Red Hat AMQ 2021.Q1

AMQ Interconnect の使用

AMQ Interconnect 1.10 向け

Last Updated: 2023-05-17

Red Hat AMQ 2021.Q1 AMQ Interconnect の使用

AMQ Interconnect 1.10 向け

法律上の通知

概要

本ガイドでは、AMQ Interconnect をインストール、設定、および管理して、大規模なメッセージングネットワークを構築する方法を説明します。

目次

パート I. 概要

第1章 AMQ INTERCONNECT の概要

AMQ Interconnect は、スケーラブルで利用可能なメッセージングネットワークを構築する軽量 AMQP メッセージルーターです。

1.1. 主な特長

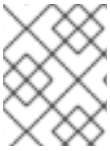
AMQ Interconnect を使用して、クライアント、サーバー、メッセージブローカーなど、AMQP が有効なエンドポイント間でメッセージを柔軟にルーティングできます。AMQ Interconnect には以下の利点があります。

- クライアントおよびメッセージブローカーを、統一アドレス指定でインターネットメッセージングネットワークに接続します。
- 高パフォーマンスの直接メッセージングをサポートします。
- 障害に関する冗長なネットワークパスを使用したルーティング
- 大規模なデプロイメントの管理の簡素化

1.2. サポートされる標準およびプロトコル

AMQ Interconnect は、以下の業界標準およびネットワークプロトコルをサポートします。

- Advanced Message Queueing Protocol (AMQP) のバージョン 1.0
- IPv6 での最新の TCP



注記

AMQP 内の分散トランザクション (XA) の詳細は、仕様の 1.0 バージョンでは提供されません。AMQ Interconnect は XA トランザクションをサポートしません。

関連情報

- OASIS AMQP 1.0 仕様。

1.3. サポートされる設定

AMQ Interconnect は、Red Hat Enterprise Linux 7 および 8 でサポートされます。詳細は、Red Hat AMQ 7 でサポートされる設定 を参照してください。

1.4. 本書の表記慣例

本書では、root 権限を必要とするすべてのコマンドに対して **sudo** が使用されています。何らかの変更がシステム全体に影響を与える可能性があるため、**sudo** を使用する場合は、常に注意が必要です。

sudo の使用の詳細は、**sudo** コマンド を参照してください。

パート II. 詳細情報

第2章 重要な用語と概念

AMQ Interconnect を使用する前に、AMQP に精通し、AMQ Interconnect に関する主要な概念を理解しておく必要があります。

2.1. AMQP の概要

AMQ Interconnect は、Advanced Message Queueing Protocol (AMQP) 仕様のバージョン 1.0 を実装します。そのため、AMQ Interconnect をデプロイまたは設定する前に、複数の主要な AMQP 用語および概念を理解する必要があります。

コンテナー

AMQP は、**コンテナー**と呼ばれるアプリケーション間でメッセージを転送するためのワイヤレベルのメッセージングプロトコルです。AMQP では、コンテナーは、クライアントアプリケーションやメッセージブローカーなどのメッセージを送受信するアプリケーションです。
コンテナーは、**通信**用のチャネルである接続を介して相互に接続します。

ノード

コンテナーには、メッセージの保存や配信を行う**ノード**と呼ばれるアドレス可能なエンティティーが含まれます。たとえば、メッセージブローカーのキューはノードです。

リンク

メッセージは、**リンク**を介して接続されたコンテナー間で転送されます。リンクはノード間の一方向ルートです。基本的には、リンクはメッセージの送受信チャネルです。
リンクは**セッション**で確立され、メッセージの送受信のコンテキストです。セッションは接続上で確立されます。

関連情報

- OASIS AMQP 1.0 仕様
- AMQP Essentials Refcard
- Video series introducing AMQP 1.0

2.2. ルーターとは

AMQ Interconnect は、通常のユーザープログラムまたはデーモンとして実行されるアプリケーションレイヤープログラムです。AMQ Interconnect の実行中のインスタンスは**ルーター**と呼ばれます。

ルーターがメッセージに関して責任を行ならない

ルーターはプロデューサーとコンシューマー間でメッセージを転送しますが、メッセージブローカーとは異なり、メッセージの処理は取りません。代わりに、ルーターは、配信の保証を満たすように、メッセージセットを伝播し、ネットワーク全体でメッセージを伝播します。つまり、ルーターネットワークは複数の中間ルーターによってメッセージを提供し、コンシューマーが同じパス全体でそのメッセージの確認応答をルーティングします。メッセージの責任は、直接接続されているかのようにプロデューサーからコンシューマーに転送されます。

ルーターは、ルーターネットワークを形成するために組み合わされます。

多くの場合、ルーターはルーターネットワークと呼ばれる複数のルーターのトポロジーにデプロイされます。ルーターは、Open Shortest Path First (OSPF) と Intermediate System (IS-IS) プロトコルに類似したリンク状態ルーティングプロトコルおよびアルゴリズムを使用して、すべてのメッ

セージソースから最適なパスを計算し、障害から迅速に復元します。ルーターネットワークは、冗長なネットワークパスに依存して、システムやネットワークに障害が発生した場合の接続を継続できます。

ルーターは直接メッセージングパターンと間接メッセージングパターンの両方を強化

メッセージングクライアントは、単一の AMQP 接続をルーターネットワークに接続することができ、その接続でメッセージをネットワーク上の任意のルーターに接続する 1 つ以上のメッセージブローカーと交換できます。同時に、クライアントはブローカーに関連することなく、他のエンドポイントとメッセージを直接交換できます。

例2.1 メッセージブローカーの使用の改良

ルーターは、スケーラブルで分散された分散作業キューを提供するメッセージブローカーのクラスターを強化できます。

ルーターネットワークは、プロデューサーを 1 つのアドレスに公開し、コンシューマーが単一のアドレスにサブスクライブすることで、ブローカークラスターを単一のキューとして表示できます。ルーターネットワークはクラスター内の任意のブローカーにワークを分散し、任意のコンシューマーに対して任意のブローカーから作業を収集できます。

ルーターはクライアントに影響を与えずにブローカーをクラスターから追加または削除できるため、ブローカークラスターのスケーラビリティーを向上させます。

また、ルーターは "stuck messages" がよく難しいことが難しくなります。ルーターネットワークがない場合、コンシューマーがメッセージを持たないブローカーに接続されている場合 (ただしクラスターの他のブローカーにメッセージがある場合)、メッセージを転送するか、stuck のままにする必要があります。ルーターは、すべてのコンシューマーがルーターネットワーク経由ですべてのブローカーに接続されているため、この問題を解決します。すべてのブローカーのメッセージをコンシューマーに配信できます。

2.3. ルータールートメッセージの使用方法

ルーターネットワークでは、**ルーティング**とは、メッセージが宛先に配信されるプロセスです。そのために、AMQ Interconnect は 2 つの異なるルーティングメカニズムを提供します。

メッセージルーティング

メッセージルーティングにより、メッセージを任意のキャストおよびマルチキャストパターンで分散できます。これらのパターンは直接ルーティングの両方に使用できます。この場合、ルーターはメッセージブローカーと間接ルーティングなしでメッセージを分散します。これにより、ルーターはメッセージブローカーを介してメッセージを交換できます。
メッセージルーティングは以下の要件に役立ちます。

- デフォルトの基本メッセージルーティング
  AMQ Interconnect はデフォルトで自動的にメッセージをルーティングするため、デフォルトとは異なる動作をルーティングする必要がある場合に限り、手動設定が必要になります。
- メッセージベースのルーティングパターン
  メッセージルーティングは、何らかのキャストおよびマルチキャストルーティングパターンの両方をサポートします。個々のメッセージを複数のコンシューマーに分散し、マルチキャスト (または fan-out) メッセージを複数のサブスクライバーに負荷分散できます。
- メッセージ配信順序が重要でない場合は、複数のメッセージブローカーにメッセージのシャーディングをシャーディングします。

あるプロデューサーからメッセージをシャーディングすると、プロデューサーのメッセージが送信順に異なる順序で受信される可能性があります。

リンクルーティング

リンクルーティングにより、送信元とルーターネットワークを通過する受信側の間で専用の仮想パスを確立できます。リンクルートは通常、直接接続が不完全であるシナリオのメッセージブローカーにクライアントを接続するために使用されます。したがって、リンクルートは、以下のようなメッセージルーティングでできないメッセージング機能を有効にします。

- トランザクションメッセージング
  リンクルーティングは、1つのブローカーへのローカルトランザクションをサポートします。分散トランザクションはサポートされません。
- 確実な、メッセージ配信順序
  シャードキューへのリンクルーティングは、リンク上のすべてのメッセージが同じブローカーインスタンスに移動することで、プロデューサーのメッセージの配信順序を保持します。
- エンドツーエンドフロー制御
  フロー制御は、受信側から送信者へのリンクルート全体のクレジットフローを real します。
- サーバー側のセレクター
  リンクルートを使用すると、コンシューマーはブローカーのサブスクリプションのサーバー側のセレクターを提供できます。
- コンシューマー固有の確認応答
  リンクルートでは、変更された配信状態はブローカーによって解釈できます。たとえば、ブローカーは **undeliverable-here=true** が変更された配信状態で、あらゆるメッセージメッセージ再配信を防ぐことができます。

関連情報

- 「メッセージルーティングの設定」
- 「リンクルートの作成」

2.4. ルーターセキュリティー

AMQ Interconnect は認証および承認メカニズムを提供し、ルーターネットワークへのアクセスユーザーとメッセージングリソースで実行できることを制御できます。

認証

AMQ Interconnect は、リモートピアの暗号化および認証を行うために SSL/TLS と SASL の両方をサポートします。これらのメカニズムを使用すると、以下の方法でルーターネットワークのセキュリティーを保護することができます。

- リモートピア (クライアントやメッセージブローカーなど) からの受信接続を認証します。
- リモートピア (クライアントやメッセージブローカーなど) への送信接続に認証クレデンシャルを提供します。
- ルーターネットワーク内のルーター間の相互接続を保護します。

承認

AMQ Interconnect は、ユーザー接続の制限や AMQP リソースアクセス制御を有効にするために使用できる **policy** メカニズムを提供します。

関連情報

- 9章 *ネットワーク接続のセキュリティー保護*
- 10章 *認証の設定*

2.5. ルーター管理

AMQ Interconnect は、ルーターネットワークを監視および管理するためのグラフィカルおよび CLI ツールの両方を提供します。

Red Hat AMQ Interconnect Console

ルーターネットワークのレイアウトと正常性を監視するための Web コンソール。

qdstat

ルーターネットワーク内のルーターのステータスを監視するためのコマンドラインツールです。このツールを使用して、ルーターについての以下の情報を表示できます。

- 着信および発信接続
- 送受信のリンク
- このルーターに関連するルーターネットワークトポロジー
- このルーターが認識するアドレス
- ルートと自動リンクのリンク
- メモリー消費情報

qdmanage

ランタイム時にルーターの設定を表示および更新するためのコマンドラインツールです。

関連情報

- 管理

パート III. 使ってみる

第3章 スタートガイド

本セクションでは、AMQ Interconnect のインストール方法、デフォルト設定でルーターを起動し、2 つのクライアント間でメッセージを分散する方法を示す簡単な AMQ Interconnect の概要について説明します。

3.1. RED HAT ENTERPRISE LINUX での AMQ INTERCONNECT のインストール

AMQ Interconnect は RPM パッケージのセットとして配布されており、お使いの Red Hat サブスクリプションから入手できます。

手順

1. サブスクリプションがアクティベートされ、システムが登録されていることを確認します。カスタマーポータルを使用して Red Hat サブスクリプションをアクティブ化し、システムを登録する方法は、付録A *サブスクリプションの使用* を参照してください。

2. 必要なリポジトリーにサブスクライブします。

   Red Hat Enterprise Linux 7

   ```
   $ sudo subscription-manager repos --enable=amq-interconnect-1-for-rhel-7-server-rpms --
   enable=amq-clients-2-for-rhel-7-server-rpms
   ```

   Red Hat Enterprise Linux 8

   ```
   $ sudo subscription-manager repos --enable=amq-interconnect-1-for-rhel-8-x86_64-rpms --
   enable=amq-clients-2-for-rhel-8-x86_64-rpms
   ```

3. **yum** または **dnf** コマンドを使用して、**qpid-dispatch-router**、**qpid-dispatch-tools**、および **qpid-dispatch-console** パッケージとその依存関係をインストールします。

   ```
   $ sudo yum install qpid-dispatch-router qpid-dispatch-tools qpid-dispatch-console
   ```

4. **which** コマンドを使用して、**qdrouterd** 実行可能ファイルが存在することを確認します。

   ```
   $ which qdrouterd
   /usr/sbin/qdrouterd
   ```

   Theqdrouterd 実行ファイルは **/usr/sbin/qdrouterd** に配置する必要があります。

3.2. デフォルトのルーター設定ファイルの使用

ルーターの設定ファイル (**qdrouterd.conf**) は、ルーターの機能を制御します。デフォルトの設定ファイルには、ルーターの実行に必要な最低限の設定が含まれます。ルーターに慣れているように、これらの設定を追加または変更することや、独自の設定ファイルを作成することができます。

デフォルトでは、ルーター設定ファイルはルーターの以下の設定を定義します。

- 操作モード

- 受信接続をリッスンする方法
- メッセージルーティングメカニズムのルーティングパターン

手順

1. **/etc/qpid-dispatch/qdrouterd.conf** ファイルを開きます。
   AMQ Interconnect がインストールされると、**qdrouterd.conf** がこのディレクトリーにインストールされます。ルーターが起動すると、このファイルは定義した設定で実行されます。

2. **qdrouterd.conf** のデフォルト設定を確認します。

   デフォルトの設定ファイル

```
router {
    mode: standalone (1)
    id: Router.A (2)
}

listener { (3)
    host: 0.0.0.0
    port: amqp
    authenticatePeer: no
}

address { (4)
    prefix: closest
    distribution: closest
}

address {
    prefix: multicast
    distribution: multicast
}

address {
    prefix: unicast
    distribution: closest
}

address {
    prefix: exclusive
    distribution: closest
}

address {
    prefix: broadcast
    distribution: multicast
}
```

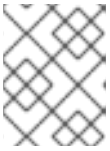
(1) デフォルトでは、ルーターは**スタンドアロン**モードで動作します。これは、直接接続されているエンドポイントとのみ通信できることを意味します。これは他のルーターに接続できないか、またはルーターネットワークに参加できません。

(2)

ルーターの一意識別子。この ID は AMQP プロトコルレベルで **container-id** (コンテナー名) として使用されます。これが指定されていない場合、ルーターは起動時にランダムな

3 **listener** エンティティーは、クライアントエンドポイントからの受信接続を処理します。デフォルトでは、ルーターはデフォルトの AMQP ポート (5672) 上のすべてのネットワークインターフェイスでリッスンします。

4 デフォルトでは、ルーターはメッセージルーティングのメカニズムを使用するように設定されます。各 **address** エンティティーは 、特定のアドレス **prefix** を持つメッセージを分散する方法を定義します。たとえば、**closest** で始まるアドレスを持つすべてのメッセージは、**closest** ディストリビューションパターンを使用して分散されます。



注記

クライアントがルーターの設定ファイルに定義されていないアドレスでメッセージを要求すると、**balanced** 分散パターンが自動的に使用されます。

関連情報

- ルーター設定ファイル (利用可能なエンティティーおよび属性を含む) の詳細は、qdrouterd の man ページ を参照してください。

3.3. ルーターの起動

AMQ Interconnect のインストール後に、**qdrouterd** コマンドを使用してルーターを起動します。

手順

1. ルーターを起動します。

```
$ qdrouterd
```

ルーターは、**/etc/qpid-dispatch/qdrouterd.conf** に保存されているデフォルトの設定ファイルを使用して起動します。

2. **qdrouterd** コマンドの出力をチェックして、ルーターのステータスを確認します。
以下の例では、ルーターが正しくインストールされ、実行中であり、クライアント間のトラフィックをルーティングする準備が整っていることを示しています。

```
$ qdrouterd
Fri May 20 09:38:03 2017 SERVER (info) Container Name: Router.A
Fri May 20 09:38:03 2017 ROUTER (info) Router started in Standalone mode
Fri May 20 09:38:03 2017 ROUTER (info) Router Core thread running. 0/Router.A
Fri May 20 09:38:03 2017 ROUTER (info) In-process subscription M/$management
Fri May 20 09:38:03 2017 AGENT (info) Activating management agent on
$_management_internal
Fri May 20 09:38:03 2017 ROUTER (info) In-process subscription L/$management
Fri May 20 09:38:03 2017 ROUTER (info) In-process subscription L/$_management_internal
Fri May 20 09:38:03 2017 DISPLAYNAME (info) Activating DisplayNameService on
$displayname
Fri May 20 09:38:03 2017 ROUTER (info) In-process subscription L/$displayname
Fri May 20 09:38:03 2017 CONN_MGR (info) Configured Listener: 0.0.0.0:amqp proto=any
role=normal
Fri May 20 09:38:03 2017 POLICY (info) Policy configured maximumConnections: 0,
```

```
policyFolder: '', access rules enabled: 'false'
Fri May 20 09:38:03 2017 POLICY (info) Policy fallback defaultApplication is disabled
Fri May 20 09:38:03 2017 SERVER (info) Operational, 4 Threads Running
```

関連情報

- qdrouterd man ページ

3.4. テストメッセージの送信

ルーターの起動後に、いくつかのテストメッセージを送信し、それらの間でメッセージを分散することで 2 つのエンドポイントを接続する方法を確認します。

以下の手順では、送信者と受信者という 2 つのクライアントで単一のルーターで設定される単純な設定を示しています。受信側は特定のアドレスでメッセージを受信する必要があり、送信側はこのアドレスにメッセージを送信します。

この手順ではブローカーは使用されないので、途中に**ストアと転送**メカニズムはありません。代わりに、受信側がオンラインの場合にのみ送信側からのメッセージフローがルーターを通過し、受信側が送信側に到達したことを確認することができます。

前提条件

AMQ Python がインストールされている必要があります。詳細は、AMQ Python クライアントの使用を参照してください。

手順

1. AMQ Python サンプルディレクトリーに移動します。

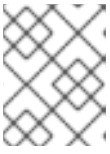
```
$ cd <install-dir>/examples/python/
```

<install-dir>
AMQ Python をインストールしたディレクトリー。

2. **simple_recv.py** レシーバークライアントを起動します。

```
$ python simple_recv.py -a 127.0.0.1:5672/examples -m 5
```

このコマンドは受信側を起動し、**example** アドレス (**127.0.0.1:5672/examples**) をリッスンします。また、受信側は最大 5 つのメッセージを受信するように設定されます。



注記

実際には、送信者および受信側を開始する順序は問題ありません。いずれの場合も、受信側がオンラインになるとすぐにメッセージが送信されます。

3. 新しいターミナルウィンドウで、Python のサンプルディレクトリーに移動して、**simple_send.py** のサンプルを実行します。

```
$ cd <install-dir>/examples/python/
$ python simple_send.py -a 127.0.0.1:5672/examples -m 5
```

このコマンドは、5 つの自動生成メッセージを **example** アドレス (**127.0.0.1:5672/examples**) に送信し、それらが配信され、受信側によって確認されたことを確認します。

```
all messages confirmed
```

4. レシーバークライアントがメッセージを受信していることを確認します。
レシーバークライアントは 5 つのメッセージの内容を表示します。

```
{u'sequence': 1L}
{u'sequence': 2L}
{u'sequence': 3L}
{u'sequence': 4L}
{u'sequence': 5L}
```

3.5. 次のステップ

AMQ Interconnect を使用して 2 つのクライアント間でメッセージを配信した後、以下のセクションを使用して AMQ Interconnect の設定、デプロイメント、および管理について詳しく説明します。

ルーター設定の変更

AMQ Interconnect には、多くの基本的なユースケースに適したデフォルト設定が含まれています。ルーターの基本プロパティー、ネットワーク接続、セキュリティー設定、ロギング、ルーティングのメカニズムを変更することで、以下の **スタート** で使用するスタンドアロンルーターをさらに試すことができます。

AMQ Interconnect のインストールおよび設定

通常、AMQ Interconnect はルーターネットワークにデプロイされます。任意のトポロジーのルーターネットワークを設計して、メッセージングネットワークでエンドポイントを相互接続することができます。

AMQ Interconnect の監視および管理

Web コンソールおよびコマンドラインツールを使用して、ルーターネットワーク内のルーターのステータスとパフォーマンスを監視できます。

パート IV. インストール

第4章 AMQ INTERCONNECT デプロイメントガイドライン

ルーターネットワークを計画し、ネットワークトポロジーを設計するには、まずさまざまなルーターモードと、それらを使用して異なるネットワーク種別の作成方法を理解する必要があります。

4.1. ルーター操作モード

AMQ Interconnect では、各ルーターは **standalone**、**interior**、または **edge** モードで動作します。ルーターネットワークでは、複数の内部ルーターまたは相互運用ルーターとエッジルーターの組み合わせをデプロイして、必要なネットワークトポロジーを作成します。

Standalone

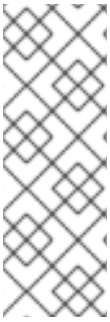
ルーターは単一のスタンドアロンネットワークノードとして機能します。スタンドアロンルーターは、ルーターネットワークでは使用できません。これは他のルーターとの接続を確立せず、直接接続されたエンドポイント間のメッセージのみをルーティングしません。

interior

ルーターはルーターネットワークのインテリアの一部です。内部ルーターは相互に接続を確立し、ネットワーク全体で最小コストのパスを自動的に算出します。

Edge

ルーターは、1つ以上の相互 io ルーターへの単一のアップリンク接続を維持します。エッジルーターはルーティングプロトコルまたはルート計算に参加しませんが、ルーティングネットワークを効率的にスケーリングできます。



注記

ルーターネットワークのパフォーマンスは、さまざまな要素により決定されます。

- トポロジー
- ルーターの数
- 基礎となるインフラストラクチャー (ホストリソース、ネットワーク速度など)

4.2. セキュリティーガイドライン

ルーターネットワークでは、相互のルーターが強固な認証メカニズムを使用して保護する必要があります。ルーターネットワークを作成する前に、この認証メカニズムを選択してプランニングする必要があります。

警告

内部ルーターが適切に保護されていない場合、未承認のルーター (またはルーターに先行しているエンドポイント) は、ルーターネットワークに加わる可能性があるため、その整合性と可用性が侵害される可能性があります。

