9.3

IBM MQ のシナリオ

注記

本書および本書で紹介する製品をご使用になる前に、223 ページの『特記事項』に記載されている情報をお読みください。

本書は、 IBM® MQ バージョン 9 リリース 3、および新しい版で明記されていない限り、以降のすべてのリリースおよびモディフィケーションに適用されます。

お客様が IBM に情報を送信する場合、お客様は IBM に対し、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で情報を使用または配布する非独占的な権利を付与します。

目次

IBM MQ のシナリオ

各シナリオにより、相当数のタスクの集合を概観できるので、主要な製品フィーチャーを構成するのに役立ちます。 このシナリオには、関心のある分野についてさらに理解を深めるうえで役立つ他のコンテンツへの有用なリンクが含まれています。

使用可能な IBM MQ シナリオについては、以下のサブトピックで説明します。

Windows IBM MQ 入門

このシナリオでは、Windows プラットフォームで IBM MQ を開始する方法について説明します。 IBM MQ をまだ使用したことがなく、すぐに使用を開始する場合に、このシナリオを使用します。

このシナリオは、IBM MQ がシステムにまだインストールされていない場合に、 Windows にインストールし、構成および検査するための基本ステップについて説明しています。 グラフィカル・ユーザー・インターフェース またはコマンド行インターフェースを使用して、シナリオのステップを完了することができます。

ソリューションの計画

Windows に IBM MQ をインストールする方法を選択します。 グラフィカル・ユーザー・インターフェースおよびインストール・プロセスと構成プロセスを実行するウィザードを使用するか、コマンド・ラインを使用してサイレント・インストールを実行します。

概要: 実現する論理トポロジー

シナリオが完成した後に実現する論理トポロジー。

インストール済み IBM MQ サーバー・インスタンスによって、IBM MQ オブジェクト (キューやキュー・マネージャー) を作成できます。 IBM MQ Explorer を使用して、ローカル・キューとキュー・マネージャーとの間のメッセージの書き込みおよび取得を実行できます。 このシナリオが完成すると、実現するトポロジーは図 1 のようになります。

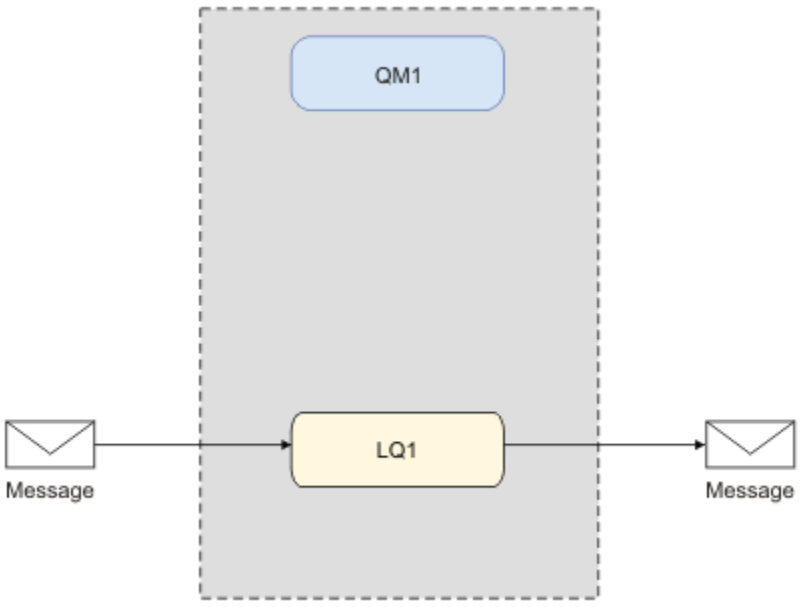


図 *1. LQ1* でのメッセージの書き込みと *LQ1* からのメッセージの取得

基本概念と重要な用語

『IBM MQ の概要』のシナリオを使用する前に理解しておく必要のある、基本概念および重要な用語について説明します。

基本概念

IBM MQ によって、アプリケーションは、キューに対するメッセージの読み取り/書き込みを実行できます。メッセージを読み取るアプリケーションは、メッセージを書き込むアプリケーションから独立しています。2 つのアプリケーションを同時に実行する必要はありません。 メッセージを読み取るアプリケーションが使用不可である場合、アプリケーションがメッセージを読み取るまで、そのメッセージは IBM MQ キューで待機します。

このシナリオでは、IBM MQ のインストールおよび構成方法を、以下のいずれかから選択できます。

7 ページの『グラフィカル・ユーザー・インターフェースを使用したインストールおよび構成』

グラフィカル・ユーザー・インターフェースを使用してインストールする場合は、適切なオプションや設定を適用できるように、複数の画面やウィザードにガイダンスが表示されます。

ランチパッド
ソフトウェア要件を確認し、ネットワーク情報を指定して、IBM MQ インストール・ウィザードを開始します。

IBM MQ インストール・ウィザード
ソフトウェアをインストールして、Prepare IBM MQ Wizard を開始します。

Prepare IBM MQ Wizard
IBM MQ サービスと IBM MQ Explorer を開始します。

IBM MQ Explorer
キューとキュー・マネージャーを管理します。

12 ページの『コマンド・ラインインターフェースを使用したインストールおよび構成』
コマンド・ライン・インターフェースでのインストールは、サイレントまたは対話式のいずれかで実行できます。 サイレント・インストールは、アクセシビリティーに完全に対応しており、このシナリオで扱われます。 コマンド・ラインを使用してインストールする間、ガイドが提供され、いくつかのステップに従って関連するオプションや設定を適用することができます。

- IBM MQ をインストール
- IBM MQ オブジェクト (キューとキュー・マネージャー) を作成および構成します。
- `amqsput` を使用してキューへのメッセージの書き込み、また `amqsget` を使用してキューからのメッセージの取得を実行することによって、インストールを検証します。

IBM MQ Explorer やコマンド・ラインを使用して IBM MQ オブジェクトを作成するのと同じように、プログラマブル・インターフェースを使用してそれらを作成することができます。 この方法は、現在のシナリオでは扱われません。

重要な用語

以下に、メッセージ・キューイングに関する重要な用語のリストを示します。

メッセージ・キューイングに関する重要な用語。

用語	説明
キュー・マネージャー	キュー・マネージャーは、所有するキューを管理し、受信したすべてのメッセージを適切なキューに格納します。
メッセージ	メッセージを使用するアプリケーションにとって、メッセージは意味のあるバイト・ストリングです。 メッセージは、ある 1 つのアプリケーション・プログラムから別のアプリケーションに情報を転送するために使用されます。 アプリケーションは、同じコンピューターで実行することも、異なるコンピューターで実行することもできます。
ローカル・キュー	ローカル・キューとは、メッセージを保管するためのデータ構造体です。 このキューは、通常キューであることも、伝送キューであることも可能です。 通常キューは、キュー・マネージャーからメッセージを直接読み取るアプリケーションによって読み取られるメッセージを保持します。 伝送キューは、別のキュー・マネージャーに転送中のメッセージを保持します。

ソリューションの実装

シナリオのソリューションを実装します。 インストール・ランチパッドを使用して Windows に IBM MQ をインストールしてから、 IBM MQ Explorer を使用してインストールを検証します。

グラフィカル・ユーザー・インターフェースを使用したインストールおよび構成

インストール・ランチパッドを使用して Windows に IBM MQ をインストールし、IBM MQ Explorer を使用してインストールを検証します。 インストール済み環境を検証した後、キュー・マネージャーの作成およびキューを作成し、その後でキューへのメッセージの書き込みおよびキューからのメッセージの取得を試行します。

Launchpad を使用したインストール

インストール済みの Launchpad を使用して Windows に IBM MQ をインストールします。

始める前に

このタスクを始める前に、以下の確認を行ってください。

- インストール時に、ローカル管理者権限が必要です。 その権限は Windows の機能で定義します。
- マシン名にスペースが含まれていないことを確認します。
- IBM MQ for Windows をフルインストールするための十分なディスク・スペースがあることを確認します。 詳しくは、ディスク・スペース要件 (Multiplatforms) を参照してください。
- いずれかの IBM MQ ユーザーに対して Windows ドメイン・ユーザー ID を定義する必要があるかどうかを判別します。

IBM MQ をインストールする前に、システムがハードウェアとソフトウェアの要件を満たしていることを確認してください。 サポートされるすべてのプラットフォームでのハードウェア要件とソフトウェア要件について詳しくは、 IBM MQ のシステム要件を参照してください。

このタスクについて

ランチパッドと後続のウィザードに従ってインストール処理を進め、ソフトウェア要件と IBM MQ 設定を確認することができます。

この作業は、ご使用のマシンに初めて IBM MQ をインストールすること、およびデフォルトの場所を使用することを想定しています。 初期設定では、 IBM MQ プログラム・ファイルのロケーションは C:¥Program Files¥IBM¥MQ であり、データおよびログ・ファイルのロケーションは C: ¥ProgramData¥IBM¥MQ です。

注 : IBM MQ 9.0 をインストールしている場合、マシン上に以前の IBM MQ のインストール済み環境がある場合は、プログラムおよびデータ・ファイルの場所はデフォルトとは異なります。 詳しくは、プログラム

およびデータのディレクトリーの場所を参照してください。 以前にこのシナリオを既に実行したことがあり、デフォルトのロケーションを使用して単一のフレッシュ・インストールでこのシナリオを繰り返す場合、シナリオを再度開始する前に、以前のインストールを削除します。 マシンから既存の IBM MQ インスタンスをアンインストールする方法については、17 ページの『IBM MQ のアンインストール』を参照してください。

インストール・プログラムには、インストール・プロセスの間に必要になる場合のために、詳細情報へのリンクが含まれます。

手順

1. ランチパッドを起動して、ソフトウェア要件およびネットワーク構成を確認し、必要であれば変更します。
   a) IBM MQ ソフトウェア・ディレクトリーに移動し、`Setup.exe` ファイルをダブルクリックして、ランチパッドを開始します。
   b)「**ソフトウェア要件**」タブを選択して、「**ソフトウェア要件**」設定を表示します。
   c) ソフトウェア要件が満たされていること、および要件の項目に OK という語を含む緑色のチェック・マークが表示されていることを確認します。 指示されている訂正を行います。

      注 :

      要件の詳細は、チェック・ボックスをクリックし、情報タブを展開して参照できます。
   d)「**ネットワーク構成**」タブを選択し、「**ネットワーク構成**」設定を表示します。
   e)「**いいえ**」を選択します。

      注 : このシナリオは、IBM MQ 用にドメイン・ユーザー ID を構成する必要がないことを想定しています。 IBM MQ for Windows ドメイン・ユーザーの構成について詳しくは、「**詳細情報**」をクリックしてください。
   f) ランチパッドの「**IBM MQ インストール**」タブで、インストール言語を選択してから、「**IBM MQ インストーラーの起動**」をクリックして、IBM MQ インストール・ウィザードを開始します。

   インストール要件を満たす、または指定することにより、IBM MQ のセットアップを完了し、IBM MQ インストール・ウィザードを開始しました。
2. IBM MQ インストール・ウィザードを使用してソフトウェアをインストールし、Prepare IBM MQ Wizard を開始します。
   a) IBM MQ インストール・ウィザードで、ご使用条件を読み、「**使用条件の条項に同意します**」チェック・ボックスをクリックして、「**次へ**」をクリックします。
   b)「**標準**」をクリックしてから、「**次へ**」をクリックします。
   c)「**IBM MQ をインストールする準備ができました**」ページで、インストール情報を確認し、「**インストール**」をクリックします。

      注 : 以下の詳細に注意してください。

      - インストール環境の名前
      - プログラム・ファイルの最上位フォルダー
      - データ・ファイルの最上位フォルダー

      以下の機能がインストールされます。

      - IBM MQ Server
      - IBM MQ: IBM MQ リソースを管理およびモニターするためのグラフィカル・インターフェース
      - Java 、 .NET のメッセージングおよび Web サービス
      - IBM MQ 開発ツールキット

      インストール・プロセスが開始します。 ご使用のシステムによっては、インストール・プロセスに数分を要する場合があります。

インストール・プロセスの最後に、IBM MQ セットアップ・ウィンドウに `Installation Wizard Completed Successfully` というメッセージが表示されます。

d)「**完了 (Finish)**」 をクリックします。

IBM MQ が正常にインストールされました。 Prepare IBM MQ Wizard が自動的に開始し、**Prepare IBM MQ Wizard** ページが表示されます。

3. Prepare IBM MQ Wizard を使用して、IBM MQ サービスを開始します。

a)「Prepare IBM MQ Wizard にようこそ」ページで、「**次へ**」を選択します。

Prepare IBM MQ Wizard には、メッセージ `Status: Checking IBM MQ Configuration` と進行状況表示バーが表示されます。 処理が完了すると、IBM MQ ネットワーク構成ページが表示されます。

b) Prepare IBM MQ Wizard の「 IBM MQ ネットワーク構成」ページで、 「**いいえ**」を選択します。

c) **次へ** をクリックします。

Prepare IBM MQ Wizard には、メッセージ `Status: starting the IBM MQ Service` と進行状況表示バーが表示されます。 処理が完了すると、ウィザードに「Prepare IBM MQ Wizard の完了」ページが表示されます。

d)「**IBM MQ エクスプローラーを起動する**」を選択し、リリース・ノートを表示するかどうかを選択して、「**完了**」ボタンをクリックします。

IBM MQ エクスプローラーが開始します。

IBM MQ がインストールされました。 IBM MQ Explorer も開始しました。

タスクの結果

IBM MQ のインストールおよび検証が完了し、キュー・マネージャーやキューなどのオブジェクトを構成する準備が整いました。

次のタスク

を参照してください。9 ページの『QM1 という名前のキュー・マネージャーの作成』.

関連概念

ディスク・スペースの要件

Windows システムでのハードウェア要件とソフトウェア要件

IBM MQ の概要

関連タスク

Windows での IBM MQ サーバーのインストール

IBM MQ サーバーの構成

QM1 という名前のキュー・マネージャーの作成

IBM MQ Explorer を使用して、`QM1` というキュー・マネージャーを作成します。 キュー・マネージャーは、IBM MQ メッセージング・ネットワークのメイン・コンポーネントです。

始める前に

IBM MQ がインストールされている必要があります。 インストールされていない場合、その方法について、7 ページの『Launchpad を使用したインストール』を参照してください。

このタスクについて

この例では、すべての名前が大文字になっています。IBM MQ の名前には大/小文字の区別があるため、すべての名前は大文字で入力する必要があります。

IBM MQ Explorer を使用してキュー・マネージャーを作成し、開始するには、以下のステップを完了してください。

手順

1. 管理者として IBM MQ Explorer を開始します。
2. 「**ナビゲーター**」ビューで、**Queue Managers** フォルダーを右クリックし、「**新規**」**>**「**キュー・マネージャー**」をクリックします。 「**キュー・マネージャーの作成**」ウィザードが開始します。
3. 「**キュー・マネージャー名**」フィールドに、QM1 と入力します。
4. Make this the default queue manager チェック・ボックスを選択します。
5. 「**送達不能キュー**」フィールドに、SYSTEM.DEAD.LETTER.QUEUE と入力します。

   これは、キュー・マネージャーを作成するときに自動的に作成される送達不能キューの名前です。
6. 他のフィールドを空のままにして「**完了**」をクリックするか、 このボタンが使用不可になっている場合は「**次へ**」をクリックします。

   ポート番号が既存のキュー・マネージャー (デフォルト構成の一部として作成されたキュー・マネージャーなど) と競合する場合は、「**完了**」ボタンが使用不可になります。 ウィザードで先に進んでデフォルトのポート番号を変更する必要があります。
7. 「**次へ**」をクリックした場合は、デフォルトを受け入れて、 各ページで「**次へ**」をクリックします。 ウィザードの最後のページに達すると、「**完了**」ボタンが使用可能になります。 指定のポート番号を 1415 などに変更し、「**完了**」をクリックします。

   IBM MQ で「**キュー・マネージャーの作成**」ダイアログ・ウィンドウが表示され、キュー・マネージャーが作成および開始されます。

次のタスク

キューの作成については、10 ページの『LQ1 という名前のキューの作成』を参照してください。

関連タスク

マルチプラットフォームでのキュー・マネージャーの作成と管理

LQ1 という名前のキューの作成

IBM MQ Explorer を使用して、キューを作成します。 キューは、メッセージを保管するために使用されるデータ構造体であり、IBM MQ キュー・マネージャーのオブジェクトです。

このタスクについて

このタスクでは、 IBM MQ Explorer を使用して IBM MQ オブジェクトを作成できます。

IBM MQ Explorer を使用してキューを作成および開始するには、以下のステップを完了してください。

手順

1. 「**ナビゲーター**」ビューで、**Queue Managers** フォルダーを展開します。
2. キュー・マネージャー「**QM1**」を展開します。
3. 「**キュー**」フォルダーを右クリックして、「**新規 > ローカル・キュー...**」をクリックします。 **新規のローカル・キュー** ウィザードが開始します。
4. **名前**フィールドに、LQ1 と入力します。
5. 「**完了 (Finish)**」 をクリックします。

   「**コンテンツ**」ビューに、新規のキュー LQ1 が表示されます。 キューが「**コンテンツ**」ビューに表示されない場合は、「**コンテンツ**」ビューの 上部にある「**リフレッシュ**」ボタンをクリックします。

次のタスク

これで、メッセージをキューに書き込む準備が整いました。 メッセージをキューに置く場合は、10 ページの『キュー LQ1 へのメッセージの書き込み』を参照してください。

キュー *LQ1* へのメッセージの書き込み

IBM MQ Explorer を使用して、メッセージをキュー LQ1 に書き込みます。

このタスクについて

このタスクでは、14 ページの『QM1 という名前のキュー・マネージャーの作成』の説明に従って QM1 と呼ばれるキュー・マネージャーが、また 10 ページの『LQ1 という名前のキューの作成』の説明に従って LQ1 と呼ばれるキューが既に作成されていることを前提とします。

IBM MQ Explorer を使用してキューにメッセージを書き込むには、以下のステップを実行します。

手順

1. 「**ナビゲーター**」ビューで、**Queue Managers** フォルダーを展開します。
2. 作成したキュー・マネージャー QM1 を展開します。
3. 「**キュー**」フォルダーをクリックします。 キュー・マネージャーのキューが「コンテンツ」ビューにリストされます。
4. 「コンテンツ」ビューでローカル・キュー LQ1 を右クリックし、 「**テスト・メッセージの書き込み...**」をクリックします。

   「**テスト・メッセージの書き込み**」ダイアログが開きます。
5. 「**メッセージ・データ**」 フィールドに、 Hello World などのテキストを入力し、 「**メッセージの書き込み**」をクリックします。

   「**メッセージ・データ**」フィールドがクリアされ、メッセージが キューに書き込まれます。
6. 「**クローズ**」をクリックします。

   「コンテンツ」ビューで、 LQ1 「**現在のキュー項目数**」 の値が 1 になったことに注意してください。 **現行キュー項目数** の列が表示されていない場合、**コンテンツ・ビュー**の右側にスクロールする必要があります。

次のタスク

キューからメッセージを読み取る場合は、11 ページの『キュー LQ1 からのメッセージの取得』を参照してください。

キュー *LQ1* からのメッセージの取得

IBM MQ Explorer を使用して、キュー LQ1 からメッセージを取得します。

このタスクについて

このタスクでは、10 ページの『キュー LQ1 へのメッセージの書き込み』の説明に従ってメッセージ QM1 が既に書き込まれていることを前提とします。

IBM MQ Explorer を使用してキューからメッセージを取得するには、以下のステップを完了します。

手順

1. 「**ナビゲーター**」ビューで「**キュー・マネージャー**」フォルダーを展開し、次に QM1 を展開します。
2. 「**キュー**」フォルダーをクリックします。
3. 「**コンテンツ**」ビューでローカル・キュー LQ1 を右クリックして、「**メッセージの参照...**」をクリックします。 「**メッセージ・ブラウザー**」が開き、 現在 QM1 に入っているメッセージのリストが表示されます。
4. 最後のメッセージをダブルクリックして、プロパティー・ダイアログを開きます。

   プロパティー・ダイアログの「**データ**」ページの「**メッセージ・データ**」フィールドに、人間が読める形式でメッセージの内容が表示されます。

次のタスク

後続のシナリオの指示に従って、その他の IBM MQ 機能について調べます。

キューイング・アプリケーションの作成、キュー・マネージャーへの接続と切断、パブリッシュ/サブスクライブ、およびオブジェクトのオープン/クローズについて学習するには、プロシージャー型キューイング・アプリケーションの作成を参照してください。

コマンド・ラインインターフェースを使用したインストールおよび構成

コマンド行を使用して IBM MQ を Windows にインストールし、サイレント・インストールを実行して環境変数をセットアップします。 インストール済み環境を検証した後、キュー・マネージャーの作成およびキューを作成し、その後でキューへのメッセージの書き込みおよびキューからのメッセージの取得を試行します。

サイレント・インストールを使用したインストール

コマンド行を使用して IBM MQ を Windows にインストールし、サイレント・インストールを実行し、インストール用の環境が正しくセットアップされていることを確認します。

始める前に

この作業を始める前に、以下の検査を完了してください。

- インストール時に、ローカル管理者権限が必要です。 その権限は Windows の機能で定義します。
- マシン名にスペースが含まれていないことを確認します。
- 十分なディスク・スペースがあることを確認します。 詳しくは、ディスク・スペース要件 (Multiplatforms) を参照してください。
- IBM MQ ユーザーの Windows ドメイン・ユーザー ID を定義する必要があるかどうかを判別します。

IBM MQ をインストールする前に、システムがハードウェアとソフトウェアの要件を満たしていることを確認してください。 サポートされているすべてのプラットフォームに関するハードウェア要件とソフトウェア要件の最新の詳細情報については、IBM MQ のシステム要件 を参照してください。

このタスクについて

このシナリオは、ご使用のマシンに初めて IBM MQ をインストールすること、およびデフォルトの場所を使用することを想定しています。 初期設定では、 IBM MQ 9.0 プログラム・ファイルのロケーションは `C:&#xa5;Program Files&#xa5;IBM&#xa5;MQ` であり、データおよびログ・ファイルのロケーションは `C: ¥ProgramData¥IBM¥MQ` です。

注 : マシン上に IBM MQ の前のインストール済み環境がある場合、プログラムおよびデータ・ファイルのデフォルトの場所は異なる可能性があります。 詳しくは、プログラムおよびデータのディレクトリーの場所を参照してください。 以前にこのシナリオを既に実行したことがあり、デフォルトのロケーションを使用して単一のフレッシュ・インストールでこのシナリオを繰り返す場合、シナリオを再度開始する前に、以前のインストールを削除します。 マシンから既存の IBM MQ インスタンスをアンインストールする方法については、17 ページの『IBM MQ のアンインストール』 を参照してください。

IBM MQ にある Windows は、 MSI テクノロジーを使用してソフトウェアをインストールします。 MSI テクノロジーを使用したインストールについて詳しくは、msiexec を使用した拡張インストールを参照してください。

コマンド・ラインを使用して IBM MQ をインストールするには、以下のパラメーターを指定する必要があります。

- `/i "`*`MQ_INSTALLATION_MEDIA`*`\MSI\IBM MQ.msi"` ここで、 *`MQ_INSTALLATION_MEDIA`* は `IBM MQ.msi` ファイルの場所です。 この引数は、.msi ファイルの場所を指定します。
- `/l*v` *`USER_LOGFILE_LOCATION`*`\install.log` ここで、 *`USER_LOGFILE_LOCATION`* は、インストール・ログの書き込み先です。

  注 : コマンドを実行する前に、作成したい `install.log` のあるフォルダが存在する必要があります。
- `/q[n|b|r|f]` `/q` は、 `n`、 `b`、 `r`、または `f` のいずれかとペアにする必要があります。 コマンド・プロンプトで **msiexec** コマンドを実行すると、正しい使用法を示すヘルプ・ファイルが開きます。

- USEINI="*RESPONSE_FILE*"。ここで、*RESPONSE_FILE* は、サイレント・インストールで使用される応答ファイルの名前と場所です。 このシナリオでは、 IBM MQ インストール・メディアに含まれているサンプル Response.ini ファイルを使用します。
- TRANSFORMS="*TRANSFORM_FILE*"。ここで、*TRANSFORM_FILE* はインストールに適用される変換ファイルの名前です。 このシナリオでは、米国英語変換 1033.mst を使用します。
- AGREETOLICENSE="YES"。このパラメーターは必ず指定します。そうしないと、インストールが完了しません。
- ADDLOCAL="Server"。このパラメーターは、インストールするコンポーネントをリストします。

手順

1. コマンド・ラインを使用してサイレント・インストールを行います。
   a) 昇格されたコマンド・プロンプトからサイレント・インストールを呼び出すには、**Windows タスクバー**の「**スタート**」ボタンをクリックして、「**プログラムとファイルの検索**」フィールドに cmd を入力します。 **cmd.exe** プログラムを右クリックし、「**管理者として実行**」を選択します。
   b) Windows コマンド・プロンプトに、以下のコマンドを入力します。

   注 : ここでは、コマンドは複数行で示されていますが、実際には 1 行で入力する必要があります。

   ```
   msiexec /i "MQ_INSTALLATION_MEDIA\MSI\IBM MQ.msi"
   /l*v c:\wmqinslogs\install.log
   /q USEINI="MQ_INSTALLATION_MEDIA\Response.ini"
   TRANSFORMS="1033.mst"
   AGREETOLICENSE="yes"
   ADDLOCAL="Server"
   ```

   ここで、*MQ_INSTALLATION_MEDIA* は使用する IBM MQ インストール・メディアへのパスです。

   注 : コマンドを実行する前に、作成したい install.log のあるフォルダが存在する必要があります。

   コマンドの入力後、コマンド・ラインはプロンプトを返します。
   c) インストールの進行状況を表示するには、指定したログ・ファイルを開きます。 インストールが正常に完了すると、ログ・ファイルの下部から 2 段落上の Product: IBM MQ (Installation1) -- Installation operation completed successfully. というメッセージが表示されます。
   d) インストールが完了すると、サービスが始動し、システム・トレイに IBM MQ アイコンが表示されます。

   IBM MQ のインストールが完了し、IBM MQ サービスが開始しました。
2. **setmqenv** コマンドを使用して、インストール済み環境に環境変数をセットアップします。
   a) コマンド・ラインで以下のコマンドを入力します。

   ```
   "MQ_INSTALLATION_PATH/bin/setmqenv" -s
   ```

   ここで、*MQ_INSTALLATION_PATH* は IBM MQ がインストールされている場所を示しています。 bin フォルダーの **setmqenv** へのパスを引用符で囲んでいることを確認し、プロンプトがエラーを返さないようにします。

   注 : 初期設定のロケーションを使用した場合は、インストーレーションへの経路は C:¥Program Files¥IBM¥MQ になります。
   b) 以下のコマンドを入力して、環境が正しく設定されていることを確認します。

   ```
   dspmqver
   ```

   コマンドが正常に完了して、予想したバージョン番号とインストール名が返されたら、環境は正しく設定されています。 メッセージに次の行が含まれているはずです。

```
Version: n.n.n.n
```

n.n.n.n はバージョン番号です。 デフォルト以外のインストール名を指定しなかった場合は、次の行が含まれます。

```
InstName: Installation1
```

サイレント・インストールを使用した IBM MQ のインストールが正常に完了しました。

タスクの結果

IBM MQ サイレント・インストールを実行し、環境が正常にセットアップされたことを確認しました。

次のタスク

- Prepare IBM MQ Wizard を実行することができます。
- 14 ページの『QM1 という名前のキュー・マネージャーの作成』 の指示に従ってください。

インストール中に問題が発生した場合は、**msiexec** コマンドで指定した場所にあるインストール・ログを確認してください。このシナリオでは、ログ・ファイルの場所は `c:\wmqinslogs\install.log` です。ログに指定されている処置を実行し、インストールを再度実行します。 コマンドに指定したパラメーターを調べて、必要なパラメーターをすべて指定したことも確認してください。

関連タスク

msiexec を使用したサーバーのインストール
msiexec での変換の使用
IBM MQ のインストール - 概要

QM1 という名前のキュー・マネージャーの作成

コマンド行インターフェースを使用して、QM1 という名前のキュー・マネージャーを作成します。 キュー・マネージャーは、IBM MQ メッセージング・ネットワークのメイン・コンポーネントです。

始める前に

IBM MQ がインストールされている必要があります。 インストールされていない場合、その方法について、12 ページの『サイレント・インストールを使用したインストール』 を参照してください。

このタスクについて

この例では、すべての名前が大文字になっています。IBM MQ の名前には大/小文字の区別があるため、すべての名前は大文字で入力する必要があります。

手順

1. 管理者としてコマンド・プロンプトを開きます。
2. 以下のコマンドを入力して、QM1 という名前でキュー・マネージャーを作成します。

```
crtmqm QM1
```

システムによってキュー・マネージャーが作成されると、以下の出力が表示されます。

```
C:\>crtmqm QM1
IBM MQ queue manager created.
Creating or replacing default objects for QM1.
Default objects statistics : 61 created. 0 replaced. 0 failed.
Completing setup.
Setup completed.
```

キュー・マネージャーが作成され、停止しています。 キュー・マネージャーを管理し、そのキューのメッセージの読み取り/書き込みを実行するには、キュー・マネージャーを開始する必要があります。

3. 次のコマンドを入力して、キュー・マネージャーを開始します。

```
strmqm QM1
```

キュー・マネージャーが正常に開始されると、次の出力が表示されます。

```
C:\>strmqm QM1
IBM MQ queue manager 'QM1' starting.
5 log records accessed on queue manager 'QM1' during the log replay phase.
Log replay for queue manager 'QM1' complete.
Transaction manager state recovered for queue manager 'QM1'.
IBM MQ queue manager 'QM1' started.
```

キュー・マネージャーが開始されます。

次のタスク

キューの作成については、15 ページの『LQ1 という名前のキューの作成』を参照してください。

関連タスク

マルチプラットフォームでのキュー・マネージャーの作成と管理

LQ1 という名前のキューの作成

コマンド行インターフェースを使用して、キューを作成します。 キューは、メッセージを保管するために使用されるデータ構造体であり、IBM MQ キュー・マネージャーのオブジェクトです。

このタスクについて

IBM MQ オブジェクトを作成する 3 つの方法があります。

- コマンド行。
- IBM MQ Explorer.
- プログラマブル・インターフェースの使用。

このタスクでは、コマンド行を使用して、IBM MQ オブジェクトを作成できます。

コマンド行インターフェースには、IBM MQ Script Commands (MQSC) と呼ばれるスクリプト言語があります。 スクリプト・ツール **runmqsc** を使用し、キュー・マネージャーに対してスクリプトを実行します。 コマンド行インターフェースを使用してキューを作成および開始するには、以下のステップを実行します。

手順

1. 次のコマンドを入力して、スクリプト・ツールを開始します。

```
runmqsc QM1
```

スクリプト・ツールを開始すると、次の出力が表示されます。

```
C:\>runmqsc QM1
5724-H72 (C) Copyright IBM Corp. 1994, 2024. ALL RIGHTS RESERVED.
Starting MQSC for queue manager QM1.
```

これで、ツールは MQSC コマンドを受け入れることができます。

2. 以下の MQSC コマンドを入力して、LQ1 という名前のローカル・キューを作成します。

```
define qlocal(LQ1)
```

キューが作成されると、以下の出力が表示されます。

```
define qlocal(LQ1)
2 : define qlocal(LQ1)
AMQ8006: IBM MQ queue created.
```

3. 次の MQSC コマンドを入力して、スクリプト・ツールを停止します。

```
end
```

スクリプト・ツールが終了する時、以下の出力が表示されます。

```
One MQSC command read.
No commands have a syntax error.
All valid MQSC commands were processed.

C:\>
```

次のタスク

これで、メッセージをキューに書き込む準備が整いました。 メッセージをキューに置く場合は、16 ページの『キュー LQ1 へのメッセージの書き込み』を参照してください。

キュー *LQ1* へのメッセージの書き込み

コマンド行インターフェースを使用してキュー LQ1 にメッセージを書き込みます。

このタスクについて

IBM MQ には、**amqsput** という名前のサンプル・アプリケーションが付属しています。 このアプリケーションは、事前定義されたキューにメッセージを書き込みます。

コマンド行インターフェースを使用してキューにメッセージを書き込むには、以下のステップを実行します。

手順

1. 以下のコマンドを入力することにより、**amqsput** サンプル・アプリケーションを使用して、メッセージをキュー LQ1 に書き込みます。

```
amqsput LQ1 QM1
```

サンプル・アプリケーションを開始すると、次の出力が表示されます。

```
C:\>amqsput LQ1 QM1
Sample AMQSPUT0 start
target queue is LQ1
```

2. Hello World と入力して、Enter キーを押します。 「Hello World」というテキストを格納したメッセージを QM1 という名前のキュー・マネージャーによって管理されるキュー LQ1 に入れました。
3. **amqsput** を終了するには、**Enter** キーを押します。 以下の出力が表示されます。

```
C:\>amqsput LQ1 QM1
Sample AMQSPUT0 start
target queue is LQ1
Hello World

Sample AMQSPUT0 end
```

次のタスク

キューからメッセージを読み取る場合は、17 ページの『キュー LQ1 からのメッセージの取得』を参照してください。

キュー *LQ1* からのメッセージの取得

コマンド行インターフェースを使用してキュー LQ1 からメッセージを取得します。

このタスクについて

IBM MQ には、**amqsget** という名前のサンプル・アプリケーションが付属しています。 このアプリケーションは、キューからメッセージを読み取ります。

コマンド行インターフェースを使用してキューからメッセージを読み取るには、以下のステップを実行します。

手順

以下のコマンドを入力することにより、**amqsget** サンプル・アプリケーションを使用して、キュー LQ1 のメッセージを読み取ります。

```
amqsget LQ1 QM1
```

サンプル・アプリケーションを開始すると、次の出力が表示されます。

```
C:\>amqsget LQ1 QM1
Sample AMQSGET0 start
message <Hello World>
no more messages
Sample AMQSGET0 end
```

メッセージを読み取った 30 秒後に、**amqsget** アプリケーションは終了します。

次のタスク

後続のシナリオの指示に従って、その他の IBM MQ 機能について調べます。

キューイング・アプリケーションの作成、キュー・マネージャーへの接続と切断、パブリッシュ/サブスクライブ、およびオブジェクトのオープン/クローズについて学習するには、プロシージャー型キューイング・アプリケーションの作成を参照してください。

IBM MQ のアンインストール

IBM MQ を停止してからアンインストールします。アンインストールすることによって、すべてのキュー・マネージャーとそのオブジェクトも削除されます。 このタスクが終了すると、IBM MQ を再インストールする準備ができます。

このタスクについて

このタスクでは、ダウンロードしたインストール・イメージを使用して Windows 上の IBM MQ をアンインストールする手順について説明します。

「準備作業」シナリオでは、ランチパッドまたはコマンド・ラインを使用して IBM MQ をインストールするオプションについて説明します。 IBM MQ を複数インストールすることはできますが、このシナリオでは、単一サーバーでの新規インストールに基づきます。 そのため、このシナリオを繰り返す、または異なるインストール方法を試す場合は、まず既存の IBM MQ コンポーネント (既存のすべてのキュー・マネージャーとそのオブジェクトを含む) をアンインストールする必要があります。 そうすることによって、フレッシュ・インストールを再び開始できます。

このセクションで示すその他のシナリオの一部でフレッシュ・インストールを実行できるようにするためにも、アンインストールする必要が生じる場合があります。

手順

1. IBM MQ サービスを停止します。
   a) システム・トレイの「**IBM MQ**」アイコンを右クリックして、「**IBM MQ の停止**」をクリックして IBM MQ サービスを停止します。
   以下のメッセージがダイアログに表示されます。
   ```
   IBM MQ インストール済み環境 "Installation1" をシャットダウンすると、実行中のすべてのキュー・マネージャーが終了します。
   IBM MQ は、Microsoft Failover Cluster の制御下にあるプロセスを除き、そのインストールのために処理します。
   続行してよろしいですか?
   ```
   b)「**はい**」 をクリックし、 IBM MQ が停止するまで待ちます。
   c) IBM MQ が停止したら、システム・トレイの **IBM MQ** アイコンを右クリックして、 「**終了**」 をクリックします。
2. アンインストール・プロセスを開始します。
   インストール・イメージを含む圧縮ファイルをダウンロードして、一時ディレクトリーに解凍します。そのディレクトリーにナビゲートし、 `setup.exe` をダブルクリックします。
   IBM MQ **インストール・ランチパッド** ・ウィンドウが表示されます。
3. IBM MQ を削除します。
   a)「**IBM MQ インストール**」をクリックします。
   b)「 **IBM MQ インストーラーの起動**」 をクリックし、「 IBM MQ **プログラム保守ペイン** 」がウェルカム・メッセージとともに表示されるまで 「**次へ**」 をクリックします。
   このペインが表示されない場合、IBM MQ for Windows はこのマシンには現在インストールされていません。
   c)「**既存のインスタンスの維持またはアップグレード (Maintain or upgrade an existing instance)**」をクリックします。 「**Installation1**」を選択して削除します。 「**次へ**」をクリックし、「**プログラム・メンテナンス**」ペインで「**除去**」をクリックしてから、「**次へ**」をクリックします。
   「サーバー機能の除去」ペインが表示されます。
   d)「**除去**」を選択して、既存のキュー・マネージャーとそのオブジェクトを除去します。
   次へ をクリックします。
   除去対象のインストール内容の要約を示した「IBM MQ の除去」ペインが表示されます。
   e)「**除去**」をクリックして、先に進みます。
   ロックされたファイルが見つかったことを示すメッセージが表示された場合は、 IBM MQ プログラムが実行されていないことを確認してください。 Windows システムでの IBM MQ のアンインストールを参照してください。
   IBM MQ がアンインストールされると、完了を知らせるメッセージが表示されます。
   f)「**完了 (Finish)**」 をクリックします。
   IBM MQ が正常にアンインストールされました。

関連タスク
Windows システムでの IBM MQ のアンインストール

次のタスク

『IBM MQ の概要』のシナリオを完了した後に実行する内容。

Windows AIX Linux インストールおよびアップグレードに役立つチュートリアルについては、 AIX®、 Linux®、および Windows で IBM MQ をインストールおよびアップグレードするためのチュートリアルのコレクションを参照してください。 チュートリアルでは、以下について説明します。

- IBM MQ 用のホストを準備します。
- IBM MQ コードのダウンロード。
- IBM MQ コードのインストールとアンインストール、およびフィックスパックの適用。

- あるバージョンの IBM MQ から別のバージョンへのアップグレード、およびあるホストから別のホストへのキュー・マネージャーの移動。

IBM MQ の製品資料には、その他にも参照できるトピックがあります。 以下のセクションを参照できます。

- 管理 IBM MQ

  IBM MQ には、使用できる制御コマンドがあります。 このシナリオでは、**crtmqm** および **strmqm** の 2 つのコマンドを使用します。 このセクションでは、メッセージ・キューイングに関する優れた概要も提供されます。

- MQSC コマンドを使用した IBM MQ の管理

  このシナリオでは、`define qlocal('LQ1')` コマンドを使用して、LQ1 という名前のローカル・キューを定義します。 このコマンドは MQSC コマンドです。 IBM MQ システム管理者は、これらのコマンドを使用してキュー・マネージャーを管理します。 このセクションでは、コマンドの概要と使用方法について説明しています。 MQSC コマンド・リファレンス・セクションでは、コマンドの詳細がアルファベット順に説明されています。

- キュー・マネージャー・クラスターの構成

  このセクションでは、クラスターとして知られる仮想グループ内のキュー・マネージャーを編成、使用、および管理する方法について説明します。 クラスタリングによって、クラスター内の各キュー・マネージャーは確実に同じクラスター内の他のすべてのキュー・マネージャーを認識します。 また、クラスタリングによって、複雑なキュー・マネージャー・ネットワークの管理がより簡単になります。

Point-to-Point シナリオ

2 つの IBM MQ キュー・マネージャーを Point-to-Point トポロジーで接続して、分散キューイングを使用可能にします。

このタスクについて

2 つのキュー・マネージャーと適切なキューおよびチャネルを作成して、片方向 Point-to-Point メッセージング・インフラストラクチャーを作成します。 別々のホスト上にキュー・マネージャーを作成して、ネットワークを介した通信を使用可能にします。 このシナリオの拡張として、チャネルに Transport Layer Security を追加してデータのセキュア通信を使用可能にします。

ソリューションの計画

Point-to-Point メッセージングは、IBM MQ で最も単純なメッセージング形式です。 Point-to-Point メッセージングでは、メッセージを送信するためには、その前に送信側アプリケーションが受信側アプリケーションについての特定の情報を知っておかなければなりません。 送信側アプリケーションは、リモート・キューをアドレス指定する方法を必要とします。 Point-to-Point メッセージングを使用して、サンプル・アプリケーションでリモート・キュー・マネージャーにメッセージを送信します。

概要: 実現する論理トポロジー

シナリオが完成した後に実現する論理トポロジー。

Point-to-Point インフラストラクチャーでは、複数の異なるホスト・マシン上のキュー・マネージャー間で一方向メッセージングが可能です。 ホスト 1 上のキュー・マネージャー 1 がホスト 2 上のキュー・マネージャー 2 にメッセージを送信します。 このシナリオが完成すると、実現するトポロジーは図 1 のようになります。

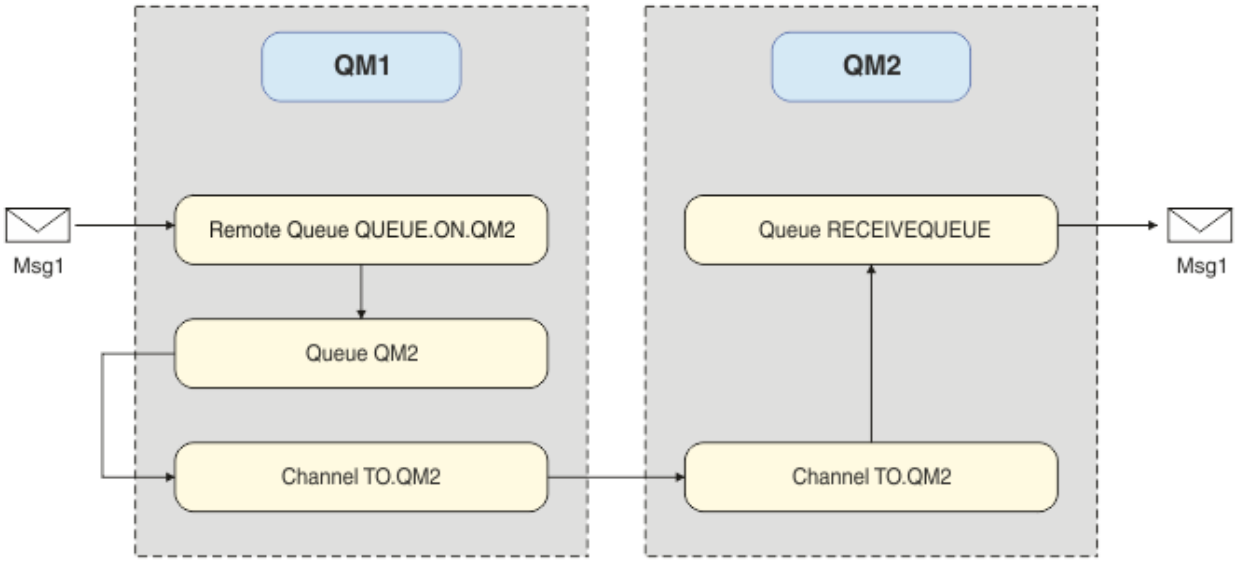


図 *2. QM1* が *QM2* にメッセージを送信

基本概念と重要な用語

Point-to-Point シナリオを完成させるために知っておく必要のある基本概念と重要な用語の説明。

基本概念

IBM MQ によって、アプリケーションは、キューに対するメッセージの読み取り/書き込みを実行できます。メッセージを読み取るアプリケーションは、メッセージを書き込むアプリケーションから独立しています。2 つのアプリケーションを同時に実行する必要はありません。 メッセージを読み取るアプリケーションが使用不可である場合、アプリケーションがメッセージを読み取るまで、そのメッセージは IBM MQ キューで待機します。

重要な用語

以下に、メッセージ・キューイングに関する重要な用語のリストを示します。

メッセージ・キューイングに関する重要な用語。

用語	説明
キュー・マネージャー	キュー・マネージャーは、所有するキューを管理し、受信したすべてのメッセージを適切なキューに格納します。
メッセージ	メッセージを使用するアプリケーションにとって、メッセージは意味のあるバイト・ストリングです。 メッセージは、ある 1 つのアプリケーション・プログラムから別のアプリケーションに情報を転送するために使用されます。 アプリケーションは、同じコンピューターで実行することも、異なるコンピューターで実行することもできます。
ローカル・キュー	ローカル・キューとは、メッセージを保管するためのデータ構造体です。 このキューは、通常キューであることも、伝送キューであることも可能です。 通常キューは、キュー・マネージャーからメッセージを直接読み取るアプリケーションによって読み取られるメッセージを保持します。 伝送キューは、別のキュー・マネージャーに転送中のメッセージを保持します。
リモート・キュー	リモート・キューは、別のキュー・マネージャーにメッセージをアドレッシングするために使用されます。
チャネル	チャネルは、キュー・マネージャー間でメッセージを送受信するために使用されます。
リスナー	リスナーは、他のキュー・マネージャーまたはクライアント・アプリケーションからネットワーク要求を受け取るプロセスで、関連付けられたチャネルを始動します。

ソリューションの実装

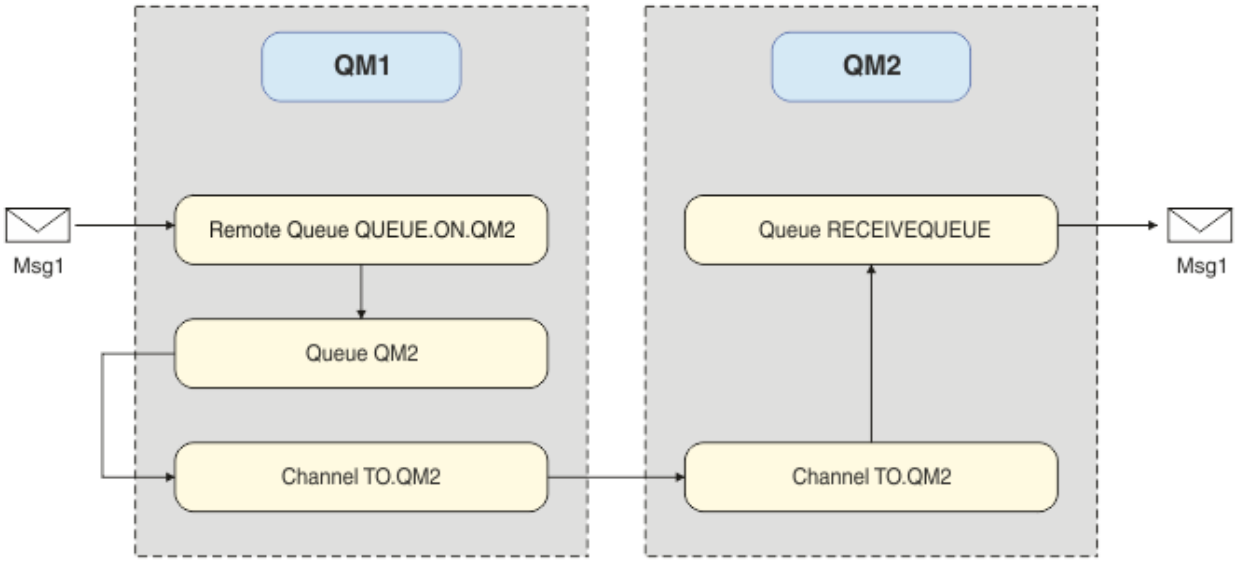
シナリオのソリューションを実装します。 2 つの別個のホスト上に 2 つの IBM MQ キュー・マネージャー (メッセージを送信するためのソース・キュー・マネージャーとメッセージを受信するためのターゲット・キュー・マネージャー) を作成します。

始める前に

このシナリオの開始点は、検証済みの既存 IBM MQ インストール済み環境です。 IBM MQ のインストールについては、Windows での IBM MQ サーバーのインストールの手順に従ってください。

このタスクについて

コマンド行インターフェースを使用してキュー・マネージャーを 2 つ作成し、必要なリスナー、キュー、およびチャネルを定義します。 実現する論理トポロジーは、ソリューションの実装によって追加された機能を示しています。



キュー・マネージャーの作成

ターゲット・キュー・マネージャーにメッセージを送信するための IBM MQ キュー・マネージャーを作成します。

始める前に

- IBM MQ がインストールされている必要があります。 IBM MQ のインストールについて詳しくは、インストールおよびアンインストールを参照してください。

このタスクについて

コマンド行インターフェースを使用して IBM MQ キュー・マネージャーを作成します。

手順

1. 名前 QM1 でキュー・マネージャーを作成します。 コマンド行で次のように入力します。

```
crtmqm QM1
```

   次のメッセージが表示され、キュー・マネージャーが作成されたことを確認できます。

```
IBM MQ queue manager created.
Creating or replacing default objects for QM1.
Default objects statistics : 61 created. 0 replaced. 0 failed.
Completing setup.
Setup completed.
```

2. キュー・マネージャーを始動します。 コマンド行で次のように入力します。

```
strmqm QM1
```

   次のメッセージが表示され、キュー・マネージャーが始動したことを確認できます。

```
IBM MQ queue manager 'QM1' starting.
5 log records accessed on queue manager 'QM1' during the log replay phase.
Log replay for queue manager 'QM1' complete.
Transaction manager state recovered for queue manager 'QM1'.
IBM MQ queue manager 'QM1' started.
```

タスクの結果

IBM MQ キュー・マネージャーの QM1 が作成されて開始されました。

次のタスク

QM1 で使用するキューを作成するには、22 ページの『キューの作成』の手順に従ってください。

キューの作成

IBM MQ キュー・マネージャーによって管理される IBM MQ キューを作成します。

始める前に

22 ページの『キュー・マネージャーの作成』の説明に従ってセットアップされた IBM MQ キュー・マネージャーが必要です。

このタスクについて

MQSC インターフェースを開始して、キュー・マネージャーに接続されるオブジェクトを管理します。 伝送キューおよびリモート・キュー定義を作成します。 **MQSC** インターフェースを終了します。

手順

1. コマンド行で次のように入力します。

```
runmqsc QM1
```

   確認メッセージの後、このツールはコマンドを受け入れることができます。
2. QM2 という伝送キューを作成します。 リモート・キュー・マネージャーと同じ名前を伝送キューに付けることをお勧めします。 MQSC インターフェースで次のように入力します。

```
DEFINE QLOCAL(QM2) DESCR('Transmission queue to QM2') USAGE(XMITQ)
```

   伝送キューが作成されます。
3. QUEUE.ON.QM2 というリモート・キュー定義を作成します。 リモート・キュー定義は、リモート・ホスト上のローカル・キューに付けられた名前を参照する必要があります。 MQSC インターフェースで次のように入力します。

```
DEFINE QREMOTE(QUEUE.ON.QM2) DESCR('Remote queue for QM2') XMITQ(QM2) RNAME(RECEIVEQUEUE)
RQMNAME(QM2)
```

   リモート・キュー定義が作成されます。
4. end と入力して MQSC インターフェースを終了します。

次のタスク

ターゲット・キュー・マネージャーへの接続に使用される送信側チャネルを作成するには、23 ページの『送信側チャネルの作成』の手順に従ってください。

送信側チャネルの作成

ソース・キュー・マネージャーに送信側チャネルを作成します。このチャネルは、ターゲット・キュー・マネージャーへの接続に使用されます。

始める前に

TLS を使用するチャネルを作成するには、29 ページの『TLS を使用するチャネルの作成』の手順に従ってください。 この作業は、TLS セキュリティーなしでソリューションをテストする場合は、後で行えます。

このタスクについて

MQSC インターフェースを開始して、キュー・マネージャーに接続されるオブジェクトを管理し、送信側チャネルを作成します。 このチャネルは、QM2 という名前のターゲット・キュー・マネージャーへの接続に使用されます。

手順

1. コマンド行で次のように入力します。

```
runmqsc QM1
```

   確認メッセージの後、このツールはコマンドを受け入れることができます。
2. TO.QM2 という送信側チャネルを作成します。 MQSC インターフェースで次のように入力します。

```
DEFINE CHANNEL(TO.QM2) CHLTYPE(SDR) CONNAME(' remoteHost ') TRPTYPE(TCP) XMITQ(QM2)
```

   注 : 変数 *remoteHost* は、ターゲット・キュー・マネージャーのホスト名または IP アドレスです。

   送信側チャネルが作成されます。

次のタスク
分散キュー・マネージャー・トポロジーを作成するには、24 ページの『分散キュー・マネージャー・トポロジーの作成』の手順に従ってください。

分散キュー・マネージャー・トポロジーの作成

Point-to-Point メッセージングは、IBM MQ で最も単純なメッセージング形式です。 Point-to-Point メッセージングでは、メッセージを送信するためには、その前に送信側アプリケーションが受信側アプリケーションについての特定の情報を知っておかなければなりません。 送信側アプリケーションは、リモート・キューをアドレス指定する方法を必要とします。 Point-to-Point メッセージングを使用して、サンプル・アプリケーションで 2 番目のキュー・マネージャーにメッセージを送信します。

始める前に
22 ページの『キュー・マネージャーの作成』の説明に従って、ソース・キュー・マネージャーをセットアップしておく必要があります。

このタスクについて
リモート・ホスト上にターゲット・キュー・マネージャーを作成します。 サンプル・アプリケーションを使用して、ソース・キュー・マネージャーとターゲット・キュー・マネージャーの間の通信を検証します。

キュー・マネージャーの作成
リモート・キュー・マネージャーに対してメッセージを受信するための IBM MQ キュー・マネージャーを作成します。

始める前に
IBM MQ がインストールされている必要があります。 IBM MQ のインストールについて詳しくは、Windows での IBM MQ サーバーのインストールを参照してください。

このタスクについて
コマンド行インターフェースを使用して IBM MQ キュー・マネージャーを作成します。

手順
1. 名前 QM2 でキュー・マネージャーを作成します。 コマンド行で次のように入力します。

   ```
   crtmqm QM2
   ```

   次のメッセージが表示されます。

   ```
   IBM MQ queue manager created.
   Creating or replacing default objects for QM2.
   Default objects statistics : 61 created. 0 replaced. 0 failed.
   Completing setup.
   Setup completed.
   ```

2. キュー・マネージャーを始動します。 コマンド行で次のように入力します。

   ```
   strmqm QM2
   ```

   次のメッセージが表示され、キュー・マネージャーが始動したことを確認できます。

   ```
   IBM MQ queue manager 'QM2' starting.
   5 log records accessed on queue manager 'QM2' during the log replay phase.
   Log replay for queue manager 'QM2' complete.
   Transaction manager state recovered for queue manager 'QM2'.
   IBM MQ queue manager 'QM2' started.
   ```

タスクの結果
IBM MQ キュー・マネージャーの QM2 が作成されて開始されました。

次のタスク
QM2 で使用するキューを作成するには、25 ページの『キューの作成』の手順に従ってください。

キューの作成
ターゲット・キュー・マネージャーでのメッセージの受信に使用されるローカル・キュー、およびインバウンド・チャネル接続を受け入れるリスナーを作成します。

このタスクについて
runmqsc スクリプト・ツールの開始後に、MQSC コマンドを使用して、ローカル・キューとリスナーを作成できます。

手順
1. 次のコマンドを入力して、スクリプト・ツールを開始します。

   ```
   runmqsc QM2
   ```

   ツールが開始されたことを確認するメッセージが表示されます。
2. RECEIVEQUEUE というローカル・キューを作成します。 キューの名前は、ソース・キュー・マネージャー上のリモート・キュー定義で示されている名前と同じでなければなりません。 MQSC インターフェースで次のように入力します。

   ```
   DEFINE QLOCAL(RECEIVEQUEUE) DESCR('Receiving queue')
   ```

   ローカル・キューが作成されます。
3. LISTENER1 というリスナーを作成します。 MQSC インターフェースで次のように入力します。

   ```
   DEFINE LISTENER(LISTENER1) TRPTYPE(TCP) PORT(1414) CONTROL(QMGR)
   ```

   注: ポート 1414 は IBM MQ のデフォルト・ポートです。 異なるポート番号を選択する場合、送信側キュー・マネージャーの送信側チャネルの CONNAME にそれを追加する必要があります。
4. リスナーを開始することで、インバウンド接続の受け入れ準備ができた状態にします。 MQSC インターフェースで次のように入力します。

   ```
   START LISTENER(LISTENER1)
   ```

   注: オプション CONTROL(QMGR) を使ってリスナーが作成されたため、次回にキュー・マネージャーが開始されるとき、リスナーも自動的に開始されます。
5. end と入力して **MQSC** インターフェースを終了します。

次のタスク
受信側チャネルを作成してソース・キュー・マネージャーとターゲット・キュー・マネージャーの間の接続を作成するには、25 ページの『受信側チャネルの作成』の手順に従ってください。

受信側チャネルの作成
ターゲット・キュー・マネージャーの受信側チャネルを作成して、ソース・キュー・マネージャーとターゲット・キュー・マネージャーの間の通信を可能にします。

始める前に

TLS を使用するチャネルを作成するには、29 ページの『TLS を使用するチャネルの作成』の手順に従ってください。 この作業は、TLS セキュリティーなしでソリューションをテストする場合は、後で行えます。

このタスクについて

MQSC インターフェースを使用して、QM2 によって管理される受信側チャネルを作成します。

手順

1. コマンド行で次のように入力します。

```
runmqsc QM2
```

   確認メッセージの後、このツールはコマンドを受け入れることができます。
2. TO.QM2 という受信側チャネルを作成します。 チャネルの名前は、ソース・キュー・マネージャー上の送信側チャネルと同じでなければなりません。 MQSC インターフェースで次のように入力します。

```
DEFINE CHANNEL(TO.QM2) CHLTYPE(RCVR) TRPTYPE(TCP)
```

   受信側チャネルが作成されます。

次のタスク

ソース・キュー・マネージャー上の送信側チャネルを開始し、これが次にターゲット・キュー・マネージャー上の受信側チャネルを開始するには、26 ページの『送信側チャネルの開始』の手順に従ってください。

送信側チャネルの開始

ソース・キュー・マネージャー上の送信側チャネルを開始します。ターゲット・キュー・マネージャー上の受信側チャネルも開始されます。 メッセージはソース・キュー・マネージャーからターゲット・キュー・マネージャーに送信できます。

このタスクについて

MQSC インターフェースを開始して、キュー・マネージャーに接続されるオブジェクトを管理します。 送信側チャネルを開始してターゲット・キュー・マネージャーに接続することで、通信を可能にします。 ソース・チャネルが開始されると、受信側チャネルが自動手的に開始します。

手順

1. コマンド行で次のように入力します。

```
runmqsc QM1
```

   確認メッセージの後、このツールはコマンドを受け入れることができます。
2. ソース・キュー・マネージャー上の送信側チャネルを開始します。 MQSC インターフェースで次のように入力します。

```
START CHANNEL(TO.QM2)
```

   送信側チャネルが開始します。ターゲット・キュー・マネージャー上の受信側チャネルも開始されます。
3. チャネルが実行中であることを確認します。 MQSC インターフェースで次のように入力します。

```
DISPLAY CHSTATUS(TO.QM2)
```

チャネルが実行されている場合、STATUS(RUNNING) という報告が表示されます。 STATUS でこれ以外の値が報告される場合は、エラー・ログを確認してください。

次のタスク
ソース・キュー・マネージャーがターゲット・キュー・マネージャーにメッセージを送信できることを検証するには、27 ページの『ソリューションの検証』の手順に従ってください。

ソリューションの検証
ソース・キュー・マネージャーがリモート・キューにメッセージを書き込めることを検証します。 ターゲット・キュー・マネージャーがキューからメッセージを取得できることを検証します。

このタスクについて
サンプル・アプリケーションの **amqsput** と **amqsget** を使用してソリューションを検証します。

手順
1. ソース・キュー・マネージャーからターゲット・キュー・マネージャー QM2 にメッセージを送信します。
   a) コマンド行インターフェースで次のように入力します。

```
amqsput QUEUE.ON.QM2 QM1
```

   ターゲット・キュー・マネージャーにメッセージを送信するには、リモート・キュー定義の名前を使用する必要があります。
   次のメッセージが表示されます。

```
Sample AMQSPUT0 start
target queue is QUEUE.ON.QM2
```

   b) `Hello world.` と入力して、Enter キーを 2 回押します。
2. ターゲット・キュー・マネージャーでメッセージを取得します。
   a) コマンド行インターフェースで次のように入力します。

```
amqsget RECEIVEQUEUE QM2
```

   次のメッセージが表示されます。

```
Sample AMQSGET0 start
message <Hello world.>
no more messages
Sample AMQSGET0 end
```

タスクの結果
ターゲット・キュー・マネージャーがソース・キュー・マネージャーからのメッセージを受信しました。 これで、Point-to-Point 通信が実現されたことを検証しました。

次のタスク
ソリューションにセキュリティーを追加する場合は、27 ページの『Point-to-Point トポロジーの保護』の手順に従ってください。

Point-to-Point トポロジーの保護
実稼働環境でメッセージを伝送できるように、Point-to-Point トポロジーを保護します。

このタスクについて
適切なレベルのアクセス権限が付与されるように、ソース・キュー・マネージャー・オブジェクトとターゲット・キュー・マネージャー・オブジェクトを保護します。 どのユーザー・グループがキューおよびキュー・マネージャーへのアクセス権限を持つかを定義します。 デジタル署名付き証明書を使用して Transport Layer Security (TLS) を使用して接続することによって、ネットワーク接続を保護します。

ソース・キュー・マネージャー・オブジェクトの保護
ソース・キュー・マネージャー上のオブジェクトに対する許可の値を設定します。

このタスクについて
setmqaut コマンドを使用して、アプリケーションを実行するユーザー・グループに権限を付与します。

手順
1. 指定したユーザー・グループにキュー・マネージャーへの *connect* 許可を付与するには、コマンド行インターフェースで次のように入力します。

```
setmqaut -m QM1 -t qmgr -g userGroup +connect
```

2. 指定したユーザー・グループにリモート・キュー定義に対する *put* 許可を付与するには、コマンド行インターフェースで次のように入力します。

```
setmqaut -m QM1 -t q -n "QUEUE.ON.QM2" -g userGroup +put
```

ターゲット・キュー・マネージャー・オブジェクトの保護
ターゲット・キュー・マネージャー上のオブジェクトに対する許可の値を設定します。

このタスクについて
setmqaut コマンドを使用して、アプリケーションを実行するユーザー・グループに権限を付与します。

手順
1. 指定したユーザー・グループにキュー・マネージャーへの *connect* 許可を付与するには、コマンド行インターフェースで次のように入力します。

```
setmqaut -m QM2 -t qmgr -g userGroup +connect
```

2. 指定したユーザー・グループにリモート・キュー定義に対する *get* 許可を付与するには、コマンド行インターフェースで次のように入力します。

```
setmqaut -m QM2 -t q -n "RECEIVEQUEUE" -g userGroup +get
```

ネットワークの保護
ソース・キュー・マネージャーとリモート・キュー・マネージャーの間のネットワーク接続を保護します。

このタスクについて
署名付き証明書を使用して、ソース・キュー・マネージャーとリモート・キュー・マネージャーの認証性を検査します。 メッセージを暗号化するために、TLS ネットワークを使用してメッセージを転送します。

TLS を使用するキュー・マネージャーの準備
IBM MQ キュー・マネージャーの鍵リポジトリーは、キュー・マネージャーの個人証明書と公開認証局 (CA) 証明書を保管するために使用されます。 IBM MQ キュー・マネージャーからの個人証明書要求は、CA によ

って署名されていなければなりません。公開証明書は、他のエンティティーが IBM MQ キュー・マネージャーを認証するために使用されます。

始める前に

公開認証局の証明書が 1 つのファイルに含まれていなければなりません。

このタスクについて

IBM MQ キュー・マネージャーの鍵リポジトリーを作成し、認証局の署名者証明書をインポートし、キュー・マネージャーの個人証明書要求を作成します。

手順

1. key.kdb というキュー・マネージャー用の CMS キー・リポジトリー・ファイルを作成します。Qmgrs\QM1\ssl ディレクトリーに移動し、コマンドラインで次のように入力します。

   ```
   runmqckm -keydb -create -db key.kdb -pw passw0rd -type cms -stash
   ```

   注: この単純な例では、パスワード passw0rd を使用しています。 これとは異なるパスワードを選んで、独自のパスワードを使用するようこれ以降の各コマンドを変更することができます。
2. ファイルに入っている CA 証明書を鍵リポジトリーに追加します。コマンド・ラインで次のように入力します。

   ```
   runmqckm -cert -add -file CA-certificate-file -db key.kdb -pw passw0rd -label TrustedCA
   ```
3. QM1req.req というリクエストファイルに書き込まれる個人証明書を要求します。

   コマンド・ラインに以下のように入力します。

   ```
   runmqckm -certreq -create -db key.kdb -pw passw0rd -label ibmwebspheremqqm1
   -dn CN="QM1" -size 1024 -file QM1req.req
   -sig_alg SHA1WithRSA
   ```

   この例ではデフォルトの証明書ラベル名が示されています。 必要に応じて独自の名前を設定できます。 詳細については、デジタル証明書ラベルを参照してください。
4. 証明書要求ファイルを CA に送ります。デジタル署名付き証明書が発行されます。 受け取った署名付き証明書ファイルを、キュー・マネージャーの鍵リポジトリーに取り込むことを想定した適した場所に入れます。
5. 署名付き個人証明書をキュー・マネージャーの鍵リポジトリーに取り込みます。

   ```
   runmqckm -cert -receive -file Signed-certificate-file -db key.kdb -pw passw0rd -format ascii
   ```
6. キュー・マネージャーごとに、キュー・マネージャー名を適宜変えて、これらのステップを実行します。

次のタスク

送信側チャネルと受信側チャネルを介したセキュア通信を使用可能にするには、29 ページの『TLS を使用するチャネルの作成』の手順に従ってください。

TLS を使用するチャネルの作成

TLS を使用する新しいチャネルを作成して接続を作成します。

始める前に

TLS を使用するチャネルを介して通信するには、まず、接続の両端に必要な証明書がなければなりません。必要な証明書を作成するには、28 ページの『TLS を使用するキュー・マネージャーの準備』の手順に従ってください。

このタスクについて

TLS 属性セットを使ってチャネルを定義するには、MQSC インターフェースを使用します。 以前のステップで TLS を使わずにチャネルを定義した場合でも、REPLACE キーワードを使用することによってこのタスクを行うことができます。

手順

1. コマンド行で次のように入力します。

```
runmqsc QM1
```

2. QM1 上に TO.QM2 という名前の送信側チャネルを作成します。MQSC インターフェースで次のように入力します。

```
DEFINE CHANNEL(TO.QM2) CHLTYPE(SDR) TRPTYPE(TCP)
CONNAME('remoteHost') XMITQ(QM2)
SSLCIPH(TLS_RSA_WITH_AES_128_CBC_SHA256)
DESCR('Sender channel using TLS from QM1 to QM2')
REPLACE
```

   注: 変数 *remoteHost* は、ターゲット・キュー・マネージャーのホスト名または IP アドレスです。

   チャネルの CERTLABL 属性を指定できます。 これを指定する場合は、28 ページの『TLS を使用するキュー・マネージャーの準備』のステップ 3 で既に実行した **runmqckm** コマンドの **-label** パラメーターの値に一致させる必要があります。 証明書ラベルの詳細については、デジタル証明書ラベルの要件に関する説明を参照してください。

3. end と入力して MQSC インターフェースを終了します。
4. コマンド行で次のように入力します。

```
runmqsc QM2
```

5. QM2 上に TO.QM2 という名前の受信側チャネルを作成します。MQSC インターフェースで次のように入力します。

```
DEFINE CHANNEL(TO.QM2) CHLTYPE(RCVR) TRPTYPE(TCP)
SSLCIPH(TLS_RSA_WITH_AES_128_CBC_SHA256) SSLCAUTH(REQUIRED)
DESCR('Receiver channel using TLS from QM1 to QM2')
REPLACE
```

6. end と入力して MQSC インターフェースを終了します。

次のタスク

ソース・キュー・マネージャーが TLS を使用してターゲット・キュー・マネージャーにメッセージを送信できることを検証するには、27 ページの『ソリューションの検証』の手順に従ってください。

V 9.3.0 ストリーミング・キュー

IBM MQ のストリーミング・キュー機能を使用すると、 すべてのメッセージのコピー (元とほぼ同じもの) を 2 番目のキューに書き込むようにキューを構成することができます。

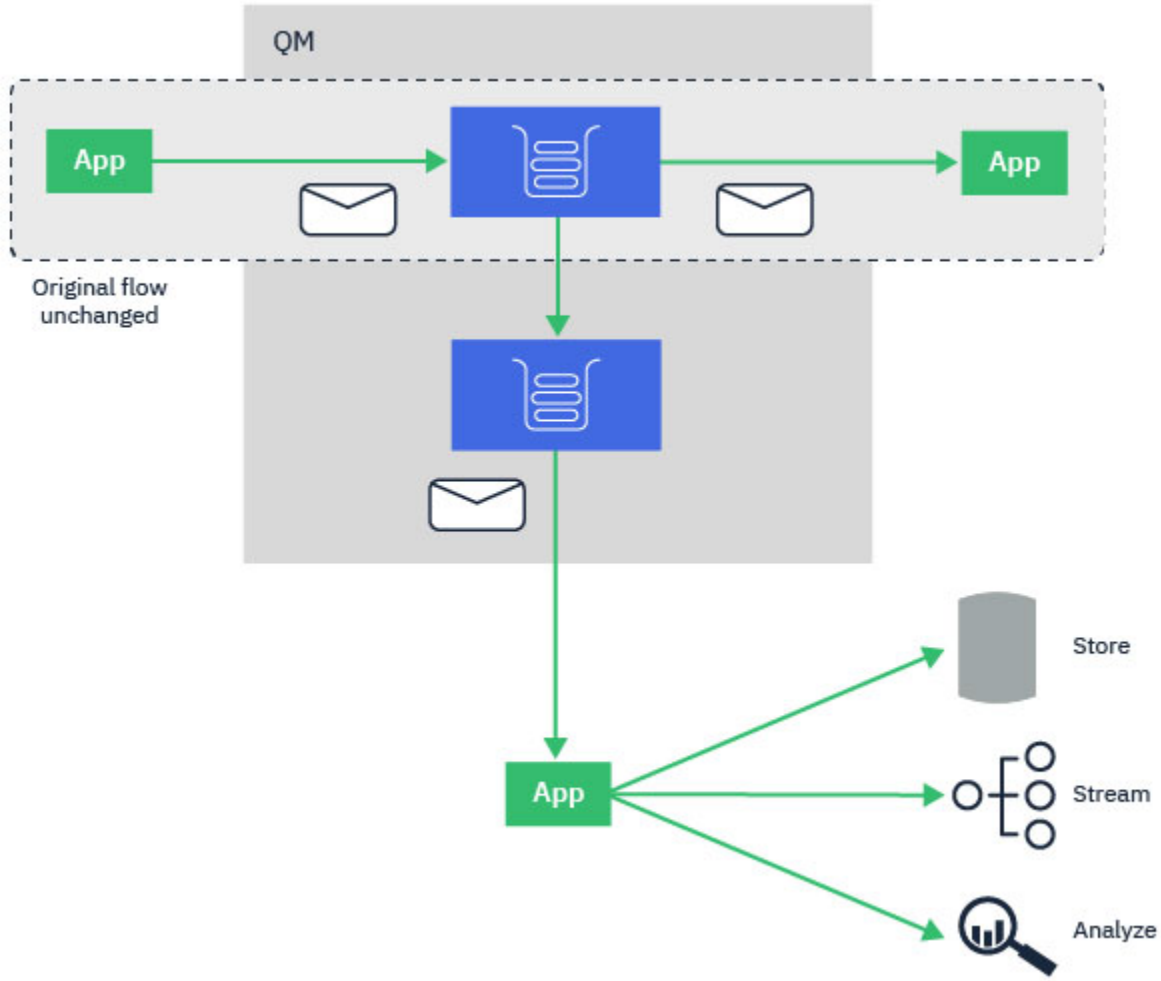
ストリーミング・キューは、メッセージのコピーを作成する必要がある特定のシナリオで役立つ場合があります。 以下に例を示します。

- IBM MQ の Kafka Connect ソース・コネクターを使用して、メッセージを Apache Kafka にストリーミングします。 詳しくは、kafka_connect_mq_source を参照してください。
- システムを通過するデータに関する分析を実行する。
- 後でリカバリーするためにメッセージを保管する。
- 開発システムとテスト・システムで使用するためにメッセージのセットを取り込む。

- システム・イベント・キューから IBM MQ イベント・メッセージをコンシュームし、他のキューまたはトピックに追加のコピーを送信する。

これらのすべてのシナリオでは、ストリーミング・キューを構成することで、ストリーミング・プロセスが元のメッセージに影響を与えないようにすることができます。 そうすれば、中核となるビジネス・アプリケーションが、ストリーミングの影響を受けることはありません。

これを図にすると、次のようになります。



関連概念

ストリーミング・キューのセキュリティー

ストリーミング・キューと AMS

V 9.3.0 ストリーミング・キュー構成

IBM MQ のストリーミング・キュー機能は、個々のキューの管理者によって構成され、 メッセージはアプリケーション自体ではなく、キュー・マネージャーによってストリームされます。

つまり、ほとんどの場合、元のキューにメッセージを書き込むアプリケーションは、ストリーミングが行われている事実をまったく認識しません。 同様に、元のキューのメッセージをコンシュームするアプリケーションも、メッセージ・ストリーミングが行われたことを認識しません。

注 : ストリーミング・キューを使用するために、IBM MQ クライアント・ライブラリーのバージョンをアップグレードする必要はありません。 また、元のメッセージは、ストリーミング・プロセスによってまったく変更されません。

ストリーミング・キューは、以下の 2 つのモードのいずれかで構成できます。

ベスト・エフォート
このモードでは、キュー・マネージャーは、ストリーム・メッセージの送達が元のメッセージの送達に影響しないことをより重要視します。

元のメッセージを配信することは可能であってもストリーム・メッセージを配信することは不可能である場合、元のメッセージが対応するキューに配信されます。 元のビジネス・アプリケーションがストリーミング・プロセスの影響を受けないことが重要である場合、そのようなアプリケーションには、このモードが最も適しています。

複製が必要
このモードでは、キュー・マネージャーは、元のメッセージとストリーム・メッセージの両方が確実にそれぞれのキューに正常に送達されるようにします。

何らかの理由 (例えば 2 番目のキューがフルであるなど) により、ストリーム・メッセージをそのキューに送達できない場合は、元のメッセージもそのキューに送達されません。 書き込み側のアプリケーションはエラー理由コードを受け取り、メッセージの書き込みを再度試行する必要があります。

メッセージ・ストリーミングを有効にするローカル・キューとモデル・キューに追加されるその他の属性については、 ストリーミング・キューの構成方法を参照してください。

ストリーム・メッセージ

ほとんどの場合、2 番目のキューに送達されるメッセージのコピーは、元のメッセージの複写です。 これには、メッセージ ID や相関 ID など、すべてのメッセージ記述子フィールドが含まれます。 ストリーム・メッセージは、元のメッセージに非常に近いコピーになるようになっているので、見つけやすく、 必要であれば、別の IBM MQ システムで再生することもできます。

一部のメッセージ記述子フィールドは、ストリーム・メッセージでは保持されません。 ストリーム・メッセージは、2 番目のキューに入れられる前に、以下の変更を加えられます。

- ストリーム・メッセージの有効期限が、元のメッセージの有効期限に関係なく、MQEI_UNLIMITED に設定されます。 **CAPEXPRY** が 2 次キューに設定されている場合は、その値を使用して、ストリーム・メッセージの有効期限時刻が設定されます。
- 元のメッセージに以下のレポート・オプションが設定されている場合、それらのオプションはストリーム・メッセージでは有効にされません。 これは、予期しないレポート・メッセージが、それらを受け取るように設計されていないアプリケーションに送達されないようにするためです。
  - アクティビティー・レポート
  - 有効期限レポート
  - 例外報告
  - 到着時の確認 (COA)
  - 配信時の確認 (COD)

ほぼ同一になるというストリーム・メッセージの性質のため、2 次キューのほとんどの属性は、ストリーム・メッセージのメッセージ記述子フィールドに影響を与えません。 例えば、2 次キューの **DEFPSIST** 属性と **DEFPRTY** 属性は、ストリーム・メッセージには影響しません。

ストリーム・メッセージには、以下の例外が適用されます。

- **CAPEXPRY** 属性

  2 次キューが **CAPEXPRY** 属性を使用して構成されている場合、この有効期限の上限は、ストリーム・メッセージの有効期限に適用されます。
- **DEFBIND** (クラスター・キューの場合)

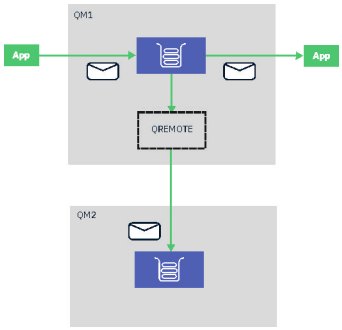
  2 次キューがクラスター・キューの場合、ストリーム・メッセージは、2 次キューの **DEFBIND** 属性に設定されたバインド・オプションを使用して書き込まれます。

V 9.3.0 リモート・キューおよび別名キューへのストリーミングです。

リモート・キューおよび別名キューにメッセージをストリームすることができます。 例えば、 Q1 は STREAMQ(MY.REMOTE.Q) を使用して構成されることがあり、MY.REMOTE.Q がリモート・キュー定義です。

リモート・キューへのストリーミングです。

ローカル・キューからリモート・キューにメッセージをストリーミングすることにより、以下の図に示すように、重複したメッセージを IBM MQ ネットワーク上の別のキュー・マネージャーのキューに送信できます。



別名キューへのストリーミングです。

別名キューにメッセージをストリーミングすることにより、重複したメッセージを別名キューのターゲットに送信できます。 別名キューのターゲットをトピックにすることもできるので、重複したメッセージをパブリッシュ/サブスクライブ・トピックに送信することが可能です。 別名トピックのサブスクライバーは全て重複したメッセージのコピーを受信します。 このようにして、元のメッセージのコピーを複数作成することができます。 ただし、パブリッシュ/サブスクライブ・メッセージの既存のルールは、重複したメッセージに適用されます。 つまり、サブスクライバーに送信されるメッセージは、元のメッセージと同じではなく、次のようなメッセージが表示されます。

- 新しいメッセージ ID を持ちます。
- サブスクリプションの構成に応じて、生成された相関 ID を持ちます。
- キュー・マネージャーが実行しているユーザーに設定されるユーザー識別子のフィールドは、メッセージを書き込んだユーザーではありません。
- 書き込み側アプリケーションの名前ではなく、PutApplName はキュー・マネージャーの名前を示します。

注 :

1. リモート・キューまたは別名キュー自体で **STREAMQ** 属性を構成することはできません。 それらからではなく、 メッセージをそれらにストリームしかできません。
2. メッセージがキューの別名にストリーミングされている場合、キューの別名のターゲットはその **STREAMQ** 属性セットを持つことができません。

V 9.3.0 ストリーミング・キューの制約事項

IBM MQ でストリーミング・キューを使用するときにサポートされない構成があります。それらをここに記載します。

以下のリストは、サポートされない構成を示しています。

- 相互にストリーミングするキューのチェーンを定義する。例: Q1->Q2, Q2->Q3, Q3->Q4
- ストリーミング・キューのループを定義する。例: Q1->Q2, Q2->Q1
- STREAMQ が定義されている提供された宛先を使用してサブスクリプションを定義する
- USAGE(XMITQ) を指定して構成されたキューに対して STREAMQ を定義する

