AMQP Addressing Version 1.0

Working Draft 07

03 December 2018

**Technical Committee:**

[OASIS Advanced Message Queuing Protocol (AMQP) TC](https://www.oasis-open.org/committees/amqp/)

**Chairs:**

Rob Godfrey (rgodfrey@redhat.com), [Red Hat](http://www.redhat.com/)

Clemens Vasters (clemensv@microsoft.com), [Microsoft](http://www.microsoft.com/)

**Editors:**

Clemens Vasters (clemensv@microsoft.com), [Microsoft](http://www.microsoft.com/)

**Related work:**

This specification is related to:

* *OASIS Advanced Message Queuing Protocol (AMQP) Version 1.0 Part 0: Overview*. Edited by Robert Godfrey, David Ingham, and Rafael Schloming. 29 October 2012. OASIS Standard. <http://docs.oasis-open.org/amqp/core/v1.0/os/amqp-core-overview-v1.0-os.html>.

**Abstract:**

The AMQP Addressing specification further defines the “AMQP network” concept introduced in the main AMQP specification as a federation of AMQP containers whose nodes communicate with each other either directly or via intermediaries. This specification also defines the semantics of the “address” archetype that was left undefined in the main AMQP specification, and the syntax for the AMQP URI scheme and a matching restriction of the AMQP “address-string” type.

**Status:**

This [Working Draft](https://www.oasis-open.org/policies-guidelines/oasis-defined-terms-2017-05-26%22%20%5Cl%20%22dWorkingDraft) (WD) has been produced by one or more TC Members; it has not yet been voted on by the TC or [approved](https://www.oasis-open.org/policies-guidelines/tc-process%22%20%5Cl%20%22committeeDraft) as a Committee Draft (Committee Specification Draft or a Committee Note Draft). The OASIS document [Approval Process](https://www.oasis-open.org/policies-guidelines/tc-process%22%20%5Cl%20%22standApprovProcess) begins officially with a TC vote to approve a WD as a Committee Draft. A TC may approve a Working Draft, revise it, and re-approve it any number of times as a Committee Draft.

This specification is provided under the [RF on RAND Terms](https://www.oasis-open.org/policies-guidelines/ipr%22%20%5Cl%20%22RF-on-RAND-Mode) Mode of the [OASIS IPR Policy](https://www.oasis-open.org/policies-guidelines/ipr), the mode chosen when the Technical Committee was established. For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the TC’s web page (<https://www.oasis-open.org/committees/amqp/ipr.php>).

Note that any machine-readable content ([Computer Language Definitions](https://www.oasis-open.org/policies-guidelines/tc-process%22%20%5Cl%20%22wpComponentsCompLang)) declared Normative for this Work Product is provided in separate plain text files. In the event of a discrepancy between any such plain text file and display content in the Work Product's prose narrative document(s), the content in the separate plain text file prevails.

**URI patterns:**

(TBD)

(Managed by OASIS TC Administration; please don’t modify.)

Copyright © OASIS Open 2018. All Rights Reserved.

All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the "OASIS IPR Policy"). The full [Policy](https://www.oasis-open.org/policies-guidelines/ipr) may be found at the OASIS website.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to OASIS, except as needed for the purpose of developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

**Table of Contents**

[1](#_30j0zll) Introduction 4

[1.1 IPR Policy 4](#_1fob9te)

[1.2 Terminology 4](#_3znysh7)

[1.3 Normative References 4](#_2et92p0)

[1.4 Non-Normative References 5](#_4d34og8)

[2](#_2s8eyo1) AMQP Networks 6

[2.1 Nodes 6](#_17dp8vu)

[2.2 Containers 7](#_3rdcrjn)

[2.3 Scopes 8](#_26in1rg)

[2.4 Relationship between Scope and Container Identifiers 8](#_lnxbz9)

[3](#_35nkun2) Addressing Elements 10

[3.1 Protocol Schemes 10](#_1ksv4uv)

[3.2 Network Endpoint (Authority) 10](#_2p2csry)

[3.3 Scope Identifier 10](#_44sinio)

[3.4 Path 11](#_2jxsxqh)

[3.5 Parameters 11](#_z337ya)

[4](#_147n2zr) URI Definitions 12

[4.1 AMQP URI Syntax 12](#_3j2qqm3)

[4.2 Examples 12](#_1y810tw)