要件に合わせて最も適したセキュリティーメカニズムを選択できます。ただし、以下の推奨事項を検討する必要があります。

- ルーターネットワークの内部部分を参照するために X.509 認証局 (CA) を作成します。
- それぞれの内部ルーター用に個別の証明書を生成します。
  それぞれの相互オリオーラールーターは、他の内部ルーターからの接続を認証できるように設定できます。

注記

エッジルーターおよびクライアントからの接続では、要件に応じて異なるレベルのセキュリティーを使用することができます。

これらの推奨事項を使用することで、CA の所有者が新規ルーターの新しい証明書を発行するまで、新しいインディオータールーターがネットワークを結合できません。さらに、ネットワークの CA が発行する有効な X.509 証明書がないため、インディオーターのインターフェーフィングングは、ルーターを偽装させることはできません。

4.3. ルーター接続ガイドライン

ルーターネットワークを作成する前に、ルーターが互いに接続する方法、およびルーター間接続を確立する方向の方向に影響を及ぼす要素を理解しておく必要があります。

ルーター間接続は双方向です
ルーター間で接続を確立する場合、メッセージトラフィックフローは、その接続に対する両方の方向に送ります。各コネクションには、接続の確立の目的でクライアント側 (**コネクター**) とサーバー側 (**リスナー**) があります。接続が確立されると、双方向接続で 2 つのサイドの参加者が同等になります。
AMQP トラフィックをネットワーク上でルーティングする目的では、接続の確立の方向は関係ありません。

接続を確立の方向に影響を及ぼす要因
ルーター間の接続を確立する場合、どのルーターを listener にするかを選択し、connector となるルーターを選択する必要があります。どのルーターのペアの間には 1 つの接続が必要です。

ネットワークトポロジーでルーター間接続の方向を判断する際に、以下の要素を考慮してください。

IP ネットワーク境界およびファイアウォール

通常、ルーター間接続は常によりプライベートからよりパブリックに確立される必要があります。たとえば、プライベート IP ネットワークのルーターをパブリックの場所 (パブリッククラウドプロバイダーなど) の別のルーターに接続するには、プライベートネットワークのルーターにコネクターとパブリックの場所のルーターが必要です。これは、VPN を使用したり、パブリックからプライベートアクセスを許可するように設計されたその他のファイアウォール機能を使用せずに、TCP/IP がプライベートの場所に到達できないためです。

ネットワークトポロジー

ルーターネットワークのトポロジーは、ルーター間で接続を確立する必要がある方向に影響を及ぼす可能性があります。たとえば、1 つまたは 2 つの中心の hub ルーターに接続された複数のルーターがあるスタートポロジーでは、 ハブ上にリスナー、スポーク上にコネクターが必要です。これにより、ハブの設定を変更せずに新規のスポークルーターを追加することができます。

第5章 AMQ INTERCONNECT のインストール

AMQ Interconnect は、単一のスタンドアロンルーターとして、またはルーターネットワークで接続された複数のルーターとしてデプロイできます。ルーターネットワークは任意のトポロジーを表す可能性があり、要件に合わせてネットワークを設計できます。

AMQ Interconnect では、ルーターネットワークトポロジーはメッセージルーティングから独立しています。つまり、メッセージングクライアントは基盤のネットワークトポロジーに関係なく、常に同じメッセージルーティング動作が発生することを意味します。マルチサイトまたはハイブリッドのクラウドルーターネットワークでも、接続されたエンドポイントは、1 つの論理ルーターに接続されているかのように動作します。

ルーターネットワークトポロジーを作成するには、以下の手順を実施します。

1. デプロイメントガイドラインを確認します。
   トポロジーにデプロイするさまざまなルーター操作モードを理解し、ルーターネットワークのインディオーター部分のセキュリティー要件に留意してください。

2. ホストに AMQ Interconnect をインストールします。
   複数のルーターでルーターネットワークを作成する場合は、各ホストでこの手順を繰り返してください。

3. ルーター設定を準備します。
   AMQ Interconnect のインストール後に、他のルーターおよびエンドポイントへの接続方法や、その動作を定義します。

4. ルーターを起動します。
   ルーターの設定後に、ルーターを起動して相互に接続し、メッセージのルーティングを開始できるようにします。

5.1. RED HAT ENTERPRISE LINUX での AMQ INTERCONNECT のインストール

AMQ Interconnect は RPM パッケージのセットとして配布されており、お使いの Red Hat サブスクリプションから入手できます。

手順

1. サブスクリプションがアクティベートされ、システムが登録されていることを確認します。
   カスタマーポータルを使用して Red Hat サブスクリプションをアクティブ化し、システムを登録する方法は、付録A *サブスクリプションの使用* を参照してください。

2. 必要なリポジトリーにサブスクライブします。

   Red Hat Enterprise Linux 7

   ```
   $ sudo subscription-manager repos --enable=amq-interconnect-1-for-rhel-7-server-rpms --enable=amq-clients-2-for-rhel-7-server-rpms
   ```

   Red Hat Enterprise Linux 8

   ```
   $ sudo subscription-manager repos --enable=amq-interconnect-1-for-rhel-8-x86_64-rpms --enable=amq-clients-2-for-rhel-8-x86_64-rpms
   ```

3. **yum** または **dnf** コマンドを使用して、**qpid-dispatch-router**、**qpid-dispatch-tools**、および **qpid-dispatch-console** パッケージとその依存関係をインストールします。

```
$ sudo yum install qpid-dispatch-router qpid-dispatch-tools qpid-dispatch-console
```

4. **which** コマンドを使用して、**qdrouterd** 実行可能ファイルが存在することを確認します。

```
$ which qdrouterd
/usr/sbin/qdrouterd
```

Theqdrouterd 実行ファイルは **/usr/sbin/qdrouterd** に配置する必要があります。

5.2. ルーター設定の準備

AMQ Interconnect のインストール後に、他のルーターおよびエンドポイントへの接続方法や、その動作を定義します。ルーターネットワークを作成する場合には、ネットワーク内の各ルーターに対してこのワークフローを完了します。

前提条件

- AMQ Interconnect がホストにインストールされている。

手順

1. 必須のルータープロパティーを設定します。
   ルーターネットワークに参加するには、ルーターは一意の ID と操作モードで設定する必要があります。

2. ネットワーク接続を設定します。

   a. ルーターをルーターネットワークの他のルーターに接続します。
   このルーターに接続する追加のルーターごとに、このステップを繰り返します。

   b. ルーターが AMQP クライアントとの接続が必要な場合には、クライアント接続を設定します。

   c. ルーターが外部 AMQP コンテナー (メッセージブローカーなど) に接続する必要がある場合、接続を設定します。

3. 前のステップで設定した各コネクションを保護します。

4. (任意) 追加のプロパティーを設定します。
   これらのプロパティーは、各ルーターでも同じ方法で設定する必要があります。したがって、それぞれを一度だけ設定してから、ルーターネットワーク内の各追加ルーターに設定をコピーする必要があります。

   - 承認
     必要な場合は、ポリシーを設定し、どのメッセージングリソースクライアントがルーターネットワークでアクセスできるかを制御します。

   - ルーティング
     AMQ Interconnect は設定なしでメッセージを自動的にルーティングします。クライアントはメッセージをルーターネットワークに送信でき、ルーターはそれらの宛先に自動的にルーティングできます。ただし、実際の要件を満たすようにルーティングを設定することができます。ルーティングパターンを、特定のアドレスに使用するルーティングパター

ン、ブローカーキューでメッセージをルーティングするためのポイントと自動リンクを作成し、リンクルートを作成してクライアントをブローカーに接続するように設定できます。

- ロギング
  デフォルトのロギング設定を設定すると、イベントがお使いの環境の正しいレベルに記録されるようにすることができます。

5. ルーターネットワークに追加する各追加のルーターに対して、このワークフローを繰り返します。

5.3. ルーターの起動

qdrouterd コマンドを使用してルーターを起動します。ルーターは、フォアグラウンド、バックグラウンド、またはサービスとして起動できます。

手順

- 次のいずれかを行います。

以下を行う場合	このコマンドを入力...
フォアグラウンドでルーターを起動します。	`$ qdrouterd`
デーモンとしてバックグラウンドでルーターを起動します。	`$ qdrouterd -d`
サービスとしてルーターを起動します。	`$ systemctl start qdrouterd.service` **注記** ルーターをサービスとして起動する場合、**systemd LimitNOFILE** 制限はルーター用に開くことのできる接続の数に影響を与えます。制限に達すると、ルーターは他の接続を受け入れできず、Too many open files を示すエラーメッセージが記録されます。この制限に到達しないように、**systemd** プロセスの **LimitNOFILE** 値を増やします。 詳細は RHEL 7 および systemd でサービスに制限を設定する方法 を参照してください。

第6章 AMQ INTERCONNECT のアップグレード

AMQ Interconnect を最新バージョンにアップグレードし、最新の機能強化および修正が確実に実行されるようにする必要があります。アップグレードプロセスでは、新しい AMQ Interconnect パッケージをインストールして、ルーターを再起動する必要があります。

この手順を使用して、AMQ Interconnect を新しい**マイナーリリース**または**メンテナーンスリリース**にアップグレードすることができます 。

マイナーリリース

AMQ Interconnect では、定期的にポイントリリースが提供されます。ポイントリリースは、新機能およびセキュリティー修正が含まれるマイナー更新です。AMQ Interconnect 1.0 から AMQ Interconnect 1.1 などのように、AMQ Interconnect ポイントリリースから別の AMQ Interconnect 1.1 にアップグレードする予定の場合、プライベート、サポートされていない、またはテクノロジープレビューコンポーネントを使用しないアプリケーションには、コードの変更は必要ありません。

メンテナーンスリリース

AMQ Interconnect では、バグ修正が含まれるメンテナーンスリリースも定期的に提供されます。メンテナーンスリリースは、1.0.0 から 1.0.1 など、最後の数字のマイナーリリースバージョンを増分します。メンテナーンスリリースにはコードの変更は必要ありませんが、一部のメンテナーンスリリースには設定の変更が必要になる場合があります。

前提条件

アップグレードを実行する前に、ターゲットリリースのリリースノートを確認し、新機能、拡張機能、修正、および問題を理解するようにしてください。ターゲットリリースのリリースノートを確認するには、Red Hat カスタマーポータル を参照してください。

手順

1. **qpid-dispatch-router** パッケージおよび **qpid-dispatch-tools** パッケージとその依存関係をアップグレードします。

   ```
   $ sudo yum update qpid-dispatch-router qpid-dispatch-tools
   ```

   詳細は、5章*AMQ Interconnect のインストール* を参照してください。

2. ルーターネットワークで各ルーターを再起動します。
   中断を防ぐには、それぞれのルーターを一度に 1 つずつ再起動する必要があります。

   以下の例では、Red Hat Enterprise Linux 7 でルーターを再起動します。

   ```
   $ systemctl restart qdrouterd.service
   ```

   ルーターの起動についての詳細は、「ルーターの起動」 を参照してください。

パート V. 設定

第7章 ルータープロパティーの設定

デフォルトでは、AMQ Interconnect は無作為に生成される ID で **standalone** モードで動作します。ルーターネットワークでこのルーターを使用する場合は、これらのプロパティーを変更する必要があります。

手順

1. **/etc/qpid-dispatch/qdrouterd.conf** 設定ファイルを開きます。
2. **router** セクションで、モードおよび ID を指定します。
   以下の例は、**interior** モードで動作するように設定されたルーターを示しています。

```
router {
    mode: interior
    id: Router.A
}
```

   mode

   以下のモードのいずれかを指定します。

   - **standalone**: ルーターが他のルーターと通信せず、ルーターネットワークの一部ではない場合は、このモードを使用します。このモードで実行している場合、ルーターは直接接続されたエンドポイント間でメッセージのみをルーティングします。
   - **interior**: ルーターがルーターネットワークの一部である場合、他のルーターと協力する必要がある場合は、このモードを使用します。
   - **edge**: ルーターが干渉ルーターのネットワークに接続するエッジルーターの場合、このモードを使用します。

   id

   ルーターの一意識別子。この ID は AMQP プロトコルレベルのコンテナー名でもあります。

3. 必要な場合は、ルーターの追加のプロパティーを設定します。
   その他の属性に関する情報は、qdrouterd.conf の man ページの ルーター を参照してください。

第8章 ネットワーク接続の設定

AMQ Interconnect は、ネットワーク接続経由でクライアント、サーバー、AMQP サービス、およびその他のルーターに接続します。ルーターを他のメッセージングエンドポイントに接続するには、**リスナー**が接続を許可するよう設定し、**コネクター**がアウトバウンド接続を行うように設定します。ただし、接続には双方向で行われます。接続が確立されると、両方の方向にメッセージトラフィックフローが表示されます。

以下を実行できます。

- ルーターを別のルーターに接続する
- クライアント接続のリッスン
- 外部 AMQP コンテナーへのルーターの接続
- 接続へのメタデータの追加
- 接続のフェイルオーバーについて

8.1. ルーターの接続

ルーターネットワークの別のルーターに接続するには、あるルーターで **connector** を設定し 、アウトバウンド接続を作成し、他のルーターの **listener** を設定して接続を受け入れます。

接続は双方向であるため、ルーターのペア間では 1 つの接続のみが必要です。接続が確立されたら、両方の方向にメッセージトラフィックフローが表示されます。

以下の手順では、ルーターをルーターネットワーク内の別のルーターに接続する方法について説明します。

手順

1. 接続の方向を決定します。
   connector であるルーターを決定し、listener にしてください。接続を確立する方向は任意ですが、以下の要素を考慮してください。

   IP ネットワーク境界およびファイアウォール

   通常、ルーター間接続は常によりプライベートからよりパブリックに確立される必要があります。たとえば、プライベート IP ネットワークのルーターをパブリックの場所 (パブリッククラウドプロバイダーなど) の別のルーターに接続するには、プライベートネットワークのルーターがコネクターで、パブリックの場所のルーターがリスナーである必要があります。これは、VPN を使用したり、パブリックからプライベートアクセスを許可するように設計されたその他のファイアウォール機能を使用せずに、TCP/IP がプライベートの場所に到達できないためです。

   ネットワークトポロジー

   ルーターネットワークのトポロジーは、ルーター間で接続を確立する必要がある方向に影響を及ぼす可能性があります。たとえば、1 つまたは 2 つの中心の hub ルーターに接続された複数のルーターがあるスタートポロジーでは、ハブ上にリスナー、スポーク上にコネクターが必要です。これにより、ハブの設定を変更せずに新規のスポークルーターを追加することができます。

2. 接続を作成するルーターで、**/etc/qpid-dispatch/qdrouterd.conf** 設定ファイルを開き、**connector** を追加します。

この例では、2 つの相互オリオータールーター間のルーター間接続の **connector** を作成します。

```
connector {
    host: 192.0.2.1
    port: 5001
    role: inter-router
    ...
}
```

host

ルーターが接続する IP アドレス (IPv4 または IPv6)、またはルーターのホスト名。

port

ルーターが接続するポート番号またはシンボリックサービス名（ **/etc/services** で定義されます）。

role

接続のロール。接続が 2 つのインテリアルーターの間にある場合は、**inter-router** を指定します。接続がインテリアルーターとエッジルーターの間にある場合は、**edge** を指定します。

3. 接続を確立すべきルーターで、**/etc/qpid-dispatch/qdrouterd.conf** 設定ファイルを開き、inter-router **listener** が設定されたことを確認します。
以下の例では、前のステップで設定された接続を確立するために **listener** を作成します。

```
listener {
    host: 0.0.0.0
    port: 5001
    role: inter-router
    ...
}
```

host

ルーターがリッスンする IP アドレス (IPv4 または IPv6)、またはルーターのホスト名。

port

ルーターがリッスンするポート番号またはシンボリックサービス名（ **/etc/services** で定義されます）。

role

接続のロール。接続が 2 つのインテリアルーターの間にある場合は、**inter-router** を指定します。接続がインテリアルーターとエッジルーターの間にある場合は、**edge** を指定します。

4. ルーターが他のルーターに接続する必要がある場合、この手順を繰り返します。
エッジルーターは、インテリアルーターにのみ接続することができます。他のエッジルーターに接続できない。

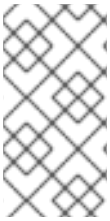
関連情報

- ルーターを別のルーターに接続したら、接続をセキュアにします。
詳細は、「ルーター間の接続のセキュリティー保護」 を参照してください。

8.2. クライアント接続をリッスンしています。

ルーターが AMQP クライアントからの接続をリッスンし、受け入れるようにするには、**listener** を設定します。

ルーターで接続を有効にすると、クライアントはブローカーへの接続に使用するものと同じ方法を使用して接続できます。クライアントの視点では、ルーターの接続とリンクの確立はブローカーの接続とリンクの確立と同じです。



注記

クライアントから接続をリッスンするように **listener** を設定する代わりに、**connector** を設定してクライアントへの接続を開始することができます。この場合、ルーターは **connector** を使用して接続を開始しますが、リンクは作成されません。リンクは、接続を受け入れるピアによってのみ作成されます。

手順

1. **/etc/qpid-dispatch/qdrouterd.conf** 設定ファイルを開きます。
2. **normal** ロールで **listener** を設定します。

```
listener {
    host: primary.example.com
    port: 5672
    role: normal
    failoverUrls: secondary.example.com:20000, tertiary.example.com
    ...
}
```

host

ルーターがリッスンする IP アドレス (IPv4 または IPv6)、またはルーターのホスト名。

port

ルーターがリッスンするポート番号またはシンボリックサービス名（ **/etc/services** で定義されます）。

role

接続のロール。AMQP クライアントのメッセージ配信にこのコネクションが使用されることを示すために **normal** を指定します。

failoverUrls (オプション)

確立された接続が失われた場合にクライアントが再接続に使用できるバックアップ URL のコンマ区切りリスト。各 URL は以下の形式を使用する必要があります。
[(amqp|amqps|ws|wss)://](HOST|IP ADDRESS)[:port]

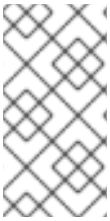
詳細は、「接続のフェイルオーバーについて」 を参照してください。

関連情報

- ルーターがクライアント接続をリッスンできるようにし、接続をセキュアにします。
詳細は、「着信クライアント接続のセキュリティー保護」 を参照してください。

8.3. 外部 AMQP コンテナーへの接続

ルーターが外部 AMQP コンテナーへの接続を確立できるようにするには (メッセージブローカーなど)、**connector** を設定します。



注記

AMQP コンテナーへのコネクションを開始するように **connector** を設定する代わりに、AMQP コンテナーから接続をリッスンするように **listener** を設定できます。ただし、この場合は、AMQP コンテナーのアドレスは、AMQP コンテナーがコネクションが作成された後にのみルーティングに使用できます。

手順

1. **/etc/qpid-dispatch/qdrouterd.conf** 設定ファイルを開きます。
2. **route-container** ロールで **connector** を設定します。
   この例では、ブローカーへのコネクションを開始する **connector** を作成します。ルーターによって接続が作成され、ブローカーによって許可される場合、ブローカーのアドレスはルーティングに利用可能になります。

```
connector {
    name: my-broker
    host: 192.0.2.10
    port: 5672
    role: route-container
    ...
}
```

name

connector の名前。ルーターが接続するエンティティーを記述する名前を指定します。

host

ルーターが接続する IP アドレス (IPv4 または IPv6)、またはルーターのホスト名。

port

ルーターが接続するポート番号またはシンボリックサービス名（ **/etc/services** で定義されます）。

role

接続のロール。**route-container** を指定して、このコネクションが既知のアドレスを保持する AMQP コンテナーのものであることを指定します。

関連情報

- ルーターが外部の AMQP コンテナーに接続できるように設定した後に、必要なセキュリティー認証情報を設定します。
  詳細は、「発信接続のセキュリティー保護」 を参照してください。

8.4. コネクションへのメタデータの追加

複雑なトポロジーでは、メッセージをプログラムで処理できるように、接続にメタデータを追加するのに便利です。

手順

1. **/etc/qpid-dispatch/qdrouterd.conf** 設定ファイルを開きます。

2. 'openProperties' 属性を使用して、任意の JSON を **connector** 設定に追加します。
この例では、プロパティー **label** に **green** という値を追加します。

```
connector {
  name: broker
  role: route-container
  host: 127.0.0.1
  port: 22180
  saslMechanisms: ANONYMOUS
  openProperties: {
    "label": "green"
  }
}
```

JSON エントリーには、以下の制限に注意してください。

- ASCII 文字 (鍵のみ)
- 以下のキーは許可されていません。
  - 製品
  - バージョン
  - failover-server-list
  - network-host
  - port
  - scheme
  - hostname
  - **qd** で始まるキー。
  - **x-opt-qd** で始まるキー。

openProperties 属性は、**normal** のロールまたは **route-container** ロールを持つコネクターにのみ設定できます。以下の設定を持つコネクターの属性を設定することはできません。

- **role: inter-router**
- **role: edge**
- **http: true**

JSON 形式は、以下のようにリスト、マップ、および複数のエントリーをサポートします。

```
connector {
  name: broker
  role: route-container
  host: 127.0.0.1
  port: 22180
  saslMechanisms: ANONYMOUS
  openProperties: {
    "foo": "bar",
```

```
      "integer": 7,
 "list":  ["a", 1, "b", -9, true],
 "map":  {"key1": null, "key2": [1, 2, 3]},
 }
   cost: 10
}
```

8.5. 接続のフェイルオーバーについて

ルーターとリモートホスト間の接続が失敗すると、接続フェイルオーバーにより、別の URL で接続が自動的に再確立されます。

ルーターは、着信接続と発信接続の両方に対して接続のフェイルオーバーを使用できます。

送信接続の接続フェイルオーバー

デフォルトでは、ルーターで **connector** を設定すると、ルーターは設定済みのリモートホストおよびポートへのオープンネットワークトランスポート接続の管理を試みます。接続を確立できない場合は、ルーターは接続が確立されるまで継続的に再試行します。接続が確立されて失敗すると、ルーターは接続を直ちに再確立しようとします。
ルーターがリモートホストへの接続を確立する場合、クライアントは接続が失われた場合に使用できる別の接続情報 (フェイルオーバー一覧と呼ばれることもあります) を持つルーターを提供する場合があります。このような場合、同じホストで接続が再確立を試行するのではなく、ルーターは別のホストも試行します。

接続フェイルオーバーは、ルーターが同じサービスを提供するサーバーのクラスターへの送信接続を確立するときに特に便利です。

受信接続のフェイルオーバー

ルーター上で **listener** を設定し、バックアップとして使用するフェイルオーバー URL の一覧を提供できます。接続が失われた場合、クライアントはこれらのフェイルオーバー URL を使用してルーターへの接続を再確立できます。

第9章 ネットワーク接続のセキュリティー保護

ルーターの接続を認証および暗号化することで、安全な方法で AMQ Interconnect をクライアント、ルーター、ブローカーと通信するように設定できます。AMQ Interconnect は以下のセキュリティープロトコルをサポートします。

- 証明書ベースの暗号化および相互認証向けの SSL/TLS
- メカニズムと認証用の SASL

SSL/TLS、SASL (または両方の組み合わせ) を設定して、以下のいずれかを保護します。

- ルーター間のセキュアな接続
- 着信クライアント接続のセキュリティー保護
- セキュアな接続

9.1. ルーター間の接続のセキュリティー保護

中間ルーター間の接続は、SSL/TLS 暗号化と認証 (相互認証とも呼ばれる) を使用してセキュリティーを確保し、承認されていないルーター (またはルーターとして準備しているエンドポイント) がネットワークに加わらないようにする必要があります。

SSL/TLS 相互認証には、各相互オーリオールーター用に生成された個別の証明書を使用して X.509 認証局 (CA) が必要です。inter-router 間の接続は暗号化され、CA は各受信ルーター間接続を認証します。

この手順では、SSL/TLS 相互認証を使用して、2 つの相互オータールーター間の接続のセキュリティーを保護する方法を説明します。

前提条件

- インテリアルーターには X.509 認証局が存在する必要があります。
- セキュリティー証明書は各ルーター用に生成され、CA によって署名される必要があります。
- inter-router 接続はルーター間で存在している必要があります。
  詳細は、「ルーターの接続」 を参照してください。

手順

1. 接続を確立するルーターで、以下を実行します。
   a. **/etc/qpid-dispatch/qdrouterd.conf** を開きます。
   b. ルーターに、ルーター間のネットワークの秘密鍵と証明書を定義する **sslProfile** が含まれていない場合、これを追加します。
      この **sslProfile** には、ルーターがピアとの認証に使用する秘密鍵の場所と証明書が含まれます。

```
sslProfile {
    name: inter-router-tls
    certFile: /etc/pki/tls/certs/tls.crt
    caCertFile: /etc/pki/tls/certs/ca.crt
```

```
    privateKeyFile: /etc/pki/tls/private/tls.key
    password: file:/etc/pki/tls/private/password.txt
    ...
}
```

name

この **sslProfile** の参照に使用できる一意の名前。

certFile

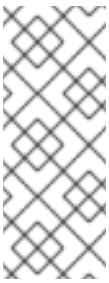
このルーターのパブリック証明書を含むファイルへの絶対パス。

caCertFile

ルーターが受信クライアントの認証に使用する CA 証明書への絶対パス。

privateKeyFile

このルーターのパブリック証明書に対する秘密鍵が含まれるファイルへの絶対パス。



注記

qdrouterd または root ユーザーが秘密鍵にアクセスできることを確認します。以下は例になります。

```
chmod 0600 /etc/pki/tls/private/tls.key
chown qdrouterd /etc/pki/tls/private/tls.key
```

パスワード

証明書キーのロックを解除するパスワード。証明書キーにパスワードがない場合は、指定する必要はありません。異なる接頭辞を使用することで、セキュリティー要件に応じてパスワードを複数回指定できます。

- パスワードを含むファイルへの絶対パスを指定します。これは、パスワードを含むファイルにパーミッションを設定することができるため、最も安全なオプションです。以下は例になります。

  ```
  password: file:/etc/qpid-dispatch-certs/inter-router/password.txt
  ```

- パスワードを保存する環境変数を指定します。他のプロセスの環境は特定のプラットフォームで表示されるため、このオプションの使用には注意が必要です。以下は例になります。

  ```
  password: env:CERT_PASSWORD
  ```

- パスワードをクリアテキストで指定します。このオプションは安全ではないため、セキュリティーが懸念されていない場合にのみ使用してください。以下は例になります。

  ```
  password: pass:mycertpassword
  ```

c. 作成した **sslProfile** を使用するように、このコネクションの相互ルーター **connector** を設定します。

```
connector {
```

```
    host: 192.0.2.1
    port: 5001
    role: inter-router
    sslProfile: inter-router-tls
    ...
}
```

sslProfile

SSL/TLS プライベートキーおよびルーター間ネットワークの証明書を定義する **sslProfile** の名前。

2. 接続をリッスンするルーターで、以下を実行します。

a. **/etc/qpid-dispatch/qdrouterd.conf** を開きます。

b. ルーターに、ルーター間のネットワークの秘密鍵と証明書を定義する **sslProfile** が含まれていない場合、これを追加します。

c. SSL/TLS を使用して接続をセキュアにするには、この接続にルーター間 **listener** を設定します。

```
listener {
    host: 0.0.0.0
    port: 5001
    role: inter-router
    sslProfile: inter_router_tls
    authenticatePeer: yes
    requireSsl: yes
    saslMechanisms: EXTERNAL
    ...
}
```

sslProfile

SSL/TLS プライベートキーおよびルーター間ネットワークの証明書を定義する **sslProfile** の名前。

authenticatePeer

yes を指定して、ピアインオリオーターの ID を認証します。

requireSsl

yes を指定して、SSL/TLS で接続を暗号化します。

saslMechanisms

X.509 クライアント証明書認証を有効にするために **EXTERNAL** を指定します。

9.2. 着信クライアント接続のセキュリティー保護

SSL/TLS および SASL を使用して、クライアントトラフィックの適切なセキュリティーレベルをルーターネットワークに提供することができます。以下の方法を使用して、AMQP クライアント、外部コンテナー、または edge ルーターから受信接続をセキュリティー保護することができます。

- SSL/TLS 暗号化の有効化
- SSL/TLS クライアント認証の有効化
- ユーザー名とパスワード認証の有効化

- Kerberos との統合

9.2.1. SSL/TLS 暗号化の有効化

SSL/TLS を使用して、クライアントから着信接続を暗号化できます。

前提条件

- X.509 認証局 (CA) がクライアント接続に存在している必要があります。
- セキュリティー証明書は CA によって生成および署名されている必要があります。

手順

1. **/etc/qpid-dispatch/qdrouterd.conf** 設定ファイルを開きます。
2. ルーターに、クライアント接続の秘密鍵と証明書を定義する **sslProfile** が含まれていない場合は、これを追加します。
   この **sslProfile** には、ルーターがクライアントからの接続の暗号化に使用する秘密鍵と証明書の場所が含まれます。

```
sslProfile {
    name: service-tls
    certFile: /etc/pki/tls/certs/tls.crt
    caCertFile: /etc/pki/tls/certs/ca.crt
    privateKeyFile: /etc/pki/tls/private/tls.key
    password: file:/etc/pki/tls/private/password.txt
    ...
}
```

name
この **sslProfile** の参照に使用できる一意の名前。

certFile
このルーターのパブリック証明書を含むファイルへの絶対パス。

caCertFile
ルーターが受信クライアントの認証に使用する CA 証明書への絶対パス。

privateKeyFile
このルーターのパブリック証明書に対する秘密鍵が含まれるファイルへの絶対パス。

注記

qdrouterd または root ユーザーが秘密鍵にアクセスできることを確認します。以下は例になります。

```
chmod 0600 /etc/pki/tls/private/tls.key
chown qdrouterd /etc/pki/tls/private/tls.key
```

パスワード

証明書キーのロックを解除するパスワード。証明書キーにパスワードがない場合は、指定する必要はありません。異なる接頭辞を使用することで、セキュリティー要件に応じてパスワードを複数回指定できます。

- パスワードを含むファイルへの絶対パスを指定します。これは、パスワードを含むファイルにパーミッションを設定することができるため、最も安全なオプションです。以下は例になります。

```
password: file:/etc/qpid-dispatch-certs/inter-router/password.txt
```

- パスワードを保存する環境変数を指定します。他のプロセスの環境は特定のプラットフォームで表示されるため、このオプションの使用には注意が必要です。以下は例になります。

```
password: env:CERT_PASSWORD
```

- パスワードをクリアテキストで指定します。このオプションは安全ではないため、セキュリティーが懸念されていない場合にのみ使用してください。以下は例になります。

```
password: pass:mycertpassword
```

3. SSL/TLS を使用して接続を暗号化するように、この接続が **listener** を設定します。
この例では、**normal** リスナーがクライアントからの接続を暗号化するように設定します。

```
listener {
    host: 0.0.0.0
    port: 5672
    role: normal
    sslProfile: inter_router_tls
    requireSsl: yes
    ...
}
```

sslProfile

SSL/TLS プライベートキーおよびクライアント接続の証明書を定義する **sslProfile** の名前。

requireSsl

true を指定して SSL/TLS で接続を暗号化します。

9.2.2. SSL/TLS クライアント認証の有効化

SSL/TLS 暗号化の他に、SSL/TLS を使用してクライアントから受信接続を認証することもできます。この方法では、クライアントが独自の X.509 証明書をルーターに提示する必要があります。ルーターを使用してクライアントのアイデンティティーを検証します。

前提条件

- SSL/TLS 暗号化を設定する必要があります。
詳細は、「SSL/TLS 暗号化の有効化」 を参照してください。

- クライアントには、ルーターに対する認証に使用できる X.509 証明書が必要です。

手順

1. **/etc/qpid-dispatch/qdrouterd.conf** 設定ファイルを開きます。

2. SSL/TLS を使用してクライアントを認証するように、この接続の **listener** を設定します。
この例では、クライアントからの受信接続を認証するために、**normal** リスナーに SSL/TLS 認証を追加します。クライアントは、独自の X.509 証明書を提示してルーターにだけ接続でき、ルーターを使用してクライアントのアイデンティティーを検証するために使用されます。

```
listener {
    host: 0.0.0.0
    port: 5672
    role: normal
    sslProfile: service-tls
    requireSsl: yes
    authenticatePeer: yes
    saslMechanisms: EXTERNAL
    ...
}
```

authenticatePeer

クライアントのアイデンティティーを認証するには **yes** を指定します。

saslMechanisms

X.509 クライアント証明書認証を有効にするために **EXTERNAL** を指定します。

9.2.3. ユーザー名とパスワード認証の有効化

SASL PLAIN メカニズムを使用して、ユーザー名とパスワードのセットに対して受信クライアント接続を認証できます。このメソッドは独自に使用することも、SSL/TLS 暗号化と組み合わせることもできます。

前提条件

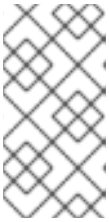
- **cyrus-sasl-plain** プラグインがインストールされている。
Cyrus SASL は、プラグインを使用して特定の SASL メカニズムをサポートします。特定の SASL メカニズムを使用する前に、関連するプラグインをインストールする必要があります。

  Red Hat Enterprise Linux の Cyrus SASL プラグインのリストを表示するには、**yum search cyrus-sasl** コマンドを使用します。Cyrus SASL プラグインをインストールするには、**yum install <plugin>** コマンドを使用します。

手順

1. 必要に応じて、ユーザー名とパスワードを SASL データベースに追加します。
この例では、新しいユーザー (user1@example.com) を SASL データベース (qdrouterd.sasldb) に追加します。

```
$ sudo saslpasswd2 -c -f qdrouterd.sasldb -u example.com user1
```



注記

完全なユーザー名は、入力したユーザー名 (**<user-name>**@**<domain-name>**) です。ユーザーをデータベースに追加する場合、ドメイン名を指定する必要はありませんが、これを指定しないと、デフォルトのドメインが自動的に追加されます (ツールが実行されているマシンのホスト名)。

2. **qdrouterd** プロセスが SASL データベースを読み取りできることを確認します。
qdrouterd プロセスが非特権ユーザーとして実行される場合は、ルーターが読み取れるように SASL データベースのパーミッションまたは所有権を調整する必要がある場合があります。

   この例では、qdrouterd ユーザーが SASL データベースの所有者になります。

```
$ sudo chown qdrouterd /var/lib/qdrouterd/qdrouterd.sasldb
```

3. **/etc/sasl2/qdrouterd.conf** 設定ファイルを開きます。
以下の例では、**/etc/sasl2/qdrouterd.conf** 設定ファイルを示しています。

```
pwcheck_method: auxprop
auxprop_plugin: sasldb
sasldb_path: qdrouterd.sasldb
mech_list: ANONYMOUS DIGEST-MD5 EXTERNAL PLAIN GSSAPI
```

4. **mech_list** 属性に **PLAIN** メカニズムが含まれることを確認します。

5. **/etc/qpid-dispatch/qdrouterd.conf** 設定ファイルを開きます。

6. **router** セクションで、SASL 設定ファイルへのパスを指定します。

```
router {
    mode: interior
    id: Router.A
    saslConfigDir: /etc/sasl2/
}
```

   saslConfigDir

   ユーザー名とパスワードを格納する SASL データベースへのパスが含まれる SASL 設定ファイルへのパス。

7. SASL PLAIN を使用してクライアントを認証するように、この接続の **listener** を設定します。
この例では、**listener** に基本的なユーザー名とパスワード認証を設定します。この場合、SSL/TLS 暗号化は使用されません。

```
listener {
    host: 0.0.0.0
    port: 5672
    authenticatePeer: yes
    saslMechanisms: PLAIN
    }
```

9.2.4. Kerberos との統合

環境に Kerberos を実装している場合、**GSSAPI** SASL メカニズムとともに使用して受信接続を認証できます。

前提条件

- Kerberos インフラストラクチャーはお使いの環境にデプロイする必要があります。
- Kerberos 環境では、**amqp/<hostname>@<realm>** のサービスプリンシパルを設定する必要があります。
  これは、AMQ Interconnect が使用するサービスプリンシパルです。
- **cyrus-sasl-gssapi** パッケージが各クライアントおよびルーターホストマシンにインストールされている。

手順

1. ルーターのホストマシンで、**/etc/sasl2/qdrouterd.conf** 設定ファイルを開きます。
   以下の例では、**/etc/sasl2/qdrouterd.conf** 設定ファイルを示しています。

```
pwcheck_method: auxprop
auxprop_plugin: sasldb
sasldb_path: qdrouterd.sasldb
keytab: /etc/krb5.keytab
mech_list: ANONYMOUS DIGEST-MD5 EXTERNAL PLAIN GSSAPI
```

2. 以下を確認します。
   - **mech_list** 属性には **GSSAPI** メカニズムが含まれます。
   - **keytab** 属性は、キータブファイルの場所を参照します。
3. **/etc/qpid-dispatch/qdrouterd.conf** 設定ファイルを開きます。
4. **router** セクションで、SASL 設定ファイルへのパスを指定します。

```
router {
    mode: interior
    id: Router.A
    saslConfigDir: /etc/sasl2/
}
```

   saslConfigDir

   SASL データベースへのパスが含まれる SASL 設定ファイルへのパス。

5. 認証に Kerberos を使用する各着信接続に対して、**listener** が **GSSAPI** メカニズムを使用するように設定します。

```
listener {
    host: 0.0.0.0
    port: 5672
    authenticatePeer: yes
    saslMechanisms: GSSAPI
    }
```

9.3. 発信接続のセキュリティー保護

外部 AMQP コンテナー (メッセージブローカーなど) への接続を作成するようにルーターが設定されている場合、以下の方法を使用して接続をセキュリティー保護することができます。

- SSL/TLS 暗号化を使用した接続 (一方向認証)
- SSL/TLS 相互認証を使用した接続
- ユーザー名およびパスワード認証 (SSL/TLS 暗号化ありまたはなしで) を使用した接続

9.3.1. 一方向 SSL/TLS 認証を使用した接続

一方向 SSL/TLS を使用して、外部 AMQP コンテナー (ブローカーなど) に接続できます。この方法では、ルーターは外部 AMQP コンテナーのサーバー証明書を検証し、そのアイデンティティーを検証します。

手順

1. **/etc/qpid-dispatch/qdrouterd.conf** 設定ファイルを開きます。
2. 外部 AMQP コンテナーのアイデンティティーの検証に使用できる証明書を定義する **sslProfile** が含まれていない場合は、これを追加します。

```
sslProfile {
    name: broker-tls
    caCertFile: /etc/qpid-dispatch-certs/ca.crt
    ...
}
```

name

この **sslProfile** の参照に使用できる一意の名前。

caCertFile

外部 AMQP コンテナーのアイデンティティーの検証に使用される CA 証明書への絶対パス。

3. SSL/TLS を使用して、SSL ハンドシェイク中にブローカーによって受信されたサーバー証明書を検証するために、この接続で **connector** を設定します。
この例では、**connector** をブローカーに設定します。ルーターはブローカーに接続すると、**broker-tls sslProfile** に定義された CA 証明書を使用して、ブローカーから受け取ったサーバー証明書を検証します。

```
connector {
    host: 192.0.2.1
    port: 5672
    role: route-container
    sslProfile: broker-tls
    ...
}
```

sslProfile

外部 AMQP コンテナーのアイデンティティーの検証に使用する証明書を定義する **sslProfile** の名前。

9.3.2. 相互 SSL/TLS 認証を使用した接続

相互 SSL/TLS 認証を使用して、外部 AMQP コンテナー (ブローカーなど) に接続できます。この方法では、クライアントとして動作するルーターは、ルーターの ID を検証できるように外部の AMQP コンテナーに証明書を提供します。

前提条件

- ルーターには X.509 認証局 (CA) が存在している必要があります。
- セキュリティー証明書はルーター用に生成され、CA によって署名される必要があります。

手順

1. **/etc/qpid-dispatch/qdrouterd.conf** 設定ファイルを開きます。
2. ルーターに、外部の AMQP コンテナーに接続するための秘密鍵と証明書を定義する **sslProfile** が含まれていない場合は、これを追加します。
   この **sslProfile** には、ルーターがピアとの認証に使用する秘密鍵と証明書の場所が含まれます。

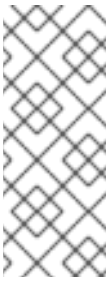
```
sslProfile {
    name: broker-tls
    certFile: /etc/pki/tls/certs/tls.crt
    caCertFile: /etc/pki/tls/certs/ca.crt
    privateKeyFile: /etc/pki/tls/private/tls.key
    password: file:/etc/pki/tls/private/password.txt
    ...
}
```

name
この **sslProfile** の参照に使用できる一意の名前。

certFile
このルーターのパブリック証明書を含むファイルへの絶対パス。

caCertFile
ルーターが受信クライアントの認証に使用する CA 証明書への絶対パス。

privateKeyFile
このルーターのパブリック証明書に対する秘密鍵が含まれるファイルへの絶対パス。



注記

qdrouterd または root ユーザーが秘密鍵にアクセスできることを確認します。以下は例になります。

```
chmod 0600 /etc/pki/tls/private/tls.key
chown qdrouterd /etc/pki/tls/private/tls.key
```

パスワード

証明書キーのロックを解除するパスワード。証明書キーにパスワードがない場合は、指定する必要はありません。異なる接頭辞を使用することで、セキュリティー要件に応じてパスワードを複数回指定できます。

- パスワードを含むファイルへの絶対パスを指定します。これは、パスワードを含むファイルにパーミッションを設定することができるため、最も安全なオプションです。以下は例になります。

```
password: file:/etc/qpid-dispatch-certs/inter-router/password.txt
```

- パスワードを保存する環境変数を指定します。他のプロセスの環境は特定のプラットフォームで表示されるため、このオプションの使用には注意が必要です。以下は例になります。

```
password: env:CERT_PASSWORD
```

- パスワードをクリアテキストで指定します。このオプションは安全ではないため、セキュリティーが懸念されていない場合にのみ使用してください。以下は例になります。

```
password: pass:mycertpassword
```

3. 作成した **sslProfile** を使用するように、このコネクションの **connector** を設定します。

```
connector {
    host: 192.0.2.1
    port: 5672
    role: route-container
    sslProfile: broker-tls
    saslMechanisms: EXTERNAL
    ...
}
```

sslProfile

SSL/TLS プライベートキーおよびルーター間ネットワークの証明書を定義する **sslProfile** の名前。

9.3.3. ユーザー名とパスワード認証を使用した接続

SASL PLAIN メカニズムを使用すると、ユーザー名とパスワードを必要とする外部 AMQP コンテナーに接続できます。このメソッドは独自に使用することも、SSL/TLS 暗号化と組み合わせることもできます。

前提条件

- **cyrus-sasl-plain** プラグインがインストールされている。
  Cyrus SASL は、プラグインを使用して特定の SASL メカニズムをサポートします。特定の SASL メカニズムを使用する前に、関連するプラグインをインストールする必要があります。

  Red Hat Enterprise Linux の Cyrus SASL プラグインのリストを表示するには、**yum search cyrus-sasl** コマンドを使用します。Cyrus SASL プラグインをインストールするには、**yum install <plugin>** コマンドを使用します。

手順

1. **/etc/qpid-dispatch/qdrouterd.conf** 設定ファイルを開きます。

2. このコネクションが、外部 AMQP コンテナーに対してユーザー名とパスワードのクレデンシャルを提供するように **connector** を設定します。

```
connector {
    host: 192.0.2.1
    port: 5672
    role: route-container
    saslMechanisms: PLAIN
    saslUsername: user
    saslPassword: file:/path/to/file/password.txt
    }
```

saslPassword

ピアに接続するためのパスワード。異なる接頭辞を使用することで、セキュリティー要件に応じてパスワードを複数回指定できます。

- パスワードを含むファイルへの絶対パスを指定します。これは、パスワードを含むファイルにパーミッションを設定することができるため、最も安全なオプションです。以下は例になります。

```
password: file:/path/to/file/password.txt
```

- パスワードを保存する環境変数を指定します。他のプロセスの環境は特定のプラットフォームで表示されるため、このオプションの使用には注意が必要です。以下は例になります。

```
password: env:PASSWORD
```

- パスワードをクリアテキストで指定します。このオプションは安全ではないため、セキュリティーが懸念されていない場合にのみ使用してください。以下は例になります。

```
password: pass:mypassword
```

