デベロッパーガイド

Amazon MQ

Amazon MQ: デベロッパーガイド

Table of Contents

Amazon MQ とは

Amazon MQ は、メッセージブローカーのセットアップ、運用、保守を管理する、Apache ActiveMQ Classic および RabbitMQ 向けのマネージドメッセージブローカーサービスです。業界標準のメッセージングプロトコルを使用して新しい Amazon MQ ブローカーを作成することも、既存のメッセージブローカーからメッセージングコードを書き換えずに Amazon MQ に移行することもできます。

ブローカーは、Amazon MQ で実行されるメッセージブローカー環境です。これは、Amazon MQ の基本的な構成要素です。メッセージブローカーを使用すると、ソフトウェアアプリケーションおよびコンポーネントが、さまざまなプログラミング言語、オペレーティングシステム、正式なメッセージングプロトコルを使用して通信できます。Amazon MQ ブローカーは、大規模なクラウドネイティブアプリケーションとコンポーネント間の通信に使用できます。

トピック

- Amazon MQ の機能
- Amazon MQ の使用を開始するにはどうすればよいですか。
- Amazon MQ にフィードバックを提供するにはどうすればよいですか。

Amazon MQ の機能

マネージドメンテナンスとバージョンアップグレード

Amazon MQ は、スケジュールされたメンテナンスウィンドウ中にメッセージブローカーのメンテナンスとバージョンアップグレードを実行します。

CloudWatch によるブローカーのモニタリング

Amazon MQ は Amazon CloudWatch と統合されているため、ブローカーとキューのメトリクスを表示および分析できます。メトリクスの表示と分析は、Amazon MQ コンソール、CloudWatch コンソール、コマンドライン、および API から行うことができます。メトリクスは自動的に収集され、1 分おきに CloudWatch にプッシュされます。

セキュリティ

Amazon MQ は、保管中および転送中のメッセージの暗号化を提供します。ブローカーへの接続には SSL が使用され、アクセスは Amazon VPC 内のプライベートエンドポイントに制限できます。さ

らに、AWS Identity and Access Management (IAM) を使用して、IAM ユーザーとグループが特定の Amazon MQ ブローカーに対して実行できるアクションを制御することも可能です。

Amazon MQ での RabbitMQ のクォーラムキュー

クォーラムキューは、1 つのリーダーノード (プライマリレプリカ) と複数のフォロワーノード (その他のレプリカ) で構成されるレプリケーション型のキュータイプです。各ノードは別々のアベイラビリティーゾーンにあるため、1 つのノードが一時的に利用できなくなっても、メッセージ配信は別のアベイラビリティーゾーンにある新しく選出されたリーダーレプリカによって続行されます。クォーラムキューは、メッセージが失敗し、キューに何度も入れ直されたときに発生する有害メッセージを処理するために役立ちます。

Amazon MQ for ActiveMQ のクロスリージョンデータレプリケーション

クロスリージョンデータレプリケーション (CRDR) を使用すると、プライマリリージョンのプライマリブローカーからレプリカ AWS リージョンのレプリカブローカーへの非同期メッセージレプリケーションが可能になります。Amazon MQ API にフェイルオーバーリクエストを発行すると、現在のレプリカブローカーはプライマリブローカーのロールに昇格され、現在のプライマリブローカーはレプリカのロールに降格されます。

Amazon MQ の使用を開始するにはどうすればよいですか。

Amazon MQ で ActiveMQ の使用を開始するには、次のドキュメントを参照してください。

- 開始方法: ActiveMQ ブローカーの作成と接続
- the section called “ブローカーのデプロイ”
- ActiveMQ チュートリアル
- the section called “Amazon MQ for ActiveMQ のベストプラクティス”

Amazon MQ で RabbitMQ の使用を開始するには、次のドキュメントを参照してください。

- 開始方法: RabbitMQ ブローカーの作成と接続
- the section called “RabbitMQ ブローカーのデプロイ”
- the section called “RabbitMQ のチュートリアル”
- the section called “Amazon MQ for RabbitMQ のベストプラクティス”

Amazon MQ REST API については、Amazon MQ REST API リファレンスを参照してください。

Amazon MQ AWS CLI コマンドの詳細については、AWS CLI 「 コマンドリファレンス」のAmazon MQ」を参照してください。

Amazon MQ にフィードバックを提供するにはどうすればよいですか。

ドキュメントに関するフィードバックをお待ちしております。右側にある高評価アイコンと低評価アイコンを使用してフィードバックを送信するか、下にリンクされている「フィードバックを送信」フォームを使用できます。

Amazon MQ チームに連絡するには、Amazon MQ ディスカッションフォーラムを使用してください。

Amazon MQ のセットアップ

Amazon MQ を使用する前に、以下のステップを完了しておく必要があります。

トピック

ステップ 1: 前提条件

にサインアップする AWS アカウント

がない場合は AWS アカウント、次の手順を実行して作成します。

にサインアップするには AWS アカウント

1. https://portal.aws.amazon.com/billing/signup を開きます。
2. オンラインの手順に従います。

   サインアップ手順の一環として、電話またはテキストメッセージを受け取り、電話キーパッドで検証コードを入力します。

   にサインアップすると AWS アカウント、 AWS アカウントのルートユーザー が作成されます。ルートユーザーには、アカウントのすべての AWS のサービス とリソースへのアクセス権があります。セキュリティベストプラクティスとして、ユーザーに管理アクセス権を割り当て、ルートユーザーアクセスが必要なタスクの実行にはルートユーザーのみを使用するようにしてください。

AWS サインアッププロセスが完了すると、 から確認メールが送信されます。https://aws.amazon.com/ の [マイアカウント] をクリックして、いつでもアカウントの現在のアクティビティを表示し、アカウントを管理することができます。

管理アクセスを持つユーザーを作成する

にサインアップしたら AWS アカウント、日常的なタスクにルートユーザーを使用しないように AWS アカウントのルートユーザー、 のセキュリティを確保し AWS IAM Identity Center、 を有効にして管理ユーザーを作成します。

を保護する AWS アカウントのルートユーザー

1. ルートユーザーを選択し、 AWS アカウント E メールアドレスを入力して、アカウント所有者AWS Management Consoleとして にサインインします。次のページでパスワードを入力します。

   ルートユーザーを使用してサインインする方法については、AWS サインイン ユーザーガイドのルートユーザーとしてサインインするを参照してください。
2. ルートユーザーの多要素認証 (MFA) を有効にします。

   手順については、IAM ユーザーガイドの AWS アカウント 「ルートユーザー (コンソール) の仮想 MFA デバイスを有効にする」を参照してください。

管理アクセスを持つユーザーを作成する

1. IAM アイデンティティセンターを有効にします。

   手順については、「AWS IAM Identity Center ユーザーガイド」の「AWS IAM Identity Centerの有効化」を参照してください。
2. IAM アイデンティティセンターで、ユーザーに管理アクセスを付与します。

   を ID ソース IAM アイデンティティセンターディレクトリ として使用する方法のチュートリアルについては、AWS IAM Identity Center 「 ユーザーガイド」の「デフォルトを使用してユーザーアクセスを設定する IAM アイデンティティセンターディレクトリ」を参照してください。

管理アクセス権を持つユーザーとしてサインインする

- IAM アイデンティティセンターのユーザーとしてサインインするには、IAM アイデンティティセンターのユーザーの作成時に E メールアドレスに送信されたサインイン URL を使用します。

  IAM Identity Center ユーザーを使用してサインインする方法については、AWS サインイン 「 ユーザーガイド」の AWS 「 アクセスポータルにサインインする」を参照してください。

追加のユーザーにアクセス権を割り当てる

1. IAM アイデンティティセンターで、最小特権のアクセス許可を適用するというベストプラクティスに従ったアクセス許可セットを作成します。

   手順については、「AWS IAM Identity Center ユーザーガイド」の「権限設定を作成する」を参照してください。
2. グループにユーザーを割り当て、そのグループにシングルサインオンアクセス権を割り当てます。

   手順については、「AWS IAM Identity Center ユーザーガイド」の「グループの結合」を参照してください。

ステップ 2: ユーザーを作成して AWS 認証情報を取得する

ユーザーが の AWS 外部で を操作する場合は、プログラムによるアクセスが必要です AWS Management Console。プログラムによるアクセスを許可する方法は、 がアクセスするユーザーのタイプによって異なります AWS。

ユーザーにプログラマチックアクセス権を付与するには、以下のいずれかのオプションを選択します。

プログラマチックアクセス権を必要とするユーザー	目的	方法
ワークフォースアイデンティティ (IAM アイデンティティセンターで管理されているユーザー)	一時的な認証情報を使用して AWS CLI、、 AWS SDKs、または AWS APIs。	使用するインターフェイスの指示に従ってください。 • については AWS CLI、AWS Command Line Interface ユーザーガイドの「 を使用する AWS CLI ように AWS IAM Identity Centerを設定する」を参照してください。 • AWS SDKs、ツール、API については、 AWS APIs「 SDK およびツールリファ

プログラマチックアクセス権を必要とするユーザー	目的	方法
		レンスガイド」の「IAM アイデンティティセンター認証」を参照してください。 AWS SDKs
IAM	一時的な認証情報を使用して AWS CLI、、 AWS SDKs、または AWS APIs。	「IAM ユーザーガイド」の「 AWS リソースでの一時的な認証情報の使用」の手順に従います。
IAM	(非推奨) 長期認証情報を使用して、AWS CLI、 AWS SDKs、または AWS APIs。	使用するインターフェイスの指示に従ってください。 • については AWS CLI、「 AWS Command Line Interface ユーザーガイド」の「IAM ユーザー認証情報を使用した認証」を参照してください。 • AWS SDKs「 SDK とツールリファレンスガイド」の「長期認証情報を使用した認証」を参照してください。 AWS SDKs • API AWS APIs「IAM ユーザーガイド」の「IAM ユーザーのアクセスキーの管理」を参照してください。

ステップ 3: サンプルコードの使用準備を整える

以下のチュートリアルでは、 を使用して Amazon MQ ブローカーを操作する方法と AWS Management Console 、Amazon MQ for ActiveMQ および Amazon MQ for RabbitMQ ブローカーにプログラムで接続する方法を示します。ActiveMQ Java サンプルコードを使用するには、Java

Standard Edition Development Kit をインストールして、コードにいくつかの変更を行う必要があります。

Amazon MQ REST API と AWS SDKs を使用して、プログラムでブローカーを作成および管理することもできます。

次のステップ

Amazon MQ を使用する準備ができたので、ブローカーを作成することによって使用を開始します。ブローカーのエンジンタイプに応じて、Amazon MQ for ActiveMQ ブローカーに Java アプリケーションを接続するか、RabbitMQ Java クライアントライブラリを使用して Amazon MQ for RabbitMQ ブローカーに JVM ベースのアプリケーションを接続します。

開始方法: ActiveMQ ブローカーの作成と接続

ブローカーは、Amazon MQ で実行されるメッセージブローカー環境です。これは、Amazon MQ の基本的な構成要素です。ブローカーインスタンスクラス (m5、t3) とサイズ (large、micro) を組み合わせた説明は、ブローカーインスタンスタイプ (など) と呼ばれますmq.m5.large。詳細については、「Amazon MQ for ActiveMQ ブローカーとは」を参照してください。

ActiveMQ ブローカーを作成する

最初に実行する最も一般的な Amazon MQ タスクは、ブローカーの作成です。次の例は、 を使用して基本的なブローカー AWS Management Console を作成する方法を示しています。

1. Amazon MQ コンソールにサインインします。
2. [Select broker engine] (ブローカーエンジンの選択) ページで [Apache ActiveMQ] を選択します。
3. [Select deployment and storage] (デプロイとストレージタイプの選択) ページの [Deployment mode and storage type] (デプロイモードとストレージタイプ) セクションで、以下を実行します。

   a. [Deployment mode] (デプロイモード) を選択します ([Active/standby broker] (アクティブ/スタンバイブローカー)など)。詳細については、「Amazon MQ for ActiveMQ ブローカーのデプロイオプション」を参照してください。

      - 単一インスタンスブローカーは 1 つのアベイラビリティーゾーンにある 1 つのブローカーで構成されます。ブローカーは、アプリケーション、および Amazon EBS または Amazon EFS ストレージボリュームと通信します。詳細については、「オプション 1: Amazon MQ 単一インスタンスブローカー」を参照してください。
      - 高可用性対応のアクティブ/スタンバイブローカーは、2 つの異なるアベイラビリティーゾーンにある 2 つのブローカーで構成され、冗長ペアで設定されます。これらのブローカーは、アプリケーションおよび Amazon EFS と同期的に通信します。詳細については、「オプション 2: 高可用性対応の Amazon MQ アクティブ/スタンバイブローカー」を参照してください。

   b. [Storage type] (ストレージタイプ) を選択します (EBS など)。詳細については、「Storage」を参照してください。

> **Note**
>
> Amazon EBS は単一のアベイラビリティーゾーン内でデータをレプリケートし、ActiveMQ アクティブ/スタンバイデプロイモードをサポートしません。

c. [Next] (次へ) をクリックします。

4. [Configure settings] (設定の定義) ページの [Details] (詳細) セクションで、以下を実行します。

a. [Broker name] (ブローカー名) を入力します。

> **Important**
>
> 個人を特定できる情報 (PII) などの機密情報や秘匿性の高い情報はタグに追加しないでください。ブローカー名には、CloudWatch Logs を含む他の AWS のサービスからアクセスできます。ブローカー名は、プライベートデータや機密データとして使用することを意図していません。

> **Note**
>
> 追加設定セクションでは、以下を設定することもできます。
>
> - 設定
> - CloudWatch Logs
> - プライベートアクセス
> - ブローカーメンテナンスウィンドウ

b. [Broker instance type] (ブローカーインスタンスタイプ) を選択します (mq.m5.large など)。詳細については、「Broker instance types」を参照してください。

5. [ActiveMQ Web Console access] (ActiveMQ ウェブコンソールアクセス) セクションで、[Username] (ユーザーネーム) と [Password] (パスワード) を入力します。ブローカーのユーザー名とパスワードには、以下の制限が適用されます:

- ユーザーネームに使用できるのは、英数字、ダッシュ、ピリオド、アンダースコア、およびチルダ (- . _ ~) のみです。

- パスワードは 12 文字以上の長さで、一意の文字を少なくとも 4 つ含める必要があり、カンマ、コロン、または等号 (,:=) は使用できません。

> **Important**
>
> 個人を特定できる情報 (PII) などの機密情報や秘匿性の高い情報はブローカーのユーザー名に追加しないでください。ブローカーユーザー名は、CloudWatch Logs を含む他の AWS のサービスからアクセスできます。ブローカーのユーザー名は、プライベートデータや機密データとして使用することを意図していません。

6. [Deploy] (デプロイ) をクリックします。

   Amazon MQ がブローカーを作成している間は、[Creation in progress] (作成中) ステータスが表示されます。

   ブローカーの作成には約 15 分かかります。

   ブローカーが正常に作成されると、Amazon MQ が [Running] (実行中) ステータスを表示します。

7. [***MyBroker***] を選択します。

   [***MyBroker***] ページの [Connect (接続)] セクションにあるブローカーの ActiveMQ web console URLをメモしておきます。以下はその例です。

   ```
   https://b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-east-2.amazonaws.com:8162
   ```

   また、ブローカーのワイヤレベルプロトコルの [Endpoints] (エンドポイント) もメモしておきます。以下は、OpenWire エンドポイントの例です。

   ```
   ssl://b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-east-2.amazonaws.com:61617
   ```

開始方法: RabbitMQ ブローカーの作成と接続

ブローカーは、Amazon MQ で実行されるメッセージブローカー環境です。これは、Amazon MQ の基本的な構成要素です。ブローカーインスタンスクラス (m5、t3) とサイズ (large、micro) を組み合わせた説明は、ブローカーインスタンスタイプ (など) と呼ばれますmq.m5.large。詳細については、Amazon MQ for RabbitMQ ブローカーとはを参照してください。

RabbitMQ ブローカーを作成する

最初に実行する最も一般的な Amazon MQ タスクは、ブローカーの作成です。次の例は、 を使用して基本的なブローカー AWS Management Console を作成する方法を示しています。

ブローカーを作成したら、RabbitMQ ブローカー Amazon MQ を使用する際のパフォーマンスを最大化し、スループットコストを最小限に抑えるための推奨事項について、RabbitMQ のベストプラクティスを確認してください。 RabbitMQ Amazon MQ

1. Amazon MQ コンソールにサインインします。
2. [Select broker engine] (ブローカーエンジンの選択) ページで [RabbitMQ] を選択し、[Next] (次へ) をクリックします。
3. [Select deployment mode] (デプロイモードの選択) ページで [Deployment mode] (デプロイモード) ([Cluster deployment] (クラスターのデプロイ) など) を選択して、[Next] (次へ) をクリックします。

   - 単一インスタンスブローカーは、Network Load Balancer (NLB) の内側にある 1 つのアベイラビリティーゾーン内の 1 つのブローカーで構成されます。ブローカーは、アプリケーション、および Amazon EBS ストレージボリュームと通信します。詳細については、「オプション 1: Amazon MQ for RabbitMQ 単一インスタンスブローカー」を参照してください。
   - 高可用性対応の RabbitMQ クラスターデプロイは、Network Load Balancer の内側にある 3 つの RabbitMQ ブローカーノードの論理グループで、それぞれがユーザー、キュー、および複数のアベイラビリティーゾーン (AZ) 間の分散状態を共有します。詳細については、「オプション 2: Amazon MQ for RabbitMQ クラスターデプロイ」を参照してください。
4. [Configure settings] (設定の定義) ページの [Details] (詳細) セクションで、以下を実行します。

   a. [Broker name] (ブローカー名) を入力します。

> **Important**
>
> 個人を特定できる情報 (PII) などの機密情報や秘匿性の高い情報はタグに追加しないでください。ブローカー名には、CloudWatch Logs を含む他の AWS のサービスからアクセスできます。ブローカー名は、プライベートデータや機密データとして使用することを意図していません。

b. [Broker instance type] (ブローカーインスタンスタイプ) を選択します (mq.m5.large など)。詳細については、「Broker instance types」を参照してください。

5. [Configure settings] (設定の定義) ページの [RabbitMQ access] (RabbitMQ アクセス) セクションで、[Username] (ユーザーネーム) と [Password] (パスワード) を入力します。ブローカーのサインイン認証情報には以下の制限が適用されます。

   - ユーザーネームに使用できるのは、英数字、ダッシュ、ピリオド、およびアンダースコア (- . _) のみです。この値にチルダ (~) 文字を含めることはできません。Amazon MQ では、ユーザーネームとしての `guest` の使用が禁止されています。
   - パスワードは 12 文字以上の長さで、一意の文字を少なくとも 4 つ含める必要があり、カンマ、コロン、または等号 (,:=) は使用できません。

> **Important**
>
> 個人を特定できる情報 (PII) などの機密情報や秘匿性の高い情報はブローカーのユーザー名に追加しないでください。ブローカーユーザー名は、CloudWatch Logs を含む他の AWS のサービスからアクセスできます。ブローカーのユーザー名は、プライベートデータや機密データとして使用することを意図していません。

> **Note**
>
> 追加設定セクションでは、以下を設定することもできます。
>
> - 設定
> - CloudWatch Logs
> - プライベートアクセス
> - ブローカーメンテナンスウィンドウ

6. [次へ] を選択します。
7. [Review and create] (確認と作成) ページで、選択内容を確認し、必要に応じて編集することができます。
8. [Create broker] (ブローカーの作成) をクリックします。

   Amazon MQ がブローカーを作成している間は、[Creation in progress] (作成中) ステータスが表示されます。

   ブローカーの作成には約 15 分かかります。

   ブローカーが正常に作成されると、Amazon MQ が [Running] (実行中) ステータスを表示します。
9. [***MyBroker***] を選択します。

   [***MyBroker***] ページの [Connect] (接続) セクションにあるブローカーの RabbitMQ web console URL をメモしておきます。以下はその例です。

   ```
   https://b-c8349341-ec91-4a78-ad9c-a57f23f235bb.mq.us-west-2.on.aws
   ```

   ブローカーの secure-AMQP エンドポイントもメモしておきます。以下は、リスナーポート 5671 を公開する amqps エンドポイントの例です。

   ```
   amqps://b-c8349341-ec91-4a78-ad9c-a57f23f235bb.mq.us-west-2.on.aws:5671
   ```

Amazon MQ ブローカーの管理

ブローカーを作成したら、Amazon MQ ブローカーのさまざまなコンポーネントを管理および維持できます。

トピック

Amazon MQ への接続

AWS サービスエンドポイントとブローカーエンドポイントを使用して、他の のサービスから Amazon MQ に接続できます。

サービスエンドポイント

Amazon MQ サービス API には、次の接続方法が使用されます。

ドメイン	接続方法
`mq.region.amazonaws.com`	IPv4
`mq.region.api.aws`	デュアルスタック (IPv4 および IPv6)
`mq-fips.region.amazonaws.com`	IPv4 のみを使用する FIPS
`mq-fips.region.api.aws`	デュアルスタックの FIPS

ブローカーエンドポイント

Amazon MQ ブローカーには、次の接続方法が使用されます。

ドメイン	接続方法
brokerId.mq.*region*.amazonaws.com	IPv4
brokerId.mq.*region*.on.aws	デュアルスタック (IPv4 および IPv6) **Note** Amazon MQ for ActiveMQ ブローカーはデュアルスタックをサポートしていません。

デュアルスタック (IPv4 および IPv6) エンドポイントを使用して Amazon MQ に接続する

デュアルスタックエンドポイントは、IPv4 と IPv6 トラフィックの両方をサポートします。デュアルスタックエンドポイントにリクエストを行うと、エンドポイント URL は IPv4 または IPv6 アドレスに解決されます。デュアルスタックおよび FIPS エンドポイントの詳細については、SDK リファレンスガイドを参照してください。

Amazon MQ はリージョンのデュアルスタックエンドポイントをサポートしています。つまり、エンドポイント名の一部として AWS リージョンを指定する必要があります。デュアルスタックエンドポイント名は、命名規則 を使用しますmq.*region*.api.aws。例えば、eu-west-1 リージョンのデュアルスタックエンドポイント名は、mq.eu-west-1.api.aws です。

Amazon MQ エンドポイントの完全なリストについては、AWS 「 全般のリファレンス」を参照してください。

AWS PrivateLink を使用して Amazon MQ に接続する

IPv4 および IPv6 をサポートする Amazon MQ API 用の AWS PrivateLink エンドポイントは、トラフィックをパブリックインターネットに公開することなく、仮想プライベートクラウド (VPCs) と Amazon MQ API 間のプライベート接続を提供します。

> **Note**
>
> PrivateLink のサポートは、ブローカーエンドポイントではなく、Amazon MQ API エンドポイントでのみ使用できます。ブローカーエンドポイントへのプライベート接続の詳細については、「」を参照してくださいConfiguring a private Amazon MQ broker。

PrivateLink を使用して Amazon MQ API にアクセスするには、まず接続元の特定の VPC にインターフェイス VPC エンドポイントを作成する必要があります。VPC エンドポイントを作成するときは、FIPS エンドポイントcom.amazonaws.*region*.mq-fipsにサービス名com.amazonaws.*region*.mqまたは を使用します。

CLI または SDK を使用して Amazon MQ AWS を呼び出す場合は、デュアルスタックドメイン名 mq.*region*.api.awsまたは を使用するエンドポイント URL を指定する必要がありますmq-fips.*region*.api.aws。PrivateLink for Amazon MQ は、 で終わるデフォルトのドメイン名をサポートしていませんamazonaws.com。詳細については、 SDK リファレンスガイドの「デュアルスタックと FIPS エンドポイント」を参照してください。

次の CLI の例は、Amazon MQ VPC エンドポイントを介してアジアパシフィック (シドニー) リージョンdescribe-broker-engine-typeで を呼び出す方法を示しています。

```
AWS_USE_DUALSTACK=true aws mq describe-broker-engine-types --region ap-southeast-2
```

CLI でエンドポイントを設定するその他の方法については、「 CLI でのエンドポイントの使用」を参照してください AWS 。

VPC エンドポイントポリシーを使用して、VPC エンドポイントへのユーザーアクセスを決定することもできます。詳細については、「エンドポイントポリシーを使用して VPC エンドポイントへのアクセスを制御する」を参照してください。

Amazon MQ ブローカーエンジンバージョンのアップグレード

Amazon MQ は、サポートされているすべてのブローカーエンジンタイプに対して、新しいブローカーエンジンバージョンを定期的に提供します。新しいエンジンバージョンには、セキュリティパッチ、バグ修正、その他のブローカーエンジンの改善が含まれています。

Amazon MQ は、X.Y.Z 形式のセマンティックバージョニングに従ってバージョン番号を分類します。Amazon MQ の実装では、X はメジャーバージョンを示し、Y はマイナーバージョンを表し、Z はパッチバージョン番号を示します。アップグレードには以下の 2 つのタイプがあります。

- メジャーバージョンアップグレード – メジャーエンジンバージョン番号が変更されたときに行われます。例えば、バージョン 1.0 からバージョン 2.0 へのアップグレードは、メジャーバージョンアップグレードと見なされます。
- マイナーバージョンアップグレード – マイナーエンジンバージョン番号のみが変更されたときに行われます。例えば、バージョン 1.5 から 1.6 へのアップグレードは、マイナーバージョンアップグレードと見なされます。

ブローカーはいつでも、サポートされている次のメジャーバージョンまたはマイナーバージョンに手動でアップグレードできます。自動マイナーバージョンアップグレードを有効にすると、Amazon MQ により、サポートされている最新のパッチバージョンにブローカーがアップグレードされます。エンジンバージョン 3.13 以降を使用しているすべてのブローカーについて、Amazon MQ は、サポートされている最新のパッチバージョンへのアップグレードをメンテナンスウィンドウ内で管理します。Amazon MQ は、現在のマイナーバージョンがサポート終了に達すると、ブローカーを次のマイナーバージョンにアップグレードします。手動および自動のバージョンアップグレードは、どちらもスケジュールされたメンテナンスウィンドウ中、またはブローカーの再起動後に行われます。

以下のトピックでは、ブローカーエンジンバージョンを手動でアップグレードする方法と、自動マイナーバージョンアップグレードをアクティブにする方法について説明します。

トピック

エンジンバージョンの手動アップグレード

ブローカーのエンジンバージョンを新しいメジャーバージョンまたはマイナーバージョンに手動でアップグレードするには、 AWS Management Console、 AWS CLI、または Amazon MQ API を使用できます。

AWS Management Console

を使用してブローカーのエンジンバージョンをアップグレードするには AWS Management Console

1. Amazon MQ コンソールにサインインします。

2. 左のナビゲーションペインで、[Brokers] (ブローカー) をクリックしてから、アップグレードするブローカーをリストから選択します。
3. ブローカーの詳細ページで [Edit] (編集) をクリックします。
4. [Specifications] (仕様) の [Broker engine version] (ブローカーエンジンバージョン) で、ドロップダウンリストから新しいバージョン番号を選択します。
5. ページの最下部までスクロールして、[Schedule modifications] (変更をスケジュールする) をクリックします。
6. [Schedule broker modifications] (ブローカー変更のスケジュール) ページの [When to apply modifications] (変更を適用するタイミング) で以下のいずれかを選択します。

   - 次にスケジュールされたメンテナンスウィンドウ中に Amazon MQ でバージョンアップグレードを完了する場合は、[After the next reboot] (次回の再起動後) を選択します。
   - 直ちにブローカーを再起動してエンジンバージョンをアップグレードする場合は、[Immediately] (即時) を選択します。

   > **Important**
   >
   > 1 つのインスタンスブローカーは再起動中にオフラインになります。クラスターブローカーの場合、ブローカーの再起動中に一度にダウンするノードは 1 つだけです。

7. [Apply] (適用) をクリックして、変更の適用を終了します。

AWS CLI

を使用してブローカーのエンジンバージョンをアップグレードするには AWS CLI

1. 以下の例にあるように、update-broker CLI コマンドを使用して、以下のパラメータを指定します。

   - `--broker-id` – Amazon MQ がブローカー用に生成する一意の ID です。ID は、ブローカー ARN から解析できます。例えば、`arn:aws:mq:us-east-2:123456789012:broker:MyBroker:b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9` という ARN の場合、ブローカー ID は `b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9` になります。
   - `--engine-version` – ブローカーエンジンをアップグレードするエンジンバージョン番号です。

```
aws mq update-broker --broker-id broker-id --engine-version version-number
```

2. (オプション) エンジンバージョンを直ちにアップグレードする場合は、reboot-broker CLI コマンドを使用してブローカーを再起動します。

```
aws mq reboot-broker --broker-id broker-id
```

直ちにブローカーを再起動して変更を適用しない場合は、次にスケジュールされたメンテナンスウィンドウ中に Amazon MQ がブローカーをアップグレードします。

> **Important**
>
> 1 つのインスタンスブローカーは再起動中にオフラインになります。クラスターブローカーの場合、ブローカーの再起動中に一度にダウンするノードは 1 つだけです。

Amazon MQ API

Amazon MQ API を使用してブローカーのエンジンバージョンをアップグレードする

1. UpdateBroker API オペレーションを使用します。パスパラメータとして broker-id を指定します。以下の例は、ブローカーが us-west-2 リージョンにあることを前提としています。利用可能な Amazon MQ エンドポイントの詳細については、「AWS 全般のリファレンス」の「Amazon MQ エンドポイントとクォータ」を参照してください。

```
PUT /v1/brokers/broker-id HTTP/1.1
Host: mq.us-west-2.amazonaws.com
Date: Mon, 7 June 2021 12:00:00 GMT
x-amz-date: Mon, 7 June 2021 12:00:00 GMT
Authorization: authorization-string
```

リクエストペイロードで engineVersion を使用して、ブローカーをアップグレードするバージョン番号を指定します。

```
{
    "engineVersion": "engine-version-number"
}
```

2. (オプション) エンジンバージョンを直ちにアップグレードする場合は、RebootBroker API オペレーションを使用してブローカーを再起動します。パスパラメータとして broker-id が指定されます。

```
POST /v1/brokers/broker-id/reboot-broker HTTP/1.1
Host: mq.us-west-2.amazonaws.com
Date: Mon, 7 June 2021 12:00:00 GMT
x-amz-date: Mon, 7 June 2021 12:00:00 GMT
Authorization: authorization-string
```

直ちにブローカーを再起動して変更を適用しない場合は、次にスケジュールされたメンテナンスウィンドウ中に Amazon MQ がブローカーをアップグレードします。

> **Important**
>
> 1 つのインスタンスブローカーは再起動中にオフラインになります。クラスターブローカーの場合、ブローカーの再起動中に一度にダウンするノードは 1 つだけです。

マイナーエンジンバージョンの自動アップグレード

ブローカーの自動マイナーバージョンアップグレードを、初めてブローカーを作成するときにアクティブにするか、ブローカー設定を変更することによってアクティブにするかは、ユーザーが制御できます。既存のブローカーの自動マイナーバージョンアップグレードを有効にするには、、 AWS Management Console、 AWS CLIまたは Amazon MQ API を使用できます。

AWS Management Console

を使用して自動マイナーバージョンアップグレードを有効にするには AWS Management Console

1. Amazon MQ コンソールにサインインします。
2. 左のナビゲーションペインで、[Brokers] (ブローカー) をクリックしてから、アップグレードするブローカーをリストから選択します。
3. ブローカーの詳細ページで [Edit] (編集) をクリックします。
4. [Maintenance] (メンテナンス) で、[Enable automatic minor version upgrades](自動マイナーバージョンアップグレードの有効化) を選択します。

> **Note**
>
> このオプションが既に選択されている場合は、何も変更する必要はありません。

5. ページの最下部で [Save] (保存) をクリックします。

AWS CLI

経由で自動マイナーバージョンアップグレードを有効にするには AWS CLI、update-broker CLI コマンドを使用して、次のパラメータを指定します。

- `--broker-id` – Amazon MQ がブローカー用に生成する一意の ID です。ID は、ブローカー ARN から解析できます。例えば、`arn:aws:mq:us-east-2:123456789012:broker:MyBroker:b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9` という ARN の場合、ブローカー ID は `b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9` になります。
- `--auto-minor-version-upgrade` – 自動マイナーバージョンアップグレードオプションをアクティブにします。

```
aws mq update-broker --broker-id broker-id --auto-minor-version-upgrade
```

> **Note**
>
> ActiveMQ ブローカーの自動マイナーバージョンアップグレードを無効にする場合は、 `--no-auto-minor-version-upgrade`パラメータを使用します。

Amazon MQ API

Amazon MQ API を使用して自動マイナーバージョンアップグレードをアクティブにするには、UpdateBroker API オペレーションを使用します。パスパラメータとして `broker-id` を指定します。以下の例は、ブローカーが `us-west-2` リージョンにあることを前提としています。利用可能な Amazon MQ エンドポイントの詳細については、「AWS 全般のリファレンス」の「Amazon MQ エンドポイントとクォータ」を参照してください。

```
PUT /v1/brokers/broker-id HTTP/1.1
```

```
Host: mq.us-west-2.amazonaws.com
Date: Mon, 7 June 2021 12:00:00 GMT
x-amz-date: Mon, 7 June 2021 12:00:00 GMT
Authorization: authorization-string
```

リクエストペイロードで autoMinorVersionUpgrade プロパティを使用して、自動マイナーバージョンアップグレードをアクティブにします。

```
{
    "autoMinorVersionUpgrade": "true"
}
```

ブローカーの自動アマイナーバージョンップグレードを非アクティブにする場合は、リクエストペイロードで "autoMinorVersionUpgrade": "false" を設定します。

Amazon MQ ブローカーインスタンスタイプのアップグレード

ブローカーインスタンスクラス (m5、t3) とサイズ (large、micro) を組み合わせた説明は、ブローカーインスタンスタイプ (など) と呼ばれますmq.m5.large。インスタンスタイプを選択するときは、ブローカーのパフォーマンスに影響する以下の要因を考慮することが重要です。

- クライアントとキューの数
- 送信されるメッセージの量
- メモリに保持されるメッセージ
- 冗長メッセージ

小さいブローカーインスタンスタイプ (mq.t3.micro) は、アプリケーションのパフォーマンスをテストする場合にのみ使用することをお勧めします。本番稼働レベルのクライアントとキュー、高スループット、メモリ内のメッセージ、冗長メッセージには、大きいブローカーインスタンスタイプ (mq.m5.large 以上) が推奨されます。

パフォーマンスの問題が発生した場合、またはテストから本番環境に移行する場合は、より大きなインスタンスタイプ (から microへlarge) にアップグレードすることをお勧めします。インスタンスタイプをアップグレードするには、 AWS Management Console、 AWS CLI、または Amazon MQ API を使用できます。

AWS Management Console

を使用してより大きなインスタンスタイプにアップグレードするには AWS Management Console、次の手順を実行します。

1. Amazon MQ コンソールにサインインします。
2. 左のナビゲーションペインで、[Brokers] (ブローカー) をクリックしてから、アップグレードするブローカーをリストから選択します。
3. ブローカーの詳細ページで [Edit] (編集) をクリックします。
4. 仕様の「ブローカーインスタンスタイプ」で、ドロップダウンリストから新しいインスタンスタイプを選択します。
5. ページの下部で、スケジュールの変更を選択します。
6. [Schedule broker modifications] (ブローカー変更のスケジュール) ページの [When to apply modifications] (変更を適用するタイミング) で以下のいずれかを選択します。
   - Amazon MQ で次のスケジュールされたメンテナンスウィンドウ中にアップグレードを完了する場合は、次の再起動後に を選択します。
   - ブローカーを再起動し、インスタンスタイプをすぐにアップグレードする場合は、「即時」を選択します。

   > **Important**
   >
   > 1 つのインスタンスブローカーは再起動中にオフラインになります。クラスターブローカーの場合、ブローカーの再起動中に一度にダウンするノードは 1 つだけです。

7. [Apply] (適用) をクリックして、変更の適用を終了します。

AWS CLI

を使用してブローカーのインスタンスタイプをアップグレードするには AWS CLI

1. この例に示すように、modify-broker CLI コマンドを使用して、次のパラメータを指定します。
   - `--broker-id` – Amazon MQ がブローカー用に生成する一意の ID です。
   - `--host-instance-type` – ブローカーエンジンをアップグレードするエンジンバージョン番号です。

```
aws mq modify-broker --broker-id broker-id --host-instance-type instance-type
```

2. （オプション）インスタンスタイプをすぐにアップグレードする場合は、reboot-broker CLI コマンドを使用してブローカーを再起動します。

```
aws mq reboot-broker --broker-id broker-id
```

直ちにブローカーを再起動して変更を適用しない場合は、次にスケジュールされたメンテナンスウィンドウ中に Amazon MQ がブローカーをアップグレードします。

> **Important**
>
> 1 つのインスタンスブローカーは再起動中にオフラインになります。クラスターブローカーの場合、ブローカーの再起動中に一度にダウンするノードは 1 つだけです。

Amazon MQ API

Amazon MQ API を使用してブローカーのインスタンスタイプをアップグレードするには

1. ModifyBroker API オペレーションを使用します。パスパラメータとして broker-id を指定します。以下の例は、ブローカーが us-west-2 リージョンにあることを前提としています。使用可能な Amazon MQ エンドポイントの詳細については、のAmazon MQ エンドポイントとクォータ」を参照してくださいAWS 全般のリファレンス。

```
PUT /v1/brokers/broker-id HTTP/1.1
Host: mq.us-west-2.amazonaws.com
Date: Mon, 7 June 2021 12:00:00 GMT
x-amz-date: Mon, 7 June 2021 12:00:00 GMT
Authorization: authorization-string
```

リクエストペイロードhost-instance-typeで を使用して、ブローカーのアップグレード先のインスタンスタイプを指定します。

```
{
    "host-instance-type": "host-instance-type"
}
```

2. (オプション) エンジンバージョンを直ちにアップグレードする場合は、RebootBroker API オペレーションを使用してブローカーを再起動します。パスパラメータとして broker-id が指定されます。

```
POST /v1/brokers/broker-id/reboot-broker HTTP/1.1
Host: mq.us-west-2.amazonaws.com
Date: Mon, 7 June 2021 12:00:00 GMT
x-amz-date: Mon, 7 June 2021 12:00:00 GMT
Authorization: authorization-string
```

直ちにブローカーを再起動して変更を適用しない場合は、次にスケジュールされたメンテナンスウィンドウ中に Amazon MQ がブローカーをアップグレードします。

> **Important**
>
> 1 つのインスタンスブローカーは再起動中にオフラインになります。クラスターブローカーの場合、ブローカーの再起動中に一度にダウンするノードは 1 つだけです。

Amazon MQ for ActiveMQ ストレージタイプ

Amazon MQ for ActiveMQ は Amazon Elastic File System (EFS) と Amazon Elastic Block Store (EBS) をサポートしています。デフォルトで、ActiveMQ ブローカーはブローカーストレージに Amazon EFS を使用します。複数のアベイラビリティーゾーン全体で優れた耐障害性とレプリケーションを活用するには、Amazon EFS を使用します。低レイテンシーと高スループットを活用するには、Amazon EBS を使用します。

> **Important**
>
> - Amazon EBS を使用できるのは、mq.m5 ブローカーインスタンスタイプファミリーのみです。
> - ブローカーインスタンスタイプを変更することはできますが、ブローカーを作成した後でブローカーストレージタイプを変更することはできません。
> - Amazon EBS は単一のアベイラビリティーゾーン内でデータをレプリケートし、ActiveMQ アクティブ/スタンバイデプロイモードをサポートしません。

ストレージタイプ間の相違点

以下の表は、ActiveMQ ブローカー向けのインメモリ、Amazon EFS、および Amazon EBS の各ストレージタイプの違いを簡単にまとめたものです。

ストレージタイプ	永続的	ユースケースの例	プロデューサーあたり、1 秒あたりのエンキューされたメッセージの概算最大数 (1 KB のメッセージ)	レプリケーション
インメモリ	非永続的	• 株価情報 • 位置情報の更新 • 頻繁に変更されるデータ	5,000	なし
Amazon EBS	永続	• 大量のテキスト • 注文処理	500	1 つのアベイラビリティーゾーン (AZ) 内の複数のコピー
Amazon EFS	永続	金融取引	80	複数の AZ にまたがる複数のコピー

インメモリメッセージストレージは、レイテンシーが最も低く、最大のスループットを提供します。ただし、メッセージはインスタンスの置き換えまたはブローカーの再起動中に失われます。

Amazon EFS は、高い耐久性を備え、複数の AZ にまたがってレプリケートされるように設計されており、単一のコンポーネントの障害、または AZ の可用性に影響する問題が原因で発生するデータの損失を防ぎます。Amazon EBS はスループット用に最適化されており、単一の AZ 内にある複数のサーバー全体にレプリケートされます。

プライベート Amazon MQ ブローカーの設定

プライベートブローカーにはパブリックアクセシビリティがないため、VPC の外部からアクセスすることはできません。プライベートブローカーを設定する前に、VPCs、サブネット、セキュリティグループに関する次の情報を確認してください。

- VPC
  - ブローカーのサブネット (複数可) とセキュリティグループ (複数可) は、同じ VPC 内にある必要があります。
  - プライベートブローカーを使用している場合は、VPC で設定していない IP アドレスが表示されることがあります。これらは Amazon MQ インフラストラクチャの IP アドレスであり、アクションは必要ありません。
- サブネット
  - サブネットが共有 VPC 内にある場合、VPC はブローカーを作成する同じアカウントによって所有されている必要があります。
  - サブネットが指定されていない場合は、デフォルト VPC のデフォルトサブネットが使用されます。
  - ブローカーが作成されると、使用されるサブネットを変更することはできません。
  - クラスターおよびアクティブ/スタンバイブローカーの場合、サブネットは異なるアベイラビリティーゾーンにある必要があります。
  - 単一インスタンスブローカーの場合、使用するサブネットを指定し、ブローカーを同じアベイラビリティーゾーン内に作成できます。
- セキュリティグループ
  - セキュリティグループが指定されていない場合は、デフォルトの VPC のデフォルトのセキュリティグループが使用されます。
  - 単一インスタンス、クラスター、アクティブ/スタンバイブローカーには、少なくとも 1 つのセキュリティグループ (デフォルトのセキュリティグループなど) が必要です。

    > **Note**
    >
    > パブリック RabbitMQ ブローカーは、サブネットやセキュリティグループを使用しません。

  - ブローカーが作成されると、使用されるセキュリティグループは変更できません。セキュリティグループ自体は引き続き変更できます。

でのプライベートブローカーの設定 AWS Management Console

プライベートブローカーを設定するには、 で新しいブローカーの作成を開始します AWS Management Console。次に、ネットワーク設定セクションで、ブローカーの接続を設定するには、以下を実行します。

1. ブローカーのプライベートアクセスを選択します。プライベートブローカーに接続するには、IPv4, IPv6、またはデュアルスタック (IPv4 および IPv6) を使用できます。詳細については、「Connecting to Amazon MQ」を参照してください。
2. 次に、デフォルトの VPC、サブネット (複数可) 、セキュリティグループ (複数可) を使用するを選択するか、既存の VPC、サブネット (複数可) 、セキュリティグループ (複数可) を選択します。デフォルトまたは既存の VPC、サブネット (複数可) 、またはセキュリティグループ (複数可) を使用しない場合は、プライベートブローカーに接続する新しい VPC を作成する必要があります。

> **Note**
>
> プライベートブローカーアクセスの場合、接続方法はサブネットの選択した IP タイプと同じになります。ブローカーが作成されると、VPC エンドポイントは変更できず、常に選択したサブネットの IP タイプになります。新しい IP タイプを使用する場合は、新しいブローカーを作成する必要があります。

> **Note**
>
> Amazon MQ for ActiveMQ は VPC エンドポイントを使用しません。ActiveMQ ブローカーを初めて作成すると、Amazon MQ は VPC に Elastic Network Interface (ENI) をプロビジョニングします。セキュリティグループは ENI に配置され、パブリックブローカーとプライベートブローカーの両方に使用できます。

パブリックアクセシビリティのない Amazon MQ ブローカーのウェブコンソールへのアクセス

ブローカーのパブリックアクセシビリティをオフにすると、ブローカーを作成した AWS アカウント ID がプライベートブローカーにアクセスできます。ブローカーのパブリックアクセシビリティを無

効にしている場合、ブローカーのウェブコンソールにアクセスするには、以下の手順を実行する必要があります。

1. public-vpc に Linux EC2 インスタンスを作成します (必要に応じて、パブリック IP を使用)。
2. VPC が正しく設定されていることを確認するには、作成した EC2 インスタンスへの ssh 接続を確立し、ブローカーの URI を指定して curl コマンドを使用します。
3. お使いのマシンから、プライベートキーファイルのパスとパブリック EC2 インスタンスの IP アドレスを使用して、EC2 インスタンスへの ssh トンネルを作成します。以下はその例です。

   ```
   ssh -i ~/.ssh/id_rsa -N -C -q -f -D 8080 ec2-user@203.0.113.0
   ```

   転送プロキシサーバーがマシン上で開始されます。
4. プロキシクライアント (例: FoxyProxy) をマシン上にインストールします。
5. 以下の設定を使用して、プロキシクライアントを設定します。

   - プロキシタイプで、SOCKS5 を指定します。
   - IP アドレス、DNS 名、サーバー名で、localhost を指定します。
   - ポートで、8080 を指定します。
   - 既存の URL パターンをすべて削除します。
   - URL パターンで、*.mq.*.amazonaws.com* を指定します。
   - 接続タイプで、HTTP(S) を指定します。

   プロキシクライアントを有効にすると、マシン上のウェブコンソールにアクセスできます。

> **Important**
>
> プライベートブローカーを使用している場合は、VPC で設定していない IP アドレスが表示されることがあります。これらは Amazon MQ インフラストラクチャ上の RabbitMQ からの IP アドレスであり、対応は必要ありません。

Amazon MQ ブローカーのメンテナンスウィンドウのスケジュール

Amazon MQ は、ハードウェア、オペレーティングシステム、またはメッセージブローカーのエンジンソフトウェアに対して、メンテナンスウィンドウ内で定期的にメンテナンスを実行します。たとえ

ば、ブローカーインスタンスタイプを変更した場合、Amazon MQ は次のスケジュールされたメンテナンスウィンドウ中に変更を適用します。メンテナンスの所要時間は、メッセージブローカーにスケジュールされている操作に応じて最大 2 時間かかることがあります。複数のアベイラビリティーゾーン (AZ) にまたがる高可用性のブローカーデプロイモードを選択すると、メンテナンスウィンドウ中のダウンタイムを最小限に抑えることができます。

Amazon MQ for ActiveMQ には、高可用性を実現するアクティブ/スタンバイデプロイが用意されています。アクティブ/スタンバイモードでは、Amazon MQ はメンテナンス操作を一度に 1 インスタンスずつ実行するため、少なくとも 1 つのインスタンスは利用可能な状態に維持されます。さらに、メンテナンスウィンドウが 1 週間の異なる時点に設定されたブローカーのネットワークを構成することもできます。Amazon MQ for RabbitMQ には、高可用性を実現するクラスターデプロイが用意されています。クラスターデプロイでは、Amazon MQ はメンテナンス操作を一度に 1 ノードずつ実行して、少なくとも 2 つのノードが常に稼働している状態を維持します。

ブローカーを最初に作成するときは、メンテナンスウィンドウを週に 1 回、指定時刻に実行するようにスケジュールできます。ブローカーのメンテナンスウィンドウは、次にスケジュールされたメンテナンスウィンドウまで、最大 4 回しか調整できません。ブローカーのメンテナンスウィンドウが完了すると、Amazon MQ はこの制限をリセットし、次回のメンテナンスウィンドウの実行前に再びスケジュールを調整できるようになります。ブローカーの可用性は、ブローカーのメンテナンスウィンドウを調整しても影響を受けません。

ブローカーのメンテナンスウィンドウを調整するには、 AWS Management Console、 AWS CLI、または Amazon MQ API を使用できます。

を使用してブローカーメンテナンスウィンドウをスケジュールする AWS Management Console

を使用してブローカーメンテナンスウィンドウを調整するには AWS Management Console

1. Amazon MQ コンソールにサインインします。
2. 左のナビゲーションペインで、[Brokers] (ブローカー) をクリックしてから、アップグレードするブローカーをリストから選択します。
3. ブローカーの詳細ページで [Edit] (編集) をクリックします。
4. [Maintenance] (メンテナンス) で、以下を実行します。

    a. [Start day (開始日)] には、ドロップダウンリストから曜日を選択します ([Sunday (日曜日)] など)。

b. [Start time (開始時刻)] には、次回のブローカーメンテナンスウィンドウをスケジュールする時間と分を選択します (12:00 など)。

> **Note**
>
> [Start time] (開始時刻) オプションは、UTC+0 タイムゾーンで設定されます。

5. 次に、[変更のスケジュール] を選択します。[次回の再起動後] または [即時] を選択します。選択次の再起動後、ブローカーを再起動せずにメンテナンスウィンドウがすぐに更新されます。[即時] を選択すると、ブローカーがすぐに再起動されます。
6. ブローカーの詳細ページにある [Maintenance window (メンテナンスウィンドウ)] で、希望する新しいスケジュールが表示されていることを確認します。

を使用してブローカーメンテナンスウィンドウをスケジュールする AWS CLI

を使用してブローカーメンテナンスウィンドウを調整するには AWS CLI

1. 以下の例にあるように、update-broker CLI コマンドを使用して、以下のパラメータを指定します。

   - `--broker-id` – Amazon MQ がブローカー用に生成する一意の ID です。ID は、ブローカー ARN から解析できます。例えば、`arn:aws:mq:us-east-2:123456789012:broker:MyBroker:b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9` という ARN の場合、ブローカー ID は `b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9` になります。
   - `--maintenance-window-start-time` – 以下の構造で提供される、週次メンテナンスウィンドウの開始時刻を決定するパラメータです。
     - `DayOfWeek`– `MONDAY| TUESDAY | WEDNESDAY | THURSDAY | FRIDAY | SATURDAY | SUNDAY` の構文で指定する曜日です。
     - `TimeOfDay` – 24 時間形式の時刻です。
     - `TimeZone` – (オプション) 国/都市、または UTC オフセット形式のいずれかで指定するタイムゾーンです。デフォルトで UTC に設定されます。

```
aws mq update-broker --broker-id broker-id \
```

```
--maintenance-window-start-time DayOfWeek=SUNDAY,TimeOfDay=13:00,TimeZone=America/
Los_Angeles
```

2. (オプション) describe-broker CLI コマンドを使用して、メンテナンスウィンドウが正常に更新されたことを検証します。

```
aws mq describe-broker --broker-id broker-id
```

Amazon MQ API を使用したブローカーメンテナンスウィンドウのスケジュール

Amazon MQ API を使用してブローカーメンテナンスウィンドウを調整する

1. UpdateBroker API オペレーションを使用します。パスパラメータとして broker-id を指定します。以下の例は、ブローカーが us-west-2 リージョンにあることを前提としています。使用可能な Amazon MQ エンドポイントの詳細については、のAmazon MQ エンドポイントとクォータ」を参照してくださいAWS 全般のリファレンス。

```
PUT /v1/brokers/broker-id HTTP/1.1
Host: mq.us-west-2.amazonaws.com
Date: Wed, 7 July 2021 12:00:00 GMT
x-amz-date: Wed, 7 July 2021 12:00:00 GMT
Authorization: authorization-string
```

リクエストペイロードには、maintenanceWindowStartTime パラメータと WeeklyStartTime リソースタイプを使用します。

```
{
"maintenanceWindowStartTime": {
    "dayOfWeek": "SUNDAY",
    "timeZone": "America/Los_Angeles",
    "timeOfDay": "13:00"
  }
}
```

2. (オプション) DescribeBroker API オペレーションを使用して、メンテナンスウィンドウが正常に更新されたことを検証します。パスパラメータとして broker-id が指定されています。

```
GET /v1/brokers/broker-id HTTP/1.1
Host: mq.us-west-2.amazonaws.com
Date: Wed, 7 July 2021 12:00:00 GMT
```

```
x-amz-date: Wed, 7 July 2021 12:00:00 GMT
Authorization: authorization-string
```

Amazon MQ ブローカーの再起動

新しい設定をブローカーに適用するには、ブローカーを再起動します。

> **Note**
>
> ActiveMQ ブローカーが応答しない場合、ブローカーを再起動して障害状態から復旧できます。

以下の例では、 AWS Management Consoleを使用して Amazon MQ ブローカーを再起動する方法を説明します。

Amazon MQ ブローカーを再起動する

1. Amazon MQ コンソールにサインインします。
2. ブローカーのリストから、ブローカーの名前 (MyBroker など) を選択します。
3. [***MyBroker***] ページで、[Actions]、[Reboot broker] の順に選択します。

   > **Important**
   >
   > 再起動中、シングルインスタンスブローカーはオフラインになります。クラスターブローカーは利用できますが、各ノードは一度に 1 つずつ再起動されます。

4. [Reboot broker] ダイアログボックスで、[Reboot] を選択します。

   Rebooting a broker takes about 5 minutes.」が表示されます。再起動にインスタンスサイズの変更が含まれているか、キューの深さが大きいブローカーで再起動が実行される場合、再起動プロセスに時間がかかることがあります。

Amazon MQ ブローカーの削除

Amazon MQ ブローカーを使用しない (および近い将来使用しない) 場合は、Amazon MQ から削除して AWS コストを削減するのがベストプラクティスです。

以下の例では、 AWS Management Consoleを使用してブローカーを削除する方法を説明します。

Amazon MQ ブローカーの削除

1. Amazon MQ コンソールにサインインします。
2. ブローカーのリストからブローカー (MyBroker など) を選択して、[Delete] (削除) をクリックします。
3. [Delete ***MyBroker***?] (MyBroker を削除しますか?) ダイアログボックスで、`delete` と入力してから [Delete] (削除) をクリックします。

   ブローカーの削除には約 5 分かかります。

Amazon MQ ブローカーのステータス

ステータスによって、ブローカーの現在の状態が示されます。以下の表には、Amazon MQ ブローカーのステータスがリストされています。

コンソール	API	説明
作成に失敗	`CREATION_FAILED`	ブローカーを作成できませんでした。
作成を実行中	`CREATION_IN_PROGRESS`	ブローカーは現在作成中です。
削除を実行中	`DELETION_IN_PROGRESS`	ブローカーは現在削除中です。
再起動の進行中	`REBOOT_IN_PROGRESS`	ブローカーは現在再起動中です。
実行中	`RUNNING`	ブローカーが機能しています。
重要なアクションは不要	`CRITICAL_ACTION_REQUIRED`	ブローカーは実行中ですが、パフォーマンスが低下した状態にあり、即時の処置が必要です。問題を解決する手順

コンソール	API	説明
		については、トラブルシューティング のリストからアクション必須コードを選択します。

Amazon MQ リソースへのタグの追加

コスト割り当てのために Amazon MQ リソースを分類して識別するには、ブローカーまたは設定の目的を特定するメタデータタグを追加できます。これはブローカーが多数ある場合に特に便利です。コスト配分タグを使用して、独自のコスト構造を反映するように AWS 請求書を整理できます。これを行うには、サインアップして AWS アカウント請求書を取得し、タグキーと値を含めます。詳細については、AWS Billing ユーザーガイドの「月別コスト配分レポートの設定」を参照してください。

例えば、コストセンターと、Amazon MQ リソースの目的を表すタグを追加できます。

リソース	キー	値
Broker1	Cost Center	34567
	Stack	Production
Broker2	Cost Center	34567
	Stack	Production
Broker3	Cost Center	12345
	Stack	Development

このタグ付けスキームでは、同じコストセンター内で関連するタスクを実行している 2 つのブローカーをグループ化しながら、関連しないブローカーに異なるコスト割り当てタグを付けることができます。

Amazon MQ コンソールでのタグの追加

次の手順に従うと、Amazon MQ コンソールでリソースを作成しているときにすばやくタグを追加できます。

1. [ブローカーの作成] ページで、[追加設定] を選択します。
2. [タグ] で、[タグの追加] を選択します。
3. [キー] と [値] のペアを入力します。
4. (オプション) [タグの追加] を選択して、ブローカーに複数のタグを追加します。
5. [バケットを作成する] を選択します。

設定を作成するときにタグを追加するには。

1. [設定の作成] ページで、[アドバンスト] を選択します。
2. [タグ] を選択し、[設定の作成] ページで、[タグの追加] を選択します。
3. [キー] と [値] のペアを入力します。
4. (オプション) [タグの追加] を選択して、設定に複数のタグを追加します。
5. [起動設定の作成] を選択します。

タグを追加した後、Amazon MQ コンソールでリソースのタグを表示、編集、削除できます。REST API を使用してリソースのタグを確認することもできます。詳細については、Amazon MQ REST API リファレンスを参照してください。

Amazon MQ for ActiveMQ の使用

Amazon MQ は、ニーズに適したコンピューティングおよびストレージリソースを使用したメッセージブローカーの作成を容易にします。ブローカーは、、Amazon MQ REST API AWS Management Console、または を使用して作成、管理、削除できます AWS Command Line Interface。

Amazon MQ for ActiveMQ ブローカーは、単一インスタンスブローカーまたはアクティブ/スタンバイブローカーとしてデプロイできます。どちらのデプロイモードでも、Amazon MQ はデータを冗長的に保存することによって優れた耐久性を提供します。

> **Note**
>
> Amazon MQ は、データストアとして Apache KahaDB を使用します。JDBC および LevelDB などの他のデータストアはサポートされていません。

ブローカーには、ActiveMQ がサポートする任意のプログラミング言語を使用し、以下のプロトコルに対して TLS を明示的に有効にすることによってアクセスできます。

- AMQP
- MQTT
- MQTT over WebSocket
- OpenWire
- STOMP
- STOMP over WebSocket

Amazon MQ REST API については、Amazon MQ REST API リファレンスを参照してください。

Amazon MQ for ActiveMQ ブローカー

Amazon MQ for ActiveMQ ブローカーとは

ブローカーは、Amazon MQ で実行されるメッセージブローカー環境です。これは、Amazon MQ の基本的な構成要素です。ブローカーインスタンスクラス (`m5`、`t3`) とサイズ (`large`、`micro`) を組み合わせた説明は、ブローカーインスタンスタイプ (など) と呼ばれます`mq.m5.large`。詳細については、「Broker instance types」を参照してください。

- 単一インスタンスブローカーは、1 つのアベイラビリティーゾーン内の 1 つのブローカーで構成されます。ブローカーは、アプリケーション、および Amazon EBS または Amazon EFS ストレージボリュームと通信します。
- アクティブ/スタンバイブローカーは、2 つの異なるアベイラビリティーゾーンにある 2 つのブローカーで構成され、冗長ペアで設定されます。これらのブローカーは、アプリケーションおよび Amazon EFS と同期的に通信します。

詳細については、「Amazon MQ for ActiveMQ ブローカーのデプロイオプション」を参照してください。

自動マイナーバージョンアップグレードを有効にして、Apache から新しいバージョンがリリースされるたびに、ブローカーエンジンの新しいマイナーバージョンにアップグレードできます。自動アップグレードは、曜日、時刻 (24 時間形式)、およびタイムゾーン (デフォルトは UTC) で定義されたメンテナンスウィンドウ中に行われます。

ブローカーを作成および管理する方法については、以下を参照してください。

- 開始方法: ActiveMQ ブローカーの作成と接続
- ブローカー
- Broker statuses

サポートされているワイヤレベルプロトコル

ブローカーには、ActiveMQ がサポートする任意のプログラミング言語を使用し、以下のプロトコルに対して TLS を明示的に有効にすることによってアクセスできます。

- AMQP
- MQTT
- MQTT over WebSocket
- OpenWire
- STOMP
- STOMP over WebSocket

属性

ActiveMQ ブローカー設定にはいくつかの属性があります。以下はその例です。

- 名前 (MyBroker)
- ID (b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9)
- Amazon リソースネーム (ARN) (arn:aws:mq:us-east-2:123456789012:broker:MyBroker:b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9)
- ActiveMQ ウェブコンソール URL (https://b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-east-2.amazonaws.com:8162)

  詳細については、Apache ActiveMQ ドキュメントの「Web Console」を参照してください。

  > **Important**
  >
  > activemq-webconsole グループが含まれない認可マップを指定する場合、Amazon MQ ブローカーにメッセージを送信する権限、またはブローカーからメッセージを受信する権限がグループにないことから、ActiveMQ ウェブコンソールは使用できません。

- ワイヤレベルプロトコルのエンドポイント:
  - amqp+ssl://b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-east-2.amazonaws.com:5671
  - mqtt+ssl://b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-east-2.amazonaws.com:8883
  - ssl://b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-east-2.amazonaws.com:61617

    > **Note**
    >
    > これは OpenWire エンドポイントです。

  - stomp+ssl://b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-east-2.amazonaws.com:61614
  - wss://b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-east-2.amazonaws.com:61619

  詳細については、Apache ActiveMQ ドキュメントの「Configuring Transports」を参照してください。

> **Note**
>
> アクティブ/スタンバイブローカーの場合、Amazon MQ は 2 つの ActiveMQ ウェブコンソール URL を提供しますが、一度に 1 つの URL しかアクティブになりません。同様に、Amazon MQ はワイヤレベルプロトコルごとに 2 つのエンドポイントを提供しますが、ペアごとに一度に 1 つのエンドポイントしかアクティブになりません。`-1` および `-2` サフィックスは冗長ペアを表します。

ブローカー属性の完全なリストについては、Amazon MQ REST API リファレンスで以下を参照してください。

- REST オペレーション ID: ブローカー
- REST オペレーション ID: ブローカー
- REST オペレーション ID: ブローカーの再起動

ブローカーユーザー

ActiveMQ ユーザーとは、ActiveMQ ブローカーのキューとトピックにアクセスできる人物またはアプリケーションです。ユーザーは、特定の許可を持つように設定できます。例えば、一部のユーザーに ActiveMQ ウェブコンソールへのアクセスを許可することができます。

グループはセマンティックラベルです。グループをユーザーに割り当てて、グループが特定のキューとトピックに対する送信、受信、管理を行うための許可を設定できます。

> **Important**
>
> ユーザーを変更しても、その変更はユーザーに直ちに適用されません。変更を適用するには、次のメンテナンスウィンドウまで待機するか、ブローカーを再起動する必要があります。

ユーザーとグループの詳細については、Apache ActiveMQ ドキュメントの以下の項目を参照してください。

- Authorization
- 認可の例

ActiveMQ ユーザーを作成、編集および削除する方法については、以下を参照してください。

- ActiveMQ ブローカーユーザーの作成
- [ユーザー]

ユーザー属性

ユーザー属性の完全なリストについては、Amazon MQ REST API リファレンスで以下を参照してください。

- REST オペレーション ID: ユーザー
- REST オペレーション ID: ユーザー

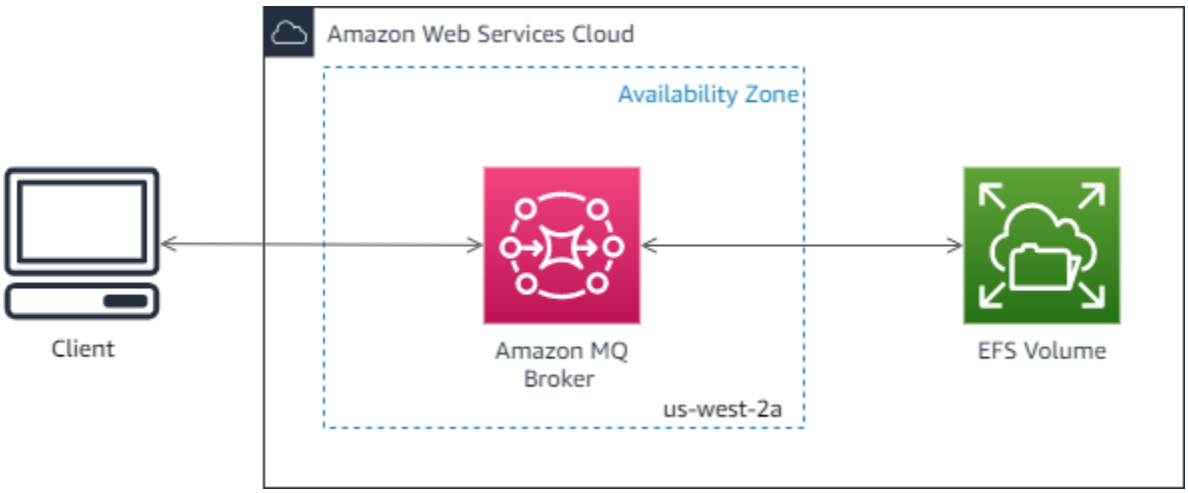
Amazon MQ for ActiveMQ ブローカーのデプロイオプション

Amazon MQ では、ブローカーのデプロイオプションとして、単一インスタンスのデプロイとクラスターデプロイが用意されています。

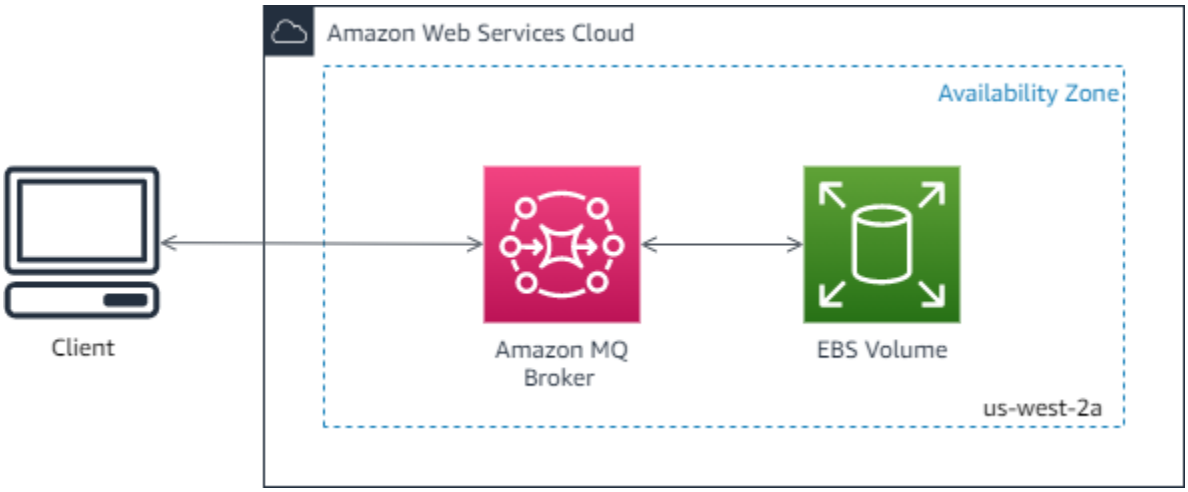
オプション 1: Amazon MQ 単一インスタンスブローカー

単一インスタンスブローカーは、1 つのアベイラビリティーゾーン内の 1 つのブローカーで構成されます。ブローカーは、アプリケーション、および Amazon EBS または Amazon EFS ストレージボリュームと通信します。Amazon EFS ストレージボリュームは、複数のアベイラビリティーゾーン (AZ) にまたがってデータを冗長的に保存することにより、最高レベルの耐久性と可用性を実現するように設計されています。Amazon EBS は、低レイテンシーと高スループット向けに最適化されたブロックレベルのストレージを提供します。ストレージオプションの詳細については、「Storage」を参照してください。

以下の図は、複数の AZ にまたがってレプリケートされている Amazon EFS ストレージを備えた単一インスタンスブローカーを図示しています。



以下の図は、単一の AZ 内にある複数のサーバーにまたがってレプリケートされている Amazon EBS ストレージを備えた単一インスタンスブローカーを図示しています。



オプション 2: 高可用性対応の Amazon MQ アクティブ/スタンバイブローカー

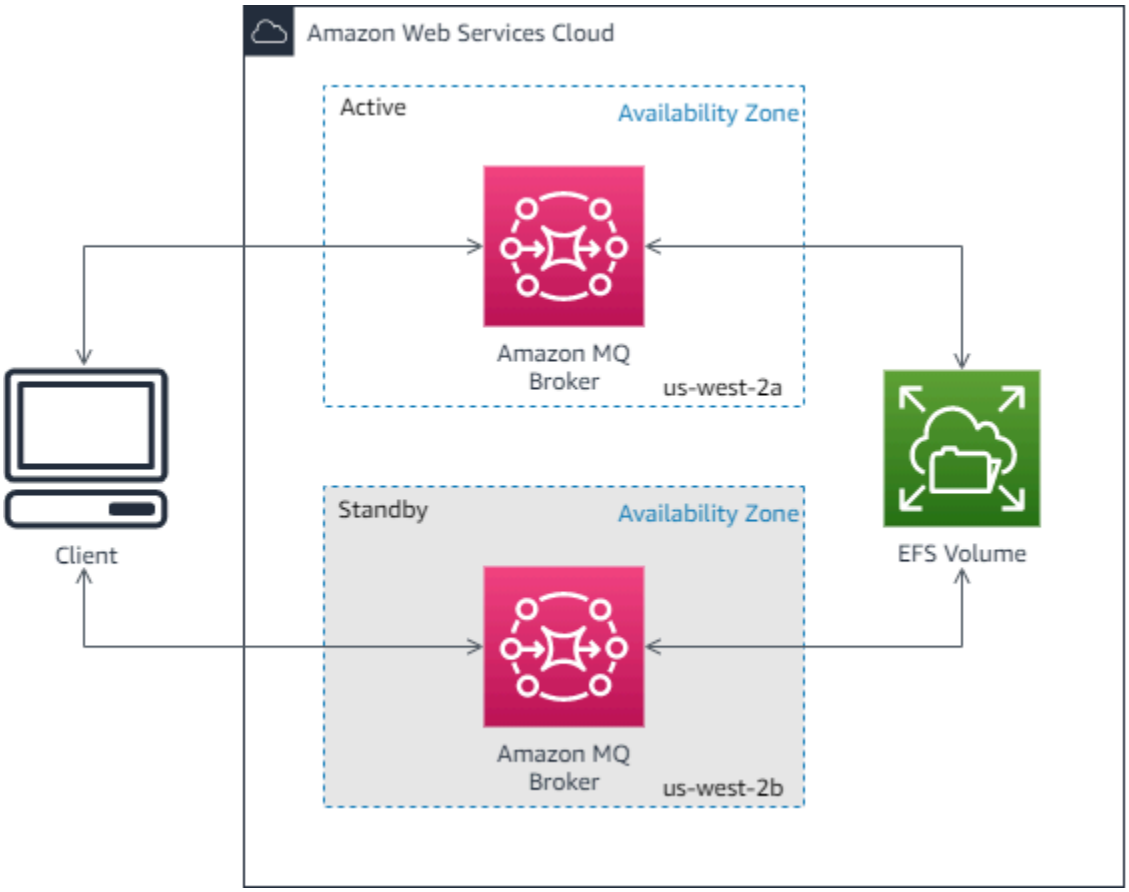
アクティブ/スタンバイブローカーは、2 つの異なるアベイラビリティーゾーンにある 2 つのブローカーで構成され、冗長ペアで設定されます。これらのブローカーは、アプリケーションおよび Amazon EFS と同期的に通信します。Amazon EFS ストレージボリュームは、複数のアベイラビリティーゾーン (AZ) にまたがってデータを冗長的に保存することにより、最高レベルの耐久性と可用性を実現するように設計されています。詳細については、「Storage」を参照してください。

通常、1 つのブローカーインスタンスのみが常時アクティブであり、他のブローカーインスタンスはスタンバイです。ブローカーインスタンスのいずれかが正常に機能しない、またはメンテナンスが行われる場合、Amazon MQ が非アクティブインスタンスを使用停止状態にするまでしばらく時間がか

かります。その間に、正常なスタンバイインスタンスがアクティブになり、着信通信の受け入れを開始できるようになります。メンテナンスウィンドウとブローカーが再起動すると、フェイルオーバーが発生します。ブローカーを再起動する場合、フェイルオーバーには数秒しかかかりません。

アクティブ/スタンバイブローカーの場合、Amazon MQ は 2 つの ActiveMQ ウェブコンソール URL を提供しますが、一度に 1 つの URL しかアクティブになりません。同様に、Amazon MQ はワイヤレベルプロトコルごとに 2 つのエンドポイントを提供しますが、ペアごとに一度に 1 つのエンドポイントしかアクティブになりません。-1 および -2 サフィックスは冗長ペアを表します。ワイヤレベルのプロトコルエンドポイントの場合、フェイルオーバートランスポートを使用してアプリケーションがいずれかのエンドポイントに接続できるようにする必要があります。

以下の図は、複数の AZ にまたがってレプリケートされている Amazon EFS ストレージを備えたアクティブ/スタンバイブローカーを図示しています。



Amazon MQ のブローカーのネットワーク

Amazon MQ は ActiveMQ のブローカーのネットワーク機能をサポートしています。

ブローカーのネットワークは、同時にアクティブな複数の単一インスタンスブローカー、またはアクティブ/スタンバイブローカーで構成されています。ブローカーのネットワークを作成すると、複数のブローカーインスタンスで可用性、耐障害性、負荷分散を向上させることができます。

ブローカーのネットワークはどのように機能しますか？

ブローカーのネットワークは、ネットワークコネクタを使用してブローカーを別のブローカーに接続することで確立されます。ネットワークコネクタは、あるブローカーから別のブローカーへのオンデマンドメッセージを提供します。ネットワークコネクタは、ブローカー設定で非二重接続または二重接続として設定されます。非二重接続の場合、メッセージは一方のブローカーから他方のブローカーにのみ転送されます。二重接続の場合、メッセージは両方のブローカー間で双方向に転送されます。

ネットワークコネクタが二重として設定されている場合、メッセージは Broker2 から Broker1 にも転送されます。例えば、ブローカー設定の二重 networkConnector エントリを次に示します。

ブローカーのネットワークでは、非二重接続と二重接続の両方を使用できます。トラフィックを改善したり、制限の引き上げを回避したりするために、別のブローカーに二重接続を導入することもできます。二重接続は、オンプレミスから Amazon MQ マネージドブローカーへの部分的な移行にも役立ちます。

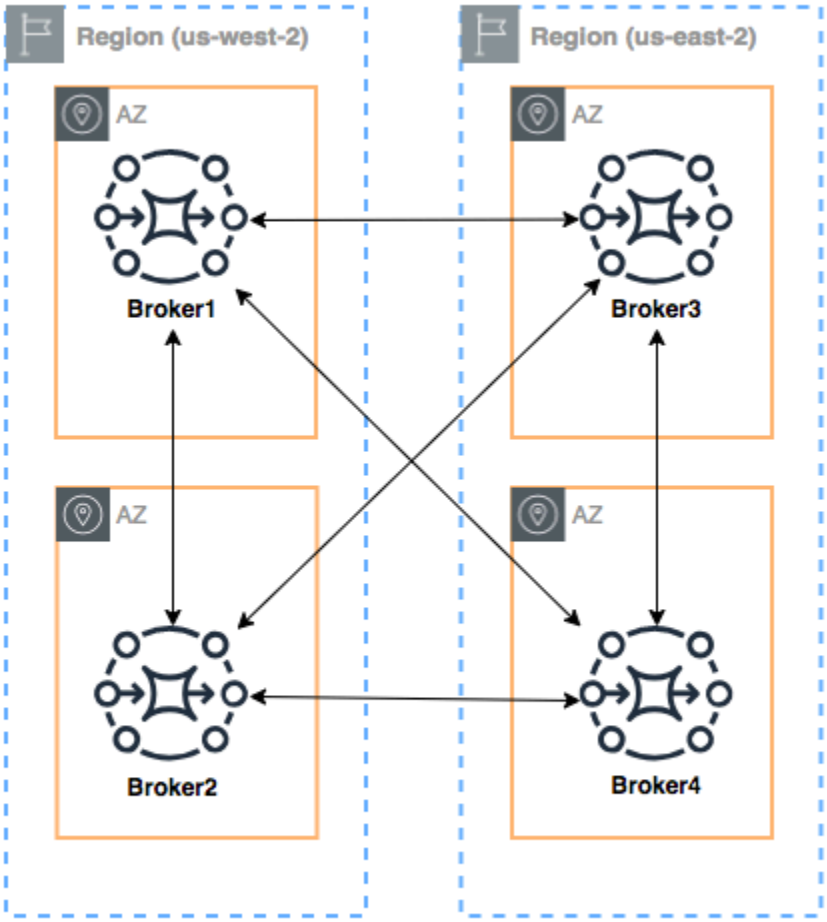
ブローカーのネットワークはどのように認証情報を処理しますか?

ブローカー A がネットワーク内でブローカー B に接続するには、ブローカー A が他のプロデューサーまたはコンシューマーと同様に有効な認証情報を使用する必要があります。ブローカー A の `<networkConnector>` 設定でパスワードを提供する代わりに、ブローカー B の別のユーザーと同じ値を持つブローカー A のユーザーを最初に作成する必要があります (これらは同じユーザー名を共有する別の、一意のユーザーです)。`<networkConnector>` 設定で `userName` 属性を指定すると、Amazon MQ は実行時にパスワードを自動的に追加します。

> **Important**
>
> `<networkConnector>` には `password` 属性を指定しないでください。パスワードが Amazon MQ コンソールに表示されてしまうため、プレーンテキストのパスワードをブローカー設定ファイルに保存することは推奨されません。詳細については、「Configure Network Connectors for Your Broker」を参照してください。

クロスリージョン

AWS リージョンにまたがるブローカーのネットワークを設定するには、それらのリージョンにブローカーをデプロイし、それらのブローカーのエンドポイントへのネットワークコネクタを設定します。



この例のようなブローカーのネットワークを設定するには、これらのブローカーのワイヤレベルのエンドポイントを参照する Broker1 と Broker4 の設定に `networkConnectors` エントリを追加できます。

Broker1 のネットワークコネクター:

```
<networkConnectors>
    <networkConnector name="1_to_2" userName="myCommonUser" duplex="true"
        uri="static:(ssl://b-9876l5k4-32ji-109h-8gfe-7d65c4b132a1-2.mq.us-
west-2.amazonaws.com:61617)"/>
```

```
    <networkConnector name="1_to_3" userName="myCommonUser" duplex="true"
        uri="static:(ssl://b-743c885d-2244-4c95-af67-a85017ff234e-3.mq.us-
east-2.amazonaws.com:61617)"/>
    <networkConnector name="1_to_4" userName="myCommonUser" duplex="true"
        uri="static:(ssl://b-62a7fb31-d51c-466a-a873-905cd660b553-4.mq.us-
east-2.amazonaws.com:61617)"/>
</networkConnectors>
```

Broker2 のネットワークコネクター:

```
<networkConnectors>
    <networkConnector name="2_to_3" userName="myCommonUser" duplex="true"
        uri="static:(ssl://b-743c885d-2244-4c95-af67-a85017ff234e-3.mq.us-
east-2.amazonaws.com:61617)"/>
</networkConnectors>
```

Broker4 のネットワークコネクター:

```
<networkConnectors>
    <networkConnector name="4_to_3" userName="myCommonUser" duplex="true"
        uri="static:(ssl://b-743c885d-2244-4c95-af67-a85017ff234e-3.mq.us-
east-2.amazonaws.com:61617)"/>
    <networkConnector name="4_to_2" userName="myCommonUser" duplex="true"
        uri="static:(ssl://b-9876l5k4-32ji-109h-8gfe-7d65c4b132a1-2.mq.us-
west-2.amazonaws.com:61617)"/>
</networkConnectors>
```

トランスポートコネクタを使用した動的なフェイルオーバー

networkConnector 要素の設定に加えて、ブローカーの transportConnector オプションを設定して動的なフェイルオーバーを有効にし、ネットワークからブローカーが追加または削除されたときに接続を再分散することができます。

```
<transportConnectors>
  <transportConnector name="openwire" updateClusterClients="true"
 rebalanceClusterClients="true" updateClusterClientsOnRemove="true"/>
</transportConnectors>
```

この例では、updateClusterClients および rebalanceClusterClients の両方が true に設定されます。この場合、クライアントにはネットワークのブローカーのリストが提供され、新しいブローカーが参加した場合は再分散がリクエストされます。

利用可能なオプション:

- `updateClusterClients`: ブローカートポロジのネットワークの変化に関する情報をクライアントに渡します。
- `rebalanceClusterClients`: 新しいブローカーがブローカーのネットワークに追加されたときに、クライアントはブローカー間で再分散されます。
- `updateClusterClientsOnRemove`: ブローカーがブローカーのネットワークを離れるときに、トポロジ情報を使用してクライアントを更新します。

`updateClusterClients` を true に設定すると、クライアントはブローカーのネットワークの 1 つのブローカーに接続するように設定されます。

```
failover:(ssl://b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-
east-2.amazonaws.com:61617)
```

新しいブローカーが接続すると、そのブローカーはネットワーク内のすべてのブローカーの URI のリストを受け取ります。ブローカーへの接続に失敗した場合、接続時に、提供されたいずれかのブローカーに動的に切り替えることができます。

フェイルオーバーの詳細については、Active MQ ドキュメントの「Broker-side Options for Failover」を参照してください。

Amazon MQ for ActiveMQ ブローカーインスタンスタイプ

ブローカーインスタンスクラス (`m5`、`t3`) とサイズ (`large`、`micro`) を組み合わせた説明は、ブローカーインスタンスタイプ (など) と呼ばれます`mq.m5.large`。次の表に、ActiveMQ ブローカーで使用できる Amazon MQ ブローカーインスタンスタイプを示します。 ActiveMQ

Amazon MQ は、インスタンスタイプがサポートを終了する少なくとも 90 日前に通知します。中断を防ぐために、end-of-supportより前にブローカーを新しいインスタンスタイプにアップグレードすることをお勧めします。

> **Important**
>
> 2025 年 3 月 17 `t2.micro`日`mq.m4.large`以降はブローカーを作成できません。

インスタンスタイプ	vCPU	メモリ (GiB)	推奨用途	[Storage (ストレージ)]	Amazon MQ でのサポート終了
mq.t3.micro	2	1	評価	EFS	
mq.m5.large	2	8	本番稼働	EFS または EBS	
mq.m5.xlarge	4	16	本番稼働	EFS または EBS	
mq.m5.2xlarge	8	32	本番稼働	EFS または EBS	
mq.m5.4xlarge	16	64	本番稼働	EFS または EBS	

スループットの考察に関する詳細は、「最良なスループットのために正しいブローカーインスタンスタイプを選択する」を参照してください。

Amazon MQ for ActiveMQ ブローカーの設定

設定には、ActiveMQ ブローカーのすべての設定が XML 形式で含まれています (ActiveMQ の `activemq.xml` ファイルに似ています)。設定は、ブローカーを作成する前に作成することができます。作成後、設定を 1 つ、または複数のブローカーに適用できます。

> **Important**
>
> 設定を変更しても、その変更はブローカーに直ちに適用されません。変更を適用するには、次のメンテナンスウィンドウまで待機するか、ブローカーを再起動する必要があります。設定は `DeleteConfiguration` API を使用してのみ削除できます。詳細については、Amazon MQ API リファレンス」の「設定」を参照してください。

属性

ブローカー設定には複数の属性があります。次に例を示します。

- 名前 (`MyConfiguration`)
- ID (`c-1234a5b6-78cd-901e-2fgh-3i45j6k178l9`)
- Amazon リソースネーム (ARN) (`arn:aws:mq:us-east-2:123456789012:configuration:c-1234a5b6-78cd-901e-2fgh-3i45j6k178l9`)

設定属性の完全なリストについては、Amazon MQ REST API リファレンスで以下を参照してください。

- REST オペレーション ID: 設定
- REST オペレーション ID: 設定

設定のリビジョン属性の詳細なリストについては、以下を参照してください。

- REST オペレーション ID: 設定のリビジョン
- REST オペレーション ID: 設定のリビジョン

Spring XML 設定ファイルの使用

Spring XML ファイルを使用して ActiveMQ ブローカーを設定します。事前定義された送信先、送信先ポリシー、認可ポリシー、およびプラグインなど、ActiveMQ ブローカーのさまざまな側面を設定できます。Amazon MQ は、ネットワーク転送およびストレージなど、これらの設定要素の一部を制御します。ブローカーのネットワーク作成など、他の設定オプションは、現在サポートされていません。

サポートされている設定オプションの完全なセットは、Amazon MQ XML スキーマに指定されています。次のリンクを使用して、サポートされているスキーマの zip ファイルをダウンロードします。

- `amazon-mq-active-mq-5.18.4.xsd.zip`
- `amazon-mq-active-mq-5.17.6.xsd.zip`
- `amazon-mq-active-mq-5.16.7.xsd.zip`
- `amazon-mq-active-mq-5.15.16.xsd.zip`

これらのスキーマは、設定ファイルの検証とサニタイズに使用できます。Amazon MQ では、XML ファイルをアップロードして設定を提供することもできます。XML ファイルをアップロードすると、Amazon MQ は、スキーマに従って無効および禁止されている設定パラメータを自動的にサニタイズし、削除します。

> **Note**
>
> 属性には静的な値のみを使用できます。Amazon MQ は、Spring 式、変数、および要素参照が含まれる要素と属性を設定からサニタイズします。

Amazon MQ for ActiveMQ ブローカー設定の作成

設定には、ActiveMQ ブローカーのすべての設定が XML 形式で含まれています (ActiveMQ の `activemq.xml` ファイルに似ています)。設定は、ブローカーを作成する前に作成することができます。作成後、設定を 1 つ、または複数のブローカーに適用できます。設定は直ちに適用する、またはメンテナンスウィンドウ中に適用することができます。

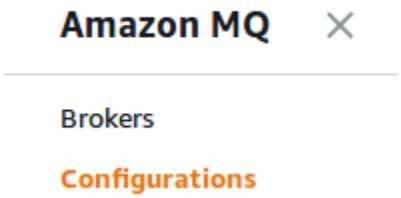
以下の例では、 AWS Management Consoleを使用して Amazon MQ ブローカーの設定を作成し、適用する方法を説明します。

> **Important**
>
> 設定は `DeleteConfiguration` API を使用してのみ削除できます。詳細については、Amazon MQ API リファレンス」の「設定」を参照してください。

新しい設定の作成

新しいブローカー設定を作成するには、まず新しい設定を作成します。

1. Amazon MQ コンソールにサインインします。
2. 左側のナビゲーションパネルを展開し、[設定] を選択します。



3. [設定] ページで、[Create configuration (設定の作成)] を選択します。
4. [Create configuration] (設定の作成) ページの [Details] (詳細) セクションで [Configuration name] (設定名)(`MyConfiguration` など) を入力し、ブローカーエンジンのバージョンを選択します。

> **Note**
>
> Amazon MQ for ActiveMQ がサポートする ActiveMQ エンジンバージョンの詳細については、「the section called “バージョン管理”」を参照してください。

5. [Create configuration] (設定の作成) をクリックします。

新しい設定リビジョンの作成

ブローカー設定を作成したら、設定リビジョンを使用して設定を編集する必要があります。

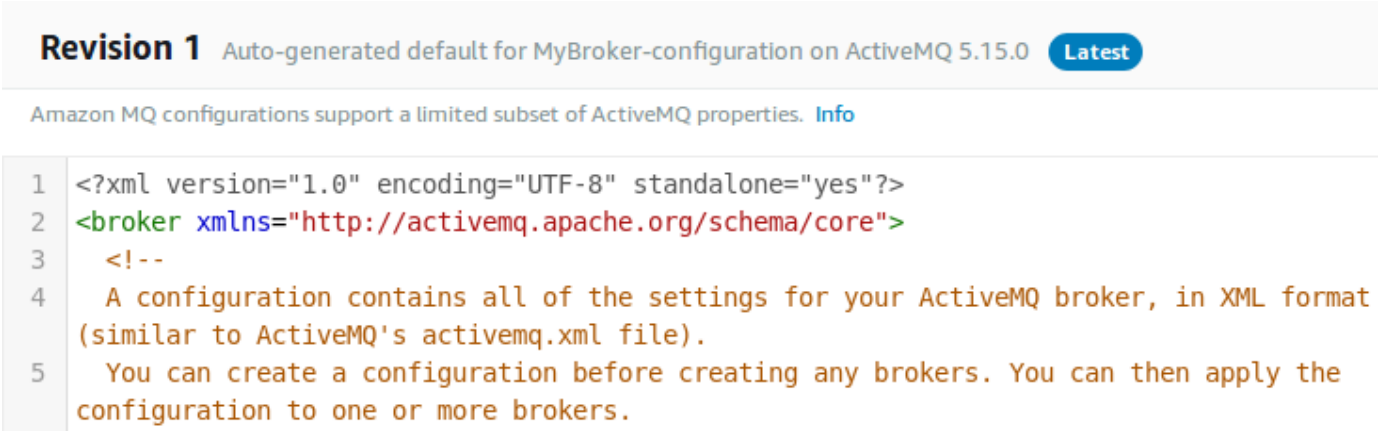
1. 設定リストから、[***MyConfiguration***] を選択します。

> **Note**
>
> 設定の最初のリビジョンは常に、Amazon MQ が設定を作成するときに作成されます。

[***MyConfiguration***] ページに、新しい設定リビジョンで使用されるブローカーのエンジンタイプとバージョン (Apache ActiveMQ 5.15.16 など) が表示されます。

2. [Configuration details] タブに、設定リビジョン番号、説明、およびブローカー設定が XML 形式で表示されます。

> **Note**
>
> 現在の設定を編集すると、設定の新しいリビジョンが作成されます。



3. [Edit configuration] (設定の編集) をクリックして、XML 設定を変更します。
4. [Save] (保存) をクリックします。

   [Save revision] (リビジョンの保存) ダイアログボックスが表示されます。
5. (オプション) `A description of the changes in this revision` を入力します。
6. [保存] を選択します。

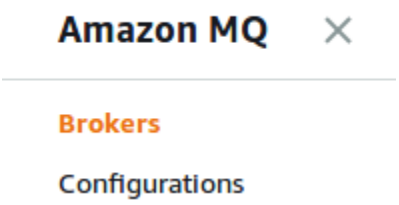
   設定の新しいリビジョンが保存されます。

> **Important**
>
> Amazon MQ コンソールは、スキーマに従って、無効および禁止されている設定パラメータを自動的にサニタイズします。許可されている XML パラメータの詳細および完全なリストについては、「Amazon MQ Broker Configuration Parameters」を参照してください。

設定リビジョンをブローカーに適用する

設定を変更したら、設定リビジョンをブローカーに適用できます。

1. 左側のナビゲーションパネルを展開し、[Brokers (ブローカー)] を選択します。



2. ブローカーリストからブローカーを選択して (MyBroker など)、[Edit] (編集) をクリックします。
3. [Edit ***MyBroker***] (MyBroker の編集) ページの [Configuration] (設定) セクションで [Configuration] (設定) と [Revision] (リビジョン) を選択してから、[Schedule Modifications] (変更をスケジュールする) をクリックします。
4. [ブローカー変更のスケジュール] セクションで、変更を [次回のスケジュールされたメンテナンスウィンドウ中] に適用するか、[即時] 適用するかを選択します。

> **Important**
>
> 1 つのインスタンスブローカーは再起動中にオフラインになります。クラスターブローカーの場合、ブローカーの再起動中に一度にダウンするノードは 1 つだけです。

5. [Apply] (適用) をクリックします。

   設定リビジョンが指定された時刻にブローカーに適用されます。

Amazon MQ for ActiveMQ 設定リビジョンの編集

ブローカーに設定リビジョンを適用した後で、そのリビジョンを編集する必要が生じることがあります。設定リビジョンを編集するには、次の手順に従います。

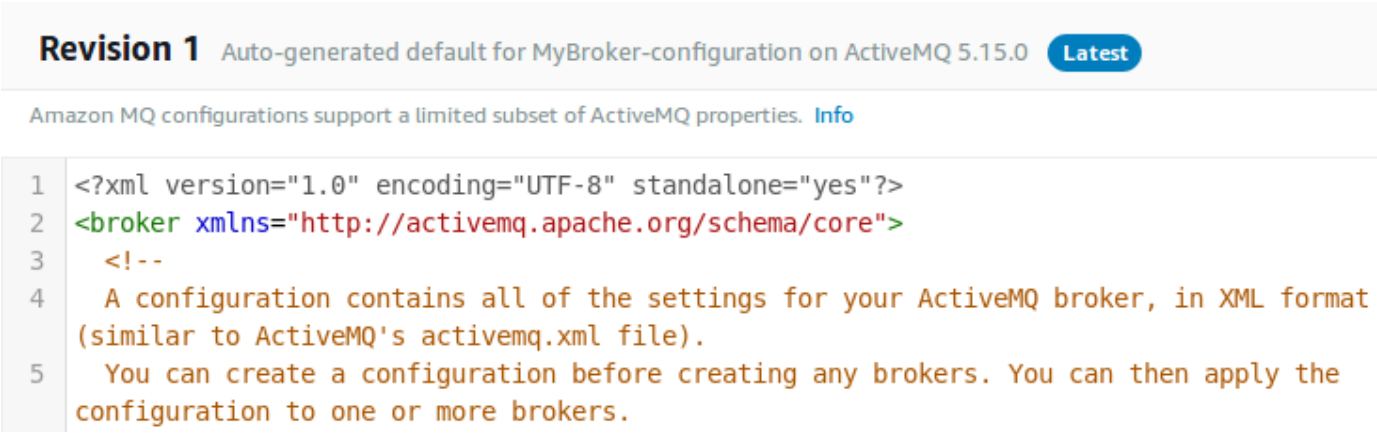
1. Amazon MQ コンソールにサインインします。
2. ブローカーリストからブローカーを選択して (MyBroker など)、[Edit] (編集) をクリックします。
3. [***MyBroker***] ページで、[編集] を選択します。
4. [Edit ***MyBroker***] (MyBroker の編集) ページの [Configuration] (設定) セクションで [Configuration] (設定) と [Revision] (リビジョン) を選択してから、[Edit] (変更) をクリックします。

> **Note**
>
> ブローカーの作成時に設定を選択する場合を除き、最初のリビジョンは、常に Amazon MQ がブローカーを作成する時に作成されます。

[***MyBroker***] ページに、設定が使用するブローカーのエンジンタイプとバージョン (Apache ActiveMQ 5.15.8 など) が表示されます。

5. [Configuration details] タブに、設定リビジョン番号、説明、およびブローカー設定が XML 形式で表示されます。

> **Note**
>
> 現在の設定を編集すると、設定の新しいリビジョンが作成されます。



6. [Edit configuration] (設定の編集) をクリックして、XML 設定を変更します。
7. [Save] (保存) をクリックします。

   [Save revision] (リビジョンの保存) ダイアログボックスが表示されます。
8. (オプション) `A description of the changes in this revision` を入力します。
9. [保存] を選択します。

   設定の新しいリビジョンが保存されます。

> **Important**
>
> Amazon MQ コンソールは、スキーマに従って、無効および禁止されている設定パラメータを自動的にサニタイズします。許可されている XML パラメータの詳細および完全なリストについては、「Amazon MQ Broker Configuration Parameters」を参照してください。

Amazon MQ の設定で許可されている要素

以下は、Amazon MQ 設定で許可されている要素の詳しいリストです。詳細については、Apache ActiveMQ ドキュメントの XML 設定を参照してください。

要素
abortSlowAckConsumerStrategy (属性)
abortSlowConsumerStrategy (属性)
authorizationEntry (属性)
authorizationMap (子コレクション要素)
authorizationPlugin (子コレクション要素)
broker (属性 | 子コレクション要素)
cachedMessageGroupMapFactory (属性)
compositeQueue (属性 | 子コレクション要素)
compositeTopic (属性 | 子コレクション要素)
constantPendingMessageLimitStrategy (属性)
discarding (属性)
discardingDLQBrokerPlugin (属性)
fileCursor
fileDurableSubscriberCursor
fileQueueCursor
filteredDestination (属性)
fixedCountSubscriptionRecoveryPolicy (属性)

要素
fixedSizedSubscriptionRecoveryPolicy (属性)
forcePersistencyModeBrokerPlugin (属性)
individualDeadLetterStrategy (属性)
lastImageSubscriptionRecoveryPolicy
messageGroupHashBucketFactory (属性)
mirroredQueue (属性)
noSubscriptionRecoveryPolicy
oldestMessageEvictionStrategy (属性)
oldestMessageWithLowestPriorityEvictionStrategy (属性)
policyEntry (属性 | 子コレクション要素)
policyMap (子コレクション要素)
prefetchRatePendingMessageLimitStrategy (属性)
priorityDispatchPolicy
priorityNetworkDispatchPolicy
queryBasedSubscriptionRecoveryPolicy (属性)
queue (属性)
redeliveryPlugin (属性 | 子コレクション要素)
redeliveryPolicy (属性)
redeliveryPolicyMap (子コレクション要素)
retainedMessageSubscriptionRecoveryPolicy (子コレクション要素)

要素
roundRobinDispatchPolicy
sharedDeadLetterStrategy (属性 | 子コレクション要素)
simpleDispatchPolicy
simpleMessageGroupMapFactory
statisticsBrokerPlugin
storeCursor
storeDurableSubscriberCursor (属性)
strictOrderDispatchPolicy
tempDestinationAuthorizationEntry (属性)
tempQueue (属性)
tempTopic (属性)
timedSubscriptionRecoveryPolicy (属性)
timeStampingBrokerPlugin (属性)
topic (属性)
transportConnector (属性)
uniquePropertyMessageEvictionStrategy (属性)
virtualDestinationInterceptor (子コレクション要素)
virtualTopic (属性)
vmCursor
vmDurableCursor

要素
vmQueueCursor

Amazon MQ 設定で許可されている要素とその属性

以下は、Amazon MQ 設定で許可されている要素とその属性の詳しいリストです。詳細については、Apache ActiveMQ ドキュメントの XML 設定を参照してください。

要素	属性
abortSlowAckConsumerStrategy	abortConnection
	checkPeriod
	ignoreIdleConsumers
	ignoreNetworkConsumers
	maxSlowCount
	maxSlowDuration
	maxTimeSinceLastAck
	name
abortSlowConsumerStrategy	abortConnection
	checkPeriod
	ignoreNetworkConsumers
	maxSlowCount
	maxSlowDuration
	name
authorizationEntry	admin

要素	属性
	queue
	read
	tempQueue
	tempTopic
	topic
	write
broker	advisorySupport
	allowTempAutoCreationOnSend
	cacheTempDestinations
	consumerSystemUsagePortion
	dedicatedTaskRunner
	deleteAllMessagesOnStartup
	keepDurableSubsActive
	enableMessageExpirationOnActiveDurableSubs
	maxPurgedDestinationsPerSweep
	maxSchedulerRepeatAllowed
	monitorConnectionSplits
	networkConnectorStartAsync
	offlineDurableSubscriberTaskSchedule

要素	属性
	offlineDurableSubscriberTimeout
	persistenceThreadPriority
	persistent
	populateJMSXUserID
	producerSystemUsagePortion
	rejectDurableConsumers
	rollbackOnlyOnAsyncException
	schedulePeriodForDestinatio nPurge
	schedulerSupport
	splitSystemUsageForProducer sConsumers
	taskRunnerPriority
	timeBeforePurgeTempDestinations
	useAuthenticatedPrincipalFo rJMSXUserID
	useMirroredQueues
	useTempMirroredQueues
	useVirtualDestSubs
	useVirtualDestSubsOnCreation
	useVirtualTopics

要素	属性
cachedMessageGroupMapFactory	cacheSize
compositeQueue	concurrentSend
	copyMessage
	forwardOnly
	name
	sendWhenNotMatched
compositeTopic	concurrentSend
	copyMessage
	forwardOnly
	name
	sendWhenNotMatched
conditionalNetworkBridgeFilterFactory	rateDuration
	rateLimit
	replayDelay
	replayWhenNoConsumers
	selectorAware 以下でサポート Apache ActiveMQ 5.16.x
constantPendingMessageLimitStrategy	limit

要素	属性
discarding	deadLetterQueue
	enableAudit
	expiration
	maxAuditDepth
	maxProducersToAudit
	processExpired
	processNonPersistent
discardingDLQBrokerPlugin	dropAll
	dropOnly
	dropTemporaryQueues
	dropTemporaryTopics
	reportInterval
filteredDestination	queue
	selector
	topic
fixedCountSubscriptionRecoveryPolicy	maximumSize
fixedSizedSubscriptionRecoveryPolicy	maximumSize
	useSharedBuffer
forcePersistencyModeBrokerPlugin	persistenceFlag
individualDeadLetterStrategy	destinationPerDurableSubscriber

要素	属性
	enableAudit
	expiration
	maxAuditDepth
	maxProducersToAudit
	processExpired
	processNonPersistent
	queuePrefix
	queueSuffix
	topicPrefix
	topicSuffix
	useQueueForQueueMessages
	useQueueForTopicMessages
messageGroupHashBucketFactory	bucketCount
	cacheSize
mirroredQueue	copyMessage
	postfix
	prefix
oldestMessageEvictionStrategy	evictExpiredMessagesHighWatermark
oldestMessageWithLowestPriorityEvictionStrategy	evictExpiredMessagesHighWatermark

要素	属性
policyEntry	advisoryForConsumed
	advisoryForDelivery
	advisoryForDiscardingMessages
	advisoryForFastProducers
	advisoryForSlowConsumers
	advisoryWhenFull
	allConsumersExclusiveByDefault
	alwaysRetroactive
	blockedProducerWarningInterval
	consumersBeforeDispatchStarts
	cursorMemoryHighWaterMark
	doOptimzeMessageStorage
	durableTopicPrefetch
	enableAudit
	expireMessagesPeriod
	gcInactiveDestinations
	gcWithNetworkConsumers
	inactiveTimeoutBeforeGC
	inactiveTimoutBeforeGC
	includeBodyForAdvisory

要素	属性
	lazyDispatch
	maxAuditDepth
	maxBrowsePageSize
	maxDestinations
	maxExpirePageSize
	maxPageSize
	maxProducersToAudit
	maxQueueAuditDepth
	memoryLimit
	messageGroupMapFactoryType
	minimumMessageSize
	optimizedDispatch
	optimizeMessageStoreInFligh tLimit
	persistJMSRedelivered
	prioritizedMessages
	producerFlowControl
	queue
	queueBrowserPrefetch
	queuePrefetch
	reduceMemoryFootprint

要素	属性
	sendAdvisoryIfNoConsumers
	sendFailIfNoSpace
	sendFailIfNoSpaceAfterTimeout 以下でサポート Apache ActiveMQ 5.16.4 以上
	sendDuplicateFromStoreToDLQ
	storeUsageHighWaterMark
	strictOrderDispatch
	tempQueue
	tempTopic
	timeBeforeDispatchStarts
	topic
	topicPrefetch
	useCache
	useConsumerPriority
usePrefetchExtension	
prefetchRatePendingMessageLimitStrategy	multiplier
queryBasedSubscriptionRecoveryPolicy	query

要素	属性
queue	DLQ
	physicalName
redeliveryPlugin	fallbackToDeadLetter
	sendToDlqIfMaxRetriesExceeded
redeliveryPolicy	backOffMultiplier
	collisionAvoidancePercent
	initialRedeliveryDelay
	maximumRedeliveries
	maximumRedeliveryDelay
	preDispatchCheck
	queue
	redeliveryDelay
	tempQueue
	tempTopic
	topic
	useCollisionAvoidance
	useExponentialBackOff
sharedDeadLetterStrategy	enableAudit
	expiration
	maxAuditDepth

要素	属性
storeDurableSubscriberCursor	maxProducersToAudit
	processExpired
	processNonPersistent
	immediatePriorityDispatch
	useCache
tempDestinationAuthorizationEntry	admin
	queue
	read
	tempQueue
	tempTopic
	topic
	write
tempQueue	DLQ
	physicalName
tempTopic	DLQ
	physicalName
timedSubscriptionRecoveryPolicy	zeroExpirationOverride
timeStampingBrokerPlugin	recoverDuration
	futureOnly
	processNetworkMessages

要素	属性
	ttlCeiling
topic	DLQ
	physicalName
transportConnector	•
	name
	updateClusterClients
	rebalanceClusterClients
	updateClusterClientsOnRemove
uniquePropertyMessageEvictionStrategy	evictExpiredMessagesHighWatermark
	propertyName
virtualTopic	concurrentSend
	local
	dropOnResourceLimit
	name
	postfix
	prefix
	selectorAware
	setOriginalDestination
	transactedSend

Amazon MQ 親要素属性

以下は、親要素属性の詳しい説明です。詳細については、Apache ActiveMQ ドキュメントの XML 設定を参照してください。

トピック

- ブローカー

ブローカー

`broker` は親コレクションの要素です。

属性

networkConnectionStartAsync

ネットワークのレイテンシーを短縮し、他のネットワークをタイムリーに起動できるようにするには、`<networkConnectionStartAsync>` タグを使用します。このタグは、ブローカーの起動とは非同期に、エグゼキューターを使用してネットワーク接続を並列に起動するようにブローカーに指示します。

デフォルト: `false`

サンプル設定

```
<broker networkConnectorStartAsync="false"/>
```

Amazon MQ 設定で許可されている要素、子コレクション要素、およびそれらの子要素

以下は、Amazon MQ 設定で許可されている要素、子コレクション要素、およびそれらの子要素の詳しいリストです。詳細については、Apache ActiveMQ ドキュメントの XML 設定を参照してください。

要素	子コレクション要素	子要素
`authorizationMap`	`authorizationEntries`	`authorizationEntry`
		`tempDestinationAuthorizationEntry`

要素	子コレクション要素	子要素
	defaultEntry	authorizationEntry
		tempDestinationAuthorizationEntry
	tempDestinationAuthorizationEntry	tempDestinationAuthorizationEntry
authorizationPlugin	map	authorizationMap
broker	destinationInterceptors	mirroredQueue
		virtualDestinationInterceptor
	destinationPolicy	policyMap
	destinations	queue
		tempQueue
		tempTopic
		topic
	networkConnectors	networkConnector
	persistenceAdapter	kahaDB
	plugins	authorizationPlugin
		discardingDLQBrokerPlugin
		forcePersistencyModeBrokerPlugin
		redeliveryPlugin

要素	子コレクション要素	子要素
		statisticsBrokerPlugin
		timeStampingBrokerPlugin
	systemUsage	systemUsage
	transportConnector	name
		updateClusterClients
		rebalanceClusterClients
		updateClusterClientsOnRemove
compositeQueue	forwardTo	queue
		tempQueue
		tempTopic
		topic
		filteredDestination
compositeTopic	forwardTo	queue
		tempQueue
		tempTopic
		topic
		filteredDestination
policyEntry	deadLetterStrategy	discarding

要素	子コレクション要素	子要素
		individualDeadLetterStrategy
		sharedDeadLetterStrategy
	destination	queue
		tempQueue
		tempTopic
		topic
	dispatchPolicy	priorityDispatchPolicy
		priorityNetworkDispatchPolicy
		roundRobinDispatchPolicy
		simpleDispatchPolicy
		strictOrderDispatchPolicy
		clientIdFilterDispatchPolicy
	messageEvictionStrategy	oldestMessageEvictionStrategy
		oldestMessageWithLowestPriorityEvictionStrategy

要素	子コレクション要素	子要素
		uniquePropertyMessageEvictionStrategy
	messageGroupMapFactory	cachedMessageGroupMapFactory
		messageGroupHashBucketFactory
		simpleMessageGroupMapFactory
	pendingDurableSubscriberPolicy	fileDurableSubscriberCursor
		storeDurableSubscriberCursor
		vmDurableCursor
	pendingMessageLimitStrategy	constantPendingMessageLimitStrategy
		prefetchRatePendingMessageLimitStrategy
	pendingQueuePolicy	fileQueueCursor
		storeCursor
		vmQueueCursor
	pendingSubscriberPolicy	fileCursor
		vmCursor

要素	子コレクション要素	子要素
	slowConsumerStrategy	abortSlowAckConsumerStrategy
		abortSlowConsumerStrategy
	subscriptionRecoveryPolicy	fixedCountSubscriptionRecoveryPolicy
		fixedSizedSubscriptionRecoveryPolicy
		lastImageSubscriptionRecoveryPolicy
		noSubscriptionRecoveryPolicy
		queryBasedSubscriptionRecoveryPolicy
		retainedMessageSubscriptionRecoveryPolicy
timedSubscriptionRecoveryPolicy		
policyMap	defaultEntry	policyEntry
	policyEntries	policyEntry
redeliveryPlugin	redeliveryPolicyMap	redeliveryPolicyMap
redeliveryPolicyMap	defaultEntry	redeliveryPolicy
	redeliveryPolicyEntries	redeliveryPolicy

要素	子コレクション要素	子要素
retainedMessageSubscriptionRecoveryPolicy	wrapped	fixedCountSubscriptionRecoveryPolicy
		fixedSizedSubscriptionRecoveryPolicy
		lastImageSubscriptionRecoveryPolicy
		noSubscriptionRecoveryPolicy
		queryBasedSubscriptionRecoveryPolicy
		retainedMessageSubscriptionRecoveryPolicy
		timedSubscriptionRecoveryPolicy
sharedDeadLetterStrategy	deadLetterQueue	queue
		tempQueue
		tempTopic
		topic
virtualDestinationInterceptor	virtualDestinations	compositeQueue
		compositeTopic
		virtualTopic

Amazon MQ 子要素属性

以下は、子要素属性の詳しい説明です。詳細については、Apache ActiveMQ ドキュメントの XML 設定を参照してください。

トピック

authorizationEntry

`authorizationEntry` は `authorizationEntries` 子コレクション要素の子です。

属性

管理|読み取り|書き込み

ユーザーのグループに付与されているアクセス許可。詳細については、「認可マップを常に設定する」を参照してください。

`activemq-webconsole` グループが含まれない認可マップを指定する場合、Amazon MQ ブローカーにメッセージを送信する権限、またはブローカーからメッセージを受信する権限がグループにないことから、ActiveMQ ウェブコンソールは使用できません。

デフォルト: `null`

サンプル設定

```
<authorizationPlugin>
                                <map>
                                    <authorizationMap>
                                        <authorizationEntries>
                                            <authorizationEntry admin="admins,activemq-
webconsole" read="admins,users,activemq-webconsole" write="admins,activemq-webconsole"
 queue=">"/>
                                            <authorizationEntry admin="admins,activemq-
webconsole" read="admins,users,activemq-webconsole" write="admins,activemq-webconsole"
 topic=">"/>
```

```
                                    </authorizationEntries>
                                </authorizationMap>
                            </map>
                            </authorizationPlugin>
```

> **Note**
>
> Amazon MQ 上の ActiveMQ の `activemq-webconsole` グループには、すべてのキューとトピックに対する管理者アクセス許可があります。このグループのすべてのユーザーは管理者アクセス権を持ちます。

networkConnector

`networkConnector` は `networkConnectors` 子コレクション要素の子です。

トピック

- 属性
- 設定例

属性

conduitSubscriptions

ブローカーのネットワークのネットワーク接続が、同じ送信先にサブスクライブしている複数のコンシューマーを 1 つのコンシューマーとして扱うかどうかを指定します。たとえば、`conduitSubscriptions` が `true` に設定されていて、2 つのコンシューマーがブローカー B に接続して送信先から消費する場合、ブローカー B は、ブローカー A へのネットワーク接続を介してサブスクリプションを単一の論理サブスクリプションに結合するので、メッセージの単一コピーのみがブローカー A からブローカー B に転送されます。

> **Note**
>
> `conduitSubscriptions` を `true` に設定すると、冗長なネットワークトラフィックを減らすことができます。ただし、この属性を使用すると、コンシューマー間でのメッセージのロードバランシングに影響が出る可能性があり、特定のシナリオ (JMS メッセージセレクタや耐久性のあるトピックなど) では正しくない動作を引き起こす可能性があります。

デフォルト: `true`

二重

ブローカーのネットワーク内の接続を使用し、またメッセージを生成するかどうかを指定します。ブ たとえば、ブローカー A が非二重モードでブローカー B への接続を作成した場合、メッセージはブローカー A からブローカー B にのみ転送できます。ただし、ブローカー A がブローカー B への二重接続を作成した場合、ブローカー B は <networkConnector> を設定しなくてもメッセージをブローカー A に転送できます。

デフォルト: `false`

名前

ブローカーのネットワークのブリッジの名前。

デフォルト: `bridge`

uri

ブローカーのネットワークの 2 つのブローカーのうちの 1 つ (または複数のブローカー) のワイヤレベルプロトコルエンドポイント。

デフォルト: `null`

username

ブローカーのネットワークのブローカーに共通のユーザー名。

デフォルト: `null`

設定例

> **Note**
>
> `networkConnector` を使用してブローカーのネットワークを定義するときは、ブローカーに共通のユーザーのパスワードを含めないでください。

2 つのブローカーとブローカーのネットワーク

この設定では、2 つのブローカーがブローカーのネットワークで接続されています。ネットワークコネクターの名前は `connector_1_to_2`、ブローカーに共通のユーザー名は `myCommonUser`、接続

は duplex、そして OpenWire エンドポイント URI は static: というプレフィックスは、ブローカー間の 1 対 1 の接続を示します。

```
<networkConnectors>
                                    <networkConnector name="connector_1_to_2"
 userName="myCommonUser" duplex="true"
                                        uri="static:(ssl://
b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-east-2.amazonaws.com:61617)"/>
                                    </networkConnectors>
```

詳細については、「Configure Network Connectors for Your Broker」を参照してください。

複数のブローカーのあるブローカーのネットワーク

この設定では、複数のブローカーがブローカーのネットワークで接続されています。ネットワークコネクターの名前は connector_1_to_2、ブローカーに共通のユーザー名は myCommonUser、接続は duplex です。OpenWire エンドポイント URI のカンマ区切りのリストの前には masterslave: というプレフィックスが付き、ブローカー間のフェイルオーバー接続を示します。ブローカーからブローカーへのフェイルオーバーはランダム化されず、再接続の試行は無期限に続きます。

```
<networkConnectors>
                                    <networkConnector name="connector_1_to_2"
 userName="myCommonUser" duplex="true"
                                        uri="masterslave:(ssl://
b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-east-2.amazonaws.com:61617,
                                        ssl://
b-9876l5k4-32ji-109h-8gfe-7d65c4b132a1-2.mq.us-west-2.amazonaws.com:61617)"/>
                                    </networkConnectors>
```

> **Note**
>
> ブローカーのネットワークの masterslave: プレフィックスを使用することをお勧めします。プレフィックスはより明示的な static:failover:()?randomize=false&maxReconnectAttempts=0 構文と完全に一致します。

> **Note**
>
> この XML 設定ではスペースを使用できません。

kahaDB

`kahaDB` は `persistenceAdapter` 子コレクション要素の子です。

属性

concurrentStoreAndDispatchQueues

キューの同時保存とディスパッチを使用するかどうかを指定します。詳細については、「低速コンシューマーのキューに対して同時保存とディスパッチを無効にする」を参照してください。

デフォルト: `true`

cleanupOnStop

> 以下でサポート
>
> Apache ActiveMQ 15.16.x 以上

無効にされていると、ブローカーが停止されたときにガベージコレクションおよびクリーンアップが実行されず、シャットダウンプロセスの速度が上がります。高速化は、大規模なデータベースやスケジューラデータベースの場合に有用です。

デフォルト: `true`

journalDiskSyncInterval

`journalDiskSyncStrategy=periodic` の場合にディスク同期を実行する間隔 (ミリ秒)。詳細については、Apache ActiveMQ kahaDB のドキュメントを参照してください。

デフォルト: `1000`

journalDiskSyncStrategy

> 以下でサポート
>
> Apache ActiveMQ 15.14.x 以上

ディスク同期ポリシーを設定します。詳細については、Apache ActiveMQ kahaDB のドキュメントを参照してください。

デフォルト: `always`

> **Note**
>
> ActiveMQ のドキュメントでは、データ損失は `journalDiskSyncInterval` の長さに制限されており、デフォルトは 1 秒です。厳密には言えませんが、データ損失はこの間隔よりも長くなる可能性があります。注意してください。

preallocationStrategy

新しいジャーナルファイルが必要になったときにブローカーがジャーナルファイルの事前割り当てを試みる方法を設定します。詳細については、Apache ActiveMQ kahaDB のドキュメントを参照してください。

デフォルト: `sparse_file`

サンプル設定

Example

```
<broker xmlns="http://activemq.apache.org/schema/core">
                                <persistenceAdapter>
                                <kahaDB preallocationStrategy="zeros"
 concurrentStoreAndDispatchQueues="false" journalDiskSyncInterval="10000"
 journalDiskSyncStrategy="periodic"/>
                            </persistenceAdapter>
                            </broker>
```

systemUsage

`systemUsage` は `systemUsage` 子コレクション要素の子です。プロデューサーの速度を遅くするまでにブローカーが使用する領域の最大量を制御します。詳細については、Apache ActiveMQ のドキュメントの Producer Flow Control を参照してください。

子要素

memoryUsage

`memoryUsage` は `systemUsage` 子要素の子です。メモリ使用量を管理します。本番稼働での作業セットの使用を制御できるように、`memoryUsage` を使用してメモリ使用量を追跡します。詳細については、Apache ActiveMQ のドキュメントの schema を参照してください。

子要素

memoryUsage は memoryUsage 子要素の子です。

属性

percentOfJvmHeap

0 ~ 70 の整数。

デフォルト: 70

属性

sendFailIfNoSpace

空き領域がない場合に send( ) メソッドが失敗するかどうかを設定します。デフォルト値は false で、領域が空くまで send( ) メソッドをブロックします。詳細については、Apache Active MQ のドキュメントの schema を参照してください。

デフォルト: false

sendFailIfNoSpaceAfterTimeout

デフォルト: null

サンプル設定

Example

```
<broker xmlns="http://activemq.apache.org/schema/core">
                                <systemUsage>
                                <systemUsage sendFailIfNoSpace="true"
 sendFailIfNoSpaceAfterTimeout="2000">
                                    <memoryUsage>
                                        <memoryUsage  percentOfJvmHeap="60" />
                                    </memoryUsage>>
                                </systemUsage>
                                </systemUsage>
                            </broker>
                            </persistenceAdapter>
```

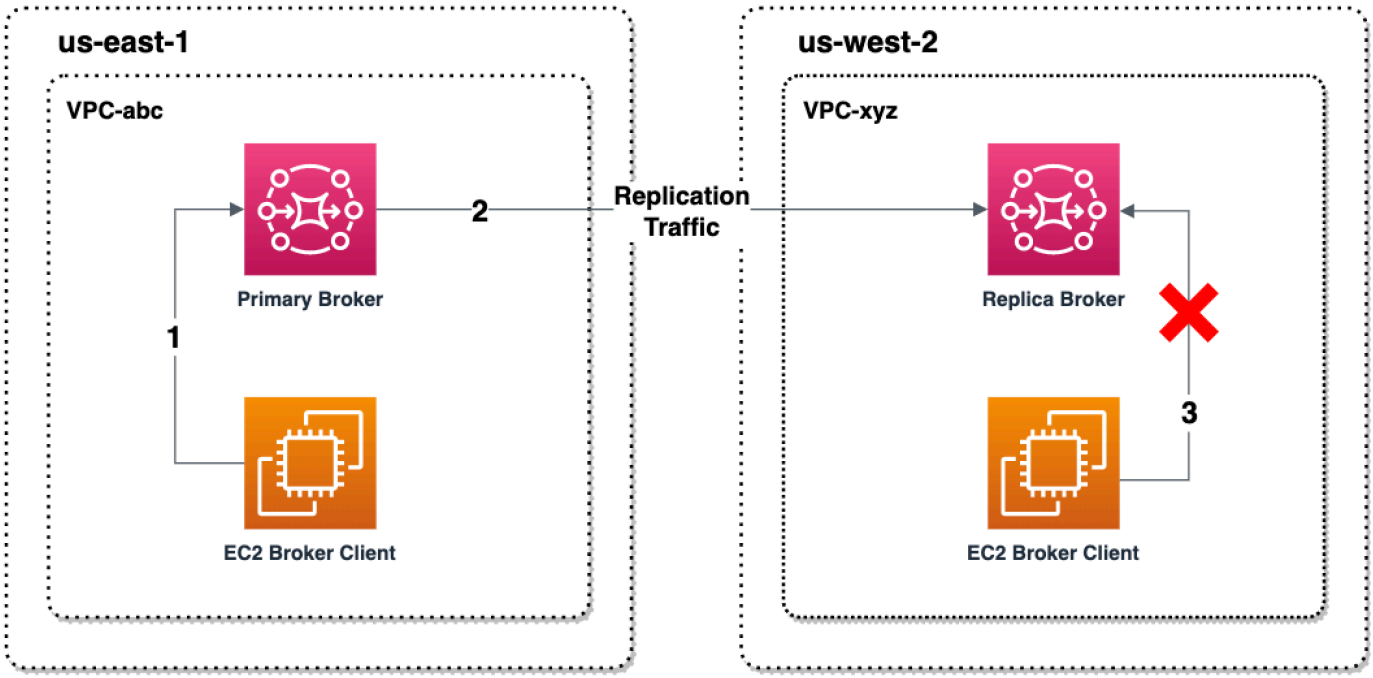
Amazon MQ for ActiveMQ のクロスリージョンデータレプリケーション

Amazon MQ for ActiveMQ は、クロスリージョンデータレプリケーション (CRDR) 機能を提供し、プライマリリージョンのプライマリブローカーからレプリカ AWS リージョンのレプリカブローカーへの非同期メッセージレプリケーションを可能にします。Amazon MQ API にフェイルオーバーリクエストを発行すると、現在のレプリカブローカーはプライマリブローカーのロールに昇格され、現在のプライマリブローカーはレプリカのロールに降格されます。

クロスリージョンデータレプリケーション用のプライマリブローカーとレプリカブローカー

プライマリリージョンのプライマリブローカーからレプリカ AWS リージョンのレプリカブローカーへの非同期データレプリケーション用のプライマリブローカーとレプリカブローカーを作成できます。プライマリリージョンは、プライマリブローカーと呼ばれるアクティブ/スタンバイブローカーの冗長ペアで構成されます。セカンダリリージョンは、レプリカブローカーと呼ばれるアクティブ/スタンバイブローカーの冗長ペアで構成されます。

次の図は、セカンダリリージョンのレプリカブローカーが、プライマリリージョンのプライマリブローカーから非同期にレプリケートされたデータを受信する様子を示しています。



プライマリブローカーとレプリカブローカーは、クロスリージョンのデータ復旧ソリューションとして機能します。プライマリリージョンのプライマリブローカーに障害が発生した場合、スイッチオーバーまたはフェイルオーバーを開始することで、セカンダリリージョンのレプリカブローカーをプライマリに昇格させることができます。元のプライマリブローカーはレプリカブローカーになり、元のレプリカブローカーはプライマリブローカーに昇格されます。プライマリブローカーとレプリカブローカーの作成手順については、「Amazon MQ クロスリージョンデータレプリケーションブローカーの作成」を参照してください。

> **Note**
>
> アクティブ/スタンバイブローカーでのみ使用できます。
> ミラーキューでは使用できません。

Amazon MQ クロスリージョンデータレプリケーションブローカーの作成

クロスリージョンデータレプリケーション (CRDR) を使用すると、必要に応じて 2 つの AWS リージョンの Amazon MQ for ActiveMQ メッセージブローカーを切り替えることができます。既存のブローカーをプライマリブローカーとして指定し、このブローカーのレプリカを作成することも、新しいプライマリブローカーとレプリカブローカーを一緒に作成することもできます。その後、Amazon MQ `Promote` API オペレーションを使用して、レプリカブローカーをプライマリブローカーのロールに昇格させることができます。プライマリブローカーとレプリカブローカーの詳細については、「クロスリージョンデータレプリケーション用のプライマリブローカーとレプリカブローカー」を参照してください。

次の手順では、Amazon MQ マネジメントコンソールを使用してレプリカブローカーを作成および設定する方法について説明します。

トピック

- 前提条件
- ステップ 1 (オプション): 新しいプライマリブローカーを作成する
- ステップ 2: 既存のブローカーのレプリカを作成する

前提条件

クロスリージョンデータレプリケーション機能を使用するには、以下の前提条件を確認して遵守する必要があります。

- バージョン: クロスリージョンデータレプリケーション機能は、バージョン 5.17.6 以降の Amazon MQ for ActiveMQ ブローカーでのみ利用できます。
- リージョン: クロスリージョンデータレプリケーションは、米国東部 (オハイオ)、米国東部 (バージニア北部)、米国西部 (オレゴン)、および米国西部 (北カリフォルニア) の各リージョンでサポートされます。
- インスタンスタイプ: クロスリージョンデータレプリケーションは、`mq.m5.large` 以上のブローカーのインスタンスサイズでのみ利用できます。
- デプロイタイプ: クロスリージョンデータレプリケーションは、複数のアベイラビリティーゾーンデプロイのアクティブ/スタンバイブローカーでのみ利用できます。
- ブローカーのステータス: ブローカーステータスが `Running` のプライマリブローカーのレプリカブローカーのみを作成できます。

ステップ 1 (オプション): 新しいプライマリブローカーを作成する

新しいプライマリブローカーを作成する

1. Amazon MQ コンソールにサインインします。
2. Amazon MQ コンソールの [ブローカー] ページで、[ブローカーの作成] を選択します。
3. [Select broker engine] (ブローカーエンジンの選択) ページで [Apache ActiveMQ] を選択します。
4. [Select deployment and storage] (デプロイとストレージタイプの選択) ページの [Deployment mode and storage type] (デプロイモードとストレージタイプ) セクションで、以下を実行します。

    - [デプロイモード] で、[アクティブ/スタンバイブローカー] を選択します。アクティブ/スタンバイブローカーは、2 つの異なるアベイラビリティーゾーンで冗長ペアとして設定された 2 つのブローカーで構成されます。これらのブローカーは、アプリケーションおよび Amazon EFS と同期的に通信します。詳細については、「Amazon MQ for ActiveMQ ブローカーのデプロイオプション」を参照してください。
5. [Next (次へ)] を選択します。
6. [Configure settings] (設定の定義) ページの [Details] (詳細) セクションで、以下を実行します。

    a. [Broker name] (ブローカー名) を入力します。

> **Important**
>
> 個人を特定できる情報 (PII) などの機密情報や秘匿性の高い情報はタグに追加しないでください。ブローカー名は、CloudWatch Logs を含む他の AWS のサービスからアクセスできます。ブローカー名は、プライベートデータや機密データとして使用することを意図していません。

b. [Broker instance type] (ブローカーインスタンスタイプ) を選択します (mq.m5.large など)。詳細については、「Broker instance types」を参照してください。

7. [ActiveMQ Web Console access] (ActiveMQ ウェブコンソールアクセス) セクションで、[Username] (ユーザーネーム) と [Password] (パスワード) を入力します。ブローカーのユーザー名とパスワードには、以下の制限が適用されます:

   - ユーザーネームに使用できるのは、英数字、ダッシュ、ピリオド、アンダースコア、およびチルデ (- . _ ~) のみです。
   - パスワードは 12 文字以上の長さで、一意の文字を少なくとも 4 つ含める必要があり、カンマ、コロン、または等号 (,:=) は使用できません。

> **Important**
>
> 個人を特定できる情報 (PII) などの機密情報や秘匿性の高い情報はブローカーのユーザー名に追加しないでください。ブローカーのユーザー名は、CloudWatch Logs を含む他の AWS のサービスからアクセスできます。ブローカーのユーザー名は、プライベートデータや機密データとして使用することを意図していません。

ページ上部の緑色のフラッシュバーは、Amazon MQ がリカバリリージョンにレプリカブローカーを作成していることを示しています。ブローカーの CRDR ロールと RPO ステータスも確認できます。[CRDR ロール] 列と [RPO ステータス] 列をオフにするには、[ブローカー] テーブルの右上隅にある歯車アイコンを選択します。次に、[設定] ページで [CRDR ロール] または [RPO ステータス] をオフにします。

ステップ 2: 既存のブローカーのレプリカを作成する

1. Amazon MQ コンソールの [ブローカー] ページで、[レプリカブローカーを作成] を選択します。

2. [プライマリブローカーを選択] ページで、CRDR プライマリブローカーとして使用する既存のブローカーを選択します。次に、[次へ] を選択します。
3. [レプリカブローカーを設定] ページで、ドロップダウンメニューを使用してレプリカリージョンを選択します。
4. [レプリカブローカーのActiveMQ コンソールユーザー] セクションで、レプリカブローカーのコンソールユーザーのユーザー名とパスワードを指定します。ブローカーのユーザー名とパスワードには、以下の制限が適用されます:

   - ユーザーネームに使用できるのは、英数字、ダッシュ、ピリオド、アンダースコア、およびチルデ (- . _ ~) のみです。
   - パスワードは 12 文字以上の長さで、一意の文字を少なくとも 4 つ含める必要があり、カンマ、コロン、または等号 (,:=) は使用できません。

   > **Important**
   >
   > 個人を特定できる情報 (PII) などの機密情報や秘匿性の高い情報はブローカーのユーザー名に追加しないでください。ブローカーのユーザー名は、CloudWatch Logs を含む他の AWS のサービスからアクセスできます。ブローカーのユーザー名は、プライベートデータや機密データとして使用することを意図していません。

5. [ブローカー間のアクセスをブリッジするデータレプリケーションユーザー] セクションで、プライマリブローカーとレプリカブローカーの両方にアクセスするユーザーのユーザー名とパスワードを入力します。ブローカーのユーザー名とパスワードには、以下の制限が適用されます:

   - ユーザーネームに使用できるのは、英数字、ダッシュ、ピリオド、アンダースコア、およびチルデ (- . _ ~) のみです。
   - パスワードは 12 文字以上の長さで、一意の文字を少なくとも 4 つ含める必要があり、カンマ、コロン、または等号 (,:=) は使用できません。

   > **Important**
   >
   > 個人を特定できる情報 (PII) などの機密情報や秘匿性の高い情報はブローカーのユーザー名に追加しないでください。ブローカーのユーザー名は、CloudWatch Logs を含む他の AWS のサービスからアクセスできます。ブローカーのユーザー名は、プライベートデータや機密データとして使用することを意図していません。

その他の設定を行います。次に、[次へ] を選択します。

6. [確認と作成] ページで、レプリカブローカーの詳細を確認します。次に、[レプリカブローカーを作成] を選択します。
7. 次に、プライマリブローカーを再起動します。これにより、レプリカブローカーも再起動されます。ブローカーを再起動する手順については、「Rebooting a Broker」を参照してください。

ActiveMQ ブローカーの追加設定の構成の詳細については、「開始方法: ActiveMQ ブローカーの作成と接続」を参照してください。

Amazon MQ クロスリージョンデータレプリケーションブローカーの削除

プライマリクロスリージョンデータレプリケーション (CRDR) ブローカーまたはレプリカ CRDR ブローカーを削除するには、まずブローカーのペアリングを解除してから、ブローカーを再起動する必要があります。次の手順は、 AWS マネジメントコンソールを使用してブローカーのペアリングを解除および再起動する方法を示しています。

1. [ブローカー] ページで、ペアリングを解除する CRDR ブローカーを選択し、[編集] を選択します。
2. ブローカーの [編集] ページの [データレプリケーション] セクションで、[ブローカーのペアリング解除] を選択します。
3. ポップアップウィンドウに「確認」と入力して、選択内容を確認します。次に、[ブローカーのペアリング解除] を選択します。
4. 次に、ペアリングされていないプライマリブローカーを再起動します。これにより、レプリカブローカーも再起動されます。ブローカーを再起動する手順については、「Rebooting a Broker」を参照してください。プライマリブローカーを再起動すると、両方のブローカーのペアリングが解除され、個別に削除できます。ブローカーを削除するには、「Deleting a broker」を参照してください。

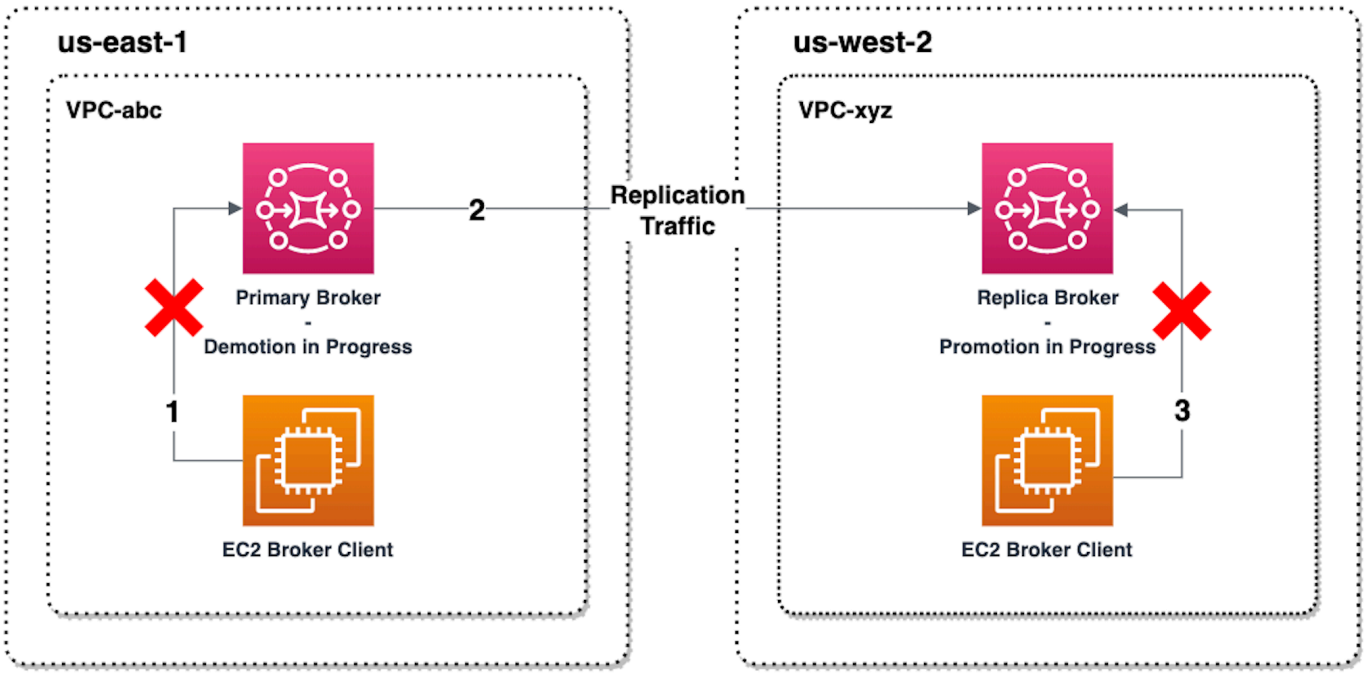
Amazon MQ レプリカブローカーをプライマリブローカーのロールに昇格させるためのスイッチオーバーまたはフェイルオーバーの開始

レプリカブローカーをプライマリブローカーのロールに昇格させる場合は、スイッチオーバーまたはフェイルオーバーを開始できます。レプリカブローカーを昇格させると、プライマリブローカーはレプリカブローカーのロールに降格されます。

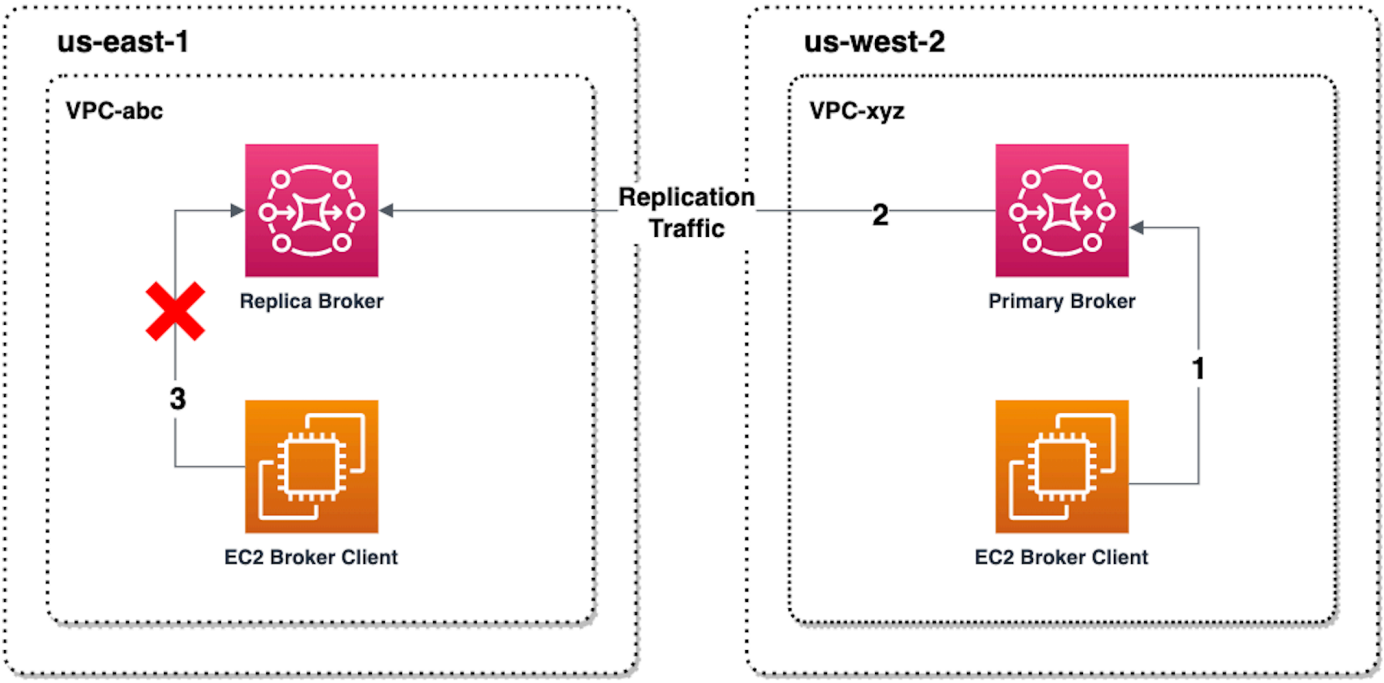
スイッチオーバーでは、可用性よりも一貫性を優先します。このフェイルオーバー操作が完了すると、ブローカーの状態が同じになることが保証されます。スイッチオーバーの場合、ブローカー間の一貫性が確立されるまでは、どちらのブローカーもクライアント接続に使用できない期間が発生する場合があります。レプリカが昇格された時点で、両方のブローカーは同じ状態になります。スイッチオーバーが成功するかどうかは、両方のリージョンの正常性とリージョン間ネットワークの成功にかかっています。

フェイルオーバーでは、一貫性よりも可用性を優先します。この操作の完了時にブローカーが同じ状態になることは保証されません。フェイルオーバーの場合、レプリケーションデータが同期されるまで、またはプライマリがシャットダウン信号を受信するまで待つことなく、レプリカブローカーがすぐにクライアントトラフィックの処理に使用可能になることが保証されます。フェイルオーバーが成功するかどうかは、元のプライマリリージョンの正常性にも、リージョン間ネットワークの成功にも依存しません。

次の図は、レプリケーションキューが空になり、ブローカーの状態が同期されるまで、どちらのブローカーもクライアント接続を受け入れないスイッチオーバーを示しています。このプロセスでは、操作が進行中で、プライマリブローカーがレプリカに降格されるまで、プライマリブローカーの VPC 内のクライアントはそれ以上の状態変更を生成できません。レプリケーションキューが空になり、2 つのブローカーが同じ状態になると、フェイルオーバー操作が完了してレプリカブローカーがプライマリに昇格されるまで、レプリカブローカーの VPC 内のクライアントはレプリカブローカーに接続できません。



次の図は、スイッチオーバープロセスが完了した後のブローカーのステータスを示しています。元のレプリカブローカーがプライマリブローカーのロールに昇格され、クライアント接続を受け入れています。クライアントはブローカーからデータを生成および利用できます。



コンソールを使用してレプリカブローカーを昇格させる

スイッチオーバーまたはフェイルオーバーを使用してレプリカブローカーを昇格させるには、Amazon MQ コンソールで次の手順に従います。

> **Note**
>
> プライマリブローカーではスイッチオーバーやフェイルオーバーを開始できません。

1. レプリカブローカーのリージョンに切り替えます。[ブローカー] テーブルで、プライマリに昇格する既存のレプリカブローカーを選択します。
2. [ブローカーの詳細] ページで、以下の操作を実行します。
    1. [レプリカを昇格させる] を選択します。
    2. ポップアップウィンドウで、[スイッチオーバー] または [フェイルオーバー] を選択します。
    3. テキストボックスに「confirm」と入力し、選択を確定します。

4. [確認] を選択してください。

フェイルオーバーを開始すると、ブローカーのステータスが [フェイルオーバー中] に変わります。フェイルオーバーが完了すると、[ブローカー] ページ上部の青い進行状況バーが緑色になります。

> **Note**
>
> 設定は、レプリカブローカーの作成時にのみレプリケートされます。それ以降の更新はレプリケートされません。

Amazon CloudWatch のクロスリージョンデータレプリケーションのメトリクス

Amazon MQ for ActiveMQ のクロスリージョンデータレプリケーション機能は、プライマリブローカーとレプリカブローカーの信頼性、可用性、パフォーマンスを維持するためのメトリクスを提供します。レプリケーションプロセス中、セカンダリリージョンのレプリカブローカーは、プライマリリージョンのプライマリブローカーから非同期でレプリケートされたデータを受信します。プライマリリージョンのプライマリブローカーに障害が発生した場合、スイッチオーバーまたはフェイルオーバーを開始することで、セカンダリリージョンのレプリカブローカーをプライマリに昇格させることができます。Amazon CloudWatch でメトリックスを表示する手順については、「Amazon MQ 向けの CloudWatch メトリクスへのアクセス」を参照してください。

CRDR のタイムスタンプ

以下のタイムスタンプは、Amazon CloudWatch でのメトリクスの計算方法を示しています。データレプリケーションプロセスには、以下の 5 つのタイムスタンプがあります。

- 現在の観測時刻 (TCO): 現在の瞬間。
- 作成時刻 (TC): プライマリブローカーがレプリケーションキューにイベントを作成した瞬間。プライマリブローカーとレプリカブローカーの両方で利用できます。
- 配信時刻 (TD): イベントがレプリカブローカーに正常に配信された瞬間。レプリカブローカーでのみ利用できます。
- 処理時刻 (TP): レプリカブローカーによってイベントが正常に処理された時刻。レプリカブローカーでのみ利用できます。
- 確認時刻 (TA): プライマリブローカーがイベントを正常に確認した瞬間。プライマリーブローカーでのみ利用できます。

CRDR CloudWatch メトリクスを使用してスイッチオーバー/フェイルオーバーのパフォーマンスを推定する

Amazon MQ は、デフォルトでブローカーのメトリクスを有効にします。Amazon CloudWatch コンソールにアクセスするか、CloudWatch API を使用して、ブローカーのメトリクスを表示できます。以下のメトリクスは、CRDR ブローカーのレプリケーションとスイッチオーバー/フェイルオーバーのパフォーマンスを理解するのに役立ちます。

Amazon MQ CloudWatch メトリクス	CRDR を使用する理由	
`TotalReplicationLag`	プライマリブローカーでの最後の未確認イベントの TA から TC までの推定時間。	
`ReplicationLag`	レプリカブローカーでの最後の未確認イベントの TP から TC までの推定時間。	
`PrimaryWaitTime`	プライマリブローカーで最後に処理されたイベントの TCO から TC までの推定時間。	
`ReplicaWaitTime`	レプリカブローカーで最後に処理されたイベントの TCO から TP までの推定時間。	
`QueueSize`	プライマリブローカーのレプリケーションキューにある未確認イベントの総数。	

`TotalReplicationLag` と `ReplicationLag` は、プライマリブローカーとレプリカブローカーの間の遅延レプリケーションについて説明します。この 2 つのメトリクスを使用して、進行中のスイッチオーバー操作やフェイルオーバー操作が完了するまでの時間を推定することもできます。