[5](#_2xcytpi) Security Considerations 14

[6](#_qsh70q) Conformance 15

[Appendix A. Acknowledgments 16](#_3as4poj)

[Appendix B. Revision History 17](#_49x2ik5)

# Introduction

The core AMQP specification [AMQP] introduces the concept of an *AMQP network* as the conceptual foundation for its architectural elements:

*An AMQP network consists of nodes connected via links. Nodes are named entities responsible for the safe storage and/or delivery of messages. Messages can originate from, terminate at, or be relayed by nodes.*

*[…]*

*Nodes exist within a container. Examples of containers are brokers and client applications. Each container MAY hold many nodes. Examples of AMQP nodes are producers, consumers, and queues.*

While the AMQP network concept is referenced several times within the core specification, there is no formal definition of the network model in the core specification since it is primarily focused on defining a peer-to-peer transfer model and protocol.

This specification provides an expanded conceptual framework for AMQP networks and for addressing the elements within them. It also formally defines the schema and syntax of the AMQP Uniform Resource Identifier.

The “AMQP Networks” section provides the conceptual framework, including examples of its application. “Addressing Elements” defines the elements of the addressing model, which includes formal constraints on the use of addressing-related constructs in the core AMQP specification.

The “AMQP URI” section defines the Uniform Resource Identifier syntax.

## IPR Policy

This specification is provided under the [RF on RAND Terms](https://www.oasis-open.org/policies-guidelines/ipr%22%20%5Cl%20%22RF-on-RAND-Mode) Mode of the [OASIS IPR Policy](https://www.oasis-open.org/policies-guidelines/ipr), the mode chosen when the Technical Committee was established. For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the TC’s web page (<https://www.oasis-open.org/committees/amqp/ipr.php>).

## Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [[RFC2119](#tyjcwt)] and [[RFC8174](#3dy6vkm)] when, and only when, they appear in all capitals, as shown here.

When used in this specification and unless explicitly stated otherwise, the term “message” always refers to an AMQP message using the default message format of [[AMQP](#1t3h5sf) 1.0, 3.2.16]

## Normative References

**[****RFC2119]** Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.

**[RFC5646]** Phillips, A., Ed., and M. Davis, Ed., "Tags for Identifying Languages", BCP 47, RFC 5646, DOI 10.17487/RFC5646, September 2009, <<https://www.rfc-editor.org/info/rfc5646>>.

**[****RFC8174]** Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<http://www.rfc-editor.org/info/rfc8174>>.

**[****AMQP 1.0]** *OASIS Advanced Message Queuing Protocol (AMQP) Version 1.0 Part 0: Overview*. Edited by Robert Godfrey, David Ingham, and Rafael Schloming. 29 October 2012. OASIS Standard. <http://docs.oasis-open.org/amqp/core/v1.0/os/amqp-core-overview-v1.0-os.html>

## Non-Normative References

**[CaseFold]** W3C Wiki, “Case Folding” <<https://www.w3.org/International/wiki/Case_folding>>

**[CharMod]** [Character Model for the World Wide Web 1.0: Fundamentals](http://www.w3.org/TR/2005/REC-charmod-20050215/), M. Dürst, F. Yergeau, R. Ishida, M. Wolf, T. Texin, Editors, W3C Recommendation, February 15, 2005, http://www.w3.org/TR/2005/REC-charmod-20050215/. [Latest version](http://www.w3.org/TR/charmod/) available at http://www.w3.org/TR/charmod/.

**Reference sources:**

For references to **IETF RFCs**, use the approved citation formats at:
<http://docs.oasis-open.org/templates/ietf-rfc-list/ietf-rfc-list.html>.

For references to **W3C Recommendations**, use the approved citation formats at:
<http://docs.oasis-open.org/templates/w3c-recommendations-list/w3c-recommendations-list.html>.

# AMQP Networks

The core AMQP specification defines an *AMQP network* to consist of *nodes* connected via *links*, and nodes existing within *containers*. The objective of the AMQP core specification is to define a peer-to-peer protocol for transferring messages between nodes in an AMQP network, and a protocol for establishing communication between nodes residing different in containers.

The AMQP concept of “network” is more abstract than that of the Internet Protocol (IP) and similar networking protocols. An AMQP network has no firm associations with underlying resources in the way that DNS names map to IP addresses and through to network hardware identifiers.