第10章 認証の設定

メッセージング環境でメッセージングリソースのセキュリティーを保護するように **ポリシー** を設定できます。ポリシーにより、認可されたユーザーのみがルーターネットワーク経由でメッセージングエンドポイントにアクセスできること、またこれらのエンドポイントのリソースが承認された方法で使用されるようにします。

- 「ポリシーの種類」
- 「ポリシーによる接続およびリソース制限の適用方法」
- 「グローバル制限の設定」
- 「メッセージングエンドポイントの接続およびリソース制限の設定」

10.1. ポリシーの種類

AMQ Interconnect では、接続およびリソース制限を制御する以下のタイプのポリシーが提供されます。

グローバルポリシー

ルーターの設定。グローバルポリシーは、ルーターの受信ユーザー接続の最大数 (すべてのメッセージングエンドポイント全体) を定義し、ルーターが vhost ポリシーをどのように使用するかを定義します。

vhost ポリシー

ルーター Ingress ポート (AMQP 仮想ホストまたは vhost と呼ばれる) の接続および AMQP リソース制限。vhost ポリシーは、ルーターネットワーク内の任意のメッセージングエンドポイントで、特定の接続を使用するクライアントがアクセスできるものを定義します。

グローバルポリシーおよび vhost ポリシーで定義されたリソース制限は、ユーザー接続にのみ適用されます。この制限は、ポイントに送信されるルーター間接続やルーター接続には影響を与えません。

指定した vhost への特定のユーザー接続に対してポリシーが許可する AMQP リソースにアクセスすると、ルーターネットワーク全体が付与されます。アクセス制限は、クライアントが接続しているルーターポートでのみ適用され、クライアントのリソース要求のみに適用されます。

10.2. ポリシーによる接続およびリソース制限の適用方法

AMQ Interconnect はポリシーを使用して接続を許可するかどうかを判断し、許可される場合は適切なリソース制限を適用します。

クライアントがルーターへの接続を作成すると、ルーターは、最初に接続を許可または拒否するか、または拒否するかを決定します。この決定は、以下の基準に基づいています。

- 接続がルーターのグローバル接続制限を超えているかどうか (グローバルポリシーで定義)
- 接続が vhost の接続制限を超過するかどうか (接続が指示されるホストに一致する vhost ポリシーで定義)

接続が許可されると、ルーターはユーザー (接続から認証済みユーザー名) をユーザーグループに割り当て、ユーザーグループのリソース制限を強制します。

10.3. グローバル制限の設定

グローバルポリシーを作成して、ルーターの着信接続およびメッセージのサイズ制限を設定できます。

手順

- **/etc/qpid-dispatch/qdrouterd.conf** 設定ファイルで、**policy** セクションを追加し、制限を設定します。
  この例では、着信接続の制限およびメッセージサイズを設定します。

```
policy {
    maxConnections: 10000
    maxMessageSize: 2000000
}
```

maxConnections

このルーターで開くことのできる同時クライアント接続の合計数。この制限は、他のポリシー設定が定義されていない場合でも常に適用されます。この制限は、リモートホスト、認証されたユーザー、またはターゲット vhost に関係なく、すべての受信接続に適用されます。デフォルトの (および最大値) 値は **65535** です。

maxMessageSize

メッセージにルーターネットワークに入るため、このルーターで許可される AMQP メッセージ転送の最大サイズ (バイト単位)。この制限は、ユーザー接続を転送し、エッジルーターから他のルーターに転送するために適用されます。この制限は、interior-to-interior ルーター接続には適用されません。この制限は、vhost または vhost ユーザーグループの設定で上書きされる可能性があります。値が **0** の場合は、この制限が無効になります。管理者は、エッジルーター管理要求や応答がブロックされるため、ルーターの最大メッセージサイズを設定しないことを推奨します。管理者は、割り当てられた相互 ior ルーターの最大サイズよりも低いエッジルーターの最大メッセージサイズを設定することも推奨されています。

10.4. メッセージングエンドポイントの接続およびリソース制限の設定

vhost ポリシーを設定して、メッセージングエンドポイントの接続制限および AMQP リソース制限を定義することができます。vhost ポリシーは、特定の接続でメッセージングエンドポイントへのアクセスを許可するリソースクライアントを定義します。

注記

通常、vhost はクライアント接続がダイレクトされるホストの名前です。たとえば、クライアントアプリケーションが **amqp://mybroker.example.com:5672/queue01** URL への接続を開くと、vhost は **mybroker.example.com** になります。

- 「vhost ポリシーの有効化」
- 「vhost ポリシーの作成」
- 「JSON ファイルとしての vhost ポリシーの作成」
- 「送信接続のリソース制限の設定」
- 「vhost ポリシーのソースおよびターゲットアドレスを指定する方法」
- 「vhost ポリシーホスト名パターン一致ルール」
- 「vhost ポリシーの例」

10.4.1. vhost ポリシーの有効化

ポリシーを作成する前に、ルーターを有効にして vhost ポリシーを使用する必要があります。

手順

- **/etc/qpid-dispatch/qdrouterd.conf** 設定ファイルでポリシーセクションを追加し、**policy** セクションが存在しない場合は追加し、ルーターの vhost ポリシーを有効にします。

```
policy {
    ...
    enableVhostPolicy: true
    enableVhostNamePatterns: true
    defaultVhost: $default
}
```

enableVhostPolicy

ルーターは、設定された vhost ポリシーで定義された接続拒否およびリソース制限を適用できるようにします。デフォルトは **false** で、ルーターは vhost ポリシーを適用しません。

enableVhostNamePatterns

vhost ホスト名のパターンの一致を有効にします。**true** に設定すると、ワイルドカードを使用して vhost のホスト名の範囲を指定できます。**false** に設定すると、vhost のホスト名はリテラル文字列として処理されます。つまり、各 vhost の正確なホスト名を指定する必要があります。デフォルトは **false** です。

defaultVhost

vhost ポリシーが設定されていない接続に適用されるデフォルトの vhost ポリシーの名前。デフォルトは **$default** です。**defaultVhost** が定義されていない場合は、デフォルトの vhost 処理が無効になります。

10.4.2. vhost ポリシーの作成

vhost ポリシーは、リモートホストからルーターに接続するユーザーの接続制限およびリソース制限を定義します。リモートホストごとに 1 つの vhost ポリシーを作成する必要があります。

前提条件

ルーターに対して vhost ポリシーを有効にする必要があります。詳細は、「vhost ポリシーの有効化」を参照してください。

手順

1. **vhost** セクションを追加し、メッセージングエンドポイントの接続およびメッセージのサイズ制限を定義します。
   接続制限は、vhost に接続されているすべてのユーザーに適用されます。これらの制限により、vhost を同時に接続できるユーザーの数を制御します。

```
vhost {
    hostname: "example.com"
    aliases: "example.org, example.net"
    maxConnections: 10000
    maxMessageSize: 500000
    maxConnectionsPerUser: 100
    maxConnectionsPerHost: 100
```

```
    allowUnknownUser: true
    ...
}
```

hostname

vhost (メッセージングエンドポイント) のリテラルホスト名または vhost ホスト名に一致するパターンのリテラルホスト名。この vhost ポリシーは、指定したホスト名に転送されるクライアント接続に適用されます。この名前は一意でなければなりません。ホスト名ごとに 1 つの vhost ポリシーのみを指定できます。
enableVhostNamePatterns が **true** に設定されている場合、ワイルドカードを使用してホスト名の範囲に一致するパターンを指定できます。詳細は、「vhost ポリシーホスト名パターン一致ルール」 を参照してください。

aliases

この vhost の設定を使用するようにルーターに指示する代替のリテラルホスト名またはパターン受信接続に一致するエイリアスホスト名には、vhost セクションで定義された設定が使用されます。マルチテナント設定では、vhost エイリアスへの接続はテナント namespace のベース vhost ホスト名を使用します。この例では、接続が vhost **example.org** に表示される場合、ベース vhost ホスト名 **example.com** の設定が適用され、**example.com** がテナントの名前空間になります。すべての vhosts の **hostname** と **aliases** の設定は一意でなければなりません。
enableVhostNamePatterns が **true** に設定されている場合、ワイルドカードを使用してホスト名エイリアスの範囲に一致するパターンを指定できます。詳細は、「vhost ポリシーホスト名パターン一致ルール」 を参照してください。

maxConnections

この vhost で許可されている同時クライアント接続のグローバル最大数デフォルトは 65535 です。

maxMessageSize

この vhost へのコネクションに対して許可される AMQP メッセージ転送の最大サイズ (バイト単位)。この制限は、ポリシーの **maxMessageSize** 値を上書きし、vhost ユーザーグループの設定で上書きされる可能性があります。値が **0** の場合は、この制限が無効になります。

maxConnectionsPerUser

すべてのユーザーが許容される同時クライアント接続の最大数。デフォルトは 65535 です。

maxConnectionsPerHost

すべてのリモートホスト (クライアントが接続しているホスト) に対して許可される同時クライアント接続の最大数。デフォルトは 65535 です。

allowUnknownUser

不明なユーザー (定義されたユーザーグループのメンバーであるユーザー) が vhost への接続を許可するかどうか。不明なユーザーは、**$default** ユーザーグループに割り当てられ、**$default** 設定を受け取ります。デフォルトは **false** で、不明なユーザーには許可されないことを意味します。

2. **vhost** セクションで、追加した接続設定の下に、**group** エンティティーを追加して、リソース制限を定義します。
ユーザーグループでリソース制限を定義します。ユーザーグループは、グループのメンバーがアクセス可能なメッセージングリソースを指定します。

以下の例は、admin、developer、および $default の 3 つのユーザーグループを示しています。

```
vhost {
    ...
    groups: {
        admin: {
            users: "admin1, admin2"
            remoteHosts: "127.0.0.1, ::1"
            sources: "*"
            targets: "*"
        }
        developers: {
            users: "dev1, dev2, dev3"
            remoteHosts: "*"
            sources: "myqueue1, myqueue2"
            targets: "myqueue1, myqueue2"
        }
        $default: {
            remoteHosts: "*"
            allowDynamicSource: true,
            allowAdminStatusUpdate: true,
            sources: "myqueue1, myqueue2"
            targets: "myqueue1, myqueue2"
        }
    }
}
```

注記

引用符を使用して文字列値を定義します。

users

このユーザーグループに認証されたユーザーの一覧。複数のユーザーを分離する場合はコンマで区切ります。ユーザーは 1 つの vhost ユーザーグループにしか属することができません。

remoteHosts

ユーザーが接続できるリモートホストの一覧。ホストには、ホスト名、IP アドレス、または IP アドレス範囲を指定できます。複数のホストを分離する場合はコンマで区切ります。すべてのリモートホストからアクセスを許可するには、ワイルドカード * を指定します。すべてのリモートホストからアクセスを拒否するには、この属性を空白のままにします。

maxConnectionsPerUser

このユーザーグループでユーザーが作成できる接続の最大数。この値が指定されている場合には、vhost **maxConnectionsPerUser** の値が上書きされます。

maxConnectionsPerHost

許可されたリモートホストからこのユーザーグループのユーザーによって作成できる同時接続の最大数。この値が指定されている場合には、vhost **maxConnectionsPerUser** の値が上書きされます。

maxMessageSize

このグループのユーザーが作成したコネクションに許可される AMQP メッセージ転送の最大サイズ (バイト単位)。この制限により、ポリシーと vhost **maxMessageSize** の値が上書きされます。値が **0** の場合は、この制限が無効になります。

allowDynamicSource

true の場合、このグループのユーザーからの接続は受信側を動的ソースに割り当てることができます。これにより、リスナーが一時的なアドレスまたは一時キューに作成できるようになります。**false** の場合、動的ソースの使用は許可されません。

allowAdminStatusUpdate

true の場合、このグループのユーザーからの接続は、接続の **adminStatus** を変更することが許可されます。これにより、送信者またはレシーバー接続の終了が許可されます。**false** の場合、このグループのユーザーは、接続を終了できなくなります。ルーター間接続はどのユーザーでも終了できません。ポリシーが設定されていない場合でも、デフォルトは **true** になります。

allowWaypointLinks

true の場合、このグループ内のユーザーからの接続は、ポイントポイント機能を使ってリンクを割り当てることができます。これにより、エンドポイントが自動リンクを設定する必要なく、ポイント (ブローカー) として機能できるようになります。**false** の場合は、ポイント機能の使用が許可されません。

allowDynamicLinkRoutes

true の場合、このグループのユーザーからの接続は、接続スコープのリンクルートの宛先を動的に作成できます。これにより、エンドポイントはリンクルートの設定を必要とせずに、リンクルートの宛先 (ブローカー) として機能できます。**false** の場合、動的リンクルートの宛先の作成は許可されません。

allowFallbackLinks

true の場合、このグループのユーザーからの接続は、フォールバックリンク機能を使用してリンクを割り当てることができます。これにより、エンドポイントはフォールバックが有効にされているアドレスのフォールバック宛先 (およびソース) として動作します。**false** の場合は、フォールバックリンク機能の使用は許可されません。

sources | sourcePattern

このグループのユーザーがメッセージを受信する AMQP ソースアドレスのリスト。**sources** を使用して、1つ以上のリテラルアドレスを指定します。複数のアドレスを指定するには、コンマ区切りリストを使用します。このグループのユーザーが任意のアドレスからメッセージを受信しないようにするには、この属性を空白のままにしておきます。特定のユーザー固有のアドレスへのアクセスを許可するには、**${user}** トークンを指定します。詳細は、「vhost ポリシーのソースおよびターゲットアドレスを指定する方法」 を参照してください。

または、**sourcePattern** を使用して、パターンに対応する1つ以上のアドレスと一致させることもできます。パターンは、**.** または / のいずれかで区切られた単語シーケンスです。ワイルドカード文字を使用して単語を表すことができます。* 文字は1つの単語にマッチし、**#** 文字はゼロ以上の単語のシーケンスと一致します。

複数のアドレス範囲を指定するには、アドレスパターンのコンマ区切りリストを使用します。詳細は、を参照してください。特定のユーザーに固有の範囲へのアクセスを許可するには、**${user}** トークンを指定します。詳細は、xref:methods-specifying-vhost-policy-source-target-addresses-router-rhel[を参照してください。

targets | targetPattern

このグループのユーザーがメッセージを送信できる AMQP ターゲットアドレスのリスト。複数の AMQP アドレスを指定し、ユーザー名の置換とアドレスパターンをソースアドレスと同じように使用できます。

3. 必要な場合は、vhost ユーザーグループに高度なユーザーグループ設定を追加します。
高度なユーザーグループ設定では、AMQP コネクションのオープン、セッション開始、および接続のリンクアタッチフェーズに基づいて、リソース制限を定義できます。詳細は、man ページの **qdrouterd.conf** で vhost を参照してください。

10.4.3. JSON ファイルとしての vhost ポリシーの作成

ルーター設定ファイルを使用する代わりに、JSON ファイルで vhost ポリシーを設定できます。同じ vhost 設定を共有する必要がある複数のルーターがある場合には、各ルーターがアクセスできる場所に vhost 設定 JSON ファイルを配置することができ、これらの JSON ファイルで定義されている vhost ポリシーを適用するようにルーターを設定することができます。

前提条件

- ルーターに対して vhost ポリシーを有効にする必要があります。詳細は、「vhost ポリシーの有効化」 を参照してください。

手順

1. **/etc/qpid-dispatch/qdrouterd.conf** 設定ファイルで、vhost ポリシー定義 JSON ファイルを保存するディレクトリーを指定します。

```
policy {
    ...
    policyDir: /etc/qpid-dispatch-policies
}
```

policyDir

JSON 形式の vhost ポリシー定義ファイルを保持するディレクトリーへの絶対パス。ルーターは、このディレクトリーにある各 JSON ファイルで、すべての vhost ポリシーを処理します。

2. vhost ポリシー定義ディレクトリーで、各 vhost ポリシーに JSON ファイルを作成します。

例10.1 vhost ポリシー定義 JSON ファイル

```
[
    ["vhost", {
        "hostname": "example.com",
        "maxConnections": 10000,
        "maxConnectionsPerUser": 100,
        "maxConnectionsPerHost": 100,
        "allowUnknownUser": true,
        "groups": {
            "admin": {
                "users": ["admin1", "admin2"],
                "remoteHosts": ["127.0.0.1", "::1"],
                "sources": "*",
                "targets": "*"
            },
            "developers": {
                "users": ["dev1", "dev2", "dev3"],
                "remoteHosts": "*",
                "sources": ["myqueue1", "myqueue2"],
                "targets": ["myqueue1", "myqueue2"]
            },
            "$default": {
                "remoteHosts": "*",
                "allowDynamicSource": true,
                "sources": ["myqueue1", "myqueue2"],
                "targets": ["myqueue1", "myqueue2"]
```

```
        }
      }
    }]
  ]
```

これらの属性についての詳細は、「vhost ポリシーの作成」 を参照してください。

10.4.4. 送信接続のリソース制限の設定

ルーターが外部 AMQP コンテナーへの発信接続を確立する場合 (クライアントやブローカーなど)、外部コンテナーがコネクター vhost ポリシーを設定して、ルーターでアクセスできるリソースを制限することができます。

コネクター vhost ポリシーに定義されているリソース制限は、外部 AMQP コンテナーによって開始されるリンクに適用されます。コネクター vhost ポリシーはルーターが作成するリンクを制限しません。

コネクター vhost ポリシーは、**normal** のロールまたは **route-container** ロールを持つコネクターにのみ適用できます。コネクター vhost ポリシーを、**inter-router** ロールまたは **edge** ロールを持つコネクターに適用することはできません。

前提条件

- ルーターに対して vhost ポリシーを有効にします。詳細は、「vhost ポリシーの有効化」 を参照してください。

手順

1. **/etc/qpid-dispatch/qdrouterd.conf** 設定ファイルで、**$connector** ユーザーグループを指定して **vhost** セクションを追加します。

```
vhost {
    hostname: "my-connector-policy"
    groups: {
        $connector: {
            sources: "*"
            targets: "*"
            maxSenders: 5
            maxReceivers: 10
            allowAnonymousSender: true
            allowWaypointLinks: true
        }
    }
}
```

hostname

コネクター vhost ポリシーを識別する一意の名前。この名前は実際のホスト名を表示しないため、実際の vhost ホスト名と競合しない名前を選択します。

$connector

この vhost ポリシーをコネクター vhost ポリシーとして特定します。適用することのできるリソース制限の詳細は、「vhost ポリシーの作成」 を参照してください。

2. コネクター vhost ポリシーを外部 AMQP コンテナーへの接続を確立するコネクターに適用します。

以下の例では、前のステップで設定したコネクター vhost ポリシーを適用します。

```
connector {
    host: 192.0.2.10
    port: 5672
    role: normal
    policyVhost: my-connector-policy
}
```

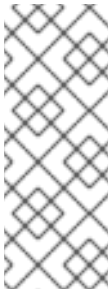
10.4.5. vhost ポリシーのソースおよびターゲットアドレスを指定する方法

vhost で複数のアドレスへのアクセスを許可または拒否する場合は、各アドレスを個別に指定せずに複数のアドレスに一致する方法が複数あります。

以下の表には、vhost ポリシーを使用して複数のソースおよびターゲットアドレスを指定するのに使用できるメソッドをまとめています。

以下を行う場合	以下を行います
ユーザーグループ内のすべてのユーザーがすべてのソースまたはターゲットアドレスにアクセスできるようにする	* ワイルドカード文字を使用してください。 **例10.2 任意のアドレスから受信** `sources: *`
ユーザーグループ内のすべてのユーザーがすべてのソースまたはターゲットアドレスにアクセスできないようにする	値は指定しないでください。 **例10.3 すべてのアドレスへのメッセージ転送の禁止** `targets:`

以下を行う場合	以下を行います

各ユーザーに固有の一部のリソースへのアクセスを許可	**${user}** ユーザー名の置換トークンを使用します。このトークンを **source**、**target**、**sourcePattern**、および **targetPattern** で使用できます。
	
	 
	注記
	AMQP アドレス名またはパターンで **${user}** トークンを一度に指定できます。アドレスに複数のトークンがある場合は、左端のトークンのみが置き換えられます。
	
	例10.4 ユーザー固有のアドレスから受信
	この定義では、ユーザーグループのユーザーは、以下のいずれかのルールを満たすアドレスからメッセージを受信できます。
	● 接頭辞 **tmp_** で開始し、ユーザー名で終わる
	● 接頭辞 **temp** で始まり、その後に追加の文字が続きます。
	● ユーザー名で始まり、**-home-** で始まり、追加の文字で終わります。
	`sources: tmp_${user}, temp*, ${user}-home-*`
	
	例10.5 ユーザー固有のアドレスパターン
	この定義では、ユーザーグループのユーザーは、以下のいずれかのルールを満たすアドレスからメッセージを受信できます。
	● 接頭辞 **tmp** で開始し、ユーザー名で終わる
	● 接頭辞 **temp** で始まりゼロまたはそれ以上の文字が続きます。
	● ユーザー名で始まり、その後に **home** が続き、追加の文字が 1 つ以上終了されます。
	`sourcePattern: tmp.${user}, temp/#, ${user}.home/*`
	
	 
	注記
	アドレスパターン (**sourcePattern** または **targetPattern**) では、ユーザー名置換トークンはパターンの最初のトークンまたは最後のトークンのいずれかである必要があります。トークンは、コンマ区切りのフィールド内でのみ含まれる必要があります。つまり、リテラルテキスト接頭辞または接尾辞で連結できません。

10.4.6. vhost ポリシーホスト名パターン一致ルール

vhost ポリシーでは、ホスト名の範囲をカバーするリテラルホスト名またはパターンのいずれかを使用することができます。

ホスト名パターンとは、以下のワイルドカード文字が 1 つ以上ある単語シーケンスになります。

- * 1 つの単語を表します。
- # ゼロ以上の単語を表します。

以下の表には、ホスト名パターンの例を紹介しています。

このパターン	以下に一致	以下には不一致
***.example.com**	**www.example.com**	**example.comsrv2.www.example.com**
#.example.com	**example.comwww.example.coma.b.c.d.example.com**	**myhost.com**
www.*.test.example.com	**www.a.test.example.com**	**www.test.example.comwww.a.b.c.test.example.com**
www.#.test.example.com	**www.test.example.comwww.a.test.example.comwww.a.b.c.test.example.com**	**test.example.com**

vhost ホスト名パターンの一致は、以下の優先順位ルールを適用します。

ポリシーパターン	優先度
完全一致	高
*	中
#	低

注記

AMQ Interconnect では、既存のパターンと競合する vhost ホスト名パターンを作成できません。これには、既存のパターンと同じ値に削減できるパターンが含まれます。たとえば、**#.com** がすでに存在している場合は、**#.#.#.#.com** パターンを作成しません。