注 : STREAMQ はリモート・キューにすることができますが、リモート・キュー定義で STREAMQ 属性を構成することはできません。

- 動的キューの STREAMQ 属性を変更する
- STREAMQ を SYSTEM.* で始まる任意の値に設定する。 ただし、 SYSTEM.DEFAULT.LOCAL.QUEUE
- SYSTEM.* という名前の任意のキューに STREAMQ を定義する。 ただし、以下の例外があります。
  - SYSTEM.DEFAULT.LOCAL.QUEUE
  - SYSTEM.ADMIN.ACCOUNTING.QUEUE
  - SYSTEM.ADMIN.ACTIVITY.QUEUE
  - SYSTEM.ADMIN.CHANNEL.EVENT
  - SYSTEM.ADMIN.COMMAND.EVENT
  - SYSTEM.ADMIN.CONFIG.EVENT
  - SYSTEM.ADMIN.LOGGER.EVENT
  - SYSTEM.ADMIN.PERFM.EVENT
  - SYSTEM.ADMIN.PUBSUB.EVENT
  - SYSTEM.ADMIN.QMGR.EVENT
  - SYSTEM.ADMIN.STATISTICS.QUEUE
  - SYSTEM.DEFAULT.MODEL.QUEUE
  - SYSTEM.JMS.TEMPQ.MODEL
- STREAMQ をモデル・キューの名前に設定する
- V 9.3.1 z/OS IBM MQ 9.3.0 で共用キューに STREAMQ を定義する (APAR PH49686 が適用されていない場合)。 この制限は、APAR PH49686 が適用されたときに IBM MQ 9.3.0 で、または IBM MQ 9.3.1 から解除されました。
- V 9.3.1 z/OS APAR PH49686 が適用されていない場合に、STREAMQ を IBM MQ 9.3.0 の共用キューの名前に設定する。 この制限は、APAR PH49686 が適用されたときに IBM MQ 9.3.0 で、または IBM MQ 9.3.1 から解除されました。

V 9.3.0 ストリーム・キューおよびトランザクション

ストリーミング・キュー機能を使用すると、1 つのキューに書き込まれたメッセージを、2 番目のキューに複写できます。 ほとんどの場合、2 つのメッセージは、1 つの作業単位の下でそれぞれのキューに書き込まれます。

元のメッセージが MQPMO_SYNCPOINT を使用して書き込まれた場合、複写メッセージは、元の書き込みのために開始されたのと同じ作業単位の下でストリーム・キューに書き込まれます。

元のメッセージが MQPMO_NO_SYNCPOINT を使用して書き込まれた場合、元の書き込みでは作業単位が要求されていなくとも作業単位が開始されます。 この動作には次のような 2 つの理由があります。

1. 元のメッセージを送達できなかった場合には、複写メッセージは送達されないようにするため。 ストリーミング・キュー機能によってストリーム・キューにメッセージが送達されるのは、元のメッセージも送達された場合に限られます。
2. 1 つの作業単位内で両方の書き込みを行うことで、パフォーマンスが向上する可能性があるため。

メッセージが 1 つの作業単位内で送達されないのは、元の MQPUT が MQPMO_NO_SYNCPOINT を使用した非永続のものであり、キューの **STRMQOS** 属性が BESTEF (ベスト・エフォート) に設定されている場合だけです。

注 :

1. ストリーム・キューへの追加の書き込みは、MAXUMSGS の制限値に対してカウントされません。
2. STRMQOS(BESTEF) を指定して構成されたキューの場合、複写メッセージを送達できないときに、作業単位はロールバックされません。

V 9.3.0 クラスター・キューとの間のストリーミング

ローカル・キューからクラスター・キューにメッセージをストリームすることも、クラスター・キュー・インスタンスからローカル・キューにメッセージをストリームすることも可能です。

クラスター・キューへのストリーミング

これは、元のメッセージが送達されるローカル・キューがあるときに、すべてのメッセージのコピーを、クラスター・キューの 1 つ以上のインスタンスにストリームする場合に便利です。 そうする目的としては、複写メッセージの処理のワークロード・バランスを取るため、または単にクラスター内にある別のキューに複写メッセージをストリームするためなどがあります。

クラスター・キューにストリーミングされるメッセージは、クラスター・ワークロード・バランシング・アルゴリズムを使用して分散させられます。 クラスター・キュー・インスタンスは、クラスター・キューの DEFBIND 属性に基づいて選択されます。

例えば、クラスター・キューが DEFBIND(OPEN) を指定して構成されている場合は、元のキューがオープンされるときに、クラスター・キューのインスタンスが選択されます。 元のキューがアプリケーションによって再オープンされるまで、複写メッセージはすべてこの同じクラスター・キュー・インスタンスに送達されます。

クラスター・キューが DEFBIND(NOTFIXED) を指定して構成されている場合は、MQPUT 操作のたびにクラスター・キューのインスタンスが選択されます。

注 : すべてのクラスター・キュー・インスタンスは、DEFBIND 属性に同じ値を指定して構成する必要があります。

クラスター・キューからのストリーミング

これは、既にクラスター・キューのいくつかのインスタンスにメッセージを送信しているときに、各メッセージのコピーを、クラスター・キュー・インスタンスと同じキュー・マネージャー上のストリーミング・キューに送達する場合に便利です。

元のメッセージがクラスター・キュー・インスタンスの 1 つに送達されると、クラスター受信側チャネルによって、複写メッセージがストリーム・キューに送達されます。

V 9.3.1 Multi ストリーミング・キューを使用したメッセージの履歴の保管

ストリーミング・キューを使用して、メッセージの履歴を一定期間保持することができます。これを行うには、メッセージのストリーミング先のキューで CAPEXPRY 属性を構成します。

概要

メッセージがあるキューから別のキューにストリーミングされると、メッセージに設定された有効期限値は、重複コピーの MQEI_UNLIMITED の値にリセットされます。 デフォルトでは、メッセージをコンシュームするアプリケーションがない場合、ストリーミング先のキューにメッセージが安定して蓄積されます。

このシナリオでは、メッセージにアクセスできるように、限られた期間だけメッセージのコピーを保持する必要があります。 例えば、元のメッセージがコンシューム側アプリケーションによって誤って削除された場合などです。

すべてのメッセージのコピーを無期限に保持することはできません。また、ストリーミング先のキューが満杯にならないようにするには、以下の 2 つのオプションがあります。

- アプリケーションを実行して、メッセージを頻繁に削除する
- 有効期限付きでメッセージを構成します。これにより、メッセージは一定期間後に IBM MQ によって削除されます。

2 番目のオプションは、キューがいっぱいにならないようにするためだけに、アプリケーションを実行して保守する必要がないため、はるかに便利です。

CAPEXPRY の構成

「有効期限を短くする」トピックでは、キューで CAPEXPRY を構成する方法について説明しています。 このシナリオでは、メッセージのストリーミング先のキューに CAPEXPRY 属性を設定する必要があります。

注 : メッセージのストリーミング元となるキューの CAPEXPRY 属性の値を変更する必要はありません。

以下の考慮事項を考慮して、重複メッセージの適切な有効期限時刻を選択します。

1. メッセージへのアクセスが必要になる可能性がある期間
2. 元のキューに書き込まれるメッセージの比率に基づいて、キューの MAXDEPTH 属性が何である必要があるか
3. 重複メッセージを保管するために必要なストレージの量。

   これには、キュー・マネージャーのファイル・システム・サイズ、およびキューの MAXFSIZE 属性を考慮する必要がある場合があります。

重複メッセージへのアクセス (必要な場合)

重複メッセージの一部またはすべてにアクセスしてリカバリーする必要がある問題が発生した場合は、**dmpmqmsg** コマンドを使用して、それらのメッセージへのアクセスに役立ててください。

dmpmqmsg には、以下のオプションがあります。

- キューからメッセージを読み取り、後でアクセスするためにファイルにコピーを書き込む
- ファイルからメッセージを読み取り、アプリケーションがコンシュームできるようにキューに書き戻す

dmpmqmsg がメッセージ・ヘッダーをファイルに書き込んだ後で、メッセージ・ヘッダーを編集することができます。 例えば、ファイル内のすべてのメッセージの EXP 値を *-1* に変更することにより、**dmqmqmsg** がメッセージを処理のためにキューに戻す前に、メッセージの有効期限を MQEI_UNLIMITED にリセットすることができます。

パフォーマンス上の考慮事項

重複メッセージを別のキューにストリーミングして、不要になったメッセージを有効期限切れにするには、わずかなコストがかかります。 ただし、このコストは、手動でコピーを 2 番目のキューに書き込み、一定期間後にアプリケーションがそれらを破壊的に除去するよりもはるかに小さくなります。 Streaming Queues Performance Report は、このシナリオのパフォーマンスに関する詳細情報を提供します。

関連概念

ストリーミング・キューのセキュリティー

ストリーミング・キューと AMS

パブリッシュ/サブスクライブのシナリオ

パブリッシュ/サブスクライブ・クラスターとパブリッシュ/サブスクライブ階層の使用法を例示する 2 つのシナリオの集合。

使用可能なパブリッシュ/サブスクライブ・シナリオについては、以下のサブトピックに説明があります。

シナリオ: パブリッシュ/サブスクライブ・クラスターの作成

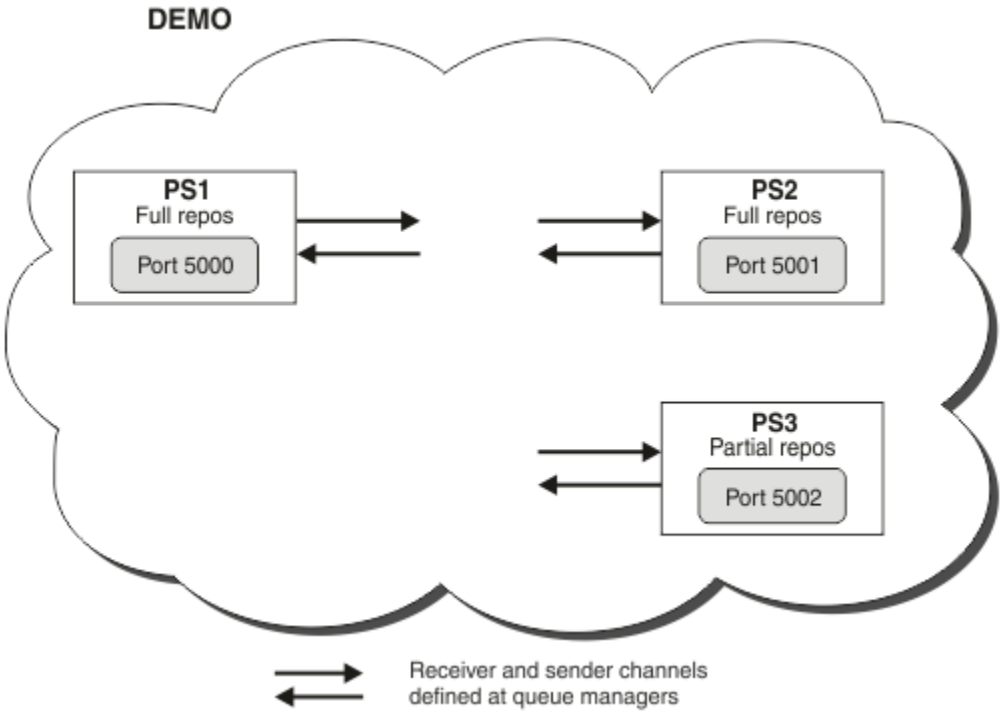
このシナリオでは、3 つのキュー・マネージャーからなる単純なクラスターを作成します。それを構成して、1 つのキュー・マネージャーで作成されたサブスクリプションが、別のキュー・マネージャーに接続されているアプリケーションによってパブリッシュされるメッセージを受信できるようにします。

始める前に

このシナリオの開始点は、既存の IBM MQ インストール済み環境です。 IBM MQ のインストールについては、Windows での IBM MQ サーバーのインストールの手順に従ってください。

このタスクについて

このシナリオの手順を完了することにより、まず次のクラスターを作成します。



このクラスターは 3 つのキュー・マネージャーからなり、そのうち 2 つはフル・リポジトリー・キュー・マネージャーとして定義されます。

次に、キュー・マネージャー PS3 上でクラスター・トピックを定義します。 クラスター・トピックを作成することにより、クラスターをパブリッシュ/サブスクライブ・クラスターにしたことになります。 パブリッシュ/サブスクライブ・クラスターをテストするには、任意のキュー・マネージャー上のトピックにサブスクライブした後、別のキュー・マネージャーからこのトピックにメッセージをパブリッシュし、サブスクリプション側がメッセージを受け取ることを確認します。

関連タスク

パブリッシュ/サブスクライブ・クラスターの設計

キュー・マネージャー・クラスターの構成

キュー・マネージャーの作成および開始

PS1、PS2 および PS3 という名前の 3 つのキュー・マネージャーを作成して開始します。

手順

1. キュー・マネージャー PS1 を作成して開始します。
   a) キュー・マネージャーを作成します。

      コマンド・ラインで、次のコマンドを入力します。

      ```
      crtmqm PS1
      ```

   b) キュー・マネージャーを始動します。

      コマンド・ラインで、次のコマンドを入力します。

      ```
      strmqm PS1
      ```

2. ステップ 1 を繰り返してキュー・マネージャー PS2 を作成および開始します。
3. ステップ 1 を繰り返してキュー・マネージャー PS3 を作成および開始します。

次のタスク
これで、最初のキュー・マネージャーを構成する準備ができました。

最初のキュー・マネージャーの構成
MQSC インターフェースを使用することで、PS1 用のリスナーと受信側チャネルを定義し、このキュー・マネージャーをクラスターのフル・リポジトリーとして設定し、PS1 から PS2 への送信側チャネルを定義します (これにより 2 つのフル・リポジトリーは情報を交換できます)。

始める前に
このタスクでは、37 ページの『キュー・マネージャーの作成および開始』のすべての手順が既に完了していることを前提とします。

手順
1. PS1 のリスナーを定義して開始します。
   a) MQSC インターフェースを起動します。

      コマンド・ラインで、次のコマンドを入力します。

      ```
      runmqsc PS1
      ```

   b) リスナーを定義します。

      次の MQSC コマンドを入力します。

      ```
      DEFINE LISTENER(PS1_LS) TRPTYPE(TCP) CONTROL(QMGR) PORT(5000)
      ```

   c) リスナーを始動します。

      次の MQSC コマンドを入力します。

      ```
      START LISTENER(PS1_LS)
      ```

2. このキュー・マネージャーをクラスターのフル・リポジトリーとして設定します。

   次の MQSC コマンドを入力します。

   ```
   ALTER QMGR REPOS(DEMO)
   ```

3. PS1 の受信側チャネルを定義して、クラスター内の他のキュー・マネージャーとの通信を可能にします。

   次の MQSC コマンドを入力します。

   ```
   DEFINE CHANNEL(DEMO.PS1) CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CONNAME('$HOSTNAME(5000)')
   CLUSTER(DEMO)
       DESCR('TCP Cluster-receiver channel for queue manager PS1')
   ```

4. PS1 から PS2 への送信側チャネルを定義することで、2 つのフル・リポジトリーが情報を交換できるようにします。

   次の MQSC コマンドを入力します。

   ```
   DEFINE CHANNEL(DEMO.PS2) CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME('$HOSTNAME(5001)')
   ```

```
CLUSTER(DEMO)
    DESCR('TCP Cluster-sender channel from PS1 to queue manager PS2')
```

次のタスク

これで、2 番目のキュー・マネージャーを構成する準備ができました。

2 番目のキュー・マネージャーの構成

MQSC インターフェースを使用することで、PS2 用のリスナーと受信側チャネルを定義し、このキュー・マネージャーをクラスターのフル・リポジトリーとして設定し、PS2 から PS1 への送信側チャネルを定義します (これにより 2 つのフル・リポジトリーは情報を交換できます)。

始める前に

このタスクでは、38 ページの『最初のキュー・マネージャーの構成』のすべての手順が既に完了していることを前提とします。

手順

1. PS2 のリスナーを定義して開始します。

   a) MQSC インターフェースを起動します。

   コマンド・ラインで、次のコマンドを入力します。

   ```
   runmqsc PS2
   ```

   b) リスナーを定義します。

   次の MQSC コマンドを入力します。

   ```
   DEFINE LISTENER(PS2_LS) TRPTYPE(TCP) CONTROL(QMGR) PORT(5001)
   ```

   c) リスナーを始動します。

   次の MQSC コマンドを入力します。

   ```
   START LISTENER(PS2_LS)
   ```

2. このキュー・マネージャーをクラスターのフル・リポジトリーとして設定します。

   次の MQSC コマンドを入力します。

   ```
   ALTER QMGR REPOS(DEMO)
   ```

3. PS2 の受信側チャネルを定義して、クラスター内の他のキュー・マネージャーとの通信を可能にします。

   次の MQSC コマンドを入力します。

   ```
   DEFINE CHANNEL(DEMO.PS2) CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CONNAME('$HOSTNAME(5001)')
   CLUSTER(DEMO)
       DESCR('TCP Cluster-receiver channel for queue manager PS2')
   ```

4. PS2 から PS1 への送信側チャネルを定義することで、2 つのフル・リポジトリーが情報を交換できるようにします。

   次の MQSC コマンドを入力します。

   ```
   DEFINE CHANNEL(DEMO.PS1) CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME('$HOSTNAME(5000)')
   ```

```
CLUSTER(DEMO)
    DESCR('TCP Cluster-sender channel from PS2 to PS1')
```

次のタスク

これで、3 番目のキュー・マネージャーを構成する準備ができました。

3 番目のキュー・マネージャーの構成

MQSC インターフェースを使用して、PS3 用のリスナーと受信側チャネルを定義します。 PS3 からいずれかのフル・リポジトリー・キュー・マネージャーへの送信側チャネルを定義することにより、PS3 をクラスターに参加させます。

始める前に

このタスクでは、39 ページの『2 番目のキュー・マネージャーの構成』のすべての手順が既に完了していることを前提とします。

手順

1. PS3 のリスナーを定義して開始します。
   a) MQSC インターフェースを起動します。

      コマンド・ラインで、次のコマンドを入力します。

      ```
      runmqsc PS3
      ```

   b) リスナーを定義します。

      次の MQSC コマンドを入力します。

      ```
      DEFINE LISTENER(PS3_LS) TRPTYPE(TCP) CONTROL(QMGR) PORT(5002)
      ```

   c) リスナーを始動します。

      次の MQSC コマンドを入力します。

      ```
      START LISTENER(PS3_LS)
      ```

2. PS3 の受信側チャネルを定義して、クラスター内の他のキュー・マネージャーとの通信を可能にします。

   次の MQSC コマンドを入力します。

   ```
   DEFINE CHANNEL(DEMO.PS3) CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CONNAME('$HOSTNAME(5002)')
   CLUSTER(DEMO)
       DESCR('TCP Cluster-receiver channel for queue manager PS3')
   ```

3. PS3 からいずれかのフル・リポジトリー・キュー・マネージャー (例えば PS1) への送信側チャネルを定義します。 これにより PS3 がクラスターに追加されます。

   次の MQSC コマンドを入力します。

   ```
   DEFINE CHANNEL(DEMO.PS1) CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME('$HOSTNAME(5000)')
   CLUSTER(DEMO)
       DESCR('TCP Cluster-sender channel from PS3 to PS1')
   ```

4. PS3 が正常にクラスターに追加されたことを検証します。

   次の MQSC コマンドを入力します。

```
DISPLAY CLUSQMGR(*) QMTYPE
```

このコマンドにより、QM1、QM2 および QM3 のそれぞれについて 1 つずつ項目が返されます。 QM1 および QM2 の **QMTYPE** が REPOS になっており、さらに QM3 の **QMTYPE** が NORMAL になっていることを確認してください。

次のタスク

これで、クラスター・トピックを定義する準備ができました。

クラスター・トピックの定義

パブリッシュ/サブスクライブ・アプリケーションは、管理対象トピック・オブジェクトを定義しなくても、任意のトピック・ストリングに向けてパブリッシュすることができます。 ただし、パブリッシュを行うアプリケーションが、サブスクリプションの作成場所であるキュー・マネージャーとは異なるクラスター・キュー・マネージャーに接続される場合には、管理対象トピック・オブジェクトを定義してそれをクラスターに追加する必要があります。 あるトピックをクラスター・トピックにするには、その定義の中でクラスターの名前を指定します。

始める前に

このタスクでは、40 ページの『3 番目のキュー・マネージャーの構成』のすべての手順が既に完了していることを前提とします。

このタスクについて

管理対象トピック・オブジェクトは、トピック・ストリングによってクラスター化されているトピック・ツリー内の地点を識別します。 パブリッシュやサブスクライブを行うアプリケーションは、この地点またはその下にある任意のトピック・ストリングを使用することが可能で、それらのメッセージはキュー・マネージャー間で自動的に伝送されます。

クラスター・トピックを定義するときには、そのルーティング・モデルも選択します。 クラスターにおけるパブリケーション・ルーティングについて詳しくは、パブリッシュ/サブスクライブ・クラスターの設計を参照してください。

このシナリオでは、デフォルト・ルーティング *DIRECT* を使用します。 つまり、パブリッシュを行うキュー・マネージャーからサブスクライブを行うキュー・マネージャーにメッセージが直接送られます。

手順

1. PS3 にクラスター・トピック SCORES を定義します。

   このトピックをクラスター・トピックにするには、クラスターの名前を指定し、このトピックのパブリケーションおよびサブスクリプションに使用するクラスター経路指定 (**CLROUTE**)を設定します。

   a) MQSC インターフェースを起動します。

   コマンド・ラインで、次のコマンドを入力します。

   ```
   runmqsc PS3
   ```

   b) クラスター・トピック SCORES を定義します。

   次の MQSC コマンドを入力します。

   ```
   DEFINE TOPIC(SCORES) TOPICSTR('/Sport/Scores') CLUSTER(DEMO) CLROUTE(DIRECT)
   ```

   c) end と入力して、PS3 用の MQSC インターフェースを終了します。

2. PS1 でトピック定義を検証します。

   a) PS1 用の MQSC インターフェースを起動します。

コマンド・ラインで、次のコマンドを入力します。

```
runmqsc PS1
```

b) クラスター・トピック SCORES のクラスター状態を表示します。

次の MQSC コマンドを入力します。

```
DISPLAY TCLUSTER(SCORES) CLSTATE
```

クラスター・トピック SCORES の **CLSTATE** が ACTIVE と表示されます。

次のタスク

このタスクの詳しい説明については、パブリッシュ/サブスクライブ・クラスターの構成を参照してください。

これで、ソリューションを検証する準備ができました。42 ページの『パブリッシュ/サブスクライブ・クラスターのテスト』 を参照してください。

パブリッシュ/サブスクライブ・クラスターのテスト

クラスター内の複数の異なるキュー・マネージャーからトピック・ストリングへのパブリッシュおよびサブスクライブを行うことで、パブリッシュ/サブスクライブ・クラスターをテストします。

始める前に

このタスクでは、41 ページの『クラスター・トピックの定義』のすべての手順が既に完了していることを前提とします。

このタスクについて

コマンド行、および IBM MQ に組み込まれている amqspub サンプル・アプリケーションと amqssub サンプル・アプリケーションを使用して、1 つのキュー・マネージャーからトピックをパブリッシュし、他のキュー・マネージャーでそのトピックにサブスクライブすることができます。 トピックにパブリッシュされたメッセージは、サブスクライブ側のキュー・マネージャーによって受信されます。

手順

1. コマンド・ラインで、次のコマンドを入力します。

```
amqspub /Sport/Scores/Football PS1
```

2. これと並行して、それぞれ別個のコマンド・ラインで以下のコマンドを入力します。

```
amqssub /Sport/Scores/Football PS2
```

```
amqssub /Sport/Scores/Football PS3
```

3. 最初のコマンド・ラインでメッセージを入力します。

このメッセージが両方のサブスクライブ側コマンド・ラインに表示されます。

注 : 10 秒間にわたってパブリケーションが受信されない場合、amqssub アプリケーションがタイムアウトになります。

タスクの結果

パブリッシュ/サブスクライブ・クラスターのセットアップが完了しました。

次のタスク

トピック・ツリー内のさまざまなブランチでさまざまなトピック・オブジェクトを定義し、さまざまなルーティング・モデルを試してみてください。

パブリッシュ/サブスクライブ階層のシナリオ

パブリッシュ/サブスクライブ階層の使用法を例示する 3 つのシナリオ。 これら 3 つのシナリオは、それぞれ、同じ単純なパブリッシュ/サブスクライブ・トポロジーをセットアップします。 各シナリオで、キュー・マネージャーは階層内の近隣の各キュー・マネージャーに接続するために異なる方法を使用します。

使用可能なパブリッシュ/サブスクライブ階層シナリオについては、以下のサブトピックに説明があります。

関連概念

パブリッシュ/サブスクライブの階層

パブリッシュ/サブスクライブ階層のシナリオ 1: キュー・マネージャー名の別名での Point-to-Point チャネルの使用

これは、キュー・マネージャー間の接続を確立するために異なる方法でパブリッシュ/サブスクライブ階層をセットアップする 3 セットのシナリオのうち最初のものです。 このシナリオでは、キュー・マネージャー名の別名で Point-to-Point チャネルを使用するパブリッシュ/サブスクライブ階層をセットアップします。

このタスクについて

このシナリオ・セットはすべて、 QM1 という親キュー・マネージャーと、 QM2 と QM3 という 2 つの子キュー・マネージャーを使用します。

プロセスを簡単にたどれるように、シナリオ 1 は小さいセクションに分割されています。

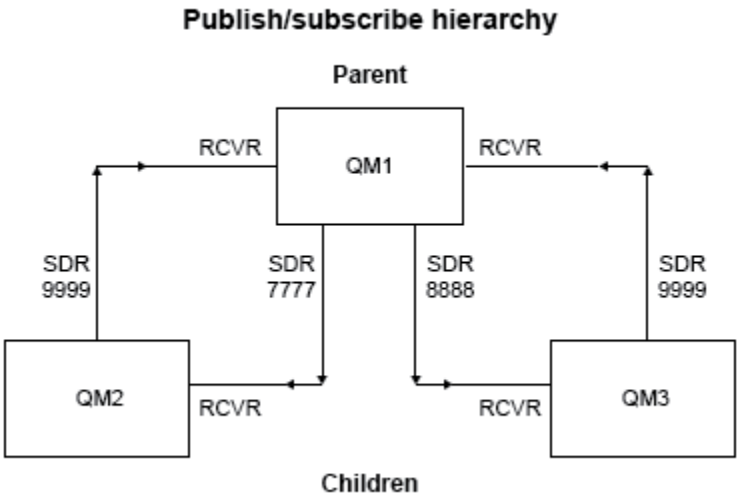


図 *3.* 標準的なパブリッシャー/サブスクライブ階層内のキュー・マネージャー間の関係を示すトポロジー図。

手順

1. キュー・マネージャーを作成します。
   a) 以下のコマンドを使用して、3 つのキュー・マネージャー QM1、QM2、および QM3 を作成して開始します。

```
crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM1
strmqm QM1

crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM2
strmqm QM2

crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM3
strmqm QM3
```

b) 3 つのすべてのキュー・マネージャーで以下のコマンドを使用して、キュー・マネージャー・パブリッシュ/サブスクライブ・モードを有効にします。

```
ALTER QMGR PSMODE(ENABLED)
```

2. 親キュー・マネージャーと同じ名前のキュー・マネージャー別名を使用して、キュー・マネージャー間の Point-to-Point チャネル接続を確立します。

a) QM2 で、QM1 に対する伝送キューおよびキュー・マネージャー別名を定義します。 QM1 に対する送信側チャネル、および QM1 で QM2 用に作成された送信側チャネルに対応する受信側チャネルを定義します。

```
DEFINE QLOCAL(QM1.XMITQ) USAGE(XMITQ)

DEFINE QREMOTE (QM1) RNAME('') RQMNAME(QM1) XMITQ(QM1.XMITQ)

DEFINE CHANNEL('QM2.TO.QM1') CHLTYPE(SDR) CONNAME('localhost(9999)') XMITQ(QM1.XMITQ)
TRPTYPE(TCP)

DEFINE CHANNEL('QM1.TO.QM2') CHLTYPE(RCVR) TRPTYPE(TCP)
```

b) QM3 で、QM1 に対する伝送キューおよびキュー・マネージャー別名を定義します。 QM1 に対する送信側チャネル、および QM1 で QM3 用に作成された送信側チャネルに対応する受信側チャネルを定義します。

```
DEFINE QLOCAL(QM1.XMITQ) USAGE(XMITQ)

DEFINE QREMOTE (QM1) RNAME('') RQMNAME(QM1) XMITQ(QM1.XMITQ)

DEFINE CHANNEL('QM3.TO.QM1') CHLTYPE(SDR) CONNAME('localhost(9999)') XMITQ(QM1.XMITQ)
TRPTYPE(TCP)

DEFINE CHANNEL('QM1.TO.QM3') CHLTYPE(RCVR) TRPTYPE(TCP)
```

c) QM1 で、QM2 および QM3 に対する伝送キューおよびキュー・マネージャー別名を定義します。 QM2 および QM3 に対する送信側チャネル、ならびに QM2 および QM3 で QM1 用に作成された送信側チャネルに対応する受信側チャネルを定義します。

```
DEFINE QLOCAL(QM2.XMITQ) USAGE(XMITQ)

DEFINE QREMOTE (QM2) RNAME('') RQMNAME(QM2) XMITQ(QM2.XMITQ)

DEFINE CHANNEL('QM1.TO.QM2') CHLTYPE(SDR) CONNAME('localhost(7777)') XMITQ(QM2.XMITQ)
TRPTYPE(TCP)

DEFINE CHANNEL('QM2.TO.QM1') CHLTYPE(RCVR) TRPTYPE(TCP)

DEFINE QLOCAL(QM3.XMITQ) USAGE(XMITQ)

DEFINE QREMOTE (QM3) RNAME('') RQMNAME(QM3) XMITQ(QM3.XMITQ)

DEFINE CHANNEL('QM1.TO.QM3') CHLTYPE(SDR) CONNAME('localhost(8888)') XMITQ(QM3.XMITQ)
TRPTYPE(TCP)

DEFINE CHANNEL('QM3.TO.QM1') CHLTYPE(RCVR) TRPTYPE(TCP)
```

d) キュー・マネージャーで該当のリスナーを開始します。

```
runmqlsr -m QM1 -t TCP -p 9999 &
runmqlsr -m QM2 -t TCP -p 7777 &
runmqlsr -m QM3 -t TCP -p 8888 &
```

e) 以下のチャネルを開始します。

i) On QM1:

```
START CHANNEL('QM1.TO.QM2')

START CHANNEL('QM1.TO.QM3')
```

ii) On QM2:

```
START CHANNEL('QM2.TO.QM1')
```

iii) On QM3:

```
START CHANNEL('QM3.TO.QM1')
```

f) すべてのチャネルが開始されたことを確認します。

```
DISPLAY CHSTATUS('QM1.TO.QM2')

DISPLAY CHSTATUS('QM1.TO.QM3')

DISPLAY CHSTATUS('QM2.TO.QM1')

DISPLAY CHSTATUS('QM3.TO.QM1')
```

g)

3. キュー・マネージャーを接続し、トピックを定義します。
   子キュー・マネージャー QM2 と QM3 を親キュー・マネージャー QM1 に接続します。

   a) QM2 および QM3 で、親キュー・マネージャーを QM1 に設定します。

```
ALTER QMGR PARENT (QM1)
```

   b) すべてのキュー・マネージャーで次のコマンドを実行して、子キュー・マネージャーが親キュー・マネージャーに接続していることを確認します。

```
DISPLAY PUBSUB TYPE(ALL)
```

   コマンド出力が表示されます。 例えば、以下は、主要な詳細が強調表示された QM1 の出力です。

```
DISPLAY PUBSUB ALL
1 : DISPLAY PUBSUB ALL
AMQ8723: Display pub/sub status details.
QMNAME(QM1)                TYPE(LOCAL)
STATUS(ACTIVE)              SUBCOUNT(6)
TPCOUNT(9)
AMQ8723: Display pub/sub status details.
QMNAME(QM2) TYPE(CHILD)
STATUS(ACTIVE) SUBCOUNT(NONE)
TPCOUNT(NONE)
AMQ8723: Display pub/sub status details.
QMNAME(QM3) TYPE(CHILD)
STATUS(ACTIVE) SUBCOUNT(NONE)
TPCOUNT(NONE)
```

4. amqspub.exe と amqssub.exe というアプリケーションを使用して、トピックを公開し、購読します。

   a) 1 番目のコマンド・ウィンドウで次のコマンドを実行します。

```
amqspub Sport/Soccer QM2
```

b) 2 番目のコマンド・ウィンドウで次のコマンドを実行します。

```
amqssub Sport/Soccer QM1
```

c) 3 番目のコマンド・ウィンドウで次のコマンドを実行します。

```
amqssub Sport/Soccer QM3
```

タスクの結果

2 番目と 3 番目のコマンド・ウィンドウの amqssub.exe アプリケーションは、最初のコマンド・ウィンドウで発行されたメッセージを受け取ります。

パブリッシュ/サブスクライブ階層のシナリオ 2: 伝送キューとリモート・キュー・マネージャーに同じ名前を指定した Point-to-Point チャネルの使用

これは、キュー・マネージャー間の接続を確立するために異なる方法でパブリッシュ/サブスクライブ階層をセットアップする 3 セットのシナリオのうち 2 番目のものです。 このシナリオでは、リモート・キュー・マネージャーと同じ伝送キュー名で Point-to-Point チャネルを使用するパブリッシュ/サブスクライブ階層をセットアップします。

このタスクについて

このシナリオ・セットはすべて、 QM1 という親キュー・マネージャーと、 QM2 と QM3 という 2 つの子キュー・マネージャーを使用します。

このシナリオでは、43 ページの『パブリッシュ/サブスクライブ階層のシナリオ 1: キュー・マネージャー名の別名での Point-to-Point チャネルの使用』のステップ 1、3、および 4 を再利用します。

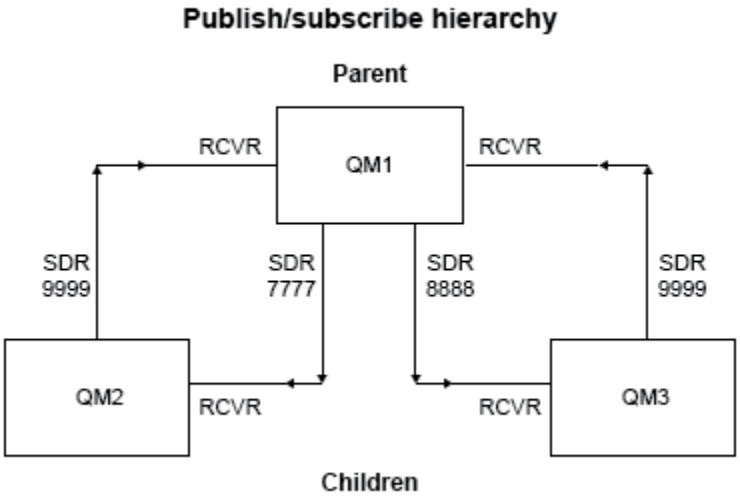


図 *4.* 標準的なパブリッシャー/サブスクライブ階層内のキュー・マネージャー間の関係を示すトポロジー図。

手順

1. キュー・マネージャーを作成します。

a) 以下のコマンドを使用して、3 つのキュー・マネージャー QM1、QM2、および QM3 を作成して開始します。

```
crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM1
strmqm QM1

crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM2
strmqm QM2

crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM3
strmqm QM3
```

b) 3 つのすべてのキュー・マネージャーで以下のコマンドを使用して、キュー・マネージャー・パブリッシュ/サブスクライブ・モードを有効にします。

```
ALTER QMGR PSMODE(ENABLED)
```

2. 親キュー・マネージャーと同じ名前の伝送キューを使用して、キュー・マネージャー間の Point-to-Point チャネル接続を確立します。

a) QM2 で、QM1 に対する伝送キューを定義します。 QM1 に対する送信側チャネル、および QM1 で QM2 用に作成された送信側チャネルに対応する受信側チャネルを定義します。

```
DEFINE QLOCAL(QM1) USAGE(XMITQ)

DEFINE CHANNEL('QM2.TO.QM1') CHLTYPE(SDR) CONNAME('localhost(9999)') XMITQ(QM1)
TRPTYPE(TCP)

DEFINE CHANNEL('QM1.TO.QM2') CHLTYPE(RCVR) TRPTYPE(TCP)
```

b) QM3 で、QM1 に対する伝送キューを定義します。 QM1 に対する送信側チャネル、および QM1 で QM3 用に作成された送信側チャネルに対応する受信側チャネルを定義します。

```
DEFINE QLOCAL(QM1) USAGE(XMITQ)

DEFINE CHANNEL('QM3.TO.QM1') CHLTYPE(SDR) CONNAME('localhost(9999)') XMITQ(QM1)
TRPTYPE(TCP)

DEFINE CHANNEL('QM1.TO.QM3') CHLTYPE(RCVR) TRPTYPE(TCP)
```

c) QM1 で、QM2 および QM3 に対する伝送キューを定義します。 QM2 および QM3 に対する送信側チャネル、ならびに QM2 および QM3 で QM1 用に作成された送信側チャネルに対応する受信側チャネルを定義します。

```
DEFINE QLOCAL(QM2) USAGE(XMITQ)

DEFINE CHANNEL('QM1.TO.QM2') CHLTYPE(SDR) CONNAME('localhost(7777)') XMITQ(QM2)
TRPTYPE(TCP)

DEFINE CHANNEL('QM2.TO.QM1') CHLTYPE(RCVR) TRPTYPE(TCP)

DEFINE QLOCAL(QM3) USAGE(XMITQ)

DEFINE CHANNEL('QM1.TO.QM3') CHLTYPE(SDR) CONNAME('localhost(8888)') XMITQ(QM3)
TRPTYPE(TCP)

DEFINE CHANNEL('QM3.TO.QM1') CHLTYPE(RCVR) TRPTYPE(TCP)
```

d) キュー・マネージャーで該当のリスナーを開始します。

```
runmqlsr -m QM1 -t TCP -p 9999 &
runmqlsr -m QM2 -t TCP -p 7777 &
runmqlsr -m QM3 -t TCP -p 8888 &
```

e) 以下のチャネルを開始します。

i) On QM1:

```
START CHANNEL('QM1.TO.QM2')
START CHANNEL('QM1.TO.QM3')
```

ii) On QM2:

```
START CHANNEL('QM2.TO.QM1')
```

iii) On QM3:

```
START CHANNEL('QM3.TO.QM1')
```

f) すべてのチャネルが開始されたことを確認します。

```
DISPLAY CHSTATUS('QM1.TO.QM2')
DISPLAY CHSTATUS('QM1.TO.QM3')
DISPLAY CHSTATUS('QM2.TO.QM1')
DISPLAY CHSTATUS('QM3.TO.QM1')
```

3. キュー・マネージャーを接続し、トピックを定義します。
   子キュー・マネージャー QM2 と QM3 を親キュー・マネージャー QM1 に接続します。
   a) QM2 および QM3 で、親キュー・マネージャーを QM1 に設定します。

```
ALTER QMGR PARENT (QM1)
```

   b) すべてのキュー・マネージャーで次のコマンドを実行して、子キュー・マネージャーが親キュー・マネージャーに接続していることを確認します。

```
DISPLAY PUBSUB TYPE(ALL)
```

   コマンド出力が表示されます。 例えば、以下は、主要な詳細が強調表示された QM1 の出力です。

```
DISPLAY PUBSUB ALL
1 : DISPLAY PUBSUB ALL
AMQ8723: Display pub/sub status details.
QMNAME(QM1)               TYPE(LOCAL)
STATUS(ACTIVE)             SUBCOUNT(6)
TPCOUNT(9)
AMQ8723: Display pub/sub status details.
QMNAME(QM2) TYPE(CHILD)
STATUS(ACTIVE) SUBCOUNT(NONE)
TPCOUNT(NONE)
AMQ8723: Display pub/sub status details.
QMNAME(QM3) TYPE(CHILD)
STATUS(ACTIVE) SUBCOUNT(NONE)
TPCOUNT(NONE)
```

4. amqspub.exe と amqssub.exe というアプリケーションを使用して、トピックを公開し、購読します。
   a) 1 番目のコマンド・ウィンドウで次のコマンドを実行します。

```
amqspub Sport/Soccer QM2
```

   b) 2 番目のコマンド・ウィンドウで次のコマンドを実行します。

```
amqssub Sport/Soccer QM1
```

c) 3 番目のコマンド・ウィンドウで次のコマンドを実行します。

```
amqssub Sport/Soccer QM3
```

タスクの結果

2 番目と 3 番目のコマンド・ウィンドウの amqssub.exe アプリケーションは、最初のコマンド・ウィンドウで発行されたメッセージを受け取ります。

関連タスク

43 ページの『パブリッシュ/サブスクライブ階層のシナリオ 1: キュー・マネージャー名の別名での Point-to-Point チャネルの使用』
これは、キュー・マネージャー間の接続を確立するために異なる方法でパブリッシュ/サブスクライブ階層をセットアップする 3 セットのシナリオのうち最初のものです。 このシナリオでは、キュー・マネージャー名の別名で Point-to-Point チャネルを使用するパブリッシュ/サブスクライブ階層をセットアップします。

49 ページの『パブリッシュ/サブスクライブ階層のシナリオ 3: クラスター・チャネルを使用してキュー・マネージャーを追加する』
これは、キュー・マネージャー間の接続を確立するために異なる方法でパブリッシュ/サブスクライブ階層をセットアップする 3 セットのシナリオのうち 3 番目のものです。 このシナリオでは、クラスター・チャネルを使ってキュー・マネージャーを階層に追加します。

パブリッシュ/サブスクライブ階層へのキュー・マネージャーの接続

パブリッシュ/サブスクライブ階層のシナリオ 3: クラスター・チャネルを使用してキュー・マネージャーを追加する

これは、キュー・マネージャー間の接続を確立するために異なる方法でパブリッシュ/サブスクライブ階層をセットアップする 3 セットのシナリオのうち 3 番目のものです。 このシナリオでは、クラスター・チャネルを使ってキュー・マネージャーを階層に追加します。

このタスクについて

このシナリオ・セットはすべて、 QM1 という親キュー・マネージャーと、 QM2 と QM3 という 2 つの子キュー・マネージャーを使用します。

注 : このシナリオでは、クラスター化トピックを介してパブリッシュ/サブスクライブ・トラフィックを伝搬するためではなく、キュー・マネージャーを互いに接続するためにのみ、クラスター構成を使用しています。 同じクラスター内のキュー・マネージャー間の親子階層関係を定義すると、キュー・マネージャー間のパブリケーションの伝搬は、トピック・ツリー内のトピックのパブリケーションおよびサブスクリプションの有効範囲設定に基づいて行われます。 クラスターの中にトピックを追加するときには、トピックのクラスター名設定を使用しないことが重要です。 クラスター名を使用すると、トポロジーはパブリッシュ/サブスクライブ・クラスターになり、親子階層関係を定義する必要がなくなってしまいます。 36 ページの『シナリオ: パブリッシュ/サブスクライブ・クラスターの作成』、および分散パブリッシュ/サブスクライブ・ネットワークの計画を参照してください。

このシナリオでは、43 ページの『パブリッシュ/サブスクライブ階層のシナリオ 1: キュー・マネージャー名の別名での Point-to-Point チャネルの使用』 のステップ 1、3、および 4 を再利用します。

このシナリオでは、QM1 と QM2 を完全リポジトリーとし、QM3 を部分リポジトリーとするクラスター DEMO を作成します。 キュー・マネージャー QM1 はキュー・マネージャー QM2 と QM3 の親です。

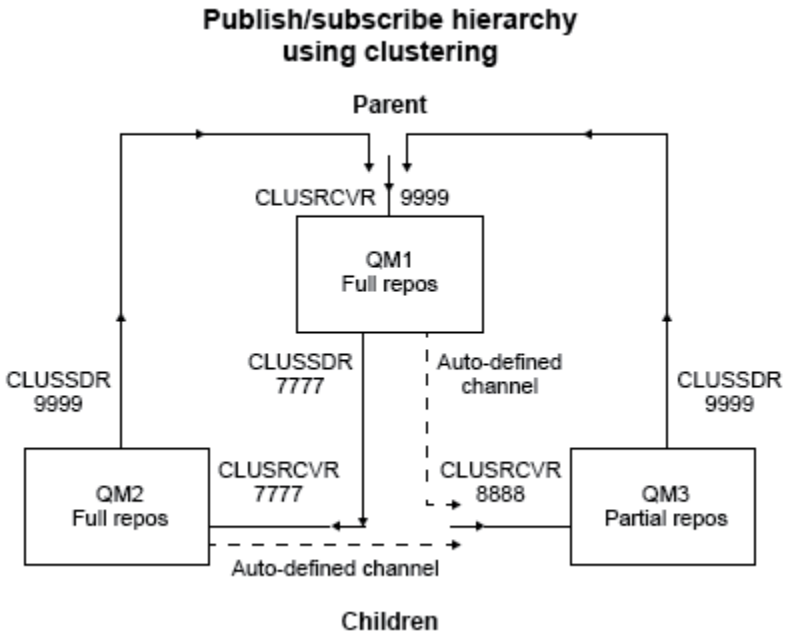


図 *5.* クラスター・チャネルを使用しているキュー・マネージャー間の関係を示すトポロジー図。

手順

1. キュー・マネージャーを作成します。
   a) 以下のコマンドを使用して、3 つのキュー・マネージャー QM1、QM2、および QM3 を作成して開始します。

```
crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM1
strmqm QM1

crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM2
strmqm QM2

crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM3
strmqm QM3
```

   b) 3 つのすべてのキュー・マネージャーで以下のコマンドを使用して、キュー・マネージャー・パブリッシュ/サブスクライブ・モードを有効にします。

```
ALTER QMGR PSMODE(ENABLED)
```

2. クラスター内のキュー・マネージャー間で Point-to-Point チャネル接続を確立します。
   a) QM1 および QM2 で、**REPOS** パラメーターをクラスター DEMO の名前に設定します。

```
ALTER QMGR REPOS(DEMO)
```

   b) キュー・マネージャーで該当のリスナーを開始します。

```
runmqlsr -m QM1 -t TCP -p 9999 &
runmqlsr -m QM2 -t TCP -p 7777 &
runmqlsr -m QM3 -t TCP -p 8888 &
```

   c) それぞれのキュー・マネージャーでクラスター受信側チャネルを定義します。

i) On QM1:

```
DEFINE CHANNEL(TO.QM1) CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CONNAME('localhost(9999)')
CLUSTER(DEMO)
```

ii) On QM2:

```
DEFINE CHANNEL(TO.QM2) CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CONNAME('localhost(7777)')
CLUSTER(DEMO)
```

iii) On QM3:

```
DEFINE CHANNEL(TO.QM3) CHLTYPE(CLUSRCVR) TRPTYPE(TCP) CONNAME('localhost(8888)')
CLUSTER(DEMO)
```

d) クラスター内の各キュー・マネージャーで完全リポジトリーに対するクラスター送信側チャネルを定義します。

i) On QM1:

```
DEFINE CHANNEL(TO.QM2) CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME('localhost(7777)')
CLUSTER(DEMO)
```

ii) On QM2:

```
DEFINE CHANNEL(TO.QM1) CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME('localhost(9999)')
CLUSTER(DEMO)
```

iii) QM3 は、QM1 と QM2 のいずれかの完全リポジトリーに対するクラスター送信側チャネルをもつことができます。 この例では、QM1 に対するチャネルを定義します。

```
DEFINE CHANNEL(TO.QM1) CHLTYPE(CLUSSDR) TRPTYPE(TCP) CONNAME('localhost(9999)')
CLUSTER(DEMO)
```

3. キュー・マネージャーを接続し、トピックを定義します。

子キュー・マネージャー QM2 と QM3 を親キュー・マネージャー QM1 に接続します。

a) QM2 および QM3 で、親キュー・マネージャーを QM1 に設定します。

```
ALTER QMGR PARENT (QM1)
```

b) すべてのキュー・マネージャーで次のコマンドを実行して、子キュー・マネージャーが親キュー・マネージャーに接続していることを確認します。

```
DISPLAY PUBSUB TYPE(ALL)
```

コマンド出力が表示されます。 例えば、以下は、主要な詳細が強調表示された QM1 の出力です。

```
DISPLAY PUBSUB ALL
1 : DISPLAY PUBSUB ALL
AMQ8723: Display pub/sub status details.
QMNAME(QM1)                TYPE(LOCAL)
STATUS(ACTIVE)              SUBCOUNT(6)
TPCOUNT(9)
AMQ8723: Display pub/sub status details.
QMNAME(QM2) TYPE(CHILD)
STATUS(ACTIVE) SUBCOUNT(NONE)
TPCOUNT(NONE)
AMQ8723: Display pub/sub status details.
QMNAME(QM3) TYPE(CHILD)
```

```
STATUS(ACTIVE) SUBCOUNT(NONE)
TPCOUNT(NONE)
```

4. amqspub.exe と amqssub.exe というアプリケーションを使用して、トピックを公開し、購読します。

   a) 1 番目のコマンド・ウィンドウで次のコマンドを実行します。

   ```
   amqspub Sport/Soccer QM2
   ```

   b) 2 番目のコマンド・ウィンドウで次のコマンドを実行します。

   ```
   amqssub Sport/Soccer QM1
   ```

   c) 3 番目のコマンド・ウィンドウで次のコマンドを実行します。

   ```
   amqssub Sport/Soccer QM3
   ```

タスクの結果

2 番目と 3 番目のコマンド・ウィンドウの amqssub.exe アプリケーションは、最初のコマンド・ウィンドウで発行されたメッセージを受け取ります。

関連タスク

43 ページの『パブリッシュ/サブスクライブ階層のシナリオ 1: キュー・マネージャー名の別名での Point-to-Point チャネルの使用』
これは、キュー・マネージャー間の接続を確立するために異なる方法でパブリッシュ/サブスクライブ階層をセットアップする 3 セットのシナリオのうち最初のものです。 このシナリオでは、キュー・マネージャー名の別名で Point-to-Point チャネルを使用するパブリッシュ/サブスクライブ階層をセットアップします。

46 ページの『パブリッシュ/サブスクライブ階層のシナリオ 2: 伝送キューとリモート・キュー・マネージャーに同じ名前を指定した Point-to-Point チャネルの使用』
これは、キュー・マネージャー間の接続を確立するために異なる方法でパブリッシュ/サブスクライブ階層をセットアップする 3 セットのシナリオのうち 2 番目のものです。 このシナリオでは、リモート・キュー・マネージャーと同じ伝送キュー名で Point-to-Point チャネルを使用するパブリッシュ/サブスクライブ階層をセットアップします。

パブリッシュ/サブスクライブ階層へのキュー・マネージャーの接続

トランザクション・サポートのシナリオ

トランザクション・サポートを使用すると、信頼性の高い方法でアプリケーションからデータベースを処理できるようになります。

このセクションでは、トランザクション・サポートについて説明します。 データベース製品を含む IBM MQ をアプリケーションで利用可能にするために必要な作業には、アプリケーション・プログラミングおよびシステム管理の分野が含まれます。 この情報および作業単位のコミットとバックアウトを使用してください。

まずトランザクションを形成する作業単位の概要を示し、次に IBM MQ がトランザクションをデータベースで調整できるようにする方法を説明します。

関連概念

52 ページの『作業単位の紹介』
このトピックでは、作業単位、コミット、バックアウト、および同期点についての一般的な概念を紹介および定義します。 また、グローバル作業単位について説明する 2 つのシナリオも記載しています。

作業単位の紹介

このトピックでは、作業単位、コミット、バックアウト、および同期点についての一般的な概念を紹介および定義します。 また、グローバル作業単位について説明する 2 つのシナリオも記載しています。

プログラムが作業単位内のキューにメッセージを書き込むと、それらのメッセージは、そのプログラムが作業単位をコミット したときにのみ、他のプログラムから見えるようになります。 作業単位をコミットするには、データの安全性を保護するためにすべての更新処理が正常終了する必要があります。

プログラムがエラーを検出し、PUT 操作を永続的にしないと決めた場合、プログラムは、その作業単位をバックアウト できます。 プログラムがバックアウトを行うと、IBM MQ はその作業単位によってキューに書き込まれたメッセージを除去することにより、キューを復元します。

同様に、プログラムが作業単位内の 1 つ以上のキューからメッセージを読み取ると、それらのメッセージは、プログラムがその作業単位をコミットするまでキューに留まりますが、それらのメッセージを他のプログラムで取り出すことはできません。 それらのメッセージは、プログラムが作業単位をコミットしたときに、キューから永続的に削除されます。 プログラムが作業単位をバックアウトすると、IBM MQ は、メッセージが他のプログラムによって検索できるようにすることによって、キューを復元します。

変更をコミットまたはバックアウトする決定は、最も単純な場合、タスクの終了時に行われます。 しかし、アプリケーションにとってはタスク内の他の論理点でデータ変更の同期をとる方が好都合な場合があります。 論理点は同期点と呼ばれ、2 つの同期点間での一連の更新を処理する期間を作業単位 といいます。 1 つの作業単位に、複数の MQGET 呼び出しや MQPUT 呼び出しを含めることができます。

IBM MQ では、ローカル 作業単位とグローバル 作業単位を区別する必要があります。

ローカル作業単位

IBM MQ キューへの書き込みおよびキューからの読み取りのみを行う作業単位。各作業単位の調整は、単一フェーズ・コミット・プロセスを使用してキュー・マネージャー内で行われます。

更新されるリソースが、単一の IBM MQ キュー・マネージャーによって管理されるキューのみである場合は、ローカル作業単位を使用します。 更新のコミットには MQCMIT verb を、更新のバックアウトには MQBACK をそれぞれ使用します。

ローカル作業単位の使用に関して、ログ管理以外にシステム管理タスクはありません。 ご使用のアプリケーションで、MQCMIT および MQBACK と共に MQPUT 呼び出しと MQGET 呼び出しを使用する場合は、MQPMO_SYNCPOINT オプションと MQGMO_SYNCPOINT オプションを使用してみてください。 (ログ管理については、ログ・ファイルを管理するを参照してください。)

グローバル作業単位

リレーショナル・データベース内のテーブルなど、他のリソースの更新も行われる作業単位。 複数のリソース・マネージャー が関与するときは、*2* フェーズ・コミット・プロセスを使用してグローバル作業単位を調整するトランザクション・マネージャー・ソフトウェアが必要です。

Db2®、Oracle、Sybase、および Informix® などのリレーショナル・データベース・マネージャー・ソフトウェアの更新も必要な場合に、グローバル作業単位を使用します。

グローバル作業単位を使用するシナリオはいくつかあります。 ここでは 2 つのシナリオについて説明します。

1. 最初のシナリオでは、キュー・マネージャー自体がトランザクション・マネージャーとして機能します。 このシナリオでは、MQI verb がグローバル作業単位を制御します。グローバル作業単位は、MQBEGIN verb を使用してアプリケーションで開始され、次いで MQCMIT を使用してコミットされるか、MQBACK を使用してバックアウトされます。
2. 2 つ目のシナリオでは、トランザクション・マネージャー役割は、TXSeries®、Encina、または Tuxedo など、他のソフトウェアによって実行されます。 このシナリオでは、トランザクション・マネージャー・ソフトウェアによって提供される API を使用して、作業単位を制御します (例えば、TXSeries の場合は EXEC CICS® SYNCPOINT)。

以上の 2 つのシナリオで編成された、グローバル作業単位を使用するために必要なすべての手順について、以下のセクションで説明します。

- 54 ページの『シナリオ 1: キュー・マネージャーが調整を行う』
- 79 ページの『シナリオ 2: 他のソフトウェアが調整を行う』

Multi シナリオ 1: キュー・マネージャーが調整を行う

シナリオ 1 では、キュー・マネージャーがトランザクション・マネージャーとして機能します。 このシナリオでは、MQI verb がグローバル作業単位を制御します。グローバル作業単位は、MQBEGIN verb を使用してアプリケーションで開始され、次いで MQCMIT を使用してコミットされるか、MQBACK を使用してバックアウトされます。

Multi 分離レベル

IBM MQ では、データベース内に実装されたトランザクション分離設計に応じて、データベースが更新される前にキュー上のメッセージが表示されることがあります。

IBM MQ キュー・マネージャーが XA トランザクション・マネージャーとして稼働している場合、XA リソース・マネージャーに対する更新を調整するには、以下のコミット・プロトコルに従います。

1. すべての XA リソース・マネージャーを準備します。
2. IBM MQ キュー・マネージャーのリソース・マネージャーをコミットします。
3. その他のリソース・マネージャーをコミットします。

ステップ 2 と 3 の間に、キューにコミットされているメッセージをアプリケーションが認識することがありますが、データベース内の対応する行はこのメッセージを反映していません。

アプリケーションのデータベース API 呼び出しが保留中の更新の完了を待つようにデータベースが構成されている場合、このことは問題にはなりません。

データベースの構成を変えると、この問題を解決できます。 必要な構成のタイプは、「分離レベル」と呼ばれます。 分離レベルについて詳しくは、データベースの資料を参照してください。 あるいは、以下の逆順でリソース・マネージャーをコミットするようにキュー・マネージャーを構成することもできます。

1. すべての XA リソース・マネージャーを準備します。
2. その他のリソース・マネージャーをコミットします。
3. IBM MQ キュー・マネージャーのリソース・マネージャーをコミットします。

プロトコルを変更すると、IBM MQ キュー・マネージャーは最後にコミットされるので、キューからメッセージを読み取るアプリケーションは、対応するデータベースの更新の完了後のみメッセージを認識します。

この変更されたプロトコルを使用するようにキュー・マネージャーを構成するには、**AMQ_REVERSE_COMMIT_ORDER** 環境変数を設定します。

この環境変数は、**strmqm** を実行してキュー・マネージャーを開始する環境で設定します。 例えば、キュー・マネージャーを開始する直前にシェルで以下のコマンドを実行します。

```
export AMQ_REVERSE_COMMIT_ORDER=1
```

注 : この環境変数を設定すると、トランザクションごとに追加のログ・エントリーが発生する可能性があるので、各トランザクションのパフォーマンスに少々影響があります。

Multi データベースの調整

キュー・マネージャーでグローバル作業単位自体を調整している場合は、データベースの更新を作業単位に統合できます。 つまり MQI と SQL の混合アプリケーションを作成し、MQCMIT verb および MQBACK verb を使用して、キューとデータベースへの変更をまとめてコミットまたはロールバックすることができます。

キュー・マネージャーは、「*X/Open Distributed Transaction Processing: The XA Specification*」に記載されている 2 フェーズ・コミット・プロトコルを使用してこれを行います。 作業単位をコミットする場合、キュー・マネージャーはまず作業単位にかかわっている各データベース・マネージャーに対して、更新する準備ができているかどうかを確認します。 キュー・マネージャーをはじめ、作業単位にかかわっているすべてのプログラムおよびデータベースがコミットできる状態にある場合にのみ、キューおよびデータベースの更新がすべてコミットされます。 更新できない状態にあるプログラムが 1 つでもあると、作業単位はバックアウトされます。

通常、グローバル作業単位は、以下の方式 (疑似コード) でアプリケーションにインプリメントされます。

```
MQBEGIN
MQGET (メッセージ・オプションにフラグ MQGMO_SYNCPOINT を組み込む)
MQPUT (メッセージ・オプションにフラグ MQPMO_SYNCPOINT を組み込む)
SQL INSERT
MQCMIT
```

MQBEGIN は、グローバル作業単位の先頭を指示するためのものです。 MQCMIT は、2 フェーズ・コミット・プロトコルを使用して、グローバル作業単位の終わりを指示し、関与するすべてのリソース・マネージャーについてこの終わりを完了するためのものです。

作業単位 (トランザクション ともいいます) が MQCMIT を使用して正常に完了されると、その作業単位で取られたすべてのアクションが永続的または不可逆的にされます。 何らかの理由で、作業単位が失敗した場合は、すべてのアクションがバックアウトされます。 1 つの作業単位内の 1 つのアクションを永続的にし、別のアクションをバックアウトすることはできません。 これが、作業単位の原理です。つまり、作業単位内のすべてのアクションを永続的にするか、すべてを非永続的にするかになります。

注 :

1. アプリケーション・プログラマーは、MQBACK を呼び出すことによって作業単位を強制的にバックアウトできます。 MQCMIT が呼び出される前にアプリケーションまたはデータベースが失敗 した場合は、キュー・マネージャーで作業単位をバックアウトすることもできます。
2. アプリケーションが MQCMIT を呼び出さずに MQDISC を呼び出した場合、キュー・マネージャーは、MQCMIT が呼び出されている場合と同様に振る舞い、作業単位をコミットします。

MQBEGIN から MQCMIT までの間では、キュー・マネージャーは、そのリソースを更新するためにデータベースへの呼び出しを行いません。 すなわち、データベースのテーブルを変更するには、コードを作成 (例えば、疑似コードでの SQL INSERT) する方法しかありません。

コミット・プロトコル中にキュー・マネージャーとデータベース・マネージャーとの間の連絡が断たれた場合には、完全リカバリー・サポートを使用できます。 データベース・マネージャーが未確定のうちに使用できなくなった場合、つまりコミットできる状態になっているがコミットまたはバックアウトの決定をまだ受信していない場合、キュー・マネージャーはその結果がデータベースに正常に送信できるまで作業単位の実行結果を保管しておきます。 同様に、コミット操作が完了していないままの状態でキュー・マネージャーが終了した場合には、キュー・マネージャーの再開時にはこの未完了の状態が保管されています。アプリケーションが突然終了した場合、作業単位の保全性に影響はありませんが、56 ページの表 2 に記載されているとおり、結果は、アプリケーションが終了したプロセスの場所によって異なります。

データベースまたはアプリケーション・プログラムが失敗した時点で起こることを、以下の表に要約します。

表 *1.* データベース・サーバーが失敗したときに起こること

失敗発生時	**結果**
アプリケーションが MQCMIT を呼び出す前。	作業単位がバックアウトされます。
アプリケーションによる MQCMIT の呼び出し中で、すべてのデータベースが正しく準備されていることを示す**前**。	理由コード MQRC_BACKED_OUT が返され、作業単位はバックアウトされます。
アプリケーションによる MQCMIT の呼び出し中で、すべてのデータベースが正しく準備されていることを示した**後**、ただし、正常にコミットしたことを示す前。	理由コード MQRC_OUTCOME_PENDING が返され、作業単位は、キュー・マネージャーによりリカバリー可能な状態に保持されます。
アプリケーションによる MQCMIT の呼び出し中で、すべてのデータベースが正しくコミットしたことを示した**後**。	理由コード MQRC_NONE が返され、作業単位はコミットされます。
アプリケーションが MQCMIT を呼び出した後。	理由コード MQRC_NONE が返され、作業単位はコミットされます。

表 2. アプリケーション・プログラムが失敗したときに起こること	
失敗発生時	**結果**
アプリケーションが MQCMIT を呼び出す前。	作業単位がバックアウトされます。
アプリケーションによる MQCMIT の呼び出し中で、キュー・マネージャーがアプリケーションの MQCMIT 要求を受信する**前**。	作業単位がバックアウトされます。
アプリケーションによる MQCMIT の呼び出し中で、キュー・マネージャーがアプリケーションの MQCMIT 要求を受信した**後**。	キュー・マネージャーは、2 フェーズ・コミットを使用してコミットを試みます (作業単位のそれぞれの部分を正しく実行およびコミットするデータベース製品の影響を受けます)。

MQCMIT から戻される途中の理由コードが MQRC_OUTCOME_PENDING の場合、作業単位は、データベース・サーバーとの連絡を再確立し、データベース・サーバーに作業単位のその部分をコミットするよう指示できるようになるまでキュー・マネージャーによって保管されます。 リカバリーの方法とタイミングについては、73 ページの『XA リソース・マネージャーとの連絡が失われる際の考慮事項』 を参照してください。

キュー・マネージャーは、「*X/Open Distributed Transaction Processing: The XA Specification* 」に記載されている XA インターフェースを使用してデータベース・マネージャーと通信します。 これらの関数呼び出しの例として、xa_open、xa_start、 xa_end、xa_prepare、xa_commit があります。 トランザクション・マネージャー とリソース・マネージャー という用語は、XA 仕様で使用されているのと同じ意味で使用されています。

Multi 制約事項

データベース調整サポートに対する制約事項があります。

次の制約事項が適用されます。

- IBM MQ の作業単位内でのデータベース更新の調整は、MQI クライアント・アプリケーションでは**サポートされていません**。 クライアント・アプリケーションでの MQBEGIN の使用は失敗します。 MQBEGIN を呼び出すプログラムは、キュー・マネージャーと同じマシン上でサーバー・アプリケーションとして実行する必要があります。

  注 : サーバー・アプリケーションとは、必要な IBM MQ サーバー・ライブラリーとリンクされているプログラムです。また、クライアント・アプリケーションとは、必要な IBM MQ クライアント・ライブラリーとリンクされているプログラムです。 プロシージャー型言語で作成するプログラムのコンパイルおよびリンクの詳細については、IBM MQ MQI clients 用のアプリケーションの作成 および プロシージャー・アプリケーションの作成 を参照してください。
- データベース・クライアントがキュー・マネージャーと同じマシン上にインストールされており、それがこの機能をサポートしている限り、データベース・サーバーをキュー・マネージャー・サーバーとは別のマシンに置くことは可能です。 データベース製品のクライアント・ソフトウェアを 2 フェーズ・コミット・システムに使用できるかどうかを判別するには、データベース製品の資料を参照してください。
- キュー・マネージャーは (シナリオ 2 のグローバル作業単位に関与する目的で) リソース・マネージャーとして動作しますが、あるキュー・マネージャーにそのシナリオ 1 のグローバル作業単位内で別のキュー・マネージャーを調整させることはできません。

Multi スイッチ・ロード・ファイル

スイッチ・ロード・ファイルは、 IBM MQ アプリケーションおよびキュー・マネージャーのコードによってロードされる共有ライブラリー (Windows システム上の DLL) です。 これは、データベースのクライアント共用ライブラリーのロードを単純化し、XA 関数にポインターを返すためのものです。

スイッチ・ロード・ファイルの詳細は、キュー・マネージャーを始動する前に指定する必要があります。 詳細は、AIX, Linux, and Windows システムの qm.ini ファイルに記載されています。

- Windows および Linux (x86 および x86-64 プラットフォーム) システムでは、IBM MQ Explorer を使用して qm.ini ファイルを更新します。

- その他のすべてのシステムでは、qm.ini ファイルを直接編集します。

スイッチ・ロード・ファイルの C ソースは、シナリオ 1 のグローバル作業単位をサポートする場合、IBM MQ のインストールにより提供されます。 ソースには MQStart という関数が含まれています。 スイッチ・ロード・ファイルがロードされると、キュー・マネージャーによってこの関数が呼び出され、この関数から *XA* スイッチ と呼ばれる構造体のアドレスが戻されます。

XA スイッチ構造体は、データベース・クライアント共用ライブラリーに入っており、57 ページの表 3 に記載されているように、多数の関数ポインターが含まれています。

表 *3. XA* スイッチ関数ポインター

関数ポインター名	**XA 関数**	**目的**
xa_open_entry	xa_open	データベースへの接続
xa_close_entry	xa_close	データベースからの切断
xa_start_entry	xa_start	グローバル作業単位のブランチの開始
xa_end_entry	xa_end	グローバル作業単位のブランチの中断
xa_rollback_entry	xa_rollback	グローバル作業単位のブランチのロールバック
xa_prepare_entry	xa_prepare	グローバル作業単位のブランチをコミットするための準備
xa_commit_entry	xa_commit	グローバル作業単位のブランチのコミット
xa_recover_entry	xa_recover	データベースに未確定の作業単位があるかどうかの判定
xa_forget_entry	xa_forget	データベースがグローバル作業単位のブランチの記録を消去できるようにする
xa_complete_entry	xa_complete	グローバル作業単位のブランチの完了

アプリケーションでの最初の MQBEGIN 呼び出し時に、MQBEGIN の一部として実行される IBM MQ コードによりスイッチ・ロード・ファイルがロードされ、データベース共用ライブラリー内の xa_open 関数が呼び出されます。 同様に、キュー・マネージャーの始動時や、後続のその他の場面で、一部のキュー・マネージャー・プロセスによりスイッチ・ロード・ファイルがロードされ、xa_open が呼び出されます。

動的登録 を使用すると、xa_* 呼び出しの回数を低減できます。 この最適化手法の詳しい説明については、77 ページの『XA 動的登録』 を参照してください。

Multi データベース調整のためのシステムの構成

キュー・マネージャーが調整するグローバル作業単位でデータベース・マネージャーを使用するには、事前に次の作業を行ってください。 これらの作業について、以下で説明します。

- 58 ページの『データベース・プロダクトのインストールおよび構成』
- 58 ページの『スイッチ・ロード・ファイルの作成』
- 59 ページの『キュー・マネージャーへの構成情報の追加』
- 61 ページの『アプリケーションの作成および変更』
- 61 ページの『システムのテスト』

Multi データベース・プロダクトのインストールおよび構成
データベース・プロダクトをインストールして構成するには、プロダクト独自の資料を参照してください。このセクションの以下のトピックでは、構成に関する一般的な問題と、それらが IBM MQ とデータベースの間の相互協調処理にどのように関連しているかを説明します。

データベースの接続

キュー・マネージャーへの標準接続を確立するアプリケーションは、個別のローカル・キュー・マネージャー・エージェント・プロセスのスレッドと関連付けられます。 (ファースト・パス 接続でない接続が、このコンテキストの標準 接続です。 MQCONNX 呼び出しを使用したキュー・マネージャーへの接続を参照してください。)

アプリケーションによって **MQBEGIN** が出されたときは、このアプリケーションとエージェント・プロセスの両方がデータベース・クライアント・ライブラリー内の **xa_open** 関数を呼び出します。 この呼び出しに対する応答で、データベース・クライアント・ライブラリー・コードは、アプリケーションとキュー・マネージャー・プロセスの両方から 作業単位に必要となるデータベースに接続 します。 アプリケーションがキュー・マネージャーに接続している限り、これらのデータベースへの接続は維持されます。

このことは、データベースによってサポートされるユーザーまたは接続の数が限られている場合には重要な考慮事項となります。1 つのアプリケーション・プログラムをサポートするのに、データベースに対して 2 つの接続が確立されるためです。

クライアント/サーバー構成

IBM MQ キュー・マネージャーおよびアプリケーション・プロセスにロードされるデータベース・クライアント・ライブラリーは、そのサーバーとの間で送受信が**可能でなければなりません**。 以下の事項について確認してください。

- データベースのクライアント/サーバー構成ファイルに正しい詳細がある
- 関係のある環境変数がキュー・マネージャー**および**アプリケーション・プロセスの環境で設定されている

Multi スイッチ・ロード・ファイルの作成
IBM MQ には、サポートされているデータベース・マネージャー用のスイッチ・ロード・ファイルの作成に使用する、サンプルの Make ファイルが付属しています。

MQ_INSTALLATION_PATH は、IBM MQ がインストールされている上位ディレクトリーを表します。

サンプルの Make ファイルは、スイッチ・ロード・ファイルの作成に必要なすべての関連 C ソース・ファイルと共に、次のディレクトリーにインストールされます。

- IBM MQ for Windows の場合、 *MQ_INSTALLATION_PATH*`\tools\c\samples\xatm\` ディレクトリーにあります。
- IBM MQ for UNIX システムおよび Linux システムの場合は、*MQ_INSTALLATION_PATH*`/samp/xatm/` ディレクトリーにあります。

スイッチ・ロード・ファイルを作成するために使用されるサンプル・ターゲットは以下のようです：

- Db2 の場合、 db2swit が使用されます。
- Oracle の場合、 oraswit が使用されます。
- Informix の場合は、infswit が使用されます。
- Sybase の場合は、 sybswit が使用されます。

Windows 生成されたスイッチ・ファイルは、 `C:&#xa5;Program Files&#xa5;IBM&#xa5;MQ\exits` に置かれます。

Linux UNIX 32 ビット・キュー・マネージャーを使用している場合は、サンプルの Make ファイル xaswit.mak により、32 ビット・スイッチ・ロード・ファイルが `/var/mqm/exits` にインストールされます。

Linux UNIX 64 ビット・キュー・マネージャーを使用している場合は、サンプルの Make ファイル xaswit.mak により、32 ビット・スイッチ・ロード・ファイルが /var/mqm/exits に、64 ビット・スイッチ・ロード・ファイルが /var/mqm/exits64 にインストールされます。

V 9.3.0 Linux UNIX ご利用のシステムが 32 ビット・コンパイルをサポートしない場合、ご使用のデータベース専用の 64 ビットを使用してください。

- Db2 の場合、 db2swit64 を使用してください。
- Oracle の場合、 oraswit64 を使用してください。
- Informix の場合、 infswit64 を使用してください。
- Sybase の場合、 sybswit64 を使用してください。

ファイル・セキュリティー

ご使用のオペレーティング・システムで、IBM MQ の制御が及ばない原因により、IBM MQ によるスイッチ・ロード・ファイルのロードが失敗することがあります。 この場合、エラー・メッセージが IBM MQ エラー・ログに書き込まれます。また、MQBEGIN 呼び出しが失敗する可能性があります。 ご使用のオペレーティング・システムでスイッチ・ロード・ファイルのロードが失敗しないようにするには、次の要件を満たす必要があります。

1. スイッチ・ロード・ファイルは、qm.ini ファイルで指定された場所で使用可能である必要があります。
2. スイッチ・ロード・ファイルは、キュー・マネージャー・プロセスおよびアプリケーション・プロセスを含む、このファイルをロードする必要のあるすべてのプロセスによってアクセス可能である必要があります。
3. データベース製品から提供されるライブラリーを含む、スイッチ・ロード・ファイルが依存するすべてのライブラリーは、存在していてアクセス可能である必要があります。

Multi キュー・マネージャーへの構成情報の追加

データベース・マネージャー用のスイッチ・ロード・ファイルを作成し、それを安全な場所に保管したら、その場所をキュー・マネージャーに指定する必要があります。

場所を指定するには、以下のステップを実行します。

- Windows 及び Linux (x86 及び x86-64 プラットフォーム) では、 システムは IBM MQ エクスプローラーを使用します. XA リソース・マネージャーのキュー・マネージャー・プロパティー・パネルで、スイッチ・ロード・ファイルの詳細を指定します。
- 他のすべてのシステムにおいて、キュー・マネージャーの qm.ini ファイルにある、XAResourceManager スタンザのスイッチ・ロード・ファイルの詳細を指定します。

キュー・マネージャーで調整する予定のデータベースについて XAResourceManager スタンザを追加してください。 通常、データベースは 1 つのみなので、XAResourceManager スタンザも 1 つです。 複数のデータベースを含む複雑な構成についての詳細は、72 ページの『複数のデータベースの構成』を参照してください。 XAResourceManager スタンザの属性は次のとおりです。

Name=name

リソース・マネージャーを識別する、ユーザーが選択したストリング。 結果的に、それにより、XAResourceManager スタンザに名前が付けられます。 キュー・マネージャーの名前は必須指定属性で、長さは 31 文字までです。

ユーザーが選択する名前は固有のものでなければなりません。この qm.ini ファイル内にある、この名前を持つ XAResourceManager スタンザは 1 つでなければなりません。 また、この名前は、分かりやすい名前にしてください。dspmqtrn コマンドの実行時に、キュー・マネージャーがこの名前を使用して、キュー・マネージャー・エラー・ログ・メッセージと出力の両方でこのリソース・マネージャーを参照するためです。 (詳細については、74 ページの『dspmqtrn コマンドを使用した未解決の作業単位の表示』を参照してください。)

名前を選択し、キュー・マネージャーを始動したら、名前属性を変更しないでください。 構成の変更についての詳細は、76 ページの『構成情報の変更』を参照してください。

SwitchFile=name
これは、以前に作成した XA スイッチ・ロード・ファイルの名前です。 この属性は必須指定属性です。 キュー・マネージャーおよび IBM MQ アプリケーション・プロセス内のコードは、次の 2 つの時点でスイッチ・ロード・ファイルのロードを試みます。

1. キュー・マネージャーの始動時
2. IBM MQ アプリケーション・プロセスで初めて MQBEGIN を呼び出したとき

スイッチ・ロード・ファイルのセキュリティー属性および許可属性により、これらのプロセスがこのアクションを実行できるようにする必要があります。

XAOpenString=string
これは、IBM MQ コードがその呼び出しに入れてデータベース・マネージャーの xa_open 関数に渡すデータ・ストリングです。 この属性はオプションです。省略すると、ゼロ長のストリングが指定されたものと見なされます。

キュー・マネージャーおよび IBM MQ アプリケーション・プロセス内のコードは、次の 2 つの時点で xa_open 関数を呼び出します。

1. キュー・マネージャーの始動時
2. IBM MQ アプリケーション・プロセスで初めて MQBEGIN を呼び出したとき

このストリングの形式は、各データベース・プロダクトに固有のものであり、そのプロダクトの資料で詳しく説明しています。 一般に、xa_open ストリングには、キュー・マネージャーとアプリケーション・プロセスの両方でデータベースに接続できるようにするために認証情報 (ユーザー名とパスワード) が含まれています。

IBM MQ 8.0.0 Fix Pack 4 以降、XAOpenString にパスワードが含まれている場合、`qm.ini` ファイル内でパスワードをプレーン・テキストで表示するのではなく、この情報を保護するために IBM MQ を取得できます。 IBM MQ は、ユーザー名とパスワードを (暗号化した形式で) 異なるファイルに保管し、それらの資格情報を使用してデータベースに接続します。 詳細については、データベース認証の保護についての詳細を参照してください。

XACloseString=string
これは、IBM MQ コードがその呼び出しに入れてデータベース・マネージャーの xa_close 関数に渡すデータ・ストリングです。 この属性はオプションです。省略すると、ゼロ長のストリングが指定されたものと見なされます。

キュー・マネージャーおよび IBM MQ アプリケーション・プロセス内のコードは、次の 2 つの時点で xa_close 関数を呼び出します。

1. キュー・マネージャーの始動時
2. 先に MQBEGIN を呼び出してから、IBM MQ アプリケーション・プロセスで MQDISC を呼び出すとき

このストリングの形式は、各データベース・プロダクトに固有のものであり、そのプロダクトの資料で詳しく説明しています。 一般に、このストリングは空であるので、通常、XACloseString 属性を XAResourceManager スタンザから除外します。

ThreadOfControl=THREAD |PROCESS (デフォルト)
ThreadOfControl 値は THREAD または PROCESS です。 キュー・マネージャーはこの属性を使用してシリアライズします。 この属性はオプションです。省略すると、値 PROCESS が指定されたものと見なされます。

データベース・クライアント・コードにより、シリアライゼーションなしでスレッドが XA 関数を呼び出せる場合、ThreadOfControl の値として THREAD を使用できます。 キュー・マネージャーは、必要に応じて、同時に複数のスレッドからデータベース・クライアント共用ライブラリーの XA 関数を呼び出せると想定しています。

データベース・クライアント・コードによりスレッドがこの方法でその XA 関数を呼び出せない場合、ThreadOfControl の値は PROCESS でなければなりません。 この場合、キュー・マネージャーは、データベース・クライアント共用ライブラリーへのすべての呼び出しをシリアライズするため、特定のプロセス内から行われる呼び出しは一度に 1 つのみです。 アプリケーションが複数のスレッドを使用して実行する場合に、同様のシリアライゼーションを実行するようにしなければならないこともあります。