`PrimaryWaitTime` と `ReplicaWaitTime` は、レプリケーションプロセスで現在発生している問題を特定するために使用できます。メトリクスの値が絶えず増加している場合は、レプリケーションプロセスのパフォーマンスが低下しているか、一時停止している可能性があります。ネットワークの分

割、ブローカーの起動、長いリカバリなどの問題が原因で、レプリケーションが遅くなることがあります。

ActiveMQ チュートリアル

以下のチュートリアルでは、ActiveMQ ブローカーを作成して接続する方法を説明します。ActiveMQ Java サンプルコードを使用するには、Java Standard Edition Development Kit をインストールして、コードにいくつかの変更を行う必要があります。

トピック

- ブローカーの Amazon MQ ネットワークの作成と設定
- Amazon MQ ブローカーへの Java アプリケーションの接続
- ActiveMQ ブローカーの LDAP との統合
- ステップ 3: (オプション) AWS Lambda 関数に接続する
- ActiveMQ ブローカーユーザーの作成
- ActiveMQ ブローカーユーザーの編集
- ActiveMQ ブローカーユーザーの削除
- ActiveMQ での Java Message Service (JMS) の使用の実用例

ブローカーの Amazon MQ ネットワークの作成と設定

ブローカーのネットワークは、同時にアクティブな複数の単一インスタンスブローカー、またはアクティブ/スタンバイブローカーで構成されています。このチュートリアルでは、ソースとシンクトポロジを使用してブローカーの 2 ブローカーネットワークを作成する方法を学びます。

概念的な概要および詳細な設定情報については、以下を参照してください。

- Amazon MQ のブローカーのネットワーク
- ブローカーのネットワークを正しく設定する
- `networkConnector`
- `networkConnectionStartAsync`
- ActiveMQ ドキュメントの「ブローカーのネットワーク」

ブローカーの Amazon MQ ネットワークは、Amazon MQ コンソールを使用して作成できます。2 つのブローカーの作成を並行して開始できるため、このプロセスには約 15 分かかります。

トピック

前提条件

ブローカーのネットワークを作成するには、以下のものが必要です。

- 同時にアクティブな 2 つ以上のブローカー (このチュートリアルでは `MyBroker2` および `MyBroker1` という名前)。ブローカー作成についての詳細は、「開始方法: ActiveMQ ブローカーの作成と接続」 を参照してください。
- 2 つのブローカーは、同じ VPC またはピア接続された VPC に属している必要があります。VPC の詳細については、Amazon VPC ユーザーガイドの「Amazon VPC とは」および Amazon VPC ピアリングガイドの「VPC ピア機能とは」を参照してください。

> **Important**
>
> デフォルトの VPC、サブネット、またはセキュリティグループがない場合は、それらを最初に作成する必要があります。詳細については、Amazon VPC ユーザーガイドの以下のトピックを参照してください。
>
> - デフォルト VPC の作成
> - デフォルトサブネットの作成
> - セキュリティグループを作成する

- 両方のブローカーに対して同じサインイン認証情報を持つ 2 人のユーザー。ユーザー作成の詳細については、「ActiveMQ ブローカーユーザーの作成」を参照してください。

> **Note**
>
> LDAP 認証をブローカーのネットワークと統合するときは、ユーザーが ActiveMQ ブローカーと LDAP ユーザーの両方として存在することを確認してください。

以下の例では、2 つの単一インスタンスブローカーを使用します。ただし、アクティブ/スタンバイブローカー、またはブローカーデプロイモードの組み合わせを使用してブローカーのネットワークを作成できます。

ステップ 1: ブローカー間のトラフィックを許可する

ブローカーを作成した後、それらの間のトラフィックを許可する必要があります。

1. Amazon MQ コンソールの [MyBroker2] ページにある [Details] (詳細) セクションの [Security and network] (セキュリティとネットワーク) で、セキュリティグループの名前または

をクリックします。

   EC2 ダッシュボードの [セキュリティグループ] ページが表示されます。
2. セキュリティグループのリストから、セキュリティグループを選択します。
3. ページ下部で、[インバウンド] を選択し、次に [編集] を選択します。
4. [インバウンドルールの編集] ダイアログボックスで、OpenWire エンドポイントのルールを追加します。

   a. [ルールの追加] を選択します。

   b. [タイプ] で、[カスタム TCP] を選択します。

   c. [ポート範囲] で、OpenWire ポートを入力します (61617)。

   d. 次のいずれかを行います :

   - 特定の IP アドレスへのアクセスを制限する場合は、[ソース] で [カスタム] を選択したままにし、`MyBroker1` の IP アドレスに続いて `/32` を入力します。(これは IP アドレスを有効な CIDR レコードに変換します)。詳細については、「Elastic Network Interfaces」を参照してください。

     > **Tip**
     >
     > `MyBroker1` の IP アドレスを取得するには、Amazon MQ コンソールでブローカーの名前を選択し、[Details] (詳細) セクションに移動します。

   - すべてのブローカーがプライベートで、同じ VPC に属している場合は、[ソース] で、[カスタム] を選択したままにし、編集しているセキュリティグループの ID を入力します。

> **Note**
>
> パブリックブローカーの場合は、IP アドレスを使用してアクセスを制限する必要があります。

e. [保存] を選択します。

これで、ブローカーはインバウンド接続を受け入れることができます。

ステップ 2: ブローカー用のネットワークコネクターを設定する

ブローカー間のトラフィックを許可すると、そのうちの 1 つのネットワーク接続を設定する必要があります。

1. ブローカー `MyBroker1` の設定リビジョンを編集します。

   a. [MyBroker1] ページで、[編集] を選択します。

   b. [MyBroker1 の編集] ページの、[設定] セクションで、[表示] を選択します。

   設定が使用するブローカーエンジンタイプとバージョン (例: [Apache ActiveMQ 5.15.0]) が表示されます。

   c. [Configuration details] タブに、設定リビジョン番号、説明、およびブローカー設定が XML 形式で表示されます。

   d. [設定の編集] を選択します。

   e. 設定ファイルの下部で、`<networkConnectors>` セクションのコメントを解除し、以下の情報を入力します。

   - ネットワークコネクターの `name`。
   - ブローカーの両方に共通の ActiveMQ ウェブコンソール`username`。
   - `duplex` 接続を有効にします。
   - 次のいずれかを行います :
     - ブローカーを単一インスタンスブローカーに接続している場合は、`MyBroker2` の `static:` プレフィックスと OpenWire エンドポイント `uri` を使用します。例:

```
<networkConnectors>
```

```
  <networkConnector name="connector_1_to_2" userName="myCommonUser"
 duplex="true"
    uri="static:(ssl://b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-
east-2.amazonaws.com:61617)"/>
</networkConnectors>
```

- ブローカーをアクティブ/スタンバイブローカーに接続している場合は、次のクエリパラメータ `?randomize=false&maxReconnectAttempts=0` を使用して、両方のブローカーの `static+failover` トランスポートと OpenWire エンドポイント `uri` を使用します。例:

```
<networkConnectors>
  <networkConnector name="connector_1_to_2" userName="myCommonUser"
 duplex="true"
    uri="static:(failover:(ssl://
b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-east-2.amazonaws.com:61617,
    ssl://b-9876l5k4-32ji-109h-8gfe-7d65c4b132a1-2.mq.us-
west-2.amazonaws.com:61617)?randomize=false&amp;maxReconnectAttempts=0)"/>
</networkConnectors>
```

> **Note**
>
> ActiveMQ ユーザーのサインイン認証情報は含めないでください。

f. [保存] を選択します。

g. [リビジョンの保存] ダイアログボックスで、「`Add network of brokers connector for MyBroker2`」と入力します。

h. [保存] を選択して設定リビジョンを保存します。

2. `MyBroker1` を編集して最新の設定リビジョンをすぐに適用するように設定します。

a. [MyBroker1] ページで、[編集] を選択します。

b. [MyBroker1 の編集] ページの、[設定] セクションで、[Schedule Modifications (スケジュールの変更)] を選択します。

c. [Schedule broker modifications (ブローカー変更のスケジュール)] セクションで、変更を適用するには、[即時] を選択します。

d. [Apply] (適用) を選択します。

`MyBroker1` が再起動され、設定リビジョンが適用されます。

ネットワークのブローカーが作成されます。

次のステップ

ブローカーのネットワークを設定したら、メッセージを作成して消費することでテストできます。

> **Important**
>
> ポート 8162 (ActiveMQ Web コンソール用) とポート 61617 (OpenWire エンドポイント用) で、ブローカー `MyBroker1` に対してローカルマシンからのインバウンド接続を有効化しておくようにしてください。
> プロデューサーとコンシューマーがブローカーのネットワークに接続できるように、セキュリティグループの設定を調整する必要がある場合があります。

1. Amazon MQ コンソールで [Connections] (接続) セクションに移動し、ブローカー `MyBroker1` の ActiveMQ ウェブコンソールエンドポイントをメモします。
2. ブローカー `MyBroker1` の ActiveMQ ウェブコンソールに移動します。
3. ネットワークブリッジが接続されていることを確認するには、[ネットワーク] を選択します。

   [Network Bridges] (ネットワークブリッジ) セクションで、`MyBroker2` の名前とアドレスが [Remote Broker] (リモートブローカー) と [Remote Address] (リモートアドレス) の列にリストされます。
4. ブローカー `MyBroker2` にアクセスできる任意のマシンから、コンシューマーを作成します。以下はその例です。

   ```
   activemq consumer --brokerUrl "ssl://
   b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-east-2.amazonaws.com:61617" \
    --user commonUser \
    --password myPassword456 \
    --destination queue://MyQueue
   ```

   コンシューマーは `MyBroker2` の OpenWire エンドポイントに接続し、キュー `MyQueue` からメッセージを消費し始めます。
5. ブローカー `MyBroker1` にアクセスできる任意のマシンから、プロデューサーを作成し、いくつかのメッセージを送信します。以下はその例です。

```
activemq producer --brokerUrl "ssl://
b-9876l5k4-32ji-109h-8gfe-7d65c4b132a1-1.mq.us-east-2.amazonaws.com:61617" \
 --user commonUser \
 --password myPassword456 \
 --destination queue://MyQueue \
 --persistent true \
 --messageSize 1000 \
 --messageCount 10000
```

プロデューサーは MyBroker1 の OpenWire エンドポイントに接続し、キュー MyQueue に永続的なメッセージを生成し始めます。

Amazon MQ ブローカーへの Java アプリケーションの接続

Amazon MQ ActiveMQ ブローカーを作成したら、ブローカーにアプリケーションを接続できます。以下の例では、Java Message Service (JMS) を使用してブローカーへの接続を作成し、キューを作成して、メッセージを送信する方法を説明します。完全な Java の実用例については、「Working Java Example」を参照してください。

ActiveMQ ブローカーには、さまざまな ActiveMQ クライアントを使用して接続できます。ActiveMQ クライアントを使用することをお勧めします。

トピック

- 前提条件
- メッセージプロデューサーを作成してメッセージを送信する
- メッセージコンシューマーを作成してメッセージを受信する

前提条件

VPC 属性 を有効にする

VPC 内でブローカーにアクセスできることを確実にするには、enableDnsHostnames および enableDnsSupport VPC 属性を有効にする必要があります。詳細については、Amazon VPC ユーザーガイドの「VPC の DNS サポート」を参照してください。

インバウンド接続を有効にする

次に、アプリケーションのインバウンド接続を有効にします。

1. Amazon MQ コンソールにサインインします。
2. ブローカーのリストからブローカーの名前 (MyBroker など) を選択します。
3. [***MyBroker***] ページの [Connections] (接続) セクションで、ブローカーのウェブコンソール URL とワイヤレベルプロトコルのアドレスとポートをメモします。
4. [Details] (詳細) セクションの [Security and network] (セキュリティとネットワーク) で、セキュリティグループの名前または
   
   をクリックします。

   EC2 ダッシュボードの [セキュリティグループ] ページが表示されます。
5. セキュリティグループのリストから、セキュリティグループを選択します。
6. ページ下部で、[インバウンド] を選択し、次に [編集] を選択します。
7. [Edit inbound rules] (インバウンドルールの編集) ダイアログボックスで、パブリックアクセスを許可する URL またはエンドポイントごとにルールを追加します (以下の例は、これをブローカーのウェブコンソールに対して行う方法を説明しています)。

   a. [ルールの追加] を選択します。
   b. [タイプ] で、[カスタム TCP] を選択します。
   c. [Port Range] (ポート範囲) にはウェブコンソールポート (8162) を入力します。
   d. [Source] (ソース) では、[Custom] (カスタム) が選択された状態のままにしておき、ウェブコンソールにアクセスできるようにするシステムの IP アドレスを入力します (`192.0.2.1` など)。
   e. [Save] (保存) をクリックします。

      これで、ブローカーはインバウンド接続を受け入れることができます。

Java の依存関係を追加する

`activemq-client.jar` パッケージと `activemq-pool.jar` パッケージを Java クラスパスに追加します。以下の例は、Maven プロジェクトの `pom.xml` ファイルにあるこれらの依存関係を示しています。

```
<dependencies>
    <dependency>
        <groupId>org.apache.activemq</groupId>
        <artifactId>activemq-client</artifactId>
```

```
        <version>5.15.16</version>
    </dependency>
    <dependency>
        <groupId>org.apache.activemq</groupId>
        <artifactId>activemq-pool</artifactId>
        <version>5.15.16</version>
    </dependency>
</dependencies>
```

`activemq-client.jar` の詳細については、Apache ActiveMQ ドキュメントの「Initial Configuration」を参照してください。

> **Important**
>
> 以下のコード例では、プロデューサーとコンシューマーが単一のスレッド内で実行されます。実稼働システム (またはブローカーインスタンスのフェイルオーバーをテストする) には、プロデューサーとコンシューマーが個別のホストまたはスレッドで実行されるようにしてください。

メッセージプロデューサーを作成してメッセージを送信する

メッセージプロデューサーを作成してメッセージを受信するには、次の手順に従います。

1. ブローカーのエンドポイントを使用してメッセージプロデューサーの JMS プール接続ファクトリを作成してから、ファクトリに対して `createConnection` メソッドを呼び出します。

   > **Note**
   >
   > アクティブ/スタンバイブローカーの場合、Amazon MQ は 2 つの ActiveMQ ウェブコンソール URL を提供しますが、一度に 1 つの URL しかアクティブになりません。同様に、Amazon MQ はワイヤレベルプロトコルごとに 2 つのエンドポイントを提供しますが、ペアごとに一度に 1 つのエンドポイントしかアクティブになりません。`-1` および `-2` サフィックスは冗長ペアを表します。詳細については、「Amazon MQ for ActiveMQ ブローカーのデプロイオプション」を参照してください。
   > ワイヤレベルのプロトコルエンドポイントの場合、フェイルオーバートランスポートを使用してアプリケーションがいずれかのエンドポイントに接続できるようにする必要があります。

```
// Create a connection factory.
final ActiveMQConnectionFactory connectionFactory = new
 ActiveMQConnectionFactory(wireLevelEndpoint);

// Pass the sign-in credentials.
connectionFactory.setUserName(activeMqUsername);
connectionFactory.setPassword(activeMqPassword);

// Create a pooled connection factory.
final PooledConnectionFactory pooledConnectionFactory = new
 PooledConnectionFactory();
pooledConnectionFactory.setConnectionFactory(connectionFactory);
pooledConnectionFactory.setMaxConnections(10);

// Establish a connection for the producer.
final Connection producerConnection = pooledConnectionFactory.createConnection();
producerConnection.start();

// Close all connections in the pool.
pooledConnectionFactory.clear();
```

> **Note**
>
> メッセージプロデューサーは、常に PooledConnectionFactory クラスを使用する必要があります。詳細については、「常に接続プールを使用する」を参照してください。

2. セッション、MyQueue という名前のキュー、およびメッセージプロデューサーを作成します。

```
// Create a session.
final Session producerSession = producerConnection.createSession(false,
 Session.AUTO_ACKNOWLEDGE);

// Create a queue named "MyQueue".
final Destination producerDestination = producerSession.createQueue("MyQueue");

// Create a producer from the session to the queue.
final MessageProducer producer =
 producerSession.createProducer(producerDestination);
producer.setDeliveryMode(DeliveryMode.NON_PERSISTENT);
```

3. メッセージ文字列 "Hello from Amazon MQ!" を作成してから、メッセージを送信します。

```
// Create a message.
final String text = "Hello from Amazon MQ!";
TextMessage producerMessage = producerSession.createTextMessage(text);

// Send the message.
producer.send(producerMessage);
System.out.println("Message sent.");
```

4. プロデューサーをクリーンアップします。

```
producer.close();
producerSession.close();
producerConnection.close();
```

メッセージコンシューマーを作成してメッセージを受信する

メッセージプロデューサーを作成してメッセージを受信するには、次の手順に従います。

1. ブローカーのエンドポイントを使用してメッセージプロデューサーの JMS 接続ファクトリを作成してから、ファクトリに対して createConnection メソッドを呼び出します。

```
// Create a connection factory.
final ActiveMQConnectionFactory connectionFactory = new
 ActiveMQConnectionFactory(wireLevelEndpoint);

// Pass the sign-in credentials.
connectionFactory.setUserName(activeMqUsername);
connectionFactory.setPassword(activeMqPassword);

// Establish a connection for the consumer.
final Connection consumerConnection = connectionFactory.createConnection();
consumerConnection.start();
```

> **Note**
>
> メッセージコンシューマーには、PooledConnectionFactory クラスを一切使用しないでください。詳細については、「常に接続プールを使用する」を参照してください。

2. セッション、MyQueue という名前のキュー、およびメッセージコンシューマーを作成します。

```
// Create a session.
final Session consumerSession = consumerConnection.createSession(false,
 Session.AUTO_ACKNOWLEDGE);

// Create a queue named "MyQueue".
final Destination consumerDestination = consumerSession.createQueue("MyQueue");

// Create a message consumer from the session to the queue.
final MessageConsumer consumer =
 consumerSession.createConsumer(consumerDestination);
```

3. メッセージの待機を開始し、メッセージの到着時にメッセージを受信します。

```
// Begin to wait for messages.
final Message consumerMessage = consumer.receive(1000);

// Receive the message when it arrives.
final TextMessage consumerTextMessage = (TextMessage) consumerMessage;
System.out.println("Message received: " + consumerTextMessage.getText());
```

> **Note**
>
> AWS メッセージングサービス (Amazon SQS など) とは異なり、コンシューマーは常にブローカーに接続されます。

4. コンシューマー、セッション、および接続を閉じます。

```
consumer.close();
consumerSession.close();
consumerConnection.close();
```

ActiveMQ ブローカーの LDAP との統合

> **Important**
>
> RabbitMQ ブローカーでは LDAP 統合はサポートされません。

Amazon MQ では、プライベート CA によって発行されたサーバー証明書はサポートされません。

ActiveMQ ブローカーには、TLS が有効化されている以下のプロトコルを使用してアクセスできます。

- AMQP
- MQTT
- MQTT over WebSocket
- OpenWire
- STOMP
- STOMP over WebSocket

Amazon MQ では、ユーザー許可の管理に、ネイティブ ActiveMQ 認証か LDAP 認証と認可のどちらかを選択できます。ActiveMQ のユーザー名とパスワードに関する制限の詳細については、「[ユーザー]」を参照してください。

ActiveMQ のユーザーおよびグループによるキューとトピックの使用を認可するには、ブローカーの設定を編集する必要があります。Amazon MQ は、ActiveMQ の Simple Authentication Plugin を使用して、送信先に対する読み込みと書き込みを制限します。詳細情報と例については、「認可マップを常に設定する」および「`authorizationEntry`」を参照してください。

> **Note**
>
> 現在、Amazon MQ はクライアント証明書認証をサポートしていません。

トピック

- LDAP を ActiveMQ に統合する
- 前提条件
- LDAP の使用開始
- LDAP 統合の仕組み

LDAP を ActiveMQ に統合する

Amazon MQ ユーザーは、Lightweight Directory Access Protocol (LDAP) サーバーに保存されている認証情報を使用して認証することができます。これを使用して、Amazon MQ ユーザーの追加、削除、変更、およびトピックとキューへの許可の割り当てを行うことも可能です。ブローカーの作成、更新、および削除といった管理操作には引き続き IAM 認証情報が必要となり、これらは LDAP と統合されません。

LDAP サーバーを使用した Amazon MQ ブローカーの認証と認可の簡素化と一元化を希望するお客様は、この機能を使用できます。すべてのユーザー認証情報を LDAP サーバーに保存することにより、これらの認証情報を保存して管理する一元的な場所が提供されるため、時間と労力を節約できます。

Amazon MQ は、Apache ActiveMQ JAAS プラグインを使用して LDAP サポートを提供します。このプラグインがサポートする LDAP サーバー (Microsoft Active Directory や OpenLDAP など) ならば、Amazon MQ でもサポートされます。プラグインの詳細については、ActiveMQ ドキュメントの「Security」セクションを参照してください。

ユーザーに加えて、特定のグループまたはユーザーのトピックとキューへのアクセスも、LDAP サーバー経由で指定できます。これは、LDAP サーバーでトピックとキューを表すエントリを作成してから、特定の LDAP ユーザーまたはグループに許可を割り当てることで実行します。その後、LDAP サーバーから認可データを取得するようにブローカーを設定できます。

> **Important**
>
> LDAP を使用する場合、認証では大文字と小文字は区別されませんが、ユーザー名では認可では大文字と小文字が区別されます。

前提条件

新規または既存の Amazon MQ ブローカーに LDAP サポートを追加する前に、サービスアカウントをセットアップする必要があります。このサービスアカウントは、LDAP サーバーへの接続を開始するために必要で、この接続を行うために適切な許可を持っている必要があります。このサービスアカウントは、ブローカーの LDAP 認証をセットアップします。後続のクライアント接続は、いずれも同じ接続を介して認証されます。

サービスアカウントは、接続を開始するためのアクセス権を持つ LDAP サーバー内のアカウントです。これは標準の LDAP 要件であり、サービスアカウントの認証情報を提供する必要があるの

は 1 度だけです。接続がセットアップされると、その後のすべてのクライアント接続が LDAP サーバー経由で認証されます。サービスアカウントの認証情報は暗号化された形態でセキュアに保存され、Amazon MQ 以外はアクセスできません。

ActiveMQ との統合には、LDAP サーバーに特定のディレクトリ情報ツリー (DIT) が必要です。この構造を明確に示すサンプル `ldif` ファイルについては、ActiveMQ ドキュメントの「Security」セクションで「Import the following LDIF file into the LDAP server」を参照してください。

LDAP の使用開始

使用を開始するには、Amazon MQ コンソールに移動し、新しい Amazon MQ ブローカーインスタンスの作成時または既存のブローカーインスタンスの編集時に [LDAP 認証と認可] を選択します。

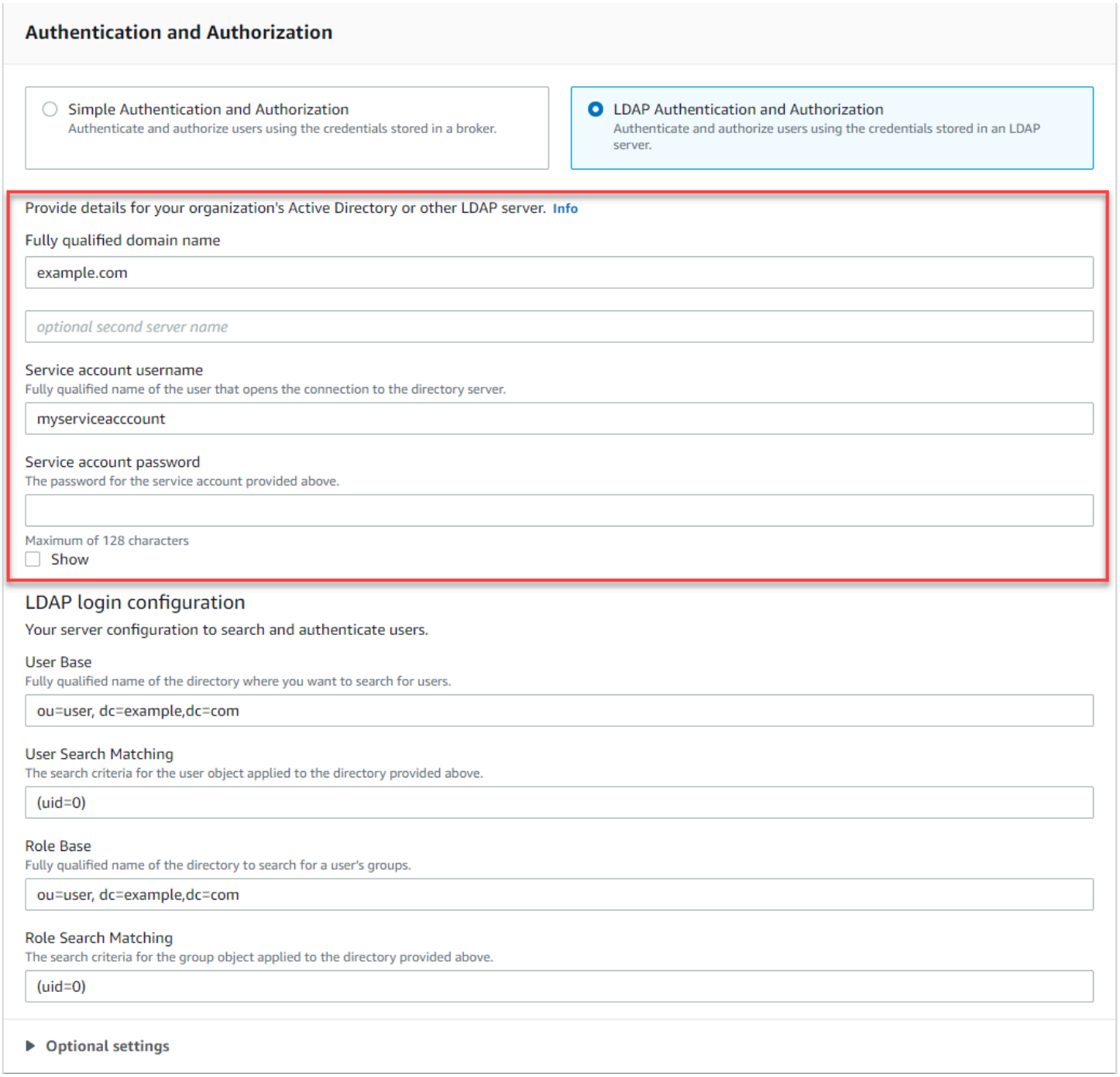
サービスアカウントに関する以下の情報を入力します。

- [完全修飾ドメイン名] 認証リクエストと認可リクエストを発行する先の LDAP サーバーの場所です。

  > **Note**
  >
  > 入力する LDAP サーバーの完全修飾ドメイン名には、プロトコルまたはポート番号を含めないでください。Amazon MQ は、完全修飾ドメイン名の先頭にプロトコル `ldaps` を付加し、末尾にポート番号 636 を付加します。
  > 例えば、`example.com` という完全修飾ドメインを指定する場合、Amazon MQ は URL `ldaps://example.com:636` を使用して LDAP サーバーにアクセスします。
  > ブローカーホストが LDAP サーバーと正常に通信できるようにするには、完全修飾ドメイン名がパブリックに解決可能である必要があります。LDAP サーバーをプライベートかつセキュアに保つには、サーバーのインバウンドルールでインバウンドトラフィックを制限して、ブローカーの VPC 内からのトラフィックのみを許可します。

- Service account username (サービスアカウントのユーザーネーム) LDAP サーバーへの初期バインドを実行するために使用されるユーザーの識別名です。
- Service account password (サービスアカウントのパスワード) 初期バインドを実行するユーザーのパスワードです。

以下の画像では、これらの詳細情報を指定する場所が強調されています。



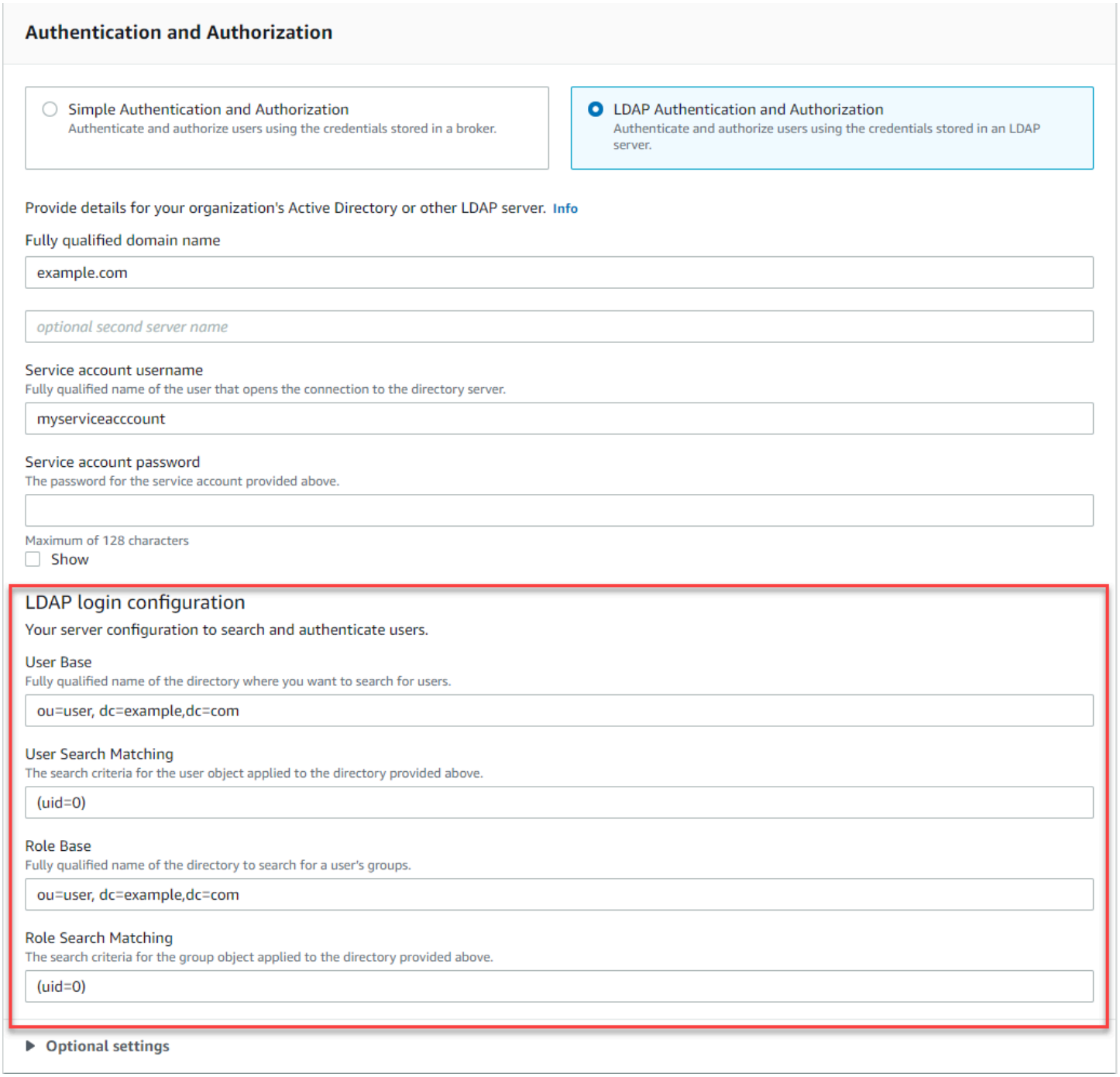
[LDAP login configuration] (LDAP ログイン設定) セクションで、以下の必須情報を入力します。

- User Base (ユーザーベース) ユーザーの検索先となる、ディレクトリ情報ツリー (DIT) 内のノードの識別名です。
- User Search Matching (ユーザー検索のマッチング) `userBase` 内のユーザーを検索するために使用される LDAP 検索フィルターです。検索フィルターの `{0}` プレースホルダーにはクライアント

のユーザーネームが代入されます。詳細については、「認証」および「Authorization」を参照してください。

- Role Base (ロールベース) ロールの検索先となる、DIT 内のノードの識別名です。ロールは、ディレクトリ内の明示的な LDAP グループエントリとして設定できます。一般的なロールエントリは、ロール名の 1 つの属性 (共通名 (CN)など)、もう一つの属性 (`member` など)、およびロールグループに属するユーザーの識別名またはユーザーネームを表す値で構成することができます。例えば、組織単位 `group` がある場合には、識別名 `ou=group,dc=example,dc=com` を指定できます。
- Role Search Matching (ロール検索のマッチング) `roleBase` 内のロールを検索するために使用される LDAP 検索フィルターです。検索フィルターの `{0}` プレースホルダーには、`userSearchMatching` に一致するユーザーの識別名が代入されます。`{1}` プレースホルダーには、クライアントのユーザーネームが代入されます。例えば、ディレクトリ内のロールエントリに `member` という名前の属性が含まれ、そのロール内のすべてのユーザーのユーザーネームが含められている場合は、検索フィルター (`member:=uid={1}`) を指定できます。

以下の画像では、これらの詳細情報を指定する場所が強調されています。



[Optional settings] (オプション設定) セクションでは、以下のオプション情報を指定できます。

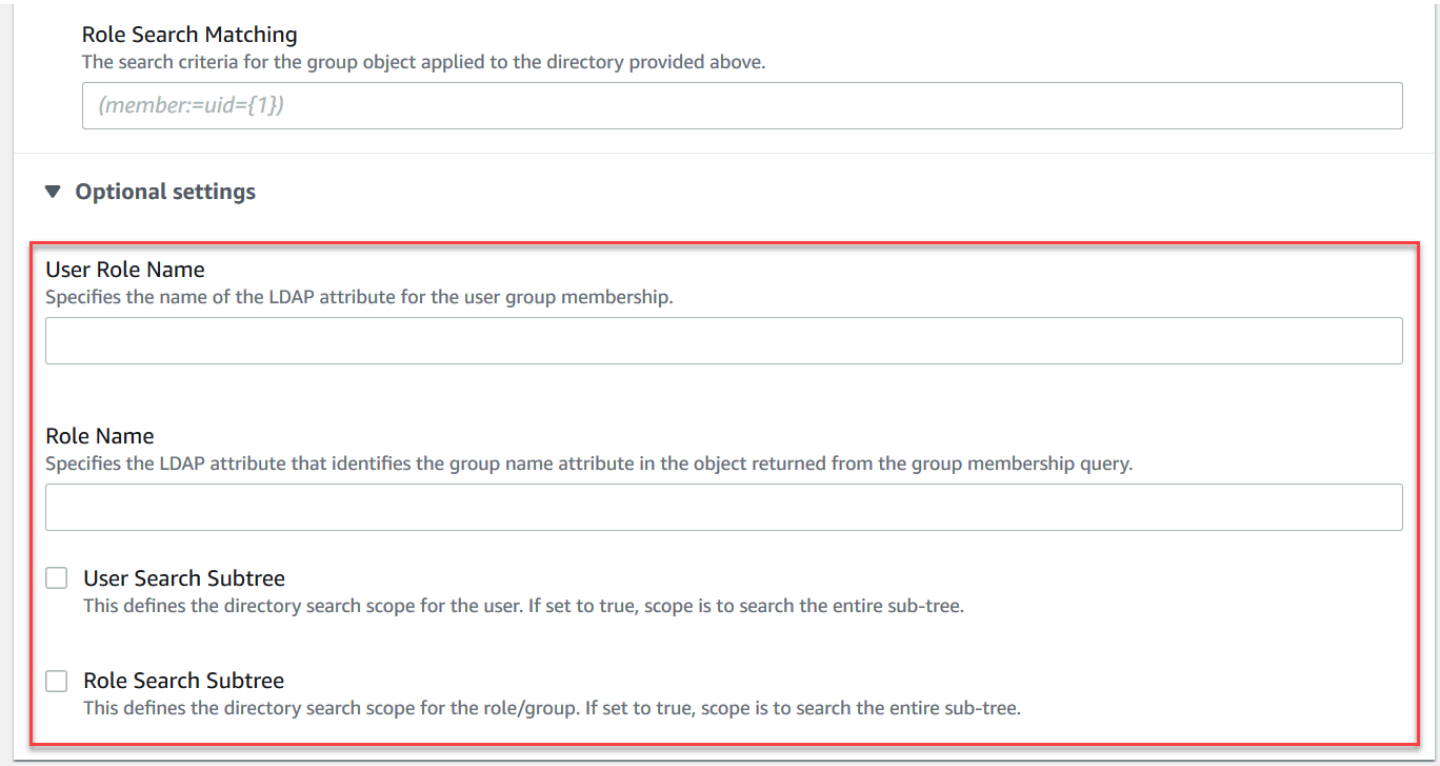
- User Role Name (ユーザーロール名) ユーザーのグループメンバーシップに関するユーザーのディレクトリエントリ内の LDAP 属性の名前です。場合によっては、ユーザーのディレクトリエントリ内の属性の値によって、ユーザーロールを識別できることもあります。`userRoleName` オプションは、この属性の名前を指定することを可能にします。例えば、以下のユーザーエントリについて考えてみましょう。

```
dn: uid=jdoe,ou=user,dc=example,dc=com
objectClass: user
uid: jdoe
sn: jane
cn: Jane Doe
mail: j.doe@somecompany.com
memberOf: role1
userPassword: password
```

上記の例に正しい userRoleName を提供するには、memberOf 属性を指定します。認証が成功すると、ユーザーにロール role1 が割り当てられます。

- Role Name (ロール名) ロールエントリ内のグループ名属性で、値がそのロールの名前になってます。例えば、グループエントリの共通名には cn を指定できます。認証が成功すると、ユーザーには、メンバーになっている各ロールエントリの cn 属性の値が割り当てられます。
- User Search Subtree (ユーザー検索サブツリー) LDAP ユーザー検索クエリの範囲を定義します。true の場合、userBase によって定義されたノード下にあるサブツリー全体を検索するように範囲が設定されます。
- Role Search Subtree (ロール検索サブツリー) LDAP ロール検索クエリの範囲を定義します。true の場合、roleBase によって定義されたノード下にあるサブツリー全体を検索するように範囲が設定されます。

以下の画像では、これらのオプション設定を指定する場所が強調されています。



LDAP 統合の仕組み

統合は、認証の構造と認可の構造という 2 つの主要カテゴリに分けて考えることができます。

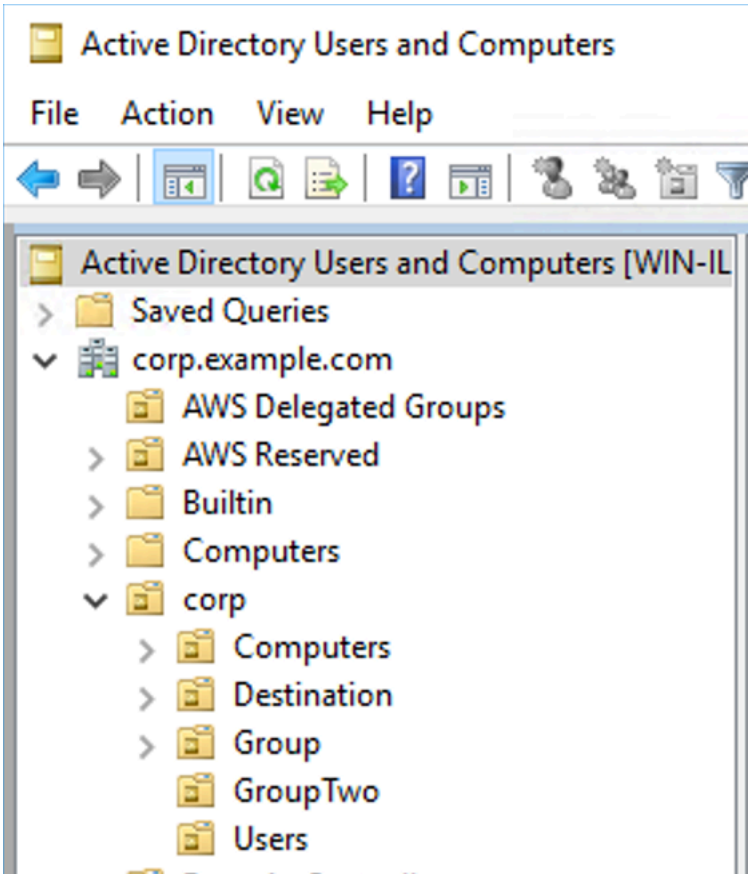
認証

認証では、クライアントの認証情報が有効である必要があります。これらの認証情報は、LDAP サーバーのユーザベース内のユーザーに対して検証されます。

ActiveMQ ブローカーに提供されるユーザーベースは、LDAP サーバーでユーザーが保存されている DIT 内のノードをポイントしている必要があります。たとえば、 を使用していて AWS Managed Microsoft AD、ドメインコンポーネント `corp`、`example`、および `com`、組織単位 `corp` と があるコンポーネント内には`Users`、ユーザーベースとして以下を使用します。

```
OU=Users,OU=corp,DC=corp,DC=example,DC=com
```

ActiveMQ ブローカーは、ブローカーに対するクライアント接続リクエストを認証するために、DIT 内のこの場所でユーザーを検索します。



ActiveMQ ソースコードは、ユーザーの属性名を `uid` にハードコードするため、各ユーザーにこの属性セットがあることを確認する必要があります。簡略化のため、ユーザーの接続ユーザーネームを使用できます。詳細については、activemq ソースコードと「Configuring ID mappings in Active Directory Users and Computers for Windows Server 2016 (and subsequent) versions」を参照してください。

特定のユーザーに対して ActiveMQ コンソールアクセスを有効にするには、ユーザーが `amazonmq-console-admins` グループに属していることを確認してください。

Authorization

認可のため、ブローカーの設定に許可の検索ベースが指定されています。認可は、ブローカーの `activemq.xml` 設定ファイルにある `cachedLdapAuthorizationMap` 要素を通じて、送信先ごと (またはワイルドカード、送信先セット) に行われます。詳細については、「Cached LDAP Authorization Module」を参照してください。

> **Note**
>
> ブローカーactivemq.xmlの設定ファイルで cachedLDAPAuthorizationMap要素を使用するには、 を使用して設定を作成する AWS Management Consoleときに LDAP 認証および認可オプションを選択するか、Amazon MQ API を使用して新しい設定を作成するLDAPときに authenticationStrategyプロパティを に設定する必要があります。

cachedLDAPAuthorizationMap 要素の一部として、以下の 3 つの属性を指定する必要があります。

- queueSearchBase
- topicSearchBase
- tempSearchBase

> **Important**
>
> ブローカーの設定ファイルに機密情報が直接配置されることを防ぐため、Amazon MQ は cachedLdapAuthorizationMap での以下の属性の使用をブロックします。
>
> - connectionURL
> - connectionUsername
> - connectionPassword
>
> ブローカーを作成すると、Amazon MQ は、 を介して AWS Management Console、または API リクエストの ldapServerMetadataプロパティで指定した値を上記の属性に置き換えます。

以下は、cachedLdapAuthorizationMap の実用例です。

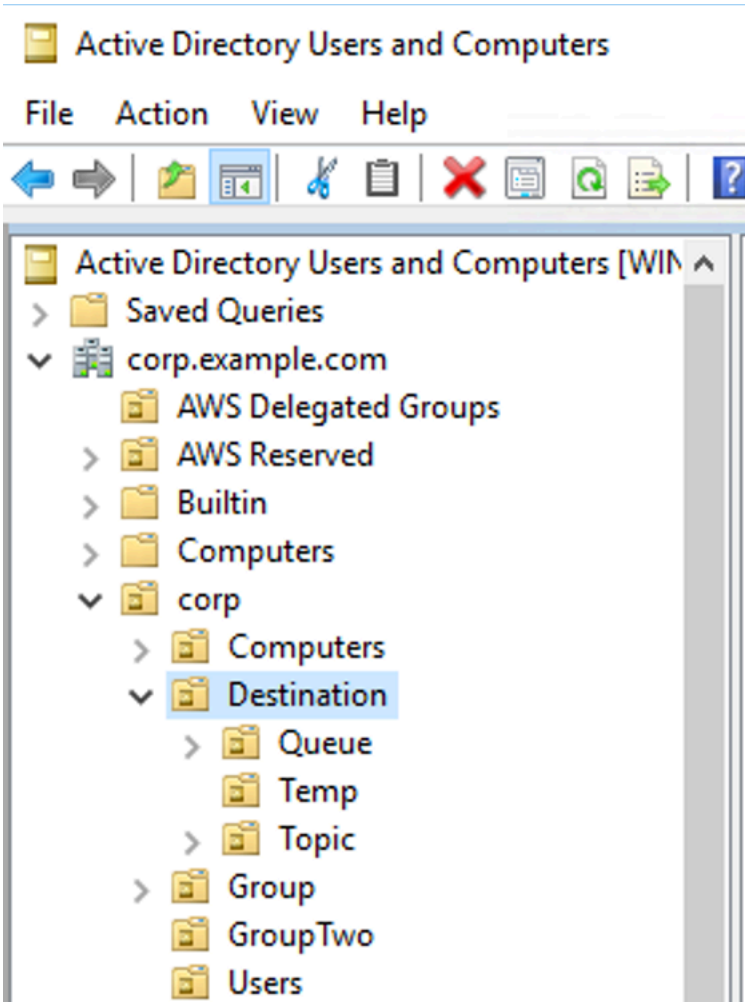
```
<authorizationPlugin>
    <map>
        <cachedLDAPAuthorizationMap
            queueSearchBase="ou=Queue,ou=Destination,ou=corp,dc=corp,dc=example,dc=com"
            topicSearchBase="ou=Topic,ou=Destination,ou=corp,dc=corp,dc=example,dc=com"
            tempSearchBase="ou=Temp,ou=Destination,ou=corp,dc=corp,dc=example,dc=com"
```

```
            refreshInterval="300000"
            legacyGroupMapping="false"
        />
    </map>
</authorizationPlugin>
```

これらの値は、送信先の各タイプに対する許可が指定されている、DIT 内の場所を特定します。したがって、上記の例では AWS Managed Microsoft AD、、corp、exampleおよび の同じドメインコンポーネントを使用してcom、すべての送信先タイプを含むdestinationように という名前の組織単位を指定します。その OU 内で、queues、topics、および temp の各送信先の OU を作成します。

これは、Queue タイプの送信先の認可情報を提供するキュー検索ベースの場所が、DIT 内の以下の場所になることを意味します。

```
OU=Queue,OU=Destination,OU=corp,DC=corp,DC=example,DC=com
```



同様に、Topics および Temp 送信先の許可ルールの場所も、DIT 内の同じレベルになります。

```
OU=Topic,OU=Destination,OU=corp,DC=corp,DC=example,DC=com
OU=Temp,OU=Destination,OU=corp,DC=corp,DC=example,DC=com
```

各送信先タイプ (Queue、Topic、Temp) の OU 内には、ワイルドカードまたは特定の送信先名を指定できます。例えば、プレフィックス DEMO.EVENTS.$. で始まるすべてのキューの認可ルールを提供するには、以下の OU を作成できます。

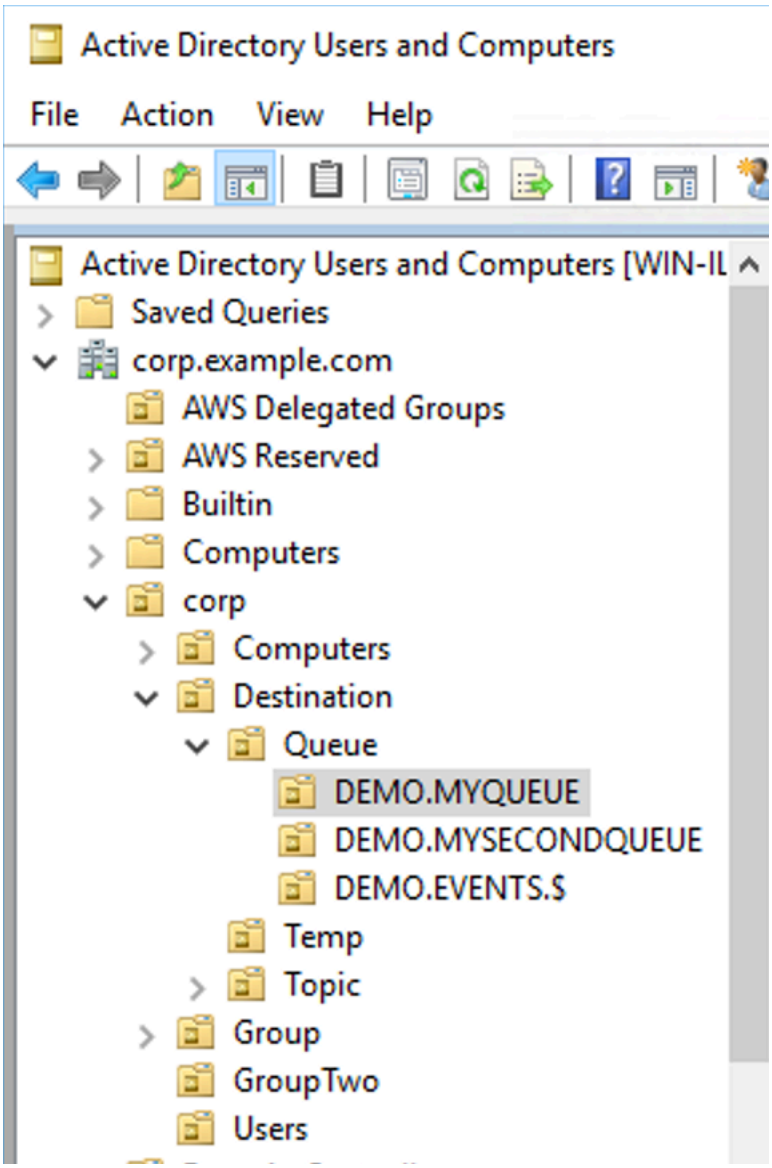
```
OU=DEMO.EVENTS.$,OU=Queue,OU=Destination,OU=corp,DC=corp,DC=example,DC=com
```

> **Note**
>
> DEMO.EVENTS.$ OU は Queue OU 内にあります。

ActiveMQ でのワイルドカードの詳細については、「Wildcards」を参照してください。

DEMO.MYQUEUE などの特定のキューの認可ルールを提供するには、以下のように指定します。

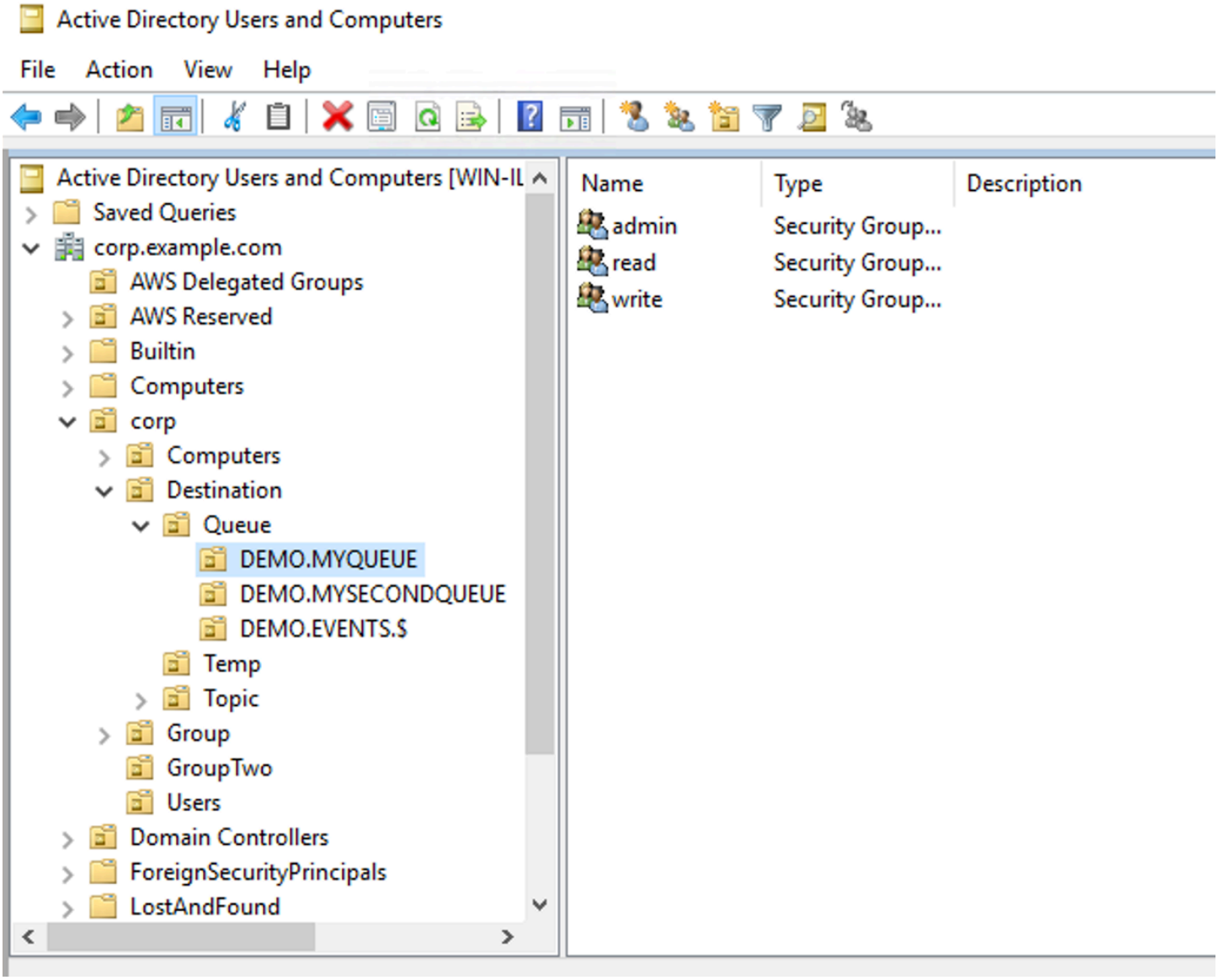
```
OU=DEMO.MYQUEUE,OU=Queue,OU=Destination,OU=corp,DC=corp,DC=example,DC=com
```



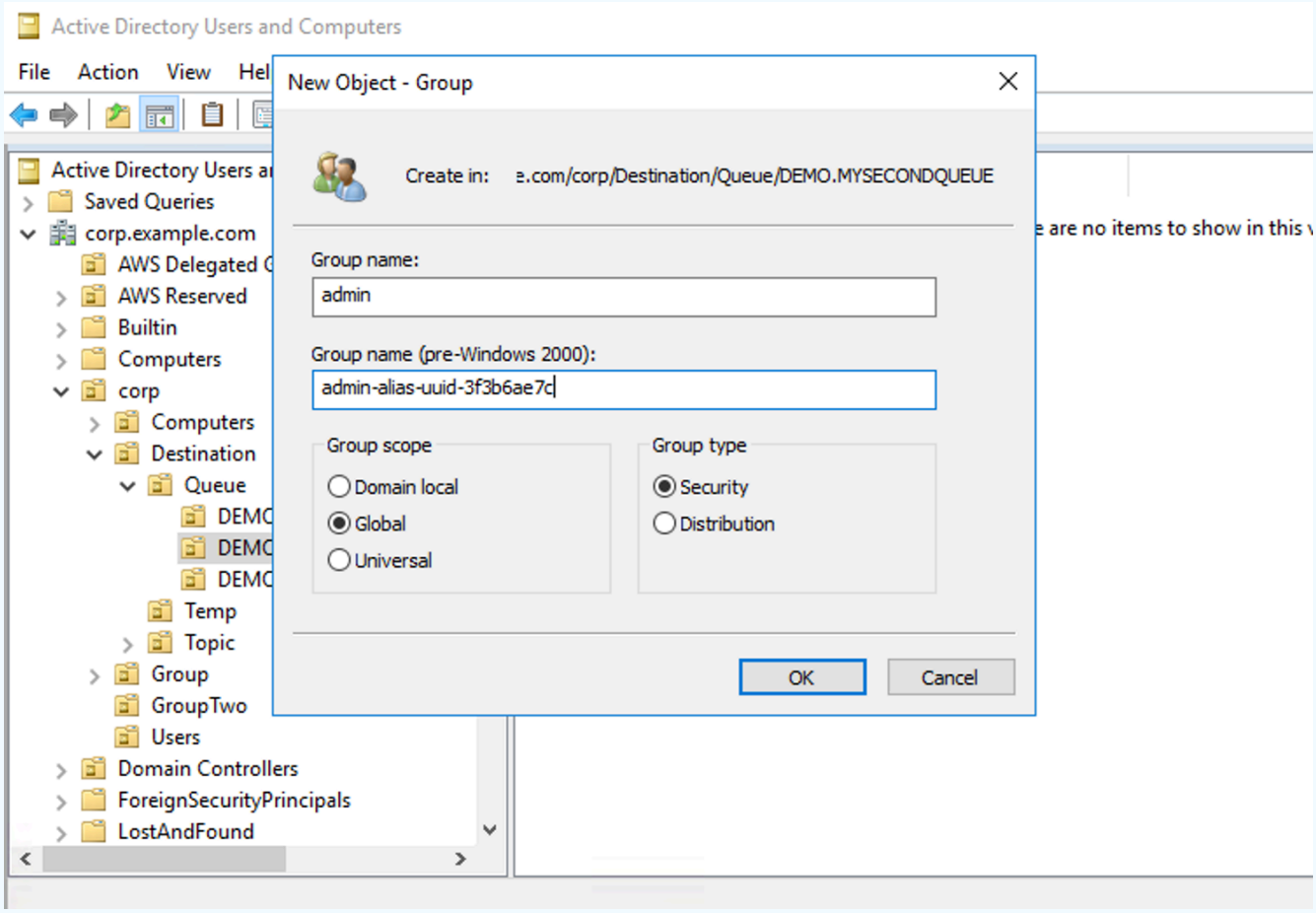
セキュリティグループ

送信先またはワイルドカードを表す各 OU 内には、3 つのセキュリティグループを作成する必要があります。ActiveMQ のすべての許可と同様に、これらは読み取り/書き込み/管理者許可です。これらの許可のそれぞれがユーザーに許可する操作の詳細については、ActiveMQ ドキュメントの「Security」を参照してください。

これらのセキュリティグループには、`read`、`write`、および `admin` という名前を付ける必要があります。これらの各セキュリティグループ内でユーザーまたはグループを追加することができ、そうすることで、そのユーザーとグループが関連付けられたアクションを実行する許可を得ます。これらのセキュリティグループは、各ワイルドカード送信先セット、または個々の送信先に必要になります。



> **Note**
>
> 管理グループを作成すると、グループ名で競合が発生します。この競合は、Windows 2000 より前のレガシールールが、グループによる同一名の共有を、グループが DIT 内の別の場所にある場合でも許可しないために発生します。[Windows 2000 より前] テキストボックス内の値はセットアップに影響しませんが、グローバルに一意である必要があります。この競合を回避するには、各 admin グループに uuid サフィックスを追加できます。



特定の送信先の admin セキュリティグループにユーザーを追加すると、ユーザーがそのトピックの作成および削除を実行できるようになります。ユーザーを read セキュリティグループに追加すると、送信先からの読み取りが可能になり、write グループに追加すると、送信先への書き込みが可能になります。

セキュリティグループ許可に個々のユーザーを追加することに加えて、グループ全体を追加することもできますが、ActiveMQ はグループの属性名をハードコードするため、activemq ソースコードにあ

るように、追加するグループにオブジェクトクラス `groupOfNames` があることを確実にする必要があります。

これを行うには、ユーザーの `uid` と同じプロセスに従ってください。「Configuring ID mappings in Active Directory Users and Computers for Windows Server 2016 (and subsequent) versions」を参照してください。

ステップ 3: (オプション) AWS Lambda 関数に接続する

AWS Lambda は Amazon MQ ブローカーに接続して、Amazon MQ ブローカーからのメッセージを消費できます。ブローカーを Lambda に接続するときは、キューからメッセージを読み取り、関数 synchronously を呼び出すイベントソースマッピングを作成します。作成するイベントソースマッピングは、ブローカーからメッセージをバッチで読み取り、それらを JSON オブジェクト形式の Lambda ペイロードに変換します。

ブローカーを Lambda 関数に接続する

1. Lambda 関数 execution role に以下の IAM ロール許可を追加します。

   - mq:DescribeBroker
   - ec2:CreateNetworkInterface
   - ec2:DeleteNetworkInterface
   - ec2:DescribeNetworkInterfaces
   - ec2:DescribeSecurityGroups
   - ec2:DescribeSubnets
   - ec2:DescribeVpcs
   - logs:CreateLogGroup
   - logs:CreateLogStream
   - logs:PutLogEvents
   - secretsmanager:GetSecretValue

   > **Note**
   >
   > 必要な IAM 許可がない場合、関数は Amazon MQ リソースからレコードを正常に読み取ることができません。

2. (オプション) パブリックアクセシビリティがないブローカーを作成した場合は、次のいずれかを実行して、Lambda のブローカーへの接続を許可する必要があります。

   - パブリックサブネットごとに 1 つの NAT ゲートウェイを設定します。詳細については、AWS Lambda デベロッパーガイドの「VPC に接続した関数のインターネットアクセスとサービスアクセス」を参照してください。
   - VPC エンドポイントを使用して、Amazon Virtual Private Cloud (Amazon VPC) と Lambda 間の接続を作成します。Amazon VPC は、 AWS Security Token Service （ AWS STS) および Secrets Manager エンドポイントにも接続する必要があります。詳細については、AWS Lambda デベロッパーガイドの「Lambda のインターフェイス VPC エンドポイントの設定」を参照してください。

3. AWS Management Consoleを使用して、Lambda 関数のイベントソースとしてブローカーを設定します。`create-event-source-mapping` AWS Command Line Interface コマンドを使用することもできます。

4. ブローカーから取り込まれたメッセージを処理するための Lambda 関数のコードをいくつか記述します。イベントソースマッピングによって取得される Lambda ペイロードは、ブローカーのエンジンタイプに依存します。以下は、Amazon MQ for ActiveMQ キューの Lambda ペイロードの例です。

> **Note**
>
> この例では、`testQueue` がキューの名前です。

```
{
  "eventSource": "aws:amq",
  "eventSourceArn": "arn:aws:mq:us-
west-2:112556298976:broker:test:b-9bcfa592-423a-4942-879d-eb284b418fc8",
  "messages": {
    [
      {
        "messageID": "ID:b-9bcfa592-423a-4942-879d-eb284b418fc8-1.mq.us-
west-2.amazonaws.com-37557-1234520418293-4:1:1:1:1",
        "messageType": "jms/text-message",
        "data": "QUJDOkFBQUE=",
        "connectionId": "myJMSCoID",
        "redelivered": false,
        "destination": {
          "physicalname": "testQueue"
```

```
        },
        "timestamp": 1598827811958,
        "brokerInTime": 1598827811958,
        "brokerOutTime": 1598827811959
      },
      {
        "messageID": "ID:b-9bcfa592-423a-4942-879d-eb284b418fc8-1.mq.us-
west-2.amazonaws.com-37557-1234520418293-4:1:1:1:1",
        "messageType":"jms/bytes-message",
        "data": "3DTOOW7crj51prgVLQaGQ82S48k=",
        "connectionId": "myJMSCoID1",
        "persistent": false,
        "destination": {
          "physicalname": "testQueue"
        },
        "timestamp": 1598827811958,
        "brokerInTime": 1598827811958,
        "brokerOutTime": 1598827811959
      }
    ]
  }
}
```

Amazon MQ の Lambda への接続、Amazon MQ イベントソースに対して Lambda がサポートするオプション、およびイベントソースマッピングエラーの詳細については、AWS Lambda デベロッパーガイドの「Amazon MQ で Lambda を使用する」を参照してください。

ActiveMQ ブローカーユーザーの作成

ActiveMQ ユーザーとは、ActiveMQ ブローカーのキューとトピックにアクセスできる人物またはアプリケーションです。ユーザーは、特定の許可を持つように設定できます。例えば、一部のユーザーに ActiveMQ ウェブコンソールへのアクセスを許可することができます。

グループはセマンティックラベルです。グループをユーザーに割り当てて、グループが特定のキューとトピックに対する送信、受信、管理を行うための許可を設定できます。

> **Note**
>
> グループをユーザーと個別に設定することはできません。グループラベルは、グループに少なくとも 1 人のユーザーを追加するときに作成され、そこからすべてのユーザーを削除するとグループも削除されます。

> **Note**
>
> Amazon MQ 上の ActiveMQ の `activemq-webconsole` グループには、すべてのキューとトピックに対する管理者アクセス許可があります。このグループのすべてのユーザーは管理者アクセス権を持ちます。

以下の例では、 AWS Management Consoleを使用して Amazon MQ ブローカーユーザーを作成、編集、および削除する方法を説明します。

新しい ActiveMQ ブローカーユーザーの作成

1. Amazon MQ コンソールにサインインします。
2. ブローカーリストからブローカーの名前 (MyBroker など) を選択して、[View details] (詳細を表示) をクリックします。

   [***MyBroker***] ページの [Users] (ユーザー) セクションに、このブローカーのすべてのユーザーがリストされます。

	Username	Console access	Groups	Pending modifications
○	paolo.santos	No	Devs	
○	jane.doe	Yes	Admins	

3. [ユーザーの作成] を選択します。
4. [ユーザーの作成] ダイアログボックスに、[ユーザー名] と [パスワード] を入力します。
5. (省略可能) ユーザーが属するグループの名前をコンマで区切って入力します (例: `Devs,Admins`)。
6. (省略可能) ユーザーが ActiveMQ ウェブコンソールにアクセスできるようにするには、[ActiveMQ ウェブコンソール] を選択します。

7. [Create user] (ユーザーの作成) をクリックします。

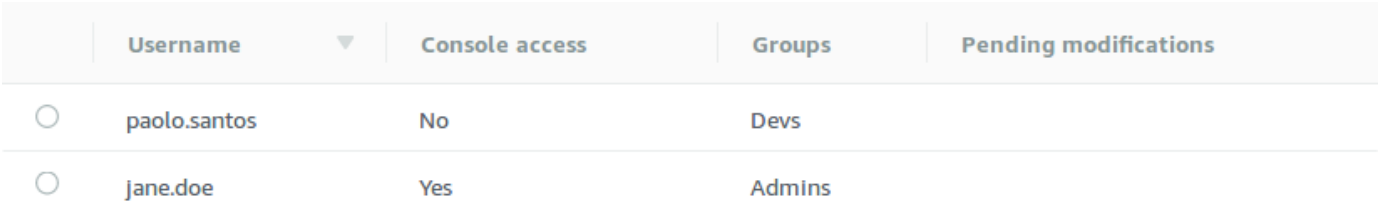
> **Important**
>
> ユーザーを変更しても、その変更はユーザーに直ちに適用されません。変更を適用するには、次のメンテナンスウィンドウまで待機するか、ブローカーを再起動する必要があります。

ActiveMQ ブローカーユーザーの編集

既存のユーザーを使用するには、以下の操作を行います。

1. Amazon MQ コンソールにサインインします。
2. ブローカーリストからブローカーの名前 (MyBroker など) を選択して、[View details] (詳細を表示) をクリックします。

   [***MyBroker***] ページの [Users] (ユーザー) セクションに、このブローカーのすべてのユーザーがリストされます。



3. サインイン認証情報を指定し、[編集] を選択します。

   [ユーザーの編集] ダイアログボックスが表示されます。
4. (省略可能) 新しい [パスワード] を入力します。
5. (省略可能) ユーザーが属するグループの名前をコンマで区切って追加または削除します (例: `Managers, Admins`)。
6. (省略可能) ユーザーが ActiveMQ ウェブコンソールにアクセスできるようにするには、[ActiveMQ ウェブコンソール] を選択します。
7. ユーザーに対する変更を保存するには、[完了] を選択します。

> **Important**
>
> ユーザーを変更しても、その変更はユーザーに直ちに適用されません。変更を適用するには、次のメンテナンスウィンドウまで待機するか、ブローカーを再起動する必要があります。

ActiveMQ ブローカーユーザーの削除

不要になったユーザーは削除できます。

1. Amazon MQ コンソールにサインインします。
2. ブローカーリストからブローカーの名前 (MyBroker など) を選択して、[View details] (詳細を表示) をクリックします。

   [***MyBroker***] ページの [Users] (ユーザー) セクションに、このブローカーのすべてのユーザーがリストされます。

	Username	Console access	Groups	Pending modifications
○	paolo.santos	No	Devs	
○	jane.doe	Yes	Admins	

3. サインイン認証情報 (***MyUser*** など) を指定し、[削除] を選択します。
4. ユーザーの削除を確認するには、[Delete ***MyUser***? (MyUser を削除しますか?)] ダイアログボックスで、[削除] を選択します。

> **Important**
>
> ユーザーを変更しても、その変更はユーザーに直ちに適用されません。変更を適用するには、次のメンテナンスウィンドウまで待機するか、ブローカーを再起動する必要があります。

ActiveMQ での Java Message Service (JMS) の使用の実用例

以下の例で、プログラムで ActiveMQ を操作する方法を示します。

- OpenWire の Java コードの例は、ブローカーに接続し、キューを作成して、メッセージを送受信します。詳細および説明については、「Connecting a Java application to your broker」を参照してください。
- MQTT のサンプル Java コードは、ブローカーへの接続、トピックの作成、およびメッセージの発行と受信を行います。
- STOMP+WSS のサンプル Java コードは、ブローカーへの接続、キューの作成、およびメッセージの発行と受信を行います。

前提条件

VPC 属性 を有効にする

VPC 内でブローカーにアクセスできることを確実にするには、`enableDnsHostnames` および `enableDnsSupport` VPC 属性を有効にする必要があります。詳細については、Amazon VPC ユーザーガイドの「VPC の DNS サポート」を参照してください。

インバウンド接続を有効にする

Amazon MQ をプログラムで操作するには、インバウンド接続を使用する必要があります。

1. Amazon MQ コンソールにサインインします。
2. ブローカーのリストからブローカーの名前 (MyBroker など) を選択します。
3. [***MyBroker***] ページの [Connections] (接続) セクションで、ブローカーのウェブコンソール URL とワイヤレベルプロトコルのアドレスとポートをメモします。
4. [Details] (詳細) セクションの [Security and network] (セキュリティとネットワーク) で、セキュリティグループの名前または
   
   をクリックします。

   EC2 ダッシュボードの [セキュリティグループ] ページが表示されます。
5. セキュリティグループのリストから、セキュリティグループを選択します。
6. ページ下部で、[インバウンド] を選択し、次に [編集] を選択します。
7. [Edit inbound rules] (インバウンドルールの編集) ダイアログボックスで、パブリックアクセスを許可する URL またはエンドポイントごとにルールを追加します (以下の例は、これをブローカーのウェブコンソールに対して行う方法を説明しています)。

   a. [ルールの追加] を選択します。

b. [タイプ] で、[カスタム TCP] を選択します。

c. [Port Range] (ポート範囲) にはウェブコンソールポート (8162) を入力します。

d. [Source] (ソース) では、[Custom] (カスタム) が選択された状態のままにしておき、ウェブコンソールにアクセスできるようにするシステムの IP アドレスを入力します (`192.0.2.1` など)。

e. [Save] (保存) をクリックします。

これで、ブローカーはインバウンド接続を受け入れることができます。

Java の依存関係を追加する

OpenWire

`activemq-client.jar` パッケージと `activemq-pool.jar` パッケージを Java クラスパスに追加します。以下の例は、Maven プロジェクトの `pom.xml` ファイルにあるこれらの依存関係を示しています。

```
<dependencies>
    <dependency>
        <groupId>org.apache.activemq</groupId>
        <artifactId>activemq-client</artifactId>
        <version>5.15.16</version>
    </dependency>
    <dependency>
        <groupId>org.apache.activemq</groupId>
        <artifactId>activemq-pool</artifactId>
        <version>5.15.16</version>
    </dependency>
</dependencies>
```

`activemq-client.jar` の詳細については、Apache ActiveMQ ドキュメントの「Initial Configuration」を参照してください。