*AMQP’s nodes* and *containers* are application concepts and therefore, resolving address information (or message metadata) to network locations, and routing messages towards those locations are application-level tasks.

As a result, an AMQP network allows an application to define an overlay network routing model that spans one or more underlying transport networks and multiple middleware infrastructures.

Safety and security critical environments often use isolated IP networks that are only reachable through application-layer gateways acting as an “air gap” for IP traffic, because IP-level access to the protected equipment potentially opens up a large attack surface area, and unauthorized access and manipulations may result in safety hazards that could result in destroyed equipment, injury, or death.

The network segmentation and mutual isolation employed in such scenarios makes it intentionally impossible to establish an IP route to services or equipment in the protected networks.

The AMQP network model aims to make it possible for AMQP intermediaries in gateways and devices to securely route application-level messages from (authorized) external parties across the boundaries of isolated IP networks without exposing unsecured or vulnerable network assets, and while enabling the information flow to be inspected by intermediaries and logged in audit trails.

When supported by the container implementations, the origins of routed AMQP traffic can also be masqueraded [AMQP-ROUT-ANN] similar to how Network Address Translation (NAT) performs it for IP, protecting sender privacy and preventing discovery of routing topologies by receivers not privileged to such information. A robot manufacturer might be authorized to see information about a robot, but it’s not necessarily authorized to discover the entire factory routing topology.

The integration of an AMQP network with an IP network is performed using “on-ramp” endpoints. An “on-ramp” is an IP-addressable container that is part of an AMQP network and to which external clients can connect. Those external clients (technically AMQP containers themselves) then establish links and transfer messages to and through the “on-ramp” container.

While AMQP containers are typically interconnected via IP networks, the IP routes between containers do not play a role in AMQP addressing. An AMQP container, like one that’s hosted inside a moving vehicle or ship or aircraft, might quite well be changing IP networks and addresses and therefore inter-container routing, but its AMQP address will remain stable.

## Nodes

Nodes are the sources or targets of messages in an AMQP network. AMQP does not differentiate between clients and servers, but only knows communicating peers. Implementations using AMQP might assign specific roles to nodes, like producer and queue and consumer, but these differences play no role at the AMQP level. A producer sending messages to a queue is just an AMQP node transferring messages to another node.

AMQP *links* exist to establish transfer routes between *nodes,* whereby the nodes on either end may reside in the same container or may be distributed across different containers. A link attaches to a node at a terminus. There are two kinds of terminus: sources and targets. A terminus is responsible for tracking the state of a particular stream of incoming or outgoing messages. Sources track outgoing messages and targets track incoming messages. Messages only travel along a link if they meet the entry criteria at the source.

Links are unidirectional transfer routes, moving messages from the source to the target, and communicating the state of the delivery at the target back to the source. If the container supports it, links may be paired [AMQPLINKPAIR] to form full-duplex, bi-directional routes between two nodes that are also suitable for request-response communication.

The *source* and *target* information that is transferred in the attach performative for establishing links contains two *addresses*, the source address identifies a node on the source container, the target identifies a node on target container.

Out of this follows that AMQP nodes themselves are not named but that termini are named. Nodes are grouping constructs in AMQP and typically reflect distinct architectural entities in the implementing container that are accepting or emitting messages.

An AMQP node MAY have any number of named termini. Termini of different kinds MAY share a name, meaning that the terminus address for *source* and *target* might be the same, but its also possible for them to diverge. Since termini are distinct elements of the architecture, they MAY also differ in terms of supported capabilities and behavior from other termini on the same node.

In the simplest case, a node might just have one named terminus or a pair of *source* and *target* termini that are named the same and therefore the node might appear named. In more complex cases, a node might have one named *target* terminus for accepting messages but multiple named *source* termini for emitting messages, or a node might even have a dynamic internal topology where each entity inside of the node has associated termini.

The naming authority for termini lies with the AMQP container which resolves names to internal objects.

A container implementation might choose to enforce a naming model such that there is at most one *source* and at most one *target* terminus with the same names on any entity represented by an AMQP node, and therefore effectively structure its addressing topology by node names.

Another container implementation might choose to have names for the entities represented by AMQP nodes and with explicit termini names relative to those, and therefore forming a dependency hierarchy. This might manifest in an address expression as a combination of the entity name and the terminus name, with a separator between them; for example, *myentity/feed1* or *myentity/feed2*.

In either case, the container might also allow for further organization of such entities in a structured graph, resulting in terminus address expression (-prefixes) that form paths through that graph; for example, */group-a/group-a-b/myentity*.

Yet another container implementation might choose for termini to have independent names or allow for individual termini to be freely aliased.

## Containers

AMQP containers are naming and management scopes for AMQP nodes. The AMQP protocol does not have explicit client/server roles, but containers fall into one of three categories: a client initiates connections a *server* accepts connections and an *intermediary* does both. In particular a *router* is an intermediaries that routes messages in a network.

A container might be a single process on a single machine, but it might also be a distributed system spanning many processes and machines and use private means to present the container as one towards external parties.

AMQP *connections* and *sessions* exist to establish communication paths for AMQP *links* across an underlying transport network.

The core AMQP specification defines a binding of the AMQP connection layer to the TCP stream transport protocol, optionally with overlaid TLS. The AMQP Web Socket Binding specification defines a binding to the Web Socket protocol, which allows for network endpoint sharing with HTTPS.

If a container is realized as a distributed system, the implementation might use AMQP *connections* and *sessions* to establish communication paths within the scope of the same container, potentially on top of a system-specific stream layer.

More typically, AMQP *connections* and *sessions* serve to interconnect different *containers* so that nodes on either side can communicate across them via *links*. The lifetime of a *link* is independent of the lifetime of its hosting *session*. If a *connection*, including its session(s)*,* gets interrupted, a previously established link MAY be recovered on a new *connection* and *session.*

An AMQP container MAY be concurrently reachable via multiple, different underlying transport network endpoints and using different protocol bindings. Especially, an AMQP container MAY be reachable via different transport network endpoints that are each bound to networks which are otherwise unrelated.

This specification mandates that each container MUST use an AMQP network-unique *container-id* to identify itself during connection establishment in the open performative.

## Scopes

AMQP *nodes* and *containers* are introduced in the core AMQP specification. This specification introduces a further addressing concept: Scopes. A scope is an addressing abstraction at the level of AMQP containers that either describes where a message or link originates from or where a message or link is destined to go. Scopes MAY form hierarchies and the MAY map to one or multiple containers.

A *scope identifier* is a string that follows DNS naming conventions. *Scope expressions* are matching conditions that can be evaluated against scope identifiers for routing or filtering.

*Scope identifiers* are used for routing messages or links across AMQP container boundaries, potentially with one or more containers acting as routing intermediaries. Instead of having to handle container-ids for all potential routing targets, routing intermediaries can evaluate *scope expressions* to determine which container to direct the message to.

A *scope identifier* is a structured name that represents a logical routing target or source. The term *scope* reflects that the addressed realm has further internal structure (nodes).

While *scope identifiers* follow DNS naming conventions, but they do not have to be registered in DNS and they don’t have an associated IP address. They are just names describing logical realms in a system, and the DNS-like structure helps expressing hierarchical relationships.

For example, scope identifiers MAY be any of the following:

* Arbitrary names for use in a private network with it's own naming conventions
* UUID-based names
* Subdomain-names of a registered DNS domain to provide internet-unique naming. DNS scope names do not need to have any IP addresses registered.

While the identifier of a container MUST be unique and stable within an AMQP network, scope identifiers are more flexible. A container MAY have any number of associated scope identifiers. Reversely, a scope identifier MAY be associated with several containers.

## Relationship between Scope and Container Identifiers

Scopes and Containers have a many-to-many relationship. Many containers can belong to one scope, a single container can belong to many scopes.

If a message is sent to an address with a scope then the AMQP network MUST deliver that message to a container in that scope. If an address does not have a scope then it MAY be handled by immediately connected container or MAY be forwarded to any other container.

How the network maps scope names to container names, or determines membership of a scope, is outside the scope of this specification.

In the core AMQP specification, the container identifier (container-id) is used for containers to identify themselves to the communicating peer in the OPEN performative that is exchanged during connection establishment. The container-id is never used to locate the container.

The *scope* *identifier* introduced in this specification is meant to facilitate locating containers, but it does not define or constrain specific methods by which the container is located.

**Non-normative example:**

Consider a lookup table inside of a container acting as a sender that pairs *scope expressions* with AMQP URIs identifying the peer containers handling further routing, for instance:

|  |  |
| --- | --- |
| **Scope expression** | **Target URI** |
| **singapore.southeast-asia.amqp.org** | amqps://sea-2.example.com/ |
| **\*.southeast-asia.amqp.org** | amqps://sea-1.example.com/queue1 |
| **west-europe.amqp.org** | amqps://weu-1.example.com/(europe.amqp.org)/ |
| **\*.amqp.org** | amqp:(world.example.com) |

When the sending container needs to resolve a scope identifier, for example by encountering it in a link target address or inside the *to* property of a message, it consults the lookup table for finding a *scope expression* match, and the lookup yields a target URI if a match is found.

The target URI is then used to determine the next routing activities. Those might involve connecting to the network endpoint indicated in the URI or using some private configuration information for creating such a connection. Scope and path information in the target URI might be used to further direct the message or link flow. The specifics of how that might occur are out of scope for this specification.

Whether the target container is the ultimate message target scope or whether it chooses to route either links or messages further onwards is a private concern of the target container. The sending container therefore does not need to understand how the *container-id* received via the target container’s OPEN performative relates to the *scope identifier*. The only relevance of the *container-id* is that for link recovery, the *container-id* of a newly established recovery connection MUST match that of the prior connection, and the *container-id* might also be handy as a lookup key for established connections.

When the *scope identifier* is absent from a fully qualified address expression or when it is empty, the address targets the “current” container, whereby the “current” container is where the address expression is being evaluated. For instance, in the ATTACH performative, the *target* and *source* addresses are evaluated by the receiving container and are therefore relative to that container unless further qualified.

# Addressing Elements

AMQP addresses are used to establish IP network connections between containers, to establish links between nodes within the same container or across different containers, and to route messages inside or across containers based on their addressing metadata.

An address needs to be able to hold the following information:

1. A protocol scheme to indicate how communication should be initiated when a network connection is required.
2. An “authority” that includes network endpoint information for establishing a connection.
3. A scope identifier that indicates which logical AMQP network scope, and, ultimately, which container the message or link is directed towards.
4. A path expression that maps to a terminus inside the target container.
5. A set of application-defined parameters that allow embedding information in the URI that is required for establishing the desired communication path.

## Protocol Schemes

For the core AMQP transport, the defined schemes are *amqp* for regular AMQP connections with in-band TLS upgrades, and *amqps* for the alternate establishment connection mode that begins with TLS connection (Section 5.2.1) [AMQP].

The default TCP port for the amqp scheme is 5672. The default TCP port for the amqps scheme is 5671.

AMQP WebSocket Binding [AMQPWS] endpoints MUST either be described with the standard *ws* (non-secure) or *wss* (*secure, TLS*) WebSocket schemes.

The default TCP port for the *ws* scheme is 80. The default TCP port for the *wss* scheme is 443.

For WebSockets, the client will subsequently rely on the *amqp* WebSocket subprotocol negotiation for discovering whether the endpoint does indeed support AMQP 1.0.

The *scope* scheme is a network and transport protocol independent scheme for URIs that only carry *scope* and *path* information. While this specification defines the *scope* scheme and its syntax, *scope* URIs and the supporting abstract address model can also be used with overlay networks that are not AMQP based or that use a mix of AMQP and other protocols.

## Network Endpoint

While AMQP’s addressing model is primarily a high-level abstraction for overlay networks, external clients outside of such an overlay network must be able to find an entry point (“on-ramp”) into such an overlay network and the overlay network parties need to be able to locate eachother on the underlying network as well. Therefore, some address expressions need to carry both logical addressing information as well as network endpoint addressing information.

For the current AMQP transport bindings, the network endpoint will generally be an IP network address or an IP address resolvable hostname along with a TCP port number. The network endpoint’s port number is only required if it deviates from the protocol scheme’s default. For uses of AMQP with non-IP transports, the conformance rules spelled out here are equivalently applicable.

The IP network endpoint identifies the “on-ramp” into an AMQP network; it helps an otherwise external party (typically referred to as “client”) to establish a connection to a container that is part of said AMQP network.

Link addresses

Link addresses SHOULD NOT include an authority. Links are made over a connection, which has already been established, so an authority is redundant. If a link address has an authority it is not an error, but it MUST be ignored (this allows AMQP addresses to be copied without checking for an authority)

### Message 'to' field

The 'to' field of a message MAY have an authority. Note the 'to' field is part of the bare message which may be signed, and cannot be modified by routers. A server (the final processor of the message) MUST ignore the IP authority (access via different on-ramps to the same AMQP address is equivalent.)

A router MAY ignore the 'to' field and forware a message within its own network, or MAY connect to the 'to' field address. How routers decide that is out of scope of this specification; they MAY use custom annotations. properties of the link or or connection that received the message, or other mechanisms.

### Message 'reply-to' and the Request Response pattern

The 'reply-to' field MAY have an authority.

Servers and routers SHOULD attempt to deliver a response on the same connection that received the request, so it can be returned or re-routed to the same client that sent the request If this is impossible or fails, the response SHOULD be sent via a connection to the 'reply-to' authority.

### Scope Identifier

As discussed in 2.3 the scope identifier is used to direct messages to the appropriate container.

If the scope identifier is empty, it is interpreted by the “current” container, which is always the one at which the AMQP operation carrying the address information is immediately directed. For instance, for an *attach* or *transfer* operation, the implied reference is determined by the container receiving the performative frame.

An AMQP address consisting of a network endpoint and an empty scope identifier will result in the overall address expression resolved by the container reachable via the network endpoint or its equivalent communication path per Error: Reference source not found.

An AMQP address consisting of a network endpoint and an non-empty scope identifier will result in the overall address expression resolving to a container identified by the scope identifier while the network endpoint or its equivalent communication path per Error: Reference source not found serve as “on-ramp” into the AMQP network if required.

How a scope identifier maps to target containers and how it is determined that a message or link has reached its intended destination is outside the scope of this specification and implementation specific.

## Path

As discussed in 2.1, the path expression is a sequence of identifier segments that reflects a path through an implementation specific relationship graph of AMQP nodes and their termini. The path expression MUST resolve to a terminus in an AMQP container. An empty path expression reflects the anonymous terminus.

How the path expression relates to the graph and terminus is outside the scope of this specification and implementation specific.

## Parameters

Parameters are a set of application-specific key-value pairs carried alongside the other address information. The OASIS AMQP TC reserves parameter name prefixed with “amqp:” for use in its specifications.

Parameters are useful when extra information needs to be given to the party handling the address information.

For instance, when a sending application wants to pass a *reply-to* address to a message consumer and the sender wants to grant the consumer limited access to the reply destination, it might include a parameter that carries a security token granting the required access. amqp://endpoint.example.com/(site-b.contoso.com)/queue

# The AMQP address

An *AMQP address* is a *URI reference* as defined by RFC3986.

This specification defines schemes “amqp” for plain TCP connections and “amqps” for TLS connections.

Implementations MAY support other connnection schemes (for examle ws, wss) and other AMQP specifications MAY introduce other AMQP-specific schemes.

## Transport independent addresses

A *transport independent address* has no network informtaion and cannot be used to connect to an AMQP service but can be used to send messages once connected – as a link source or destination address, or a message “to” or “reply-to” address. This is compatible with AMQP practice prior to this specification. In terms of RFC398 such an address is a *relatifve URI reference*. For example:

* myqueue
* amqp:myqueue
* /area/mailbox
* /(site.example.com)/foo/bar/thing

Note in the last example “site.example.com” is an AMQP scope, not a DNS name.

## AMQP URLs

A URL provides a network endpoint to connect to an AMQP service, and the address of an AMQP node, for example:

* amqps://onramp.example.com/(site.net)/target

An AMQP URL may have no path, for example

* amqp://service.org

Such a URL can be used to establish a connection. If used as the source or target address of a link it refers to the *anonymous terminus* ***XREF***.

## AMQP URI Syntax

The following ABNF notation builds on the defined elements from Appendix A of RFC3986, New syntax is marked “AMQP Specific”, the rest is directly from RFC3986. This syntax can be parsed using a standard URI parser by examining the first element of the URI path to see if it matches the 'path-amqp-scope' syntax below.

 URI = scheme ":" hier-part [ "?" query ] [ "#" fragment ]

 scheme = “amqp” / “amqps” ; AMQP specific

 hier-part = "//" authority path-abempty

 / path-absolute

 / path-rootless

 / path-empty

 / path-amqp-scope ;; AMQP-specific

 URI-reference = URI / relative-ref

 absolute-URI = scheme ":" hier-part [ "?" query ]

 relative-ref = relative-part [ "?" query ] [ "#" fragment ]

 relative-part = "//" authority path-abempty

 / path-absolute

 / path-noscheme

 / path-empty

 / path-amqp-scope ;; AMQP-specific

// AMQP specific

 path-amqp-scope = "/" amqp-scope path-absolute

 amqp-scope = “(“ reg-name “)”

The syntax elements map to the AMQP address elements as follows:

 scheme ::= Protocol Scheme (3.1)

 authority ::= Network Endpoint (3.2)

 amqp-scope ::= Scope (3.3)

 path-\* ::= Path (3.4)

 query ::= Parameters (3.5)

## Examples

Examples of URLs with network endpoint information.

1. AMQP URLs with just a network endpoint. Those URIs identify the container reachable via the network endpoint and the anonymous terminus of that container:
	* amqp://endpoint.example.com
	* amqp://endpoint.example.com:15671
2. AMQP URLs with a network endpoint and a path. Those URIs identify the container reachable via the network endpoint and the node identified by the path:
* amqp://endpoint.example.com/queue
* amqp://endpoint.example.com/area/queue
1. AMQP URLs with just a network endpoint and a parameter. Those URIs identify the container reachable via the network endpoint and the anonymous terminus of that container:
* amqp://endpoint.example.com:15671/?access\_token={token}
1. AMQP URLs with a network endpoint and scope identifier. Those URIs identify a container and its anonymous terminus by the scope identifier and provide an on-ramp endpoint.
* amqp://endpoint.example.com/(site-a.contoso.com)/
1. AMQP URLs with a network endpoint, scope identifier, and path. Those URIs identify a container by the scope identifier and a node inside the container with an on-ramp endpoint.
* amqp://endpoint.example.com/(site-b.contoso.com)/queue
* amqp://endpoint.example.com/(site-c.contoso.com)/area/mailbox

Examples of URIs without network endpoint information. May be used when a connection is already established.

1. AMQP URIs with a scope identifier and path. Those URIs identify a container by the scope identifier and a node inside the container.
* amqp:(site-c.contoso.com)/area/mailbox
* amqp:(site-b.contoso.com)/queue
1. AMQP URIs with just a path. Those URIs identify the current container and a node inside the container.
* amqp:/queue
* amqp:/area/mailbox
* amqp:queue

Examples of URI references without a scheme. May be used where AMQP is implied, for example in AMQP message to/reply-to fields and link source/target fields. May also be used externally where an application knows that AMQP is intended and has some other way to make a connection.

1. AMQP URI references with a scope identifier and path.
* (site-c.contoso.com)/area/mailbox
* (site-b.contoso.com)/queue
1. AMQP URIs with just a path. Those URIs identify the current container and a node inside the container.
* /queue
* /area/mailbox
* queue

# Security Considerations

[TBD]

* Complex expressions may consume significant compute resources
* Malformed messages
* Inducing errors
* Injection

# Conformance

(**Note**: The [OASIS TC Process](https://www.oasis-open.org/policies-guidelines/tc-process%22%20%5Cl%20%22wpComponentsConfClause) requires that a specification approved by the TC at the Committee Specification Public Review Draft, Committee Specification or OASIS Standard level must include a separate section, listing a set of numbered conformance clauses, to which any implementation of the specification must adhere in order to claim conformance to the specification (or any optional portion thereof). This is done by listing the conformance clauses here.

For the definition of "conformance clause," see [OASIS Defined Terms](https://www.oasis-open.org/policies-guidelines/oasis-defined-terms-2017-05-26%22%20%5Cl%20%22dConformanceClause).

See "Guidelines to Writing Conformance Clauses":
<http://docs.oasis-open.org/templates/TCHandbook/ConformanceGuidelines.html>.

Remove this note before submitting for publication.)

1. **Acknowledgments**

(**Note:** A Work Product approved by the TC must include a list of people who participated in the development of the Work Product. This is generally done by collecting the list of names in this appendix. This list shall be initially compiled by the Chair, and any Member of the TC may add or remove their names from the list by request.

Remove this note before submitting for publication.)

The following individuals have participated in the creation of this specification and are gratefully acknowledged:

**Participants:!!br0ken!!**

[Participant Name, Affiliation | Individual Member]

[Participant Name, Affiliation | Individual Member]

1. **Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision** | **Date** | **Editor** | **Changes Made** |
| [Rev number] | [Rev Date] | [Modified By] | [Summary of Changes] |