データベース・プロダクトがこの方法でマルチスレッド・プロセスに対処できるかどうかというこの問題は、そのプロダクトのベンダーの問題です。 ThreadOfControl 属性を THREAD または PROCESS に設定できるかどうかについては、該当するデータベース・プロダクトの資料で詳しく説明しています。 できれば、ThreadOfControl を THREAD に設定することをお勧めします。 未確定の場合、安全性のより高い オプションは、PROCESS に設定することです。ただし、THREAD を使用した場合に可能なパフォーマンスの向上は実現されません。

Multi アプリケーションの作成および変更

グローバル作業単位のインプリメント方法を示します。

IBM MQ のインストールにより提供されるシナリオ 1 のグローバル作業単位用のサンプル・アプリケーション・プログラムについては、52 ページの『作業単位の紹介』で説明されています。

通常、グローバル作業単位は、以下の方式 (疑似コード) でアプリケーションにインプリメントされます。

```
MQBEGIN
MQGET
MQPUT
SQL INSERT
MQCMIT
```

MQBEGIN は、グローバル作業単位の先頭を指示するためのものです。 MQCMIT は、2 フェーズ・コミット・プロトコルを使用して、グローバル作業単位の終わりを指示し、関与するすべてのリソース・マネージャーについてこの終わりを完了するためのものです。

MQBEGIN から MQCMIT までの間では、キュー・マネージャーは、そのリソースを更新するためにデータベースへの呼び出しを行いません。 すなわち、データベースのテーブルを変更するには、コードを作成 (例えば、疑似コードでの SQL INSERT) する方法しかありません。

キュー・マネージャーの役割は、データベースに関する限り、グローバル作業単位の開始時、終了時、およびグローバル作業単位をコミットまたはロールバックするかどうかをデータベースに指示することです。

アプリケーションに関する限り、キュー・マネージャーは、グローバル作業単位に対して、リソース・マネージャー (リソースがキュー上のメッセージである場合) およびトランザクション・マネージャーという 2 つの役割を担います。

付属のサンプル・プログラムから始め、これらのプログラム内で実行されるさまざまな IBM MQ 呼び出しおよびデータベース API 呼び出しの動作を確認してください。 関係する API 呼び出しについては、IBM MQ プロシージャー型プログラムのサンプル、MQI で使用されるデータ・タイプ、およびデータベース固有の資料 (データベース固有の API の場合) に詳しく記載されています。

Multi システムのテスト

アプリケーションおよびシステムが正しく構成されていることを確認する方法は、テスト時にそれらを実行する方法のみです。 付属のサンプル・プログラムの 1 つを作成して実行することにより、システムの構成 (キュー・マネージャーとデータベース間での通信が正常に行えるかどうか) をテストできます。

Multi Db2 の構成

Db2 のサポートおよび構成情報。

サポートされる Db2 のレベルは、IBM MQ のシステム要件 のページで定義されています。

注 : Db2 の 32 ビット・インスタンスは、キュー・マネージャーが 64 ビットのプラットフォームではサポートされていません。

以下の作業を行います。

1. 環境変数の設定の確認
2. Db2 スイッチ・ロード・ファイルの作成
3. リソース・マネージャー構成情報の追加
4. 必要な場合に、Db2 構成パラメーターの変更

この情報を、57 ページの『データベース調整のためのシステムの構成』に記載されている一般情報と共にお読みください。

警告 : AIX and Linux プラットフォームで db2profile を実行する場合は、環境変数 LIBPATH および LD_LIBRARY_PATH が設定されます。 これらの環境変数を unset することをお勧めします。 詳細情報は crtmqenv または setmqenv を参照してください。

Db2 環境変数の設定の確認

Db2 環境変数が、アプリケーション・プロセス **だけでなく** 、キュー・マネージャー・プロセスに対しても設定されていることを確認してください。 特に、キュー・マネージャーを開始する **前に** 、常に Db2INSTANCE 環境変数を設定する必要があります。 DB2INSTANCE 環境変数は、更新する Db2 データベースが含まれる Db2 インスタンスを識別します。 以下に例を示します。

- AIX and Linux システムでは、以下を使用します。

```
export DB2INSTANCE=db2inst1
```

- Windows システムでは、以下を使用します。

```
set DB2INSTANCE=Db2
```

Db2 データベースを使用する Windows では、キュー・マネージャーを開始できるようにするために、ユーザー MUSR_MQADMIN を DB2USERS グループに追加する必要があります。

Db2 スイッチ・ロード・ファイルの作成

Db2 スイッチ・ロード・ファイルを作成する最も簡単な方法は、サンプル・ファイル xaswit.mak を使用する方法です。このファイルは、各種のデータベース・プロダクトについてスイッチ・ロード・ファイルを作成するために IBM MQ により提供されます。

Windows Windows システムの場合、 xaswit.mak はディレクトリー *MQ_INSTALLATION_PATH*\tools\c\samples\xatm にあります。 *MQ_INSTALLATION_PATH* は、IBM MQ がインストールされている上位ディレクトリーを表します。 Microsoft Visual C++ で Db2 スイッチ・ロード・ファイルを作成するには、以下を使用します。

```
nmake /f xaswit.mak db2swit.dll
```

生成されたスイッチ・ファイルは、C:¥Program Files¥IBM¥MQ\exits に置かれます。

Linux UNIX xaswit.mak は、ディレクトリー *MQ_INSTALLATION_PATH*/samp/xatm にあります。 *MQ_INSTALLATION_PATH* は、IBM MQ がインストールされている上位ディレクトリーを表します。

xaswit.mak を編集して、使用している Db2 のバージョンに該当する行を コメント解除 します。 その後で、次のコマンドを使用して MAKE ファイルを実行します。

```
make -f xaswit.mak db2swit
```

生成された 32 ビットのスイッチ・ロード・ファイルは、 /var/mqm/exits にあります。

生成された 64 ビット・スイッチ・ロード・ファイルは、 /var/mqm/exits64 にあります。

V 9.3.0 ご使用のシステムが 32 ビット・コンパイルをサポートしない場合は、 64 ビットのみのターゲットを使用してください。

```
make -f xaswit.mak db2swit64
```

Db2 用のリソース・マネージャー構成情報の追加

キュー・マネージャー用の構成情報を変更して、Db2 をグローバル作業単位に参加するプログラムとして宣言する必要があります。 この方法による構成情報の変更は、59 ページの『キュー・マネージャーへの構成情報の追加』で詳しく説明されています。

- Windows および Linux (x86 および x86-64 プラットフォーム) システムでは、IBM MQ Explorer を使用します。 XA リソース・マネージャーのキュー・マネージャー・プロパティー・パネルで、スイッチ・ロード・ファイルの詳細を指定します。
- 他のすべてのシステムにおいて、キュー・マネージャーの qm.ini ファイルにある、XAResourceManager スタンザのスイッチ・ロード・ファイルの詳細を指定します。

63 ページの図 6 は、UNIX サンプルで、調整されるデータ ベースが mydbname と呼ばれる XAResourceManager 項目を示しています。この名前が XAOpenString で指定されます。

```
XAResourceManager:
  Name=mydb2
  SwitchFile=db2swit
  XAOpenString=mydbname,myuser,mypasswd,toc=t
  ThreadOfControl=THREAD
```

図 *6. Db2* 用サンプル *XAResourceManager* 項目 *(UNIX)*

注 :

1. ThreadOfControl=THREAD は、8 より前の Db2 バージョンでは使用できません。 ThreadOfControl および XAOpenString パラメーター toc を、以下のいずれかの組み合わせに設定します。
   - ThreadOfControl=THREAD と toc=t
   - ThreadOfControl=PROCESS と toc=p

   jdbcdb2 XA スイッチ・ロード・ファイルを使用して JDBC/JTA 調整を有効にする場合は、ThreadOfControl=PROCESS および toc=p を使用する必要があります。

Db2 構成パラメーターの変更

キュー・マネージャーが調整する各 Db2 データベースごとに、データベース権限を設定し、tp_mon_name パラメーターを変更し、maxappls パラメーターをリセットする必要があります。 これを行うには、以下のステップを実行します。

データベース特権の設定

キュー・マネージャー・プロセスは、AIX and Linux システム上では有効なユーザーおよびグループ mqm で稼働します。 Windows システムでは、それらのプロセスは、キュー・マネージャーを始動したユーザーとして稼働します。 このユーザーは、次のいずれかになります。

1. strmqm コマンドを発行したユーザー
2. IBM MQ Service COM サーバーを実行するユーザー

特に指定のない限り、このユーザーは MUSR_MQADMIN と呼ばれます。

xa_open ストリングでユーザー名とパスワードを指定しなかった場合、 Db2 は xa_open 呼び出しを認証するために **キュー・マネージャーを実行しているユーザー** を使用します。 このユーザー (例えば、AIX and Linux システム上のユーザー mqm) がデータベース内に最小限の特権を持っていない場合、データベースは xa_open 呼び出しの認証を拒否します。

同じことが、アプリケーション・プロセスにも当てはまります。 xa_open ストリングにユーザー名とパスワードを指定しなかった場合、Db2 は、アプリケーションが実行されているユーザーを使用して、最初の MQBEGIN 中に作成される xa_open 呼び出しを認証します。 さらに、このユーザーがデータベース内に最小限の特権を持っていないと、これは機能しません。

例えば、以下の Db2 コマンドを発行して、mydbname データベース内で mqm ユーザーに接続権限を付与してください。

```
db2 connect to mydbname
db2 grant connect on database to user mqm
```

セキュリティーの詳細については、73 ページの『セキュリティーに関する考慮事項』を参照してください。

Windows **TP_MON_NAME パラメーターの変更**

Windows システム上の Db2 について、TP_MON_NAME 構成パラメーターを Db2 が動的登録にキュー・マネージャーを喚起するために使用する DLL に変更します。

コマンド `db2 update dbm cfg using TP_MON_NAME mqmax` を使用して、Db2 がキュー・マネージャーを呼び出すために使用するライブラリーとして MQMAX.DLL を指定します。 これは PATH 内のディレクトリーに入っていなければなりません。

maxappls パラメーターのリセット

場合によっては *maxappls* パラメーターの設定を確認する必要があります。このパラメーターにより、1 つのデータベースに接続できるアプリケーションの数が制限されます。 58 ページの『データベース・プロダクトのインストールおよび構成』を参照してください。

Multi Oracle の構成

Oracle のサポートと構成に関する情報。

以下のステップを実行します。

1. 環境変数の設定の確認
2. Oracle スイッチ・ロード・ファイルの作成
3. リソース・マネージャー構成情報の追加
4. 必要な場合には、Oracle 構成パラメーターの変更

IBM MQ でサポートされている Oracle のレベルの現行リストについては、IBM MQ のシステム要件を参照してください。

Oracle 環境変数の設定の確認

Oracle 環境変数が、アプリケーション・プロセスだけでなく、キュー・マネージャー・プロセスに対しても設定されていることを確認します。 特に、キュー・マネージャーを始動する前に、必ず以下の環境変数を設定してください。

ORACLE_HOME

Oracle ホーム・ディレクトリー。 例えば、AIX and Linux システムでは、以下を使用します。

```
export ORACLE_HOME=/opt/oracle/product/8.1.6
```

Windows システムでは、以下を使用します。

```
set ORACLE_HOME=c:\oracle\ora81
```

ORACLE_SID

使用している Oracle SID。 クライアント/サーバーの接続に Net8 を使用している場合、この環境変数を設定する必要はありません。 お手元の Oracle 資料を参照してください。

次の例は、AIX and Linux システムにおけるこの環境変数の設定例です。

```
export ORACLE_SID=sid1
```

Windows システム上での同等値は、次のとおりです。

```
set ORACLE_SID=sid1
```

注: バイナリー・ディレクトリー (例えば、ORACLE_INSTALL_DIR/VERSION/32BIT_NAME/bin または ORACLE_INSTALL_DIR/VERSION/64BIT_NAME/bin) を組み込むように PATH 環境変数を設定する必要があります。そうでない場合、オラクライアントライブラリーがコンピューターから欠落していることを示すメッセージが表示されるかもしれません。

Windows 64 ビット・システムでキュー・マネージャーを実行する場合は、64 ビットの Oracle クライアントのみをインストールする必要があります。 64 ビットのキュー・マネージャーによってロードされたスイッチ・ロード・ファイルは、 Oracle 64 ビット・クライアント・ライブラリーにアクセスする必要があります。

Oracle スイッチ・ロード・ファイルの作成

Oracle スイッチ・ロード・ファイルを作成するには、サンプル・ファイルの xaswit.mak を使用します。そして IBM MQ がさまざまなデータベースプロダクトのスイッチ・ロード・ファイルを作成するために提供されます。

Windows Windows システムの場合、 xaswit.mak はディレクトリー C:¥Program Files¥IBM¥MQ\tools\c\samples\xatm にあります。 Microsoft Visual C++ を使用して Oracle スイッチ・ロード・ファイルを作成するには、以下のものを使用します。

```
nmake /f xaswit.mak oraswit.dll
```

注: これらのスイッチ・ロード・ファイルは、C アプリケーションでのみ使用できます。 Java アプリケーションについては、IBM MQ classes for Java を使用した JTA/JDBC の調整を参照してください。

生成されたスイッチ・ファイルは、 *MQ_INSTALLATION_PATH*\exits に置かれます。 *MQ_INSTALLATION_PATH* は、IBM MQ がインストールされている上位ディレクトリーを表します。

Linux UNIX xaswit.mak は、ディレクトリー *MQ_INSTALLATION_PATH*/samp/xatm にあります。 *MQ_INSTALLATION_PATH* は、IBM MQ がインストールされている上位ディレクトリーを表します。

xaswit.mak を編集して、ご使用の Oracle のバージョンに該当する行のコメントを外します。 その後で、次のコマンドを使用して MAKE ファイルを実行します。

```
make -f xaswit.mak oraswit
```

IBM MQ がインストールされている場合、 MQ_INSTALLATION_PATH/samp/xatm の内容は読み取り専用です。したがって、 xaswit.mak を編集するには、すべてのファイルを samp/xatm から別のディレクトリーにコピーし、xaswit.mak を変更して、そのディレクトリーから make -f xaswit.makoraswit を実行します。

生成された 32 ビット・スイッチ・ロード・ファイルは、/var/mqm/exits に保管されます。

生成された 64 ビット・スイッチ・ロード・ファイルは、/var/mqm/exits64 に保管されます。

V 9.3.0 ご使用のシステムが 32 ビット・コンパイルをサポートしない場合は、 64 ビットのみのターゲットを使用してください。

```
make -f xaswit.mak oraswit64
```

Oracle 用のリソース・マネージャー構成情報の追加

キュー・マネージャー用の構成情報を変更し、Oracle をグローバル作業単位に参加するプログラムとして宣言する必要があります。 キュー・マネージャー用の構成情報をこのように変更する方法については、59 ページの『キュー・マネージャーへの構成情報の追加』で詳しく説明されています。

- Windows および Linux (x86 および x86-64 プラットフォーム) システムでは、IBM MQ Explorer を使用します。 XA リソース・マネージャーのキュー・マネージャー・プロパティー・パネルで、スイッチ・ロード・ファイルの詳細を指定します。
- それ以外のすべてのシステムでは、キュー・マネージャーの qm.ini ファイルにある XAResourceManager スタンザのスイッチ・ロード・ファイルの詳細を指定します。

66 ページの図 7 は、XAResourceManager 項目を示した AIX and Linux システムのサンプルです。 XA オープン・ストリングに LogDir を追加して、すべてのエラーおよびトレース情報のログが同一の場所に記録されるようにする必要があります。

```
XAResourceManager:
  Name=myoracle
  SwitchFile=oraswit
  XAOpenString=Oracle_XA+Acc=P/myuser/mypasswd+SesTm=35+LogDir=/tmp+threads=true
  ThreadOfControl=THREAD
```

図 7. *Oracle* 用サンプル *XAResourceManager* 項目 *(AIX and Linux* プラットフォーム*)*

注 :

1. 66 ページの図 7 では、xa_open ストリングと 4 つのパラメーターが使用されています。 Oracle の資料に説明されているように、追加のパラメーターを指定できます。
2. IBM MQ パラメーターの ThreadOfControl=THREAD を使用する場合は、XAResourceManager スタンザで Oracle パラメーター +threads=true を使用する必要があります。

xa_open ストリングの詳細については、「*Oracle8 Server Application Developer's Guide*」を参照してください。

Oracle 構成パラメーターの変更

キュー・マネージャーが調整する各 Oracle データベースごとに、最大セッション数を確認してデータベース特権を設定する必要があります。 これを行うには、以下のステップを完了します。

最大セッションの確認
キュー・マネージャーにより制御されるプロセスでは、接続を増やす必要があることを考慮して、LICENSE_MAX_SESSIONS および PROCESSES の設定値を確認する必要がある場合があります。 詳しくは、58 ページの『データベース・プロダクトのインストールおよび構成』を参照してください。

データベース特権の設定
xa_open ストリングに指定される Oracle ユーザー名は、Oracle 資料に記載されているとおり、DBA_PENDING_TRANSACTIONS ビューへのアクセス権限が付与されているものでなければなりません。

必要な特権は、以下のコマンド例を使用して付与できます。

```
grant select on DBA_PENDING_TRANSACTIONS to myuser;
```

Multi Informix の構成

Informix のサポートおよび構成情報。

以下のステップを実行します。

1. 適切な Informix クライアント SDK がインストールされていることを確認します。
   - 32 ビットのキュー・マネージャーとアプリケーションでは、32 ビット Informix クライアント SDK が必要です。
   - 64 ビットのキュー・マネージャーとアプリケーションでは、64 ビット Informix クライアント SDK が必要です。
2. Informix データベースが正しく作成されていることを確認します。
3. 環境変数の設定の確認
4. Informix スイッチ・ロード・ファイルを作成します。
5. リソース・マネージャー構成情報の追加

IBM MQ でサポートされている Informix のレベルの現行リストについては、 IBM MQ のシステム要件を参照してください。

Informix データベースが正しく作成されていることの確認

IBM MQ キュー・マネージャーによって調整されるすべての Informix データベースは、 **log** パラメーターを指定して作成する必要があります。 以下に例を示します。

```
create database mydbname with log;
```

IBM MQ キュー・マネージャーは、作成時に `log` パラメーターが指定されていない Informix データベースを調整できません。 作成時に `log` パラメーターが指定されていない Informix データベースをキュー・マネージャーが調整しようとすると、 Informix への xa_open 呼び出しが失敗し、いくつかの FFST エラーが生成されます。

Informix 環境変数の設定の確認

Informix 環境変数が、アプリケーション・プロセス **だけでなく** 、キュー・マネージャー・プロセスに対しても設定されていることを確認してください。 特に、キュー・マネージャーを始動する**前**に、必ず以下の環境変数を設定してください。

INFORMIXDIR
Informix プロダクト・インストールのディレクトリー。

- 32 ビット AIX and Linux アプリケーションの場合は、以下のコマンドを使用します。

  ```
  export INFORMIXDIR=/opt/informix/32-bit
  ```

- 64 ビット AIX and Linux アプリケーションの場合は、以下のコマンドを使用します。

  ```
  export INFORMIXDIR=/opt/informix/64-bit
  ```

- Windows アプリケーションの場合は、以下のコマンドを使用します。

  ```
  set INFORMIXDIR=c:\informix
  ```

32 ビットと 64 ビットの両方のアプリケーションをサポートする必要がある 64 ビットのキュー・マネージャーを使用しているシステムの場合は、32 ビットと 64 ビット両方の Informix クライアント SDK をインストールする必要があります。 スイッチ・ロード・ファイルを作成する場合に使用するサンプル MAKE ファイル `xaswit.mak` も、両方の製品インストール・ディレクトリーを設定します。

INFORMIXSERVER
Informix サーバーの名前。 例えば、AIX and Linux システムでは、以下を使用します。

```
export INFORMIXSERVER=hostname_1
```

Windows システムでは、以下を使用します。

```
set INFORMIXSERVER=hostname_1
```

ONCONFIG
Informix サーバーの構成ファイルの名前。 例えば、AIX and Linux システムでは、以下を使用します。

```
export ONCONFIG=onconfig.hostname_1
```

Windows システムでは、以下を使用します。

```
set ONCONFIG=onconfig.hostname_1
```

Informix スイッチ・ロード・ファイルの作成

Informix スイッチ・ロード・ファイルを作成するには、サンプル・ファイル xaswit.mak を使用します。このファイルは、各種のデータベース製品用にスイッチ・ロード・ファイルを作成するために IBM MQ により提供されます。

Windows Windows システムでは、 xaswit.mak はディレクトリー *MQ_INSTALLATION_PATH*\tools\c\samples\xatm に入っています。 *MQ_INSTALLATION_PATH* は、IBM MQ がインストールされている上位ディレクトリーを表します。 Microsoft Visual C++ で Informix スイッチ・ロード・ファイルを作成するには、以下を使用します。

```
nmake /f xaswit.mak infswit.dll
```

生成されたスイッチ・ファイルは、C:¥Program Files¥IBM¥MQ\exits に置かれます。

Linux UNIX xaswit.mak は、ディレクトリー *MQ_INSTALLATION_PATH*/samp/xatm に入っています。 *MQ_INSTALLATION_PATH* は、IBM MQ がインストールされている上位ディレクトリーを表します。

xaswit.mak を編集し、使用している Informix のバージョンに該当する行を コメント解除 してください。その後で、次のコマンドを使用して MAKE ファイルを実行します。

```
make -f xaswit.mak infswit
```

生成された 32 ビット・スイッチ・ロード・ファイルは、/var/mqm/exits に保管されます。

生成された 64 ビット・スイッチ・ロード・ファイルは、/var/mqm/exits64 に保管されます。

V 9.3.0 ご使用のシステムが 32 ビット・コンパイルをサポートしない場合は、 64 ビットのみのターゲットを使用してください。

```
make -f xaswit.mak infswit64
```

Informix 用のリソース・マネージャー構成情報の追加

キュー・マネージャー用の構成情報を変更して、Informix をグローバル作業単位に参加するプログラムとして宣言する必要があります。 キュー・マネージャー用の構成情報をこのように変更する方法については、59 ページの『キュー・マネージャーへの構成情報の追加』 で詳しく説明されています。

- Windows および Linux (x86 および x86-64 プラットフォーム) システムでは、IBM MQ エクスプローラーを使用します。 XA リソース・マネージャーのキュー・マネージャー・プロパティー・パネルで、スイッチ・ロード・ファイルの詳細を指定します。
- 他のすべてのシステムでは、キュー・マネージャーの qm.ini ファイルにある XAResourceManager スタンザでスイッチ・ロード・ファイルの詳細を指定します。

69 ページの図 8 は、UNIX サンプルで、調整されるデータベースが mydbname と呼ばれる qm.ini の XAResourceManager 項目を示しています。この名前が XAOpenString で指定されています。

```
XAResourceManager:
  Name=myinformix
  SwitchFile=infswit
  XAOpenString=DB=mydbname@myinformixserver\;USER=myuser\;PASSWD=mypasswd
  ThreadOfControl=THREAD
```

図 *8. Informix* 用サンプル *XAResourceManager* 項目 *(UNIX)*

注 : デフォルトでは、UNIX 上のサンプル xaswit.mak は、スレッド化された Informix ライブラリーを使用するスイッチ・ロード・ファイルを作成します。 これらの Informix ライブラリーを使用するときには、ThreadOfControl が THREAD に設定されていることを確認する必要があります。 69 ページの図 8 では、qm.ini ファイルの XAResourceManager スタンザ属性 ThreadOfControl が THREAD に設定されています。 THREAD が指定されている場合、スレッド化された Informix ライブラリーおよび IBM MQ のスレッド化された API ライブラリーを使用してアプリケーションを構築する必要があります。

XAOpenString 属性には、データベース名の後に @ 記号、その後に Informix サーバー名を付けた値を指定する必要があります。

スレッド化されていない Informix ライブラリーを使用する場合は、qm.ini ファイルの XAResourceManager スタンザの ThreadOfControl 属性を PROCESS に設定する必要があります。 さらに、サンプル xaswit.mak に以下の変更を加える必要があります。

1. スレッド化されていないスイッチ・ロード・ファイルの生成をアンコメントする。
2. スレッド化されたスイッチ・ロード・ファイルの生成をコメント化する。

Multi Sybase 構成

Sybase のサポートおよび構成情報。

以下のステップを実行します。

1. 例えば XA DTM オプションをインストールすることにより、Sybase XA ライブラリーをインストールしていることを確認します。
2. 環境変数の設定の確認
3. Sybase XA サポートの使用可能化
4. Sybase スイッチ・ロード・ファイルの作成
5. リソース・マネージャー構成情報の追加

IBM MQ でサポートされている Sybase のレベルの現行リストについては、IBM MQ のシステム要件を参照してください。

Sybase 環境変数の設定の確認

Sybase 環境変数が、アプリケーション・プロセスだけでなく、キュー・マネージャー・プロセスに対しても設定されていることを確認します。 特に、キュー・マネージャーを始動する前に、必ず以下の環境変数を設定してください。

SYBASE

Linux AIX Sybase プロダクト・インストールの位置。 例えば、AIX and Linux システムでは、以下を使用します。

```
export SYBASE=/sybase
```

Windows Windows システムでは、以下を使用します。

```
set SYBASE=c:\sybase
```

SYBASE_OCS

Sybase クライアント・ファイルをインストールした SYBASE の下のディレクトリー。

Linux AIX 例えば、AIX and Linux システムでは、以下を使用します。

```
export SYBASE_OCS=OCS-12_0
```

Windows Windows システムでは、以下を使用します。

```
set SYBASE_OCS=OCS-12_0
```

Sybase XA サポートの使用可能化

Sybase XA 構成ファイル $SYBASE/$SYBASE_OCS/xa_config 内で、更新されている Sybase サーバーへの接続ごとに、ロジカルリソース・マネージャー (LRM) を定義します。 $SYBASE/$SYBASE_OCS/xa_config の内容の例を以下のようです。

```
# The first line must always be a comment
[xa]
  LRM=lrmname
  server=servername
```

Sybase スイッチ・ロード・ファイルの作成

Sybase スイッチ・ロード・ファイルを作成するには、IBM MQ に付属しているサンプル・ファイルを使用します。

Windows Windows システムの場合、 xaswit.mak はディレクトリー C:¥Program Files¥IBM¥MQ\tools\c\samples\xatm にあります。 Microsoft Visual C++ で Sybase スイッチ・ロード・ファイルを作成するには、以下を使用します。

```
nmake /f xaswit.mak sybswit.dll
```

生成されたスイッチ・ファイルは、C:¥Program Files¥IBM¥MQ\exits に置かれます。

Linux UNIX xaswit.mak は、ディレクトリー *MQ_INSTALLATION_PATH*/samp/xatm にあります。 *MQ_INSTALLATION_PATH* は、IBM MQ がインストールされている上位ディレクトリーを表します。

Sybase のバージョンに該当するラインから コメントを外す ために、xaswit.mak を編集します。 その後で、次のコマンドを使用して MAKE ファイルを実行します。

```
make -f xaswit.mak sybswit
```

生成された 32 ビットのスイッチ・ロード・ファイルは、 /var/mqm/exits にあります。

生成された 64 ビット・スイッチ・ロード・ファイルは、 /var/mqm/exits64 にあります。

V 9.3.0 ご使用のシステムが 32 ビット・コンパイルをサポートしない場合は、 64 ビットのみのターゲットを使用してください。

```
make -f xaswit.mak sybswit64
```

注 : AIX AIX では、次の例に示すようにサンプル Make ファイルが変更され、Sybase 15 ESD#5 以降を使用しているか、それより前のバージョンの Sybase を使用しているかに応じて異なる SYBLINKFLAG64 値を選択できるようになりました。

```
SYBLINKFLAGS32=-brtl
# The following line is for Sybase 15
#SYBLINKFLAGS64=-brtl
# The following line is for Sybase 16
SYBLINKFLAGS64=-bstatic -bdynamic
```

Make ファイルに加えなければならない変更は、SYBLINKFLAGS64 値のいずれかのコメントを外すことだけです。 デフォルト値は「Sybase 16」で、15 #ESD5 以降に対して使用します。

生成される XA スイッチ・ファイルは、Sybase の特定のリリースに関連付けられているため、他のプラットフォームに移動することはできません。

Sybase のレベルが変更された場合、XA スイッチ・ファイルを再ビルドする必要があります。

Sybase 用のリソース・マネージャー構成情報の追加

キュー・マネージャー用の構成情報を変更して、Sybase をグローバル作業単位に参加するプログラムとして宣言する必要があります。 構成情報の変更は、59 ページの『キュー・マネージャーへの構成情報の追加』で詳しく説明されています。

- Windows Linux Windows および Linux (x86 および x86-64 プラットフォーム) システムでは、IBM MQ Explorer を使用します。 XA リソース・マネージャーのキュー・マネージャー・プロパティー・パネルで、スイッチ・ロード・ファイルの詳細を指定します。
- その他すべてのシステムでは、キュー・マネージャーの qm.ini ファイルにある XAResourceManager スタンザでスイッチ・ロード・ファイルの詳細を指定します。

Linux AIX Sybase XA configuration ファイル$SYBASE/$SYBASE_OCS/xa_config の lrmname LRM 定義に関連するデータベースが使用された AIX and Linux での Sybase について、サンプル XAResourceManager の入力例は以下のようです。 XA 関数呼び出しをログに記録する場合は、ログ・ファイル名を組み込みます。

```
XAResourceManager:
  Name=mysybase
  SwitchFile=sybswit
  XAOpenString=-Uuser -Ppassword -Nlrmname -L/tmp/sybase.log -Txa
  ThreadOfControl=THREAD
```

Sybase でのマルチスレッド・プログラムの使用

Sybase への更新を組み込む IBM MQ グローバル作業単位でマルチスレッド化プログラムを使用する場合は、ThreadOfControl パラメーターに値 THREAD を使用する必要があります。 さらに、プログラム (およびスイッチ・ロード・ファイル) とスレッド・セーフな Sybase ライブラリー (_r バージョン) とをリンクしていることを確認してください。 ThreadOfControl パラメーターでの値 THREAD の使用については、前述の例に示されています。

Multi 複数のデータベースの構成

複数のデータベースに対する更新がグローバル作業単位に含まれるようにキュー・マネージャーを構成するには、データベースごとに XAResourceManager スタンザを追加します。

すべてのデータベースが同じデータベース・マネージャーによって管理される場合、データベースは各スタンザにより個別に定義されます。 各スタンザは同一の *SwitchFile* を指定しますが、*XAOpenString* の内容は異なります。これは、各スタンザにより、更新されるデータベースの名前が指定されるためです。 例えば、 72 ページの図 9 に示すスタンザは、 AIX and Linux システム上の Db2 データベース *MQBankDB* および *MQFeeDB* を使用してキュー・マネージャーを構成します。

重要 : 複数のスタンザで同じデータベースを指定することはできません。 どのような環境においても、このような構成は機能しません。このような構成を試した場合は失敗します。

when the MQ code makes its second xa_open call in any process in this environment, the database software fails the second xa_open with a -5 error, XAER_INVAL という形式のエラーを受け取ります。

```
XAResourceManager:
Name=DB2 MQBankDB
SwitchFile=db2swit
XAOpenString=MQBankDB

XAResourceManager:
Name=DB2 MQFeeDB
SwitchFile=db2swit
XAOpenString=MQFeeDB
```

図 *9.* 複数の *Db2* データベース用サンプル *XAResourceManager* 項目

更新されるデータベースが異なるデータベース・マネージャーによって管理される場合、それぞれに対して XAResourceManager スタンザを追加します。 この場合、各スタンザには異なる *SwitchFile* が指定されます。 例えば、*MQFeeDB* が Db2 ではなく Oracle により管理されている場合、AIX and Linux システムで次のスタンザを使用します。

```
XAResourceManager:
Name=DB2 MQBankDB
SwitchFile=db2swit
XAOpenString=MQBankDB

XAResourceManager:
Name=Oracle MQFeeDB
SwitchFile=oraswit
XAOpenString=Oracle_XA+Acc=P/myuser/mypassword+SesTm=35+LogDir=/tmp/ora.log+DB=MQFeeDB
```

図 *10. Db2* および *Oracle* データベース用サンプル *XAResourceManager* 項目

原則として、1 つのキュー・マネージャーで構成できるデータベース・インスタンスの数に制限はありません。

注 : グローバル作業単位内の複数のデータベース更新に Informix データベースを組み込む場合のサポートについて詳しくは、製品の README ファイルを確認してください。

Multi セキュリティーに関する考慮事項

XA モデルでのデータベースの実行に関する考慮事項。

以下の情報は、参考のためにのみ記載しています。 いかなる場合でも、XA モデルでのデータベースの運用に関するセキュリティーの含意事項については、データベース・マネージャーに添付されている資料を参照してください。

アプリケーション・プロセスは、MQBEGIN によってグローバル作業単位の開始を指示します。 アプリケーションが実行する最初の MQBEGIN 呼び出しでは、各参加データベースのクライアント・ライブラリー・コードが xa_open エントリー・ポイントで呼び出されて、参加データベースにすべて接続されます。 すべてのデータベース・マネージャーが、ユーザー ID およびパスワードを XAOpenString に指定できるメカニズムを提供しています。 認証情報が流れるのは、このときだけです。

AIX and Linux プラットフォームで MQI を呼び出す場合、ファースト・パス・アプリケーションを実行するには、mqm の有効なユーザー ID を使用しなければならないことに注意してください。

Multi XA リソース・マネージャーとの連絡が失われる際の考慮事項

キュー・マネージャーは、データベース・マネージャーが使用できない状況でもその影響をあまり受けません。 これは、データベース・サーバーとは別にキュー・マネージャーを始動および停止できることを意味しています。 連絡が復元されるとき、キュー・マネージャーとデータベースは再同期されます。 rsvmqtrn コマンドを使用して、未確定の作業単位を手動で解決することもできます。

通常の運用では、構成手順の完了後には、最小限の管理作業のみが必要とされます。 キュー・マネージャーは、データベース・マネージャーが使用できない状況でもその影響を受けることが少ないため、管理ジョブは単純なものになっています。 具体的には、次のような特徴があります。

- キュー・マネージャーを始動する際に、事前に各データベース・マネージャーを始動する必要はありません。
- いずれかのデータベース・マネージャーが使用できなくなった場合に、キュー・マネージャーを停止して再始動する必要はありません。

これにより、データベース・サーバーとは独立してキュー・マネージャーを始動および停止できます。

キュー・マネージャーとデータベース間の連絡が失われた場合は、両方が使用可能になった後に、再同期を取る必要があります。 再同期とは、該当するデータベースが関連する未確定の作業単位を完了するプロセスです。 通常、再同期は自動的に実行されるため、ユーザーが介入する必要はありません。 キュー・マネージャーはデータベースに対して、未確定の作業単位のリストを要求します。 キュー・マネージャーは次に、未確定の作業単位をそれぞれコミットまたはロールバックするようにデータベースに指示します。

キュー・マネージャーが始動すると、各データベースと再同期を取ります。 あるデータベースが使用できない状態になった場合、このデータベースが使用可能な状態に戻ったとキュー・マネージャーが次に認識した時点で、キュー・マネージャーが再同期を取る必要があるのはこのデータベースのみです。

MQBEGIN を使用して新しいグローバル作業単位が開始されると、キュー・マネージャーは、以前に使用できない状態にあったデータベースとの連絡を自動的に回復します。 これは、データベース・クライアント・ライブラリー内の xa_open 関数を呼び出して行われます。 この xa_open の呼び出しが失敗した場合、MQBEGIN は、完了コード MQCC_WARNING と理由コード MQRC_PARTICIPANT_NOT_AVAILABLE を戻します。 MQBEGIN 呼び出しは、後で再試行できます。

データベースへの更新に関与するグローバル作業単位で、MQBEGIN 中に失敗したことが示される場合、この作業単位の試行を続行しないでください。 そのデータベースに接続して更新を行うことはできません。 行える操作は、プログラムを終了するか、データベースが再度使用可能な状態になることを期待して MQBEGIN を定期的に再試行することのみです。

別の方法として、rsvmqtrn コマンドを使用して、未確定の作業単位をすべて明示的に解決することもできます。

Multi 未確定の作業単位

キュー・マネージャーからデータベース・マネージャーに対して準備の指示が出された後にキュー・マネージャーとの接続が失われた場合、データベースに未確定の作業単位が発生することがあります。 データ

ベース・サーバーは、キュー・マネージャーから実行結果 (コミットまたはロールバック) を受け取るまでは、更新に関連するデータベース・ロックを保持する必要があります。

このロックにより、他のアプリケーションがデータベース・レコードの更新または読み取りを行えなくなるため、できるだけ早く再同期を実行する必要があります。

何らかの理由により、キュー・マネージャーによるデータベースの自動再同期よりも早い時点で再同期を行わなければならない場合には、データベース・マネージャーに付属の機能を使用してデータベースの更新を手動でコミットまたはロールバックすることもできます。 「*X/Open Distributed Transaction Processing: The XA Specification*」では、これをヒューリスティック 判定の実行と呼んでいます。 データ保全性を損なう可能性があるため、これはできるだけ使用しないでください。例えば、その他のすべての参加プログラムがそれぞれの更新をコミット済みであるときに、誤ってデータベースの更新をロールバックしてしまう可能性があります。

キュー・マネージャーを再始動するか、データベースが既に再始動している場合には `rsvmqtrn` コマンドを使用して、自動再同期を開始するほうがはるかに良い方法です。

Multi *dspmqtrn* コマンドを使用した未解決の作業単位の表示

-i パラメーターを指定した **dspmqtrn** コマンドを使用して、内部で発生した未確定トランザクションを表示できます。

データベース・マネージャーが使用不可になっている場合には、**dspmqtrn** コマンドを使用して、該当するデータベースが関与する未解決のグローバル作業単位の状態を確認できます。

dspmqtrn コマンドを実行すると、1 つ以上の参加プログラムが未確定の状態になっている作業単位のみが表示されます。 参加プログラムは、準備済みの更新をコミットするかロールバックするかについてキュー・マネージャーからの決定を待機しています。

これらのグローバル作業単位ごとに、それぞれの参加プログラムの状態が **dspmqtrn** からの出力に表示されます。 特定のリソース・マネージャーのリソースを更新しなかった作業単位は表示されません。

未確定の作業単位に関して、リソース・マネージャーは、次のいずれかが済んだ状態と見なされます。

準備済み
リソース・マネージャーでは更新をコミットする準備ができています。

コミット済み
リソース・マネージャーによる更新のコミットが既に実行されています。

ロールバック済み
リソース・マネージャーによる更新のロールバックが既に実行されています。

参加済み
リソース・マネージャーは参加プログラムですが、更新の準備、コミット、およびロールバックが済んでいません。

キュー・マネージャーは、再始動時に、XAResourceManager スタンザを持つ各データベースに、その未確定のグローバル作業単位のリストを要求します。 データベースが再開されていない場合、または使用できない状態の場合、キュー・マネージャーは、それらの作業単位の最終結果をデータベースにまだ送信できません。 未確定の作業単位の結果は、その後初めてデータベースが再度使用できる状態になったときに、データベースに送信されます。

この場合、データベース・マネージャーは、再同期が行われるまで、準備済み状態にあるとして報告されます。

dspmqtrn コマンドにより未確定の作業単位が表示される場合には常に、参加している可能性のあるリソース・マネージャーすべてのリストが最初に示されます。 これらのリソース・マネージャーには固有 ID である RMId が割り振られています。この ID は、未確定の作業単位に関してリソース・マネージャーの状態が報告される場合に、リソース・マネージャーの名前 の代わりに使用されます。

dspmqtrn による出力の例に、次のコマンドの発行結果を示します。

```
dspmqtrn -m MY_QMGR
```

```
AMQ7107: Resource manager 0 is MQSeries.
AMQ7107: Resource manager 1 is DB2 MQBankDB.
AMQ7107: Resource manager 2 is DB2 MQFeeDB.

AMQ7056: Transaction number 0,1.
    XID: formatID 5067085, gtrid_length 12, bqual_length 4
         gtrid [3291A50600002013746573 74]
         bqual [00000001]
AMQ7105: Resource manager 0 has committed.
AMQ7104: Resource manager 1 has prepared.
AMQ7104: Resource manager 2 has prepared.
```

Transaction number は、**rsvmqtrn** コマンドで使用できるトランザクションの ID です。 詳しくは、AMQ7xxx: IBM MQ 製品のメッセージを参照してください。 *XID* 変数は、*X/Open XA* 仕様の一部です。 この仕様の最新情報については、https://publications.opengroup.org/c193 を参照してください。

図 *11. dspmqtrn* による出力の例

dspmqtrn による出力の例の出力は、キュー・マネージャーに 3 つのリソース・マネージャーが関連付けられていることを示しています。 最初のリソース・マネージャー 0 は、キュー・マネージャー自身です。 その他の 2 つのリソース・マネージャー・インスタンスは、MQBankDB および MQFeeDB Db2 データベースです。

この例では、未確定の作業単位は 1 つのみです。 メッセージは、3 つのリソース・マネージャーすべてに対して出されます。これは、作業単位内のキュー・マネージャーおよび両方の Db2 データベースに対して更新が行われたことを意味します。

キュー・マネージャー、つまりリソース・マネージャー 0 に対して行われた更新は、コミット済み の状態です。 Db2 データベースに対して行われた更新は、準備済み の状態です。これは、Db2 が MQBankDB データベースおよび MQFeeDB データベースに対する更新をコミットするために呼び出される前に、DB2 が使用不可の状態になっていたはずであることを意味します。

未確定の作業単位には、XID (トランザクション *ID*) という外部 ID が付いています。 これは、グローバル作業単位における未確定部分が占める部分を識別するためにキュー・マネージャーが Db2 に渡すデータです。

詳しくは、 **dspmqtrn** を参照してください。

Multi *rsvmqtrn* コマンドを使用した未解決の作業単位の解決

未解決の作業単位は、キュー・マネージャーと Db2 が再同期したときに完了します。

75 ページの図 11 に示す出力には、両方の Db2 データベースに対してコミットの決定がまだ送信されていない未確定の作業単位が 1 つあることが示されています。

この作業単位を完了するには、Db2 が次に使用可能な状態になった時点で、キュー・マネージャーと Db2 を再同期する必要があります。 キュー・マネージャーは、新しい作業単位の開始を利用して、Db2 との連絡を再度獲得します。 または、**rsvmqtrn** コマンドを使用して明示的に再同期を実行するよう、キュー・マネージャーに指示することもできます。

未確定の作業単位に関連するすべてのデータベース・ロックをできるだけ早く解放するために、Db2 が再始動した直後にこの指示を行ってください。 キュー・マネージャーに対してすべての未確定の作業単位を解決するよう指示する -a オプションを使用します。 次の例では、Db2 が再始動しているため、キュー・マネージャーが未確定の作業単位を解決できます。

```
> rsvmqtrn -m MY_QMGR -a
Any in-doubt transactions have been resolved.
```

Multi 実行結果の混在とエラー

キュー・マネージャーは 2 フェーズ・コミット・プロトコルを使用しますが、これにより作業単位の実行結果が混在する可能性が完全に排除されるわけではありません。 参加プログラムのうち、一部が更新をコミットし、一部が更新をバックアウトした場合に、このような混在が発生します。

完了した作業単位の実行結果が混在していると、深刻な影響が出ます。1 つの作業単位として更新すべきだったのに更新されなかった共用リソースは、整合性のない状態になってしまうためです。

実行結果の混在は、キュー・マネージャーによって未確定の作業単位を解決せずに、ヒューリスティック判定に基づいて作業単位の処理を行った場合に主に発生します。 この判定は、キュー・マネージャーの制御を受けません。

キュー・マネージャーは実行結果の混在を検出するたびに、FFST 情報を作成し、エラー・ログに障害を記録して、次の 2 つのメッセージのいずれかを表示します。

- データベース・マネージャーがコミットの代わりにロールバックを行った場合には次のメッセージが表示されます。

```
AMQ7606 A transaction has been committed but one or more resource
        managers have rolled back.
```

- データベース・マネージャーがロールバックの代わりにコミットを行った場合には次のメッセージが表示されます。

```
AMQ7607 A transaction has been rolled back but one or more resource
        managers have committed.
```

さらに、ヒューリスティック判定によって障害が発生したデータベースを識別するメッセージが表示されます。 この場合、ユーザーは障害のあるデータベースとの整合性をローカルで復元する必要があります。この手順は複雑です。まず、誤ってコミットまたはロールバックした更新を分離し、次にデータベースの変更を手動で取り消すか、再実行しなければなりません。

Multi 構成情報の変更

キュー・マネージャーが正常に始動し、グローバル作業単位を調整する準備ができた後は、リソース・マネージャーの構成情報を変更しないでください。

構成情報は、変更する必要があればいつでも変更できますが、変更内容が有効になるのはキュー・マネージャーの再始動後です。

データベースに対するリソース・マネージャー構成情報を削除すると、実際にはキュー・マネージャーがデータベース・マネージャーと連絡をとる機能が除去されることになります。

決して リソース・マネージャー構成情報の *Name* 属性を変更しないでください。 この属性により、キュー・マネージャーに対してデータベース・マネージャーのインスタンスが一意的に識別されます。 この固有の ID が変更されると、キュー・マネージャーは、データベースが削除され、まったく新しいインスタンスが追加されているものと想定します。 その場合でも、キュー・マネージャーは未解決の作業単位をその古い *Name* に関連付けており、データベースを未確定の状態にする可能性があります。

Multi データベース・マネージャーのインスタンスの削除

構成からデータベースを完全に削除する必要がある場合は、キュー・マネージャーを再始動する前に、データベースが未確定状態でないことを確認してください。

データベース製品には、未確定のトランザクションをリストするためのコマンドが用意されています。 未確定のトランザクションがある場合、まず、キュー・マネージャーとデータベースを再同期させます。 これを行うには、キュー・マネージャーを始動します。 **rsvmqtrn** コマンドか、未確定の作業単位を表示するためのデータベース固有のコマンドを使用して、再同期が行われたかどうかを確認できます。 再同期が行われたことを確認できたら、キュー・マネージャーを終了し、データベースの構成情報を削除します。

この手順に従わない場合、キュー・マネージャーはそのデータベースに関与するすべての未確定の作業単位を記憶したままになります。 キュー・マネージャーの再始動時に毎回警告メッセージ AMQ7623 が出されます。 今後、キュー・マネージャーを使ってこのデータベースの構成を再び行うことがない場合は、**rsvmqtrn** コマンドの -r オプションを使用して、未確定トランザクションへのこのデータベースの関与に関する記憶を消去するようキュー・マネージャーに指示します。 キュー・マネージャーは、すべての参加プログラムで未確定トランザクションが完了した場合にのみ、これらのトランザクションに関する記憶を消去します。

いくつかのリソース・マネージャー構成情報を一時的に削除する必要がある場合があります。 AIX and Linux システムでは、後で簡単に復元できるように、スタンザをコメント化して構成情報を一時的に削除するのが最も良い方法です。 キュー・マネージャーが特定のデータベースまたはデータベース・マネージャーと連絡を取るたびにエラーが発生する場合、この処置を行うことをお勧めします。 対象のリソース・マネージャー構成情報を一時的に削除すると、キュー・マネージャーはその他すべての参加プログラムが関与するグローバル作業単位を開始できます。 XAResourceManager スタンザのコメント化の例を以下に示します。

```
# This database has been temporarily removed
#XAResourceManager:
#   Name=mydb2
#   SwitchFile=db2swit
#   XAOpenString=mydbname,myuser,mypassword,toc=t
#   ThreadOfControl=THREAD
```

図 *12. AIX and Linux* システムでの *XAResourceManager* スタンザのコメント化

Windows システムでは、IBM MQ Explorer を使用して、データベース・マネージャー・インスタンスに関する情報を削除します。 *Name* フィールドを復元する際に、正しい名前を入力するよう、十分注意してください。 間違った名前を入力した場合、76 ページの『構成情報の変更』に記載されているとおり、未確定問題が発生することがあります。

Multi XA 動的登録

XA 仕様には、トランザクション・マネージャーがリソース・マネージャーに対して実行する xa_* 呼び出しの回数を減らす仕組みがあります。 この最適化の仕組みを動的登録 といいます。

動的登録は Db2 でサポートされています。 他のデータベースでもサポートしているものがあります。詳しくは、ご使用のデータベース製品の資料を参照してください。

動的登録最適化はなぜ有用なのでしょうか。 アプリケーションには、データベース・テーブルの更新が含まれているグローバル作業単位もあれば、そのような更新が含まれていないグローバル作業単位もあります。 データベースのテーブルに対して永続的な更新が行われていない場合、MQCMIT 中に発生するコミット・プロトコルにそのデータベースを組み込む必要はありません。

データベースが動的登録をサポートしているかどうかに関係なく、アプリケーションは IBM MQ 接続での最初の MQBEGIN 呼び出し時に xa_open を呼び出します。 アプリケーションは、後続の MQDISC 呼び出しで xa_close も呼び出します。 後続の XA 呼び出しのパターンは、データベースが動的登録をサポートしているかどうかによって異なります。

データベースが動的登録をサポートしない場合
作業単位内のそのデータベースのテーブルに対して永続的な更新を行ったかどうかにかかわらず、各グローバル作業単位には、IBM MQ コードによって実行されてデータベース・クライアント・ライブラリーに入れられたいくつかの XA 関数呼び出しが含まれます。 以下が含まれます。

- アプリケーション・プロセスからの xa_start および xa_end。 これらは、グローバル作業単位の始めと終わりを宣言するために使用されます。
- キュー・マネージャー・エージェント・プロセス amqzlaa0 からの xa_prepare、xa_commit、および xa_rollback。 これらは、グローバル作業単位の結果、つまりコミットまたはロールバックの決定を送信するために使用されます。

さらに、キュー・マネージャー・エージェント・プロセスは、最初の MQBEGIN 中に xa_open も呼び出します。

データベースが動的登録をサポートする場合
IBM MQ コードは、必要な XA 関数呼び出しのみを実行します。 データベース・リソースに対する永続的更新が**伴わない**グローバル作業単位の場合、データベースへの XA 呼び出しは**ありません**。 そのような永続的更新が**伴う**グローバル作業単位の場合、次の呼び出しが行われます。

- アプリケーション・プロセスからの、グローバル作業単位の終わりを宣言する xa_end。

- キュー・マネージャー・エージェント・プロセス amqzlaa0 からの xa_prepare、xa_commit、および xa_rollback。 これらは、グローバル作業単位の結果、つまりコミットまたはロールバックの決定を送信するために使用されます。

動的登録が機能するためには、データベースが現行のグローバル作業単位に含める永続的更新を実行したことを IBM MQ に通知する仕組みがデータベースに用意されていることが不可欠です。 IBM MQ には、この目的のために ax_reg 関数が用意されています。

アプリケーション・プロセスで実行するデータベースのクライアント・コードは、ax_reg 関数を検出してそれを呼び出し、現行のグローバル作業単位内で永続的な作業を実行したという事実を動的に登録 します。 この ax_reg 呼び出しに対する応答として、IBM MQ は、データベースが参加したことを記録します。 これがこの IBM MQ 接続での最初の ax_reg 呼び出しである場合、キュー・マネージャー・エージェント・プロセスは xa_open を呼び出します。

データベース・クライアント・コードは、プロセスで実行中に (例えば SQL UPDATE 呼び出し中や、データベースのクライアント API が関与するあらゆる呼び出し中に)、この ax_reg 呼び出しを実行します。

Multi エラー状態

XA 動的登録では、分かりにくい障害がキュー・マネージャーで発生する可能性があります。

一般的な例としては、キュー・マネージャーを始動する前にデータベース環境変数を正しく設定することを忘れたために、キュー・マネージャーの xa_open に対する呼び出しが失敗する場合があります。 どのグローバル作業単位も使用できません。

このようなことを回避するために、キュー・マネージャーを始動する前に関連する環境変数を設定してあることを確認してください。 ご使用のデータベース製品の資料、および 61 ページの『Db2 の構成』、64 ページの『Oracle の構成』、および 69 ページの『Sybase 構成』 に記載されているアドバイスを確認してください。

すべてのデータベース製品において、キュー・マネージャーは、リカバリー・セッションの一部として、キュー・マネージャーの始動時に xa_open を一度呼び出します (73 ページの『XA リソース・マネージャーとの連絡が失われる際の考慮事項』 で説明しています)。 データベース環境変数を間違って設定した場合、この xa_open 呼び出しは失敗しますが、これはキュー・マネージャーが始動できない原因ではありません。 キュー・マネージャーが始動できないのは、データベース・サーバーが使用できない状態であることを示すのにデータベース・クライアント・ライブラリーが同じ xa_open エラー・コードを使用するためです。 キュー・マネージャーは、該当のデータベースが関与するグローバル作業単位の外側でデータ処理の続行を開始できるはずなので、IBM MQ では、これを重大なエラーとしては扱いません。

xa_open への後続の呼び出しは、IBM MQ 接続での最初の MQBEGIN 中 (動的登録が使用されていない場合) またはデータベース・クライアント・コードによる IBM MQ 提供の ax_reg 関数への呼び出し中 (動的登録が使用されている場合) に、キュー・マネージャーから実行されます。

エラー条件の ***タイミング*** (場合によっては FFST レポート) は、動的登録を使用しているかどうかによって異なります。

- 動的登録を使用している場合、MQBEGIN 呼び出しは正常に実行されますが、SQL UPDATE (または類似の) データベース呼び出しは失敗します。
- 動的登録を使用していない場合、MQBEGIN 呼び出しは失敗します。

アプリケーション・プロセスおよびキュー・マネージャー・プロセスに環境変数が正しく設定されていることを確認してください。

Multi *XA* 呼び出しの要約

ここでは、グローバル作業単位を制御するさまざまな MQI 呼び出しの結果として、データベース・クライアント・ライブラリー内の XA 関数に対して行われる呼び出しのリストを示します。 ここでは、XA 仕様で記述されるプロトコルの完全な説明ではなく、概要を示します。

xa_start 呼び出し、および xa_end 呼び出しは、必ずアプリケーション・プロセス内で IBM MQ コードによって行われるのに対し、xa_prepare、xa_commit、および xa_rollback は、必ずキュー・マネージャー・エージェント・プロセス amqzlaa0 から呼び出されます。

この表に示されている xa_open 呼び出しと xa_close 呼び出しはすべて、アプリケーション・プロセスから実行されます。 キュー・マネージャー・エージェント・プロセスは、78 ページの『エラー状態』で説明されている状況で、xa_open を呼び出します。

表 *4. XA* 関数呼び出しの要約

MQI 呼び出し	**動的登録付きで行われる XA 呼び出し**	**動的登録なしで行われる XA 呼び出し**
最初の MQBEGIN	`xa_open`	`xa_open` `xa_start`
後続の MQBEGIN	XA 呼び出しなし	`xa_start`

MQCMIT (ax_reg は現行のグローバル作業単位中に**呼び出されない**)	XA 呼び出しなし	`xa_end`
		`xa_prepare`
		`xa_commit`
		`xa_rollback`

MQCMIT (ax_reg は現行のグローバル作業単位中に**呼び出される**)	`xa_end`
	`xa_prepare`
	`xa_commit`
	`xa_rollback`
		適用外 非動的モードでは ax_reg への呼び出しは行われません。

MQBACK (ax_reg は現行のグローバル作業単位中に**呼び出されない**)	XA 呼び出しなし	`xa_end` `xa_rollback`
MQBACK (ax_reg は現行のグローバル作業単位中に**呼び出される**)	`xa_end` `xa_rollback`	適用外 非動的モードでは ax_reg への呼び出しは行われません。
MQDISC。MQCMIT または MQBACK が最初に呼び出されたもの。 それらが呼び出されなかった場合、MQCMIT 処理は、MQDISC 中に最初に行われます。	`xa_close`	`xa_close`

注 :

1. MQCMIT では、xa_prepare が正常に実行された場合に xa_commit が呼び出されます。 そうでない場合は、xa_rollback が呼び出されます。

シナリオ 2: 他のソフトウェアが調整を行う

シナリオ 2 では、外部トランザクション・マネージャーがグローバル作業単位を調整し、トランザクション・マネージャーの API の制御を受けてそれらを開始してコミットします。 MQBEGIN、MQCMIT、および MQBACK の各 verb は使用できません。

ここでは、次の事項を含め、このシナリオについて説明します。

- 80 ページの『外部同期点調整』
- 82 ページの『CICS の使用』
- 86 ページの『Microsoft Transaction Server (COM+) の使用』

外部同期点調整

グローバル作業単位は、外部 X/Open XA 準拠のトランザクション・マネージャーによって調整することもできます。 この場合、IBM MQ キュー・マネージャーは作業単位に関与していますが、作業単位の調整は行いません。

外部トランザクション・マネージャーによって調整されるグローバル作業単位の制御のフローは次のとおりです。

1. トランザクションを開始することをアプリケーションが外部同期点調整プログラム (TXSeries など) に通知します。
2. 同期点調整プログラムが IBM MQ などの既知のリソース・マネージャーに現行トランザクションを通知します。
3. アプリケーションは、現行トランザクションに関連付けられたリソース・マネージャーに対して呼び出しを実行します。 例えば、アプリケーションは MQGET 呼び出しを IBM MQ に対して実行することがあります。
4. アプリケーションは外部同期点調整プログラムにコミット要求またはバックアウト要求を出します。
5. 同期点調整プログラムは、通常 2 フェーズ・コミット・プロトコルを使用して、適切な呼び出しを各リソース・マネージャーに出し、トランザクションを完了します。

IBM MQ が関与するトランザクションの 2 フェーズ・コミット・プロセスを提供する外部同期点調整プログラムのサポートされるレベルは、IBM MQ のシステム要件で定義されています。

このセクションの後半では、外部作業単位を使用可能にする方法について説明します。

IBM MQ XA スイッチ構造

外部から調整される作業単位に関与する各リソース・マネージャーには、XA スイッチ構造体がなければなりません。 この構造体により、リソース・マネージャーの機能と、同期点調整プログラムによって呼び出される関数の両方が定義されます。

IBM MQ には、この構造体について、次の 2 種類が用意されています。

- *MQRMIXASwitch* (静的 XA リソース管理用)
- *MQRMIXASwitchDynamic* (動的 XA リソース管理用)

ご使用のトランザクション・マネージャーの資料を参照して、静的リソース管理インターフェースまたは動的リソース管理インターフェースのいずれを使用するかを決定してください。 トランザクション・マネージャーでサポートされている場合は必ず、動的 XA リソース管理を使用することをお勧めします。

64 ビットのトランザクション・マネージャーの中には、XA 仕様にある *long* 型を 64 ビットとして扱うものもあれば、32 ビットとして扱うものもあります。 IBM MQ は両方のモデルをサポートします。

- トランザクション・マネージャーが 32 ビットの場合、またはトランザクション・マネージャーが 64 ビットであるが long タイプを 32 ビットとして扱う場合は、81 ページの表 5 にリストされているスイッチ・ロード・ファイルを使用します。
- トランザクション・マネージャーが 64 ビットで、 long タイプを 64 ビットとして扱う場合は、81 ページの表 6 にリストされているスイッチ・ロード・ファイルを使用します。

64 ビットのトランザクション・マネージャーの中には、long 型を 64 ビットとして扱うものもあります。次の 64 ビットのトランザクション・マネージャーには、64 ビットの代替スイッチ・ロード・ファイルが必要であることが確認されています。

- Tuxedo

ご使用のトランザクション・マネージャーが使用するモデルが不明な場合は、トランザクション・マネージャーの資料を参照してください。

表 5. XA スイッチ・ロード・ファイル名		
プラットフォーム	**スイッチ・ロード・ファイル名 (サーバー)**	**スイッチ・ロード・ファイル名 (拡張トランザクション・クライアント)**
Windows Windows	*mqmxa.dll*	*mqcxa.dll*
AIX AIX (スレッドなし)	*libmqmxa.a*	*libmqcxa.a*
AIX AIX (スレッドあり)	*libmqmxa_r.a*	*libmqcxa_r.a*
Linux Linux (スレッドなし)	*libmqmxa.so*	*libmqcxa.so*
Linux Linux (スレッドあり)	*libmqmxa_r.so*	*libmqcxa_r.so*

表 6. 64 ビットの代替 XA スイッチ・ロード・ファイル名		
プラットフォーム	**スイッチ・ロード・ファイル名 (サーバー)**	**スイッチ・ロード・ファイル名 (拡張トランザクション・クライアント)**
AIX AIX (スレッドなし)	*libmqmxa64.a*	*libmqcxa64.a*
AIX AIX (スレッドあり)	*libmqmxa64_r.a*	*libmqcxa64_r.a*
Linux Linux (スレッドなし)	*libmqmxa64.so*	*libmqcxa64.so*
Linux Linux (スレッドあり)	*libmqmxa64_r.so*	*libmqcxa64_r.so*

一部の外部同期点調整プログラム (CICS 以外) では、XA スイッチ構造体の名前フィールドに作業単位に関与する各リソース・マネージャーの名前を指定する必要があります。 IBM MQ リソース・マネージャー名は MQSeries_XA_RMI です。

IBM MQ XA スイッチ構造体が同期点調整プログラムにリンクする方法は、同期点調整プログラムが定義します。 IBM MQ XA スイッチ構造体と CICS とのリンクの詳細については、82 ページの『CICS の使用』に記載されています。 IBM MQ XA スイッチ構造体とその他の XA 準拠同期点調整プログラムとのリンクの詳細については、各プログラムに付属の資料を参照してください。

IBM MQ を XA 準拠同期点調整プログラムと共に使用する場合には必ず、次の事項を考慮してください。

- トランザクション・マネージャー・ライブラリー・コード (アプリケーション・プログラマーが呼び出した API の一部として実行) は、MQCONN を呼び出す前のある時点において **xa_open** を呼び出して IBM MQ を開く必要があります。

xa_open 呼び出しは、MQCONN 呼び出しを実行するスレッドと同じスレッドで実行する必要があります。 この要件の理由は、XA 仕様が、コンテキストを示すためにこのスレッドの使用を要求するためです。

これは、サンプル・プログラム amqstxsx.c で実行されるアプローチであることに注意してください。 このサンプル・プログラムは、 **xa_open** 呼び出しが、トランザクション・マネージャーのライブラリー・コードから、関数 tpopen 内で IBM MQ に対して行われることを想定しています。

MQCONN 呼び出しの前に、同じスレッドで **xa_open** 呼び出しが行われなかった場合、IBM MQ キュー・マネージャー接続は XA コンテキストと関連付けられません。

詳細については、MQCTL を参照してください。

- 同期点調整プログラムによる xa_open 呼び出しに渡される xa_info 構造体には必ず、IBM MQ キュー・マネージャーの名前が含まれています。 この名前の形式は、MQCONN 呼び出しに渡されるキュー・マネージャーの名前と同じです。 xa_open 呼び出しに渡される名前がブランクの場合、デフォルトのキュー・マネージャーが使用されます。

  あるいは、xa_info 構造体に *TPM* パラメーターおよび *AXLIB* パラメーターの値を含める方法もあります。 *TPM* パラメーターは、使用されるトランザクション・マネージャーを指定します。 有効な値は、CICS、TUXEDO、および ENCINA です。 *AXLIB* パラメーターは、トランザクション・マネージャーの ax_reg 関数および ax_unreg 関数を含むライブラリーの名前を指定します。 これらのパラメーターの詳細については、拡張トランザクション・クライアントの構成を参照してください。 xa_info 構造体にこれらのパラメーターのいずれかが含まれる場合、デフォルトのキュー・マネージャーが使用されていない限り、キュー・マネージャー名が *QMNAME* パラメーターで指定されます。
- 外部同期点調整プログラムのインスタンスによって調整されるトランザクションには、一度に 1 つのキュー・マネージャーのみが参加できます。 これにより同期点調整プログラムは、キュー・マネージャーと有効な接続関係を持つことができ、サポートされる接続は一度につき 1 つという規則が適用されます。
- 外部同期点調整プログラムへの呼び出しを含むあらゆるアプリケーションが接続できるキュー・マネージャーは、この外部同期点調整プログラムが管理するトランザクションに参加しているキュー・マネージャーのみです (このようなアプリケーションは既にキュー・マネージャーと有効な接続関係にあるためです)。 ただし、このようなアプリケーションは、MQCONN 呼び出しを実行して接続ハンドルを獲得し、終了前に MQDISC 呼び出しを実行する必要があります。
- リソースの更新が外部同期点調整プログラムによって調整されるキュー・マネージャーは、外部同期点調整プログラムより前に始動する必要があります。 同様に、同期点調整プログラムはキュー・マネージャーより前に終了しなければなりません。
- 外部同期点調整プログラムが異常終了した場合、この同期点調整プログラムを再始動する前にキュー・マネージャーを停止して再始動し、異常終了時にコミットされていなかったすべてのメッセージング操作が適切に解決されるようにします。

CICS の使用

CICS は、TXSeries のエレメントの 1 つです。

XA 準拠 (かつ 2 フェーズ・コミット・プロセスを使用する) TXSeries のバージョンは、IBM MQ のシステム要件の資料で定義されています。

IBM MQ は、他のトランザクション・マネージャーもサポートします。 サポートされるソフトウェアに関する情報へのリンクについては、IBM MQ のシステム要件 Web ページを参照してください。

2 フェーズ・コミット・プロセスの要件

IBM MQ で CICS 2 フェーズ・コミット・プロセスを使用する場合の 2 フェーズ・コミット・プロセスの要件。 これらの要件は、z/OS® には適用されません。

次の要件に注意してください。

- IBM MQ および CICS は同一の物理マシン上になければなりません。
- IBM MQ は、 IBM MQ MQI client 上の CICS をサポートしません。

- CICS を始動しようとする **前に** 、XAD リソース定義スタンザに指定されている名前を使用してキュー・マネージャーを始動する必要があります。 CICS 領域に IBM MQ 用の XAD リソース定義スタンザを追加してある場合には、このようにキュー・マネージャーを始動しないと、CICS を始動できません。
- 単一の CICS 領域から一度にアクセスできる IBM MQ キュー・マネージャーは 1 つのみです。
- CICS トランザクションは、MQCONN 要求を実行しないと IBM MQ リソースにアクセスできません。 MQCONN 呼び出しで名前が指定される IBM MQ キュー・マネージャーは、CICS 領域の XAD リソース定義スタンザの XAOpen 項目に指定されているものでなければなりません。 この項目がブランクである場合、MQCONN 要求ではデフォルトのキュー・マネージャーを指定しなければなりません。
- IBM MQ リソースにアクセスする CICS トランザクションは、 CICS に戻る前に、そのトランザクションから MQDISC 呼び出しを発行する必要があります。 これを行わない場合、キューがオープンの状態で、CICS アプリケーション・サーバーが接続されたままになることがあります。 さらに、タスク終了出口 (86 ページの『サンプル・タスク終了出口』を参照) をインストールしないと、CICS アプリケーション・サーバーが後で (おそらく次のトランザクション中に) 異常終了する可能性があります。
- CICS ユーザー ID (cics) が mqm グループのメンバーであり、CICS コードが IBM MQ を呼び出す権限を持っていることを確認する必要があります。

  CICS 環境で実行されるトランザクションの場合、キュー・マネージャーは、次のような許可方法およびコンテキストの決定方法を採用しています。

  – キュー・マネージャーは、CICS がトランザクションを実行するユーザー ID を照会します。 このユーザー ID はオブジェクト権限マネージャーにより検査されたもので、コンテキスト情報に使用されます。
  – メッセージ・コンテキストでは、アプリケーション・タイプは MQAT_CICS です。
  – コンテキスト内のアプリケーション名は、CICS トランザクション名からコピーされます。

ALW 一般 *XA* サポート

CICS と IBM MQ for AIX, Linux, and Windows システムのリンクを可能にする XA スイッチ・ロード・モジュールが提供されています。 また、用意されているサンプル・ソース・コード・ファイルを使用して、他のトランザクション・メッセージ用の XA スイッチを作成できます。 IBM i General XA は IBM i でサポートされません。

用意されているスイッチ・ロード・モジュールの名前は以下のとおりです。

表 *7. CICS* アプリケーションの必須コード*: XA* 初期化ルーチン

C (ソース)	**C (exec) - 以下のいずれかを追加してください。** **XAD.Stanza**
amqzscix.c	AIX amqzsc- TXSeries for AIX 5.1
amqzscin.c	Windows mqmc4swi - TXSeries for Windows 5.1

Multi *TXSeries for Multiplatforms* で使用するためのライブラリーの作成

TXSeries for Multiplatforms で使用するためのライブラリーを作成するときには、以下の情報を使用してください。

事前作成スイッチ・ロード・ファイル は、XA プロトコルを使用する 2 フェーズ・コミット・トランザクションを必要とする CICS プログラムで使用できる共有ライブラリー (Windows システムでは *DLL* と呼ばれます) です。 この事前作成ライブラリーの名前は、CICS アプリケーションの必須コード: XA 初期化ルーチンの表に記載されています。 また、サンプル・ソース・コードは、以下のディレクトリーにあります。

表 8. AIX, Linux, and Windows オペレーティング・システムでのインストール・ディレクトリー		
プラットフォーム	**ディレクトリー**	**ソース・ファイル**
Linux AIX AIX and Linux	*MQ_INSTALLATION_PATH*/サンプ/	amqzscix.c
Windows Windows	*MQ_INSTALLATION_PATH*¥ Tools¥c¥Samples	amqzscin.c

ここで *MQ_INSTALLATION_PATH* は、IBM MQ をインストールしたディレクトリーです。

注 : スイッチ・ロード・ファイルを使用する IBM MQ のバージョンに対してビルドされた CICS スイッチ・ロード・ファイルを使用する必要があります。

サンプル・ソースからスイッチ・ロード・ファイルを作成するには、ご使用のオペレーティング・システムに該当する指示に従ってください。

AIX **AIX**

以下のコマンドを発行します。

```
export MQM_HOME=/usr/mqm
echo "amqzscix" > tmp.exp
xlc_r $MQM_HOME/samp/amqzscix.c -I/usr/lpp/cics/include -I$MQM_HOME/inc -e amqzscix -bE:tmp.exp -bM:SRE
    -o amqzsc /usr/lpp/cics/lib/regxa_swxa.o -L$MQM_HOME/lib -L/usr/lpp/cics/lib -lcicsrt -lEncina
    -lEncServer -lpthreads -lsarpc -lmqmcics_r -lmqmxa_r -lmqzi_r -lmqmcs_r
rm tmp.exp
```

Linux **Linux プラットフォーム**

以下のコマンドを発行します。

```
gcc -m32 -shared -fPIC -o amqzscix amqzscix.c
    \-IMQ_INSTALLATION_PATH/inc -I CICS_INSTALLATION_PATH/include
    \-LMQ_INSTALLATION_PATH/lib -Wl,-rpath=MQ_INSTALLATION_PATH/lib
    \-Wl,-rpath=/usr/lib -Wl,-rpath-link,/usr/lib -Wl,--no-undefined
    -Wl,--allow-shlib-undefined \-L CICS_LIB_PATH/regxa_swxa.o \-lpthread -ldl -lc
    -shared -lmqzi_r -lmqmxa_r -lmqmcics_r -ldl -lc
```

Windows **Windows**

次のステップを行います。

1. cl コマンドを使用し、少なくとも以下の変数でコンパイルして、amqzscin.obj をビルドします。

   ```
   cl.exe -c -I EncinaPath\include -I MQ_INSTALLATION_PATH\include -Gz -LD amqzscin.c
   ```

2. 以下のラインを含む `mqmc1415.def` というのモジュール定義ファイルを作成します。

   ```
   LIBRARY MQMC4SWI
   EXPORTS
   CICS_XA_Init
   ```

3. **`lib`** コマンドを使用し、少なくとも以下のオプションを使用してエクスポート・ファイルとインポート・ライブラリーを作成します。

   ```
   lib -def:mqmc4swi.def -out:mqmc4swi.lib
   ```

   lib コマンドが正常に実行された場合に `mqmc4swi.exp` ファイルも作成されます。

4. リンク・コマンドを使用して `mqmc4swi.dll` を作成するには、少なくとも以下のオプションを使用してください。

   ```
   link.exe -dll -nod -out:mqmc4swi.dll
       amqzscin.obj CicsPath\lib\regxa_swxa.obj
   ```

```
mqmc4swi.exp mqmcics4.lib
CicsPath\lib\libcicsrt.lib
DcePath\lib\libdce.lib DcePath\lib\pthreads.lib
EncinaPath\lib\libEncina.lib
EncinaPath\lib\libEncServer.lib
msvcrt.lib kernel32.lib
```

IBM MQ XA のサポートおよび *Tuxedo*
IBM MQ (AIX, Linux, and Windows システム用) では、Tuxedo によって調整された XA アプリケーションが xa_start で無限にブロックされる場合があります。

これが発生するのは、単一のグローバル・トランザクションの中で Tuxedo によって調整される複数のプロセスが、同じトランザクション・ブランチ ID (XID) を使用して IBM MQ にアクセスしようとする場合のみです。 グローバル・トランザクションの各プロセスに対して、IBM MQ で使用するための XID としてそれぞれ異なるものを Tuxedo が割り当てる場合は、この問題が発生することはありません。

この問題を回避するには、単一のグローバル・トランザクション ID (gtrid) のもとで IBM MQ にアクセスする Tuxedo 内の各アプリケーションを、独自の Tuxedo サーバー・グループ内で構成するようにしてください。 同じサーバー・グループに属するプロセスは、リソース・マネージャーにアクセスする際に単一の gtrid の代わりに同じ XID を使用するため、IBM MQ において xa_start でブロッキングの影響を受けることになります。 異なるサーバー・グループに属するプロセスは、別々の XID を使用してリソース・マネージャーにアクセスするので、IBM MQ でそのトランザクション処理を直列化する必要はありません。

CICS 2 フェーズ・コミット・プロセスの使用可能化

CICS で 2 フェーズ・コミット・プロセスを使用して MQI 呼び出しが関与するトランザクションを調整できるようにするには、CICS XAD リソース定義スタンザ・エントリーを CICS 領域に追加します。 このトピックは z/OS に適用されないことに注意してください。

次に IBM MQ for Windows の XAD スタンザ・エントリーの追加例を示します。 ここで、*Drive* は IBM MQ がインストールされるドライブ (例えば、D:) です。

```
cicsadd -cxad -rcics_region \
     ResourceDescription="MQM XA Product Description" \
     SwitchLoadFile="Drive:\Program Files\IBM\IBM MQ\bin\mqmc4swi.dll" \
     XAOpen=queue_manager_name
```

拡張トランザクション・クライアントでは、スイッチ・ロード・ファイル mqcc4swi.dll を使用します。

以下は、IBM MQ for AIX or Linux システムの XAD スタンザ・エントリーを追加する例です。
MQ_INSTALLATION_PATH は、IBM MQ がインストールされている上位ディレクトリーを表します。

```
cicsadd -cxad -rcics_region \
      ResourceDescription="MQM XA Product Description" \
      SwitchLoadFile="MQ_INSTALLATION_PATH/lib/amqzsc" \
      XAOpen=queue_manager_name
```

拡張トランザクション・クライアントには、スイッチ・ロード・ファイル amqczsc を使用します。

cicsadd コマンドの使用法については、 CICS の資料を参照してください。

IBM MQ の呼び出しを CICS トランザクションに組み込むことができます。この場合、IBM MQ リソースのコミットおよびロールバックは CICS の指示により実行されます。 このサポートはクライアント・アプリケーションでは使用できません。

IBM MQ リソースにアクセスするには、 CICS トランザクションから MQCONN を発行し、その後に出口で対応する MQDISC を発行する必要があります。

CICS ユーザー出口の使用可能化

CICS ユーザー出口ポイント (通常、ユーザー出口 と呼ばれる) とは、CICS モジュール内で CICS が制御をユーザー作成プログラム (ユーザー出口プログラム) に渡し、ユーザー出口プログラムが処理を完了すると CICS に制御が戻るポイントです。

CICS ユーザー出口を使用する前に、ご使用のプラットフォーム用の「*CICS* 管理ガイド」を参照してください。

サンプル・タスク終了出口
IBM MQ では、CICS タスク終了出口のサンプル・ソース・コードが提供されています。

サンプル・ソース・コードは、以下のディレクトリーにあります。

表 *9. CICS* タスク終了出口		
プラットフォーム	**ディレクトリー**	**ソース・ファイル**
AIX and Linux システム	*MQ_INSTALLATION_PATH*/samp	amqzscgx.c
Windows	*MQ_INSTALLATION_PATH*¥Tools¥c¥Samples	amqzscgn.c

MQ_INSTALLATION_PATH は、IBM MQ がインストールされている上位ディレクトリーを表します。

各ソース・ファイルのコメント内に、サンプル・タスク終了出口の作成に関する説明が記載されています。

タスクの通常終了時および異常終了時 (同期点がとられた後) に、CICS からこの出口が呼び出されます。 出口プログラムでは、リカバリー可能作業は実行できません。

これらの関数を使用するのは、IBM MQ においてと、CICS バージョンが XA インターフェースをサポートしている CICS コンテキストにおいてのみです。 CICS は、これらのライブラリーを `programs` または `user exits` と呼びます。

CICS にはいくつかのユーザー出口があり、amqzscgx (使用されている場合) は `Task termination user exit` (`UE014015`)(出口番号 15) として CICS に定義され、使用可能になっています。

タスク終了出口が CICS から呼び出された時点で、そのタスクの終了状態が CICS から IBM MQ に既に通知されており、IBM MQ が適切な操作 (コミットまたはロールバック) を実行しています。 出口が実行する内容は、クリーンアップのための MQDISC の発行のみです。

タスク終了出口を使用するために CICS システムをインストールして構成する目的の 1 つは、障害のあるアプリケーション・コードの何らかの影響を受けないようにシステムを保護することにあります。 例えば、CICS トランザクションが最初に MQDISC を呼び出さずに異常終了した場合に、タスク終了出口がインストールされていないと、続けて (約 10 秒以内に) CICS 領域のリカバリー不能な障害が発生することがあります。 その原因は、cicsas プロセスで稼働する IBM MQ の正常性スレッドが、ポストされずにしかもクリーンアップおよび戻りのための時間も与えられなくなることにあります。 その徴候としては、FFST レポートを /var/mqm/errors (Windows ではこれに相当する場所) に書き込んだうえで cicsas プロセスが即時に終了することがあります。

Microsoft Transaction Server (COM+) の使用

COM+ (Microsoft Transaction Server) は、一般的な中間層サーバーでのビジネス・ロジック・アプリケーションの実行を支援するように設計されています。

重要な情報については、Windows 上のプライマリー・インストールでのみ使用できる機能 を参照してください。

COM+ は作業を アクティビティー に分割します。通常、口座 *A* から口座 *B* への資金の振替などの、ビジネス・ロジックに基づく短い独立したチャンクです。 COM+ はオブジェクト方向、特に COM に大きく依存します。緩やかに COM + のアクティビティーは COM (ビジネス) オブジェクトによって表されます。

COM+ は、オペレーティング・システムの統合パーツの 1 つです。

COM+ は以下 3 つのサービスをビジネス・オブジェクト管理者に提供し、ビジネス・オブジェクト・プログラマーのかなりの負荷を軽減します。

- トランザクション管理
- セキュリティー
- リソース・プール

一般に COM+ は、COM+ 内に保持されているオブジェクトに対する COM クライアントであるフロントエンド・コードと、データベースなどのバックエンド・サービスと共に使用され、IBM MQ によって COM+ ビジネス・オブジェクトとバックエンドの間がブリッジングされます。

フロントエンド・コードは、スタンドアロン・プログラムか、Microsoft Internet Information Server (IIS) をホストとするアクティブ・サーバー・ページ (ASP) にすることができます。 フロントエンド・コードは、COM の接続を通して、COM+ およびビジネス・オブジェクトと同じコンピューターに置くことができます。また、フロントエンド・コードは、DCOM の接続を通して、異なるコンピューターに置くことができます。状況別に異なるクライアントを使用して、同じ COM+ ビジネス・オブジェクトにアクセスすることもできます。

