# Qpid Dispatch Router (QDR) Access Policy

# Table of Contents

# Introduction

Qpid Dispatch Router (QDR) access policy (Policy) is an optional authorization mechanism enforcing

- Per-router total and per-application user connection limits.
- Per-user, per-application resource limits.

## Connection Limits

Policy enforces a total connection limit.

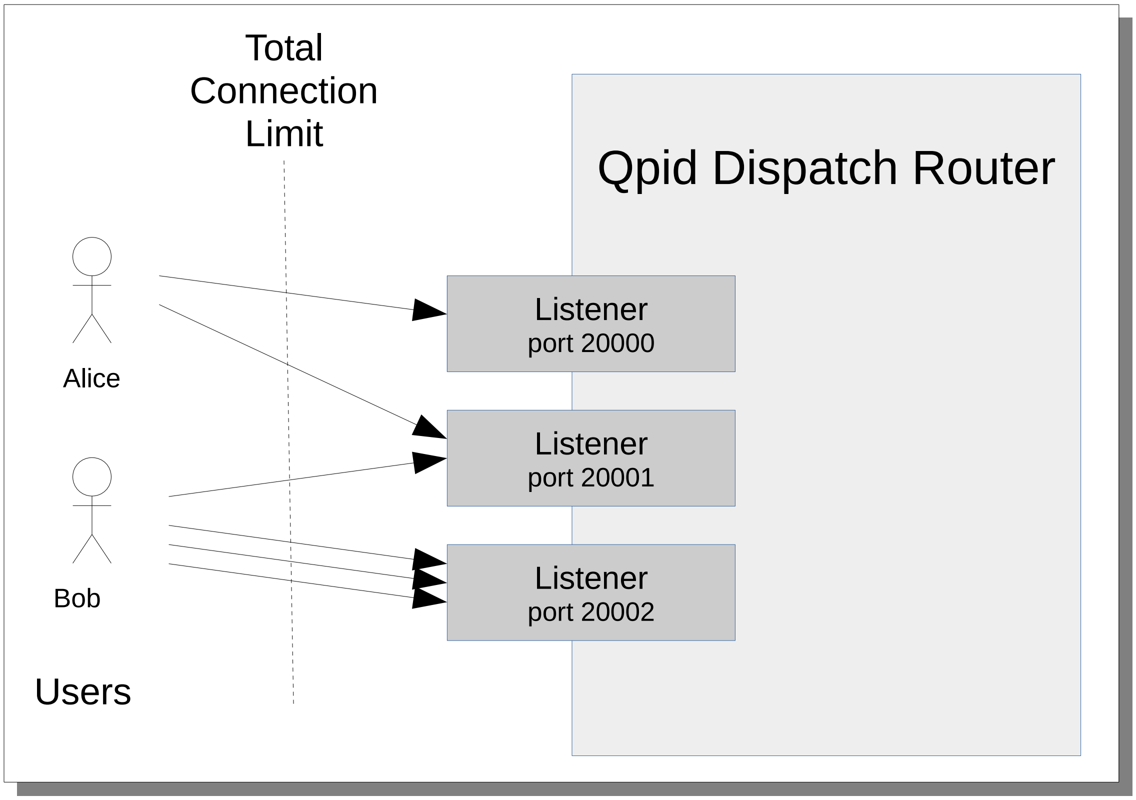

*Figure 1 – Policy Total Connection Limit*

The Total Connection Limit enforces a maximum number of user connections to the QDR service connectors. This limit protects against file descriptor resource exhaustion.

Policy enforces per-application connection limits.