MQTT

`org.eclipse.paho.client.mqttv3.jar` パッケージを Java クラスパスに追加します。次の例では、この依存関係を Maven プロジェクトの `pom.xml` ファイルで示しています。

```
<dependencies>
                    <dependency>
```

```
                        <groupId>org.eclipse.paho</groupId>
                        <artifactId>org.eclipse.paho.client.mqttv3</artifactId>
                        <version>1.2.0</version>
                    </dependency>
                    </dependencies>
```

org.eclipse.paho.client.mqttv3.jar の詳細については、Eclipse Paho Java Client を参照してください。

STOMP+WSS

次のパッケージを Java クラスパスに追加しました。

- spring-messaging.jar
- spring-websocket.jar
- javax.websocket-api.jar
- jetty-all.jar
- slf4j-simple.jar
- jackson-databind.jar

以下の例は、Maven プロジェクトの pom.xml ファイルにあるこれらの依存関係を示しています。

```
<dependencies>
                    <dependency>
                        <groupId>org.springframework</groupId>
                        <artifactId>spring-messaging</artifactId>
                        <version>5.0.5.RELEASE</version>
                    </dependency>
                    <dependency>
                        <groupId>org.springframework</groupId>
                        <artifactId>spring-websocket</artifactId>
                        <version>5.0.5.RELEASE</version>
                    </dependency>
                    <dependency>
                        <groupId>javax.websocket</groupId>
                        <artifactId>javax.websocket-api</artifactId>
                        <version>1.1</version>
                    </dependency>
                    <dependency>
                        <groupId>org.eclipse.jetty.aggregate</groupId>
```

```
                    <artifactId>jetty-all</artifactId>
                    <type>pom</type>
                    <version>9.3.3.v20150827</version>
                </dependency>
                <dependency>
                    <groupId>org.slf4j</groupId>
                    <artifactId>slf4j-simple</artifactId>
                    <version>1.6.6</version>
                </dependency>
                <dependency>
                    <groupId>com.fasterxml.jackson.core</groupId>
                    <artifactId>jackson-databind</artifactId>
                    <version>2.5.0</version>
                </dependency>
                </dependencies>
```

詳細については、Spring Framework ドキュメントの「STOMP Support」を参照してください。

AmazonMQExample.java

> **Important**
>
> 以下のコード例では、プロデューサーとコンシューマーが単一のスレッド内で実行されます。実稼働システム (またはブローカーインスタンスのフェイルオーバーをテストする) には、プロデューサーとコンシューマーが個別のホストまたはスレッドで実行されるようにしてください。

OpenWire

```
/*
 * Copyright 2010-2019 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 *  https://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
```

```
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 *
 */

                    import org.apache.activemq.ActiveMQConnectionFactory;
                    import org.apache.activemq.jms.pool.PooledConnectionFactory;

                    import javax.jms.*;

                    public class AmazonMQExample {

                    // Specify the connection parameters.
                    private final static String WIRE_LEVEL_ENDPOINT
                            = "ssl://b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-
east-2.amazonaws.com:61617";
                    private final static String ACTIVE_MQ_USERNAME =
 "MyUsername123";
                    private final static String ACTIVE_MQ_PASSWORD =
 "MyPassword456";

                    public static void main(String[] args) throws JMSException {
                        final ActiveMQConnectionFactory connectionFactory =
                                createActiveMQConnectionFactory();
                        final PooledConnectionFactory pooledConnectionFactory =
                                createPooledConnectionFactory(connectionFactory);

                        sendMessage(pooledConnectionFactory);
                        receiveMessage(connectionFactory);

                        pooledConnectionFactory.stop();
                    }

                    private static void
                    sendMessage(PooledConnectionFactory pooledConnectionFactory)
 throws JMSException {
                        // Establish a connection for the producer.
                        final Connection producerConnection =
 pooledConnectionFactory
                                .createConnection();
                        producerConnection.start();

                        // Create a session.
                        final Session producerSession = producerConnection
```

```
                                .createSession(false, Session.AUTO_ACKNOWLEDGE);

                        // Create a queue named "MyQueue".
                        final Destination producerDestination = producerSession
                                .createQueue("MyQueue");

                        // Create a producer from the session to the queue.
                        final MessageProducer producer = producerSession
                                .createProducer(producerDestination);
                        producer.setDeliveryMode(DeliveryMode.NON_PERSISTENT);

                        // Create a message.
                        final String text = "Hello from Amazon MQ!";
                        final TextMessage producerMessage = producerSession
                                .createTextMessage(text);

                        // Send the message.
                        producer.send(producerMessage);
                        System.out.println("Message sent.");

                        // Clean up the producer.
                        producer.close();
                        producerSession.close();
                        producerConnection.close();
                    }

                    private static void
                    receiveMessage(ActiveMQConnectionFactory connectionFactory)
throws JMSException {
                        // Establish a connection for the consumer.
                        // Note: Consumers should not use PooledConnectionFactory.
                        final Connection consumerConnection =
connectionFactory.createConnection();
                        consumerConnection.start();

                        // Create a session.
                        final Session consumerSession = consumerConnection
                                .createSession(false, Session.AUTO_ACKNOWLEDGE);

                        // Create a queue named "MyQueue".
                        final Destination consumerDestination = consumerSession
                                .createQueue("MyQueue");

                        // Create a message consumer from the session to the queue.
```

```
                        final MessageConsumer consumer = consumerSession
                                .createConsumer(consumerDestination);

                        // Begin to wait for messages.
                        final Message consumerMessage = consumer.receive(1000);

                        // Receive the message when it arrives.
                        final TextMessage consumerTextMessage = (TextMessage)
consumerMessage;
                        System.out.println("Message received: " +
consumerTextMessage.getText());

                        // Clean up the consumer.
                        consumer.close();
                        consumerSession.close();
                        consumerConnection.close();
                    }

                    private static PooledConnectionFactory
                    createPooledConnectionFactory(ActiveMQConnectionFactory
connectionFactory) {
                        // Create a pooled connection factory.
                        final PooledConnectionFactory pooledConnectionFactory =
                                new PooledConnectionFactory();

pooledConnectionFactory.setConnectionFactory(connectionFactory);
                        pooledConnectionFactory.setMaxConnections(10);
                        return pooledConnectionFactory;
                    }

                    private static ActiveMQConnectionFactory
createActiveMQConnectionFactory() {
                        // Create a connection factory.
                        final ActiveMQConnectionFactory connectionFactory =
                                new ActiveMQConnectionFactory(WIRE_LEVEL_ENDPOINT);

                        // Pass the sign-in credentials.
                        connectionFactory.setUserName(ACTIVE_MQ_USERNAME);
                        connectionFactory.setPassword(ACTIVE_MQ_PASSWORD);
                        return connectionFactory;
                    }
                    }
```

MQTT

```
/*
 * Copyright 2010-2019 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 *  https://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 *
 */

                    import org.eclipse.paho.client.mqttv3.*;

                    public class AmazonMQExampleMqtt implements MqttCallback {

                    // Specify the connection parameters.
                    private final static String WIRE_LEVEL_ENDPOINT =
                            "ssl://b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-
east-2.amazonaws.com:8883";
                    private final static String ACTIVE_MQ_USERNAME =
 "MyUsername123";
                    private final static String ACTIVE_MQ_PASSWORD =
 "MyPassword456";

                    public static void main(String[] args) throws Exception {
                        new AmazonMQExampleMqtt().run();
                    }

                    private void run() throws MqttException, InterruptedException {

                        // Specify the topic name and the message text.
                        final String topic = "myTopic";
                        final String text = "Hello from Amazon MQ!";

                        // Create the MQTT client and specify the connection
 options.
                        final String clientId = "abc123";
```

```
                        final MqttClient client = new
MqttClient(WIRE_LEVEL_ENDPOINT, clientId);
                        final MqttConnectOptions connOpts = new
MqttConnectOptions();

                        // Pass the sign-in credentials.
                        connOpts.setUserName(ACTIVE_MQ_USERNAME);
                        connOpts.setPassword(ACTIVE_MQ_PASSWORD.toCharArray());

                        // Create a session and subscribe to a topic filter.
                        client.connect(connOpts);
                        client.setCallback(this);
                        client.subscribe("+");

                        // Create a message.
                        final MqttMessage message = new
MqttMessage(text.getBytes());

                        // Publish the message to a topic.
                        client.publish(topic, message);
                        System.out.println("Published message.");

                        // Wait for the message to be received.
                        Thread.sleep(3000L);

                        // Clean up the connection.
                        client.disconnect();
                    }

                    @Override
                    public void connectionLost(Throwable cause) {
                        System.out.println("Lost connection.");
                    }

                    @Override
                    public void messageArrived(String topic, MqttMessage message)
throws MqttException {
                        System.out.println("Received message from topic " + topic +
": " + message);
                    }

                    @Override
                    public void deliveryComplete(IMqttDeliveryToken token) {
                        System.out.println("Delivered message.");
```

```
                    }
                    }
```

STOMP+WSS

```
/*
 * Copyright 2010-2019 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 *  https://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 *
 */

                    import
 org.springframework.messaging.converter.StringMessageConverter;
                    import org.springframework.messaging.simp.stomp.*;
                    import org.springframework.web.socket.WebSocketHttpHeaders;
                    import org.springframework.web.socket.client.WebSocketClient;
                    import
 org.springframework.web.socket.client.standard.StandardWebSocketClient;
                    import
 org.springframework.web.socket.messaging.WebSocketStompClient;

                    import java.lang.reflect.Type;

                    public class AmazonMQExampleStompWss {

                    // Specify the connection parameters.
                    private final static String DESTINATION = "/queue";
                    private final static String WIRE_LEVEL_ENDPOINT =
                            "wss://b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-
east-2.amazonaws.com:61619";
                    private final static String ACTIVE_MQ_USERNAME =
 "MyUsername123";
```

```
                    private final static String ACTIVE_MQ_PASSWORD =
"MyPassword456";

                    public static void main(String[] args) throws Exception {
                        final AmazonMQExampleStompWss example = new
AmazonMQExampleStompWss();

                        final StompSession stompSession = example.connect();
                        System.out.println("Subscribed to a destination using
session.");
                        example.subscribeToDestination(stompSession);

                        System.out.println("Sent message to session.");
                        example.sendMessage(stompSession);
                        Thread.sleep(60000);
                    }

                    private StompSession connect() throws Exception {
                        // Create a client.
                        final WebSocketClient client = new
StandardWebSocketClient();
                        final WebSocketStompClient stompClient = new
WebSocketStompClient(client);
                        stompClient.setMessageConverter(new
StringMessageConverter());

                        final WebSocketHttpHeaders headers = new
WebSocketHttpHeaders();

                        // Create headers with authentication parameters.
                        final StompHeaders head = new StompHeaders();
                        head.add(StompHeaders.LOGIN, ACTIVE_MQ_USERNAME);
                        head.add(StompHeaders.PASSCODE, ACTIVE_MQ_PASSWORD);

                        final StompSessionHandler sessionHandler = new
MySessionHandler();

                        // Create a connection.
                        return stompClient.connect(WIRE_LEVEL_ENDPOINT, headers,
head,
                                sessionHandler).get();
                    }
```

```
                    private void subscribeToDestination(final StompSession
stompSession) {
                        stompSession.subscribe(DESTINATION, new MyFrameHandler());
                    }

                    private void sendMessage(final StompSession stompSession) {
                        stompSession.send(DESTINATION, "Hello from Amazon
MQ!".getBytes());
                    }

                    private static class MySessionHandler extends
StompSessionHandlerAdapter {
                        public void afterConnected(final StompSession stompSession,
                                                   final StompHeaders stompHeaders) {
                            System.out.println("Connected to broker.");
                        }
                    }

                    private static class MyFrameHandler implements StompFrameHandler
{
                        public Type getPayloadType(final StompHeaders headers) {
                            return String.class;
                        }

                        public void handleFrame(final StompHeaders stompHeaders,
                                                final Object message) {
                            System.out.print("Received message from topic: " +
message);
                        }
                    }
                    }
```

Amazon MQ for ActiveMQ エンジンバージョンの管理

Apache ActiveMQ は、X.Y.Z 形式のセマンティックバージョニングに従ってバージョン番号を分類します。Amazon MQ for ActiveMQ の実装では、X はメジャーバージョンを示し、Y はマイナーバージョンを表し、Z はパッチバージョン番号を示します。Amazon MQ は、メジャーバージョン番号が変更される場合に、バージョン変更がメジャーであると見なします。例えば、バージョン 5.17 から 6.0 へのアップグレードは、メジャーバージョンアップグレードと見なされます。マイナーバージョン番号またはパッチバージョン番号のみが変わる場合、バージョン変更はマイナーと見なされます。例えば、バージョン 5.17 から 5.18 へのアップグレードは、マイナーバージョンアップグレードと見

なされます。をオンにすると、Amazon MQ `autoMinorVersionUpgrade`はブローカーを利用可能な最新のパッチバージョンにアップグレードします。

Amazon MQ for ActiveMQ では、すべてのブローカーについて、サポートされている最新のマイナーバージョンを使用することをお勧めします。ブローカーエンジンバージョンをアップグレードする手順については、「Amazon MQ ブローカーエンジンバージョンのアップグレード」を参照してください。

Amazon MQ for ActiveMQ でサポートされるエンジンバージョン

Amazon MQ バージョンサポートカレンダーは、ブローカーエンジンバージョンがサポート終了に達するタイミングを示します。あるバージョンがサポート終了に達すると、Amazon MQ は、そのバージョンのすべてのブローカーを、サポートされている次のバージョンに自動的にアップグレードします。このアップグレードは、ブローカーのスケジュールされたメンテナンスウィンドウ内で、サポート終了日から 45 日以内に行われます。

Amazon MQ は、バージョンがサポート終了に達する少なくとも 90 日前に通知を送信します。中断を防ぐために、サポート終了日より前にブローカーをアップグレードすることをお勧めします。また、サポート終了が 30 日以内に予定されているバージョンで新しいブローカーを作成することはできません。

Apache ActiveMQ のバージョン	Amazon MQ でのサポート終了
ActiveMQ 5.18 (推奨)	
ActiveMQ 5.17	2025 年 6 月 16 日
ActiveMQ 5.16	2024 年 11 月 15 日
ActiveMQ 5.15	2024 年 9 月 16 日

新しい Amazon MQ for ActiveMQ ブローカーを作成するときは、サポートされている任意の ActiveMQ エンジンバージョンを指定できます。ブローカーの作成時にエンジンバージョン番号を指定しない場合は、Amazon MQ により、デフォルトで自動的に最新のエンジンバージョン番号が選択されます。

エンジンバージョンのアップグレード

ブローカーはいつでも、サポートされている次のメジャーバージョンまたはマイナーバージョンに手動でアップグレードできます。自動マイナーバージョンアップグレードを有効にすると、Amazon MQ はメンテナンスウィンドウ内で、サポートされている最新のパッチバージョンにブローカーをアップグレードします。

ブローカーの手動アップグレードの詳細については、「the section called “エンジンバージョンのアップグレード”」を参照してください。

サポートされているエンジンバージョンのリスト化

describe-broker-instance-options AWS CLI コマンドを使用して、サポートされているすべてのマイナーエンジンバージョンとメジャーエンジンバージョンを一覧表示できます。

```
aws mq describe-broker-instance-options
```

エンジンおよびインスタンスタイプで結果をフィルタリングするには、以下にあるように、--engine-type および --host-instance-type オプションを使用します。

```
aws mq describe-broker-instance-options --engine-type engine-type --host-instance-type instance-type
```

例えば、ActiveMQ と mq.m5.large インスタンスタイプで結果をフィルタリングするには、*engine-type* を ACTIVEMQ、*instance-type* を mq.m5.large に置き換えます。

Amazon MQ for ActiveMQ のベストプラクティス

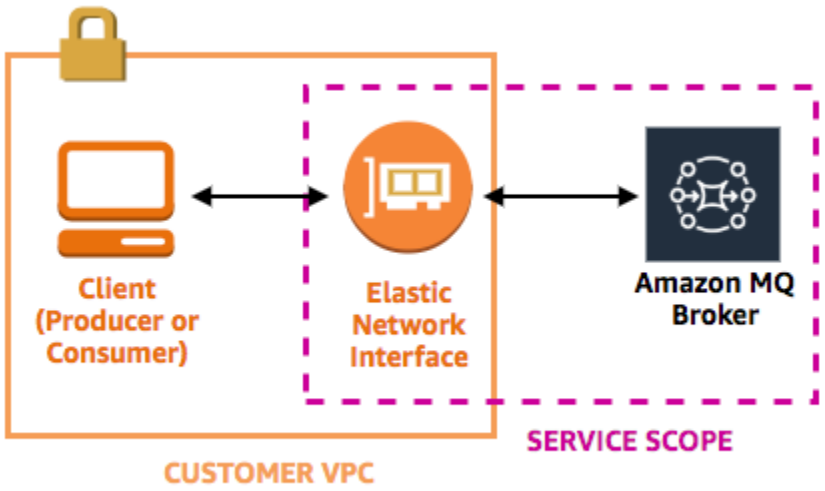
このセクションは、Amazon MQ での ActiveMQ ブローカーの使用時にパフォーマンスを最大限に引き出し、スループットコストを最小限に抑えるための推奨事項をすばやく見つけるために使用してください。

Amazon MQ Elastic Network Interface を変更または削除しない

初めて Amazon MQ ブローカーを作成するときは、Amazon MQ がアカウントの Virtual Private Cloud (VPC) 内に Elastic Network Interface をプロビジョンするため、多数の EC2 許可が必要になります。このネットワークインターフェイスは、クライアント (プロデューサーまたはコンシュー

マー) が Amazon MQ ブローカーと通信することを可能にします。このネットワークインターフェイスは、アカウントの VPC の一部であるにもかかわらず、Amazon MQ のサービス範囲内であると見なされます。

> **Warning**
>
> このネットワークインターフェイスを変更または削除しないでください。このネットワークインターフェイスを変更または削除すると、VPC とブローカーとの間の接続が完全に失われる可能性があります。



常に接続プールを使用する

単一のプロデューサーと単一のコンシューマーを使用するシナリオ (開始方法: ActiveMQ ブローカーの作成と接続チュートリアルなど) では、各プロデューサーおよびコンシューマーに単一の `ActiveMQConnectionFactory` クラスを使用できます。以下はその例です。

```
// Create a connection factory.
final ActiveMQConnectionFactory connectionFactory = new
 ActiveMQConnectionFactory(wireLevelEndpoint);

// Pass the sign-in credentials.
connectionFactory.setUserName(activeMqUsername);
connectionFactory.setPassword(activeMqPassword);
```

```
// Establish a connection for the consumer.
final Connection consumerConnection = connectionFactory.createConnection();
consumerConnection.start();
```

ただし、複数のプロデューサーやコンシューマーが関与するより現実的なシナリオでは、複数のプロデューサーのために多数の接続を作成することはコスト高および非効率的になる場合があります。このようなシナリオでは、PooledConnectionFactory クラスを使用して複数のプロデューサーリクエストをグループ化する必要があります。以下はその例です。

> **Note**
>
> メッセージコンシューマーには、PooledConnectionFactory クラスを一切使用しないでください。

```
// Create a connection factory.
final ActiveMQConnectionFactory connectionFactory = new
 ActiveMQConnectionFactory(wireLevelEndpoint);

// Pass the sign-in credentials.
connectionFactory.setUserName(activeMqUsername);
connectionFactory.setPassword(activeMqPassword);

// Create a pooled connection factory.
final PooledConnectionFactory pooledConnectionFactory = new PooledConnectionFactory();
pooledConnectionFactory.setConnectionFactory(connectionFactory);
pooledConnectionFactory.setMaxConnections(10);

// Establish a connection for the producer.
final Connection producerConnection = pooledConnectionFactory.createConnection();
producerConnection.start();
```

常にフェイルオーバートランスポートを使用して複数のブローカーエンドポイントに接続する

アクティブ/スタンバイデプロイモードを使用するとき、またはオンプレミスメッセージブローカーから Amazon MQ に移行するときなど、アプリケーションを複数のブローカーエンドポイントに接続する必要がある場合は、フェイルオーバートランスポートを使用して、コンシューマーがそれらのいずれかにランダムに接続できるようにします。例:

```
failover:(ssl://b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-
east-2.amazonaws.com:61617,ssl://b-9876l5k4-32ji-109h-8gfe-7d65c4b132a1-2.mq.us-
west-2.amazonaws.com:61617)?randomize=true
```

Important

マルチアベイラビリティーゾーンブローカーでは、メンテナンスウィンドウやブローカーの再起動中にフェイルオーバーが発生する可能性があります。フェイルオーバートランスポートを使用して、ブローカーの可用性を確保します。

メッセージセレクタを使用しない

JMS セレクタを使用して、トピックのサブスクリプションにフィルターをアタッチする (コンテンツに基づいてコンシューマーにメッセージを送信するため) ことは可能ですが、JMS セレクタを使用すると Amazon MQ ブローカーのフィルターバッファが満杯になり、メッセージをフィルタリングできなくなります。

一般的に、コンシューマーによるメッセージのルーティングは避けます。コンシューマーとプロデューサーが適切に非干渉化されるために、コンシューマーとプロデューサーはどちらもエフェメラルである必要があるためです。

永続サブスクリプションよりも仮想送信先を優先する

たとえば接続が失われて復元された後などに、トピックに発行されたすべてのメッセージをコンシューマーが受信するには、永続サブスクリプションが役立ちます。ただし、永続サブスクリプションを使用する場合、競合するコンシューマーの使用は不可能であり、パフォーマンスの大規模な問題が発生する可能性があります。代わりに、仮想送信先を使用することを検討してください。

Amazon VPC ピアリングを使用する場合は、CIDR 範囲 **10.0.0.0/16** 内のクライアント IP を避けてください。

オンプレミスインフラストラクチャと Amazon MQ ブローカーの間に Amazon VPC ピアリングをセットアップしている場合は、CIDR 範囲 `10.0.0.0/16` 内の IP でクライアント接続を設定しない必要があります。

低速コンシューマーのキューに対して同時保存とディスパッチを無効にする

デフォルトで、Amazon MQ は高速コンシューマーのキューに対して最適化を行います。

- コンシューマーは、プロデューサーによって生成されるメッセージの速度に対応できる場合、高速とみなされます。
- キューによって未確認メッセージのバックログが生成され、プロデューサーのスループットが低下する可能性がある場合、コンシューマーは低速とみなされます。

低速コンシューマーのキューに対して最適化を行うよう Amazon MQ に指示するには、`concurrentStoreAndDispatchQueues` 属性を `false` に設定します。設定の例については、「concurrentStoreAndDispatchQueues」を参照してください。

最良なスループットのために正しいブローカーインスタンスタイプを選択する

ブローカーインスタンスタイプのメッセージスループットは、アプリケーションのユースケースおよび以下の要因に依存します。

- ActiveMQ を永続モードで使用する
- メッセージサイズ
- プロデューサーとコンシューマーの数
- 送信先の数

メッセージサイズ、レイテンシー、およびスループット間の関係の理解

ユースケースによっては、より大きなブローカーインスタンスタイプはシステムスループットを向上させない場合があります。ActiveMQ が耐久性のあるストレージにメッセージを書き込むと、メッセージのサイズはシステムの制限要因を決定します。

- メッセージが 100 KB 未満の場合、永続的ストレージのレイテンシーが制限要因となります。
- メッセージが 100 KB 以上の場合、永続的ストレージのスループットが制限要因となります。

ActiveMQ を永続モード使用すると、ストレージへの書き込みは通常、前のコンシューマーがいくつか存在するか、あるいはコンシューマーが低速の場合に発生します。非永続的なモードでは、ブロー

カーインスタンスのヒープメモリに空き容量がない場合にも、低速のコンシューマーによるストレージへの書き込みが発生します。

アプリケーションにおける最適なブローカーインスタンスタイプを決定するには、異なるブローカーインスタンスタイプをテストすることが推奨されます。詳細については、「Broker instance types」および「Measuring the Throughput for Amazon MQ using the JMS Benchmark」を参照してください。

より大きなブローカーインスタンスタイプのユースケース

より大きなブローカーインスタンスタイプがスループットを向上させるには、3 つの一般的なユースケースがあります。

- 非永続モード – アプリケーションがブローカーインスタンスのフェイルオーバー中におけるメッセージの喪失による影響を受けにくいときは、多くの場合 ActiveMQ の非永続モードを使用できます。このモードでは、ブローカーインスタンスのヒープメモリに空き容量がない場合にのみ、ActiveMQ は永続的ストレージにメッセージを書き込みます。非永続モードを使用するシステムは、大きなブローカーインスタンスタイプで利用できるより大きなメモリ容量、高速の CPU、および高速のネットワークの利点を活用できます。
- 高速コンシューマー – アクティブなコンシューマーが利用可能で、`concurrentStoreAndDispatchQueues` フラグが有効になっていると、ActiveMQ は、永続モードになっている場合でも、ストレージにメッセージを送信することなく、プロデューサーからコンシューマーへの直接的なメッセージのフローを許可します。アプリケーションが素早くメッセージを消費できる場合 (あるいは、コンシューマーがその処理を行えるように設計できる場合)、アプリケーションはより大きなブローカーインスタンスタイプの利点を活用できます。アプリケーションがより素早くメッセージを消費できるようにするには、アプリケーションインスタンスにコンシューマースレッドを追加するか、あるいはアプリケーションインスタンスを水平あるいは垂直にスケールアップします。
- バッチトランザクション – 永続的モードを使用しており、トランザクションごとに複数のメッセージを送信するときは、より大きなブローカーインスタンスタイプを使用することによって、全体的に高いメッセージスループットを達成することができます。詳細については、ActiveMQ ドキュメントの「Should I Use Transactions?」を参照してください。

最高のスループットのために正しいブローカーストレージタイプを選択する

複数のアベイラビリティーゾーン全体で優れた耐障害性とレプリケーションを活用するには、Amazon EFS を使用します。低レイテンシーと高スループットを活用するには、Amazon EBS を使用します。詳細については、「Storage」を参照してください。

ブローカーのネットワークを正しく設定する

ブローカーのネットワークを作成するときは、アプリケーションに合わせて正しく設定します。

- 永続モードを有効にする – 同等のものと比べると、各ブローカーインスタンスはプロデューサーまたはコンシューマーのように動作するため、ブローカーのネットワークはメッセージの分散レプリケーションを提供しません。コンシューマーとして機能する最初のブローカーはメッセージを受信し、それをストレージに永続化します。このブローカーは確認をプロデューサーに送信し、そのメッセージを次のブローカーに転送します。2 番目のブローカーがメッセージの持続性を確認すると、最初のブローカーはそのメッセージを削除します。

  永続モードが無効になっている場合、最初のブローカーはメッセージをストレージに保持せずにプロデューサーに確認します。詳細については、Apache ActiveMQ ドキュメントの「レプリケートされたメッセージストア」および「永続的配信と非永続的配信の違い」を参照してください。
- ブローカーインスタンスのアドバイザリーメッセージを無効にしない – 詳細については、Apache ActiveMQ ドキュメントの「Advisory Message」を参照してください。
- マルチキャストブローカー検出を使用しない – Amazon MQ はマルチキャストを使用したブローカー検出をサポートしません。詳細については、Apache ActiveMQ ドキュメントの「検出、マルチキャスト、および zeroconf の違い」を参照してください。

準備された XA トランザクションを復旧することで再起動が遅くならないようにする

ActiveMQ は分散型 (XA) トランザクションをサポートしています。ActiveMQ が XA トランザクションを処理する方法を理解しておくと、Amazon MQ でのブローカーの再起動とフェイルオーバーにかかる長い復旧時間の回避に役立ちます。

未解決の準備済み XA トランザクションは、再起動のたびに再実行されます。これらのトランザクションが未解決のままである場合、その数は時間の経過とともに大きくなり、ブローカーの起動に必要な時間が大幅に長くなります。これにより、再起動とフェイルオーバー時間に影響がありま

す。commit() および rollback() を使用してこれらのトランザクションを解決し、時間の経過とともにパフォーマンスが低下しないようにする必要があります。

未解決の準備された XA トランザクションをモニタリングするには、Amazon CloudWatch Logs の JournalFilesForFastRecovery メトリクスを使用できます。この数値が増えるか、常に 1 より高い場合は、次の例のようなコードを使用して、未解決のトランザクションを復旧します。詳細については、「Amazon MQ のクォータ」を参照してください。

以下のコード例は、準備された XA トランザクションを確認し、rollback() でそれらを終了します。

```
import org.apache.activemq.ActiveMQXAConnectionFactory;

import javax.jms.XAConnection;
import javax.jms.XASession;
import javax.transaction.xa.XAResource;
import javax.transaction.xa.Xid;

public class RecoverXaTransactions {
    private static final ActiveMQXAConnectionFactory ACTIVE_MQ_CONNECTION_FACTORY;
    final static String WIRE_LEVEL_ENDPOINT =
            "tcp://localhost:61616";;
    static {
        final String activeMqUsername = "MyUsername123";
        final String activeMqPassword = "MyPassword456";
        ACTIVE_MQ_CONNECTION_FACTORY = new
 ActiveMQXAConnectionFactory(activeMqUsername, activeMqPassword, WIRE_LEVEL_ENDPOINT);
        ACTIVE_MQ_CONNECTION_FACTORY.setUserName(activeMqUsername);
        ACTIVE_MQ_CONNECTION_FACTORY.setPassword(activeMqPassword);
    }

    public static void main(String[] args) {
        try {
            final XAConnection connection =
 ACTIVE_MQ_CONNECTION_FACTORY.createXAConnection();
            XASession xaSession = connection.createXASession();
            XAResource xaRes = xaSession.getXAResource();

            for (Xid id : xaRes.recover(XAResource.TMENDRSCAN)) {
                xaRes.rollback(id);
            }
            connection.close();
```

```
        } catch (Exception e) {
        }
    }
}
```

実際のシナリオでは、XA トランザクションマネージャーに対して準備済み XA トランザクションを確認することができます。その後、rollback() または commit() を使用して準備されたトランザクションのそれぞれを処理するかどうかを決定できます。

Amazon MQ for RabbitMQ の使用

Amazon MQ は、ニーズに適したコンピューティングおよびストレージリソースを使用したメッセージブローカーの作成を容易にします。ブローカーは、、Amazon MQ REST API AWS Management Console、または を使用して作成、管理、削除できます AWS Command Line Interface。

このセクションでは、ActiveMQ エンジンタイプと RabbitMQ エンジンタイプ向けのメッセージブローカーの基本的要素を説明し、利用可能な Amazon MQ ブローカーのインスタンスタイプとステータスをリストして、ブローカーのアーキテクチャと設定オプションの概要を説明します。

Amazon MQ REST API については、Amazon MQ REST API リファレンスを参照してください。

Amazon MQ for RabbitMQ ブローカー

Amazon MQ for RabbitMQ ブローカーとは

ブローカーは、Amazon MQ で実行されるメッセージブローカー環境です。これは、Amazon MQ の基本的な構成要素です。ブローカーインスタンスクラス (`m5`、`t3`) とサイズ (`large`、`micro`) を組み合わせた説明は、ブローカーインスタンスタイプ (など) と呼ばれます`mq.m5.large`。

- 単一インスタンスブローカーは、ネットワークロードバランサー (NLB) の内側にある 1 つのアベイラビリティーゾーン内の 1 つのブローカーで構成されます。ブローカーは、アプリケーション、および Amazon EBS ストレージボリュームと通信します。
- クラスターデプロイは、ネットワークロードバランサーの内側にある 3 つの RabbitMQ ブローカーノードの論理グループで、それぞれがユーザー、キュー、および複数のアベイラビリティーゾーン (AZ) 間の分散状態を共有します。

詳細については、「Amazon MQ for RabbitMQ ブローカーのデプロイオプション」を参照してください。

マイナーバージョンの自動アップグレードを有効にして、RabbitMQ エンジンの新しいマイナーバージョンがリリースされたときに、ブローカーエンジンを新しいマイナーバージョンにアップグレードできます。自動アップグレードは、曜日、時刻 (24 時間形式)、およびタイムゾーン (デフォルトは UTC) で定義されたメンテナンスウィンドウ中に行われます。

サポートされるプロトコル

RabbitMQ ブローカーには、RabbitMQ がサポートする任意のプログラミング言語を使用し、以下のプロトコルに対して TLS を有効にすることによってアクセスできます。

- AMQP (0-9-1)

リスナーポート

Amazon MQ マネージド RabbitMQ ブローカーは、amqps 経由でのアプリケーションレベルの接続、および RabbitMQ ウェブコンソールと Management API を使用したクライアント接続に対して以下のリスナーポートをサポートします。

- リスナーポート 5671 – セキュアな AMQP URL 経由で行われる接続に使用されます。例えば、us-west-2 リージョンでデプロイされた、ブローカー ID が b-c8352341-ec91-4a78-ad9c-a43f23d325bb のブローカーの場合、ブローカーの完全な amqp URL は b-c8352341-ec91-4a78-ad9c-a43f23d325bb.mq.us-west-2.amazonaws.com:5671 になります。
- リスナーポート 443 および 15671 – RabbitMQ ウェブコンソールまたは Management API 経由でのブローカーへのアクセスには、両方のリスナーポートを区別なく使用できます。

属性

RabbitMQ ブローカーには、いくつかの属性があります。

- 名前。例えば、MyBroker。
- ID。例えば、b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9。
- Amazon リソースネーム (ARN)。例えば、arn:aws:mq:us-east-2:123456789012:broker:MyBroker:b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9。
- RabbitMQ ウェブコンソール URL。例えば、https://b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-east-2.amazonaws.com。

  詳細については、RabbitMQ ドキュメントの「RabbitMQ web console」を参照してください。
- セキュアな AMQP エンドポイント。例えば、amqps://b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-east-2.amazonaws.com。

ブローカー属性の完全なリストについては、Amazon MQ REST API リファレンスで以下を参照してください。

- REST オペレーション ID: ブローカー
- REST オペレーション ID: ブローカー
- REST オペレーション ID: ブローカーの再起動

Amazon MQ for RabbitMQ ブローカーのユーザー

すべての AMQP 0-9-1 クライアント接続には関連付けられたユーザーがあり、認証される必要があります。各クライアント接続は仮想ホスト (vhost) もターゲットにしており、ユーザーにはこのホストに対する一連の許可が必要です。ユーザーは、vhost 内のキューとエクスチェンジに対して設定、書き込み、および読み込みを行う許可を持つことができます。ユーザーの認証情報、およびターゲット vhost は、接続の確立時に指定されます。

Amazon MQ for RabbitMQ ブローカーを初めて作成する場合、Amazon MQ は、指定されたサインイン認証情報を使用して、`administrator` タグで RabbitMQ ユーザーを作成します。その後、RabbitMQ Management API、または RabbitMQ ウェブコンソールを使用してユーザーを追加および管理することができます。また、RabbitMQ ウェブコンソールまたは Management API を使用して、ユーザーの認証情報とタグを設定または変更することもできます。

> **Note**
>
> RabbitMQ ユーザーは、Amazon MQ のユーザー API 経由で保存または表示されません。

> **Important**
>
> Amazon MQ for RabbitMQ では、ユーザー名「guest」はサポートされず、デフォルトのゲストアカウントは新しいブローカーの作成時に削除されます。ユーザーが作成した「guest」というアカウントも、Amazon MQ によって定期的に削除されます。

RabbitMQ Management API を使用して新しいユーザーを作成するには、以下の API エンドポイントとリクエストボディを使用します。*#####*と*#####*を、新しいサインイン認証情報に置き換えます。

```
PUT /api/users/username HTTP/1.1

            {"password":"password","tags":"administrator"}
```

Important

- 個人を特定できる情報 (PII) などの機密情報や秘匿性の高い情報はブローカーのユーザー名に追加しないでください。ブローカーユーザー名は、CloudWatch Logs を含む他の AWS のサービスからアクセスできます。ブローカーのユーザー名は、プライベートデータや機密データとして使用することを意図していません。
- ブローカーの作成時に設定した管理者パスワードを忘れた場合は、認証情報をリセットできません。複数の管理者を作成した場合は、別の管理者ユーザーを使用してログインし、認証情報をリセットまたは再作成できます。管理者ユーザーが 1 人しかない場合は、ブローカーを削除し、新しい認証情報で新しいブローカーを作成する必要があります。ブローカーを削除する前に、メッセージを使用またはバックアップすることをお勧めします。

`tags` キーは必須です。これは、ユーザーのタグのカンマで区切られたリストです。Amazon MQ は、`administrator`、`management`、`monitoring`、および `policymaker` ユーザータグをサポートします。

個々のユーザーに対する許可は、以下の API エンドポイントとリクエストボディを使用して設定できます。*`vhost`* および *`username`* を、独自の情報に置き換えます。デフォルト vhost `/` には、`%2F` を使用します。

```
PUT /api/permissions/vhost/username HTTP/1.1

            {"configure":".*","write":".*","read":".*"}
```

Note

`configure`、`read`、および `write` キーはすべて必須です。

ワイルドカード `.*` 値を使用することによって、このオペレーションは、指定された vhost 内のすべてのキューに対する読み取り、書き込み、および設定許可をユーザーに付与します。RabbitMQ Management API を使用したユーザーの管理の詳細については、「RabbitMQ Management HTTP API」を参照してください。

Amazon MQ for RabbitMQ のブローカーデフォルト

Amazon MQ for RabbitMQ ブローカーを作成するときは、ブローカーのパフォーマンスを最適化するために、Amazon MQ がブローカーポリシーと vhost 制限のデフォルトセットを適用します。Amazon MQ が vhost 制限を適用するのは、デフォルト (/) vhost のみです。Amazon MQ は、新しく作成された vhost にデフォルトポリシーを適用しません。すべての新規および既存のブローカーに対してこれらのデフォルトを維持することが推奨されますが、これらのデフォルトはいつでも変更、上書き、または削除できます。

Amazon MQ は、ブローカーの作成時に選択されたインスタンスタイプとブローカーデプロイモードに基づいてポリシーと制限を作成します。デフォルトポリシーの名前は、以下のように、デプロイモードに従って命名されます。

- 単一インスタンス – `AWS-DEFAULT-POLICY-SINGLE-INSTANCE`
- クラスターデプロイ – `AWS-DEFAULT-POLICY-CLUSTER-MULTI-AZ`

単一インスタンスブローカーの場合、Amazon MQ はポリシーの優先順位値を `0` に設定します。デフォルトの優先順位値を上書きするには、より高い優先順位値を持つ独自のカスタムポリシーを作成することができます。クラスターデプロイの場合、Amazon MQ はブローカーデフォルトに対して優先順位値を 1 に設定します。クラスター用に独自のカスタムポリシーを作成するには、1 を超える優先順位値を割り当てます。

> **Note**
>
> クラシックミラーリングと高可用性 (HA) のため、クラスターデプロイでは `ha-mode` および `ha-sync-mode` のブローカーポリシーが必要になります。
> デフォルトの `AWS-DEFAULT-POLICY-CLUSTER-MULTI-AZ` ポリシーを削除する場合、Amazon MQ は優先順位値が `0` の `ha-all-AWS-OWNED-DO-NOT-DELETE` ポリシー使用します。これは、必要な `ha-mode` および `ha-sync-mode` ポリシーが引き続き有効であることを確実にします。独自のカスタムポリシーを作成する場合、Amazon MQ はポリシー定義に `ha-mode` および `ha-sync-mode` を自動的に付加します。

トピック

- ポリシーと制限の説明
- 推奨されるデフォルト値

ポリシーと制限の説明

以下のリストには、新しく作成されたブローカーに Amazon MQ が適用するデフォルトのポリシーと制限の説明があります。`max-length`、`max-queues`、および `max-connections` の値は、ブローカーのインスタンスタイプとデプロイモードに応じて異なります。これらの値は、推奨されるデフォルト値 セクションにリストされています。

- **`queue-mode: lazy`** (ポリシー) – レイジーキューを有効にします。デフォルトで、キューはメッセージのインメモリキャッシュを保持し、ブローカーがコンシューマーにメッセージを可能な限り速く配信できるようにします。これは、ブローカーのメモリが不足し、高メモリアラームが発生する原因になる場合があります。レイジーキューは、現実的な範囲でできる限り早急にメッセージをディスクに移動しようとします。つまり、通常の動作条件下では、メモリに保持されるメッセージはそれほど多くないということです。レイジーキューを使用することにより、RabbitMQ for Amazon MQ は、はるかに大きなメッセージング負荷とはるかに長いキューをサポートできます。特定のユースケースでは、レイジーキューを使用するブローカーのパフォーマンスがわずかに遅くなる可能性があることに注意してください。これは、メッセージがインメモリキャッシュから配信されるのではなく、ディスクからブローカーに移動されるためです。

  > デプロイモード
  >
  > 単一インスタンス、クラスター

- **`max-length: number-of-messages`** (ポリシー) – キュー内のメッセージ数に対する制限を設定します。クラスターデプロイでは、この制限が、ブローカーの再起動やメンテナンスウィンドウの後などにキューの同期が一時停止されることを防ぎます。

  > デプロイモード
  >
  > クラスター

- **`overflow: reject-publish`** (ポリシー) – キュー内の数が `max-length` 値に達した後、`max-length` ポリシーを持つキューが新しいメッセージを拒否するようにします。キューがオーバーフロー状態になった場合にメッセージが失われないようにするには、ブローカーにメッセージを発行するクライアントアプリケーションがパブリッシャー確認を実装する必要があります。パブリッシャー確認の実装の詳細については、RabbitMQ ウェブサイトの「Publisher Confirms」を参照してください。

デプロイモード

クラスター

- **`max-queues:`** ***`number-of-queues-per-vhost`*** (vhost 制限) – ブローカー内のキューの数に対する制限を設定します。`max-length` ポリシー定義と同様に、クラスターデプロイ内のキュー数の制限は、ブローカーの再起動やメンテナンスウィンドウの後などにキューの同期が一時停止されることを防ぎます。キューの制限は、キューを維持するための過剰な CPU 量の使用も防ぎます。

  デプロイモード

  単一インスタンス、クラスター

- **`max-connections:`** ***`number-of-connections-per-vhost`*** (vhost 制限) – ブローカーへのクライアント接続数に対する制限を設定します。推奨される値に従って接続数を制限すると、ブローカーがメモリアラームを発し、操作を一時停止させる原因となり得るブローカーメモリの過剰な使用を防ぎます。

  デプロイモード

  単一インスタンス、クラスター

推奨されるデフォルト値

Note

`max-length` および `max-queue` のデフォルト制限は、5 kB の平均メッセージサイズに基づいてテストおよび評価されます。メッセージが 5 kB を大幅に超える場合は、`max-length` および `max-queue` 制限を調整して低くする必要があります。

以下の表には、新しく作成されたブローカーに対するデフォルト制限値がリストされています。Amazon MQ は、ブローカーのインスタンスタイプとデプロイモードに従ってこれらの値を適用します。

インスタンスタイプ	デプロイモード	max-length	max-queues	max-connections
t3.micro	単一インスタンス	該当なし	500	500
m5.large	単一インスタンス	該当なし	20,000	4,000
	クラスター	8,000,000	4,000	15,000
m5.xlarge	単一インスタンス	該当なし	30,000	8,000
	クラスター	9,000,000	5,000	20,000
m5.2xlarge	単一インスタンス	該当なし	60,000	15,000
	クラスター	10,000,000	6,000	40,000
m5.4xlarge	単一インスタンス	該当なし	150,000	30,000
	クラスター	12,000,000	10,000	100,000

Amazon MQ for RabbitMQ のサイズ設定ガイドライン

アプリケーションに最適なブローカーインスタンスタイプを選択できます。インスタンスタイプを選択するときは、ブローカーのパフォーマンスに影響する以下の要因を考慮することが重要です。

- クライアントとキューの数
- 送信されるメッセージの量
- メモリに保持されるメッセージ
- 冗長メッセージ

小さいブローカーインスタンスタイプ (t3.micro) は、アプリケーションのパフォーマンスをテストする場合にのみ使用することをお勧めします。本番稼働レベルのクライアントとキュー、高スループット、メモリ内のメッセージ、冗長メッセージには、大きいブローカーインスタンスタイプ (m5.large 以上) が推奨されます。

ブローカーをテストして、ワークロードのメッセージング要件に適したインスタンスタイプとサイズを決定することが重要です。以下のサイズ設定ガイドラインを使用して、アプリケーションに最適なインスタンスタイプを決定してください。

単一インスタンスデプロイのサイズ設定ガイドライン

次の表は、単一インスタンスブローカーの各インスタンスタイプの上限値を示しています。

インスタンスタイプ	Connections	チャンネル	キュー	チャネルあたりのコンシューマー	シャベル
t3.micro	500	1,500	2,500	1,000	150
m5.large	5,000	15,000	30,000	1,000	250
m5.xlarge	10,000	30,000	60,000	1,000	500
m5.2xlarge	20,000	60,000	120,000	1,000	1,000
m5.4xlarge	40,000	120,000	240,000	1,000	2,000

クラスターデプロイのサイズ設定ガイドライン

次の表は、クラスターブローカーの各インスタンスタイプの上限値を示しています。

インスタンスタイプ	キュー	チャネルあたりのコンシューマー	シャベル
m5.large	10,000	1,000	150
m5.xlarge	15,000	1,000	300
m5.2xlarge	20,000	1,000	600

インスタンスタイプ	キュー	チャネルあたりのコンシューマー	シャベル
m5.4xlarge	30,000	1,000	1200

ノードごとに次の接続とチャネルの制限が適用されます。

インスタンスタイプ	Connections	チャンネル
m5.large	5000	15,000
m5.xlarge	10,000	30,000
m5.2xlarge	20,000	60,000
m5.4xlarge	40,000	120,000

クラスターブローカーの正確な制限値は、利用可能なノードの数と、利用可能なノード間で RabbitMQ がどのようにリソースを分散するかに応じて、示されている値よりも低くなる場合があります。制限値を超えた場合は、別のノードへの新しい接続を作成して再試行するか、インスタンスのサイズをアップグレードして最大制限を増やすことができます。

エラーメッセージ

制限を超えると、以下のエラーメッセージが返されます。すべての値は、`m5.large` の単一インスタンスの制限が基になっています。

> **Note**
>
> 以下のメッセージのエラーコードは、使用しているクライアントライブラリによって異なる場合があります。

Connection

```
ConnectionClosedByBroker 500 "NOT_ALLOWED - connection refused: node
connection limit (500) is reached"
```

チャンネル

```
ConnectionClosedByBroker 1500 "NOT_ALLOWED - number of channels opened on
node 'rabbit@ip-10-0-23-173.us-west-2.compute.internal' has reached the
maximum allowed limit of (15,000)"
```

コンシューマー

```
ConnectionClosedByBroker: (530, 'NOT_ALLOWED - reached maximum (1,000) of
consumers per channel')
```

> **Note**
>
> 次のエラーメッセージは、HTTP 管理 API 形式を使用します。

キュー

```
{"error":"bad_request","reason":"cannot declare queue 'my_queue': queue
limit in cluster (30,000) is reached"}]
```

シャベル

```
{"error":"bad_request","reason":"Validation failed\n\ncomponent shovel is
limited to 250 per node\n"}
```

Vhost

```
{"error":"bad_request","reason":"cannot create vhost 'my_vhost': vhost
limit of 4,000 is reached"}
```

Amazon MQ for RabbitMQ のプラグイン

Amazon MQ for RabbitMQ は、この Management API と RabbitMQ ウェブコンソールを動作させる RabbitMQ の Management プラグインをサポートします。ブローカーのユーザーとポリシーの作成と管理には、ウェブコンソールと Management API を使用できます。

管理プラグインに加えて、Amazon MQ for RabbitMQ は以下のプラグインもサポートします。

トピック

- シャベルプラグイン
- フェデレーションプラグイン
- コンシステントハッシュエクスチェンジプラグイン

シャベルプラグイン

Amazon MQ マネージドブローカーは RabbitMQ シャベルをサポートしており、1 つのブローカーインスタンス上にあるキューとエクスチェンジからのメッセージを、別のブローカーインスタンスに移動することを可能にします。シャベルは、疎結合されたブローカーを接続し、メッセージ負荷が高いノードを避けてメッセージを分散するために使用できます。

Amazon MQ マネージド RabbitMQ ブローカーは、動的シャベルをサポートします。動的シャベルはランタイムパラメータを使用して設定され、クライアント接続によってプログラム的にいつでも開始および停止できます。例えば、RabbitMQ Management API を使用して、以下の API エンドポイントに対する PUT リクエストを作成し、動的シャベルを設定することができます。この例では、{vhost} をブローカーの vhost の名前、{name} を新しい動的シャベルの名前に置き換えることができます。

```
/api/parameters/shovel/{vhost}/{name}
```

リクエストボディでは、キューまたはエクスチェンジのどちらかを指定する必要がありますが、両方を指定する必要はありません。以下の例は、src-queue で指定されたローカルキューと、dest-queue で定義されたリモートキューの間で動的シャベルを設定します。同様に、src-exchange および dest-exchange パラメータを使用して、2 つのエクスチェンジ間でシャベルを設定することもできます。

```
{
                "value": {
                    "src-protocol": "amqp091",
                    "src-uri":  "amqp://localhost",
                    "src-queue":  "source-queue-name",
                    "dest-protocol": "amqp091",
                    "dest-uri": "amqps://b-c8352341-ec91-4a78-ad9c-a43f23d325bb.mq.us-
west-2.amazonaws.com:5671",
                    "dest-queue": "destination-queue-name"
                }
                }
```

> **Important**
>
> シャベル先がプライベートブローカーの場合は、キューまたはエクスチェンジの間でシャベルを構成することはできません。

動的シャベルの使用の詳細については、「RabbitMQ dynamic shovel plugin」を参照してください。

> **Note**
>
> Amazon MQ は、静的シャベルの使用をサポートしません。

フェデレーションプラグイン

Amazon MQ は、フェデレートされたエクスチェンジとキューをサポートします。フェデレーションを使用すると、個別のブローカー上にあるキュー、エクスチェンジ、およびコンシューマー間でメッセージのフローをレプリケートできます。フェデレートされたキューとエクスチェンジは、他のブローカー内のピアへの接続にポイントツーポイントリンクを使用します。フェデレートされたエクスチェンジでは、デフォルトでメッセージが 1 回送信されますが、フェデレートされたキューでは、コンシューマーが必要とする回数だけメッセージを移動できます。

フェデレーションを使用して、アップストリームのエクスチェンジまたはキューからのメッセージをダウンストリームブローカーが消費できるようにすることが可能です。RabbitMQ ウェブコンソールまたは Management API を使用して、ダウンストリームブローカーでフェデレーションを有効にできます。

> **Important**
>
> アップストリームキューまたはエクスチェンジがプライベートブローカーにある場合は、フェデレーションを設定できません。フェデレーションは、パブリックブローカーのキューまたはエクスチェンジの間、または、パブリックブローカーのアップストリームキューかエクスチェンジと、プライベートブローカーのダウンストリームキューかエクスチェンジの間のみ設定できます。

例えば、Management API を使用して以下を実行することにより、フェデレーションを設定できます。

- 他のノードへのフェデレーション接続を定義する 1 つ、または複数のアップストリームを設定する。フェデレーション接続は、RabbitMQ ウェブコンソールまたは Management API を使用して定義できます。Management API を使用して、以下のリクエストボディで `/api/parameters/federation-upstream/%2f/my-upstream` に対する POST リクエストを作成できます。

```
{"value":{"uri":"amqp://server-name","expires":3600000}}
```

- キューまたはエクスチェンジがフェデレートされるようにするポリシーを設定する。ポリシーは、RabbitMQ ウェブコンソールまたは Management API を使用して設定できます。Management API を使用して、以下のリクエストボディで `/api/policies/%2f/federate-me` に対する POST リクエストを作成できます。

```
{"pattern":"^amq\.", "definition":{"federation-upstream-set":"all"}, "apply-
to":"exchanges"}
```

> **Note**
>
> リクエストボディは、サーバー上のエクスチェンジの名前が amq で始まることを前提としています。正規表現 `^amq\.` の使用は、名前が「amq」で始まるすべてのエクスチェンジに対してフェデレーションが有効化されることを確実にします。RabbitMQ サーバー上のエクスチェンジには、異なる名前を付けることができます。

フェデレーションプラグインの設定に関する詳細については、「RabbitMQ federation plugin」を参照してください。

コンシステントハッシュエクスチェンジプラグイン

デフォルトで、Amazon MQ for RabbitMQ はコンシステントハッシュエクスチェンジタイプのプラグインをサポートします。コンシステントハッシュエクスチェンジは、メッセージのルーティングキーから計算されたハッシュ値に基づいてメッセージをキューに送信します。合理的に均等なルーティングキーが提供されると、コンシステントハッシュエクスチェンジはキュー間でメッセージを合理的にむらなく分散できます。

コンシステントハッシュエクスチェンジにバインドされたキューの場合、バインディングキーは各キューのバインドの重みを決定する文字列数値です。バインドの重みが高いキューでは、それらがバインドされているコンシステントハッシュエクスチェンジから受け取るメッセージの配分が相対的に高くなります。コンシステントハッシュエクスチェンジトポロジでは、パブリッシャーは単にメッ

セージをエクスチェンジに発行できますが、コンシューマーは特定のキューからのメッセージを消費するように明示的に設定される必要があります。

コンシステントハッシュエクスチェンジの詳細については、GitHub ウェブサイトの「RabbitMQ Consistent Hash Exchange Type」を参照してください。

Amazon MQ for RabbitMQ へのポリシーの適用

Amazon MQ の推奨デフォルト値を持つカスタムのポリシーと制限を適用できます。推奨されるデフォルトポリシーと制限を削除したが、それらを再作成したい、または追加の vhost を作成して、新しい vhost にデフォルトのポリシーと制限を適用したいという場合は、以下のステップを実行できます。

Important

Amazon MQ for RabbitMQ エンジンバージョン 3.12 以前では、現在のデフォルトのオペレータポリシーは次のとおりです。

```
vhost name pattern apply-to definition priority/
 default_operator_policy_AWS_managed .* all {"queue-version":2} 0
```

バージョン 3.13 以降では、デフォルトの演算子ポリシーが次のように変更されました。

```
vhost name pattern apply-to definition priority/
 default_operator_policy_AWS_managed .* classic_queues {"ha-mode":"all","ha-
sync-mode":"automatic","queue-version":2} 0
```

この更新では、RabbitMQ アプリケーションの動作に機能変更はありません。
従来のミラーキューとクォーラムキューの両方に適用されるポリシーを作成することはできません。ポリシーをクォーラムキューにのみ適用する場合は、`--apply-to` を `quorum_queues` に設定する必要があります。クラシックミラーキューとクォーラムキューを使用している場合は、 `--apply-to:classic_queues`とクォーラムキューポリシーで別のポリシーを作成する必要があります。

Important

以下のステップを実行するには、管理者権限を持つ Amazon MQ for RabbitMQ ブローカーユーザーが必要です。ブローカーを初めて作成したときに作成された管理者ユーザー、また

はその後で作成した別のユーザーを使用できます。以下の表は、正規表現 (regexp) パターンとしての必要な管理者ユーザータグと許可です。

[タグ]	読み込み regexp	設定 regexp	書き込み regexp
`administrator`	`.*`	`.*`	`.*`

RabbitMQ ユーザーの作成、およびユーザータグと許可の管理の詳細については、「Amazon MQ for RabbitMQ ブローカーのユーザー」を参照してください。

RabbitMQ ウェブコンソールを使用してデフォルトのポリシーと仮想ホスト制限を適用する

1. Amazon MQ コンソールにサインインします。
2. 左側のナビゲーションペインで [Brokers] (ブローカー) をクリックします。
3. ブローカーのリストから、新しいポリシーを適用するブローカーの名前を選択します。
4. ブローカーの詳細ページの [Connections] (接続) セクションで、RabbitMQ ウェブコンソール URL をクリックします。RabbitMQ ウェブコンソールが新しいブラウザタブまたはウィンドウで開きます。
5. ブローカー管理者のユーザー名とパスワードを使用して RabbitMQ ウェブコンソールにログインします。
6. RabbitMQ ウェブコンソールのページ上部で、[Admin] (管理) をクリックします。
7. [Admin] (管理) ページの右側にあるナビゲーションペインで [Policies] (ポリシー) をクリックします。
8. [Policies] (ポリシー) ページに、ブローカーの現在の [User policies] (ユーザーポリシー) が表示されます。[User policies] (ユーザーポリシー) の下で、[Add / update a policy] (ポリシーの追加/更新) を展開します。
9. 新しいブローカーポリシーを作成するには、[Add / update a policy] (ポリシーの追加/更新) で以下を実行します。

    a. [Virtual host] (仮想ホスト) には、ドロップダウンリストからポリシーをアタッチする仮想ホストの名前を選択します。デフォルト vhost を選択するには、[/] を選択します。

> **Note**
>
> 追加の vhost を作成していない場合は、RabbitMQ コンソールに [Virtual host] (仮想ホスト) オプションが表示されず、デフォルト vhost のみにポリシーが適用されます。

b. [Name] (名前) には、ポリシーの名前 (**`policy-defaults`** など) を入力します。

c. [Pattern] (パターン) には regexp パターン **`.*`** を入力して、ポリシーがブローカー上のすべてのキューと一致するようにします。

d. [Apply to] (適用先) には、ドロップダウンリストから [Exchanges and queues] (エクスチェンジとキュー) を選択します。

e. [Priority] (優先順位) には、vhost に適用されたその他すべてのポリシーよりも大きい整数を入力します。RabbitMQ のキューとエクスチェンジに適用できるのは、常に 1 つのポリシー定義セットのみです。RabbitMQ は、一致するポリシーで、最高の優先順位値を持つものを選択します。ポリシーの優先順位とポリシーの結合方法の詳細については、RabbitMQ サーバードキュメントの「Policies」を参照してください。

f. [Definition] (定義) には、以下のキーバリューペアを追加します。

- **`queue-mode=lazy`**。ドロップダウンリストから [String] (文字列) を選択します。
- **`overflow=reject-publish`**。ドロップダウンリストから [String] (文字列) を選択します。

  > **Note**
  >
  > 単一インスタンスブローカーには適用されません。

- **`max-length=`*`number-of-messages`***。*`number-of-messages`* は、ブローカーのインスタンスサイズとデプロイモードに従った Amazon MQ の推奨値 (例えば、`mq.m5.large` クラスターには **`8000000`**) に置き換えます。ドロップダウンリストから [Number] (数値) を選択します。

  > **Note**
  >
  > 単一インスタンスブローカーには適用されません。

g. [Add / update policy] (ポリシーを追加/更新) をクリックします。

10. [User policies] (ユーザーポリシー) リストに新しいポリシーが表示されることを確認します。

> **Note**
>
> クラスターブローカーの場合、Amazon MQ が `ha-mode: all` および `ha-sync-mode: automatic` ポリシー定義を自動的に適用します。

11. 右側のナビゲーションペインで [Limits] (制限) をクリックします。
12. [Limits] (制限) ページに、ブローカーの現在の [Virtual host limits] (仮想ホストの制限) が表示されます。[Virtual host limits] (仮想ホスト制限) で、[Set / update a virtual host limit] (仮想ホスト制限の設定/更新) を展開します。
13. 新しい vhost 制限を作成するには、[Set / update a virtual host limit] (仮想ホスト制限の設定/更新) で以下を実行します。
    a. [Virtual host] (仮想ホスト) には、ドロップダウンリストからポリシーをアタッチする仮想ホストの名前を選択します。デフォルト vhost を選択するには、[/] を選択します。
    b. [Limit] (制限) には、ドロップダウンオプションから [max-connections] を選択します。
    c. [Value] (値) には、ブローカーのインスタンスサイズとデプロイモードに従った Amazon MQ の推奨値 (例えば、`mq.m5.large` クラスターには **`15000`**) を入力します。
    d. [Set / update limit] (制限を設定/更新) をクリックします。
    e. 上記のステップを繰り返します。[Limit] (制限) には、ドロップダウンオプションから [max-queues] を選択します。
14. 新しい制限が [Virtual host limits] (仮想ホスト制限) リストにが表示されていることを確認します。

RabbitMQ Management API を使用してデフォルトのポリシーと仮想ホスト制限を適用する

1. Amazon MQ コンソールにサインインします。
2. 左側のナビゲーションペインで [Brokers] (ブローカー) をクリックします。
3. ブローカーのリストから、新しいポリシーを適用するブローカーの名前を選択します。
4. ブローカーのページの [Connections] (接続) セクションで、RabbitMQ ウェブコンソール URL をメモします。これは、HTTP リクエストで使用するブローカーエンドポイントです。
5. 任意の新しいターミナルまたはコマンドラインウィンドウを開きます。

6. 新しいブローカーポリシーを作成するには、以下の curl コマンドを入力します。このコマンドでは、%2F としてエンコードされているデフォルト / vhost 上のキューを前提としています。別の vhost にポリシーを適用するには、%2F をその vhost の名前に置き換えてください。

> **Note**
>
> *#####*と*#####*を、管理者のサインイン認証情報に置き換えます。*number-of-messages* を、ブローカーのインスタンスサイズとデプロイモードに従った Amazon MQ の推奨値に置き換えます。*policy-name* をポリシーの名前に置き換えます。*broker-endpoint* を先ほどメモした URL に置き換えます。

```
curl -i -u username:password -H "content-type:application/json" -XPUT \
-d '{"pattern":".*", "priority":1, "definition":{"queue-mode":lazy,
 "overflow":"reject-publish", "max-length":"number-of-messages"}}' \
broker-endpoint/api/policies/%2F/policy-name
```

7. 新しいポリシーがブローカーのユーザーポリシーに追加されていることを確認するには、以下の curl コマンドを入力して、すべてのブローカーポリシーをリストします。

```
curl -i -u username:password broker-endpoint/api/policies
```

8. 新しい max-connections 仮想ホスト制限を作成するには、以下の curl コマンドを入力します。このコマンドでは、%2F としてエンコードされているデフォルト / vhost 上のキューを前提としています。別の vhost にポリシーを適用するには、%2F をその vhost の名前に置き換えてください。

> **Note**
>
> *#####*と*#####*を、管理者のサインイン認証情報に置き換えます。*max-connections* を、ブローカーのインスタンスサイズとデプロイモードに従った Amazon MQ の推奨値に置き換えます。ブローカーエンドポイントを先ほどメモした URL に置き換えます。

```
curl -i -u username:password -H "content-type:application/json" -XPUT \
-d '{"value":"number-of-connections"}' \
```

```
broker-endpoint/api/vhost-limits/%2F/max-connections
```

9. 新しい max-queues 仮想ホスト制限を作成するには、前のステップを繰り返しますが、curl コマンドを以下のように変更します。

```
curl -i -u username:password -H "content-type:application/json" -XPUT \
-d '{"value":"number-of-queues"}' \
broker-endpoint/api/vhost-limits/%2F/max-queues
```

10. 新しい制限がブローカーの仮想ホスト制限に追加されていることを確認するには、以下の curl コマンドを入力して、すべてのブローカー仮想ホスト制限をリストします。

```
curl -i -u username:password broker-endpoint/api/vhost-limits
```

Amazon MQ for RabbitMQ ブローカーのデプロイオプション

RabbitMQ ブローカーは、単一インスタンスブローカーとして、またはクラスターデプロイで作成できます。どちらのデプロイモードでも、Amazon MQ はデータを冗長的に保存することによって優れた耐久性を提供します。

RabbitMQ ブローカーには、RabbitMQ がサポートする任意のプログラミング言語を使用し、以下のプロトコルに対して TLS を有効にすることによってアクセスできます。

- AMQP (0-9-1)

トピック

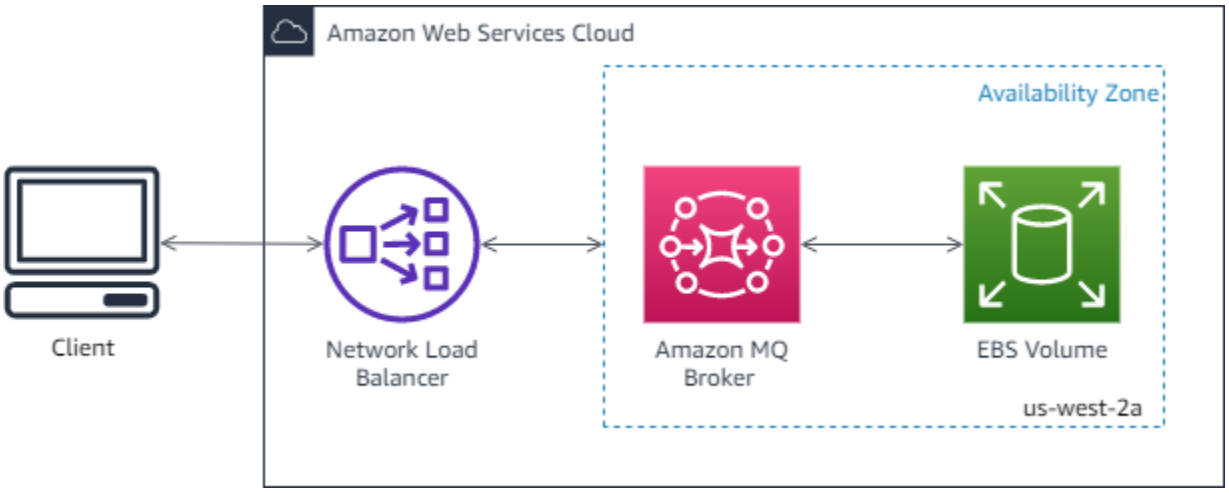
- オプション 1: Amazon MQ for RabbitMQ 単一インスタンスブローカー
- オプション 2: Amazon MQ for RabbitMQ クラスターデプロイ

オプション 1: Amazon MQ for RabbitMQ 単一インスタンスブローカー

単一インスタンスブローカーは、ネットワークロードバランサー (NLB) の内側にある 1 つのアベイラビリティーゾーン内の 1 つのブローカーで構成されます。ブローカーは、アプリケーション、および Amazon EBS ストレージボリュームと通信します。Amazon EBS は、低レイテンシーと高スループット向けに最適化されたブロックレベルのストレージを提供します。

ネットワークロードバランサーの使用は、メンテナンスウィンドウ中に、または基盤となる Amazon EC2 ハードウェア障害が理由でブローカーインスタンスが置き換えられた場合でも、Amazon MQ for RabbitMQ ブローカーエンドポイントがそのまま変更されないことを確実にします。ネットワークロードバランサーは、アプリケーションとユーザーが引き続き同じエンドポイントを使用してブローカーに接続できるようにします。

以下の図は、Amazon MQ for RabbitMQ の単一インスタンスブローカーを示しています。



オプション 2: Amazon MQ for RabbitMQ クラスターデプロイ

クラスターデプロイは、ネットワークロードバランサーの内側にある 3 つの RabbitMQ ブローカーノードの論理グループで、それぞれがユーザー、キュー、および複数のアベイラビリティーゾーン (AZ) 間の分散状態を共有します。

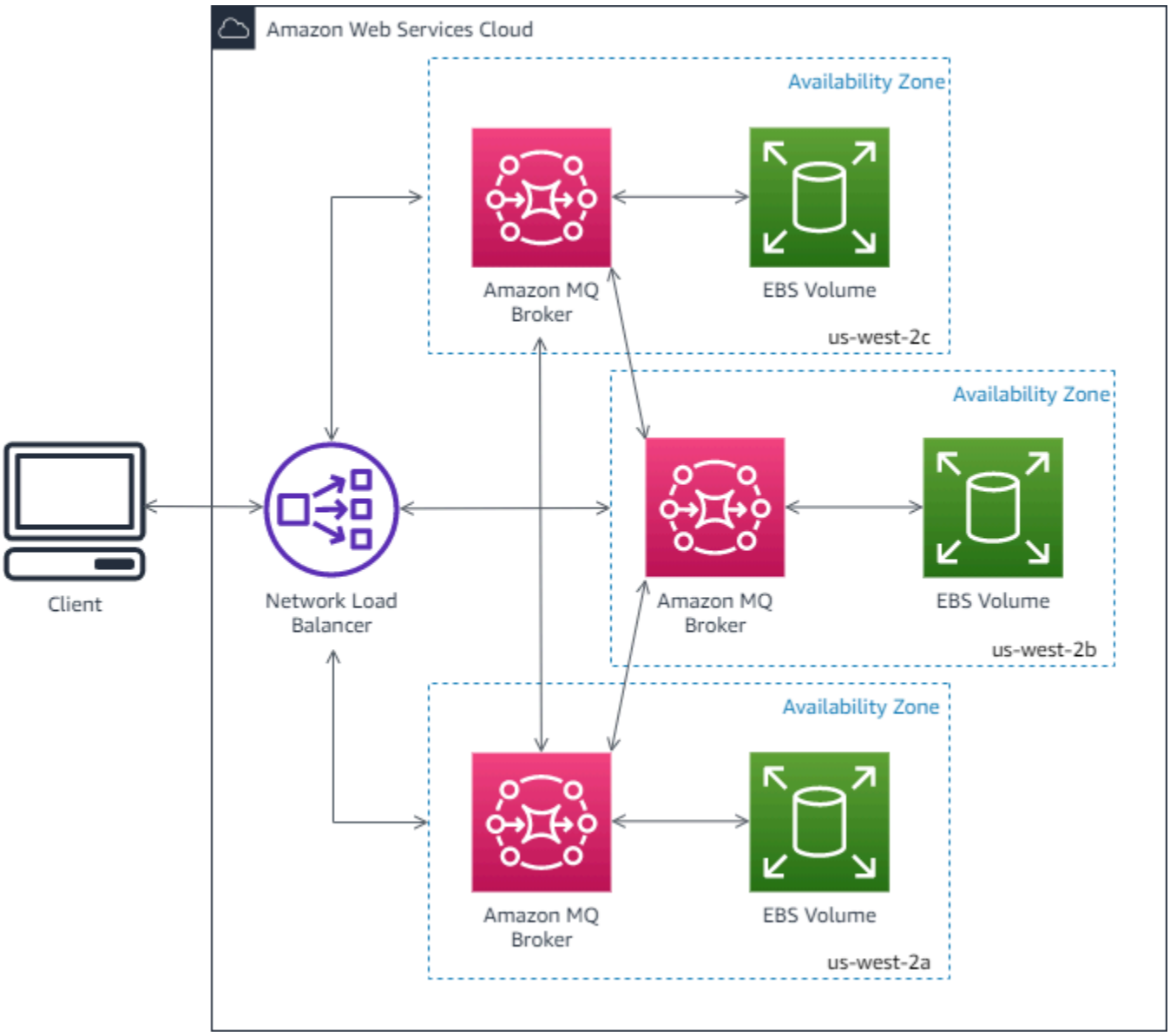
クラスターデプロイでは、Amazon MQ がブローカーポリシーを自動的に管理してすべてのノードでクラシックミラーリングを有効にするため、高可用性 (HA) が確保されます。ミラーされたキューはそれぞれ、1 つのメインノードと、1 つ、または複数のミラーで構成されます。各キューには独自のメインノードがあります。所定のキューに対するすべての操作は、まずキューのメインノードに適用されてから、ミラーに伝播されます。Amazon MQ は、`ha-mode` を `all`、および `ha-sync-mode` を `automatic` に設定するデフォルトのシステムポリシーを作成します。これは、より優れた耐久性のために、異なるアベイラビリティーゾーンにまたがるクラスター内のすべてのノードにデータがレプリケートされることを確実にします。

> **Note**
>
> メンテナンスウィンドウ中、クラスターに対するメンテナンスはすべて一度に 1 ノードずつ実行されるので、少なくとも 2 つのノードが常に実行され続けます。ノードへのクライアント接続は、ノードがダウンするたびに切断され、再確立されなければなりません。クライア

ントコードが、クラスターに自動的に再接続するように設計されていることを確認する必要があります。接続リカバリの詳細については、「the section called “ネットワーク障害から自動的に回復する”」を参照してください。
Amazon MQ は `ha-sync-mode: automatic` を設定するため、メンテナンスウィンドウ中、各ノードがクラスターに再参加するときにキューが同期されます。キューの同期は、その他すべてのキュー操作をブロックします。メンテナンスウィンドウ中におけるキューの同期の影響は、キューを短くしておくことによって軽減できます。

デフォルトポリシーは削除しないようにしてください。このポリシーを削除しても、Amazon MQ によって自動的に再作成されます。また、Amazon MQ は、クラスターブローカーで作成するその他すべてのポリシーに HA プロパティが適用されることも確実にします。HA プロパティのないポリシーを追加すると、Amazon MQ がそれらのプロパティを追加します。異なる高可用性プロパティを持つポリシーを追加すると、Amazon MQ がプロパティを置き換えます。クラシックミラーリングの詳細については、「Classic mirrored queues」を参照してください。

以下の図は、それぞれが独自の Amazon EBS ボリュームと共有状態を持つ 3 つのアベイラビリティーゾーン (AZ) 内に 3 つのノードがある RabbitMQ クラスターブローカーデプロイを示しています。Amazon EBS は、低レイテンシーと高スループット向けに最適化されたブロックレベルのストレージを提供します。



Amazon MQ for RabbitMQ ブローカーのインスタンスタイプ

ブローカーインスタンスクラス (`m5`、`t3`) とサイズ (`large`、`micro`) を組み合わせた説明は、ブローカーインスタンスタイプ (など) と呼ばれます`mq.m5.large`。次の表に、RabbitMQ ブローカーで使用できる Amazon MQ ブローカーインスタンスタイプを示します。 RabbitMQ

Amazon MQ は、インスタンスタイプがサポートを終了する少なくとも 90 日前に通知します。中断を防ぐために、end-of-support前にブローカーを新しいインスタンスタイプにアップグレードすることをお勧めします。

> **Important**
>
> ブローカーを mq.m5. インスタンスタイプから mq.t3.micro インスタンスタイプにダウングレードすることはできません。

インスタンスタイプ	vCPU	メモリ (GiB)	推奨使用法	ストレージ	サポートの終了
mq.t3.micro	2	1	評価 **Important** mq.t3.micro インスタンスタイプはクラスターデプロイをサポートしません。	EBS	
mq.m5.large	2	8	本番稼働	EBS	
mq.m5.xlarge	4	16	本番稼働	EBS	

インスタンスタイプ	vCPU	メモリ (GiB)	推奨使用法	ストレージ	サポートの終了
mq.m5.2xlarge	8	32	本番稼働	EBS	
mq.m5.4xlarge	16	64	本番稼働	EBS	

Amazon MQ for RabbitMQ ブローカーの設定

configuration には、RabbitMQ ブローカーのすべての設定が Cuttlefish 形式で含まれています。設定は、ブローカーを作成する前に作成することができます。作成後、設定を 1 つ、または複数のブローカーに適用できます。

属性

ブローカー設定には複数の属性があります。次に例を示します。

- 名前 (MyConfiguration)
- ID (c-1234a5b6-78cd-901e-2fgh-3i45j6k178l9)
- Amazon リソースネーム (ARN) (arn:aws:mq:us-east-2:123456789012:configuration:c-1234a5b6-78cd-901e-2fgh-3i45j6k178l9)

設定属性の完全なリストについては、Amazon MQ REST API リファレンスで以下を参照してください。

- REST オペレーション ID: 設定
- REST オペレーション ID: 設定

設定のリビジョン属性の詳細なリストについては、以下を参照してください。

- REST オペレーション ID: 設定のリビジョン
- REST オペレーション ID: 設定のリビジョン

トピック

- RabbitMQ ブローカー設定の作成と適用
- Amazon MQ for RabbitMQ 設定リビジョンの編集
- Amazon MQ 上の RabbitMQ の設定可能な値

RabbitMQ ブローカー設定の作成と適用

configuration には、ActiveMQ ブローカーのすべての設定が Cuttlefish 形式で含まれています。設定は、ブローカーを作成する前に作成することができます。次に、設定を 1 つ以上のブローカーに適用できます。

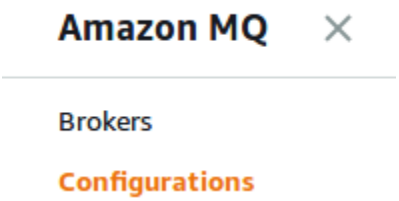
以下の例では、 AWS Management Consoleを使用して Amazon MQ ブローカーの設定を作成および適用する方法を示します。

> **Important**
>
> 設定は `DeleteConfiguration` API を使用してのみ削除できます。詳細については、Amazon MQ API リファレンス」の「設定」を参照してください。

新しい設定の作成

設定をブローカーに適用するには、まず設定を作成する必要があります。

1. Amazon MQ コンソールにサインインします。
2. 左側のナビゲーションパネルを展開し、[設定] を選択します。



3. [設定] ページで、[Create configuration (設定の作成)] を選択します。
4. [Create configuration] (設定の作成) ページの [Details] (詳細) セクションで [Configuration name] (設定名)(`MyConfiguration` など) を入力し、ブローカーエンジンのバージョンを選択します。

   Amazon MQ for ActiveMQ がサポートする RabbitMQ エンジンバージョンの詳細については、「the section called “バージョン管理”」を参照してください。

5. [設定を作成] を選択します。

新しい設定リビジョンの作成

設定を作成したら、設定リビジョンを使用して設定を編集する必要があります。

1. 設定リストから、[***MyConfiguration***] を選択します。

> **Note**
>
> 設定の最初のリビジョンは常に、Amazon MQ が設定を作成するときに作成されます。

[***MyConfiguration***] ページに、新しい設定リビジョンが使用するブローカーのエンジンタイプとバージョン (例: RabbitMQ 3.xx.xx) が表示されます。

2. [設定の詳細] タブに、設定リビジョン番号、説明、およびブローカー設定が Cuttlefish 形式で表示されます。

> **Note**
>
> 現在の設定を編集すると、設定の新しいリビジョンが作成されます。

3. [設定の編集] を選択して、Cuttlefish 設定を変更します。
4. [保存] を選択します。

[Save revision] (リビジョンの保存) ダイアログボックスが表示されます。

5. (オプション) `A description of the changes in this revision` を入力します。
6. [保存] を選択します。

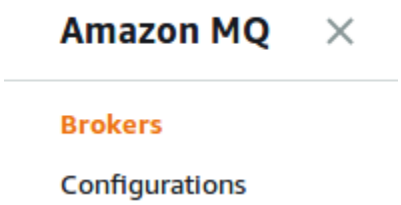
設定の新しいリビジョンが保存されます。

> **Important**
>
> 設定を変更しても、その変更はブローカーに直ちに適用されません。変更を適用するには、次のメンテナンスウィンドウまで待機するか、ブローカーを再起動する必要があります。
> 現在、設定を削除することはできません。

設定リビジョンをブローカーに適用する

設定リビジョンを作成したら、設定リビジョンをブローカーに適用できます。

1. 左側のナビゲーションパネルを展開し、[Brokers (ブローカー)] を選択します。



2. ブローカーリストからブローカーを選択して (MyBroker など)、[Edit] (編集) をクリックします。
3. [Edit ***MyBroker***] (MyBroker の編集) ページの [Configuration] (設定) セクションで [Configuration] (設定) と [Revision] (リビジョン) を選択してから、[Schedule Modifications] (変更をスケジュールする) をクリックします。
4. [ブローカー変更のスケジュール] セクションで、変更を [次回のスケジュールされたメンテナンスウィンドウ中] に適用するか、[即時] 適用するかを選択します。

> **Important**
>
> 1 つのインスタンスブローカーは再起動中にオフラインになります。クラスターブローカーの場合、ブローカーの再起動中に一度にダウンするノードは 1 つだけです。

5. [Apply] (適用) をクリックします。

   設定リビジョンが指定された時刻にブローカーに適用されます。

Amazon MQ for RabbitMQ 設定リビジョンの編集

以下の手順では、ブローカーの設定リビジョンを編集する方法について説明します。

1. Amazon MQ コンソールにサインインします。
2. ブローカーリストからブローカーを選択して (MyBroker など)、[Edit] (編集) をクリックします。
3. [***MyBroker***] ページで、[編集] を選択します。
4. [Edit ***MyBroker***] (MyBroker の編集) ページの [Configuration] (設定) セクションで [Configuration] (設定) と [Revision] (リビジョン) を選択してから、[Edit] (変更) をクリックします。

> **Note**
>
> ブローカーの作成時に設定を選択する場合を除き、最初のリビジョンは、常に Amazon MQ がブローカーを作成する時に作成されます。

[***MyBroker***] ページに、設定が使用するブローカーのエンジンタイプとバージョン (RabbitMQ 3.xx.xx など) が表示されます。

5. [設定の詳細] タブに、設定リビジョン番号、説明、およびブローカー設定が Cuttlefish 形式で表示されます。

> **Note**
>
> 現在の設定を編集すると、設定の新しいリビジョンが作成されます。

6. [設定の編集] を選択して、Cuttlefish 設定を変更します。
7. [保存] を選択します。

   [Save revision] (リビジョンの保存) ダイアログボックスが表示されます。
8. (オプション) `A description of the changes in this revision` を入力します。
9. [保存] を選択します。

   設定の新しいリビジョンが保存されます。

> **Important**
>
> 設定を変更しても、その変更はブローカーに直ちに適用されません。変更を適用するには、次のメンテナンスウィンドウまで待機するか、ブローカーを再起動する必要があります。
> 現在、設定を削除することはできません。

Amazon MQ 上の RabbitMQ の設定可能な値

AWS Management Consoleでブローカー設定ファイルを変更することで、以下のブローカー設定オプションの値を設定できます。

設定	デフォルト値	推奨値	値	適用可能なバージョン	説明
consumer_timeout	1800000 ミリ秒 (30 分)	1800000 ミリ秒 (30 分)	0 ~ 2,147,483,647 ミリ秒。Amazon MQ は、「無限」を意味する値 0 をサポートします。	すべてのバージョン	コンシューマーの配信承認のタイムアウト。コンシューマーが配信を受け付けない状況を検出するために使用されます。
heartbeat	60 秒	60 秒	60 ~ 3600 秒	すべてのバージョン	RabbitMQ で接続が利用できないと見なされるまでの時間を定義します。
management.restrictions.operator_policy_changes.disabled	真	真	true、false	3.11 以降	オペレーターポリシーへの変更を無効にします。この変更を行う場合は、HA プロパティを独自のオペレーターポリシーに含めることを強くお勧めします。
quorum_queue.property_equiv	真	真	true、false	3.13 以降	TRUE に設定すると、アプリケーション

設定	デフォルト値	推奨値	値	適用可能なバージョン	説明
alence.relaxed_checks_on_redeclaration					はクォーラムキューを再宣言するときのチャネル例外を回避します。
secure.management.http.headers.enabled	2024 年 7 月 9 日以降に作成された 3.10 のブローカーでは true。2024 年 7 月 9 日より前に作成されたブローカーでは false。	真	true または false	3.10 以降	変更不可能な HTTP セキュリティヘッダーを有効にします。

コンシューマーの配信承認の設定

consumer_timeout を設定すると、コンシューマーから配信承認が届かない状況を検出できます。コンシューマーがタイムアウト値の時間内に承認を送信しない場合、チャネルは閉じられます。例えば、デフォルト値の 1800000 ミリ秒を使用している場合は、コンシューマーが 1800000 ミリ秒以内に配信承認を送信しないとチャネルが閉じられます。

ハートビートの設定

ハートビートタイムアウトを設定すると、接続の中断や失敗が発生したときに検出できます。ハートビート値は、接続がダウンと見なされるまでの時間制限を定義します。

オペレーターポリシーの設定

各仮想ホストのデフォルトのオペレーターポリシーには、以下の推奨される HA プロパティがあります。

```
{
  "name": "default_operator_policy_AWS_managed",
  "pattern": ".*",
  "apply-to": "all",
  "priority": 0,
  "definition": {
    "ha-mode": "all",
    "ha-sync-mode": "automatic"
  }
}
```

AWS Management Console または Management API を介したオペレーターポリシーの変更は、デフォルトでは利用できません。ブローカー設定に次の行を追加することで変更を有効にすることができます。

```
management.restrictions.operator_policy_changes.disabled=false
```

この変更を行う場合は、HA プロパティを独自のオペレーターポリシーに含めることを強くお勧めします。

キューの宣言時の緩和チェックの設定

従来のキューをクォーラムキューに移行してもクライアントコードを更新していない場合は、`quorum_queue.property_equivalence.relaxed_checks_on_redeclaration` を true に設定することで、クォーラムキューを再宣言するときのチャネル例外を回避できます。

HTTP セキュリティヘッダーの設定

`secure.management.http.headers.enabled` 設定は、次の HTTP セキュリティヘッダーを有効にします。

- X-Content-Type-Options: nosniff: ブラウザによるコンテンツスニッフィングの実行を防ぎます。コンテンツスニッフィングは、ウェブサイトのファイル形式を推定するために使用されるアルゴリズムです。
- X-Frame-Options: DENY: 第三者が独自のウェブサイト上のフレームに管理プラグインを埋め込んで他者をだますことを防ぎます。

- Strict-Transport-Security: max-age=47304000; includeSubDomains: ウェブサイトとそのサブドメインに接続を行うときに、今後長期間 (1.5 年) にわたって HTTPS を使用するようにブラウザに強制します。

バージョン 3.10 以降で作成された Amazon MQ for RabbitMQ ブローカーでは、デフォルトで `secure.management.http.headers.enabled` が `true` に設定されます。`secure.management.http.headers.enabled` を `true` に設定すると、これらの HTTP セキュリティヘッダーが有効になります。これらの HTTP セキュリティヘッダーからオプトアウトする場合は、`secure.management.http.headers.enabled` を `false` に設定します。

Amazon MQ での RabbitMQ のクォーラムキュー

> **Important**
>
> クォーラムキューは、Amazon MQ for RabbitMQ バージョン 3.13 以降のブローカーでのみ使用できます。

クォーラムキューは、1 つのリーダー (プライマリレプリカ) と複数のフォロワー (その他のレプリカ) で構成されるレプリケーション型のキュータイプです。リーダーが利用不可能になった場合、クォーラムキューでは、Raft コンセンサスアルゴリズムを使用して多数決で新しいリーダーノードが選出され、前のリーダーは同じクラスター内のフォロワーノードに降格されます。残りのフォロワーは、以前と同様にレプリケーションを続行します。各ノードは別々のアベイラビリティーゾーンにあるため、1 つのノードが一時的に利用できなくなっても、メッセージ配信は別のアベイラビリティーゾーンにある新しく選出されたリーダーレプリカによって続行されます。

クォーラムキューは、メッセージが失敗し、キューに何度も入れ直されたときに発生する有害メッセージを処理するために役立ちます。

以下の場合はクォーラムキューを使用しないでください。

- 一時キューを使用する場合
- 長いキューバックログがある場合
- 低レイテンシーを優先する場合

クォーラムキューを宣言するには、ヘッダー `x-queue-type` を `quorum` に設定します。

トピック

- Amazon MQ for RabbitMQ での従来のキューからクォーラムキューへの移行
- Amazon MQ for RabbitMQ のクォーラムキューのポリシー設定
- Amazon MQ for RabbitMQ のクォーラムキューのベストプラクティス

Amazon MQ for RabbitMQ での従来のキューからクォーラムキューへの移行

従来のミラーキューを、バージョン 3.13 以降の Amazon MQ ブローカーのクォーラムキューに移行することができます。そのためには、同じクラスター上に新しい仮想ホストを作成する方法と、インプレースで移行する方法があります。

オプション 1: 新しい仮想ホストを使用した従来のミラーキューからクォーラムキューへの移行

同じクラスター上に新しい仮想ホストを作成することで、従来のミラーキューをバージョン 3.13 以降の Amazon MQ ブローカーのクォーラムキューに移行できます。

1. 既存のクラスターで、デフォルトのキュータイプをクォーラムとする新しい仮想ホスト (vhost) を作成します。
2. 新しい vhost から フェデレーションプラグインを作成して、従来のミラーキューを使用する以前の vhost を指す URL を指定します。
3. `rabbitmqadmin` を使用して、以前の vhost から新しいファイルに定義を エクスポートします。このスキーマファイルを変更して、クォーラムキューとの互換性を持たせる必要があります。ファイルに加える必要のある変更の完全なリストについては、RabbitMQ クォーラムキュードキュメントの「Moving definitions」を参照してください。必要な変更をファイルに適用したら、新しい vhost に定義を再インポートします。
4. 新しい vhost に新しいポリシーを作成します。クォーラムキュー向けの Amazon MQ ポリシー設定に関する推奨事項については、「Amazon MQ for RabbitMQ のクォーラムキューのポリシー設定」を参照してください。次に、前の手順で作成した、以前の vhost から新しい vhost へのフェデレーションを開始します。
5. コンシューマーとプロデューサーが新しい vhost を指すように設定します。
6. Shovel プラグインを設定して、残っているメッセージをすべて移動します。キューが空になったら、Shovel を削除します。

従来のミラーキューからクォーラムキューへのインプレース移行

従来のミラーキューを、バージョン 3.13 以降の Amazon MQ ブローカーのクォーラムキューにインプレースで移行できます。

1. コンシューマーとプロデューサーを停止します。
2. 新しい一時クォーラムキューを作成します。
3. Shovel プラグインを設定して、以前の従来のミラーキューから新しい一時クォーラムキューにすべてのメッセージを移動します。すべてのメッセージが一時クォーラムキューに移動されたら、Shovel を削除します。
4. 移行元の従来のミラーキューを削除します。次に、移行元の従来のミラーキューと同じ名前とバインディングでクォーラムキューを再作成します。
5. 新しい Shovel を作成して、一時クォーラムキューから新しいクォーラムキューにメッセージを移動します。

Amazon MQ for RabbitMQ のクォーラムキューのポリシー設定

Amazon MQ 上の RabbitMQ ブローカーのクォーラムキューに、特定のポリシー設定を追加できます。

クォーラムキューのポリシーを作成するときは、次の操作を実行する必要があります。

- ha で始まるポリシー属性をすべて削除します。`ha-mode`、`ha-params`、`ha-sync-mode`、`ha-sync-batch-size`、`ha-promote-on-shutdown`、`ha-promote-on-failure` などが該当します。
- `queue-mode` を削除します。
- オーバーフローが `reject-publish-dlx` に設定されている場合は変更します。

> **Important**
>
> Amazon MQ for RabbitMQ は、ポリシー内のすべての属性を適用するか、一切適用しないかのどちらかの動作になります。従来のミラーキューとクォーラムキューの両方に適用されるポリシーを作成することはできません。ポリシーをクォーラムキューにのみ適用する場合は、`--apply-to` を `quorum_queues` に設定する必要があります。従来のミラーキューと

クォーラムキューを使用している場合は、クォーラムキューポリシーに加えて、`--apply-to:classic_queues` を設定した別のポリシーを作成する必要があります。

`AWS-DEFAULT` ポリシーは、「適用先」パラメータに自動的に新しいキュータイプを採用するため、変更する必要はありません。Amazon MQ for RabbitMQ のデフォルトポリシーの詳細については、「RabbitMQ configuration policies」を参照してください。

Amazon MQ for RabbitMQ のクォーラムキューのベストプラクティス

クォーラムキューを使用する際のパフォーマンスを向上させるには、次のベストプラクティスを使用することをお勧めします。

配信制限を設定して有害メッセージを処理する

有害メッセージは、メッセージが失敗し、何度も再配信される場合に発生します。`delivery-limit` ポリシー引数を使用してメッセージ配信制限を設定すると、何度も再配信されるメッセージを削除できます。配信制限で許容される回数を超えてメッセージが再配信された場合、メッセージは RabbitMQ によってドロップされ、削除されます。配信制限を設定すると、メッセージはキューの先頭の近くに再び入れられます。

クォーラムキューのメッセージ優先度

クォーラムキューにはメッセージ優先度がありません。メッセージ優先度が必要な場合は、複数のクォーラムキューを作成する必要があります。複数のクォーラムキューを持つメッセージの優先度設定の詳細については、RabbitMQ ドキュメントの「Message priority」を参照してください。

デフォルトのレプリケーション係数の使用

Amazon MQ for RabbitMQ では、クォーラムキューを使用するクラスターブローカーのレプリケーション係数はデフォルトで 3 ノードになります。`x-quorum-initial-group-size` に変更を加えると、Amazon MQ は、再びデフォルトでレプリケーション係数 3 を使用するようになります。

RABBITMQ_QUORUM_QUEUES_NOT_SUPPORTED_ON_CURRENT_VERSION のトラブルシューティング

Amazon MQ for RabbitMQ では、バージョン 3.12 以前を使用して単一インスタンスブローカーまたはクラスターブローカーにクォーラムキューを作成しようとすると、重要なアクションを必要と

するコード RABBITMQ_QUORUM_QUEUES_NOT_SUPPORTED_ON_CURRENT_VERSION が発生します。RABBITMQ_QUORUM_QUEUES_NOT_SUPPORTED_ON_CURRENT_VERSION のトラブルシューティングの詳細については、「Amazon MQ クォーラムキューの RabbitMQ アラーム Amazon MQ 」を参照してください。

RabbitMQ のチュートリアル

以下のチュートリアルでは、Amazon MQ で RabbitMQ を設定して使用する方法を説明します。サポートされているクライアントライブラリを Node.js、Python、.NET などのさまざまなプログラミング言語で使用する方法の詳細については、「RabbitMQ Getting Started Guide」の「RabbitMQ Tutorials」を参照してください。

トピック

- ブローカー設定の編集
- Amazon MQ for RabbitMQ でPython Pika を使う
- RabbitMQ の一時停止されたキュー同期の解決
- ステップ 2: ブローカーに JVM ベースのアプリケーションを接続する
- ステップ 3: (オプション) AWS Lambda 関数に接続する

ブローカー設定の編集

を使用して CloudWatch ログの有効化や無効化など、ブローカーの設定を編集できます AWS Management Console。

RabbitMQ ブローカーオプションを編集する

1. Amazon MQ コンソールにサインインします。
2. ブローカーリストからブローカーを選択して (MyBroker など)、[Edit] (編集) をクリックします。
3. [***MyBroker*** の編集] ページの [仕様] セクションで、[ブローカーエンジンのバージョン] または [ブローカーインスタンスタイプ] を選択します。
4. [CloudWatch Logs] セクションのトグルボタンをクリックして、一般ログを有効化または無効化します。これ以上のステップは必要ありません。

> **Note**
>
> - RabbitMQ ブローカーの場合、Amazon MQ は自動的にサービスリンクロール (SLR) を使用して、CloudWatch に一般ログを発行します。詳細については、「the section called “サービスリンクロールの使用”」を参照してください。
> - Amazon MQ は、RabbitMQ ブローカーに対する監査ロギングをサポートしません。

5. [Maintenance (メンテナンス)] セクションで、ブローカーのメンテナンススケジュールを設定します。

   ブローカーを AWS リリース時に新しいバージョンにアップグレードするには、自動マイナーバージョンアップグレードを有効にするを選択します。自動アップグレードは、曜日、時刻 (24 時間形式)、およびタイムゾーン (デフォルトは UTC) で定義されたメンテナンスウィンドウ中に行われます。

6. [Schedule modifications (スケジュールの変更)] を選択します。

   > **Note**
   >
   > [自動マイナーバージョンのアップグレードを有効にする] のみを選択した場合、ブローカーの再起動が必要ないため、ボタンは [保存] に変わります。

   設定が指定された時刻にブローカーに適用されます。

Amazon MQ for RabbitMQ でPython Pika を使う

次のチュートリアルでは、Amazon MQ for RabbitMQ ブローカーに接続するように構成された TLS を使用して Python Pika クライアントをセットアップする方法を示しています。Pika は RabbitMQ のための AMQP 0-9-1 プロトコルの Python 実装です。このチュートリアルでは、Pika のインストール、キューの宣言、ブローカーのデフォルトエクスチェンジにメッセージを送信するパブリッシャーの設定、およびキューからメッセージを受信するようにコンシューマを設定する手順を説明します。

トピック

- 前提条件
- アクセス許可

前提条件

このチュートリアルの最初のステップを完了するには、以下のものが必要です。

- Amazon MQ for RabbitMQ ブローカー。詳細については、「Amazon MQ for RabbitMQ ブローカーを作成する」を参照してください。
- オペレーティングシステム用に Python 3 がインストールされています。
- Python `pip` を使用して、Pika がインストールされました。Pika をインストールするには、新しいターミナルウィンドウを開き、以下を実行します。

```
$ python3 -m pip install pika
```

アクセス許可

このチュートリアルでは、vhost への書き込みおよび読み取りの許可を持つ Amazon MQ for RabbitMQ ブローカーユーザーが少なくとも 1 人必要です。以下の表は、正規表現 (regexp) パターンとして必要な最低限の許可を説明しています。

[タグ]	設定 regexp	書き込み regexp	読み込み regexp
none		.*	.*

リストされているユーザー許可は、ブローカで管理オペレーションを実行するための管理プラグインへのアクセスを付与することなく、ユーザーに読み取りおよび書き込み許可のみを提供します。特定のキューへのユーザーのアクセスを制限する正規表現パターンを提供することで、許可をさらに制限できます。例えば、読み取り regexp パターンを `^[hello world].*` に変更する場合、ユーザーには `hello world` で始まるキューからの読み取り許可のみが付与されます。

RabbitMQ ユーザーの作成、およびユーザータグと許可の管理の詳細については、「Amazon MQ for RabbitMQ ブローカーのユーザー」を参照してください。

ステップ 1: 基本的な Python Pika クライアントを作成する

Amazon MQ for RabbitMQ ブローカーと対話するときに、コンストラクタを定義し、TLS 設定に必要な SSL コンテキストを提供する Python Pika クライアント基本クラスを作成するには、次の手順を実行します。

1. 新しいターミナルウィンドウを開き、プロジェクトの新しいディレクトリを作成し、そのディレクトリに移動します。

```
$ mkdir pika-tutorial
$ cd pika-tutorial
```

2. 以下の Python コードを含む basicClient.py というファイルを作成します。

```
import ssl
import pika

class BasicPikaClient:

    def __init__(self, rabbitmq_broker_id, rabbitmq_user, rabbitmq_password,
 region):

        # SSL Context for TLS configuration of Amazon MQ for RabbitMQ
        ssl_context = ssl.SSLContext(ssl.PROTOCOL_TLSv1_2)
        ssl_context.set_ciphers('ECDHE+AESGCM:!ECDSA')

        url = f"amqps://{rabbitmq_user}:
{rabbitmq_password}@{rabbitmq_broker_id}.mq.{region}.amazonaws.com:5671"
        parameters = pika.URLParameters(url)
        parameters.ssl_options = pika.SSLOptions(context=ssl_context)

        self.connection = pika.BlockingConnection(parameters)
        self.channel = self.connection.channel()
```

パブリッシャーとコンシューマに対して、BasicPikaClient から継承する追加のクラスを定義できるようになりました。

ステップ 2: パブリッシャーを作成してメッセージを送信する

キューを宣言し、1 つのメッセージを送信するパブリッシャを作成するには、次の手順を実行します。

1. 次のコードサンプルの内容をコピーし、前のステップで作成した同じディレクトリで、publisher.py と名前を付けてローカルに保存します。

```
from basicClient import BasicPikaClient

class BasicMessageSender(BasicPikaClient):

    def declare_queue(self, queue_name):
        print(f"Trying to declare queue({queue_name})...")
        self.channel.queue_declare(queue=queue_name)

    def send_message(self, exchange, routing_key, body):
        channel = self.connection.channel()
        channel.basic_publish(exchange=exchange,
                              routing_key=routing_key,
                              body=body)
        print(f"Sent message. Exchange: {exchange}, Routing Key: {routing_key},
 Body: {body}")

    def close(self):
        self.channel.close()
        self.connection.close()

if __name__ == "__main__":

    # Initialize Basic Message Sender which creates a connection
    # and channel for sending messages.
    basic_message_sender = BasicMessageSender(
        "<broker-id>",
        "<username>",
        "<password>",
        "<region>"
    )

    # Declare a queue
    basic_message_sender.declare_queue("hello world queue")

    # Send a message to the queue.
```

```
    basic_message_sender.send_message(exchange="", routing_key="hello world queue",
body=b'Hello World!')

    # Close connections.
    basic_message_sender.close()
```

BasicMessageSender クラスは BasicPikaClient から継承され、キューの宣言、キューへのメッセージの送信、および接続を閉じるための追加のメソッドを実装します。コードサンプルでは、キューの名前と等しいルーティングキーを使用して、メッセージをデフォルトの交換にルーティングします。

2. [if __name__ == "__main__":] で、渡されたパラメータを次の情報を含む BasicMessageSender コンストラクターステートメントで置換します。

   - **<broker-id>** – Amazon MQ がブローカー用に生成する一意の ID です。ID は、ブローカー ARN から解析できます。例えば、arn:aws:mq:us-east-2:123456789012:broker:MyBroker:b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9 という ARN の場合、ブローカー ID は b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9 になります。
   - **<username>** - ブローカにメッセージを書き込むのに十分な許可を持つブローカユーザーのユーザー名。
   - **<password>** - ブローカにメッセージを書き込むのに十分な許可を持つブローカユーザーのパスワード。
   - **<region>** – Amazon MQ for RabbitMQ ブローカーを作成した AWS リージョン。例えば、us-west-2 と指定します。

3. publisher.py を作成した同じディレクトリで次のコマンドを実行します。

```
$ python3 publisher.py
```

コードが正常に実行された場合、ターミナルウィンドウに次の出力が表示されます。

```
Trying to declare queue(hello world queue)...
Sent message. Exchange: , Routing Key: hello world queue, Body: b'Hello World!'
```

ステップ 3: コンシューマを作成してメッセージを受信する

キューから 1 つのメッセージを受信するコンシューマを作成するには、次の手順を実行します。

1. 次のコードサンプルの内容をコピーし、同じディレクトリで、consumer.py と名前を付けてローカルに保存します。

```
from basicClient import BasicPikaClient

class BasicMessageReceiver(BasicPikaClient):

    def get_message(self, queue):
        method_frame, header_frame, body = self.channel.basic_get(queue)
        if method_frame:
            print(method_frame, header_frame, body)
            self.channel.basic_ack(method_frame.delivery_tag)
            return method_frame, header_frame, body
        else:
            print('No message returned')

    def close(self):
        self.channel.close()
        self.connection.close()


if __name__ == "__main__":

    # Create Basic Message Receiver which creates a connection
    # and channel for consuming messages.
    basic_message_receiver = BasicMessageReceiver(
        "<broker-id>",
        "<username>",
        "<password>",
        "<region>"
    )

    # Consume the message that was sent.
    basic_message_receiver.get_message("hello world queue")

    # Close connections.
    basic_message_receiver.close()
```

前のステップで作成したパブリッシャーと同様に、BasicMessageReciever は BasicPikaClient から継承し、単一のメッセージを受信し、接続を閉じるための追加のメソッドを実装します。

2. `if __name__ == "__main__":` ステートメントで、渡されたパラメータを次の情報を含む BasicMessageReciever コンストラクターに置換します。
3. プロジェクトディレクトリで次のコマンドを実行します。

```
$ python3 consumer.py
```

コードが正常に実行されると、メッセージ本文とルーティングキーを含むヘッダーがターミナルウィンドウに表示されます。

```
<Basic.GetOk(['delivery_tag=1', 'exchange=', 'message_count=0',
 'redelivered=False', 'routing_key=hello world queue'])> <BasicProperties> b'Hello
 World!'
```

ステップ 4: (オプション) イベントループを設定し、メッセージを消費する

キューから複数のメッセージを消費するには、Pika の [basic_consume] メソッドと、次に示すコールバック関数を使用します

1. consumer.py で、BasicMessageReceiver クラスに以下のメソッド定義を追加します。

```
def consume_messages(self, queue):
    def callback(ch, method, properties, body):
        print(" [x] Received %r" % body)

    self.channel.basic_consume(queue=queue, on_message_callback=callback,
 auto_ack=True)

    print(' [*] Waiting for messages. To exit press CTRL+C')
    self.channel.start_consuming()
```

2. consumer.py の `if __name__ == "__main__":` の下で、前のステップで定義した consume_messages メソッドを呼び出します。

```
if __name__ == "__main__":

    # Create Basic Message Receiver which creates a connection and channel for
 consuming messages.
    basic_message_receiver = BasicMessageReceiver(
        "<broker-id>",
```

```
        "<username>",
        "<password>",
        "<region>"
    )

    # Consume the message that was sent.
    # basic_message_receiver.get_message("hello world queue")

    # Consume multiple messages in an event loop.
    basic_message_receiver.consume_messages("hello world queue")

    # Close connections.
    basic_message_receiver.close()
```

3. consumer.py をもう一度実行し、成功すると、キューに入れられたメッセージがターミナルウィンドウに表示されます。

```
[*] Waiting for messages. To exit press CTRL+C
[x] Received b'Hello World!'
[x] Received b'Hello World!'
...
```

次のステップ

- サポートされている他の RabbitMQ クライアントライブラリの詳細については、RabbitMQ のウェブサイトの「RabbitMQ クライアントドキュメント」を参照してください。

RabbitMQ の一時停止されたキュー同期の解決

Amazon MQ for RabbitMQ クラスターデプロイでは、各キューに発行されたメッセージが 3 つのブローカーノード全体にレプリケートされます。ミラーリングと呼ばれるこのレプリケーションは、RabbitMQ ブローカーに高可用性 (HA) を提供します。クラスターデプロイ内のキューは、1 つのノード上にあるメインレプリカと、1 つ、または複数のミラーで構成されています。ミラーキューに適用されるすべての操作 (メッセージのキュー登録など) は、まずメインキューに適用され、その後ミラー全体にレプリケートされます。

例えば、メインノード (main) と 2 つのミラー (mirror-1 および mirror-2) の 3 つのノード全体にレプリケートされたミラーキューについて考えてみましょう。このミラーキュー内のすべてのメッセージがすべてのミラーに正常に伝播されると、キューが同期されたことになります。ノード

(mirror-1) が一定期間使用できなくなった場合でも、キューは引き続き動作可能で、メッセージのキュー登録を継続できますが、キューを同期するには、mirror-1 が使用不可である間に main に発行されたメッセージが mirror-1 にレプリケートされる必要があります。

ミラーリングの詳細については、RabbitMQ ウェブサイトで「Classic Mirrored Queues」を参照してください。

メンテナンスとキューの同期

メンテナンスウィンドウ中、Amazon MQ はすべてのメンテナンス作業を一度に 1 ノードずつ実行して、ブローカーが動作可能な状態を維持することを確実にします。その結果、各ノードが操作を再開するときに、キューが同期する必要が生じる場合があります。同期中、ミラーにレプリケートする必要があるメッセージは、バッチで処理されるように、対応する Amazon Elastic Block Store (Amazon EBS) ボリュームからメモリにロードされます。メッセージをバッチで処理することにより、キューの同期が速くなります。

キューを短くし、メッセージを小さくしておくと、キューが正常に同期し、期待通りに操作を再開します。ただし、バッチ内のデータ量がノードのメモリ制限に近づいた場合は、ノードが高メモリアラームを発し、キューの同期を一時停止します。メモリ使用量は、CloudWatch で RabbitMemUsed および RabbitMqMemLimit のブローカーノードメトリクスを比較することで確認できます。同期は、メッセージが消費もしくは削除される、またはバッチ内のメッセージの数が減るまで完了できません。

> **Note**
>
> キューの同期のバッチサイズを小さくすると、レプリケーショントランザクション数の増加につながる可能性があります。

一時停止されたキューの同期を解決するには、ha-sync-batch-size ポリシーの適用とキューの同期の再開について説明する、このチュートリアルのステップに従ってください。

トピック

- 前提条件
- ステップ 1: ha-sync-batch-size ポリシーを適用する
- ステップ 2: キューの同期を再開する
- 次のステップ
- 関連リソース

前提条件

このチュートリアルには、管理者権限を持つ Amazon MQ for RabbitMQ ブローカーユーザーが必要です。ブローカーを初めて作成したときに作成された管理者ユーザー、またはその後で作成した別のユーザーを使用できます。以下の表は、正規表現 (regexp) パターンとしての必要な管理者ユーザータグと許可です。

[タグ]	読み込み regexp	設定 regexp	書き込み regexp
`administrator`	`.*`	`.*`	`.*`

RabbitMQ ユーザーの作成、およびユーザータグと許可の管理の詳細については、「Amazon MQ for RabbitMQ ブローカーのユーザー」を参照してください。

ステップ 1: **ha-sync-batch-size** ポリシーを適用する

以下の手順では、ブローカーで作成されたすべてのキューに適用されるポリシーの追加について説明します。RabbitMQ ウェブコンソールまたは RabbitMQ Management API を使用できます。詳細については、RabbitMQ ウェブサイトの「Management Plugin」を参照してください。

RabbitMQ ウェブコンソールを使用して **ha-sync-batch-size** ポリシーを適用する

1. Amazon MQ コンソールにサインインします。
2. 左側のナビゲーションペインで [Brokers] (ブローカー) をクリックします。
3. ブローカーのリストから、新しいポリシーを適用するブローカーの名前を選択します。
4. ブローカーのページの [Connections] (接続) セクションで、RabbitMQ ウェブコンソール URL をメモします。新しいブラウザタブまたはウィンドウに RabbitMQ ウェブコンソールが開きます。
5. ブローカー管理者のサインイン認証情報を使用して RabbitMQ ウェブコンソールにログインします。
6. RabbitMQ ウェブコンソールのページ上部で、[Admin] (管理) をクリックします。
7. [Admin] (管理) ページの右側にあるナビゲーションペインで [Policies] (ポリシー) をクリックします。
8. [Policies] (ポリシー) ページに、ブローカーの現在の [User policies] (ユーザーポリシー) が表示されます。[User policies] (ユーザーポリシー) の下で、[Add / update a policy] (ポリシーの追加/更新) を展開します。

> **Note**
>
> デフォルトで、Amazon MQ for RabbitMQ クラスターは、`ha-all-AWS-OWNED-DO-NOT-DELETE` という名前の初期ブローカーポリシーを使用して作成されます。Amazon MQ はこのポリシーを管理して、ブローカー上のすべてのキューが 3 つのノードすべてにレプリケートされ、キューが自動的に同期化されることを確実にします。

9. 新しいブローカーポリシーを作成するには、[Add / update a policy] (ポリシーの追加/更新) で以下を実行します。

   a. [Name] (名前) には、ポリシーの名前 (**`batch-size-policy`** など) を入力します。

   b. [Pattern] (パターン) には regexp パターン **`.*`** を入力して、ポリシーがブローカー上のすべてのキューと一致するようにします。

   c. [Apply to] (適用先) には、ドロップダウンリストから [Exchanges and queues] (エクスチェンジとキュー) を選択します。

   d. [Priority] (優先順位) には、vhost に適用されたその他すべてのポリシーよりも大きい整数を入力します。RabbitMQ のキューと エクスチェンジに適用できるのは、常に 1 つのポリシー定義セットのみです。RabbitMQ は、一致するポリシーで、最高の優先順位値を持つものを選択します。ポリシーの優先順位とポリシーの結合方法の詳細については、RabbitMQ サーバードキュメントの「Policies」を参照してください。

   e. [Definition] (定義) には、以下のキーバリューペアを追加します。

      - **`ha-sync-batch-size`**=*`100`*。ドロップダウンリストから [Number] (数値) を選択します。

        > **Note**
        >
        > `ha-sync-batch-size` の値は、キュー内の同期されていないメッセージの数とサイズに基づいて調整と較正を行う必要がある場合があります。

      - **`ha-mode=all`**。ドロップダウンリストから [String] (文字列) を選択します。

        > **Important**
        >
        > `ha-mode` 定義は、すべての HA 関連ポリシーに必須です。省略すると、検証が失敗します。

- **`ha-sync-mode=automatic`**。ドロップダウンリストから [String] (文字列) を選択します。

> **Note**
>
> `ha-sync-mode` 定義は、すべてのカスタムポリシーに必須です。省略すると、Amazon MQ が定義を自動的に付加します。

f. [Add / update policy] (ポリシーを追加/更新) をクリックします。

10. [User policies] (ユーザーポリシー) リストに新しいポリシーが表示されることを確認します。

RabbitMQ Management API を使用して **`ha-sync-batch-size`** ポリシーを適用する

1. Amazon MQ コンソールにサインインします。
2. 左側のナビゲーションペインで [Brokers] (ブローカー) をクリックします。
3. ブローカーのリストから、新しいポリシーを適用するブローカーの名前を選択します。
4. ブローカーのページの [Connections] (接続) セクションで、RabbitMQ ウェブコンソール URL をメモします。これは、HTTP リクエストで使用するブローカーエンドポイントです。
5. 任意の新しいターミナルまたはコマンドラインウィンドウを開きます。
6. 新しいブローカーポリシーを作成するには、以下の `curl` コマンドを入力します。このコマンドでは、`%2F` としてエンコードされているデフォルト / vhost 上のキューを前提としています。

> **Note**
>
> *`#####`*と*`#####`*を、ブローカー管理者のサインイン認証情報に置き換えます。`ha-sync-batch-size` の値 (*`100`*) は、キュー内の同期されていないメッセージの数とサイズに基づいて調整と較正を行う必要がある場合があります。ブローカーエンドポイントを先ほどメモした URL に置き換えます。

```
curl -i -u username:password -H "content-type:application/json" -XPUT \
-d '{"pattern":".*", "priority":1, "definition":{"ha-sync-batch-size":100, "ha-
mode":"all", "ha-sync-mode":"automatic"}}' \
https://b-589c045f-f8ln-4ab0-a89c-co62e1c32ef8.mq.us-west-2.amazonaws.com/api/
policies/%2F/batch-size-policy
```

7. 新しいポリシーがブローカーのユーザーポリシーに追加されていることを確認するには、以下の curl コマンドを入力して、すべてのブローカーポリシーをリストします。

```
curl -i -u username:password https://b-589c045f-f8ln-4ab0-a89c-co62e1c32ef8.mq.us-west-2.amazonaws.com/api/policies
```

ステップ 2: キューの同期を再開する

ブローカーに新しい ha-sync-batch-size ポリシーを適用したら、キューの同期を再開します。

RabbitMQ ウェブコンソールを使用してキューの同期を再開する

> **Note**
>
> RabbitMQ ウェブコンソールを開くには、このチュートリアルのステップ 1 にある前述の手順を参照してください。

1. RabbitMQ ウェブコンソールのページ上部で、[Queues] (キュー) をクリックします。
2. [Queues] (キュー) ページの [All queues] (すべてのキュー) で、一時停止されたキューを見つけます。ポリシー行では、キューに、作成した新しいポリシーの名前 (など) が一覧表示されますbatch-size-policy。
3. バッチサイズを小さくして同期プロセスを再起動するには、まずキューの同期をキャンセルします。次に、キュー同期を再起動します。

> **Note**
>
> 同期が一時停止して正常に終了しない場合は、ha-sync-batch-size の値を低くして、もう一度キューの同期を再開してみてください。

次のステップ

- キューが正常に同期化されたら、Amazon CloudWatch メトリクス RabbitMQMemUsed を表示することで、RabbitMQ ノードが使用するメモリの量をモニタリングできます。RabbitMQMemLimit メトリクスを表示して、ノードのメモリ制限をモニタリングすることも

できます。詳細については、「Amazon MQ 向けの CloudWatch メトリクスへのアクセス」および「Amazon MQ for RabbitMQ ブローカーで利用可能な CloudWatch メトリクス」を参照してください。

- キューの同期が一時停止しないようにするため、キューを短くしておき、メッセージを処理することをお勧めします。メッセージサイズが大きいワークロードの場合は、より多くのメモリを備えたより大きなインスタンスサイズにブローカーインスタンスタイプをアップグレードすることもお勧めします。ブローカーインスタンスタイプとブローカー設定の編集の詳細については、「」を参照してくださいブローカー設定の編集。
- 新しい Amazon MQ for RabbitMQ ブローカーを作成するときは、ブローカーのパフォーマンスを最適化するために、Amazon MQ が一連のデフォルトブローカーポリシーと仮想ホスト制限を適用します。お使いのブローカーに推奨されるデフォルトのポリシーと制限がない場合は、独自のポリシーと制限を作成することをお勧めします。デフォルトのポリシーと vhost 制限の作成に関する詳細については、「the section called “ブローカーのデフォルト”」を参照してください。

関連リソース

- UpdateBrokerInput – Amazon MQ API を使用してブローカーインスタンスタイプを更新するには、このブローカープロパティを使用します。
- Parameters and Policies (RabbitMQ サーバードキュメント) – RabbitMQ のウェブサイトで、RabbitMQ のパラメータとポリシーの詳細について学びます。
- RabbitMQ Management HTTP API – RabbitMQ Management API の詳細について学びます。

ステップ 2: ブローカーに JVM ベースのアプリケーションを接続する

RabbitMQ ブローカーを作成したら、ブローカーにアプリケーションを接続できます。以下の例では、RabbitMQ Java クライアントライブラリを使用してブローカーへの接続を作成し、キューを作成して、メッセージを送信する方法を説明します。RabbitMQ ブローカーには、サポートされているさまざまな言語の RabbitMQ クライアントライブラリを使用して接続することができます。サポートされている RabbitMQ クライアントライブラリの詳細については、「RabbitMQ client libraries and developer tools」を参照してください。

前提条件

> **Note**
>
> 以下の前提条件ステップは、パブリックアクセシビリティなしで作成された RabbitMQ ブローカーのみに適用されます。パブリックアクセシビリティがあるブローカーを作成している場合は、スキップすることができます。

VPC 属性 を有効にする

VPC 内でブローカーにアクセスできることを確実にするには、`enableDnsHostnames` および `enableDnsSupport` VPC 属性を有効にする必要があります。詳細については、Amazon VPC ユーザーガイドの「VPC の DNS サポート」を参照してください。

インバウンド接続を有効にする

1. Amazon MQ コンソールにサインインします。
2. ブローカーのリストからブローカーの名前 (MyBroker など) を選択します。
3. [***MyBroker***] ページの [Connections] (接続) セクションで、ブローカーのウェブコンソール URL とワイヤレベルプロトコルのアドレスとポートをメモします。
4. [Details] (詳細) セクションの [Security and network] (セキュリティとネットワーク) で、セキュリティグループの名前または
   
   をクリックします。

   EC2 ダッシュボードの [セキュリティグループ] ページが表示されます。
5. セキュリティグループのリストから、セキュリティグループを選択します。
6. ページ下部で、[インバウンド] を選択し、次に [編集] を選択します。
7. [Edit inbound rules] (インバウンドルールの編集) ダイアログボックスで、パブリックアクセスを許可する URL またはエンドポイントごとにルールを追加します (以下の例は、これをブローカーのウェブコンソールに対して行う方法を説明しています)。

   a. [ルールの追加] を選択します。

   b. [タイプ] で、[カスタム TCP] を選択します。

c. [Source] (ソース) では、[Custom] (カスタム) が選択された状態のままにしておき、ウェブコンソールにアクセスできるようにするシステムの IP アドレスを入力します (`192.0.2.1` など)。

d. [Save] (保存) をクリックします。

これで、ブローカーはインバウンド接続を受け入れることができます。

Java の依存関係を追加する

ビルドの自動化のために Apache Maven を使用している場合は、以下の依存関係を `pom.xml` ファイルに追加します。Apache Maven のプロジェクトオブジェクトモデルファイルの詳細については、「Introduction to the POM」を参照してください。

```
<dependency>
    <groupId>com.rabbitmq</groupId>
    <artifactId>amqp-client</artifactId>
    <version>5.9.0</version>
</dependency>
```

ビルドの自動化のために Gradle を使用している場合は、以下の依存関係を宣言します。

```
dependencies {
    compile 'com.rabbitmq:amqp-client:5.9.0'
}
```

`Connection` と **`Channel`** クラスをインポートする

RabbitMQ Java クライアントは、そのトップレベルパッケージとして `com.rabbitmq.client` を使用し、それぞれが AMQP 0-9-1 接続とチャネルを表す `Connection` および `Channel` API クラスがあります。以下の例にあるように、使用する前に `Connection` と `Channel` クラスをインポートします。

```
import com.rabbitmq.client.Connection;
import com.rabbitmq.client.Channel;
```

ConnectionFactory を作成してブローカーに接続する

以下の例を使用して、所定のパラメータで ConnectionFactory クラスのインスタンスを作成します。setHost メソッドを使用して、先ほどメモしておいたブローカーエンドポイントを設定します。AMQPS のワイヤレベル接続には、ポート 5671 を使用します。

```
ConnectionFactory factory = new ConnectionFactory();

factory.setUsername(username);
factory.setPassword(password);

//Replace the URL with your information
factory.setHost("b-c8352341-ec91-4a78-ad9c-a43f23d325bb.mq.us-west-2.amazonaws.com");
factory.setPort(5671);

// Allows client to establish a connection over TLS
factory.useSslProtocol();

// Create a connection
Connection conn = factory.newConnection();

// Create a channel
Channel channel = conn.createChannel();
```

エクスチェンジにメッセージを発行する

エクスチェンジにメッセージを発行するには、Channel.basicPublish を使用できます。以下の例では、AMQP Builder クラスを使用して、content-type が plain/text のメッセージプロパティオブジェクトを構築します。

```
byte[] messageBodyBytes = "Hello, world!".getBytes();
channel.basicPublish(exchangeName, routingKey,
             new AMQP.BasicProperties.Builder()
               .contentType("text/plain")
               .userId("userId")
               .build(),
               messageBodyBytes);
```

> **Note**
>
> BasicProperties は自動生成されたホルダークラス AMQP の内部クラスであることに注意してください。

キューにサブスクライブしてメッセージを受信する

メッセージは、Consumer インターフェイスを使用してキューにサブスクライブすることによって受信できます。サブスクライブすると、メッセージが到着すると同時に自動配信されます。

Consumer を実装する最も簡単な方法は、サブクラス DefaultConsumer の使用です。以下の例にあるように、DefaultConsumer オブジェクトは、サブスクリプションをセットアップするための basicConsume コールの一部として渡すことがきます。

```
boolean autoAck = false;
channel.basicConsume(queueName, autoAck, "myConsumerTag",
     new DefaultConsumer(channel) {
         @Override
         public void handleDelivery(String consumerTag,
                                    Envelope envelope,
                                    AMQP.BasicProperties properties,
                                    byte[] body)
             throws IOException
         {
             String routingKey = envelope.getRoutingKey();
             String contentType = properties.getContentType();
             long deliveryTag = envelope.getDeliveryTag();
             // (process the message components here ...)
             channel.basicAck(deliveryTag, false);
         }
     });
```

> **Note**
>
> autoAck = false を指定したので、Consumer に配信されたメッセージを承認する必要があります。これは、上記の例にあるように、handleDelivery で実行することが最も便利です。

接続を閉じてブローカーへの接続を切断する

RabbitMQ ブローカーへの接続を切断するには、以下に示すように、チャネルと接続の両方を閉じます。

```
channel.close();
conn.close();
```

> **Note**
>
> RabbitMQ Java クライアントライブラリの使用に関する詳細については、RabbitMQ Java Client API Guide を参照してください

ステップ 3: (オプション) AWS Lambda 関数に接続する

AWS Lambda は、Amazon MQ ブローカーに接続し、Amazon MQ ブローカーからのメッセージを消費できます。ブローカーを Lambda に接続するときは、キューからメッセージを読み取り、関数 synchronously を呼び出すイベントソースマッピングを作成します。作成するイベントソースマッピングは、ブローカーからメッセージをバッチで読み取り、それらを JSON オブジェクト形式の Lambda ペイロードに変換します。

ブローカーを Lambda 関数に接続する

1. Lambda 関数 execution role に以下の IAM ロール許可を追加します。

   - mq:DescribeBroker
   - ec2:CreateNetworkInterface
   - ec2:DeleteNetworkInterface
   - ec2:DescribeNetworkInterfaces
   - ec2:DescribeSecurityGroups
   - ec2:DescribeSubnets
   - ec2:DescribeVpcs
   - logs:CreateLogGroup
   - logs:CreateLogStream
   - logs:PutLogEvents
   - secretsmanager:GetSecretValue

> **Note**
>
> 必要な IAM 許可がない場合、関数は Amazon MQ リソースからレコードを正常に読み取ることができません。

2. (オプション) パブリックアクセシビリティがないブローカーを作成した場合は、次のいずれかを実行して、Lambda のブローカーへの接続を許可する必要があります。

   - パブリックサブネットごとに 1 つの NAT ゲートウェイを設定します。詳細については、AWS Lambda デベロッパーガイドの「VPC に接続した関数のインターネットアクセスとサービスアクセス」を参照してください。
   - VPC エンドポイントを使用して、Amazon Virtual Private Cloud (Amazon VPC) と Lambda 間の接続を作成します。Amazon VPC は、 AWS Security Token Service （ AWS STS) および Secrets Manager エンドポイントにも接続する必要があります。詳細については、AWS Lambda デベロッパーガイドの「Lambda のインターフェイス VPC エンドポイントの設定」を参照してください。

3. AWS Management Consoleを使用して、Lambda 関数のイベントソースとしてブローカーを設定します。`create-event-source-mapping` AWS Command Line Interface コマンドを使用することもできます。

4. ブローカーから取り込まれたメッセージを処理するための Lambda 関数のコードをいくつか記述します。イベントソースマッピングによって取得される Lambda ペイロードは、ブローカーのエンジンタイプに依存します。以下は、Amazon MQ for RabbitMQ キューの Lambda ペイロードの例です。

> **Note**
>
> この例では、`test` がキューの名前で、`/` がデフォルト仮想ホストの名前です。メッセージを受信すると、イベントソースは `test::/` の下にメッセージを一覧表示します。

```
{
  "eventSource": "aws:rmq",
  "eventSourceArn": "arn:aws:mq:us-
west-2:112556298976:broker:test:b-9bcfa592-423a-4942-879d-eb284b418fc8",
  "rmqMessagesByQueue": {
```

```
    "test::/": [
      {
        "basicProperties": {
          "contentType": "text/plain",
          "contentEncoding": null,
          "headers": {
            "header1": {
              "bytes": [
                118,
                97,
                108,
                117,
                101,
                49
              ]
            },
            "header2": {
              "bytes": [
                118,
                97,
                108,
                117,
                101,
                50
              ]
            },
            "numberInHeader": 10
          }
          "deliveryMode": 1,
          "priority": 34,
          "correlationId": null,
          "replyTo": null,
          "expiration": "60000",
          "messageId": null,
          "timestamp": "Jan 1, 1970, 12:33:41 AM",
          "type": null,
          "userId": "AIDACKCEVSQ6C2EXAMPLE",
          "appId": null,
          "clusterId": null,
          "bodySize": 80
        },
        "redelivered": false,
        "data": "eyJ0aW1lb3V0IjowLCJkYXRhIjoiQ1pybWYwR3c4T3Y0YnFMUXhENEUifQ=="
      }
```

```
    ]
  }
}
```

Amazon MQ の Lambda への接続、Amazon MQ イベントソースに対して Lambda がサポートするオプション、およびイベントソースマッピングエラーの詳細については、AWS Lambda デベロッパーガイドの「Amazon MQ で Lambda を使用する」を参照してください。

Amazon MQ for RabbitMQ エンジンバージョンの管理

RabbitMQ は、X.Y.Z 形式のセマンティックバージョニングに従ってバージョン番号を分類します。Amazon MQ for RabbitMQ の実装では、X はメジャーバージョンを示し、Y はマイナーバージョンを表し、Z はパッチバージョン番号を示します。Amazon MQ は、メジャーバージョン番号が変更される場合に、バージョン変更がメジャーであると見なします。例えば、バージョン 3.13 から 4.0 へのアップグレードは、メジャーバージョンアップグレードと見なされます。マイナーバージョン番号またはパッチバージョン番号のみが変わる場合、バージョン変更はマイナーと見なされます。例えば、バージョン 3.11.28 から 3.12.13 へのアップグレードは、マイナーバージョンアップグレードと見なされます。

Amazon MQ for RabbitMQ では、すべてのブローカーについて、サポートされている最新のマイナーバージョンを使用することをお勧めします。ブローカーエンジンバージョンをアップグレードする手順については、「Amazon MQ ブローカーエンジンバージョンのアップグレード」を参照してください。

> **Important**
>
> Amazon MQ では、ストリームはサポートされません。ストリームを作成すると、データが失われます。
> Amazon MQ は RabbitMQ 3.9 で導入された JSON での構造化ロギングの使用はサポートしません。

サポートされる Amazon MQ for RabbitMQ エンジンバージョン

Amazon MQ バージョンサポートカレンダーは、ブローカーエンジンバージョンがサポート終了に達するタイミングを示します。あるバージョンがサポート終了に達すると、Amazon MQ は、そのバージョンのすべてのブローカーを、サポートされている次のバージョンに自動的にアップグレードしま

す。このアップグレードは、ブローカーのスケジュールされたメンテナンスウィンドウ内で、サポート終了日から 45 日以内に行われます。

Amazon MQ は、バージョンがサポート終了に達する少なくとも 90 日前に通知を送信します。中断を防ぐために、サポート終了日より前にブローカーをアップグレードすることをお勧めします。また、サポート終了が 30 日以内に予定されているバージョンで新しいブローカーを作成することはできません。

RabbitMQ バージョン	Amazon MQ でのサポート終了
3.13 (推奨)	
3.12	2025 年 3 月 17 日
3.11	2025 年 2 月 17 日
3.10	2024 年 10 月 15 日
3.9	2024 年 9 月 16 日

新しい Amazon MQ for RabbitMQ ブローカーを作成するときは、サポートされている任意の RabbitMQ エンジンバージョンを指定できます。ブローカーの作成時にエンジンバージョン番号を指定しない場合は、Amazon MQ により、デフォルトで自動的に最新のエンジンバージョン番号が選択されます。

エンジンバージョンのアップグレード

ブローカーはいつでも、サポートされている次のメジャーバージョンまたはマイナーバージョンに手動でアップグレードできます。自動マイナーバージョンアップグレードを有効にすると、Amazon MQ はメンテナンスウィンドウ内で、サポートされている最新のパッチバージョンにブローカーをアップグレードします。

ブローカーの手動アップグレードの詳細については、「the section called “エンジンバージョンのアップグレード”」を参照してください。

エンジンバージョン 3.13 以降を使用しているすべてのブローカーについて、Amazon MQ は、サポートされている最新のパッチバージョンへのアップグレードをメンテナンスウィンドウ内で管理します。

> **Important**
>
> RabbitMQ では、バージョンの増分アップデート (例: 3.9.x から 3.10.x) のみが可能です。アップデートでマイナーバージョンをスキップすること (例: 3.8.x から 3.11.x) はできません。

再起動中、シングルインスタンスブローカーはオフラインになります。クラスターブローカーでは、ミラーキューは再起動時に同期する必要があります。キューが長い場合、キューの同期プロセスに時間がかかることがあります。キューの同期プロセス中、コンシューマーとプロデューサーはキューを利用できません。キューの同期プロセスが完了すると、ブローカーは再び利用可能になります。影響を最小限に抑えるために、トラフィックの少ない時間帯にアップグレードすることをお勧めします。バージョンアップグレードのベストプラクティスの詳細については、「Amazon MQ for RabbitMQ のベストプラクティス」を参照してください。

サポートされているエンジンバージョンのリスト化

describe-broker-instance-options AWS CLI コマンドを使用して、サポートされているすべてのマイナーエンジンバージョンとメジャーエンジンバージョンを一覧表示できます。

```
aws mq describe-broker-instance-options
```

エンジンおよびインスタンスタイプで結果をフィルタリングするには、以下にあるように、--engine-type および --host-instance-type オプションを使用します。

```
aws mq describe-broker-instance-options --engine-type engine-type --host-instance-type instance-type
```

例えば、ActiveMQ と mq.m5.large インスタンスタイプで結果をフィルタリングするには、*engine-type* を RABBITMQ、*instance-type* を mq.m5.large に置き換えます。

Amazon MQ for RabbitMQ のベストプラクティス

このセクションは、Amazon MQ での RabbitMQ ブローカーの使用時にパフォーマンスを最大限に引き出し、スループットコストを最小限に抑えるための推奨事項をすばやく見つけるために使用してください。

> **Important**
>
> 現在、Amazon MQ はストリームや、RabbitMQ 3.9.x で導入された JSON での構造化ロギングの使用をサポートしていません。

> **Important**
>
> Amazon MQ for RabbitMQ では、ユーザー名「guest」はサポートされず、デフォルトのゲストアカウントは新しいブローカーの作成時に削除されます。ユーザーが作成した「guest」というアカウントも、Amazon MQ によって定期的に削除されます。

トピック

最高のスループットのために正しいブローカーインスタンスタイプを選択する

ブローカーインスタンスタイプのメッセージスループットは、アプリケーションのユースケースに依存します。`t3.micro` などの小さいブローカーインスタンスタイプは、アプリケーションのパフォーマンスをテストする場合にのみ使用することをお勧めします。大規模なインスタンスを本番環境で使用する前にこれらのマイクロインスタンスを使用すると、アプリケーションのパフォーマンスが向上し、開発コストを抑えることができます。`m5.large` 以上のインスタンスタイプでは、クラスターデプロイを使用して高可用性とメッセージの耐久性を確保できます。大きいブローカーイン

スタンスタイプは、本番稼働レベルのクライアントとキュー、高スループット、メモリ内のメッセージ、冗長メッセージを処理できます。正しいインスタンスタイプの選択の詳細については、「」を参照してくださいthe section called “サイズ設定ガイドライン”。

複数のチャネルを使用する

接続チャーンを回避するには、1 つの接続で複数のチャネルを使用します。アプリケーションでは、チャネルに対する 1:1 の接続を避ける必要があります。プロセスごとに 1 つの接続を使用し、スレッドごとに 1 つのチャネルを使用することをお勧めします。チャネルのリークを防ぐために、チャネルを過剰に使用することは避けてください。

永続メッセージと持続キューを使用する

永続メッセージは、ブローカーがクラッシュまたは再起動するという状況におけるデータ損失の防止に役立ちます。永続メッセージは、到着するとすぐにディスクに書き込まれますが、レイジーキューとは異なり、ブローカーがより多くのメモリを必要とする場合を除き、永続メッセージはメモリとディスクの両方にキャッシュされます。より多くのメモリが必要な場合は、ディスクへのメッセージの保存を管理する RabbitMQ ブローカーメカニズム (一般に永続レイヤーと呼ばれます) によって、メモリからメッセージが削除されます。

メッセージの永続性を有効にするには、キューを `durable` として宣言し、メッセージ配信モードを `persistent` に設定できます。以下の例は、RabbitMQ Java クライアントライブラリを使用した持続キューの宣言を示しています。AMQP 0-9-1 を使用している場合は、配信モード「2」を設定することで、メッセージを永続としてマークできます。

```
boolean durable = true;
channel.queueDeclare("my_queue", durable, false, false, null);
```

キューを持続キューとして設定したら、以下の例にあるように、`MessageProperties` を `PERSISTENT_TEXT_PLAIN` に設定することによって永続メッセージをキューに送信できます。

```
import com.rabbitmq.client.MessageProperties;

channel.basicPublish("", "my_queue",
            MessageProperties.PERSISTENT_TEXT_PLAIN,
            message.getBytes());
```

キューを短くしておく

クラスターデプロイでは、多数のメッセージを持つキューがリソースの過剰な使用につながる場合があります。ブローカーが過剰に使用されているときは、Amazon MQ for RabbitMQ ブローカーの再起動がパフォーマンスをさらに低下させる原因となる可能性があります。過剰に使用されているブローカーが再起動されると、REBOOT_IN_PROGRESS 状態のまま応答しなくなることがあります。

Amazon MQ はメンテナンスウィンドウ中、すべてのメンテナンス作業を一度に 1 ノードずつ実行して、ブローカーが動作可能な状態を維持することを確実にします。その結果、各ノードが操作を再開するときに、キューが同期する必要が生じる場合があります。同期中、ミラーにレプリケートする必要があるメッセージは、バッチで処理されるように、対応する Amazon Elastic Block Store (Amazon EBS) ボリュームからメモリにロードされます。メッセージをバッチで処理することにより、キューの同期が速くなります。

キューを短くし、メッセージを小さくしておくと、キューが正常に同期し、期待通りに操作を再開します。ただし、バッチ内のデータ量がノードのメモリ制限に近づいた場合は、ノードが高メモリアラームを発し、キューの同期を一時停止します。メモリ使用量は、CloudWatch で RabbitMemUsed および RabbitMqMemLimit のブローカーノードメトリクスを比較することで確認できます。同期は、メッセージが消費もしくは削除される、またはバッチ内のメッセージの数が減るまで完了できません。

クラスターデプロイのためにキューの同期化が一時停止される場合は、メッセージを消費または削除して、キュー内のメッセージの数を減らすことをお勧めします。キュー深度が減少し、キューの同期が完了すると、ブローカーのステータスが RUNNING に変更されます。一時停止されたキューの同期を解決するには、キューの同期のバッチサイズを小さくするポリシーを適用することも可能です。

また、自動削除ポリシーと TTL ポリシーを定義すると、リソースの使用量をプロアクティブに削減するとともに、コンシューマーからの NACK を最小限に抑えることができます。ブローカーへのメッセージの再キュー処理は CPU 負荷が高いため、大量の NACK が発生するとブローカーのパフォーマンスに影響する可能性があります。

パブリッシャーの確認とコンシューマーの配信承認の設定

ブローカーにメッセージが送信されたことを確認するプロセスは、パブリッシャーの確認と呼ばれます。パブリッシャーは、メッセージが確実に格納されたときにアプリケーションに通知します。パブリッシャーの確認は、ブローカーに格納されるメッセージの割合を制御するためにも役立ちます。パブリッシャーが確認しないと、メッセージが正常に処理されたことは確認されず、ブローカーは処理できないメッセージを削除する可能性があります。

同様に、クライアントアプリケーションはメッセージの配信と消費の確認をブローカーに返送します。これはコンシューマーの配信承認と呼ばれます。RabbitMQ ブローカーの使用時にデータの安全性を確保するには、確認と承認の両方が不可欠です。

コンシューマーの配信承認は、通常クライアントアプリケーションで設定されています。AMQP 0-9-1 を使用している場合は、`basic.consume` メソッドを設定することで承認を有効化できます。AMQP 0-9-1 クライアントでは、`confirm.select` メソッドを送信してパブリッシャーの確認を設定することもできます。

通常、配信承認はチャネルで有効化されます。例えば、RabbitMQ Java クライアントライブラリの使用時には、以下の例にあるように、`Channel#basicAck` を使用してシンプルな `basic.ack` 肯定承認をセットアップできます。

```
// this example assumes an existing channel instance

boolean autoAck = false;
channel.basicConsume(queueName, autoAck, "a-consumer-tag",
     new DefaultConsumer(channel) {
         @Override
         public void handleDelivery(String consumerTag,
                                    Envelope envelope,
                                    AMQP.BasicProperties properties,
                                    byte[] body)
             throws IOException
         {
             long deliveryTag = envelope.getDeliveryTag();
             // positively acknowledge a single delivery, the message will
             // be discarded
             channel.basicAck(deliveryTag, false);
         }
     });
```

> **Note**
>
> 未承認メッセージは、メモリにキャッシュする必要があります。コンシューマーがプリフェッチするメッセージの数は、クライアントアプリケーションのプリフェッチを設定することによって制限できます。

`consumer_timeout` を設定すると、コンシューマーから配信承認が届かない状況を検出できます。コンシューマーがタイムアウト値の時間内に承認を送信しない場合、チャネルは閉じら

れ、PRECONDITION_FAILED が発生します。エラーを診断するには、UpdateConfiguration API を使用して consumer_timeout 値を大きくします。

プリフェッチを設定する

RabbitMQ のプリフェッチ値を使用して、コンシューマーがメッセージを消費する方法を最適化できます。RabbitMQ は、プリフェッチ数をチャネルではなくコンシューマーに適用することによって、AMQP 0-9-1 が提供するチャネルプリフェッチメカニズムを実装します。プリフェッチ値は、特定の時間にコンシューマに送信されるメッセージの数を指定するために使用されます。デフォルトで、RabbitMQ はクライアントアプリケーションに無制限のバッファサイズを設定します。

RabbitMQ コンシューマーにプリフェッチ数を設定するときに考慮する要因にはさまざまなものがあります。まず、コンシューマーの環境と設定を考慮します。コンシューマーは、メッセージが処理されるときにそれらすべてをメモリに保持する必要があるため、高いプリフェッチ値はコンシューマーのパフォーマンスに悪影響を及ぼし、場合によってはコンシューマー全体がクラッシュする原因になることもあります。同様に、RabbitMQ ブローカー自体も、コンシューマー承認を受け取るまで、送信するすべてのメッセージをメモリにキャッシュしておきます。コンシューマに自動承認が設定されておらず、コンシューマによるメッセージの処理に比較的長い時間がかかる場合、高いプリフェッチ値は RabbitMQ サーバーのメモリがすぐになくなる原因になる可能性があります。

上記の考慮事項を踏まえて、大量の未処理または未承認のメッセージが原因で RabbitMQ ブローカー、またはそのコンシューマーでメモリ不足が発生する状況を防ぐため、常にプリフェッチ値を設定することが推奨されます。大量のメッセージを処理するためにブローカーを最適化する必要がある場合は、さまざまなプリフェッチ数を使用してブローカーとコンシューマーをテストし、コンシューマーがメッセージを処理するためにかかる時間と比較して、ネットワークオーバーヘッドがおおむね軽微なものになる値を判断します。

> **Note**
>
> - コンシューマーへのメッセージの配信を自動承認するようにクライアントアプリケーションが設定されている場合、プリフェッチ値を設定しても効果はありません。
> - プリフェッチされたメッセージはすべて、キューから削除されます。

以下の例は、RabbitMQ Java クライアントライブラリを使用した単一のコンシューマーへのプリフェッチ値 10 の設定を示しています。

```
ConnectionFactory factory = new ConnectionFactory();
```

```
Connection connection = factory.newConnection();
Channel channel = connection.createChannel();

channel.basicQos(10, false);

QueueingConsumer consumer = new QueueingConsumer(channel);
channel.basicConsume("my_queue", false, consumer);
```

> **Note**
>
> RabbitMQ Java クライアントライブラリでは、global フラグのデフォルト値が false に設定されているので、上記の例は単純に channel.basicQos(10) として記述できます。

クォーラムキューで Celery 5.5 以降を使用する

分散タスクキューシステムである Python Celery は、タスク負荷が高い場合に、重要ではないメッセージを多数生成できます。この追加のブローカーアクティビティにより、RabbitMQ メモリアラームがトリガーされ、ブローカーが使用できなくなる可能性があります。メモリアラームがトリガーされる可能性を減らすには、以下を実行します。

すべての Celery バージョンの場合

1. をオフにtask_create_missing_queuesしてキューのチャーンを軽減します。
2. 次に、 をオフにworker_enable_remote_controlして、celery@...pidboxキューの動的作成を停止します。これにより、ブローカーのキューチャーンが減少します。

   ```
   worker_enable_remote_control = false
   ```
3. 重要でないメッセージアクティビティをさらに減らすには、Celery アプリケーションを起動するときに -Eまたは --task-events フラグを付けずに、Celery worker-send-task-events をオフにします。
4. 次のパラメータを使用して Celery アプリケーションを起動します。

   ```
   celery -A app_name worker --without-heartbeat --without-gossip --without-mingle
   ```

Celery バージョン 5.5 以降の場合

1. Celery バージョン 5.5、クォーラムキューをサポートする最小バージョン、またはそれ以降のバージョンにアップグレードします。使用している Celery のバージョンを確認するには、 を使用しますcelery --version。クォーラムキューの詳細については、「」を参照してくださいthe section called “クォーラムキュー”。
2. Celery 5.5 以降にアップグレードした後、 task_default_queue_type を「クォーラム」に設定します。
3. 次に、ブローカートランスポートオプションで発行確認を有効にする必要もあります。

```
broker_transport_options = {"confirm_publish": True}
```

ネットワーク障害から自動的に回復する

RabbitMQ ノードへのクライアント接続が失敗した場合の大幅なダウンタイムを防ぐため、自動ネットワークリカバリを常に有効にしておくことをお勧めします。バージョン 4.0.0 以降の RabbitMQ Java クライアントライブラリは、自動ネットワークリカバリをデフォルトでサポートします。

自動接続リカバリは、接続の I/O ループで未処理の例外がスローされた場合、ソケット読み取り操作のタイムアウトが検出された場合、またはサーバーがハートビートを受信しない場合にトリガーされます。

クライアントと RabbitMQ ノード間の初期接続が失敗した場合、自動リカバリはトリガーされません。アプリケーションコードは、接続の再試行によって、初期接続障害を考慮するように記述することをお勧めします。以下の例は、RabbitMQ Java クライアントライブラリを使用した初期ネットワーク障害の再試行を示しています。

```
ConnectionFactory factory = new ConnectionFactory();
// enable automatic recovery if using RabbitMQ Java client library prior to version
 4.0.0.
factory.setAutomaticRecoveryEnabled(true);
// configure various connection settings

try {
  Connection conn = factory.newConnection();
} catch (java.net.ConnectException e) {
  Thread.sleep(5000);
  // apply retry logic
}
```

> **Note**
>
> アプリケーションが `Connection.Close` メソッド使用して接続を閉じる場合、自動ネットワークリカバリは有効化またはトリガーされません。

メッセージサイズを 1 MB 未満に維持する

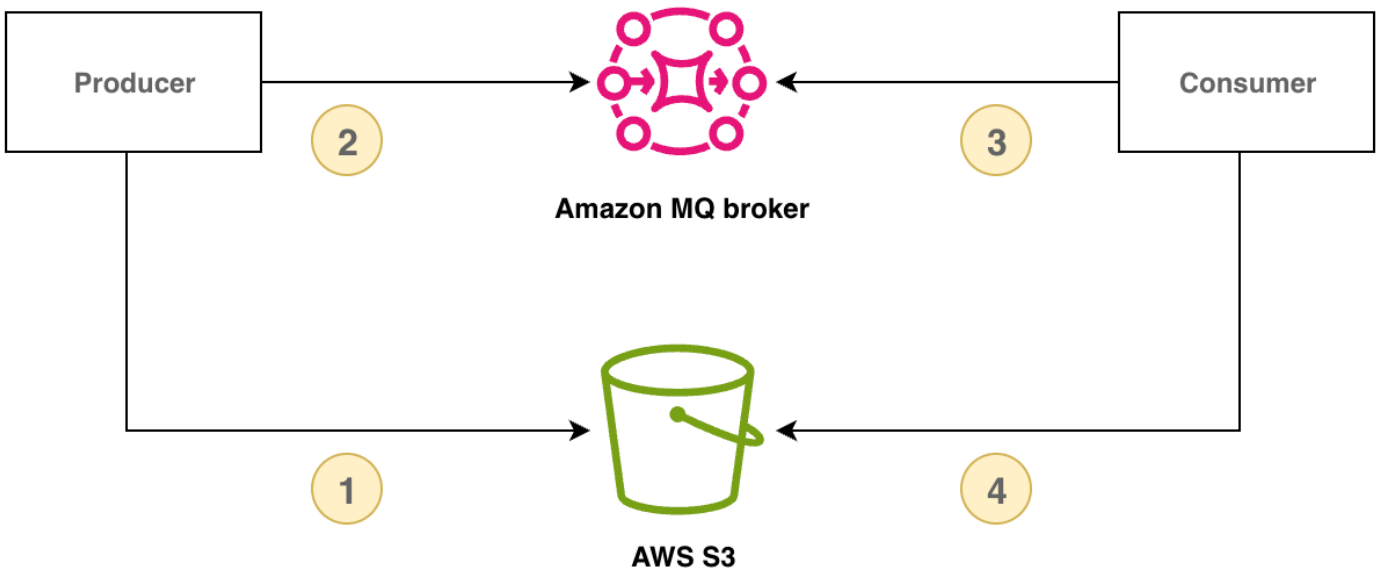
最適なパフォーマンスと信頼性を得るには、メッセージを 1 MB (1 MB) 未満にしておくことをお勧めします。

RabbitMQ 3.13 はデフォルトで最大 128 MB のメッセージサイズをサポートしますが、大きなメッセージは、発行をブロックし、ノード間でメッセージをレプリケートしながら高いメモリ負荷を発生させる可能性のある予測不可能なメモリアラームをトリガーする可能性があります。メッセージのサイズが大きすぎると、ブローカーの再起動プロセスや復旧プロセスにも影響し、サービス継続性のリスクが高まり、パフォーマンスが低下する可能性があります。

クレームチェックパターンを使用して大きなペイロードを保存および取得する

大きなメッセージを管理するには、メッセージペイロードを外部ストレージに保存し、RabbitMQ を介してペイロード参照識別子のみを送信することで、クレームチェックパターンを実装できます。コンシューマーはペイロード参照識別子を使用して、大きなメッセージを取得して処理します。

次の図は、Amazon MQ for RabbitMQ と Amazon S3 を使用してクレームチェックパターンを実装する方法を示しています。



次の例は、Amazon MQ、 AWS SDK for Java 2.x、Amazon S3 を使用したこのパターンを示しています。

1. まず、Amazon S3 参照識別子を保持する Message クラスを定義します。

```
class Message {
    // Other data fields of the message...

    public String s3Key;
    public String s3Bucket;
}
```

2. Amazon S3 にペイロードを保存し、RabbitMQ を介してリファレンスメッセージを送信するパブリッシャーメソッドを作成します。

```
public void publishPayload() {
    // Store the payload in S3.
    String payload = PAYLOAD;
    String prefix = S3_KEY_PREFIX;
    String s3Key = prefix + "/" + UUID.randomUUID();
    s3Client.putObject(PutObjectRequest.builder()
        .bucket(S3_BUCKET).key(s3Key).build(),
        RequestBody.fromString(payload));

    // Send the reference through RabbitMQ.
    Message message = new Message();
    message.s3Key = s3Key;
    message.s3Bucket = S3_BUCKET;
    // Assign values to other fields in your message instance.

    publishMessage(message);
}
```

3. Amazon S3 からペイロードを取得し、ペイロードを処理し、Amazon S3 オブジェクトを削除するコンシューマーメソッドを実装します。

```
public void consumeMessage(Message message) {
    // Retrieve the payload from S3.
    String payload = s3Client.getObjectAsBytes(GetObjectRequest.builder()
        .bucket(message.s3Bucket).key(message.s3Key).build())
        .asUtf8String();
```

```
    // Process the complete message.
    processPayload(message, payload);

    // Delete the S3 object.
    s3Client.deleteObject(DeleteObjectRequest.builder()
        .bucket(message.s3Bucket).key(message.s3Key).build());
}
```

basic.consume と存続期間の長いコンシューマーを使用する

存続期間の長いコンシューマーbasic.consumeで を使用すると、 を使用して個々のメッセージをポーリングするよりも効率的ですbasic.get。詳細については、「個々のメッセージのポーリング」を参照してください。

Amazon MQ のセキュリティ

のクラウドセキュリティが最優先事項 AWS です。お客様は AWS 、セキュリティの影響を受けやすい組織の要件を満たすように構築されたデータセンターとネットワークアーキテクチャを活用できます。

セキュリティは、 AWS とユーザーの間で共有される責任です。責任共有モデルでは、これをクラウドのセキュリティおよびクラウド内のセキュリティと説明しています。

- クラウドのセキュリティ – AWS クラウドで AWS サービスを実行するインフラストラクチャを保護する AWS 責任があります。 AWS また、 では、安全に使用できるサービスも提供しています。サードパーティーの監査者は、AWS コンプライアンスプログラムコンプライアンスプログラムの一環として、当社のセキュリティの有効性を定期的にテストおよび検証。Amazon MQ に適用されるコンプライアンスプログラムの詳細については、「コンプライアンスプログラムAWS による対象範囲内のサービスコンプライアンスプログラム」を参照してください。
- クラウド内のセキュリティ — お客様の責任は、使用する AWS サービスによって決まります。また、ユーザーは、データの機密性、会社の要件、適用される法律や規制など、その他の要因についても責任を負います。

このドキュメントは、Amazon MQ の使用時に責任共有モデルがどのように適用されるかを理解するために役立ちます。以下のトピックでは、セキュリティおよびコンプライアンス上の目的を達成するように Amazon MQ を設定する方法について説明します。また、Amazon MQ リソースのモニタリングや保護に役立つ他の AWS サービスの使用方法についても説明します。

トピック

- Amazon MQ のデータ保護
- Amazon MQ のための Identity and Access Management
- Amazon MQ のコンプライアンス検証
- Amazon MQ の耐障害性
- Amazon MQ のインフラストラクチャセキュリティ
- Amazon MQ のセキュリティベストプラクティス

Amazon MQ のデータ保護

責任 AWS 共有モデル、Amazon MQ でのデータ保護に適用されます。このモデルで説明されているように、 AWS はすべての を実行するグローバルインフラストラクチャを保護する責任があります AWS クラウド。ユーザーは、このインフラストラクチャでホストされるコンテンツに対する管理を維持する責任があります。また、使用する「 AWS のサービス 」のセキュリティ設定と管理タスクもユーザーの責任となります。データプライバシーの詳細については、データプライバシーに関するよくある質問を参照してください。欧州でのデータ保護の詳細については、AWS セキュリティブログに投稿された AWS 責任共有モデルおよび GDPR のブログ記事を参照してください。

データ保護の目的で、認証情報を保護し AWS アカウント 、 AWS IAM Identity Center または AWS Identity and Access Management (IAM) を使用して個々のユーザーを設定することをお勧めします。この方法により、それぞれのジョブを遂行するために必要な権限のみが各ユーザーに付与されます。また、次の方法でデータを保護することもお勧めします:

- 各アカウントで多要素認証 (MFA) を使用します。
- SSL/TLS を使用して AWS リソースと通信します。TLS 1.2 が必須で、TLS 1.3 をお勧めします。
- で API とユーザーアクティビティのログ記録を設定します AWS CloudTrail。CloudTrail 証跡を使用して AWS アクティビティをキャプチャする方法については、「 AWS CloudTrail ユーザーガイド」のCloudTrail 証跡の使用」を参照してください。
- AWS 暗号化ソリューションと、 内のすべてのデフォルトのセキュリティコントロールを使用します AWS のサービス。
- Amazon Macie などの高度な管理されたセキュリティサービスを使用します。これらは、Amazon S3 に保存されている機密データの検出と保護を支援します。
- コマンドラインインターフェイスまたは API AWS を介して にアクセスするときに FIPS 140-3 検証済み暗号化モジュールが必要な場合は、FIPS エンドポイントを使用します。利用可能な FIPS エンドポイントの詳細については、「連邦情報処理規格 (FIPS) 140-3」を参照してください。

お客様の E メールアドレスなどの極秘または機密情報を、タグ、または [名前] フィールドなどの自由形式のテキストフィールドに含めないことを強くお勧めします。これは、コンソール、API、または SDK を使用して Amazon MQ AWS CLIまたは他の AWS のサービス を使用する場合も同様です。AWS SDKs タグ、または名前に使用される自由記述のテキストフィールドに入力したデータは、請求または診断ログに使用される場合があります。外部サーバーに URL を提供する場合、そのサーバーへのリクエストを検証できるように、認証情報を URL に含めないことを強くお勧めします。

Amazon MQ for ActiveMQ と Amazon MQ for RabbitMQ ブローカーのどちらでも、ブローカーのウェブコンソールまたは Amazon MQ API を使用してリソースを作成するときに、ブローカー名またはユーザー名に個人を特定できる情報 (PII) またはその他の秘密情報や機密情報を使用しないでください。ブローカー名とユーザー名は、CloudWatch Logs を含む他の AWS のサービスからアクセスできます。ブローカーのユーザー名は、プライベートデータや機密データとして使用することを意図していません。

> **Important**
>
> TLS 1.3 は RabbitMQ ブローカーでは使用できません。

Encryption

Amazon MQ に保存されているユーザーデータは、保管中暗号化されています。Amazon MQ による保管時の暗号化は、 AWS Key Management Service (KMS) に保存されている暗号化キーを使用してデータを暗号化することによって、セキュリティを強化します。このサービスは、機密データの保護における負担と複雑な作業を減らすのに役立ちます。保管時に暗号化することで、セキュリティを重視したアプリケーションを構築して、暗号化のコンプライアンスと規制の要件を満たすことができます。

Amazon MQ ブローカー間のすべての接続は、転送時の暗号化を提供するために Transport layer Security (TLS) を使用します。

Amazon MQ は、Amazon MQ がセキュアな方法で管理して保存する暗号化キーを使用して、保管中および転送中のメッセージを暗号化します。詳細については、AWS Encryption SDK デベロッパーガイドを参照してください。

保管中の暗号化

Amazon MQ は AWS Key Management Service (KMS) と統合して、透過的なサーバー側の暗号化を提供します。Amazon MQ は、保管中のデータを常に暗号化します。

Amazon MQ for ActiveMQ ブローカーまたは Amazon MQ for RabbitMQ ブローカーを作成するときに、Amazon MQ AWS KMS key が保管中のデータの暗号化に使用する を指定できます。KMS キーを指定しない場合、Amazon MQ は AWS 所有の KMS キーを作成し、ユーザーに代わって使用します。Amazon MQ は現在、対称 KMS キーをサポートしています。KMS キーに関する詳細については、「AWS KMS keys」を参照してください。

ブローカーを作成するときに以下のいずれかを選択することによって、Amazon MQ が暗号化キーに何を使用するかを選択できます。

- Amazon MQ 所有の KMS キー (デフォルト) – キーは Amazon MQ が所有、管理し、ユーザーのアカウントにはありません。
- AWS マネージド KMS キー — AWS マネージド KMS キー (`aws/mq`) は、Amazon MQ によってユーザーに代わって作成、管理、使用されるアカウントの KMS キーです。
- 既存のカスタマーマネージド KMS キーを選択する – カスタマーマネージド KMS キーは、ユーザーが AWS Key Management Service (KMS) で作成し、管理します。

> **Important**
>
> - 付与の取り消しを元に戻すことはできません。アクセス権を取り消す必要がある場合は、ブローカーを削除することをお勧めします。
> - Amazon Elastic File System (EFS) を使用してメッセージデータを保存する Amazon MQ for ActiveMQ ブローカーの場合、アカウントで KMS キーを使用する許可を Amazon EFS に提供する付与を取り消すと、すぐには有効にはなりません。
> - EBS を使用してメッセージデータを保存する Amazon MQ for RabbitMQ ブローカーおよび Amazon MQ for ActiveMQ ブローカー場合、アカウントで KMS キーを使用する許可を Amazon EBS に提供する付与を無効にした、削除をスケジュールした、または取り消した場合、Amazon MQ はブローカーを維持できず、パフォーマンスが低下する可能性があります。
> - キーを無効にした場合、またはキーの削除をスケジュールした場合は、キーを再び有効にするか、キーの削除をキャンセルして、ブローカーの機能を維持できます。
> - キーを無効にするか、付与を取り消しても、すぐには有効にはなりません。

RabbitMQ の KMS キーを使用して単一インスタンスブローカーを作成すると、2 つの `CreateGrant` イベントが AWS CloudTrailに記録されます。最初のイベントは、Amazon MQ による KMS キー用の許可の作成です。2 つ目のイベントは、EBS による EBS 用の許可の作成です。

CreateGrant AWS CloudTrail ログエントリ: 単一インスタンスブローカー

mq_grant

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AKIAIOSFODNN7EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/AmazonMqConsole",
        "accountId": "111122223333",
        "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AKIAIOSFODNN7EXAMPLE",
                "arn": "arn:aws:iam::111122223333:user/AmazonMqConsole",
                "accountId": "111122223333",
                "userName": "AmazonMqConsole"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2023-02-23T18:59:10Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "mq.amazonaws.com"
    },
    "eventTime": "2018-06-28T22:23:46Z",
    "eventSource": "amazonmq.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "203.0.113.0",
    "userAgent": "PostmanRuntime/7.1.5",
    "requestParameters": {
        "granteePrincipal": "mq.amazonaws.com",
        "keyId": "arn:aws:kms:us-east-1:316438333700:key/bdbe42ae-f825-4e78-
a8a1-828d411c4be2",
        "retiringPrincipal": "mq.amazonaws.com",
        "operations": [
            "CreateGrant",
            "Decrypt",
            "GenerateDataKeyWithoutPlaintext",
            "ReEncryptFrom",
            "ReEncryptTo",
            "DescribeKey"
        ]
    },
```

```
    "responseElements": {
        "grantId":
 "0ab0ac0d0b000f00ea00cc0a0e00fc00bce000c000f0000000c0bc0a0000aaafSAMPLE",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",

    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": false,
    "resources": [
        {
           "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "sessionCredentialFromConsole": "true"
}
```

EBS grant creation

EBS の許可の作成に関するイベントが 1 つ表示されます。

```
                                {
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "mq.amazonaws.com"
    },
    "eventTime": "2023-02-23T19:09:40Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "mq.amazonaws.com",
    "userAgent": "ExampleDesktop/1.0 (V1; OS)",
    "requestParameters": {
        "granteePrincipal": "mq.amazonaws.com",
```

```
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "constraints": {
            "encryptionContextSubset": {
                "aws:ebs:id": "vol-0b670f00f7d5417c0"
            }
        },
        "operations": [
            "Decrypt"
        ],
        "retiringPrincipal": "ec2.us-east-1.amazonaws.com"
    },
    "responseElements": {
        "grantId":
 "0ab0ac0d0b000f00ea00cc0a0e00fc00bce000c000f0000000c0bc0a0000aaafSAMPLE",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
    },
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "sharedEventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventCategory": "Management"
}
```


RabbitMQ の KMS キーを使用してクラスターのデプロイを作成すると、5 つの `CreateGrant` イベントが AWS CloudTrailに記録されます。最初の 2 つのイベントは、Amazon MQ 用の許可の作成です。次の 3 つのイベントは、EBS による EBS 用の許可の作成です。

CreateGrant AWS CloudTrail ログエントリ: クラスターデプロイ

mq_grant

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AKIAIOSFODNN7EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/AmazonMqConsole",
        "accountId": "111122223333",
        "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AKIAIOSFODNN7EXAMPLE",
                "arn": "arn:aws:iam::111122223333:user/AmazonMqConsole",
                "accountId": "111122223333",
                "userName": "AmazonMqConsole"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2023-02-23T18:59:10Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "mq.amazonaws.com"
    },
    "eventTime": "2018-06-28T22:23:46Z",
    "eventSource": "amazonmq.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "203.0.113.0",
    "userAgent": "PostmanRuntime/7.1.5",
    "requestParameters": {
        "granteePrincipal": "mq.amazonaws.com",
        "keyId": "arn:aws:kms:us-east-1:316438333700:key/bdbe42ae-f825-4e78-
a8a1-828d411c4be2",
        "retiringPrincipal": "mq.amazonaws.com",
        "operations": [
            "CreateGrant",
            "Encrypt",
            "Decrypt",
```

```
            "ReEncryptFrom",
            "ReEncryptTo",
            "GenerateDataKey",
            "GenerateDataKeyWithoutPlaintext",
            "DescribeKey"
        ]
    },
    "responseElements": {
        "grantId":
 "0ab0ac0d0b000f00ea00cc0a0e00fc00bce000c000f0000000c0bc0a0000aaafSAMPLE",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",

    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": false,
    "resources": [
        {
           "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "sessionCredentialFromConsole": "true"
}
```

mq_rabbit_grant

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AKIAIOSFODNN7EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/AmazonMqConsole",
        "accountId": "111122223333",
        "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
```

```
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AKIAIOSFODNN7EXAMPLE",
                "arn": "arn:aws:iam::111122223333:user/AmazonMqConsole",
                "accountId": "111122223333",
                "userName": "AmazonMqConsole"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2023-02-23T18:59:10Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "mq.amazonaws.com"
    },
    "eventTime": "2018-06-28T22:23:46Z",
    "eventSource": "amazonmq.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "203.0.113.0",
    "userAgent": "PostmanRuntime/7.1.5",
    "requestParameters": {
        "granteePrincipal": "mq.amazonaws.com",
        "retiringPrincipal": "mq.amazonaws.com",
        "operations": [
            "DescribeKey"
        ],
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
    },
    "responseElements": {
        "grantId":
 "0ab0ac0d0b000f00ea00cc0a0e00fc00bce000c000f0000000c0bc0a0000aaafSAMPLE",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",

    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": false,
    "resources": [
        {
           "accountId": "111122223333",
            "type": "AWS::KMS::Key",
```

```
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "sessionCredentialFromConsole": "true"
}
```

EBS grant creation

EBS の許可の作成に関する 3 つのイベントが表示されます。

```
                                    {
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "mq.amazonaws.com"
    },
    "eventTime": "2023-02-23T19:09:40Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "mq.amazonaws.com",
    "userAgent": "ExampleDesktop/1.0 (V1; OS)",
    "requestParameters": {
        "granteePrincipal": "mq.amazonaws.com",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "constraints": {
            "encryptionContextSubset": {
                "aws:ebs:id": "vol-0b670f00f7d5417c0"
            }
        },
        "operations": [
            "Decrypt"
        ],
        "retiringPrincipal": "ec2.us-east-1.amazonaws.com"
    },
    "responseElements": {
```

```
        "grantId":
 "0ab0ac0d0b000f00ea00cc0a0e00fc00bce000c000f0000000c0bc0a0000aaafSAMPLE",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
    },
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "sharedEventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventCategory": "Management"
}
```

KMS キーの詳細については、「AWS Key Management Service デベロッパーガイド」の「AWS KMS keys」を参照してください。

転送中の暗号化

Amazon MQ for ActiveMQ: Amazon MQ for ActiveMQ は強力な Transport Layer Security (TLS) を必要とし、Amazon MQ デプロイのブローカー間で転送されるデータを暗号化します。Amazon MQ ブローカー間で渡されるすべてのデータは、強力な Transport Layer Security (TLS) を使用して暗号化されています。これはすべての利用可能なプロトコルに当てはまります。

Amazon MQ for RabbitMQ: Amazon MQ for RabbitMQ は、すべてのクライアント接続に強力な Transport Layer Security (TLS) 暗号化を必要とします。RabbitMQ クラスターレプリケーショントラフィックはブローカーの VPC のみを転送し、 AWS データセンター間のすべてのネットワークトラフィックは物理レイヤーで透過的に暗号化されます。Amazon MQ for RabbitMQ クラスター化ブローカーは、現在、クラスターレプリケーションのノード間暗号化をサポートしていません。転送中のデータの詳細については、「保管中および転送中のデータの暗号化」を参照してください。

Amazon MQ for ActiveMQ のプロトコル

ActiveMQ ブローカーには、TLS が有効化されている以下のプロトコルを使用してアクセスできます。

- AMQP
- MQTT
- MQTT over WebSocket
- OpenWire
- STOMP
- STOMP over WebSocket

ActiveMQ 向けにサポートされている TLS 暗号スイート

ActiveMQ on Amazon MQ は、以下の暗号スイートをサポートしています。

- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
- TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA256
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA
- TLS_RSA_WITH_AES_256_GCM_SHA384
- TLS_RSA_WITH_AES_256_CBC_SHA256
- TLS_RSA_WITH_AES_256_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
- TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA256
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_AES_128_GCM_SHA256

- TLS_RSA_WITH_AES_128_CBC_SHA256
- TLS_RSA_WITH_AES_128_CBC_SHA

Amazon MQ for RabbitMQ のプロトコル

RabbitMQ ブローカーには、TLS が有効化されている以下のプロトコルを使用してアクセスできます。

- AMQP (0-9-1)

RabbitMQ 向けにサポートされている TLS 暗号スイート

RabbitMQ on Amazon MQ は、以下の暗号スイートをサポートしています。

- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256

Amazon MQ のための Identity and Access Management

AWS Identity and Access Management (IAM) は、管理者が AWS リソースへのアクセスを安全に制御 AWS のサービス するのに役立つ です。IAM 管理者は、誰が認証 (サインイン) され、Amazon MQ リソースを使用する認可 を受ける (許可がある) ことができるかを制御します。IAM は、追加料金なしで AWS のサービス 使用できる です。

トピック

- 対象者
- アイデンティティを使用した認証
- ポリシーを使用したアクセスの管理
- Amazon MQ で IAM が機能する仕組み
- Amazon MQ のアイデンティティベースポリシーの例
- Amazon MQ の API 認証と認可
- AWS Amazon MQ の マネージドポリシー
- Amazon MQ のサービスリンクロールの使用
- Amazon MQ アイデンティティとアクセスのトラブルシューティング

対象者

AWS Identity and Access Management (IAM) の使用方法は、Amazon MQ で行う作業によって異なります。

サービスユーザー – 業務を行うために Amazon MQ サービスを使用する場合は、管理者から必要な認証情報と許可が提供されます。業務のために使用する Amazon MQ 機能が増えるにつれて、追加の許可が必要になる可能性があります。アクセスの管理方法を理解しておくことは、管理者に適切な許可をリクエストするために役に立ちます。Amazon MQ の機能にアクセスできない場合は、「Amazon MQ アイデンティティとアクセスのトラブルシューティング」を参照してください。

サービス管理者 – 社内の Amazon MQ リソースを担当している場合は、Amazon MQ に対する完全なアクセス権があると思われます。サービスのユーザーがどの Amazon MQ 機能やリソースにアクセスするかを決めるのは管理者の仕事です。その後、IAM 管理者にリクエストを送信して、サービスユーザーの権限を変更する必要があります。このページの情報を点検して、IAM の基本概念を理解してください。会社で Amazon MQ と IAM を併用する方法の詳細については、「Amazon MQ で IAM が機能する仕組み」を参照してください。

IAM 管理者 – IAM 管理者には、Amazon MQ へのアクセスを管理するポリシーの作成方法の詳細を理解することが推奨されます。IAM で使用できる Amazon MQ のアイデンティティベースポリシーの例を確認するには、「Amazon MQ のアイデンティティベースポリシーの例」を参照してください。

アイデンティティを使用した認証

認証は、ID 認証情報 AWS を使用して にサインインする方法です。として、IAM ユーザーとして AWS アカウントのルートユーザー、または IAM ロールを引き受けることによって、認証 (にサインイン AWS) される必要があります。

ID ソースを介して提供された認証情報を使用して、フェデレーティッド ID AWS として にサインインできます。 AWS IAM Identity Center （ IAM Identity Center) ユーザー、会社のシングルサインオン認証、Google または Facebook 認証情報は、フェデレーティッド ID の例です。フェデレーティッド ID としてサインインする場合、IAM ロールを使用して、前もって管理者により ID フェデレーションが設定されています。フェデレーション AWS を使用して にアクセスすると、間接的にロールを引き受けることになります。

ユーザーの種類に応じて、 AWS Management Console または AWS アクセスポータルにサインインできます。へのサインインの詳細については AWS、 AWS サインイン ユーザーガイドの「 へのサインイン方法 AWS アカウント」を参照してください。

AWS プログラムで にアクセスする場合、 は、ソフトウェア開発キット (SDK) とコマンドライン インターフェイス (CLI) AWS を提供し、認証情報を使用してリクエストを暗号化して署名します。AWS ツールを使用しない場合は、自分でリクエストに署名する必要があります。リクエストに自分で署名する推奨方法の使用については、「IAM ユーザーガイド」の「API リクエストに対するAWS Signature Version 4」を参照してください。

使用する認証方法を問わず、追加セキュリティ情報の提供をリクエストされる場合もあります。例えば、 では、多要素認証 (MFA) を使用してアカウントのセキュリティを向上させる AWS ことをお勧めします。詳細については、「AWS IAM Identity Center ユーザーガイド」の「多要素認証」および「IAM ユーザーガイド」の「IAM のAWS 多要素認証」を参照してください。

AWS アカウント ルートユーザー

を作成するときは AWS アカウント、アカウント内のすべての およびリソースへの AWS のサービス 完全なアクセス権を持つ 1 つのサインインアイデンティティから始めます。この ID は AWS アカウント ルートユーザーと呼ばれ、アカウントの作成に使用した E メールアドレスとパスワードでサインインすることでアクセスできます。日常的なタスクには、ルートユーザーを使用しないことを強くお勧めします。ルートユーザーの認証情報は保護し、ルートユーザーでしか実行できないタスクを実行するときに使用します。ルートユーザーとしてサインインする必要があるタスクの完全なリストについては、「IAM ユーザーガイド」の「ルートユーザー認証情報が必要なタスク」を参照してください。

ユーザーとグループ

IAM ユーザーは、1 人のユーザーまたはアプリケーションに対して特定のアクセス許可 AWS アカウント を持つ 内の ID です。可能であれば、パスワードやアクセスキーなどの長期的な認証情報を保有する IAM ユーザーを作成する代わりに、一時的な認証情報を使用することをお勧めします。ただし、IAM ユーザーでの長期的な認証情報が必要な特定のユースケースがある場合は、アクセスキーをローテーションすることをお勧めします。詳細については、「IAM ユーザーガイド」の「長期的な認証情報を必要とするユースケースのためにアクセスキーを定期的にローテーションする」を参照してください。

IAM グループは、IAM ユーザーの集団を指定するアイデンティティです。グループとしてサインインすることはできません。グループを使用して、複数のユーザーに対して一度に権限を指定できます。多数のユーザーグループがある場合、グループを使用することで権限の管理が容易になります。例えば、IAMAdmins という名前のグループを設定して、そのグループに IAM リソースを管理する許可を与えることができます。

ユーザーは、ロールとは異なります。ユーザーは 1 人の人または 1 つのアプリケーションに一意に関連付けられますが、ロールはそれを必要とする任意の人が引き受けるようになっています。ユーザーには永続的な長期の認証情報がありますが、ロールでは一時認証情報が提供されます。詳細については、「IAM ユーザーガイド」の「IAM ユーザーに関するユースケース」を参照してください。

IAM ロール

IAM ロールは、特定のアクセス許可 AWS アカウント を持つ 内のアイデンティティです。これは IAM ユーザーに似ていますが、特定のユーザーには関連付けられていません。で IAM ロールを一時的に引き受けるには AWS Management Console、ユーザーから IAM ロール (コンソール) に切り替えることができます。ロールを引き受けるには、 または AWS API オペレーションを AWS CLI 呼び出すか、カスタム URL を使用します。ロールを使用する方法の詳細については、「IAM ユーザーガイド」の「ロールを引き受けるための各種方法」を参照してください。

IAM ロールと一時的な認証情報は、次の状況で役立ちます:

- フェデレーションユーザーアクセス – フェデレーティッド ID に許可を割り当てるには、ロールを作成してそのロールの許可を定義します。フェデレーティッド ID が認証されると、その ID はロールに関連付けられ、ロールで定義されている許可が付与されます。フェデレーションのロールについては、「IAM ユーザーガイド」の「サードパーティー ID プロバイダー (フェデレーション) 用のロールを作成する」を参照してください。IAM Identity Center を使用する場合は、許可セットを設定します。アイデンティティが認証後にアクセスできるものを制御するため、IAM Identity Center は、権限セットを IAM のロールに関連付けます。アクセス許可セットの詳細については、「AWS IAM Identity Center User Guide」の「Permission sets」を参照してください。
- 一時的な IAM ユーザー権限 - IAM ユーザーまたはロールは、特定のタスクに対して複数の異なる権限を一時的に IAM ロールで引き受けることができます。
- クロスアカウントアクセス - IAM ロールを使用して、自分のアカウントのリソースにアクセスすることを、別のアカウントの人物 (信頼済みプリンシパル) に許可できます。クロスアカウントアクセス権を付与する主な方法は、ロールを使用することです。ただし、一部の では AWS のサービス、 (ロールをプロキシとして使用する代わりに) ポリシーをリソースに直接アタッチできます。クロスアカウントアクセスにおけるロールとリソースベースのポリシーの違いについては、「IAM ユーザーガイド」の「IAM でのクロスアカウントのリソースへのアクセス」を参照してください。
- クロスサービスアクセス — 一部の は他の の機能 AWS のサービス を使用します AWS のサービス。例えば、あるサービスで呼び出しを行うと、通常そのサービスによって Amazon EC2 でアプリケーションが実行されたり、Amazon S3 にオブジェクトが保存されたりします。サービスで

は、呼び出し元プリンシパルの許可、サービスロール、またはサービスリンクロールを使用してこれを行う場合があります。

- 転送アクセスセッション (FAS) – IAM ユーザーまたはロールを使用して でアクションを実行すると AWS、プリンシパルと見なされます。一部のサービスを使用する際に、アクションを実行することで、別のサービスの別のアクションがトリガーされることがあります。FAS は、 を呼び出すプリンシパルのアクセス許可と AWS のサービス、ダウンストリームサービス AWS のサービス へのリクエストのリクエストをリクエストする を使用します。FAS リクエストは、サービスが他の AWS のサービス またはリソースとのやり取りを完了する必要があるリクエストを受け取った場合にのみ行われます。この場合、両方のアクションを実行するためのアクセス許可が必要です。FAS リクエストを行う際のポリシーの詳細については、「転送アクセスセッション」を参照してください。
- サービスロール - サービスがユーザーに代わってアクションを実行するために引き受ける IAM ロールです。IAM 管理者は、IAM 内からサービスロールを作成、変更、削除することができます。詳細については、「IAM ユーザーガイド」の「AWS のサービスに許可を委任するロールを作成する」を参照してください。
- サービスにリンクされたロール – サービスにリンクされたロールは、 にリンクされたサービスロールの一種です AWS のサービス。サービスは、ユーザーに代わってアクションを実行するロールを引き受けることができます。サービスにリンクされたロールは に表示され AWS アカウント 、 サービスによって所有されます。IAM 管理者は、サービスリンクロールのアクセス許可を表示できますが、編集することはできません。

- Amazon EC2 で実行されているアプリケーション – IAM ロールを使用して、EC2 インスタンスで実行され、 AWS CLI または AWS API リクエストを行うアプリケーションの一時的な認証情報を管理できます。これは、EC2 インスタンス内でのアクセスキーの保存に推奨されます。EC2 インスタンスに AWS ロールを割り当て、そのすべてのアプリケーションで使用できるようにするには、インスタンスにアタッチされたインスタンスプロファイルを作成します。インスタンスプロファイルにはロールが含まれ、EC2 インスタンスで実行されるプログラムは一時的な認証情報を取得できます。詳細については、「IAM ユーザーガイド」の「Amazon EC2 インスタンスで実行されるアプリケーションに IAM ロールを使用して許可を付与する」を参照してください。

ポリシーを使用したアクセスの管理

でアクセスを制御する AWS には、ポリシーを作成し、ID AWS またはリソースにアタッチします。ポリシーは AWS 、アイデンティティまたはリソースに関連付けられているときにアクセス許可を定義する のオブジェクトです。 は、プリンシパル (ユーザー、ルートユーザー、またはロールセッション) がリクエストを行うときに、これらのポリシー AWS を評価します。ポリシーでの権限に

より、リクエストが許可されるか拒否されるかが決まります。ほとんどのポリシーは JSON ドキュメント AWS として に保存されます。JSON ポリシードキュメントの構造と内容の詳細については、IAM ユーザーガイドの JSON ポリシー概要を参照してください。

管理者は JSON AWS ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどのリソースに対してどのような条件下でアクションを実行できるかということです。

デフォルトでは、ユーザーやロールに権限はありません。IAM 管理者は、リソースで必要なアクションを実行するための権限をユーザーに付与する IAM ポリシーを作成できます。その後、管理者はロールに IAM ポリシーを追加し、ユーザーはロールを引き受けることができます。

IAM ポリシーは、オペレーションの実行方法を問わず、アクションの許可を定義します。例えば、`iam:GetRole` アクションを許可するポリシーがあるとします。そのポリシーを持つユーザーは、 AWS Management Console、、 AWS CLIまたは AWS API からロール情報を取得できます。

アイデンティティベースのポリシー

アイデンティティベースポリシーは、IAM ユーザーグループ、ユーザーのグループ、ロールなど、アイデンティティにアタッチできる JSON 許可ポリシードキュメントです。これらのポリシーは、ユーザーとロールが実行できるアクション、リソース、および条件をコントロールします。アイデンティティベースポリシーの作成方法については、「IAM ユーザーガイド」の「カスタマー管理ポリシーでカスタム IAM アクセス許可を定義する」を参照してください。

アイデンティティベースのポリシーは、さらにインラインポリシーまたはマネージドポリシーに分類できます。インラインポリシーは、単一のユーザー、グループ、またはロールに直接埋め込まれています。管理ポリシーは、 内の複数のユーザー、グループ、ロールにアタッチできるスタンドアロンポリシーです AWS アカウント。管理ポリシーには、 AWS 管理ポリシーとカスタマー管理ポリシーが含まれます。マネージドポリシーまたはインラインポリシーのいずれかを選択する方法については、「IAM ユーザーガイド」の「管理ポリシーとインラインポリシーのいずれかを選択する」を参照してください。

リソースベースのポリシー

リソースベースのポリシーは、リソースに添付する JSON ポリシードキュメントです。リソースベースのポリシーには例として、IAM ロールの信頼ポリシーや Amazon S3 バケットポリシーがあげられます。リソースベースのポリシーをサポートするサービスでは、サービス管理者はポリシーを使用して特定のリソースへのアクセスを制御できます。ポリシーがアタッチされているリソースの場合、指定されたプリンシパルがそのリソースに対して実行できるアクションと条件は、ポリシーに

よって定義されます。リソースベースのポリシーでは、プリンシパルを指定する必要があります。プリンシパルには、アカウント、ユーザー、ロール、フェデレーティッドユーザー、または を含めることができます AWS のサービス。

リソースベースのポリシーは、そのサービス内にあるインラインポリシーです。リソースベースのポリシーでは、IAM の AWS マネージドポリシーを使用できません。

アクセスコントロールリスト (ACL)

アクセスコントロールリスト (ACL) は、どのプリンシパル (アカウントメンバー、ユーザー、またはロール) がリソースにアクセスするための許可を持つかを制御します。ACL はリソースベースのポリシーに似ていますが、JSON ポリシードキュメント形式は使用しません。

Amazon S3、 AWS WAF、および Amazon VPC は、ACLs。ACL の詳細については、「Amazon Simple Storage Service デベロッパーガイド」の「アクセスコントロールリスト (ACL) の概要」を参照してください。

その他のポリシータイプ

AWS は、追加のあまり一般的ではないポリシータイプをサポートします。これらのポリシータイプでは、より一般的なポリシータイプで付与された最大の権限を設定できます。

- アクセス許可の境界 - アクセス許可の境界は、アイデンティティベースポリシーによって IAM エンティティ (IAM ユーザーまたはロール) に付与できる権限の上限を設定する高度な機能です。エンティティにアクセス許可の境界を設定できます。結果として得られる権限は、エンティティのアイデンティティベースポリシーとそのアクセス許可の境界の共通部分になります。`Principal` フィールドでユーザーまたはロールを指定するリソースベースのポリシーでは、アクセス許可の境界は制限されません。これらのポリシーのいずれかを明示的に拒否した場合、権限は無効になります。アクセス許可の境界の詳細については、「IAM ユーザーガイド」の「IAM エンティティのアクセス許可の境界」を参照してください。
- サービスコントロールポリシー (SCPs) – SCPsは、 の組織または組織単位 (OU) の最大アクセス許可を指定する JSON ポリシーです AWS Organizations。 AWS Organizations は、ビジネスが所有する複数の をグループ化して一元管理するためのサービス AWS アカウント です。組織内のすべての機能を有効にすると、サービスコントロールポリシー (SCP) を一部またはすべてのアカウントに適用できます。SCP は、各 を含むメンバーアカウントのエンティティのアクセス許可を制限します AWS アカウントのルートユーザー。Organizations と SCP の詳細については、「AWS Organizations ユーザーガイド」の「サービスコントロールポリシー (SCP)」を参照してください。

- リソースコントロールポリシー (RCP) – RCP は、所有する各リソースにアタッチされた IAM ポリシーを更新することなく、アカウント内のリソースに利用可能な最大数のアクセス許可を設定するために使用できる JSON ポリシーです。RCP は、メンバーアカウントのリソースに対するアクセス許可を制限し、組織に属するかどうかにかかわらず AWS アカウントのルートユーザー、 を含む ID に対する有効なアクセス許可に影響を与える可能性があります。RCP AWS のサービス をサポートする のリストを含む Organizations と RCPs「リソースコントロールポリシー (RCPs」を参照してください。 AWS Organizations
- セッションポリシー - セッションポリシーは、ロールまたはフェデレーションユーザーの一時的なセッションをプログラムで作成する際にパラメータとして渡す高度なポリシーです。結果としてセッションの権限は、ユーザーまたはロールのアイデンティティベースポリシーとセッションポリシーの共通部分になります。また、リソースベースのポリシーから権限が派生する場合もあります。これらのポリシーのいずれかを明示的に拒否した場合、権限は無効になります。詳細については、「IAM ユーザーガイド」の「セッションポリシー」を参照してください。

複数のポリシータイプ

1 つのリクエストに複数のタイプのポリシーが適用されると、結果として作成される権限を理解するのがさらに難しくなります。複数のポリシータイプが関係する場合にリクエストを許可するかどうか AWS を決定する方法については、IAM ユーザーガイドの「ポリシー評価ロジック」を参照してください。

Amazon MQ で IAM が機能する仕組み

IAM を使用して Amazon MQ へのアクセスを管理する前に、Amazon MQ で使用できる IAM 機能について理解しておく必要があります。Amazon MQ およびその他の AWS のサービスが IAM と連携する方法の概要を把握するには、IAM ユーザーガイドのAWS 「IAM と連携する のサービス」を参照してください。

Amazon MQ は、作成、更新、および削除操作に IAM を使用しますが、ブローカーにはネイティブ ActiveMQ 認証を使用します。詳細については、「ActiveMQ ブローカーの LDAP との統合」を参照してください。

トピック

- Amazon MQ のアイデンティティベースポリシー
- Amazon MQ のリソースベースポリシー
- Amazon MQ タグに基づいた認可
- Amazon MQ の IAM ロール

Amazon MQ のアイデンティティベースポリシー

IAM アイデンティティベースポリシーでは、許可または拒否するアクションとリソース、またアクションを許可または拒否する条件を指定できます。Amazon MQ は、特定のアクション、リソース、および条件キーをサポートしています。JSON ポリシーで使用するすべての要素については、「IAM ユーザーガイド」の「IAM JSON ポリシー要素のリファレンス」を参照してください。

アクション

管理者は JSON AWS ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどのリソースに対してどのような条件下でアクションを実行できるかということです。

JSON ポリシーの `Action` 要素にはポリシー内のアクセスを許可または拒否するために使用できるアクションが記述されます。ポリシーアクションの名前は通常、関連付けられた AWS API オペレーションと同じです。一致する API オペレーションのない許可のみのアクションなど、いくつかの例外があります。また、ポリシーに複数のアクションが必要なオペレーションもあります。これらの追加アクションは依存アクションと呼ばれます。

このアクションは関連付けられたオペレーションを実行するためのアクセス許可を付与するポリシーで使用されます。

Amazon MQ のポリシーアクションは、アクションの前にプレフィックス `mq:` を使用します。例えば、Amazon MQ `CreateBroker` API オペレーションで Amazon MQ インスタンスを実行する許可を付与するには、ユーザーのポリシーに `mq:CreateBroker` アクションを含めます。ポリシーステートメントには、`Action` または `NotAction` エレメントを含める必要があります。Amazon MQ は、このサービスで実行できるタスクを記述する独自のアクションのセットを定義します。

単一のステートメントに複数のアクションを指定するには次のようにコンマで区切ります。

```
"Action": [
      "mq:action1",
      "mq:action2"
```

ワイルドカード (*) を使用して複数アクションを指定できます。例えば、`Describe` という単語で始まるすべてのアクションを指定するには、次のアクションを含めます。

```
"Action": "mq:Describe*"
```

Amazon MQ アクションのリストを確認するには、IAM ユーザーガイドの「Amazon MQ で定義されるアクション」を参照してください。

リソース

管理者は JSON AWS ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルが、どのリソースに対してどのような条件下でアクションを実行できるかということです。

Resource JSON ポリシー要素はアクションが適用されるオブジェクトを指定します。ステートメントにはResource または NotResource 要素を含める必要があります。ベストプラクティスとして、Amazon リソースネーム (ARN) を使用してリソースを指定します。これは、リソースレベルの許可と呼ばれる特定のリソースタイプをサポートするアクションに対して実行できます。

オペレーションのリスト化など、リソースレベルの権限をサポートしないアクションの場合は、ステートメントがすべてのリソースに適用されることを示すために、ワイルドカード (*) を使用します。

```
"Resource": "*"
```

Amazon MQ では、プライマリ AWS リソースは Amazon MQ メッセージブローカーとその設定です。Amazon MQ ブローカーと設定には、以下の表にあるとおり、それぞれ一意の Amazon リソースネーム (ARN) が関連付けられています。

リソースタイプ	ARN	条件キー
brokers	arn:aws:mq:us-east-1:123456789012:broker:${brokerName}:${brokerId}	aws:ResourceTag/${TagKey}
configurations	arn:${Partition}:mq:${Region}:${Account}:configuration:${configuration-id}	aws:ResourceTag/${TagKey}

ARN の形式の詳細については、「Amazon リソースネーム (ARNs AWS 「サービス名前空間」を参照してください。

例えば、ステートメントでブローカー ID b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9 を持つ MyBroker というブローカーを指定するには、次の ARN を使用します。

```
"Resource": "arn:aws:mq:us-
east-1:123456789012:broker:MyBroker:b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9"
```

特定のアカウントに属するすべてのブローカーと設定を指定するには、ワイルドカード (*) を使用します。

```
"Resource": "arn:aws:mq:us-east-1:123456789012:*"
```

リソースを作成するためのアクションなど、Amazon MQ アクションには特定のリソースで実行できないものがあります。このような場合はワイルドカード *を使用する必要があります。

```
"Resource": "*"
```

API アクション CreateTags には、ブローカーと設定の両方が必要です。複数リソースを単一ステートメントで指定するには、ARN をカンマで区切ります。

```
"Resource": [
      "resource1",
      "resource2"
```

Amazon MQ のリソースタイプとそれらの ARN のリストを確認するには、IAM ユーザーガイドの「Amazon MQ で定義されるリソースタイプ」を参照してください。どのアクションで各リソースの ARN を指定できるかについては、「Amazon MQ で定義されるアクション」を参照してください。

条件キー

管理者は JSON AWS ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルが、どのリソースに対してどのような条件下でアクションを実行できるかということです。

Condition 要素 (または Condition ブロック) を使用すると、ステートメントが有効な条件を指定できます。Condition 要素はオプションです。イコールや未満などの 条件演算子 を使用して条件式を作成して、ポリシーの条件とリクエスト内の値を一致させることができます。

1 つのステートメントに複数の Condition 要素を指定する場合、または 1 つの Condition 要素に複数のキーを指定する場合、 AWS では AND 論理演算子を使用してそれらを評価します。1 つの条

件キーに複数の値を指定すると、 は論理ORオペレーションを使用して条件 AWS を評価します。ステートメントの権限が付与される前にすべての条件が満たされる必要があります。

条件を指定する際にプレースホルダー変数も使用できます。例えば IAM ユーザーに、IAM ユーザー名がタグ付けされている場合のみリソースにアクセスできる権限を付与することができます。詳細については、「IAM ユーザーガイド」の「IAM ポリシーの要素: 変数およびタグ」を参照してください。

AWS は、グローバル条件キーとサービス固有の条件キーをサポートしています。すべての AWS グローバル条件キーを確認するには、IAM ユーザーガイドのAWS 「グローバル条件コンテキストキー」を参照してください。

Amazon MQ はサービス固有の条件キーを定義しませんが、いくつかのグローバル条件キーの使用がサポートされています。Amazon MQ の条件キーのリストを確認するには、IAM ユーザーガイドの「Amazon MQ の条件キー」を参照してください。条件キーを使用できるアクションとリソースについては、「Amazon MQ で定義されるアクション」を参照してください。

条件キー	説明	[Type] (タイプ)
aws:RequestTag/${TagKey}	リクエストで渡されたタグに基づいてアクションをフィルタリングします。	String
aws:ResourceTag/${TagKey}	リソースに関連付けられているタグに基づいてアクションをフィルタリングします。	String
aws:TagKeys	リクエストで渡されたタグキーに基づいてアクションをフィルタリングします。	String

例

Amazon MQ のアイデンティティベースポリシーの例を確認するには、「Amazon MQ のアイデンティティベースポリシーの例」を参照してください。

Amazon MQ のリソースベースポリシー

現在、Amazon MQ はリソースベースの許可またはリソースベースのポリシーを使用した IAM 認証をサポートしていません。

Amazon MQ タグに基づいた認可

タグは、Amazon MQ リソースにアタッチする、または Amazon MQ へのリクエストで渡すことができます。タグに基づいてアクセスを制御するには`mq:ResourceTag/`*`key-name`*、`aws:RequestTag/`*`key-name`*、または `aws:TagKeys` の条件キーを使用して、ポリシーの条件要素でタグ情報を提供します。

Amazon MQ はタグベースのポリシーをサポートしています。例えば、キー `environment` および値 `production` を持つタグが含まれる Amazon MQ リソースへのアクセスを拒否することができます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "mq:DeleteBroker",
                "mq:RebootBroker",
                "mq:DeleteTags"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/environment": "production"
                }
            }
        }
    ]
}
```

このポリシーは、`environment/production` タグが含まれる Amazon MQ ブローカーを削除または再起動する能力を `Deny` します。

タグ付けの詳細については、以下を参照してください。

- Amazon MQ リソースへのタグの追加
- IAM タグを使用したアクセスの制御

Amazon MQ の IAM ロール

IAM ロールは、特定のアクセス許可を持つ AWS アカウント内のエンティティです。

Amazon MQ での一時的な認証情報の使用

一時的な認証情報を使用して、フェデレーションでサインインする、IAM 役割を引き受ける、またはクロスアカウント役割を引き受けることができます。一時的なセキュリティ認証情報を取得するには、AssumeRole や GetFederationToken などの AWS STS API オペレーションを呼び出します。

Amazon MQ は、一時的な認証情報の使用をサポートします。

サービス役割

この機能により、ユーザーに代わってサービスがサービス役割を引き受けることが許可されます。この役割により、サービスがお客様に代わって他のサービスのリソースにアクセスし、アクションを完了することが許可されます。サービス役割はIAM アカウントに表示され、アカウントによって所有されます。つまり、IAM 管理者はこの役割の権限を変更できます。ただし、それにより、サービスの機能が損なわれる場合があります。

Amazon MQ は、サービスロールをサポートします。

Amazon MQ のアイデンティティベースポリシーの例

デフォルトでは、ユーザーとロールには Amazon MQ リソースを作成または変更するアクセス許可がありません。また、 AWS Management Console、 AWS CLI、または AWS API を使用してタスクを実行することはできません。IAM 管理者は、ユーザーとロールに必要な、指定されたリソースで特定の API オペレーションを実行する権限をユーザーとロールに付与する IAM ポリシーを作成する必要があります。続いて、管理者はそれらの権限が必要な IAM ユーザーまたはグループにそのポリシーをアタッチする必要があります。

JSON ポリシードキュメントのこれらの例を使用して、IAM アイデンティティベースのポリシーを作成する方法については、「IAM ユーザーガイド」の「JSON タブでのポリシーの作成」を参照してください。

トピック

- ポリシーに関するベストプラクティス
- Amazon MQ コンソールの使用
- ユーザーが自分の許可を表示できるようにする

ポリシーに関するベストプラクティス

ID ベースのポリシーは、ユーザーのアカウント内で誰かが Amazon MQ リソースを作成、アクセス、または削除できるどうかを決定します。これらのアクションを実行すると、 AWS アカウントに料金が発生する可能性があります。アイデンティティベースポリシーを作成したり編集したりする際には、以下のガイドラインと推奨事項に従ってください:

- AWS 管理ポリシーを開始し、最小特権のアクセス許可に移行 – ユーザーとワークロードにアクセス許可の付与を開始するには、多くの一般的なユースケースにアクセス許可を付与するAWS 管理ポリシーを使用します。これらは で使用できます AWS アカウント。ユースケースに固有の AWS カスタマー管理ポリシーを定義することで、アクセス許可をさらに減らすことをお勧めします。詳細については、「IAM ユーザーガイド」の「AWS マネージドポリシー」または「ジョブ機能の AWS マネージドポリシー」を参照してください。
- 最小特権を適用する – IAM ポリシーで許可を設定する場合は、タスクの実行に必要な許可のみを付与します。これを行うには、特定の条件下で特定のリソースに対して実行できるアクションを定義します。これは、最小特権アクセス許可とも呼ばれています。IAM を使用して許可を適用する方法の詳細については、「IAM ユーザーガイド」の「IAM でのポリシーとアクセス許可」を参照してください。
- IAM ポリシーで条件を使用してアクセスをさらに制限する - ポリシーに条件を追加して、アクションやリソースへのアクセスを制限できます。例えば、ポリシー条件を記述して、すべてのリクエストを SSL を使用して送信するように指定できます。条件を使用して、 などの特定の を通じてサービスアクションが使用される場合に AWS のサービス、サービスアクションへのアクセスを許可することもできます AWS CloudFormation。詳細については、「IAM ユーザーガイド」の「IAM JSON ポリシー要素:条件」を参照してください。
- IAM Access Analyzer を使用して IAM ポリシーを検証し、安全で機能的な権限を確保する - IAM Access Analyzer は、新規および既存のポリシーを検証して、ポリシーが IAM ポリシー言語 (JSON) および IAM のベストプラクティスに準拠するようにします。IAM アクセスアナライザーは 100 を超えるポリシーチェックと実用的な推奨事項を提供し、安全で機能的なポリシーの作成をサポートします。詳細については、「IAM ユーザーガイド」の「IAM Access Analyzer でポリシーを検証する」を参照してください。
- 多要素認証 (MFA) を要求する – で IAM ユーザーまたはルートユーザーを必要とするシナリオがある場合は AWS アカウント、セキュリティを強化するために MFA を有効にします。API オペレーションが呼び出されるときに MFA を必須にするには、ポリシーに MFA 条件を追加します。詳細については、「IAM ユーザーガイド」の「MFA を使用した安全な API アクセス」を参照してください。

IAM でのベストプラクティスの詳細については、IAM ユーザーガイドの IAM でのセキュリティのベストプラクティスを参照してください。

Amazon MQ コンソールの使用

Amazon MQ コンソールにアクセスするには、許可の最小限のセットが必要です。これらのアクセス許可により、 AWS アカウントの Amazon MQ リソースの詳細を一覧表示および表示できます。最小限必要な許可よりも厳しく制限されたアイデンティティベースポリシーを作成すると、そのポリシーを添付したエンティティ (IAM ユーザーまたはロール) に対してコンソールが意図したとおりに機能しません。

これらのエンティティが引き続き Amazon MQ コンソールを使用できるようにするには、エンティティに次の AWS 管理ポリシーもアタッチします。詳細については、「IAM ユーザーガイド」の「ユーザーへのアクセス許可の追加」を参照してください。

```
AmazonMQReadOnlyAccess
```

AWS CLI または AWS API のみを呼び出すユーザーには、最小限のコンソールアクセス許可を付与する必要はありません。代わりに、実行しようとしている API オペレーションに一致するアクションのみへのアクセスが許可されます。

ユーザーが自分の許可を表示できるようにする

この例では、ユーザーアイデンティティにアタッチされたインラインおよびマネージドポリシーの表示を IAM ユーザーに許可するポリシーの作成方法を示します。このポリシーには、コンソールで、または AWS CLI または AWS API を使用してプログラムでこのアクションを実行するアクセス許可が含まれています。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
```

```
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Amazon MQ の API 認証と認可

Amazon MQ は API 認証に標準 AWS リクエスト署名を使用します。詳細については、『AWS』の「AWS 全般のリファレンス API リクエストの署名」を参照してください。

> **Note**
>
> 現在、Amazon MQ はリソースベースの許可またはリソースベースのポリシーを使用した IAM 認証をサポートしていません。

ブローカー、設定、 AWS およびユーザーの使用をユーザーに許可するには、IAM ポリシーのアクセス許可を編集する必要があります。

トピック

- Amazon MQ ブローカーを作成するために必要な IAM 許可
- Amazon MQ REST API 許可リファレンス
- Amazon MQ API アクションに対するリソースレベルの許可

Amazon MQ ブローカーを作成するために必要な IAM 許可

ブローカーを作成するには、AmazonMQFullAccess IAM ポリシーを使用するか、以下の EC2 許可を IAM ポリシーに含める必要があります。

以下のカスタムポリシーは、ActiveMQ ブローカーを作成するために Amazon MQ が必要とするリソースを操作するための許可を付与する 2 つのステートメント (1 つは条件付き) で構成されています。

> **Important**
>
> - ec2:CreateNetworkInterface アクションは、ユーザーに代わってアカウントに Elastic Network Interface (ENI) を作成することを Amazon MQ に許可するために必要です。
> - ec2:CreateNetworkInterfacePermission アクションは、Amazon MQ が ENI を ActiveMQ ブローカーにアタッチすることを認可します。
> - ec2:AuthorizedService 条件キーは、ENI 許可が Amazon MQ サービスアカウントのみに付与されることを確実にします。

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Action": [
            "mq:*",
            "ec2:CreateNetworkInterface",
            "ec2:DeleteNetworkInterface",
            "ec2:DetachNetworkInterface",
            "ec2:DescribeInternetGateways",
            "ec2:DescribeNetworkInterfaces",
            "ec2:DescribeRouteTables",
            "ec2:DescribeSecurityGroups",
            "ec2:DescribeSubnets",
            "ec2:DescribeVpcs"
        ],
        "Effect": "Allow",
        "Resource": "*"
    },{
        "Action": [
            "ec2:CreateNetworkInterfacePermission",
```

```
            "ec2:DeleteNetworkInterfacePermission",
            "ec2:DescribeNetworkInterfacePermissions"
        ],
        "Effect": "Allow",
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "ec2:AuthorizedService": "mq.amazonaws.com"
            }
        }
    }]
}
```

詳細については、ステップ 2: ユーザーを作成して AWS 認証情報を取得するおよびAmazon MQ Elastic Network Interface を変更または削除しないを参照してください。

Amazon MQ REST API 許可リファレンス

以下の表には、Amazon MQ REST API と、それらに対応する IAM 許可がリストされています。

Amazon MQ REST API と必要な許可

Amazon MQ REST API	必要な許可
CreateBroker	mq:CreateBroker
CreateConfiguration	mq:CreateConfiguration
CreateTags	mq:CreateTags
CreateUser	mq:CreateUser
DeleteBroker	mq:DeleteBroker
DeleteUser	mq:DeleteUser
DescribeBroker	mq:DescribeBroker
DescribeConfiguration	mq:DescribeConfiguration
DescribeConfigurationRevision	mq:DescribeConfigurationRevision
DescribeUser	mq:DescribeUser

Amazon MQ REST API	必要な許可
ListBrokers	mq:ListBrokers
ListConfigurationRevisions	mq:ListConfigurationRevisions
ListConfigurations	mq:ListConfigurations
ListTags	mq:ListTags
ListUsers	mq:ListUsers
RebootBroker	mq:RebootBroker
UpdateBroker	mq:UpdateBroker
UpdateConfiguration	mq:UpdateConfiguration
UpdateUser	mq:UpdateUser

Amazon MQ API アクションに対するリソースレベルの許可

リソースレベルの許可とは、ユーザーがアクションを実行できるリソースを指定する能力を意味します。Amazon MQ は、リソースレベルの許可を部分的にサポートします。特定の Amazon MQ アクションでは、満たす必要がある条件、またはユーザーが使用できる特定のリソースに基づいて、ユーザーにこれらのアクションの使用が許可されるタイミングを制御できます。

以下の表では、現在リソースレベルの許可をサポートしている Amazon MQ API アクションと、各アクションに対してサポートされるリソース、リソース ARN、条件キーを説明します。

> **Important**
>
> Amazon MQ API アクションがこの表に示されていない場合、そのアクションはリソースレベルの許可をサポートしていません。Amazon MQ API アクションがリソースレベルの許可をサポートしない場合、アクションを使用する許可をユーザーに付与できますが、ポリシーステートメントのリソース要素にワイルドカード (*) を指定する必要があります。

API アクション	リソースタイプ (* 必須)
CreateConfiguration	設定*
CreateTags	ブローカー、設定
CreateUser	ブローカー
DeleteBroker	ブローカー
DeleteUser	ブローカー
DescribeBroker	ブローカー
DescribeConfiguration	設定*
DescribeConfigurationRevision	設定*
DescribeUser	ブローカー
ListConfigurationRevisions	設定*
ListConfigurationRevisions	設定*
ListTags	ブローカー、設定
ListUsers	ブローカー
RebootBroker	ブローカー
UpdateBroker	ブローカー
UpdateConfiguration	設定*
UpdateUser	ブローカー

AWS Amazon MQ の マネージドポリシー

AWS 管理ポリシーは、 によって作成および管理されるスタンドアロンポリシーです AWS。 AWS 管理ポリシーは、多くの一般的なユースケースに対するアクセス許可を付与するように設計されているため、ユーザー、グループ、ロールへのアクセス許可の割り当てを開始できます。

AWS 管理ポリシーは、すべての AWS お客様が使用できるため、特定のユースケースに対して最小特権のアクセス許可を付与しない場合があることに注意してください。ユースケースに固有のカスタマー管理ポリシーを定義して、アクセス許可を絞り込むことをお勧めします。

AWS 管理ポリシーで定義されているアクセス許可は変更できません。が AWS 管理ポリシーで定義されたアクセス許可 AWS を更新すると、ポリシーがアタッチされているすべてのプリンシパル ID (ユーザー、グループ、ロール) が更新されます。 AWS は、新しい が起動されるか、新しい API オペレーション AWS のサービス が既存のサービスで使用できるようになったときに、 AWS 管理ポリシーを更新する可能性が最も高くなります。

詳細については「IAM ユーザーガイド」の「AWS マネージドポリシー」を参照してください。

Amazon MQ は、以下の AWS 管理ポリシーをサポートしています。

- AmazonMQApiFullAccess
- AmazonMQApiReadOnlyAccess
- AmazonMQFullAccess
- AmazonMQReadOnlyAccess
- AmazonMQServiceRolePolicy

AWS マネージドポリシー: AmazonMQServiceRolePolicy

IAM エンティティに `AmazonMQServiceRolePolicy` をアタッチすることはできません。このポリシーは、Amazon MQ がユーザーに代わってアクションを実行することを許可するサービスリンクロールにアタッチされます。この許可ポリシーと、それが Amazon MQ に実行を許可するアクションの詳細については、「the section called “Amazon MQ のサービスリンクロール許可”」を参照してください。

AWS マネージドポリシーへの Amazon MQ 更新

このサービスがこれらの変更の追跡を開始してからの Amazon MQ の AWS マネージドポリシーの更新に関する詳細を表示します。このページへの変更に関する自動アラートを受け取るには、Amazon MQ のドキュメント履歴ページで RSS フィードにサブスクライブしてください。

変更	説明	日付
Amazon MQ が変更の追跡を開始しました。	Amazon MQ は、 AWS マネージドポリシーの変更の追跡を開始しました。	2021 年 5 月 5 日

Amazon MQ のサービスリンクロールの使用

Amazon MQ は AWS Identity and Access Management (IAM) サービスにリンクされたロールを使用します。サービスリンクロールは、Amazon MQ に直接リンクされた特殊なタイプの IAM ロールです。サービスにリンクされたロールは Amazon MQ によって事前定義されており、サービスがユーザーに代わって他の AWS サービスを呼び出すために必要なすべてのアクセス許可が含まれています。

必要な許可を手動で追加する必要がないため、サービスリンクロールは Amazon MQ のセットアップを容易にします。サービスリンクロールの許可は Amazon MQ が定義し、別段の定義がない限り、Amazon MQ のみがそのロールを引き受けることができます。定義される許可は信頼ポリシーと許可ポリシーに含まれており、その許可ポリシーを他の IAM エンティティにアタッチすることはできません。

サービスリンクロールを削除するには、最初に関連リソースを削除する必要があります。これは、リソースにアクセスするための許可を誤って削除できないため、Amazon MQ リソースを保護します。

サービスにリンクされたロールをサポートするその他のサービスについては、IAM と連携するAWS のサービスを参照の上、 サービスにリンクされたロール 列が はい になっているサービスを検索してください。サービスリンクロールに関するドキュメントをサービスで表示するには、[Yes] (はい) リンクを選択します。

Amazon MQ のサービスリンクロール許可

Amazon MQ は、AWSServiceRoleForAmazonMQ という名前のサービスにリンクされたロールを使用します。Amazon MQ は、このサービスにリンクされたロールを使用してユーザーに代わって AWS サービスを呼び出します。

AWSServiceRoleForAmazonMQ サービスリンクロールは、ロールの引き受けに以下のサービスを信頼します。

- mq.amazonaws.com

Amazon MQ は、指定されたリソースで以下のアクションを完了するために、AWSServiceRoleForAmazonMQ サービスリンクロールにアタッチされる許可ポリシー AmazonMQServiceRolePolicy を使用します。

- アクション: vpc リソースでの ec2:CreateVpcEndpoint アクション。
- アクション: subnet リソースでの ec2:CreateVpcEndpoint アクション。
- アクション: security-group リソースでの ec2:CreateVpcEndpoint アクション。
- アクション: vpc-endpoint リソースでの ec2:CreateVpcEndpoint アクション。
- アクション: vpc リソースでの ec2:DescribeVpcEndpoints アクション。
- アクション: subnet リソースでの ec2:DescribeVpcEndpoints アクション。
- アクション: vpc-endpoint リソースでの ec2:CreateTags アクション。
- アクション: log-group リソースでの logs:PutLogEvents アクション。
- アクション: log-group リソースでの logs:DescribeLogStreams アクション。
- アクション: log-group リソースでの logs:DescribeLogGroups アクション。
- アクション: log-group リソースでの CreateLogStream アクション。
- アクション: log-group リソースでの CreateLogGroup アクション。

Amazon MQ for RabbitMQ ブローカーの作成時、AmazonMQServiceRolePolicy 許可ポリシーは、Amazon MQ がユーザーに代わって以下のタスクを実行することを許可します。

- ユーザー指定の Amazon VPC、サブネット、およびセキュリティグループを使用して、ブローカーの Amazon VPC エンドポイントを作成する。ブローカー用に作成されたエンドポイントは、RabbitMQ マネジメントコンソール、Management API、またはプログラム経由でブローカーに接続するために使用できます。
- ロググループを作成して、ブローカーログを Amazon CloudWatch Logs に発行する。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeVpcEndpoints"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateVpcEndpoint"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:vpc/*",
                "arn:aws:ec2:*:*:subnet/*",
                "arn:aws:ec2:*:*:security-group/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateVpcEndpoint"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:vpc-endpoint/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/AMQManaged": "true"
                }
            }
        },
```

```
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": "arn:aws:ec2:*:*:vpc-endpoint/*",
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": "CreateVpcEndpoint"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DeleteVpcEndpoints"
            ],
            "Resource": "arn:aws:ec2:*:*:vpc-endpoint/*",
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/AMQManaged": "true"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:PutLogEvents",
                "logs:DescribeLogStreams",
                "logs:DescribeLogGroups",
                "logs:CreateLogStream",
                "logs:CreateLogGroup"
            ],
            "Resource": [
                "arn:aws:logs:*:*:log-group:/aws/amazonmq/*"
            ]
        }
    ]
}
```

サービスリンクロールの作成、編集、削除を IAM エンティティ (ユーザー、グループ、ロールなど) に許可するには、許可を設定する必要があります。詳細については、IAM ユーザーガイドの「サービスリンクロールの許可」を参照してください。

Amazon MQ のサービスリンクロールの作成

サービスリンクロールを手動で作成する必要はありません。ブローカーを初めて作成すると、Amazon MQ はユーザーに代わって AWS サービスを呼び出すサービスにリンクされたロールを作成します。その後作成するすべてのブローカーには同じロールが使用され、新しいロールは作成されません。

> **Important**
>
> このサービスリンクロールがアカウントに表示されるのは、このロールでサポートされている機能を使用する別のサービスでアクションが完了した場合です。詳細については、「IAM アカウントに新しいロールが表示される」を参照してください。

このサービスリンクロールを削除した後で再度作成する必要が生じた場合は、同じ手順でアカウントにロールを再作成できます。

IAM コンソールを使用して、Amazon MQ ユースケースでサービスリンクロールを作成することもできます。 AWS CLI または AWS API で、サービス名を使用して`mq.amazonaws.com`サービスにリンクされたロールを作成します。詳細については、「IAM ユーザーガイド」の「サービスリンクロールの作成」を参照してください。このサービスリンクロールを削除しても、同じ方法でロールを再作成できます。

> **Important**
>
> サービスにリンクされたロールは、Amazon MQ for RabbitMQ に対してのみ作成されます。

Amazon MQ のサービスリンクロールの編集

Amazon MQ は、AWSServiceRoleForAmazonMQ サービスリンクロールの編集を許可しません。ただし、IAM を使用してロールの説明を編集することはできます。詳細については、IAM ユーザーガイドの「サービスリンクロールの編集」を参照してください。

Amazon MQ のサービスリンクロールの削除

サービスリンクロールを必要とする機能やサービスが不要になった場合は、ロールを削除することをお勧めします。そうすることで、積極的にモニタリングまたは保守されていない未使用のエンティ

ティを排除できます。ただし、手動で削除する前に、サービスリンクロールのリソースをクリーンアップする必要があります。

> **Note**
>
> リソースを削除しようとしているときに Amazon MQ サービスがロールを使用している場合は、削除が失敗する可能性があります。失敗した場合は数分待ってから操作を再試行してください。

AWSServiceRoleForAmazonMQ が使用する Amazon MQ リソースを削除する

- AWS Management Console、Amazon MQ CLI、または Amazon MQ API を使用して Amazon MQ ブローカーを削除します。ブローカーの削除の詳細については、「???」を参照してください。

サービスリンクロールを IAM で手動削除するには

IAM コンソール、 AWS CLI、または AWS API を使用して、AWSServiceRoleForAmazonMQ サービスにリンクされたロールを削除します。詳細については、IAM ユーザーガイドの「サービスリンクロールの削除」を参照してください。

Amazon MQ サービスリンクロールがサポートされるリージョン

Amazon MQ は、このサービスを利用できるすべてのリージョンでサービスリンクロールの使用をサポートします。詳細については、「AWS リージョンとエンドポイント」を参照してください。

リージョン名	リージョン識別子	Amazon MQ でのサポート
米国東部 (バージニア北部)	us-east-1	はい
米国東部 (オハイオ)	us-east-2	はい
米国西部 (北カリフォルニア)	us-west-1	はい
米国西部 (オレゴン)	us-west-2	はい
アジアパシフィック (ムンバイ)	ap-south-1	はい

リージョン名	リージョン識別子	Amazon MQ でのサポート
アジアパシフィック (大阪)	ap-northeast-3	はい
アジアパシフィック (ソウル)	ap-northeast-2	はい
アジアパシフィック (シンガポール)	ap-southeast-1	はい
アジアパシフィック (シドニー)	ap-southeast-2	はい
アジアパシフィック (東京)	ap-northeast-1	はい
カナダ (中部)	ca-central-1	はい
欧州 (フランクフルト)	eu-central-1	はい
欧州 (アイルランド)	eu-west-1	はい
欧州 (ロンドン)	eu-west-2	はい
欧州 (パリ)	eu-west-3	はい
南米 (サンパウロ)	sa-east-1	はい
AWS GovCloud (US)	us-gov-west-1	いいえ

Amazon MQ アイデンティティとアクセスのトラブルシューティング

以下の情報を使用して、Amazon MQ と IAM の使用時に発生する可能性がある一般的な問題の診断と修正に役立てます。

トピック

- Amazon MQ でアクションを実行する認可がない
- iam:PassRole を実行する権限がない
- AWS アカウント外のユーザーに Amazon MQ リソースへのアクセスを許可したい

Amazon MQ でアクションを実行する認可がない

がアクションを実行する権限がないと AWS Management Console 通知した場合は、管理者に連絡してサポートを依頼する必要があります。管理者は、サインイン認証情報を提供した担当者です。

以下の例のエラーは、mateojackson ユーザーがコンソールを使用して、######の詳細を表示しようとしましたが、mq:*GetWidget* アクセス許可がない場合に発生します。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 mq:GetWidget on resource: my-example-widget
```

この場合、Mateo は、mq:*GetWidget* アクションを使用して *my-example-widget* リソースにアクセスできるように、管理者にポリシーの更新を依頼します。

iam:PassRole を実行する権限がない

iam:PassRole アクションを実行する権限がないというエラーが表示された場合は、ポリシーを更新して Amazon MQ にロールを渡せるようにする必要があります。

一部の AWS のサービス では、新しいサービスロールまたはサービスにリンクされたロールを作成する代わりに、既存のロールをそのサービスに渡すことができます。そのためには、サービスにロールを渡す権限が必要です。

以下のエラー例は、marymajor という名前の IAM ユーザーが、コンソールを使用して Amazon MQ でアクションを実行しようするときに発生します。ただし、このアクションをサービスが実行するには、サービスロールから付与された権限が必要です。メアリーには、ロールをサービスに渡す許可がありません。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

この場合、Mary のポリシーを更新してメアリーに iam:PassRole アクションの実行を許可する必要があります。

サポートが必要な場合は、 AWS 管理者にお問い合わせください。サインイン認証情報を提供した担当者が管理者です。

AWS アカウント外のユーザーに Amazon MQ リソースへのアクセスを許可したい

他のアカウントのユーザーや組織外の人が、リソースにアクセスするために使用できるロールを作成できます。ロールの引き受けを委託するユーザーを指定できます。リソースベースのポリシーまた

はアクセスコントロールリスト (ACL) をサポートするサービスの場合、それらのポリシーを使用して、リソースへのアクセスを付与できます。

詳細については、以下を参照してください:

- Amazon MQ がこれらの機能をサポートしているかどうかを確認するには、「Amazon MQ で IAM が機能する仕組み」を参照してください。
- 所有 AWS アカウント している のリソースへのアクセスを提供する方法については、IAM ユーザーガイドの「所有 AWS アカウント している別の の IAM ユーザーへのアクセスを提供する」を参照してください。
- リソースへのアクセスをサードパーティーに提供する方法については AWS アカウント、IAM ユーザーガイドの「サードパーティー AWS アカウント が所有する へのアクセスを提供する」を参照してください。
- ID フェデレーションを介してアクセスを提供する方法については、「IAM ユーザーガイド」の「外部で認証されたユーザー (ID フェデレーション) へのアクセスの許可」を参照してください。
- クロスアカウントアクセスにおけるロールとリソースベースのポリシーの使用方法の違いについては、「IAM ユーザーガイド」の「IAM でのクロスアカウントのリソースへのアクセス」を参照してください。

Amazon MQ のコンプライアンス検証

サードパーティーの監査者は、複数のコンプライアンスプログラムの一環として Amazon MQ のセキュリティと AWS コンプライアンスを評価します。このプログラムには、SOC、PCI、HIPAA などを含みます。

AWS のサービス が特定のコンプライアンスプログラムの範囲内にあるかどうかを確認するには、AWS のサービス「コンプライアンスプログラムによる範囲内」を参照して、関心のあるコンプライアンスプログラムを選択します。一般的な情報については、AWS「コンプライアンスプログラム」を参照してください。

を使用して、サードパーティーの監査レポートをダウンロードできます AWS Artifact。詳細については、「Downloading Reports in AWS Artifact」を参照してください。

を使用する際のお客様のコンプライアンス責任 AWS のサービス は、お客様のデータの機密性、貴社のコンプライアンス目的、適用される法律および規制によって決まります。 は、コンプライアンスに役立つ以下のリソース AWS を提供します。

- セキュリティのコンプライアンスとガバナンス – これらのソリューション実装ガイドでは、アーキテクチャ上の考慮事項について説明し、セキュリティとコンプライアンスの機能をデプロイする手順を示します。
- HIPAA 対応サービスのリファレンス – HIPAA 対応サービスの一覧が提供されています。すべて AWS のサービス HIPAA の対象となるわけではありません。
- AWS コンプライアンスリソース – このワークブックとガイドのコレクションは、お客様の業界や地域に適用される場合があります。
- AWS カスタマーコンプライアンスガイド – コンプライアンスの観点から責任共有モデルを理解します。このガイドでは、複数のフレームワーク (米国国立標準技術研究所 (NIST)、Payment Card Industry Security Standards Council (PCI)、国際標準化機構 (ISO) など) にわたるセキュリティコントロールを保護し、そのガイダンスに AWS のサービス マッピングするためのベストプラクティスをまとめています。
- 「 デベロッパーガイド」の「ルールによるリソースの評価」 – この AWS Config サービスは、リソース設定が内部プラクティス、業界ガイドライン、規制にどの程度準拠しているかを評価します。 AWS Config
- AWS Security Hub – これにより AWS のサービス 、 内のセキュリティ状態を包括的に把握できます AWS。Security Hub では、セキュリティコントロールを使用して AWS リソースを評価し、セキュリティ業界標準とベストプラクティスに対するコンプライアンスをチェックします。サポートされているサービスとコントロールの一覧については、Security Hub のコントロールリファレンスを参照してください。
- Amazon GuardDuty – 環境をモニタリングして AWS アカウント、疑わしいアクティビティや悪意のあるアクティビティがないか調べることで、、ワークロード、コンテナ、データに対する潜在的な脅威 AWS のサービス を検出します。GuardDuty を使用すると、特定のコンプライアンスフレームワークで義務付けられている侵入検知要件を満たすことで、PCI DSS などのさまざまなコンプライアンス要件に対応できます。
- AWS Audit Manager – これにより AWS のサービス 、 AWS 使用状況を継続的に監査し、リスクの管理方法と規制や業界標準への準拠を簡素化できます。

Amazon MQ の耐障害性

AWS グローバルインフラストラクチャは、 AWS リージョンとアベイラビリティーゾーンを中心に構築されています。 AWS リージョンは、低レイテンシー、高スループット、および冗長性の高いネットワークで接続された、物理的に分離された複数のアベイラビリティーゾーンを提供します。アベイラビリティーゾーンでは、ゾーン間で中断することなく自動的にフェイルオーバーするアプリケーションとデータベースを設計および運用することができます。アベイラビリティーゾーンは、従

来の単一または複数のデータセンターインフラストラクチャよりも可用性が高く、フォールトトレラントで、スケーラブルです。

AWS リージョンとアベイラビリティーゾーンの詳細については、AWS 「 グローバルインフラストラクチャ」を参照してください。

Amazon MQ のインフラストラクチャセキュリティ

マネージドサービスとして、 は AWS グローバルネットワークセキュリティで保護されています。AWS セキュリティサービスと がインフラストラクチャ AWS を保護する方法については、AWS 「 クラウドセキュリティ」を参照してください。インフラストラクチャセキュリティのベストプラクティスを使用して AWS 環境を設計するには、「Security Pillar AWS Well-Architected Framework」の「Infrastructure Protection」を参照してください。

AWS 公開された API コールを使用して、ネットワーク経由で にアクセスします。クライアントは以下をサポートする必要があります。

- Transport Layer Security (TLS)。TLS 1.2 が必須で、TLS 1.3 をお勧めします。
- DHE (楕円ディフィー・ヘルマン鍵共有) や ECDHE (楕円曲線ディフィー・ヘルマン鍵共有) などの完全前方秘匿性 (PFS) による暗号スイート。これらのモードはJava 7 以降など、ほとんどの最新システムでサポートされています。

また、リクエストにはアクセスキー ID と、IAM プリンシパルに関連付けられているシークレットアクセスキーを使用して署名する必要があります。または、AWS Security Token Service (AWS STS) を使用して、一時的なセキュリティ認証情報を生成し、リクエストに署名することもできます。

Amazon MQ のセキュリティベストプラクティス

以下の設計パターンは、Amazon MQ ブローカーのセキュリティを向上させることができます。

トピック

- パブリックアクセスビリティのないブローカーを優先する
- 認可マップを常に設定する
- VPC セキュリティグループを使用して不要なプロトコルをブロックする

Amazon MQ がデータを暗号化する方法、およびサポートされるプロトコルのリストの詳細については、「データ保護」を参照してください。

パブリックアクセスビリティのないブローカーを優先する

パブリックアクセシビリティなしで作成されたブローカーには、VPC 外からアクセスできません。これにより、ブローカーがパブリックインターネットからの分散サービス妨害 (DDoS) 攻撃を受ける可能性が大幅に低減されます。詳細については、 AWS セキュリティブログの「攻撃領域を減らして DDoS 攻撃に備える方法」を参照してください。

認可マップを常に設定する

デフォルトでは、ActiveMQ には承認された認可マップがないため、認証されたすべてのユーザーが、ブローカーであらゆるアクションを実行することができます。したがって、グループごとにアクセス許可を制限することがベストプラクティスとなります。詳細については、「`authorizationEntry`」を参照してください。

> **Important**
>
> `activemq-webconsole` グループが含まれない認可マップを指定する場合、Amazon MQ ブローカーにメッセージを送信する権限、またはブローカーからメッセージを受信する権限がグループにないことから、ActiveMQ ウェブコンソールは使用できません。

VPC セキュリティグループを使用して不要なプロトコルをブロックする

プライベートブローカーのセキュリティを向上させるには、Amazon VPC セキュリティグループを適切に設定して、不要なプロトコルとポートの接続を制限する必要があります。例えば、OpenWire および ウェブコンソールへのアクセスを許可する一方で、ほとんどのプロトコルへのアクセスを制限するには、61617 および 8162 へのアクセスのみを許可することができます。これは、OpenWire とウェブコンソールが正常に機能することを可能にしながら、使用していないプロトコルをブロックすることによって、露出を制限します。

使用しているプロトコルポートのみを許可します。

- AMQP: 5671
- MQTT: 8883
- OpenWire: 61617
- STOMP: 61614
- WebSocket: 61619

詳細については、以下を参照してください。

- VPC のセキュリティグループ
- VPC のデフォルトセキュリティグループ
- セキュリティグループを操作する

Amazon MQ ブローカーのロギングとモニタリング

モニタリングは、 AWS ソリューションの信頼性、可用性、パフォーマンスを維持する上で重要な部分です。マルチポイント障害が発生した場合は、その障害をより簡単にデバッグできるように、 AWS ソリューションのすべての部分からモニタリングデータを収集する必要があります。 には、Amazon MQ リソースをモニタリングし、潜在的なインシデントに対応するための複数のツール AWS が用意されています。

CloudWatch を使用して、Amazon MQ ブローカーのメトリクスを表示および分析できます。CloudWatch コンソール、、 AWS CLIまたは CloudWatch AWS CLIからブローカーメトリクスを表示および分析できます。Amazon MQ 向けの CloudWatch メトリクスは、1 分おきにブローカーから自動的にポーリングされ、その後 CloudWatch にプッシュされます。ActiveMQ ブローカーの場合、CloudWatch は最初の 1000 個の送信先のみをモニタリングします。RabbitMQ ブローカーの場合、CloudWatch はコンシューマーの数順に並べられた最初 500 個の送信先のみをモニタリングします。

Amazon MQ メトリクスの完全なリストについては、「Amazon MQ for ActiveMQ ブローカーで利用可能な CloudWatch メトリクス」を参照してください。

メトリクスに対する CloudWatch アラームの作成については、Amazon CloudWatch ユーザーガイドで Amazon CloudWatch アラームの作成と編集を参照してください。

Amazon MQ 向けの CloudWatch メトリクスへのアクセス

CloudWatch メトリクスには AWS Management Console、、 AWS CLI、および API を使用してアクセスできます。

を使用せずに CloudWatch メトリクスにアクセスすることもできます AWS Management Console。

を使用して Amazon MQ メトリクスにアクセスするには AWS CLI、`get-metric-statistics` コマンドを使用します。詳細については、Amazon CloudWatch ユーザーガイドの「メトリクスの統計の取得」を参照してください。

CloudWatch API を使用して Amazon MQ メトリクスにアクセスするには、`GetMetricStatistics` アクションを使用します。詳細については、Amazon CloudWatch ユーザーガイドの「メトリクスの統計の取得」を参照してください。

を使用した CloudWatch メトリクスの取得 AWS Management Console

次の例は、 AWS Management Consoleを使用して Amazon MQ の CloudWatch メトリクスにアクセスする方法を示しています。Amazon MQ コンソールに既にサインインしている場合は、ブローカーの [詳細] ページで、[アクション]、[CloudWatch メトリクスの表示] の順に選択します。

1. CloudWatchコンソールにサインインします。
2. ナビゲーションパネルで [Metrics] を選択します。
3. [AmazonMQ] メトリクスの名前空間を選択します。
4. 次のいずれかのメトリクスディメンションを選択します。

   - ブローカーのメトリクス
   - ブローカー別のキューメトリクス
   - ブローカー別のトピックメトリクス

   この例では、[ブローカーのメトリクス] が選択されています。
5. これで、Amazon MQ メトリクスを調べることができるようになりました。

   - メトリクスを並べ替えるには、列見出しを使用します。
   - メトリクスをグラフ表示するには、メトリクスの横にあるチェックボックスを選択します。
   - メトリクスでフィルタするには、メトリクスの名前を選択し、[Add to search] を選択します。

Amazon MQ for ActiveMQ ブローカーで利用可能な CloudWatch メトリクス

Amazon MQ for ActiveMQ メトリクス

メトリクス	単位	説明
`AmqpMaximumConnections`	カウント	AMQP を使用してブローカーに接続できるクライアントの最大数。接続クォータの詳細については、「Quotas in

メトリクス	単位	説明
		Amazon MQ」を参照してください。
BurstBalance	割合 (%)	スループット最適化ブローカーのメッセージデータを永続化するために使用される Amazon EBS ボリュームに残っているバーストクレジットの割合 (%)。この残量がゼロになると、バーストバランスが補充されるまで、Amazon EBS ボリューム提供の IOPS が減少します。Amazon EBS でのバーストバランスの仕組みに関する詳細については、「I/O クレジットおよびバーストパフォーマンス」を参照してください。

メトリクス	単位	説明
CpuCreditBalance	クレジット (vCPU 分)	**Important** このメトリクスは、mq.t2.micro ブローカーインスタンスタイプでのみ使用できます。 CPU クレジットメトリクスは、5 分間隔でのみ利用可能です。 インスタンスが起動または開始後に蓄積した獲得 CPU クレジットの数 (起動クレジットの数を含む)。クレジット残高は、ブローカーインスタンスがそのベースライン CPU 使用率を超えてバーストするために消費できます。 クレジットは、獲得後にクレジット残高に蓄積され、消費後にクレジット残高から削除されます。クレジット残高には上限があります。制限に到達すると、新しく獲得されたクレジットはすべて破棄されます。
CpuUtilization	割合 (%)	割り当てられた Amazon EC2 コンピュートユニット (ECU) のうち、現在ブローカーが使用しているユニットの割合。

メトリクス	単位	説明
CurrentConnectionsCount	カウント	現在のブローカーでのアクティブな接続の現在の数。
EstablishedConnectionsCount	カウント	ブローカーで確立された、アクティブと非アクティブな接続の合計数。
HeapUsage	割合 (%)	ブローカーが現在使用している ActiveMQ JVM メモリ制限の割合。
InactiveDurableTopicSubscribersCount	カウント	非アクティブな永続トピックサブスクライバーの数 (最大 2000)。
JobSchedulerStorePercentUsage	割合 (%)	ジョブスケジューラストアで使用するディスク領域の割合 (%)。
JournalFilesForFastRecovery	カウント	クリーンシャットダウン後に再生されるジャーナルファイルの数。
JournalFilesForFullRecovery	カウント	クリーンでないシャットダウン後に再生されるジャーナルファイルの数。
MqttMaximumConnections	カウント	MQTT を使用してブローカーに接続できるクライアントの最大数。接続クォータの詳細については、「Quotas in Amazon MQ」を参照してください。

メトリクス	単位	説明
NetworkConnectorConnectionCount	カウント	NetworkConnector を使用してブローカーのネットワーク内のブローカーに接続されているノードの数。
NetworkIn	バイト	ブローカーの受信トラフィックのボリューム。
NetworkOut	バイト	ブローカーの送信トラフィックのボリューム。
OpenTransactionCount	カウント	進行中のトランザクションの総数。
OpenwireMaximumConnections	カウント	OpenWire を使用してブローカーに接続できるクライアントの最大数。接続クォータの詳細については、「Quotas in Amazon MQ」を参照してください。
StompMaximumConnections	カウント	STOMP を使用してブローカーに接続できるクライアントの最大数。接続クォータの詳細については、「Quotas in Amazon MQ」を参照してください。
StorePercentUsage	割合 (%)	ストレージ制限によって使用されている割合。これが 100 に達すると、ブローカーはメッセージを拒否します。
TempPercentUsage	割合 (%)	非永続的メッセージで使用可能な一時ストレージの割合 (%)。

メトリクス	単位	説明
TotalConsumerCount	カウント	現在のブローカーの送信先にサブスクライブされたメッセージコンシューマーの数。
TotalMessageCount	カウント	ブローカーに保存されたメッセージの数。
TotalProducerCount	カウント	現在のブローカーの送信先でのアクティブなメッセージプロデューサーの数。
VolumeReadOps	カウント	Amazon EBS ボリュームで実行された読み取り操作の数。
VolumeWriteOps	カウント	Amazon EBS ボリュームで実行された書き込み操作の数。
WsMaximumConnections	カウント	WebSocket を使用してブローカーに接続できるクライアントの最大数。接続クォータの詳細については、「Quotas in Amazon MQ」を参照してください。

ActiveMQ ブローカーメトリクスのディメンション

ディメンション	説明
Broker	ブローカーの名前 **Note** 単一インスタンスブローカーにはサフィックス -1 が付いています。高可用性対応のアクティブ/スタンバイブロー

ディメンション	説明
	カーには、その冗長ペアにサフィックス -1 と -2 が付いています。

ActiveMQ の送信先 (キューとトピック) メトリクス

> **Important**
>
> 以下のメトリクスには、CloudWatch のポーリング期間中の 1 分あたりの数が含まれます。
>
> - `EnqueueCount`
> - `ExpiredCount`
> - `DequeueCount`
> - `DispatchCount`
> - `InFlightCount`
>
> 例えば、5 分間の CloudWatch 期間では、`EnqueueCount` に 5 つの計数値があり、それぞれがその期間の 1 分間に対応します。`Maximum` および `Minimum` 統計は、指定した期間内の 1 分あたりの最小値と最大値を提供します。

メトリクス	単位	説明
`ConsumerCount`	カウント	送信先にサブスクライブされる消費者の数。
`EnqueueCount`	カウント	送信先に送信されるメッセージの数 (1 分あたり)。
`EnqueueTime`	時間 (ミリ秒)	メッセージがブローカーに届いてからコンシューマーに配信されるまでの、エンドツーエンドのレイテンシー。

メトリクス	単位	説明
		Note EnqueueTime は、プロデューサーがメッセージを送信してから、それがブローカーに到達するまでのエンドツーエンドレイテンシーを測定しません。また、ブローカーがメッセージを受信してから、ブローカーがそれを承認するまでのレイテンシーも測定しません。EnqueueTime は、ブローカーがメッセージを受信した瞬間から、コンシューマーに正常に配信されるまでのミリ秒数です。
ExpiredCount	カウント	期限切れのために配信できなかったメッセージの数 (1 分あたり)。
DispatchCount	カウント	コンシューマーに送信されたメッセージの数 (1 分あたり)。
DequeueCount	カウント	コンシューマーによって確認されたメッセージの数 (1 分あたり)。

メトリクス	単位	説明
InFlightCount	カウント	確認されていないコンシューマーに送信されたメッセージの数。
ReceiveCount	カウント	二重ネットワークコネクターに対してリモートブローカーから受信したメッセージの数。
MemoryUsage	割合 (%)	送信先が現在使用しているメモリ制限の割合。
ProducerCount	カウント	宛先のプロデューサーの数。
QueueSize	カウント	キュー内のメッセージの数。 **Important** このメトリクスは、キューにのみ適用されます。
TotalEnqueueCount	カウント	ブローカーに送信されたメッセージの合計数。
TotalDequeueCount	カウント	クライアントによって消費されたメッセージの合計数。

Note

TotalEnqueueCount および TotalDequeueCount メトリクスには、アドバイザリトピックのメッセージが含まれます。アドバイザリトピックメッセージの詳細については、ActiveMQ のドキュメントを参照してください。

ActiveMQ の送信先 (キューとトピック) メトリクスのディメンション

ディメンション	説明
Broker	ブローカーの名前。 **Note** 単一インスタンスブローカーにはサフィックス -1 があります。高可用性対応アクティブ/スタンバイブローカーには、冗長なペアに対してサフィックス -1 および -2 があります。
Topic、または Queue	トピックまたはキューの名前。
NetworkConnector	ネットワークコネクタの名前。

Amazon MQ for RabbitMQ ブローカーで利用可能な CloudWatch メトリクス

RabbitMQ ブローカーメトリクス

メトリクス	単位	説明
ExchangeCount	カウント	ブローカーで設定されたエクスチェンジの合計数。
QueueCount	カウント	ブローカーで設定されたキューの合計数。
ConnectionCount	カウント	ブローカーで確立された接続の合計数。
ChannelCount	カウント	ブローカーで確立されたチャネルの合計数。

メトリクス	単位	説明
ConsumerCount	カウント	ブローカーに接続されたコンシューマーの合計数。
MessageCount	カウント	キュー内のメッセージの合計数。 **Note** 生成される数値は、ブローカー上にある準備完了および未承認のメッセージの合計数です。
MessageReadyCount	カウント	キュー内の準備完了メッセージの合計数。
MessageUnacknowledgedCount	カウント	キュー内の未承認メッセージの合計数。
PublishRate	カウント	メッセージがブローカーに発行される速度。 生成される数値は、サンプリング時における 1 秒あたりのメッセージ数を表します。

メトリクス	単位	説明
ConfirmRate	カウント	RabbitMQ サーバーが発行されたメッセージを確認する速度。このメトリクスを PublishRate を比較して、ブローカーのパフォーマンスをより良く理解することができます。 生成される数値は、サンプリング時における 1 秒あたりのメッセージ数を表します。
AckRate	カウント	メッセージがコンシューマーによって承認される速度。 生成される数値は、サンプリング時における 1 秒あたりのメッセージ数を表します。
SystemCpuUtilization	割合 (%)	割り当てられた Amazon EC2 コンピュートユニット (ECU) のうち、現在ブローカーが使用しているユニットの割合。クラスターデプロイの場合、この値は 3 つの各 RabbitMQ ノードに対応するメトリクス値の集計を表します。
RabbitMQMemLimit	バイト	RabbitMQ ブローカーに対する RAM 制限。クラスターデプロイの場合、この値は 3 つの各 RabbitMQ ノードに対応するメトリクス値の集計を表します。

メトリクス	単位	説明
RabbitMQMemUsed	バイト	RabbitMQ ブローカーによって使用される RAM の量。クラスターデプロイの場合、この値は 3 つの各 RabbitMQ ノードに対応するメトリクス値の集計を表します。
RabbitMQDiskFreeLimit	バイト	RabbitMQ ブローカーに対するディスク制限。クラスターデプロイの場合、この値は 3 つの各 RabbitMQ ノードに対応するメトリクス値の集計を表します。このメトリクスは、インスタンスサイズごとに異なります。
RabbitMQDiskFree	バイト	RabbitMQ ブローカーで利用できる空きディスク領域の合計容量。ディスクの使用量が上限を超えると、クラスターはすべてのプロデューサー接続をブロックします。クラスターデプロイの場合、この値は 3 つの各 RabbitMQ ノードに対応するメトリクス値の集計を表します。
RabbitMQFdUsed	カウント	使用されたファイルディスクリプタの数。クラスターデプロイの場合、この値は 3 つの各 RabbitMQ ノードに対応するメトリクス値の集計を表します。

メトリクス	単位	説明
RabbitMQIOReadAverageTime	カウント	RabbitMQ が 1 回の読み込みオペレーションを実行する平均時間 (ミリ秒単位)。値はメッセージサイズに比例します。
RabbitMQIOWriteAverageTime	カウント	RabbitMQ が 1 回の書き込みオペレーションを実行する平均時間 (ミリ秒単位)。値はメッセージサイズに比例します。

RabbitMQ ブローカーメトリクスのディメンション

ディメンション	説明
Broker	ブローカーの名前。

RabbitMQ ノードメトリクス

メトリクス	単位	説明
SystemCpuUtilization	割合 (%)	割り当てられた Amazon EC2 コンピュートユニット (ECU) のうち、現在ブローカーが使用しているユニットの割合。
RabbitMQMemLimit	バイト	RabbitMQ ノードに対する RAM 制限。
RabbitMQMemUsed	バイト	RabbitMQ ノードによって使用される RAM の容量。メモリの使用量が制限を超えると、クラスターはすべてのプ

メトリクス	単位	説明
		ロデューサー接続をブロックします。
RabbitMQDiskFreeLimit	バイト	RabbitMQ ノードのディスク制限。このメトリクスは、インスタンスサイズごとに異なります。
RabbitMQDiskFree	バイト	RabbitMQ ノードで利用できる空きディスク領域の合計容量。ディスクの使用量が上限を超えると、クラスターはすべてのプロデューサー接続をブロックします。
RabbitMQFdUsed	カウント	使用されたファイルディスクリプタの数。

RabbitMQ ノードメトリクスのディメンション

ディメンション	説明
Node	ノードの名前。 **Note** ノード名は、プレフィックス (通常 rabbit) とホスト名の 2 つの部分で構成されます。例えば、rabbit@ip-10-0-0-230.us-west-2.compute.internal はプレフィックス rabbit とホスト名 ip-10-0-0-230.us-west-2.com

ディメンション	説明
	pute.internal を持つノード名です。
Broker	ブローカーの名前。

RabbitMQ キューメトリクス

メトリクス	単位	説明
ConsumerCount	カウント	キューにサブスクライブしているコンシューマーの数。
MessageReadyCount	カウント	現在配信可能なメッセージの数。
MessageUnacknowledgedCount	カウント	サーバーが承認を待機しているメッセージの数。
MessageCount	カウント	MessageReadyCount と MessageUnacknowledgedCount の合計数 (キュー深度とも呼ばれます)。

RabbitMQ キューメトリクスのディメンション

> **Note**
>
> Amazon MQ for RabbitMQ は、空白、タブ、またはその他の非 ASCII 文字が含まれた名前を持つ仮想ホストおよびキューのメトリクスを発行しません。
> ディメンション名の詳細については、Amazon CloudWatch API リファレンスの「Dimension」を参照してください。

ディメンション	説明
Queue	キューの名前。
VirtualHost	仮想ホストの名前。
Broker	ブローカーの名前。

Amazon MQ for RabbitMQ ログの設定

RabbitMQ ブローカーに対して CloudWatch ロギングを有効にすると、Amazon MQ はサービスリンクロールを使用して CloudWatch に一般ログを発行します。ブローカーを初めて作成するときに Amazon MQ サービスリンクロールが存在しない場合、Amazon MQ がそのロールを自動的に作成します。すべての後続 RabbitMQ ブローカーは、同じサービスリンクロールを使用して CloudWatch にログを発行します。

サービスリンクロールの詳細については、「AWS Identity and Access Management ユーザーガイド」の「サービスリンクロールの使用」を参照してください。Amazon MQ がサービスリンクロールを使用する方法の詳細については、「the section called “サービスリンクロールの使用”」を参照してください。

AWS CloudTrailを使用した Amazon MQ API コールのロギング

Amazon MQ は、ユーザー AWS CloudTrail、ロール、またはサービスが行う Amazon MQ 呼び出しの記録を提供する AWS サービスである と統合されています。CloudTrail は、Amazon MQ ブローカーと設定に関連する API コールをイベントとしてキャプチャします。これには Amazon MQ コンソールからのコールと Amazon MQ API からのコードコールが含まれます。CloudTrail の詳細については、「AWS CloudTrail ユーザーガイド」を参照してください。

> **Note**
>
> CloudTrail は、ActiveMQ 操作 (メッセージの送受信など) や ActiveMQ ウェブコンソールに関連する API コールをログしません。ActiveMQ 操作に関連する情報をログするには、一般ログと監査ログを Amazon CloudWatch Logs に発行するように Amazon MQ を設定することができます。

CloudTrail が収集する情報を使用して、Amazon MQ API に対する特定のリクエスト、リクエスタの IP アドレス、リクエスタのアイデンティティ、およびリクエストの日時などを特定することができます。追跡を設定する場合は、Amazon S3 バケットへの CloudTrail イベントの継続的な配信を有効にすることができます 追跡を設定しない場合でも、CloudTrailコンソールのイベント履歴で最近のイベントを表示できます。詳細については、AWS CloudTrail ユーザーガイドの「Overview for Creating a Trail」を参照してください。

CloudTrail 内の Amazon MQ 情報

AWS アカウントを作成すると、CloudTrail が有効になります。サポートされている Amazon MQ イベントアクティビティが発生すると、イベント履歴内の他の AWS サービスイベントとともに CloudTrail イベントに記録されます。 AWS アカウントの最近のイベントを、表示、検索、およびダウンロードすることができます。詳細については、AWS CloudTrail ユーザーガイドの「CloudTrail イベント履歴でのイベントの表示」を参照してください。

追跡は、CloudTrailがログファイルを Amazon S3バケットに配信できるようにします。証跡を作成して、 AWS アカウントのイベントの継続的な記録を保持できます。デフォルトでは、 を使用して証跡を作成すると AWS Management Console、証跡はすべての AWS リージョンに適用されます。証跡は、すべての AWS リージョンからのイベントをログに記録し、ログファイルを指定された Amazon S3 バケットに配信します。CloudTrail ログで収集されたイベントデータをさらに分析して処理するように、他の AWS サービスを設定することもできます。詳細については、AWS CloudTrail ユーザーガイドの次のトピックを参照してください。

- CloudTrail がサポートするサービスと統合
- CloudTrail の Amazon SNS 通知の設定
- CloudTrail ログファイルの複数のリージョンからの受け取り
- 複数のアカウントから CloudTrailログファイルを受け取る

Amazon MQ は、以下の API のリクエストパラメータとレスポンスの両方を、イベントとして CloudTrail ログファイルにログすることをサポートします。

- `CreateConfiguration`
- `DeleteBroker`
- `DeleteUser`
- `RebootBroker`
- `UpdateBroker`

> **Note**
>
> RebootBroker ログファイルは、ブローカーを再起動したときに記録されます。メンテナンス期間中、サービスは自動的に再起動し、RebootBroker ログファイルは記録されません。

> **Important**
>
> 以下 API の GET メソッドの場合、リクエストパラメータはログ記録されますが、レスポンスは加工されます。
>
> - DescribeBroker
> - DescribeConfiguration
> - DescribeConfigurationRevision
> - DescribeUser
> - ListBrokers
> - ListConfigurationRevisions
> - ListConfigurations
> - ListUsers
>
> 以下の API では、data と password のリクエストパラメータはアスタリスク (***) によって非表示になります。
>
> - CreateBroker (POST)
> - CreateUser (POST)
> - UpdateConfiguration (PUT)
> - UpdateUser (PUT)

各イベントまたはログエントリには、リクエスタに関する情報が含まれます。この情報は以下のことを確認するのに役立ちます:

- リクエストが、ルートと ユーザー認証情報のどちらを使用して送信されたか。
- リクエストが、ロールとフェデレーティッドユーザーのどちらの一時的なセキュリティ認証情報を使用して送信されたか。

- リクエストは別の AWS サービスによって行われましたか？

詳細については、「AWS CloudTrail ユーザーガイド」の「CloudTrail userIdentity Element」を参照してください。

Amazon MQ ログファイルエントリの例

追跡は、イベントをログファイルとして指定された Amazon S3 バケットに配信することを可能にする設定です。CloudTrail のログファイルには、単一か複数のログエントリがあります。

イベントは、任意のソースからの単一のリクエストを表し、Amazon MQ API へのリクエスト、リクエスタの IP アドレス、リクエスタのアイデンティティ、およびリクエストの日時などに関する情報が含まれます。

以下の例は、`CreateBroker` API コールの CloudTrail ログエントリを示しています。

> **Note**
>
> CloudTrail ログファイルはパブリック API の順序付けられたスタックトレースではないため、特定の順序で情報が表示されることはありません。


```
{
    "eventVersion": "1.06",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AKIAIOSFODNN7EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/AmazonMqConsole",
        "accountId": "111122223333",
        "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
        "userName": "AmazonMqConsole"
    },
    "eventTime": "2018-06-28T22:23:46Z",
    "eventSource": "amazonmq.amazonaws.com",
    "eventName": "CreateBroker",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "203.0.113.0",
    "userAgent": "PostmanRuntime/7.1.5",
    "requestParameters": {
        "engineVersion": "5.15.9",
        "deploymentMode": "ACTIVE_STANDBY_MULTI_AZ",
```

```
        "maintenanceWindowStartTime": {
            "dayOfWeek": "THURSDAY",
            "timeOfDay": "22:45",
            "timeZone": "America/Los_Angeles"
        },
        "engineType": "ActiveMQ",
        "hostInstanceType": "mq.m5.large",
        "users": [
            {
                "username": "MyUsername123",
                "password": "***",
                "consoleAccess": true,
                "groups": [
                    "admins",
                    "support"
                ]
            },
            {
                "username": "MyUsername456",
                "password": "***",
                "groups": [
                    "admins"
                ]
            }
        ],
        "creatorRequestId": "1",
        "publiclyAccessible": true,
        "securityGroups": [
            "sg-a1b234cd"
        ],
        "brokerName": "MyBroker",
        "autoMinorVersionUpgrade": false,
        "subnetIds": [
            "subnet-12a3b45c",
            "subnet-67d8e90f"
        ]
    },
    "responseElements": {
        "brokerId": "b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9",
        "brokerArn": "arn:aws:mq:us-
east-2:123456789012:broker:MyBroker:b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9"
    },
    "requestID": "a1b2c345-6d78-90e1-f2g3-4hi56jk7l890",
    "eventID": "a12bcd3e-fg45-67h8-ij90-12k34d5l16mn",
```

```
    "readOnly": false,
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

Amazon MQ for ActiveMQ ログの設定

CloudWatch Logs へのログの発行を Amazon MQ に許可するには、ブローカーを作成および再起動する前に、Amazon MQ ユーザーに許可を追加するとともに、Amazon MQ のリソースベースポリシーも設定する必要があります。

> **Note**
>
> ActiveMQ ウェブコンソールからログを有効にしてメッセージを発行すると、メッセージのコンテンツが CloudWatch に送信され、ログに表示されます。

以下は、ActiveMQ ブローカー用の CloudWatch Logs を設定するステップの説明です。

トピック

- CloudWatch Logs でのロギングの構造を理解する
- Amazon MQ ユーザーへの CreateLogGroup 許可の追加
- Amazon MQ のリソースベースポリシーを設定する
- サービス間での不分別な代理処理の防止

CloudWatch Logs でのロギングの構造を理解する

ブローカーの作成時、またはブローカーの編集時に高度なブローカー設定を構成するときに、一般ログ記録と監査ログ記録を有効にできます。

一般ロギングは、デフォルトの `INFO` ロギングレベルを有効にし (`DEBUG` ロギングはサポートされません)、`activemq.log` を CloudWatch アカウントのロググループに発行します。ロググループの形式は次のようになります。

```
/aws/amazonmq/broker/b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9/general
```

監査ロギングは、JMX または ActiveMQ ウェブコンソールを使用して行われた管理アクションのロギングを有効にし、audit.log を CloudWatch アカウントのロググループに発行します。ロググループの形式は次のようになります。

```
/aws/amazonmq/broker/b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9/audit
```

Amazon MQ は、単一インスタンスブローカーかアクティブ/スタンバイブローカーのどちらを使用しているかに応じて、各ロググループ内に 1 つまたは 2 つのログストリームを作成します。ログストリームの形式は次のようになります。

```
activemq-b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.log
                activemq-b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-2.log
```

サフィックスが -1 および -2 の場合は、個々のブローカーインスタンスを示します。詳細については、Amazon CloudWatch Logs ユーザーガイドの「ロググループとログストリームの操作」を参照してください。

Amazon MQ ユーザーへの **CreateLogGroup** 許可の追加

CloudWatch Logs ロググループの作成を Amazon MQ に許可するには、ブローカーを作成または再起動するユーザーに logs:CreateLogGroup アクセス許可があることを確認する必要があります。

> **Important**
>
> ユーザーがブローカーの作成または再起動を行う前に CreateLogGroup 許可をユーザーに追加しなければ、Amazon MQ はロググループを作成しません。

以下のサンプル IAM ベースポリシーは、このポリシーがアタッチされているユーザーの logs:CreateLogGroup に対する許可を付与します。

```
{
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "logs:CreateLogGroup",
                            "Resource": "arn:aws:logs:*:*:log-group:/aws/amazonmq/*"
```

```
            }
        ]
        }
```

> **Note**
>
> ここで、ユーザーという用語は、新しいブローカーの設定時に作成した Amazon MQ ユーザーではなく、ユーザーを指しています。ユーザーのセットアップと IAM ポリシーの設定の詳細については、IAM ユーザーガイドの「ID 管理の概要」セクションを参照してください。

詳細については、Amazon CloudWatch Logs API リファレンスの「CreateLogGroup」を参照してください。

Amazon MQ のリソースベースポリシーを設定する

> **Important**
>
> Amazon MQ にリソースベースポリシーを設定しない場合、ブローカーは CloudWatch Logs にログを発行できません。

CloudWatch Logs ロググループへのログの発行を Amazon MQ に許可するには、以下の CloudWatch Logs API アクションに対するアクセス権を Amazon MQ に付与するリソースベースポリシーを設定します。

- CreateLogStream – 指定したロググループの CloudWatch Logs ログストリームを作成します。
- PutLogEvents – 指定された CloudWatch Logs ログストリームにイベントを配信します。

次のリソースベースのポリシーは、 `logs:CreateLogStream`および へのアクセス許可を付与`logs:PutLogEvents`します AWS。

```
{
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": { "Service": "mq.amazonaws.com" },
```

```
                                    "Action": [ "logs:CreateLogStream",
 "logs:PutLogEvents" ],
                                    "Resource": "arn:aws:logs:*:*:log-group:/aws/
amazonmq/*"
                                }
                            ]
                        }
```

このリソースベースのポリシーは、次のコマンド AWS CLI に示すように、 を使用して設定する必要があります。この例では、*us-east-1* を独自の情報に置き換えます。

```
aws --region us-east-1 logs put-resource-policy --policy-name AmazonMQ-logs \
                        --policy-document "{\"Version\": \"2012-10-17\", \"Statement\":
[{ \"Effect\": \"Allow\", \"Principal\": { \"Service\": \"mq.amazonaws.com\" },
                        \"Action\": [\"logs:CreateLogStream\", \"logs:PutLogEvents\"],
 \"Resource\": \"arn:aws:logs:*:*:log-group:\/aws\/amazonmq\/*\" }]}"
```

> **Note**
>
> この例では /aws/amazonmq/ プレフィックスを使用するため、リソースベースのポリシーは AWS アカウントごと、リージョンごとに 1 回のみ設定する必要があります。

サービス間での不分別な代理処理の防止

混乱した代理問題は、アクションを実行するためのアクセス許可を持たないエンティティが、より特権のあるエンティティにアクションの実行を強制できてしまう場合に生じる、セキュリティ上の問題です。では AWS、サービス間のなりすましにより、混乱した代理問題が発生する可能性があります。サービス間でのなりすましは、1 つのサービス (呼び出し元サービス) が、別のサービス (呼び出し対象サービス) を呼び出すときに発生する可能性があります。呼び出し元サービスは、本来ならアクセスすることが許可されるべきではない方法でその許可を使用して、別のお客様のリソースに対する処理を実行するように操作される場合があります。これを防ぐために、 は、アカウント内のリソースへのアクセス権が付与されたサービスプリンシパルを持つすべてのサービスのデータを保護するのに役立つツール AWS を提供します。

CloudWatch Logs アクセスを指定された 1 つまたは複数のブローカーに制限するには、Amazon MQ のリソースベースポリシーで aws:SourceArn および aws:SourceAccount のグローバル条件コンテキストキーを使用することをお勧めします。

> **Note**
>
> 両方のグローバル条件コンテキストキーを同じポリシーステートメントで使用する場合は、aws:SourceAccount 値と、aws:SourceArn 値に含まれるアカウントが、同じアカウント ID を示している必要があります。

次の例は、CloudWatch Logs アクセスを単一の Amazon MQ ブローカーに制限するリソースベースポリシーを示しています。

```
{
                        "Version": "2012-10-17",
                        "Statement": [
                            {
                            "Effect": "Allow",
                            "Principal": {
                                "Service": "mq.amazonaws.com"
                            },
                            "Action": [
                                "logs:CreateLogStream",
                                "logs:PutLogEvents"
                            ],
                            "Resource": "arn:aws:logs:*:*:log-group:/aws/amazonmq/*",
                            "Condition": {
                                "StringEquals": {
                                "aws:SourceAccount": "123456789012",
                                "aws:SourceArn": "arn:aws:mq:us-
east-2:123456789012:broker:MyBroker:b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9"
                                }
                            }
                            }
                        ]
                        }
```

以下に示すように、CloudWatch Logs アクセスをアカウント内のすべてのブローカーに制限するように、リソースベースポリシーを設定することもできます。

```
{
                            "Version": "2012-10-17",
                            "Statement": [
                            {
```

```
                                "Effect": "Allow",
                                "Principal": {
                                "Service": [
                                    "mq.amazonaws.com"
                                ]
                                },
                                "Action": [
                                "logs:CreateLogStream",
                                "logs:PutLogEvents"
                                ],
                                "Resource": "arn:aws:logs:*:*:log-group:/aws/amazonmq/
*",
                                "Condition": {
                                "ArnLike": {
                                    "aws:SourceArn":
 "arn:aws:mq:*:123456789012:broker:*"
                                },
                                "StringEquals": {
                                    "aws:SourceAccount": "123456789012"
                                }
                                }
                            }
                            ]
                        }
```

「混乱した代理」セキュリティ問題の詳細については、ユーザーガイドの「混乱した代理問題」を参照してください。

Amazon MQ での CloudWatch Logs 設定のトラブルシューティング

場合によっては、CloudWatch Logs が常に期待通りに動作しないことがあります。このセクションでは、一般的な問題の概要とそれらの解決方法を説明します。

ロググループが CloudWatch に表示されない

CreateLogGroup 許可を Amazon MQ ユーザーに追加して、ブローカーを再起動します。そうすることで、Amazon MQ がロググループを作成できるようになります。

ログストリームが CloudWatch ロググループに表示されない

Amazon MQ のリソースベースポリシーを設定します。これにより、ブローカーよりログを発行することができます。

Amazon MQ のクォータ

このトピックでは、Amazon MQ の制限の一覧を示します。以下の制限の多くは、特定の AWS アカウントで変更できます。制限緩和のリクエスト方法については、「Amazon Web Services 全般のリファレンス」の「AWS のサービスクォータ」を参照してください。上限の引き上げが適用された後でも、更新された上限は表示されません。Amazon CloudWatch での現在の接続上限の表示に関する詳細については、「Amazon CloudWatch を使用した Amazon MQ ブローカーのモニタリング」を参照してください。

トピック

ブローカー

以下の表は、Amazon MQ ブローカーに関連するクォータのリストです。

[制限]	説明
ブローカー名	• AWS アカウント内で一意である必要があります。 • 1 ~ 50 文字にする必要があります。 • 使用できるのは、印刷可能な ASCII 文字に指定された文字のみです。 • 使用できるのは、英数字、ダッシュ、ピリオド、アンダースコア、チルダ (- . _ ~) のみです。
リージョンあたりのブローカー数	50

[制限]	説明
小規模ブローカーのプロトコルあたりのワイヤレベルの接続	**Important** RabbitMQ ブローカーには適用されません。 `mq.*.micro` インスタンスタイプのブローカーに対して 300 個。
大規模ブローカーのプロトコルあたりのワイヤレベルの接続	**Important** RabbitMQ ブローカーには適用されません。 `mq.*.*large` インスタンスタイプのブローカーに対して 2,000 個。
ブローカーあたりのセキュリティグループ	5
CloudWatch でモニタリングされる ActiveMQ 送信先 (キューとトピック)	CloudWatch は、最初の 1000 個の送信先のみをモニタリングします。
CloudWatch でモニタリングされる RabbitMQ 送信先 (キュー)	CloudWatch は、コンシューマーの数順に並べられた最初 500 個の送信先のみをモニタリングします。
ブローカーあたりのタグ	50

設定

以下の表は、Amazon MQ の設定に関連するクォータのリストです。

[制限]	説明
設定名	• 1 ~ 150 文字にする必要があります。 • 使用できるのは、印刷可能な ASCII 文字に指定された文字のみです。 • 使用できるのは、英数字、ダッシュ、ピリオド、アンダースコア、チルダ (- . _ ~) のみです。
設定あたりのリビジョン	300

[ユーザー]

以下の表は、Amazon MQ ActiveMQ ブローカーのユーザーに関連するクォータのリストです。

[制限]	説明
ユーザーネーム	• 1 ~ 100 文字にする必要があります。 • 使用できるのは、印刷可能な ASCII 文字に指定された文字のみです。 • 使用できるのは、英数字、ダッシュ、ピリオド、アンダースコア、チルダ (- . _ ~) のみです。 • カンマ (,) を含めることはできません。
パスワード	• 12~250 文字にする必要があります。 • 使用できるのは、印刷可能な ASCII 文字に指定された文字のみです。 •

[制限]	説明
	少なくとも 4 個の一意文字を含める必要があります。 • カンマ (,) を含めることはできません。
ブローカーあたりのユーザー (simple auth)	250
ユーザーあたりのグループ (simple auth)	20

データストレージ

以下の表は、Amazon MQ のデータストレージに関連するクォータのリストです。

[制限]	説明
小規模なブローカーごとのストレージ容量	`mq.*.micro` インスタンスタイプのブローカーに対して 20 GB。Amazon MQ のインスタンスタイプの詳細については、「Broker instance types」を参照してください。
大規模なブローカーごとのストレージ容量	`mq.m5.*` インスタンスタイプのブローカーに対して 200 GB。Amazon MQ のインスタンスタイプの詳細については、「Broker instance types」を参照してください。
Amazon EBS によってバックアップされるブローカーごとのジョブスケジューラの使用制限	**Important** RabbitMQ ブローカーには適用されません。 50 GB。ジョブスケジューラの使用に関する詳細については、Apache ActiveMQ API ドキュメ

[制限]	説明
	ントの「JobSchedulerUsage 」を参照してください。
小規模なブローカーごとの一時的なストレージ容量	**Important** RabbitMQ ブローカーには適用されません。 mq.*.micro インスタンスタイプのブローカーに対して 5 GB。
大規模なブローカーごとの一時的なストレージ容量	**Important** RabbitMQ ブローカーには適用されません。 mq.m5.* インスタンスタイプのブローカーに対して 50 GB。

API スロットリング

以下のスロットリングクォータは、サービス帯域幅を維持するために、すべての Amazon MQ APIs で AWS アカウントごとに集計されます。Amazon MQ API の詳細については、Amazon MQ REST API リファレンスを参照してください。

Important

これらのクォータは、Amazon MQ for ActiveMQ または Amazon MQ for RabbitMQ のブローカーメッセージング API には適用されません。例えば、Amazon MQ はメッセージの送信または受信をスロットリングしません。

API バースト制限	API レート制限
100	15

Amazon MQ のトラブルシューティング

このセクションでは、Amazon MQ ブローカーの使用時に発生する可能性がある一般的な問題と、それらを解決するために実行できるステップについて説明します。一般的なトラブルシューティングについては、「」を参照してくださいthe section called “トラブルシューティング: 一般的な Amazon MQ”。特定のエンジンバージョンのトラブルシューティングについては、以下のセクションを参照してください。

Amazon MQ での ActiveMQ のトラブルシューティング Amazon MQ

トラブルシューティング情報	説明
一般的なトラブルシューティング	このセクションの情報は、Amazon MQ ブローカーで ActiveMQ を使用する際に発生する可能性がある一般的な問題の診断と解決に役立ちます。
BROKER_ENI_DELETED	Amazon MQ の ActiveMQ は、ブローカーの Elastic Network Interface (ENI) を削除するとBROKER_ENI_DELETED アラームを生成します。
BROKER_OOM	Amazon MQ の ActiveMQ は、メモリ容量不足のためにブローカーが再起動ループを経ると、BROKER_OOM アラームを生成します。

Amazon MQ での RabbitMQ のトラブルシューティング Amazon MQ

トラブルシューティング情報	説明
一般的なトラブルシューティング	RabbitMQ ブローカーを使用する際に発生する可能性が

トラブルシューティング情報	説明
	ある一般的な問題を診断します。
RABBITMQ_MEMORY_ALARM	RabbitMQ は、CloudWatch メトリクス によって識別されるブローカーのメモリ使用量が、 によって識別されるメモリ制限を超えると`RabbitMQMemUsed` 、高メモリアラームを生成します`RabbitMQMemLimit` 。
RABBITMQ_INVALID_KMS_KEY	Amazon MQ の RabbitMQ は、カスタマーマネージド AWS KMS key(CMK) で作成されたブローカーが AWS Key Management Service (KMS) キーが無効であることを検出したときに、INVALID_KMS_KEY の重要なアクション必須コードを生成します。
RABBITMQ_DISK_ALARM	ディスク制限アラームは、新しいメッセージが追加される一方で消費されないメッセージが多いため、RabbitMQ ノードが使用するディスク量が減少したことを示します。

トラブルシューティング情報	説明
RABBITMQ_QUORUM_QUEUES_NOT _SUPPORTED_ON_CURRENT_VERSION	Amazon MQ の RabbitMQ は、バージョン 3.12 以前を使用して単一のインスタンスまたはクラスターブローカーでクォーラムキューを作成しようとすると、RABBITMQ_QUORUM_QUEUES_NOT_SUPPORTED_ON_CURRENT_VERSION アラームを生成します。

トラブルシューティング: 一般的な Amazon MQ

このセクションの情報を使用して、ブローカーへの接続問題、またはブローカーの再起動などの、Amazon MQ ブローカーの使用時に発生する可能性がある一般的な問題の診断に役立てます。

目次

ブローカーのウェブコンソールまたはエンドポイントに接続できません。

ウェブコンソールまたはワイヤレベルのエンドポイントを使用したブローカーへの接続で問題が発生する場合は、以下の手順が推奨されます。

1. ファイアウォールの内側からブローカーに接続しようとしているかどうかをチェックします。ブローカーへのアクセスを許可するようにファイアウォールを設定する必要がある場合があります。

2. FIPS エンドポイントを使用して、ブローカーに接続しようとしているかどうかをチェックしてください。Amazon MQ では、API オペレーションを使用する場合のみ FIPS エンドポイントがサポートされ、ブローカーインスタンス自体へのワイヤレベルの接続はサポートされません。
3. ブローカーの [Public Accessibility] (パブリックアクセシビリティ) オプションが [Yes] (はい) に設定されているかどうかをチェックします。これが [No] (いいえ) に設定されている場合は、サブネットのネットワークアクセスコントロールリスト (ACL) ルールをチェックしてください。カスタムネットワーク ACL を作成した場合は、ブローカーへのアクセス権を提供するようにネットワーク ACL ルールを変更する必要がある場合があります。Amazon VPC ネットワークの詳細については、Amazon VPC ユーザーガイドの「インターネットアクセスを有効にする」を参照してください。
4. ブローカーのセキュリティグループルールをチェックします。以下のポートへの接続が許可されていることを確認してください。

> **Note**
>
> Amazon MQ の ActiveMQ と Amazon MQ の RabbitMQ Amazon MQ は接続に異なるポートを使用するため、次のポートはエンジンタイプに従ってグループ化されています。

Amazon MQ での ActiveMQ Amazon MQ

- ウェブコンソール – ポート 8162
- OpenWire – ポート 61617
- AMQP – ポート 5671
- STOMP — ポート 61614
- MQTT – ポート 8883
- WSS – ポート 61619

Amazon MQ の RabbitMQ Amazon MQ

- ウェブコンソールおよび Management API – ポート 443 および 15671
- AMQP – ポート 5671

5. ブローカーエンジンタイプに対して、以下のネットワーク接続テストを実行します。

> **Note**
>
> パブリックアクセシビリティがないブローカーの場合は、Amazon MQ ブローカーと同じ Amazon VPC 内の Amazon EC2 インスタンスからテストを実行して、レスポンスを評価してください。

ActiveMQ on Amazon MQ

Amazon MQ ブローカーのネットワーク接続で ActiveMQ をテストするには Amazon MQ

1. 新規のターミナルまたはコマンドラインウィンドウを開きます。
2. 以下の nslookup コマンドを実行して、ブローカー DNS レコードをクエリします。アクティブ/スタンバイデプロイの場合は、アクティブエンドポイントとスタンバイエンドポイントの両方をテストします。アクティブ/スタンバイエンドポイントは、一意のブローカー ID に追加された -1 または -2 サフィックスで特定されます。エンドポイントを独自の情報に置き換えます。

   ```
   $ nslookup b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-west-2.amazonaws.com
   ```

   クエリが正常に完了すると、以下のような出力が表示されます。

   ```
   Non-authoritative answer:
   Server:  dns-resolver-corp-sfo-1.sfo.corp.amazon.com
   Address:  172.10.123.456

   Name:    ec2-12-345-123-45.us-west-2.compute.amazonaws.com
   Address:  12.345.123.45
   Aliases:  b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-west-2.amazonaws.com
   ```

   解決された IP アドレスが、Amazon MQ コンソールで指定した IP アドレスと一致している必要があります。これは、ドメイン名が DNS サーバーで正しく解決されていることを示すので、次のステップに進むことができます。
3. 以下の telnet コマンドを実行して、ブローカーのネットワークパスをテストします。エンドポイントを独自の情報に置き換えます。必要に応じて、*port* をウェブコンソールのポート番号 8162、またはその他のワイヤレベルのポートに置き換えて、追加のプロトコルをテストします。

> **Note**
>
> アクティブ/スタンバイデプロイの場合、スタンバイエンドポイントで telnet を実行すると、Connect failed エラーメッセージが返されます。スタンバイインスタンス自体は実行されていますが、ActiveMQ プロセスは実行されておらず、ブローカーの Amazon EFS ストレージボリュームへのアクセス権がないため、これは期待どおりの動作です。アクティブインスタンスとスタンバイインスタンスの両方をテストできるように、-1 および -2 両方のエンドポイントにこのコマンドを実行してください。

```
$ telnet b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-
west-2.amazonaws.com port
```

アクティブインスタンスには、以下のような出力が表示されます。

```
Connected to b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-
west-2.amazonaws.com.
Escape character is '^]'.
```

4. 以下のいずれかを行ってください。

   - telnet コマンドが正常に完了する場合は、EstablishedConnectionsCount メトリクスをチェックして、ブローカーがワイヤレベル接続の上限に到達していないことを確認します。ブローカーの General ログを調べて、上限に到達したかどうかを確認することも可能です。このメトリクスがゼロより大きい場合は、現在少なくとも 1 つのクライアントがブローカーに接続されています。メトリクスがゼロ個の接続を示している場合は、telnet パステストを再度実行し、少なくとも 1 分待ってから接続を切断してください (ブローカーメトリクスは毎分発行されるため)。
   - telnet コマンドが失敗する場合は、ブローカーの Elastic Network Interface のステータスをチェックして、ステータスが in-use になっていることを確認します。各インスタンスのネットワークインターフェイスに関する Amazon VPC フローログを作成して、生成されたフローログを検証します。telnet コマンドを実行したときのブローカーの IP アドレスを調べて、応答パケットを含む接続パケットが ACCEPTED であることを確認します。フローログの詳細と例については、Amazon VPC デベロッパーガイドの「フローログレコードの例」を参照してください。

5. 以下の curl コマンドを実行して、ActiveMQ の管理ウェブコンソールへの接続をチェックします。

```
$ curl https://b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-
west-2.amazonaws.com:8162/index.html
```

コマンドが正常に完了すると、出力は以下のような HTML ドキュメントになります。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://
www.w3.org/TR/html4/loose.dtd">
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
 charset=iso-8859-1" />
        <title>Apache ActiveMQ</title>
        ...
```

RabbitMQ on Amazon MQ

Amazon MQ ブローカーのネットワーク接続で RabbitMQ をテストするには Amazon MQ

1. 新規のターミナルまたはコマンドラインウィンドウを開きます。
2. 以下の nslookup コマンドを実行して、ブローカー DNS レコードをクエリします。エンドポイントを独自の情報に置き換えます。

```
$ nslookup b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-west-2.amazonaws.com
```

クエリが正常に完了すると、以下のような出力が表示されます。

```
Non-authoritative answer:
Server:  dns-resolver-corp-sfo-1.sfo.corp.amazon.com
Address:  172.10.123.456

Name:    rabbit-broker-1c23e456ca78-b9000123b4ebbab5.elb.us-
west-2.amazonaws.com
Addresses:  52.12.345.678
          52.23.234.56
          41.234.567.890
          54.123.45.678
```

```
Aliases:  b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-west-2.amazonaws.com
```

3. 以下の telnet コマンドを実行して、ブローカーのネットワークパスをテストします。エンドポイントを独自の情報に置き換えます。*port* をウェブコンソールのポート 443 に置き換える、および 5671 に置き換えてワイヤレベルの AMQP 接続をテストすることができます。

```
$ telnet b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-
west-2.amazonaws.com port
```

コマンドが正常に完了する場合は、以下のような出力が表示されます。

```
Connected to b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-
west-2.amazonaws.com.
Escape character is '^]'.
```

> **Note**
>
> Telnet 接続は、数秒後に自動的に終了します。

4. 以下のいずれかを行ってください。

- telnet コマンドが正常に完了する場合は、ConnectionCount メトリクスをチェックして、max-connections デフォルトポリシーで設定されている値にブローカーが到達していないことを確認します。ブローカーの Connection.log ロググループを調べて、上限に到達したかどうかを確認することも可能です。このメトリクスがゼロより大きい場合は、現在少なくとも 1 つのクライアントがブローカーに接続されています。メトリクスがゼロ個の接続を示している場合は、telnet パステストを再度実行します。ブローカーが新しい接続メトリクスを CloudWatch に発行する前に接続が終了する場合は、このプロセスを繰り返す必要がある場合があります。メトリクスは毎分発行されます。
- パブリックアクセシビリティがないブローカーで telnet コマンドが失敗する場合は、ブローカーの Elastic Network Interface のステータスをチェックして、ステータスが in-use になっていることを確認します。各ネットワークインターフェイスに関する Amazon VPC フローログを作成して、生成されたフローログを検証します。telnet コマンドが呼び出されたときのブローカーのプライベート IP アドレスを調べて、応答パケットを含む接続パケットが ACCEPTED であることを確認します。フローログの詳細

と例については、Amazon VPC デベロッパーガイドの「フローログレコードの例」を参照してください。

> **Note**
>
> このステップは、パブリックアクセシビリティを備えた Amazon MQ ブローカーの RabbitMQ には適用されません。 Amazon MQ

5. 以下の `curl` コマンドを実行して、RabbitMQ の管理ウェブコンソールへの接続をチェックします。

```
$ curl https://b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-west-2.amazonaws.com:443/index.html
```

コマンドが正常に完了すると、出力は以下のような HTML ドキュメントになります。

```
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="X-UA-Compatible" content="IE=edge" />
        <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
        <title>RabbitMQ Management</title>
        ...
```

ブローカーが実行中であり、**telnet** を使用して接続を検証できますが、クライアントは接続できず、SSL 例外を返しています。

ブローカーのエンドポイント証明書がブローカーのメンテナンスウィンドウ中に更新されている可能性があります。Amazon MQ ブローカー証明書は定期的にローテーションされ、ブローカーの可用性とセキュリティが引き続き維持されます。

Amazon Trust Services で Amazon ルート認証局 (CA) を使用して、クライアントのトラストストアで認証することをお勧めします。すべての Amazon MQ ブローカーの証明書は、このルート CA で署名されています。Amazon ルート CA を使用することで、ブローカーで証明書が更新されるたびに、新しい Amazon MQ ブローカーの証明書をダウンロードする必要がなくなります。

ブローカーを作成しましたが、ブローカーの作成に失敗しました。

ブローカーが CREATION_FAILED ステータスになっている場合は、以下の手順を実行します。

- IAM 許可をチェックします。ブローカーを作成するには、 AWS マネージド IAM ポリシーを使用するか、カスタム IAM ポリシーに正しい Amazon EC2 アクセス許可のセットAmazonMQFullAccessが必要です。必要な Amazon EC2 許可の詳細については、「Amazon MQ ブローカーの作成に必要な IAM 許可」を参照してください。
- ブローカー用に選択しているサブネットが、共有 Amazon Virtual Private Cloud (VPC) 内にあるかどうかをチェックします。共有 Amazon VPC 内で Amazon MQ ブローカーを作成するには、その Amazon VPC を所有するアカウントでブローカーを作成する必要があります。

ブローカーが再起動したのですが、その理由がよくわかりません。

ブローカーが自動的に再起動した場合は、以下の理由のいずれかが原因で再起動した可能性があります。

- スケジュールされた毎週のメンテナンスウィンドウが原因でブローカーが再起動した可能性があります。Amazon MQ は、ハードウェア、オペレーティングシステム、またはメッセージブローカーのエンジンソフトウェアに対して定期的にメンテナンスを実行します。メンテナンスの所要時間はさまざまですが、メッセージブローカーに対してスケジュールされている操作によっては、最長 2 時間継続することがあります。ブローカーは、この 2 時間のメンテナンスウィンドウのどの時点でも再起動する可能性があります。ブローカーのメンテナンスウィンドウの詳細については、「the section called “ブローカーのメンテナンスのスケジュール”」を参照してください。
- ブローカーのインスタンスタイプがアプリケーションワークロードに適していない可能性があります。例えば、mq.t2.micro で実稼働ワークロードを実行すると、ブローカーのリソースが不足する原因になる場合があります。CPU 使用率が高い、またはブローカーのメモリ使用率が高いと、ブローカーが予期せず再起動する原因になる場合があります。ブローカーが使用する CPU とメモリの量を確認するには、エンジンタイプに対応する以下の CloudWatch メトリクスを使用してください。
  - Amazon MQ での ActiveMQ Amazon MQ – ブローカーが現在使用している割り当てられた Amazon EC2 コンピューティングユニットの割合CpuUtilizationを確認します。ActiveMQ JVM メモリ制限のうち、ブローカーが現在使用している割合について HeapUsage をチェックします。
  - Amazon MQ の RabbitMQ Amazon MQ – ブローカーが現在使用している割り当てられた Amazon EC2 コンピューティングユニットの割合SystemCpuUtilizationを確認します。

使用済みの RAM の量 (バイト単位) について `RabbitMQMemUsed` をチェックし、それを `RabbitMQMemLimit` で除算して、RabbitMQ ノードが使用したメモリの割合を算出します。

ブローカーのインスタンスタイプ、およびワークロードに適したインスタンスタイプを選択する方法の詳細については、「Broker instance types」を参照してください。

Amazon MQ での ActiveMQ のトラブルシューティング Amazon MQ

このセクションの情報は、Amazon MQ ブローカーで ActiveMQ を使用する際に発生する可能性がある一般的な問題の診断と解決に役立ちます。

目次

ロギングをアクティブにしているにもかかわらず、CloudWatch Logs にブローカーの一般ログまたは監査ログが表示されません。

CloudWatch Logs でブローカーのログを表示できない場合は、以下の手順を実行します。

1. ブローカーを作成または再起動するユーザーに `logs:CreateLogGroup` アクセス許可があるかどうかを確認します。ユーザーがブローカーの作成または再起動を行う前に `CreateLogGroup` 許可をユーザーに追加しなければ、Amazon MQ はロググループを作成しません。
2. Amazon MQ が CloudWatch Logs にログを発行することを許可するリソースベースポリシーが設定されているかどうかをチェックします。CloudWatch Logs ロググループへのログの発行を Amazon MQ に許可するには、以下の CloudWatch Logs API アクションに対するアクセス権を Amazon MQ に付与するリソースベースポリシーを設定します。
   - `CreateLogStream` – 指定したロググループの CloudWatch Logs ログストリームを作成します。

- PutLogEvents – 指定された CloudWatch Logs ログストリームにイベントを配信します。

CloudWatch Logs にログを発行するように Amazon MQ で ActiveMQ を設定する方法の詳細については、「ログ記録の設定」を参照してください。 Amazon MQ

ブローカーの再起動またはメンテナンスウィンドウ後、ステータスが RUNNING であってもブローカーに接続できない。なぜですか?

開始したブローカーの再起動後、スケジュールされたメンテナンスウィンドウの完了後、またはスタンバイインスタンスがアクティブ化された障害イベントで、接続の問題が発生する可能性があります。いずれの場合も、ブローカーの再起動後の接続の問題は、ブローカーの Amazon EFS または Amazon EBS ストレージボリュームに保持されるメッセージが異常に多数であることが原因である可能性が最も高いです。再起動中、Amazon MQ は永続化されたメッセージをストレージからブローカメモリに移動します。この診断を確認するには、CloudWatch で Amazon MQ for ActiveMQ ブローカーの次のメトリクスを監視します。

- **StoragePercentUsage** — 100% に近づくほど割合が大きいと、ブローカーは接続を拒否する可能性があります。
- **JournalFilesForFullRecovery** — クリーンでないシャットダウンおよび再起動後に再生されるジャーナルファイルの数を示します。値が増加する、または常に高い値は、再起動後に接続の問題を引き起こす可能性のある未解決のトランザクションを示します。
- **OpenTransactionCount** — 再起動後にゼロより大きい数字は、ブローカーが以前に消費されたメッセージを保存しようとし、その結果、接続の問題が発生することを示します。

この問題を解決するには、XA トランザクションを `rollback()` または `commit()` で解決することをお勧めします。詳細および `rollback()` を使用して XA トランザクションを解決するコード例を確認するには、「XAトランザクションの回復」を参照してください。

一部のクライアントはブローカーに接続していますが、他のクライアントは接続できません。

ブローカーが RUNNING ステータスであり、一部のクライアントはブローカーに正常に接続できますが、他のクライアントは正常に接続できません。ブローカーのワイヤレベル接続の上限に達している可能性があります。ワイヤレベルの接続制限に達したことを確認するには、次の手順を実行します。

- CloudWatch Logs で Amazon MQ ブローカーの ActiveMQ の一般的なブローカーログを確認します。 Amazon MQ 上限に達した場合は、ブローカーログに `Reached Maximum Connections` が表示されます。Amazon MQ ブローカーでの ActiveMQ の CloudWatch Logs の詳細については、「」を参照してくださいthe section called “CloudWatch Logs での口ギングの構造を理解する”。

ワイヤレベルの接続制限に達すると、ブローカーは追加の着信接続を積極的に拒否します。この問題を解決するには、ブローカーインスタンスタイプをアップグレードすることをお勧めします。ワークロードに最適なインスタンスタイプの選択の詳細については、「Broker instance types」を参照してください。

ワイヤレベル接続の数がブローカー接続制限を下回っていることを確認できた場合、問題はクライアントの再起動に関連している可能性があります。ブローカーのログで、`... Inactive for longer than 600000 ms - removing ...` の多数で頻繁なエントリを確認してください。ログエントリは、クライアントの再起動または接続の問題を示しています。この影響は、頻繁にブローカーを切断して再接続するクライアントと Network Load Balancer (NLB) を介してブローカーに接続する場合に顕著になります。これは通常、コンテナベースのクライアントで観察されます。

詳細については、クライアント側のログを確認してください。ブローカーは 600000 ミリ秒後に非アクティブな TCP 接続をクリーンアップし、接続ソケットを解放します。

オペレーションを実行すると、ActiveMQ コンソールに例外 **`org.apache.jasper.JasperException: An exception occurred processing JSP page`** が表示されます。

簡易認証を使用していて、キューとトピックの認可に `AuthorizationPlugin` を設定している場合は、XML 設定ファイルで `AuthorizationEntries` 要素を使用し、`activemq-webconsole` グループにすべてのキューとトピックへのアクセス許可を付与してください。これにより、ActiveMQ ウェブコンソールが ActiveMQ ブローカーと通信できるようになります。

次のサンプルの `AuthorizationEntry` は、`activemq-webconsole` グループにすべてのキューとトピックの読み取りおよび書き込みの許可を付与します。

```
<authorizationEntries>
    <authorizationEntry admin="activemq-webconsole,admins,users" topic=">"
 read="activemq-webconsole,admins,users" write="activemq-webconsole,admins,users" />
    <authorizationEntry admin="activemq-webconsole,admins,users" queue=">"
 read="activemq-webconsole,admins,users" write="activemq-webconsole,admins,users" />
</authorizationEntries>
```

同様に、ブローカーを LDAP に統合する場合は、必ず `amazonmq-console-admins` グループに許可を付与してください。LDAP 統合の詳細については、the section called “LDAP 統合の仕組み” を参照してください。

トラブルシューティング: Amazon MQ での RabbitMQ Amazon MQ

このセクションの情報は、Amazon MQ ブローカーで RabbitMQ を使用する際に発生する可能性がある一般的な問題の診断と解決に役立ちます。

目次

CloudWatch にキューまたは仮想ホストのメトリクスが表示されません。

CloudWatch にキューまたは仮想ホストのメトリクスが表示されない場合は、キューまたは仮想ホストの名前に、空白、タブ、またはその他の非 ASCII 文字が含まれていないか確認してください。

Amazon MQ は、空白、タブ、またはその他の非 ASCII 文字が含まれた名前を持つ仮想ホストおよびキューのメトリクスを発行できません。

ディメンション名の詳細については、Amazon CloudWatch API リファレンスの「Dimension」を参照してください。

Amazon MQ で RabbitMQ でプラグインを有効にするにはどうすればよいですか？

Amazon MQ の RabbitMQ は現在、デフォルトで有効になっている RabbitMQ 管理、シャベル、フェデレーション、整合性ハッシュ交換プラグインのみをサポートしています。 Amazon MQ サポートされているプラグインの詳細については、「the section called “プラグイン”」を参照してください。

ブローカーの Amazon VPC 設定を変更できません。

Amazon MQ は、ブローカーが作成された後の Amazon VPC 設定の変更をサポートしていません。新しい Amazon VPC 設定で新しいブローカーを作成し、クライアント接続 URL を新しいブローカー接続 URL で更新する必要があることに注意してください。

Amazon MQ の ActiveMQ: Elastic Network Interface アラームを削除 Amazon MQ

Amazon MQ の ActiveMQ は、ブローカーの Elastic Network Interface (ENI) を削除すると BROKER_ENI_DELETED アラームを生成します。初めて Amazon MQ ブローカーを作成するときは、Amazon MQ がアカウントの Virtual Private Cloud (VPC) 内に Elastic Network Interface をプロビジョンするため、多数の EC2 許可が必要になります。

このネットワークインターフェイスを変更または削除しないでください。このネットワークインターフェイスを変更または削除すると、VPC とブローカーとの間の接続が完全に失われる可能性があります。ネットワークインターフェイスを削除する場合は、まずブローカーを削除します。

Amazon MQ の ActiveMQ: ブローカーメモリ不足アラーム Amazon MQ

Amazon MQ の ActiveMQ は、メモリ容量不足のためにブローカーが再起動ループが発生すると、BROKER_OOM アラームを生成します。ブローカーが再起動ループ (バウンスループとも呼ばれる) の状態になると、ブローカーは短時間内にリカバリの試行を繰り返します。メモリ容量不足でスタートアップを完了できないブローカーは、再起動のループに入る可能性があり、その間はブローカーとのやり取りが制限されます。

Amazon MQ は、デフォルトでブローカーのメトリクスを有効にします。Amazon CloudWatch コンソールにアクセスするか、CloudWatch API を使用して、ブローカーのメトリクスを表示できます。次のメトリクスは、ActiveMQ BROKER_OOM アラームを診断する場合に役立ちます。

Amazon MQ CloudWatch メトリクス	メモリ使用量が多い理由	
`TotalMessageCount`	メッセージは、消費または破棄されるまでメモリに格納されます。メッセージ数が多いと、リソースの過剰使用が表示され、高メモリアラームの原因となる可能性があります。	

Amazon MQ CloudWatch メトリクス	メモリ使用量が多い理由	
HeapUsage	ブローカーが現在使用している ActiveMQ JVM メモリ制限の割合。パーセンテージが高い場合は、ブローカーが大量のリソースを使用していることを示し、OOM アラームが発生する可能性があります。	
ConnectionCount	クライアント接続にはメモリを使用するため、同時接続が多すぎると高メモリアラームの原因となる可能性があります。	
CpuUtilization	割り当てられた EC2 コンピューティングユニットのうち、現在ブローカーが使用しているものの比率。	
TotalConsumerCount	ブローカーに接続されているすべてのコンシューマーについて、設定された数のメッセージは、コンシューマーに配信される前にストレージからメモリにロードされます。コンシューマーの接続が多いと、メモリ使用率が高くなり、高メモリアラームの原因となる可能性があります。	

再起動ループを防ぎ、BROKER_OOM アラームを回避するには、メッセージがすばやく消費されるようにします。これを行うには、最も効果的なブローカーインスタンスタイプを選択し、配信不能または期限切れのメッセージを破棄するために、デッドレターキューをクリーニングします。Amazon

MQ のベストプラクティスの ActiveMQ で、効果的なパフォーマンスを確保する方法の詳細を確認できます。

Amazon MQ の RabbitMQ: ハイメモリアラーム Amazon MQ

RabbitMQ では、CloudWatch メトリクス `RabbitMQMemUsed` で特定されるブローカーのメモリ使用率が、`RabbitMQMemLimit` で特定されるメモリ制限を超えたときに高メモリアラームが発生します。`RabbitMQMemLimit` は Amazon MQ によって設定され、各ホストインスタンスタイプで使用可能なメモリを考慮して特別に調整されています。CloudWatch ログを有効にして、`Memory resource limit alarm set on host node rabbit@hostname` というメッセージによって高メモリアラームを特定することもできます。

高メモリアラームを発生させた Amazon MQ ブローカーの RabbitMQ は、メッセージを発行しているすべてのクライアントをブロックします。 Amazon MQ メモリ使用率が高いために、ブローカーではアラームの診断および解決を困難にする他の問題が発生することがあります。

メモリ使用率が高いためにスタートアップを完了できない単一インスタンスブローカーは、再起動のループに入る可能性があり、その間はブローカーとのやり取りが制限されます。クラスターのデプロイでは、異なるノード上のレプリカ間でのメッセージの同期がキューで一時停止することがあります。キューで同期が一時停止すると、キューからのメッセージの消費が妨げられるため、メモリアラームを解決する際にはこれに個別に対処する必要があります。

Amazon MQ では、高メモリアラームが発生しているブローカーの再起動は行われません。また、ブローカーでアラームが発生し続ける限り `RebootBroker` API オペレーションに対して例外が返されます。

このセクションの情報は、ブローカーで発生した RabbitMQ の高メモリアラームの診断と解決に役立ちます。

> **Note**
>
> 必要なアクションを実行した後、RABBITMQ_MEMORY_ALARM ステータスがクリアされるまでに数時間かかる場合があります。

> **Note**
>
> ブローカーを mq.m5. インスタンスタイプから mq.t3.micro インスタンスタイプにダウングレードすることはできません。ダウングレードするには、ブローカーを削除し、新しいブローカーを作成する必要があります。

トピック

RabbitMQ ウェブコンソールを使用した高メモリアラームの診断

RabbitMQ ウェブコンソールでは、各ノードのメモリ使用率の詳細情報を生成して表示できます。この情報は、次の手順を実行することで確認できます。

1. にサインイン AWS Management Console し、ブローカーの RabbitMQ ウェブコンソールを開きます。
2. RabbitMQ コンソールの [Overview] (概要) ページで、[Nodes] (ノード) リストからノードの名前を選択します。
3. ノードの詳細ページで、[Memory details] (メモリの詳細) を選択してセクションを展開し、ノードにおけるメモリ使用率の情報を表示します。

RabbitMQ がウェブコンソールで提供するメモリ使用率の情報は、メモリを消費しすぎている可能性や、高メモリアラームの原因となる可能性のあるリソースを特定するのに役立ちます。RabbitMQ ウェブコンソールで使用できるメモリ使用率の詳細については、RabbitMQ Server Documentation ウェブサイトの「Reasoning About Memory Use」を参照してください。

Amazon MQ メトリクスを使用した高メモリアラームの診断

Amazon MQ は、デフォルトでブローカーのメトリクスを有効にします。CloudWatch コンソールにアクセスするか、CloudWatch API を使用して、ブローカーのメトリクスを表示できます。次のメトリクスは、RabbitMQ の高メモリアラームを診断する際に便利です。

Amazon MQ CloudWatch メトリクス	メモリ使用量が多い理由
`MessageCount`	メッセージは、消費または破棄されるまでメモリに格納されます。メッセージ数が多いと、リソースの過剰使用が表示され、高メモリアラームの原因となる可能性があります。
`QueueCount`	また、キューはメモリに格納されます。キューの数が多いと高メモリアラームの原因となる可能性があります。
`ConnectionCount`	クライアント接続にはメモリを使用するため、同時接続が多すぎると高メモリアラームの原因となる可能性があります。
`ChannelCount`	接続と同様に、各接続を使用して確立されたチャネルもノードメモリに格納されます。チャネルの数が多いと高メモリアラームの原因となる可能性があります。
`ConsumerCount`	ブローカーに接続されているすべてのコンシューマーについて、設定された数のメッ

Amazon MQ CloudWatch メトリクス	メモリ使用量が多い理由	
	セージは、コンシューマーに配信される前にストレージからメモリにロードされます。コンシューマーの接続が多いと、メモリ使用率が高くなり、高メモリアラームの原因となる可能性があります。	
PublishRate	メッセージの発行には、ブローカーのメモリが使用されます。メッセージがブローカーに発行される速度が高すぎて、ブローカーがコンシューマーにメッセージを配信する速度を大幅に上回ると、ブローカーで高メモリアラームが発生する可能性があります。	

高メモリアラームへの対応

特定したコントリビューターごとに、ブローカーの高メモリアラームを軽減して解決するため、次の一連のアクションをお勧めします。

メモリ使用量が多い理由	Amazon MQ の推奨	
キュー内のメッセージ数が多すぎます。	次のいずれかを実行します。 • キューに発行されたメッセージを消費します。 • キューからメッセージをパージします。	

メモリ使用量が多い理由	Amazon MQ の推奨	
	• ブローカーからキューを削除します。	
ブローカーで設定されたキューの数が多すぎます。	キューの数を減らします。	
ブローカーで確立された接続の数が多すぎます。	接続の数を減らします。詳細については、「the section called “接続およびチャネルの数の削減”」を参照してください。	
ブローカーで確立されたチャネルの数が多すぎます。	チャネルの数を減らします。詳細については、「the section called “接続およびチャネルの数の削減”」を参照してください。	
ブローカーに接続されたコンシューマーの数が多すぎます。	ブローカーに接続されたコンシューマーの数を減らします。	
メッセージ発行速度が高すぎます。	パブリッシャーがメッセージをブローカーに発行する速度を低くします。	
クライアント接続試行速度が高すぎます。	メッセージを発行または消費できるようにクライアントがブローカーへの接続を試行する頻度を減らすか、ブローカーを設定します。	

接続およびチャネルの数の削減

Amazon MQ ブローカー上の RabbitMQ への接続は、クライアントアプリケーションによって、または RabbitMQ ウェブコンソールを使用して手動で閉じることで閉じられます。RabbitMQ ウェブコンソールを使用して接続を終了するには、次の手順を実行します。

1. にサインイン AWS Management Console し、ブローカーの RabbitMQ ウェブコンソールを開きます。
2. RabbitMQ コンソールで、[Connections] (接続) タブを選択します。
3. [Connections] (接続) ページの [All connections] (すべての接続) から、終了する接続の名前をリストから選択します。
4. 接続の詳細ページで、[Close this connection] (この接続を終了する) を選択してセクションを展開し、[Force Close] (強制終了) を選択します。オプションで、理由のデフォルトのテキストをお客様自身の説明に置き換えることもできます。Amazon MQ の RabbitMQ は、接続を閉じるときに指定した理由をクライアントに返します。 Amazon MQ
5. ダイアログボックスで [OK] を選択し、確認して接続を終了します。

接続を終了すると、終了した接続に関連付けられているすべてのチャネルも終了します。

> **Note**
>
> クライアントアプリケーションは、終了後にブローカーが自動的に接続を再確立するように設定されている場合があります。この場合、接続またはチャネルの数を減らすには、ブローカーのウェブコンソールからの接続を終了するだけでは不十分です。

パブリックアクセスがないブローカーの場合、適切なメッセージプロトコルのポート (例えば AMQP 接続の場合、ポート 5671) でインバウンドトラフィックを拒否することで、一時的に接続をブロックできます。ブローカーの作成時に Amazon MQ に指定したセキュリティグループのポートをブロックできます。セキュリティグループの変更方法の詳細については、Amazon VPC ユーザーガイドの「セキュリティグループへのルールの追加」を参照してください。

クラスターのデプロイで一時停止したキューの同期への対応

RabbitMQ の高メモリアラームに対処しているときに、1 つまたは複数のキューのメッセージを消費できないことがあります。これらのキューは、ノード間でメッセージを同期中である可能性がありま

す。その間、それぞれのキューは、メッセージの発行および消費に使用できなくなります。高メモリアラームが原因でキューの同期が一時停止し、メモリアラームの原因になることさえあります。

一時停止したキューの同期の停止と再試行の詳細については、「the section called “一時停止されたキュー同期の解決”」を参照してください。

単一インスタンスブローカーでの再起動ループへの対応

高メモリアラームを発生させる Amazon MQ シングルインスタンスブローカーの RabbitMQ は、再起動して起動するのに十分なメモリがない場合、使用できなくなるリスクがあります。 Amazon MQ これにより、RabbitMQ が再起動のループに入り、問題が解決するまでブローカーとのやり取りが妨げられる可能性があります。ブローカーが再起動のループ状態にある場合、このセクションで前述した Amazon MQ で推奨されるアクションを適用して、高メモリアラームを解決することはできません。

ブローカーを回復させるには、より多くのメモリを持つ大きなインスタンスタイプにアップグレードすることをお勧めします。クラスターのデプロイとは異なり、再起動中にノード間で実行するキューの同期がないため、高メモリアラームの発生時に単一インスタンスブローカーをアップグレードできます。

高メモリアラームの防止

特定する要因ごとに、RabbitMQ の高メモリアラームの発生を防止および低減するため、次の一連のアクションを推奨します。

メモリ使用量が多い理由	Amazon MQ の推奨
キュー内のメッセージ数が多すぎます。	以下の操作を実行します。 • レイジーキューを有効にします。 • 設定を行うか、キューの深度の制限を減らします。
ブローカーで設定されたキューの数が多すぎます。	設定を行うか、キューの数の制限を減らします。
ブローカーで確立された接続の数が多すぎます。	設定を行うか、接続の数の制限を減らします。

メモリ使用量が多い理由	Amazon MQ の推奨
ブローカーで確立されたチャネルの数が多すぎます。	クライアントアプリケーションで、接続あたりのチャネルの最大数を設定します。
ブローカーに接続されたコンシューマーの数が多すぎます。	小さいコンシューマーのプリフェッチの制限を設定します。
クライアント接続試行速度が高すぎます。	より長時間の接続を使用して、接続の試行回数と頻度を減らします。

ブローカーのメモリアラームが解決したら、ホストインスタンスタイプを追加のリソースを含むインスタンスにアップグレードできます。ブローカーのインスタンスタイプを更新する方法については、Amazon MQ REST API リファレンスの「`UpdateBrokerInput`」を参照してください。

Amazon MQ の RabbitMQ: 無効な AWS Key Management Service キー Amazon MQ

Amazon MQ の RabbitMQ は、カスタマーマネージド AWS KMS key(CMK) で作成されたブローカーが AWS Key Management Service (KMS) キーが無効であることを検出したときに、INVALID_KMS_KEY の重要なアクション必須コードを生成します。CMK を備えた RabbitMQ ブローカーは、KMS キーが有効になっていることと、ブローカーに必要な権限がすべて付与されていることを定期的に確認します。キーが有効になっていることを RabbitMQ が確認できない場合、ブローカーは隔離され、RabbitMQ は INVALID_KMS_KEY を返します。

有効な KMS キーがない場合、ブローカーにはカスタマー管理の KMS キーに対する基本的なアクセス許可がありません。ユーザーがキーを再度有効にしてブローカーが再起動するまで、ブローカーはキーを使用して暗号化操作を実行できません。KMS キーが無効になっている RabbitMQ ブローカーは、劣化を防ぐために隔離されます。KMS キーが再び有効になったことを RabbitMQ が確認すると、ブローカーは隔離から除外されます。Amazon MQ は、KMS キーが無効になっているブローカーを再起動せず、ブローカーが無効な KMS キーを保持し続ける限り、`RebootBroker` API オペレーションに対して例外を返します。

INVALID_KMS_KEY の診断と対処

INVALID_KMS_KEY アクションの必須コードを診断して対処するには、 コマンドラインインターフェイス (CLI) AWS と AWS Key Management Service コンソールを使用する必要があります。

KMS キーを再度有効にするには

1. `DescribeBroker` メソッドを呼び出して CMK ブローカーの `kmsKeyId` を取得します。
2. AWS Key Management Service コンソールにサインインします。
3. [カスタマー管理キー] ページで、問題のあるブローカーの KMS キー ID を見つけて、ステータスが [有効] であることを確認します。
4. KMS キーが無効になっている場合は、[キーアクション]、[有効化] の順に選択してキーを再度有効にします。キーを再度有効にしたら、RabbitMQ がブローカーを隔離から除外するまで待つ必要があります。

必要な許可がまだブローカーの KMS キーに関連付けられていることを確認するには、`ListGrantListGrant` メソッドを呼び出して、`mq_rabbit_grant` と `mq_grant` が存在することを確認します。KMS 許可またはキーが削除されている場合は、ブローカーを削除し、必要な許可をすべて備えた新しいブローカーを作成する必要があります。ブローカーを削除する手順については、「ブローカーの削除」を参照してください。

重要なアクションが必要なコード INVALID_KMS_KEY が発生しないようにするには、KMS キーまたは CMK 許可を手動で削除または無効化しないでください。キーを削除する場合は、まずブローカーを削除します。

Amazon MQ の RabbitMQ: ディスク制限アラーム Amazon MQ

ディスク制限アラームは、新しいメッセージが追加される一方で消費されないメッセージが多いため、RabbitMQ ノードが使用するディスク量が減少したことを示します。RabbitMQ は、Amazon CloudWatch メトリクス `RabbitMQDiskFree` で識別されるブローカーの空きディスク容量が、`RabbitMQDiskFreeLimit` で識別されるディスク制限に達すると、ディスク制限アラームを発生させます。`RabbitMQDiskFreeLimit` は Amazon MQ によって設定され、各ブローカーインスタンスタイプで使用可能なディスク容量を考慮して定義されています。

ディスク制限アラームを生成した Amazon MQ 上の RabbitMQ ブローカーは、新しいメッセージを発行できなくなります。同じ接続上にパブリッシャーとコンシューマーがある場合、コンシューマーもメッセージを受信できなくなります。RabbitMQ をクラスターで実行する場合、ディスクアラーム

はクラスター全体に適用されます。1 つのノードが制限を下回ると、他のすべてのノードが受信メッセージをブロックします。ディスク容量の不足のために、ブローカーではアラームの診断および解決を困難にする他の問題が発生することがあります。

Amazon MQ では、ディスクアラームが発生しているブローカーの再起動は行われません。また、ブローカーでアラームが発生し続ける限り `RebootBroker` API オペレーションに対して例外が返されます。

> **Note**
>
> ブローカーを `mq.m5` インスタンスタイプから `mq.t3.micro` インスタンスタイプにダウングレードすることはできません。ダウングレードするには、ブローカーを削除し、新しいブローカーを作成する必要があります。

ディスク制限アラームの診断と対処

Amazon MQ は、デフォルトでブローカーのメトリクスを有効にします。Amazon CloudWatch コンソールにアクセスするか、CloudWatch API を使用して、ブローカーのメトリクスを表示できます。`MessageCount` は、RabbitMQ ディスク制限アラームを診断する際に役立つメトリクスです。メッセージは、消費または破棄されるまでメモリに格納されます。メッセージ数が多い場合は、ディスクストレージが過剰に使用されていることを示し、ディスクアラームの原因となる可能性があります。

ディスク制限アラームを診断するには、Amazon MQ マネジメントコンソールを使用して次の操作を行います。

- キューに発行されたメッセージを消費する新しい接続を作成します。
- キューからメッセージをパージします。
- ブローカーからキューを削除します。

> **Note**
>
> 必要なアクションを実行した後、RABBITMQ_DISK_ALARM ステータスがクリアされるまでに数時間かかる場合があります。

ディスク制限アラームの再発を防ぐには、ホストインスタンスタイプを追加のリソースを含むインスタンスにアップグレードします。ブローカーのインスタンスタイプを更新する方法については、Amazon MQ REST API リファレンスの「UpdateBrokerInput」を参照してください。また、パブリッシャーとコンシューマーは別々の接続上に保持することをお勧めします。

Amazon MQ クォーラムキューの RabbitMQ アラーム Amazon MQ

クォーラムキューは、Amazon MQ バージョン 3.13 以降の RabbitMQ でのみサポートされています。 Amazon MQ Amazon MQ の RabbitMQ は、バージョン 3.12 以前を使用して単一のインスタンスまたはクラスターブローカーでクォーラムキューを作成しようとするRABBITMQ_QUORUM_QUEUES_NOT_SUPPORTED_ON_CURRENT_VERSIONと、必要な重要なアクションコードを生成します。

RABBITMQ_QUORUM_QUEUES_NOT_SUPPORTED_ON_CURRENT_VERSION アラームを診断して対処するには、RabbitMQ 管理ダッシュボードでクォーラムキューの一覧を確認します。

- メッセージを保持する必要がない場合は、クォーラムキューを削除し、ブローカーをバージョン 3.13 以降にアップグレードして、ブローカーのアップグレード後にクォーラムキューを再作成できます。
- メッセージを保持する必要がある場合は、バージョン 3.13 以降で新しいブローカーを作成し、新しいブローカーにクォーラムキューを作成する必要があります。新しいブローカーとクォーラムキューを作成したら、Shovel または Federation プラグインを使用して、以前のブローカーから新しいブローカーにメッセージを移行できます。その後、以前のブローカーを削除します。

RABBITMQ_QUORUM_QUEUES_NOT_SUPPORTED_ON_CURRENT_VERSION を防ぐには、ブローカーをバージョン 3.13 以降にアップグレードしてから、そのブローカーにクォーラムキューを作成します。

関連リソース

Amazon MQ のリソース

以下の表は、Amazon MQ の使用に役立つリソースのリストです。

リソース	説明
Amazon MQ REST API リファレンス	REST リソース、サンプルリクエスト、HTTP メソッド、スキーマ、パラメータ、およびサービスから返されるエラーの説明です。
「AWS CLI コマンドリファレンス」内の Amazon MQ	メッセージブローカーの操作に使用できる AWS CLI コマンドの説明。
「AWS CloudFormation ユーザーガイド」内の Amazon MQ	`AWS::Amazon MQ::Broker` リソースを使用すると、Amazon MQ ブローカーを作成する、指定されたブローカーに対して設定変更の追加またはユーザーの変更を行う、指定されたブローカーに関する情報を返す、および指定されたブローカーを削除することができます。 `AWS::Amazon MQ::Configuration` リソースを使用すると、Amazon MQ 設定を作成する、設定変更の追加とユーザーの変更を行う、および指定された設定に関する情報を返すことができます。
のリージョンとエンドポイント	Amazon MQ のリージョンとエンドポイントに関する情報
製品ページ	Amazon MQ に関する情報のメインウェブページです。
フォーラム	デベロッパーが Amazon MQ に関連する技術的な質問について話し合うためのコミュニティベースのフォーラムです。

リソース	説明
AWS プレミアムサポート情報	インフラストラクチャサービスで AWS アプリケーションを構築および実行するための one-on-one の高速応答サポートチャネルである AWS プレミアムサポートに関する情報のプライマリウェブページ

Amazon MQ for ActiveMQ のリソース

以下の表は、Apache ActiveMQ の使用に役立つリソースのリストです。

リソース	説明
Apache ActiveMQ Getting Started Guide	Apache ActiveMQ の公式ドキュメントです。
ActiveMQ in Action	JMS メッセージ、コネクタ、メッセージの持続性、認証、認可の構造を説明した Apache ActiveMQ のガイドです。
言語間のクライアント	プログラミング言語と対応する Apache ActiveMQ ライブラリのリストです。「ActiveMQ クライアント」と「QpidJMS クライアント」も参照してください。

Amazon MQ for RabbitMQ のリソース

以下の表は、RabbitMQ の使用に役立つリソースのリストです。

リソース	説明
The RabbitMQ Getting Started Guide	RabbitMQ の公式ドキュメントです。
RabbitMQ Client Libraries and Developer Tools	さまざまなプログラミング言語とプラットフォームを使用した RabbitMQ での作業のための、公式にサポートされているクライアントラ

リソース	説明
	イブラリとデベロッパーツールに関するガイドです。
RabbitMQ Best Practices	RabbitMQ を使用するためのベストプラクティスと推奨事項に関する CloudAMQP のガイドです。

Amazon MQ リリースノート

以下の表には、Amazon MQ 機能のリリースおよび改善がリストされています。

日付	ドキュメントの更新
2025 年 4 月 22 日	`DeleteConfiguration` API を使用して Amazon MQ ブローカー設定を削除できるようになりました。詳細については、Amazon MQ API リファレンス」の「設定」を参照してください。
2025 年 4 月 16 日	Amazon MQ for RabbitMQ で、デュアルスタック (IPv4 および IPv6) エンドポイントを使用してパブリックブローカーとプライベートブローカーに接続できるようになりました。詳細については、「Connecting to Amazon MQ」および「Configuring a private Amazon MQ broker」を参照してください。
2025 年 4 月 7 日	Amazon MQ がアジアパシフィック (タイ) およびメキシコ (中部) リージョンで利用可能になりました。 利用可能なリージョンの詳細については、AWS 全般リファレンスガイドの「AWS リージョンとエンドポイント」を参照してください。
2025 年 2 月 13 日	Amazon MQ API FIPS エンドポイントが、カナダ (中部) およびカナダ西部 (カルガリー) リージョンで利用可能になりました。 Amazon MQ API で FIPS エンドポイントを使用する方法の詳細については、「」を参照してくださいConnecting to Amazon MQ。 利用可能なリージョンの詳細については、AWS 全般リファレンスガイドの「AWS リージョンとエンドポイント」を参照してください。
2025 年 2 月 12 日	Amazon MQ は、次のインスタンスタイプのサポート終了日を発表しました。 Broker instance types • ActiveMQ `mq.t2.micro` : 2025 年 5 月 12 日 • ActiveMQ `mq.m4.large` : 2025 年 5 月 12 日

日付	ドキュメントの更新
	2025 年 3 月 17 `mq.t2.micro` `mq.m4.large` 日以降はブローカーを作成できません。
2024 年 12 月 10 日	Amazon MQ は AWS PrivateLink を使用して、トラフィックをパブリックインターネットに公開することなく、仮想プライベートクラウド (VPCs) と Amazon MQ API 間の接続をサポートするようになりました。詳細については、「the section called “AWS PrivateLink を使用して Amazon MQ に接続する”」を参照してください。
2024 年 11 月 18 日	Amazon MQ がアジアパシフィック (マレーシア) リージョンで利用可能になりました。利用可能なリージョンの詳細については、AWS 全般リファレンスガイドの「AWS リージョンとエンドポイント」を参照してください。
2024 年 11 月 14 日	Amazon MQ は、次のエンジンバージョンのサポート終了日を発表しました。 Amazon MQ for ActiveMQ エンジンバージョンの管理 • ActiveMQ 5.17: 2025 年 6 月 16 日 Amazon MQ for RabbitMQ エンジンバージョンの管理 • RabbitMQ 3.11: 2025 年 2 月 17 日 • RabbitMQ 3.12: 2025 年 3 月 17 日 最新バージョンへのアップグレードの詳細については、「」を参照してください。 Amazon MQ ブローカーエンジンバージョンのアップグレード
2024 年 11 月 13 日	Amazon MQ は、IPv4 または IPv6 を使用して に接続できるデュアルスタックサービスエンドポイントをサポートするようになりました。Amazon MQ デュアルスタックリージョンサービスエンドポイントは、 Aと DNS AAAA レコードの両方で解決できます。詳細については、「???」を参照してください。

日付	ドキュメントの更新
2024 年 7 月 25 日	Amazon MQ が、新しいマイナーエンジンバージョンのリリースである ActiveMQ 5.18 をサポートするようになりました。詳細については次を参照してください: • ActiveMQ 5.18 リリースページ • Amazon MQ for ActiveMQ エンジンバージョンの管理 • Amazon MQ ブローカーエンジンバージョンのアップグレード • Spring XML 設定ファイルの使用
2024 年 7 月 22 日	Amazon MQ が、バージョン 3.13 以降のブローカーでのみクォーラムキューをサポートするようになりました。クォーラムキューは、Raft コンセンサスアルゴリズムを使用してデータ整合性を維持する、レプリケーション型の FIFO キュータイプです。クォーラムキューは有害メッセージの処理を提供するため、未処理のメッセージの管理に役立つ可能性があります。 クォーラムキューの使用を開始するには、「Amazon MQ での RabbitMQ のクォーラムキュー」を参照してください。
2024 年 7 月 2 日	Amazon MQ for RabbitMQ が、マイナーバージョンリリースである RabbitMQ 3.13 をサポートするようになりました。エンジンバージョン 3.13 以降を使用しているすべてのブローカーについて、Amazon MQ は、サポートされている最新のパッチバージョンへのアップグレードをメンテナンスウィンドウ内で管理します。詳細については、「Amazon MQ ブローカーエンジンバージョンのアップグレード」を参照してください。 「Amazon MQ for RabbitMQ のサイズ設定ガイドライン」が更新され、エンジンバージョン 3.13 を使用するブローカーのキュー、チャネルあたりのコンシューマー、シャベルの新しい制限値が含められました。 このリリースでの修正と機能の詳細については、RabbitMQ サーバー GitHub リポジトリの RabbitMQ 3.13 リリースノートを参照してください。 サポートされる Amazon MQ for RabbitMQ のバージョンとブローカーアップグレードの詳細については、「Amazon MQ for RabbitMQ エンジンバージョンの管理」を参照してください。

日付	ドキュメントの更新
2024 年 6 月 10 日	Amazon MQ がカナダ西部 (カルガリー) リージョンで利用可能になりました。利用可能なリージョンの詳細については、AWS 全般リファレンスガイドの「AWS リージョンとエンドポイント」を参照してください。
2024 年 5 月 10 日	Amazon MQ バージョンサポートカレンダーは、ブローカーエンジンバージョンがサポート終了に達するタイミングを示します。あるエンジンバージョンがサポート終了に達すると、Amazon MQ は、そのバージョンのすべてのブローカーを、サポートされている次のマイナーバージョンに自動的に更新します。Amazon MQ は、エンジンバージョンがサポート終了に達する少なくとも 90 日前に通知を送信します。 バージョンサポートカレンダーとサポート終了日を確認するには、以下を参照してください。 • Amazon MQ for ActiveMQ エンジンバージョンの管理 • Amazon MQ for RabbitMQ エンジンバージョンの管理 自動マイナーバージョンアップグレードを有効にして、メンテナンスウィンドウ内でブローカーが次のパッチバージョンに更新されるようにすることもできます。詳細については、Amazon MQ ブローカーエンジンバージョンのアップグレードを参照してください。

日付	ドキュメントの更新
2024 年 5 月 9 日	Amazon MQ for RabbitMQ が、マイナーバージョンリリースである RabbitMQ 3.12 をサポートするようになりました。3.12.13 以降のすべてのブローカーは Classic Queues バージョン 2 (CQv2) を使用し、3.12.13 以降のすべてのキューはレイジーキューとして動作します。 3.12.13 より前のバージョンのブローカーは、CQv2 キューとレイジーキューを有効にするか、Amazon MQ for RabbitMQ の最新バージョンにアップグレードすることをお勧めします。 このリリースでの修正と機能の詳細については、以下を参照してください。 • RabbitMQ サーバー GitHub リポジトリの RabbitMQ 3.12 リリースノート サポートされる Amazon MQ for RabbitMQ のバージョンとブローカーアップグレードの詳細については、「Amazon MQ for RabbitMQ エンジンバージョンの管理」を参照してください。
2024 年 3 月 4 日	Amazon MQ for RabbitMQ が RabbitMQ 3.11.28 をサポートするようになりました。 このリリースでの修正と機能の詳細については、以下を参照してください。 • RabbitMQ サーバー GitHub リポジトリの RabbitMQ 3.11.28 リリースノート • RabbitMQ changelog サポートされる Amazon MQ for RabbitMQ のバージョンとブローカーアップグレードの詳細については、「Amazon MQ for RabbitMQ エンジンバージョンの管理」を参照してください。
2024 年 1 月 19 日	Amazon MQ for RabbitMQ では、ユーザー名「guest」はサポートされず、デフォルトのゲストアカウントは新しいブローカーの作成時に削除されます。ユーザーが作成した「guest」というアカウントも、Amazon MQ によって定期的に削除されます。

日付	ドキュメントの更新
2023 年 12 月 15 日	Amazon MQ がイスラエル (テルアビブ) リージョンで利用可能になりました。利用可能なリージョンの詳細については、AWS 全般リファレンスガイドの「AWS リージョンとエンドポイント」を参照してください。
2023 年 12 月 11 日	Amazon MQ for RabbitMQ が RabbitMQ 3.10.25 をサポートするようになりました。 このリリースでの修正と機能の詳細については、以下を参照してください。 • RabbitMQ サーバー GitHub リポジトリの RabbitMQ 3.10.25 リリースノート • RabbitMQ changelog サポートされる Amazon MQ for RabbitMQ のバージョンとブローカーアップグレードの詳細については、「Amazon MQ for RabbitMQ エンジンバージョンの管理」を参照してください。
2023 年 10 月 26 日	Amazon MQ は、重要な更新を含む最新の ActiveMQ マイナーバージョン 5.15.16、5.16.7、5.17.6 をリリースしました。ActiveMQ の古いマイナーバージョンを非推奨とし、すべてのブローカーについて 5.15 のすべてのバージョンを 5.15.16、5.16 のすべてのバージョンを 5.16.7、5.17 のすべてのバージョンを 5.17.6 にアップデートします。 ActiveMQ ブローカーの更新の詳細については、「Amazon MQ for ActiveMQ エンジンバージョンの管理」を参照してください。

日付	ドキュメントの更新
2023 年 9 月 27 日	Amazon MQ for RabbitMQ が RabbitMQ 3.11.20 をサポートするようになりました。 このリリースでの修正と機能の詳細については、以下を参照してください。 • RabbitMQ サーバー GitHub リポジトリの「RabbitMQ 3.11.20 リリースノート」 • RabbitMQ changelog サポートされる Amazon MQ for RabbitMQ のバージョンとブローカーアップグレードの詳細については、「Amazon MQ for RabbitMQ エンジンバージョンの管理」を参照してください。
2023 年 7 月 27 日	Amazon MQ for RabbitMQ が RabbitMQ 3.11.16 をサポートするようになりました。 このリリースでの修正と機能の詳細については、以下を参照してください。 • RabbitMQ サーバー GitHub リポジトリの RabbitMQ 3.11.16 リリースノート • RabbitMQ changelog サポートされる Amazon MQ for RabbitMQ のバージョンとブローカーアップグレードの詳細については、「Amazon MQ for RabbitMQ エンジンバージョンの管理」を参照してください。
2023 年 7 月 27 日	Amazon MQ for RabbitMQ は、RabbitMQ ブローカーの設定の作成と適用をサポートするようになりました。 ブローカーに設定を追加する方法の詳細については、「RabbitMQ Broker Configurations」を参照してください。 この機能の詳細については、以下を参照してください。 • オペレーターポリシー • オペレーターポリシーの変更

日付	ドキュメントの更新
2023 年 6 月 23 日	Amazon MQ が、新しいマイナーエンジンバージョンのリリースである ActiveMQ 5.17.3 をサポートするようになりました。このリリースでは、Amazon MQ の新しいクロスリージョンデータレプリケーション (CRDR) 機能をサポートしています。 詳細については次を参照してください: • CRDR の開始方法については、開発者ガイドの「Amazon MQ for ActiveMQ のクロスリージョンデータレプリケーション」を参照してください。 • ActiveMQ 5.17.3 リリースページ • Amazon MQ for ActiveMQ エンジンバージョンの管理 • Amazon MQ ブローカーエンジンバージョンのアップグレード • Spring XML 設定ファイルの使用
2023 年 6 月 21 日	Amazon MQ for ActiveMQ では、クロスリージョンデータレプリケーション (CRDR) 機能が提供されるようになりました。これにより、プライマリリージョンのプライマリブローカーからレプリカ AWS リージョンのレプリカブローカーへの非同期メッセージレプリケーションが可能になります。プライマリリージョンのプライマリブローカーに障害が発生した場合、スイッチオーバーまたはフェイルオーバーを開始することで、セカンダリリージョンのレプリカブローカーをプライマリに昇格させることができます。 CRDR の開始方法については、開発者ガイドの「Amazon MQ for ActiveMQ のクロスリージョンデータレプリケーション」を参照してください。

日付	ドキュメントの更新
2023 年 5 月 18 日	Amazon MQ は、以下のリージョンでご利用いただけるようになりました。 • アジアパシフィック (メルボルン) • アジアパシフィック (ハイデラバード) • 欧州 (スペイン) • 欧州 (チューリッヒ) 利用可能なリージョンの詳細については、AWS 全般リファレンスガイドの「AWS リージョンとエンドポイント」を参照してください。
2023 年 4 月 14 日	Amazon MQ for RabbitMQ が RabbitMQ バージョン 3.9.27 をサポートするようになりました。 このリリースでの修正と機能の詳細については、以下を参照してください。 • RabbitMQ サーバー GitHub リポジトリの RabbitMQ 3.9.27 リリースノート • RabbitMQ changelog サポートされる Amazon MQ for RabbitMQ のバージョンとブローカーアップグレードの詳細については、「Amazon MQ for RabbitMQ エンジンバージョンの管理」を参照してください。
2023 年 4 月 14 日	Amazon MQ for RabbitMQ が RabbitMQ バージョン 3.10.20 をサポートするようになりました。 このリリースでの修正と機能の詳細については、以下を参照してください。 • RabbitMQ サーバー GitHub リポジトリの RabbitMQ 3.10.20 リリースノート • RabbitMQ changelog サポートされる Amazon MQ for RabbitMQ のバージョンとブローカーアップグレードの詳細については、「Amazon MQ for RabbitMQ エンジンバージョンの管理」を参照してください。

日付	ドキュメントの更新
2023 年 3 月 31 日	Amazon MQ for RabbitMQ が RabbitMQ エンジンバージョン 3.10.17 を無効化 Amazon MQ for RabbitMQ チームと RabbitMQ のオープンソース保守管理者は、バージョン 3.10.17 の RabbitMQ マネジメントコンソールに関する問題を特定しました。Amazon MQ はこのバージョンを撤回しました。この問題の影響を軽減するために、RabbitMQ の新しいパッチバージョンのサポートに取り組んでいる間、バージョン 3.10.20 で新しいブローカーを作成してください。マイナーバージョンのauto アップグレードオプションを有効にして、最新のバグ修正、セキュリティアップデート、パフォーマンスの向上を自動的に受けることをお勧めします。 Amazon MQ for RabbitMQ の利用可能なバージョンの詳細については、「Amazon MQ for RabbitMQ エンジンバージョン」を参照してください。
2023 年 3 月 1 日	Amazon MQ for RabbitMQ が RabbitMQ バージョン 3.10.17 をサポートするようになりました。 このリリースでの修正と機能の詳細については、以下を参照してください。 • RabbitMQ サーバー GitHub リポジトリの RabbitMQ 3.10.17 リリースノート • RabbitMQ changelog サポートされる Amazon MQ for RabbitMQ のバージョンとブローカーアップグレードの詳細については、「Amazon MQ for RabbitMQ エンジンバージョンの管理」を参照してください。

日付	ドキュメントの更新
2023 年 2 月 21 日	Amazon MQ for RabbitMQ が AWS Key Management Service (KMS) と統合され、サーバー側の暗号化が提供されるようになりました。独自のカスタマーマネージド CMK を選択するか、 AWS KMS アカウントで AWS マネージド KMS キーを使用できるようになりました。詳細については、「保管中の暗号化」を参照してください。 Amazon MQ は、次の方法で AWS KMS キーの使用をサポートしています。 • Amazon MQ 所有の KMS キー (デフォルト) – キーは Amazon MQ が所有、管理し、ユーザーのアカウントにはありません。 • AWS マネージド KMS キー — AWS マネージド KMS キー (aws/mq) は、Amazon MQ によってユーザーに代わって作成、管理、使用されるアカウントの KMS キーです。 • 既存のカスタマーマネージド KMS キーを選択する – カスタマーマネージド KMS キーは、ユーザーが AWS Key Management Service (KMS) で作成し、管理します。
2023 年 1 月 13 日	Amazon MQ for RabbitMQ が RabbitMQ バージョン 3.8.34 をサポートするようになりました。 このリリースでの修正と機能の詳細については、以下を参照してください。 • RabbitMQ サーバー GitHub リポジトリの RabbitMQ 3.8.34 リリースノート • RabbitMQ changelog サポートされる Amazon MQ for RabbitMQ のバージョンとブローカーアップグレードの詳細については、「Amazon MQ for RabbitMQ エンジンバージョンの管理」を参照してください。

日付	ドキュメントの更新
2022 年 12 月 15 日	Amazon MQ for RabbitMQ が RabbitMQ バージョン 3.9.24 をサポートするようになりました。 このリリースでの修正と機能の詳細については、以下を参照してください。 • RabbitMQ サーバー GitHub リポジトリの RabbitMQ 3.9.24 リリースノート • RabbitMQ changelog サポートされる Amazon MQ for RabbitMQ のバージョンとブローカーアップグレードの詳細については、「Amazon MQ for RabbitMQ エンジンバージョンの管理」を参照してください。
2022 年 12 月 13 日	Amazon MQ が、中東 (UAE) リージョンで利用可能になりました。利用可能なリージョンの詳細については、AWS 全般リファレンスガイドの「AWS リージョンとエンドポイント」を参照してください。
2022 年 11 月 14 日	Amazon MQ for RabbitMQ が、メジャーエンジンバージョンのリリースである 3.10 をサポートするようになりました。RabbitMQ キューで Queues バージョン 2 (CQv2) を有効にできるようになりました。3.8 から 3.10 への直接更新はサポートされていません。詳細については次を参照してください: • RabbitMQ 3.10.10 リリースノート • RabbitMQ changelog サポートされる Amazon MQ for RabbitMQ のバージョンとブローカーアップグレードの詳細については、「Amazon MQ for RabbitMQ エンジンバージョンの管理」を参照してください。

日付	ドキュメントの更新
2022 年 11 月 9 日	Amazon MQ が、新しいマイナーエンジンバージョンのリリースである ActiveMQ 5.17.2 をサポートするようになりました。詳細については次を参照してください: • ActiveMQ 5.17.2 リリースページ • Amazon MQ for ActiveMQ エンジンバージョンの管理 • Amazon MQ ブローカーエンジンバージョンのアップグレード • Spring XML 設定ファイルの使用
2022 年 8 月 17 日	Amazon MQ が、新しいメジャーエンジンバージョンのリリースである ActiveMQ 5.17.1 をサポートするようになりました。詳細については次を参照してください: • ActiveMQ 5.17.1 リリースページ • Amazon MQ for ActiveMQ エンジンバージョンの管理 • Amazon MQ ブローカーエンジンバージョンのアップグレード • Spring XML 設定ファイルの使用
2022 年 7 月 14 日	Amazon MQ が、マイナーエンジンバージョンのリリースである ActiveMQ 5.16.5 をサポートするようになりました。詳細については次を参照してください: • ActiveMQ 5.16.5 リリースページ • Amazon MQ for ActiveMQ エンジンバージョンの管理 • Spring XML 設定ファイルの使用 • Amazon MQ ブローカーエンジンバージョンのアップグレード
2022 年 5 月 4 日	Amazon MQ は、ブローカー設定の `networkConnector` 要素に包括的な言語を追加します。 • ブローカーの Amazon MQ ネットワークの作成と設定

日付	ドキュメントの更新
2022 年 4 月 25 日	Amazon MQ このリリースでは、`CRITICAL_ACTION_REQUIRED` ブローカーステートと `ActionRequired` API プロパティを追加します。`CRITICAL_ACTION_REQUIRED` は、ブローカーが低下したときに通知します。`ActionRequired` には、デベロッパーガイドで問題の解決方法を見つけるために使用するコードが用意されています。 • トラブルシューティング • Amazon MQ API リファレンス 内の `ActionRequired` ドキュメント。
2022 年 4 月 20 日	Amazon MQ が、新しいマイナーエンジンバージョンのリリースである ActiveMQ 5.16.4 をサポートするようになりました。詳細については次を参照してください: • ActiveMQ 5.16.4 Release ページ • Amazon MQ for ActiveMQ エンジンバージョンの管理 • Spring XML 設定ファイルの使用 • Amazon MQ ブローカーエンジンバージョンのアップグレード
2022 年 3 月 1 日	Amazon MQ がアジアパシフィック (ジャカルタ) リージョンで利用可能になりました。利用可能なリージョンの詳細については、AWS 全般リファレンスガイドの「AWS リージョンとエンドポイント」を参照してください。
2022 年 2 月 25 日	Amazon MQ for RabbitMQ が RabbitMQ バージョン 3.8.27 をサポートするようになりました。 このリリースでの修正と機能の詳細については、以下を参照してください。 • RabbitMQ サーバー GitHub リポジトリの RabbitMQ 3.8.27 リリースノート • RabbitMQ changelog サポートされる Amazon MQ for RabbitMQ のバージョンとブローカーアップグレードの詳細については、「Amazon MQ for RabbitMQ エンジンバージョンの管理」を参照してください。

日付	ドキュメントの更新
2022 年 2 月 16 日	Amazon MQ が アフリカ (ケープタウン) リージョン で利用可能になりました。利用可能なリージョンの詳細については、AWS 全般リファレンスガイドの「AWS リージョンとエンドポイント」を参照してください。
2022 年 2 月 14 日	Amazon MQ for RabbitMQ が RabbitMQ version 3.9.13 をサポートするようになりました。マイナーバージョンの自動アップグレードは、Rabbit 3.8 から 3.9 へのアップグレードには使用できません。これを行うには、ブローカーを手動でアップグレードします。 RabbitMQ 3.9 で導入された新機能の詳細については、GitHub ウェブサイトのバージョン 3.9.0 のリリースノートページを参照してください。 **Note** 現在、Amazon MQ はストリーム、または RabbitMQ 3.9 で導入された JSON での構造化ロギングの使用はサポートしません。 このリリースでの修正と機能の詳細については、以下を参照してください。 • RabbitMQ サーバー GitHub リポジトリの RabbitMQ 3.9.13 リリースノート • RabbitMQ changelog サポートされる Amazon MQ for RabbitMQ のバージョンとブローカーアップグレードの詳細については、「Amazon MQ for RabbitMQ エンジンバージョンの管理」を参照してください。
2022 年 2 月 7 日	Amazon MQ for RabbitMQ では、新しいブローカーメトリクスが導入され、クラスターデプロイの 3 つのノードすべてで平均リソース使用率を監視できます。 詳細については次を参照してください: • the section called “RabbitMQ のメトリクス”

日付	ドキュメントの更新
2022 年 1 月 18 日	Amazon MQ for RabbitMQ が RabbitMQ バージョン 3.8.26 をサポートするようになりました。 このリリースでの修正と機能の詳細については、以下を参照してください。 • RabbitMQ サーバー GitHub リポジトリの RabbitMQ 3.8.26 リリースノート • RabbitMQ changelog サポートされる Amazon MQ for RabbitMQ のバージョンとブローカーアップグレードの詳細については、「Amazon MQ for RabbitMQ エンジンバージョンの管理」を参照してください。
2022 年 1 月 13 日	Amazon MQ では、ブローカーが高メモリアラームを発して異常な状態にあるときに通知するための RABBITMQ_MEMORY_ALARM ステータスコードが導入されました。Amazon MQ では、高メモリアラームの診断、解決、および防止に役立つ詳細情報と推奨事項が提供されています。詳細については、以下を参照してください。 • the section called “ RABBITMQ_MEMORY_ALARM ”
2022 年 1 月 6 日	Amazon MQ for ActiveMQ ブローカーの CloudWatch Logs を設定すると、Amazon MQ が IAM リソースベースのポリシーの aws:SourceArn および aws:SourceAccount グローバル条件コンテキストキーを使用して「混乱した代理」問題の防止をサポートします。詳細については、以下を参照してください。 • the section called “サービス間での不分別な代理処理の防止”
2021 年 12 月 20 日	Amazon MQ for ActiveMQ では、一連の新しいメトリクスが導入され、サポートされている各種トランスポートプロトコルを使用してブローカーに接続できる最大数をモニタリングできるようになりました。また、ブローカーのネットワークでブローカーに接続されているノードの数をモニタリングできる追加の新しいメトリクスも導入されています。詳細については、以下を参照してください。 • the section called “ActiveMQ のメトリクス”

日付	ドキュメントの更新
2021 年 11 月 16 日	Amazon MQ for RabbitMQ が RabbitMQ バージョン 3.8.23 をサポートするようになりました。 このリリースでの修正と機能の詳細については、以下を参照してください。 • RabbitMQ サーバー GitHub リポジトリの RabbitMQ 3.8.23 リリースノート • RabbitMQ changelog サポートされる Amazon MQ for RabbitMQ のバージョンとブローカーアップグレードの詳細については、「Amazon MQ for RabbitMQ エンジンバージョンの管理」を参照してください。
2021 年 10 月 12 日	Amazon MQ が、新しいマイナーエンジンバージョンのリリースである ActiveMQ 5.16.3 をサポートするようになりました。詳細については次を参照してください: • ActiveMQ 5.16.3 Release ページ • Amazon MQ for ActiveMQ エンジンバージョンの管理 • Amazon MQ ブローカーエンジンバージョンのアップグレード • Spring XML 設定ファイルの使用

日付	ドキュメントの更新
2021 年 9 月 8 日	Amazon MQ for RabbitMQ が RabbitMQ バージョン 3.8.22 をサポートするようになりました。 このリリースには、以前にサポートされていたバージョンの RabbitMQ 3.8.17 で特定された、メッセージごとの TTL (有効期限) を使用するキューの問題に対する修正が含まれます。既存のブローカーをバージョン 3.8.22 にアップグレードすることをお勧めします。 このリリースでの修正と機能の詳細については、以下を参照してください。 • RabbitMQ サーバー GitHub リポジトリの RabbitMQ 3.8.22 リリースノート • RabbitMQ changelog サポートされる Amazon MQ for RabbitMQ のバージョンとブローカーアップグレードの詳細については、「Amazon MQ for RabbitMQ エンジンバージョンの管理」を参照してください。
2021 年 8 月 25 日	Amazon MQ for RabbitMQ は、メッセージごとの有効期限 (TTL) を使用するキューで特定された問題のため、RabbitMQ エンジンバージョン 3.8.17 を一時的に無効化しました。バージョン 3.8.11 の使用をお勧めします。
2021 年 7 月 29 日	Amazon MQ for RabbitMQ が RabbitMQ バージョン 3.8.17 をサポートするようになりました。この更新に含まれる修正と機能の詳細については、以下を参照してください。 • RabbitMQ サーバー GitHub リポジトリの RabbitMQ 3.8.17 リリースノート • RabbitMQ changelog • Amazon MQ for RabbitMQ エンジンバージョンの管理
2021年7月16日	Amazon MQ ブローカーのメンテナンスウィンドウは AWS Management Console、、 AWS CLI、または Amazon MQ API を使用して調整できるようになりました。ブローカーのメンテナンスウィンドウの詳細については、以下を参照してください。 • Amazon MQ ブローカーのメンテナンスウィンドウのスケジュール

日付	ドキュメントの更新
2021 年 7 月 6 日	Amazon MQ for RabbitMQ がコンシステントハッシュエクスチェンジタイプのサポートを導入しました。コンシステントハッシュエクスチェンジは、メッセージのルーティングキーから計算されたハッシュ値に基づいてメッセージをキューに送信します。詳細については次を参照してください: • コンシステントハッシュエクスチェンジプラグイン • RabbitMQ GitHub リポジトリの RabbitMQ Consistent Hash Exchange Type
2021 年 6 月 7 日	Amazon MQ が、新しいメジャーエンジンバージョンのリリースである ActiveMQ 5.16.2 をサポートするようになりました。詳細については次を参照してください: • ActiveMQ 5.16.2 Release ページ • Amazon MQ for ActiveMQ エンジンバージョンの管理 • Amazon MQ ブローカーエンジンバージョンのアップグレード • Spring XML 設定ファイルの使用
2021 年 5 月 26 日	Amazon MQ for RabbitMQ が、中国 (北京) および中国 (寧夏) リージョンで利用可能になりました。利用可能なリージョンについては、AWS のリージョンとエンドポイントを参照してください。
2021 年 5 月 18 日	Amazon MQ for RabbitMQ がブローカーデフォルトを実装します。 ブローカーを初めて作成するときは、ブローカーのパフォーマンスを最適化するために、Amazon MQ が選択されたインスタンスタイプとブローカーデプロイモードに基づいて一連のブローカーポリシーと vhost 制限を作成します。詳細については次を参照してください: • Amazon MQ for RabbitMQ のブローカーデフォルト

日付	ドキュメントの更新
2021 年 5 月 5 日	Amazon MQ が ActiveMQ 5.15.15 をサポートするようになりました。詳細については次を参照してください: • ActiveMQ 5.15.15 Release ページ • Amazon MQ for ActiveMQ エンジンバージョンの管理 • Spring XML 設定ファイルの使用
2021 年 5 月 5 日	Amazon MQ は、 AWS 管理ポリシーの変更の追跡を開始しました。詳細については次を参照してください: • the section called “AWS マネージドポリシー”
2021 年 4 月 14 日	Amazon MQ が中国 (北京) および中国 (寧夏) リージョンで利用可能になりました。利用可能なリージョンについては、AWS のリージョンとエンドポイントを参照してください。
2021 年 4 月 7 日	Amazon MQ が RabbitMQ 3.8.11 をサポートするようになりました。この更新に含まれる修正と機能の詳細については、以下を参照してください。 • RabbitMQ サーバー GitHub リポジトリの RabbitMQ 3.8.11 リリースノート • RabbitMQ changelog • Amazon MQ for RabbitMQ エンジンバージョンの管理
2021 年 4 月 1 日	Amazon MQ がアジアパシフィック (大阪) リージョンで利用可能になりました。利用可能なリージョンについては、Amazon MQ のリージョンとエンドポイントを参照してください。

日付	ドキュメントの更新
2020 年 12 月 21 日	Amazon MQ が ActiveMQ 5.15.14 をサポートするようになりました。詳細については次を参照してください: • ActiveMQ 5.15.14 リリースノート • Amazon MQ for ActiveMQ エンジンバージョンの管理 • Spring XML 設定ファイルの使用 • **Important** このリリースでの既知の Apache ActiveMQ 問題のため、Amazon MQ for ActiveMQ では、ActiveMQ ウェブコンソールの新しい [Pause Queue] ボタンを使用できません。この問題の詳細については、「AMQ-8104」を参照してください。

日付	ドキュメントの更新
2020 年 11 月 4 日	Amazon MQ が、人気のあるオープンソースのメッセージブローカーである RabbitMQ をサポートするようになりました。これにより、コードを書き換え AWS ることなく、既存の RabbitMQ メッセージブローカーを に移行できます。 Amazon MQ for RabbitMQ は、個々のメッセージブローカーとクラスター化されたメッセージブローカーの両方を管理し、インフラストラクチャのプロビジョニング、ブローカーのセットアップ、およびソフトウェアの更新などのタスクを処理します。 • Amazon MQ は RabbitMQ 3.8.6 をサポートします。サポートされるエンジンバージョンの詳細については、「the section called “バージョン管理”」を参照してください。 • AWS 無料利用枠には、1 年間毎月最大 750 時間の単一インスタンス `mq.t3.micro` ブローカーと、最大 20GB のストレージが含まれています。サポートされているインスタンスタイプの詳細については、「Broker instance types」を参照してください。 • Amazon MQ for RabbitMQ では、AMQP 0-9-1、および RabbitMQ クライアントライブラリでサポートされる任意の言語を使用してブローカーにアクセスできます。サポートされるプロトコルと暗号化スイートの詳細については、「the section called “Amazon MQ for RabbitMQ のプロトコル”」を参照してください。 • RabbitMQ for Amazon MQ は、現在 Amazon MQ を利用できるすべてのリージョンでご利用いただけます。利用可能なすべてのリージョンの詳細については、「AWS リージョン表」を参照してください。 Amazon MQ の使用を開始し、ブローカーを作成して、JVM ベースのアプリケーションを RabbitMQ ブローカーに接続するには、「開始方法: RabbitMQ ブローカーの作成と接続」を参照してください。

日付	ドキュメントの更新
2020 年 10 月 22 日	Amazon MQ は ActiveMQ 5.15.13 をサポートします。詳細については次を参照してください: • ActiveMQ 5.15.13 リリースノート • Amazon MQ for ActiveMQ エンジンバージョンの管理 • Spring XML 設定ファイルの使用
2020 年 9 月 30 日	Amazon MQ が欧州 (ミラノ) リージョンで利用可能になりました。利用可能なリージョンについては、Amazon MQ のリージョンとエンドポイントを参照してください。
2020 年 7 月 27 日	Amazon MQ ユーザーは、アクティブディレクトリまたはその他の LDAP サーバーに保存されている認証情報を使用して認証することができます。Amazon MQ ユーザーの追加、削除、変更、およびトピックとキューへの許可の割り当てを行うことも可能です。詳細については、「LDAP を ActiveMQ に統合する」を参照してください。
2020 年 7 月 17 日	Amazon MQ が `mq.t3.micro` インスタンスタイプをサポートするようになりました。詳細については、「Broker instance types」を参照してください。
2020 年 6 月 30 日	Amazon MQ は ActiveMQ 5.15.12 をサポートします。詳細については次を参照してください: • ActiveMQ 5.15.12 リリースノート • Amazon MQ for ActiveMQ エンジンバージョンの管理 • Spring XML 設定ファイルの使用

日付	ドキュメントの更新
2020 年 4 月 30 日	Amazon MQ は、broker 要素の新しい子コレクション要素 systemUsage をサポートしています。詳細については、「systemUsage」を参照してください。 Amazon MQ は、kahaDB 子要素の 3 つの新しい属性もサポートします。 • journalDiskSyncInterval - journalDiskSyncStrategy=periodic の場合にディスク同期を実行する間隔 (ミリ秒)。 • journalDiskSyncStrategy - ディスク同期ポリシーを設定します。 • preallocationStrategy - 新しいジャーナルファイルが必要になったときにブローカーがジャーナルファイルの事前割り当てを試みる方法を設定します。 詳細については、「属性」を参照してください。
2020 年 3 月 3 日	Amazon MQ が 2 つの新しい CloudWatch メトリクスをサポート • TempPercentUsage - 非永続的メッセージで使用可能な一時ストレージの割合 (%)。 • JobSchedulerStorePercentUsage - ジョブスケジューラストアで使用するディスク領域の割合 (%)。 詳細については、「Monitoring and logging Amazon MQ brokers」を参照してください。
2020 年 2 月 4 日	Amazon MQ をアジアパシフィック (香港) および中東 (バーレーン) リージョンでご利用いただけます。利用可能なリージョンについては、AWS のリージョンとエンドポイントを参照してください。
2020 年 1 月 22 日	Amazon MQ は ActiveMQ 5.15.10 をサポートします。詳細については次を参照してください: • ActiveMQ 5.15.10 リリースノート • Amazon MQ for ActiveMQ エンジンバージョンの管理 • Spring XML 設定ファイルの使用

日付	ドキュメントの更新
2019 年 12 月 19 日	Amazon MQ を欧州 (ストックホルム) および南米 (サンパウロ) リージョンでご利用いただけます。利用可能なリージョンについては、AWS のリージョンとエンドポイントを参照してください。
2019 年 12 月 16 日	Amazon MQ は、デフォルトの Amazon Elastic File System (Amazon EFS) ではなく、ブローカーストレージ用の Amazon Elastic Block Store (EBS) を使用することによるスループット最適化ブローカーの作成をサポートします。複数のアベイラビリティーゾーン全体で優れた耐障害性とレプリケーションを活用するには、Amazon EFS を使用します。低レイテンシーと高スループットを活用するには、Amazon EBS を使用します。 **Important** • Amazon EBS を使用できるのは、`mq.m5` ブローカーインスタンスタイプファミリーのみです。 • ブローカーインスタンスタイプを変更することはできますが、ブローカーを作成した後でブローカーストレージタイプを変更することはできません。 • Amazon EBS は単一のアベイラビリティーゾーン内でデータをレプリケートし、ActiveMQ アクティブ/スタンバイデプロイモードをサポートしません。 詳細については次を参照してください: • Storage • 最高のスループットのために正しいブローカーストレージタイプを選択する • Amazon MQ REST API リファレンスの `broker-instance-options` リソースの `storageType` プロパティ • Monitoring and logging Amazon MQ brokers セクションの `BurstBalance` 、`VolumeReadOps` 、および `VolumeWriteOps` メトリクス。

日付	ドキュメントの更新
2019 年 10 月 18 日	`TotalEnqueueCount` および `TotalDequeueCount` の 2 つの Amazon CloudWatch メトリクスをご利用いただけます。詳細については、「Monitoring and logging Amazon MQ brokers」を参照してください。
2019 年 10 月 11 日	Amazon MQ が、米国商用リージョンで米国連邦情報処理規格 140-2 (FIPS) 準拠のエンドポイントをサポートするようになりました。 詳細については、以下を参照してください。 • 連邦情報処理規格 (FIPS) 140-2 • Amazon MQ のリージョンとエンドポイント
2019 年 9 月 30 日	Amazon MQ に、ホストインスタンスタイプを変更してブローカーをスケーリングする機能が組み込まれました。詳細については、`UpdateBrokerInput` の `hostInstanceType` プロパティおよび `DescribeBrokerOutput` の `pendingHostInstanceType` プロパティを参照してください。
2019 年 8 月 30 日	コンソールと `UpdateBrokerInput` の両方で、ブローカーに関連付けられたセキュリティグループを更新できるようになりました。
2019 年 7 月 22 日	Amazon MQ は AWS Key Management Service (KMS) と統合して、サーバー側の暗号化を提供します。独自のカスタマーマネージド CMK を選択するか、 AWS KMS アカウントで AWS マネージド KMS キーを使用できるようになりました。詳細については、「保管中の暗号化」を参照してください。 Amazon MQ は、次の方法で AWS KMS キーの使用をサポートしています。 • AWS 所有の KMS キー — キーは Amazon MQ を所有しており、アカウントにはありません。 • AWS マネージド KMS キー — AWS マネージド KMS キー (aws/mq) は、Amazon MQ によってユーザーに代わって作成、管理、使用されるアカウントの KMS キーです。 • 既存のカスタマーマネージド CMK を選択する – カスタマーマネージド CMK は、 AWS Key Management Service (KMS) でユーザーが作成し、管理します。

日付	ドキュメントの更新
2019 年 6 月 19 日	Amazon MQ を欧州 (パリ) およびアジアパシフィック (ムンバイ) リージョンでご利用いただけます。利用可能なリージョンについては、AWS のリージョンとエンドポイントを参照してください。
2019 年 6 月 12 日	Amazon MQ をカナダ (中部) リージョンでご利用いただけます。利用可能なリージョンについては、AWS のリージョンとエンドポイントを参照してください。
2019 年 6 月 3 日	`EstablishedConnectionsCount` および `InactiveDurableSubscribers` の 2 つの新しい Amazon CloudWatch メトリクスをご利用いただけます。詳細については次を参照してください: • Monitoring and logging Amazon MQ brokers • Monitoring and logging Amazon MQ brokers
2019 年 5 月 10 日	新しい `mq.t2.micro` インスタンスタイプのデータストレージが 20 GB に制限されました。詳細については次を参照してください: • the section called “データストレージ” • Broker instance types
2019 年 4 月 29 日	タグベースのポリシーとリソースレベルのアクセス権限を使用できるようになりました。詳細については次を参照してください: • Amazon MQ で IAM が機能する仕組み • Amazon MQ API アクションに対するリソースレベルの許可
2019 年 4 月 16 日	REST API を使用して、ブローカーエンジンとブローカーインスタンスのオプションに関する情報を取得できるようになりました。詳細については次を参照してください: • ブローカーインスタンスのオプション • ブローカーエンジンタイプ

日付	ドキュメントの更新
2019 年 4 月 8 日	Amazon MQ は ActiveMQ 5.15.9 をサポートします。詳細については次を参照してください: • ActiveMQ 5.15.9 リリースノート • Amazon MQ for ActiveMQ エンジンバージョンの管理 • Spring XML 設定ファイルの使用
2019 年 3 月 4 日	動的なフェイルオーバーの設定と、ブローカーのネットワークのクライアントの再分散のため、ドキュメントを改善しました。transportConnectors と networkConnectors 設定オプションを設定することにより、動的なフェイルオーバーを有効にします。詳細については次を参照してください: • トランスポートコネクタを使用した動的なフェイルオーバー • Amazon MQ のブローカーのネットワーク • Amazon MQ Broker Configuration Parameters
2019 年 2 月 27 日	Amazon MQ は、以下のリージョンに加えて、欧州 (ロンドン) リージョンでもご利用いただけます。 • アジアパシフィック (シンガポール) • 米国東部(オハイオ) • 米国東部 (バージニア北部) • 米国西部 (北カリフォルニア) • 米国西部 (オレゴン) • アジアパシフィック (東京) • アジアパシフィック (ソウル) • アジアパシフィック (シドニー) • 欧州 (フランクフルト) • 欧州 (アイルランド)
2019 年 1 月 24 日	デフォルト設定に、非アクティブな送信先を消去するポリシーが含まれるようになりました。

日付	ドキュメントの更新
2019 年 1 月 17 日	Amazon MQ `mq.t2.micro` インスタンスタイプが、ワイヤレベルプロトコルあたり 100 個の接続のみをサポートするようになりました。詳細については、「Quotas in Amazon MQ」を参照してください。
2018 年 12 月 19 日	ブローカーのネットワークで一連の Amazon MQ ブローカーを設定できます。詳細については、次のセクションを参照してください。 • Amazon MQ のブローカーのネットワーク • Creating and Configuring a Network of Brokers • ブローカーのネットワークを正しく設定する • `networkConnector` • `networkConnectionStartAsync`

2018 年 12 月 11 日	Amazon MQ は ActiveMQ 5.15.8、5.15.6、および 5.15.0 をサポートします。
	• 解決されたバグと ActiveMQ の改善点。
	  • ActiveMQ 5.15.8 リリースノート
	  • ActiveMQ 5.15.7 リリースノート

2018 年 12 月 5 日	AWS は、コスト配分の追跡に役立つリソースのタグ付けをサポートしています。リソースを作成するとき、またはそのリソースの詳細を表示することによって、リソースにタグを付けることができます。詳細については、「リソースにタグを付ける」を参照してください。
2018 年 11 月 19 日	AWS は、SOC コンプライアンスプログラムを拡張し、SOC 準拠サービスとして Amazon MQ を含めました。
2018 年 10 月 15 日	• ユーザーあたりのグループの最大数は 20 です。詳細については、「[ユーザー]」を参照してください。 • 接続の最大数は、ブローカーあたり、ワイヤレベルプロトコルあたり 1,000 です。詳細については、「ブローカー」を参照してください。
2018 年 10 月 2 日	AWS は、HIPAA コンプライアンスプログラムを拡張し、Amazon MQ を HIPAA 対応サービスとして含めました。

日付	ドキュメントの更新

2018 年 9 月 27 日	ActiveMQ 5.15.0 に加えて、Amazon MQ が 5.15.6 をサポートします。詳細については次を参照してください:
	• 開始方法: ActiveMQ ブローカーの作成と接続
	• 解決されたバグと ActiveMQ ドキュメントの改善点。
	  • ActiveMQ 5.15.6 リリースノート
	  • ActiveMQ 5.15.5 リリースノート
	  • ActiveMQ 5.15.4 リリースノート
	  • ActiveMQ 5.15.3 リリースノート
	  • ActiveMQ 5.15.2 リリースノート
	  • ActiveMQ 5.15.1 リリースノート
	• ActiveMQ Client 5.15.6

2018 年 8 月 31 日	• 以下のメトリクスが利用可能です。
	  • `CurrentConnectionsCount`
	  • `TotalConsumerCount`
	  • `TotalProducerCount`
	  詳細については「Monitoring and logging Amazon MQ brokers」セクションを参照してください。
	• また、ブローカーの IP アドレスが [詳細] ページに表示されます。
	  Note
	  パブリックアクセシビリティが無効なブローカーの場合、内部 IP アドレスが表示されます。

日付	ドキュメントの更新
2018 年 8 月 30 日	Amazon MQ は、以下のリージョンに加えて、アジアパシフィック (シンガポール) リージョンでもご利用いただけます。 • 米国東部(オハイオ) • 米国東部 (バージニア北部) • 米国西部 (北カリフォルニア) • 米国西部 (オレゴン) • アジアパシフィック (東京) • アジアパシフィック (ソウル) • アジアパシフィック (シドニー) • 欧州 (フランクフルト) • 欧州 (アイルランド)
2018 年 7 月 30 日	一般ログと監査ログを Amazon CloudWatch Logs に発行するように Amazon MQ を設定できます。詳細については、「Monitoring and logging Amazon MQ brokers」を参照してください。
2018 年 7 月 25 日	Amazon MQ は、以下のリージョンに加えて、アジアパシフィック (東京) およびアジアパシフィック (ソウル) リージョンでもご利用いただけます。 • 米国東部(オハイオ) • 米国東部 (バージニア北部) • 米国西部 (北カリフォルニア) • 米国西部 (オレゴン) • アジアパシフィック (シドニー) • 欧州 (フランクフルト) • 欧州 (アイルランド)
2018 年 7 月 19 日	を使用して Amazon MQ API コール AWS CloudTrail を記録できます。詳細については、「Logging Amazon MQ API calls using CloudTrail」を参照してください。

日付	ドキュメントの更新
2018 年 6 月 29 日	mq.t2.micro および mq.m4.large に加えて、次のブローカーインスタンスタイプが一般的な開発、テスト、および高度なスループットが必要なプロダクションワークロードに利用できます。 • mq.m5.large • mq.m5.xlarge • mq.m5.2xlarge • mq.m5.4xlarge 詳細については、「Broker instance types」を参照してください。
2018 年 6 月 27 日	Amazon MQ は、以下のリージョンに加えて、米国西部 (北カリフォルニア) リージョンでもご利用いただけます。 • 米国東部(オハイオ) • 米国東部 (バージニア北部) • 米国西部 (オレゴン) • アジアパシフィック (シドニー) • 欧州 (フランクフルト) • 欧州 (アイルランド)

日付	ドキュメントの更新

2018 年 6 月 14 日	• AWS::Amazon MQ::Broker AWS CloudFormation リソースを使用して、次のアクションを実行できます。
	  • ブローカーの作成。
	  • 指定されたブローカーの設定の変更またはユーザーの変更。
	  • 指定されたブローカーに関する情報の戻し。
	  • 指定されたブローカーの削除。
	
	  Note
	  Amazon MQ の Broker ConfigurationId または Amazon MQ の Broker User プロパティタイプのプロパティを変更すると、ブローカーは直ちに再起動されます。
	• AWS::Amazon MQ::Configuration AWS CloudFormation リソースを使用して、次のアクションを実行できます。
	  • 設定の作成。
	  • 指定された構成の更新。
	  • 指定された設定に関する情報の戻し。
	
	  Note
	  を使用して AWS CloudFormation 、Amazon MQ 設定を変更できますが、削除はできません。

2018 年 6 月 7 日	Amazon MQ コンソールは、ドイツ語、ポルトガル語 (ブラジル)、スペイン語、イタリア語、および繁体字中国語をサポートします。
2018 年 5 月 17 日	ブローカーあたりのユーザー数の制限は 250 です。詳細については、「[ユーザー]」を参照してください。
2018 年 3 月 13 日	ブローカーの作成には約 15 分かかります。詳細については、「ブローカー作成の完了」を参照してください。

日付	ドキュメントの更新
2018 年 3 月 1 日	• ??? 属性を使用して Apache KahaDB の concurrentStoreAndDispatchQueues 同時保存とディスパッチを設定できます。 • mq.t2.micro ブローカーインスタンスタイプに CpuCreditBalance CloudWatch メトリクスを利用できます。
2018 年 1 月 10 日	以下の変更は Amazon MQ コンソールに影響を及ぼします。 • ブローカーのリストで、[Creation (作成)] 列はデフォルトで非表示になります。ページサイズと列をカスタマイズするには、  。 • [*MyBroker*] ページの [Connections] (接続) セクションでセキュリティグループの名前または  を選択すると、(VPC コンソールではなく) EC2 コンソールが開きます。EC2 コンソールでは、インバウンドおよびアウトバウンドルールのより直感的な設定ができます。詳細については、更新された「Connecting a Java application to your broker」セクションを参照してください。
2018 年 1 月 9 日	• REST オペレーション ID UpdateBroker の許可が、IAM コンソールで mq:UpdateBroker として正しく表示されるようになりました。 • 誤った mq:DescribeEngine 許可は IAM コンソールから削除されました。

日付	ドキュメントの更新

2017 年 11 月 28 日	これは、Amazon MQ と Amazon MQ デベロッパーガイドの初回リリースです。
	• Amazon MQ は、以下のリージョンでご利用いただけます。
	  • 米国東部(オハイオ)
	  • 米国東部 (バージニア北部)
	  • 米国西部 (オレゴン)
	  • アジアパシフィック (シドニー)
	  • 欧州 (フランクフルト)
	  • 欧州 (アイルランド)
	  `mq.t2.micro` インスタンスタイプの使用は CPU クレジットとベースラインパフォーマンスの対象となり、ベースラインレベルを超えてバーストする機能を備えています (詳細については、`CpuCreditBalance` メトリクスを参照してください)。アプリケーションに一定のパフォーマンスが必要な場合は、`mq.m5.large` インスタンスタイプの使用を検討してください。
	• `mq.m4.large` および `mq.t2.micro` ブローカーを作成できます。
	  `mq.t2.micro` インスタンスタイプの使用は CPU クレジットとベースラインパフォーマンスの対象となり、ベースラインレベルを超えてバーストする機能を備えています (詳細については、`CpuCreditBalance` メトリクスを参照してください)。アプリケーションに一定のパフォーマンスが必要な場合は、`mq.m5.large` インスタンスタイプの使用を検討してください。
	• ActiveMQ 5.15.0 ブローカーエンジンを使用できます。
	• Amazon MQ REST API と AWS SDKs を使用して、プログラムでブローカーを作成および管理することもできます。
	• ブローカーには、ActiveMQ がサポートする任意のプログラミング言語を使用し、以下のプロトコルに対して TLS を明示的に有効にすることによってアクセスできます。
	  • AMQP
	  • MQTT

日付	ドキュメントの更新
	• MQTT over WebSocket • OpenWire • STOMP • STOMP over WebSocket • ActiveMQ ブローカーには、さまざまな ActiveMQ クライアントを使用して接続できます。ActiveMQ クライアントを使用することをお勧めします。詳細については、「Connecting a Java application to your broker」を参照してください。 • ブローカーは任意のサイズのメッセージを送受信できます。

翻訳は機械翻訳により提供されています。提供された翻訳内容と英語版の間で齟齬、不一致または矛盾がある場合、英語版が優先します。