バックエンド・コードは、COM+ およびビジネス・オブジェクトと同じコンピューターに置くことも、IBM MQ でサポートされている任意のプロトコルの接続を通して異なるコンピューターに置くこともできます。

グローバル作業単位の有効期限切れ

事前に構成された非アクティブ間隔が経過した後に、グローバル作業単位の有効期限が切れるようキュー・マネージャーを構成することができます。

この動作を有効にするには、以下の環境変数を設定します。

- **AMQ_TRANSACTION_EXPIRY_RESCAN**=再スキャン間隔 (ミリ秒)
- **AMQ_XA_TRANSACTION_EXPIRY**=タイムアウト間隔 (ミリ秒)

重要 : この環境変数は、OPEN Group の XA 仕様の表 6-4 で **Idle** 状態となっているトランザクションにのみ影響します。

つまり、アプリケーション・スレッドに関連付けられていないものの、外部トランザクション・マネージャー・ソフトウェアによって **xa_prepare** 関数呼び出しがまだ呼び出されていないトランザクションです。

外部トランザクション・マネージャーは、準備、コミット、またはロールバックされるトランザクションのみをログに記録します。 何らかの理由で外部トランザクション・マネージャーがダウンして復帰した場合、準備、コミット、およびロールバックされたトランザクションは完了されますが、まだ準備されていないすべてのアクティブ・トランザクションは孤立します。 これを防ぐには **AMQ_XA_TRANSACTION_EXPIRY** を設定します。その際、アプリケーションが MQI トランザクション API 呼び出しを行い、他のリソース・マネージャーでのトランザクション作業を実行してトランザクションを完了するまでの予想される時間間隔を考慮に入れます。

AMQ_XA_TRANSACTION_EXPIRY 満了後の適切な時点で確実にクリーンアップするには、**AMQ_TRANSACTION_EXPIRY_RESCAN** の値を **AMQ_XA_TRANSACTION_EXPIRY** 間隔の値よりも小さい値に設定してください。**AMQ_XA_TRANSACTION_EXPIRY** 間隔の中で再スキャンが複数回にわたって発生するのが理想的です。

回復単位後処理

IBM MQ for z/OS は、リカバリー単位処理を提供します。 この機能によって、2 フェーズ・コミット・トランザクションの 2 番目のフェーズを駆動可能にするかどうかを構成できます。 例えば、リカバリー中に、同じキュー共用グループ (QSG) 内の別のキュー・マネージャーに接続される場合などです。

リカバリー単位処理は、IBM MQ for z/OS V7.0.1 以降でサポートされます。

回復単位後処理

リカバリー単位処理は、アプリケーションの接続およびそれ以降に開始されるすべてのトランザクションに関係するものです。 可能なリカバリー単位処理には、次の 2 つがあります。

- GROUP リカバリー単位処理は、トランザクション・アプリケーションが論理的にキュー共用グループに接続されていて、特定のキュー・マネージャーに対するアフィニティーがないことを識別します。 QSG 内のいずれかのキュー・マネージャーに接続されている場合に、開始される 2 フェーズ・コミット・トランザクションのうち、コミット・プロセスのフェーズ 1 を完了している (つまり、未確定である) ものを照会して解決できます。 リカバリー・シナリオでは、これは、トランザクション

調整プログラムが、同じキュー・マネージャーに再接続する必要がない (使用不可の可能性がある) ことを意味します。

- QMGR リカバリー単位処理は、アプリケーションに接続先のキュー・マネージャーに対する直接のアフィニティーがあり、このアプリケーションが開始するすべてのトランザクションにもこの処理があることを識別します。

  リカバリー・シナリオでは、そのキュー・マネージャーがキュー共用グループに属するかどうかに関係なく、トランザクション調整プログラムは同じキュー・マネージャーに再接続して、未確定トランザクションをすべて照会して解決する必要があります。

z/OS この仕様の実装方法の詳細については、 キュー共用グループ内のリカバリー処理の単位 を参照してください。

セキュリティーのシナリオ

さまざまな構成へのセキュリティーの適用を例示するシナリオの集合。

使用可能なセキュリティー・シナリオについては、以下のサブトピックに説明があります。

関連タスク

z/OS z/OS でのセキュリティーのセットアップ

z/OS セキュリティー・シナリオ: z/OS で 2 つのキュー・マネージャーを使用する場合

このシナリオで取り上げるアプリケーションは、**MQPUT1** 呼び出しを使用して、キュー・マネージャー QM1 のキューにメッセージを書き込みます。 さらに、TCP と LU 6.2 のチャネルを使用して、一部のメッセージを QM2 のキューに転送します。 SSL/TLS を使用できる TCP チャネルもあれば、使用できない TCP チャネルもあります。 アプリケーションの種類としては、バッチ・アプリケーションまたは CICS アプリケーションが考えられます。メッセージを書き込むときには、MQPMO_SET_ALL_CONTEXT オプションを使用します。

これは 88 ページの図 13 に図示されています。

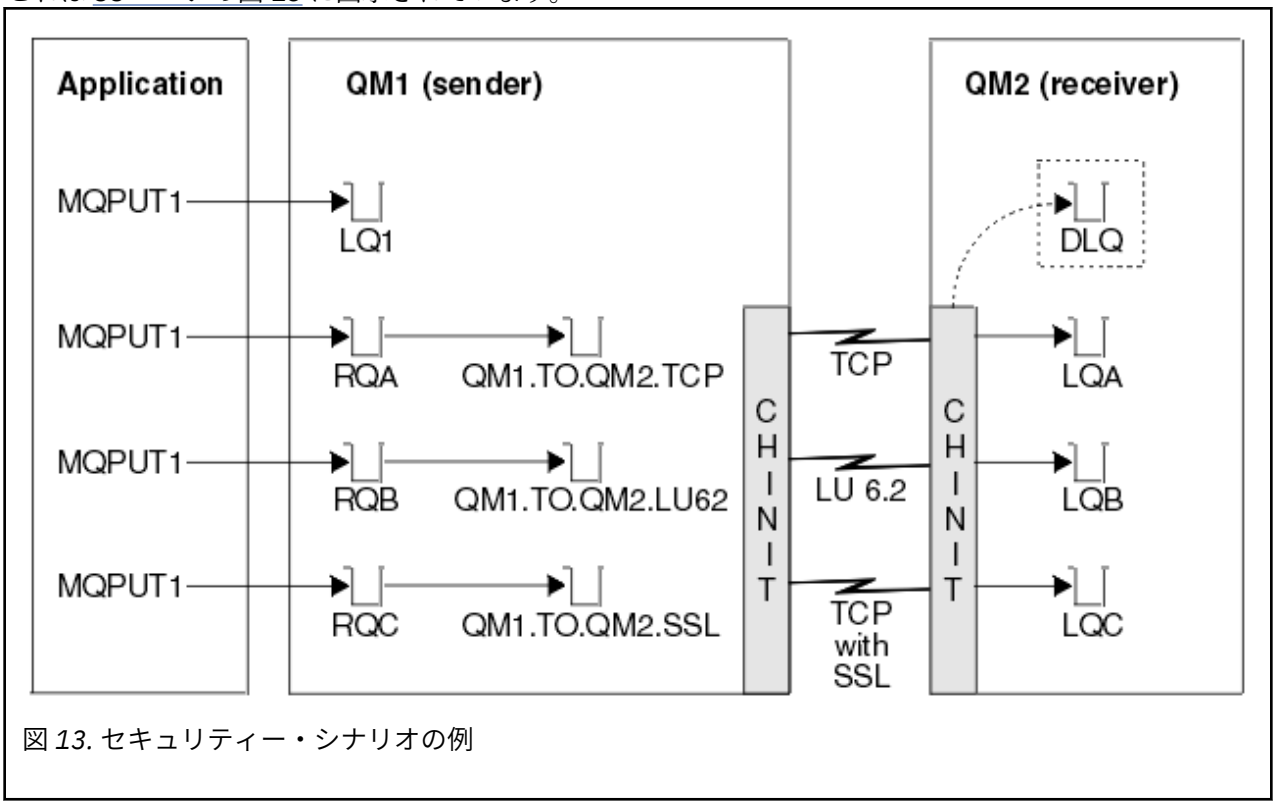


図 13. セキュリティー・シナリオの例

キュー・マネージャーについては、以下の前提条件を設定します。

- すべての必要な IBM MQ 定義は、事前定義のものであるか、キュー・マネージャーの開始時に処理される CSQINP2 データ・セットによって作成されています。

  そうでない場合は、それらのオブジェクトを定義するためのコマンドに関する適切なアクセス権限が必要になります。
- キュー・マネージャーとチャネル・イニシエーターを開始する前に、すべての必要な RACF® プロファイルが定義されていて、適切なアクセス権限が与えられています。

  そうでない場合は、すべての必要なプロファイルを定義し、それらのプロファイルに適切なアクセス権限を与えるための RACF コマンドの実行に関する適切な権限が必要になります。 さらに、新しいセキュリティー・プロファイルを使用し始めるための MQSC セキュリティー・コマンドの実行に関する適切な権限も必要です。
- すべての必要なデジタル証明書が作成されていて、鍵リングに接続されています。 QM1 から SSL/TLS ハンドシェークの一部として送信されるデジタル証明書は、QM2 のシステムの RACF によって認識されます。そのデジタル証明書がその RACF プロファイルでもインストールされているか、一致する証明書名フィルター (CNF) フィルターが存在するからです。

関連タスク

z/OS z/OS でのセキュリティーのセットアップ

z/OS 2 つのキュー・マネージャーのシナリオで使用するセキュリティー・スイッチ設定

スイッチ設定と RACF プロファイル。

両方のキュー・マネージャーで以下のセキュリティー・スイッチを設定します。

- サブシステム・セキュリティーはオン
- キュー・セキュリティーはオン
- 代替ユーザー・セキュリティーはオン
- コンテキスト・セキュリティーはオン
- プロセス・セキュリティーはオフ
- 名前リスト・セキュリティーはオフ
- トピックのセキュリティーはオフ
- 接続セキュリティーはオン
- コマンド・セキュリティーはオン
- コマンド・リソース・セキュリティーはオン

プロセス、名前リスト、およびトピックのセキュリティーをオフにするために、MQADMIN クラスの中で以下のプロファイルが定義されます。

```
QM1.NO.PROCESS.CHECKS
QM1.NO.NLIST.CHECKS
QM1.NO.TOPIC.CHECKS
QM2.NO.PROCESS.CHECKS
QM2.NO.NLIST.CHECKS
QM2.NO.TOPIC.CHECKS
```

z/OS 2 つのキュー・マネージャーのシナリオで使用するキュー・マネージャー QM1

QM1 のキューとチャネル。

キュー・マネージャー QM1 で以下のキューを定義します。

LQ1
ローカル・キュー。

RQA
リモート・キュー定義 (以下の属性を設定します)。

- RNAME(LQA)
- RQMNAME(QM2)
- XMITQ(QM1.TO.QM2.TCP)

RQB
リモート・キュー定義 (以下の属性を設定します)。

- RNAME(LQB)
- RQMNAME(QM2)
- XMITQ(QM1.TO.QM2.LU62)

RQC
リモート・キュー定義 (以下の属性を設定します)。

- RNAME(LQC)
- RQMNAME(QM2)
- XMITQ(QM1.TO.QM2.TLS)

QM1.TO.QM2.TCP
伝送キュー。

QM1.TO.QM2.LU62
伝送キュー。

QM1.TO.QM2.TLS
伝送キュー。

QM1 で以下のチャネルを定義します。

QM1.TO.QM2.TCP
送信側チャネル定義 (以下の属性を設定します)。

- CHLTYPE(SDR)
- TRPTYPE(TCP)
- XMITQ(QM1.TO.QM2.TCP)
- CONNAME(QM2TCP)

QM1.TO.QM2.LU62
送信側チャネル定義 (以下の属性を設定します)。

- CHLTYPE(SDR)
- TRPTYPE(LU62)
- XMITQ(QM1.TO.QM2.LU62)
- CONNAME(QM2LU62)

(APPC セキュリティーのセットアップについては、z/OS で使用するチャネル・イニシエーターのセキュリティーに関する考慮事項を参照してください。)

QM1.TO.QM2.TLS
送信側チャネル定義 (以下の属性を設定します)。

- CHLTYPE(SDR)
- TRPTYPE(TCP)
- XMITQ(QM1.TO.QM2.TLS)
- CONNAME(QM2TCP)

- SSLCIPH(TLS_RSA_WITH_AES_128_CBC_SHA256)

z/OS 2 つのキュー・マネージャーのシナリオで使用するキュー・マネージャー QM2

QM2 のキューとチャネル。

キュー・マネージャー QM2 で以下のキューを定義します。

LQA
ローカル・キュー。

LQB
ローカル・キュー。

LQC
ローカル・キュー。

DLQ
送達不能キューとして使用するローカル・キュー。

QM2 で以下のチャネルを定義します。

QM1.TO.QM2.TCP
受信側チャネル定義 (以下の属性を設定します)。

- CHLTYPE(RCVR)
- TRPTYPE(TCP)
- PUTAUT(CTX)
- MCAUSER(MCATCP)

QM1.TO.QM2.LU62
受信側チャネル定義 (以下の属性を設定します)。

- CHLTYPE(RCVR)
- TRPTYPE(LU62)
- PUTAUT(CTX)
- MCAUSER(MCALU62)

(APPC セキュリティーのセットアップについては、z/OS で使用するチャネル・イニシエーターのセキュリティーに関する考慮事項を参照してください。)

QM1.TO.QM2.TLS
受信側チャネル定義 (以下の属性を設定します)。

- CHLTYPE(RCVR)
- TRPTYPE(TCP)
- PUTAUT(CTX)
- MCAUSER(MCASSL)
- SSLCIPH(TLS_RSA_WITH_AES_128_CBC_SHA256)

z/OS 2 つのキュー・マネージャーのシナリオで使用するユーザー ID

このシナリオで使用するユーザー ID について説明します。

以下のユーザー ID を使用します。

BATCHID
バッチ・アプリケーション (ジョブ ID または TSO ID)

MSGUSR
MQMD の *UserIdentifier* (コンテキスト・ユーザー ID)

MOVER1
QM1 のチャネル・イニシエーター・アドレス・スペースのユーザー ID

MOVER2
QM2 のチャネル・イニシエーター・アドレス・スペースのユーザー ID

MCATCP
TCP/IP (SSL/TLS なし) 受信側チャネル定義で指定されている MCAUSER

MCALU62
LU 6.2 受信側チャネル定義で指定されている MCAUSER

MCASSL
TCP/IP (SSL/TLS あり) 受信側チャネル定義で指定されている MCAUSER

CICSAD1
CICS アドレス・スペース ID

CICSTX1
CICS タスク・ユーザー ID

CERTID
RACF によって、流れてくる証明書に関連付けられるユーザー ID

z/OS 2 つのキュー・マネージャーのシナリオで必要なセキュリティー・プロファイルとアクセス権

2 つのキュー・マネージャーのシナリオのバッチ実装または CICS 実装で必要なセキュリティー・プロファイルとアクセス権に関する情報。

以下の表は、2 つのキュー・マネージャーからなるシナリオを機能させるために必要なセキュリティー・プロファイルを示しています。 さらに、バッチまたは CICS のどちらでシナリオを実装するかに応じて、追加のセキュリティー・プロファイルが必要になります。 詳しくは、93 ページの『バッチ・アプリケーションで必要なセキュリティー・プロファイル』および 95 ページの『CICS アプリケーションで必要なセキュリティー・プロファイル』を参照してください。

表 *10.* このシナリオの例で使用するセキュリティー・プロファイル.

この表の 4 つの列は、2 つのキュー・マネージャーからなるシナリオでのクラス、プロファイル、ユーザー ID、およびアクセス権限を示しています。

クラス	プロファイル	ユーザー ID	アクセス
MQCONN	QM1.CHIN	MOVER1	READ
MQADMIN	QM1.RESLEVEL	BATCHID CICSAD1 MOVER1	NONE
MQADMIN	QM1.CONTEXT.**	MOVER1	CONTROL
MQQUEUE	QM1.SYSTEM.COMMAND.INPUT	MOVER1	UPDATE
MQQUEUE	QM1.SYSTEM.CHANNEL.SYNCQ	MOVER1	UPDATE
MQQUEUE	QM1.SYSTEM.CHANNEL.INITQ	MOVER1	UPDATE
MQQUEUE	QM1.SYSTEM.COMMAND.REPLY.MODEL	MOVER1	UPDATE
MQQUEUE	QM1.SYSTEM.ADMIN.CHANNEL.EVENT	MOVER1	UPDATE
MQQUEUE	QM1.QM1.TO.QM2.TCP	MOVER1	ALTER
MQQUEUE	QM1.QM1.TO.QM2.LU62	MOVER1	ALTER
MQQUEUE	QM1.QM1.TO.QM2.TLS	MOVER1	ALTER
MQCONN	QM2.CHIN	MOVER2	READ

表 *10.* このシナリオの例で使用するセキュリティー・プロファイル.

この表の 4 つの列は、2 つのキュー・マネージャーからなるシナリオでのクラス、プロファイル、ユーザー ID、およびアクセス権限を示しています。

(続き)

クラス	プロファイル	ユーザー ID	アクセス
MQADMIN	QM2.RESLEVEL	MOVER2	NONE
MQADMIN	QM2.CONTEXT.**	MOVER2	CONTROL
MQQUEUE	QM2.SYSTEM.COMMAND.INPUT	MOVER2	UPDATE
MQQUEUE	QM2.SYSTEM.CHANNEL.SYNCQ	MOVER2	UPDATE
MQQUEUE	QM2.SYSTEM.CHANNEL.INITQ	MOVER2	UPDATE
MQQUEUE	QM2.SYSTEM.COMMAND.REPLY.MODEL	MOVER2	UPDATE
MQQUEUE	QM2.SYSTEM.ADMIN.CHANNEL.EVENT	MOVER2	UPDATE
MQQUEUE	QM2.DLQ	MOVER2	UPDATE

z/OS バッチ・アプリケーションで必要なセキュリティー・プロファイル

2 つのキュー・マネージャーのシナリオのバッチ実装で必要な追加のセキュリティー・プロファイル。

QM1 でユーザー ID BATCHID を使用してバッチ・アプリケーションを実行します。 このバッチ・アプリケーションは、キュー・マネージャー QM1 に接続して、以下のキューにメッセージを書き込みます。

- LQ1
- RQA
- RQB
- RQC

MQPMO_SET_ALL_CONTEXT オプションを使用します。 メッセージ記述子 (MQMD) の *UserIdentifier* フィールドにある代替ユーザー ID は、MSGUSR です。

キュー・マネージャー QM1 では、以下のプロファイルが必要です。

表 *11.* キュー・マネージャー *QM1* のバッチ・アプリケーションで使用するセキュリティー・プロファイルの例

クラス	プロファイル	ユーザー ID	アクセス
MQCONN	QM1.BATCH	BATCHID	READ
MQADMIN	QM1.CONTEXT.**	BATCHID	CONTROL
MQQUEUE	QM1.LQ1	BATCHID	UPDATE
MQQUEUE	QM1.RQA	BATCHID	UPDATE
MQQUEUE	QM1.RQB	BATCHID	UPDATE
MQQUEUE	QM1.RQC	BATCHID	UPDATE

キュー・マネージャー QM2 では、キュー・マネージャー QM1 のキュー RQA に書き込まれるメッセージのために以下のプロファイルが必要です (TLS を使用しない TCP/IP チャネル)。

表 *12. TCP/IP (TLS* なし*)* を使用するキュー・マネージャー *QM2* で必要なセキュリティー・プロファイルの例

クラス	プロファイル	ユーザー ID	アクセス
MQADMIN	QM2.ALTERNATE.USER.MSGUSR	MCATCP MOVER2	UPDATE
MQADMIN	QM2.CONTEXT.**	MCATCP MOVER2	CONTROL
MQQUEUE	QM2.LQA	MOVER2 MSGUSR	UPDATE
MQQUEUE	QM2.DLQ	MOVER2 MSGUSR	UPDATE

注 :

1. 受信側チャネルは、PUTAUT(CTX) と MCAUSER(MCATCP) で定義されているので、メッセージの MQMD で渡されるユーザー ID がキュー・マネージャー QM2 の MQPUT1 のユーザー ID として使用されます。
2. 受信側チャネル定義の MCAUSER フィールドは、MCATCP に設定されています。代替ユーザー ID とコンテキスト・プロファイルに対して実行される検査では、このユーザー ID とチャネル・イニシエーター・アドレス・スペースのユーザー ID が使用されます。
3. キューに対して実行されるリソース検査では、MOVER2 のユーザー ID とメッセージ記述子 (MQMD) の *UserIdentifier* が使用されます。
4. MOVER2 と MSGUSR の両方のユーザー ID では、宛先キューに書き込めなかったメッセージを送達不能キューに送信する操作のために、送達不能キューに対するアクセス権が必要です。
5. RESLEVEL が NONE に設定されているので、実行される 3 つの検査ではいずれも 2 つのユーザー ID が対象になります。

キュー・マネージャー QM2 では、キュー・マネージャー QM1 のキュー RQB に書き込まれるメッセージのために以下のプロファイルが必要です (LU 6.2 チャネル)。

表 *13. LU 6.2* を使用するキュー・マネージャー *QM2* で必要なセキュリティー・プロファイルの例

クラス	プロファイル	ユーザー ID	アクセス
MQADMIN	QM2.ALTERNATE.USER.MSGUSR	MCALU62 MOVER1	UPDATE
MQADMIN	QM2.CONTEXT.**	MCALU62 MOVER1	CONTROL
MQQUEUE	QM2.LQB	MOVER1 MSGUSR	UPDATE
MQQUEUE	QM2.DLQ	MOVER1 MSGUSR	UPDATE

注 :

1. 受信側チャネルは、PUTAUT(CTX) と MCAUSER(MCALU62) で定義されているので、メッセージの MQMD で渡されるユーザー ID がキュー・マネージャー QM2 の MQPUT1 のユーザー ID として使用されます。
2. MCA ユーザー ID は、受信側チャネル定義の MCAUSER フィールドの値 (MCALU62) に設定されます。
3. LU 6.2 では、チャネルの通信システムのセキュリティーがサポートされているので、ネットワークから受け取るユーザー ID がチャネルのユーザー ID として使用されます (MOVER1)。
4. RESLEVEL が NONE に設定されているので、実行される 3 つの検査ではいずれも 2 つのユーザー ID が対象になります。
5. 代替ユーザー ID とコンテキスト・プロファイルに対して実行される検査では MCALU62 と MOVER1 が使用され、キュー・プロファイルに対して実行される検査では MSGUSR と MOVER1 が使用されます。
6. MOVER1 と MSGUSR の両方のユーザー ID では、宛先キューに書き込めなかったメッセージを送達不能キューに送信する操作のために、送達不能キューに対するアクセス権が必要です。

キュー・マネージャー QM2 では、キュー・マネージャー QM1 のキュー RQC に書き込まれるメッセージのために以下のプロファイルが必要です (TLS を使用する TCP/IP チャネル)。

表 *14. TCP/IP (TLS* あり*)* を使用するキュー・マネージャー *QM2* で必要なセキュリティー・プロファイルの例

クラス	プロファイル	ユーザー ID	アクセス
MQADMIN	QM2.ALTERNATE.USER.MSGUSR	MCASSL CERTID	UPDATE
MQADMIN	QM2.CONTEXT.**	MCASSL CERTID	CONTROL
MQQUEUE	QM2.LQC	CERTID MSGUSR	UPDATE
MQQUEUE	QM2.DLQ	CERTID MSGUSR	UPDATE

注 :

1. 受信側チャネルは、PUTAUT(CTX) と MCAUSER(MCASSL) で定義されているので、メッセージの MQMD で渡されるユーザー ID がキュー・マネージャー QM2 の MQPUT1 のユーザー ID として使用されます。
2. MCA ユーザー ID は、受信側チャネル定義の MCAUSER フィールドの値 (MCASSL) に設定されます。
3. TLS ハンドシェークの一部として QM1 からチャネルを経由して流れてくる証明書は、QM2 のシステムでもインストールされているか、QM2 のシステムの証明書名フィルターと一致する可能性があるので、その突き合わせで検出されたユーザー ID がチャネルのユーザー ID として使用されます (CERTID)。
4. RESLEVEL が NONE に設定されているので、実行される 3 つの検査ではいずれも 2 つのユーザー ID が対象になります。
5. 代替ユーザー ID とコンテキスト・プロファイルに対して実行される検査では MCASSL と CERTID が使用され、キュー・プロファイルに対して実行される検査では MSGUSR と MOVER1 が使用されます。
6. CERTID と MSGUSR の両方のユーザー ID では、宛先キューに書き込めなかったメッセージを送達不能キューに送信する操作のために、送達不能キューに対するアクセス権が必要です。

z/OS *CICS* アプリケーションで必要なセキュリティー・プロファイル

2 つのキュー・マネージャーのシナリオの CICS 実装で必要な追加のセキュリティー・プロファイル。

CICS アプリケーションは、CICS アドレス・スペース・ユーザー ID (CICSAD1) と CICS タスク・ユーザー ID (CICSTX1) を使用します。 キュー・マネージャー QM1 で必要なセキュリティー・プロファイルは、バッチ・アプリケーションの場合に必要なプロファイルとは異なります。 キュー・マネージャー QM2 で必要なプロファイルは、バッチ・アプリケーションの場合に必要なプロファイルと同じです。

キュー・マネージャー QM1 では、以下のプロファイルが必要です。

表 *15.* キュー・マネージャー *QM1* の *CICS* アプリケーションで使用するセキュリティー・プロファイルの例

クラス	プロファイル	ユーザー ID	アクセス
MQCONN	QM1.CICS	CICSAD1	READ
MQADMIN	QM1.CONTEXT.**	CICSAD1 CICSTX1	CONTROL
MQQUEUE	QM1.LQ1	CICSAD1 CICSTX1	UPDATE
MQQUEUE	QM1.RQA	CICSAD1 CICSTX1	UPDATE
MQQUEUE	QM1.RQB	CICSAD1 CICSTX1	UPDATE

z/OS セキュリティー・シナリオ: z/OS でキュー共有グループを使用する場合

このシナリオで取り上げるアプリケーションは、**MQPUT1** 呼び出しを使用して、キュー・マネージャー QM1 のキューにメッセージを書き込みます。 さらに、TCP と LU 6.2 のチャネルを使用して、一部のメッセージ

を QM2 のキューに転送します。 アプリケーションは、バッチ・アプリケーションです。メッセージを書き込むときには、MQPMO_SET_ALL_CONTEXT オプションを使用します。

これは 88 ページの図 13 に図示されています。

キュー・マネージャーについては、以下の前提条件を設定します。

- すべての必要な IBM MQ 定義は、事前定義のものであるか、キュー・マネージャーの開始時に処理される CSQINP2 データ・セットによって作成されています。

  そうでない場合は、それらのオブジェクトを定義するためのコマンドに関する適切なアクセス権限が必要になります。
- キュー・マネージャーとチャネル・イニシエーターを開始する前に、すべての必要な RACF プロファイルが定義されていて、適切なアクセス権限が与えられています。

  そうでない場合は、すべての必要なプロファイルを定義し、それらのプロファイルに適切なアクセス権限を与えるための RACF コマンドの実行に関する適切な権限が必要になります。 さらに、新しいセキュリティー・プロファイルを使用し始めるための MQSC セキュリティー・コマンドの実行に関する適切な権限も必要です。

関連タスク

z/OS z/OS でのセキュリティーのセットアップ

z/OS キュー共有グループのシナリオで使用するセキュリティー・スイッチ設定

スイッチ設定と RACF プロファイル。

キュー共有グループで以下のセキュリティー・スイッチを設定します。

- サブシステム・セキュリティーはオン
- キュー共有グループ・セキュリティーはオン
- キュー・マネージャー・セキュリティーはオフ
- キュー・セキュリティーはオン
- 代替ユーザー・セキュリティーはオン
- コンテキスト・セキュリティーはオン
- プロセス・セキュリティーはオフ
- 名前リスト・セキュリティーはオフ
- トピックのセキュリティーはオフ
- 接続セキュリティーはオン
- コマンド・セキュリティーはオン
- コマンド・リソース・セキュリティーはオン

プロセス、名前リスト、トピック、およびキュー・マネージャー・レベルのセキュリティーをオフにするために、MQADMIN クラスの中で以下のプロファイルが定義されます。

```
QSGA.NO.PROCESS.CHECKS
QSGA.NO.NLIST.CHECKS
QSGA.NO.TOPIC.CHECKS
QSGA.NO.QMGR.CHECKS
```

z/OS キュー共有グループのシナリオで使用するキュー・マネージャー QM1

QM1 のキューとチャネル。

キュー・マネージャー QM1 で以下のキューを定義します。

LQ1
ローカル・キュー。

RQA
リモート・キュー定義 (以下の属性を設定します)。

- RNAME(LQA)
- RQMNAME(QM2)
- XMITQ(QM1.TO.QM2.TCP)

RQB
リモート・キュー定義 (以下の属性を設定します)。

- RNAME(LQB)
- RQMNAME(QM2)
- XMITQ(QM1.TO.QM2.LU62)

QM1.TO.QM2.TCP
伝送キュー。

QM1.TO.QM2.LU62
伝送キュー。

QM1 で以下のチャネルを定義します。

QM1.TO.QM2.TCP
送信側チャネル定義 (以下の属性を設定します)。

- CHLTYPE(SDR)
- TRPTYPE(TCP)
- XMITQ(QM1.TO.QM2.TCP)
- CONNAME(QM2TCP)

QM1.TO.QM2.LU62
送信側チャネル定義 (以下の属性を設定します)。

- CHLTYPE(SDR)
- TRPTYPE(LU62)
- XMITQ(QM1.TO.QM2.LU62)
- CONNAME(QM2LU62)

(APPC セキュリティーのセットアップについては、z/OS で使用するチャネル・イニシエーターのセキュリティーに関する考慮事項を参照してください。)

z/OS キュー共有グループのシナリオで使用するキュー・マネージャー QM2

QM2 のキューとチャネル。

キュー・マネージャー QM2 で以下のキューを定義します。

LQA
ローカル・キュー。

LQB
ローカル・キュー。

DLQ
送達不能キューとして使用するローカル・キュー。

QM2 で以下のチャネルを定義します。

QM1.TO.QM2.TCP
受信側チャネル定義 (以下の属性を設定します)。

- CHLTYPE(RCVR)

- TRPTYPE(TCP)
- PUTAUT(CTX)
- MCAUSER(MCATCP)

QM1.TO.QM2.LU62
受信側チャネル定義 (以下の属性を設定します)。

- CHLTYPE(RCVR)
- TRPTYPE(LU62)
- PUTAUT(CTX)
- MCAUSER(MCALU62)

(APPC セキュリティーのセットアップについては、z/OS で使用するチャネル・イニシエーターのセキュリティーに関する考慮事項を参照してください。)

z/OS キュー共有グループのシナリオで使用するユーザー ID

このシナリオで使用するユーザー ID について説明します。

以下のユーザー ID を使用します。

BATCHID
バッチ・アプリケーション (ジョブ ID または TSO ID)

MSGUSR
MQMD の *UserIdentifier* (コンテキスト・ユーザー ID)

MOVER1
QM1 のチャネル・イニシエーター・アドレス・スペースのユーザー ID

MOVER2
QM2 のチャネル・イニシエーター・アドレス・スペースのユーザー ID

MCATCP
TCP/IP 受信側チャネル定義で指定されている MCAUSER

MCALU62
LU 6.2 受信側チャネル定義で指定されている MCAUSER

z/OS キュー共有グループのシナリオで必要なセキュリティー・プロファイルとアクセス権

キュー共有グループのシナリオのバッチ実装または CICS 実装で使用するセキュリティー・プロファイルとアクセス権

以下の表は、キュー共用グループのシナリオを機能させるために必要なセキュリティー・プロファイルを示しています。 また、このシナリオをバッチで実装する場合には、追加のセキュリティー・プロファイルが必要になります (99 ページの『バッチ・アプリケーションで必要なセキュリティー・プロファイル』の説明を参照してください)。

表 *16.* このシナリオの例で使用するセキュリティー・プロファイル.

この表の 4 つの列は、キュー共用グループのシナリオでのクラス、プロファイル、ユーザー ID、およびアクセス権限を示しています。

クラス	プロファイル	ユーザー ID	アクセス
MQCONN	QSGA.CHIN	MOVER1 MOVER2	READ
MQADMIN	QSGA.RESLEVEL	BATCHID MOVER1 MOVER2	NONE

表 16. このシナリオの例で使用するセキュリティー・プロファイル.

この表の 4 つの列は、キュー共用グループのシナリオでのクラス、プロファイル、ユーザー ID、およびアクセス権限を示しています。

(続き)

クラス	プロファイル	ユーザー ID	アクセス
MQADMIN	QSGA.CONTEXT.**	MOVER1 MOVER2	CONTROL
MQQUEUE	QSGA.SYSTEM.COMMAND.INPUT	MOVER1 MOVER2	UPDATE
MQQUEUE	QSGA.SYSTEM.CHANNEL.SYNCQ	MOVER1 MOVER	UPDATE
MQQUEUE	QSGA.SYSTEM.CHANNEL.INITQ	MOVER1 MOVER2	UPDATE
MQQUEUE	QSGA.SYSTEM.COMMAND.REPLY.MODEL	MOVER1 MOVER2	UPDATE
MQQUEUE	QSGA.SYSTEM.ADMIN.CHANNEL.EVENT	MOVER1 MOVER2	UPDATE
MQQUEUE	QSGA.SYSTEM.QSG.CHANNEL.SYNCQ	MOVER1 MOVER2	UPDATE
MQQUEUE	QSGA.SYSTEM.QSG.TRANSMIT.QUEUE	MOVER1 MOVER2	UPDATE
MQQUEUE	QSGA.QM1.TO.QM2.TCP	MOVER1	ALTER
MQQUEUE	QSGA.QM1.TO.QM2.LU62	MOVER1	ALTER
MQQUEUE	QSGA.DLQ	MOVER2	UPDATE

z/OS バッチ・アプリケーションで必要なセキュリティー・プロファイル

キュー共有グループのシナリオのバッチ実装で必要な追加のセキュリティー・プロファイル。

QM1 でユーザー ID BATCHID を使用してバッチ・アプリケーションを実行します。 このバッチ・アプリケーションは、キュー・マネージャー QM1 に接続して、以下のキューにメッセージを書き込みます。

- LQ1
- RQA
- RQB

MQPMO_SET_ALL_CONTEXT オプションを使用します。 メッセージ記述子 (MQMD) の *UserIdentifier* フィールドにあるユーザー ID は MSGUSR です。

キュー・マネージャー QM1 では、以下のプロファイルが必要です。

表 17. キュー・マネージャー *QM1* のバッチ・アプリケーションで使用するセキュリティー・プロファイルの例

クラス	プロファイル	ユーザー ID	アクセス
MQCONN	QSGA.BATCH	BATCHID	READ
MQADMIN	QSGA.CONTEXT.**	BATCHID	CONTROL
MQQUEUE	QSGA.LQ1	BATCHID	UPDATE
MQQUEUE	QSGA.RQA	BATCHID	UPDATE

表 *17.* キュー・マネージャー *QM1* のバッチ・アプリケーションで使用するセキュリティー・プロファイルの例 (続き)			
クラス	**プロファイル**	**ユーザー ID**	**アクセス**
MQQUEUE	QSGA.RQB	BATCHID	UPDATE

キュー・マネージャー QM2 では、キュー・マネージャー QM1 のキュー RQA に書き込まれるメッセージのために以下のプロファイルが必要です (TCP/IP チャネル)。

表 *18. TCP/IP* を使用するキュー・マネージャー *QM2* で必要なセキュリティー・プロファイルの例			
クラス	**プロファイル**	**ユーザー ID**	**アクセス**
MQADMIN	QSGA.ALTERNATE.USER.MSGUSR	MCATCP MOVER2	UPDATE
MQADMIN	QSGA.CONTEXT.**	MCATCP MOVER2	CONTROL
MQQUEUE	QSGA.LQA	MOVER2 MSGUSR	UPDATE
MQQUEUE	QSGA.DLQ	MOVER2 MSGUSR	UPDATE

注 :

1. 受信側チャネルは、PUTAUT(CTX) と MCAUSER(MCATCP) で定義されているので、メッセージの MQMD で渡されるユーザー ID がキュー・マネージャー QM2 の MQPUT1 のユーザー ID として使用されます。
2. 受信側チャネル定義の MCAUSER フィールドは、MCATCP に設定されています。代替ユーザー ID とコンテキスト・プロファイルに対して実行される検査では、このユーザー ID とチャネル・イニシエーター・アドレス・スペースのユーザー ID が使用されます。
3. キューに対して実行されるリソース検査では、MOVER2 のユーザー ID とメッセージ記述子 (MQMD) の *UserIdentifier* が使用されます。
4. MOVER2 と MSGUSR の両方のユーザー ID では、宛先キューに書き込めなかったメッセージを送達不能キューに送信する操作のために、送達不能キューに対するアクセス権が必要です。
5. RESLEVEL が NONE に設定されているので、実行される 3 つの検査ではいずれも 2 つのユーザー ID が対象になります。

キュー・マネージャー QM2 では、キュー・マネージャー QM1 のキュー RQB に書き込まれるメッセージのために以下のプロファイルが必要です (LU 6.2 チャネル)。

表 *19. LU 6.2* を使用するキュー・マネージャー *QM2* で必要なセキュリティー・プロファイルの例			
クラス	**プロファイル**	**ユーザー ID**	**アクセス**
MQADMIN	QSGA.ALTERNATE.USER.MSGUSR	MCALU62 MOVER1	UPDATE
MQADMIN	QSGA.CONTEXT.**	MCALU62 MOVER1	CONTROL
MQQUEUE	QSGA.LQB	MOVER1 MSGUSR	UPDATE
MQQUEUE	QSGA.DLQ	MOVER1 MSGUSR	UPDATE

注 :

1. 受信側チャネルは、PUTAUT(CTX) と MCAUSER(MCALU62) で定義されているので、メッセージの MQMD で渡されるユーザー ID がキュー・マネージャー QM2 の MQPUT1 のユーザー ID として使用されます。
2. MCA ユーザー ID は、受信側チャネル定義の MCAUSER フィールドの値 (MCALU62) に設定されます。

3. LU 6.2 では、チャネルの通信システムのセキュリティーがサポートされているので、ネットワークから受け取るユーザー ID がチャネルのユーザー ID として使用されます (MOVER1)。
4. RESLEVEL が NONE に設定されているので、実行される 3 つの検査ではいずれも 2 つのユーザー ID が対象になります。
5. 代替ユーザー ID とコンテキスト・プロファイルに対して実行される検査では MCALU62 と MOVER1 が使用され、キュー・プロファイルに対して実行される検査では MSGUSR と MOVER1 が使用されます。
6. MOVER1 と MSGUSR の両方のユーザー ID では、宛先キューに書き込めなかったメッセージを送達不能キューに送信する操作のために、送達不能キューに対するアクセス権が必要です。

z/OS サーバー間メッセージ・チャネル・インターセプトの構成例

サーバー間メッセージ・チャネル・インターセプトでは、インバウンド・メッセージとアウトバウンド・メッセージを正しく保護または無保護にできるように、チャネル定義と Advanced Message Security ポリシーを構成する必要があります。 構成は、チャネルがインバウンドであるかアウトバウンドであるかによって異なります。

インバウンド・チャネル

以下の例は、受信側タイプのインバウンド・チャネルの標準的な構成と、無保護のインバウンド・メッセージを保護するために必要な AMS ポリシーの詳細を示しています。

図 *14.* インバウンド構成

この例は次のものを示しています。

- キュー・マネージャー QMA
- チャネル TO.QMA
- ローカル・キュー DESTQ

次のコードを使用します。

```
DEFINE CHANNEL(TO.QMA) CHLTYPE(RCVR) SSLCAUTH(REQUIRED) SSLCIPH(ANY_TLS12) TRPTYPE(TCP)
SPLPROT(ASPOLICY)

DEFINE QLOCAL(DESTQ) DESCR('AMS PROTECTED QUEUE')

setmqspl -m QMA -p DESTQ -e AES256 -r CN=TEST,O=ORG,C=US
```

注 : 上記の本文で説明したポリシーは、メッセージのみを暗号化します。つまり、AMS の機密性です。

z/OS での **setmqspl** の使用については、 setmqspl および メッセージ・セキュリティー・ポリシー (CSQ0UTIL) を参照してください。

アウトバウンド・チャネル

以下の例は、送信側タイプのアウトバウンド・チャネルの標準的な構成を示しています。 この例では、リモート・キューに書き込まれたメッセージを保護し、伝送キューから取得したメッセージの保護を解除して送信するために必要な、AMS ポリシーの詳細を説明します。

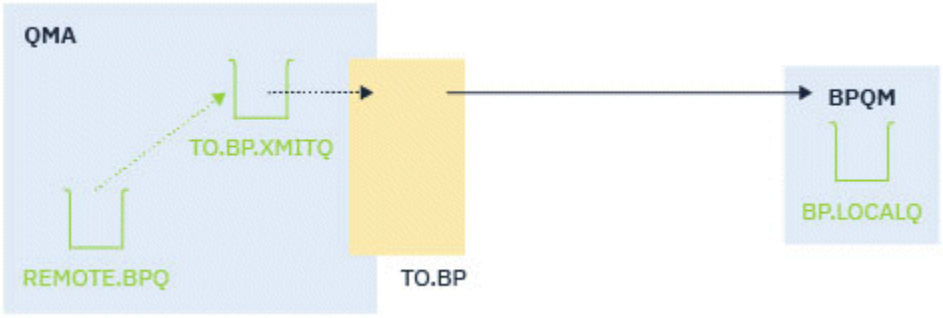


図 *15.* アウトバウンド構成

この例は次のものを示しています。

- キュー・マネージャー QMA
- チャネル TO.BP
- ローカル伝送キュー TO.BP.XMITQ
- リモート・キュー REMOTE.BPQ

次のコードを使用します。

```
DEFINE CHANNEL(TO.BP) CHLTYPE(SDR) SSLCAUTH(REQUIRED) SSLCIPH(ANY_TLS12) TRPTYPE(TCP)
SPLPROT(REMOVE) CONNAME('server(1414)') XMITQ(TO.BP.XMITQ)

DEFINE QLOCAL(TO.BP.XMITQ) DESCR('TRANSMISSION QUEUE FOR TO.BP') USAGE(XMITQ)

DEFINE QREMOTE(REMOTE.BPQ) DESCR('REMOTE QUEUE TO BP') RNAME(BP.LOCALQ) RQMANME(BPQM)
XMITQ(TO.BP.XMITQ)

setmqspl -m QMA -p TO.BP.XMITQ -e AES256 -r CN=TEST,O=ORG,C=US

setmqspl -m QMA -p REMOTE.BPQ -e AES256 -r CN=TEST,O=ORG,C=US
```

注 : 上記の本文で説明したポリシーは、メッセージのみを暗号化します。つまり、AMS の機密性です。

SSL/TLS による 2 つのキュー・マネージャーの接続

TLS 暗号セキュリティー・プロトコルを使用するセキュア通信では、通信チャネルをセットアップし、認証に使用するデジタル証明書を管理する必要があります。

SSL/TLS インストール環境をセットアップするには、TLS を使用するようにチャネルを定義する必要があります。 また、デジタル証明書を取得し、管理することも必要です。 テスト・システムでは、自己署名証明書か、ローカル認証局 (CA) で発行された証明書を使用できます。 実動システムでは、自己署名証明書を使用しないでください。

証明書の作成と管理の詳細については、以下のトピックを参照してください。

- IBM i IBM i での SSL または TLS の取り扱い
- ALW AIX, Linux, and Windows・システムでの SSL または TLS の取り扱い
- z/OS z/OS での SSL または TLS の取り扱い

このトピック集では、SSL/TLS 通信のセットアップに関連したタスクを取り上げ、それらのタスクを実行するための段階的な手順を説明します。

また、プロトコルのオプション部分である SSL/TLS クライアント認証をテストすることもできます。 SSL/TLS ハンドシェーク中に、SSL/TLS クライアントは常にサーバーからデジタル証明書を取得し検証します。 IBM MQ の実装では、SSL/TLS サーバーは、常にクライアントから証明書を要求します。

注 :

1. この状況では、SSL/TLS クライアントはハンドシェークを開始する接続を参照します。

2. z/OS z/OS キュー・マネージャーは、SSL/TLS クライアントの役割を果たす場合は、証明書のみを送信します。

SSL/TLS クライアントと SSL/TLS クライアントは、ラベルが一致する証明書が見つかった場合にのみ、証明書を送信します。 詳細は デジタル証明書ラベル を参照してください。

SSL/TLS サーバーは、クライアント証明書が送信される場合は、常にそのクライアント証明書を検証します。 クライアントが証明書を送信しない場合に認証が失敗するのは、チャネルの SSL/TLS サーバー側の定義で、**SSLCAUTH** パラメーターが REQUIRED に設定されている場合、または **SSLPEER** パラメーターの値が設定されている場合に限られます。 匿名によるキュー・マネージャーへの接続 (つまり、SSL/TLS クライアントから証明書を送信しない場合) について詳しくは、107 ページの『片方向認証による 2 つのキュー・マネージャーの接続』を参照してください。

2 つのキュー・マネージャーの相互認証への自己署名証明書の使用

自己署名 TLS 証明書を使用して 2 つのキュー・マネージャーの間で相互認証を実装するための手順の例を取り上げます。

このタスクについて

シナリオ

- 安全に通信する必要のあるキュー・マネージャーが 2 つ (QM1 と QM2) あります。 QM1 と QM2 の間で、相互認証を実行する必要があります。
- そこで、自己署名証明書を使用して安全な通信をテストすることにしました。

構成の結果は次のようになります。

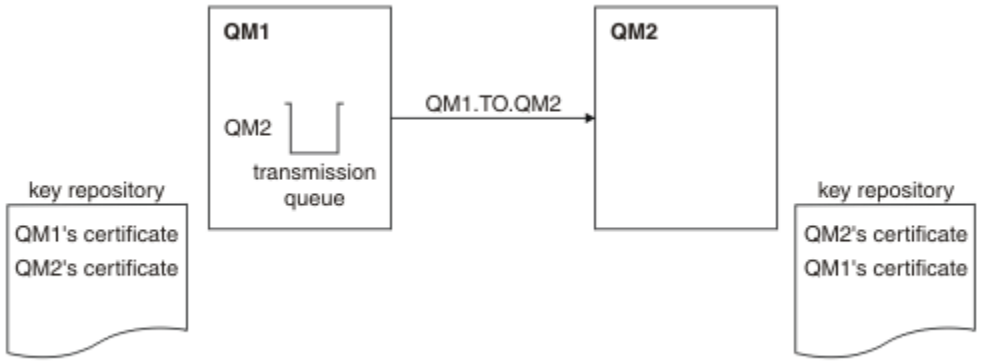


図 *16.* このタスクで実装する構成

103 ページの図 16 にあるとおり、QM1 の鍵リポジトリーには QM1 の証明書と QM2 の公開証明書を格納します。 QM2 の鍵リポジトリーには、QM2 の証明書と QM1 の公開証明書を格納します。

手順

1. オペレーティング・システムに従って、各キュー・マネージャーで鍵リポジトリーを準備します。
   - ALW AIX, Linux, and Windows・システムの場合。
   - z/OS z/OS・システムの場合。
2. キュー・マネージャーごとに自己署名証明書を作成します。
   - ALW AIX, Linux, and Windows・システムの場合。
   - z/OS z/OS・システムの場合。
3. 各証明書のコピーを取り出します。

- ALW AIX, Linux, and Windows・システムの場合。
- z/OS z/OS・システムの場合。

4. FTP などのユーティリティーを使用して QM1 証明書の公開部分を QM2 システムに転送し、またその逆を実行します z/OS (自己署名証明書の交換を参照)。
5. キュー・マネージャーごとにパートナーの証明書を鍵リポジトリーに追加します。
   - ALW AIX, Linux, and Windows・システムの場合。
   - z/OS z/OS・システムの場合。
6. QM1 で以下の例のようなコマンドを実行して、送信側チャネルとそれに関連する伝送キューを定義します。

```
DEFINE CHANNEL(QM1.TO.QM2) CHLTYPE(SDR) TRPTYPE(TCP) CONNAME(QM1.MACH.COM) XMITQ(QM2)
SSLCIPH(TLS_RSA_WITH_AES_128_CBC_SHA) DESCR('Sender channel using TLS from QM1 to QM2')

DEFINE QLOCAL(QM2) USAGE(XMITQ)
```

   この例では、CipherSpec TLS_RSA を使用します。 チャネルの両端の CipherSpec は同じでなければなりません。
7. QM2 で以下の例のようなコマンドを実行して、受信側チャネルを定義します。

```
DEFINE CHANNEL(QM1.TO.QM2) CHLTYPE(RCVR) TRPTYPE(TCP) SSLCIPH(TLS_RSA_WITH_AES_128_CBC_SHA)
SSLCAUTH(REQUIRED) DESCR('Receiver channel using TLS from QM1 to QM2')
```

   このチャネルは、ステップ 104 ページの『6』 で定義した送信側チャネルと同じ名前でなければならず、使用する CipherSpec も同じでなければなりません。
8. チャネルを始動してください。

   z/OS z/OS の場合は、送信側チャネルの開始を参照してください。

タスクの結果

作成した鍵リポジトリーとチャネルを図にまとめたのが、103 ページの図 16 です。

次のタスク

DISPLAY コマンドを使用して、タスクが正常に完了していることを確認します。 タスクが正常に完了していれば、以下の例のような出力が表示されます。

キュー・マネージャー QM1 から以下のコマンドを入力します。

```
DISPLAY CHS(QM1.TO.QM2) SSLPEER SSLCERTI
```

結果の出力は、以下の例のようになります。

```
DISPLAY CHSTATUS(QM1.TO.QM2) SSLPEER SSLCERTI
     4 : DISPLAY CHSTATUS(QM1.TO.QM2) SSLPEER SSLCERTI
AMQ8417: Display Channel Status details.
   CHANNEL(QM1.TO.QM2)                     CHLTYPE(SDR)
   CONNAME(9.20.25.40)                     CURRENT
   RQMNAME(QM2)
   SSLCERTI("CN=QM2,OU=<Department>,O=<Organization>,ST=<State>,C=<Country>")

SSLPEER("SERIALNUMBER=4C:D0:49:D5:02:5E:02,CN=QM2,OU=<Department>,O=<Organization>,ST=<State>,C=
<Country>")
   STATUS(RUNNING)                         SUBSTATE(MQGET)
   XMITQ(QM2)
```

キュー・マネージャー QM2 から以下のコマンドを入力します。

```
DISPLAY CHS(QM1.TO.QM2) SSLPEER SSLCERTI
```

結果の出力は、以下の例のようになります。

```
DISPLAY CHSTATUS(QM1.TO.QM2) SSLPEER SSLCERTI
     5 : DISPLAY CHSTATUS(QM1.TO.QM2) SSLPEER SSLCERTI
AMQ8417: Display Channel Status details.
   CHANNEL(QM2.TO.QM1)                     CHLTYPE(RCVR)
   CONNAME(9.20.35.92)                     CURRENT
   RQMNAME(QM1)
   SSLCERTI("CN=QM1,OU=<Department>,O=<Organization>,ST=<State>,C=<Country>")

SSLPEER("SERIALNUMBER=4C:D0:49:D5:02:5F:38,CN=QM1,OU=<Department>,O=<Organization>,ST=<State>,C=
<Country>")
   STATUS(RUNNING)                         SUBSTATE(RECEIVE)
   XMITQ( )
```

どの場合でも、SSLPEER の値は、手順 103 ページの『2』 で作成したパートナー証明書の DN の値と一致している必要があります。 この証明書は自己署名証明書なので、発行者の名前はピアの名前と一致しています。

SSLPEER はオプションです。 これを指定する場合は、パートナー証明書 (ステップ 103 ページの『2』 で作成された) の DN が許可されるように値を設定する必要があります。 SSLPEER の使用について詳しくは、SSLPEER 値についての IBM MQ の規則を参照してください。

2 つのキュー・マネージャーの相互認証への CA 署名証明書の使用

CA 署名 TLS 証明書を使用して 2 つのキュー・マネージャーの間で相互認証を実装するための手順の例を取り上げます。

このタスクについて

シナリオ

- 安全に通信する必要のあるキュー・マネージャーが 2 つあります (QM1 と QM2)。 QM1 と QM2 の間で、相互認証を実行する必要があります。
- 将来的にはこのネットワークを実稼働環境で使用することを計画しているので、最初から CA 署名証明書を使用することにしました。

構成の結果は次のようになります。

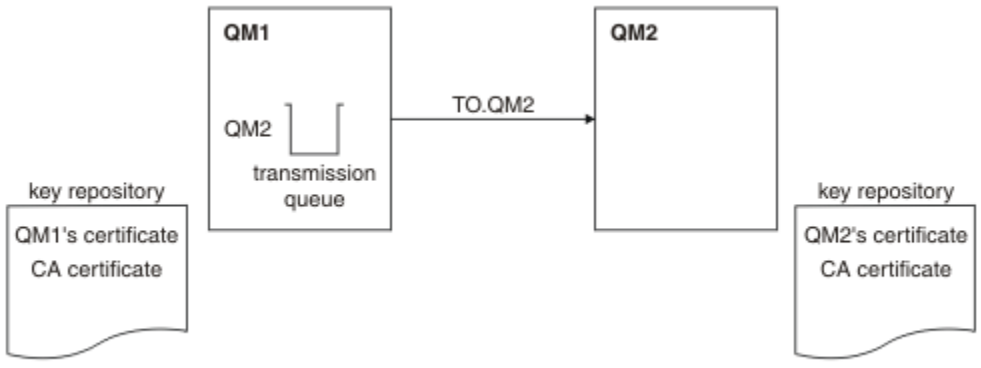


図 *17.* このタスクで実装する構成

105 ページの図 17 にあるとおり、QM1 の鍵リポジトリーには、QM1 の証明書と CA 証明書を格納します。 QM2 の鍵リポジトリーには、 QM2 の証明書と CA 証明書を格納します。

この例では、QM1 の証明書と QM2 の証明書の両方が同じ CA から発行されました。 QM1 の証明書と QM2 の証明書が別々の CA から発行された場合は、QM1 と QM2 の鍵リポジトリーには両方の CA 証明書が含まれている必要があります。

手順

1. 社内で使用しているオペレーティング・システムに合わせて、各キュー・マネージャーで鍵リポジトリーを準備します。
   - IBM i IBM i システムの場合。
   - ALW AIX, Linux, and Windows・システムの場合。
   - z/OS z/OS・システムの場合。
2. キュー・マネージャーごとに CA 署名証明書を要求します。
   2 つのキュー・マネージャーで異なる CA を使用することができます。
   - IBM i IBM i システムの場合。
   - ALW AIX, Linux, and Windows・システムの場合。
   - z/OS z/OS・システムの場合。
3. キュー・マネージャーごとに認証局証明書を鍵リポジトリーに追加します。
   これらのキュー・マネージャーがそれぞれ異なる認証局を使用する場合は、各認証局の CA 証明書を両方の鍵リポジトリーに追加する必要があります。
   - IBM i IBM i システムでは、この手順を実行しないでください。
   - ALW AIX, Linux, and Windows・システムの場合。
   - z/OS z/OS・システムの場合。
4. キュー・マネージャーごとに CA 署名証明書を鍵リポジトリーに取り込みます。
   - IBM i IBM i システムの場合。
   - ALW AIX, Linux, and Windows・システムの場合。
   - z/OS z/OS・システムの場合。
5. QM1 で以下の例のようなコマンドを実行して、送信側チャネルとそれに関連する伝送キューを定義します。

```
DEFINE CHANNEL(TO.QM2) CHLTYPE(SDR) TRPTYPE(TCP)
CONNAME(QM2.MACH.COM) XMITQ(QM2) SSLCIPH(TLS_RSA_WITH_AES_128_CBC_SHA256)
DESCR('Sender channel using TLS from QM1 to QM2')

DEFINE QLOCAL(QM2) USAGE(XMITQ)
```

   この例では、CipherSpec TLS_RSA_WITH_AES_128_CBC_SHA256 を使用します。 チャネルの両端の CipherSpec は同じでなければなりません。
6. QM2 で以下の例のようなコマンドを実行して、受信側チャネルを定義します。

```
DEFINE CHANNEL(TO.QM2) CHLTYPE(RCVR) TRPTYPE(TCP)
SSLCIPH(TLS_RSA_WITH_AES_128_CBC_SHA256) SSLCAUTH(REQUIRED)
DESCR('Receiver channel using TLS to QM2')
```

   このチャネルは、ステップ 106 ページの『5』 で定義した送信側チャネルと同じ名前でなければならず、使用する CipherSpec も同じでなければなりません。
7. チャネルを開始します。
   - IBM i IBM i システムの場合。

- ALW AIX, Linux, and Windows・システムの場合。
- z/OS z/OS・システムの場合。

タスクの結果

作成した鍵リポジトリーとチャネルを図にまとめたのが、105 ページの図 17 です。

次のタスク

DISPLAY コマンドを使用して、タスクが正常に完了していることを確認します。 タスクが正常に完了していれば、以下の例のような出力が表示されます。

キュー・マネージャー QM1 から以下のコマンドを入力します。

```
DISPLAY CHS(TO.QM2) SSLPEER SSLCERTI
```

結果の出力は、以下の例のようになります。

```
DISPLAY CHSTATUS(TO.QM2) SSLPEER SSLCERTI
     4 : DISPLAY CHSTATUS(TO.QM2) SSLPEER SSLCERTI
AMQ8417: Display Channel Status details.
   CHANNEL(TO.QM2)                         CHLTYPE(SDR)
   CONNAME(192.0.0.2)                      CURRENT
   RQMNAME(QM2)
   SSLCERTI("CN=<Division> CA,OU=<Department>,O=<Organization>,ST=<State>,C=<Country>")

SSLPEER("SERIALNUMBER=4C:D0:49:D5:02:5F:38,CN=QM2,OU=<Department>,O=<Organization>,ST=<State>,C=
<Country>")
   STATUS(RUNNING)                         SUBSTATE(MQGET)
   XMITQ(QM2)
```

キュー・マネージャー QM2 から以下のコマンドを入力します。

```
DISPLAY CHS(TO.QM2) SSLPEER SSLCERTI
```

結果の出力は、以下の例のようになります。

```
DISPLAY CHSTATUS(TO.QM2) SSLPEER SSLCERTI
     5 : DISPLAY CHSTATUS(TO.QM2) SSLPEER SSLCERTI
AMQ8417: Display Channel Status details.
   CHANNEL(TO.QM2)                         CHLTYPE(RCVR)
   CONNAME(192.0.0.1)                      CURRENT
   RQMNAME(QM1)
   SSLCERTI("CN=<Division> CA,OU=<Department>,O=<Organization>,ST=<State>,C=<Country>")

SSLPEER("SERIALNUMBER=4C:D0:49:D5:02:5F:38,CN=QM1,OU=<Department>,O=<Organization>,ST=<State>,C=
<Country>")
   STATUS(RUNNING)                         SUBSTATE(RECEIVE)
   XMITQ( )
```

いずれの場合でも、SSLPEER の値は、手順 106 ページの『2』 で作成したパートナー証明書の識別名 (DN) の値と一致している必要があります。 発行者の名前は、手順 106 ページの『4』 で追加した個人証明書に署名した CA 証明書のサブジェクトの DN と一致しています。

片方向認証による 2 つのキュー・マネージャーの接続

相互認証を使用するシステムを変更して、キュー・マネージャーが片方向認証 (つまり、SSL/TLS クライアントが証明書を送信しない方式) を使用して別のキュー・マネージャーに接続できるようにするには、以下のサンプル手順を実行してください。

このタスクについて

シナリオ

- 105 ページの『2 つのキュー・マネージャーの相互認証への CA 署名証明書の使用』で、2 つのキュー・マネージャー (QM1 と QM2) がセットアップされました。
- 片方向認証を使用して QM2 に接続されるように QM1 を変更する必要があるとします。

構成の結果は次のようになります。

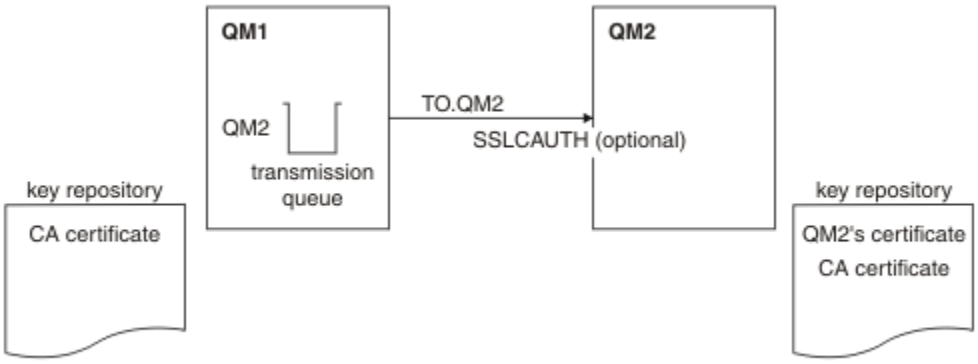


図 *18.* 片方向認証を許可するキュー・マネージャー

手順

1. QM1 の個人証明書を鍵リポジトリーから削除します。
   - IBM i IBM i システム上の証明書の削除.
   - ALW AIX, Linux, and Windows 上の証明書の削除.
   - z/OS z/OS システムで証明書を削除します。 このステップを 2 回実行したら、 QMA の個人証明書とデフォルト証明書の両方を削除できます。

   証明書のラベルを設定する方法について詳しくは、デジタル証明書ラベルを参照してください。
2. オプション: QM1 で SSL/TLS チャネルを実行したことがある場合は、SSL/TLS 環境をリフレッシュします (TLS 環境のリフレッシュを参照)。
3. 受信側で匿名接続を許可します (受信側チャネルで匿名の接続を許可する操作の説明を参照してください)。

   変更した鍵リポジトリーとチャネルを図にまとめたのが、108 ページの図 18 です。
4. 送信側チャネルが稼働していなかった場合は、送信側チャネルを始動します。

   注 : 送信側チャネルの稼働中に REFRESH SECURITY TYPE(SSL) コマンドを実行した場合は (手順 2)、チャネルが自動的に再始動されます。

   チャネルのサーバー側で、チャネル状況表示にピア名パラメーター値が表示される場合は、クライアント証明書が流れたことを意味します。
5. DISPLAY コマンドを使用して、タスクが正常に完了していることを確認します。
   タスクが正常に完了していれば、以下の例のような出力が表示されます。
   - QM1 キュー・マネージャーから、次のコマンドを入力します。

     ```
     DISPLAY CHS(TO.QM2) SSLPEER SSLCERTI
     ```

     結果の出力は、以下の例のようになります。

     ```
     DISPLAY CHSTATUS(TO.QMB) SSLPEER SSLCERTI
          4 : DISPLAY CHSTATUS(TO.QMB) SSLPEER
     AMQ8417: Display Channel Status details.
        CHANNEL(TO.QM2)                         CHLTYPE(SDR)
        CONNAME(192.0.0.1)                      CURRENT
     ```

```
   RQMNAME(QM2)
   SSLCERTI("CN=IBM MQ CA,OU=IBM MQ Devt,O=IBM,ST=Hampshire,C=UK")
   SSLPEER("SERIALNUMBER=4C:D0:49:D5:02:5F:38,CN=QMB,OU=IBM MQ
Development,O=IBM,ST=Hampshire,C=UK")
   STATUS(RUNNING)                          SUBSTATE(MQGET)
   XMITQ(QM2)
```

- QM2 キュー・マネージャーから、次のコマンドを入力します。

```
DISPLAY CHS(TO.QM2) SSLPEER SSLCERTI
```

結果の出力は、以下の例のようになります。

```
DISPLAY CHSTATUS(TO.QM2) SSLPEER SSLCERTI
     5 : DISPLAY CHSTATUS(TO.QM2) SSLPEER SSLCERTI
AMQ8417: Display Channel Status details.
   CHANNEL(TO.QM2)                          CHLTYPE(RCVR)
   CONNAME(192.0.0.2)                       CURRENT
   RQMNAME(QMA)                             SSLCERTI( )
   SSLPEER( )                               STATUS(RUNNING)
   SUBSTATE(RECEIVE)                        XMITQ( )
```

QM2 で SSLPEER フィールドが空になっています。これは、QM1 が証明書を送信しなかったことを示しています。 QM1 では、SSLPEER の値が QM2 の個人証明書の DN の値と一致しています。

キュー・マネージャーへのクライアントのセキュア接続

TLS 暗号セキュリティー・プロトコルを使用するセキュア通信では、通信チャネルをセットアップし、認証に使用するデジタル証明書を管理する必要があります。

SSL/TLS インストール環境をセットアップするには、TLS を使用するようにチャネルを定義する必要があります。 また、デジタル証明書を取得し、管理することも必要です。 テスト・システムでは、自己署名証明書か、ローカル認証局 (CA) で発行された証明書を使用できます。 実動システムでは、自己署名証明書を使用しないでください。

証明書の作成と管理の詳細については、以下のトピックを参照してください。

- IBM i IBM i での SSL または TLS の取り扱い
- ALW AIX, Linux, and Windows・システムでの SSL または TLS の取り扱い
- z/OS z/OS での SSL または TLS の取り扱い

このトピック集では、SSL/TLS 通信のセットアップに関連したタスクを取り上げ、それらのタスクを実行するための段階的な手順を説明します。

また、プロトコルのオプション部分である SSL/TLS クライアント認証をテストすることもできます。 SSL/TLS ハンドシェーク中に、SSL/TLS クライアントは常にサーバーからデジタル証明書を取得し検証します。 IBM MQ の実装では、SSL/TLS サーバーは、常にクライアントから証明書を要求します。

IBM i ALW IBM i、 AIX, Linux, and Windows システムでは、 SSL/TLS クライアントが証明書を送信するのは、それが正しい IBM MQ の形式でラベルが付けられている場合に限られます。そのフォーマットは、 `ibmwebspheremq` の後に小文字でログオン・ユーザー ID が続くか、 **CERTLABL** 属性の値になるかです。 デジタル証明書ラベルを参照してください。

SSL/TLS サーバーは、クライアント証明書が送信される場合は、常にそのクライアント証明書を検証します。 クライアントが証明書を送信しない場合に認証が失敗するのは、チャネルの SSL/TLS サーバー側の定義で、**SSLCAUTH** パラメーターが REQUIRED に設定されている場合か、 **SSLPEER** パラメーターの値が設定されている場合に限られます。 匿名でのキュー・マネージャーの接続について詳しくは、113 ページの『キュー・マネージャーへのクライアントの匿名接続』を参照してください。

関連概念

IBM MQ classes for Java での TLS CipherSpec と CipherSuite

IBM MQ classes for JMS での TLS CipherSpec と CipherSuite

関連タスク
管理対象 .NET クライアント用の証明書の使用

クライアントとキュー・マネージャーの相互認証への自己署名証明書の使用

自己署名 TLS 証明書を使用してクライアントとキュー・マネージャーの間で相互認証を実装するための手順の例を取り上げます。

このタスクについて

IBM i IBM i での DCM は自己署名証明書をサポートしていないため、このタスクは IBM i システムでは適用されません。

シナリオ

- クライアント C1、およびキュー・マネージャー QM1 が存在し、これらは安全に通信する必要があります。 C1 と QM1 の間で、相互認証を実行する必要があります。
- そこで、自己署名証明書を使用して安全な通信をテストすることにしました。

構成の結果は次のようになります。

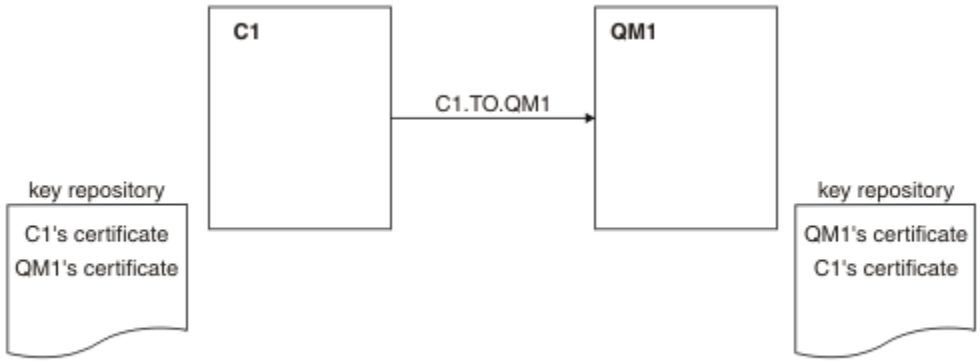


図 *19.* このタスクで実装する構成

110 ページの図 19 にあるとおり、QM1 の鍵リポジトリーには QM1 の証明書と C1 の公開証明書を格納します。 C1 の鍵リポジトリーには、C1 の証明書と QM1 の公開証明書を格納します。

手順

1. オペレーティング・システムに従って、クライアントとキュー・マネージャーで鍵リポジトリーを準備します。
   - ALW AIX, Linux, and Windows・システムの場合。
   - z/OS z/OS システムの場合 (キュー・マネージャーのみ)。
2. クライアントおよびキュー・マネージャー用の自己署名証明書を作成します。
   - ALW AIX, Linux, and Windows・システムの場合。
   - z/OS z/OS システムの場合 (キュー・マネージャーのみ)。
3. 各証明書のコピーを取り出します。
   - ALW AIX, Linux, and Windows・システムの場合。
   - z/OS z/OS・システムの場合。
4. FTP などのユーティリティーを使用して C1 証明書の公開部分を QM1 システムに転送し、またその逆を実行します。

z/OS の場合は、自己署名証明書の交換を参照してください。

5. パートナーの証明書を、クライアントとキュー・マネージャーの鍵リポジトリーに追加します。
   - ALW AIX, Linux, and Windows・システムの場合。
   - z/OS z/OS・システムの場合。
6. キュー・マネージャーでコマンド REFRESH SECURITY TYPE(SSL) を実行します。
7. 以下のいずれかの方法で、クライアント接続チャネルを定義します。
   - C1 で MQSCO 構造を指定した MQCONNX 呼び出しを使用する (MQCNO を使用した IBM MQ MQI client でのクライアント接続チャネルの作成を参照)。
   - クライアント・チャネル定義テーブルを使用します (サーバー側でのサーバー接続およびクライアント接続の定義の作成を参照)。
8. QM1 で以下の例のようなコマンドを実行して、サーバー接続チャネルを定義します。

```
DEFINE CHANNEL(C1.TO.QM1) CHLTYPE(SVRCONN) TRPTYPE(TCP) SSLCIPH(TLS_RSA_WITH_AES_128_CBC_SHA)
SSLCAUTH(REQUIRED) DESCR('Receiver channel using TLS from C1 to QM1')
```

   このチャネルの名前は、ステップ 6 で定義したクライアント接続チャネルと同じ名前でなければならず、使用する CipherSpec も同じでなければなりません。

タスクの結果

作成した鍵リポジトリーとチャネルを図にまとめたのが、110 ページの図 19 です。

次のタスク

DISPLAY コマンドを使用して、タスクが正常に完了していることを確認します。 タスクが正常に完了していれば、以下の例のような出力が表示されます。

キュー・マネージャー QM1 から以下のコマンドを入力します。

```
DISPLAY CHSTATUS(C1.TO.QM1) SSLPEER SSLCERTI
```

結果の出力は、以下の例のようになります。

```
DISPLAY CHSTATUS(C1.TO.QM1) SSLPEER SSLCERTI
     5 : DISPLAY CHSTATUS(C1.TO.QM1) SSLPEER SSLCERTI
AMQ8417: Display Channel Status details.
   CHANNEL(C1.TO.QM1)                      CHLTYPE(SVRCONN)
   CONNAME(192.0.0.1)                      CURRENT
   SSLCERTI("CN=QM1,OU=IBM MQ Development,O=IBM,ST=Hampshire,C=UK")
   SSLPEER("SERIALNUMBER=4C:D0:49:D5:02:5E:02,CN=QM2,OU=IBM MQ
Development,O=IBM,ST=Hampshire,C=UK")
   STATUS(RUNNING)                          SUBSTATE(RECEIVE)
```

オプションで、チャネル定義の **SSLPEER** フィルター属性を設定できます。 チャネル定義 **SSLPEER** が設定されている場合、その値はステップ 2 で作成したビジネスパートナー証明書の識別名と一致する必要があります。 接続が成功すると、 **DISPLAY CHSTATUS** 出力の **SSLPEER** フィールドに、リモート・クライアント証明書のサブジェクト DN が表示されます。

クライアントとキュー・マネージャーの相互認証への CA 署名証明書の使用

CA 署名 TLS 証明書を使用してクライアントとキュー・マネージャーの間で相互認証を実装するための手順の例を取り上げます。

このタスクについて

シナリオ

- クライアント C1、およびキュー・マネージャー QM1 が存在し、これらは安全に通信する必要があります。 C1 と QM1 の間で、相互認証を実行する必要があります。
- 将来的にはこのネットワークを実稼働環境で使用することを計画しているので、最初から CA 署名証明書を使用することにしました。

構成の結果は次のようになります。

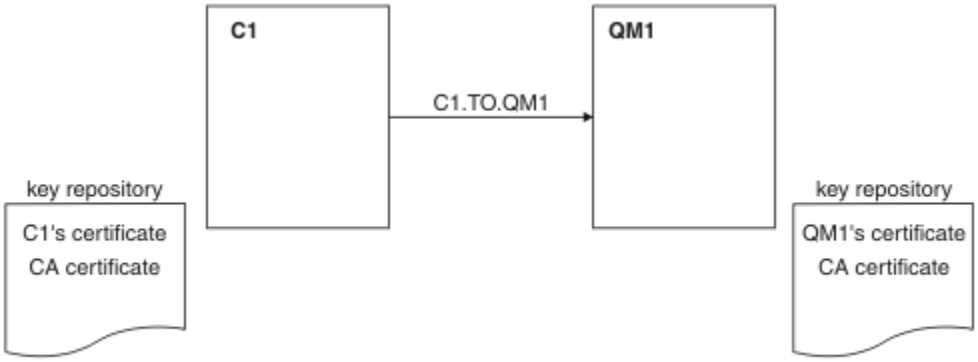


図 *20.* このタスクで実装する構成

112 ページの図 20 にあるとおり、C1 の鍵リポジトリーには、C1 の証明書と CA 証明書を格納します。 QM1 の鍵リポジトリーには、QM1 の証明書と CA 証明書を格納します。 この例では、C1 の証明書と QM1 の証明書の両方が同じ CA から発行されました。 C1 の証明書と QM1 の証明書が異なる CA から発行された場合は、C1 と QM1 の鍵リポジトリーに両方の CA 証明書が含まれている必要があります。

手順

1. オペレーティング・システムに従って、クライアントとキュー・マネージャーで鍵リポジトリーを準備します。
   - IBM i IBM i システムの場合。
   - ALW AIX, Linux, and Windows・システムの場合。
   - z/OS z/OS システムの場合 (キュー・マネージャーのみ)。
2. クライアントおよびキュー・マネージャー用の CA 署名証明書を要求します。
   クライアントとキュー・マネージャーで異なる CA を使用することがあります。
   - IBM i IBM i システムの場合。
   - ALW AIX, Linux, and Windows・システムの場合。
   - z/OS z/OS システムの場合 (キュー・マネージャーのみ)。
3. クライアントとキュー・マネージャーの鍵リポジトリーに認証局証明書を追加します。
   クライアントとキュー・マネージャーがそれぞれ異なる認証局を使用する場合は、各認証局の CA 証明書を両方の鍵リポジトリーに追加する必要があります。
   - IBM i IBM i システムでは、この手順を実行しないでください。
   - ALW AIX, Linux, and Windows・システムの場合。
   - z/OS z/OS システムの場合 (キュー・マネージャーのみ)。
4. クライアントとキュー・マネージャーの鍵リポジトリーに CA 署名証明書を取り込みます。
   - IBM i IBM i システムの場合。
   - ALW AIX, Linux, and Windows・システムの場合。

- z/OS z/OS システムの場合 (キュー・マネージャーのみ)。

5. 以下のいずれかの方法で、クライアント接続チャネルを定義します。
   - C1 で MQSCO 構造を指定した MQCONNX 呼び出しを使用する (MQCNO を使用した IBM MQ MQI client でのクライアント接続チャネルの作成を参照)。
   - クライアント・チャネル定義テーブルを使用します (サーバー側でのサーバー接続およびクライアント接続の定義の作成を参照)。
6. QM1 で以下の例のようなコマンドを実行して、サーバー接続チャネルを定義します。

```
DEFINE CHANNEL(C1.TO.QM1) CHLTYPE(SVRCONN) TRPTYPE(TCP)
SSLCIPH(TLS_RSA_WITH_AES_128_CBC_SHA) SSLCAUTH(REQUIRED)
DESCR('Receiver channel using TLS from C1 to QM1')
```

   このチャネルの名前は、ステップ 6 で定義したクライアント接続チャネルと同じ名前でなければならず、使用する CipherSpec も同じでなければなりません。

タスクの結果

作成した鍵リポジトリーとチャネルを図にまとめたのが、112 ページの図 20 です。

次のタスク

DISPLAY コマンドを使用して、タスクが正常に完了していることを確認します。 タスクが正常に完了していれば、以下の例のような出力が表示されます。

キュー・マネージャー QM1 から以下のコマンドを入力します。

```
DISPLAY CHSTATUS(TO.QMB) SSLPEER SSLCERTI
```

結果の出力は、以下の例のようになります。

```
DISPLAY CHSTATUS(C1.TO.QM1) SSLPEER SSLCERTI
     5 : DISPLAY CHSTATUS(C1.TO.QM1) SSLPEER SSLCERTI
AMQ8417: Display Channel Status details.
   CHANNEL(C1.TO.QM1)                      CHLTYPE(SVRCONN)
   CONNAME(192.0.0.1)                      CURRENT
   SSLCERTI("CN=IBM MQ CA,OU=IBM MQ Devt,O=IBM,ST=Hampshire,C=UK")
   SSLPEER("SERIALNUMBER=4C:D0:49:D5:02:5F:38,CN=QMA,OU=IBM MQ
Development,O=IBM,ST=Hampshire,C=UK")
   STATUS(RUNNING)                         SUBSTATE(RECEIVE)
```

DISPLAY CHSTATUS 出力の SSLPEER フィールドには、ステップ 2 で作成されたリモート・クライアント証明書の識別名が表示されます。 発行者名は、ステップ 4 で追加された個人証明書に署名した CA 証明書の識別名と一致します。

キュー・マネージャーへのクライアントの匿名接続

相互認証のシステムを変更して、キュー・マネージャーが別のキュー・マネージャーに匿名で接続できるようにするための手順の例を取り上げます。

このタスクについて

シナリオ

- 111 ページの『クライアントとキュー・マネージャーの相互認証への CA 署名証明書の使用』 で、キュー・マネージャーとクライアント (QM1 と C1) がセットアップされました。
- C1 が QM1 に匿名接続されるように変更する必要があるとします。

構成の結果は次のようになります。

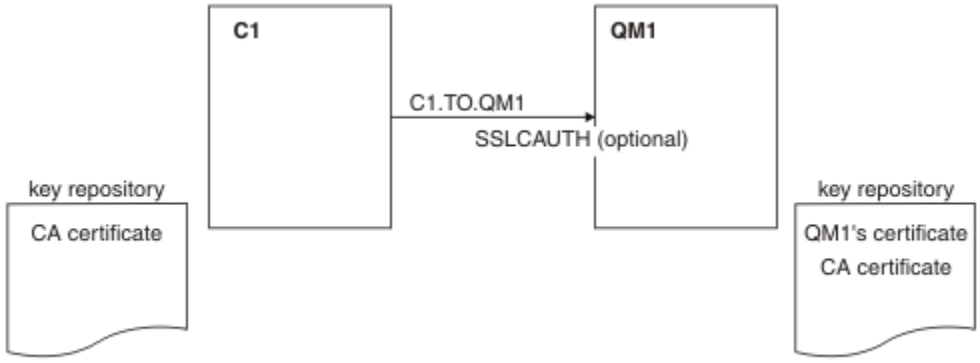


図 *21.* 匿名接続を可能にするクライアントとキュー・マネージャー

手順

1. オペレーティング・システムに従って、個人証明書を C1 の鍵リポジトリーから削除します。
   - IBM i IBM i システム
   - ALW AIX, Linux, and Windows システム