10.4.7. vhost ポリシーの例

以下の例では、vhost ポリシーを使用してメッセージングリソースへのアクセスを承認する方法を示しています。

例10.6 メッセージングエンドポイントの基本的なリソース制限の定義

この例では、vhost ポリシーで、**example.com** ホストに接続するクライアントのリソース制限を定義します。

```
[
   ["vhost", {
      "hostname": "example.com", (1)
      "maxConnectionsPerUser": 10, (2)
      "allowUnknownUser": true, (3)
      "groups": {
         "admin": {
            "users": ["admin1", "admin2"], (4)
            "remoteHosts": ["127.0.0.1", "::1"], (5)
            "sources": "*", (6)
            "targets": "*" (7)
         },
         "$default": {
            "remoteHosts": "*", (8)
            "sources": ["news*", "sports*" "chat*"], (9)
            "targets": "chat*" (10)
         }
      }
   }]
]
```

(1) この vhost ポリシーで定義されたルールは、**example.com** へ接続するユーザーに適用されます。

(2) 各ユーザーは、vhost への最大 10 の接続を開くことができます。

(3) 任意のユーザーはこの vhost に接続できます。**admin** グループの一部ではないユーザーは **$default** グループに割り当てられます。

(4) **admin1** または **admin2** ユーザーが vhost に接続する場合は、**admin** ユーザーグループに割り当てられます。

(5) **admin** ユーザーグループのユーザーは、ローカルホストから接続する必要があります。admin ユーザーが他のホストから接続を試みると、接続は拒否されます。

(6) admin ユーザーグループのユーザーは、任意のアドレスから受信できます。

(7) admin ユーザーグループのユーザーは任意のアドレスに送信できます。

(8) 管理者以外のユーザーは、どのホストからでも接続できるようになります。

(9) 管理者以外のユーザーは、**news**、**sports**、または **chat** 接頭辞で始まるアドレスからメッセージを受信できます。

(10) 管理者以外のユーザーは、**chat** 接頭辞で始まる任意のアドレスにメッセージを送信できます。

例10.7 メモリー消費の制限

高度な vhost ポリシー属性を使用することで、ユーザー接続が消費できるシステムバッファーメモリーを制御することができます。

この例では、株式取引サイトで株式取引先責任者のサービスを提供します。ただし、サイトは high-capacity も受け入れる必要があり、株式エクスチェンジから自動データフィードも受け付ける必要があります。受信側がフィードに必要なメモリーを消費しないようにするため、通信者よりも新しいシステムバッファーメモリーはすべてフィードに割り当てられます。

この例では、**maxSessions** および **maxSessionWindow** 属性を使用して、各 AMQP セッションでバッファーメモリー消費制限を設定します。この設定は AMQP 接続およびセッションネゴシエーションに直接渡され、ルーターで処理サイクルは必要ありません。

この例では、バッファー割り当てに関係しない vhost ポリシー設定は表示されません。

```
[
    ["vhost", {
        "hostname": "traders.com",  1
        "groups": {
            "traders": {
                "users": ["trader1", "trader2"],  2
                "maxFrameSize": 10000,
                "maxSessionWindow": 5000000,  3
                "maxSessions": 1  4
            },
            "feeds": {
                "users": ["nyse-feed", "nasdaq-feed"],  5
                "maxFrameSize": 60000,
                "maxSessionWindow": 1200000000,  6
                "maxSessions": 3  7
            }
        }
    }]
]
```

1. この vhost ポリシーで定義されたルールは、**traders.com** へ接続するすべてのユーザーに適用されます。
2. **traders** グループには、**trader1**、**trader2** で定義されるその他のユーザーが含まれます。
3. 最大 5,000 のデータ量として、各セッションで 5,000 のデータを処理できるようになります。
4. 接続ごとに 1 つのセッションのみが許可されます。
5. **feeds** グループには 2 人のユーザーが含まれます。
6. 大概の 1,200,000,000 バイトのデータは各セッションでフライトになります。
7. 接続ごとに最大 3 つのセッションが許可されます。

第11章 ロギングの設定

AMQ Interconnect には、各ルーターに関する重要な情報を提供する内部ロギングモジュールが含まれています。各モジュールに対して、ロギングレベル、ログファイルのフォーマット、およびログを書き込む場所を設定できます。

11.1. ロギングモジュール

AMQ Interconnect ログは、**ロギングモジュール**と呼ばれるカテゴリーに分けられます。各モジュールは、AMQ Interconnect の特定の側面に関する重要な情報を提供します。

DEFAULT

デフォルトのモジュール。このモジュールは、デフォルトでその他のロギングモジュールすべてに適用されます。

ROUTER

このモジュールは、ローカルルーターに関する情報および統計を提供します。これには、ルーターがネットワークの他のルーターと、ルーターから直接到達可能なリモート宛先に関する情報 (リンクルート、ポイント、自動リンクなど) が含まれます。

ROUTER_HELLO

このモジュールは、Hello メッセージの交換に使用される **Hello** プロトコルについての情報を提供します。これには、ルーターの ID および到達可能な近接リスト (このルーターが双方向接続のある他のルーター) に関する情報が含まれます。

ROUTER_LS

このモジュールには、ルーター広告 (RA)、Link State Request (LSR)、Link State Update (LSU) メッセージなど、ルーター間のリンク状態データに関する情報が提供されます。
定期的に、各ルーターは LSR を他のルーターに送信し、必要な情報で LSU を受信します。上記の情報を試すことで、各ルーターはトポロジー内の次のホップを計算し、関連するコストを計算できます。

ROUTER_MA

このモジュールは、モバイルブロードバンドアドレスリクエスト (MAR) および Mobile Address Update (MAU) メッセージ、ルーター間で交換される Mobile Address Update (MAU) メッセージを含む、ルーター間のモバイルアドレス情報の交換に関する情報を提供します。このログを使用して、各ルーターに接続されたモバイルアドレスの状態を監視することができます。

MESSAGE

このモジュールは、ルーターによって送受信された AMQP メッセージに関する情報を提供します。これには、アドレス、ボディー、リンクに関する情報が含まれます。このログを使用して、特定のルーター上のメッセージの概要情報を見つけることができます。

SERVER

このモジュールは、ルーターがネットワーク上の他のコンテナーをリッスンし、接続する方法 (クライアント、ルーター、ブローカーなど) に関する情報を提供します。この情報には、ブローカーによって送受信された AMQP メッセージの状態 (オープン、開始、割り当て、転送、フローなど)、これらのメッセージの関連するコンテンツが含まれます。

AGENT

このモジュールは、ルーターの設定ファイルの編集または **qdmanage** の使用からルーターに加えられた変更に関する情報を提供します。

CONTAINER

このモジュールは、ルーターに関連するノードに関する情報を提供します。これには、AMQP リレーノードのみが含まれます。

ERROR

このモジュールは、実行中に発生したエラー状態に関する詳細情報を提供します。

POLICY

このモジュールは、ルーターに設定されたポリシーに関する情報を提供します。

関連情報

- これらのロギングモジュールの例は、「ログを使用したトラブルシューティング」 を参照してください。

11.2. デフォルトロギングの設定

ログする必要のあるイベントのタイプ、ログエントリーの形式、およびエントリー送信先を指定できます。

手順

1. **/etc/qpid-dispatch/qdrouterd.conf** 設定ファイルで、**log** セクションを追加してデフォルトのロギングプロパティーを設定します。
この例では、すべてのロギングモジュールを設定して、**info** レベルで開始するイベントをロギングします。

```
log {
    module: DEFAULT
    enable: info+
    includeTimestamp: yes
}
```

module

DEFAULT を指定します。

enable

ログレベル。以下のレベルのいずれかを指定できます (最も低いレベルから高いレベル)。

- **trace**: 最も情報を提供しますが、システムパフォーマンスに大きく影響します。
- **debug**: デバッグに便利ですが、システムパフォーマンスに影響します。
- **info**: システムパフォーマンスに影響を与えずに一般的な情報を提供します。
- **notice**: 一般情報を提供しますが、**info** よりも詳細は少なくなります。
- **warning**: 注意すべき問題に関する情報を提供しますが、エラーではありません。
- **error**: アドレスを指定するエラー
- **critical**: すぐに対応が必要な、重大なシステム問題

複数のレベルを指定するには、コンマ区切りリストを使用します。また、**+** を使用して、レベルとそれ上のすべてのレベルを指定することもできます。たとえば、**trace,debug,warning+** は、トレース、デバッグ、警告、エラー、および重要なレベルを有効にします。デフォルトのロギングでは、通常 **info+** または **notice+** レベルを使用する必要があります。これらのレベルは、AMQ Interconnect のパフォーマンスに影響を与えずに、すべてのモジュールの一般的な情報、警告、およびエラーを提供します。

includeTimestamp

すべてのログにタイムスタンプを追加するには、このパラメーターを **yes** に設定します。

追加のログ属性の詳細は、**logdrouterd.conf** の man ページの ログ を参照してください。

2. ロギングモジュールにデフォルト以外のロギングを設定する場合は、デフォルトに準拠しない各モジュールに追加の **log** セクションを追加します。
この例では、**ROUTER** ロギングモジュールが **debug** イベントを記録するように設定します。

```
log {
    module: ROUTER
    enable: debug
    includeTimestamp: yes
}
```

関連情報

- ログの表示および使用の詳細は、16章*AMQ Interconnect のトラブルシューティング* を参照してください。

第12章 ルーティングの設定

ルーティングとは、メッセージが宛先に配信されるプロセスです。そのため、AMQ Interconnect は**メッセージルーティング**と**リンクルーティング**の 2 つのルーティングメカニズムを提供します。

メッセージルーティング

メッセージルーティングは、デフォルトのルーティングメカニズムです。これにより、メッセージごとにメッセージを直接 (直接ルーティングするメッセージング)、またはブローカーキュー (ブローカー化されたメッセージング) の間でメッセージをルーティングすることができます。

リンクルーティング

リンクルートは、送信元と受信側間のプライベートメッセージングパスを表し、ルーターは終了点間でメッセージを渡します。クライアントをサービス (ブローカーキューなど) に接続するために使用できます。

12.1. メッセージルーティングの設定

メッセージルーティングは、デフォルトのルーティングメカニズムです。これにより、メッセージごとにメッセージを直接 (直接ルーティングするメッセージング)、またはブローカーキュー (ブローカー化されたメッセージング) の間でメッセージをルーティングすることができます。

メッセージルーティングでは、以下を実行できます。

- メッセージルーティングの概念を理解している
- アドレスセマンティクスの設定 (クライアント間のルートメッセージ)
- メッセージ配信の優先順位付けのアドレスを設定
- ブローカー化されたメッセージングの設定
- アドレスパターンの一致について

12.1.1. メッセージルーティングについて

メッセージルーティングにより、プロデューサーがそれらをルーターに送信するため、メッセージに対してルーティングが実行されます。メッセージがルーターに到達すると、ルーターはメッセージの**アドレス**および**ルーティングパターン**に基づいてメッセージとその**セット**をルーティングします。

12.1.1.1. メッセージルーティングのフロー制御

AMQ Interconnect は、**クレジットベース**のフロー制御メカニズムを使用して、1 つ以上のコンシューマーが受信できる場合にのみプロデューサーをルーターに送信できます。AMQ Interconnect はメッセージを保存しないため、この信用ベースのフロー制御により、コンシューマーが存在しない場合にプロデューサーがメッセージを送信できなくなります。

メッセージをルーターに送信するクライアントは、ルーターがクレジットが含まれるまで待機する必要があります。クレジットなしでメッセージをパブリッシュしようとすると、クライアントはブロックします。クレジットが利用可能になると、クライアントはブロックされず、メッセージはルーターに送信されます。

注記

ほとんどの AMQP クライアントライブラリーを使用すると、プロデューサーで利用できるクレジット量を判断できます。詳細は、クライアントのドキュメントを参照してください。

12.1.1.2. アドレス

アドレスは、メッセージをルーターネットワークを通過する方法を決定します。以下のように、メッセージングネットワークのエンドポイントを指定するアドレス。

- データを使用するエンドポイントプロセス、またはサービスを提供するエンドポイントプロセス
- 複数のコンシューマーと複数のプロデューサーに一致するトピック
- メッセージングブローカー内のエンティティー:
  - キュー
  - 永続トピック
  - エクスチェンジ

ルーターがメッセージを受信すると、メッセージのアドレスを使用して、メッセージの送信先を決定します (宛先または 1 つのステップのいずれか、宛先に近いステップのいずれか)。

AMQ Interconnect ではアドレスが**モバイル**として見なれば\、アドレスがルーターネットワーク内のルーターに直接接続されていることや、トポロジーの周りにある場合でもアドレスがモバイルである可能性がある点を考慮します。メッセージを複数のコンシューマーにブロードキャストして、または複数のコンシューマーに分散している場合には、アドレスのユーザーはネットワーク上の複数のルーターに接続する場合があります。

通常のルーター操作中にモバイルアドレスを検出したり、管理設定を介して設定したりできます。

12.1.1.3. ルーティングパターン

ルーティングパターンは、モバイルアドレスを持つメッセージがネットワーク全体にかかるパスを定義します。これらのルーティングパターンは直接ルーティングの両方に使用できます。この場合、ルーターはブローカーと間接ルーティングなしでメッセージを分散します。これにより、ルーターはブローカーを介してメッセージを交換できます。

ルーティングパターンは、anycast (Balanced and Closest) と Multicast の 2 つのカテゴリーに分類されます。アドレスには 1 つのコンシューマーのみがある unicast という概念はありません。

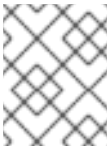
キャストディストリビューションは各メッセージを 1 つのコンシューマーに提供しますが、マルチキャストは各メッセージをすべてのコンシューマーに提供します。

各アドレスには、以下のルーティングパターンの 1 つがあります。これは、アドレスを持つメッセージがメッセージングネットワーク全体に分散できるパスを定義します。

Balanced

複数のコンシューマーが同じアドレスを使用できる任意のキャストメソッド。各メッセージは単一のコンシューマーにのみ提供され、AMQ Interconnect はルーターネットワーク間のトラフィックの負荷を分散しようとします。

複数のコンシューマーが同じアドレスに割り当てられている場合、各ルーターは各パスの現在未設定配信の数を考慮して、どのアウトバウンドパスが存在するかを決定します。これは、配信がより高いレートで設定されるパスと共に配信されることを意味します。

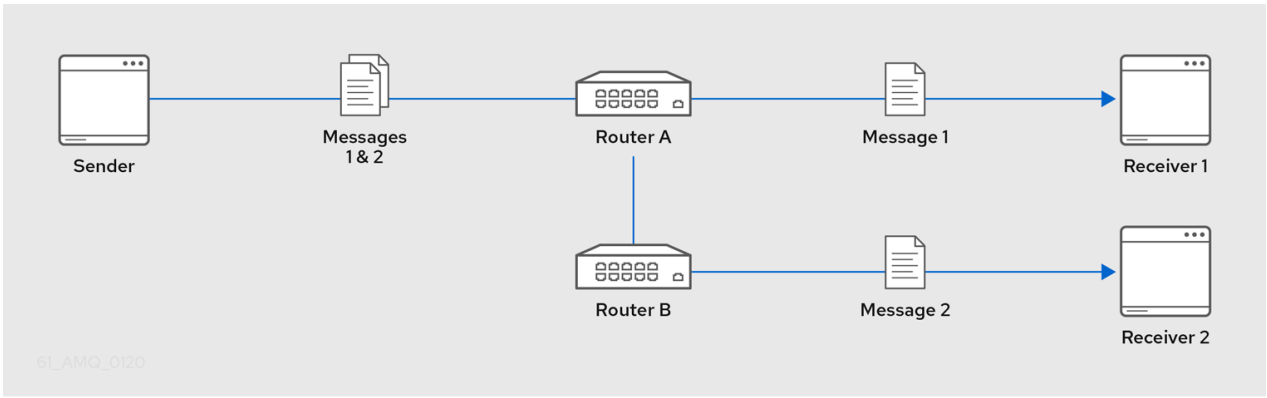


注記

AMQ Interconnect は、メッセージセットを計測して、使用するアウトバウンドパスを決定しません。

このシナリオでは、パスの長さに関係なく、メッセージは両方のレシーバーに分散されます。

図12.1 分散メッセージルーティング

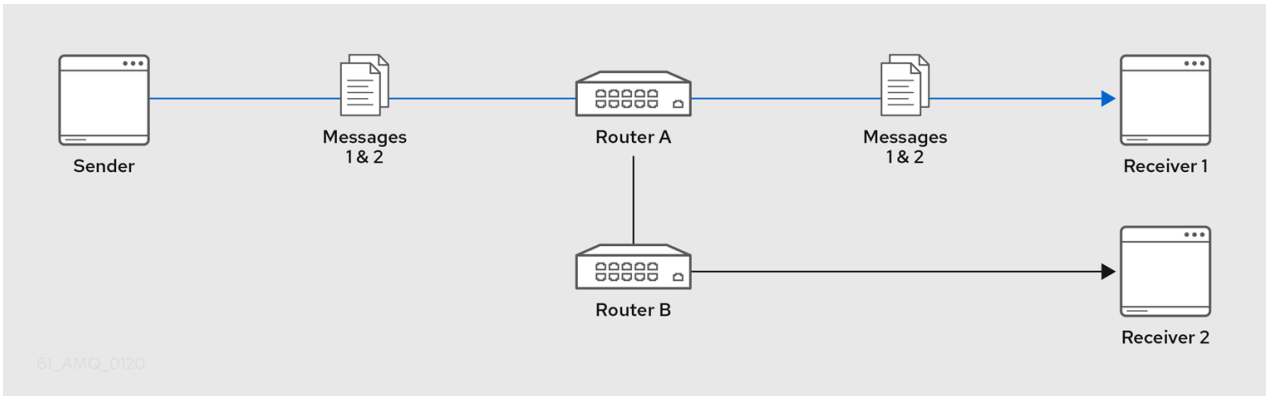


最寄り

同じアドレスに他のコンシューマーがある場合でも、すべてのメッセージが宛先に到達する短いパスと共に送信されるキャストメソッド。
AMQ Interconnect は、各コンシューマーに到達できるように、トポロジーコストに基づいて最も短いパスを決定します。同じコストが最も低いコンシューマーが複数ある場合、メッセージはこれらのコンシューマー間で均等に分散されます。

このシナリオでは、**Sender** によって送信されたすべてのメッセージが **Receiver 1** に送信されます。

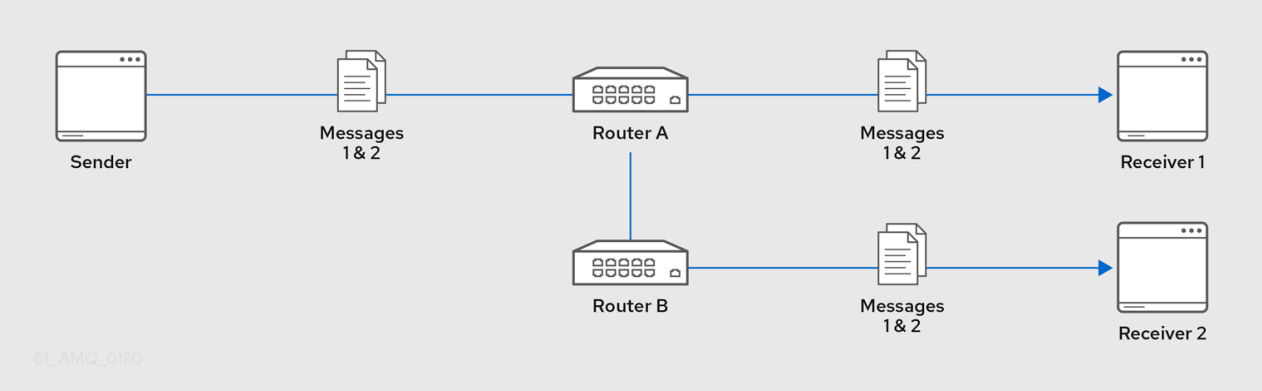
図12.2 最寄りのメッセージルーティング



Multicast

メッセージはアドレスにアタッチされたすべてのコンシューマーに送信されます。各コンシューマーはメッセージの 1 つのコピーを受け取ります。
このシナリオでは、すべてのメッセージがすべての受信側に送信されます。

図12.3 マルチキャストメッセージルーティング



12.1.1.4. メッセージセットおよび信頼性

AMQ Interconnect では、以下の信頼性でメッセージを配信できます。

- 少なくとも 1 回
- 少なくとも 1 回
- 一度だけ

信頼性のレベルは、プロデューサーがルーターへのリンクを確立する際に、プロデューサーとルーター間でネゴシエートされます。信頼性のレベルをネゴシエートするために、AMQ Interconnect はすべてのメッセージを**事前設定**または**未設定**として扱います。

事前設定済み

fire および forget という名称は、ルーターセットで受信および送信の配信が可能で、セットがメッセージの送信先に伝播されます。ただし、配信を保証しているわけではありません。

未設定

AMQ Interconnect は、プロデューサーとコンシューマーの間でセットを伝播します。任意のキャストアドレスの場合、ルーターは受信配信を作成された送信配信に関連付けます。この関連付けに基づいて、ルーターはコンシューマーからプロデューサーに配信状態の変更を伝播します。
マルチキャストアドレスの場合は、ルーターは受信配信をすべてアウトバウンド配信に関連付けます。ルーターは、各コンシューマーが配信の最終状態を設定するのを待機します。すべての送信配信が最終状態になった後に、ルーターは元のインバウンド配信の最終配信状態を維持し、プロデューサーに渡します。

以下の表は、すべてのキャストまたはマルチキャストアドレスに送信される未設定メッセージに対する信頼性の保証を示しています。

最終的な配置	anycast	マルチキャスト
accepted	コンシューマーはメッセージを受け入れました。	少なくとも 1 つのコンシューマーはメッセージを受け入れますが、コンシューマーは拒否しません。
released	メッセージはその宛先に到達しませんでした。	このメッセージはコンシューマーのいずれかに到達しませんでした。

最終的な配置	anycast	マルチキャスト
modified	メッセージはその送信先に達した場合もあります。この配信はインダウト状態のであると見なされ、1 回以上配送が必要な場合に再送信される必要があります。	このメッセージはコンシューマーのいずれの場合でも、アクセスされなかった場合もあります。ただし、コンシューマーが拒否または受け入れられません。
rejected	コンシューマーはメッセージを拒否しました。	少なくとも 1 つのコンシューマーはメッセージを拒否しました。

12.1.2. アドレスセマンティクスの設定

ブローカーを使用せずにクライアント間でメッセージをルーティングできます。ブローカーレスシナリオ (**direct-routed メッセージングと呼ばれることもあります**) では、AMQ Interconnect はクライアント間でメッセージを直接ルーティングします。

クライアント間でメッセージをルーティングするには、ルーティング分散パターンでアドレスを設定します。ルーターがこのアドレスを持つメッセージを受信すると、メッセージはアドレスのルーティング分散パターンに基づいて宛先または宛先にルーティングされます。

手順

1. **/etc/qpid-dispatch/qdrouterd.conf** 設定ファイルで **address** セクションを追加します。

```
address {
    prefix: my_address
    distribution: multicast
    ...
}
```

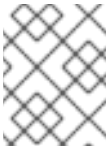
prefix | pattern

アドレス設定を適用するアドレスのアドレスまたはグループ。接頭辞を指定して、アドレスの正確なアドレスまたは開始セグメントと一致するように指定できます。または、ワイルドカードを使用してアドレスに一致するパターンを指定できます。
接頭辞は、**.** または / 文字のいずれかで区切られたアドレス内の正確なアドレスまたは最初のセグメントと一致します。たとえば、接頭辞 **my_address** は、**my_address** と **my_address.1** および **my_address/1** のアドレスと一致します。ただし、**my_address1** と一致しません。

パターンは、パターンに対応するアドレスと一致します。パターンは、**.** または / のいずれかで区切られた単語シーケンスです。ワイルドカード文字を使用して単語を表すことができます。* 文字は 1 つの単語にマッチし、**#** 文字はゼロ以上の単語のシーケンスと一致します。

* および **#** 文字はワイルドカードとして予約されています。したがって、メッセージアドレスでは使用しないでください。

アドレスパターンの作成に関する詳細は、「アドレスパターンの一致」 を参照してください。



注記

prefix 値を **pattern** に変換するには、**/#** をそれに追加します。たとえば、接頭辞 **a/b/c** は、パターン **a/b/c/#** と同等です。

distribution

メッセージディストリビューションパターン。デフォルトは **balanced** ですが、以下のオプションのいずれかを指定できます。

- **balanced**: アドレスに送信されたメッセージは受信側のいずれかにルーティングされ、ルーティングネットワークはセットレートに基づいてトラフィックの負荷のバランスを取ります。
- **closest**: アドレスに送信されたメッセージは、送信先に到達するために最短パスで送信されます。つまり、同じアドレスに複数の受信側がある場合は、最も近いもののみがメッセージを受信します。
- **multicast**: メッセージは**パブリッシュ/サブスクライブ**モデルのアドレスにアタッチされるすべてのレシーバーに送信されます。
  メッセージディストリビューションパターンに関する詳細は、「ルーティングパターン」 を参照してください。

その他の属性の詳細は、**qdrouterd.conf** の man ページの address を参照してください。

2. アドレスを使用する必要のある他のルーターに同じ **address** セクションを追加します。
   このルーター設定ファイルに追加した **address** は、このルーターがアドレスに送信されるメッセージを配布する方法のみ制御します。このアドレスにメッセージを分散する必要のあるルーターネットワークに追加のルーターがある場合、設定ファイルに同じ **address** セクションを追加する必要があります。

12.1.3. メッセージ配信の優先順位付けのアドレス設定

アドレスの優先度レベルを設定して、AMQ Interconnect がそのアドレスに送信されるメッセージを処理する方法を制御できます。AMQ Interconnect はコネクションの範囲内で、優先度に基づいてメッセージを処理しようとします。大量のメッセージを処理している接続の場合、優先度の高いメッセージのレイテンシーが短縮されます。

アドレスに高優先度の高いレベルを割り当てると、アドレスに送信されたメッセージが優先度の低いアドレスに送信されることを保証する訳ではありません。ただし、優先順位の高いメッセージは、ルーターネットワークを通じてより迅速に移動します。

注記

メッセージヘッダーに優先度レベルを設定することで、個々のメッセージの優先度レベルを制御することもできます。ただし、アドレスの優先度が優先されます。優先順位を異なるアドレスに優先したメッセージを送信した場合、ルーターはアドレスの優先度レベルを使用します。

手順

- **/etc/qpid-dispatch/qdrouterd.conf** 設定ファイルでアドレスを追加または編集し、優先度レベルを割り当てます。
  この例では、最も優先度の高いアドレスを追加します。ルーターは、優先度が低いメッセージの前に、このアドレスに送信されたメッセージの配信を試みます。

```
address {
    prefix: my-high-priority-address
    priority: 9
    ...
}
```

priority

このアドレスに送信されたすべてのメッセージに割り当てる優先度レベル。有効な優先度のレベルの範囲は 0-9 で、数字が大きいほど優先度が高くなります。デフォルトでは 4 回です。

関連情報

- メッセージに優先順位レベルを設定する方法は、AMQP 1.0 仕様 を参照してください。

12.1.4. ブローカー化されたメッセージングの設定

ストアおよび転送機能が必要な場合は、AMQ Interconnect をブローカー化されたメッセージングを使用するように設定できます。このシナリオでは、クライアントはルーターに接続してメッセージを送受信し、ルーターはメッセージブローカーのキューとの間でメッセージを送受信します。

以下を設定できます。

- ブローカーキューによるメッセージのルーティング
  メッセージを単一ブローカーでホストされるキューにルーティングしたり、複数のブローカーに分散した**シャードキュー**にメッセージをルーティングしたりできます。

- 非配信可能なメッセージをブローカーキューに保存および取得

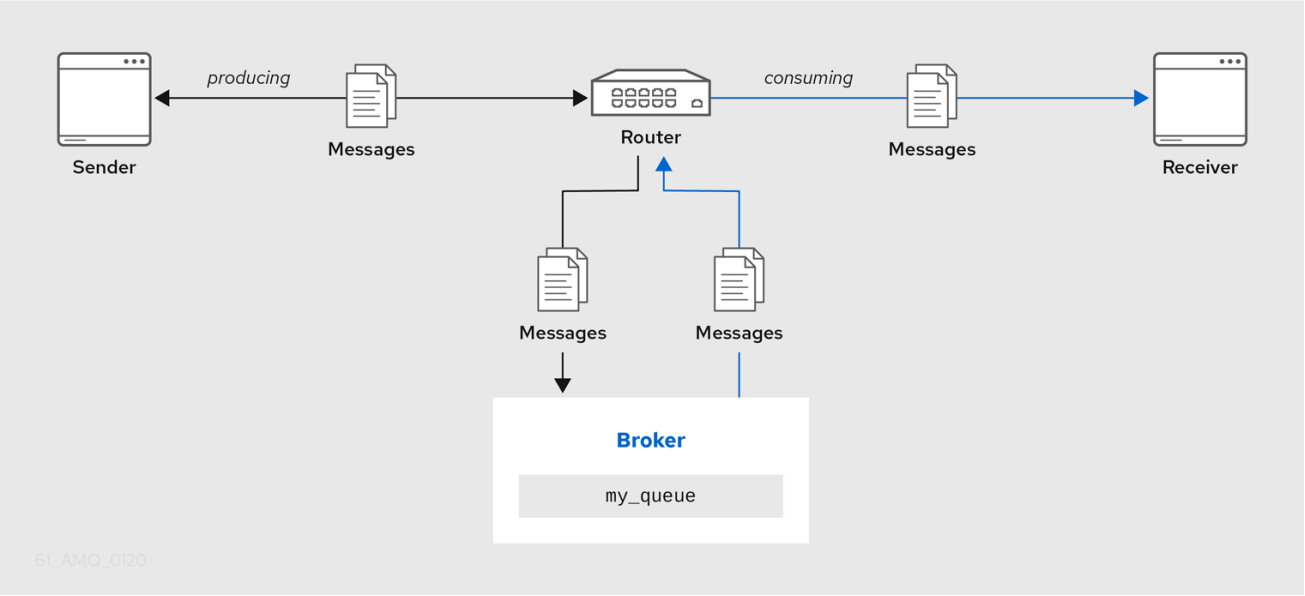
12.1.4.1. AMQ Interconnect によるブローカー化されたメッセージングの有効化方法

ブローカー化されたメッセージングにより、AMQ Interconnect はメッセージをブローカーキューに保存できます。これには、ブローカーへの接続、ブローカーキューを表すための方法指定アドレス、および**ウェイポイント**アドレスにアタッチする**自動リンク**が必要です。

オートリンクは、ルーターポイントアドレスにアタッチするためにルーターによって自動作成されるリンクです。自動リンクでは、クライアントトラフィックはブローカーではなく、ルーターで処理されます。クライアントはルーターにリンクをアタッチし、ルーターは内部の自動リンクを使用してブローカーのキューに接続します。そのため、ルーターに接続されているクライアントの数にかかわらず、キューは常に単一のプロデューサーと単一のコンシューマーを持ちます。

リンクルーティングとは異なり、自動リンクの使用は**メッセージルーティング**の形式です。コンシューマーに関連するセマンティクス (例: **undeliverable-here=true** 変更済み配信状態) を使用する場合は、リンクルーティングを使用することが推奨されます。

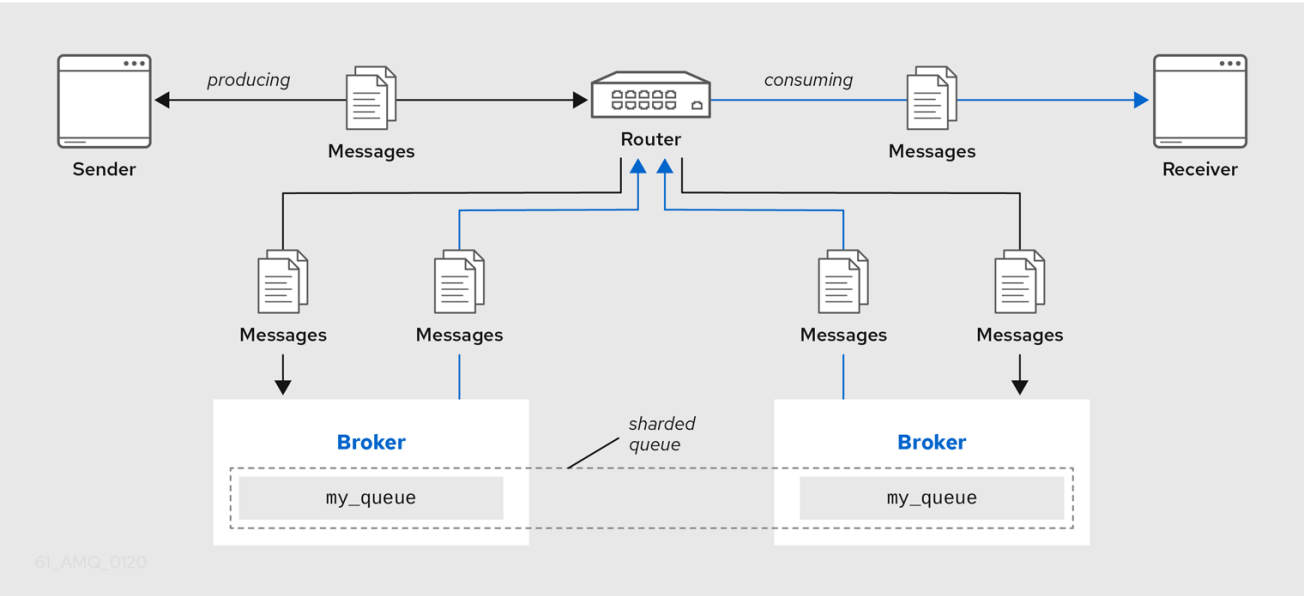
図12.4 ブローカー化されたメッセージング



この図では、送信側はルーターに接続し、my_queue にメッセージを送信します。ルーターは、ブローカーに送信リンクを接続し、そのメッセージを my_queue に送信します。後に、受信側はルーターに接続し、my_queue からメッセージを要求します。ルーターは、受信リンクをブローカーに割り当て、my_queue からメッセージを受信し、それらを受信側に提供します。

また、複数の基礎となる物理キューで設定される単一の論理キューである**シャードされたキュー**にメッセージをルーティングすることもできます。キューのシャーディングを使用すると、複数のブローカーで単一のキューを分散できます。クライアントは、シャードを保持するブローカーに接続してメッセージを送受信できます。

図12.5 シャードキューのあるブローカーメッセージング



この図では、シャーディングされたキュー (my_queue) は 2 つのブローカーに分散されます。ルーターはクライアントと両方のブローカーに接続されている必要があります。送信側はルーターに接続し、そのメッセージを my_queue に送信します。ルーターは各ブローカーに送信リンクを割り当て、各シャードにメッセージを送信します (デフォルトではルーティングの分散は **balanced** です)。後に、受信側はルーターに接続し、my_queue からすべてのメッセージを要求します。ルーターは、受信リンクをブローカーのいずれかに割り当て、my_queue からメッセージを受信し、それらを受信側に提供します。

12.1.4.2. ブローカーキューによるメッセージのルーティング

ブローカーキューとの間でメッセージをルーティングし、ルーター経由でキューにアクセスできるクライアントを提供できます。このシナリオでは、クライアントはルーターに接続してメッセージを送受信し、ルーターはブローカーキューとの間でメッセージを送受信します。

メッセージを単一ブローカーでホストされるキューにルーティングしたり、複数のブローカーに分散した**シャードキュー**にメッセージをルーティングしたりできます。

手順

1. **/etc/qpid-dispatch/qdrouterd.conf** 設定ファイルで、ブローカーキューのを追加します。
ポイントポイントアドレスは、メッセージをルーティングするブローカーのキューを識別します。以下の例では、**my_queue** キューのポイントアドレスを追加します。

```
address {
    prefix: my_queue
    waypoint: yes
}
```

prefix | pattern

メッセージの送信先のブローカーキューに一致するアドレス接頭辞またはパターン。接頭辞を指定して、アドレスの正確なアドレスまたは開始セグメントと一致するように指定できます。または、ワイルドカードを使用してアドレスに一致するパターンを指定できます。
接頭辞は、**.** または / 文字のいずれかで区切られたアドレス内の正確なアドレスまたは最初のセグメントと一致します。たとえば、接頭辞 **my_address** は、**my_address** と **my_address.1** および **my_address/1** のアドレスと一致します。ただし、**my_address1** と一致しません。

パターンは、パターンに対応するアドレスと一致します。パターンは、**.** または / のいずれかで区切られた単語シーケンスです。ワイルドカード文字を使用して単語を表すことができます。* 文字は 1 つの単語にマッチし、**#** 文字はゼロ以上の単語のシーケンスと一致します。

* および **#** 文字はワイルドカードとして予約されています。したがって、メッセージアドレスでは使用しないでください。

アドレスパターンの作成に関する詳細は、「アドレスパターンの一致」 を参照してください。

注記

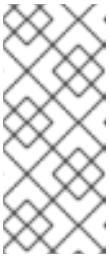
prefix 値を **pattern** に変換するには 、**/#** をそれに追加します。たとえば、接頭辞 **a/b/c** は、パターン **a/b/c/#** と同等です。

waypoint

ルーターがこのアドレスに送信されたメッセージをポイントとして処理するように、この属性を **yes** に設定します。

2. ルーターをブローカーに接続します。

a. ブローカーに外向き接続がない場合は、その接続を追加します。
キューが複数のブローカーにシャードされている場合は、各ブローカーの接続を追加する必要があります。詳細は、「外部 AMQP コンテナーへの接続」 を参照してください。



注記

ブローカーへの接続に失敗すると、AMQ Interconnect は自動的に接続を再確立し、メッセージ配信を利用可能な別の宛先に再ルーティングしようとします。ただし、一部の配信は **RELEASED** または **MODIFIED** で送信側に返される場合があります。したがって、クライアントがこれらの配信を適切に処理できるようにする必要があります (通常は再送することです)。

b. メッセージをブローカーキューに送信する場合は、**送信**自動リンクをブローカーキューに追加します。
キューが複数のブローカーにシャードされている場合は、各ブローカーに出力自動リンクを追加する必要があります。

この例では、送信自動リンクを設定し、メッセージをブローカーキューに送信します。

```
autoLink {
    address: my_queue
    connection: my_broker
    direction: out
    ...
}
```

address

ブローカーキューのアドレス。自動リンクが作成されると、このアドレスに割り当てられます。

externalAddress

ブローカーキューのオプションの代替アドレス。ブローカーキューが送信者が使用するアドレスとは別のアドレスを持つ必要がある場合、外部アドレスを使用します。このシナリオでは、送信側は **address** アドレスにメッセージを送信し、ルーターはそれらを **externalAddress** アドレスで表されるブローカーキューにルーティングします。

connection | containerID

ルーターによるブローカーへの接続方法。発信接続 (**connection**) またはブローカーのコンテナー ID (**containerID**) のいずれかを指定できます。

direction

この属性を **out** に設定して、この自動リンクがルーターからブローカーにメッセージを送信できるように指定します。

その他の属性の詳細は、**qdrouterd.conf** の man ページの autoLink を参照してください。

3. ブローカーキューからメッセージを受信したい場合は、ブローカーキューから**受信**自動リンクを追加します。
キューが複数のブローカーにシャードされている場合は、各ブローカーに出力自動リンクを追加する必要があります。

この例では、受信自動リンクを設定し、ブローカーキューからメッセージを受信します。

```
autoLink {
    address: my_queue
    connection: my_broker
    direction: in
    ...
}
```

address

ブローカーキューのアドレス。自動リンクが作成されると、このアドレスに割り当てられます。

externalAddress

ブローカーキューのオプションの代替アドレス。ブローカーキューが受信者が使用するアドレスとは別のアドレスを持つ必要がある場合、外部アドレスを使用します。このシナリオでは、受信側は **address** アドレスからメッセージを受信し、ルーターは **externalAddress** アドレスで表されるブローカーキューからメッセージを取得します。

connection | containerID

ルーターによるブローカーへの接続方法。発信接続 (**connection**) またはブローカーのコンテナー ID (**containerID**) のいずれかを指定できます。

direction

この自動リンクがブローカーからルーターにメッセージを送受信するように指定するには、この属性を **in** に設定します。

その他の属性の詳細は、**qdrouterd.conf** の man ページの autoLink を参照してください。

12.1.4.3. 再配信不可能なメッセージの処理

フォールバック先をポイントする autolinks を設定して、アドレスの配信不可能なメッセージを処理します。フォールバック先 (ブローカーのキューなど) は、どのコンシューマーにも直接ルーティングできないメッセージを保存します。

通常のメッセージの配信中に、AMQ Interconnect はルーターネットワークに接続されているコンシューマーにメッセージを提供します。ただし、コンシューマーに到達できない場合、メッセージはアドレスに設定されたフォールバック宛先に迂回されます (フォールバック先を参照する自動リンクがアクティブである場合)。コンシューマーを再接続し、再度到達すると、フォールバック先に保存されているメッセージを受信します。

注記

AMQ Interconnect は、フォールバック先に保存されたメッセージの元の配信順序を保持します。ただし、コンシューマーが再接続すると、キューのドレインの間に作成された新しいメッセージが、フォールバック先に保存されるメッセージと干渉されます。

前提条件

- ルーターはブローカーに接続されている。
  詳細は、「外部 AMQP コンテナーへの接続」 を参照してください。

手順

この手順では、アドレスのフォールバックを有効にし、アドレスのフォールバック宛先を提供するブローカーキューに接続するように自動リンクを設定します。

1. **/etc/qpid-dispatch/qdrouterd.conf** 設定ファイルで、アドレスのフォールバック宛先を有効にします。

```
address {
    prefix: my_address
    enableFallback: yes
}
```

2. **発信**自動リンクをブローカーのキューに追加します。
フォールバックを有効にしたアドレスで、メッセージがコンシューマーにルーティング可能でない場合、ルーターはこの自動リンクを使用してメッセージをブローカーのキューに送信します。

```
autoLink {
    address: my_address.2
    direction: out
    connection: my_broker
    fallback: yes
}
```

3. ルーターネットワークに接続した直後に、ルーターがキューに置かれたメッセージを接続させる場合は、**受信**自動リンクを追加します。
コンシューマーがルーターに接続するとすぐに、ブローカーキューに保管されたメッセージと、プロデューサーによって送信される新しいメッセージを受信します。キューに格納されたメッセージの元の配信順序は保持されますが、キューに置かれたメッセージは新しいメッセージによってインターリーブされます。

   受信自動リンクを追加しない場合、メッセージはブローカーに保存されますが、ルーターに割り当てるとコンシューマーには送信されません。

```
autoLink {
    address: my_address.2
    direction: in
    connection: my_broker
    fallback: yes
}
```

12.1.5. アドレスパターンの一致

一部のルーター設定のシナリオでは、1つのリテラルアドレスではなく、アドレスの範囲と一致するようにパターンの一致を使用する必要がある場合があります。アドレスのパターンは、パターンに対応するアドレスに一致します。

アドレスパターンは、トークン (通常は 単語) のシーケンスで、**.** または / で区切られています。また、単語を表す特別なワイルドカード文字を含めることもできます。

- ***** 1つの単語を表します。
- **#** ゼロ以上の単語を表します。

例12.1 アドレスパターン

このアドレスには、/ 区切り文字で区切られた 2 つのトークンが含まれます。

my/address

例12.2 ワイルドカードを使用したアドレスパターン

このアドレスには 3 つのトークンが含まれます。* はワイルドカードで、**my** および **address** の間にある単一の単語を表します。

> **my/*/address**

以下の表は、アドレスパターンと、一致するアドレスの例を示しています。

このパターン	以下に一致	以下には不一致
news/*	**news/europe** **news/usa**	**news** **news/usa/sports**
news/#	**news** **news/europe** **news/usa/sports**	**europe** **usa**
news/europe/#	**news/europe** **news/europe/sports** **news/europe/politics/fr**	**news/usa** **europe**
news/*/sports	**news/europe/sports** **news/usa/sports**	**news** **news/europe/fr/sports**

12.2. リンクルートの作成

リンクルートは、送信元と受信側間のプライベートメッセージングパスを表し、ルーターは終了点間でメッセージを渡します。クライアントをサービス (ブローカーキューなど) に接続するために使用できます。

12.2.1. リンクルーティングについて

リンクルーティングは、ブローカー化されたメッセージングの代替ストラテジーを提供します。リンクルートは、送信元と受信側間のプライベートメッセージングパスを表し、ルーターは終了点間でメッセージを渡します。リンクルートは、ルーターネットワークを介して送信元から受信側を通過する仮想接続または tunnel と考えることができます。

リンクルーティングにより、ルーティングはリンク接続フレームで実行され、送信者および受信側を直接接続する仮想メッセージングパスを形成します。リンクルートが確立されると、メッセージの配信、フローフレーム、破棄の転送がリンクルート全体で実行されます。

12.2.1.1. リンクルーティングのフロー制御

リンクルーティングとは異なり、送信側および受信側はフロー制御を直接処理します。受信側は、受信できるメッセージの数であるリンククレジットを付与します。ルーターは送信側に直接送信し、受信側が付与されたクレジットに基づいてメッセージを送信します。

12.2.1.2. リンクルートアドレス

リンクルートアドレスは、ブローカーキュー、トピック、またはその他のサービスを表します。クライアントがリンクルートアドレスをルーターに割り当てると、ルーターはアドレスで識別されるブローカーリソースへのリンクを伝播します。

リンクルートアドレスを使用して、ルーターネットワークは集約されたメッセージ分散に参加しません。ルーターは単に 2 つの終了ポイント間でメッセージの配信とセットを渡します。

12.2.1.3. リンクルーティングのルーティングパターン

ルーティングパターンは、送信者と受信者との間に直接リンクがあるため、リンクルーティングとは使用されません。ルーターは、最初のリンク接続要求フレームを受信する場合にのみルーティングの決定を行います。リンクが確立されたら、ルーターは分散したディストリビューションのリンクとメッセージを渡します。

12.2.2. リンクルートの作成

リンクルートは、送信元とルーターを通過する受信側間のリンクを確立します。着信リンクと発信リンクルートを設定し、ルーターがクライアントからリンク接続を受信し、特定の宛先に送信することができます。

リンクルーティングでは、クライアントトラフィックはルーターではなくブローカーで処理されます。クライアントには、ルーターからブローカーのキューへの直接のリンクがあります。そのため、各クライアントは個別のプロデューサーまたはコンシューマーです。

注記

ブローカーへの接続に失敗すると、ルーティングされたリンクがデタッチされ、ルーターはブローカー (またはバックアップ) への再接続を試みます。接続が再確立されると、ブローカーへのリンクルートが再度到達可能になります。

クライアントの視点からは、クライアントは切り離されたリンク (送信者またはレシーバー) を認識しますが、失敗した接続ではありません。したがって、ブローカー接続障害が発生した場合に、クライアントがリンクを再アタッチする場合は、クライアントでこの機能を設定する必要があります。または、リンクルーティングの代わりに、自動リンクでメッセージルーティングを使用できます。詳細は、「ブローカーキューによるメッセージのルーティング」 を参照してください。

手順

1. ブローカーに外向き接続がない場合は、その接続を追加します。
   キューが複数のブローカーにシャードされている場合は、各ブローカーの接続を追加する必要があります。詳細は、「外部 AMQP コンテナーへの接続」 を参照してください。

2. クライアントがローカルトランザクションをブローカーに送信する場合は、トランザクションコーディネーターのリンクルートを作成します。

```
linkRoute {
    prefix: $coordinator (1)
    connection: my_broker
    direction: in
}
```

(1) **$coordinator** 接頭辞は、このリンクルートをトランザクションコーディネーターとして指定します。クライアントがトランザクションセッションを開くと、トランザクションの開始および終了リクエストは、ブローカーへのこのリンクルートと共に伝播されます。

AMQ Interconnect では、複数のブローカーへのトランザクションのルーティングはサポートされません。環境に複数のブローカーがある場合は、単一のブローカーを選択し、すべてのトランザクションをそのブローカーにルーティングします。

3. クライアントがこのリンクルートにメッセージを送信するには、受信リンクルートを作成します。

```
linkRoute {
    prefix: my_queue
    connection: my_broker
    direction: in
    ...
}
```

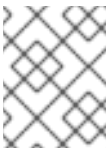
prefix | pattern

ルーティングされたリンク添付の宛先となるブローカーキューに一致するアドレス接頭辞またはパターン。この接頭辞またはパターンに一致するメッセージはすべて、リンクルートとともに分散されます。接頭辞を指定して、アドレスの正確なアドレスまたは開始セグメントと一致するように指定できます。または、ワイルドカードを使用してアドレスに一致するパターンを指定できます。
接頭辞は、**.** または / 文字のいずれかで区切られたアドレス内の正確なアドレスまたは最初のセグメントと一致します。たとえば、接頭辞 **my_address** は、**my_address** と **my_address.1** および **my_address/1** のアドレスと一致します。ただし、**my_address1** と一致しません。

パターンは、パターンに対応するアドレスと一致します。パターンは、**.** または / のいずれかで区切られた単語シーケンスです。ワイルドカード文字を使用して単語を表すことができます。***** 文字は 1 つの単語にマッチし、**#** 文字はゼロ以上の単語のシーケンスと一致します。

***** および **#** 文字はワイルドカードとして予約されています。したがって、メッセージアドレスでは使用しないでください。

アドレスパターンの作成に関する詳細は、「アドレスパターンの一致」 を参照してください。



注記

prefix 値を **pattern** に変換するには 、**/#** をそれに追加します。たとえば、接頭辞 **a/b/c** は、パターン **a/b/c/#** と同等です。

connection | containerID

ルーターによるブローカーへの接続方法。発信接続 (**connection**) またはブローカーのコンテナー ID (**containerID**) のいずれかを指定できます。
この接続で複数のブローカーがルーターに接続されている場合、リンクルートの接頭辞またはパターンに一致するアドレスの要求がブローカー全体で分散されます。または、特定のブローカーを指定する場合は、**containerID** を使用し、ブローカーのコンテナー ID を追加します。

direction

このリンクルート上のルーターネットワークにメッセージを送信するように、この属性を **in** に設定します。

その他の属性の詳細は、**qdrouterd.conf** の man ページの linkRoute を参照してください。

4. クライアントがこのリンクルート上のメッセージを受信するには、出力リンクルートを作成します。

```
linkRoute {
    prefix: my_queue
    connection: my_broker
    direction: out
    ...
}
```

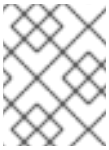
prefix | pattern

ルーティングされたリンク割り当てを受信するブローカーキューに一致するアドレス接頭辞またはパターン。この接頭辞またはパターンに一致するメッセージはすべて、リンクルートとともに分散されます。接頭辞を指定して、アドレスの正確なアドレスまたは開始セグメントと一致するように指定できます。または、ワイルドカードを使用してアドレスに一致するパターンを指定できます。
接頭辞は、**.** または / 文字のいずれかで区切られたアドレス内の正確なアドレスまたは最初のセグメントと一致します。たとえば、接頭辞 **my_address** は、**my_address** と **my_address.1** および **my_address/1** のアドレスと一致します。ただし、**my_address1** と一致しません。

パターンは、パターンに対応するアドレスと一致します。パターンは、**.** または / のいずれかで区切られた単語シーケンスです。ワイルドカード文字を使用して単語を表すことができます。***** 文字は 1 つの単語にマッチし、**#** 文字はゼロ以上の単語のシーケンスと一致します。

***** および **#** 文字はワイルドカードとして予約されています。したがって、メッセージアドレスでは使用しないでください。

アドレスパターンの作成に関する詳細は、「アドレスパターンの一致」 を参照してください。



注記

prefix 値を **pattern** に変換するには 、**/#** をそれに追加します。たとえば、接頭辞 **a/b/c** は、パターン **a/b/c/#** と同等です。

connection | containerID

ルーターによるブローカーへの接続方法。発信接続 (**connection**) またはブローカーのコンテナー ID (**containerID**) のいずれかを指定できます。
この接続で複数のブローカーがルーターに接続されている場合、リンクルートの接頭辞またはパターンに一致するアドレスの要求がブローカー全体で分散されます。または、特定のブローカーを指定する場合は、**containerID** を使用し、ブローカーのコンテナー ID を追加します。

direction

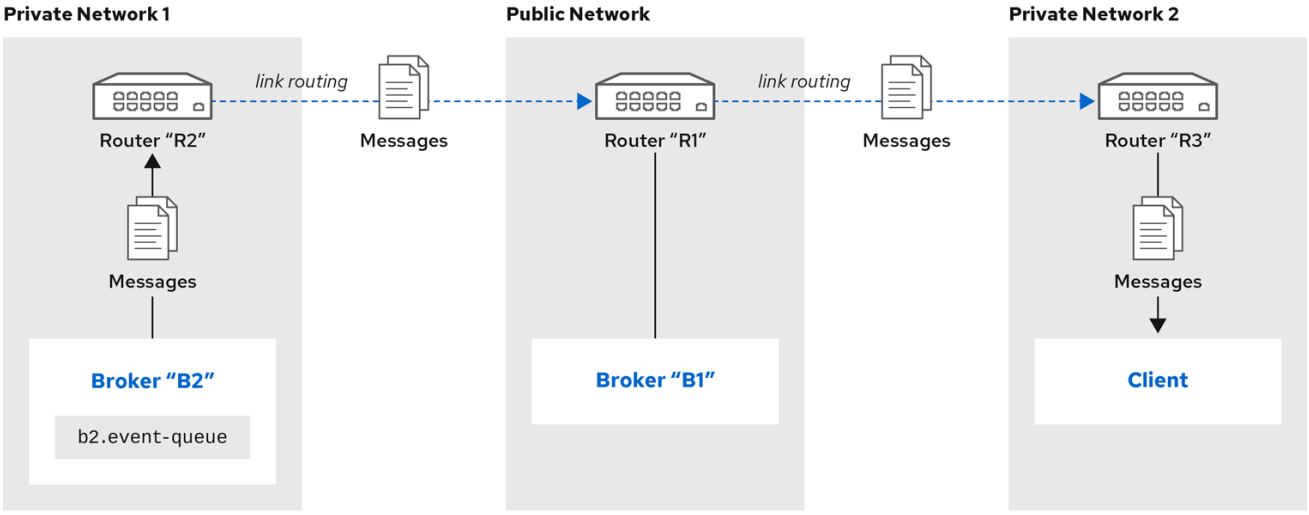
このリンクルートが受信側用であることを指定するには、この属性を **out** に設定します。

その他の属性の詳細は、**qdrouterd.conf** の man ページの linkRoute を参照してください。

12.2.3. リンクルートの例: 異なるネットワーク上でのクライアントおよびブローカーの接続

この例は、リンクルートがクライアントを別のプライベートネットワークにあるメッセージブローカーに接続できる方法を示しています。

図12.6 分離されたクライアントを使用するルーターネットワーク



クライアントは、独自のネットワーク (**R3**) でルーターに接続するためにファイアウォールポリシーによって制限されます。ただし、リンクルートを使用して、異なるネットワーク上にある場合でも、メッセージブローカー **B1** および **B2** を指定した他の AMQP サービスにアクセスできます。

この例では、クライアントは **Private Network 1** のブローカー **B2** でホストされる **b2.event-queue** からメッセージを受信する必要があります。リンクルートは、クライアントとサーバー間にルーターネットワークがあることを認識していても、クライアントとブローカーを接続する。

ルーターの設定
クライアントがブローカー **B2** で **b2.event-queue** からメッセージを受信できるようにするには、ルーター **R2** が以下を実行できる必要があります。

- ブローカー **B2** への接続
- ブローカー **B2** 間のルートリンク
- **b2.event-queue** アドレスを持つリンクの有効な宛先として、ルーターネットワークにアドバタイズします。

ルーター **R2** の設定ファイルの関連する部分で、以下が表示されます。

```
connector { (1)
    name: broker
    role: route-container
    host: 192.0.2.1
    port: 61617
    saslMechanisms: ANONYMOUS
}

linkRoute { (2)
    prefix: b2
    direction: in
    connection: broker
}
```

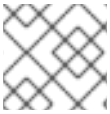
```
linkRoute { 3
    prefix: b2
    direction: out
    connection: broker
}
```

1. ルーターからブローカー **B2** への外向き接続。**route-container** ロールにより、ルーターは外部 AMQP コンテナー (この場合はブローカー) に接続できるようになります。

2. クライアント送信者からリンクを受信するための受信リンクルート。**b2** で始まるアドレスがブローカー **B2** で始まるターゲットを持つ送信者は、**broker** コネクターを使用してブローカー B2 にルーティングされます。

3. クライアントレシーバーへのリンクを送信するための出力リンクルート。ソースアドレスが **b2** で始まるレシーバーは、**broker** コネクターを使用してブローカー **B2** にルーティングされます。

この設定では、ルーター **R2** は、**b2** で始まるターゲットおよびソースの有効な宛先としてアドバタイズできます。また、ルーターはブローカー **B2** に接続でき、**b2** 接頭辞で始まるキューとキューからリンクをルーティングできるようにします。



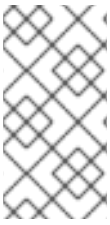
注記

必須ではありませんが、ルーター **R1** と **R3** でも同じ設定が必要です。

クライアントがメッセージを受信する方法
設定されたリンクルートを使用すると、クライアントは異なるネットワーク上にある場合でもブローカー **B2** からメッセージを受信できます。

ルーター **R2** はブローカー **B2** への接続を確立します。接続が開かれると、**R2** は他のルーター (**R1** および **R3**) に、これが **b2** 接頭辞へのリンクルート用の有効な宛先であることを通知します。つまり、**R1** または **R3** に割り当てられた送信元と受信側のリンクは、最短のパスは **R2** にルーティングされ、それをブローカー **B2** にルーティングします。

ブローカー **B2** で **b2.event-queue** からメッセージを受信するには、クライアントは受信側リンクを **b2.event-queue** のソースアドレスでそのローカルルーター (**R3**) に割り当てます。アドレスは **b2** 接頭辞に一致するため、**R3** はリンクを **R1** にルーティングします。これは、ルート内の次のホップから宛先にルーティングします。**R1** は、リンクを **R2** にルーティングし、これをブローカー **B2** にルーティングします。クライアントが受信側が確立され、メッセージの受信を開始できます。



注記

ブローカー **B2** が何らかの理由で利用できない場合、ルーター **R2** は **b2** アドレスの宛先としてアドバタイズしません。この場合、ルーター **R1** および **R3** は、宛先に利用可能なルートがないことを示すエラーメッセージで、ブローカー **B2** にルーティングされるべきリンク接続を拒否します。

パート VI. 管理

第13章 AMQ 管理コンソールを使用した監視

AMQ Management Console は、AMQ Interconnect ルーターネットワークのステータスとパフォーマンスを監視するための Web コンソールです。

前提条件

- AMQ 管理コンソールには **qpid-dispatch-console** パッケージが必要です。
  パッケージのインストールの詳細は、5章*AMQ Interconnect のインストール* を参照してください。

13.1. AMQ 管理コンソールへのアクセスの設定

Web コンソールにアクセスする前に、Web コンソールの HTTP 接続を許可し、コンソールファイルを提供するように **listener** を設定する必要があります。

手順

1. Web コンソールにアクセスするルーターで、**/etc/qpid-dispatch/qdrouterd.conf** 設定ファイルを開きます。
2. コンソールを提供するために **listener** を追加します。
   以下の例では、クライアントが Web コンソールにアクセスするために使用できる **listener** を作成します。

```
listener {
    host: 0.0.0.0
    port: 8672
    role: normal
    http: true
    httpRootDir: /usr/share/qpid-dispatch/console
}
```

host

ルーターがリッスンする IP アドレス (IPv4 または IPv6)、またはルーターのホスト名。

port

ルーターがリッスンするポート番号またはシンボリックサービス名（ **/etc/services** で定義されます）。

role

接続のロール。**normal** を指定して、このコネクションがクライアントトラフィックに使用されることを示します。

http

この属性を **true** に設定すると、この **listener** がプレーン AMQP コネクションではなく HTTP 接続を受け入れる必要があります。

httpRootDir

Web コンソールの HTML ファイルが含まれるディレクトリーへの絶対パスを指定します。デフォルトのディレクトリーはスタンドアロンのコンソールインストールディレクトリーで、通常は **/usr/share/qpid-dispatch/console** です。

3. コンソールへのアクセスのセキュリティーを保護する場合は、**listener** をセキュアにします。

詳細は、「着信クライアント接続のセキュリティー保護」 を参照してください。この例では、SASL PLAIN を使用して基本的なユーザー名とパスワード認証を追加します。

```
listener {
    host: 0.0.0.0
    port: 8672
    role: normal
    http: true
    httpRootDir: /usr/share/qpid-dispatch/console
    authenticatePeer: yes
    saslMechanisms: PLAIN
}
```

4. ルーターネットワーク内の他のルーターから Web コンソールへのアクセスを設定する場合は、それぞれのルーターについてこの手順を繰り返してください。

13.2. AMQ 管理コンソールへのアクセス

Web ブラウザーから Web コンソールにアクセスできます。

手順

1. Web ブラウザーで、Web コンソール URL に移動します。
   Web コンソール URL は、Web コンソールを提供するために作成した **listener** の **<host>**:**<port>** です。たとえば、**localhost:8672** です。

   AMQ Management Console が開きます。ユーザー名とパスワードの認証を設定すると、**Connect** タブが表示されます。

2. 必要に応じて、Web コンソールにログインします。
   ユーザー名とパスワードの認証を設定する場合は、Web コンソールにアクセスするためのユーザー名とパスワードを入力します。

   ユーザー名の構文は **<user>**@**<domain>** です。例: **admin@my-domain**

13.3. AMQ 管理コンソールを使用したルーターネットワークの監視

Web コンソールは、ルーターネットワークを監視するために使用できる複数のセクションを提供します。

このセクション...	提供する内容

このセクション...	提供する内容
概要	ルーターのネットワークに関する情報を集計します。この情報には以下が含まれます。 ● ダッシュボード (ルーターネットワークの統計を表示) ● ルーター ● アドレス ● リンク ● 接続 ● ログ
可視化	ルーターネットワークのグラフィカルビュー。以下のタイプのビジュアライゼーションが表示されます。 **Topology** ルーター、クライアント、ブローカーなど、ルーターネットワークのトポロジー。このビジュアライゼーションは、メッセージがネットワーク経由で流れる方法も示します。 **Message flow** アドレス別にリアルタイムメッセージフローを表示する図。
詳細	ルーターネットワーク上の各 AMQP 管理エンティティーについての詳細設定情報。ネットワーク内のルーターの設定を表示および変更できます。

第14章 QDSTAT を使用したモニターリング

qdstat ツールは、AMQ Interconnect ルーターネットワークのステータスおよびパフォーマンスを監視するためのコマンドラインツールです。

14.1. QDSTAT を使用するための構文

qdstat は、以下の構文で使用できます。

```
$ qdstat <option> [<connection-options>] [<secure-connection-options>]
```

これにより、以下が指定されます。

- 表示する情報タイプの**オプション**。
- 情報を表示するルーターを指定する 1 つ以上のオプションの**接続オプション**。
  接続オプションを指定しない場合、**qdstat** は localhost でリッスンするルーターおよびデフォルトの AMQP ポート (5672) に接続します。
- 情報を表示するルーターでセキュアな接続のみを受け入れる場合の**セキュアな接続オプション**。

関連情報

- **qdstat** の詳細は、man ページの qdstat を参照してください。

14.2. ルーターネットワークをモニターするためのコマンド

qdstat を使用して、ルーターネットワーク上のルーターのステータスを表示することができます。たとえば、割り当てられたリンクおよび設定されたアドレス、利用可能な接続、およびルーターネットワークのノードについての情報を表示できます。

以下を行う場合	このコマンド...
すべてのルーターに関する統計をすべて含む状態ダンプを作成します。 状態ダンプには、ルーターネットワークの現在の状態が表示されます。	`$ qdstat --all-routers --all-entities` インテリアルーターでこのコマンドを実行すると、すべてのインオリオータールーターの統計が表示されます。エッジルーターでコマンドを実行すると、そのエッジルーターのみの統計が表示されます。
すべてのルーターについて単一の統計を含む状態ダンプを作成します。	`$ qdstat -l|-a|-c|--autolinks|--linkroutes|-g|-m --all-routers` インテリアルーターでこのコマンドを実行すると、すべてのインオリオータールーターの統計が表示されます。エッジルーターでコマンドを実行すると、そのエッジルーターのみの統計が表示されます。

以下を行う場合	このコマンド...
単一ルーターのすべての統計を含む状態ダンプを作成します。	$ qdstat --all-entities このコマンドにより、ローカルルーターの統計のみが表示されます。
ルーターの一般的な統計の表示	$ qdstat -g [all-routers|**<connection-options>**]
ルーターへの接続の一覧を表示します。	$ qdstat -c [all-routers|**<connection-options>**]
ルーターに割り当てられた AMQP リンクの表示 AMQP のリンクの一覧をクライアントから (送信側/受信側)、他のルーター、または他のルーターから他のコンテナー (ブローカーなど)、およびツール自体から確認できます。	$ qdstat -l [all-routers|**<connection-options>**]
ルーターネットワーク上の既知のルーターの表示	$ qdstat -n [all-routers|**<connection-options>**]
ルーターが認識するアドレスの表示	$ qdstat -a [all-routers|**<connection-options>**]
ルーターの自動リンクの表示	$ qdstat --autolinks [all-routers|**<connection-options>**]
ルーターのリンクルートのステータスを表示します。	$ qdstat --linkroutes [all-routers|**<connection-options>**]
ルーターのポリシーグローバル設定および統計の表示	$ qdstat --policy [all-routers|**<connection-options>**]
ルーターのポリシー vhost 設定の表示	$ qdstat --vhosts [all-routers|**<connection-options>**]

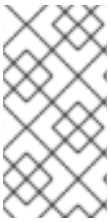
以下を行う場合	このコマンド…
ルーターのポリシー vhost 統計の表示	$ qdstat --vhoststats [all-routers|**<connection-options>**]
ルーターの vhostgroup 設定の表示	$ qdstat --vhostgroups [all-routers|**<connection-options>**]
ルーターのメモリー消費の表示	$ qdstat -m [all-routers|**<connection-options>**]

関連情報

- 各 **qdstat** コマンドで表示されるフィールドの詳細は、man ページの qdstat を参照してください。

第15章 QDMANAGE を使用した管理

qdmanage ツールは、ランタイム時に実行中のルーターの設定を表示および変更するコマンドラインツールです。



注記

qdmanage を使用してルーターに変更を加えると、変更はすぐに有効になりますが、ルーターは停止した場合に失われます。ルーターの設定に永続的な変更を加える場合は、ルーターの **/etc/qpid-dispatch/qdrouterd.conf** 設定ファイルを編集する必要があります。

以下の構文で **qdmanage** を使用できます。

```
$ qdmanage [<connection-options>] <operation> [<options>]
```

これにより、以下が指定されます。

- 操作を実行するルーターを指定する 1 つ以上のオプションの **接続オプション** や、ルーターがセキュアな接続のみを受け入れる場合はセキュリティー認証情報を指定するための 1 つ以上のオプション。
  接続オプションを指定しない場合、**qdmanage** は localhost でリッスンするルーターおよびデフォルトの AMQP ポート (5672) に接続します。
- ルーターで実行する**操作**。
- 操作を実行する設定エンティティーを指定する 1 つ以上の**オプション**、またはコマンド出力のフォーマット方法を指定します。

qdmanage コマンドを入力すると、AMQP 管理操作のリクエストとして実行され、JSON 形式のコマンド出力として応答が返されます。

たとえば、以下のコマンドはルーターでクエリー操作を実行してから、JSON 形式で応答を返します。

```
$ qdmanage query --type listener
[
  {
    "stripAnnotations": "both",
    "addr": "127.0.0.1",
    "multiTenant": false,
    "requireSsl": false,
    "idleTimeoutSeconds": 16,
    "saslMechanisms": "ANONYMOUS",
    "maxFrameSize": 16384,
    "requireEncryption": false,
    "host": "0.0.0.0",
    "cost": 1,
    "role": "normal",
    "http": false,
    "maxSessions": 32768,
    "authenticatePeer": false,
    "type": "org.apache.qpid.dispatch.listener",
    "port": "amqp",
    "identity": "listener/0.0.0.0:amqp",
```

```
    "name": "listener/0.0.0.0:amqp"
  }
]
```

関連情報

- **qdmanage** の詳細は、man ページの qdmanage を参照してください。

第16章 AMQ INTERCONNECT のトラブルシューティング

AMQ Interconnect ログを使用すると、ルーターネットワークのルーターに関するエラーおよびパフォーマンスの問題を診断し、トラブルシューティングすることができます。

16.1. ログエントリーの表示

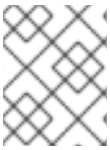
エラー、パフォーマンスの問題、およびその他の重要な問題の診断には、ログエントリーを表示する必要がある場合があります。ログエントリーは、オプションのタイムスタンプ、ロギングモジュール、ロギングレベル、およびログメッセージで設定されます。

手順

- 次のいずれかを行います。
  - コンソールでログエントリーを表示します。
    デフォルトでは、イベントはコンソールに記録され、そこで表示できます。ただし、**output** 属性が特定のロギングモジュールに設定されている場合、これらのログエントリーは指定された場所 (**stderr**、**syslog**、またはファイル) で確認できます。
  - **qdstat --log** コマンドを使用して最新のログエントリーを表示します。
    --limit パラメーターを使用して、表示されるログエントリーの数を制限できます。qdstat の詳細は、**qdstat man ページ** を参照してください。

    以下の例では、**Router.A** の最後の 3 つのログエントリーを表示します。

    ```
    $ qdstat --log --limit=3 -r ROUTER.A
    Wed Jun  7 17:49:32 2019 ROUTER (none) Core action 'link_deliver'
    Wed Jun  7 17:49:32 2019 ROUTER (none) Core action 'send_to'
    Wed Jun  7 17:49:32 2019 SERVER (none) [2]:0 -> @flow(19) [next-incoming-id=1,
    incoming-window=61, next-outgoing-id=0, outgoing-window=2147483647, handle=0,
    delivery-count=1, link-credit=250, drain=false]
    ```



注記

/etc/qpid-dispatch/qdrouterd.conf 設定ファイルで **multiTenant** が **true** に設定されている場合にのみ、**vhost** エントリーが設定されます。

関連情報

- ロギングモジュールの設定に関する詳細は、「デフォルトロギングの設定」 を参照してください。

16.2. ログを使用したトラブルシューティング

AMQ Interconnect ログエントリーを使用すると、ネットワークのルーターに関するエラーおよびパフォーマンスの問題を診断できます。

例16.1 接続およびリンクのトラブルシューティング

この例では、**ROUTER** ログには接続のライフサイクルと、それに関連するリンクが表示されます。

```
2019-04-05 14:54:38.037248 -0400 ROUTER (info) [C1] Connection Opened: dir=in
```

```
host=127.0.0.1:55440 vhost= encrypted=no auth=no user=anonymous container_id=95e55424-6c0a-4a5c-8848-65a3ea5cc25a props= (1)
2019-04-05 14:54:38.038137 -0400 ROUTER (info) [C1][L6] Link attached: dir=in source={<none> expire:sess} target={$management expire:sess} (2)
2019-04-05 14:54:38.041103 -0400 ROUTER (info) [C1][L6] Link lost: del=1 presett=0 psdrop=0 acc=1 rej=0 rel=0 mod=0 delay1=0 delay10=0 (3)
2019-04-05 14:54:38.041154 -0400 ROUTER (info) [C1] Connection Closed (4)
```

1. 接続が開きます。各接続には一意の ID (**C1**) があります。ログには、接続に関する情報も表示されます。
2. リンクは接続上でアタッチされます。リンクは一意の ID (**L6**) で識別されます。ログには、リンクの方向と、ソースアドレスとターゲットアドレスも表示されます。
3. リンクがデタッチされます。ログには、リンクの端末統計が表示されます。
4. 接続は閉じられます。

注記

必要な場合は **qdmanage** を使用して特定の接続のプロトコルレベルのトレースロギングを有効にすることができます。これを使用して、AMQP フレームを追跡できます。以下は例になります。

```
$ qdmanage update --type=connection --id=C1 enableProtocolTrace=true
```

例16.2 ネットワークトポロジーのトラブルシューティング

この例では、**Router.A** の **ROUTER_HELLO** ログは **Router.B** に接続されており、その **Router.B** がその **Router.A** および **Router.C** に接続されていることを示しています。

```
Tue Jun  7 13:50:21 2016 ROUTER_HELLO (trace) RCVD: HELLO(id=Router.B area=0 inst=1465307413 seen=['Router.A', 'Router.C']) (1)
Tue Jun  7 13:50:21 2016 ROUTER_HELLO (trace) SENT: HELLO(id=Router.A area=0 inst=1465307416 seen=['Router.B']) (2)
Tue Jun  7 13:50:22 2016 ROUTER_HELLO (trace) RCVD: HELLO(id=Router.B area=0 inst=1465307413 seen=['Router.A', 'Router.C'])
Tue Jun  7 13:50:22 2016 ROUTER_HELLO (trace) SENT: HELLO(id=Router.A area=0 inst=1465307416 seen=['Router.B'])
```

1. **Router.A** は **Router.B** から Hello メッセージを受信し、**Router.A** と **Router.C** を参照してください。
2. **router.A** は Hello メッセージを **Router.B** に送信し、確認できる唯一のルーターです。

Router.B では、**ROUTER_HELLO** ログは、別のパースペクティブからの同じルータートポロジーを表示します。

```
Tue Jun  7 13:50:18 2016 ROUTER_HELLO (trace) SENT: HELLO(id=Router.B area=0 inst=1465307413 seen=['Router.A', 'Router.C']) (1)
```

```
Tue Jun  7 13:50:18 2016 ROUTER_HELLO (trace) RCVD: HELLO(id=Router.A area=0
inst=1465307416 seen=['Router.B']) 2
Tue Jun  7 13:50:19 2016 ROUTER_HELLO (trace) RCVD: HELLO(id=Router.C area=0
inst=1465307411 seen=['Router.B']) 3
```

1. **router.B** は Hello メッセージを **Router.A** および **Router.C** に送信します。
2. **Router.B** は **Router.A** から Hello メッセージを受信したので、 **Router.B** だけが表示されます。
3. **Router.B** は **Router.C** から Hello メッセージを受信したので、 **Router.B** だけが表示されます。

例16.3 ルーター間のリンク状態のトレース

定期的に、各ルーターは他のルーターに Link State Request (LSR) を送信し、要求された情報で Link State Update (LSU) を受信します。上記の情報を試すことで、各ルーターはトポロジー内の次のホップを計算し、関連するコストを計算できます。

この例では、**ROUTER_LS** のログは、3 つのルーター間で送信される RA、LSR、および LSU メッセージを表示します。

```
Tue Jun  7 14:10:02 2016 ROUTER_LS (trace) SENT: LSR(id=Router.A area=0) to: Router.C
Tue Jun  7 14:10:02 2016 ROUTER_LS (trace) SENT: LSR(id=Router.A area=0) to: Router.B
Tue Jun  7 14:10:02 2016 ROUTER_LS (trace) SENT: RA(id=Router.A area=0 inst=1465308600
ls_seq=1 mobile_seq=1) 1
Tue Jun  7 14:10:02 2016 ROUTER_LS (trace) RCVD: LSU(id=Router.B area=0
inst=1465308595 ls_seq=2 ls=LS(id=Router.B area=0 ls_seq=2 peers={'Router.A': 1L, 'Router.C':
1L})) 2
Tue Jun  7 14:10:02 2016 ROUTER_LS (trace) RCVD: LSR(id=Router.B area=0)
Tue Jun  7 14:10:02 2016 ROUTER_LS (trace) SENT: LSU(id=Router.A area=0 inst=1465308600
ls_seq=1 ls=LS(id=Router.A area=0 ls_seq=1 peers={'Router.B': 1}))
Tue Jun  7 14:10:02 2016 ROUTER_LS (trace) RCVD: RA(id=Router.C area=0 inst=1465308592
ls_seq=1 mobile_seq=0)
Tue Jun  7 14:10:02 2016 ROUTER_LS (trace) SENT: LSR(id=Router.A area=0) to: Router.C
Tue Jun  7 14:10:02 2016 ROUTER_LS (trace) RCVD: LSR(id=Router.C area=0) 3
Tue Jun  7 14:10:02 2016 ROUTER_LS (trace) SENT: LSU(id=Router.A area=0 inst=1465308600
ls_seq=1 ls=LS(id=Router.A area=0 ls_seq=1 peers={'Router.B': 1}))
Tue Jun  7 14:10:02 2016 ROUTER_LS (trace) RCVD: LSU(id=Router.C area=0
inst=1465308592 ls_seq=1 ls=LS(id=Router.C area=0 ls_seq=1 peers={'Router.B': 1L})) 4
Tue Jun  7 14:10:03 2016 ROUTER_LS (trace) Computed next hops: {'Router.C': 'Router.B',
'Router.B': 'Router.B'} 5
Tue Jun  7 14:10:03 2016 ROUTER_LS (trace) Computed costs: {'Router.C': 2L, 'Router.B': 1}
Tue Jun  7 14:10:03 2016 ROUTER_LS (trace) Computed valid origins: {'Router.C': [], 'Router.B':
[]}
```

1. **Router.A** は LSR 要求を送信し、RA アドバータイズメントをネットワーク上の他のルーターに送信します。
2. **Router.A** は **Router.B** から LSU を受信しました。これには、 **Router.A**, と **Router.C** (**1** の消費) の 2 つのピアがあります。
3. **Router.A** は **Router.B** および **Router.C** から LSR を受信し、LSU と共に復元します。
4.

Router.A は **Router.C** から LSU を受け取ります。これには、1 つのピア **Router.B** (コストは **1**) のみがあります。

5 LSR および LSU メッセージが交換されると、**Router.A** は関連するコストでルートポロジーを計算します。

例16.4 ルーターに割り当てられたモバイルアドレスの状態の追跡

この例では、**ROUTER_MA** ログには、3 つのルーター間で送信される Mobile Address Request (MAR) および Mobile Address Update (MAU) メッセージが表示されています。

```
Tue Jun  7 14:27:20 2016 ROUTER_MA (trace) SENT: MAU(id=Router.A area=0 mobile_seq=1
add=['Cmy_queue', 'Dmy_queue', 'M0my_queue_wp'] del=[]) 1
Tue Jun  7 14:27:21 2016 ROUTER_MA (trace) RCVD: MAR(id=Router.C area=0 have_seq=0)
2
Tue Jun  7 14:27:21 2016 ROUTER_MA (trace) SENT: MAU(id=Router.A area=0 mobile_seq=1
add=['Cmy_queue', 'Dmy_queue', 'M0my_queue_wp'] del=[])
Tue Jun  7 14:27:22 2016 ROUTER_MA (trace) RCVD: MAR(id=Router.B area=0 have_seq=0)
3
Tue Jun  7 14:27:22 2016 ROUTER_MA (trace) SENT: MAU(id=Router.A area=0 mobile_seq=1
add=['Cmy_queue', 'Dmy_queue', 'M0my_queue_wp'] del=[])
Tue Jun  7 14:27:39 2016 ROUTER_MA (trace) RCVD: MAU(id=Router.C area=0 mobile_seq=1
add=['M0my_test'] del=[]) 4
Tue Jun  7 14:27:51 2016 ROUTER_MA (trace) RCVD: MAU(id=Router.C area=0 mobile_seq=2
add=[] del=['M0my_test']) 5
```

1 **Router.A** は、**my_queue** および **my_queue_wp** に追加されたアドレスについて通知するために、ネットワーク内の他のルーターに MAU メッセージを送信しました。

2 **Router.A** は **Router.C** から応答で MAR メッセージを受信しました。

3 **Router.A** は、**Router.B** から応答として別の MAR メッセージを受信しました。

4 **Router.C** は MAU メッセージを送信して、**my_test** の追加およびアドレスを他のルーターに通知します。

5 **Router.C** は別の MAU メッセージを送信し、**my_test** のアドレスが削除されたことを他のルーターに通知します (受信側はデタッチされます)。

例16.5 ルーターによって送受信されたメッセージに関する情報の検索

以下の例では、**MESSAGE** ログには **Router.A** が Hello プロトコルに関連する一部のメッセージが送受信され、モバイルアドレスのリンク上の他のメッセージを送信および受信したことを示しています。

```
Tue Jun  7 14:36:54 2016 MESSAGE (trace) Sending
Message{to='amqp:/_topo/0/Router.B/qdrouter'
body='\d1\00\00\00\1b\00\00\00\04\a1\02id\a1\08R'} on link qdlink.p9XmBm19uDqx50R
Tue Jun  7 14:36:54 2016 MESSAGE (trace) Received
Message{to='amqp:/_topo/0/Router.A/qdrouter' body='\d1\00\00\00\8e\00\00\00
\a1\06ls_se'} on link qdlink.phMsJOq7YaFsGAG
```

```
Tue Jun  7 14:36:54 2016 MESSAGE (trace) Received Message{
body='\d1\00\00\00\10\00\00\00\02\a1\08seque'} on link qdlink.FYHqBX+TtwXZHfV
Tue Jun  7 14:36:54 2016 MESSAGE (trace) Sending Message{
body='\d1\00\00\00\10\00\00\00\02\a1\08seque'} on link qdlink.yU1tnPs5KbMlieM
Tue Jun  7 14:36:54 2016 MESSAGE (trace) Sending Message{to='amqp:/_local/qdhello'
body='\d1\00\00\00G\00\00\00\08\a1\04seen\d0'} on link qdlink.p9XmBm19uDqx50R
Tue Jun  7 14:36:54 2016 MESSAGE (trace) Sending
Message{to='amqp:/_topo/0/Router.C/qdrouter'
body='\d1\00\00\00\1b\00\00\00\04\a1\02id\a1\08R'} on link qdlink.p9XmBm19uDqx50R
```

例16.6 ルーターへの設定変更の追跡

この例では、**AGENT** ログは **Router.A**、**address**、**linkRoute**、および **autoLink** エンティティー上がルーターの設定ファイルに追加されていることを示しています。ルーターが起動すると **AGENT** モジュールがこれらの変更を適用し、それらがログで表示可能になりました。

```
Tue Jun  7 15:07:32 2016 AGENT (debug) Add entity: ConnectorEntity(addr=127.0.0.1,
allowRedirect=True, cost=1, host=127.0.0.1, identity=connector/127.0.0.1:5672:BROKER,
idleTimeoutSeconds=16, maxFrameSize=65536, name=BROKER, port=5672, role=route-
container, stripAnnotations=both, type=org.apache.qpid.dispatch.connector,
verifyHostname=True)
Tue Jun  7 15:07:32 2016 AGENT (debug) Add entity:
RouterConfigAddressEntity(distribution=closest, identity=router.config.address/0,
name=router.config.address/0, prefix=my_address,
type=org.apache.qpid.dispatch.router.config.address, waypoint=False)
Tue Jun  7 15:07:32 2016 AGENT (debug) Add entity:
RouterConfigAddressEntity(distribution=balanced, identity=router.config.address/1,
name=router.config.address/1, prefix=my_queue_wp,
type=org.apache.qpid.dispatch.router.config.address, waypoint=True)
Tue Jun  7 15:07:32 2016 AGENT (debug) Add entity:
RouterConfigLinkrouteEntity(connection=BROKER, direction=in, distribution=linkBalanced,
identity=router.config.linkRoute/0, name=router.config.linkRoute/0, prefix=my_queue,
type=org.apache.qpid.dispatch.router.config.linkRoute)
Tue Jun  7 15:07:32 2016 AGENT (debug) Add entity:
RouterConfigLinkrouteEntity(connection=BROKER, direction=out, distribution=linkBalanced,
identity=router.config.linkRoute/1, name=router.config.linkRoute/1, prefix=my_queue,
type=org.apache.qpid.dispatch.router.config.linkRoute)
Tue Jun  7 15:07:32 2016 AGENT (debug) Add entity:
RouterConfigAutolinkEntity(address=my_queue_wp, connection=BROKER, direction=in,
identity=router.config.autoLink/0, name=router.config.autoLink/0,
type=org.apache.qpid.dispatch.router.config.autoLink)
Tue Jun  7 15:07:32 2016 AGENT (debug) Add entity:
RouterConfigAutolinkEntity(address=my_queue_wp, connection=BROKER, direction=out,
identity=router.config.autoLink/1, name=router.config.autoLink/1,
type=org.apache.qpid.dispatch.router.config.autoLink)
```

例16.7 ポリシーおよび vhost アクセスルールのトラブルシューティング

以下の例では、**POLICY** ログにはこのルーターに最大接続に制限がなく、デフォルトのアプリケーションポリシーが無効になっていることが分かります。

```
Tue Jun  7 15:07:32 2016 POLICY (info) Policy configured maximumConnections: 0, policyFolder:
'', access rules enabled: 'false'
Tue Jun  7 15:07:32 2016 POLICY (info) Policy fallback defaultApplication is disabled
```

例16.8 エラーの診断

以下の例では、**ERROR** ログには、ルーターの設定ファイルに対して正しくないパスが指定されている場合にルーターの開始に失敗することを示しています。

```
$ qdrouterd --conf my_config
Wed Jun 15 09:53:28 2016 ERROR (error) Python: Exception: Cannot load configuration file
my_config: [Errno 2] No such file or directory: 'my_config'
Wed Jun 15 09:53:28 2016 ERROR (error) Traceback (most recent call last):
  File "/usr/lib/qpid-dispatch/python/qpid_dispatch_internal/management/config.py", line 155, in
configure_dispatch
    config = Config(filename)
  File "/usr/lib/qpid-dispatch/python/qpid_dispatch_internal/management/config.py", line 41, in
__init__
    self.load(filename, raw_json)
  File "/usr/lib/qpid-dispatch/python/qpid_dispatch_internal/management/config.py", line 123, in
load
    with open(source) as f:
Exception: Cannot load configuration file my_config: [Errno 2] No such file or directory: 'my_config'

Wed Jun 15 09:53:28 2016 MAIN (critical) Router start-up failed: Python: Exception: Cannot load
configuration file my_config: [Errno 2] No such file or directory: 'my_config'
qdrouterd: Python: Exception: Cannot load configuration file my_config: [Errno 2] No such file or
directory: 'my_config'
```

関連情報

- ロギングモジュールの詳細は、「ロギングモジュール」 を参照してください。

付録A サブスクリプションの使用

AMQ は、ソフトウェアサブスクリプションから提供されます。サブスクリプションを管理するには、Red Hat カスタマーポータルでアカウントにアクセスします。

A.1. アカウントへのアクセス

手順

1. access.redhat.com に移動します。
2. アカウントがない場合は、作成します。
3. アカウントにログインします。

A.2. サブスクリプションのアクティベート

手順

1. access.redhat.com に移動します。
2. **サブスクリプション** に移動します。
3. **Activate a subscription** に移動し、16 桁のアクティベーション番号を入力します。

A.3. リリースファイルのダウンロード

.zip、.tar.gz およびその他のリリースファイルにアクセスするには、カスタマーポータルを使用してダウンロードする関連ファイルを検索します。RPM パッケージまたは Red Hat Maven リポジトリーを使用している場合は、この手順は必要ありません。

手順

1. ブラウザーを開き、access.redhat.com/downloads で Red Hat カスタマーポータルの **Product Downloads** ページにログインします。
2. **JBOSS INTEGRATION AND AUTOMATION** カテゴリーの **Red Hat AMQ** エントリーを見つけます。
3. 必要な AMQ 製品を選択します。**Software Downloads** ページが開きます。
4. コンポーネントの **Download** リンクをクリックします。

A.4. パッケージ用システムの登録

この製品の RPM パッケージを Red Hat Enterprise Linux にインストールするには、システムが登録されている必要があります。ダウンロードしたリリースファイルを使用している場合は、この手順は必要ありません。

手順

1. access.redhat.com に移動します。

2. **Registration Assistant** に移動します。
3. ご使用の OS バージョンを選択し、次のページに進みます。
4. システムの端末に一覧表示されたコマンドを使用して、登録を完了します。

システムを登録する方法は、以下のリソースを参照してください。

- Red Hat Enterprise Linux 7 - システム登録およびサブスクリプション管理
- Red Hat Enterprise Linux 8 - システム登録およびサブスクリプション管理

改訂日時: 2023-05-17