending addresses into one list instead of separate add and delete lists (addresses are only ever in one list at a time).
  - Added “update” handling for changes in address state. These are sent as “add” operations in the protocol, and are interpreted as add/update on the receive side.
- `include/qpid/dispatch/discriminator.h`
  - Added QD_DISCRIMINATOR_BYTES constant to denote discriminator length not including the terminating null. This makes it easier to generate and compare fields in annotations.

Updates for Mesh of Edges

Configuration and Management

- Added a new role for connections: inter-edge
- Added QD_CAPABILITY_INTER_EDGE terminus capability.
- Added QD_LINK_INTER_EDGE link-type.

Edge-peer tracking

- Track connections to edge-peers using container-id
- Select a primary connection for each edge-peer
- Data types in src/router_core/router_core_private.h
  - qdr_edge_peer_t
- Functions in src/router_core/modules/edge_router/connection_manager.c
  - qdr_find_edge_peer_CT
  - qdr_inter_edge_connection_setup_CT
  - qdr_inter_edge_connection_cleanup_CT

Dynamic address support

Note: The format for a dynamic source address issued by an edge router remains unchanged and is: `amqp:/_edge/<router-id>/temp.<discriminator>`

- Updates to iterator:
    - Maintain a list of edge-peer-router-ids in the iterator module.
    - Use `H<router-id>` hash format for temporary addresses in the set of edge-peers.
    - There are now 3 cases for edge-format addresses on an edge router:
        - RID is this router => Ltemp.<discriminator> (deliver to consumer)
        - RID is different edge, same mesh => H<router-id> (deliver to peer edge)
        - RID is different edge, different mesh => L_edge (deliver to interior)
    - Files
        - include/qpid/dispatch/iterator.h
        - src/iterator.c
        - tests/field_test.c
- Use of iterator enhancements
    - src/router_core/modules/edge_router/connection_manager.c
        - Add/delete router-ids in the iterator module.
        - qdr_inter_edge_peer_activate_CT - Establish an outgoing link on each edge-peer primary connection, bound to the H-class edge-summary address for the connected edge router.
    - src/router_core/router_core.c - cleanup of iterator module's id list during shutdown.
- tests/system_tests_edge_mesh.py - Dynamic address tests for standalone mesh (no interior router) and connected mesh (interior router connected).

Proxy links for local destinations

- Introduced "proxy" flag for outgoing links to differentiate real local destinations from proxy local destinations. Proxy links are links that reach a destination via another router.
- For each local destination on an edge router, maintain an inbound link from each edge-peer in the mesh over that peer's primary connection.

Negotiate an identity shared by all edges in a mesh

- include/qpid/dispatch/amqp.h - Added constants for new addresses and keys
- src/amqp.c - Added constants for new addresses and keys
- src/router_core/modules/mesh_discovery/mesh_discovery_edge.c
    - Implementation of the edge-side identity negotiation and notification to the interior
- src/router_core/modules/mesh_discovery/mesh_discovery_interior.c
    - Implementation of the interior-side storage and use of edge-mesh identities

Added a new inter-router message annotation field: ingress-mesh

This is used for messages originating from an edge mesh. The field is set with the negotiated mesh identity.

Sole Destination Mesh

If all of the destinations for an address are on the same edge-mesh, this is called “sole-destination-mesh”. This is tracked for local and remote destinations per-address. This is used to block flow of deliveries from a particular edge-mesh into the interior. If all possible destinations for an address are back on the same mesh, don’t allow the flow of deliveries from that mesh.

- Local-sole-destination-mesh is true when all local destinations for the address (addr->rlinks) are to the same mesh.
    - Local-sole-destination-mesh state is synchronized between interior routers via the mobile-address-sync protocol (MAU/MAR). This synchronization is used to compute remote-sole-destination-mesh state.
- Remote-sole-destination-mesh is true when all remote (addr->rnodes) destinations are to the same mesh. This is important because it is possible for routers in an edge mesh to be connected to different interior routers.

Generalization of the detection of edge-echo

- Prevents two problems:
    - Deliveries looping back to the same edge-mesh via the interior. This causes duplicate deliveries with multicast distribution.
    - Deliveries being erroneously sent to the interior when there is no reachable destination via the interior. This causes spurious delivery RELEASES.
- src/router_core/forwarder.c: qdr_forward_edge_echo_CT - Prevents forwarding of deliveries back to the edges from which it came.
- src/router_core/modules/edge_addr_tracking/edge_addr_tracking.c: qdrc_can_send_address - Controls the start/stop protocol for address-specific uplinks from edge to interior.