   証明書ラベルは、ibmwebspheremq の後に小文字のログオン・ユーザー ID、または **CERTLABL** 属性の値のいずれかです。 デジタル証明書ラベルを参照してください。
2. クライアント・アプリケーションを再始動します。あるいは、クライアント・アプリケーションを閉じ、すべての SSL/TLS 接続を再開します。
3. 次のコマンドを発行して、キュー・マネージャーでの匿名接続を許可します。

```
ALTER CHANNEL(C1.TO.QM1) CHLTYPE(SVRCONN) SSLCAUTH(OPTIONAL)
```

タスクの結果

変更した鍵リポジトリーとチャネルを図にまとめたのが、114 ページの図 21 です。

次のタスク

チャネルのサーバー側で、チャネル状況表示にピア名パラメーター値が表示される場合は、クライアント証明書が流れたことを意味します。

DISPLAY コマンドを使用して、タスクが正常に完了していることを確認します。 タスクが正常に完了していれば、以下の例のような出力が表示されます。

キュー・マネージャー QM1 から以下のコマンドを入力します。

```
DISPLAY CHSTATUS(C1.TO.QM1) SSLPEER SSLCERTI
```

結果の出力は、以下の例のようになります。

```
DISPLAY CHSTATUS(C1.TO.QM1) SSLPEER SSLCERTI
     5 : DISPLAY CHSTATUS(C1.TO.QM1) SSLPEER SSLCERTI
AMQ8417: Display Channel Status details.
   CHANNEL(C1.TO.QM1)                      CHLTYPE(SVRCONN)
   CONNAME(192.0.0.1)                      CURRENT
   SSLCERTI( )                             SSLPEER( )
   STATUS(RUNNING)                         SUBSTATE(RECEIVE)
```

SSLCERTI フィールドと SSLPEER フィールドは空です。 これは、C1 が証明書を送信しなかったことを示しています。

Windows Windows でのマイグレーション

このシナリオでは、IBM MQ 9.2 の既存のインストール済み環境から IBM MQ 9.3 にアップグレードしてデータをマイグレーションするための主なタスクを取り上げます。 両方のバージョンを同じ Windows 環境にインストールします。

Windows ソリューションの計画

このセクションのトピックを参照して、このシナリオで網羅される内容、ビジネスがこのシナリオに従うべき理由、およびこのシナリオで提案されるソリューションの概要を理解できます。

関連タスク

インストールの計画

Windows 前提事項

このシナリオには、サンプル IT 構成のセットアップと作業に使用するシステムについて、いくつかの前提事項があります。 それらの前提事項には、使用するオペレーティング・システムと製品のバージョン、IBM MQ のセキュリティーを構成しているかどうか、などが含まれます。

このシナリオでは、以下を想定しています。

- このシナリオ用に Windows オペレーティング・システムがある 1 台のコンピューターを使用していて、そこに最初の IBM MQ 9.2 構成をインストールしてから、IBM MQ 9.3 もインストールします。

  注: このシナリオでは、クラスター化については説明していません。 提供されている指示は、シナリオが最初に開発されたときと同じ方法でシナリオを試行するための開始点として使用できる、IBM MQ サンプル・シングル・サーバー構成をインストールするためのものです。
- 以下のバージョンの IBM MQ を使用します。
  - サンプル初期構成では、IBM MQ 9.2 を使用します。
  - マイグレーション後の構成では、IBM MQ 9.3 を使用します。
- このシナリオでは、IBM MQ のセキュリティー構成を取り上げていません。 セキュリティーが構成される場合でも、このシナリオを完了することができます。
- Windows コマンド・プロンプトとグラフィカル・ユーザー・インターフェース IBM MQ Explorer を使用して、このシナリオに説明されているタスクを実行することができます。

関連概念

移行パス

Windows ビジネスの概要

ある企業が、 Windows オペレーティング・システム上の既存の IBM MQ 9.2 IT 構成を IBM MQ 9.3 にマイグレーションしようとしています。

この企業は、以下のようなビジネス上の価値を得るために、ビジネス・ソリューションを IBM MQ 9.3 にマイグレーションすることにしました。

- IBM MQ 9.3 に用意されている新機能や更新機能の利用。
- IBM MQ 9.3 から利用できるようになった新しいタイプのリリース、つまり、継続的デリバリー・リリース (CDR) の検討。
- Windows プラットフォームで利用できる LDAP 権限のメリットの活用。

関連概念

IBM MQ のリリース・タイプとバージョン管理

関連情報

IBM MQ FAQ for Long Term Support and Continuous Delivery リリース

Windows マイグレーション・パスの選択

IBM MQ 9.2 と IBM MQ 9.3 の間でマイグレーションする場合は、使用できるマイグレーション・パスがいくつかあります。

このトピックでは、以下のマイグレーション・パスの概要を示します。

- 一段階マイグレーション (別名: スタンドアロン・マイグレーション) のパス
- 横並びマイグレーションのパス
- 段階的マイグレーションのパス

注記 : このシナリオでは、一段階マイグレーション方式と横並びマイグレーション方式についてのみ説明します。

各パスの利点と制約を考慮したうえで、どのパスがお客様の要件に最も適しているかを判断してください。

一段階マイグレーション

一段階マイグレーションでは、前のバージョンの製品のインストール済み環境が最新バージョンに置き換わり、インストール場所は同じになります。

一段階マイグレーションの利点は、前のバージョンのキュー・マネージャーの構成の変更が最小限になることです。 既存のアプリケーションは、前のバージョンのライブラリーのロードから最新バージョンのライブラリーのロードに自動的に切り替えます。 この手法を使用する場合は、このプロセス中はシステムを使用できなくなります。

横並びマイグレーション

横並びマイグレーションでは、キュー・マネージャーを旧バージョンに関連付けたままの状態で、最新バージョンの IBM MQ をインストールします。

準備ができたら、キュー・マネージャーとアプリケーションを最新バージョンにマイグレーションします。

この手法では、前のバージョンをアンインストールしてからキュー・マネージャーを開始します。したがって、最新バージョンのインストール済み環境をプライマリー・インストールとして割り当てることができます。

詳しくは、プライマリー・インストールの選択を参照してください。

段階的マイグレーション

段階的マイグレーションでは、前のバージョンに関連付けられたままの実行中のキュー・マネージャーと共存させる形で、最新バージョンの製品をインストールします。 最新バージョンのインストール済み環境を使用して、キュー・マネージャーを作成し、新規アプリケーションを実行できます。 前のバージョンのキュー・マネージャーとアプリケーションのマイグレーションを始める準備ができたら、マイグレーションを 1 つずつ行えます。 最新バージョンへのマイグレーションが完了したら、前のバージョンをアンインストールし、最新バージョンのインストール済み環境をプライマリー・インストールにします。

段階的手法では、前のバージョンをアンインストールするまでに、最新バージョンのキュー・マネージャーに接続するアプリケーションを実行するように環境を構成する必要があります。 また、IBM MQ コマンドを実行するためのパスを指定する必要もあります。 これらのタスクは両方とも、 **setmqenv** コマンドを使用して実行します。

関連概念

マイグレーションの概念や方式の概要

関連タスク

AIX and Linux でのマイグレーション: 一段階

AIX and Linux でのマイグレーション: 横並び

AIX and Linux でのマイグレーション: 段階的

関連資料

マイグレーションに影響を与える変更のリスト

Windows テクニカル・ソリューション

このシナリオでは、前のバージョンの IBM MQ から新しいバージョンにマイグレーションするための 2 つの方法を説明します。両方のバージョンが同じサーバー上の Windows オペレーティング・システム上で稼働するものとします。

Windows 概要: 初期 *IT* 構成

ある企業が、 Windows オペレーティング・システムのサーバーにインストールされている IBM MQ 9.2 によって提供される既存の IT 構成を使用しています。 このシナリオでは、その初期 IT 構成を、同じサーバー上の IBM MQ 9.3 で提供される同等の IT 構成にマイグレーションする方法を取り上げます。

117 ページの図 22 に示されているように、初期 IT 構成には、管理者が構成したり使用したりする複数のコンポーネントが含まれています。

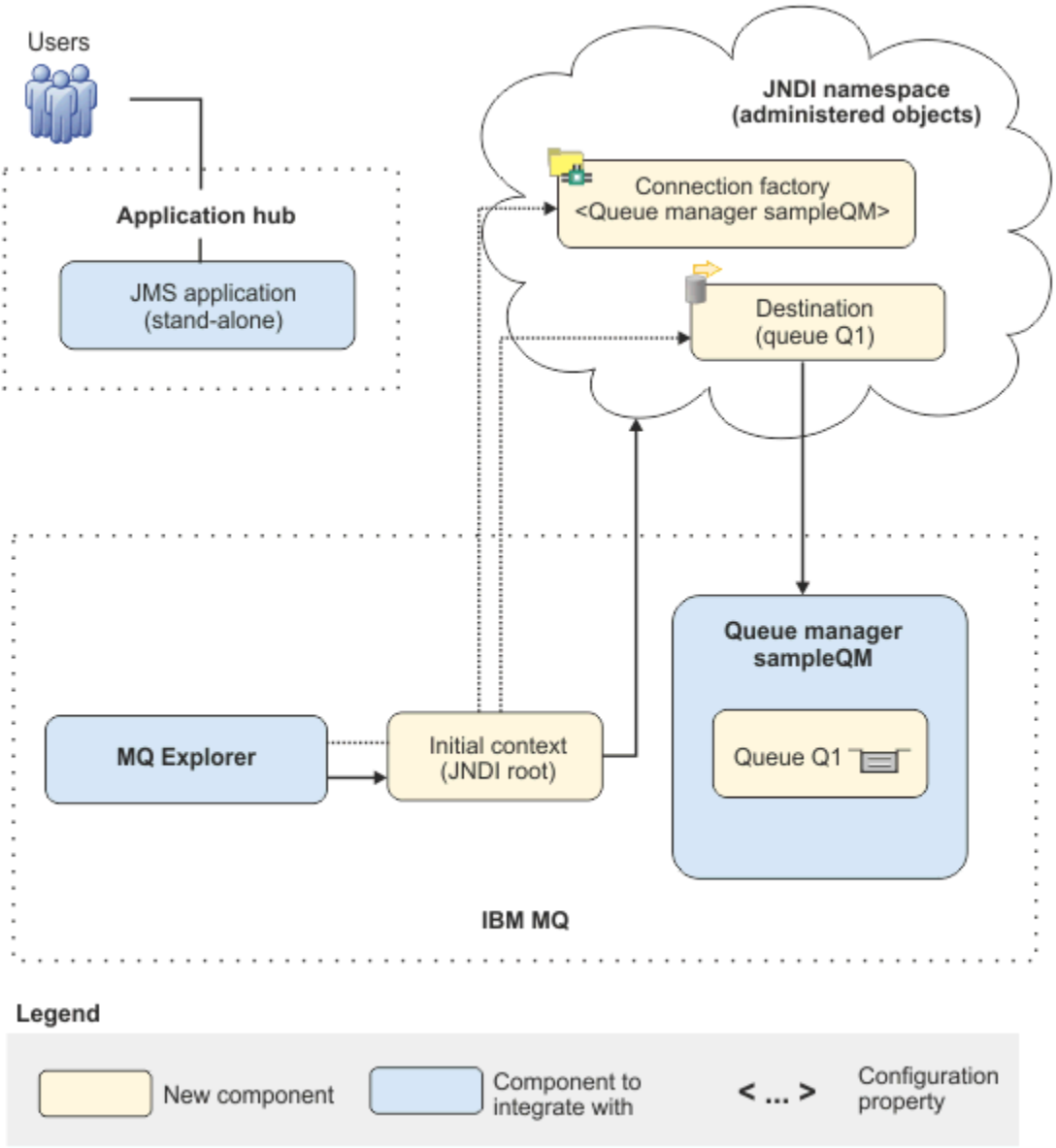


図 *22.* 初期 *IT* 構成

JMS アプリケーション
ビジネス・ユーザーが (例えばオーダーを登録するために) 対話するスタンドアロン・アプリケーション。 アプリケーションは、非同期メッセージングのために Java Message Service (JMS) を使用します。

- JMS は、広くサポートされている Java 拡張版メッセージング規格です。 したがって、JMS に基づくアプリケーションは、多くのメッセージング製品にわたって移植可能です。
- JMS は、メッセージ層の詳細から一定のレベルの抽象化を提供し、アプリケーション開発のプロセスを簡単にします。
- JMS は非同期通信を提供して、リモート・プロシージャー・コール (RPC) のような密結合のシステムとは違って、アプリケーションが応答を待機しなくても実行できるようにします。
- JMS を使用するアプリケーションは、リソースにアクセスするための詳細を直接指定しません。 その代わりに、接続ファクトリーや宛先などの管理 JMS オブジェクトを検索して使用します。

いくつかの状況では、他のメッセージング規格が JMS よりも適切な場合があります。 例えば、IBMMessage Service Clients for C、C++ 、.NET および XMS は非 Java アプリケーションと同様に JMS に利点を提供する APIs です。 そのため、.NET プラットフォームを使用する場合や、既存の C++ アプリケーションをより新しい Java EE アプリケーションと統合する場合には、XMS の方がより適しています。

アプリケーションは Point-to-Point メッセージングを使用して、メッセージをインフラストラクチャー内のキューに送信し、応答メッセージを処理してビジネス・ユーザーに適切な応答を提供します。

このメッセージング・モデルでは、1 つのアプリケーションがメッセージをキューに送り、別のアプリケーションがキューからメッセージを受け取ってメッセージの受信の確認応答を行います。 このモデルは、2 つのエンドポイントだけを含むので、最も単純なメッセージング形式です。 このモデルはまた、単一のクライアントが単一のサーバーに情報を要求するという、シナリオのサンプル・アプリケーションに最も適切です。

代わりのメッセージング・モデルであるパブリッシュ/サブスクライブでは、パブリッシャーがメッセージをメッセージ・トピックにパブリッシュします。 サブスクライバーはトピックにサブスクライブして、メッセージを受け取ります。 パブリッシャーとサブスクライバーは互いに関する情報を持ちません。そしてメッセージはゼロ以上の数の受信者によって受け取られます。

キュー・マネージャー sampleQM
初期のメッセージング・インフラストラクチャーを提供する IBM MQ キュー・マネージャー。 JMS アプリケーションが処理するキューをホストします。

Q1 [メッセージ・キュー]
JMS アプリケーションがメッセージを送信する先の IBM MQ キュー。

JNDI 名前空間
Java Naming Directory Interface JNDI 名前空間は、JMS 管理対象オブジェクトを保持するために使用されます。アプリケーションはそのオブジェクト使用して IBM MQ に接続し、メッセージを送受信するための宛先にアクセスできます。

JNDI は Java EE の一部であり、アプリケーションがさまざまなタイプのネーミング・サービスやディレクトリー・サービスにアクセスして、アプリケーション・コンポーネントを取得するための標準方式を提供します。 例えば、JNDI を使用してファイル・システム上のネーミング・サービスにアクセスし、プリンター・オブジェクトの場所を取得したり、LDAP サーバー上のディレクトリー・サービスにアクセスして ID とパスワード情報を格納するユーザー・オブジェクトを取得したりすることができます。 したがって、JNDI により JMS ベースのアプリケーションの移植性が強化され、それらのアプリケーションを互いに統合することや既存のシステムと統合することが容易になります。 JMS メッセージングでは JNDI を使用して、メッセージのターゲットの宛先を表すオブジェクトやアプリケーションとそのメッセージングの宛先との間に接続を作成する接続ファクトリーを表すオブジェクトを保管します。

JNDI 名前空間へのアクセス権限を持つすべてのアプリケーションやプロセスは、同じ管理対象オブジェクトを使用できます。 JNDI では管理対象オブジェクトのプロパティーを変更することができ、すべてのアプリケーションやプロセスはそれらの変更の利点を同じように活用できます。

初期コンテキスト
初期コンテキストは、JNDI 名前空間のルートを定義します。 IBM MQ Explorer を使用して管理対象オブジェクトを作成および構成するには、まず JNDI 名前空間のルートを定義する初期コンテキストを追加します。 同様に、JMS アプリケーションは、まず初期コンテキストを取得した後に、JNDI 名前空間から管理対象オブジェクトを取得できるようになります。

接続ファクトリー、myCF
JMS 接続ファクトリー・オブジェクトは、接続の標準的な構成プロパティーのセットを定義します。 アプリケーションは、接続ファクトリーを使用して IBM MQ への接続を作成します。

宛先、myQueue
JMS の宛先には、トピックまたはキューを使用できます。 このシナリオでは、宛先はキューであり、アプリケーションのメッセージ送信先、アプリケーションのメッセージ受信元、またはその両方に使用する IBM MQ キューを示します。 アプリケーションは、JNDI 名前空間で宛先を検索し、IBM MQ キューへの接続を作成します。

Windows 概要: 実現する論理トポロジー

この会社は、IBM MQ 9.2 から IBM MQ 9.3 にマイグレーションしました。

119 ページの図 23 にあるとおり、マイグレーション後の IT 構成は変わっていません。 キュー・マネージャーとサンプル・アプリケーションをマイグレーションした結果、この会社は IBM MQ の最新バージョンの新機能を利用できるようになりました。

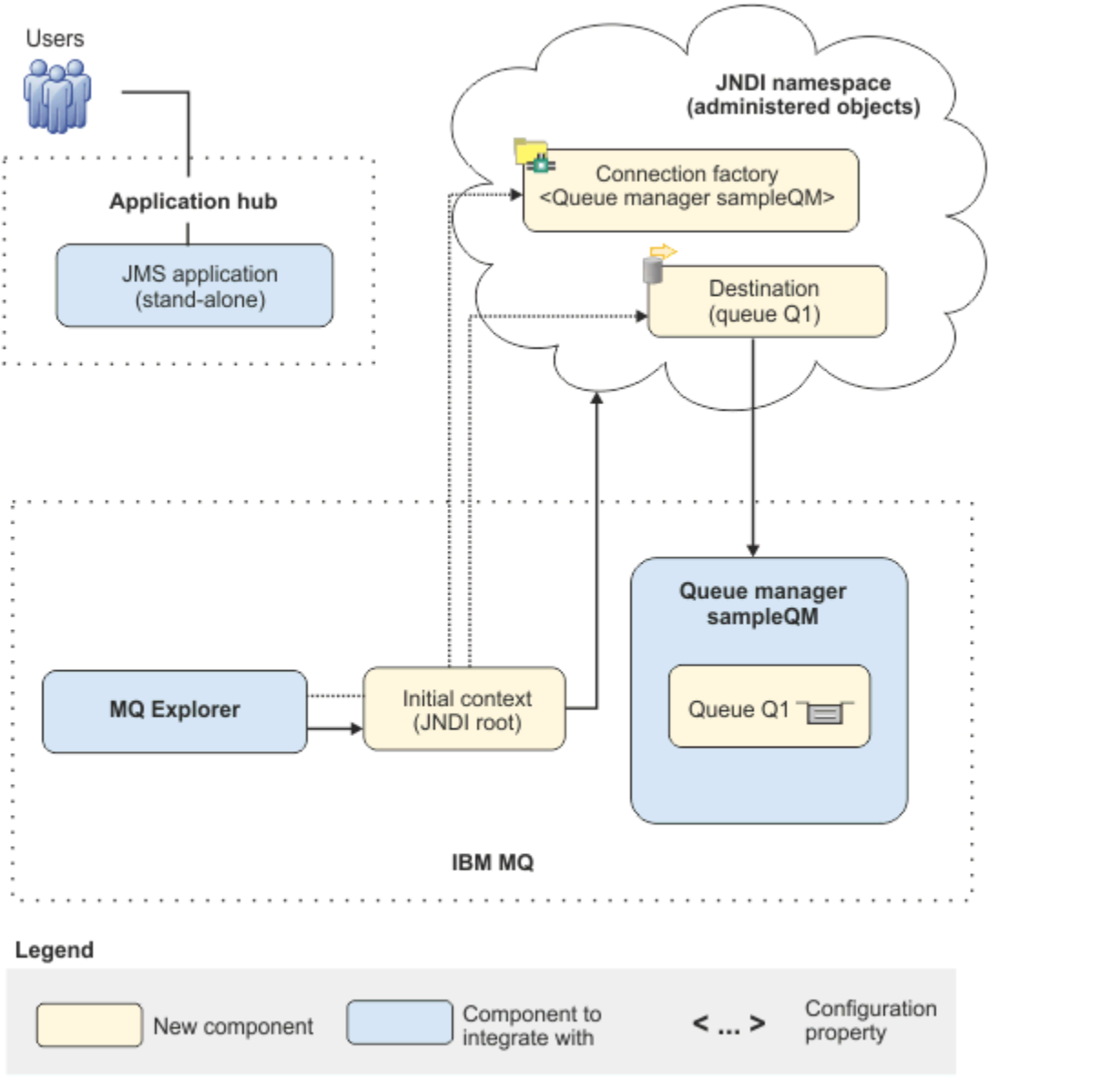


図 *23.* デリバリー済みの *IT* 構成

Windows グラフィカル・ユーザー・インターフェースを使用したソリューションの実装

このシナリオでのソリューションの実装には、グラフィカル・ユーザー・インターフェースを使用して、Windows オペレーティング・システム上で実行されている以前のリリースの IBM MQ から、同じく Windows オペレーティング・システム上で実行されている新しいリリースにマイグレーションし、キュー・マネージャーとキューが新しいリリースに正常にマイグレーションされたことを確認することが含まれます。

始める前に

シナリオを試す場合は、121 ページの『初期 IT 構成の作成』で説明されているように、 まずサンプル・メッセージング・インフラストラクチャーのコピーをセットアップするための指示に従います。 このサンプル構成は、IBM MQ 9.2 に基づいています。

このタスクについて

このシナリオの移行プロセスは、IBM MQ 9.2 から IBM MQ 9.3 へのキュー・マネージャーの移行について記述しています。

シナリオには、マイグレーション方式の 2 つのオプションが含まれています。 これらのオプションのいずれか、または両方を選択して試すことができます。

オプション 1: 一段階マイグレーション

一段階マイグレーションでは、前のバージョンの製品のインストール済み環境が新しいバージョンに置き換わり、インストール場所は同じになります。

一段階マイグレーションの利点は、前のバージョンのキュー・マネージャーの構成の変更が最小限になることです。 既存のアプリケーションは、前のバージョンのライブラリーのロードから最新バージョンのライブラリーのロードに自動的に切り替えます。 この手法を使用する場合は、このプロセス中はシステムを使用できなくなります。

オプション 2: 横並びマイグレーション

横並びマイグレーションでは、新しいバージョンの IBM MQ を以前のバージョンと共存するようにインストールします。 キュー・マネージャーは、新しいバージョンにマイグレーションする準備ができるまで、前のバージョンと関連付けられたままになります。

横並び手法では、マイグレーションされたキュー・マネージャーを新しいバージョンで開始する前に前のバージョンをアンインストールので、新しいバージョンのインストール済み環境をプライマリー・インストールとして割り当てることができます。

手順

1. 121 ページの『初期 IT 構成の作成』で説明されているように、シナリオの開始点として使用する初期 IT 構成のサンプルを作成します。
2. 製品のマイグレーションに使用する方式を選択してから、選択したオプションのための指示に従います。
   - 130 ページの『オプション 1: 一段階マイグレーション』
   - 137 ページの『オプション 2: 横並びマイグレーション』

関連タスク

AIX and Linux でのマイグレーション: 一段階
AIX and Linux でのマイグレーション: 横並び
Windows での旧バージョンのキュー・マネージャーから最新バージョンへのマイグレーション
プライマリー・インストールの選択

Windows 初期 IT 構成の作成

このシナリオは、サンプル初期 (IT) 構成を使用して作成されています。 このサンプル構成をセットアップして、当初作成されたのと同様にこのシナリオを試すには、以下の手順に従ってください。

このタスクについて

このシナリオで使用する初期 IT 構成 (117 ページの『概要: 初期 IT 構成』を参照) には、IBM MQ Explorer が JNDI 名前空間のルートに接続するために追加された初期コンテキストが含まれます。 JNDI 名前空間には、 IBM MQ への接続に使用するサンプルの JMS アプリケーション用に追加された接続ファクトリーと、IBM MQ キューに接続するためにサンプルの JMS アプリケーション用に追加された出力先が含まれています。 その IBM MQ キューは初期 IT 構成にも追加され、サンプルの JMS アプリケーションによって使用されます。

手順

1. IBM MQ 9.2 をインストールして、インストール済み環境を検証します。
2. JNDI 名前空間と管理対象オブジェクトを構成します。
3. サンプル IT 構成を検証します。

Windows ランチパッドを使用した *IBM MQ 9.3* のインストール

このシナリオの開始点として使用する初期 IT 構成としてセットアップする IBM MQ のバージョンをインストールするには、インストール・ランチパッドとウィザードを使用します。

始める前に

このタスクを始める前に、以下の確認を行ってください。

- インストール時に、ローカル管理者権限が必要です。
- マシン名にスペースが含まれていないことを確認します。
- 十分なディスク・スペースがあることを確認します。 詳しくは、ディスク・スペース要件 (Multiplatforms) を参照してください。

このシナリオでは、 IBM MQ ユーザーの Windows ドメイン・ユーザー ID を定義する必要があるかどうかを判別する必要はありません。この要件は、このシナリオの範囲外であるためです。 詳細については、IBM MQ 用の Active Directory および DNS ドメインの作成を参照してください。

IBM MQ をインストールする前に、システムがハードウェアとソフトウェアの要件を満たしていることを確認してください。 ハードウェア要件とソフトウェア要件の最新の詳細情報については、IBM MQ のシステム要件を参照してください。

このタスクについて

このタスクでは、 Windows オペレーティング・システムに IBM MQ をインストールするための基本的なステップについて説明します。

インストール・プログラムには、詳細情報へのリンクが含まれています。 インストール・プロセスは、次の部分からなります。

1. インストール処理を開始します。
2. インストール・ランチパッドを使用して、ソフトウェア要件を検査してインストールし、ネットワーク情報を指定し、IBM MQ インストール・ウィザードを開始します。
3. IBM MQ インストール・ウィザードを使用してソフトウェアをインストールし、Prepare IBM MQ Wizard を開始します。
4. Prepare IBM MQ Wizard を使用して、IBM MQ サービスを開始します。

手順

1. インストール処理を開始します。

   Windows エクスプローラーで、インストール・イメージをダウンロードした一時フォルダーにナビゲートし、 `setup.exe` をダブルクリックします。

   インストール・ランチパッドが開始します。
2. ランチパッドを使用して、ソフトウェア要件およびネットワーク構成を確認し、必要であれば変更します。
   a)「**ソフトウェア要件**」ボタンをクリックして、「**ソフトウェア要件**」タブを表示します。
   b) ソフトウェア要件が満たされていること、および要件の項目に OK という語を含む緑色のチェック・マークが表示されていることを確認します。 指示されている訂正を行います。

      注 : 要件に関するより詳細を表示するには、プラス (+) ボタンをクリックします。
   c)「**ネットワーク構成**」ボタンをクリックし、「**ネットワーク構成**」タブを表示します。
   d)「**いいえ**」ラジオ・ボタンをクリックします。

      注 : このシナリオは、IBM MQ 用にドメイン・ユーザー ID を構成する必要がないことを想定しています。 IBM MQ for Windows ドメイン・ユーザーの構成について詳しくは、「**詳細情報**」ボタンをクリックしてください。
   e) ランチパッドの「**IBM MQ インストール**」タブで、インストール言語を選択してから、「**IBM MQ インストーラーの起動**」ボタンをクリックして、IBM MQ インストール・ウィザードを開始します。

   IBM MQ のインストール要件の確認を完了し、必要な変更を行い、IBM MQ インストール・ウィザードを開始しました。
3. IBM MQ インストール・ウィザードを使用してソフトウェアをインストールし、Prepare IBM MQ Wizard を開始します。
   a) IBM MQ インストール・ウィザードで、ご使用条件を読み、「**使用条件の条項に同意します**」チェック・ボックスを選択して、「**次へ**」をクリックします。
   b)「**標準**」をクリックしてから、「**次へ**」をクリックします。
   c)「**IBM MQ をインストールする準備ができました**」ページで、インストール情報を確認し、「**インストール**」をクリックします。

      注 : 以下の詳細に注意してください。

      - インストール環境の名前
      - プログラム・ファイルの最上位フォルダー
      - データ・ファイルの最上位フォルダー

      次の機能がインストールされます。

      - IBM MQ Server
      - IBM MQ: IBM MQ リソースを管理およびモニターするためのグラフィカル・インターフェース
      - Java™、 .NET メッセージングおよび Web サービス
      - IBM MQ 開発ツールキット

      インストール・プロセスが開始します。 ご使用のシステムによっては、インストール・プロセスに数分を要する場合があります。

      インストール・プロセスの最後に、 IBM MQ セットアップ・ウィンドウに `Installation Wizard Completed Successfully` というメッセージが表示されます。
   d)「**完了 (Finish)**」 をクリックします。

   IBM MQ が正常にインストールされました。 Prepare IBM MQ Wizard が自動的に開始し、「**Prepare IBM MQ Wizard にようこそ**」ページが表示されます。
4. MQ 準備ウィザードを使用して、IBM MQ サービスを開始します。
   a)「Prepare IBM MQ Wizard にようこそ」ページで、「**次へ**」を選択します。

Prepare IBM MQ Wizard には、メッセージ `Status: Checking IBM MQ Configuration` と進行状況表示バーが表示されます。 処理が完了すると、IBM MQ ネットワーク構成ページが表示されます。

b) Prepare IBM MQ Wizard の「IBM MQ ネットワーク構成」ページで、 「**いいえ**」を選択します。

c) **次へ** をクリックします。

Prepare IBM MQ Wizard には、メッセージ `Status: starting the IBM MQ Service` と進行状況表示バーが表示されます。 処理が完了すると、ウィザードに「Prepare IBM MQ Wizard の完了」ページが表示されます。

d)「**IBM MQ Explorer を起動する**」を選択し、メモ帳を起動してリリース・ノートを表示するかどうかを選択して、「**完了**」をクリックします。

IBM MQ Explorer が始動します。

IBM MQ のインストールが完了し、IBM MQ Explorer が開始しました。

タスクの結果

IBM MQ がコンピューターにインストールされます。

次のタスク

このシナリオで使用される管理対象オブジェクトを、123 ページの『JNDI 名前空間と管理対象オブジェクトの構成』の説明に従って作成できる状態になりました。

関連概念

Windows システムでのハードウェア要件とソフトウェア要件

IBM MQ の概要

関連タスク

Windows での IBM MQ サーバーのインストール

IBM MQ サーバーの構成

Windows *JNDI* 名前空間と管理対象オブジェクトの構成

JNDI 名前空間の初期コンテキストを IBM MQ Explorer で定義してから、サンプル・アプリケーションで使用できる管理対象オブジェクトを名前空間内に定義します。

このタスクについて

このタスクでは、IBM MQ で以下のオブジェクトを作成します。

- ローカル・ファイル・システムにある JNDI 名前空間。 サンプル・シナリオ向けの最も単純な JNDI メカニズムであることから、ファイル・システムを使用します。

  JNDI 名前空間は、ファイル・システム、Lightweight Directory Access Protocol (LDAP) サーバー、または別の JNDI 実装環境に配置できます。 LDAP サーバーまたは別の JNDI 実装環境で JNDI 名前空間を使用する場合、JNDI 名前空間を構成し、実装環境の要件に応じて JNDI 名前空間を参照するサンプル・アプリケーションを変更する必要があります。
- JNDI 名前空間内の管理対象オブジェクト。 JMS アプリケーションは、管理対象オブジェクトを検索して IBM MQ に接続し、メッセージを送受信する IBM MQ 宛先にアクセスします。

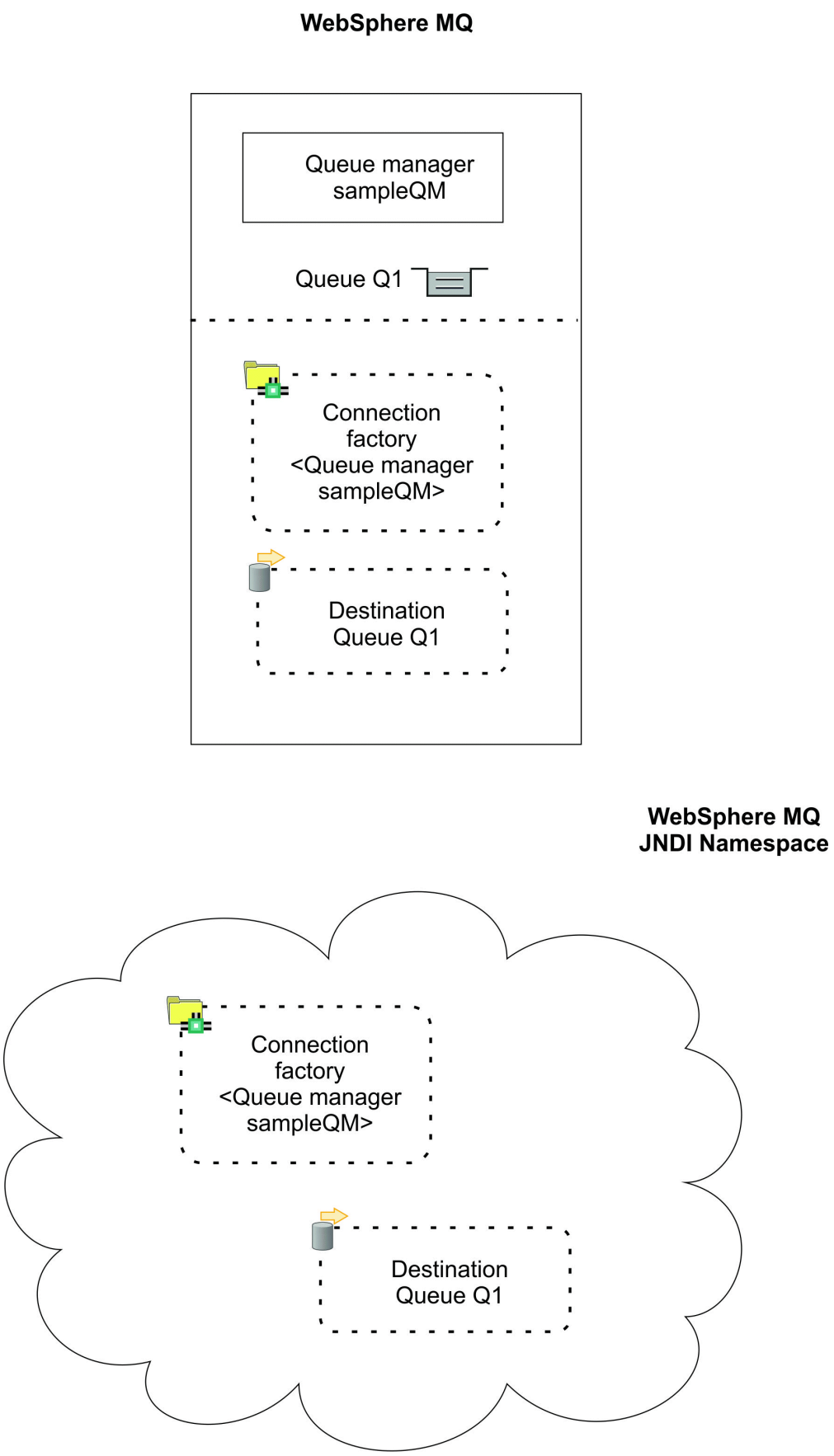


図 *24. IBM MQ* で作成されるオブジェクト

手順

1. 「**スタート**」>「**すべてのプログラム**」> **IBM MQ** > **IBM MQ Explorer** をクリックして、IBM MQ Explorer を開始します (まだ開始していない場合)。

   IBM MQ Explorer が実行されていて、ウェルカム・ページが表示されている場合は、ウェルカム・ページを閉じて IBM MQ オブジェクトの管理を開始します。
2. サンプル・アプリケーションに使用するキュー・マネージャーを作成します。

   a)「**キュー・マネージャー**」を右クリックして、「**新規**」 >「**キュー・マネージャー...**」を選択します。

   「キュー・マネージャーの作成」ウィザードが開始します。

   b)「**キュー・マネージャー名**」フィールドに、`sampleQM` と入力します。

   キュー・マネージャーには別の名前を選択できますが、後の構成ステップでは、`sampleQM` の代わりに必ずその名前を使用するようにしてください。

   注 : 名前は 48 文字以下で、次の文字セットを使用する必要があります。

   - 大文字または小文字 (A から Z、a から z)
   - 数字 (0 から 9)
   - ピリオド (.)
   - スラッシュ (/)
   - 下線 (_)
   - パーセント記号 (%)

   名前は大文字小文字を区別します。 同じタイプのオブジェクトには、別の名前を使用する必要があります。 例えば、2 つのキューを同じ名前にすることはできませんが、キュー・マネージャーとキューを同じ名前にすることはできます。

   c)「**送達不能キュー**」フィールドに、`SYSTEM.DEAD.LETTER.QUEUE` と入力します。

   このフィールドは、キュー・マネージャーを作成するときに自動的に作成される送達不能キューの名前です。

   送達不能キューには、(例えば、キューがいっぱいであるために) 正しい宛先へ配信できないメッセージが保管されます。 すべてのキュー・マネージャーには、関連付けられた送達不能キューがなければなりません。

   d) 他のフィールドを空のままにして「**完了**」をクリックするか、 このボタンが使用不可になっている場合は「**次へ**」をクリックします。

   ポート番号が既存のキュー・マネージャー (デフォルト構成の一部として作成されたキュー・マネージャーなど) と競合する場合は、「**完了**」ボタンが使用不可になります。 ウィザードで先に進んでデフォルトのポート番号を変更する必要があります。

   e)「**次へ**」をクリックした場合は、デフォルトを受け入れて、 各ページで「**次へ**」をクリックします。 ウィザードの最後のページに達すると、「**完了**」ボタンが使用可能になります。 指定のポート番号を 1415 などに変更し、「**完了**」をクリックします。

   IBM MQ でダイアログ・ウィンドウが表示され、キュー・マネージャーが作成および開始されます。
3. *JNDI* 名前空間 に初期コンテキストを追加してから、IBM MQ Explorer をそのコンテキストに接続します。

   IBM MQ Explorer を使用して JMS 管理対象オブジェクトを作成および構成するためには、その前に初期コンテキストを追加して、管理対象オブジェクトが保管される JNDI 名前空間のルートを定義する必要があります。

   IBM MQ Explorer を使用して JNDI 名前空間で管理対象オブジェクトを作成または管理する場合は、必ず、IBM MQ Explorer を JNDI 名前空間の初期コンテキストに接続する必要があります。

   a)「IBM MQ Explorer - **ナビゲーター**」ペインで、「**JMS 管理対象オブジェクト**」を右クリックして、「**初期コンテキストの追加...**」を選択します。

   このアクションは " **接続の詳細**" というページを表示します

   b) " 「**JNDI 名前空間の場所**」 "の下で、 「**ファイル・システム**」 チェック・ボックスを選択します。

c)「**バインディング・ディレクトリー**」 フィールドに、 C:\JNDI-Directory と入力します。

この値は、サンプル JMS アプリケーションで指定した JNDI 名前空間の場所と一致します。 別の JNDI ディレクトリーを指定する必要がある場合は、一致するようにアプリケーションを変更してください。

ディレクトリーがシステム上に存在しない場合は、ウィンドウに「Specified location does not exist or is not readable」というメッセージが表示されます。「**参照 ...**」 をクリックします。 ファイル・システム・ウィンドウを開くには、 Local Disk (C:)にナビゲートし、「**新規フォルダーの作成**」 をクリックして JNDI-Directory フォルダーを作成します。「**OK**」をクリックします。

次へ をクリックします。

d)「**ユーザー設定**」ページでは、デフォルト設定のままにします。

- **コンテキスト・ニックネーム**: IBM MQ Explorer に初期コンテキストを表示するためのニックネームとして、JNDI 名前空間の場所が使用されます。
- **終了時に即時に接続**: このオプションにより、初期コンテキストの作成完了時に IBM MQ Explorer が JNDI 名前空間に接続されるので、管理対象オブジェクトを即時に作成できます。
- **開始時に自動的にコンテキストに再接続**: このオプションは選択されていません。 これは、通常、IBM MQ Explorer を閉じて再度開くたびに IBM MQ Explorer を自動的に初期コンテキストに再接続する必要がないためです。

日常的に IBM MQ Explorer を使用して JNDI 名前空間で管理対象オブジェクトを作成または管理する場合は、「**開始時に自動的にコンテキストに再接続**」チェック・ボックスを選択すると、IBM MQ Explorer を開始するたびに IBM MQ Explorer が自動的に初期コンテキストに再接続されます。 このオプションを使用すると、手動で IBM MQ Explorer を初期コンテキストに接続する手間が省けます。

「**完了**」をクリックして、初期コンテキストを作成して表示します。

4. 接続ファクトリー管理対象オブジェクトを作成します。

接続ファクトリー管理対象オブジェクトは、接続の標準的な構成プロパティーのセットを定義します。アプリケーションは、接続ファクトリーを使用して IBM MQ への接続を作成します。

a)「IBM MQ Explorer - **ナビゲーター**」ペインで、「**JMS 管理対象オブジェクト**」を展開してから、**file:/C:/JNDI-Directory/** というラベルが付いた初期コンテキストを展開します。

b)「**接続ファクトリー**」を右クリックしてから、「**新規**」 > 「**接続ファクトリー...**」を選択します。

このアクションにより、「**新規接続ファクトリー**」ウィザードが表示されます。

c)「名前」フィールドに myCF と入力します。

サンプル JMS アプリケーションには、myCF という名前の接続ファクトリーを検索するコードが含まれています。 別の名前を使用する必要がある場合は、一致するようにアプリケーションを変更してください。

サンプル・アプリケーションでは Point-to-Point メッセージングが使用されるため、IBM MQ がメッセージング・プロバイダーに使用されます。

次へ をクリックします。

d) 接続ファクトリーのタイプは「**接続ファクトリー**」のままにします。 JMS の一般的な用途としては、このオプションが最も柔軟であるためです。

ドメインに依存しない接続ファクトリーを使用すると、JMS アプリケーションで Point-to-Point メッセージングとパブリッシュ/サブスクライブ・メッセージングの両方を使用できるようになります。これは特に、JMS アプリケーションを使用して、同じトランザクションのもとで両方のタイプのメッセージングを実行する場合に役立ちます。

JMS アプリケーションが Point-to-Point メッセージングまたはパブリッシュ/サブスクライブ・メッセージングの一方のみを使用する設計になっている場合は、接続ファクトリーの作成時に特定のメッセージング・ドメインを選択すると、ドメイン固有の (キューまたはトピック) 接続ファクトリーが作成されます。

e) XA トランザクションのサポートは、クリアしたままにしておきます。

サンプル・アプリケーションでは、XA 準拠のトランザクションは使用しません。

IBM MQ JMS は、バインディング・モードでの XA 準拠のトランザクションをサポートしています。サンプル・アプリケーションで XA 準拠のトランザクションを使用する必要がある場合は、サンプル・アプリケーションを変更してください。

次へ をクリックします。

f) トランスポートは「**バインディング**」のままにしておきます。

接続ファクトリーを使用するサンプル JMS アプリケーションはキュー・マネージャーと同じコンピューター上で実行されるため、「バインディング」モードのトランスポートを使用できます。 このオプションは、JMS アプリケーションがキュー・マネージャーに直接接続されることを意味し、代替クライアント・モードよりもパフォーマンス上の利点があります。

「**次へ**」をクリックし、再度「**次へ**」をクリックします。

g)「**プロパティーの変更**」ページで、左側のメニューから「**接続**」を選択します。 次に、「**接続**」ペインで、sampleQM を「**基本キュー・マネージャー**」として選択します。

基本キュー・マネージャーとは、アプリケーションの接続先となるキュー・マネージャーです。 アプリケーションが複数のキュー・マネージャーに接続できるようにする場合は、この値をブランクのままにします。

h)「**完了 (Finish)**」 をクリックします。

IBM MQ に、オブジェクトが正常に作成されたことを示すダイアログ・ウィンドウが表示されます。「**OK**」をクリックして、ダイアログ・ウィンドウを閉じます。

5. 宛先の管理対象オブジェクトを作成します。

宛先の管理対象オブジェクトは、アプリケーションが送信するメッセージの送信先、またはアプリケーションが受信するメッセージの送信元、あるいはその両方の IBM MQ キューを識別します。 アプリケーションは、JNDI 名前空間で宛先を検索し、IBM MQ キューへの接続を作成します。

パブリッシュ/サブスクライブ・メッセージングでは、宛先は、キューではなくトピックを識別します。

a)「IBM MQ Explorer - **ナビゲーター**」ペインで、「**JMS 管理対象オブジェクト**」を展開してから、**file:/C:/JNDI-Directory/** というラベルが付いた初期コンテキストを展開します。

b)「**宛先**」を右クリックしてから、「**新規**」 >「**宛先...**」を選択します。

「**新規宛先**」ウィザードが表示されます。

c) **名前**フィールドに、myQueue と入力します。

「**タイプ**」は「**キュー**」のままにしておきます。

サンプル JMS アプリケーションには、myQueue という名前の宛先を検索するコードが含まれています。 サンプル JMS アプリケーションでは Point-to-Point メッセージングが使用されるため、キューというタイプの宛先が必要です。 トピックというタイプの宛先は、パブリッシュ/サブスクライブ・メッセージングに使用されます。

d)「**マッチングする MQ キューを作成するためのウィザードを開始します**」チェック・ボックスを選択します。

宛先オブジェクトには、マッチングする IBM MQ キューが必要であるため、IBM MQ Explorer を使用して両方を一緒に作成すると便利です。「**新規宛先**」ウィザードが完了すると、「**MQ キューの作成**」ウィザードが開いて、宛先の詳細情報の多くが IBM MQ キューにマップされます。

次へ をクリックします。

再度「**次へ**」をクリックします。

e) "「**プロパティーの変更**」" ページで、「**選択 ...**」 をクリックします。 **キュー・マネージャー**の横に表示されます。 前に作成した sampleQM キュー・マネージャーを選択して、「**OK**」をクリックします。

f) IBM MQ キューの名前として Q1 を指定します。

キューには別の名前を選択できますが、後の構成ステップでは、Q1 の代わりに必ずその名前を使用するようにしてください。

注 : 名前は 48 文字以下で、次の文字セットを使用する必要があります。

- 大文字または小文字 (A から Z、a から z)

- 数字 (0 から 9)
- ピリオド (.)
- スラッシュ (/)
- 下線 (_)
- パーセント記号 (%)

名前は大文字小文字を区別します。 同じタイプのオブジェクトには、別の名前を使用する必要があります。 例えば、2 つのキューを同じ名前にすることはできませんが、キュー・マネージャーとキューを同じ名前にすることはできます。

g)「**完了 (Finish)**」 をクリックします。

「**MQ キューの作成**」ウィザードが開始します。

ウィザードが開始しない場合は、前のステップで「**マッチングする MQ キューを作成するためのウィザードを開始します**」チェック・ボックスが選択されなかった可能性があります。「IBM MQ Explorer - **ナビゲーター**」ペインで、「**sampleQM**」キュー・マネージャーを展開し、「**キュー**」を右クリックしてから、「**新規**」 > 「**ローカル・キュー...**」を選択します。

6. マッチングする IBM MQ キューを作成します。

前に作成した宛先の管理対象オブジェクトは、IBM MQ キューです。 このキューに JMS メッセージが保管されます。

a)「**次へ**」をクリックして、前に指定した sampleQM キュー・マネージャーを受け入れます。

b) **次へ** をクリックします。

c)「**完了**」をクリックし、前に作成した宛先の管理対象オブジェクトからの情報を使用して IBM MQ キューを作成します。

IBM MQ にダイアログ・ウィンドウが表示され、オブジェクトが正常に作成されたことを示すメッセージが表示されます。

これで、新規キューがキュー・マネージャーの下の「**キュー**」セクションに表示されるようになりました。

タスクの結果

これで、サンプル JMS アプリケーションを使用するために必要な IBM MQ オブジェクトが作成されました。

次のタスク

これで、サンプル・アプリケーションで使用する IBM MQ が正しく構成されたことを 128 ページの『サンプル IT 構成の検証』 の説明に従って検証する準備ができました。

Windows サンプル *IT* 構成の検証

サンプルのスタンドアロン JMS アプリケーションを実行して IBM MQ を介してメッセージを送受信し、サンプル・アプリケーションで使用する IBM MQ が正しく構成されていることを検証します。

始める前に

サンプル・アプリケーション・パッケージをダウンロードします。 次のリンクをクリックし、IBM MQ をインストールするコンピューターにファイルを保存します: sampleJMSApp.zip。 その後、内容を抽出します。 パッケージには、サンプル JMS アプリケーション `.jar` ファイルおよびアプリケーションを実行するためのバッチ・ファイルが含まれています。

- サンプルの `sampleJMSApp.jar` ファイルと `.cmd` ファイルは、同じディレクトリーに存在しなければなりません。
- `.cmd` というファイルは環境変数を使用して、JMS アプリケーションを実行するためのクラスパスを設定します。 JMS アプリケーションを実行するとき、Java `java.lang.NoClassDefFoundError` が表示される場合は、 コマンド・ファイル内のクラスパスの行を調整することが必要である可能性があります。

このタスクについて

JMS アプリケーションは、最初のメッセージを送信する要求側クライアント、 およびメッセージを受信して応答を送信する応答側クライアントから構成されます。 提供されるバッチ・ファイルは、以下の操作を実行します。

- runresponder.cmd は、応答側クライアントが開始したコマンド・プロンプト・ウィンドウを開いてから、メッセージを待機します。
- runrequester.cmd は、リクエスター・クライアントが開始した別のコマンド・プロンプト・ウィンドウを開いてから、要求メッセージを送信し、応答を受信します。

2 つのコマンド・プロンプト・ウィンドウを表示すると、要求側と応答側のアクションを個別にはっきりと確認することができます。

手順

1. runresponder.cmd というファイルをダブルクリックしてください。

   「**応答側ウィンドウ (Responder window)**」というラベルのあるコマンド・プロンプト・ウィンドウで、応答側クライアントが開始してメッセージを待機します。

   ```
   > Connection factory located in JNDI.> Destination located in JNDI.> Creating connection to
   QueueManager.> Created connection.
   > Waiting for message.
   ```

2. runrequester.cmd というファイルをダブルクリックしてください。

   「**要求側ウィンドウ (Requester window)**」で、要求側メッセージを監視します。 「**応答側ウィンドウ (Responder window)**」で、更新される応答側メッセージ (要求側クライアントから受け取るメッセージ、および送信する応答メッセージ) を監視します。

タスクの結果

「**要求側ウィンドウ (Requester window)**」というラベルのあるコマンド・プロンプト・ウィンドウで、要求側クライアントは接続状況、送信したメッセージ、そして応答側クライアントから受け取る応答メッセージを表示します。

```
> Connection factory located in JNDI.> Destination located in JNDI.> Creating connection to
QueueManager.> Connection created.
> Sending stock request for 'BakedBeans'> Sent Message
ID=ID:414d5120514d5f4c33344c3238482020c3cd094d20002b02
> Received Message ID=ID:414d5120514d5f4c33344c3238482020c3cd094d20002902 for 'B
akedBeans - 15 tins in stock'
> Closing connection to QueueManager.> Closed Connection.
------------------------------------------------------
In this window, observe the messages sent through IBM MQ:
- The request message sent
- The reply message received
-----
When ready, press any key to close this window
Press any key to continue . . .
```

「**応答側ウィンドウ (Responder window)**」で、更新される応答側メッセージ (要求側クライアントから受け取るメッセージ、および送信する応答メッセージ) を監視します。

```
> Connection factory located in JNDI.> Destination located in JNDI.> Creating connection to
QueueManager.> Created connection.
> Waiting for message.

> Received Message ID=ID:414d5120514d5f4c33344c3238482020c3cd094d20002b02 for 'B
akedBeans'
> Sending Reply Message 'BakedBeans - 15 tins in stock'> Sent Message
ID=ID:414d5120514d5f4c33344c3238482020c3cd094d20002902
> Closing connection to QueueManager.> Closed connection.
------------------------------------------------------
In this window, observe the updated responder messages
- The request message received (from the requester)
- The reply message sent
-----
```

```
When ready, press any key to close this window
Press any key to continue . . .
```

2 つのコマンド・ウィンドウに表示されるメッセージによって、サンプル・アプリケーションの要求側クライアントと応答側クライアントが IBM MQ を介して相互に通信可能であることが検証されます。

次のタスク

これで、以下の 2 つのマイグレーション・オプションのいずれかを使用して、サンプルの IBM MQ 9.2 インストール済み環境を新しいリリースの IBM MQ にマイグレーションする作業を開始する準備ができました。

- 一段階マイグレーション方式を使用してマイグレーションを行うには、130 ページの『オプション 1: 一段階マイグレーション』の手順に従ってください。
- 横並びマイグレーション方式を使用してマイグレーションを行うには、137 ページの『オプション 2: 横並びマイグレーション』の手順に従ってください。

オプション 1: 一段階マイグレーション

このシナリオのオプション 1 では、一段階マイグレーション方式を使用した場合に IBM MQ を前のバージョンから新しいバージョンにマイグレーションする手順を示しています。 一段階マイグレーションでは、前のバージョンの IBM MQ のインストール環境がより新しいバージョンに置き換わり、インストール・ロケーションは同じになります。

始める前に

このシナリオの開始点は、117 ページの『概要: 初期 IT 構成』で説明されている初期 IT 構成です。

この作業を開始する前に、121 ページの『初期 IT 構成の作成』の指示に従って初期 IT 構成をセットアップします。

このタスクについて

一段階マイグレーションでは、IBM MQ の以前のバージョンをアンインストールしてから新しいバージョンをインストールする方法、 または以前のバージョンを最初にアンインストールしないで新しいバージョンをインストールする方法 (つまりインプレースでのマイグレーション) を選択できます。 どちらのケースでも、新しいリリースは以前のリリースと同じディレクトリーにインストールされます。 このシナリオのオプション 1 は、以前のバージョンをアンインストールしてから新しいバージョンをインストールする、一段階マイグレーションを示しています。 キュー・マネージャー・データは、アンインストール・プロセスの一部として削除されません。つまり、このシナリオで使用されるサンプル・キュー・マネージャーは、新しいバージョンの IBM MQ をインストールするときに保持され、検出されます。

手順

1. 以前のバージョンの IBM MQ で実行されている キュー・マネージャーを停止 し、キュー・マネージャー・データをバックアップします。
2. キュー・マネージャー・データを削除しないで、マイグレーション元である以前のバージョンの IBM MQ をアンインストールします。
3. ランチパッドを使用して IBM MQ 9.3 をインストールします。
4. IBM MQ Explorer を使用して新しい IBM MQ 9.3 インストール済み環境を検証します。

   キュー・マネージャーが以前のリリースから正常にマイグレーションされたこと、およびマイグレーションされたキューに対してメッセージの書き込みとメッセージの読み取りができることを検証します。

関連タスク

AIX and Linux でのマイグレーション: 一段階

Windows マイグレーションの準備

より新しいバージョンの IBM MQ にマイグレーションする前に、まずキュー・マネージャーを停止して、キュー・マネージャーのデータをバックアップする必要があります。

このタスクについて

IBM MQ の以前のバージョンからマイグレーションするとき、 最初にシステムのバックアップを行わないと、マイグレーションを続行しないことに決めた場合でも、以前のバージョンに戻すことはできません。新しいバージョンをインストールする前にシステムをバックアップすれば、必要な場合にアップグレードを取り消すことができます。 ただし、アップグレードを取り消す場合には、新しいバージョンの IBM MQ によって行われた処理 (メッセージやオブジェクトの変更など) をリカバリーすることはできません。

バックアップを行う前に、バックアップ対象のキュー・マネージャー (このシナリオでは sampleQM) を停止します。 実行中のキュー・マネージャーのバックアップを行う場合、ファイルのコピー時にも更新が進行しているので、バックアップの整合性が得られないこともあります。

手順

1. IBM MQ Explorer を開きます。
   「**スタート**」 > 「**すべてのアプリケーション**」 > **IBM MQ** > **IBM MQ Explorer** をクリックします。
2. キュー・マネージャー sampleQM を停止します。
   a) ナビゲーター・ビューでキュー・マネージャー sampleQM を右クリックします。
   b)「**停止**」をクリックします。
   「**キュー・マネージャーの終了 (End Queue Manager)**」ウィンドウが開きます。
   c)「**制御**」を選択し、「**OK**」をクリックします。
   「制御」オプションを選択し、 制御下のキュー・マネージャーを適切な順序で停止します。 「**即時**」オプションは、キュー・マネージャーを強制的に停止します。通常は、制御された停止が正常に完了しなかった場合にのみ、使用します。
   キュー・マネージャーが停止します。 IBM MQ で、キュー・マネージャー sampleQM の隣りのアイコンが赤い下矢印付きに変わります。
3. IBM MQ Explorer を閉じます。
4. キュー・マネージャーのデータをバックアップします。
   以下のすべてのデータのコピーを作成し、すべてのバックアップ・ディレクトリーも含まれていることを確認します。 一部のディレクトリーは空である場合がありますが、後日バックアップをリストアする必要が生じた場合にはすべて必要になるので、それらも保存してください。
   - `C: ¥ProgramData¥IBM¥MQ\Qmgrs` にあるキュー・マネージャー・データです。
   - `C: ¥ProgramData¥IBM¥MQ\log` にあるキュー・マネージャーのログ・ファイル・ディレクトリーには、ログ制御ファイル `amqhlctl.lfh` も含まれます。
   - `C: ¥ProgramData¥IBM¥MQ\Config` にある構成ファイルです。
   - IBM MQ 9.2 .ini ファイルおよびレジストリー項目。 キュー・マネージャーの情報は .ini ファイルに保管され、前のバージョンの製品に戻すために使用できます。
5. IBM MQ を停止します。
   a) IBM MQ サービスを停止します。
   システム・トレイの **IBM MQ** アイコンを右クリックし、 「**停止**」 **IBM MQ** をクリックします。
   以下のメッセージがダイアログ・ボックスに表示されます。

   ```
   IBM MQ をシャットダウンすると、実行中のすべてのキュー・マネージャーが終了し、
   IBM MQ プロセス。 続行してよろしいですか? (AMQ4102)
   ```

   b)「**はい**」をクリックして、IBM MQ が停止するのを待機します。
   c) IBM MQ が停止したら、システム・トレイの **IBM MQ** アイコンを右クリックし、 「**終了**」をクリックします。

タスクの結果

新しいリリースの IBM MQ にマイグレーションするキュー・マネージャーを停止して、キュー・マネージャーのデータのバックアップを行いました。

次のタスク

これで、 132 ページの『前のバージョンのアンインストール』で説明されているように、 IBM MQ をアンインストールする準備ができました。

関連タスク

キュー・マネージャー・データのバックアップ

Windows 前のバージョンのアンインストール

コントロール・パネルを使用して、前のバージョンをアンインストールします。 Windows での一段階マイグレーションの場合、オプションで、前のバージョンの製品をアンインストールしてからより新しいバージョンをインストールしてください。

始める前に

この作業を開始する前に、 131 ページの『マイグレーションの準備』で説明されているように、まずキュー・マネージャーを停止し、 IBM MQ Explorer を閉じて、 IBM MQ を停止する必要があります。

このタスクについて

このタスクでは、 Windows コントロール・パネルを使用して IBM MQ をアンインストールします。 キュー・マネージャー・データは、アンインストール・プロセスの一部として削除されません。つまり、このシナリオで使用されるサンプル・キュー・マネージャーは、新しいバージョンの製品をインストールするときに保持され、検出できます。

新しいバージョンをインストールする前に以前のバージョンの製品をアンインストールする必要があるかどうかは、オペレーティング・システムによって異なります。 Windows システムではアンインストールは任意選択であり、以前のバージョンをアンインストールしないで新しいバージョンをインストールすることもできます。 このケースでは、インストール中に表示されるオプションやメッセージのいくつかが、以前のバージョンをまずアンインストールした場合に表示されるものと異なることに注意してください。 新しいバージョンをインストールする前に古いバージョンをアンインストールする必要のあるプラットフォームについて詳しくは、 AIX and Linux でのマイグレーション: 一段階を参照してください。

手順

1. 「**スタート**」>「**コントロールパネル**」>「**プログラムのアンインストール**」をクリックして、 Windows コントロールパネルを開きます。
2. 「**プログラムと機能**」ウィンドウで、削除するインストール済み環境の項目 (IBM WebSphere® MQ (Installation1) など) を選択して、「**アンインストール**」をクリックします。

   アンインストール・プロセスが開始し、実行して完了します。 プロセスが完了すると、古いバージョンの IBM MQ はコンピューターから削除されて、プログラムのリストに表示されなくなります。

タスクの結果

IBM MQ の前のバージョンは、ご使用のコンピューターから削除されました。 しかし、キュー・マネージャーのデータは削除されていません。

次のタスク

これで、 133 ページの『ランチパッドを使用した IBM MQ 9.3 のインストール』で説明されているように、新しいバージョンの IBM MQ をインストールする準備ができました。

関連タスク
Windows システムでの IBM MQ のアンインストール

Windows ランチパッドを使用した *IBM MQ 9.3* のインストール

インストール・ランチパッドとウィザードを使用して、新しいバージョンの IBM MQ を、旧バージョンがインストールされていたものと同じ Windows コンピューター上にインストールします。

始める前に

このタスクを開始する前に、インストール・イメージを含む圧縮ファイルをダウンロードして、一時ディレクトリーに解凍します。

このタスクでは、132 ページの『前のバージョンのアンインストール』で説明されているように、マイグレーション元の旧バージョンの IBM MQ が既にアンインストールされていることを前提としています。 以前のバージョンを最初にアンインストールしないで新しいバージョンをインストールする場合、 インストール・プロセス中に表示されるオプションやメッセージのいくつかは、このタスクで説明されるものとは異なります。

このタスクを始める前に、以下の確認を行ってください。

- インストール時に、ローカル管理者権限が必要です。 その権限は Windows の機能で定義します。
- マシン名にスペースが含まれていないことを確認します。
- 十分なディスク・スペースがあることを確認します。 詳しくは、ディスク・スペース要件 (Multiplatforms) を参照してください。

このシナリオでは、 IBM MQ ユーザーの Windows ドメイン・ユーザー ID を定義する必要があるかどうかを判別する必要はありません。この要件は、このシナリオの範囲外であるためです。 詳細については、IBM MQ 用の Active Directory および DNS ドメインの作成を参照してください。

IBM MQ をインストールする前に、システムがハードウェアとソフトウェアの要件を満たしていることを確認してください。 ハードウェア要件とソフトウェア要件の最新の詳細情報については、IBM MQ のシステム要件を参照してください。

このタスクについて

このタスクでは、旧バージョンからマイグレーションする場合に Windows オペレーティング・システムに IBM MQ をインストールするための基本的なステップについて説明します。

注 : デフォルトのプログラムおよびデータのディレクトリーの場所は、IBM MQ 9.0 以降のバージョンでは同じです。 そのため、IBM MQ 9.0 からそれ以降のバージョンにマイグレーションする場合、プログラムおよびデータのディレクトリーの指定を変更する必要はありません。 ただし、 IBM MQ 9.0 より前のバージョンからマイグレーションする場合は、デフォルト・ロケーションに違いがあることを考慮する必要があります。 詳細については、プログラム・ディレクトリーとデータ・ディレクトリーの場所 (Windows) を参照してください。

インストール・プログラムには、インストール・プロセスの間に必要になる場合のために、詳細情報へのリンクが含まれます。 インストール・プロセスは、次の部分からなります。

1. ランチパッドを使用して、ソフトウェア要件を検査し、インストールして、ネットワーク情報を指定し、IBM MQ インストール・ウィザードを開始します。
2. IBM MQ インストール・ウィザードを使用してソフトウェアをインストールし、Prepare IBM MQ Wizard を開始します。
3. Prepare IBM MQ Wizard を使用して、IBM MQ サービスを開始します。

手順

1. インストール処理を開始します。

   Windows エクスプローラーで、インストール・イメージをダウンロードした一時フォルダーにナビゲートし、 `setup.exe` をダブルクリックします。

インストール・ランチパッドが開始します。

2. ランチパッドを起動して、ソフトウェア要件およびネットワーク構成を確認し、必要であれば変更します。
   a) IBM MQ ソフトウェア・ディレクトリーに移動し、`Setup.exe` ファイルをダブルクリックして、ランチパッドを開始します。
   b)「**ソフトウェア要件**」ボタンをクリックして、「**ソフトウェア要件**」タブを表示します。
   c) ソフトウェア要件が満たされていること、および要件の項目に OK という語を含む緑色のチェック・マークが表示されていることを確認します。 指示されている訂正を行います。

      注 :

      要件の詳細は、チェック・ボックスをクリックし、情報タブを展開して参照できます。

   d)「**ネットワーク構成**」ボタンをクリックし、「**ネットワーク構成**」タブを表示します。
   e)「**いいえ**」ラジオ・ボタンをクリックします。

      注 : このシナリオは、IBM MQ 用にドメイン・ユーザー ID を構成する必要がないことを想定しています。 IBM MQ for Windows ドメイン・ユーザーの構成について詳しくは、「**詳細情報**」ボタンをクリックしてください。

   f) ランチパッドの「**IBM MQ インストール**」タブで、インストール言語を選択してから、「**IBM MQ インストーラーの起動**」をクリックして、IBM MQ インストール・ウィザードを開始します。

   インストール要件を満たす、または指定することにより、IBM MQ のセットアップを完了し、IBM MQ インストール・ウィザードを開始しました。
3. IBM MQ インストール・ウィザードを使用してソフトウェアをインストールし、Prepare IBM MQ Wizard を開始します。
   a) IBM MQ インストール・ウィザードで、ご使用条件を読み、「**使用条件の条項に同意します**」チェック・ボックスをクリックして、「**次へ**」をクリックします。
   b)「**標準**」をクリックしてから、「**次へ**」をクリックします。
   c)「**IBM MQ をインストールする準備ができました**」ページで、インストール情報を確認し、「**インストール**」をクリックします。

      インストール情報には、以下の詳細が含まれます。

      - インストール環境の名前
      - プログラム・ファイルの最上位フォルダー
      - データ・ファイルの最上位フォルダー

      以下の機能がインストールされます。

      - IBM MQ Server
      - IBM MQ: IBM MQ リソースを管理およびモニターするためのグラフィカル・インターフェース
      - Java™、 .NET メッセージングおよび Web サービス
      - IBM MQ 開発ツールキット

      インストール・プロセスが開始します。 ご使用のシステムによっては、インストール・プロセスに数分を要する場合があります。

      インストール・プロセスの最後に、IBM MQ セットアップ・ウィンドウに `Installation Wizard Completed Successfully` というメッセージが表示されます。

   d)「**完了 (Finish)**」 をクリックします。

   IBM MQ が正常にインストールされました。 Prepare IBM MQ Wizard が自動的に開始し、「**Prepare IBM MQ Wizard にようこそ**」ページが表示されます。
4. Prepare IBM MQ Wizard を使用して、IBM MQ サービスを開始します。
   a)「Prepare IBM MQ Wizard にようこそ」ページで、「**次へ**」を選択します。

Prepare IBM MQ Wizard には、メッセージ `Status: Checking IBM MQ Configuration` と進行状況表示バーが表示されます。 処理が完了すると、IBM MQ ネットワーク構成ページが表示されます。

b) Prepare IBM MQ Wizard の「 IBM MQ ネットワーク構成」ページで、 「**いいえ**」を選択します。

c) **次へ** をクリックします。

Prepare IBM MQ Wizard には、メッセージ `Status: starting the IBM MQ Service` と進行状況表示バーが表示されます。 処理が完了すると、ウィザードに「Prepare IBM MQ Wizard の完了」ページが表示されます。

d)「**IBM MQ Explorer を起動する**」を選択し、メモ帳を起動してリリース・ノートを表示するかどうかを選択して、「**完了**」ボタンをクリックします。

IBM MQ Explorer が始動します。

タスクの結果

IBM MQ のインストールが完了し、IBM MQ Explorer が開始しました。

次のタスク

新しいバージョンの IBM MQ がインストールされたので、135 ページの『IBM MQ 9.3 のインストールの検証』で説明されているように、サンプルのキュー・マネージャーが正常にマイグレーションされたことの確認と、マイグレーションされたキューに対してメッセージの書き込みとメッセージの読み取りができることの確認を行う準備ができました。

関連概念

ダウンロード可能なインストール・イメージの入手先

関連タスク

Windows での IBM MQ サーバーのインストール

IBM MQ 9.3 のインストールの検証

IBM MQ 9.3 をインストールした後に、IBM MQ Explorer を使用して、キュー・マネージャーとキューが旧リリースから正常にマイグレーションされたことを確認し、さらにサンプル・アプリケーションを使用できることを確認します。

このタスクについて

マイグレーションしたキュー・マネージャー sampleQM が IBM MQ Explorer のナビゲーター・ビューに表示されていることを確認してから、マイグレーションしたキューに対してメッセージの書き込みと読み取りができることを確認し、さらにサンプル・アプリケーションを実行できることを確認します。

手順

1. IBM MQ Explorer が実行されていない場合は、この時点でそれを開始します。
   「**スタート**」>「**すべてのプログラム**」>「**IBM MQ**」>「**IBM MQ Explorer**」をクリックします。
2. 以下のようにして、キュー・マネージャーが新しいバージョンの IBM MQ に正常にマイグレーションされたかどうかを検査します。
   a) ナビゲーター・ビューで **Queue Managers** フォルダーを展開します。
   b) **Queue Managers** フォルダーにキュー・マネージャー sampleQM が含まれていることを確認します。
   c) キュー・マネージャー sampleQM を展開し、**Queues** フォルダーをクリックして、「コンテンツ」ビューにキュー Q1 が表示されていることを確認します。
3. キュー・マネージャー sampleQM がまだ開始されていない場合は、ここで開始します。
   a) ナビゲーター・ビューでキュー・マネージャー・ノードを展開します。
   b) キュー・マネージャー sampleQM を右クリックして「**開始**」をクリックします。
4. メッセージをキュー Q1 に書き込めるかどうかを検査します。

a) ナビゲーター・ビューで **Queue Managers** フォルダーを展開します。

b) キュー・マネージャー sampleQM を展開して、**Queues** フォルダーをクリックします。

c)「コンテンツ」ビューでキュー Q1 を右クリックし、「**テスト・メッセージの書き込み**」をクリックします。

「**テスト・メッセージの書き込み**」ダイアログが開きます。

d)「**メッセージ・データ**」 フィールドに、 Hello queue!などのテキストを入力し、 「**メッセージの書き込み**」をクリックします。

「**メッセージ・データ**」フィールドがクリアされ、メッセージが キューに書き込まれます。

e)「**クローズ**」をクリックします。

コンテンツビューで、キューの **現行キュー項目数** の値が 1 になっていることに注目してください。「現行キュー項目数」の列が表示されない場合、コンテンツビューの右側にスクロールする必要があるかもしれません。

5. キュー Q1 からメッセージを読み取ることができることを検証します。

a) ナビゲーター・ビューで **Queue Managers** フォルダーを展開します。

b) キュー・マネージャー sampleQM を展開して、**Queues** フォルダーをクリックします。

c)「コンテンツ」ビューでキュー Q1 を右クリックして、「**メッセージの参照**」をクリックします。

「メッセージ・ブラウザー」が開き、現在キューに入っているメッセージのリストが表示されます。

d) 最後のメッセージをダブルクリックして、プロパティー・ダイアログを開きます。

プロパティー・ダイアログの「**データ**」ページの「**メッセージ・データ**」フィールドに、人間が読める形式でメッセージの内容が表示されます。

6. サンプル・アプリケーションを実行できることを検証します。

a) runresponder.cmd というファイルをダブルクリックしてください。

「**応答側ウィンドウ (Responder window)**」というラベルのあるコマンド・プロンプト・ウィンドウで、応答側クライアントが開始してメッセージを待機します。

```
> Connection factory located in JNDI.> Destination located in JNDI.> Creating connection to
QueueManager.> Created connection.
> Waiting for message.
```

b) runrequester.cmd というファイルをダブルクリックしてください。

「**要求側ウィンドウ (Requester window)**」で、要求側メッセージを監視します。「**応答側ウィンドウ (Responder window)**」で、更新される応答側メッセージ (要求側クライアントから受け取るメッセージ、および送信する応答メッセージ) を監視します。

「**要求側ウィンドウ (Requester window)**」というラベルのあるコマンド・プロンプト・ウィンドウで、要求側クライアントは接続状況、送信したメッセージ、そして応答側クライアントから受け取る応答メッセージを表示します。

```
> Connection factory located in JNDI.> Destination located in JNDI.> Creating connection to
QueueManager.> Connection created.
> Sending stock request for 'BakedBeans'> Sent Message
ID=ID:414d5120514d5f4c33344c3238482020c3cd094d20002b02
> Received Message ID=ID:414d5120514d5f4c33344c3238482020c3cd094d20002902 for 'B
akedBeans - 15 tins in stock'
> Closing connection to QueueManager.> Closed Connection.
------------------------------------------------------
In this window, observe the messages sent through IBM MQ:
- The request message sent
- The reply message received
-----
When ready, press any key to close this window
続行するには、任意のキーを押してください. . .
```

「**応答側ウィンドウ (Responder window)**」で、更新される応答側メッセージ (要求側クライアントから受け取るメッセージ、および送信する応答メッセージ) を監視します。

```
> Connection factory located in JNDI.> Destination located in JNDI.> Creating connection to
QueueManager.> Created connection.
> Waiting for message.

> Received Message ID=ID:414d5120514d5f4c33344c3238482020c3cd094d20002b02 for 'B
akedBeans'
> Sending Reply Message 'BakedBeans - 15 tins in stock'> Sent Message
```

```
ID=ID:414d5120514d5f4c33344c3238482020c3cd094d20002902
> Closing connection to QueueManager.> Closed connection.
------------------------------------------------------
In this window, observe the updated responder messages
- The request message received (from the requester)
- The reply message sent
-----
When ready, press any key to close this window
続行するには、任意のキーを押してください. . .
```

2 つのコマンド・ウィンドウに表示されるメッセージによって、サンプル・アプリケーションの要求側クライアントと応答側クライアントが IBM MQ を介して相互に通信可能であることが検証されます。

タスクの結果

新しいバージョンの IBM MQ へのマイグレーションが正常に完了しました。

Windows オプション 2: 横並びマイグレーション

このシナリオのオプション 2 では、横並びマイグレーション方式を使用した場合に IBM MQ を前のリリースから新しいリリースにマイグレーションする手順を示しています。 「横並び」マイグレーションでは、最新バージョンの IBM MQ を、マイグレーションする前のバージョンと共存するようにインストールします。 キュー・マネージャーおよびアプリケーションは、新しいリリースにマイグレーションされるまで、前のリリースと関連付けられたままになります。

始める前に

このシナリオの開始点は、117 ページの『概要: 初期 IT 構成』で説明されている初期 IT 構成です。

この作業を開始する前に、121 ページの『初期 IT 構成の作成』の指示に従って初期 IT 構成をセットアップします。

このタスクについて

このシナリオで説明されている横並びのマイグレーション方式に従うと、新しいバージョンが以前のバージョンと共存して代替場所にインストールされます。 以前のバージョンをアンインストールしてから新しいバージョンのキュー・マネージャーを開始するので、 新しいバージョンの IBM MQ のインストール済み環境をプライマリー・インストールとして割り当てることができます。 プライマリー・インストールについて詳しくは、プライマリー・インストールを参照してください。

手順

1. ランチパッドを使用して IBM MQ 9.3 をインストールしてから、インストール済み環境を検証します。
2. 以前のバージョンの IBM MQ で実行されているキュー・マネージャーを停止します。
3. 以前のバージョンの IBM MQ をアンインストールします。
4. IBM MQ 9.3 をプライマリー・インストールにします。
5. オプション: キュー・マネージャーを IBM MQ 9.3 に関連付けます。
6. IBM MQ Explorer を使用して、IBM MQ 9.3 のインストール済み環境を検証します。

   キュー・マネージャーが以前のリリースから正常にマイグレーションされたこと、およびマイグレーションされたキューに対してメッセージの書き込みとメッセージの読み取りができることを確認します。

関連タスク

Windows でのマイグレーション: 横並び

Windows ランチパッドを使用した *IBM MQ 9.3* のインストール

インストール・ランチパッドおよびウィザードを使用して、新しいバージョンの IBM MQ を古いバージョンと共存するように Windows にインストールします。

始める前に

この作業を実行する前に、以下の検査を完了してください。

- インストール時に、ローカル管理者権限が必要です。 その権限は Windows の機能で定義します。
- マシン名にスペースが含まれていないことを確認します。
- IBM MQ for Windows をフルインストールするために、十分なディスク・スペース (最大 1005 MB) があることを確認します。
- いずれかの IBM MQ ユーザーに対して Windows ドメイン・ユーザー ID を定義する必要があるかどうかを判別します。

IBM MQ をインストールする前に、システムがハードウェアとソフトウェアの要件を満たしていることを確認してください。 サポートされているすべてのプラットフォームに関するハードウェア要件とソフトウェア要件の最新の詳細情報については、IBM MQ のシステム要件 を参照してください。

このタスクについて

この作業は、IBM MQ がシステムにまだインストールされていない場合に、 Windows にインストールするための基本ステップについて説明しています。

この作業では、IBM MQ のプログラム・ファイルとデータ・ファイルを配置するためにデフォルトの場所を使用することを想定しています。

注 : デフォルトのプログラムおよびデータのディレクトリーの場所は、IBM MQ 9.0 以降のバージョンでは同じです。 そのため、IBM MQ 9.0 からそれ以降のバージョンにマイグレーションする場合、プログラムおよびデータのディレクトリーの指定を変更する必要はありません。 ただし、 IBM MQ 9.0 より前のバージョンからマイグレーションする場合は、デフォルト・ロケーションに違いがあることを考慮する必要があります。 詳細については、プログラム・ディレクトリーとデータ・ディレクトリーの場所 (Windows) を参照してください。

インストール・プログラムには、インストール・プロセスの間に必要になる場合のために、詳細情報へのリンクが含まれます。 インストール・プロセスは、次の部分からなります。

1. ランチパッドを使用して、ソフトウェア要件を検査し、インストールして、ネットワーク情報を指定し、IBM MQ インストール・ウィザードを開始します。
2. IBM MQ インストール・ウィザードを使用してソフトウェアをインストールし、Prepare IBM MQ Wizard を開始します。
3. Prepare IBM MQ Wizard を使用して、IBM MQ サービスを開始します。

手順

1. インストール処理を開始します。

   Windows エクスプローラーで、インストール・イメージをダウンロードした一時フォルダーにナビゲートし、 `setup.exe` をダブルクリックします。

   インストール・ランチパッドが開始します。
2. ランチパッドを使用して、ソフトウェア要件およびネットワーク構成を確認し、必要であれば変更します。
   a)「**ソフトウェア要件**」ボタンをクリックして、「**ソフトウェア要件**」タブを表示します。
   b) ソフトウェア要件が満たされていること、および要件の項目に OK という語を含む緑色のチェック・マークが表示されていることを確認します。 指示されている訂正を行います。

      注 : 要件に関するより詳細を表示するには、プラス (+) ボタンをクリックします。
   c)「**ネットワーク構成**」ボタンをクリックし、「**ネットワーク構成**」タブを表示します。
   d)「**いいえ**」ラジオ・ボタンをクリックします。