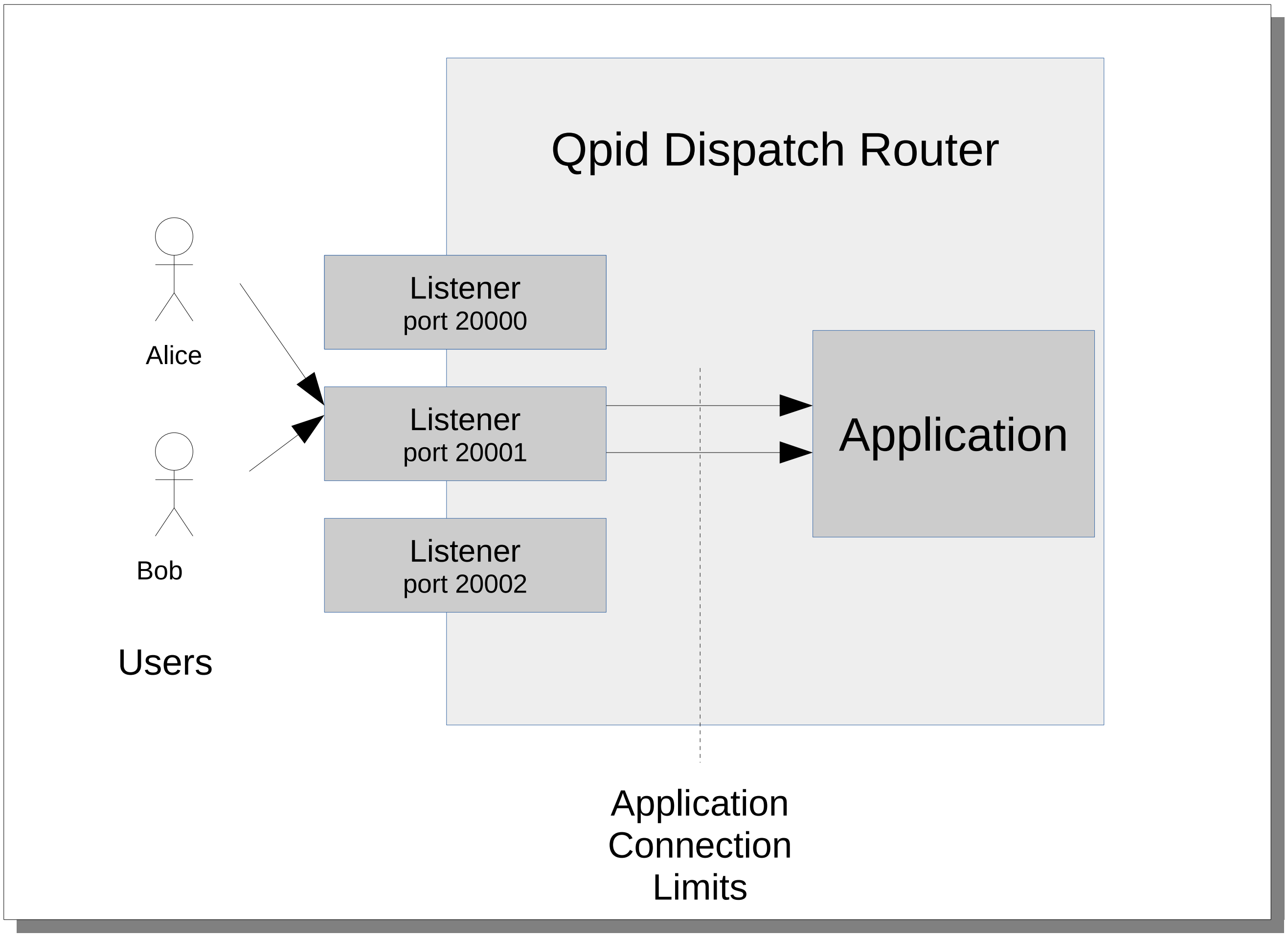

*Figure 2 – Policy Application Connection Limits*

The Application Connection Limits enforce a maximum number of user connections that the QDR will allow to a given application. Several limits may be specified.

| **Application Connection Limit** | **Enforces** |
|---|---|
| maximumConnections | The total number of connections allowed regardless of source. |
| maximumConnectionsPerUser | The total number of connections allowed for a given authenticated  user regardless of which remote host the user is connecting from. |
| maximumConnectionsPerHost | The total number of connections allowed from a remote host regardless of which user is originating the connection. |

*Table 1 – Policy Application Connection Limits*

## Resource Limits

If an authenticated user is allowed to connect to an application then the connection is subject to further restrictions.

