AMQP Technical Workshop Meeting Notes

February 18-22, 2013.
Microsoft Campus, Redmond, WA

Contents

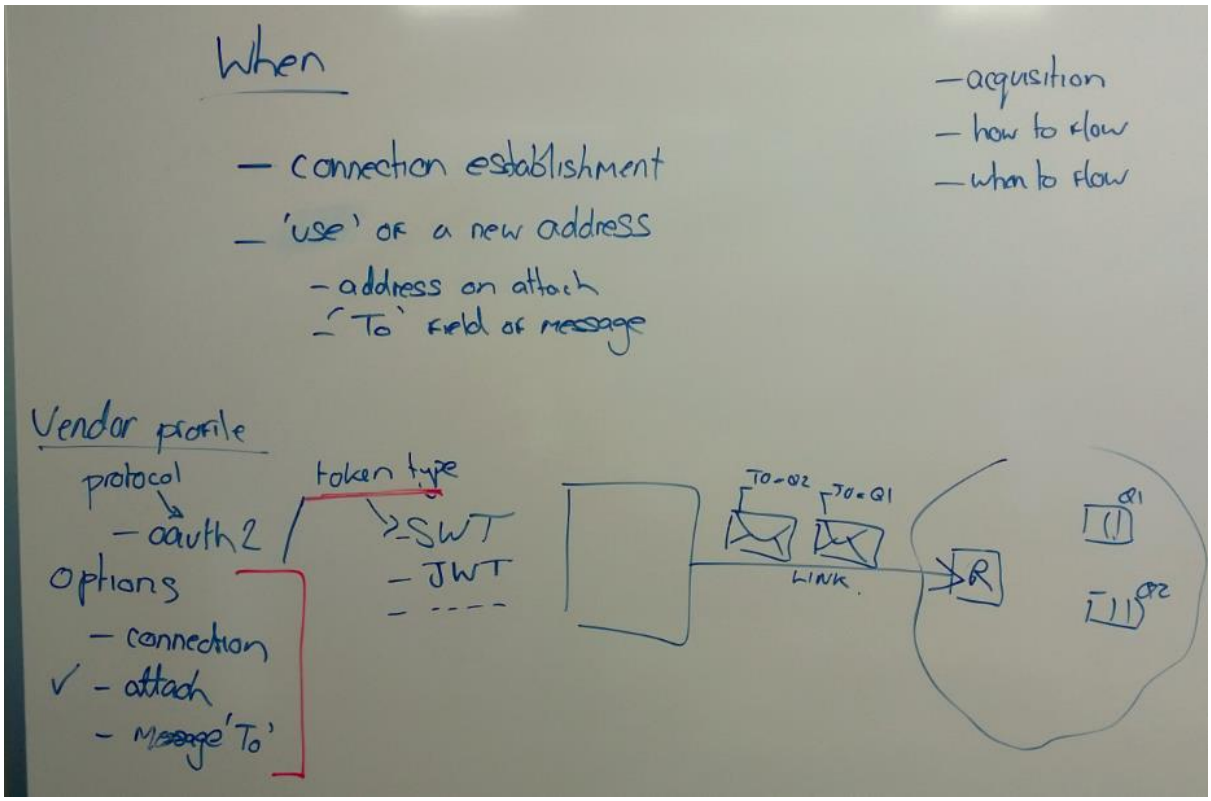
Attendees

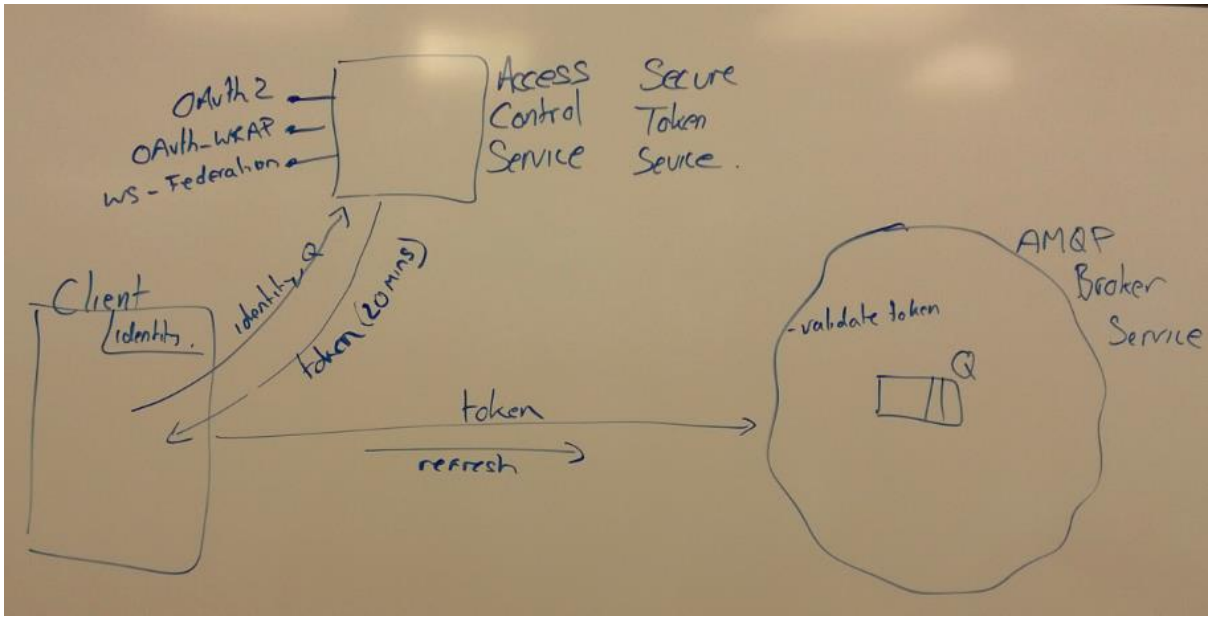
- Rob Godfrey (JPMC)
- Robbie Gemmell (JPMC)
- Rafael Schloming (Red Hat)
- William Henry (Red Hat)
- Alex Kritikos (SoftwareAG)
- Xin Chen (Microsoft)
- Affan Dar (Microsoft)
- David Ingham (Microsoft)

Planned agenda

- Mon
  - AM: claims-based security
  - PM: addressing
- Tue
  - AM: management
  - PM: JMS / OpenMAMA
- Wed
  - AM: WebSockets
  - PM: SCTP
- Thu
  - Authoring: management, WebSockets, security, JMS as appropriate
- Fri
  - Authoring continued

Claims-based security





Decisions

- Token can flow in properties of open, attach, and flow frames and also on Message delivery annotations.
- Connection open configuration
  - Desired capabilities:
    - CLAIMS_BASED_AUTH
      - Support for claims-based auth
  - Properties
    - CBA_CHECKS : List (entity checks)
      - LCL_TRM (local terminus)
      - RMT_TRM (remote terminus)
      - MSG (message)
    - CBA_TOKENS : List (token type preference list)
      - SWT
      - JWT
      - SAML
- Link attach configuration
  - Properties
    - CBA_MSG_CHECK
      - true/false. Omission implies false.
- Token property name
  - CBA_TOKEN
    - Map from address to token.
    - Token value is a single string or an array of strings

AMQP Global Addressing

Use cases

- AMQP firewall
- Receivers bind on a port
- Receivers await remotely-initiated links
- Request-response

    - Peer-to-peer
    - Intermediated
- Reply-to
- Mobile address
    - Endpoint owns address
    - No dedicated network mailbox
- Redirect
    - Connection
    - Link
- Collective addresses
    - Load balancing
    - Funnelling
    - Fanout
- DNS based load balancing
- Simple host addressing
- Global federated topics
    - Trusted intermediaries
    - No direct trust required between senders and receivers
- Return receipts (positive and ngative)
- Anonymous back-channel

Address characteristics

- Comparison?
- Canonicalization
- Post-it-able
    - “Means the same thing everywhere”
- Have attributes/annotations?
    - Is this a characteristic of address or node?

Data required to send/receive a message

Connection establishment

- IP address/port
- Protocol: ssl, sasl, websocket, …
- Hostname (for v-host/multi-tenant)
- Credentials
    - SASL scheme
    - STS host/port, token acquisition scheme
    - SASL username/password
    - SSL cert stuff (DBs, credentials)

Link attach

- Local terminus
- Remote terminus
- Filters
- Properties
- Capabilities

Transfer

- to

- reply-to

Syntax

AMQP addresses follow the URL syntax. Examples:

1) amqp://<domain>/<path>
2) amqp://!<identifier>/<path>

Addresses are interpreted in two parts <namespace>/<name>. At different points in the network the division between the namespace and the path will be different. At any given point in the network, the "namespace" is the portion of the total path which is can be used for routing, the "name" portion can be considered opaque.

For example: a broker *A* within an organisation *foo.com* may have a queue *Q*. The full global address of this queue may be *amqp://foo.com/A/Q*. At the broker *A* the namespace portion of the address will be *foo.com/A* and the name will be *Q*. If a message with reply-to "amqp://foo.com/A/Q" is sent through the global AMQP network and arrives at a receiver *R* within the organisation *bar.com*, then R may consider the namespace of the address to be "*foo.com*" and the name to be "*A/Q*". That is *R* can route to "*foo.com*" but has no visibility of how to route directly to addresses within that namespace.

We define two forms of global addresses. The first form uses as the first element of the path a Domain/IP Address/Host (that is an address resolvable to an IP address and port using DNS). The second form uses an identifier that is not resolvable using DNS, but it guaranteed to uniquely identify a routable address within the AMQP network. Details of how the routing information for such addresses is propagated through the AMQP network are not within the scope of this document.

An AMQP container within the namespace N which contains the address X MUST be capable of correctly routing links where the terminus local to the container is expressed in the form "amqp://N/X". Additionally it SHOULD be capable of correctly routing links where the terminus is expressed in the form "X". This if a container in namespace "foo.com/server1" contains an address "queue1", then a client connecting to the container MUST be able to attach a sending link with the target address "amqp://foo.com/server1/queue1" and SHOULD be able to attach a sending link with the target address "queue1" where the messages sent on the link will arrive at the same node.

The mechanism by which a container establishes the namespace within which it resides are out of scope of this document. Also out of scope is the definition of any mechanism by which one peer can discover the namespace in which another AMQP container resides.

If scheme is omitted it must be interpreted as *amqp:* . The scheme used in the address of a source or target in an Attach performative MUST be *amqp:* (either explicitly or implicitly). If the scheme used in an address supplied in the **to** or **reply-to** properties of a message is not amqp: then the behaviour is undefined. Intermediaries should allow reply-to addresses which the intermediary cannot resolve or for which it does not recognise the scheme.

Routing

For addresses of the form amqp://<domain>/path an AMQP container which does not already have routing information sufficient to route to the address can fall back to using DNS SRV records to identify an AMQP entry point for the specified domain. The mechanism for obtaining credentials for connecting to this domain is out of scope.

Node Level Routing

When using the request-response pattern, the requestor may generate an address for a temporary back channel using the second form of the address syntax. In order for responses to be routed correctly, the requestor opens a receiving link to the address of the service providing the address of the back-channel in the target address of the link.

In order for responses to be correctly routed, the service located at the node MUST detect the presence of an outgoing link from the service node with a target address matching that of the reply-to address in the request message.

Addressing/routing scenarios

Scenario 1: Peer to Peer

- DNS Name (of well-known service): service.rh.com
- AMQP Address: amqp://service.rh.com
- DNS Resolve service.rh.com (look up SRV record)
- src is unimportant on request path for “temporary” response queues
- Option 1: Use IP Addr/Port for requestor name
    - Establish request path:
        - attach(target=”amqp://service.rh.com”, source=null)
    - Establish response path:
        - attach(src=”amqp://service.rh.com”, target=amqp://<host/ip:port>/<name unique to host:port>)
    - Message
        - to: ”amqp://service.rh.com”
        - reply-to: amqp://<host/ip:port>/<name unique to host:port>
- Option 2: Use Globally Unique (random) identifier for requestor name
    - Cannot fall back to DNS lookup to initiate a response connection, MUST use matching of address… may not be on same connection

Scenario 2: Service is outside local domain

- attach(target=”amqp://<client namespace>/<name unique for client namespace>”, source=” amqp://rafi.rh.com/ messenger”)
- Examples of client namespace: foo.rh.com ; rh.com/foo

Scenario 3: Traditional Broker

- AMQP Address: amqp://dingham.sb.com/q1$management
- DNS Resolve dingham.sb.com
- attach(target=” amqp://dingham.sb.com/q1$management”, src=null)
- Option 1:
    - Establish request path:
        - attach(target=”amqp://!<Globally Unique Identifier>” , source=” amqp://dingham.sb.com/q1$management”)
    - Establish response path:
        - attach(target=”amqp://<client namespace>/<name unique for client namespace>”, source=” amqp://dingham.sb.com/q1$management”)
- Examples of client namespace: foo.rh.com ; rh.com/foo

Resolving return routing:

First try exact string match for address. Ultimate fall-back is DNS resolution on domain. Deployment specific other resolutions may occur.

Scenario 4: Transparent Intermediaries

- AMQP Address: amqp://!rafi/service
- Out of band uplink into intermediaries
- Establish request path:
  - attach(target=" amqp://!rafi/ service", src=null)
- Establish response path:
  - attach(src=" amqp://!rafi/ service", target="amqp://!dave/responses/rafi")
- Message
  - to: amqp://!rafi/ service
  - reply-to: amqp://!dave/responses/rafi

Management

Notes are captured in the current work-in-progress draft of the management specification document:

amqp-man-v1 0-wd01.doc

Questions

- Batched management operators?
- Text representation?

WebSockets

- Should we transport standard AMQP frames (binary) or invest in a text representation for WebSockets. Argument is that cracking binary in the browser is tricky.
  - Decision: for now, stick with binary frames, maybe revisit later.
- Mapping of AMQP frame to WebSocket frame/message?
  - Decision: map AMQP frame to WebSocket message.
- How to handle SASL. Issue is that in standard usage there are two protocol headers, one for SASL, one for AMQP. How does this map to WebSockets?
  - Decision: define 2 WebSockets subprotocols:
    - Raw AMQP: AMQP0100.amqp.org
    - AMQP over SASL: AMQP3100_AMQP0100.amqp.org
- How would SASL auth interact with browser-based auth?
  - If a Web app is authenticating using SSL client certs, shouldn't need to re-authenticate for AMQP.
  - Similarly, if an app is authenticated using GoogleId then that could be used for AMQP authentication.
- Connection recovery. If the TCP connection between the browser and the load balancer fails then how do we ensure that on re-connection the LB routes to the original broker instance. Does this work automagically due to HTTP session affinity?
- What environments do we want to support?
  - Browsers, Tunnelling