注 : このシナリオは、IBM MQ 用にドメイン・ユーザー ID を構成する必要がないことを想定しています。 IBM MQ for Windows ドメイン・ユーザーの構成について詳しくは、「**詳細情報**」ボタンをクリックしてください。

e) ランチパッドの「**IBM MQ インストール**」タブで、インストール言語を選択してから、「**IBM MQ インストーラーの起動**」ボタンをクリックして、IBM MQ インストール・ウィザードを開始します。

IBM MQ のインストール要件の確認を完了し、必要な変更を行い、IBM MQ インストール・ウィザードを開始しました。

3. IBM MQ インストール・ウィザードを使用してソフトウェアをインストールし、Prepare IBM MQ Wizard を開始します。

   IBM MQ インストール・ウィザードは、既存のインストール済み環境がないかを調べ、使用可能なアップグレード・オプションやインストール・オプションを表示します。 このシナリオのケースでは、以下の 2 つのオプションがあります。

   - 既存のインストール済み環境には影響を与えずにインストールする
   - 8.0.0.5 のインストール済み環境 'Installation 1' をアップグレードする

   a)「**既存のインストール済み環境には影響を与えずにインストールする**」を選択してから、「**次へ**」をクリックします。

   b) ご使用条件を読んで、「**使用条件の条項に同意します**」チェック・ボックスをクリックし、「**次へ**」をクリックします。

   c)「**標準**」をクリックしてから、「**次へ**」をクリックします。

   d)「**IBM MQ をインストールする準備ができました**」ページで、表示されるインストール情報を確認し、「**インストール**」をクリックします。

   インストール情報には、以下の詳細が含まれます。

   - インストール環境の名前
   - プログラム・ファイルの最上位フォルダー
   - データ・ファイルの最上位フォルダー

   以下の機能がインストールされます。

   - IBM MQ Server
   - IBM MQ: IBM MQ リソースを管理およびモニターするためのグラフィカル・インターフェース
   - Java™、 .NET メッセージングおよび Web サービス
   - IBM MQ 開発ツールキット

   インストール・プロセスが開始します。 ご使用のシステムによっては、インストール・プロセスに数分を要する場合があります。

   インストール・プロセスの最後に、IBM MQ セットアップ・ウィンドウに `Installation Wizard Completed Successfully` というメッセージが表示されます。

   e)「**完了 (Finish)**」 をクリックします。

   IBM MQ が正常にインストールされました。 Prepare IBM MQ Wizard が自動的に開始し、「**Prepare IBM MQ Wizard にようこそ**」ページが表示されます。

4. Prepare IBM MQ Wizard を使用して、IBM MQ サービスを開始します。

   a)「Prepare IBM MQ Wizard にようこそ」ページで、「**次へ**」を選択します。

   Prepare IBM MQ Wizard には、メッセージ `Status: Checking IBM MQ Configuration` と進行状況表示バーが表示されます。 処理が完了すると、IBM MQ ネットワーク構成ページが表示されます。

   b) Prepare IBM MQ Wizard の「 IBM MQ ネットワーク構成」ページで、 「**いいえ**」を選択します。

   c) **次へ** をクリックします。

   Prepare IBM MQ Wizard には、メッセージ `Status: starting the IBM MQ Service` と進行状況表示バーが表示されます。 処理が完了すると、ウィザードに「Prepare IBM MQ Wizard の完了」ページが表示されます。

d)「**IBM MQ Explorer を起動する**」を選択し、メモ帳を起動してリリース・ノートを表示するかどうかを選択して、「**完了**」ボタンをクリックします。

IBM MQ Explorer が始動します。

次のタスク

新しいバージョンの IBM MQ を古いバージョンと共存するように (ただし別のインストール・ディレクトリーに) インストールし、 IBM MQ Explorer を開始しました。

これで、140 ページの『キュー・マネージャーの停止』で説明されているように、旧バージョンの IBM MQ で実行されているキュー・マネージャーを停止する準備ができました。

Windows キュー・マネージャーの停止

より新しいバージョンの IBM MQ にマイグレーションする前に、まずキュー・マネージャーを停止して、キュー・マネージャーのデータをバックアップする必要があります。

このタスクについて

バックアップを行う前に、バックアップ対象のキュー・マネージャーを停止します。 実行中のキュー・マネージャーのバックアップを行う場合、ファイルのコピー時にも更新が進行しているので、バックアップの整合性が得られないこともあります。

手順

1. IBM MQ Explorer を開きます。

   「**スタート**」 >「**すべてのアプリケーション**」 > **IBM MQ** > **IBM MQ Explorer** をクリックします。
2. キュー・マネージャー sampleQM を停止します。

   a) ナビゲーター・ビューでキュー・マネージャー sampleQM を右クリックします。

   b)「**停止**」をクリックします。

   「**キュー・マネージャーの終了 (End Queue Manager)**」ウィンドウが開きます。

   c)「**制御**」を選択し、「**OK**」をクリックします。

   「制御」オプションを選択し、 制御下のキュー・マネージャーを適切な順序で停止します。 「**即時**」オプションは、キュー・マネージャーを強制的に停止します。通常は、制御された停止が正常に完了しなかった場合にのみ、使用します。

   キュー・マネージャーが停止します。 IBM MQ で、キュー・マネージャー sampleQM の隣りのアイコンが赤い下矢印付きに変わります。
3. IBM MQ Explorer を閉じます。
4. キュー・マネージャーのデータをバックアップします。

   以下のすべてのデータのコピーを作成し、すべてのバックアップ・ディレクトリーも含まれていることを確認します。 一部のディレクトリーは空である場合がありますが、後日バックアップをリストアする必要が生じた場合にはすべて必要になるので、それらも保存してください。

   - `C: ¥ProgramData¥IBM¥MQ\Qmgrs` にあるキュー・マネージャー・データです。
   - `C: ¥ProgramData¥IBM¥MQ\log` にあるキュー・マネージャーのログ・ファイル・ディレクトリーには、ログ制御ファイル `amqhlctl.lfh` も含まれます。
   - `C: ¥ProgramData¥IBM¥MQ\Config` にある構成ファイルです。
   - IBM MQ 9.2 .ini ファイルおよびレジストリー項目。 キュー・マネージャーの情報は .ini ファイルに保管され、前のバージョンの製品に戻すために使用できます。
5. IBM MQ を停止します。

   a) IBM MQ サービスを停止します。

   システム・トレイの **IBM MQ** アイコンを右クリックし、 「**停止**」 **IBM MQ** をクリックします。

   以下のメッセージがダイアログ・ボックスに表示されます。

```
IBM MQ をシャットダウンすると、実行中のすべてのキュー・マネージャーが終了し、
IBM MQ プロセス。 続行してよろしいですか? (AMQ4102)
```

b)「**はい**」をクリックして、IBM MQ が停止するのを待機します。

c) IBM MQ が停止したら、システム・トレイの **IBM MQ** アイコンを右クリックし、「**終了**」をクリックします。

次のタスク

キュー・マネージャーを停止すると、142 ページの『キュー・マネージャーと IBM MQ 9.3 の関連付け』で説明されているように、新しいバージョンの IBM MQ の新しいインストール済み環境にキュー・マネージャーを関連付けることができます。

Windows 前のバージョンのアンインストール

Windows コントロール・パネルを使用して、前のバージョンの製品をアンインストールします。

始める前に

この作業を開始する前に、140 ページの『キュー・マネージャーの停止』で説明されているように、まずキュー・マネージャーを停止し、 IBM MQ Explorer を閉じて、 IBM MQ を停止する必要があります。

このタスクについて

このタスクでは、 Windows コントロール・パネルを使用して IBM MQ をアンインストールします。 キュー・マネージャー・データは、アンインストール・プロセスの一部として削除されません。つまり、このシナリオで使用されるサンプル・キュー・マネージャーは、新しいバージョンの製品をインストールするときに保持され、検出できます。

手順

1.「**スタート**」>「**コントロールパネル**」>「**プログラムのアンインストール**」をクリックして、 Windows コントロールパネルを開きます。
2.「**プログラムと機能**」ウィンドウで、削除するインストール済み環境の項目 (IBM WebSphere MQ (Installation1) など) を選択して、「**アンインストール**」をクリックします。

   アンインストール・プロセスが開始し、実行して完了します。 プロセスが完了すると、古いバージョンの IBM MQ はコンピューターから削除されて、プログラムのリストに表示されなくなります。

タスクの結果

前のバージョンの製品が、コンピューターから削除されました。 しかし、キュー・マネージャーのデータは削除されていません。

次のタスク

これで、141 ページの『IBM MQ 9.3 のプライマリー・インストール設定』で説明されているように、新しいバージョンの IBM MQ をプライマリー・インストールにする準備ができました。

関連タスク

Windows システムでの IBM MQ のアンインストール

Windows *IBM MQ 9.3* のプライマリー・インストール設定

より新しいバージョンの IBM MQ の新しいインストール済み環境でキュー・マネージャーを開始する前に、オプションで、その新しいバージョンをプライマリー・インストールとして設定することができます。

このタスクについて

IBM MQ の複数インストールをサポートするシステムにおいて、プライマリー・インストールとは IBM MQ システム全体が関係する場所が参照するインストールです。 プライマリー・インストールはオプションですが、便利です。

このシナリオで説明されている横並びのマイグレーション方式に従うと、以前のバージョンをアンインストールしてから新しいリリースでキュー・マネージャーを開始するので、 より新しいバージョンの製品のインストール済み環境をプライマリー・インストールとして割り当てることができます。

プライマリー・インストールについて詳しくは、プライマリー・インストールを参照してください。

手順

1. **dspmqinst** をコマンド・プロンプトに入力し、現行のプライマリー・インストールを確認します。

   コマンド・プロンプトに現行インストールの詳細が表示されます。 現行のプライマリー・インストールには、`Primary: Yes` という行があります。
2. **setmqinst** コマンドを使用して、現行のプライマリー・インストールを変更します。

   コマンド行に以下のように入力します。

   ```
   setmqinst -x -n Installation_Name
   ```

   ここで、*Installation_Name*> は現行のプライマリー・インストールの名前です。

   コマンドが正常に実行されると、コマンド・プロンプトにメッセージ '*Installation_Name*' (*Filepath*) has been unset as the Primary Installation が表示されます。
3. **setmqinst** コマンドを使用して、新規 IBM MQ 9.3 インストールをプライマリー・インストールとして設定します。

   コマンド行に以下のように入力します。

   ```
   setmqinst -i -n V9_Installation
   ```

   V9_Installation は IBM MQ 9.3 インストール済み環境の名前です。

   コマンドが正常に実行されると、コマンド・プロンプトにメッセージ '*V9_Installation*' (*Filepath*) has been set as the primary installation. You must restart the operating system to complete the update. が表示されます。

   注: 成功のメッセージに示されているとおり、更新を完了するには、オペレーティング・システムを再始動する必要があります。

次のタスク

142 ページの『キュー・マネージャーと IBM MQ 9.3 の関連付け』で説明されているように、マイグレーションされたキュー・マネージャーを新しいバージョンの IBM MQ に関連付ける準備ができました。

Windows キュー・マネージャーと *IBM MQ 9.3* の関連付け

「キュー・マネージャーの転送」ウィザードを使用して、 sampleQM キュー・マネージャーを、新しいバージョンの IBM MQ のインストール済み環境に関連付けます。

始める前に

この作業を開始する前に、140 ページの『キュー・マネージャーの停止』で説明されているようにキュー・マネージャーを停止したことを確認してください。停止していないと、転送を完了できません。

このタスクについて

IBM MQ Explorer の「キュー・マネージャーの転送」ウィザード・フィーチャーを使用すると、1 つ以上のキュー・マネージャーを他のインストール済み環境から現行のインストール済み環境に転送することができます。 このウィザードは **setmqm** コマンドと同等ですが、必要なパスやパラメーターを入力する手間が省けます。 転送できるのは、停止中のキュー・マネージャーだけです。実行中のキュー・マネージャーは参照のために表示されます。

新しいバージョンの製品のインストール済み環境でキュー・マネージャーを転送して開始した後は、 以前のバージョンに戻すことはできません。

手順

1. IBM MQ Explorer を開始します。
   「**スタート**」 > 「**すべてのアプリケーション**」 > **IBM MQ** > **IBM MQ Explorer** をクリックします。
2. ナビゲーター・ビューで、キュー・マネージャー・ノードを右クリックして、「**キュー・マネージャーの転送**」を選択します。
3. 右クリックしてからキュー・マネージャー sampleQM を選択し、「**転送**」をクリックします。
   選択したキュー・マネージャーが指定された状態で **setmqm** コマンドが呼び出されます。 転送が成功すると、ナビゲーター・ツリーが更新され、転送されたキュー・マネージャーが組み込まれます。 問題がある場合、ダイアログが表示され、コマンドからのエラー・メッセージが示されます。
4. キュー・マネージャー sampleQM を開始します。
   a) ナビゲーター・ビューでキュー・マネージャー・ノードを展開します。
   b) キュー・マネージャーの名前を右クリックし、「**開始**」をクリックします。

タスクの結果

キュー・マネージャー sampleQM が新しいバージョンの IBM MQ に正常に関連付けられました。

次のタスク

143 ページの『IBM MQ 9.3 インストール済み環境の検証』で説明されているように、キューに対してメッセージの書き込みと読み取りができることを確認して、キュー・マネージャー sampleQM が正常にマイグレーションされたことを検証します。

IBM MQ 9.3 インストール済み環境の検証

IBM MQ の新しいバージョンをインストールした後、 IBM MQ Explorer を使用して、キュー・マネージャーとキューが前のリリースから移行済みだと確認してから、サンプル・アプリケーションを使用できることを確認してください。

このタスクについて

マイグレーションしたキュー・マネージャー sampleQM が IBM MQ Explorer のナビゲーター・ビューに表示されていることを確認してから、マイグレーションしたキューに対してメッセージの書き込みと読み取りができることを確認し、さらにサンプル・アプリケーションを実行できることを確認します。

手順

1. IBM MQ Explorer が実行されていない場合は、この時点でそれを開始します。
   「**スタート**」**>**「**すべてのプログラム**」**>**「**IBM MQ**」**>**「**IBM MQ Explorer**」をクリックします。
2. 以下のようにして、キュー・マネージャーが新しいバージョンの IBM MQ に正常にマイグレーションされたかどうかを検査します。
   a) ナビゲーター・ビューで **Queue Managers** フォルダーを展開します。
   b) **Queue Managers** フォルダーにキュー・マネージャー sampleQM が含まれていることを確認します。

c) キュー・マネージャー sampleQM を展開し、**Queues** フォルダーをクリックして、「コンテンツ」ビューにキュー Q1 が表示されていることを確認します。

3. キュー・マネージャー sampleQM がまだ開始されていない場合は、ここで開始します。
   a) ナビゲーター・ビューでキュー・マネージャー・ノードを展開します。
   b) キュー・マネージャー sampleQM を右クリックして「**開始**」をクリックします。
4. メッセージをキュー Q1 に書き込めるかどうかを検査します。
   a) ナビゲーター・ビューで **Queue Managers** フォルダーを展開します。
   b) キュー・マネージャー sampleQM を展開して、**Queues** フォルダーをクリックします。
   c)「コンテンツ」ビューでキュー Q1 を右クリックし、「**テスト・メッセージの書き込み**」をクリックします。
   「**テスト・メッセージの書き込み**」ダイアログが開きます。
   d)「**メッセージ・データ**」 フィールドに、 `Hello queue!`などのテキストを入力し、 「**メッセージの書き込み**」をクリックします。
   「**メッセージ・データ**」フィールドがクリアされ、メッセージが キューに書き込まれます。
   e)「**クローズ**」をクリックします。
   コンテンツビューで、キューの **現行キュー項目数** の値が 1 になっていることに注目してください。「現行キュー項目数」の列が表示されない場合、コンテンツビューの右側にスクロールする必要があるかもしれません。
5. キュー Q1 からメッセージを読み取ることができることを検証します。
   a) ナビゲーター・ビューで **Queue Managers** フォルダーを展開します。
   b) キュー・マネージャー sampleQM を展開して、**Queues** フォルダーをクリックします。
   c)「コンテンツ」ビューでキュー Q1 を右クリックして、「**メッセージの参照**」をクリックします。
   「メッセージ・ブラウザー」が開き、現在キューに入っているメッセージのリストが表示されます。
   d) 最後のメッセージをダブルクリックして、プロパティー・ダイアログを開きます。
   プロパティー・ダイアログの「**データ**」ページの「**メッセージ・データ**」フィールドに、人間が読める形式でメッセージの内容が表示されます。
6. サンプル・アプリケーションを実行できることを検証します。
   a) `runresponder.cmd` というファイルをダブルクリックしてください。
   「**応答側ウィンドウ (Responder window)**」というラベルのあるコマンド・プロンプト・ウィンドウで、応答側クライアントが開始してメッセージを待機します。

```
> Connection factory located in JNDI.> Destination located in JNDI.> Creating connection to
QueueManager.> Created connection.
> Waiting for message.
```

   b) `runrequester.cmd` というファイルをダブルクリックしてください。
   「**要求側ウィンドウ (Requester window)**」で、要求側メッセージを監視します。 「**応答側ウィンドウ (Responder window)**」で、更新される応答側メッセージ (要求側クライアントから受け取るメッセージ、および送信する応答メッセージ) を監視します。

「**要求側ウィンドウ (Requester window)**」というラベルのあるコマンド・プロンプト・ウィンドウで、要求側クライアントは接続状況、送信したメッセージ、そして応答側クライアントから受け取る応答メッセージを表示します。

```
> Connection factory located in JNDI.> Destination located in JNDI.> Creating connection to
QueueManager.> Connection created.
> Sending stock request for 'BakedBeans'> Sent Message
ID=ID:414d5120514d5f4c33344c3238482020c3cd094d20002b02
> Received Message ID=ID:414d5120514d5f4c33344c3238482020c3cd094d20002902 for 'B
akedBeans - 15 tins in stock'
> Closing connection to QueueManager.> Closed Connection.
------------------------------------------------------------
In this window, observe the messages sent through IBM MQ:
- The request message sent
- The reply message received
-----
When ready, press any key to close this window
続行するには、任意のキーを押してください. . .
```

「**応答側ウィンドウ (Responder window)**」で、更新される応答側メッセージ (要求側クライアントから受け取るメッセージ、および送信する応答メッセージ) を監視します。

```
> Connection factory located in JNDI.> Destination located in JNDI.> Creating connection to
QueueManager.> Created connection.
> Waiting for message.

> Received Message ID=ID:414d5120514d5f4c33344c3238482020c3cd094d20002b02 for 'B
akedBeans'
> Sending Reply Message 'BakedBeans - 15 tins in stock'> Sent Message
ID=ID:414d5120514d5f4c33344c3238482020c3cd094d20002902
> Closing connection to QueueManager.> Closed connection.
-------------------------------------------------------
In this window, observe the updated responder messages
- The request message received (from the requester)
- The reply message sent
-----
When ready, press any key to close this window
続行するには、任意のキーを押してください. . .
```

2 つのコマンド・ウィンドウに表示されるメッセージによって、サンプル・アプリケーションの要求側クライアントと応答側クライアントが IBM MQ を介して相互に通信可能であることが検証されます。

タスクの結果

新しいバージョンの IBM MQ へのマイグレーションが正常に完了しました。

次のタスク

マイグレーションとアップグレードについて詳しくは、保守およびマイグレーションを参照してください。

Windows LTS 以前のバージョンと共存するために、IBM MQ で Windows の新しいバージョンをインストールします。

このシナリオでは、旧バージョンの製品と共存するように (横並びに) Long Term Support (LTS) バージョンの IBM MQ をインストールするためのすべてのステップを示します。 また、これらのステップには、新しいバージョンへのフィックスパックのインストールも含まれます。 常に最新のレベルのフィックスパックをインストールする必要があります。

複数インストールの概要

複数インストール、およびこのシナリオで使用されるハードウェアおよびソフトウェアについての説明。

このタスクについて

同じホストへの IBM MQ の複数インストール (マルチインストール) の機能の重要な特徴として、このタイプのインストールでは、 新しいバージョンの製品でアクティビティーを実行する際に、旧バージョンの製品のキュー・マネージャーおよびアプリケーションを停止する必要がありません。

つまり、新しいバージョンの製品をインストールしても、旧バージョンの製品でのアプリケーションの実行は影響を受けません。 これは、旧バージョンの製品のキュー・マネージャーを新しいバージョンに段階的にマイグレーションしようとする場合に役立ちます。

このシナリオでは、旧バージョンの製品として IBM MQ 8.0.0 を使用し、新しいバージョンの製品として IBM MQ 9.1.0 を使用します。

Long Term Support バージョンの IBM MQ の使用

このシナリオでは、新しいバージョンの製品として LTS バージョンの IBM MQ を使用します。

CD 新しいバージョンの製品に Continuous Delivery (CD) バージョンの IBM MQ を使用している場合は、新しいバージョン (IBM MQ 9.1.5 など) をインストールする前に、使用している CD バージョン (IBM MQ 9.1.1 など) をアンインストールする必要があります。 詳しくは、異なる Continuous Delivery リリース間のマイグレーションを参照してください。

このシナリオで使用されるハードウェアおよびソフトウェア

オペレーティング・システム

Windows 10
Hostname: `johndoe1.fyre.<yourdomainname>.com`

キュー・マネージャー

QM80
IBM MQ 8.0.0 で作成されたもの。IBM MQ 8.0.0 のままとなります。

QMMIG
IBM MQ 8.0.0 で作成されたもの。IBM MQ 9.1.0 にマイグレーションされます。

QM910
IBM MQ 9.1.0 で作成されたもの。IBM MQ 9.1.0 のままとなります。

新しいバージョンの IBM MQ を前のバージョンと横並びでインストールする。

IBM MQ 9.3 を、既存のバージョンの製品と横並びに同じマシン上にインストールする方法。 IBM MQ 9.1 インストールは、プライマリー・インストールとして指定されません。

始める前に

IBM MQ 9.1 がシステム上にインストールされていることを確認します。 IBM MQ 9.1 がインストールされていない場合に製品をインストールするには、Windows への IBM MQ サーバーのインストールの手順を使用します。

クライアントを明示的に選択できるように、「カスタム」オプションを選択しておく必要があります。 詳しくは、Windows の場合のインストール方法を参照してください。

また、**-n** パラメーターを指定して **setmqenv** コマンドを呼び出して、インストール名を `Installation1` に設定しておく必要があります。 コマンドを呼び出すにはバッチ・ファイルを使用すると便利です。

このタスクについて

すべてのキュー・マネージャー・データは、共通のディレクトリー構造に保管されます。 各バージョンの IBM MQ の実行可能コードは異なるディレクトリー構造に保管されていますが、バージョンに関係なく、すべてのキュー・マネージャーのデータは、 `MQ_DATA_PATH=C:\ProgramData\IBM\MQ` に保管されます。

IBM MQ 9.3 をインストールする場合 :

手順

1. 管理者としてログインします。
   IBM MQ 9.3 は、`C:\Program Files\IBM\MQ` という初期設定のディレクトリーにインストールされます。
2. ダウンロード・ファイルがあるディレクトリー (例えば、 `C:\downloads\mq9300`) に移動します。
3. ダウンロードしたファイルを unzip します。
   ファイルは、`MQServer` という新しいサブディレクトリーに解凍されます。
4. 新しいディレクトリーに切り替え、コマンド **setup.exe** を実行してインストーラーを開始します。
   a)「ソフトウェア要件」をクリックして、必要なソフトウェアがエンタープライズにインストールされていることを確認します。
   詳しくは、Windows の場合の要件の確認を参照してください。 このシナリオでは、システムに必要条件があります。
   b)「ネットワーク構成」をクリックします。
   このシナリオでは、マシンはドメインの一部ではないので、ドメイン・ユーザーを指定する必要はありません。質問への回答は No になります。

c)「*IBM MQ* インストール」をクリックします。

d)「*IBM MQ* インストーラーの起動」をクリックします。

インストーラーは、 システムに他のインストール済み環境があることを検出し、以下のメッセージを表示します。

アップグレードまたはインストール
既存のインストール済み環境をアップグレードするか、またはそれと共存するように新規バージョンをインストールする

このシナリオでは、他のインストール済み環境をそのまま残すため、 最初の項目である「既存のインストール済み環境には影響を与えずにインストールする」を選択します。

e)「**次へ**」をクリックし、ライセンスを受け入れます。

f) インストール・オプションとして「カスタム」を選択します。

詳しくは、Windows の場合のインストール方法を参照してください。

g) **次へ** をクリックします。

では

インストールの詳細
インストールの詳細を定義します

に、以下が表示されます。

インストール環境の名前
Installation2

プログラム・ファイル用のインストール・フォルダー
`C:\ProgramFiles\IBM\MQ`

インストールするフィーチャーのリストが表示されます。 項目によっては以下のことが当てはまるため、注意してください。

- デフォルトではインストールされない。
- インストールする場合は、選択する必要がある。

このシナリオでは、「*MQI* クライアント」を選択します。また、その他の場合も、このオプションを選択してください。

h) **次へ** をクリックします。

「*IBM MQ* をインストールする準備ができました」という画面に、 インストール名、ロケーション、インストールするコンポーネントなどの要約が表示されます。

i)「**インストール**」をクリックして、先に進みます。

インストール・ディレクトリー構造へのファイルのコピーが始まります。 ファイルをコピーすると、ダイアログ `The IBM MQ Installation Wizard has successfully installed IBM MQ` が表示されます。

j)「**完了 (Finish)**」 をクリックします。

インストール後に、ダイアログ `Welcome to the Prepare IBM MQ Wizard` が表示されます。

このシナリオでは Windows ドメインを使用していないため、デフォルトの「**いいえ**」を受け入れます。

IBM MQ Explorer を起動することを受け入れます。特別な理由がない限り、 デフォルト設定の「新規ワークスペースを使用して *MQ* エクスプローラーを開始する」を受け入れることができます。

タスクの結果

既存のバージョンの製品と共存するように、別のバージョンの IBM MQ for Windows を正常にインストールすることができました。

次のタスク

いずれかのバージョンでコマンドを使用するには、**setmqenv** コマンドを実行する必要があります。 詳細は 148 ページの『setmqenv コマンドを使用して、両方のバージョンの IBM MQ で実行する』 を参照してください。

setmqenv コマンドを使用して、両方のバージョンの IBM MQ で実行する

インストール・アクティビティーが完了し、IBM MQ コードが含まれているディレクトリーを検証できるようになりました。

始める前に

IBM MQ 8.0.0 製品 Installation1 が C:\Program Files\IBM\WebSphere MQ\ に、 IBM MQ 9.1 製品 Installation2 が C:\Program Files\IBM\MQ に正しくインストールされていることを確認します。

このタスクについて

dspmqinst というコマンドを使用して、システムにインストール済みバージョンのインストール情報を表示するには、ディレクトリーを調べ、 **dspmqver** コマンドを使用して、バージョン情報を表示する必要がありません。

Windows システムでの情報はレジストリーに保管され、インストールの構成情報は Computer\HKEY_LOCAL_MACHINE\SOFTWARE\IBM\WebSphere MQ\Installation というキーに保管されます。

重要 : このキーは、直接編集しないでください。また参照することもできません。

手順

1. **dspmqinst** コマンドを使用して、インストール情報を表示します。

```
InstName:      Installation1
InstDesc:
Identifier:    1
InstPath:      C:\Program Files\IBM\WebSphere MQ
Version:       8.0.0.9
Primary:       Yes
State:         Available
MSIProdCode:   {74F6B169-7CE6-4EFB-8A03-2AA7B2DBB57C}
MSIMedia:      8.0 Server
MSIInstanceId: 1

InstName:      Installation2
InstDesc:
Identifier:    2
InstPath:      C:\Program Files\IBM\MQ
Version:       9.1.0.0
Primary:       No
State:         Available
MSIProdCode:   {5D3ECA81-BF8D-4E80-B36C-CBB1D69BC110}
MSIMedia:      9.1 Server
MSIInstanceId: 1
```

   注 : それぞれのインストール名 (InstName) は重要です。

2. デフォルト (またはプライマリー) インストールの **dspmqver** を使用して IBM MQ のバージョンを表示します。

```
C:\> dspmqver
Name:        WebSphere MQ
Version:     8.0.0.9
Level:       p800-009-180321.1
BuildType:   IKAP - (Production)
Platform:    WebSphere MQ for Windows (x64 platform)
Mode:        64-bit
O/S:         Windows 10 Professional x64 Edition, Build 18363
```

```
InstName:    Installation1
InstDesc:
Primary:     Yes
InstPath:    C:\Program Files\IBM\WebSphere MQ
DataPath:    C:\ProgramData\IBM\MQ
MaxCmdLevel: 802
LicenseType: Production
```

コマンドを実行した後、メッセージ `Note there are a number (1) of other installations, use the '-i' parameter to display them` を受け取ります。

3. コマンド `C:\> where dspmqver` を発行すると、プライマリー・インストールに関する情報が表示されます。

```
C:\> where dspmqver
C:\Program Files\IBM\WebSphere MQ\bin64\dspmqver.exe
C:\Program Files\IBM\WebSphere MQ\bin\dspmqver.exe
```

4. IBM MQ 9.1 製品に関する情報を表示するには、コマンド `C:\> setmqenv -n Installation2` を発行します。
5. コマンド `C:\> where dspmqver` を再度発行すると、2 番目のインストールに関する情報が表示されます。

```
C:\> where dspmqver
C:\Program Files\IBM\MQ\bin64\dspmqver.exe
C:\Program Files\IBM\MQ\bin\dspmqver.exe
```

6. コマンド `C:\dspmqver` を再発行してください。

   以下が表示されます。

```
C:\> dspmqver
Name:        IBM MQ
Version:     9.1.0.0
Level:       p910-L180705
BuildType:   IKAP - (Production)
Platform:    IBM MQ for Windows (x64 platform)
Mode:        64-bit
O/S:         Windows 10 Professional x64 Edition, Build 18363
InstName:    Installation2
InstDesc:
Primary:     No
InstPath:    C:\Program Files\IBM\MQ
DataPath:    C:\ProgramData\IBM\MQ
MaxCmdLevel: 910
LicenseType: Production
```

7. コマンド `C:\ set MQ` を実行します。**setmqenv** を使用した後に、2 つ目のインストールに関する情報が表示されます。

```
C:\> set MQ
MQ_DATA_PATH=C:\ProgramData\IBM\MQ
MQ_ENV_MODE=64
MQ_FILE_PATH=C:\Program Files\IBM\MQ
MQ_INSTALLATION_NAME=Installation2
MQ_INSTALLATION_PATH=C:\Program Files\IBM\MQ
MQ_JAVA_DATA_PATH=C:\ProgramData\IBM\MQ
MQ_JAVA_INSTALL_PATH=C:\Program Files\IBM\MQ\java
MQ_JAVA_LIB_PATH=C:\Program Files\IBM\MQ\java\lib64
MQ_JRE_PATH=C:\Program Files\IBM\MQ\java\jre
```

指定された構文を使用して **setmqenv** コマンドを実行する、バッチ・ファイルを作成することができます。 このバッチ・ファイルが PATH のディレクトリーにあることを確認してください。例えば、`C:\WinTools` などです。

例えば、次のコンテンツを使用してバッチ・ファイル `set-mq-910.bat` を作成することができます。

```
REM Setup the environment to run MQ 9.1
CALL "C:\Program Files\IBM\MQ\bin\setmqenv" -n Installation2
REM Adding Samples to the path
SET PATH=%PATH%;%MQ_FILE_PATH%\Tools\c\Samples\Bin;%MQ_FILE_PATH%\Tools\c\Samples\Bin64
;%MQ_FILE_PATH%\Tools\jms\samples;%MQ_JAVA_INSTALL_PATH%\bin\ dspmqver -f 2
```

注:

a. **setmqenv** を呼び出すときには、「CALL」引数を使用する必要があります。 この引数が指定されていない場合、**setmqenv** の処理によってバッチが終了するため、後続のステートメントが実行されません。 つまり、CALL 引数を使用することにより、バッチ・ファイル内の他のステートメントの処理が可能になります。

b. IBM MQ ディレクトリーを PATH に追加した場合 (例えば、C サンプルの場所 PATH= ...;C:\Program Files\IBM\MQ\tools\c\Samples\Bin; ... など)、このディレクトリーは、コマンドの次回実行時に **setmqenv** によって除去されます。

IBM MQ 9.1 から C のサンプルを実行できるようにするには、サンプルのディレクトリーを PATH に戻すために、前述のバッチ・ファイルの最後の行が必要になります。

MQ_FILE_PATH が使用されることで、IBM MQ 9.1: SET PATH=%PATH%;%MQ_FILE_PATH%\tools\c\Samples\Bin の適切なディレクトリー構造を使用していることにも注意してください。

キュー・マネージャーの作成

crtmqm コマンドを使用してキュー・マネージャーを作成する方法。 IBM MQ Explorer を使用して、このタスクを実行できます。

始める前に

dspmq コマンドに **-o** *installation* パラメーターと **-s** パラメーターを指定して使用すると、現在のキュー・マネージャーのインストール名と状況が表示されます。

```
C:\> dspmq -o installation -s
QMNAME(QM80)                                  STATUS(Running) INSTNAME(Installation1)
  INSTPATH(C:\Program Files\IBM\WebSphere MQ) INSTVER(9.1.0.9)
QMNAME(QMMIG)                                 STATUS(Running) INSTNAME(Installation1)
  INSTPATH(C:\Program Files\IBM\WebSphere MQ) INSTVER(9.1.0.9)
```

このタスクについて

以下のプロセスを実行するには、Windows のコマンド・プロンプトを開き、自分自身を管理者としてセットアップする必要があります。 コマンド・プロンプトを使用しないで **crtmqm** コマンドを実行しようとすると、 メッセージ AMQ7077 (「要求された操作の実行は許可されていません」) を受け取ります。

手順

1. 「**開始**」>「**Windows システム**」>「**コマンド・プロンプト**」>「**その他**」>「**管理者として実行 (Run as administrator)**」を選択します

   作成されるウィンドウのタイトルは、「管理者*:* コマンド・プロンプト」です。

   注: コマンド・プロンプト内の IBM MQ のバージョンは IBM MQ 9.1 です。
2. **setmqenv** コマンド、または作成したバッチ・ファイル **set-mq-930** を実行します。

   詳細は、148 ページの『setmqenv コマンドを使用して、両方のバージョンの IBM MQ で実行する』を参照してください。

   いずれの場合も、バージョン 9.3.0.0 が表示されます。
3. コマンド C:\> crtmqm -u SYSTEM.DEAD.LETTER.QUEUE QM930 を発行します。

   次の出力が表示されます。

```
IBM MQ queue manager created.
Directory 'C:\ProgramData\IBM\MQ\qmgrs\QM930' created.
The queue manager is associated with installation 'Installation2'.
Creating or replacing default objects for queue manager 'QM930'.
Default objects statistics : 87 created. 0 replaced. 0 failed.
Completing setup.
Setup completed.
```

4. 次のコマンドを発行して、キュー・マネージャー C:\> strmqm QM930 を開始します。

   次の出力が表示されます。 キュー・マネージャーが実行されているインストール済み環境とバージョンを示す行に注目してください。

```
IBM MQ queue manager 'QM930' starting.
The queue manager is associated with installation 'Installation2'.
5 log records accessed on queue manager 'QM930' during the log replay phase.
Log replay for queue manager 'QM930' complete.
Transaction manager state recovered for queue manager 'QM910'.
IBM MQ queue manager 'QM930' started using V9.3.0.0.
```

5. コマンド C:\> dspmq -o installation -s を再発行して、インストールされているキュー・マネージャーを表示します。

```
C:\> dspmq -o installation -s
QMNAME(QM80)                                      STATUS(Running) INSTNAME(Installation1)
  INSTPATH(C:\Program Files\IBM\WebSphere MQ) INSTVER(9.1.0.9)
QMNAME(QMMIG)                                     STATUS(Running) INSTNAME(Installation1)
  INSTPATH(C:\Program Files\IBM\WebSphere MQ) INSTVER(9.1.0.9)
QMNAME(QM910)                                     STATUS(Running) INSTNAME(Installation2)
  INSTPATH(C:\Program Files\IBM\MQ) INSTVER(9.3.0.0)
++ Cannot use MQ 9.3.0 administrative commands to run an MQ 9.1 queue manager
```

   重要 : 使用しているものとは異なるバージョンの IBM MQ で管理コマンドを使用することはできません。

   これを試行しようとすると、 AMQ5691E: キュー・マネージャー <qmname> は別のインストール環境 (<installation name>) に関連付けられています。というメッセージが表示されます。

新しいバージョンの IBM MQ へのキュー・マネージャーのマイグレーション

IBM MQ 9.3 をインストールした後は、IBM MQ 9.1 キュー・マネージャーをマイグレーションおよびアップグレードして、 IBM MQ 9.3 で使用できるようにする必要があります。

このタスクについて

次の 2 つの主なステップを実行する必要があります。

1. **setmqm** コマンドを使用して、キュー・マネージャーを適切なインストール済み環境 (つまり IBM MQ のバージョン) に関連付けます。
2. 適切なバージョンの下で **strmqm** コマンドを使用します。これにより、新しいバージョンに対応するようにキュー・マネージャーのデータが更新されます。

このシナリオでは、 IBM MQ 9.1 を使用して作成されたキュー・マネージャー QMMIG をマイグレーション手順の図として使用します。

重要 : キュー・マネージャーを新しいバージョンの IBM MQ にマイグレーションした後は、以前のバージョンの IBM MQ でそれを使用することはできなくなります。 マイグレーション・プロセスは、いくつかのファイルとオブジェクト定義を変更します。それらを元に戻すことはできません。

手順

1. **dmpmqcfg** コマンドを使用して、キュー・マネージャーのバックアップを取ってください。

   詳しくは、IBM MQ キュー・マネージャー・データのバックアップと復元、およびキュー・マネージャー構成のバックアップを参照してください。

   a) デフォルトの属性を含むすべての属性 (出力に含まれない **setmqaut** は除く) を指定するには、以下のコマンドを指定します。

```
dmpmqcfg -m QMgr -a > QMgr.dmpmqcfg.out.all.mqsc
```

   b) **setmqaut** 形式で属性をキャプチャーするには、次のコマンドを発行します。

```
dmpmqcfg -m QMgr -o setmqaut > QMgr.dmpmqcfg.setmqaut.bat
```

注 : setmqaut コマンドを含む出力ファイルには、コマンドごとに、キュー・マネージャーの名前が含まれています。 そのため、コマンドを別のキュー・マネージャーに復元する場合は、そのファイルを編集して、適切なキュー・マネージャー名を指定する必要があります。

2. 復元手順:

   a) **runmqsc** のコマンドの場合、以下を実行します。

   ```
   runmqsc Qmgr < QMgr.dmpmqcfg.out.mqsc
   ```

   または

   ```
   runmqsc Qmgr < QMgr.dmpmqcfg.out.all.mqsc
   ```

   b) **setmqaut** コマンドの場合、以下を実行します。

   ```
   QMgr.dmpmqcfg.setmqaut.bat
   ```

3. マイグレーションするキュー・マネージャーは IBM MQ 9.1 にあるので、実行環境を IBM MQ 9.3 に設定するスクリプトを実行する必要があります。

   ```
   C:\> set-mq-80
   ```

   a) コマンド C:\> dspmqver を発行して、 IBM MQ 9.1 上でキュー・マネージャーが実行しているバージョンを確認します。

   b) コマンド C:\> where dspmq を発行して、キュー・マネージャーが実行されていることを確認します。

   ```
   C:\Program Files\IBM\WebSphere MQ\bin64\dspmq.exe
   C:\Program Files\IBM\WebSphere MQ\bin\dspmq.exe
   ```

   c) 以下のコマンドを発行します。 C:\> dspmq -m QMMIG -o installation -s

   ```
   QMNAME(QMMIG)               STATUS(Running) INSTNAME(Installation1)
   INSTPATH(C:\Program Files\IBM\WebSphere MQ) INSTVER(9.1.0.9)
   ```

   d) 以下のコマンドを発行します。 C:\> runmqsc QMMIG

   ```
   display qmgr cmdlevel version
          1 : display qmgr cmdlevel
   AMQ8408: Display Queue Manager details.
     QMNAME(QMMIG)                   CMDLEVEL(910)
     VERSION(09001009)
   end
   ```

   また、CMDLEVEL が IBM MQ 9.1 であることに注目してください。

   e) コマンド C:\ endmqm -i QMMIG を発行して、キュー・マネージャーを停止します。

   以下のメッセージを受け取ります。

   ```
   WebSphere MQ queue manager 'QMMIG' ending.
   WebSphere MQ queue manager 'QMMIG' ended.
   ```

4. IBM MQ 9.3 コマンドを実行するように環境を変更します。そのためには、バッチ・ファイルを作成した場合はコマンド C:\> set-mq-930 を発行し、次に **setmqenv** コマンドを使用し、 **dspmqver** コマンドを発行してバージョンを確認します。

5. コマンド C:\> dspmq -o installation -s を発行して、キュー・マネージャーの状況を表示します。

   以下の出力が返されます。

   ```
   QMNAME(QM91)                    STATUS(Running)INSTNAME(Installation1)
   INSTPATH(C:\Program Files\IBM\WebSphere MQ) INSTVER(9.1.0.9)

   QMNAME(QMMIG)                             STATUS(Ended immediately) INSTNAME(Installation1)
   INSTPATH(C:\Program Files\IBM\WebSphere MQ) INSTVER(9.1.0.9)
   ```

```
QMNAME(QM910)                             STATUS(Running) INSTNAME(Installation2)
INSTPATH(C:\Program Files\IBM\MQ) INSTVER(9.3.0.0)
```

重要 : キュー・マネージャー QMMIG はまだ Installation1 (IBM MQ 9.1) に関連付けられていますが、 Installation2 は IBM MQ 9.3 に関連付けられています。

そのため、キュー・マネージャー QMMIG を Installation1 から関連付け解除して、Installation2 に関連付ける必要があります。

6. 次のコマンドを発行して、キュー・マネージャー QMMIG を Installation2 に関連付けます。

```
C:\> setmqm -m QMMIG -n Installation2
```

以下のメッセージを受け取ります。

```
The setmqm command completed successfully.
```

これにより、QMMIG が IBM MQ 9.3 に関連付けられたことが分かります。

7. コマンド C:\> strmqm QMMIG を発行して、キュー・マネージャー QMMIG を開始します。

以前に旧バージョンで使用されていたキュー・マネージャーで IBM MQ 9.3 **strmqm** コマンドが実行されるのはこれが初めてなので、マイグレーションが実行されます。

次のような出力が表示されます。

```
IBM MQ queue manager 'QMMIG' starting.
The queue manager is associated with installation 'Installation2'.
5 log records accessed on queue manager 'QMMIG' during the log replay phase.
Log replay for queue manager 'QMMIG' complete.
Transaction manager state recovered for queue manager 'QMMIG'.
Migrating objects for queue manager 'QMMIG'.
Default objects statistics : 5 created. 0 replaced. 0 failed.
IBM MQ queue manager 'QMMIG' started using V9.3.0.0.
```

8. 次のコマンド C:\> runmqsc QMMIG を発行して、キュー・マネージャーの属性を表示し、VERSION フィールドと CMDLEVEL フィールドをメモします。

```
display qmgr cmdlevel version
     1 : display qmgr cmdlevel version
AMQ8408I: Display Queue Manager details.
   QMNAME(QMMIG)                           CMDLEVEL(930)
   VERSION(09300000)
end
```

タスクの結果

これで、キュー・マネージャーは新しいバージョンの製品に正常にマイグレーションされました。

IBM MQ 9.3 へのフィックスパックのインストール

IBM MQ のマルチバージョン・インストレーションのあるシステム上に、インストールされた IBM MQ 9.3 の中にフィックスパックをインストールする方法。

始める前に

キュー・マネージャー QMMIG が IBM MQ 9.3 にマイグレーションされていることを確認します。 詳しくは、151 ページの『新しいバージョンの IBM MQ へのキュー・マネージャーのマイグレーション』を参照してください。

このタスクについて

このシナリオでは、別のバージョン (IBM MQ 9.1) がインストールされており、その別のバージョンで実行されているキュー・マネージャーは停止されません。これは、 IBM MQ 9.3.0.n の保守アクティビティーを実行しながら、それらの他のバージョンを引き続き使用できることを示すためです。

更新のインストール時に拡張オプションは選択されないことに注意してください。

手順

1. 管理者としてログインします。
   a) バッチ・ファイルを作成した場合は、コマンド `C:\> set-mq-930` を発行するか、 **setmqenv** コマンドを使用して、 IBM MQ 9.3.0.n を使用していることを確認します。ここで、 n は、このシナリオでは `0` です。
   b) コマンド `C:\> dspmq -o installation -s` を発行して、キュー・マネージャーの状況を表示します。
   以下の出力が返されます。

```
QMNAME(QM80)                    STATUS(Running)INSTNAME(Installation1)
INSTPATH(C:\Program Files\IBM\MQ) INSTVER(9.1.0.0)

QMNAME(QMMIG)                             STATUS(Ended unexpectedly)
INSTNAME(Installation2)
INSTPATH(C:\Program Files\IBM\MQ) INSTVER(9.3.0.0)

QMNAME(QM910)                             STATUS(Ended immediately) INSTNAME(Installation2)
INSTPATH(C:\Program Files\IBM\MQ) INSTVER(9.3.0.0)
```

2. IBM MQ 9.3.0 プロセスを停止します。
   プロセスを停止する方法について詳しくは、Windows での保守の適用と削除を参照してください。
   要約すると、以下のようになります。
   - キュー・マネージャー QMMIG および QM930 に対して **endmqm immediate** コマンドを発行します。
   - タスクバーの IBM MQ アイコンを右クリックし、「**停止**」 **IBM MQ** をクリックして、インストールの IBM MQ サービスを停止します。
3. フィックスパックを見つけます。 このシナリオでは、 IBM MQ 9.3.0.5 を使用します。
   最新のリストについては、 IBM MQ の推奨フィックス を参照してください。
   - Continuous Delivery リリース、および Continuous Delivery リリースのフィックス・リスト
   - Long Term Support リリースおよび Long Term Support リリースのフィックス・リストの累積セキュリティー更新

   注 : 最新のフィックスパックを使用していることを確認するには、該当するフィックスパックにアクセスしてダウンロードしてください。
   a) 該当するタブをクリックします。
   このシナリオでは、 *V9.3.0.5 LTS* です。
   b) Fix Central または Passport Advantage から Windows ソフトウェアをダウンロードします。 製品全体をダウンロードする必要がある場合は、
   このシナリオでは、ファイル・セットをディレクトリー `C:\downloads\mq9305` に入れます。ファイル名は `9.3.0-IBM-MQ-Win64-FP0005.zip` です。
4. `.zip` ファイルからファイルを解凍し、コマンド **IBM-MQ-9.3.0-FP0005.exe** を実行します。
   「*Install Anywhere*」 ダイアログが表示されます。このダイアログには、以下の情報が含まれています。
   `InstallAnywhere is preparing to install ...`
   準備が終わるまで待機する必要があります。これには数分かかる場合があります。
5. ウィンドウ *IBM MQ (フィックスパック 9.3.0.5 ファイル)*」 が表示されたら、「**OK**」 をクリックして続行します。
   a)「概要」セクションが表示されたら、「**次へ**」をクリックします。
   b)「インストール・タイプ」セクションが表示されたら、 自分のエンタープライズに最適なオプション (ほとんどの場合、「**ファイルのロードとフィックスパックの適用**」) を選択して、「**次へ**」をクリックします。
   c)「情報」セクションが表示されたら、「**次へ**」をクリックします。

d) 出力先フォルダー セクションが表示されたら、初期設定のロケーション `C:\Program Files\IBM\source\MQ 9.3.0.5` を選択して、 **次へ**をクリックします。

e)「拡張オプション」セクションが表示されたら、「**次へ**」をクリックします。

f)「インストール前の要約」セクションが表示されたら、表示された情報を確認し、「**インストール**」をクリックします。

g) コードがロードされるまで待ちます。

「ロード中」セクションには進行標識があり、 プロセスが完了すると「ロードが完了しました」にチェック・マークが付いて、「フィックスパックの適用」セクションが表示されます。

h)「**完了**」をクリックします。

システムには複数のインストール済み環境があるので、 アップグレードするインストール済み環境を選択できるダイアログが表示されます。 この場合は、 `Installation2 (9.3.0.0)`です。

i)「**OK**」をクリックします。

j)「バックアップ・フォルダー」でデフォルトを受け入れ、「**適用**」をクリックします。

進行状況表示を示すダイアログがさらに表示され、最終ダイアログには フィックスパック *9.3.0.5* が適用されたことが示されます。 *[* 完了 *]* をクリックして終了します。

k)「**完了 (Finish)**」 をクリックします。

タスクの結果

既存のバージョンの製品と共存するように、IBM MQ for Windows のバージョンを正常にアップグレードすることができました。

Managed File Transfer シナリオ

一般的な Managed File Transfer トポロジーの概要、およびシステムをセットアップしてテスト・メッセージを転送する方法を示すことによって Managed File Transfer 機能の使用法を示すシナリオ。

- 一般的なトポロジー
- 基本サーバーの構成

MFT の一般的なトポロジー

このセクションでは、一般的な Managed File Transfer トポロジーについて説明します。 各図の両方向矢印は、キュー・マネージャーへの接続を表しています。

キュー・マネージャーの接続オプションについて詳しくは、158 ページの『接続に関する考慮事項』を参照してください。

1 つのキュー・マネージャーが存在する基本トポロジー

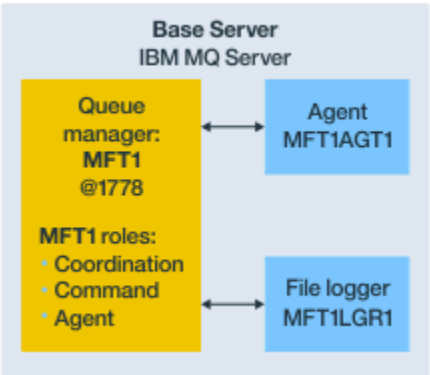


図 *25. 1* つのキュー・マネージャーが存在する基本トポロジー

基本トポロジーは、調整キュー・マネージャーを含んだ完全な構成を表します。 構成名は調整キュー・マネージャーの名前と同じです。 調整キュー・マネージャー名が MFT1 であれば、構成名も MFT1 です。

基本トポロジーは、最初に完成させる Managed File Transfer 構成です。 基本構成が完成した後、ファイルを交換するために、リモート・サーバー上のパートナー・エージェントが基本構成に追加されます。

基本トポロジーでは、基本トポロジー・サーバー外のファイルを交換しません。 ただし、基本トポロジーでは、同じサーバー内の異なる場所にファイルを移動できるので、開発の目的で使用することもできます。

1 つのパートナー・エージェントが存在する基本トポロジー

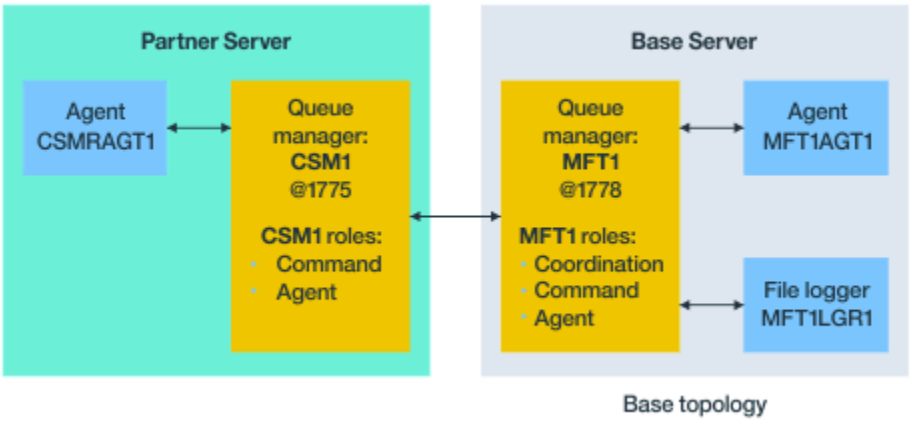


図 *26. 1* つのパートナー・エージェントが存在する基本トポロジー

このトポロジーでは、2 つのエージェントの間でファイルを交換できます。 最初に追加されたエージェントと同様の方法で、さらに別のパートナー・エージェントを追加できます。

Managed File Transfer キュー・マネージャーの 3 つの役割すべてに単一のキュー・マネージャーを使用することも、特定の役割に専用のキュー・マネージャーを使用することもできます。

例えば、調整キュー・マネージャー役割に専用のキュー・マネージャーを 1 つ使用し、コマンド役割とエージェント役割で 2 つ目のキュー・マネージャーを共有することもできます。

基本構成とは別個のサーバーにあるリモート・エージェント・キュー・マネージャーと基本構成の調整キュー・マネージャーの間の接続は、IBM MQ クライアント・チャネル、つまり MQI チャネルとして構成する必要があります。

調整キュー・マネージャーへの接続は、**fteSetupCoordination** コマンドで確立されます。 パートナー・サーバーで調整キュー・マネージャー接続が IBM MQ クライアント・チャネルとして構成されていなければ、パートナー・エージェント・サーバーから **fteListAgents** などのコマンドを発行すると失敗します。

別個の調整キュー・マネージャーと 1 つのパートナー・エージェントが存在する基本トポロジー

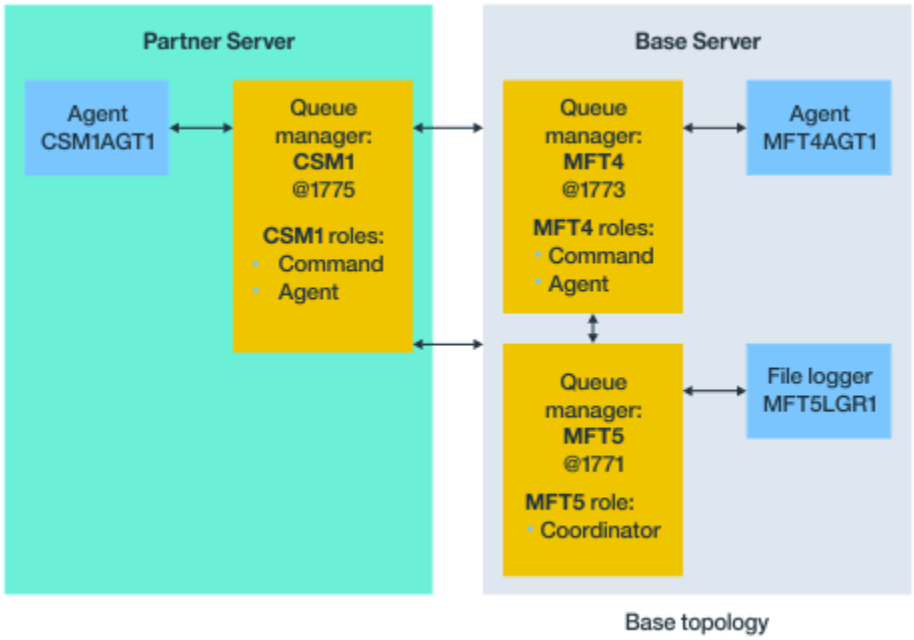


図 *27.* 別個の調整キュー・マネージャーと *1* つのパートナー・エージェントが存在する基本トポロジー

図 3 の基本トポロジーでは、基本サーバーのキュー・マネージャー MFT4 はコマンド役割とエージェント役割として共有され、キュー・マネージャー MFT5 は調整キュー・マネージャー役割専用です。

基本トポロジー内のキュー・マネージャー MFT4 および MFT5 を含め、このトポロジーに含まれるすべてのキュー・マネージャー間に接続が存在しなければなりません。

パートナー・サーバーのキュー・マネージャーでは、キュー・マネージャー CSM1 がエージェント・キュー・マネージャーとコマンド・キュー・マネージャーの役割を持っています。

このトポロジーでは、2 つのエージェントの間でファイルを交換できます。 図に示すように、各パートナー・エージェントをキュー・マネージャーに接続する必要があります。 最初のパートナー・エージェントを追加したときと同様の方法で、さらに別のパートナー・エージェントを追加できます。

Managed File Transfer Agent パートナーが存在する基本トポロジー

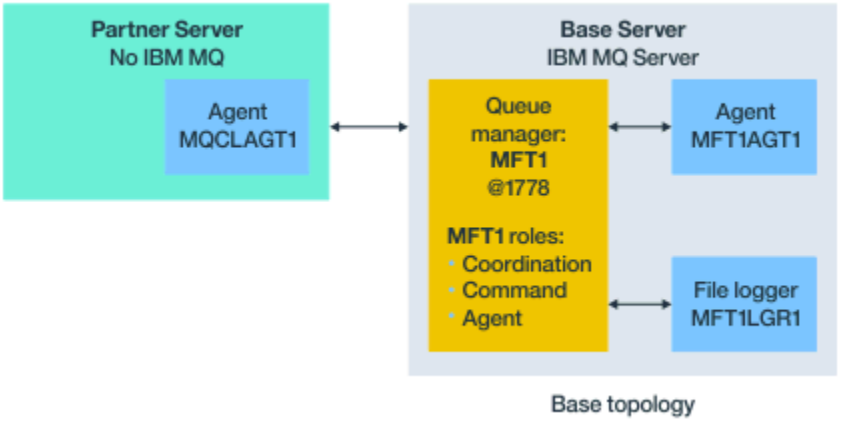


図 *28. Managed File Transfer Agent* パートナーが存在する基本トポロジー

このトポロジーでは、2 つのエージェントの間でファイルを交換できます。

パートナー・エージェント (図では MQCLAGT1) のサーバーには、IBM MQ サーバーがインストールされていません。

パートナー・エージェントは IBM MQ がインストールされたサーバーと同じコマンドを使用して構成されますが、以下に示すいくつかの例外があります。

- このパートナー・エージェントの構成では、基本キュー・マネージャーまたはキュー・マネージャーへの IBM MQ クライアント接続を使用する必要があります。
- パートナー・エージェント・サーバーで構成コマンドによって作成される IBM MQ 調整キュー・マネージャー役割定義を実行する必要はありません。 調整キュー・マネージャー定義は既に基本サーバーに存在します。

  ただし、以下の操作が必要です。

  – パートナー・サーバーでエージェントが作成されるときに生成されるエージェント・オブジェクト定義をコピーします。
  – 定義ファイルを基本構成サーバーに転送します。
  – 基本サーバーでエージェント・キュー・マネージャーの役割を持つキュー・マネージャーに定義を作成します。

    この例では、MFT1 は 3 つの役割すべてを果たすので、MFT1 キュー・マネージャーにエージェント MQCLAGT1 のオブジェクトを作成します。

  オブジェクト定義を基本サーバーにコピーする代わりに、エージェント MQCLAGT1 に対する **fteDefine** コマンドを、エージェント・キュー・マネージャーが置かれている基本サーバーで実行することもできます。 **fteDefine** コマンドによって生成される定義を使用して、エージェント・キュー・マネージャーに必要なエージェント定義を作成します。

  例えば、この図の場合、パートナー・サーバーのエージェント・ディレクトリーにあるファイル MQCLAGT1_create.mqsc を基本構成サーバーにコピーして、必要なエージェント定義を MFT1 キュー・マネージャーに作成します。

  パートナー・エージェント・サーバーで構成を完了すると、Managed File Transfer 構成ディレクトリーと必要なプロパティー・ファイルが作成されます。

パートナー・サーバーでは、 IBM MQ 9.1.0 以降、 Managed File Transfer 再配布可能クライアント (Fix Central)をインストールできます。

注 : MQMFT 再配布可能クライアントは、既にパッケージ化されており、**genmqpkg** ユーティリティーを使用する必要がないという点で、 IBM MQ 再配布可能クライアントとは異なります。 詳しくは、再配布可能クライアントを参照してください。

接続に関する考慮事項

上記の図では、エージェントとキュー・マネージャーを結ぶ各線は、キュー・マネージャーへの接続を表しています。

この接続として、以下が考えられます。

- ローカル接続
- バインディング接続、つまりメッセージ・チャネル接続
- IBM MQ クライアント接続、つまり MQI 接続

構成で選択する接続のタイプは、指定するパラメーターによって決まります。

- キュー・マネージャー名パラメーターを指定して他の接続パラメーターを指定しなければ、バインディング接続を指定することになります。

  使用されるキュー・マネージャーが Managed File Transfer 構成上ローカルであれば、基本構成サーバーで使用される接続もローカル接続になります。
- キュー・マネージャー名パラメーターを、対応するホスト、ポート、およびチャネル名パラメーターとともに指定すると、IBM MQ クライアント接続を指定することになります。

エージェント・キュー・マネージャーと同じホストにエージェントが置かれるときは、バインディング・タイプ (結果的にローカル接続になる) を指定した方がより効率的です。

基本サーバーの構成

別個の構成キュー・マネージャーが存在する基本サーバーをセットアップする方法。

始める前に

以下の例は、次のことを前提としています。

- 158 ページの『接続に関する考慮事項』のセクションを確認して、構成におけるキュー・マネージャーへの接続のタイプへの影響を理解していること。
- IBM MQ インフラストラクチャーが機能していること。 キュー・マネージャーのセットアップについては、IBM MQ のキュー・マネージャーの構成を参照してください。
- IBM MQ セキュリティー・タスクが完了していること。

  ファイルに対するアクセス権限など、すべてのシステム・リソースに適切なセキュリティーが構成されていること。

  Managed File Transfer のセキュリティー構成については、Securing Managed File Transfer と、MFT エージェント・アクションに対するユーザー権限の制限を参照してください。
- すべての IBM MQ 接続は、 IBM MQ の構成後に、サンプル・プログラムを使用してメッセージを送受信するか、サンプル **amqscnxc** を使用して IBM MQ クライアント・タイプの接続をテストすることによってテストされます。

  amqscnxc サンプルは、サンプル・コードでチャネル接続を定義することによってキュー・マネージャーに接続します。これは、 Managed File Transfer が MQI クライアント・タイプまたは IBM MQ クライアント・タイプの接続を使用する場合の接続方法に似ています。
- この説明は、基本構成に使用するサーバーに IBM MQ の 1 つのバージョンがインストールされていることを前提としています。 基本サーバーに複数の IBM MQ インストール済み環境が存在する場合は、使用するバージョンの IBM MQ に対応する正しいファイル・パスを使用するよう注意してください。
- この説明で使用されるキュー・マネージャーは、接続認証を必要としません。

  接続認証を必要とせずに最初の構成を完了する方が簡単な場合がありますが、企業で即時に接続認証を使用する必要がある場合は、`MQMFTCredentials.xml` 資格情報ファイルの構成方法について、MFT および IBM MQ 接続認証 を参照してください。

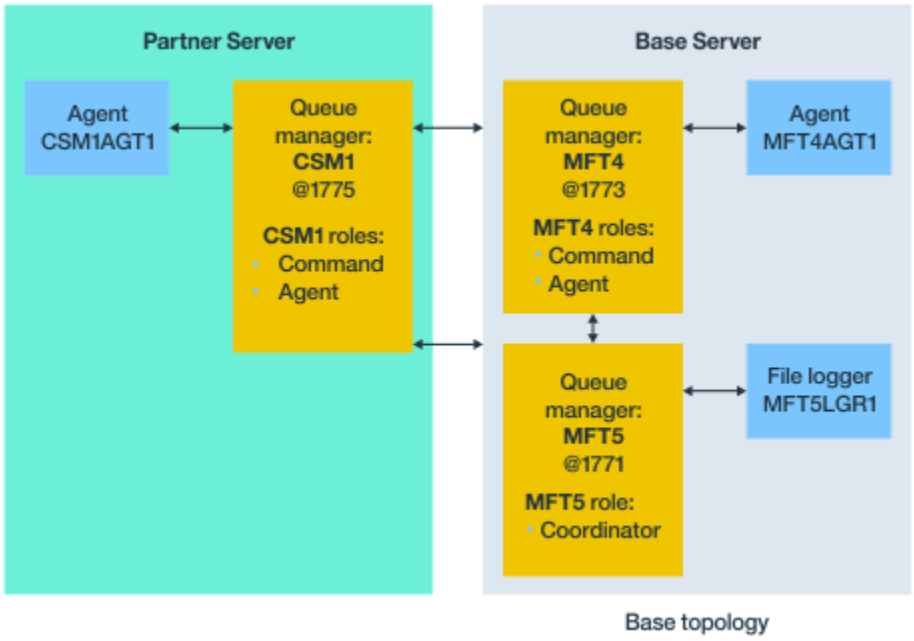


図 *29.* 別個の調整キュー・マネージャーと *1* つのパートナー・エージェントが存在する基本トポロジー

このタスクについて

この構成例でのキュー・マネージャー役割は、次のとおりです。

- 基本サーバー
  - キュー・マネージャー MFT5 は調整キュー・マネージャーです。
  - キュー・マネージャー MFT4 は、エージェント MFT4AGT1 のエージェント・キュー・マネージャーとして使用されるとともに、MFT5 構成における基本サーバー上のコマンド・キュー・マネージャーとしても機能します。
- パートナー・サーバー
  - キュー・マネージャー CSM1 は、エージェント CSM1AGT1 のエージェント・キュー・マネージャーと、MFT5 構成におけるパートナー・サーバー上のコマンド・キュー・マネージャーを兼ねています。
  - 基本サーバーのキュー・マネージャー MFT5 は、調整キュー・マネージャーです。

手順

1. 調整キュー・マネージャーの構成
2. コマンド・キュー・マネージャーの構成
3. エージェントのセットアップ
4. ロガーのセットアップ
5. パートナー・サーバーの構成

次のタスク

セットアップ例をテストできるように MQExplorer と MQMFT をセットアップします。

調整キュー・マネージャーの構成

ファイル転送を調整するための調整キュー・マネージャーを構成する方法。

始める前に

このシナリオ用にセットアップしたキュー・マネージャー間の完全な接続性を確保します。

このタスクについて

このタスクでは、調整キュー・マネージャー MFT5 をセットアップします。このセクションの説明は、IBM MQ インストール済み環境が 1 つ存在する環境で作業することを前提としています。

複数のインストール済み環境がある場合は、構成タスクを開始する前に、 setmqenv コマンドを使用して、IBM MQ パスを必要な IBM MQ のバージョンに設定する必要があります。

手順

1. Managed File Transfer 管理者としてログインします。
2. 次のコマンドを発行して調整キュー・マネージャーを指定し、構成ディレクトリー構造をセットアップします。

   ```
   fteSetupCoordination -coordinationQMgr MFT5
   ```

   調整キュー・マネージャーのディレクトリー
   C:*data*\mqft\config\MFT5

   coordination.properties file
   C:*data*\mqft\config\MFT5\coordination.properties

   このコマンドは、調整キュー・マネージャー C:*data*\mqft\config\MFT5\MFT5.mqsc に対して実行する必要がある MQSC コマンド・ファイルも生成します。
3. C:*data*\mqft\config\MFT5 ディレクトリーに移動します。
4. 次のコマンドを実行して、調整キュー・マネージャーとなるようにキュー・マネージャーを構成します。

   ステップ 161 ページの『2』 で発行したコマンドによって生成された MQSC コマンド・ファイルを指定する必要があります。

   ```
   runmqsc MFT5 < MFT5.mqsc > mft5.txt
   ```
5. 任意のエディターで mft5.txt 結果ファイルを開きます。 定義が正常に作成されていることを確認してください。

次のタスク

コマンド・キュー・マネージャーをセットアップします。

コマンド・キュー・マネージャーの構成

コマンド・キュー・マネージャーを構成する方法。

始める前に

調整キュー・マネージャーが構成済みであることを確認します。 詳しくは、 160 ページの『調整キュー・マネージャーの構成』 を参照してください。

このタスクについて

このタスクでは、コマンド・キュー・マネージャーを指定します。

手順

以下のコマンドを発行します。

```
fteSetupCommands -connectionQMgr MFT4
```

次のメッセージが表示されます。 BFGCL0245I: ファイル C:*data*\mqft\config\MFT4\command.properties が正常に作成されました。

コマンド・キュー・マネージャーは、追加の IBM MQ 定義を必要としません。 **fteSetupCommands** を実行すると、 command.properties ファイルが MFT5 構成ディレクトリーに作成されます。

次のタスク

エージェントをセットアップします。

エージェントのセットアップ

ファイル転送エージェント MFT4AGT1 と、実行する必要がある MQSC スクリプトを準備する方法。

始める前に

コマンド・キュー・マネージャーをセットアップしておく必要があります。 詳しくは、161 ページの『コマンド・キュー・マネージャーの構成』 を参照してください。

このタスクについて

このタスクでは、Windows ファイル転送エージェント MFT4AGT1 を準備します。

手順

1. 以下のコマンドを発行します。

   ```
   fteCreateAgent -agentName MFT4AGT1 -agentQMgr MFT4
   ```

   fteCreateAgent コマンドでエージェントを作成すると、エージェント MFT4AGT1 の agents ディレクトリーとサブディレクトリーが MFT5 ディレクトリーに追加されます。

   data\MFT5\agents\MFT4AGT1 ディレクトリーには、以下のものがあります。

   - agent.properties file
   - MFT4AGT1_create.mqsc ファイル、エージェントが必要とする IBM MQ 定義が含まれています。

2. *data*\MFT5\agents\MFT4AGT1 ディレクトリーに移動し、以下のコマンドを発行して、必要なエージェント・キュー・マネージャー定義を作成します。

   ```
   runmqsc MFT4 < MFT4AGT1_create.mqsc > mft4.txt
   ```

3. 任意のエディターで mft4.txt 結果ファイルを開き、定義が正常に作成されていることを確認します。
4. コマンド **fteStartAgent** MFT4AGT1 を入力してエージェントを開始します。
5. コマンド **fteListAgents** を入力してエージェントを表示します。

   次のような出力が表示されます。

   ```
   5655-MFT, 5724-H72 Copyright IBM Corp. 2008, 2024.  ALL RIGHTS RESERVED
   BFGPR0127W: No credentials file has been specified to connect to IBM MQ.
   Therefore, the assumption is that IBM MQ authentication has been disabled.
   Agent Name:     Queue Manager Name:     Status:
   MFT4AGT1        MFT4                    READY
   ```

   注 : Managed File Transfer 環境での接続認証を有効にしていない場合、BFGPR0127W メッセージは無視できます。

ftelistAgents コマンドを発行して、BFGCL0014W: 現在の選択基準に一致するエージェントが存在しません。というメッセージを受け取った場合、 MFT エージェントが **fteListAgents** コマンドによってリストされない場合の対処方法を参照。

次のタスク

ロガーをセットアップします。

ロガーのセットアップ

構成における転送アクティビティーに関する履歴情報と監査情報を保持するためには、ファイル・ロガーまたはデータベース・ロガーが必要です。 この例では、ファイル・ロガーを作成します。

始める前に

次のものをセットアップしておく必要があります。

- 構成キュー・マネージャー
- コマンド・キュー・マネージャー
- エージェント

手順

1. 以下のコマンドを発行します。

   ```
   fteCreateLogger -loggerQMgr MFT5 -loggerType FILE
   -fileLoggerMode CIRCULAR -fileSize 5MB -fileCount 3 MFT5lgr1
   ```

   fteCreateLogger コマンドを実行すると、*data*\mqft\config\MFT5\loggers ディレクトリーが作成され、 MFT5LGR1 サブディレクトリーが作成されます。

   MFT5LGR1 サブディレクトリーには logger.properties ファイルが格納されます。 このディレクトリーには、ロガーが必要とする IBM MQ 定義が含まれる MFT5LGR1_create.mqsc というファイルもあります。
2. *data*\mqft\config\MFT5\loggers\MFT5LGR1 ディレクトリーに移動します。
3. 関連する MQSC コマンド・ファイルを実行します。

   ```
   runmqsc MFT5 < MFT5_create.mqsc
   ```

   これにより、ロガーが必要とする定義が作成されます。

   a) オブジェクト定義の結果を調べて、必要なオブジェクトが正常に作成されたことを確認します。
4. コマンド **fteStartLogger** MFT5LGR1 を発行してロガーを開始します。
5. *data*\mqft\logs\MFT5\loggers\MFT5LGR1\logs にあるファイル output0.log の内容を確認します。
   ロガーに関する情報に続く最後のステートメントに、「BFGDB0023I: ロガーは開始アクティビティーを完了し、現在、実行中です」というメッセージがあるはずです。

   ロガーが初めて開始されたときは、output0.log にログ情報が書き込まれないことがあります。 output0.log ファイルが空の場合は、**fteStopLogger** MFT5LGR1 と入力してから **Enter** キーを押して、ロガーを再始動します。

   fteStartLogger MFTULGR1 と入力してから **Enter** キーを押して、ロガーを再始動します。 ファイル output0.log にデータが示されるようになります。

   エージェント版の output0.log ファイルでも、エージェントが初めて開始されたときは同じ動作になります。

fteStopAgent コマンドと **fteStartAgent** コマンドを使用して、エージェントを停止してから開始します。 その後、エージェントの output0.log ファイルにログ・データが書き込まれるようになります。

タスクの結果

この構成における調整キュー・マネージャーを設定した基本サーバーを構成しました。

次のタスク

次は、リモート・エージェントが含まれるパートナー・サーバーに関して同様の作業をします。

パートナー・サーバーの構成

基本サーバーに別個の調整キュー・マネージャーがある場合にパートナー・サーバーを構成する方法。

始める前に

構成キュー・マネージャーを含む基本サーバーをセットアップするためのすべてのタスクが完全に完了していることを確認します。

このタスクについて

IBM MQ とセキュリティー構成、および IBM MQ パスに関する前提と同じ前提が、パートナー・サーバーにも適用されます。

初めに **fteSetupCoordination** コマンドを使用して、MFT5 構成ディレクトリーをセットアップし、調整キュー・マネージャーを指定します。

手順

1. 次のコマンドを発行して、パートナー・サーバー構成ディレクトリーを作成します。

```
fteSetupCoordination -coordinationQMgr MFT5
-coordinationQMgrHost 177.16.20.15 -coordinationQMgrPort 1771
-coordinationQMgrChannel MQMFT.MFT5.SVRCONN
```

   注:

   a. 調整キュー・マネージャーがパートナー・サーバーとは異なるサーバーにある場合、基本サーバーの調整キュー・マネージャーへの接続は、クライアント接続として定義する必要があります。

      パートナー・サーバーで調整キュー・マネージャー接続を IBM MQ クライアント接続として定義しなければ、調整キュー・マネージャーに接続する Managed File Transfer コマンドはどれも、失敗することになります。

      調整キュー・マネージャーに接続するコマンドの例として、**fteListAgents** があります。

   b. IBM MQ 定義を作成する必要はありません。調整キュー・マネージャーが必要とする定義は、基本サーバーを構成したときに完了しているからです。

2. 次のコマンドを発行して、コマンド・キュー・マネージャーを指定します。

```
fteSetupCommands -connectionQMgr CSM1
```

   コマンド・キュー・マネージャーは、追加の IBM MQ 定義を必要としません。

3. 次のコマンドを発行して、パートナー・エージェント・キュー・マネージャーを指定し、パートナー・エージェント・キュー・マネージャーを作成します。

```
fteCreateAgent -agentName CSM1AGT1 -agentQMgr CSM1
```

4. CSM1AGT1 ディレクトリーに変更します。
5. 次のコマンドを発行して、エージェントが必要とする IBM MQ 定義を作成します。

```
runmqsc CSM1 < CSM1AGT1_create.mqsc > csm1.txt
```

   a) 任意のエディターでファイル csm1.txt を開き、すべてのエージェント必須定義が正常に作成されたことを確認します。
6. 次のコマンドを発行して、エージェントを開始します。

```
fteStartAgent CSM1AGT1
```

7. **fteListAgents** と入力してエージェントを表示します。
   次のような出力が表示されます。

```
C:\>fteListAgents
5655-MFT, 5724-H72 Copyright IBM Corp. 2008, 2024.  ALL RIGHTS RESERVED
BFGPR0127W: No credentials file has been specified to connect to IBM MQ. Therefo
re, the assumption is that IBM MQ authentication has been disabled.
Agent Name:     Queue Manager Name:     Status:
CSM1AGT1        CSM1                    READY
MFT4AGT1        MFT4                    READY
```

   注 : Managed File Transfer 環境での接続認証を有効にしていない場合、BFGPR0127W メッセージは無視できます。

   ftelistAgents コマンドを発行して、BFGCL0014W: 現在の選択基準に一致するエージェントが存在しません。というメッセージを受け取った場合、 MFT エージェントが **fteListAgents** コマンドによってリストされない場合の対処方法を参照。

   いずれかのエージェントの状況が UNREACHABLE である場合は、 エージェントが UNKNOWN 状態として表示される場合の対処方法を参照してください。

IBM MQ Explorer と MFT のセットアップ

このタスクは、IBM MQ Explorer を Managed File Transfer 構成に接続するためのものです。

手順

1. IBM MQ Explorer を開始します。
2. 左側の Navigator パネルで、スクロールダウンしてフォルダー Managed File Transfer を展開します。
   調整キュー・マネージャー MFT5 の項目があります。
3. MFT5 を右クリックして、「**接続**」を選択します。
   a) 表示されるドロップダウン・メニューで Agents を選択し、 MFT4AGT1 と CSMAGT1 の両方のエージェントが Ready 状態であることを確認します。

次のタスク

IBM MQ Explorer を使用してセットアップ例をテストします。

IBM MQ Explorer を使用したファイル転送のテスト

このタスクでは、前のトピックの説明に従って IBM MQ Explorer をセットアップした後に、 Managed File Transfer で IBM IBM MQ Explorer を使用してファイル転送をテストする方法の例を示します。

始める前に

システムが機能していることと、エージェントが READY で、IBM MQ Explorer が機能していることを確認します。 詳しくは、165 ページの『IBM MQ Explorer と MFT のセットアップ』 を参照してください。

このタスクについて

転送のテストに使用するファイルと、そのコピー先となるディレクトリーを決定します。 この例では、ファイル `test-file.txt` out of directory `C:\temp\mft` が使用されることを想定しています。

```
C:\temp\mft> dir *
Date stamp 61 test-file.txt
1 File(s) 61 bytes
```

手順

1. Windows で IBM MQ Explorer を開始します。
2. 左側のナビゲーター・パネルで、`Managed File Transfer` フォルダーを展開します。
   調整キュー・マネージャー MFT5 の項目があります。
3. MFT5 を右クリックして、「**接続**」を選択します。
4. 接続されたら、MFT5 を右クリックして、「**新規の転送**」を選択します。
   a) プルダウン・メニューを使用して、ソース・エージェントとして MFT4AGT1 を、宛先エージェントとして CSMAGT1 を選択します。
   b) **次へ** をクリックします。
   c) 次のウィンドウで、「**追加**」をクリックします。
   幅広のダイアログが表示されます。 左側は `Source` 用で、右側は `Destination` 用です。
5. 「`Source`」パネルで、以下を行います。
   a) ファイルはテキストなので、「**テキスト転送**」を選択します。
   b)「**参照**」を選択してファイルを見つけます。
   この場合、当該ファイルは `C:\temp\mft\test-file.txt` になります。