| **Application Resource Limit** | **Enforces** |
|---|---|
| maxFrameSize | Largest frame that may be sent on this connection. (AMQP Open, max-frame-size) |
| maxMessageSize | Largest message size supported by links created on this connection. (AMQP Attach, max-message-size) |
| maxSessionWindow | Largest incoming and outgoing window for sessions created on this connection. (AMQP Begin, incoming-window, outgoing-window) |
| maxSessions | Maximum number of sessions that may be created on this connection. (AMQP Open, channel-max) |
| maxSenders | Maximum number of sending links that may be created on this connection. |
| maxReceivers | Maximum number of receiving links that may be created on this connection. |
| allowDynamicSrc | This connection is allowed to use the dynamic link source feature. |
| allowAnonymousSender | This connection is allowed to use the Anonymous Sender feature. |
| sources | List of Source addresses allowed when creating receiving links. |
| targets | List of Target addresses allowed when creating sending links. |

*Table 2 – Policy Application Resource Limits*

# Policy Definition

## Total Connection Limit

The total connection limit is enforced at a low level in QDR when a TCP socket is being opened. The same limit value applies to all user connections from any remote host. If the total connection limit is reached then the socket is simply closed.

Note that the total connection limit applies only to incoming user network connections. The limit does not apply to outbound connections created by QDR nor to inter-router connections.

## Application Resource Limits

Application resource limits are applied later in the AMQP protocol processing starting with the reception of an AMQP Open performative. The following illustration shows an incoming connection.

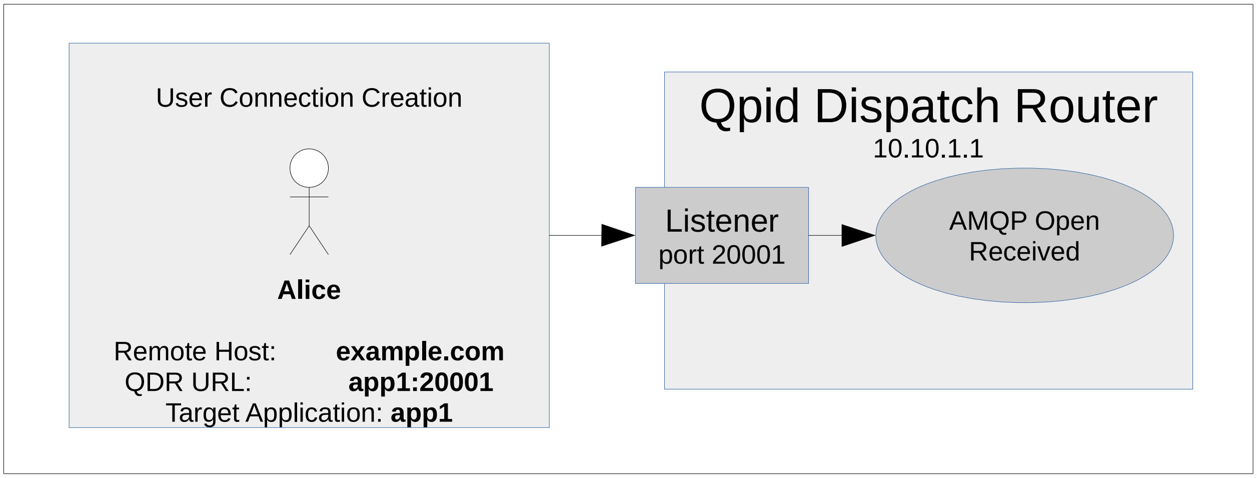

*Figure 3 – User Connection Creation*

In this example remote user ***Alice*** on system example.com (***93.184.216.34***) opens a connection to the Qpid Dispatch Router at URL **app1:20001**. Dns translates the host name 'app1' to 10.10.1.1 to reach the router. The hostname field of the AMQP Open, in this case ***app1***, identifies the application targeted by this connection. When QDR receives the AMQP Open the following information is known:

| **Open Received** | **Value Source** | **Value** |
|---|---|---|
| Remote Host | TCP socket peer | **93.184.216.34** |
| Authenticated User | SASL user name | **Alice** |
| QDR Application | AMQP Open hostname field | **app1** |

Table 3 – AMQP Open Received information known by QDR

QDR defines a policy scoped to the application. In this case there is a rule set for application **app1**. The policy defines:

- User group mappings.
- Ingress host definitions and user group ingress host policy.
- Application connection limits described above.
- User group resource limits described above.

Application Resource Limits are enforced in several stages.

## User Group Assignment

QDR configuration for user group assignment consists of a ***userGroups*** dictionary where the key is the user group name and the value is the list of authenticated user names in that group. For instance:

```
userGroups[“admins”] = “Alice, bob, harry”
userGroups[“users”]  = “user1, user2, user3”
```

When user ***Alice*** connects she is assigned to the ***admins*** user group.

- A user may be in only one group.
- Users in a user group are subject to Host Ingress policy rules.
- Users who are not in any group are assigned to the group ***default***. A configuration boolean ***connectionAllowDefault*** controls whether uses in the default group are allowed to connect. If so then these users are not subjected to Host Ingress policy rules.

## Host Ingress Rules

QDR configuration defines groups of network hosts and which user group users are allowed to connect from those hosts. These are effectively white lists of hosts that limit user connections. The configuration defines two tables to effect ingress checks.

Table ***ingressHostGroups*** is a dictionary where the key is an ingress host group name and the value is a list of IP hosts in that group.

```
ingressHostGroups[“localhost”] = “127.0.0.1, ::1”
ingressHostGroups[“Ten18”]     = “10.18.0.0-10.18.255.255”
ingressHostGroups[“social”]    = “example.com”
```

A user connecting from 10.18.33.145 would be in the ***Ten18*** ingress host group.

- Numeric host IP addresses may be IPv4 or IPv6.
- Host ranges are two numeric addresses of the same family separated by a hyphen. Do not mix IPv4 and IPv6 addresses in a range.

Table ***ingressPolicies*** is a dictionary where the key is a user group name and the value is a list if ingressHostGroups for those users.

```
ingressPolicies[“admins”] = “localhost, Ten18”
ingressPolicies[“users”]  = “social”
```

User ***Alice*** in the user group ***admins*** must connect from a host in the ***localhost*** or the ***Ten18*** ingressHostGroups entry. Similarly user ***user1*** would be assigned to the user group ***users*** and constrained to connecting from a host in the ***social*** ingressHostGroups entry.

- If a user group has no ingress policy then all connections from that group are allowed.

## Application Connection Limits

After a user connection passes the group assignment and ingress policy checks it is subjected to application connection limit checks. If that passes then the connection is counted and allowed to proceed.

## Policy Application Resource Limits

QDR configuration defines resource limits in a ***settings*** dictionary where the key is the user group name and the value is a dictionary of Policy Application Resource Limits shown in Table 2.

```
settings["admin"] = {
  "maxFrameSize":     555555,
  "maxMessageSize":   555555,
  "maxSessionWindow": 555555,
  "maxSessions":           5,
  "maxSenders":           55,
  "maxReceivers":         55,
  "allowDynamicSrc":      true,
  "allowAnonymousSender": true,
  "sources": "public, private, management",
  "targets": "public, private, management"}
```

## Connection Approval

With no Policy in place QDR returns an AMQP Open to the client.

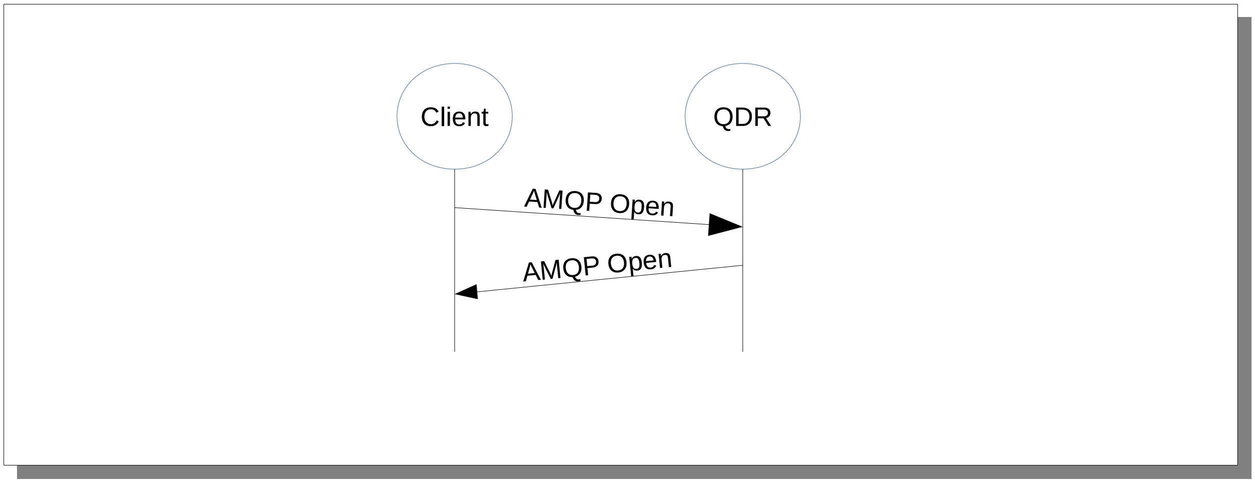

Figure 4 – AMQP Open Forwarding with No Policy Approval

With Policy enforcement QDR intercepts the AMQP Open and queries Policy for approval.

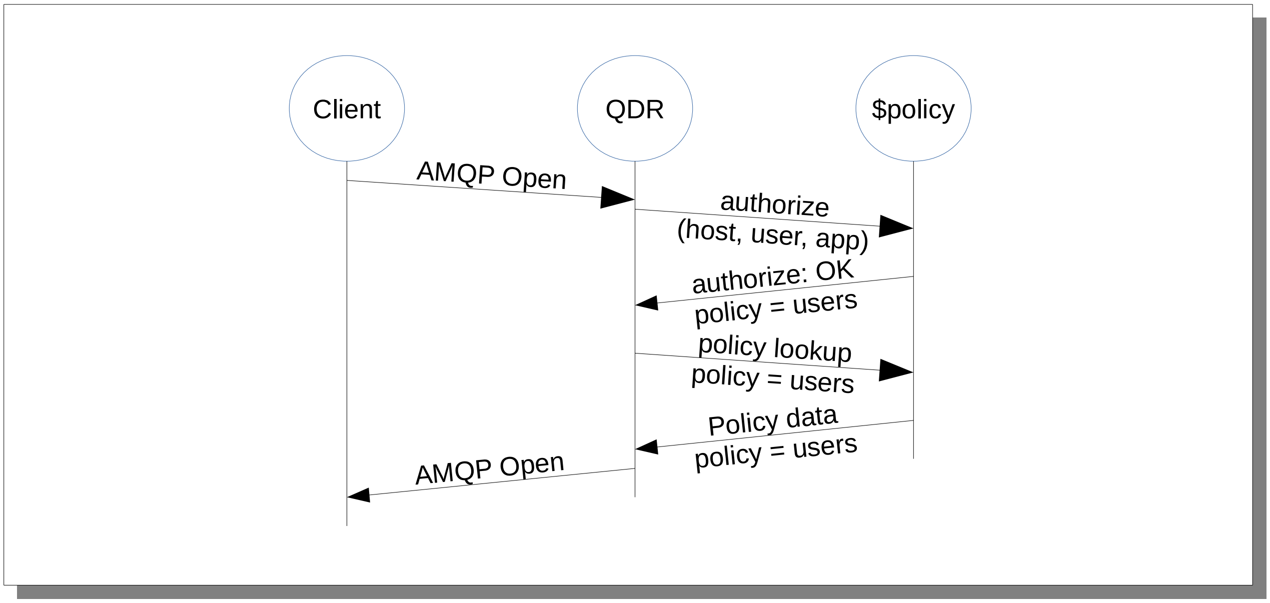

Figure 5 – AMQP Open Forwarding with Policy Approval

The first step is the ***authorize(host, user, app)*** query to approve the user, verify the ingress host, and check application connection counts. The result of this query on approval is the name of the user group. The next step is the ***policy lookup(usergroup)*** where the user group name is presented and the policy engine returns the Resource Limit settings.