   重要 : `Destination` パネルに入力する必要があるため、 「**OK**」 をクリックしないでください。
6. 「`Destination`」パネルで、以下を行います。
   a) 宛先でのファイルに付ける名前を入力します (例えば `test-file.txt`)。
   相対パスの使用がサポートされています。 絶対パスの先頭部分は、宛先エージェントを開始するユーザー ID のホーム・ディレクトリーです。
   b) このオプションが必要な場合は、 `Overwrite files if present` を選択します。
   c)「**OK**」をクリックします。
   選択したファイルが「**新規の転送**」パネルに表示されます。
7. MFT5 構成メニューが閉じていて +MFT5 が表示されている場合は、「**+**」符号をクリックしてメニューを展開します。
8. MFT 構成を選択したままにしておきます。
   次に、以下の手順を実行して、転送の状況を確認します。
9. 調整キュー・マネージャー MFT5 の項目の中の「**転送ログ**」をクリックします。
10.「**転送ログ**」 上部パネルのすぐ下にある `Managed File Transfer - Current Transfer progress panel` の状況を確認し、転送が完了するまで待ちます。
    転送が成功を示していて、背景が緑色であれば、構成のテストを正常に完了したことになります。
    転送が失敗して背景が赤色であれば、エラーが発生したことになります。
    ほとんどの場合、上部「**転送ログ**」パネルの下のスクロール・バーを使用して、失敗の理由の要約を見ることができます。
    a) 転送が失敗した理由を判別できない場合は、上部「**転送ログ**」パネルで転送の項目をダブルクリックします。
    b) 表示されるポップアップ・パネルの左ペインで XML を選択します。

c) 情報をスクロールして、エラーの原因を判別します。

d) 必要な修正を行った後に、転送を再度テストします。

IBM MQ Internet Pass-Thru 入門

これらのシナリオでは、いくつかの単純な IBM MQ Internet Pass-Thru (MQIPT) 構成をセットアップする方法を示します。 これらの作業は、製品が正常にインストールされたことを確認するためにも実行できます。

始める前に

これらのシナリオの使用を開始する前に、以下の前提条件が満たされていることを確認してください。

- IBM MQ のキュー・マネージャー、キュー、およびチャネルの定義を理解している。
- IBM MQ クライアントおよびサーバーを既にインストール済み。
- MQIPT は、Windows システム上の `C:\mqipt` というディレクトリーにインストールされています。 (例は Windows システム用に書かれているが、サポートされるどのプラットフォームでも実行可能。) MQIPT のインストールについて詳しくは、MQIPT のインストールを参照してください。
- MQIPT のクライアント、サーバー、および各インスタンスが個別のコンピューターにインストールされている。
- **amqsputc** コマンドを使用したキューへのメッセージの書き込みを理解している。
- **amqsgetc** コマンドを使用したキューからのメッセージの読み取りを理解している。
- IBM MQ でのクライアント権限の設定を理解している。

このタスクについて

前提条件を満たしたら、以下の初期ステップを実行して、シナリオを実行できるようにシステムをセットアップします。

手順

1. IBM MQ サーバーで以下の作業を実行します。
   - キュー・マネージャー `MQIPT.QM1` を定義する。
   - サーバー接続チャネル `MQIPT.CONN.CHANNEL` を定義する。
   - ローカル・キュー `MQIPT.LOCAL.QUEUE` を定義する。
   - ポート 1414 で `MQIPT.QM1` の TCP/IP リスナーを開始する。 ポート 1414 が既に別のアプリケーションにより使用されている場合、空いているポート・アドレスを選択して、以下の例では置換してください。
   - クライアント・マシンからユーザー ID を使用してクライアント接続できるように接続認証とチャネル認証が構成されていることを確認します。 接続認証が、アプリケーションがクライアント接続の認証資格情報を提供することを要求するように設定されている場合は、**amqsputc** および **amqsgetc** コマンドを実行する前に、以下のいずれかの環境変数を設定する必要があります。

     MQSAP ユーザー ID
     キュー・マネージャーでの認証にユーザー ID とパスワードを使用する場合は、接続認証に使用するユーザー ID に設定します。

     V 9.3.4 Linux AIX **MQSAP トークン**
     キュー・マネージャーでの認証に認証トークンを使用する場合は、非ブランク値に設定します。
2. **amqsputc** コマンドを使用してキュー・マネージャーのローカル・キューにメッセージを書き込み、**amqsgetc** コマンドを使用して取得することにより、IBM MQ クライアントからキュー・マネージャーへの経路をテストします。

   このセクションのシナリオにを作成するには、次のように `mqipt.conf` ファイルを作成して編集します。

a. MQIPT インストール・ディレクトリーの samples サブディレクトリーにある mqiptSample.conf ファイルを、選択した MQIPT ホーム・ディレクトリー内の mqipt.conf にコピーします。 次のシナリオでは、C:\mqiptHome を MQIPT のホーム・ディレクトリーとして使用しています。

b. mqipt.conf と並んで、errors および logs という二つのディレクトリーを作成します。 これらのディレクトリーのファイル許可を設定して、MQIPT を実行するユーザー ID が書き込み可能になるようにします。

c. mqipt.conf ファイルからすべてのルートを削除します。

d. 残りの [global] セクションで、**ClientAccess** が存在し、true に設定されていることを確認します。

次のタスク

システムをセットアップした後、以下のシナリオを開始する準備ができました。

- 168 ページの『MQIPT が正常に機能しているかどうかの検証』
- 170 ページの『鍵リング・ファイルの作成』
- 172 ページの『テスト証明書の作成』
- 174 ページの『TLS サーバーの認証』
- 176 ページの『TLS クライアントの認証』
- 182 ページの『HTTP トンネリングの構成』
- 184 ページの『アクセス制御の構成』
- 186 ページの『SOCKS プロキシーの構成』
- 188 ページの『SOCKS クライアントの構成』
- 189 ページの『MQIPT クラスタリング・サポートの構成』
- 192 ページの『ポート番号の割り振り』
- 193 ページの『LDAP サーバーを使用した CRL の取得』
- 196 ページの『MQIPT の TLS プロキシー・モードでの実行』
- 198 ページの『セキュリティー・マネージャーを使用した TLS プロキシー・モードでの MQIPT の実行』
- 200 ページの『セキュリティー出口の使用』
- 202 ページの『セキュリティー出口を使用した IBM MQ キュー・マネージャー・サーバーへのクライアント接続要求の経路指定』
- 205 ページの『クライアント接続要求の動的経路指定』
- 208 ページの『TLS サーバーを認証するための証明書出口の使用』

MQIPT が正常に機能しているかどうかの検証

この単純な構成セットアップを使用して、MQIPT が正しくインストールされているかどうかを確認します。

始める前に

- このシナリオを開始する前に、167 ページの『IBM MQ Internet Pass-Thru 入門』 でリストしている前提条件作業を必ず完了してください。

このタスクについて

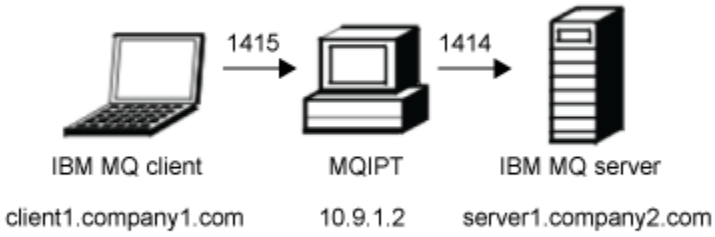


図 *30.* インストール検証テストのネットワーク図

この図では、IBM MQ クライアント (ポート 1415 の client1.company1.com) から MQIPT を通った IBM MQ サーバー (ポート 1414 の server1.company2.com) への接続を示しています。

手順

MQIPT が正常に機能していることを検証するには、以下の手順を実行します。

1. MQIPT 経路を定義します。

   MQIPT コンピューターで、 `mqipt.conf` を編集し、ルート定義を追加します。

   ```
   [route]
   ListenerPort=1415
   Destination=server1.company2.com
   DestinationPort=1414
   ```

2. MQIPT を開始します。

   コマンド・プロンプトを開いて、以下のコマンドを入力します。

   ```
   C:\mqipt\bin\mqipt C:\mqiptHome -n ipt1
   ```

   ここで、`C:\mqiptHome` は MQIPT 構成ファイル、`mqipt.conf` の場所を示し、および `ipt1` は MQIPT のインスタンスに指定される名前です。

   以下のメッセージは、MQIPT が正常に開始したことを示します。

   ```
   5724-H72 (C) Copyright IBM Corp. 2000, 2024. All Rights Reserved
   MQCPI001 IBM MQ Internet Pass-Thru V9.2.0.0 starting
   MQCPI004 Reading configuration information from mqipt.conf
   MQCPI152 MQIPT name is ipt1
   MQCPI021 Password checking has been enabled on the command port
   MQCPI011 The path C:\mqiptHome\logs will be used to store the log files
   MQCPI006 Route 1415 has started and will forward messages to:
   MQCPI034 ....server1.company2.com(1414)
   MQCPI035 ....using MQ protocol
   MQCPI078 Route 1415 ready for connection requests
   ```

3. IBM MQ クライアント・システムのコマンド・プロンプトで、以下のコマンドを入力します。

   a) **MQSERVER** 環境変数を設定します。

   ```
   SET MQSERVER=MQIPT.CONN.CHANNEL/tcp/10.9.1.2(1415)
   ```

   b) メッセージを書き込みます。

   ```
   amqsputc MQIPT.LOCAL.QUEUE MQIPT.QM1
   Hello world
   ```

   メッセージのストリングを入力した後、Enter キーを 2 回押します。

   c) メッセージを読み取ります。

```
amqsgetc MQIPT.LOCAL.QUEUE MQIPT.QM1
```

メッセージ「Hello world」が戻ります。

4. IBM MQ を停止するには、以下のコマンドを入力します。

```
mqiptAdmin -stop -n ipt1
```

鍵リング・ファイルの作成

このシナリオでは、証明書を要求し、鍵リング・ファイルを作成して、 MQIPT を使用して TLS を使用できるようにすることができます。

始める前に

このシナリオを開始する前に、167 ページの『IBM MQ Internet Pass-Thru 入門』でリストしている前提条件作業を必ず完了してください。

このタスクでは、 **mqiptKeyman** (iKeyman) を使用してトラステッド認証局 (CA) から新しい証明書を要求し、個人証明書がファイル (例えば、 `server.cer`) に戻されることを想定しています。 サーバー認証を実行するには、これで十分です。 クライアント認証を必要とする場合は、再認証(例えば、 `client.cer`) が要求され、以下のステップを 2 回実行して 2 つの鍵リング・ファイルを作成する必要があります。

このタスクについて

証明書を要求するために **mqiptKeycmd** コマンド・ラインインターフェース (CLI) または **mqiptKeyman** GUI を使用できます。 次に、証明書がインバウンド接続またはアウトバウンド接続のどちらで使用されるかに応じて、 **SSLServerKeyRing** または **SSLClientKeyRing** MQIPT 経路プロパティーを使用する必要があります。

手順

以下のいずれかの方法を使用して、鍵リング・ファイルを作成します。

- **mqiptKeycmd** コマンド・ラインインターフェース (CLI) の使用

  a) 次のコマンドを入力して、新しい PKCS #12 鍵リング・ファイルを作成します。

  ```
  mqiptKeycmd -keydb -create -db server_name.pfx -pw password -type pkcs12
  ```

  ここで、

  - **-db** は、鍵リング・ファイル (*server_name*.pfx) の名前を指定します。
  - **-pw** は、鍵リングのパスワード (*password*) を指定します。これは、後で **mqiptPW** コマンドを使用して暗号化する必要があります。

  b) 次のコマンドを入力して、新しい証明書要求を作成します。

  ```
  mqiptKeycmd -certreq -create -db server_name.pfx -pw password -type pkcs12
              -file cert_file_name.req -label label -dn DN_identity
              -sig_alg signature_algorithm -size key_size
  ```

  ここで、

  - **-file** は、要求した証明書のファイル名を指定します。
  - **-label** は、任意の固有の名前を指定しますが、スペース文字が含まれないことが望まれます。
  - **-dn** は、MQIPT 経路の適切な識別名 ID (例えば、`"CN=Test Certificate,OU=Sales,O=Example,C=US"`) を指定します。
  - **-sig_alg** は、ハッシュ・アルゴリズム (例えば、SHA256WithRSA) を指定します。
  - **-size** は、公開鍵のサイズ (例えば、2048) を指定します。

例に挙げた値を使用した場合、このコマンドによって、2048 ビット RSA 公開鍵のデジタル証明書と、SHA-256 ハッシュ・アルゴリズムの RSA を使用するデジタル署名が作成されます。

証明書を作成する場合は、組織のセキュリティー・ニーズを満たす適切な公開鍵暗号化アルゴリズム、鍵サイズ、およびデジタル署名アルゴリズムを選択するように注意してください。 詳しくは、MQIPT のデジタル証明書に関する考慮事項を参照してください。

コマンドによって作成された証明書要求ファイル (*cert_file_name*.req) を CA に送信し、署名を要求します。

c) 署名付き個人証明書を CA から受け取ったら、次のコマンドを入力して、証明書をサーバー鍵リングに追加します。

```
mqiptKeycmd -cert -receive -db server_name.pfx -pw password
            -type pkcs12 -file cert_file_name.crt
```

- **mqiptKeyman** GUI を使用します

a) 以下のコマンドを実行して、GUI を開きます。

```
mqiptKeyman
```

b)「**鍵データベース・ファイル**」 > 「**新規**」をクリックします。

c) 鍵データベースのタイプ (PKCS12) を選択します。

d) 新規鍵リング・ファイルのファイル名と場所を入力します。
「**OK**」をクリックします。

e) 新規鍵リング・ファイルのパスワードを入力し、確認します。
これは、後で **mqiptPW** コマンドを使用して暗号化する必要のある鍵リング・パスワードです。
「**OK**」をクリックして、新規個人証明書鍵リング・ファイルを作成します。

f)「**作成**」 > 「**新規認証要求**」をクリックして、認証要求を作成します。

g)「**鍵ラベル**」フィールドに新規証明書のラベルを入力します。
ラベルは、任意の固有の名前を指定できますが、スペース文字が含まれないことが望まれます。

h) 組織のセキュリティー・ニーズを満たす適切な鍵サイズおよびデジタル署名アルゴリズムを選択します。
詳しくは、 MQIPT のデジタル証明書に関する考慮事項を参照してください。

i) オプションの DN フィールドに、MQIPT 経路の適切な識別名 ID を入力します。

j) 作成する認証要求のファイル名を入力し、「**OK**」をクリックします。
認証要求が生成され、指定した名前で保存されます。 このファイルを CA に送信して署名をもらいます。

k) 署名付き個人証明書を CA から受け取る場合、それを鍵リング・ファイルで受信する必要があります。
「**鍵データベースの内容**」パネルで、ドロップダウン・リストから「個人証明書」を選択します。
「**受信**」をクリックします。

l) 署名付き証明書を保管するファイルの名前を入力し、「**OK**」をクリックします。

次のタスク

個人証明書に署名した CA の CA 証明書が CA 鍵リング・ファイルに存在することを確認する必要があります。 MQIPT 構成によっては、CA 鍵リング・ファイルは個人証明書鍵リング・ファイルとは異なるファイルにある場合があります。

分離したの CA 鍵リング・ファイルを使用するには、 MQIPT で提供される `sslCAdefault.pfx` というサンプル CA 鍵リング・ファイルを使用するか、新規の PKCS #12 鍵リング・ファイルを作成します。 個人証明書に署名した CA の公開 CA 証明書を CA 鍵リングに追加する必要があります (サンプルの鍵リング・ファイルに既に存在する場合を除く)。 公開 CA 証明書は、個人証明書と共に返されている場合があります。返されていない場合は、個人証明書を提供した CA の CA 証明書を要求して、それを鍵リングに追加する必要があります。

CA 証明書を追加する場合は、**mqiptKeycmd** CLI または **mqiptKeyman** GUI を使用できます。

mqiptKeycmd CLI を使用して CA 証明書を追加するには、以下のコマンドを実行します。

```
mqiptKeycmd -cert -add -db sslCAdefault.pfx -pw password -type pkcs12
            -file ca_file_name.crt -label label
```

ここで、

- **-db** は、CA 鍵リング・ファイル名 (この場合は、sslCAdefault.pfx) を指定します。
- **-pw** は、鍵リングのパスワードを指定します。 sslCAdefault.pfx という名前のサンプル CA 鍵リング・ファイルのパスワードは、mqiptSample です。
- **-file** は、CA によって返されるファイルの名前を指定します。
- **-label** は、任意の固有の名前を指定しますが、スペース文字を使用しないことが望まれます。

iKeyman GUI を使用して CA 証明書を追加するには、以下のようにします。

- 「鍵データベースの内容」パネルで、ドロップダウン・リストから「署名者証明書」を選択します。
- **追加** をクリックします。
- CA 証明書を含むファイルの名前を入力し、「**OK**」をクリックします。
- CA 証明書のラベルを入力します。 ラベルは、任意の固有の名前を指定できますが、スペース文字を使用しないことが望まれます。 「**OK**」をクリックします。

次のコマンドを発行して、鍵リング・パスワードを暗号化します。

```
mqiptPW
```

プロンプトが表示されたら、暗号化する鍵リング・パスワードを入力します。 **mqipt.conf** 構成ファイル内の該当するプロパティーの値を、**mqiptPW** コマンドで出力される暗号化されたパスワードに設定します。例えば、証明書がインバウンド接続で使用されるものかアウトバウンド接続で使用されるものかに応じて、**SSLServerKeyRingPW** または **SSLClientKeyRingPW** で指定されている必要があります。 鍵リング・パスワードの暗号化の詳細について、 保管されるパスワードの暗号化を参照し、鍵リング・パスワードを暗号化します。

これらの新規鍵リング・ファイルをサーバー認証に使用するには、次のように MQIPT ホーム・ディレクトリの下の ssl というディレクトリに鍵リング・ファイルを置き、以下のルート・プロパティを設定します:

```
SSLClientCAKeyRing=C:\\mqiptHome\\ssl\\sslCAdefault.pfx
SSLClientCAKeyRingPW=encrypted_password
SSLServerKeyRing=C:\\mqiptHome\\ssl\\myServer.pfx
SSLServerKeyRingPW=encrypted_password
SSLServerCAKeyRing=C:\\mqiptHome\\ssl\\sslCAdefault.pfx
SSLServerCAKeyRingPW=encrypted_password
```

TLS を使用するように MQIPT を構成する方法について詳しくは、シナリオ 174 ページの『TLS サーバーの認証』 を参照してください。

テスト証明書の作成

このシナリオでは、MQIPT 経路のテストに使用できる自己署名証明書を作成できます。 この証明書は、リモート・ピアであることを識別するために MQIPT 経路で使用できます。

自己署名証明書は、証明書のために認証局 (CA) に費用を支払わずに TLS 接続を確実にする必要があるテスト・シナリオで便利です。 ただし、実稼働環境では、自己署名証明書は使用しないでください。 実動で使用する証明書が必要な場合は、170 ページの『鍵リング・ファイルの作成』 を参照してください。

始める前に

- このシナリオを開始する前に、167 ページの『IBM MQ Internet Pass-Thru 入門』 でリストしている前提条件作業を必ず完了してください。

このタスクについて

証明書を要求するために **mqiptkeycmd** (iKeyman) コマンド行インターフェース (CLI) または **mqiptkeyman** GUI を使用できます。 証明書がインバウンド接続とアウトバウンド接続のどちらで使用するものであるかに応じて、**SSLServerKeyRing** または **SSLClientKeyRing** MQIPT 経路プロパティー内の、証明書が含まれる鍵リング・ファイルを指定する必要があります。

手順

以下のいずれかの方法を使用して、テスト証明書を作成します。

- コマンド行インターフェース (CLI) の使用
  a) 次のコマンドを入力して、新しい PKCS #12 鍵リング・ファイルを作成します。

  ```
  mqiptKeycmd -keydb -create -db server_name.pfx -pw password -type pkcs12
  ```

  ここで、

  - **-db** は、鍵リング・ファイル (*server_name*.pfx) の名前を指定します。
  - **-pw** は、鍵リングのパスワード (*password*) を指定します。これは、後で **mqiptPW** ユーティリティーを使用して暗号化する必要があります。

  b) テスト目的の自己署名個人証明書を作成するには、次のコマンドを入力します。

  ```
  mqiptKeycmd -cert -create -db server_name.pfx -pw password -type pkcs12
              -label label -dn DN_identity
              -sig_alg signature_algorithm -size key_size
  ```

  ここで、

  - **-label** は、任意の固有の名前を指定しますが、スペース文字が含まれないことが望まれます。
  - **-dn** は、MQIPT 経路の適切な識別名 ID (例えば、"CN=Test Certificate,OU=Sales,O=Example,C=US") を指定します。
  - **-sig_alg** は、ハッシュ・アルゴリズム (例えば、SHA256WithRSA) を指定します。
  - **-size** は、公開鍵のサイズ (例えば、2048) を指定します。

  例に挙げた値を使用した場合、このコマンドによって、2048 ビット RSA 公開鍵のデジタル証明書と、SHA-256 ハッシュ・アルゴリズムの RSA を使用するデジタル署名が作成されます。

  証明書を作成する場合は、組織のセキュリティー・ニーズを満たす適切な公開鍵暗号化アルゴリズム、鍵サイズ、およびデジタル署名アルゴリズムを選択するように注意してください。 詳しくは、MQIPT のデジタル証明書に関する考慮事項を参照してください。

- GUI の使用
  a) 以下のコマンドを実行して、GUI を開きます。

  ```
  mqiptKeyman
  ```

  b)「**鍵データベース・ファイル**」 >「**新規**」をクリックします。
  c) 鍵データベースのタイプ (例えば、PKCS12) を選択します。
  d) 新規鍵リング・ファイルのファイル名と場所を入力します。
  「**OK**」をクリックします。
  e) 新規鍵リング・ファイルのパスワードを入力します。
  確認のためにもう一度パスワードを入力します。 これは、 後で **mqiptPW** ユーティリティーを使用して暗号化する必要のある鍵リング・パスワードです。 「**OK**」をクリックして、新規個人証明書鍵リング・ファイルを作成します。
  f)「**作成**」 >「**新規自己署名証明書**」をクリックして、新規自己署名個人証明書を作成します。
  g)「**鍵ラベル**」フィールドに新規証明書のラベルを入力します。
  ラベルは、任意の固有の名前を指定できますが、スペース文字が含まれないことが望まれます。

h) 組織のセキュリティー・ニーズを満たす適切な鍵サイズおよびデジタル署名アルゴリズムを選択します。

詳しくは、 MQIPT のデジタル証明書に関する考慮事項を参照してください。

i) オプションの DN フィールドに、MQIPT 経路の適切な識別名 ID を入力して、「**OK**」をクリックします。

次のタスク

次のコマンドを発行して、鍵リング・パスワードを暗号化します。

```
mqiptPW
```

プロンプトが表示されたら、暗号化する鍵リング・パスワードを入力します。 **mqipt.conf** 構成ファイル内の該当するプロパティーの値を、**mqiptPW** コマンドで出力される暗号化されたパスワードに設定します。例えば、証明書がインバウンド接続で使用されるものかアウトバウンド接続で使用されるものかに応じて、**SSLServerKeyRingPW** または **SSLClientKeyRingPW** で指定されている必要があります。 鍵リング・パスワードの暗号化の詳細について、 保管されるパスワードの暗号化を参照し、鍵リング・パスワードを暗号化します。

TLS サーバーの認証

このシナリオでは、MQIPT インストール済みのディレクトリーの samples/ssl サブディレクトリーにある MQIPT に付属するサンプル (sslSample.pfx) 鍵リング・ファイル内のテスト証明書を使用して、 TLS 接続をテストできます。

始める前に

このシナリオの使用を開始する前に、167 ページの『IBM MQ Internet Pass-Thru 入門』にリストされている前提条件タスクを完了していること、およびトピック「 MQIPT での SSL/TLS サポート」をお読みください。

このタスクについて

接続は、MQIPT の 2 つのインスタンスを介して IBM MQ クライアントと IBM MQ サーバーの間で行われます。 MQIPT 1 と MQIPT 2 の間の接続では TLS が使用され、MQIPT 1 は TLS クライアントの働きをし、MQIPT 2 は TLS サーバーの働きをします。

TLS ハンドシェークの間に、サーバーはテスト証明書をクライアントに送信し、クライアントはその証明書のコピーを使用してピアとして信頼するフラグを設定してサーバーを認証します。 CipherSuite SSL_RSA_WITH_AES_256_CBC_SHA256 が使用されます。 このシナリオの mqipt.conf という構成ファイルは、168 ページの『MQIPT が正常に機能しているかどうかの検証』のシナリオで作成された構成ファイルに基づいたものです。 この例で使用するテスト証明書の作成方法の詳細については、172 ページの『テスト証明書の作成』を参照してください。

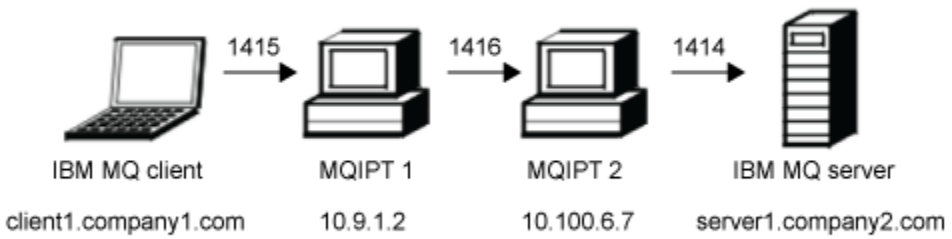


図 *31. SSL/TLS* サーバーのネットワーク図

この図では、IBM MQ クライアント (ポート 1415 の client1.company1.com) から MQIPT の 2 つのインスタンスを通った IBM MQ サーバー (ポート 1414 の server1.company2.com) への接続を示しています。

手順

TLS サーバーを認証するには、以下の手順を実行します。

1. MQIPT 1 システムの場合:

   a) `mqipt.conf` を編集して、次のルート定義を追加してください。

   ```
   [route]
   ListenerPort=1415
   Destination=10.100.6.7
   DestinationPort=1416
   SSLClient=true
   SSLClientKeyRing=C:\\mqipt\\samples\\ssl\\sslSample.pfx
   SSLClientKeyRingPW=<mqiptPW>1!PCaB1HWrFMOp43ngjwgArg==!6N/vsbqru7iqMhFN+wozxQ==
   SSLClientCipherSuites=SSL_RSA_WITH_AES_256_CBC_SHA256
   ```

   b) コマンド・プロンプトを開き、MQIPT を開始します。

   ```
   C:\mqipt\bin\mqipt C:\mqiptHome -n ipt1
   ```

   ここで、`C:\mqiptHome` は MQIPT 構成ファイル、`mqipt.conf` の場所を示し、および `ipt1` は MQIPT のインスタンスに指定される名前です。

   以下のメッセージは、MQIPT が正常に開始したことを示します。

   ```
   5724-H72 (C) Copyright IBM Corp. 2000, 2024. All Rights Reserved
   MQCPI001 IBM MQ Internet Pass-Thru V9.2.0.0 starting
   MQCPI004 Reading configuration information from mqipt.conf
   MQCPI152 MQIPT name is ipt1
   MQCPI021 Password checking has been enabled on the command port
   MQCPI011 The path C:\mqiptHome\logs will be used to store the log files
   MQCPI006 Route 1415 is starting and will forward messages to :
   MQCPI034 ....10.100.6.7(1416)
   MQCPI035 ....using MQ protocol
   MQCPI036 ....SSL Client side enabled with properties :
   MQCPI139 ......secure socket protocols <NULL>
   MQCPI031 ......cipher suites SSL_RSA_WITH_AES_256_CBC_SHA256
   MQCPI032 ......key ring file C:\\mqipt\\samples\\ssl\\sslSample.pfx
   MQCPI047 ......CA key ring file <NULL>
   MQCPI071 ......site certificate uses
   UID=*,CN=*,T=*,OU=*,DC=*,O=*,STREET=*,L=*,ST=*,PC=*,C=*,DNQ=*
   MQCPI038 ......peer certificate uses
   UID=*,CN=*,T=*,OU=*,DC=*,O=*,STREET=*,L=*,ST=*,PC=*,C=*,DNQ=*
   MQCPI078 Route 1415 ready for connection requests
   ```

2. MQIPT 2 システムの場合:

   a) `mqipt.conf` を編集して、次のルート定義を追加してください。

   ```
   [route]
   ListenerPort=1416
   Destination=Server1.company2.com
   DestinationPort=1414
   SSLServer=true
   SSLServerKeyRing=C:\\mqipt\\samples\\ssl\\sslSample.pfx
   SSLServerKeyRingPW=<mqiptPW>1!PCaB1HWrFMOp43ngjwgArg==!6N/vsbqru7iqMhFN+wozxQ==
   SSLServerCipherSuites=SSL_RSA_WITH_AES_256_CBC_SHA256
   ```

   b) コマンド・プロンプトを開き、MQIPT を開始します。

   ```
   C:
   cd \mqipt\bin
   mqipt .. -n ipt2
   ```

   ここで、`..` は MQIPT 構成ファイル `mqipt.conf` が親ディレクトリーにあることを示し、`ipt2` は MQIPT のインスタンスに付けられる名前です。

以下のメッセージは、MQIPT が正常に開始したことを示します。

```
5724-H72 (C) Copyright IBM Corp. 2000, 2024. All Rights Reserved
MQCPI001 IBM MQ Internet Pass-Thru V9.2.0.0 starting
MQCPI004 Reading configuration information from mqipt.conf
MQCPI152 MQIPT name is ipt2
MQCPI021 Password checking has been enabled on the command port
MQCPI011 The path C:\mqipt\logs will be used to store the log files
MQCPI006 Route 1416 is starting and will forward messages to :
MQCPI034 ....Server1.company2.com(1414)
MQCPI035 ....using MQ protocol
MQCPI037 ....SSL Server side enabled with properties :
MQCPI139 ......secure socket protocols <NULL>
MQCPI031 ......cipher suites SSL_RSA_WITH_AES_256_CBC_SHA256
MQCPI032 ......key ring file C:\\mqipt\\samples\\ssl\\sslSample.pfx
MQCPI047 ......CA key ring file <NULL>
MQCPI071 ......site certificate uses
UID=*,CN=*,T=*,OU=*,DC=*,O=*,STREET=*,L=*,ST=*,PC=*,C=*,DNQ=*
MQCPI038 ......peer certificate uses
UID=*,CN=*,T=*,OU=*,DC=*,O=*,STREET=*,L=*,ST=*,PC=*,C=*,DNQ=*
MQCPI033 ......client authentication set to false
MQCPI078 Route 1416 ready for connection requests
```

3. IBM MQ クライアントのコマンド・プロンプトで以下のコマンドを入力します。
   a) **MQSERVER** 環境変数を設定します。

```
SET MQSERVER=MQIPT.CONN.CHANNEL/tcp/10.9.1.2(1415)
```

   b) メッセージを書き込みます。

```
amqsputc MQIPT.LOCAL.QUEUE MQIPT.QM1
Hello world
```

   メッセージのストリングを入力した後、Enter キーを 2 回押します。
   c) メッセージを読み取ります。

```
amqsgetc MQIPT.LOCAL.QUEUE MQIPT.QM1
```

   メッセージ「Hello world」が戻ります。

TLS クライアントの認証

このシナリオでは、サンプルのテスト証明書を使用してサーバーとクライアントの認証を実行し、TLS 接続をテストできます。

始める前に

このシナリオの使用を開始する前に、167 ページの『IBM MQ Internet Pass-Thru 入門』にリストされている前提条件タスクを完了していること、およびトピック「MQIPT での SSL/TLS サポート」をお読みください。

このタスクについて

接続は、MQIPT の 2 つのインスタンスを介して IBM MQ クライアントと IBM MQ サーバーの間で行われます。 MQIPT 1 と MQIPT 2 の間の接続では TLS が使用され、MQIPT 1 は TLS クライアントの働きをし、MQIPT 2 は TLS サーバーの働きをします。

TLS ハンドシェークの間に、サーバーはテスト証明書をクライアントに送信します。 クライアントはその証明書のコピーを使用してピアとして信頼するフラグでサーバーを認証します。 その後、クライアントはテスト証明書をサーバーに送信します。 サーバーはその証明書のコピーを使用してピアとして信頼するフラグでクライアントを認証します。 CipherSuite SSL_RSA_WITH_AES_256_CBC_SHA256 が使用されます。 このシナリオの `mqipt.conf` という構成ファイルは、168 ページの『MQIPT が正常に機能しているかどうかの検証』のシナリオで作成された構成ファイルに基づいたものです。

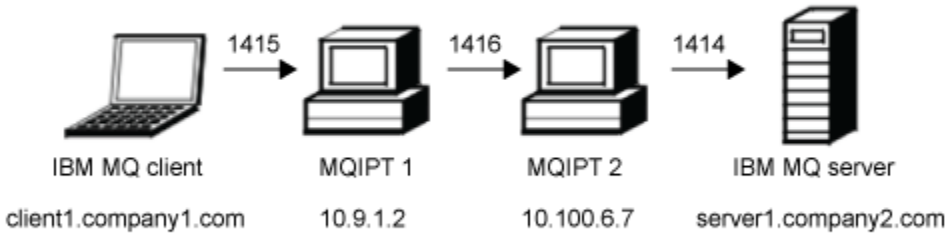


図 *32. SSL/TLS* クライアントのネットワーク図

この図では、IBM MQ クライアント (ポート 1415 の client1.company1.com) から MQIPT の 2 つのインスタンスを通った IBM MQ サーバー (ポート 1414 の server1.company2.com) への接続を示しています。

手順

TLS クライアントを認証するには、以下の手順を実行します。

1. MQIPT 1 システムの場合:

   a) `mqipt.conf` を編集して、次のルート定義を追加してください。

   ```
   [route]
   ListenerPort=1415
   Destination=10.100.6.7
   DestinationPort=1416
   SSLClient=true
   SSLClientKeyRing=C:\\mqipt\\samples\\ssl\\sslSample.pfx
   SSLClientKeyRingPW=<mqiptPW>1!PCaB1HWrFMOp43ngjwgArg==!6N/vsbqru7iqMhFN+wozxQ==
   SSLClientCipherSuites=SSL_RSA_WITH_AES_256_CBC_SHA256
   ```

   b) コマンド・プロンプトを開き、MQIPT を開始します。

   ```
   C:\mqipt\bin\mqipt C:\mqiptHome -n ipt1
   ```

   ここで、`C:\mqiptHome` は MQIPT 構成ファイル、`mqipt.conf` の場所を示し、および `ipt1` は MQIPT のインスタンスに指定される名前です。

   以下のメッセージは、MQIPT が正常に開始したことを示します。

   ```
   5724-H72 (C) Copyright IBM Corp. 2000, 2024. All Rights Reserved
   MQCPI001 IBM MQ Internet Pass-Thru V9.2.0.0 starting
   MQCPI004 Reading configuration information from mqipt.conf
   MQCPI152 MQIPT name is ipt1
   MQCPI021 Password checking has been enabled on the command port
   MQCPI011 The path C:\mqiptHome\logs will be used to store the log files
   MQCPI006 Route 1415 is starting and will forward messages to :
   MQCPI034 ....10.100.6.7(1416)
   MQCPI035 ....using MQ protocol
   MQCPI036 ....SSL Client side enabled with properties :
   MQCPI139 ......secure socket protocols <NULL>
   MQCPI031 ......cipher suites SSL_RSA_WITH_AES_256_CBC_SHA256
   MQCPI032 ......key ring file C:\\mqipt\\samples\\ssl\\sslSample.pfx
   MQCPI047 ......CA key ring file <NULL>
   MQCPI071 ......site certificate uses
   UID=*,CN=*,T=*,OU=*,DC=*,O=*,STREET=*,L=*,ST=*,PC=*,C=*,DNQ=*
   MQCPI038 ......peer certificate uses
   UID=*,CN=*,T=*,OU=*,DC=*,O=*,STREET=*,L=*,ST=*,PC=*,C=*,DNQ=*
   MQCPI078 Route 1415 ready for connection requests
   ```

2. MQIPT 2 システムの場合:

   a) `mqipt.conf` を編集して、次のルート定義を追加してください。

   ```
   [route]
   ListenerPort=1416
   ```

```
Destination=Server1.company2.com
DestinationPort=1414
SSLServer=true
SSLServerAskClientAuth=true
SSLServerKeyRing=C:\\mqipt\\samples\\ssl\\sslSample.pfx
SSLServerKeyRingPW=<mqiptPW>1!PCaB1HWrFMOp43ngjwgArg==!6N/vsbqru7iqMhFN+wozxQ==
SSLServerCipherSuites=SSL_RSA_WITH_AES_256_CBC_SHA256
```

b) コマンド・プロンプトを開き、MQIPT を開始します。

```
C:
cd \mqipt\bin
mqipt .. -n ipt2
```

ここで、`..` は MQIPT 構成ファイル `mqipt.conf` が親ディレクトリーにあることを示し、`ipt2` は MQIPT のインスタンスに付けられる名前です。

以下のメッセージは、MQIPT が正常に開始したことを示します。

```
5724-H72 (C) Copyright IBM Corp. 2000, 2024. All Rights Reserved
MQCPI001 IBM MQ Internet Pass-Thru V9.2.0.0 starting
MQCPI004 Reading configuration information from mqipt.conf
MQCPI152 MQIPT name is ipt2
MQCPI021 Password checking has been enabled on the command port
MQCPI011 The path C:\mqipt\logs will be used to store the log files
MQCPI006 Route 1416 is starting and will forward messages to :
MQCPI034 ....Server1.company2.com(1414)
MQCPI035 ....using MQ protocol
MQCPI037 ....SSL Server side enabled with properties :
MQCPI139 ......secure socket protocols <NULL>
MQCPI031 ......cipher suites SSL_RSA_WITH_AES_256_CBC_SHA256
MQCPI032 ......key ring file C:\\mqipt\\samples\\ssl\\sslSample.pfx
MQCPI047 ......CA key ring file <NULL>
MQCPI071 ......site certificate uses
UID=*,CN=*,T=*,OU=*,DC=*,O=*,STREET=*,L=*,ST=*,PC=*,C=*,DNQ=*
MQCPI038 ......peer certificate uses
UID=*,CN=*,T=*,OU=*,DC=*,O=*,STREET=*,L=*,ST=*,PC=*,C=*,DNQ=*
MQCPI033 ......client authentication set to true
MQCPI078 Route 1416 ready for connection requests
```

3. IBM MQ クライアント・システムのコマンド・プロンプトで、以下のコマンドを入力します。

a) **MQSERVER** 環境変数を設定します。

```
SET MQSERVER=MQIPT.CONN.CHANNEL/tcp/10.9.1.2(1415)
```

b) メッセージを書き込みます。

```
amqsputc MQIPT.LOCAL.QUEUE MQIPT.QM1
Hello world
```

メッセージのストリングを入力した後、Enter キーを 2 回押します。

c) メッセージを読み取ります。

```
amqsgetc MQIPT.LOCAL.QUEUE MQIPT.QM1
```

メッセージ「Hello world」が戻ります。

TLS クライアントおよびサーバーの認証

MQIPT を TLS サーバーとクライアントの両方として実行して、着信 TLS セッションを終了し、別個の TLS 接続を使用してデータを宛先に転送することができます。

始める前に

このシナリオの使用を開始する前に、167 ページの『IBM MQ Internet Pass-Thru 入門』にリストされている前提条件タスクを完了していること、およびトピック「MQIPT での SSL/TLS サポート」を読んでいることを確認してください。

注 : このシナリオでは、便宜上、自己署名証明書を使用します。 実稼働環境では自己署名証明書を使用しないでください。 代わりに、信頼できる認証局 (CA) によって署名された証明書を取得してください。

このタスクについて

接続は、 MQIPT の単一インスタンスを介して IBM MQ クライアントと IBM MQ サーバーの間で行われます。 IBM MQ クライアントと MQIPT の間、および MQIPT と IBM MQ サーバーの間の接続は、両方とも TLS を使用します。 したがって、 MQIPT 経路は TLS サーバーと TLS クライアントの両方です。

クライアントと MQIPT の間の TLS ハンドシェーク中に、クライアントと MQIPT は、接続を認証するために証明書を相互に送信します。 クライアントと MQIPT の間の接続が確立されると、 MQIPT は IBM MQ サーバーへの別個の TLS 接続を確立します。 MQIPT と IBM MQ サーバーは、接続を認証するために相互に証明書を送信します。

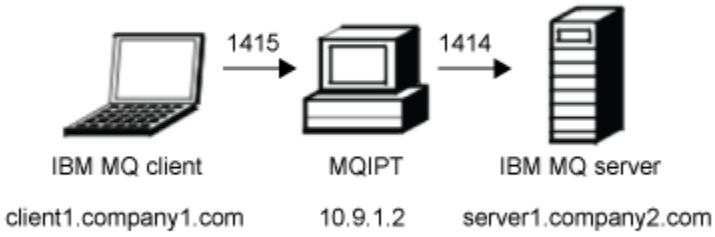


図 *33. SSL/TLS* サーバーとクライアントのネットワーク図

この図は、IBM MQ クライアント (client1.company1.com と呼ばれる) から MQIPT の単一インスタンスを介して IBM MQ サーバー (ポート 1414 では server1.company2.com と呼ばれる) への接続を示しています。

手順

MQIPT 経路を TLS サーバーと TLS クライアントの両方として構成するには、以下の手順を実行します。

1. クライアントが実行されているシステムで、以下のステップを実行して、TLS 接続を使用するように IBM MQ クライアントを構成します。
   a) クライアントのキー・リポジトリーを作成する。
   以下のコマンドを入力して、 `C:\ProgramData\IBM\MQ` ディレクトリーに `clientkey.kdb` という名前の新しい鍵リポジトリーを作成します。

   ```
   runmqakm -keydb -create -db C:\ProgramData\IBM\MQ\clientkey.kdb -pw password -stash
   ```

   ここで、 *password* は鍵リポジトリーのパスワードです。
   詳しくは、AIX, Linux, and Windows での鍵リポジトリーのセットアップを参照してください。
   b) ステップ 179 ページの『1.a』 で作成したクライアントの鍵リポジトリー内に、クライアントの個人証明書を作成します。
   以下のコマンドを入力して、クライアントのラベル `clientcert` を持つ新しい自己署名証明書を作成します。

   ```
   runmqakm -cert -create -db C:\ProgramData\IBM\MQ\clientkey.kdb -stashed
            -label clientcert -dn CN=Client
   ```

   詳しくは、AIX, Linux, and Windows での自己署名個人証明書の作成を参照してください。
   c) クライアント鍵リポジトリーから証明書を抽出します。
   以下のコマンドを入力して、クライアント証明書を抽出します。

```
runmqakm -cert -extract -db C:\ProgramData\IBM\MQ\clientkey.kdb -stashed -label clientcert
         -target C:\ProgramData\IBM\MQ\client.crt -format ascii
```

詳しくは、 AIX, Linux, and Windows 上の鍵リポジトリーからの自己署名証明書の公開部分の抽出を参照してください。

d) 抽出した証明書ファイルを MQIPT が実行されるシステムにコピーします。

2. IBM MQ サーバーが実行されているシステムで、以下のステップを実行して、TLS 接続を使用するようにキュー・マネージャーを構成します。

a) キュー・マネージャーのキー・リポジトリーを作成する。

以下のコマンドを入力して、キュー・マネージャー用の key.kdb という名前の新しい鍵リポジトリーを作成します。

```
runmqakm -keydb -create -db C:\ProgramData\IBM\MQ\qmgrs\MQIPT!QM1\ssl\key.kdb -pw
password -stash
```

ここで、 *password* は鍵リポジトリーのパスワードです。

暗号化された鍵リポジトリー・パスワードを保管するためにコマンドが作成する C:\ProgramData\IBM\MQ\qmgrs\MQIPT!QM1\ssl\key.sth stash ファイルに対する読み取り権限が mqm ユーザーに付与されていることを確認します。

詳しくは、AIX, Linux, and Windows での鍵リポジトリーのセットアップを参照してください。

b) ステップ 180 ページの『2.a』 で作成したキュー・マネージャーの鍵リポジトリー内に、キュー・マネージャーの個人証明書を作成します。

以下のコマンドを入力して、キュー・マネージャーの ibmwebspheremqmqipt.qm1 というラベルの新しい自己署名証明書を作成します。

```
runmqakm -cert -create -db C:\ProgramData\IBM\MQ\qmgrs\MQIPT!QM1\ssl\key.kdb -stashed
         -label ibmwebspheremqmqipt.qm1 -dn CN=MQIPT.QM1
```

詳しくは、AIX, Linux, and Windows での自己署名個人証明書の作成を参照してください。

c) キュー・マネージャーの鍵リポジトリーから証明書を抽出します。

以下のコマンドを入力して、キュー・マネージャー証明書を抽出します。

```
runmqakm -cert -extract -db C:\ProgramData\IBM\MQ\qmgrs\MQIPT!QM1\ssl\key.kdb -stashed
-label ibmwebspheremqmqipt.qm1
         -target C:\ProgramData\IBM\MQ\qmgrs\MQIPT!QM1\ssl\mqipt.qm1.crt -format ascii
```

詳しくは、 AIX, Linux, and Windows 上の鍵リポジトリーからの自己署名証明書の公開部分の抽出を参照してください。

d) 抽出した証明書ファイルを MQIPT が実行されるシステムにコピーします。

e) 以下の MQSC コマンドを発行して、 MQIPT.CONN.CHANNEL サーバー接続チャネル:

```
ALTER CHANNEL(MQIPT.CONN.CHANNEL) CHLTYPE(SVRCONN) TRPTYPE(TCP)
SSLCIPH(ANY_TLS12_OR_HIGHER)
```

3. MQIPT が実行されているシステムで、以下のステップを実行して、TLS を使用するように MQIPT 経路を構成します。

a) 以下のコマンドを入力して、 MQIPT 用の PKCS #12 鍵リポジトリーを作成します。

```
mqiptKeycmd -keydb -create -db C:\mqiptHome\ssl\mqipt.p12 -pw password -type pkcs12
```

ここで、 *password* は鍵リポジトリーのパスワードです。

b) ステップ 180 ページの『3.a』 で作成した MQIPT 鍵リポジトリー内に、 MQIPT 用の個人証明書を作成します。

以下のコマンドを入力して、 MQIPT のラベル mqiptcert を持つ新しい自己署名証明書を作成します。

```
mqiptKeycmd -cert -create -db C:\mqiptHome\ssl\mqipt.p12 -pw password -type pkcs12
            -label mqiptcert -dn "CN=MQIPT Test Certificate"
```

ここで、 *password* は、ステップ 180 ページの『3.a』 で鍵リポジトリーを作成したときに指定した鍵リポジトリー・パスワードです。

c) クライアント証明書とキュー・マネージャー証明書を MQIPT 鍵リポジトリーに追加します。

以下のコマンドを入力して、 MQIPT 鍵リポジトリーに証明書を追加します。

```
mqiptKeycmd -cert -add -db C:\mqiptHome\ssl\mqipt.p12 -pw password -type pkcs12 -label
clientcert
            -file client.crt -format ascii
mqiptKeycmd -cert -add -db C:\mqiptHome\ssl\mqipt.p12 -pw password -type pkcs12 -label
qm1cert
            -file mqipt.qm1.crt -format ascii
```

ここで、 *password* はキー・リポジトリー・パスワード、 *client.crt* はステップ 179 ページの『1.c』 で作成したクライアント証明書ファイル、 *mqipt.qm1.crt* はステップ 180 ページの『2.c』 で作成したキュー・マネージャー証明書です。

d) 鍵リポジトリーから MQIPT 証明書を抽出します。

以下のコマンドを入力して、 MQIPT 証明書を抽出します。

```
mqiptKeycmd -cert -extract -db C:\mqiptHome\ssl\mqipt.p12 -pw password -type pkcs12
            -label mqiptcert -target C:\mqiptHome\ssl\mqipt.crt -format ascii
```

ここで、 *password* は鍵リポジトリーのパスワードです。

e) 抽出した証明書ファイルを、クライアントが実行されるシステムと IBM MQ サーバーが実行されるシステムの両方にコピーします。

f) 以下のコマンドを入力して、 MQIPT 鍵リポジトリー・パスワードを暗号化します。

```
mqiptPW
```

プロンプトが出されたら、ステップ 180 ページの『3.a』 で鍵リポジトリーを作成したときに指定した鍵リポジトリー・パスワードを入力します。

g) mqipt.conf ファイルを編集して、以下の経路定義を追加します。

```
[route]
ListenerPort=1415
Destination=server1.company2.com
DestinationPort=1414
SSLServer=true
SSLServerKeyRing=C:\\mqiptHome\\ssl\\mqipt.p12
SSLServerKeyRingPW=encrypted_password
SSLClient=true
SSLClientKeyRing=C:\\mqiptHome\\ssl\\mqipt.p12
SSLClientKeyRingPW=encrypted_password
```

ここで、 *encrypted_password* は、ステップ 181 ページの『3.f』 で **mqiptPW** コマンドを実行して作成した暗号化された鍵リポジトリー・パスワードです。

4. MQIPT 証明書をクライアント鍵リポジトリーとキュー・マネージャー鍵リポジトリーの両方に追加します。

a) クライアントが実行されているシステムで、以下のコマンドを入力して、 MQIPT 証明書をクライアント鍵リポジトリーに追加します。

```
runmqakm -cert -add -db C:\ProgramData\IBM\MQ\clientkey.kdb -stashed
         -label mqiptcert -file mqipt.crt -format ascii
```

ここで、 *mqipt.crt* は、ステップ 181 ページの『3.d』 で作成した MQIPT 証明書ファイルです。

b) IBM MQ サーバーが実行されているシステムで、以下のコマンドを入力して、 MQIPT 証明書をキュー・マネージャーの鍵リポジトリーに追加します。

```
runmqakm -cert -add -db C:\ProgramData\IBM\MQ\qmgrs\MQIPT!QM1\ssl\key.kdb -stashed
         -label mqiptcert -file mqipt.crt -format ascii
```

ここで、 *mqipt.crt* は、ステップ 181 ページの『3.d』 で作成した MQIPT 証明書ファイルです。

詳しくは、 AIX, Linux, and Windows システムでの鍵リポジトリーへの CA 証明書 (または自己署名証明書の公開部分) の追加を参照してください。

5. MQIPT が実行されているシステムで、コマンド・プロンプトを開き、以下のコマンドを入力して MQIPT を開始します。

```
C:\mqipt\bin\mqipt C:\mqiptHome -n ipt1
```

ここで、`C:\mqiptHome` は MQIPT 構成ファイル、`mqipt.conf` の場所を示し、および `ipt1` は MQIPT のインスタンスに指定される名前です。

以下のメッセージは、MQIPT が正常に開始したことを示します。

```
5724-H72 (C) Copyright IBM Corp. 2000, 2024. All Rights Reserved
MQCPI001 IBM MQ Internet Pass-Thru V9.2.0.0 starting
MQCPI004 Reading configuration information from mqipt.conf
MQCPI152 MQIPT name is ipt1
MQCPI021 Password checking has been enabled on the command port
MQCPI011 The path C:\mqiptHome\logs will be used to store the log files
MQCPI006 Route 1415 is starting and will forward messages to :
MQCPI034 ....server1.company2.com(1414)
MQCPI035 ....using MQ protocol
MQCPI036 ....SSL Client side enabled with properties :
MQCPI139 ......secure socket protocols <NULL>
MQCPI031 ......cipher suites <NULL>
MQCPI032 ......key ring file C:\\mqiptHome\\ssl\\mqipt.p12
MQCPI047 ......CA key ring file <NULL>
MQCPI071 ......site certificate uses
UID=*,CN=*,T=*,OU=*,DC=*,O=*,STREET=*,L=*,ST=*,PC=*,C=*,DNQ=*
MQCPI038 ......peer certificate uses
UID=*,CN=*,T=*,OU=*,DC=*,O=*,STREET=*,L=*,ST=*,PC=*,C=*,DNQ=*
MQCPI037 ....SSL Server side enabled with properties :
MQCPI139 ......secure socket protocols <NULL>
MQCPI031 ......cipher suites <NULL>
MQCPI032 ......key ring file C:\\mqiptHome\\ssl\\mqipt.p12
MQCPI047 ......CA key ring file <NULL>
MQCPI071 ......site certificate uses
UID=*,CN=*,T=*,OU=*,DC=*,O=*,STREET=*,L=*,ST=*,PC=*,C=*,DNQ=*
MQCPI038 ......peer certificate uses
UID=*,CN=*,T=*,OU=*,DC=*,O=*,STREET=*,L=*,ST=*,PC=*,C=*,DNQ=*
MQCPI033 ......client authentication set to false
MQCPI078 Route 1415 ready for connection requests
```

6. IBM MQ クライアント・システムのコマンド・プロンプトで、以下のコマンドを入力して TLS サンプル・プログラムを実行します。

```
AMQSSSLC -m MQIPT.QM1 -c MQIPT.CONN.CHANNEL -x 10.9.1.2(1415)
         -k "C:\ProgramData\IBM\MQ\clientkey" -l clientcert -s ANY_TLS12_OR_HIGHER
```

以下のメッセージは、アプリケーションがキュー・マネージャーに正常に接続されたことを示します。

```
Connection established to queue manager MQIPT.QM1
```

HTTP トンネリングの構成

このシナリオでは、HTTP を介した MQIPT の 2 つのインスタンス間の単純接続をテストできます。

始める前に

このシナリオを開始する前に、167 ページの『IBM MQ Internet Pass-Thru 入門』 でリストしている前提条件作業を必ず完了してください。

このタスクについて

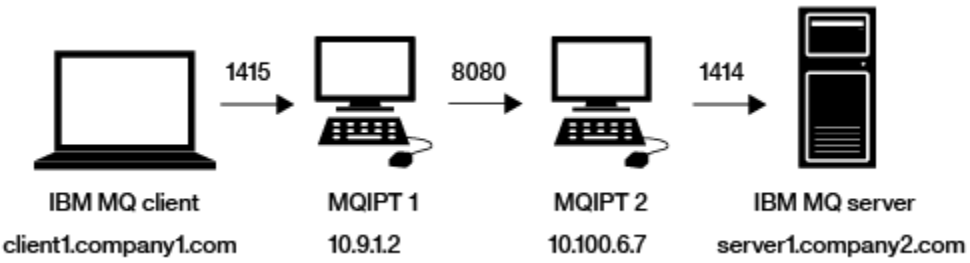


図 *34. HTTP* トンネリングネットワーク図

この図では、IBM MQ クライアント (ポート 1415 の client1.company1.com) から MQIPT の 2 つのインスタンス (HTTP を介したトンネル接続) を通って最終的に IBM MQ サーバー (ポート 1414 の server1.company2.com) に至る接続を示しています。

手順

MQIPT の 2 つのインスタンス間で HTTP トンネリングを構成するには、以下の手順を実行してください。

1. MQIPT 1 システムの場合:

   a) `mqipt.conf` を編集して、次のルート定義を追加してください。

   ```
   [route]
   ListenerPort=1415
   Destination=10.100.6.7
   DestinationPort=8080
   HTTP=true
   HTTPServer=10.100.6.7
   HTTPServerPort=8080
   ```

   b) コマンド・プロンプトを開き、MQIPT を開始します。

   ```
   C:\mqipt\bin\mqipt C:\mqiptHome -n ipt1
   ```

   ここで、`C:\mqiptHome` は MQIPT 構成ファイル、`mqipt.conf` の場所を示し、および `ipt1` は MQIPT のインスタンスに指定される名前です。

   以下のメッセージは、MQIPT が正常に開始したことを示します。

   ```
   5724-H72 (C) Copyright IBM Corp. 2000, 2024. All Rights Reserved
   MQCPI001 IBM MQ Internet Pass-Thru V9.2.0.0 starting
   MQCPI004 Reading configuration information from mqipt.conf
   MQCPI152 MQIPT name is ipt1
   MQCPI021 Password checking has been enabled on the command port
   MQCPI011 The path C:\mqiptHome\logs will be used to store the log files
   MQCPI006 Route 1415 is starting and will forward messages to :
   MQCPI034 ....10.100.6.7(8080)
   MQCPI035 ....using HTTP
   MQCPI066 ....and HTTP server at 10.100.6.7(8080)
   MQCPI078 Route 1415 ready for connection requests
   ```

2. MQIPT 2 システムの場合:

   a) `mqipt.conf` を編集して、次のルート定義を追加してください。

   ```
   [route]
   ListenerPort=8080
   Destination=Server1.company2.com
   DestinationPort=1414
   ```

   b) コマンド・プロンプトを開き、MQIPT を開始します。

   ```
   C:\mqipt\bin\mqipt C:\mqiptHome -n ipt2
   ```

ここで、C:\mqiptHome は MQIPT 構成ファイル、mqipt.conf の場所を示し、および ipt2 は MQIPT のインスタンスに指定される名前です。

以下のメッセージは、MQIPT が正常に開始したことを示します。

```
5724-H72 (C) Copyright IBM Corp. 2000, 2024. All Rights Reserved
MQCPI001 IBM MQ Internet Pass-Thru V9.2.0.0 starting
MQCPI004 Reading configuration information from mqipt.conf
MQCPI152 MQIPT name is ipt2
MQCPI021 Password checking has been enabled on the command port
MQCPI011 The path C:\mqiptHome\logs will be used to store the log files
MQCPI006 Route 8080 is starting and will forward messages to :
MQCPI034 ....Server1.company2.com(1414)
MQCPI035 ....using MQ protocols
MQCPI078 Route 8080 ready for connection requests
```

3. IBM MQ クライアントのコマンド・プロンプトで以下のコマンドを入力します。
   a) **MQSERVER** 環境変数を設定します。

   ```
   SET MQSERVER=MQIPT.CONN.CHANNEL/tcp/10.9.1.2(1415)
   ```

   b) メッセージを書き込みます。

   ```
   amqsputc MQIPT.LOCAL.QUEUE MQIPT.QM1
   Hello world
   ```

   メッセージのストリングを入力した後、Enter キーを 2 回押します。

   c) メッセージを読み取ります。

   ```
   amqsgetc MQIPT.LOCAL.QUEUE MQIPT.QM1
   ```

   メッセージ「Hello world」が戻ります。

アクセス制御の構成

このシナリオでは、 Java security manager を使用して MQIPT リスナー・ポートでセキュリティー検査を追加することにより、特定のクライアントからの接続のみを受け入れるように MQIPT をセットアップできます。

始める前に

- このシナリオを開始する前に、167 ページの『IBM MQ Internet Pass-Thru 入門』 でリストしている前提条件作業を必ず完了してください。

このタスクについて

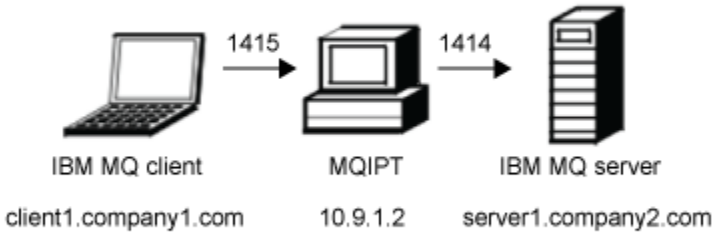


図 *35.* アクセス制御のネットワーク図

この図では、IBM MQ クライアント (ポート 1415 の client1.company1.com) から MQIPT を通った IBM MQ サーバー (ポート 1414 の server1.company2.com) への接続を示しています。

手順

アクセス制御を構成するには、以下の手順を実行します。

1. MQIPT をセットアップします。
   a) コマンド・プロンプトで次のコマンドを入力して、サンプルの Java security manager・ポリシーを MQIPT のホーム・ディレクトリーにコピーします。

   ```
   copy C:\mqipt\samples\mqiptSample.policy C:\mqiptHome\mqipt.policy
   ```

   b) 以下のコマンドを使用して、ポリシー・ツール・ユーティリティーを開始します。

   ```
   C:\mqipt\java\jre\bin\policytool
   ```

   c) **ファイル** > **オープン** をクリックして C:\mqiptHome\mqipt.policy.を選択します。
   d)「**ポリシー項目の編集**」をクリックして、CodeBase を変更します。変更前:

   ```
   file:/C:/Program Files/IBM/IBM MQ Internet Pass-Thru/lib/com.ibm.mq.ipt.jar
   ```

   to:-->

   ```
   file:/C:/mqipt/lib/com.ibm.mq.ipt.jar
   ```

   e) IBM MQ Internet Pass-Thru、errors および logs ディレクトリーのファイルへのアクセス権限を変更します。変更前:

   ```
   C:\Program Files\IBM\IBM MQ Internet Pass-Thru
   ```

   to:-->

   ```
   C:\mqiptHome
   ```

   f) 他のファイルの許可を変更します。変更前:

   ```
   C:\Program Files\IBM\IBM MQ Internet Pass-Thru
   ```

   to:-->

   ```
   C:\mqipt
   ```

   g)「**アクセス権の追加**」をクリックします。
   以下のようにフィールドに入力します。

   アクセス権: java.net.SocketPermission
   ターゲット: client1.company1.com:1024-
   アクション: accept, listen, resolve

   h)「**ファイル**」>「**保存**」をクリックして、ポリシー・ファイルへの変更を保存します。
   i) mqipt.conf を編集します。
      i) 以下の 2 つのプロパティーを [global] セクションに追加します。

      ```
      SecurityManager=true
      SecurityManagerPolicy=C:\mqiptHome\mqipt.policy
      ```

      ii) 以下の経路定義を追加します。

      ```
      [route]
      ListenerPort=1415
      Destination=server1.company2.com
      DestinationPort=1414
      ```

2. MQIPT を開始します。
   コマンド・プロンプトを開いて、以下のように入力します。

```
C:\mqipt\bin\mqipt C:\mqiptHome -n ipt1
```

ここで、C:\mqiptHome は MQIPT 構成ファイル、mqipt.conf の場所を示し、および ipt1 は MQIPT のインスタンスに指定される名前です。

以下のメッセージは、MQIPT が正常に開始したことを示します。

```
5724-H72 (C) Copyright IBM Corp. 2000, 2024. All Rights Reserved
MQCPI001 IBM MQ Internet Pass-Thru V9.2.0.0 starting
MQCPI004 Reading configuration information from mqipt.conf
MQCPI152 MQIPT name is ipt1
MQCPI055 Setting the java.security.policy to C:\mqiptHome\mqipt.policy
MQCPI053 Starting the Java Security Manager
MQCPI021 Password checking has been enabled on the command port
MQCPI011 The path C:\mqiptHome\logs will be used to store the log files
MQCPI006 Route 1415 has started and will forward messages to :
MQCPI034 ....server1.company2.com(1414)
MQCPI035 ....using MQ protocol
MQCPI078 Route 1415 ready for connection requests
```

3. IBM MQ クライアント・システムのコマンド・プロンプトで、以下のコマンドを入力します。
   a) **MQSERVER** 環境変数を設定します。

   ```
   SET MQSERVER=MQIPT.CONN.CHANNEL/tcp/10.9.1.2(1415)
   ```

   b) メッセージを書き込みます。

   ```
   amqsputc MQIPT.LOCAL.QUEUE MQIPT.QM1
   Hello world
   ```

   メッセージのストリングを入力した後、Enter キーを 2 回押します。
   c) メッセージを読み取ります。

   ```
   amqsgetc MQIPT.LOCAL.QUEUE MQIPT.QM1
   ```

   メッセージ「Hello world」が戻ります。

SOCKS プロキシーの構成

このシナリオでは、MQIPT を SOCKS プロキシーとして機能させることができます。

始める前に

- このシナリオを開始する前に、167 ページの『IBM MQ Internet Pass-Thru 入門』でリストしている前提条件作業を必ず完了してください。
- IBM MQ コンピューター全体または IBM MQ クライアント・アプリケーション (**amqsputc** および **amqsgetc**) のみで SOCKS を使用可能にしてください。
- SOCKS クライアントを以下のように構成してください。
  1. MQIPT を SOCKS プロキシーとして使用します。
  2. SOCKS 5 サポートを使用可能にします。
  3. ユーザー認証を使用不可にします。
  4. 接続を MQIPT ネットワーク・アドレスに制限します。

このタスクについて

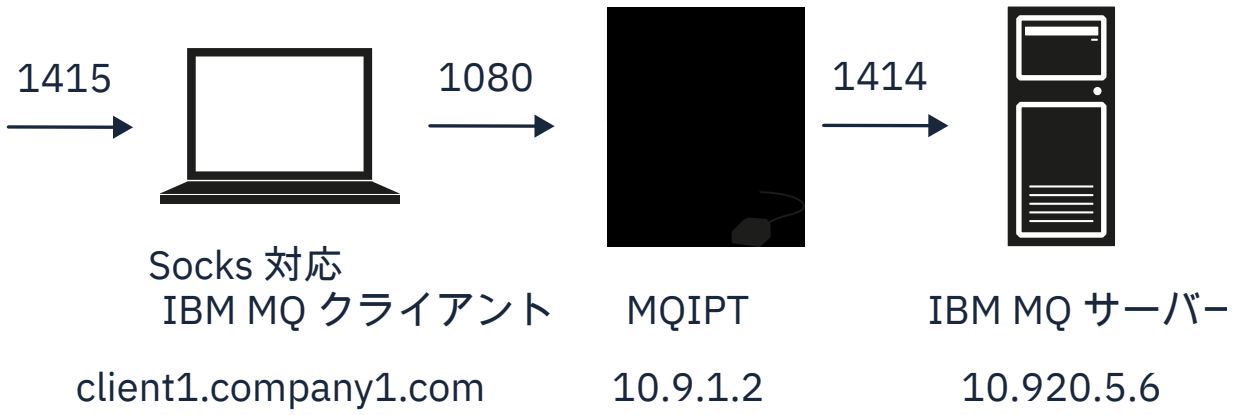


図 *36. SOCKS* プロキシーのネットワーク図

この図では、IBM MQ クライアント (ポート 1415 の client1.company1.com) から MQIPT を通った IBM MQ サーバー (ポート 1414 の server1.company2.com) への接続フローを示しています。

手順

SOCKS プロキシーを構成するには、以下の手順を実行します。

1. 以下のように、MQIPT を構成して開始します。
   a) `mqipt.conf` を編集して、次のルート定義を追加してください。

   ```
   [route]
   ListenerPort=1080
   Destination=server1.company2.com
   DestinationPort=1414
   SocksServer=true
   ```

   実際の宛先は SOCKS ハンドシェーク・プロセス中に IBM MQ クライアントから取得されるため、**Destination** および **DestinationPort** 経路プロパティーの値は無視されます。

   b) コマンド・プロンプトを開き、MQIPT を開始します。

   ```
   C:\mqipt\bin\mqipt C:\mqiptHome -n ipt1
   ```

   ここで、`C:\mqiptHome` は MQIPT 構成ファイル、`mqipt.conf` の場所を示し、および `ipt1` は MQIPT のインスタンスに指定される名前です。

   以下のメッセージは、MQIPT が正常に開始したことを示します。

   ```
   5724-H72 (C) Copyright IBM Corp. 2000, 2024. All Rights Reserved
   MQCPI001 IBM MQ Internet Pass-Thru V9.2.0.0 starting
   MQCPI004 Reading configuration information from mqipt.conf
   MQCPI152 MQIPT name is ipt1
   MQCPI021 Password checking has been enabled on the command port
   MQCPI011 The path C:\mqiptHome\logs will be used to store the log files
   MQCPI006 Route 1080 has started and will forward messages to :
   MQCPI034 ....server1.company2.com(1414)
   MQCPI035 ....using MQ protocol
   MQCPI052 ....SOCKS server side enabled
   MQCPI078 Route 1080 ready for connection requests
   ```

2. IBM MQ クライアント・システムのコマンド・プロンプトで、以下のコマンドを入力します。
   a) **MQSERVER** 環境変数を設定します。

   ```
   SET MQSERVER=MQIPT.CONN.CHANNEL/tcp/10.20.5.6(1414)
   ```

b) メッセージを書き込みます。

```
amqsputc MQIPT.LOCAL.QUEUE MQIPT.QM1
Hello world
```

メッセージのストリングを入力した後、Enter キーを 2 回押します。

c) メッセージを読み取ります。

```
amqsgetc MQIPT.LOCAL.QUEUE MQIPT.QM1
```

メッセージ「Hello world」が戻ります。

SOCKS クライアントの構成

このシナリオでは、既存の SOCKS プロキシーを使用して、MQIPT を SOCKS 対応であるかのように実行できます。

これは、シナリオ 186 ページの『SOCKS プロキシーの構成』と似ていますが、 MQIPT が IBM MQ クライアントの代わりに SOCKS 対応接続を行う点が異なります。

始める前に

このシナリオを開始する前に、167 ページの『IBM MQ Internet Pass-Thru 入門』でリストしている前提条件作業を必ず完了してください。

このタスクについて

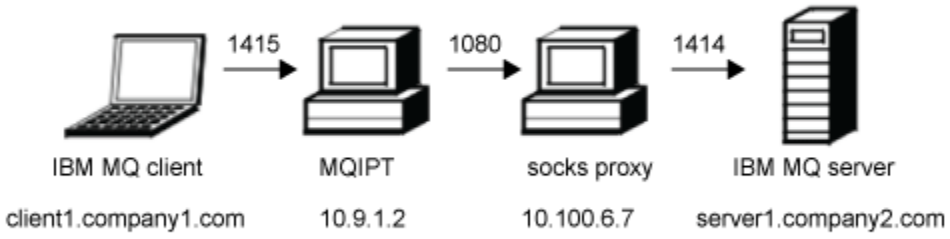


図 *37. SOCKS* クライアントのネットワーク図

この図では、IBM MQ クライアント (ポート 1415 の client1.company1.com) から MQIPT および SOCKS プロキシー (ポート 1080) を通った IBM MQ サーバー (ポート 1414 の server1.company2.com) へのネットワーク接続を示しています。

手順

SOCKS クライアントを構成するには、以下の手順を実行します。

1. MQIPT をセットアップします。

   MQIPT コンピューターで、 `mqipt.conf` を編集し、ルート定義を追加します。

   ```
   [route]
   ListenerPort=1415
   Destination=server1.company2.com
   DestinationPort=1414
   SocksClient=true
   SocksProxyHost=10.9.6.7
   SocksProxyPort=1080
   ```

2. MQIPT を開始します。

   コマンド・プロンプトを開いて、以下のコマンドを入力します。

```
C:\mqipt\bin\mqipt C:\mqiptHome -n ipt1
```

ここで、C:\mqiptHome は MQIPT 構成ファイル、mqipt.conf の場所を示し、および ipt1 は MQIPT のインスタンスに指定される名前です。

以下のメッセージは、MQIPT が正常に開始したことを示します。

```
5724-H72 (C) Copyright IBM Corp. 2000, 2024. All Rights Reserved
MQCPI001 IBM MQ Internet Pass-Thru V9.2.0.0 starting
MQCPI004 Reading configuration information from mqipt.conf
MQCPI152 MQIPT name is ipt1
MQCPI021 Password checking has been enabled on the command port
MQCPI011 The path C:\mqiptHome\logs will be used to store the log files
MQCPI006 Route 1415 has started and will forward messages to :
MQCPI034 ....server1.company2.com(1414)
MQCPI035 ....using MQ protocol
MQCPI039 ....and SOCKS proxy at 10.9.6.7(1080)
MQCPI078 Route 1415 ready for connection requests
```

3. IBM MQ クライアントのコマンド・プロンプトで以下のコマンドを入力します。

   a) **MQSERVER** 環境変数を設定します。

   ```
   SET MQSERVER=MQIPT.CONN.CHANNEL/tcp/10.9.1.2(1415)
   ```

   b) メッセージを書き込みます。

   ```
   amqsputc MQIPT.LOCAL.QUEUE MQIPT.QM1
   Hello world
   ```

   メッセージのストリングを入力した後、Enter キーを 2 回押します。

   c) メッセージを読み取ります。

   ```
   amqsgetc MQIPT.LOCAL.QUEUE MQIPT.QM1
   ```

   メッセージ「Hello world」が戻ります。

MQIPT クラスタリング・サポートの構成

このシナリオでは、クラスタリング環境をセットアップできます。

始める前に

- このシナリオを開始する前に、167 ページの『IBM MQ Internet Pass-Thru 入門』でリストしている前提条件作業を必ず完了してください。
- IBM MQ サーバー LONDON の場合:
  - キュー・マネージャー LONDON を定義します。
  - サーバー接続チャネル MQIPT.CONN.CHANNEL を定義します。
  - ポート 1414 で LONDON の TCP/IP リスナーを開始します。
  - キュー・マネージャーを SOCKS 対応にします。
- IBM MQ サーバー NEWYORK の場合:
  - キュー・マネージャー NEWYORK を定義します。
  - サーバー接続チャネル MQIPT.CONN.CHANNEL を定義します。
  - ポート 1414 で NEWYORK の TCP/IP リスナーを開始します。
  - キュー・マネージャーを SOCKS 対応にします。

注 : キュー・マネージャーを SOCKS 対応にするには、コンピューター全体または IBM MQ サーバー・アプリケーションのみを使用可能にします。 SOCKS クライアントを以下のように構成してください。

- MQIPT のクライアントを SOCKS プロキシーとして指します。
- SOCKS V5 サポートを使用可能にします。

- ユーザー認証を使用不可にします。
- MQIPT へのリモート接続のみ行います。

このタスクについて

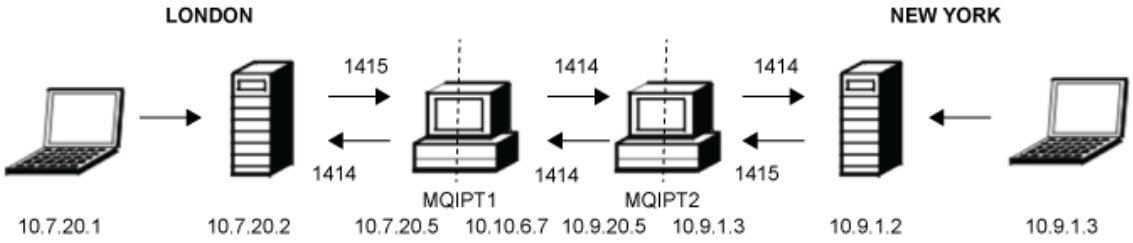


図 *38.* クラスタリングのネットワーク図

この図では、IBM MQ クライアントから MQIPT を通った IBM MQ サーバーへの接続を示しています。

同じコンピューターの指定のポートで listen できるのは 1 つのアプリケーションのみです。 ポート 1414 が既に使用されている場合、空いているポートを選択して、例では置換してください。

LONDON サーバーのローカル・キューにメッセージを書き込み、NEWYORK サーバーからそれを取り出すことによって、キュー・マネージャー間の経路をテストできます。

手順

MQIPT クラスタリング・サポートを構成するには、以下の手順を実行します。

1. LONDON サーバーをセットアップします。
   コマンド・プロンプトを開いて、以下のコマンドを入力します。

```
runmqsc
DEFINE CHANNEL(TO.LONDON) +
       CHLTYPE(CLUSRCVR) TRPTYPE(TCP) +
       CLUSTER(INVENTORY) +
       CONNAME('10.10.6.7(1414)')
DEFINE CHANNEL(TO.NEWYORK) +
       CHLTYPE(CLUSSDR) TRPTYPE(TCP) +
       CLUSTER(INVENTORY) +
       CONNAME('10.9.20.5(1414)')
```

2. NEWYORK サーバーをセットアップします。
   コマンド・プロンプトを開いて、以下のコマンドを入力します。

```
runmqsc
ALTER QMGR REPOS(INVENTORY)
DEFINE QLOCAL(MQIPT.LOCAL.QUEUE) +
       CLUSTER(INVENTORY)
DEFINE CHANNEL(TO.NEWYORK) +
       CHLTYPE(CLUSRCVR) TRPTYPE(TCP) +
       CLUSTER(INVENTORY) +
       CONNAME('10.9.20.5(1414)')
DEFINE CHANNEL(TO.LONDON) +
       CHLTYPE(CLUSSDR) TRPTYPE(TCP) +
       CLUSTER(INVENTORY) +
       CONNAME('10.10.6.7(1414)')
```

3. MQIPT 1 をセットアップします。
   `mqipt.conf` を編集して、次のルート定義を追加します。

```
[route]
Name=LONDON to NEWYORK
ListenerPort=1415
Destination=10.9.20.5
DestinationPort=1414
```

```
SocksServer=true

[route]
Name=MQIPT1 to LONDON
ListenerPort=1414
Destination=10.7.20.2
DestinationPort=1414
```

4. MQIPT 1 を開始します。
   コマンド・プロンプトを開いて、以下のコマンドを入力します。

```
C:\mqipt\bin\mqipt C:\mqiptHome -n ipt1
```

   ここで、C:\mqiptHome は MQIPT 構成ファイル、mqipt.conf の場所を示し、および ipt1 は MQIPT のインスタンスに指定される名前です。

   以下のメッセージは、MQIPT が正常に開始したことを示します。

```
5724-H72 (C) Copyright IBM Corp. 2000, 2024. All Rights Reserved
MQCPI001 IBM MQ Internet Pass-Thru V9.2.0.0 starting
MQCPI004 Reading configuration information from mqipt.conf
MQCPI152 MQIPT name is ipt1
MQCPI021 Password checking has been enabled on the command port
MQCPI011 The path C:\mqiptHome\logs will be used to store the log files
MQCPI006 Route 1415 has started and will forward messages to :
MQCPI034 ....10.9.20.5(1414)
MQCPI035 ....using MQ protocol
MQCPI052 ....SOCKS server side enabled
MQCPI078 Route 1415 ready for connection requests
MQCPI006 Route 1414 has started and will forward messages to :
MQCPI034 ....10.7.20.2(1414)
MQCPI035 ....using MQ protocol
MQCPI078 Route 1414 ready for connection requests
```

5. MQIPT 2 をセットアップします。
   mqipt.conf を編集して、次のルート定義を追加します。

```
[route]
Name=NEWYORK to LONDON
ListenerPort=1415
Destination=10.10.6.7
DestinationPort=1414
SocksServer=true

[route]
Name=MQIPT2 to NEWYORK
ListenerPort=1414
Destination=10.9.1.2
DestinationPort=1414
```

6. MQIPT 2 を開始します。
   コマンド・プロンプトを開いて、以下のコマンドを入力します。

```
C:
cd \mqipt\bin
mqipt .. -n ipt2
```

   ここで、.. は MQIPT 構成ファイル mqipt.conf が親ディレクトリーにあることを示し、ipt2 は MQIPT のインスタンスに付けられる名前です。

   以下のメッセージは、MQIPT が正常に開始したことを示します。

```
5724-H72 (C) Copyright IBM Corp. 2000, 2024. All Rights Reserved
MQCPI001 IBM MQ Internet Pass-Thru V9.2.0.0 starting
MQCPI004 Reading configuration information from mqipt.conf
MQCPI152 MQIPT name is ipt2
MQCPI021 Password checking has been enabled on the command port
MQCPI011 The path C:\mqipt\logs will be used to store the log files
MQCPI006 Route 1415 has started and will forward messages to :
MQCPI034 ....10.10.6.7(1414)
MQCPI035 ....using MQ protocol
MQCPI052 ....SOCKS server side enabled
MQCPI078 Route 1415 ready for connection requests
MQCPI006 Route 1414 has started and will forward messages to :
```

```
MQCPI034 ....10.9.1.2(1414)
MQCPI035 ....using MQ protocol
MQCPI078 Route 1414 ready for connection requests
```

7. LONDON IBM MQ クライアント (10.7.20.1) のコマンド・プロンプトで以下のコマンドを入力します。
   a) **MQSERVER** 環境変数を設定します。

   ```
   SET MQSERVER=MQIPT.CONN.CHANNEL/tcp/10.7.20.2(1414)
   ```

   b) メッセージを書き込みます。

   ```
   amqsputc MQIPT.LOCAL.QUEUE LONDON
   Hello world
   ```

   メッセージのストリングを入力した後、Enter キーを 2 回押します。

   これによって、LONDON キュー・マネージャーが NEW YORK キュー・マネージャーのキューにメッセージを送信します。
8. NEW YORK IBM MQ クライアント (10.9.1.3) のコマンド・プロンプトで以下のコマンドを入力します。
   a) **MQSERVER** 環境変数を設定します。

   ```
   SET MQSERVER=MQIPT.CONN.CHANNEL/TCP/10.9.1.2(1414)
   ```

   b) メッセージを読み取ります。

   ```
   amqsgetc MQIPT.LOCAL.QUEUE NEWYORK
   ```

   メッセージ「Hello world」が戻ります。

ポート番号の割り振り

発信接続時に使用されるローカル・ポート・アドレスを制御できます。 例えば、ファイアウォールで特定範囲のポート番号のみが許可されている場合、MQIPT を使用して、有効なポートから出力が発信されるようにできます。

始める前に

- このシナリオを開始する前に、167 ページの『IBM MQ Internet Pass-Thru 入門』でリストしている前提条件作業を必ず完了してください。
- マルチホームのコンピューターに MQIPT をインストールします。

このタスクについて

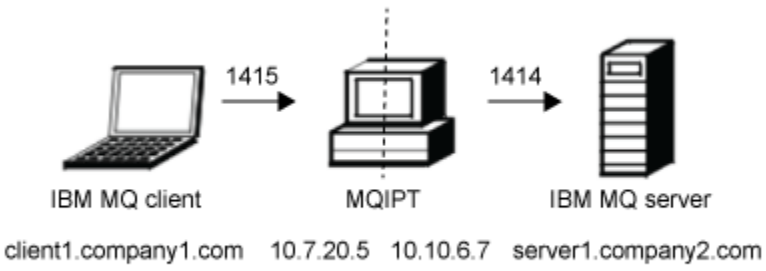


図 *39.* ポート割り振りのネットワーク図

この図では、IBM MQ クライアント (ポート 1415 の client1.company1.com) から MQIPT を通った IBM MQ サーバー (ポート 1414 の server1.company2.com) への接続を示しています。

手順

ポート番号を割り振るには、以下の手順を実行します。

1. MQIPT をセットアップします。

   mqipt.conf を編集して、次のルート定義を追加してください。

   ```
   [route]
   ListenerPort=1415
   Destination=server1.company2.com
   DestinationPort=1414
   LocalAddress=10.10.6.7
   OutgoingPort=2000
   MaxConnectionThreads=20
   ```

2. MQIPT を開始します。

   IBM MQ システムのコマンド・プロンプトを開き、以下のコマンドを入力します。

   ```
   C:\mqipt\bin\mqipt C:\mqiptHome -n ipt1
   ```

   ここで、C:\mqiptHome は MQIPT 構成ファイル、mqipt.conf の場所を示し、および ipt1 は MQIPT のインスタンスに指定される名前です。

   以下のメッセージは、MQIPT が正常に開始したことを示します。

   ```
   5724-H72 (C) Copyright IBM Corp. 2000, 2024. All Rights Reserved
   MQCPI001 IBM MQ Internet Pass-Thru V9.2.0.0 starting
   MQCPI004 Reading configuration information from mqipt.conf
   MQCPI152 MQIPT name is ipt1
   MQCPI021 Password checking has been enabled on the command port
   MQCPI011 The path C:\mqiptHome\logs will be used to store the log files
   MQCPI006 Route 1415 is starting and will forward messages to :
   MQCPI034 ....server1.company2.com(1414)
   MQCPI035 ....using MQ protocol
   MQCPI069 ....binding to local address 10.10.6.7 when making new connections
   MQCPI070 ....using local port address range 2000-2019 when making new connections
   MQCPI078 Route 1415 ready for connection requests
   ```

3. IBM MQ クライアント・システムのコマンド・プロンプトで、以下のコマンドを入力します。

   a) **MQSERVER** 環境変数を設定します。

   ```
   SET MQSERVER=MQIPT.CONN.CHANNEL/tcp/10.7.20.5(1415)
   ```

   b) メッセージを書き込みます。

   ```
   amqsputc MQIPT.LOCAL.QUEUE MQIPT.QM1
   Hello world
   ```

   メッセージのストリングを入力した後、Enter キーを 2 回押します。

   c) メッセージを読み取ります。

   ```
   amqsgetc MQIPT.LOCAL.QUEUE MQIPT.QM1
   ```

   メッセージ「Hello world」が戻ります。

LDAP サーバーを使用した CRL の取得

LDAP サーバーを使用して証明書取り消しリスト (CRL) を取得するように MQIPT を構成できます。

始める前に

- このシナリオを開始する前に、167 ページの『IBM MQ Internet Pass-Thru 入門』 でリストしている前提条件作業を必ず完了してください。
- MQIPT 2 には、 myCert.pfx という鍵リング・ファイルに保管されている、信頼できる認証局 (CA) によって発行された個人証明書があることを確認してください。

- MQIPT 1 に、 MQIPT 2 によって送信された証明書の認証に使用される信頼できる CA 証明書のコピーがあることを確認してください。 この証明書は、 `caCerts.pfx` という鍵リング・ファイルに保管されます。
- 鍵リングにアクセスするためのパスワードは、**mqiptPW** コマンドを使用して暗号化されています。

このタスクについて

このシナリオでは、IBM MQ クライアントをキュー・マネージャー (QM) に接続でき、IBM MQ メッセージをターゲット・キューに置くことができます。 MQIPT トレースを MQIPT 1 で実行すると、LDAP サーバーが使用されていることが示されます。

CRL がどのように機能するかを実演するには、MQIPT 2 で使用される個人証明書がトラステッド CA によって取り消されることを確認します。 その後、 MQIPT 1 から MQIPT 2 への接続が拒否されるため、 IBM MQ クライアントは QM への接続を許可されません。

このシナリオの目的は、LDAP サーバーのインストール方法やセットアップ方法を説明したり、個人証明書やトラステッド証明書を含む鍵リング・ファイルの作成方法を説明したりすることではありません。 LDAP サーバーは既知のトラステッド CA によって使用可能にされているものとします。 バックアップ LDAP サーバーは使用されませんが、適切な経路プロパティーを追加することによって実装することもできます。

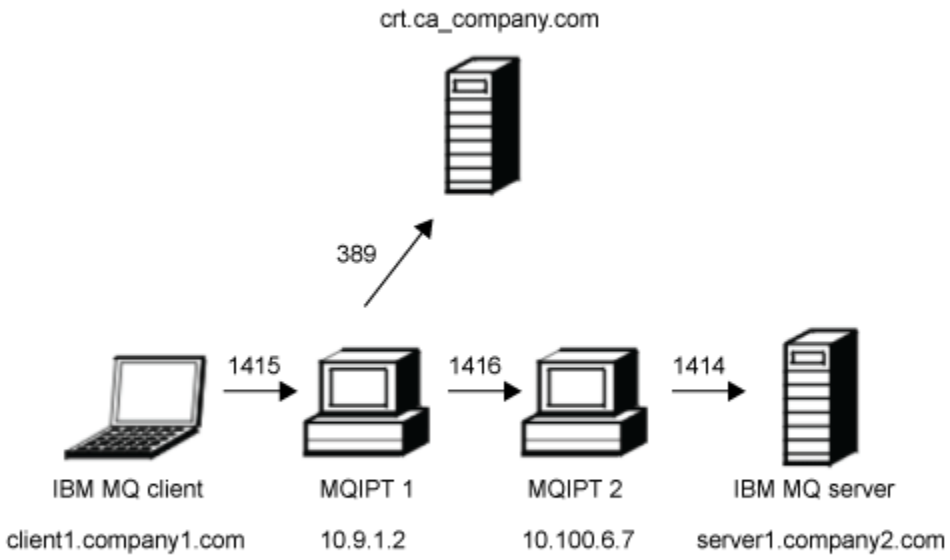


図 *40. LDAP* サーバーのネットワーク図

この図では、IBM MQ クライアント (ポート 1415 の client1.company1.com) から MQIPT の 2 つのインスタンスを通った IBM MQ サーバー (ポート 1414 の server1.company2.com) への接続を示しています。 最初の MQIPT は LDAP サーバー (ポート 389 の crl.ca_company.com) に接続しています。

手順

LDAP サーバーを使用して CRL を取得するには、以下の手順を実行します。

1. MQIPT 1 システムの場合:

   a) `mqipt.conf` を編集して、次のルート定義を追加してください。

```
[route]
ListenerPort=1415
Destination=10.100.6.7
DestinationPort=1416
SSLClient=true
SSLClientCAKeyRing=C:\mqiptHome\ssl\caCerts.pfx
SSLClientCAKeyRingPW=encrypted_key_ring_password
LDAP=true
LDAPServer1=crl.ca_company.com
LDAPServer1Timeout=4
```

encrypted_key_ring_password は、**mqiptPW** コマンドを使用して暗号化された、caCerts.pfx 鍵リングのパスワードです。

b) コマンド・プロンプトを開き、MQIPT を開始します。

```
C:\mqipt\bin\mqipt C:\mqiptHome -n ipt1
```

ここで、C:\mqiptHome は MQIPT 構成ファイル、mqipt.conf の場所を示し、および ipt1 は MQIPT のインスタンスに指定される名前です。

以下のメッセージは、MQIPT が正常に開始したことを示します。

```
5724-H72 (C) Copyright IBM Corp. 2000, 2024. All Rights Reserved
MQCPI001 IBM MQ Internet Pass-Thru V9.2.0.0 starting
MQCPI004 Reading configuration information from mqipt.conf
MQCPI152 MQIPT name is ipt1
MQCPI021 Password checking has been enabled on the command port
MQCPI011 The path C:\mqiptHome\logs will be used to store the log files
MQCPI006 Route 1415 has started and will forward messages to :
MQCPI034 ....10.100.6.7(1416)
MQCPI035 ....using MQ protocol
MQCPI036 ....SSL Client side enabled with properties :
MQCPI031 ......CipherSuites <NULL>
MQCPI032 ......key ring file <NULL>
MQCPI047 ......CA key ring file C:\mqiptHome\ssl\caCerts.pfx
MQCPI071 ......site certificate uses UID=*,CN=*,T=*,OU=*,DC=*,O=*,
                                                STREET=*,L=*,ST=*,PC=*,C=*,DNQ=*
MQCPI038 ......peer certificate uses UID=*,CN=*,T=*,OU=*,DC=*,O=*,
                                                STREET=*,L=*,ST=*,PC=*,C=*,DNQ=*
MQCPI075 ....LDAP main server at crl.ca_company.com(389)
MQCPI086 ......timeout of 4 second(s)
MQCPI084 ....CRL cache expiry timeout is 1 hour(s)
MQCPI085 ....CRLs will be saved in the key-ring file(s)
MQCPI078 Route 1415 ready for connection requests
```

2. MQIPT 2 システムの場合:

a) mqipt.conf を編集して、次のルート定義を追加してください。

```
[route]
ListenerPort=1416
Destination=server1.company2.com
DestinationPort=1414
SSLServer=true
SSLServerKeyRing=C:\mqipt\ssl\myCert.pfx
SSLServerKeyRingPW=encrypted_key_ring_password
```

encrypted_key_ring_password は、**mqiptPW** コマンドを使用して暗号化された、myCert.pfx 鍵リングのパスワードです。

b) コマンド・プロンプトを開き、MQIPT を開始します。

```
C:
cd \mqipt\bin
mqipt .. -n ipt2
```

ここで、.. は MQIPT 構成ファイル mqipt.conf が親ディレクトリーにあることを示し、ipt2 は MQIPT のインスタンスに付けられる名前です。

以下のメッセージは、正常に完了したことを示します。

```
5724-H72 (C) Copyright IBM Corp. 2000, 2024. All Rights Reserved
MQCPI001 IBM MQ Internet Pass-Thru V9.2.0.0 starting
```

```
MQCPI004 Reading configuration information from mqipt.conf
MQCPI152 MQIPT name is ipt2
MQCPI021 Password checking has been enabled on the command port
MQCPI011 The path C:\mqipt\logs will be used to store the log files
MQCPI006 Route 1416 is starting and will forward messages to :
MQCPI034 ....server1.company2.com(1414)
MQCPI035 ....using MQ protocol
MQCPI037 ....SSL Server side enabled with properties :
MQCPI031 ......CipherSuites <NULL>
MQCPI032 ......key ring file C:\mqipt\ssl\myCert.pfx
MQCPI047 ......CA key ring file <NULL>
MQCPI071 ......site certificate uses UID=*,CN=*,T=*,OU=*,DC=*,O=*,
                                          STREET=*,L=*,ST=*,PC=*,C=*,DNQ=*
MQCPI038 ......peer certificate uses UID=*,CN=*,T=*,OU=*,DC=*,O=*,
                                          STREET=*,L=*,ST=*,PC=*,C=*,DNQ=*
MQCPI033 ......client authentication set to false
MQCPI078 Route 1416 ready for connection requests
```

3. IBM MQ クライアント・システムのコマンド・プロンプトで、以下のコマンドを入力します。

   a) **MQSERVER** 環境変数を設定します。

   ```
   SET MQSERVER=MQIPT.CONN.CHANNEL/tcp/10.9.1.2(1415)
   ```

   b) メッセージを書き込みます。

   ```
   amqsputc MQIPT.LOCAL.QUEUE MQIPT.QM1
   Hello world
   ```

   メッセージのストリングを入力した後、Enter キーを 2 回押します。

   c) メッセージを読み取ります。

   ```
   amqsgetc MQIPT.LOCAL.QUEUE MQIPT.QM1
   ```

   メッセージ「Hello world」が戻ります。

MQIPT の TLS プロキシー・モードでの実行

MQIPT を TLS プロキシー・モードで実行できるので、IBM MQ TLS クライアントからの TLS 接続要求を受け入れて IBM MQ TLS サーバーにトンネル接続させることができます。

始める前に

このシナリオを開始する前に、167 ページの『IBM MQ Internet Pass-Thru 入門』でリストしている前提条件作業を必ず完了してください。

このタスクについて

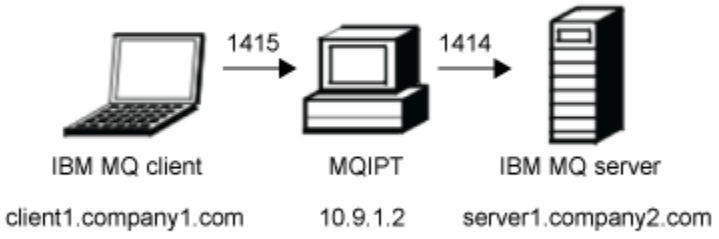


図 *41. SSL/TLS* プロキシー・モードのネットワーク図

この図では、IBM MQ クライアント (ポート 1415 の client1.company1.com) から MQIPT を通った IBM MQ サーバー (ポート 1414 の server1.company2.com) への接続フローを示しています。

IBM MQ の TLS 構成の詳細については、SSL または TLS の取り扱いを参照してください。

手順

MQIPT を TLS プロキシー・モードで実行するには、以下の手順を実行します。

1. IBM MQ クライアントおよびサーバーで TLS 接続が使用されるように構成します。
   a) キュー・マネージャーのキー・リポジトリーを作成する。
      詳しくは、AIX, Linux, and Windows での鍵リポジトリーのセットアップを参照してください。
   b) `C:\ProgramData\IBM\MQ` ディレクトリーにクライアント用の鍵リポジトリーを作成します。 このリポジトリーに *clientkey.kdb* という名前を付けます。
   c) ステップ 197 ページの『1.a』 で作成したキュー・マネージャーの鍵リポジトリー内に、キュー・マネージャーの個人証明書を作成します。
      詳しくは、AIX, Linux, and Windows での自己署名個人証明書の作成を参照してください。
   d) ステップ 197 ページの『1.b』 で作成したクライアントの鍵リポジトリー内に、クライアントの個人証明書を作成します。
   e) サーバーのキー・リポジトリーから個人証明書を抽出し、クライアント・リポジトリーに追加する。
      詳しくは、AIX, Linux, and Windows で鍵リポジトリーから自己署名証明書の公開部分を抽出する、および AIX, Linux, and Windows システムで CA 証明書 (または自己署名証明書の公開部分) を鍵リポジトリーに追加するを参照してください。
   f) クライアントのキー・リポジトリーから個人証明書を抽出し、サーバーのキー・リポジトリーに追加する。
   g) 以下の MQSC コマンドを使用して、MQIPT.CONN.CHANNEL サーバー接続チャネルを変更し、TLS が使用されるようにします。

```
ALTER CHANNEL(MQIPT.CONN.CHANNEL) CHLTYPE(SVRCONN) TRPTYPE(TCP)
SSLCIPH(TLS_RSA_WITH_AES_128_CBC_SHA256)
```

2. MQIPT を TLS プロキシー・モードで実行するには、以下の手順を実行します。
   a) `mqipt.conf` を編集して、次のルート定義を追加してください。

```
[route]
ListenerPort=1415
Destination=server1.company2.com
DestinationPort=1414
SSLProxyMode=true
```

   b) MQIPT を開始します。
      コマンド・プロンプトを開いて、以下のコマンドを入力します。

```
C:\mqipt\bin\mqipt C:\mqiptHome -n ipt1
```

      ここで、`C:\mqiptHome` は MQIPT 構成ファイル、`mqipt.conf` の場所を示し、および `ipt1` は MQIPT のインスタンスに指定される名前です。

      以下のメッセージは、MQIPT が正常に開始したことを示します。

```
5724-H72 (C) Copyright IBM Corp. 2000, 2024. All Rights Reserved
MQCPI001 IBM MQ Internet Pass-Thru V9.2.0.0 starting
MQCPI004 Reading configuration information from mqipt.conf
MQCPI152 MQIPT name is ipt1
MQCPI021 Password checking has been enabled on the command port
MQCPI011 The path C:\mqiptHome\logs will be used to store the log files
MQCPI006 Route 1415 has started and will forward messages to :
MQCPI034 ....server1.company2.com(1414)
MQCPI035 ....using SSLProxyMode protocol
MQCPI078 Route 1415 ready for connection requests
```

3. IBM MQ クライアント・システムのコマンド・プロンプトで、以下のコマンドを入力して TLS サンプル・プログラムを実行します。

```
AMQSSSLC -m MQIPT.QM1 -c MQIPT.CONN.CHANNEL -x 10.9.1.2(1415)
         -k "C:\ProgramData\IBM\MQ\clientkey" -l cert_label -s
TLS_RSA_WITH_AES_128_CBC_SHA256
```

`cert_label` は、ステップ 197 ページの『1.d』 で作成したクライアント証明書のラベルです。

セキュリティー・マネージャーを使用した TLS プロキシー・モードでの MQIPT の実行

MQIPT を TLS プロキシー・モードで実行できるので、IBM MQ TLS クライアントからの TLS 接続要求を受け入れて IBM MQ TLS サーバーにトンネル接続させることができます。 MQIPT でセキュリティー・マネージャーを使用することによって、メッセージを送信できるアドレスを制限できます。

始める前に

注 : Deprecated MQIPT での Java security manager の使用は、 Java の将来のリリースで削除するために Java security manager が非推奨になったため、非推奨になりました。

このシナリオを開始する前に、167 ページの『IBM MQ Internet Pass-Thru 入門』 でリストしている前提条件作業を必ず完了してください。

このタスクについて

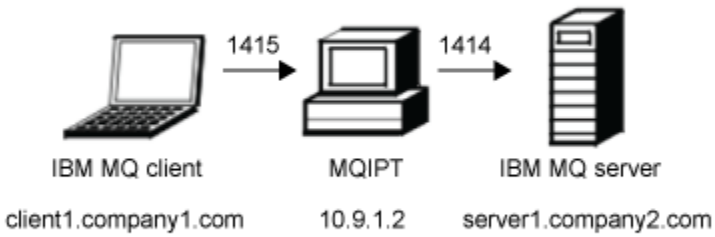


図 *42. SSL/TLS* プロキシー・モードのネットワーク図

この図では、IBM MQ クライアント (ポート 1415 の client1.company1.com) から MQIPT を通った IBM MQ サーバー (ポート 1414 の server1.company2.com) への接続フローを示しています。

IBM MQ の TLS 構成の詳細については、SSL または TLS の取り扱いを参照してください。

手順

セキュリティー・マネージャーを使用して MQIPT を TLS プロキシー・モードで実行するには、以下の手順を実行します。

1. IBM MQ クライアントおよびサーバーで TLS 接続が使用されるように構成します。
   a) キュー・マネージャーのキー・リポジトリーを作成する。
      詳しくは、AIX, Linux, and Windows での鍵リポジトリーのセットアップを参照してください。
   b) `C:\ProgramData\IBM\MQ` ディレクトリーにクライアント用の鍵リポジトリーを作成します。 このリポジトリーに *clientkey.kdb* という名前を付けます。
   c) ステップ 198 ページの『1.a』 で作成したキュー・マネージャーの鍵リポジトリー内に、キュー・マネージャーの個人証明書を作成します。
      詳しくは、AIX, Linux, and Windows での自己署名個人証明書の作成を参照してください。
   d) ステップ 198 ページの『1.b』 で作成したクライアントの鍵リポジトリー内に、クライアントの個人証明書を作成します。

e) サーバーのキー・リポジトリーから個人証明書を抽出し、クライアント・リポジトリーに追加する。
詳しくは、AIX, Linux, and Windows で鍵リポジトリーから自己署名証明書の公開部分を抽出する、および AIX, Linux, and Windows システムで CA 証明書 (または自己署名証明書の公開部分) を鍵リポジトリーに追加するを参照してください。

f) クライアントのキー・リポジトリーから個人証明書を抽出し、サーバーのキー・リポジトリーに追加する。

g) 以下の MQSC コマンドを使用して、MQIPT.CONN.CHANNEL サーバー接続チャネルを変更し、TLS が使用されるようにします。

```
ALTER CHANNEL(MQIPT.CONN.CHANNEL) CHLTYPE(SVRCONN) TRPTYPE(TCP)
SSLCIPH(TLS_RSA_WITH_AES_128_CBC_SHA256)
```

2. MQIPT コンピューター上で (図を参照)、コマンド・プロンプトで次のコマンドを入力して、サンプルの Java security manager・ポリシーを MQIPT のホーム・ディレクトリーにコピーします。

```
copy C:\mqipt\samples\mqiptSample.policy C:\mqiptHome\mqipt.policy
```

3. 以下のコマンドを使用して、ポリシー・ツール・ユーティリティーを開始します。

```
C:\mqipt\java\jre\bin\policytool
```

ポリシー・ツールで以下のようにします。

a) **ファイル** > **オープン** をクリックして C:\mqiptHome\mqipt.policy.を選択します。

b) 以下を選択します。

```
file:/C:/Program Files/IBM/IBM MQ Internet Pass-Thru/lib/com.ibm.mq.ipt.jar
```

次に、「**ポリシー項目の編集**」をクリックします。

c) CodeBase を以下のものから変更します。

```
file:/C:/Program Files/IBM/IBM MQ Internet Pass-Thru/lib/com.ibm.mq.ipt.jar
```

to:-->

```
file:/C:/mqipt/lib/com.ibm.mq.ipt.jar
```

d) IBM MQ Internet Pass-Thru、errors および logs ディレクトリーのファイルへのアクセス権限を変更します。変更前:

```
C:\Program Files\IBM\IBM MQ Internet Pass-Thru
```

to:-->

```
C:\mqiptHome
```

e) 他のファイルの許可を変更します。変更前:

```
C:\Program Files\IBM\IBM MQ Internet Pass-Thru
```

to:-->

```
C:\mqipt
```

f)「**アクセス権の追加**」をクリックします。
以下のようにフィールドに入力します。

アクセス権: java.net.SocketPermission
ターゲット: client1.company1.com:1024-
アクション: accept, listen, resolve

g)「**ファイル**」>「**保存**」をクリックして、ポリシー・ファイルへの変更を保存します。

4. mqipt.conf を編集します。 [global] セクションに以下のプロパティーを追加し、以下の経路定義を追加します。

```
[global]
SecurityManager=true
SecurityManagerPolicy=C:\mqiptHome\mqipt.policy

[route]
ListenerPort=1415
Destination=server1.company2.com
DestinationPort=1414
SSLProxyMode=true
```

5. MQIPT を開始します。
   コマンド・プロンプトを開いて、以下のコマンドを入力します。

```
C:\mqipt\bin\mqipt C:\mqiptHome -n ipt1
```

   ここで、C:\mqiptHome は MQIPT 構成ファイル、mqipt.conf の場所を示し、および ipt1 は MQIPT のインスタンスに指定される名前です。

   以下のメッセージは、MQIPT が正常に開始したことを示します。

```
5724-H72 (C) Copyright IBM Corp. 2000, 2024. All Rights Reserved
MQCPI001 IBM MQ Internet Pass-Thru V9.2.0.0 starting
MQCPI004 Reading configuration information from mqipt.conf
MQCPI152 MQIPT name is ipt1
MQCPI055 Setting the java.security.policy to C:\mqiptHome\mqipt.policy
MQCPI053 Starting the Java Security Manager
MQCPI021 Password checking has been enabled on the command port
MQCPI011 The path C:\mqiptHome\mqipt\logs will be used to store the log files
MQCPI006 Route 1415 has started and will forward messages to :
MQCPI034 ....server1.company2.com(1414)
MQCPI035 ....using SSLProxyMode protocol
MQCPI078 Route 1415 ready for connection requests
```

6. IBM MQ クライアント・システムのコマンド・プロンプトで、以下のコマンドを入力して TLS サンプル・プログラムを実行します。

```
AMQSSSLC -m MQIPT.QM1 -c MQIPT.CONN.CHANNEL -x 10.9.1.2(1415)
         -k "C:\ProgramData\IBM\MQ\clientkey" -l cert_label -s
TLS_RSA_WITH_AES_128_CBC_SHA256
```

   cert_label は、ステップ 198 ページの『1.d』 で作成したクライアント証明書のラベルです。

セキュリティー出口の使用

このシナリオでは、提供されているサンプルのセキュリティー出口 SampleSecurityExit を使用して、文字 MQIPT. で始まるチャネル名を使用するクライアント接続のみを許可するようにできます。

始める前に

- このシナリオを開始する前に、167 ページの『IBM MQ Internet Pass-Thru 入門』 でリストしている前提条件作業を必ず完了してください。
- JavaJava 8.0 JDK をインストールします。
- Java bin サブディレクトリーを **PATH** 環境変数に追加します。

このタスクについて

このシナリオで使用されたサンプル出口は SampleSecurityExit.java です。 インストール済みディレクトリーの samples/exits サブディレクトリー MQIPT の MQIPT が提供されます。

推奨されるサーバー接続チャネル名の MQIPT.CONN.CHANNEL (シナリオの大部分で使用されている) を使用すると、クライアント接続は完了を許可され、IBM MQ メッセージをキューに置くことができます。

セキュリティー出口が予期したとおりに機能していることを示すには、文字 MQIPT. で始まらない任意の名前 (例えば、TEST.CONN.CHANNEL) を使用して別のサーバー接続チャネルを定義し、**amqsputc** コマンドを再試行します。ただし、新しいチャネル名を使用するように **MQSERVER** 環境変数を変更してください。この時の接続は拒否され、 2059 (MQRC_Q_MGR_NOT_AVAILABLE) エラーが返されます。

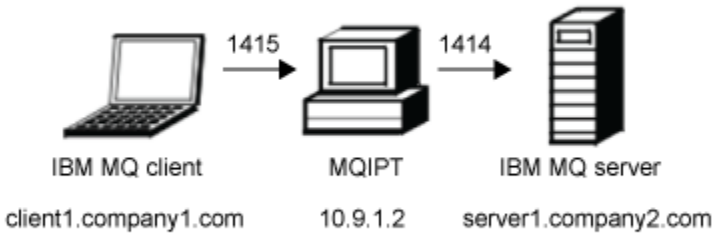


図 *43.* セキュリティー出口のネットワーク図

この図では、IBM MQ クライアント (ポート 1415 の client1.company1.com) から MQIPT を通った IBM MQ サーバー (ポート 1414 の server1.company2.com) への接続フローを示しています。

手順

セキュリティー出口を使用するには、以下の手順を実行します。

1. MQIPT コンピューターの場合:
   a) MQIPT というホーム・ディレクトリーに exits というディレクトリーを作成するため、コマンド・プロンプトで以下のコマンドを発行します。

   ```
   md C:\mqiptHome\exits
   ```

   b) 以下のコマンドを入力して、出口をコンパイルします。 コンパイルされたサンプル出口が MQIPT に付属しているため、出口コードを変更していない場合は、これを行う必要はありません。

   ```
   C:
   cd \mqipt\samples\exits
   javac -classpath C:\mqipt\lib\com.ibm.mq.ipt.jar;. SampleSecurityExit.java
   ```

   c) 以下のコマンドを入力して、コンパイル済みの SampleSecurityExit.class という出口クラス・ファイル を C:\mqiptHome\exits というディレクトリーにコピーします。

   ```
   copy C:\mqipt\samples\exits\SampleSecurityExit.class C:\mqiptHome\exits
   ```

   d) mqipt.conf を編集して、ルート定義を追加します。

   ```
   [route]
   ListenerPort=1415
   Destination=server1.company2.com
   DestinationPort=1414
   SecurityExit=true
   SecurityExitName=SampleSecurityExit
   ```

   e) コマンド・プロンプトを開き、MQIPT を開始します。

   ```
   C:\mqipt\bin\mqipt C:\mqiptHome -n ipt1
   ```

   ここで、C:\mqiptHome は MQIPT 構成ファイル、mqipt.conf の場所を示し、および ipt1 は MQIPT のインスタンスに指定される名前です。

以下のメッセージは、MQIPT が正常に開始したことを示します。

```
5724-H72 (C) Copyright IBM Corp. 2000, 2024. All Rights Reserved
MQCPI001 IBM MQ Internet Pass-Thru V9.2.0.0 starting
MQCPI004 Reading configuration information from mqipt.conf
MQCPI152 MQIPT name is ipt1
MQCPI021 Password checking has been enabled on the command port
MQCPI011 The path C:\mqiptHome\logs will be used to store the log files
MQCPI006 Route 1415 has started and will forward messages to :
MQCPI034 ....server1.company2.com(1414)
MQCPI035 ....using MQ protocol
MQCPI079 ....using security exit C:\mqiptHome\exits\SampleSecurityExit
MQCPI080 ......and timeout of 30 seconds
MQCPI078 Route 1415 ready for connection requests
```

2. IBM MQ クライアント・システムのコマンド・プロンプトで、以下のコマンドを入力します。
   a) **MQSERVER** 環境変数を設定します。

   ```
   SET MQSERVER=MQIPT.CONN.CHANNEL/tcp/10.9.1.2(1415)
   ```

   b) メッセージを書き込みます。

   ```
   amqsputc MQIPT.LOCAL.QUEUE MQIPT.QM1
   Hello world
   ```

   メッセージのストリングを入力した後、Enter キーを 2 回押します。
   c) メッセージを読み取ります。

   ```
   amqsgetc MQIPT.LOCAL.QUEUE MQIPT.QM1
   ```

   メッセージ「Hello world」が戻ります。

セキュリティー出口を使用した IBM MQ キュー・マネージャー・サーバーへのクライアント接続要求の経路指定

このシナリオでは、3 つの IBM MQ キュー・マネージャー・サーバーのグループにラウンドロビン方式でクライアント接続要求を動的に経路指定できます。 グループ内の各サーバーのキュー・マネージャーは同一である必要があります。

始める前に

- このシナリオを開始する前に、167 ページの『IBM MQ Internet Pass-Thru 入門』でリストしている前提条件作業を必ず完了してください。
- Java 8.0 JDK をインストールします。
- Java `bin` サブディレクトリーを **PATH** 環境変数に追加します。

このタスクについて

このシナリオで使用されたサンプル出口は `SampleRoutingExit.java` です。 インストール済みディレクトリーの `samples/exits` サブディレクトリー MQIPT の MQIPT が提供されます。

コンパイルされた出口クラス・ファイルの名前と場所は、MQIPT **SecurityExitName** および **SecurityExitPath** プロパティーで定義されます。

使用される全てのキュー・マネージャーとサーバー名は、`SampleRoutingExit.conf` という構成ファイルから読み取られます。 出口は、構成ファイルが出口クラス・ファイルと同じディレクトリーに存在すると想定します。

amqsputc コマンドの初回実行時には、IBM MQ メッセージは 1 番目のサーバーの MQIPT.LOCAL.QUEUE キューに置かれます。 2 回目の実行時には、メッセージは 2 番目のサーバーのキューに置かれるという具合です。 このセットアップを使用すると、**amqsgetc** コマンドで使用されるクライアント接続要求はリストにある次のキューに渡されるため、キューに置いたばかりのメッセージを **amqsgetc** コマンドで取り出

すことはできません。 ただし、**amqsputc** コマンドの 3 回の実行と、それに続く 3 回の **amqsgetc** コマンドの実行により、各メッセージは同じ順序で取り出されます。

別の IBM MQ クライアントを使用して、キュー・マネージャーに直接接続することによって (つまり、このサンプルの MQIPT を使用しない場合)、キュー・マネージャーのいずれかからメッセージを選択的に取り出すことができます。

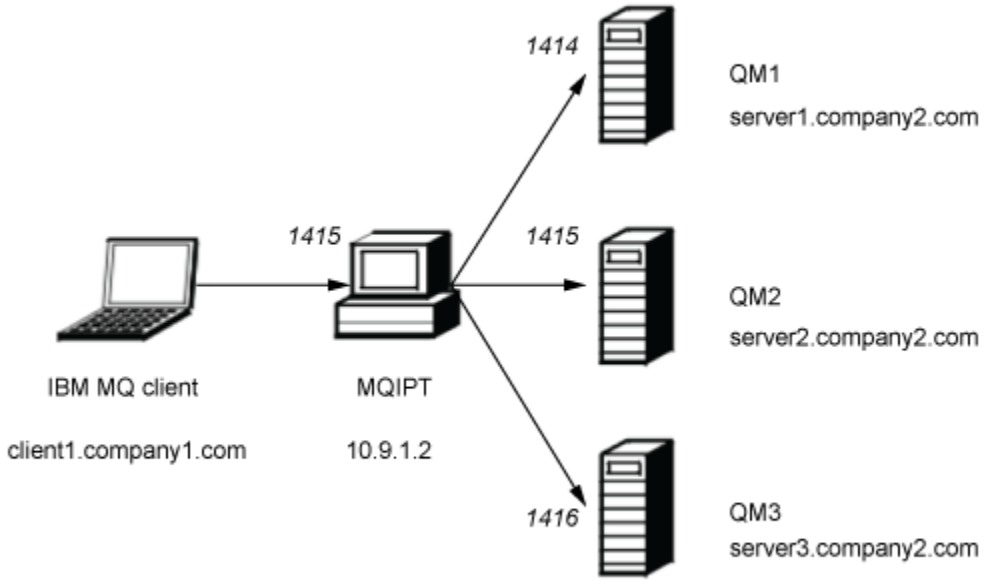


図 *44.* セキュリティー出口の経路指定のネットワーク図

この図では、IBM MQ クライアント (ポート 1415 の client1.company1.com) から MQIPT を通った 3 つの IBM MQ サーバー (server1.company2.com、server2.company2.com、および server3.company2.com) への接続フローを示しています。

手順

セキュリティー出口を使用して、3 つの異なる IBM MQ キュー・マネージャー・サーバーにクライアント接続要求を順次経路指定するには、以下の手順を実行します。

1. 3 つの別個のサーバーに MQIPT.QM1 という 3 つの同一のキュー・マネージャーを作成します。

   各キュー・マネージャーには、SVRCONN チャネル MQIPT.CONN.CHANNEL と空のローカル・キュー MQIPT.LOCAL.QUEUE があります。
2. MQIPT サーバーの場合:

   a) MQIPT というホーム・ディレクトリーに `exits` というディレクトリーを作成するため、コマンド・プロンプトで以下のコマンドを発行します。

   ```
   md C:\mqiptHome\exits
   ```

   b) `C:\mqiptHome\exits` というディレクトリー (ここで、`C:\mqiptHome` は `mqipt.conf` ファイルが位置するディレクトリーである) で、 3 つのキュー・マネージャーの名前を含む `SampleRoutingExit.conf` というサンプル構成ファイルを作成してください。

例えば、構成ファイルには以下のようなエントリーが含まれます。

```
server1.company2.com:1414
server2.company2.com:1415
server3.company2.com:1416
```

ファイルの最初の入力の前にブランク行がないことと、各入力が有効なサーバー名であることを確認します。 異なるサーバー名を使用している場合は、それらの名前を環境に一致するように変更します。

c) コマンド・プロンプトを開き、以下のコマンドを入力して出口をコンパイルします。 コンパイルされたサンプル出口が MQIPT に付属しているため、出口コードを変更していない場合は、これを行う必要はありません。

```
C:
cd \mqipt\samples\exits
javac -classpath C:\mqipt\lib\com.ibm.mq.ipt.jar;. SampleRoutingExit.java
```

d) 以下のコマンドを入力して、コンパイル済みの SampleRoutingExit.class という出口クラス・ファイル を C:\mqiptHome\exits というディレクトリーにコピーします。

```
copy C:\mqipt\samples\exits\SampleRoutingExit.class C:\mqiptHome\exits
```

e) mqipt.conf を編集して、ルート定義を追加します。

```
[route]
ListenerPort=1415
Destination=server1.company2.com
DestinationPort=1414
SecurityExit=true
SecurityExitPath=C:\mqiptHome\exits
SecurityExitName=SampleRoutingExit
```

SampleRoutingExit.conf をデフォルトの C:\mqiptHome\exits ディレクトリーに配置する場合は、 **SecurityExitPath** を設定する必要はありません。

f) MQIPT を開始します。

コマンド・プロンプトを開いて、以下のコマンドを入力します。

```
C:\mqipt\bin\mqipt C:\mqiptHome -n ipt1
```

ここで、C:\mqiptHome は MQIPT 構成ファイル、mqipt.conf の場所を示し、および ipt1 は MQIPT のインスタンスに指定される名前です。

以下のメッセージは、MQIPT が正常に開始したことを示します。

```
5724-H72 (C) Copyright IBM Corp. 2000, 2024. All Rights Reserved
MQCPI001 IBM MQ Internet Pass-Thru V9.2.0.0 starting
MQCPI004 Reading configuration information from mqipt.conf
MQCPI152 MQIPT name is ipt1
MQCPI021 Password checking has been enabled on the command port
MQCPI011 The path C:\mqiptHome\logs will be used to store the log files
MQCPI006 Route 1415 has started and will forward messages to :
MQCPI034 ....server1.company2.com(1414)
MQCPI035 ....using MQ protocol
MQCPI079 ....using security exit C:\mqiptHome\exits\SampleRoutingExit
MQCPI080 ......and timeout of 30 seconds
MQCPI078 Route 1415 ready for connection requests
```

3. IBM MQ クライアント・システムのコマンド・プロンプトで、以下のコマンドを入力します。

a) **MQSERVER** 環境変数を設定します。

```
SET MQSERVER=MQIPT.CONN.CHANNEL/TCP/10.9.1.2(1415)
```

b) 3 つのメッセージを書き込みます。

```
amqsputc MQIPT.LOCAL.QUEUE MQIPT.QM1
Hello world 1
amqsputc MQIPT.LOCAL.QUEUE MQIPT.QM1
```

```
Hello world 2
amqsputc MQIPT.LOCAL.QUEUE MQIPT.QM1
Hello world 3
```

各メッセージのストリングを入力した後、Enter キーを 2 回押します。

c) メッセージを読み取ります。

```
amqsgetc MQIPT.LOCAL.QUEUE MQIPT.QM1
amqsgetc MQIPT.LOCAL.QUEUE MQIPT.QM1
amqsgetc MQIPT.LOCAL.QUEUE MQIPT.QM1
```

メッセージ Hello world 1、 Hello world 2、および Hello world 3 が返されます。

クライアント接続要求の動的経路指定

このシナリオでは、使用されるチャネルの名前に基づいて、ターゲット・サーバーにクライアント接続要求を動的に経路指定できます。

始める前に

- このシナリオを開始する前に、167 ページの『IBM MQ Internet Pass-Thru 入門』でリストしている前提条件作業を必ず完了してください。
- Java 8.0 JDK をインストールします。
- Java bin サブディレクトリーを **PATH** 環境変数に追加します。

このタスクについて

チャネル名の最初の部分にキュー・マネージャーの名前を使用すれば、1 つの MQIPT 経路だけで、すべての接続要求を処理できます。 例えば、QM1 に接続するには、SVRCONN チャネルの名前は QM1.MQIPT.CHANNEL です。

このシナリオで使用されたサンプル出口は SampleOneRouteExit.java です。 インストール済みディレクトリーの samples/exits サブディレクトリー MQIPT の MQIPT が提供されます。

コンパイルされた出口クラス・ファイルの名前と場所は、MQIPT **SecurityExitName** および **SecurityExitPath** プロパティーで定義されます。

使用される全てのキュー・マネージャーとサーバー名は、 SampleOneRouteExit.conf という構成ファイルから読み取られます。 出口は、構成ファイルが出口クラス・ファイルと同じディレクトリーに存在すると想定します。

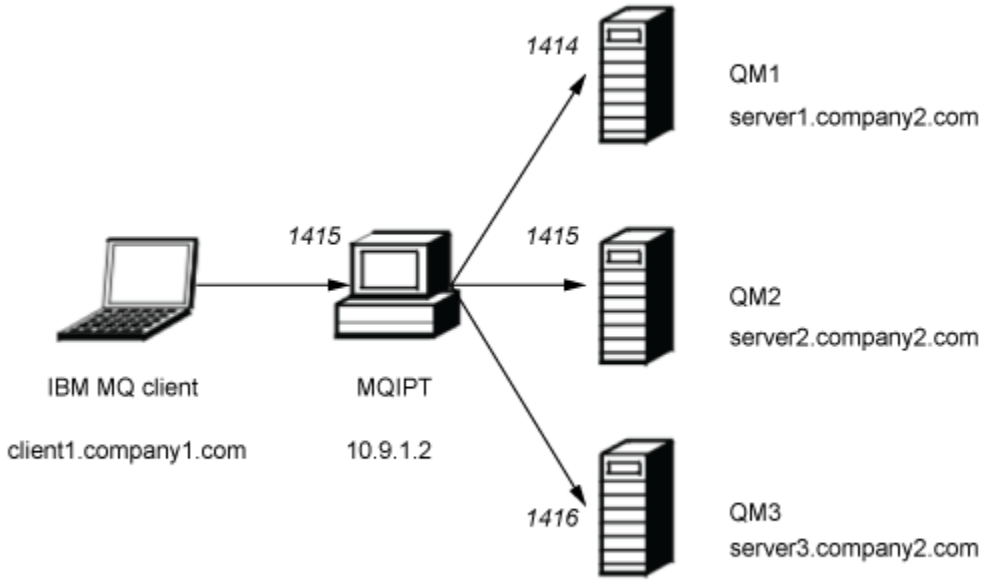


図 *45. 1* つの動的経路指定出口のネットワーク図

この図では、IBM MQ クライアント (ポート 1415 の client1.company1.com) から MQIPT を通った 3 つの IBM MQ サーバー (server1.company2.com、server2.company2.com、および server3.company2.com) への接続フローを示しています。

手順

クライアント接続要求を動的に経路指定するには、以下の手順を実行します。

1. 3 つの別個のサーバーに 3 つの異なるキュー・マネージャーを作成します。

   各キュー・マネージャーには、それぞれにちなんだ SVRCONN チャネル (例えば、キュー・マネージャー QM1 の QM1.MQIPT.CHANNEL) と空のローカル・キュー MQIPT.LOCAL.QUEUE があります。
2. MQIPT サーバーの場合:

   a) MQIPT というホーム・ディレクトリーに `exits` というディレクトリーを作成するため、コマンド・プロンプトで以下のコマンドを発行します。

   ```
   md C:\mqiptHome\exits
   ```

   b) `C\mqiptHome\exits` というディレクトリー (ここで、 `C:\mqiptHome` は `mqipt.conf` ファイルが位置するディレクトリーである) で、 3 つのキュー・マネージャーの名前を含む `SampleOneRouteExit.conf` というサンプル構成ファイルを作成してください。

   例えば、構成ファイルには以下のようなエントリーが含まれます。

   ```
   server1.company2.com:1414
   server2.company2.com:1415
   server3.company2.com:1416
   ```

   ファイルの最初の入力の前にブランク行がないことと、各入力が有効なサーバー名であることを確認します。 異なるサーバー名を使用している場合は、それらの名前を環境に一致するように変更します。

リスト内のすべてのキュー・マネージャー名は固有である必要があります。 同じ名前を複数回リストした場合は、キュー・マネージャーが異なるサーバーにある場合でも、その名前の最後の項目のみが登録されます。

c) コマンド・プロンプトを開き、以下のコマンドを入力して出口をコンパイルします。 コンパイルされたサンプル出口が MQIPT に付属しているため、出口コードを変更していない場合は、これを行う必要はありません。

```
C:
cd \mqipt\samples\exits
javac -classpath C:\mqipt\lib\com.ibm.mq.ipt.jar;. SampleOneRouteExit.java
```

d) 以下のコマンドを入力して、コンパイル済みの SampleOneRouteExit.class という出口クラス・ファイル を C:\mqiptHome\exits というディレクトリーにコピーします。

```
copy C:\mqipt\samples\exits\SampleOneRouteExit.class C:\mqiptHome\exits
```

e) mqipt.conf を編集して、次のルート定義を追加してください。

```
[route]
ListenerPort=1415
Destination=server1.company2.com
DestinationPort=1414
SecurityExit=true
SecurityExitName=SampleOneRouteExit
```

f) コマンド・プロンプトを開き、MQIPT を開始します。

```
C:\mqipt\bin\mqipt C:\mqiptHome -n ipt1
```

ここで、C:\mqiptHome は MQIPT 構成ファイル、mqipt.conf の場所を示し、および ipt2 は MQIPT のインスタンスに指定される名前です。

以下のメッセージは、MQIPT が正常に開始したことを示します。

```
5724-H72 (C) Copyright IBM Corp. 2000, 2024. All Rights Reserved
MQCPI001 IBM MQ Internet Pass-Thru V9.2.0.0 starting
MQCPI004 Reading configuration information from mqipt.conf
MQCPI152 MQIPT name is ipt2
MQCPI021 Password checking has been enabled on the command port
MQCPI011 The path C:\mqiptHome\logs will be used to store the log files
MQCPI006 Route 1415 has started and will forward messages to :
MQCPI034 ....server1.company2.com(1414)
MQCPI035 ....using MQ protocol
MQCPI079 ....using security exit C:\mqiptHome\exits\SampleOneRouteExit
MQCPI080 ......and timeout of 5 seconds
MQCPI078 Route 1415 ready for connection requests
```

3. IBM MQ クライアント・システムのコマンド・プロンプトで、以下のコマンドを入力します。

a) **MQSERVER** 環境変数を設定します。

```
SET MQSERVER=QM1.MQIPT.CHANNEL/TCP/10.9.1.2(1415)
```

b) メッセージを書き込みます。

```
amqsputc MQIPT.LOCAL.QUEUE QM1
Hello world 1
```

メッセージのストリングを入力した後、Enter キーを 2 回押します。

SVRCONN チャネル名が QM1 で始まっているため、メッセージは MQIPT によって QM1 に経路指定されます。

c) QM1 からメッセージを取得します。

```
amqsgetc MQIPT.LOCAL.QUEUE QM1
```

メッセージ Hello world 1 が戻ります。

d) **MQSERVER** 環境変数を再設定します。

```
SET MQSERVER=QM2.MQIPT.CHANNEL/TCP/10.9.1.2(1415)
```

e) メッセージを書き込みます。

```
amqsputc MQIPT.LOCAL.QUEUE QM2
Hello world 2
```

メッセージのストリングを入力した後、Enter キーを 2 回押します。

SVRCONN チャネル名が QM2 で始まっているため、メッセージは MQIPT によって QM2 に経路指定されます。

f) QM2 からメッセージを取得します。

```
amqsgetc MQIPT.LOCAL.QUEUE QM2
```

メッセージ `Hello world 2` が戻ります。

g) **MQSERVER** 環境変数をもう一度再設定します。

```
SET MQSERVER=QM3.MQIPT.CHANNEL/TCP/10.9.1.2(1415)
```

h) メッセージを書き込みます。

```
amqsputc MQIPT.LOCAL.QUEUE QM3
Hello world 3
```

メッセージのストリングを入力した後、Enter キーを 2 回押します。

SVRCONN チャネル名が QM3 で始まっているため、メッセージは MQIPT によって QM3 に経路指定されます。

i) QM3 からメッセージを取得します。

```
amqsgetc MQIPT.LOCAL.QUEUE QM3
```

メッセージ `Hello world 3` が戻ります。

TLS サーバーを認証するための証明書出口の使用

このシナリオでは、証明書出口を使用して TLS 接続を認証できます。

始める前に

- このシナリオを開始する前に、167 ページの『IBM MQ Internet Pass-Thru 入門』でリストしている前提条件作業を必ず完了してください。
- Java 8.0 JDK をインストールします。
- Java `bin` サブディレクトリーを **PATH** 環境変数に追加します。

このタスクについて

このシナリオでは、174 ページの『TLS サーバーの認証』のシナリオと同じ機能を、証明書出口を追加して実行します。

このシナリオで使用されたサンプル出口は `SampleCertificateExit.java` です。 インストール済みディレクトリーの `samples/exits` サブディレクトリー MQIPT の MQIPT が提供されます。

SSLExitData プロパティーの値を変更することによって、2 つの MQIPT サーバー間の TLS 接続を許可または拒否できます。

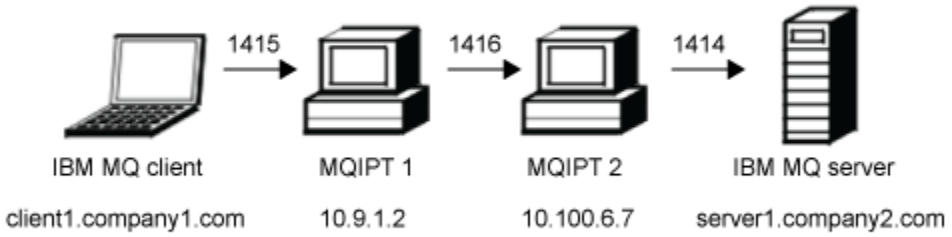


図 *46. SSL/TLS* サーバーのネットワーク図

この図では、IBM MQ クライアント (ポート 1415 の client1.company1.com) から MQIPT の 2 つのインスタンスを通った IBM MQ サーバー (ポート 1414 の server1.company2.com) への接続を示しています。

手順

TLS サーバーを認証するために証明書出口を使用するには、以下の手順を実行します。

1. MQIPT 1 システムの場合:

   a) MQIPT というホーム・ディレクトリーに `exits` というディレクトリーを作成するため、コマンド・プロンプトで以下のコマンドを発行します。

   ```
   md C:\mqiptHome\exits
   ```

   b) コマンド・プロンプトを開き、以下のコマンドを入力して出口をコンパイルします。 コンパイルされたサンプル出口が MQIPT に付属しているため、出口コードを変更していない場合は、これを行う必要はありません。

   ```
   C:
   cd \mqipt\samples\exits
   javac -classpath C:\mqipt\lib\com.ibm.mq.ipt.jar;. SampleCertificateExit.java
   ```

   c) 以下のコマンドを入力して、コンパイル済みの `SampleCertificateExit.class` という出口クラス・ファイル を `C:\mqiptHome\exits` というディレクトリーにコピーします。

   ```
   copy C:\mqipt\samples\exits\SampleCertificateExit.class C:\mqiptHome\exits
   ```

   d) `mqipt.conf` を編集して、次のルート定義を追加してください。

   ```
   [route]
   ListenerPort=1415
   Destination=9.100.6.7
   DestinationPort=1416
   SSLClient=true
   SSLClientKeyRing=C:\mqipt\samples\ssl\sslSample.pfx
   SSLClientKeyRingPW=<mqiptPW>1!PCaB1HWrFMOp43ngjwgArg==!6N/vsbqru7iqMhFN+wozxQ==
   SSLClientExit=true
   SSLExitName=SampleCertificateExit
   SSLExitPath=C:\mqiptHome\exits
   SSLExitData=allow
   ```

   e) コマンド・プロンプトを開き、MQIPT を開始します。

   ```
   C:\mqipt\bin\mqipt C:\mqiptHome -n ipt1
   ```

   ここで、`C:\mqiptHome` は MQIPT 構成ファイル、`mqipt.conf` の場所を示し、および `ipt1` は MQIPT のインスタンスに指定される名前です。

以下のメッセージは、MQIPT が正常に開始したことを示します。

```
5724-H72 (C) Copyright IBM Corp. 2000, 2024. All Rights Reserved
MQCPI001 IBM MQ Internet Pass-Thru V9.2.0.0 starting
MQCPI004 Reading configuration information from mqipt.conf
MQCPI152 MQIPT name is ipt1
MQCPI021 Password checking has been enabled on the command port
MQCPI011 The path C:\mqiptHome\logs will be used to store the log files
MQCPI006 Route 1415 has started and will forward messages to :
MQCPI034 ....9.100.6.7(1416)
MQCPI035 ....using MQ protocol
MQCPI036 ....SSL Client side enabled with properties :
MQCPI031 ......CipherSuites <null>
MQCPI032 ......keyring file C:\mqipt\samples\ssl\sslSample.pfx
MQCPI047 ......CA keyring file <null>
MQCPI038 ......peer certificate uses UID=*,CN=*,T=*,OU=*,DC=*,O=*,
                                              STREET=*,L=*,ST=*,PC=*,C=*,DNQ=*
MQCPI129 ......using certificate exit C:\mqiptHome\exits\SampleCertificateExit
MQCPI131 ......and certificate exit data 'allow'
MQCPI078 Route 1415 ready for connection requests
```

2. MQIPT 2 システムの場合:

   a) `mqipt.conf` を編集して、次のルート定義を追加してください。

```
[route]
ListenerPort=1416
Destination=Server1.company2.com
DestinationPort=1414
SSLServer=true
SSLServerKeyRing=C:\mqipt\samples\ssl\sslSample.pfx
SSLServerKeyRingPW=C:\mqipt\samples\ssl\sslSample.pwd
```

   b) コマンド・プロンプトを開き、MQIPT を開始します。

```
C:
cd \mqipt\bin
mqipt .. -n ipt2
```

   ここで、`..` は MQIPT 構成ファイル `mqipt.conf` が親ディレクトリーにあることを示し、`ipt2` は MQIPT のインスタンスに付けられる名前です。

   以下のメッセージは、MQIPT が正常に開始したことを示します。

```
5724-H72 (C) Copyright IBM Corp. 2000, 2024. All Rights Reserved
MQCPI001 IBM MQ Internet Pass-Thru V9.2.0.0 starting
MQCPI004 Reading configuration information from mqipt.conf
MQCPI152 MQIPT name is ipt2
MQCPI021 Password checking has been enabled on the command port
MQCPI011 The path C:\mqipt\logs will be used to store the log files
MQCPI006 Route 1416 has started and will forward messages to :
MQCPI034 ....server1.company2.com(1414)
MQCPI035 ....using MQ protocol
MQCPI037 ....SSL Server side enabled with properties :
MQCPI031 ......CipherSuites <null>
MQCPI032 ......key ring file C:\mqipt\samples\ssl\sslSample.pfx
MQCPI047 ......CA key ring file <null>
MQCPI038 ......peer certificate uses UID=*,CN=*,T=*,OU=*,DC=*,O=*,
                                              STREET=*,L=*,ST=*,PC=*,C=*,DNQ=*
MQCPI033 ......client authentication set to false
MQCPI078 Route 1416 ready for connection requests
```

3. IBM MQ クライアント・システムのコマンド・プロンプトで、以下のコマンドを入力します。

   a) **MQSERVER** 環境変数を設定します。

```
SET MQSERVER=MQIPT.CONN.CHANNEL/tcp/10.9.1.2(1415)
```

   b) メッセージを書き込みます。

```
amqsputc MQIPT.LOCAL.QUEUE MQIPT.QM1
Hello world
```

   メッセージのストリングを入力した後、Enter キーを 2 回押します。

c) メッセージを読み取ります。

```
amqsgetc MQIPT.LOCAL.QUEUE MQIPT.QM1
```

メッセージ「Hello world」が戻ります。

MQ Adv. V 9.3.3 MQ Adv. VUE Kafka Connect のシナリオ

IBM MQ および Apache Kafka がメッセージング範囲のさまざまな側面 (1 つは接続性に関するもの、もう 1 つはデータに関するもの) に特化しているため、ソリューションでは、多くの場合、2 つの間を流れるデータを必要とします。 これは、 Kafka Connect を使用して行うことができます。

Kafka Connect は、外部システムから Kafka クラスターに、または Kafka クラスターから外部システムにデータを移動するためのフレームワークを提供します。 これは、コネクターによって実現されます。

さまざまなタイプのコネクターが使用可能であり、 IBM には IBM MQ で使用するためのコネクターが用意されています。 コネクターには、以下の 2 つの異なるタイプがあります。

- ソース・コネクターは、外部システムから Kafka にデータを転送します。

  IBM MQ ソース・コネクターは、IBM MQ キューからメッセージをコンシュームし、それらをイベントとして Kafka トピックにパブリッシュします。
- シンク・コネクターは、 Kafka から外部システムにデータを転送します。

  IBM MQ シンク・コネクターは、 Kafka トピックからイベントを取り込み、それらをメッセージとして MQ キューに送信します。

詳しくは、 Kafka Connect およびコネクター を参照してください。

Kafka Connect のシナリオには、以下が含まれます。

- 接続バックボーンとして使用される IBM MQ を備えたコア・バンキング・システム。 IBM MQ 内を移動するメッセージのコピーを作成し、それらを分析のために Kafka にプッシュする必要がある。
- コア・バンキング・システムを拡張してデータを Kafka に出力したいが、バンキング・トランザクションが正常に完了したときにのみデータを Kafka に入力したいため、 IBM MQ をトランザクション・ブリッジとして使用します。
- Multiplatforms から z/OS にデータを取り込む必要があります。 マルチプラットフォーム開発チームには Kafka の経験があります。 z/OS チームは、 CICS / IMS との IBM MQ 統合を活用したいと考えています。

IBM MQ 9.3.3 以降、企業に IBM MQ Advanced for z/OS VUE ライセンス、 IBM MQ Advanced for Multiplatforms ライセンス、または IBM MQ Appliance ライセンスがある場合は、 IBM 提供のサポート対象ソース・コネクターおよびシンク・コネクターにアクセスできます。

V 9.3.4 LTS IBM MQ Advanced ライセンスをお持ちの場合、これは IBM MQ 9.3.4 または APAR PH56722 が適用された Long Term Support からも適用されます。

以前は、サポートなしでコネクターを無料で入手することも、 IBM Event Streams ライセンスを使用してサポートを受けることもできました。

注 :

1. これらの方法は、 Kafka の任意のバリアント (例えば、 Apache Kafka および IBM Event Streams) で使用できます。
2. サポートは、 Kafka Connect フレームワーク自体ではなく、2 つの IBM コネクターに対してのみ提供されます。

MQ Adv. V 9.3.3 MQ Adv. VUE Kafka Connect の一般的なトポロジー

このセクションでは、 IBM コネクターを介して IBM MQ を Kafka と統合する場合に使用できる 3 つの方法について説明します。

コネクターの取得について詳しくは、 218 ページの『コネクターの取得』 を参照してください。キュー・マネージャーの接続オプションおよび構成オプションについて詳しくは、 219 ページの『コネクターの使用』 を参照してください。

直接キュー (ソース)

IBM MQ を使用して Kafka にデータを送信するアプリケーションは、 IBM MQ ソース・コネクターが使用するキューにそれらのメッセージを送信できます。 その後、 IBM MQ ソース・コネクターはそれらのメッセージを取得し、関連する Kafka トピックに転送します。

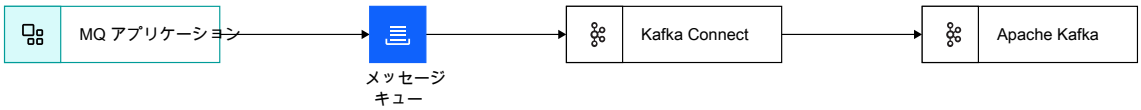


図 *47.* 直接キュー *(*ソース*)*

この方法は、アプリケーションが Kafka にデータを送信する必要があり、そのデータがまだ IBM MQ に送信されていない場合に使用する必要があります。

IBM MQ を使用してデータを送信するということは、メッセージの送信が、データベースなどの他の更新で調整されたトランザクション内で実行できることを意味します。 また、この方法により、 Kafka への存続期間が短くなる可能性がある接続をセットアップする必要がなくなり、代わりに IBM MQ への既存の接続を使用する必要がなくなります。

ストリーミング・キュー・コピー (ソース)

多くの場合、 IBM MQ を介して移動する既存のデータのコピーを取り、それを Kafka に送信する必要があります (例えば、分析のため)。 IBM MQ 9.3 以降、これはストリーミング・キューを使用して簡単に行うことができます。 ストリーミング・キューを使用すると、1 つのキューに書き込まれるメッセージを、1 つ目のキューを使用するアプリケーションに影響を与えることなく、キュー・マネージャーによって 2 つ目のキューにコピーすることができます。 詳しくは、30 ページの『ストリーミング・キュー』 を参照してください。

以下に例を示します。

```
DEF QL(TO.APP) STREAMQ(TO.KAFKA) STRMQOS(MUSTDUP)
DEF QL(TO.KAFKA)
```

メッセージが TO.APP に送信されるときに、そのメッセージのコピーを TO.KAFKA に送信する必要があることを意味します。 IBM MQ ソース・コネクターは、これらのメッセージを TO.KAFKA を使用して、関連する Kafka トピックに転送します。

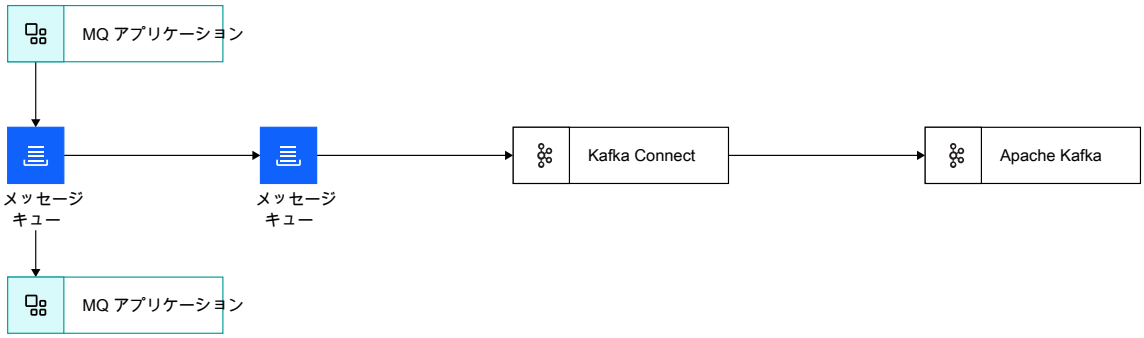


図 *48.* ストリーミング・キュー・コピー *(*ソース*)*

ストリーミング・キューを有効にしても、元のメッセージは変更されないため、既存のアプリケーションには影響しません。 2 番目のキューに送信されるメッセージは、同じペイロード、メッセージ ID、相関 ID などを持つ元のメッセージと同じです。

キューに送信 (シンク)

ソース・コネクターと同様に、シンク・コネクターは、 Kafka トピックからキューに直接データを受信するように構成できます。

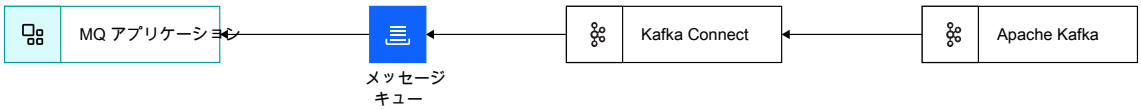


図 *49.* キューに送信 *(*シンク*)*

IBM MQ を介してデータを受信するということは、データベースなどの他の更新に合わせて調整されたトランザクション内でメッセージを受信できることを意味します。

また、この方法により、 Kafka への存続期間が短くなる可能性のある接続をセットアップする必要がなくなり、代わりに IBM MQ への既存の接続を使用する必要がなくなります。

コネクターの取得

IBM MQ に付属するコネクターのバージョンは、時間の経過とともに変化します。 IBM MQ 9.3.3 に同梱されているバージョン 1 コネクターは、最低 1 回のメッセージ配信を提供します。

V 9.3.4 IBM MQ 9.3.4 から出荷されるバージョン 2 コネクターは、少なくとも 1 回、正確に 1 回のメッセージ配信を提供します。

V 9.3.4 最低 1 回の配信と正確に 1 回の配信の違い、および正確に 1 回の配信を構成する方法について詳しくは、 220 ページの『1 回のみのサポート』を参照してください。

IBM MQ Advanced for z/OS Value Unit Edition、 V 9.3.4 LTS 、および IBM MQ Advanced for z/OS の IBM MQ 9.3.4 、または APAR PH56722 が適用された Long Term Support から、コネクターとそのサンプルは、z/OS UNIX System Services (USS) のコネクター・パック・コンポーネントの `kafka-connect` ディレクトリーに提供されています。

IBM MQ Advanced for Multiplatforms および IBM MQ Appliance では、これらのコネクターおよび必要な構成ファイルを取得するには、Fix Central にログインし、`V.R.M.F-IBM-MQ-Kafka-Connectors.tar.gz` を検索します (例: `9.3.3.0-IBM-MQ-Kafka-Connectors.tar.gz`)。

MQ Adv. MQ Adv. VUE MQ Adv. z/OS 各 IBM MQ バージョンには、以下のコネクターが付属しています。

IBM MQ のバージョン番号	IBM MQ for Multiplatforms tar ファイル名	ソース・コネクター・バージョン	シンク・コネクターのバージョン	正確に 1 回のデリバリー・サポート
V 9.3.3 9.3.3	9.3.3.0-IBM-MQ-Kafka_Connectors.tar.gz	1.3.2	1.5.0	いいえ
V 9.3.4 9.3.4	9.3.4.0-IBM-MQ-Kafka_Connectors.tar.gz	2.0	2.0	はい

GitHub からリリース・ページを通じてバージョン 1 コネクターを取得することもできます。

Source
Kafka-connect-mq-source

Sink
Kafka-connect-mq-シンク

これらのリンクから取得されるすべてのバージョンは、 IBM によってサポートされることに注意してください。

注 :

1. ただし、常に最新バージョンのコネクターを使用して、更新がないか定期的に確認する必要があります。 IBM MQ で提供されるコネクターは、製品の出荷時の最新バージョンであり、最新バージョンに定期的に更新されます。
2. IBM MQ Advanced for z/OS Value Unit Edition、 IBM MQ Advanced for Multiplatforms、または IBM MQ Appliance のライセンスによって IBM MQ コネクターのサポートが提供される場合、コネクターは、そのライセンスで実行されているキュー・マネージャーに接続する必要があります。
3. IBM MQ 9.3.3 より前のバージョンの IBM MQ Advanced for z/OS Value Unit Edition、 IBM MQ Advanced for Multiplatforms、または IBM MQ Appliance でも IBM MQ コネクターのライセンスが提供されていますが、前のテキストで参照されている GitHub のリリース・ページからコネクターをダウンロードする必要があります。

4. V 9.3.4 IBM MQ 9.3.4 より前の IBM MQ Advanced for z/OS の Continuous Delivery バージョンでも IBM MQ コネクターのライセンスが提供されていますが、前のテキストで参照されている GitHub のリリース・ページからコネクターをダウンロードする必要があります。

コネクターの使用

コネクターは、プロパティーまたは JSON ファイルのいずれかを使用して構成されます。 コネクターにはサンプル・ファイルが用意されています。

構成オプションの詳細と、コネクターのセットアップ方法については、以下を参照してください。

ソース・コネクター: IBM MQ の Kafka Connect ソース・コネクター
シンク・コネクター: Kafka Connect シンク・コネクター (IBM MQ 用)

V 9.3.4 ソース・コネクターで 1 回のみのサポートを有効にするには、 MQ ソース・コネクターの実行を参照してください。シンク・コネクターについては、 MQ シンク・コネクターの実行を参照してください。

V 9.3.4 最低 1 回の配信と正確に 1 回の配信の違い、および正確に 1 回の配信を構成する方法について詳しくは、 220 ページの『1 回のみのサポート』を参照してください。

Kafka Connect が IBM MQ Connectors を実行するには、そのクラスパスにコネクター jar と各種の IBM MQ jar ファイルが存在している必要があります。 以下の JAR ファイルが必要です。

`jms.jar`
`com.ibm.mq.allclient.jar`
`org.json.jar`
V 9.3.5 `bcpkix-jdk18on.jar` (IBM MQ 9.3.5 から)
`bcpkix-jdk15to18.jar` (IBM MQ 9.3.3 および IBM MQ 9.3.4)
V 9.3.5 `bcprov-jdk18on.jar` (IBM MQ 9.3.5 から)
`bcprov-jdk15to18.jar` (IBM MQ 9.3.3 および IBM MQ 9.3.4)
V 9.3.5 `bcutil-jdk18on.jar` (IBM MQ 9.3.5 から)
`bcutil-jdk15to18.jar` (IBM MQ 9.3.3 および IBM MQ 9.3.4)

以下に例を示します。

ソース・コネクター

V 9.3.5 IBM MQ 9.3.5 以降:

```
export CLASSPATH=$CLASSPATH:/path-to-kafka-jars/kafka-connect-mq-source-1.3.2.jar:
/path-to-mq-jars/jms.jar:/path-to-mq-jars/com.ibm.mq.allclient.jar:/path-to-mq-jars/org.json.jar:
/path-to-mq-jars/bcpkix-jdk18on.jar:/path-to-mq-jars/bcprov-jdk18on.jar:/path-to-mq-jars/bcutil-
jdk18on.jar
```

IBM MQ 9.3.3 および IBM MQ 9.3.4

```
export CLASSPATH=$CLASSPATH:/path-to-kafka-jars/kafka-connect-mq-source-1.3.2.jar:
/path-to-mq-jars/jms.jar:/path-to-mq-jars/com.ibm.mq.allclient.jar:/path-to-mq-jars/org.json.jar:
/path-to-mq-jars/bcpkix-jdk15to18.jar:/path-to-mq-jars/bcprov-jdk15to18.jar:/path-to-mq-jars/bcutil-
jdk15to18.jar
```

シンク・コネクター

V 9.3.5 IBM MQ 9.3.5 以降:

```
export CLASSPATH=$CLASSPATH:/path-to-kafka-jars/kafka-connect-mq-sink-1.5.0.jar:
/path-to-mq-jars/jms.jar:/path-to-mq-jars/com.ibm.mq.allclient.jar:/path-to-mq-jars/org.json.jar:
/path-to-mq-jars/bcpkix-jdk18on.jar:/path-to-mq-jars/bcprov-jdk18on.jar:/path-to-mq-jars/bcutil-
jdk18on.jar
```

IBM MQ 9.3.3 および IBM MQ 9.3.4

```
export CLASSPATH=$CLASSPATH:/path-to-kafka-jars/kafka-connect-mq-sink-1.5.0.jar:
/path-to-mq-jars/jms.jar:/path-to-mq-jars/com.ibm.mq.allclient.jar:/path-to-mq-jars/org.json.jar:
/path-to-mq-jars/bcpkix-jdk15to18.jar:/path-to-mq-jars/bcprov-jdk15to18.jar:/path-to-mq-jars/bcutil-
jdk15to18.jar
```

ここで、

`path-to-kafka-jars` は、 IBM MQ コネクターがインストールされている場所へのパスです。

`path-to-mq-jars` は、 IBM JMS クライアントがインストールされている場所へのパスです。

z/OS z/OS、`USS_ROOT/kafka-connect/source/kafka-connect-mq-source.jar`、および Connector Pack コンポーネントで実行されている場合は、ソース・コネクターの最新バージョンを指し、`USS_ROOT/kafka-connect/sink/kafka-connect-mq-sink.jar` はシンク・コネクターの最新バージョンを指します。

Kafka Connect、および IBM MQ コネクターは、 Java 仮想マシンを備えた任意のプラットフォームで実行できます。 キュー・マネージャーと同じプラットフォームで実行する必要も、接続先の Kafka クラスターで実行する必要もありません。

ただし、キュー・マネージャーと Kafka クラスターの間に長距離がある場合は、コネクターをキュー・マネージャーの比較的近く (理想的には同じアベイラビリティー・ゾーンまたはデータ・センター内) に配置する必要があります。

z/OS でのコネクターの使用

z/OS

コネクターは、 z/OS を含むすべてのプラットフォームで実行されるキュー・マネージャーで完全にサポートされます。 z/OS キュー・マネージャーへの接続は、サーバー接続チャネルまたはローカル・バインディングのいずれかを介して行うことができます。

IBM z/OS および IBM MQ for z/OS のパフォーマンス・テスト環境では、 z/OS のコネクターを z/OS UNIX System Services (USS) で実行し、ローカル・バインディングを使用してキュー・マネージャーに接続することで、最適なパフォーマンスが得られました。 これらの検出事項について詳しくは、 Kafka Connectors for IBM MQ -an MQ for z/OS perspective を参照してください。

USS on z/OS で Kafka Connect を実行するには、いくつかの追加のセットアップ手順が必要です。これらの手順については、 IBM z/OS でのコネクターの実行を参照してください。

V 9.3.4 MQ Adv. MQ Adv. VUE MQ Adv. z/OS 1 回のみのサポート

IBM MQ Kafka コネクターには、1 と 2 の 2 つのバージョンがあります。 バージョン 2 のコネクターは、「1 回のみ」および「1 回以上」のメッセージ配信のサポートを提供します。一方、バージョン 1 のコネクターは、「1 回以上」のメッセージ配信のサポートを提供します。

最低 1 回のメッセージ配信とは、IBM MQ、IBM MQ Kafka コネクター、または Kafka のいずれかで障害が発生した場合に、以下のようになることを意味します。

- ソース・コネクターの場合、 IBM MQ メッセージは失われませんが、 Kafka に複数回配信される可能性があるため、 Kafka メッセージが重複することがあります。
- シンク・コネクターの場合、 Kafka メッセージは失われませんが、 IBM MQ に複数回配信され、 IBM MQ メッセージが重複する可能性があります。

正確に 1 回のメッセージ配信とは、IBM MQ、IBM MQ Kafka コネクター、または Kafka のいずれかで障害が発生した場合に、以下のようになることを意味します。

- ソース・コネクターの場合、IBM MQ メッセージは失われず、重複する Kafka メッセージがあっても Kafka に配信されません。
- シンク・コネクターの場合、Kafka メッセージは失われず、IBM MQ メッセージが重複することなく IBM MQ に配信されます。

正確に 1 回のサポートは、 IBM MQ に付属するバージョン 2 コネクター、または IBM Event Streams にのみ提供されます。 バージョン 1 コネクターでは使用できません。

バージョン 2 コネクターは、最低 1 回または正確に 1 回のいずれかのモードで実行できます。 Kafka の適切な構成によって、および「状態キュー」を利用することによって、1 回のみのサポートが有効になります。 正確に 1 回のモードで実行されているコネクターの各インスタンスは、独自の状態キューを必要とします。

正確に 1 回のモードで実行されているコネクターのスループットとスケーラビリティーは、少なくとも 1 回のモードで実行されている場合よりも低くなります。 アプリケーションが重複メッセージを処理するように設計されていない場合にのみ、1 回限りのモードを有効にします。

ソース・コネクターでの「正確に 1 回」モードの構成について詳しくは、 MQ ソース・コネクターの実行 を参照してください。シンク・コネクターでの「正確に 1 回」モードの構成について詳しくは、 MQ シンク・コネクターの実行 を参照してください。

特記事項

本書は米国 IBM が提供する製品およびサービスについて作成したものです。

IBM 本書に記載の製品、サービス、または機能が日本においては提供されていない場合があります。 日本で利用可能な製品、サービス、および機能については、日本 IBM の営業担当員にお尋ねください。 本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、またはサービスのみが使用可能であることを意味するものではありません。 これらに代えて、IBM の知的所有権を侵害することのない、機能的に同等の製品、プログラム、またはサービスを使用することができます。 ただし、IBM 以外の製品とプログラムの操作またはサービスの 評価および検証は、お客様の責任で行っていただきます。

IBM は、本書に記載されている内容に関して特許権 (特許出願中のものを含む) を保有している場合があります。 本書の提供は、お客様にこれらの特許権について 実施権を許諾することを意味するものではありません。 実施権についてのお問い合わせは、書面にて下記宛先にお送りください。

〒 103-8510
東京都中央区日本橋箱崎町 19 番 21 号
日本アイ・ビー・エム株式会社
日本アイ・ビー・エム株式会社
法務・知的財産
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
〒 103-8510
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

以下の保証は、国または地域の法律に沿わない場合は、適用されません。 INTERNATIONAL BUSINESS MACHINES CORPORATION は、法律上の瑕疵担保責任、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。 "" 国または地域によっては、法律の強行規定により、保証責任の制限が 禁じられる場合、強行規定の制限を受けるものとします。

この情報には、技術的に不適切な記述や誤植を含む場合があります。 本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。 IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものでは ありません。 それらの Web サイトにある資料は、この IBM 製品の資料の一部ではありません。それらの Web サイトは、お客様の責任でご使用ください。

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム (本プログラムを含む) との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

東京都中央区日本橋箱崎町 19 番 21 号
日本アイ・ビー・エム株式会社
Software Interoperability Coordinator, Department 49XA
3605 Highway 52 N
Rochester, MN 55901
U.S.A.

本プログラムに関する上記の情報は、適切な使用条件の下で使用すること ができますが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他の ライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、 IBM より提供されます。

この文書に含まれるいかなるパフォーマンス・データも、管理環境下で 決定されたものです。 そのため、他の操作環境で得られた結果は、異なる可能性があります。 一部の測定が、開発レベルのシステムで行われた可能性がありますが、その測定値が、一般に利用可能なシステムのものと同じである保証はありません。 さらに、一部の測定値が、推定値である可能性があります。 実際の結果は、異なる可能性があります。 お客様は、お客様の特定の環境に適したデータを確かめる必要があります。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、 もしくはその他の公に利用可能なソースから入手したものです。 IBM は、それらの製品のテストは行っておりません。 したがって、他社製品に関する実行性、互換性、またはその他の要求については確証できません。 IBM 以外の製品の性能に関する質問は、それらの製品の供給者にお願いします。

IBM の将来の方向または意向に関する記述については、予告なしに変更または撤回される場合があり、単に目標を示しているものです。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。 より具体性を与えるために、それらの例には、個人、企業、ブランド、 あるいは製品などの名前が含まれている場合があります。これらの名称はすべて架空のものであり、 名称や住所が類似する企業が実在しているとしても、それは偶然にすぎません。

著作権使用許諾:

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。 お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。 このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。 従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性が あることをほのめかしたり、保証することはできません。

この情報をソフトコピーでご覧になっている場合は、写真やカラーの図表は表示されない場合があります。

プログラミング・インターフェース情報

プログラミング・インターフェース情報 (提供されている場合) は、このプログラムで使用するアプリケーション・ソフトウェアの作成を支援することを目的としています。

本書には、プログラムを作成するユーザーが WebSphere MQ のサービスを使用するためのプログラミング・インターフェースに関する情報が記載されています。

ただし、この情報には、診断、修正、および調整情報が含まれている場合があります。 診断、修正、調整情報は、お客様のアプリケーション・ソフトウェアのデバッグ支援のために提供されています。

重要 : この診断、修正、およびチューニング情報は、変更される可能性があるため、プログラミング・インターフェースとして使用しないでください。

商標

IBM、 IBM ロゴ、 ibm.com®は、世界の多くの国で登録された IBM Corporation の商標です。 現時点での IBM の商標リストについては、 "Copyright and trademark information"www.ibm.com/legal/copytrade.shtml をご覧ください。 他の製品名およびサービス名等は、それぞれ IBM または各社の商標である場合があります。

Microsoft および Windows は、Microsoft Corporation の米国およびその他の国における商標です。

UNIX は The Open Group の米国およびその他の国における登録商標です。

Linux は、Linus Torvalds 氏の米国およびその他の国における登録商標です。

この製品には、 Eclipse Project (https://www.eclipse.org/) により開発されたソフトウェアが含まれています。

Java およびすべての Java 関連の商標およびロゴは Oracle やその関連会社の米国およびその他の国における商標または登録商標です。

部品番号:

(1P) P/N: