aws

개발자 가이드

Amazon MQ

Amazon MQ: 개발자 가이드

Table of Contents

Amazon MQ란 무엇인가요?

Amazon MQ는 Apache ActiveMQ Classic 및 RabbitMQ용 관리형 메시지 브로커 서비스로, 메시지 브로커의 설정, 운영 및 유지 관리를 담당합니다. 업계 표준 메시징 프로토콜을 사용하여 새 Amazon MQ 브로커를 생성하거나, 메시징 코드를 다시 작성하지 않고 기존 메시지 브로커를 Amazon MQ로 마이그레이션할 수 있습니다.

브로커는 Amazon MQ에서 실행하는 메시지 브로커 환경입니다. 이 인스턴스는 Amazon MQ의 기본 빌딩 블록입니다. 메시지 브로커를 사용하면 소프트웨어 애플리케이션 및 구성 요소가 다양한 프로그래밍 언어, 운영 체제 및 공식 메시징 프로토콜을 사용하여 통신할 수 있습니다. 대규모 클라우드 네이티브 애플리케이션과 구성 요소 간의 통신에 Amazon MQ 브로커를 사용할 수 있습니다.

주제

- Amazon MQ 기능
- Amazon MQ를 사용하려 하는데 처음에 어떻게 시작해야 하나요?
- Amazon MQ에 피드백을 제공하려면 어떻게 해야 하나요?

Amazon MQ 기능

관리형 유지 관리 및 버전 업그레이드

Amazon MQ는 예약된 유지 관리 기간 동안 메시지 브로커에 대한 유지 관리 및 버전 업그레이드를 수행합니다.

CloudWatch를 사용한 브로커 모니터링

Amazon MQ는 Amazon CloudWatch와 통합되어 있으므로 브로커 및 대기열에 대한 지표를 확인하고 분석할 수 있습니다. Amazon MQ 콘솔, CloudWatch 콘솔, 명령줄 및 API에서 지표를 확인하고 분석할 수 있습니다. 지표는 1분마다 자동으로 수집되어 CloudWatch로 푸시됩니다.

보안.

Amazon MQ는 메시지의 저장 중 및 전송 중 암호화를 제공합니다. 브로커에 대한 연결에는 SSL이 사용되며, 액세스는 Amazon VPC 내의 프라이빗 엔드포인트로 제한될 수 있습니다. 또한 AWS Identity and Access Management(IAM)을 사용하여 IAM 사용자 및 그룹이 특정 Amazon MQ 브로커에 대해 수행할 수 있는 작업을 제어할 수 있습니다.

Amazon MQ 기반 RabbitMQ의 쿼럼 대기열

쿼럼 대기열은 리더 노드(기본 복제본)와 팔로워 노드(기타 복제본)로 구성된 복제 대기열 유형입니다. 각 노드는 서로 다른 가용 영역에 있으므로 한 노드를 일시적으로 사용할 수 없는 경우에도 다른 가용 영역에서 새로 선출된 리더 복제본을 통해 메시지 전송이 계속됩니다. 쿼럼 대기열은 메시지가 실패하여 여러 번 대기열에 추가될 때 발생하는 유해 메시지를 처리하는 데 유용합니다.

Amazon MQ 기반 ActiveMQ의 리전 간 데이터 복제

리전 간 데이터 복제(CRDR)를 사용하면 기본 AWS 리전의 기본 브로커에서 복제본 리전의 복제본 브로커로 비동기 메시지를 복제할 수 있습니다. Amazon MQ API에 대한 장애 조치 요청을 실행하면 현재 복제본 브로커가 기본 브로커 역할로 승격되고 현재 기본 브로커는 복제본 역할로 강등됩니다.

Amazon MQ를 사용하려 하는데 처음에 어떻게 시작해야 하나요?

Amazon MQ 기반 ActiveMQ를 시작하려면 다음 설명서를 검토하세요.

- 시작하기: ActiveMQ 브로커 생성 및 연결
- the section called “브로커 배포”
- ActiveMQ 자습서
- the section called “ActiveMQ용 Amazon MQ 모범 사례”

Amazon MQ 기반 RabbitMQ를 시작하려면 다음 설명서를 검토하세요.

- 시작하기: RabbitMQ 브로커 생성 및 연결
- the section called “RabbitMQ 브로커 배포”
- the section called “RabbitMQ 자습서”
- the section called “RabbitMQ용 Amazon MQ 모범 사례”

Amazon MQ REST API에 대한 자세한 내용은 Amazon MQ REST API 참조를 참조하세요.

Amazon MQ AWS CLI 명령에 대한 자세한 내용은 AWS CLI 명령 참조의 Amazon MQ를 참조하세요.

Amazon MQ에 피드백을 제공하려면 어떻게 해야 하나요?

설명서에 대한 여러분의 피드백을 환영하고 장려합니다. 오른쪽의 좋아요 및 싫어요 아이콘을 사용하여 피드백을 제출하거나 아래에 링크된 "피드백 제공" 양식을 사용할 수 있습니다.

Amazon MQ 팀에 연락하려면 Amazon MQ 토론 포럼을 사용하세요.

Amazon MQ 설정

Amazon MQ를 사용하려면 먼저 다음 단계를 완료해야 합니다.

주제

1단계: 사전 조건

에 가입 AWS 계정

이 없는 경우 다음 단계를 AWS 계정완료하여 생성합니다.

에 가입하려면 AWS 계정

1. https://portal.aws.amazon.com/billing/signup을 엽니다.
2. 온라인 지시 사항을 따르세요.

   등록 절차 중 전화 또는 텍스트 메시지를 받고 전화 키패드로 확인 코드를 입력하는 과정이 있습니다.

   에 가입하면 AWS 계정AWS 계정 루트 사용자이 생성됩니다. 루트 사용자에게는 계정의 모든 AWS 서비스 및 리소스에 액세스할 권한이 있습니다. 보안 모범 사례는 사용자에게 관리 액세스 권한을 할당하고, 루트 사용자만 사용하여 루트 사용자 액세스 권한이 필요한 작업을 수행하는 것입니다.

AWS 는 가입 프로세스가 완료된 후 확인 이메일을 보냅니다. 언제든지 https://aws.amazon.com/으로 이동하고 내 계정을 선택하여 현재 계정 활동을 확인하고 계정을 관리할 수 있습니다.

관리자 액세스 권한이 있는 사용자 생성

에 가입한 후 일상적인 작업에 루트 사용자를 사용하지 않도록 관리 사용자를 AWS 계정보호 AWS IAM Identity Center, AWS 계정 루트 사용자활성화 및 생성합니다.

보안 AWS 계정 루트 사용자

1. 루트 사용자를 선택하고 AWS 계정 이메일 주소를 입력하여 계정 소유자AWS Management Console로에 로그인합니다. 다음 페이지에서 비밀번호를 입력합니다.

   루트 사용자를 사용하여 로그인하는 데 도움이 필요하면 AWS 로그인 사용 설명서의 루트 사용자로 로그인을 참조하세요.
2. 루트 사용자의 다중 인증(MFA)을 활성화합니다.

   지침은 IAM 사용 설명서의 AWS 계정 루트 사용자(콘솔)에 대한 가상 MFA 디바이스 활성화를 참조하세요.

관리자 액세스 권한이 있는 사용자 생성

1. IAM Identity Center를 활성화합니다.

   지침은 AWS IAM Identity Center 사용 설명서의 AWS IAM Identity Center설정을 참조하세요.
2. IAM Identity Center에서 사용자에게 관리 액세스 권한을 부여합니다.

   를 자격 증명 소스 IAM Identity Center 디렉터리 로 사용하는 방법에 대한 자습서는 사용 AWS IAM Identity Center 설명서의 기본값으로 사용자 액세스 구성을 IAM Identity Center 디렉터리 참조하세요.

관리 액세스 권한이 있는 사용자로 로그인

- IAM IDentity Center 사용자로 로그인하려면 IAM Identity Center 사용자를 생성할 때 이메일 주소로 전송된 로그인 URL을 사용합니다.

  IAM Identity Center 사용자를 사용하여 로그인하는 데 도움이 필요하면 사용 설명서의 AWS 액세스 포털에 로그인을 참조하세요. AWS 로그인

추가 사용자에게 액세스 권한 할당

1. IAM Identity Center에서 최소 권한 적용 모범 사례를 따르는 권한 세트를 생성합니다.

   지침은AWS IAM Identity Center 사용 설명서의 Create a permission set를 참조하세요.
2. 사용자를 그룹에 할당하고, 그룹에 Single Sign-On 액세스 권한을 할당합니다.

   지침은 AWS IAM Identity Center 사용 설명서의 Add groups를 참조하세요.

2단계: 사용자 생성 및 자격 AWS 증명 가져오기

사용자는 AWS 외부에서와 상호 작용하려는 경우 프로그래밍 방식으로 액세스해야 합니다 AWS Management Console. 프로그래밍 방식 액세스를 부여하는 방법은에 액세스하는 사용자 유형에 따라 다릅니다 AWS.

사용자에게 프로그래밍 방식 액세스 권한을 부여하려면 다음 옵션 중 하나를 선택합니다.

프로그래밍 방식 액세스가 필요한 사용자	목적	방법
IAM	(권장) 콘솔 자격 증명을 임시 자격 증명으로 사용하여 AWS CLI, AWS SDKs 또는 AWS APIs.	사용하고자 하는 인터페이스에 대한 지침을 따릅니다. • 자세한 AWS CLI내용은 AWS Command Line Interface 사용 설명서의 AWS 로컬 개발을 위한 로그인을 참조하세요. • AWS SDKs 경우 SDK 및 도구 참조 안내서의 AWS 로컬 개발을 위한 로그인을 참조하세요. AWS SDKs
작업 인력 ID (IAM Identity Center에서 관리되는 사용자)	임시 자격 증명을 사용하여 AWS CLI, AWS SDKs 또는 AWS APIs.	사용하고자 하는 인터페이스에 대한 지침을 따릅니다. • 자세한 AWS CLI내용은 AWS Command Line Interface 사용 설명서의 AWS CLI 를 사용하도록 구성을 AWS IAM Identity Center 참조하세요. • AWS SDKs, 도구 및 AWS APIs 경우 SDK 및 도구 참조 안내서의 IAM Identity

프로그래밍 방식 액세스가 필요한 사용자	목적	방법
		Center 인증을 참조하세요. AWS SDKs
IAM	임시 자격 증명을 사용하여 AWS CLI, AWS SDKs 또는 AWS APIs.	IAM 사용 설명서의 AWS 리소스에서 임시 자격 증명 사용의 지침을 따릅니다.
IAM	(권장되지 않음) 장기 자격 증명을 사용하여 AWS CLI, AWS SDKs 또는 AWS APIs.	사용하고자 하는 인터페이스에 대한 지침을 따릅니다. • 자세한 AWS CLI내용은 AWS Command Line Interface 사용 설명서의 IAM 사용자 자격 증명을 사용한 인증을 참조하세요. • AWS SDKs 및 도구의 경우 SDK 및 도구 참조 안내서의 장기 자격 증명을 사용하여 인증을 참조하세요. AWS SDKs • AWS APIs 경우 IAM 사용 설명서의 IAM 사용자의 액세스 키 관리를 참조하세요.

3단계: 예제 코드 사용 준비

다음 자습서에서는를 사용하여 Amazon MQ 브로커를 사용하는 방법과 ActiveMQ Amazon MQ 및 RabbitMQ용 Amazon MQ 브로커에 프로그래밍 방식으로 연결하는 AWS Management Console 방법을 보여줍니다. ActiveMQ Java 예제 코드를 사용하려면 Java Standard Edition Development Kit를 설치하고 코드를 일부 수정해야 합니다.

Amazon MQ REST API 및 AWS SDKs.

다음 단계

이제 Amazon MQ를 사용할 준비가 되었으므로 브로커를 생성하여 시작합니다. 브로커 엔진 유형에 따라 Java 애플리케이션을 ActiveMQ용 Amazon MQ 브로커에 연결하거나 RabbitMQ Java 클라이언트 라이브러리를 사용하여 JVM 기반 애플리케이션을 RabbitMQ용 Amazon MQ 브로커에 연결할 수 있습니다.

시작하기: ActiveMQ 브로커 생성 및 연결

브로커는 Amazon MQ에서 실행하는 메시지 브로커 환경입니다. 이 인스턴스는 Amazon MQ의 기본 빌딩 블록입니다. 브로커 인스턴스 클래스(m5) 및 크기(large, medium)의 설명 조합은 브로커 인스턴스 유형(예: mq.m5.large)입니다. 자세한 내용은 ActiveMQ용 Amazon MQ 브로커란? 섹션을 참조하세요.

ActiveMQ 브로커 생성

가장 먼저 이루어지고 가장 흔한 Amazon MQ 태스크는 브로커를 생성하는 것입니다. 다음 예제에서는 AWS Management Console을 사용하여 기본 브로커를 생성하는 방법을 보여줍니다.

1. Amazon MQ 콘솔에 로그인합니다.
2. Select broker engine(브로커 엔진 선택) 페이지에서 Apache ActiveMQ를 선택합니다.
3. Select deployment and storage(배포 및 스토리지 선택) 페이지의 Deployment mode and storage type(배포 모드 및 스토리지 유형) 섹션에서 다음을 수행합니다.

   a. Deployment mode(배포 모드)를 선택합니다(예: Active/standby broker(활성/대기 브로커)). 자세한 정보는 ActiveMQ용 Amazon MQ 브로커의 배포 옵션을 참조하세요.

      - 단일 인스턴스 브로커는 하나의 가용 영역에 있는 하나의 브로커로 구성됩니다. 브로커는 애플리케이션 및 Amazon EBS 또는 Amazon EFS 스토리지 볼륨과 통신합니다. 자세한 정보는 옵션 1: Amazon MQ 단일 인스턴스 브로커을 참조하세요.
      - 고가용성을 위한 활성/대기 브로커는 두 개의 서로 다른 가용 영역에 있는 두 개의 브로커가 중복 페어로 구성됩니다. 이러한 브로커는 애플리케이션 및 Amazon EFS와 동기식으로 통신합니다. 자세한 내용은 옵션 2: 고가용성을 위한 Amazon MQ 활성/대기 브로커 섹션을 참조하세요.

   b. 스토리지 유형을 선택합니다(예: EBS). 자세한 정보는 Storage을 참조하세요.

      > **Note**
      >
      > Amazon EBS는 단일 가용 영역 내에서 데이터를 복제하며 ActiveMQ 활성/대기 배포 모드를 지원하지 않습니다.

   c. 다음을 선택합니다.
4. Configure settings(설정 구성) 페이지의 세부 정보 섹션에서 다음을 선택합니다.

a. Broker name(브로커 이름)을 입력합니다.

> **Important**
>
> 개인 식별 정보(PII)나 기타 기밀 정보 또는 민감한 정보를 브로커 이름에 추가하지 마십시오. 브로커 이름을 통해 CloudWatch Logs를 포함하여 다른 AWS 서비스에 액세스할 수 있습니다. 브로커 이름은 개인 데이터나 민감한 데이터에 사용하기 위한 것이 아닙니다.

> **Note**
>
> 추가 설정 섹션에서 다음을 구성할 수도 있습니다.
>
> - 구성
> - CloudWatch 로그
> - 프라이빗 액세스
> - 브로커 유지 관리 기간

b. Broker instance type(브로커 인스턴스 유형)을 선택합니다(예: mq.m5.large). 자세한 정보는 Broker instance types을 참조하세요.

5. ActiveMQ Web Console access(ActiveMQ 웹 콘솔 액세스) 섹션에서 Username(사용자 이름) 및 Password(암호)를 입력합니다. 브로커 사용자 이름과 암호에는 다음 제한이 적용됩니다.

   - 사용자 이름은 영숫자, 대시, 마침표, 밑줄 및 물결 기호(- . _ ~)만 포함할 수 있습니다.
   - 암호는 최소 12자 길이이고 최소 4개의 고유 문자가 있어야 하며 쉼표, 콜론 또는 등호(,:=)는 포함할 수 없습니다.

   > **Important**
   >
   > 개인 식별 정보(PII)나 기타 기밀 정보 또는 민감한 정보를 브로커 사용자 이름에 추가하지 마십시오. 브로커 사용자 이름을 통해 CloudWatch Logs를 포함하여 다른 AWS 서비스에 액세스할 수 있습니다. 브로커 사용자 이름은 개인 데이터나 민감한 데이터에 사용하기 위한 것이 아닙니다.

6. 배포(Deploy)를 선택합니다.

Amazon MQ에서 브로커를 생성하는 동안 Creation in progress(생성 진행 중) 상태가 표시됩니다.

브로커 생성은 약 15분 정도 소요됩니다.

브로커가 생성되면 Amazon MQ에서 Running(실행 중) 상태가 표시됩니다.

7. ***MyBroker***를 선택합니다.

   MyBroker 페이지의 연결(Connect) 섹션에서 브로커의 ActiveMQ 웹 콘솔 URL을 확인합니다. 예를 들면 다음과 같습니다.

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

   또한 브로커의 와이어 레벨 프로토콜 엔드포인트도 기록합니다. 다음은 OpenWire 엔드포인트의 예입니다.

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

시작하기: RabbitMQ 브로커 생성 및 연결

브로커는 Amazon MQ에서 실행하는 메시지 브로커 환경입니다. 이 인스턴스는 Amazon MQ의 기본 빌딩 블록입니다. 브로커 인스턴스 클래스(m5) 및 크기(large, medium)의 설명 조합은 브로커 인스턴스 유형(예: mq.m5.large)입니다. 자세한 내용은 RabbitMQ용 Amazon MQ 브로커란? 섹션을 참조하세요.

RabbitMQ 브로커 생성

가장 먼저 이루어지고 가장 흔한 Amazon MQ 태스크는 브로커를 생성하는 것입니다. 다음 예제에서는 AWS Management Console을 사용하여 기본 브로커를 생성하는 방법을 보여줍니다.

RabbitMQ용 Amazon MQ 브로커를 생성할 때 RabbitMQ의 브로커 설정 모범 사례를 따라 브로커 성능을 극대화하고 메시지 처리량 효율성을 최적화합니다.

1. Amazon MQ 콘솔에 로그인합니다.
2. Select broker engine(브로커 엔진 선택) 페이지에서 RabbitMQ를 선택한 후 Next(다음)를 선택합니다.
3. Select deployment mode(배포 모드 선택) 페이지에서 Deployment mode(배포 모드)(예: Cluster deployment(클러스터 배포))를 선택한 후 Next(다음)를 선택합니다.
   - 단일 인스턴스 브로커는 Network Load Balancer (NLB) 뒤에서 하나의 가용 영역에 있는 하나의 브로커로 구성됩니다. 브로커는 애플리케이션 및 Amazon EBS 스토리지 볼륨과 통신합니다. 자세한 정보는 옵션 1: RabbitMQ용 Amazon MQ 단일 인스턴스 브로커을 참조하세요.
   - 고가용성을 위한 RabbitMQ 클러스터 배포는 Network Load Balancer 뒤에 있는 3개의 RabbitMQ 브로커 노드(각각 사용자, 대기열 및 여러 가용 영역(AZ) 간에 분산된 상태 공유)로 이루어진 논리적 그룹입니다. 자세한 정보는 옵션 2: RabbitMQ용 Amazon MQ 클러스터 배포을 참조하세요.
4. Configure settings(설정 구성) 페이지의 Details(세부 정보) 섹션에서 다음을 수행합니다.
   a. Broker name(브로커 이름)을 입력합니다.

> **Important**
>
> 개인 식별 정보(PII)나 기타 기밀 정보 또는 민감한 정보를 브로커 이름에 추가하지 마십시오. 브로커 이름을 통해 CloudWatch Logs를 포함하여 다른 AWS 서비스에 액세

스할 수 있습니다. 브로커 이름은 개인 데이터나 민감한 데이터에 사용하기 위한 것이 아닙니다.

b. Broker instance type(브로커 인스턴스 유형)을 선택합니다(예: mq.m7g.large). 자세한 내용은 Broker instance types 섹션을 참조하세요.

5. Configure settings(설정 구성) 페이지의 RabbitMQ access(RabbitMQ 액세스) 섹션에서 Username(사용자 이름)과 Password(암호)를 입력합니다. 브로커 로그인 보안 인증 정보에 다음 제한이 적용됩니다.

- 사용자 이름은 영숫자, 대시, 마침표 및 밑줄(- . _)만 포함할 수 있습니다. 이 값에 물결표(~) 문자를 포함하면 안 됩니다. Amazon MQ에서는 `guest`를 사용자 이름으로 사용할 수 없습니다.
- 암호는 최소 12자 길이이고 최소 4개의 고유 문자가 있어야 하며 쉼표, 콜론 또는 등호(,:=)는 포함할 수 없습니다.

> **Important**
>
> 개인 식별 정보(PII)나 기타 기밀 정보 또는 민감한 정보를 브로커 사용자 이름에 추가하지 마십시오. 브로커 사용자 이름을 통해 CloudWatch Logs를 포함하여 다른 AWS 서비스에 액세스할 수 있습니다. 브로커 사용자 이름은 개인 데이터나 민감한 데이터에 사용하기 위한 것이 아닙니다.

> **Note**
>
> 추가 설정 섹션에서 다음을 구성할 수도 있습니다.
>
> - 구성
> - CloudWatch 로그
> - 프라이빗 액세스
> - 브로커 유지 관리 기간

6. 다음을 선택합니다.
7. Review and create(검토 및 생성) 페이지에서 선택 항목을 확인하고 필요한 경우 편집합니다.
8. Create broker(브로커 생성)를 선택합니다.

Amazon MQ에서 브로커를 생성하는 동안 Creation in progress(생성 진행 중) 상태가 표시됩니다.

브로커 생성은 약 15분 정도 소요됩니다.

브로커가 생성되면 Amazon MQ에서 Running(실행 중) 상태가 표시됩니다.

9. ***MyBroker***를 선택합니다.

MyBroker 페이지의 Connect(연결) 섹션에서 브로커의 RabbitMQ 웹 콘솔 URL을 기록합니다. 예를 들면 다음과 같습니다.

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

또한 브로커의 보안 AMQP 엔드포인트도 기록합니다. 다음은 리스너 포트 5671을 노출하는 amqps 엔드포인트 예제입니다.

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

Amazon MQ 브로커 관리

브로커를 생성한 후에는 Amazon MQ 브로커의 다양한 구성 요소를 관리하고 유지할 수 있습니다.

주제

Amazon MQ에 연결

서비스 엔드포인트 및 브로커 엔드포인트를 사용하여 다른 AWS 서비스에서 Amazon MQ에 연결할 수 있습니다.

서비스 엔드포인트

Amazon MQ 서비스 API에는 다음 연결 메서드가 사용됩니다.

도메인	연결 방법
`mq.region.amazonaws.com`	IPv4
`mq.region.api.aws`	이중 스택(IPv4 및 IPv6)
`mq-fips.region.amazonaws.com`	IPv4 전용 FIPS

도메인	연결 방법
`mq-fips.region.api.aws`	듀얼 스택을 사용하는 FIPS

브로커 엔드포인트

Amazon MQ 브로커에는 다음 연결 메서드가 사용됩니다.

도메인	연결 방법
`brokerId.mq.region.amazonaws.com`	IPv4
`brokerId.mq.region.on.aws`	이중 스택(IPv4 및 IPv6) **Note** ActiveMQ용 Amazon MQ 브로커는 듀얼 스택을 지원하지 않습니다.

듀얼 스택(IPv4 및 IPv6) 엔드포인트를 사용하여 Amazon MQ에 연결

이중 스택 엔드포인트는 IPv4 트래픽과 IPv6 트래픽을 모두 지원합니다. 듀얼 스택 엔드포인트에 요청하는 경우, 엔드포인트 URL이 IPv4 또는 IPv6 주소로 확인됩니다. 자세한 내용은 SDK 참조 가이드의 듀얼 스택 및 FIPS 엔드포인트를 참조하세요.

Amazon MQ는 리전별 듀얼 스택 엔드포인트를 지원합니다. 따라서 엔드포인트 이름의 일부로 AWS 리전을 지정해야 합니다. 듀얼 스택 엔드포인트 이름에는 `mq.region.api.aws` 명명 규칙이 사용됩니다. 예를 들어 `eu-west-1` 리전의 이중 스택 엔드포인트 이름은 `mq.eu-west-1.api.aws`입니다.

Amazon MQ 엔드포인트의 전체 목록은 AWS 일반 참조를 참조하세요.

AWS PrivateLink를 사용하여 Amazon MQ에 연결

IPv4 및 IPv6를 지원하는 Amazon MQ API용 AWS PrivateLink 엔드포인트는 트래픽을 퍼블릭 인터넷에 노출하지 않고도 가상 프라이빗 클라우드(VPC)와 Amazon MQ API 간에 프라이빗 연결을 제공합니다.

> **Note**
>
> PrivateLink에 대한 지원은 브로커 엔드포인트가 아닌 Amazon MQ API 엔드포인트에서만 사용할 수 있습니다. 브로커 엔드포인트에 비공개로 연결하는 방법에 대한 자세한 내용은 Configuring a private Amazon MQ broker 섹션을 참조하세요.

PrivateLink를 사용하여 Amazon MQ API에 액세스하려면 먼저 연결하려는 특정 VPC에서 인터페이스 VPC 엔드포인트를 생성해야 합니다. VPC 엔드포인트를 생성할 때 FIPS 엔드포인트에 서비스 이름 `com.amazonaws.region.mq` 또는 `com.amazonaws.region.mq-fips`를 사용합니다.

AWS CLI 또는 SDK를 사용하여 Amazon MQ를 호출할 때는 듀얼 스택 도메인 이름 `mq.region.api.aws` 또는 `mq-fips.region.api.aws`를 사용할 엔드포인트 URL을 지정해야 합니다. Amazon MQ용 PrivateLink는 `amazonaws.com`으로 끝나는 기본 도메인 이름을 지원하지 않습니다. 자세한 내용은 SDK 참조 가이드의 듀얼 스택 및 FIPS 엔드포인트를 참조하세요.

다음 CLI 예제에서는 Amazon MQ VPC 엔드포인트를 통해 아시아 태평양(시드니) 리전에서 `describe-broker-engine-type`을 호출하는 방법을 보여줍니다.

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

CLI에서 엔드포인트를 구성하는 다른 방법은 AWS CLI의 엔드포인트 사용을 참조하세요.

VPC 엔드포인트 정책을 사용하여 VPC 엔드포인트에 대한 사용자 액세스를 확인할 수도 있습니다. 자세한 정보는 엔드포인트 정책을 사용하여 VPC 엔드포인트에 대한 액세스 제어를 참조하세요.

Amazon MQ 브로커에 대한 인증 및 권한 부여

Amazon MQ는 조직의 요구 사항에 따라 메시징 인프라를 보호하기 위해 여러 인증 및 권한 부여 방법을 제공합니다.

RabbitMQ용 Amazon MQ에 대한 인증 및 권한 부여

RabbitMQ용 Amazon MQ는 다음과 같은 인증 및 권한 부여 방법을 지원합니다.

간편한 인증 및 권한 부여

이 방법에서 브로커 사용자는 RabbitMQ 브로커에 내부적으로 저장되고 웹 콘솔 또는 관리 API를 통해 관리됩니다. vhost, Exchange, 대기열 및 주제에 대한 권한은 RabbitMQ에서 직접 구성됩니다. 기본 메서드입니다. 자세한 내용은 단순 인증 및 권한 부여를 참조하세요.

OAuth 2.0 인증 및 권한 부여

이 메서드에서 브로커 사용자와 해당 권한은 외부 OAuth 2.0 ID 제공업체(IdP)에서 관리합니다. vhost, Exchange, 대기열 및 주제에 대한 사용자 인증 및 리소스 권한은 OAuth 2.0 공급자의 범위 시스템을 통해 중앙 집중화됩니다. 이를 통해 사용자 관리를 간소화하고 기존 자격 증명 시스템과 통합할 수 있습니다. 자세한 내용은 OAuth 2.0 인증 및 권한 부여를 참조하세요.

IAM 인증 및 권한 부여

이 방법에서 브로커 사용자는 AWS IAM 아웃바운드 페더레이션을 통해 IAM 자격 증명을 사용하여 인증합니다. IAM 자격 증명은 AWS Security Token Service(STS)에서 JWT 토큰을 얻는 데 사용되며, 이러한 JWT 토큰은 인증을 위한 OAuth 2.0 토큰 역할을 합니다. 이 방법은 RabbitMQ용 Amazon MQ에서 기존 OAuth 2.0 지원을 활용합니다. 여기서 AWS 는 OAuth 2.0 자격 증명 공급자 역할을 합니다. 사용자 인증은 AWS IAM에서 처리되는 반면, vhost, Exchange, 대기열 및 주제에 대한 리소스 권한은 RabbitMQ에 구성된 IAM 정책 및 범위 별칭을 통해 관리됩니다. 자세한 내용은 IAM 인증 및 권한 부여를 참조하세요.

LDAP 인증 및 권한 부여

이 방법에서는 브로커 사용자와 해당 권한이 외부 LDAP 디렉터리 서비스에 의해 관리됩니다. 사용자 인증 및 리소스 권한은 LDAP 서버를 통해 중앙 집중화되므로 사용자는 기존 디렉터리 서비스 자격 증명을 사용하여 RabbitMQ에 액세스할 수 있습니다. 자세한 내용은 LDAP 인증 및 권한 부여를 참조하세요.

HTTP 인증 및 권한 부여

이 방법에서는 브로커 사용자와 해당 권한이 외부 HTTP 서버에서 관리됩니다. 사용자 인증 및 리소스 권한은 HTTP 서버를 통해 중앙 집중화되므로 사용자는 자체 인증 및 권한 부여 공급자를 사용하여 RabbitMQ에 액세스할 수 있습니다. 이 방법에 대한 자세한 내용은 HTTP 인증 및 권한 부여를 참조하세요.

SSL 인증서 인증

Amazon MQ는 RabbitMQ 브로커에 대해 상호 TLS(mTLS)를 지원합니다. SSL 인증 플러그인은 mTLS 연결의 클라이언트 인증서를 사용하여 사용자를 인증합니다. 이 방법에서 브로커 사용자는 사용자 이

름 및 암호 자격 증명 대신 X.509 클라이언트 인증서를 사용하여 인증됩니다. 클라이언트의 인증서는 신뢰할 수 있는 인증 기관(CA)에 대해 검증되고 사용자 이름은 일반 이름(CN) 또는 주체 대체 이름(SAN)과 같은 인증서의 필드에서 추출됩니다. 이 방법은 네트워크를 통해 자격 증명을 전송하지 않고 강력한 인증을 제공합니다. 자세한 내용은 SSL 인증서 인증을 참조하세요.

> **Note**
>
> RabbitMQ는 동시에 사용할 수 있는 여러 인증 및 권한 부여 방법을 지원합니다. 예를 들어 OAuth 2.0과 단순(내부) 인증을 모두 활성화할 수 있습니다. 자세한 내용은 OAuth 2.0 및 단순(내부) 인증 활성화에 대한 OAuth 2.0 자습서 섹션과 RabbitMQ 액세스 제어 설명서를 참조하세요.
>
> Amazon MQ는 인증 구성을 테스트할 때 내부 사용자를 생성할 것을 권장합니다. 이렇게 하면 RabbitMQ 관리 API를 사용하여 액세스 구성을 검증할 수 있습니다. 자세한 내용은 액세스 검증을 참조하세요.

ActiveMQ용 Amazon MQ에 대한 인증 및 권한 부여

ActiveMQ용 Amazon MQ는 다음과 같은 인증 및 권한 부여 방법을 지원합니다.

간편한 인증 및 권한 부여

이 메서드에서 브로커 사용자는 Amazon MQ 콘솔 또는 API를 통해 생성되고 관리됩니다. 대기열, 주제 및 ActiveMQ 웹 콘솔에 액세스할 수 있는 특정 권한으로 사용자를 구성할 수 있습니다. 이 방법에 대한 자세한 내용은 ActiveMQ 브로커 사용자 생성을 참조하세요.

LDAP 인증 및 권한 부여

이 방법에서 브로커 사용자는 LDAP 서버에 저장된 자격 증명을 통해 인증합니다. LDAP 서버를 통해 사용자를 추가, 삭제 및 수정하고 주제 및 대기열에 권한을 할당하여 중앙 집중식 인증 및 권한 부여를 제공할 수 있습니다. 이 방법에 대한 자세한 내용은 ActiveMQ 브로커와 LDAP 통합을 참조하세요.

Amazon MQ 브로커 엔진 버전 업그레이드

Amazon MQ는 지원되는 모든 브로커 엔진 유형에 대해 정기적으로 새 브로커 엔진 버전을 제공합니다. 새 엔진 버전에는 보안 패치, 버그 수정 및 기타 브로커 엔진 개선 사항이 포함되어 있습니다.

Amazon MQ는 의미 체계 버전 관리 사양에 따라 버전 번호를 `X.Y.Z`로 구성합니다. Amazon MQ 구현에서 `X`는 메이저 버전, `Y`는 마이너 버전, `Z`는 패치 버전 번호를 나타냅니다. Amazon MQ는 두 가지 유형의 업그레이드를 지원합니다.

- 메이저 버전 업그레이드 - 메이저 엔진 버전 번호가 변경되면 발생합니다. 예를 들어 RabbitMQ 버전 3.13에서 버전 4.2로 업그레이드하는 것은 메이저 버전 업그레이드로 간주됩니다.
- 마이너 버전 업그레이드 - 마이너 엔진 버전 번호가 변경되면 발생합니다. 예를 들어 버전 3.11에서 버전 3.12로 업그레이드하는 것은 마이너 버전 업그레이드로 간주됩니다.

언제든지 브로커를 지원되는 다음 메이저 또는 마이너 버전으로 수동으로 업그레이드할 수 있습니다. Amazon MQ는 예약된 유지 관리 기간 동안 모든 브로커에 대해 지원되는 최신 패치 버전으로의 업그레이드를 관리합니다. 수동 및 자동 버전 업그레이드는 모두 예약된 유지 관리 기간 동안 또는 브로커를 재부팅한 후에 수행됩니다. 현재 마이너 버전이 지원 종료 시점에 도달하면 Amazon MQ는 브로커를 다음 마이너 버전으로 업그레이드합니다.

엔진 버전 수동 업그레이드

AWS Management Console AWS CLI, 또는 Amazon MQ API를 사용하여 브로커의 엔진 버전을 업그레이드할 수 있습니다.

AWS Management Console

를 사용하여 브로커의 엔진 버전을 업그레이드하려면 AWS Management Console

1. 브로커 세부 정보 페이지에서 Edit(편집)을 선택합니다.
2. Specifications(사양) 아래에서 Broker engine version(브로커 엔진 버전)의 드롭다운 목록에서 새 버전 번호를 선택합니다.
3. 페이지 하단으로 스크롤하고 Schedule modifications(수정 예약)를 선택합니다.
4. Schedule broker modifications(브로커 수정 예약) 페이지의 When to apply modifications(수정을 적용할 시기)에서 다음 중 하나를 선택합니다.
   - Amazon MQ가 예약된 다음 유지 관리 기준 중 버전 업그레이드를 완료하도록 하려면 After the next reboot(다음 재부팅 후)를 선택합니다.
   - 브로커를 재부팅하고 엔진 버전을 즉시 업그레이드하려면 Immediately(즉시)를 선택합니다.

> **Important**
>
> 단일 인스턴스 브로커는 재부팅되는 동안 오프라인 상태입니다. 클러스터 브로커의 경우 브로커가 재부팅되는 동안 한 번에 하나의 노드만 다운됩니다.

5. Apply(적용)를 선택하여 변경 사항 적용을 완료합니다.

AWS CLI

를 사용하여 브로커의 엔진 버전을 업그레이드하려면 AWS CLI

1. 다음 예제와 같이 update-broker CLI 명령을 사용하여 다음 파라미터를 지정합니다.

   - `--broker-id` - Amazon MQ가 브로커에 대해 생성하는 고유한 ID입니다. 브로커 ARN에서 ID를 구문 분석할 수 있습니다. 예를 들어 ARN이 `arn:aws:mq:us-east-2:123456789012:broker:MyBroker:b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9`인 경우 브로커 ID는 `b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9`입니다.
   - `--engine-version` - 업그레이드할 브로커 엔진의 엔진 버전 번호입니다.

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

2. (선택 사항) 엔진 버전을 즉시 업그레이드하려면 reboot-broker CLI 명령을 사용하여 브로커를 재부팅합니다.

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

   브로커를 재부팅하여 변경 사항을 즉시 적용하지 않으려는 경우 Amazon MQ가 예약된 다음 유지 관리 기간 중에 브로커를 업그레이드합니다.

> **Important**
>
> 단일 인스턴스 브로커는 재부팅되는 동안 오프라인 상태입니다. 클러스터 브로커의 경우 브로커가 재부팅되는 동안 한 번에 하나의 노드만 다운됩니다.

Amazon MQ API

Amazon MQ API를 사용하여 브로커 엔진 버전을 업그레이드하려면

1. UpdateBroker API 작업을 사용합니다. `broker-id`를 경로 파라미터로 지정합니다. 다음 예제에서는 브로커가 `us-west-2` 리전에 있다고 가정합니다. 사용할 수 있는 Amazon MQ 엔드포인트에 대한 자세한 내용은 AWS 일반 참조의 Amazon MQ 엔드포인트 및 할당량을 참조하세요.

   ```
   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
   ```

   요청 페이로드에서 `engineVersion`을 사용하여 업그레이드할 브로커의 버전 번호를 지정합니다.

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

2. (선택 사항) 엔진 버전을 즉시 업그레이드하려면 RebootBroker API 작업을 사용하여 브로커를 재부팅합니다. `broker-id`는 경로 파라미터로 지정됩니다.

   ```
   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
   ```

   브로커를 재부팅하여 변경 사항을 즉시 적용하지 않으려는 경우 Amazon MQ가 예약된 다음 유지 관리 기간 중에 브로커를 업그레이드합니다.

   > **Important**
   >
   > 단일 인스턴스 브로커는 재부팅되는 동안 오프라인 상태입니다. 클러스터 브로커의 경우 브로커가 재부팅되는 동안 한 번에 하나의 노드만 다운됩니다.

Amazon MQ 브로커 인스턴스 유형 업그레이드

> **Important**
>
> `mq.m7g.x` 인스턴스는 RabbitMQ용 Amazon MQ 브로커에서만 사용할 수 있습니다. ActiveMQ용 Amazon MQ 브로커는 `mq.m5.x` 인스턴스만 사용합니다.

브로커 인스턴스 클래스(`m7g`) 및 크기(`large`)의 설명 조합은 브로커 인스턴스 유형(예: `mq.m7g.large`)입니다. 인스턴스 유형을 선택할 때는 브로커 성능에 영향을 미치는 다음과 같은 요인을 고려하는 것이 중요합니다.

- 클라이언트 및 대기열 수
- 전송된 메시지의 양
- 메모리에 보관된 메시지
- 중복 메시지

작은 브로커 인스턴스 유형(`mq.m7g.medium`)은 애플리케이션 성능을 테스트할 때만 사용하는 것이 좋습니다. 프로덕션 수준의 클라이언트 및 대기열, 높은 처리량, 메모리 내 메시지, 중복 메시지에 대해서는 더 큰 브로커 인스턴스 유형(`mq.m7g.large` 이상)을 권장합니다.

성능 문제가 발생하거나 테스트에서 프로덕션 환경으로 이동하는 경우 더 큰 인스턴스 유형(예: `micro`에서 `large`로)으로 업그레이드하는 것이 좋습니다. 인스턴스 유형을 업그레이드하려면 AWS Management Console, AWS CLI 또는 Amazon MQ API를 사용할 수 있습니다.

AWS Management Console

AWS Management Console을 사용하여 더 큰 인스턴스 유형으로 업그레이드하려면 다음을 수행합니다.

1. Amazon MQ 콘솔에 로그인합니다.
2. 왼쪽 탐색 창에서 브로커를 선택한 다음 목록에서 을 업그레이드할 브로커를 선택합니다.
3. 브로커 세부 정보 페이지에서 Edit(편집)을 선택합니다.
4. 사양의 브로커 인스턴스 유형에서 드롭다운 목록의 새 인스턴스 유형을 선택합니다.
5. 페이지 하단으로 스크롤하고 Schedule modifications(수정 예약)를 선택합니다.

6. Schedule broker modifications(브로커 수정 예약) 페이지의 When to apply modifications(수정을 적용할 시기)에서 다음 중 하나를 선택합니다.

   - Amazon MQ가 예약된 다음 유지 관리 기간 중 버전 업그레이드를 완료하도록 하려면 After the next reboot(다음 재부팅 후)를 선택합니다.
   - 브로커를 재부팅하고 인스턴스 유형을 즉시 업그레이드하려면 Immediately(즉시)를 선택합니다.

   > **Important**
   >
   > 단일 인스턴스 브로커는 재부팅되는 동안 오프라인 상태입니다. 클러스터 브로커의 경우 브로커가 재부팅되는 동안 한 번에 하나의 노드만 다운됩니다.

7. Apply(적용)를 선택하여 변경 사항 적용을 완료합니다.

AWS CLI

AWS CLI를 사용하여 브로커의 인스턴스 유형을 업그레이드하려면

1. 다음 예제와 같이 modify-broker CLI 명령을 사용하여 다음 파라미터를 지정합니다.

   - `--broker-id` - Amazon MQ가 브로커에 대해 생성하는 고유한 ID입니다.
   - `--host-instance-type` - 업그레이드할 브로커 엔진의 엔진 버전 번호입니다.

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

2. (선택 사항) 인스턴스 유형을 즉시 업그레이드하려는 경우 reboot-broker CLI 명령을 사용하여 브로커를 재부팅합니다.

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

   브로커를 재부팅하여 변경 사항을 즉시 적용하지 않으려는 경우 Amazon MQ가 예약된 다음 유지 관리 기간 중에 브로커를 업그레이드합니다.

> **Important**
>
> 단일 인스턴스 브로커는 재부팅되는 동안 오프라인 상태입니다. 클러스터 브로커의 경우 브로커가 재부팅되는 동안 한 번에 하나의 노드만 다운됩니다.

Amazon MQ API

Amazon MQ API를 사용하여 브로커의 인스턴스 유형을 업그레이드하려면

1. UpdateBroker API 작업을 사용합니다. `broker-id`를 경로 파라미터로 지정합니다. 다음 예제에서는 브로커가 `us-west-2` 리전에 있다고 가정합니다. 사용할 수 있는 Amazon MQ 엔드포인트에 대한 자세한 내용은 AWS 일반 참조의 Amazon MQ 엔드포인트 및 할당량을 참조하세요.

   ```
   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
   ```

   요청 페이로드에서 `host-instance-type`을 사용하여 업그레이드할 브로커의 인스턴스 유형을 지정합니다.

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

2. (선택 사항) 엔진 버전을 즉시 업그레이드하려면 RebootBroker API 작업을 사용하여 브로커를 재부팅합니다. `broker-id`는 경로 파라미터로 지정됩니다.

   ```
   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
   ```

   브로커를 재부팅하여 변경 사항을 즉시 적용하지 않으려는 경우 Amazon MQ가 예약된 다음 유지 관리 기간 중에 브로커를 업그레이드합니다.

> **Important**
>
> 단일 인스턴스 브로커는 재부팅되는 동안 오프라인 상태입니다. 클러스터 브로커의 경우 브로커가 재부팅되는 동안 한 번에 하나의 노드만 다운됩니다.

ActiveMQ용 Amazon MQ 스토리지 유형

ActiveMQ용 Amazon MQ는 Amazon Elastic File System(EFS) 및 Amazon Elastic Block Store(EBS)를 지원합니다. 기본적으로 ActiveMQ 브로커는 브로커 스토리지로 Amazon EFS를 사용합니다. 다중 가용 영역에 걸친 높은 내구성 및 복제를 활용하려면 Amazon EFS를 사용하세요. 짧은 지연 시간 및 높은 처리량을 활용하려면 Amazon EBS를 사용세요.

> **Important**
>
> - Amazon EBS는 `mq.m5` 브로커 인스턴스 유형 패밀리에서만 사용할 수 있습니다.
> - 브로커를 생성한 후에는 브로커 인스턴스 유형은 변경할 수 있지만, 브로커 스토리지 유형은 변경할 수 없습니다.
> - Amazon EBS는 단일 가용 영역 내에서 데이터를 복제하며 ActiveMQ 활성/대기 배포 모드를 지원하지 않습니다.

스토리지 유형 간 차이점

다음 표에서는 인 메모리, Amazon EFS 및 Amazon EBS 스토리지 유형 간의 차이점에 대해 간략하게 설명합니다.

스토리지 유형	Persistence	사용 사례	생산자당, 초당 대기열에 추가되는 대략적인 최대 메시지 수(1KB 메시지)	복제
인 메모리	비영구	• 주식 시세 • 위치 데이터 업데이트	5,000	없음

스토리지 유형	Persistence	사용 사례	생산자당, 초당 대기열에 추가되는 대략적인 최대 메시지 수(1KB 메시지)	복제
		• 자주 변경되는 데이터		
Amazon EBS	지속적	• 대량의 텍스트 • 주문 처리	500	단일 가용 영역(AZ) 내의 여러 복사본
Amazon EFS	지속적	금융 트랜잭션	80	여러 AZ에 걸쳐 있는 여러 개의 복사본

인 메모리 메시지 스토리지는 지연 시간이 가장 짧고 처리량이 가장 많습니다. 그러나 인스턴스 교체 또는 브로커 재시작 중에 메시지가 손실됩니다.

Amazon EFS는 단일 구성 요소의 실패 또는 AZ의 가용성에 영향을 미치는 문제로 인한 데이터 손실을 방지하기 위해 여러 AZ에 걸쳐 복제되어 높은 내구성을 제공하도록 설계되었습니다. Amazon EBS는 처리량에 맞게 최적화되어 단일 AZ 내의 여러 서버에 복제됩니다.

프라이빗 Amazon MQ 브로커 구성

프라이빗 브로커는 퍼블릭 액세스 권한이 없으며 VPC 외부에서 액세스할 수 없습니다. 프라이빗 브로커를 구성하기 전에 VPC, 서브넷 및 보안 그룹에 대한 다음 정보를 확인합니다.

- VPC
  - 브로커의 서브넷과 보안 그룹은 동일한 VPC에 있어야 합니다.
  - 프라이빗 브로커를 사용하는 경우 VPC로 구성하지 않은 IP 주소가 표시될 수 있습니다. 이는 Amazon MQ 인프라에서 가져온 IP 주소이므로 작업이 필요하지 않습니다.
- 서브넷
  - 서브넷이 공유 VPC 내에 있는 경우 VPC는 브로커를 생성하는 동일한 계정이 소유해야 합니다.
  - 서브넷이 제공되지 않으면 기본 VPC의 기본 서브넷이 사용됩니다.

  - 브로커가 생성되면 사용된 서브넷을 변경할 수 없습니다.
  - 클러스터 및 활성/대기 브로커의 경우 서브넷이 서로 다른 가용 영역에 있어야 합니다.
  - 단일 인스턴스 브로커의 경우 사용할 서브넷을 지정할 수 있으며 브로커는 동일한 가용 영역 내에 생성됩니다.
- 보안 그룹
  - 보안 그룹이 제공되지 않으면 기본 VPC의 기본 보안 그룹이 사용됩니다.
  - 단일 인스턴스 브로커, 클러스터 브로커 및 활성/대기 브로커에 모두 하나 이상의 보안 그룹(예: 기본 보안 그룹)이 필요합니다.

    > **Note**
    >
    > 퍼블릭 RabbitMQ 브로커는 서브넷 또는 보안 그룹을 사용하지 않습니다.

  - 브로커가 생성되면 사용된 보안 그룹을 변경할 수 없습니다. 보안 그룹 자체는 계속 수정할 수 있습니다.

에서 프라이빗 브로커 구성 AWS Management Console

프라이빗 브로커를 구성하려면 AWS Management Console에서 새 브로커 생성을 시작합니다. 그런 다음 네트워크 설정 섹션에서 브로커의 연결을 구성하려면 다음을 수행합니다.

1. 브로커에 대한 프라이빗 액세스를 선택합니다. 프라이빗 브로커에 연결하려면 IPv4, IPv6 또는 듀얼 스택(IPv4 및 IPv6)을 사용할 수 있습니다. 자세한 내용은 Connecting to Amazon MQ 단원을 참조하십시오.
2. 그런 다음 기본 VPC, 서브넷 및 보안 그룹 사용을 선택하거나 기존 VPC, 서브넷 및 보안 그룹 선택을 선택합니다. 기본 또는 기존 VPC, 서브넷 또는 보안 그룹을 사용하지 않으려면 프라이빗 브로커에 연결할 새 VPC를 생성해야 합니다.

   > **Note**
   >
   > 프라이빗 브로커 액세스의 경우 연결 방법은 서브넷의 선택한 IP 유형과 동일합니다. 브로커가 생성되면 VPC 엔드포인트는 변경할 수 없으며 항상 선택한 서브넷의 IP 유형을 갖습니다. 새 IP 유형을 사용하려면 새 브로커를 생성해야 합니다.

> **Note**
>
> ActiveMQ용 Amazon MQ는 VPC 엔드포인트를 사용하지 않습니다. ActiveMQ 브로커를 처음 생성할 때 Amazon MQ는 VPC에 탄력적 네트워크 인터페이스(ENI)를 프로비저닝합니다. 보안 그룹은 ENI에 배치되며 퍼블릭 브로커와 프라이빗 브로커 모두에 사용할 수 있습니다.

퍼블릭 액세스 가능성이 없이 Amazon MQ 브로커 웹 콘솔에 액세스

브로커에 대한 퍼블릭 액세스를 끄면 브로커를 생성한 AWS 계정 ID가 프라이빗 브로커에 액세스할 수 있습니다. 브로커의 퍼블릭 액세스 가능성을 끈 경우 브로커 웹 콘솔에 액세스하려면 다음 단계를 수행해야 합니다.

1. `public-vpc`에 Linux EC2 인스턴스를 생성합니다(필요한 경우 퍼블릭 IP 사용).
2. VPC가 올바르게 구성되어 있는지 확인하려면 EC2 인스턴스에 대한 `ssh` 연결을 설정하고 해당 브로커의 URI와 함께 `curl` 명령을 사용합니다.
3. 머신에서 프라이빗 키 파일의 경로 및 퍼블릭 EC2 인스턴스의 IP 주소를 사용하여 EC2 인스턴스에 대한 `ssh` 터널을 생성합니다. 예:

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

   머신에서 전달 프록시 서버가 시작됩니다.
4. 머신에 FoxyProxy와 같은 프록시 클라이언트를 설치합니다.
5. 다음 설정을 사용하여 프록시 클라이언트를 구성합니다.

   - 프록시 유형에 대해 `SOCKS5`를 지정합니다.
   - IP 주소, DNS 이름 및 서버 이름에 대해 `localhost`를 지정합니다.
   - 포트에 대해 `8080`을 지정합니다.
   - 기존 URL 패턴을 제거합니다.
   - URL 패턴에 대해 `*.mq.*.amazonaws.com*`을 지정합니다.
   - 연결 유형에 대해 `HTTP(S)`를 지정합니다.

   프록시 클라이언트를 활성화하면, 머신에서 웹 콘솔에 액세스할 수 있습니다.

> **Important**
>
> 프라이빗 브로커를 사용하는 경우 VPC로 구성하지 않은 IP 주소가 표시될 수 있습니다. 이는 Amazon MQ 기반 RabbitMQ 인프라에서 가져온 IP 주소이므로 작업이 필요하지 않습니다.

Amazon MQ 브로커의 유지 관리 기간 예약

Amazon MQ는 유지 관리 기간 동안 주기적으로 메시지 브로커의 하드웨어, 운영 체제 또는 엔진 소프트웨어에 대한 유지 관리를 수행합니다. 예를 들어 브로커 인스턴스 유형을 변경한 경우 Amazon MQ에서는 예약된 다음 유지 관리 기간 동안 변경 사항을 적용합니다. 유지 관리 기간은 메시지 브로커에 대해 예약된 작업에 따라 최대 2시간까지 지속될 수 있습니다. 여러 가용 영역(AZ)에서 고가용성을 제공하는 브로커 배포 모드를 선택하면 유지보수 기간 동안 가동 중지 시간을 최소화할 수 있습니다.

ActiveMQ용 Amazon MQ는 고가용성을 위해 활성/대기 배포를 제공합니다. 활성/대기 모드에서 Amazon MQ는 유지 관리 작업을 한 번에 한 인스턴스씩 수행하므로 적어도 하나의 인스턴스가 사용 가능한 상태로 유지됩니다. 또한 유지 관리 기간을 주중에 다양하게 설정하여 브로커 네트워크를 구성할 수도 있습니다. RabbitMQ용 Amazon MQ는 고가용성을 위해 클러스터 배포를 제공합니다. 클러스터 배포에서 Amazon MQ는 유지 관리 작업을 한 번에 한 노드씩 수행하므로 두 개 이상의 노드가 항상 실행 중인 상태로 유지됩니다.

브로커를 처음 생성할 때 유지 관리 기간이 지정된 시간에 주 1회 발생하도록 예약할 수 있습니다. 브로커의 유지 관리 기간은 예약된 다음 유지 관리 기간 이전에 네 번까지만 조정할 수 있습니다. 브로커 유지 관리 기간이 완료되면 Amazon MQ에서 한도를 재설정하므로 다음 유지 관리 기간이 발생하기 전에 일정을 다시 조정할 수 있습니다. 브로커 유지 관리 기간을 조정할 때 브로커 가용성은 영향을 받지 않습니다.

브로커 유지 관리 기간을 조정하려면 AWS Management Console, AWS CLI 또는 Amazon MQ API를 사용할 수 있습니다.

AWS Management Console을 사용하여 브로커 유지 관리 기간 예약

AWS Management Console을 사용하여 브로커 유지 관리 기간을 조정하려면

1. Amazon MQ 콘솔에 로그인합니다.
2. 왼쪽 탐색 창에서 브로커를 선택한 다음 목록에서 을 업그레이드할 브로커를 선택합니다.
3. 브로커 세부 정보 페이지에서 Edit(편집)을 선택합니다.
4. Maintenance(유지 관리)에서 다음을 수행합니다.

a. 시작 날짜(Start day)의 드롭다운 목록에서 요일(예: 일요일(Sunday))을 선택합니다.

b. 시작 시간(Start time)에서 다음 브로커 유지 관리 기간으로 예약할 시간과 분을 선택합니다(예:12:00).

> **Note**
>
> Start time(시작 시간) 옵션은 UTC+0 표준 시간대로 구성됩니다.

5. 수정 예약을 선택합니다. 그런 후 다음 재부팅 후 또는 즉시를 선택합니다. 다음 재부팅 후를 선택하면 브로커를 재부팅하지 않고 유지 관리 기간이 즉시 업데이트됩니다. 즉시를 선택하면 브로커가 즉시 재부팅됩니다.
6. 브로커 세부 정보 페이지의 유지 관리 기간(Maintenance window) 아래에 새 기본 설정 일정이 표시되는지 확인합니다.

AWS CLI을 사용하여 브로커 유지 관리 기간 예약

AWS CLI를 사용하여 브로커 유지 관리 기간을 조정하려면

1. 다음 예제와 같이 update-broker CLI 명령을 사용하여 다음 파라미터를 지정합니다.

   - `--broker-id` - Amazon MQ가 브로커에 대해 생성하는 고유한 ID입니다. 브로커 ARN에서 ID를 구문 분석할 수 있습니다. 예를 들어 ARN이 `arn:aws:mq:us-east-2:123456789012:broker:MyBroker:b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9`인 경우 브로커 ID는 `b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9`입니다.
   - `--maintenance-window-start-time` - 주별 유지 관리 기간 시작 시간을 결정하는 파라미터이며 다음 구조로 지정됩니다.
     - `DayOfWeek` - 요일입니다. 구문: `MONDAY| TUESDAY | WEDNESDAY | THURSDAY | FRIDAY | SATURDAY | SUNDAY`
     - `TimeOfDay` - 24시간 형식의 시간입니다.
     - `TimeZone` - (선택 사항) 국가/도시 또는 UTC 오프셋 형식의 시간대입니다. 기본적으로 UTC로 설정됩니다.

```
aws mq update-broker --broker-id broker-id \
--maintenance-window-start-time DayOfWeek=SUNDAY,TimeOfDay=13:00,TimeZone=America/Los_Angeles
```

2. (선택 사항) describe-broker CLI 명령을 실행하여 유지 관리 기간이 업데이트되었는지 확인합니다.

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

Amazon MQ API를 사용하여 브로커 유지 관리 기간 예약

Amazon MQ API를 사용하여 브로커 유지 관리 기간을 조정하려면

1. UpdateBroker API 작업을 사용합니다. `broker-id`를 경로 파라미터로 지정합니다. 다음 예제에서는 브로커가 `us-west-2` 리전에 있다고 가정합니다. 사용할 수 있는 Amazon MQ 엔드포인트에 대한 자세한 내용은 AWS 일반 참조의 Amazon MQ 엔드포인트 및 할당량을 참조하세요.

```
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
```

   요청 페이로드에서 `maintenanceWindowStartTime` 파라미터 및 `WeeklyStartTime` 리소스 유형을 사용합니다.

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

2. (선택 사항) DescribeBroker API 작업을 사용하여 유지 관리 기간이 업데이트되었는지 확인합니다. `broker-id`는 경로 파라미터로 지정됩니다.

```
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 브로커 재부팅

브로커에 새 구성을 적용하려면 브로커를 재부팅할 수 있습니다.

> **Note**
>
> 또한 ActiveMQ 브로커가 응답하지 않는 경우 재부팅하여 오류 상태에서 복구할 수 있습니다.

다음 예제에서는 AWS Management Console을 사용하여 Amazon MQ 브로커를 재부팅하는 방법을 보여줍니다.

Amazon MQ 브로커를 재부팅하려면

1. Amazon MQ 콘솔에 로그인합니다.
2. 브로커 목록에서 브로커 이름(예: MyBroker)을 선택합니다.
3. ***MyBroker*** 페이지에서 작업, Reboot broker(브로커 재부팅)를 선택합니다.

   > **Important**
   >
   > 단일 인스턴스 브로커는 재부팅되는 동안 오프라인 상태가 됩니다. 클러스터 브로커를 사용할 수 있지만 각 노드는 한 번에 하나씩 재부팅됩니다.

4. Reboot broker(브로커 재부팅) 대화 상자에서 재부팅을 선택합니다.

   브로커 재부팅은 약 5분 정도 소요됩니다. 재부팅이 인스턴스 크기 변경을 포함하거나 대기열 깊이가 큰 브로커에서 수행되는 경우 재부팅 프로세스가 더 오래 걸릴 수 있습니다.

Amazon MQ 브로커 삭제

Amazon MQ 브로커를 사용하지 않는 경우(조만간 사용을 예측하지 않는 경우) AWS 비용을 절감하기 위해 Amazon MQ에서 삭제하는 것이 좋습니다.

다음 예제에서는 AWS Management Console을 사용하여 브로커를 삭제하는 방법을 보여줍니다.

Amazon MQ 브로커 삭제

1. Amazon MQ 콘솔에 로그인합니다.

2. 브로커 목록에서 브로커(예: MyBroker)를 선택한 다음 Delete(삭제)를 선택합니다.
3. Delete ***MyBroker***?(MyBroker를 삭제할까요?) 대화 상자에서 `delete`를 입력한 후 Delete(삭제)를 선택합니다.

   브로커 삭제는 약 5분 정도 소요됩니다.

Amazon MQ 브로커 상태

브로커의 현재 조건은 상태로 표시됩니다. 다음 표에는 Amazon MQ 브로커의 상태가 나열되어 있습니다.

콘솔	API	설명
생성 실패	`CREATION_FAILED`	브로커를 생성할 수 없습니다.
생성 진행 중	`CREATION_IN_PROGRESS`	브로커를 만드는 중입니다.
삭제 진행 중	`DELETION_IN_PROGRESS`	브로커를 삭제하는 중입니다.
재부팅 진행 중	`REBOOT_IN_PROGRESS`	브로커를 재부팅하는 중입니다.
실행	`RUNNING`	브로커가 작동합니다.
중요 작업 필요	`CRITICAL_ACTION_REQUIRED`	브로커가 실행 중이지만 성능이 저하되어 즉각적인 조치가 필요합니다. 문제 해결의 목록에서 작업 필요 코드를 선택하여 문제를 해결하기 위한 지침을 찾을 수 있습니다.

Amazon MQ 리소스에 태그 추가

비용 할당을 위해 Amazon MQ 리소스를 구성하고 식별하려면 브로커 또는 구성의 목적을 식별하는 메타데이터 태그를 추가할 수 있습니다. 이 기능은 브로커가 많을 때 특히 유용합니다. 비용 할당 태그를 사용하여 비용 구조를 반영하도록 AWS 청구서를 구성할 수 있습니다. 이렇게 하려면 태그 키와 값이

포함될 AWS 계정 청구서를 가져오도록 등록합니다. 자세한 내용은 AWS Billing 사용 설명서에서 월간 비용 할당 보고서 설정을 참조하세요.

예를 들어 Amazon MQ 리소스의 비용 센터 및 목적을 나타내는 태그를 추가할 수 있습니다.

리소스	키	값
Broker1	Cost Center	34567
	Stack	Production
Broker2	Cost Center	34567
	Stack	Production
Broker3	Cost Center	12345
	Stack	Development

이 태그 지정 체계에서는 동일한 비용 센터에서 관련된 작업을 수행하는 2개의 브로커를 그룹화할 수 있고, 관련이 없는 브로커는 다른 비용 할당 태그를 사용해 태그 지정할 수 있습니다.

Amazon MQ 콘솔에서 태그 추가

다음 단계에 따라 Amazon MQ 콘솔에서 생성하는 리소스에 태그를 빠르게 추가할 수 있습니다.

1. Create a broker(브로커 생성) 페이지에서 Additional settings(추가 설정)를 선택합니다.
2. Tags(태그) 아래에서 Add tag(태그 추가)를 선택합니다.
3. Key(키) 및 Value(값) 페어를 입력합니다.
4. (선택 사항) Add tag(태그 추가)를 선택하여 브로커에 여러 태그를 추가합니다.
5. Create broker(브로커 생성)를 선택합니다.

구성을 생성할 때 태그를 추가하려면

1. Create configuration(구성 생성) 페이지에서 Advanced(고급)를 선택합니다.
2. Create configuration(구성 생성) 페이지의 Tags(태그) 아래에서 Add tag(태그 추가)를 선택합니다.
3. Key(키) 및 Value(값) 페어를 입력합니다.

4. (선택 사항) Add tag(태그 추가)를 선택하여 구성에 여러 태그를 추가합니다.
5. Create configuration(구성 생성)을 선택합니다.

태그를 추가한 후 Amazon MQ 콘솔에서 리소스에 대한 태그를 확인, 편집, 제거할 수 있습니다. REST API를 사용하여 리소스의 태그를 확인할 수도 있습니다. 자세한 내용은 Amazon MQ REST API 참조를 참조하세요.

ActiveMQ용 Amazon MQ 사용

Amazon MQ를 사용하면 필요에 맞는 컴퓨팅 및 스토리지 리소스를 사용하여 메시지 브로커를 쉽게 생성할 수 있습니다. AWS Management Console, Amazon MQ REST API 또는 AWS Command Line Interface를 사용하여 브로커를 생성, 관리 및 삭제할 수 있습니다.

ActiveMQ용 Amazon MQ 브로커는 단일 인스턴스 브로커 또는 활성/대기 브로커로 배포할 수 있습니다. 두 배포 모드 모두에서 Amazon MQ는 데이터를 중복 저장하여 높은 내구성을 제공합니다.

> **Note**
>
> Amazon MQ는 데이터 스토어로 Apache KahaDB를 사용합니다. JDBC와 LevelDB와 같은 기타 데이터 스토어는 지원되지 않습니다.

ActiveMQ가 지원하는 모든 프로그래밍 언어를 사용하고 다음 프로토콜에 대해 명시적으로 TLS를 활성화하여 브로커에 액세스할 수 있습니다.

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

Amazon MQ REST API에 대한 자세한 내용은 Amazon MQ REST API 참조를 참조하세요.

ActiveMQ용 Amazon MQ 브로커

ActiveMQ용 Amazon MQ 브로커란?

브로커는 Amazon MQ에서 실행하는 메시지 브로커 환경입니다. 이 인스턴스는 Amazon MQ의 기본 빌딩 블록입니다. 브로커 인스턴스 클래스(`m5`) 및 크기(`large`, `medium`)의 설명 조합은 브로커 인스턴스 유형(예: `mq.m5.large`)입니다. 자세한 내용은 Broker instance types 섹션을 참조하세요.

- 단일 인스턴스 브로커는 하나의 가용 영역에 있는 하나의 브로커로 구성됩니다. 브로커는 애플리케이션 및 Amazon EBS 또는 Amazon EFS 스토리지 볼륨과 통신합니다.

- 활성/대기 브로커는 두 개의 서로 다른 가용 영역에 있는 두 개의 브로커가 중복 페어로 구성됩니다. 이러한 브로커는 애플리케이션 및 Amazon EFS와 동기식으로 통신합니다.

자세한 정보는 ActiveMQ용 Amazon MQ 브로커의 배포 옵션을 참조하세요.

Apache에서 새 버전이 출시될 때 브로커 엔진의 새 마이너 버전으로 업그레이드하려면 마이너 버전 자동 업그레이드를 활성화 수 있습니다. 자동 업그레이드는 요일, 시간(24시간 형식) 및 시간대(기본값은 UTC)에 의해 정의된 유지 관리 기간 도중에 발생합니다.

브로커 생성 및 관리에 대한 자세한 내용은 다음을 참조하세요.

- 시작하기: ActiveMQ 브로커 생성 및 연결
- 브로커
- Broker statuses

지원되는 와이어 레벨 프로토콜

ActiveMQ가 지원하는 모든 프로그래밍 언어를 사용하고 다음 프로토콜에 대해 명시적으로 TLS를 활성화하여 브로커에 액세스할 수 있습니다.

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

속성

ActiveMQ 브로커에는 여러 속성이 있습니다. 예를 들면 다음과 같습니다.

- 이름 (`MyBroker`)
- ID(`b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9`)
- Amazon 리소스 이름(ARN)(`arn:aws:mq:us-east-2:123456789012:broker:MyBroker:b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9`)

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

  자세한 내용은 Apache ActiveMQ 설명서의 웹 콘솔을 참조하세요.

  > **Important**
  >
  > `activemq-webconsole` 그룹을 포함하지 않는 권한 부여 맵을 지정하는 경우, 그룹이 Amazon MQ 브로커에 메시지를 보내거나 브로커에서 메시지를 수신할 권한이 없기 때문에 ActiveMQ 웹 콘솔을 사용할 수 없습니다.

- 와이어 레벨 프로토콜 엔드포인트:
  - `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**
    >
    > OpenWire 엔드포인트입니다.

  - `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`

  자세한 내용은 Apache ActiveMQ 설명서의 전송 구성을 참조하세요.

> **Note**
>
> 활성/대기 브로커의 경우 Amazon MQ는 2개의 ActiveMQ 웹 콘솔 URL을 제공하지만, 한 번에 하나의 URL만 활성화됩니다. 마찬가지로 Amazon MQ는 각 와이어 레벨 프로토콜에 대해 2개의 엔드포인트를 제공하지만, 한 번에 각 페어의 한 엔드포인트만 활성화됩니다. `-1` 및 `-2` 접미사는 중복 페어를 나타냅니다.

브로커 속성의 전체 목록은 Amazon MQ REST API 참조의 다음 단원을 참조하세요.

- REST 작업 ID: 브로커
- REST 작업 ID: 브로커
- REST 작업 ID: 브로커 재부팅

브로커 사용자

ActiveMQ 사용자는 ActiveMQ 브로커의 대기열 및 주제에 액세스할 수 있는 사람 또는 애플리케이션입니다. 사용자가 특정 권한을 갖도록 구성할 수 있습니다. 예를 들어 일부 사용자가 ActiveMQ 웹 콘솔에 액세스하도록 허용할 수 있습니다.

그룹을 의미 체계 레이블입니다. 사용자에 그룹을 할당하고 그룹에 대해 특정 대기열 및 주제에 보내고 대기열 및 주제에서 받고 대기열 및 주제를 관리할 권한을 구성할 수 있습니다.

> **Important**
>
> 사용자를 변경해도 변경 사항이 사용자에게 즉시 적용되지는 않습니다. 변경 내용을 적용하려면 다음 유지 관리 기간을 기다리거나 브로커를 재부팅해야 합니다.

사용자 및 그룹에 대한 자세한 내용은 Apache ActiveMQ 설명서의 다음 단원을 참조하세요.

- 권한 부여
- 권한 부여 예제

ActiveMQ 사용자 생성, 편집 및 삭제에 대한 자세한 내용은 다음을 참조하세요.

- ActiveMQ 브로커 사용자 생성
- Users

사용자 속성

사용자 속성의 전체 목록은 Amazon MQ REST API 참조의 다음 단원을 참조하세요.

- REST Operation ID: User
- REST Operation ID: Users

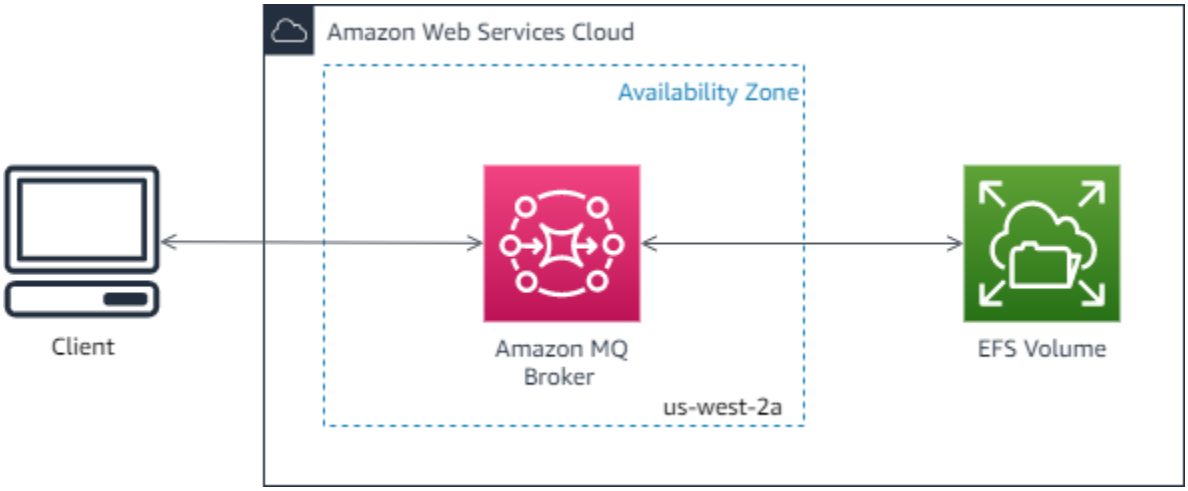
ActiveMQ용 Amazon MQ 브로커의 배포 옵션

Amazon MQ는 브로커에 대해 단일 인스턴스 및 클러스터 배포 옵션을 제공합니다.

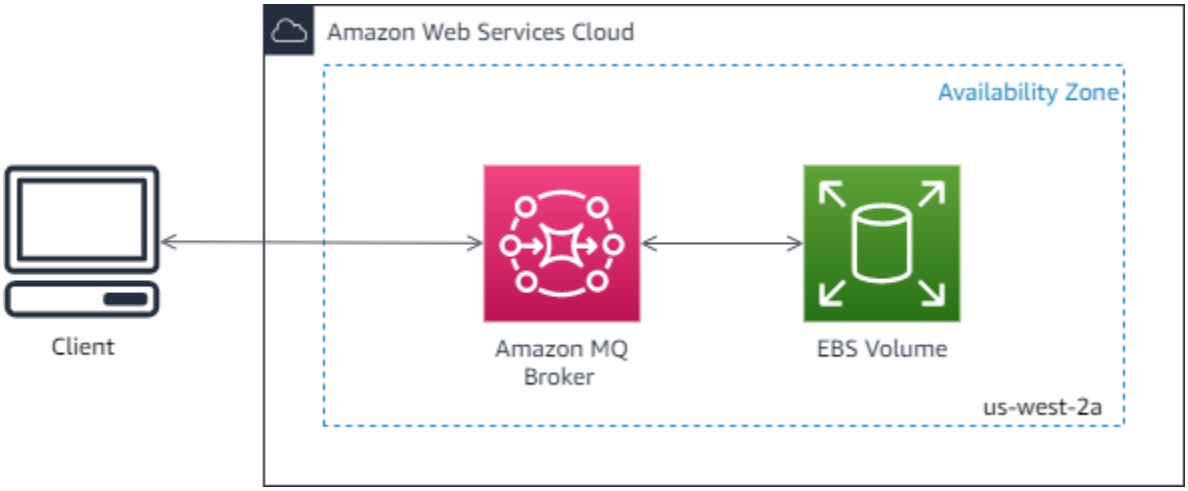
옵션 1: Amazon MQ 단일 인스턴스 브로커

단일 인스턴스 브로커는 하나의 가용 영역에 있는 하나의 브로커로 구성됩니다. 브로커는 애플리케이션 및 Amazon EBS 또는 Amazon EFS 스토리지 볼륨과 통신합니다. Amazon EFS 스토리지 볼륨은 데이터를 여러 가용 영역(AZ)에 중복 저장하여 최고 수준의 내구성과 가용성을 제공하도록 설계되었습니다. Amazon EBS는 짧은 대기 시간과 높은 처리량에 최적화된 블록 수준 스토리지를 제공합니다. 스토리지 옵션에 대한 자세한 정보는 Storage 단원을 참조하세요.

다음 다이어그램은 여러 AZ에 복제된 Amazon EFS 스토리지가 있는 단일 인스턴스 브로커를 보여줍니다.



다음 다이어그램은 단일 AZ 내의 여러 서버에 복제된 Amazon EBS 스토리지가 있는 단일 인스턴스 브로커를 보여줍니다.



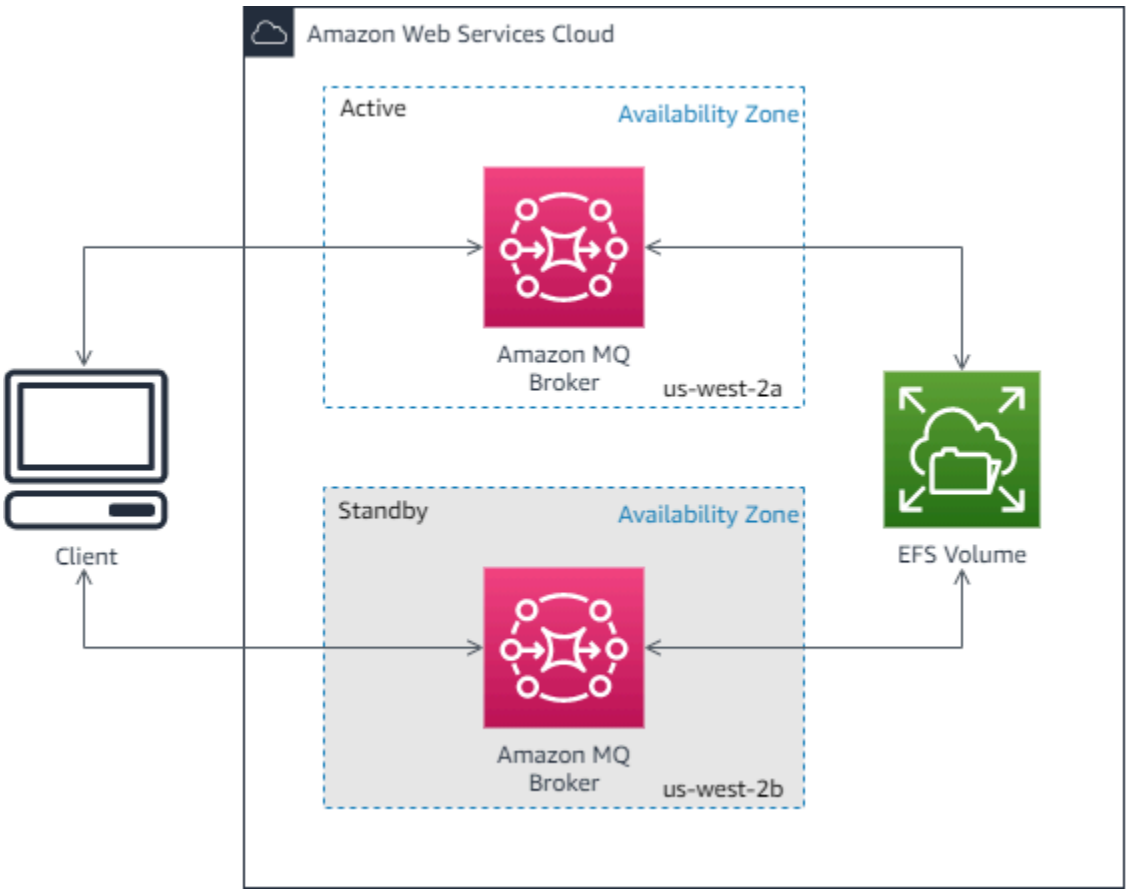
옵션 2: 고가용성을 위한 Amazon MQ 활성/대기 브로커

활성/대기 브로커는 두 개의 서로 다른 가용 영역에 있는 두 개의 브로커가 중복 페어로 구성됩니다. 이러한 브로커는 애플리케이션 및 Amazon EFS와 동기식으로 통신합니다. Amazon EFS 스토리지 볼륨은 데이터를 여러 가용 영역(AZ)에 중복 저장하여 최고 수준의 내구성과 가용성을 제공하도록 설계되었습니다. 자세한 정보는 Storage을 참조하세요.

일반적으로 한 번에 하나의 브로커 인스턴스만 활성 상태이고, 다른 브로커 인스턴스는 대기 상태입니다. 브로커 인스턴스 중 하나가 제대로 작동하지 않거나 유지 관리 중이면 Amazon MQ가 비활성 인스턴스를 서비스 중지하는 데 잠깐 시간이 걸립니다. 그런 다음 정상 대기 인스턴스가 활성화되고 들어오는 통신을 수신하기 시작할 수 있습니다. 시작한 유지 관리 기간과 브로커 재부팅으로 인해 장애 조치가 발생합니다. 브로커를 재부팅하면 몇 초 만에 장애 조치가 수행됩니다.

활성/대기 브로커의 경우 Amazon MQ는 2개의 ActiveMQ 웹 콘솔 URL을 제공하지만, 한 번에 하나의 URL만 활성화됩니다. 마찬가지로 Amazon MQ는 각 와이어 레벨 프로토콜에 대해 2개의 엔드포인트를 제공하지만, 한 번에 각 페어의 한 엔드포인트만 활성화됩니다. `-1` 및 `-2` 접미사는 중복 페어를 나타냅니다. 와이어 레벨 프로토콜 엔드포인트의 경우 장애 조치 전송을 사용하여 애플리케이션이 다른 엔드포인트와 연결하도록 허용할 수 있습니다.

다음 다이어그램은 여러 AZ에 복제된 Amazon EFS 스토리지가 있는 활성/대기 브로커를 보여줍니다.



Amazon MQ 브로커 네트워크

Amazon MQ는 ActiveMQ의 브로커 네트워크 기능을 지원합니다.

브로커 네트워크는 동시에 여러 활성 단일 인스턴스 브로커 또는 활성/대기 브로커로 구성됩니다. 브로커 네트워크를 생성하면 여러 브로커 인스턴스와의 가용성, 내결함성 및 로드 밸런싱이 향상될 수 있습니다.

브로커 네트워크 작동 방식

브로커 네트워크는 네트워크 커넥터를 사용하여 한 브로커를 다른 브로커에 연결함으로써 설정됩니다. 네트워크 커넥터는 한 브로커에서 다른 브로커로 온디맨드 메시지를 제공합니다. 네트워크 커넥터는 브로커 구성에서 비이중 또는 이중 연결로 구성됩니다. 비 전이중 연결에서는 메시지가 특정 브로커에서 다른 브로커로만 전달됩니다. 이중 연결의 경우 메시지는 두 브로커 간에 양방향으로 전달됩니다.

네트워크 커넥터가 이중으로 구성되면 메시지가 Broker2에서 Broker1로 전달될 수도 있습니다.

브로커 네트워크에서 비이중 연결과 이중 연결을 모두 사용할 수 있습니다. 트래픽을 개선하거나 한도 증가를 방지하기 위해 다른 브로커에 이중 연결을 도입할 수 있습니다. 또한 이중 연결은 온프레미스에서 Amazon MQ 관리형 브로커로 부분적으로 마이그레이션하는 데 유용합니다.

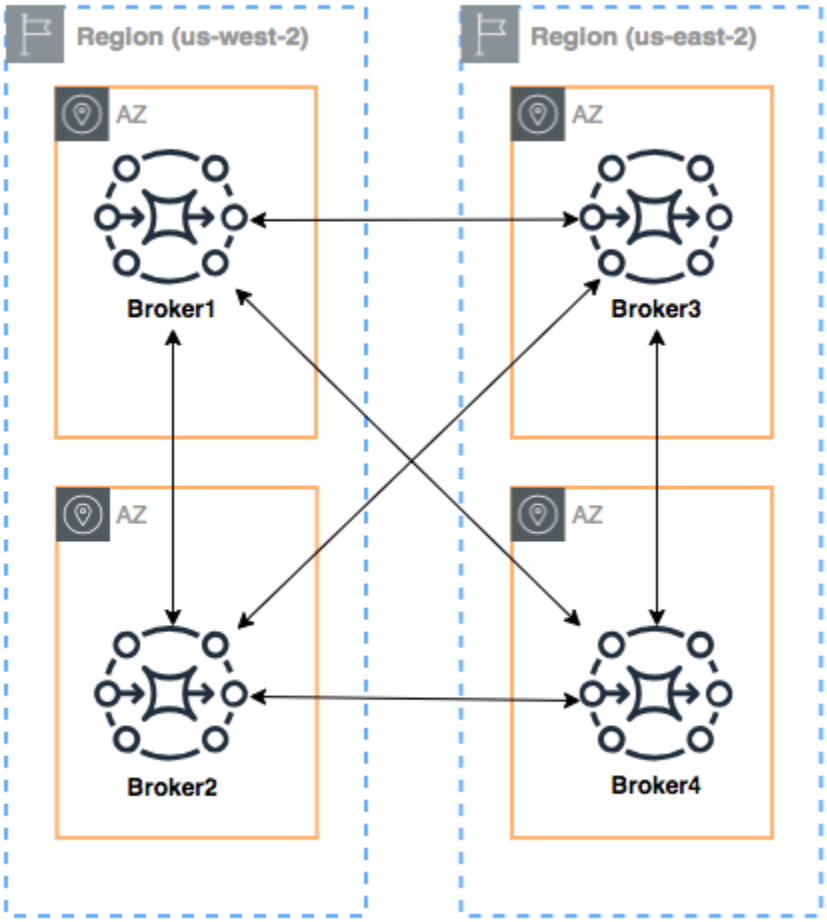
브로커 네트워크가 자격 증명을 처리하는 방식

네트워크에서 브로커 A가 브로커 B에 연결하려면 브로커 A는 다른 생산자 또는 소비자와 마찬가지로 유효한 자격 증명을 사용해야 합니다. 브로커 A의 `<networkConnector>` 구성에 암호를 제공하는 대신, 먼저 브로커 B의 다른 사용자와 동일한 값을 사용하여 브로커 A 사용자를 생성해야 합니다(이들은 사용자 이름 및 암호 값을 공유하는 별도의 고유한 사용자임). `<networkConnector>` 구성에서 `userName` 속성을 지정하면 Amazon MQ가 런타임 시 자동으로 암호를 추가합니다.

> **Important**
>
> `password`에 `<networkConnector>` 속성을 지정하지 마세요. 브로커 구성 파일에 일반 텍스트 암호를 저장하면 Amazon MQ 콘솔에서 해당 암호가 표시되므로 권장하지 않습니다. 자세한 내용은 Configure Network Connectors for Your Broker 단원을 참조하십시오.

교차 리전

AWS 리전에 걸쳐 있는 브로커 네트워크를 구성하려면 해당 리전에 브로커를 배포하고 해당 브로커의 엔드포인트에 네트워크 커넥터를 구성합니다.



이 예제와 같은 브로커 네트워크를 구성하려면 브로커의 와이어 레벨 끝점을 참조하는 `networkConnectors` 항목을 Broker1 및 Broker4의 구성에 추가할 수 있습니다.

Broker1용 네트워크 커넥터:

```
<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>
```

Broker2용 네트워크 커넥터:

```
<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>
```

Broker4용 네트워크 커넥터:

```
<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>
```

전송 커넥터를 사용한 동적 장애 조치

networkConnector 요소를 구성하는 것 외에도 브로커 transportConnector 옵션을 구성하여 동적 장애 조치를 활성화하고 네트워크에서 브로커를 추가하거나 제거할 때 연결을 리밸런싱할 수 있습니다.

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

이 예제에서는 updateClusterClients 및 rebalanceClusterClients를 모두 true로 설정합니다. 이 경우 클라이언트는 네트워크의 브로커 목록을 제공받으며 새 브로커가 참여하면 브로커에 리밸런싱하도록 요청합니다.

사용 가능한 옵션:

- updateClusterClients: 브로커 토폴로지 네트워크의 변경 사항에 대한 정보를 클라이언트에 전달합니다.

- `rebalanceClusterClients`: 브로커 네트워크에 새 브로커가 추가될 때 클라이언트가 브로커 간에 리밸런싱하도록 합니다.
- `updateClusterClientsOnRemove`: 브로커가 브로커 네트워크를 나가면 토폴로지 정보로 클라이언트를 업데이트합니다.

`updateClusterClients`를 true로 설정하면 브로커 네트워크의 단일 브로커에 연결하도록 클라이언트를 구성할 수 있습니다.

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

새 브로커가 연결되면 클라이언트는 네트워크에 있는 모든 브로커의 URI 목록을 받습니다. 브로커 연결에 실패하면 클라이언트는 연결될 때 제공되는 브로커 중 하나로 동적으로 전환할 수 있습니다.

장애 조치에 대한 자세한 내용은 Active MQ 설명서에서 장애 조치를 위한 브로커 측 옵션을 참조하세요.

ActiveMQ용 Amazon MQ 브로커 인스턴스 유형

브로커 인스턴스 클래스(`m5`) 및 크기(`large`, `medium`)의 설명 조합은 브로커 인스턴스 유형(예: `mq.m5.large`)입니다. 다음 표에는 ActiveMQ 브로커에 사용 가능한 Amazon MQ 브로커 인스턴스 유형이 나열되어 있습니다.

Amazon MQ는 인스턴스 유형 지원이 종료되기 최소 90일 전에 미리 통지합니다. 중단이 발생하지 않도록 지원 종료일 전에 브로커를 새 인스턴스 유형으로 업그레이드하는 것이 좋습니다.

> **Important**
>
> 2025년 3월 17일 이후에는 `t2.micro` 또는 `mq.m4.large`에서 브로커를 생성할 수 없습니다.

인스턴스 유형	vCPU	메모리(GiB)	권장 사용	스토리지	Amazon MQ 지원 종료
`mq.t3.micro`	2	1	평가	EFS	

인스턴스 유형	vCPU	메모리(GiB)	권장 사용	스토리지	Amazon MQ 지원 종료
mq.m5.large	2	8	프로덕션	EFS 또는 EBS	
mq.m5.xlarge	4	16	프로덕션	EFS 또는 EBS	
mq.m5.2xlarge	8	32	프로덕션	EFS 또는 EBS	
mq.m5.4xlarge	16	64	프로덕션	EFS 또는 EBS	

처리량 고려 사항에 대한 자세한 내용은 처리량을 최대화하기 위해 올바른 브로커 인스턴스 유형 선택 단원을 참조하세요.

ActiveMQ용 Amazon MQ 브로커 구성

구성에는 ActiveMQ 브로커에 대한 모든 설정이 XML 형식(ActiveMQ의 `activemq.xml` 파일과 유사)으로 포함됩니다. 브로커를 생성하기 전에 구성을 생성할 수 있습니다. 그런 다음 구성을 하나 이상의 브로커에 적용할 수 있습니다.

> **Important**
>
> 구성을 변경해도 변경 사항이 브로커에 즉시 적용되지는 않습니다. 변경 내용을 적용하려면 다음 유지 관리 기간을 기다리거나 브로커를 재부팅해야 합니다.
> `DeleteConfiguration` API를 사용해서만 구성을 삭제할 수 있습니다. 자세한 내용은 Amazon MQ API 참조의 구성을 참조하세요.

속성

브로커 구성에는 여러 속성이 있습니다. 예시는 다음과 같습니다.

- 이름 (`MyConfiguration`)

- ID(`c-1234a5b6-78cd-901e-2fgh-3i45j6k178l9`)
- Amazon 리소스 이름(ARN)(`arn:aws:mq:us-east-2:123456789012:configuration:c-1234a5b6-78cd-901e-2fgh-3i45j6k178l9`)

구성 속성의 전체 목록은 Amazon MQ REST API 참조의 다음 단원을 참조하세요.

- REST 작업 ID: 구성
- REST 작업 ID: 구성

구성 개정 속성의 전체 목록은 다음 단원을 참조하세요.

- REST 작업 ID: 구성 개정
- REST 작업 ID: 구성 개정

Spring XML 구성 파일 사용

ActiveMQ 브로커는 Spring XML 파일을 사용하여 구성됩니다. 사전 정의된 대상, 대상 정책, 권한 부여 정책 및 플러그 인과 같은 ActiveMQ 브로커의 여러 측면을 구성할 수 있습니다. Amazon MQ는 이러한 구성 요소 중 네트워크 전송 및 스토리지와 같은 일부 요소를 제어합니다. 브로커의 네트워크 생성과 같은 기타 구성 옵션은 현재 지원되지 않습니다.

지원되는 구성 옵션의 전체 세트는 Amazon MQ XML 스키마에 지정되어 있습니다. 다음 링크를 사용하여 지원되는 스키마의 zip 파일을 다운로드합니다.

- `amazon-mq-active-mq-5.19.1.xsd.zip`
- `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`

이러한 스키마를 사용하여 구성 파일을 검증하고 폐기할 수 있습니다. Amazon MQ에서는 XML 파일을 업로드하여 구성을 제공할 수도 있습니다. XML 파일을 업로드하면 Amazon MQ는 스키마에 따라 잘못된 구성 파라미터와 금지된 구성 파라미터를 자동으로 폐기하고 제거합니다.

> **Note**
>
> 속성의 정적 값만 사용할 수 있습니다. Amazon MQ는 Spring 표현식, 변수 및 요소 참조가 포함된 요소와 속성을 구성에서 폐기합니다.

ActiveMQ용 Amazon MQ 브로커 구성 생성

구성에는 ActiveMQ 브로커에 대한 모든 설정이 XML 형식(ActiveMQ의 `activemq.xml` 파일과 유사)으로 포함됩니다. 브로커를 생성하기 전에 구성을 생성할 수 있습니다. 그런 다음 구성을 하나 이상의 브로커에 적용할 수 있습니다. 구성을 즉시 적용하거나 유지 관리 기간 중에 적용할 수 있습니다.

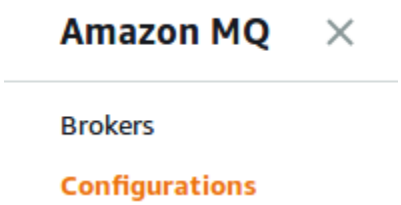
다음 예제에서는 AWS Management Console을 사용하여 Amazon MQ 브로커 구성을 생성하고 적용하는 방법을 보여줍니다.

> **Important**
>
> `DeleteConfiguration` API를 사용해서만 구성을 삭제할 수 있습니다. 자세한 내용은 Amazon MQ API 참조의 구성을 참조하세요.

새 구성 생성

새 브로커 구성을 생성하려면 먼저 새 구성을 생성합니다.

1. Amazon MQ 콘솔에 로그인합니다.
2. 왼쪽에서 탐색 창을 확장하고 Configurations(구성)를 선택합니다.



3. Configurations(구성) 페이지에서 Create configuration(구성 생성)을 선택합니다.
4. 구성 생성 페이지의 세부 정보(Details) 섹션에서 구성 이름(Configuration name)(예: `MyConfiguration`)을 입력하고 브로커 엔진(Broker engine) 버전을 선택합니다.

> **Note**
>
> ActiveMQ용 Amazon MQ에서 지원하는 ActiveMQ 엔진 버전에 대한 자세한 내용은 the section called “버전 관리” 단원을 참조하세요.

5. 구성 생성을 선택합니다.

새로운 구성 개정 생성

브로커 구성을 생성한 후에는 구성 개정을 사용하여 구성을 편집해야 합니다.

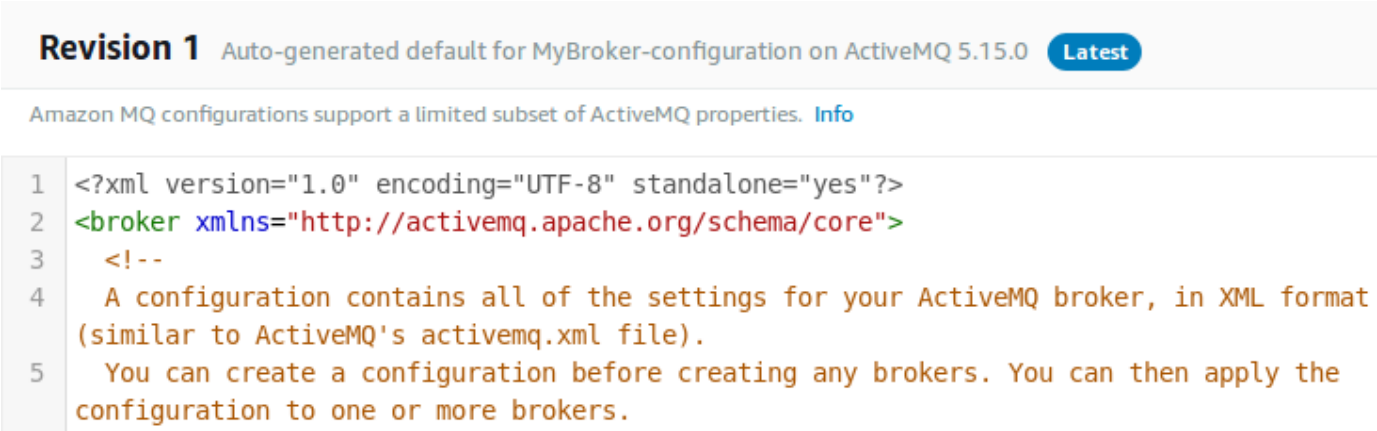
1. 구성 목록에서 ***MyConfiguration***을 선택합니다.

> **Note**
>
> 첫 번째 구성 개정은 Amazon MQ가 구성을 생성할 때 항상 생성됩니다.

MyConfiguration 페이지에 새로운 구성 개정에 사용할 브로커 엔진 유형 및 버전(예: Apache ActiveMQ 5.15.16)이 표시됩니다.

2. [Configuration details] 탭에 구성 개정 번호, 설명 및 XML 형식의 브로커 구성이 표시됩니다.

> **Note**
>
> 현재 구성을 편집하면 새 구성 개정이 생성됩니다.



3. [Edit configuration]을 선택하고 XML 구성을 변경합니다.
4. 저장을 선택합니다.

   Save revision(개정 버전 저장) 대화 상자가 표시됩니다.
5. (선택 사항) `A description of the changes in this revision`을 입력합니다.
6. 저장을 선택합니다.

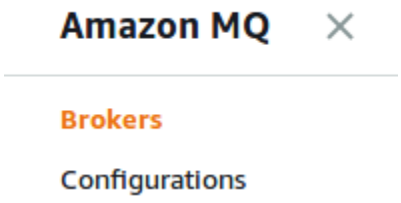
   구성의 새 개정 버전이 저장됩니다.

> **Important**
>
> Amazon MQ 콘솔이 스키마에 따라 잘못되고 금지된 구성 파라미터를 자동으로 정리합니다. 자세한 내용과 허용된 XML 파라미터의 전체 목록은 Amazon MQ Broker Configuration Parameters 단원을 참조하세요.

브로커에 구성 개정 적용

구성을 수정한 후 브로커에 구성 개정을 적용할 수 있습니다.

1. 왼쪽에서 탐색 창을 확장하고 Brokers(브로커)를 선택합니다.



2. 브로커 목록에서 브로커(예: MyBroker)를 선택한 다음 Edit(편집)을 선택합니다.
3. [Edit ***MyBroker***] 페이지의 [Configuration] 섹션에서 [Configuration] 및 [Revision]을 선택하고 [Schedule Modifications]를 선택합니다.
4. Schedule broker modifications(브로커 수정 예약) 섹션에서 During the next scheduled maintenance window(예약된 다음 유지 관리 기간 동안) 또는 Immediately(즉시) 중에 수정을 적용할 시점을 선택합니다.

> **Important**
>
> 단일 인스턴스 브로커는 재부팅되는 동안 오프라인 상태입니다. 클러스터 브로커의 경우 브로커가 재부팅되는 동안 한 번에 하나의 노드만 다운됩니다.

5. 적용을 선택합니다.

   지정된 시간에 구성 개정이 브로커에 적용됩니다.

ActiveMQ용 Amazon MQ 구성 개정 편집

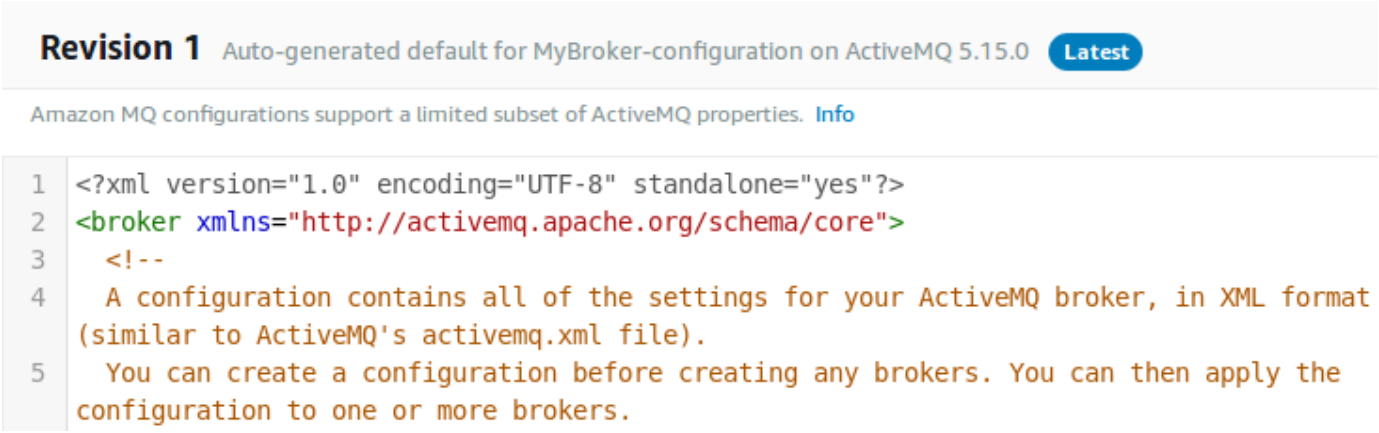
브로커에 구성 개정을 적용한 후 이를 편집할 수 있습니다. 다음 지침에 따라 구성 개정을 편집합니다.

1. Amazon MQ 콘솔에 로그인합니다.
2. 브로커 목록에서 브로커(예: MyBroker)를 선택한 다음 Edit(편집)을 선택합니다.
3. ***MyBroker*** 페이지에서 Edit(편집)를 선택합니다.
4. Edit ***MyBroker***(MyBroker 편집) 페이지의 구성 섹션에서 구성 및 개정을 선택하고 편집을 선택합니다.

   > **Note**
   >
   > 브로커를 생성할 때 구성을 선택하지 않는 한 첫 번째 구성 개정은 Amazon MQ가 브로커를 생성할 때 항상 생성됩니다.

   MyBroker 페이지에서 구성에 사용되는 브로커 엔진 유형 및 버전(예: Apache ActiveMQ 5.15.8)을 볼 수 있습니다.
5. [Configuration details] 탭에 구성 개정 번호, 설명 및 XML 형식의 브로커 구성이 표시됩니다.

   > **Note**
   >
   > 현재 구성을 편집하면 새 구성 개정이 생성됩니다.



6. [Edit configuration]을 선택하고 XML 구성을 변경합니다.
7. 저장을 선택합니다.

   Save revision(개정 버전 저장) 대화 상자가 표시됩니다.
8. (선택 사항) `A description of the changes in this revision`을 입력합니다.
9. 저장을 선택합니다.

   구성의 새 개정 버전이 저장됩니다.

> **Important**
>
> Amazon MQ 콘솔이 스키마에 따라 잘못되고 금지된 구성 파라미터를 자동으로 정리합니다. 자세한 내용과 허용된 XML 파라미터의 전체 목록은 Amazon MQ Broker Configuration Parameters 단원을 참조하세요.

Amazon MQ 구성에 허용되는 요소

다음은 Amazon MQ 구성에서 허용되는 요소의 세부 목록입니다. 자세한 내용은 Apache ActiveMQ 설명서의 XML 구성을 참조하세요.

요소
`abortSlowAckConsumerStrategy` (속성)
`abortSlowConsumerStrategy` (속성)

요소
authorizationEntry (속성)
authorizationMap (하위 컬렉션 요소)
authorizationPlugin (하위 컬렉션 요소)
broker(속성 | 하위 컬렉션 요소)
cachedMessageGroupMapFactory (속성)
compositeQueue (속성 | 하위 컬렉션 요소)
compositeTopic (속성 | 하위 컬렉션 요소)
constantPendingMessageLimitStrategy (속성)
discarding (속성)
discardingDLQBrokerPlugin (속성)
fileCursor
fileDurableSubscriberCursor
fileQueueCursor
filteredDestination (속성)
fixedCountSubscriptionRecoveryPolicy (속성)
fixedSizedSubscriptionRecoveryPolicy (속성)
forcePersistencyModeBrokerPlugin (속성)
individualDeadLetterStrategy (속성)
lastImageSubscriptionRecoveryPolicy
messageGroupHashBucketFactory (속성)

요소
mirroredQueue (속성)
noSubscriptionRecoveryPolicy
oldestMessageEvictionStrategy (속성)
oldestMessageWithLowestPriorityEvictionStrategy (속성)
policyEntry (속성 | 하위 컬렉션 요소)
policyMap (하위 컬렉션 요소)
prefetchRatePendingMessageLimitStrategy (속성)
priorityDispatchPolicy
priorityNetworkDispatchPolicy
queryBasedSubscriptionRecoveryPolicy (속성)
queue(속성)
redeliveryPlugin (속성 | 하위 컬렉션 요소)
redeliveryPolicy (속성)
redeliveryPolicyMap (하위 컬렉션 요소)
retainedMessageSubscriptionRecoveryPolicy (하위 컬렉션 요소)
roundRobinDispatchPolicy
sharedDeadLetterStrategy (속성 | 하위 컬렉션 요소)
simpleDispatchPolicy
simpleMessageGroupMapFactory
statisticsBrokerPlugin

요소
storeCursor
storeDurableSubscriberCursor (속성)
strictOrderDispatchPolicy
tempDestinationAuthorizationEntry (속성)
tempQueue (속성)
tempTopic (속성)
timedSubscriptionRecoveryPolicy (속성)
timeStampingBrokerPlugin (속성)
topic(속성)
transportConnector (속성)
uniquePropertyMessageEvictionStrategy (속성)
virtualDestinationInterceptor (하위 컬렉션 요소)
virtualTopic (속성)
vmCursor
vmDurableCursor
vmQueueCursor

Amazon MQ 구성에서 허용되는 요소 및 해당 속성

다음은 Amazon MQ 구성에서 허용되는 요소 및 해당 속성의 세부 목록입니다. 자세한 내용은 Apache ActiveMQ 설명서의 XML 구성을 참조하세요.

요소	속성
abortSlowAckConsumerStrategy	abortConnection
	checkPeriod
	ignoreIdleConsumers
	ignoreNetworkConsumers
	maxSlowCount
	maxSlowDuration
	maxTimeSinceLastAck
	name
abortSlowConsumerStrategy	abortConnection
	checkPeriod
	ignoreNetworkConsumers
	maxSlowCount
	maxSlowDuration
	name
authorizationEntry	admin
	queue
	read
	tempQueue
	tempTopic
	topic

요소	속성
	write
broker	advisorySupport
	allowTempAutoCreationOnSend
	cacheTempDestinations
	consumerSystemUsagePortion
	dedicatedTaskRunner
	deleteAllMessagesOnStartup
	keepDurableSubsActive
	enableMessageExpirationOnActiveDurableSubs
	maxPurgedDestinationsPerSweep
	maxSchedulerRepeatAllowed
	monitorConnectionSplits
	networkConnectorStartAsync
	offlineDurableSubscriberTaskSchedule
	offlineDurableSubscriberTimeout
	persistenceThreadPriority
	persistent
	populateJMSXUserID
	producerSystemUsagePortion

요소	속성
	rejectDurableConsumers
	rollbackOnlyOnAsyncException
	schedulePeriodForDestinatio nPurge
	schedulerSupport
	splitSystemUsageForProducer sConsumers
	taskRunnerPriority
	timeBeforePurgeTempDestinations
	useAuthenticatedPrincipalFo rJMSXUserID
	useMirroredQueues
	useTempMirroredQueues
	useVirtualDestSubs
	useVirtualDestSubsOnCreation
	useVirtualTopics
cachedMessageGroupMapFactory	cacheSize
compositeQueue	concurrentSend
	copyMessage
	forwardOnly
	name

요소	속성
	sendWhenNotMatched
compositeTopic	concurrentSend
	copyMessage
	forwardOnly
	name
	sendWhenNotMatched
conditionalNetworkBridgeFilterFactory	rateDuration
	rateLimit
	replayDelay
	replayWhenNoConsumers
	selectorAware 지원 버전: Apache ActiveMQ 5.16.x
constantPendingMessageLimitStrategy	limit
discarding	deadLetterQueue
	enableAudit
	expiration
	maxAuditDepth
	maxProducersToAudit

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

요소	속성
	processNonPersistent
	queuePrefix
	queueSuffix
	topicPrefix
	topicSuffix
	useQueueForQueueMessages
	useQueueForTopicMessages
messageGroupHashBucketFactory	bucketCount
	cacheSize
mirroredQueue	copyMessage
	postfix
	prefix
oldestMessageEvictionStrategy	evictExpiredMessagesHighWat ermark
oldestMessageWithLowestPrio rityEvictionStrategy	evictExpiredMessagesHighWat ermark
policyEntry	advisoryForConsumed
	advisoryForDelivery
	advisoryForDiscardingMessages
	advisoryForFastProducers
	advisoryForSlowConsumers

요소	속성
	advisoryWhenFull
	allConsumersExclusiveByDefault
	alwaysRetroactive
	blockedProducerWarningInterval
	consumersBeforeDispatchStarts
	cursorMemoryHighWaterMark
	doOptimzeMessageStorage
	durableTopicPrefetch
	enableAudit
	expireMessagesPeriod
	gcInactiveDestinations
	gcWithNetworkConsumers
	inactiveTimeoutBeforeGC
	inactiveTimoutBeforeGC
	includeBodyForAdvisory
	lazyDispatch
	maxAuditDepth
	maxBrowsePageSize
	maxDestinations
	maxExpirePageSize

요소	속성
	maxPageSize
	maxProducersToAudit
	maxQueueAuditDepth
	memoryLimit
	messageGroupMapFactoryType
	minimumMessageSize
	optimizedDispatch
	optimizeMessageStoreInFligh tLimit
	persistJMSRedelivered
	prioritizedMessages
	producerFlowControl
	queue
	queueBrowserPrefetch
	queuePrefetch
	reduceMemoryFootprint
	sendAdvisoryIfNoConsumers
	sendFailIfNoSpace

요소	속성
	sendFailIfNoSpaceAfterTimeout 지원 버전: Apache ActiveMQ 5.16.4 이상
	sendDuplicateFromStoreToDLQ
	storeUsageHighWaterMark
	strictOrderDispatch
	tempQueue
	tempTopic
	timeBeforeDispatchStarts
	topic
	topicPrefetch
	useCache
	useConsumerPriority
usePrefetchExtension	
prefetchRatePendingMessageLimitStrategy	multiplier
queryBasedSubscriptionRecoveryPolicy	query
queue	DLQ
	physicalName

요소	속성
redeliveryPlugin	fallbackToDeadLetter
	sendToDlqIfMaxRetriesExceeded
redeliveryPolicy	backOffMultiplier
	collisionAvoidancePercent
	initialRedeliveryDelay
	maximumRedeliveries
	maximumRedeliveryDelay
	preDispatchCheck
	queue
	redeliveryDelay
	tempQueue
	tempTopic
	topic
	useCollisionAvoidance
	useExponentialBackOff
sharedDeadLetterStrategy	enableAudit
	expiration
	maxAuditDepth
	maxProducersToAudit
	processExpired

요소	속성
	processNonPersistent
storeDurableSubscriberCursor	immediatePriorityDispatch
	useCache
tempDestinationAuthorizationEntry	admin
	queue
	read
	tempQueue
	tempTopic
	topic
	write
tempQueue	DLQ
	physicalName
tempTopic	DLQ
	physicalName
timedSubscriptionRecoveryPolicy	zeroExpirationOverride
timeStampingBrokerPlugin	recoverDuration
	futureOnly
	processNetworkMessages
	ttlCeiling
topic	DLQ

요소	속성
	physicalName
transportConnector	name
	updateClusterClients
	rebalanceClusterClients
	updateClusterClientsOnRemove
uniquePropertyMessageEvictionStrategy	evictExpiredMessagesHighWatermark
	propertyName
virtualTopic	concurrentSend
	local
	dropOnResourceLimit
	name
	postfix
	prefix
	selectorAware
	setOriginalDestination
	transactedSend

Amazon MQ 상위 요소 속성

다음은 상위 요소 속성에 대한 자세한 설명입니다. 자세한 내용은 Apache ActiveMQ 설명서의 XML 구성을 참조하세요.

주제

- 브로커

브로커

broker는 상위 컬렉션 요소입니다.

속성

networkConnectionStartAsync

네트워크 대기 시간을 줄이고 다른 네트워크가 적시에 시작하도록 하려면<networkConnectionStartAsync> 태그를 사용합니다. 해당 태그는 브로커가 실행기를 사용하여 브로커 시작과 비동기적으로 네트워크 연결을 병렬로 시작하도록 지시합니다.

기본값: false

구성의 예제

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

Amazon MQ 구성에서 허용되는 요소, 하위 컬렉션 요소 및 해당 하위 요소

다음은 Amazon MQ 구성에서 허용되는 요소, 하위 컬렉션 요소 및 해당 하위 요소의 세부 목록입니다. 자세한 내용은 Apache ActiveMQ 설명서의 XML 구성을 참조하세요.

요소	하위 컬렉션 요소	하위 요소
authorizationMap	authorizationEntries	authorizationEntry
		tempDestinationAuthorizationEntry
	defaultEntry	authorizationEntry
		tempDestinationAuthorizationEntry
	tempDestinationAuthorizationEntry	tempDestinationAuthorizationEntry
authorizationPlugin	map	authorizationMap

요소	하위 컬렉션 요소	하위 요소
broker	destinationInterceptors	mirroredQueue
		virtualDestinationInterceptor
	destinationPolicy	policyMap
	destinations	queue
		tempQueue
		tempTopic
		topic
	networkConnectors	networkConnector
	persistenceAdapter	kahaDB
	plugins	authorizationPlugin
		discardingDLQBrokerPlugin
		forcePersistencyModeBrokerPlugin
		redeliveryPlugin
		statisticsBrokerPlugin
		timeStampingBrokerPlugin
	systemUsage	systemUsage
	transportConnector	name

요소	하위 컬렉션 요소	하위 요소
		updateClusterClients
		rebalanceClusterClients
		updateClusterClientsOnRemove
compositeQueue	forwardTo	queue
		tempQueue
		tempTopic
		topic
		filteredDestination
compositeTopic	forwardTo	queue
		tempQueue
		tempTopic
		topic
		filteredDestination
policyEntry	deadLetterStrategy	discarding
		individualDeadLetterStrategy
		sharedDeadLetterStrategy
	destination	queue
		tempQueue

요소	하위 컬렉션 요소	하위 요소
		tempTopic
		topic
	dispatchPolicy	priorityDispatchPolicy
		priorityNetworkDispatchPolicy
		roundRobinDispatchPolicy
		simpleDispatchPolicy
		strictOrderDispatchPolicy
		clientIdFilterDispatchPolicy
	messageEvictionStrategy	oldestMessageEvictionStrategy
		oldestMessageWithLowestPriorityEvictionStrategy
		uniquePropertyMessageEvictionStrategy
	messageGroupMapFactory	cachedMessageGroupMapFactory
		messageGroupHashBucketFactory

요소	하위 컬렉션 요소	하위 요소
		simpleMessageGroupMapFactory
	pendingDurableSubscriberPolicy	fileDurableSubscriberCursor
		storeDurableSubscriberCursor
		vmDurableCursor
	pendingMessageLimitStrategy	constantPendingMessageLimitStrategy
		prefetchRatePendingMessageLimitStrategy
	pendingQueuePolicy	fileQueueCursor
		storeCursor
		vmQueueCursor
	pendingSubscriberPolicy	fileCursor
		vmCursor
	slowConsumerStrategy	abortSlowAckConsumerStrategy
		abortSlowConsumerStrategy
	subscriptionRecoveryPolicy	fixedCountSubscriptionRecoveryPolicy

요소	하위 컬렉션 요소	하위 요소
		fixedSizedSubscriptionRecoveryPolicy
		lastImageSubscriptionRecoveryPolicy
		noSubscriptionRecoveryPolicy
		queryBasedSubscriptionRecoveryPolicy
		retainedMessageSubscriptionRecoveryPolicy
timedSubscriptionRecoveryPolicy		
policyMap	defaultEntry	policyEntry
	policyEntries	policyEntry
redeliveryPlugin	redeliveryPolicyMap	redeliveryPolicyMap
redeliveryPolicyMap	defaultEntry	redeliveryPolicy
	redeliveryPolicyEntries	redeliveryPolicy
retainedMessageSubscriptionRecoveryPolicy	wrapped	fixedCountSubscriptionRecoveryPolicy
		fixedSizedSubscriptionRecoveryPolicy
		lastImageSubscriptionRecoveryPolicy

요소	하위 컬렉션 요소	하위 요소
		noSubscriptionRecoveryPolicy
		queryBasedSubscriptionRecoveryPolicy
		retainedMessageSubscriptionRecoveryPolicy
		timedSubscriptionRecoveryPolicy
sharedDeadLetterStrategy	deadLetterQueue	queue
		tempQueue
		tempTopic
		topic
virtualDestinationInterceptor	virtualDestinations	compositeQueue
		compositeTopic
		virtualTopic

Amazon MQ 하위 요소 속성

다음은 하위 요소 속성에 대한 자세한 설명입니다. 자세한 내용은 Apache ActiveMQ 설명서의 XML 구성을 참조하세요.

주제

- authorizationEntry
- networkConnector
- kahaDB

- systemUsage

authorizationEntry

`authorizationEntry`는 `authorizationEntries` 하위 컬렉션 요소의 하위입니다.

속성

admin|read|write

사용자 그룹에게 부여된 권한. 자세한 정보는 항상 권한 부여 맵 구성을 참조하세요.

`activemq-webconsole` 그룹을 포함하지 않는 권한 부여 맵을 지정하는 경우, 그룹이 Amazon MQ 브로커에 메시지를 보내거나 브로커에서 메시지를 수신할 권한이 없기 때문에 ActiveMQ 웹 콘솔을 사용할 수 없습니다.

기본값: `null`

구성의 예제

```
<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 기반 ActiveMQ의 `activemq-webconsole` 그룹은 모든 대기열 및 주제에 대한 관리자 권한을 보유하고 있습니다. 이 그룹의 모든 사용자도 관리자 액세스 권한을 갖습니다.

networkConnector

`networkConnector`는 `networkConnectors` 하위 컬렉션 요소의 하위입니다.

주제

속성

conduitSubscriptions

브로커 네트워크 내 네트워크 연결이 동일한 대상에 구독하는 여러 소비자를 한 소비자로 취급하는지 여부를 지정합니다. 예를 들어 `conduitSubscriptions`가 `true`로 설정되고 두 소비자가 브로커 B에 연결하여 대상으로부터 소비할 경우, 브로커 B는 이들의 구독을 네트워크 연결을 통한 브로커 A에 대한 단일의 논리적 구독으로 결합합니다. 그러므로 메시지의 단일 복사본만 브로커 A에서 브로커 B로 전달됩니다.

> **Note**
>
> `conduitSubscriptions`를 `true`로 설정하면 중복 네트워크 트래픽을 줄일 수 있습니다. 하지만 이 속성을 사용하면 소비자 사이의 로드 밸런싱 문제가 발생할 수 있고 일부 시나리오(예: JMS 메시지 선택기 또는 장기적 주제)에서 잘못된 동작을 초래할 수 있습니다.

기본값: `true`

duplex

브로커 네트워크 내 연결이 메시지를 생산 및 소비하는 데 사용되는지 여부를 지정합니다. 예를 들어 브로커 A가 비 전이중 모드로 브로커 B와의 연결을 생성한 경우 메시지는 브로커 A에서 브로커 B로만 전달될 수 있습니다. 하지만 브로커 A가 브로커 B와 전이중 연결을 생성한 경우에는 브로커 B가 `<networkConnector>`를 구성할 필요 없이 브로커 A로 메시지를 전달할 수 있습니다.

기본값: `false`

이름

브로커 네트워크 내 브리지의 이름.

기본값: `bridge`

uri

브로커 네트워크를 구성하는 두 브로커 중 하나(또는 여러 브로커)의 와이어 레벨 프로토콜 엔드포인트.

기본값: `null`

사용자 이름

브로커 네트워크의 브로커에 공통된 사용자 이름.

기본값: `null`

구성의 예

> **Note**
>
> `networkConnector`를 사용하여 브로커 네트워크를 정의할 때 브로커에 공통된 사용자의 암호를 포함시키지 마세요.

2개 브로커로 구성된 브로커 네트워크

이 구성에서는 2개의 브로커가 브로커 네트워크로 연결되어 있습니다. 네트워크 커넥터의 이름은 `connector_1_to_2`이고, 브로커에 공통된 사용자 이름은 `myCommonUser`이고, 연결은 `duplex`이고, OpenWire 엔드포인트 URI는 접두사 `static:`을 사용하여 브로커 간 일대일 연결을 나타냅니다.

```
<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>
```

자세한 정보는 Configure Network Connectors for Your Broker을 참조하세요.

여러 브로커로 구성된 브로커 네트워크

이 구성에서는 여러 개의 브로커가 브로커 네트워크로 연결되어 있습니다. 네트워크 커넥터의 이름은 `connector_1_to_2`이고, 브로커에 공통된 사용자 이름은 `myCommonUser`이며, 연결은 `duplex`이

고, OpenWire 엔드포인트 URI의 쉼표로 구분된 목록은 접두사 masterslave:을 사용하여 브로커 간 장애 조치 연결을 나타냅니다. 브로커 간 장애 조치는 랜덤화되지 않으며 재연결 시도가 무한정 계속됩니다.

```
<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**
>
> 브로커 네트워크에는 masterslave: 접두사를 사용하는 것이 좋습니다. 이 접두사는 더 명시적인 static:failover:()?randomize=false&maxReconnectAttempts=0 구문과 동일합니다.

> **Note**
>
> 이 XML 구성에는 공백이 허용되지 않습니다.

kahaDB

kahaDB는 persistenceAdapter 하위 컬렉션 요소의 하위입니다.

속성

concurrentStoreAndDispatchQueues

대기열에 동시 저장 및 디스패치를 사용할지 여부를 지정합니다. 자세한 정보는 느린 소비자를 통해 대기열 동시 저장 및 디스패치 비활성화을 참조하세요.

기본값: true

cleanupOnStop

> 지원 버전:
>
> Apache ActiveMQ 15.16.x 이상

비활성화된 경우 브로커가 중지될 때 가비지 수집 및 정리가 수행되지 않으므로 종료 프로세스가 빨라집니다. 속도가 증가하면 대형 데이터베이스 또는 스케줄러 데이터베이스인 경우 유용합니다.

기본값: `true`

journalDiskSyncInterval

`journalDiskSyncStrategy=periodic`의 경우 디스크 동기화를 수행할 때의 간격(ms)입니다. 자세한 내용은 Apache ActiveMQ kahaDB 설명서를 참조하세요.

기본값: `1000`

journalDiskSyncStrategy

> 지원 버전:
>
> Apache ActiveMQ 15.14.x 이상

디스크 동기화 정책을 구성합니다. 자세한 내용은 Apache ActiveMQ kahaDB 설명서를 참조하세요.

기본값: `always`

> Note
>
> ActiveMQ 설명서에는 데이터 손실이 `journalDiskSyncInterval`의 시간으로 제한된다고 기술되어 있습니다. 즉, 기본값은 1초입니다. 데이터 손실은 해당 간격보다 길 수는 있지만 정확하게 명시하기는 어렵습니다. 사용 시 주의해야 합니다.

preallocationStrategy

새 저널 파일이 필요한 경우 브로커가 저널 파일을 미리 할당하는 방법을 구성합니다. 자세한 내용은 Apache ActiveMQ kahaDB 설명서를 참조하세요.

기본값: `sparse_file`

구성의 예제

Example

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

systemUsage

`systemUsage`는 `systemUsage` 하위 컬렉션 요소의 하위입니다. 생산자 속도가 느려지기 전에 브로커가 사용할 최대 공간을 제어합니다. 자세한 내용은 Apache ActiveMQ 설명서의 생산자 흐름 제어를 참조하세요.

하위 요소

memoryUsage

`memoryUsage`는 `systemUsage` 하위 요소의 하위입니다. 메모리 사용량을 관리합니다. `memoryUsage`를 이용하면 항목이 얼마나 많이 사용되고 있는지 계속 추적할 수 있으므로 작업 세트 사용을 생산적으로 제어할 수 있습니다. 자세한 내용은 Apache Active MQ 설명서의 스키마를 참조하세요.

하위 요소

`memoryUsage`는 `memoryUsage` 하위 요소의 하위입니다.

속성

percentOfJvmHeap

0(포함)에서 70(포함) 사이의 정수입니다.

기본값: `70`

속성

sendFailIfNoSpace

여유 공간이 없는 경우 `send()` 메서드가 실패할지 여부를 설정합니다. 기본값은 false이며 공간을 사용할 수 있을 때까지 `send()` 메서드를 차단합니다. 자세한 내용은 Apache Active MQ 설명서의 스키마를 참조하세요.

기본값: `false`

sendFailIfNoSpaceAfterTimeout

기본값: `null`

구성의 예제

Example

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

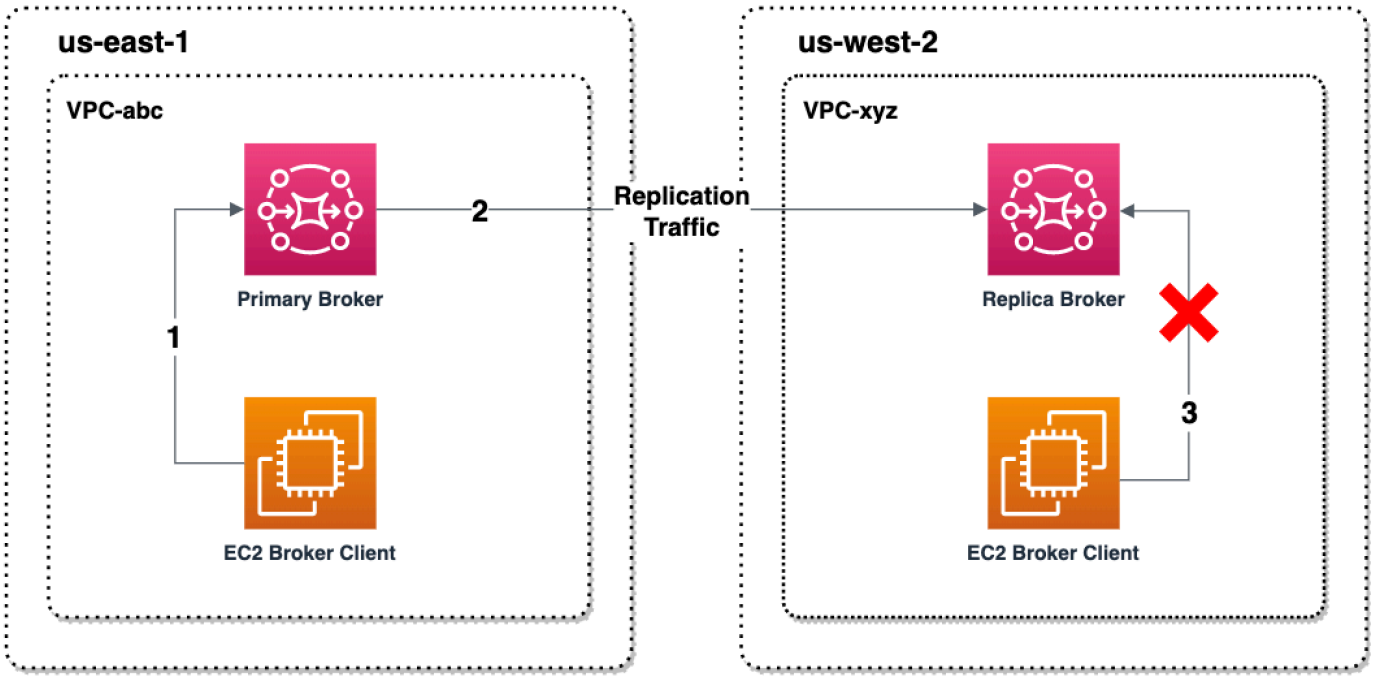
ActiveMQ용 Amazon MQ의 리전 간 데이터 복제

ActiveMQ용 Amazon MQ는 기본 리전의 기본 브로커에서 복제본 AWS 리전의 복제본 브로커로 비동기식 메시지 복제를 허용하는 리전 간 데이터 복제(CRDR) 기능을 제공합니다. ActiveMQ Amazon MQ API에 대한 장애 조치 요청을 실행하면 현재 복제본 브로커가 기본 브로커 역할로 승격되고 현재 기본 브로커는 복제본 역할로 강등됩니다.

리전 간 데이터 복제를 위한 기본 브로커 및 복제본 브로커

기본 AWS 리전의 기본 브로커에서 복제본 리전의 복제본 브로커로 비동기식 데이터 복제를 위한 기본 및 복제본 브로커를 생성할 수 있습니다. 기본 리전은 기본 브로커라고 하는 활성/대기 브로커의 중복 페어로 구성됩니다. 보조 리전은 복제본 브로커라고 하는 활성/대기 브로커의 중복 페어로 구성됩니다.

다음 다이어그램은 보조 리전의 복제본 브로커가 기본 리전의 기본 브로커로부터 비동기 복제 데이터를 수신하는 것을 보여줍니다.



기본 브로커와 복제본 브로커는 리전 간 데이터 복구 솔루션 역할을 합니다. 기본 리전의 기본 브로커에 장애가 발생하면 전환 또는 장애 조치를 시작하여 보조 리전의 복제본 브로커를 기본 브로커로 승격할 수 있습니다. 그러면 이전의 기본 브로커가 복제본 브로커가 되고 이전 복제본 브로커가 기본 브로커로 승격됩니다. 기본 브로커 및 복제본 브로커 생성에 대한 지침은 Amazon MQ 리전 간 데이터 복제 브로커 생성 섹션을 참조하세요.

> **Note**
>
> 활성/대기 브로커에서만 사용할 수 있습니다.
> 미러링된 대기열에는 사용할 수 없습니다.

Amazon MQ 리전 간 데이터 복제 브로커 생성

리전 간 데이터 복제(CRDR)를 사용하면 필요에 따라 두 AWS 리전의 ActiveMQ용 Amazon MQ 메시지 브로커 간에 전환할 수 있습니다. 기존 브로커를 기본 브로커로 지정하고 이 브로커의 복제본을 생성하거나 새 기본 브로커와 복제본 브로커를 함께 생성할 수 있습니다. 그런 다음 Amazon MQ `Promote` API 작업을 사용하여 복제본 브로커를 기본 브로커 역할로 승격할 수 있습니다. 기본 브로커

와 복제본 브로커에 대한 자세한 내용은 리전 간 데이터 복제를 위한 기본 브로커 및 복제본 브로커 섹션을 참조하세요.

다음 지침은 Amazon MQ Management Console을 사용하여 복제본 브로커를 생성하고 구성하는 방법을 설명합니다.

주제

사전 조건

크로스 리전 데이터 복제 기능을 사용하려면 다음 사전 요구 사항을 검토하고 준수해야 합니다.

- 버전: 크로스 리전 데이터 복제 기능은 ActiveMQ용 Amazon MQ 브로커 버전 5.17.6 이상에서만 사용할 수 있습니다.
- 리전: 크로스 리전 데이터 복제는 미국 동부(오하이오), 미국 동부(버지니아 북부), 미국 서부(오레곤) 및 미국 서부(캘리포니아 북부) 리전에서 지원됩니다.
- 인스턴스 유형: 크로스 리전 데이터 복제는 브로커 인스턴스 크기가 `mq.m5.large` 이상인 경우에만 사용할 수 있습니다.
- 배포 유형: 크로스 리전 데이터 복제는 다중 가용 영역 배포를 사용하는 액티브/스탠바이 브로커에만 사용할 수 있습니다.
- 브로커 상태: 브로커 상태가 `Running`인 기본 브로커에 대해서만 복제본 브로커를 생성할 수 있습니다.

1단계(선택 사항): 새 기본 브로커 생성

새 기본 브로커 생성

1. Amazon MQ 콘솔에 로그인합니다.
2. Amazon MQ 콘솔의 브로커 페이지에서 브로커 생성을 선택합니다.
3. Select broker engine(브로커 엔진 선택) 페이지에서 Apache ActiveMQ를 선택합니다.
4. Select deployment and storage(배포 및 스토리지 선택) 페이지의 Deployment mode and storage type(배포 모드 및 스토리지 유형) 섹션에서 다음을 수행합니다.

- 배포 모드에서 활성/대기 브로커를 선택합니다. 활성/대기 브로커는 두 개의 서로 다른 가용 영역에 있는 두 개의 브로커가 중복 페어로 구성됩니다. 이러한 브로커는 애플리케이션 및 Amazon EFS와 동기식으로 통신합니다. 자세한 내용은 ActiveMQ용 Amazon MQ 브로커의 배포 옵션 섹션을 참조하세요.

5. 다음을 선택합니다.
6. Configure settings(설정 구성) 페이지의 세부 정보 섹션에서 다음을 선택합니다.

   a. Broker name(브로커 이름)을 입력합니다.

   > **Important**
   >
   > 개인 식별 정보(PII)나 기타 기밀 정보 또는 민감한 정보를 브로커 이름에 추가하지 마십시오. 브로커 이름을 통해 CloudWatch Logs를 포함하여 다른 AWS 서비스에 액세스할 수 있습니다. 브로커 이름은 개인 데이터나 민감한 데이터에 사용하기 위한 것이 아닙니다.

   b. Broker instance type(브로커 인스턴스 유형)을 선택합니다(예: mq.m5.large). 자세한 정보는 Broker instance types을 참조하세요.

7. ActiveMQ Web Console access(ActiveMQ 웹 콘솔 액세스) 섹션에서 Username(사용자 이름) 및 Password(암호)를 입력합니다. 브로커 사용자 이름과 암호에는 다음 제한이 적용됩니다.

   - 사용자 이름은 영숫자, 대시, 마침표, 밑줄 및 물결 기호(- . _ ~)만 포함할 수 있습니다.
   - 암호는 최소 12자 길이이고 최소 4개의 고유 문자가 있어야 하며 쉼표, 콜론 또는 등호(,:=)는 포함할 수 없습니다.

   > **Important**
   >
   > 개인 식별 정보(PII)나 기타 기밀 정보 또는 민감한 정보를 브로커 사용자 이름에 추가하지 마십시오. 브로커 사용자 이름을 통해 CloudWatch Logs를 포함하여 다른 AWS 서비스에 액세스할 수 있습니다. 브로커 사용자 이름은 개인 데이터나 민감한 데이터에 사용하기 위한 것이 아닙니다.

페이지 상단의 녹색 플래시 막대는 Amazon MQ가 복구 리전에서 복제본 브로커를 생성하고 있음을 나타냅니다. 브로커의 CRDR 역할 및 RPO 상태도 확인할 수 있습니다. CRDR 역할 및 RPO 상태 열을

해제하려면 브로커 테이블의 오른쪽 상단에 있는 기어 모양 아이콘을 선택합니다. 그런 다음 기본 설정 페이지에서 CRDR 역할 또는 RPO 상태를 해제합니다.

2단계: 기존 브로커의 복제본 생성

1. Amazon MQ 콘솔의 브로커 페이지에서 복제본 브로커 생성을 선택합니다.
2. 기본 브로커 선택 페이지에서 CRDR 기본 브로커로 사용할 기존 브로커를 선택합니다. 그리고 다음을 선택합니다.
3. 복제본 브로커 구성 페이지에서 드롭다운 메뉴를 사용하여 복제본 리전을 선택합니다.
4. 복제본 브로커용 ActiveMQ 콘솔 사용자 섹션에서 복제본 브로커 콘솔 사용자의 사용자 이름 및 암호를 입력합니다. 브로커 사용자 이름과 암호에는 다음 제한이 적용됩니다.

   - 사용자 이름은 영숫자, 대시, 마침표, 밑줄 및 물결 기호(- . _ ~)만 포함할 수 있습니다.
   - 암호는 최소 12자 길이이고 최소 4개의 고유 문자가 있어야 하며 쉼표, 콜론 또는 등호(,:=)는 포함할 수 없습니다.

   > **Important**
   >
   > 개인 식별 정보(PII)나 기타 기밀 정보 또는 민감한 정보를 브로커 사용자 이름에 추가하지 마십시오. 브로커 사용자 이름을 통해 CloudWatch Logs를 포함하여 다른 AWS 서비스에 액세스할 수 있습니다. 브로커 사용자 이름은 개인 데이터나 민감한 데이터에 사용하기 위한 것이 아닙니다.

5. 브로커 간 액세스를 연결하는 데이터 복제 사용자 섹션에서 기본 브로커와 복제본 브로커 모두에 액세스할 사용자의 사용자 이름 및 암호를 입력합니다. 브로커 사용자 이름과 암호에는 다음 제한이 적용됩니다.

   - 사용자 이름은 영숫자, 대시, 마침표, 밑줄 및 물결 기호(- . _ ~)만 포함할 수 있습니다.
   - 암호는 최소 12자 길이이고 최소 4개의 고유 문자가 있어야 하며 쉼표, 콜론 또는 등호(,:=)는 포함할 수 없습니다.

   > **Important**
   >
   > 개인 식별 정보(PII)나 기타 기밀 정보 또는 민감한 정보를 브로커 사용자 이름에 추가하지 마십시오. 브로커 사용자 이름을 통해 CloudWatch Logs를 포함하여 다른 AWS 서비스에

액세스할 수 있습니다. 브로커 사용자 이름은 개인 데이터나 민감한 데이터에 사용하기 위한 것이 아닙니다.

추가 설정을 구성합니다. 그리고 다음을 선택합니다.

6. 검토 및 생성 페이지에서 복제본 브로커 세부 정보를 검토합니다. 복제본 브로커 생성을 선택합니다.
7. 그런 다음 기본 브로커를 재부팅합니다. 이렇게 하면 복제본 브로커도 재부팅됩니다. 브로커 재부팅에 대한 지침은 Rebooting a Broker 섹션을 참조하세요.

ActiveMQ 브로커의 추가 설정 구성에 대한 자세한 내용은 시작하기: ActiveMQ 브로커 생성 및 연결 섹션을 참조하세요.

Amazon MQ 리전 간 데이터 복제 브로커 삭제

기본 또는 복제본 CRDR(리전 간 데이터 복제) 브로커를 삭제하려면 먼저 브로커의 페어링을 해제한 다음 재부팅해야 합니다. 다음 지침은 AWS Management Console을 사용하여 브로커의 페어링을 해제하고 재부팅하는 방법을 보여줍니다.

1. 브로커 페이지에서 페어링을 해제하려는 CRDR 브로커를 선택한 다음 편집을 선택합니다.
2. 브로커 편집 페이지의 데이터 복제 섹션에서 브로커 페어링 해제를 선택합니다.
3. 팝업 창에 "confirm"을 입력하여 선택을 확인합니다. 브로커 페어링 해제를 선택합니다.
4. 그런 다음 페어링 해제된 기본 브로커를 재부팅합니다. 이렇게 하면 복제본 브로커도 재부팅됩니다. 브로커 재부팅에 대한 지침은 Rebooting a Broker 섹션을 참조하세요. 기본 브로커가 재부팅되면 두 브로커 모두 페어링이 해제되며 개별적으로 삭제할 수 있습니다. 브로커를 삭제하려면 Deleting a broker 섹션을 참조하세요.

Amazon MQ 복제본 브로커를 기본 브로커 역할로 승격하기 위한 전환 또는 장애 조치 시작

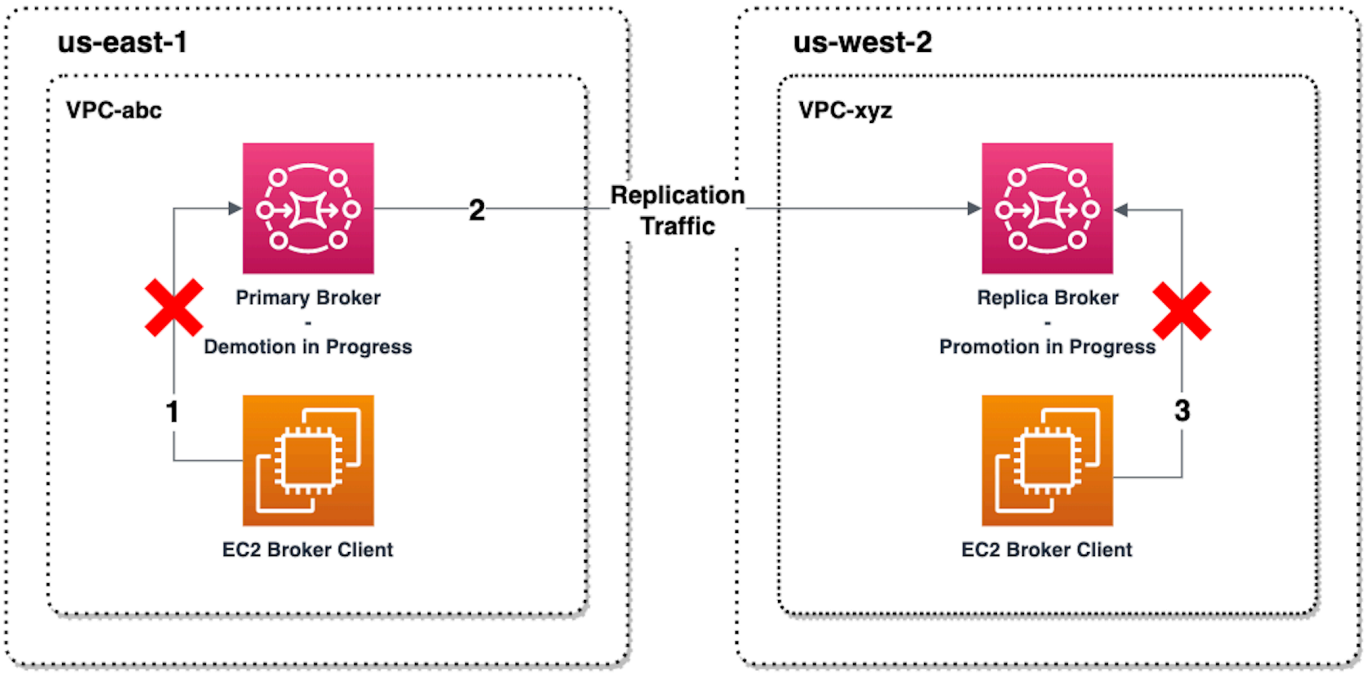
복제본 브로커를 기본 브로커 역할로 승격하려는 경우 전환 또는 장애 조치를 시작할 수 있습니다. 복제본 브로커를 승격하면 기본 브로커가 복제본 브로커 역할로 강등됩니다.

전환은 가용성보다 일관성을 우선시합니다. 이 장애 조치 작업이 완료될 때 브로커들은 동일한 상태가 됩니다. 전환을 수행하면 브로커 간 일관성이 설정되는 동안 클라이언트 연결에 어느 브로커도 사용할

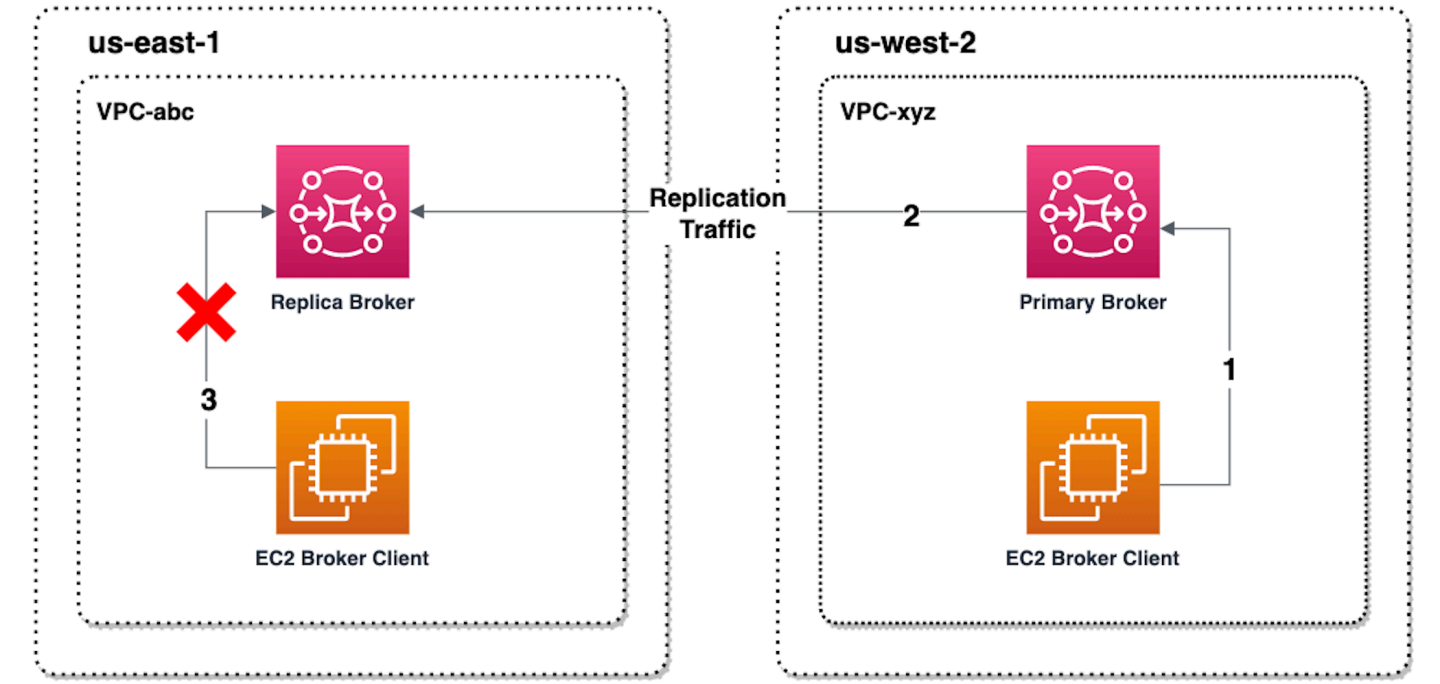
수 없는 기간이 생길 수 있습니다. 복제본이 승격되는 순간 두 브로커가 모두 동일한 상태가 됩니다. 전환의 성공 여부는 두 리전의 상태와 리전 간 네트워크에 좌우됩니다.

장애 조치는 일관성보다 가용성을 우선시합니다. 이 작업이 완료될 때 브로커가 동일한 상태로 유지되지 않을 수 있습니다. 장애 조치를 수행하면 복제 데이터가 동기화되거나 기본 브로커가 종료 신호를 수신할 때까지 기다리지 않고도 복제본 브로커를 클라이언트 트래픽 처리에 즉시 사용할 수 있습니다. 장애 조치의 성공 여부는 원래 기본 리전의 상태나 리전 간 네트워크에 좌우되지 않습니다.

다음 다이어그램은 전환에서 복제 대기열이 비워지고 브로커 상태가 동기화되는 동안 어느 브로커도 클라이언트 연결을 수락하지 않는 것을 보여줍니다. 이 프로세스에서 기본 브로커의 VPC에 있는 클라이언트는 작업이 진행되는 동안 더 이상의 상태 변경을 생성할 수 없으며, 기본 브로커는 복제본으로 강등됩니다. 복제 대기열이 비워지고 두 브로커가 동일한 상태가 되면 복제본 브로커의 VPC에 있는 클라이언트는 장애 조치 작업이 완료되고 복제본 브로커가 기본 브로커로 승격될 때까지 복제본 브로커에 연결할 수 없습니다.



다음 다이어그램은 전환 프로세스가 완료된 후의 브로커 상태를 보여줍니다. 이제 원래 복제본 브로커가 기본 브로커 역할로 승격되었으며 클라이언트 연결을 수락하고 있습니다. 클라이언트는 브로커로부터 데이터를 생성하고 소비할 수 있습니다.



콘솔을 사용하여 복제본 브로커 승격

전환 또는 장애 조치를 사용하여 복제본 브로커를 승격하려면 Amazon MQ 콘솔에서 다음 단계를 따릅니다.

> **Note**
>
> 기본 브로커에서는 전환 또는 장애 조치를 시작할 수 없습니다.

1. 복제본 브로커의 리전으로 전환합니다. 브로커 테이블에서 기본 브로커로 승격할 기존 복제본 브로커를 선택합니다.
2. 브로커 세부 정보 페이지에서 다음을 수행합니다.
    1. 복제본 승격을 선택합니다.
    2. 팝업 창에서 전환 또는 장애 조치를 선택합니다.
    3. 텍스트 상자에 'confirm'을 입력하여 선택을 확인합니다.
    4. 확인을 선택합니다.

장애 조치가 시작되면 브로커 상태가 장애 조치 진행 중으로 변경됩니다. 장애 조치가 완료되면 브로커 페이지 상단의 파란색 진행률 표시줄이 녹색으로 바뀝니다.

> **Note**
>
> 구성은 복제본 브로커가 생성될 때만 복제됩니다. 이후 업데이트는 복제되지 않습니다.

Amazon CloudWatch의 리전 간 데이터 복제 지표

ActiveMQ용 Amazon MQ 리전 간 데이터 복제 기능은 기본 브로커와 복제본 브로커의 신뢰성, 가용성 및 성능을 유지하기 위한 지표를 제공합니다. 복제 프로세스 중에 보조 리전의 복제본 브로커는 기본 리전의 기본 브로커로부터 비동기식으로 복제된 데이터를 받습니다. 기본 리전의 기본 브로커에 장애가 발생하면 전환 또는 장애 조치를 시작하여 보조 리전의 복제본 브로커를 기본 브로커로 승격할 수 있습니다. Amazon CloudWatch에서 지표를 보는 방법에 대한 지침은 Amazon MQ의 CloudWatch 지표 액세스 섹션을 참조하세요.

CRDR 타임스탬프

다음 타임스탬프는 Amazon CloudWatch에 나오는 지표가 계산되는 방식을 설명합니다. 데이터 복제 프로세스에는 5개의 타임스탬프가 있습니다.

- 현재 관측 시간(TCO): 현재 시점입니다.
- 생성 시간(TC): 기본 브로커에 의해 복제 대기열에 이벤트가 생성된 시점입니다. 기본 브로커와 복제본 브로커 모두에서 사용할 수 있습니다.
- 전송 시간(TD): 이벤트가 복제본 브로커에 성공적으로 전달된 시점입니다. 복제본 브로커에서만 사용할 수 있습니다.
- 처리 시간(TP): 복제본 브로커에 의해 이벤트가 성공적으로 처리된 시점입니다. 복제본 브로커에서만 사용할 수 있습니다.
- 승인 시간(TA): 기본 브로커에 의해 이벤트가 성공적으로 승인된 시점입니다. 기본 브로커에서만 사용할 수 있습니다.

CRDR CloudWatch 지표를 사용한 전환/장애 조치 성능 추정

Amazon MQ는 기본적으로 브로커에 대한 지표를 활성화합니다. Amazon CloudWatch 콘솔에 액세스하거나 CloudWatch API를 사용하여 브로커 지표를 볼 수 있습니다. 다음 지표는 CRDR 브로커의 복제 및 전환/장애 조치 성능을 파악하는 데 유용합니다.

Amazon MQ CloudWatch 지표	CRDR 사용 이유	
TotalReplicationLag	기본 브로커에서 승인되지 않은 마지막 이벤트의 TA와 TC 사이의 예상 시간입니다.	
ReplicationLag	복제본 브로커에서 승인되지 않은 마지막 이벤트의 TP와 TC 사이의 예상 시간입니다.	
PrimaryWaitTime	기본 브로커에서 처리된 마지막 이벤트의 TCO와 TC 사이의 예상 시간입니다.	
ReplicaWaitTime	복제본 브로커에서 처리된 마지막 이벤트의 TCO와 TP 사이의 예상 시간입니다.	
QueueSize	기본 브로커의 복제 대기열에 있는 승인되지 않은 이벤트의 총 수입니다.	

TotalReplicationLag 및 ReplicationLag는 기본 브로커와 복제본 브로커 간의 지연된 복제를 설명합니다. 또한 두 지표를 사용하여 진행 중인 전환 또는 장애 조치 작업이 완료될 때까지의 시간을 예측할 수 있습니다.

PrimaryWaitTime 및 ReplicaWaitTime을 사용하여 복제 프로세스에서 진행 중인 문제를 식별할 수 있습니다. 지표 값이 계속 증가하는 경우 복제 프로세스가 성능 저하 상태이거나 일시 중지된 것일 수 있습니다. 네트워크 파티셔닝, 브로커 시작 및 긴 복구와 같은 문제로 인해 복제 속도가 느려질 수 있습니다.

ActiveMQ 자습서

다음 자습서에서는 ActiveMQ 브로커를 만들고 연결하는 방법을 보여줍니다. ActiveMQ Java 예제 코드를 사용하려면 Java Standard Edition Development Kit를 설치하고 코드를 일부 수정해야 합니다.

주제

브로커의 Amazon MQ 네트워크 생성 및 구성

브로커 네트워크는 동시에 여러 활성 단일 인스턴스 브로커 또는 활성/대기 브로커로 구성됩니다. 이 자습서에서는 소스 및 싱크 토폴로지를 사용하여 2개 브로커로 구성된 브로커 네트워크를 생성하는 방법을 알아봅니다.

개념 개요 및 세부 구성 정보는 다음 단원을 참조하세요.

- Amazon MQ 브로커 네트워크
- 브로커 네트워크를 올바로 구성
- `networkConnector`
- `networkConnectionStartAsync`
- ActiveMQ 설명서의 브로커 네트워크

Amazon MQ 콘솔을 사용하여 브로커의 Amazon MQ 네트워크를 생성할 수 있습니다. 2개 브로커를 동시에 생성하기 시작할 수 있으므로 이 프로세스는 약 15분이 소요됩니다.

주제

사전 조건

브로커 네트워크를 생성하려면 다음이 구비되어 있어야 합니다.

- 2개 이상의 동시 활성 브로커(이 자습서에서는 각각 `MyBroker1` 및 `MyBroker2`로 명명). 브로커 생성에 대한 자세한 내용은 시작하기: ActiveMQ 브로커 생성 및 연결 단원을 참조하세요.
- 두 브로커가 동일한 VPC 또는 피어링된 VPC에 있어야 합니다. VPC에 대한 자세한 내용은 Amazon VPC 사용 설명서의 Amazon VPC란 무엇이가요? 및 Amazon VPC 피어링 가이드의 VPC 피어링이란 무엇인가요?를 참조하세요.

  > **Important**
  >
  > 기본 VPC, 서브넷 또는 보안 그룹이 없는 경우 이들 항목부터 생성해야 합니다. 자세한 내용은 Amazon VPC 사용 설명서에서 다음 항목을 참조하세요.
  >
  > - 기본 VPC 만들기
  > - 기본 서브넷 생성
  > - 보안 그룹 생성

- 두 브로커에 대해 동일한 로그인 보안 인증 정보를 가진 두 명의 사용자입니다. 사용자 생성에 대한 자세한 내용은 ActiveMQ 브로커 사용자 생성 단원을 참조하세요.

  > **Note**
  >
  > LDAP 인증을 브로커 네트워크와 통합할 때는 사용자가 LDAP 사용자뿐 아니라 ActiveMQ 브로커로도 존재하는지 확인합니다.

다음 예제에서는 2개의 단일 인스턴스 브로커를 사용합니다. 하지만 활성/대기 브로커를 사용하거나 브로커 배포 모드를 조합하여 브로커 네트워크를 생성할 수 있습니다.

1단계: 브로커 간 트래픽 허용

브로커를 생성한 후 브로커 사이에 트래픽을 허용해야 합니다.

1. Amazon MQ 콘솔에서 MyBroker2 페이지의 Details(세부 정보) 섹션에 있는 Security and network(보안 및 네트워크) 아래에서 보안 그룹 또는 을 선택합니다.

EC2 Dashboard의 보안 그룹 페이지가 표시됩니다.

2. 보안 그룹 목록에서 보안 그룹을 선택합니다.
3. 페이지 하단에서 인바운드를 선택한 후 편집을 선택합니다.
4. 인바운드 규칙 편집 대화 상자에서 OpenWire 엔드포인트에 대한 규칙을 추가합니다.

   a. 규칙 추가(Add Rule)를 선택합니다.

   b. 유형에서 Custom TCP(사용자 지정 TCP)를 선택합니다.

   c. 포트 범위에 OpenWire 포트(61617)를 입력합니다.

   d. 다음 중 하나를 수행하세요.

   - 특정 IP 주소에 대한 액세스를 제한하려면 소스에서 사용자 지정은 선택한 채로 두고 `MyBroker1`의 IP 주소와 `/32`를 차례로 입력합니다. (그러면 IP 주소가 유효한 CIDR 레코드로 변환됩니다.) 자세한 내용은 탄력적 네트워크 인터페이스를 참조하세요.

     > **Tip**
     >
     > `MyBroker1`의 IP 주소를 검색하려면, Amazon MQ 콘솔에서 브로커 이름을 선택하고 Details(세부 정보) 섹션으로 이동합니다.

   - 모든 브로커가 프라이빗이고 동일한 VPC에 속한 경우 소스에서 사용자 지정은 선택한 채로 두고 현재 편집 중인 보안 그룹의 ID를 입력합니다.

     > **Note**
     >
     > 퍼블릭 브로커의 경우 IP 주소를 사용하여 액세스를 제한해야 합니다.

   e. 저장을 선택합니다.

   이제 브로커가 인바운드 연결을 허용할 수 있습니다.

2단계: 브로커에 대한 네트워크 커넥터 구성

브로커 사이의 트래픽을 허용한 후 둘 중 하나에 대해 네트워크 커넥터를 구성해야 합니다.

1. 브로커 `MyBroker1`의 구성 개정을 편집합니다.

   a. MyBroker1 페이지에서 편집을 선택합니다.

b. Edit MyBroker1 페이지의 구성 섹션에서 보기를 선택합니다.

   구성이 사용하는 브로커 엔진 유형 및 버전(예: Apache ActiveMQ 5.15.0)이 표시됩니다.

c. [Configuration details] 탭에 구성 개정 번호, 설명 및 XML 형식의 브로커 구성이 표시됩니다.

d. 구성 편집을 선택합니다.

e. 구성 파일의 하단에서 <networkConnectors> 섹션에 입력된 주석을 제거하고 다음 정보를 포함시킵니다.

   - 네트워크 커넥터의 name.
   - 두 브로커에 공통된 ActiveMQ 웹 콘솔 username.
   - duplex 연결을 활성화합니다.
   - 다음 중 하나를 수행하세요.
     - 브로커를 단일 인스턴스 브로커에 연결하는 경우 MyBroker2에 static: 접두사 및 OpenWire 엔드포인트 uri를 사용합니다. 예를 들면 다음과 같습니다.

```
<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>
```

     - 브로커를 활성/대기 브로커에 연결하는 경우 쿼리 파라미터 ?randomize=false&maxReconnectAttempts=0과 함께 두 브로커 모두에 static+failover 전송 및 OpenWire 엔드포인트 uri를 사용합니다. 예제:

```
<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 사용자의 로그인 보안 인증 정보는 포함하지 마세요.

f. 저장을 선택합니다.

g. Save revision(개정 저장) 대화 상자에서 `Add network of brokers connector for MyBroker2`를 입력합니다.

h. 저장을 선택하여 구성의 새 개정을 저장합니다.

2. `MyBroker1`를 편집하여 최신 구성이 즉시 적용되도록 설정합니다.

a. MyBroker1 페이지에서 편집을 선택합니다.

b. Edit MyBroker1(MyBroker1 편집) 페이지의 구성 섹션에서 Schedule Modifications(수정 예약)를 선택합니다.

c. Schedule broker modifications(브로커 수정 예약) 섹션에서 즉시 수정 사항 적용을 선택합니다.

d. 적용을 선택합니다.

`MyBroker1`이 다시 부팅되고 구성 개정이 적용됩니다.

브로커 네트워크가 생성됩니다.

다음 단계

브로커 네트워크를 구성한 후 메시지를 생산하고 소비하여 테스트할 수 있습니다.

> **Important**
>
> 포트 8162(ActiveMQ 웹 콘솔용) 및 포트 61617(OpenWire 엔드포인트용)에서 브로커 `MyBroker1`에 대해 로컬 시스템에서의 인바운드 연결을 활성화해야 합니다.
> 생산자와 소비자가 브로커 네트워크에 연결하도록 허용하기 위해 보안 그룹 설정을 조정해야 할 수 있습니다.

1. Amazon MQ 콘솔에서 Connections(연결) 섹션으로 이동하여 브로커 `MyBroker1`의 ActiveMQ 웹 콘솔 엔드포인트를 기록합니다.

2. 브로커 MyBroker1의 ActiveMQ 웹 콘솔로 이동합니다.
3. 네트워크 브리지가 연결되었는지 확인하기 위해 네트워크를 선택합니다.

   네트워크 브리지(Network Bridges) 섹션에서 MyBroker2의 이름 및 주소가 원격 브로커(Remote Broker) 및 원격 주소(Remote Address) 열에 나열됩니다.
4. 브로커 MyBroker2에 액세스할 수 있는 아무 시스템에서 소비자를 생성합니다. 예:

   ```
   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
   ```

   소비자가 MyBroker2의 OpenWire 엔드포인트에 연결하고 대기열 MyQueue에서 메시지를 소비합니다.
5. 브로커 MyBroker1에 액세스할 수 있는 아무 시스템에서 생산자를 생성하고 몇몇 메시지를 전송합니다. 예:

   ```
   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
   ```

   생산자가 MyBroker1의 OpenWire 엔드포인트에 연결하고 대기열 MyQueue로 지속적 메시지를 생산합니다.

Amazon MQ 브로커에 Java 애플리케이션 연결

Amazon MQ ActiveMQ 브로커를 생성한 후 애플리케이션을 브로커에 연결할 수 있습니다. 다음 예는 JMS(Java Message Service)를 사용하여 브로커 연결을 생성하고, 대기열을 생성하고, 메시지를 전송하는 방법을 보여줍니다. 완전한 실제 Java 예제는 Working Java Example 단원을 참조하세요.

다양한 ActiveMQ 클라이언트를 사용하여 ActiveMQ 브로커에 연결할 수 있습니다. ActiveMQ 클라이언트를 사용하는 것이 좋습니다.

주제

사전 조건

VPC 속성 활성화

VPC 내에서 브로커에 액세스할 수 있도록 하려면 `enableDnsHostnames` 및 `enableDnsSupport` VPC 속성을 활성화해야 합니다. 자세한 내용은 Amazon VPC 사용 설명서의 VPC에서 DNS 지원을 참조하세요.

인바운드 연결 활성화

다음으로 애플리케이션에 대한 인바운드 연결을 활성화합니다.

1. Amazon MQ 콘솔에 로그인합니다.
2. 브로커 목록에서 브로커 이름(예: MyBroker)을 선택합니다.
3. ***MyBroker*** 페이지의 Connections(연결) 섹션에서 브로커의 웹 콘솔 URL 및 와이어 레벨 프로토콜의 주소 및 포트를 적어둡니다.
4. 세부 정보 섹션의 보안 및 네트워크에서 보안 그룹의 이름 또는 을 선택합니다.

   EC2 Dashboard의 보안 그룹 페이지가 표시됩니다.
5. 보안 그룹 목록에서 보안 그룹을 선택합니다.
6. 페이지 하단에서 인바운드를 선택한 후 편집을 선택합니다.
7. Edit inbound rules(인바운드 규칙 편집) 대화 상자에서 공개적으로 액세스하고자 하는 모든 URL 또는 엔드포인트에 대한 규칙을 추가합니다. 다음 예제에서는 브로커 웹 콘솔에 대해 이 작업을 수행하는 방법을 보여줍니다.

   a. 규칙 추가(Add Rule)를 선택합니다.

   b. 유형에서 Custom TCP(사용자 지정 TCP)를 선택합니다.

   c. Port Range(포트 범위)에 웹 콘솔 포트(8162)를 입력합니다.

d. Source(소스)에서 Custom(사용자 지정)을 선택한 상태에서 웹 콘솔에 액세스하는 데 사용할 시스템의 IP 주소(예: `192.0.2.1`)를 입력합니다.

e. 저장을 선택합니다.

이제 브로커가 인바운드 연결을 허용할 수 있습니다.

Java 종속성 추가

`activemq-client.jar` 및 `activemq-pool.jar` 패키지를 Java 클래스 경로에 추가합니다. 다음 예제는 Maven 프로젝트의 `pom.xml` 파일 내에 존재하는 이러한 종속성을 보여줍니다.

```
<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`에 대한 자세한 정보는 Apache ActiveMQ 설명서의 Initial Configuration을 참조하세요.

> **Important**
>
> 다음 예제 코드에서 생성자와 소비자는 단일 스레드에서 실행됩니다. 프로덕션 시스템(또는 브로커 인스턴스 장애 조치 테스트)의 경우 생산자와 소비자가 별도의 호스트 또는 스레드에서 실행되는지 확인합니다.

메시지 생산자를 생성하고 메시지를 전송하려면

다음 지침에 따라 메시지 생산자를 생성하고 메시지를 수신합니다.

1. 브로커의 엔드포인트를 사용하여 메시지 생산자에 대한 JMS 풀링된 연결 팩토리를 생성한 다음 팩토리에 대해 `createConnection` 메서드를 호출합니다.

> **Note**
>
> 활성/대기 브로커의 경우 Amazon MQ는 2개의 ActiveMQ 웹 콘솔 URL을 제공하지만, 한 번에 하나의 URL만 활성화됩니다. 마찬가지로 Amazon MQ는 각 와이어 레벨 프로토콜에 대해 2개의 엔드포인트를 제공하지만, 한 번에 각 페어의 한 엔드포인트만 활성화됩니다. `-1` 및 `-2` 접미사는 중복 페어를 나타냅니다. 자세한 정보는 ActiveMQ용 Amazon MQ 브로커의 배포 옵션 단원을 참조하세요.
>
> 와이어 레벨 프로토콜 엔드포인트의 경우 장애 조치 전송을 사용하여 애플리케이션이 다른 엔드포인트와 연결하도록 허용할 수 있습니다.

```
// 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**
>
> 메시지 생산자는 항상 `PooledConnectionFactory` 클래스를 사용해야 합니다. 자세한 정보는 항상 연결 풀 사용을 참조하세요.

2. 세션, `MyQueue`라는 이름의 대기열, 그리고 메시지 생산자를 생성합니다.

```
// 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. 문자열 "Hello from Amazon MQ!"를 생성한 다음 메시지를 전송합니다.

```
// 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. 생산자를 정리합니다.

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

메시지 소비자를 생성하고 메시지를 수신하려면

다음 지침에 따라 메시지 생산자를 생성하고 메시지를 수신합니다.

1. 브로커의 엔드포인트를 사용하여 메시지 생산자에 대한 JMS 연결 팩토리를 생성한 다음 팩토리에 대해 createConnection 메서드를 호출합니다.

```
// 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**
>
> 메시지 소비자는 PooledConnectionFactory 클래스를 사용하면 안 됩니다. 자세한 정보는 항상 연결 풀 사용을 참조하세요.

2. 세션, MyQueue라는 이름의 대기열, 그리고 메시지 소비자를 생성합니다.

```
// 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. 메시지가 오기를 기다리다가 메시지가 도착하면 이를 수신합니다.

```
// 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 메시징 서비스(예: Amazon SQS)와 달리 소비자는 브로커에 지속적으로 연결됩니다.

4. 소비자, 세션 및 연결을 종료합니다.

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

ActiveMQ 브로커와 LDAP 통합

> **Important**
>
> Amazon MQ는 프라이빗 CA에서 발급한 서버 인증서를 지원하지 않습니다.

TLS가 활성화된 상태에서 다음 프로토콜을 사용하여 ActiveMQ 브로커에 액세스할 수 있습니다.

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

Amazon MQ에서는 기본 ActiveMQ 인증과 LDAP 인증 및 권한 부여 중에서 선택하여 사용자 권한을 관리할 수 있습니다. ActiveMQ 사용자 이름 및 암호와 관련된 제한에 대한 자세한 내용은 Users 단원을 참조하세요.

ActiveMQ 사용자 및 그룹이 대기열 및 주제를 사용하도록 권한을 부여하려면 브로커의 구성을 편집해야 합니다. Amazon MQ는 ActiveMQ의 단순 인증 플러그 인을 사용하여 대상에 대한 읽기 및 쓰기를 제한합니다. 자세한 정보와 예제를 보려면 항상 권한 부여 맵 구성 단원과 `authorizationEntry` 단원을 참조하세요.

> **Note**
>
> 현재 Amazon MQ는 클라이언트 인증서 인증을 지원하지 않습니다.

주제

LDAP와 ActiveMQ 통합

Lightweight Directory Access Protocol(LDAP) 서버에 저장된 자격 증명을 통해 Amazon MQ 사용자를 인증할 수 있습니다. 또한 해당 자격 증명을 통해 Amazon MQ 사용자를 추가, 삭제 및 수정하고 주제 및 대기열에 대한 권한을 할당할 수도 있습니다. 브로커 생성, 업데이트 및 삭제와 같은 관리 작업은 계속 IAM 자격 증명이 필요하며 LDAP와 통합되지 않습니다.

LDAP 서버를 사용하여 Amazon MQ 브로커 인증 및 권한 부여를 간소화하고 중앙 집중화하려는 고객은 이 기능을 사용할 수 있습니다. 모든 사용자 자격 증명을 LDAP 서버에 보관하면 중앙 위치를 통해 해당 자격 증명을 저장하고 관리할 수 있어서 시간과 노력이 절약됩니다.

Amazon MQ는 Apache ActiveMQ JAAS 플러그 인을 사용하여 LDAP 지원을 제공합니다. 플러그 인에서 지원하는 Microsoft Active Directory 또는 OpenLDAP와 같은 LDAP 서버는 Amazon MQ에서도 지원됩니다. 플러그 인에 대한 자세한 내용은 Active MQ 설명서의 보안 단원을 참조하세요.

사용자 외에 특정 그룹 또는 사용자를 위한 주제 및 대기열에 대한 액세스 권한도 LDAP 서버를 통해 지정할 수 있습니다. 이렇게 하려면 LDAP 서버에서 주제 및 대기열을 나타내는 항목을 생성한 다음 특정 LDAP 사용자 또는 그룹에 사용 권한을 할당합니다. 그런 다음 LDAP 서버에서 권한 부여 데이터를 검색하도록 브로커를 구성할 수 있습니다.

> **Important**
>
> LDAP를 사용하는 경우 인증은 대/소문자를 구분하지 않지만 사용자 이름에 대한 권한 부여는 대/소문자를 구분합니다.

사전 조건

신규 또는 기존 Amazon MQ 브로커에 LDAP 지원을 추가하려면 먼저 서비스 계정을 설정해야 합니다. 이 서비스 계정은 LDAP 서버에 대한 연결을 시작하는 데 필요하며 해당 연결을 설정할 올바른 권한이 있어야 합니다. 이 서비스 계정이 브로커의 LDAP 인증을 설정합니다. 이후의 모든 클라이언트 연결은 동일한 연결을 통해 인증됩니다.

서비스 계정은 연결을 시작할 수 있는 액세스 권한이 있는 LDAP 서버의 계정입니다. 이는 표준 LDAP 요구 사항이며 서비스 계정 자격 증명을 한 번만 제공하면 됩니다. 연결이 설정되면 이후의 모든 클라이언트 연결은 LDAP 서버를 통해 인증됩니다. 서비스 계정 자격 증명은 Amazon MQ만 액세스할 수 있는 암호화된 형식으로 안전하게 저장됩니다.

ActiveMQ와 통합하려면 LDAP 서버에 특정 디렉터리 정보 트리(DIT)가 필요합니다. 이 구조를 명확하게 보여주는 예제 `ldif` 파일은 ActiveMQ 설명서의 보안 단원에서 LDAP 서버로 다음 LDIF 파일 가져오기를 참조하세요.

LDAP 시작하기

시작하려면 새 Amazon MQ를 생성하거나 기존 브로커 인스턴스를 편집할 때 Amazon MQ 콘솔로 이동하여 LDAP authentication and authorization(LDAP 인증 및 권한 부여)을 선택합니다.

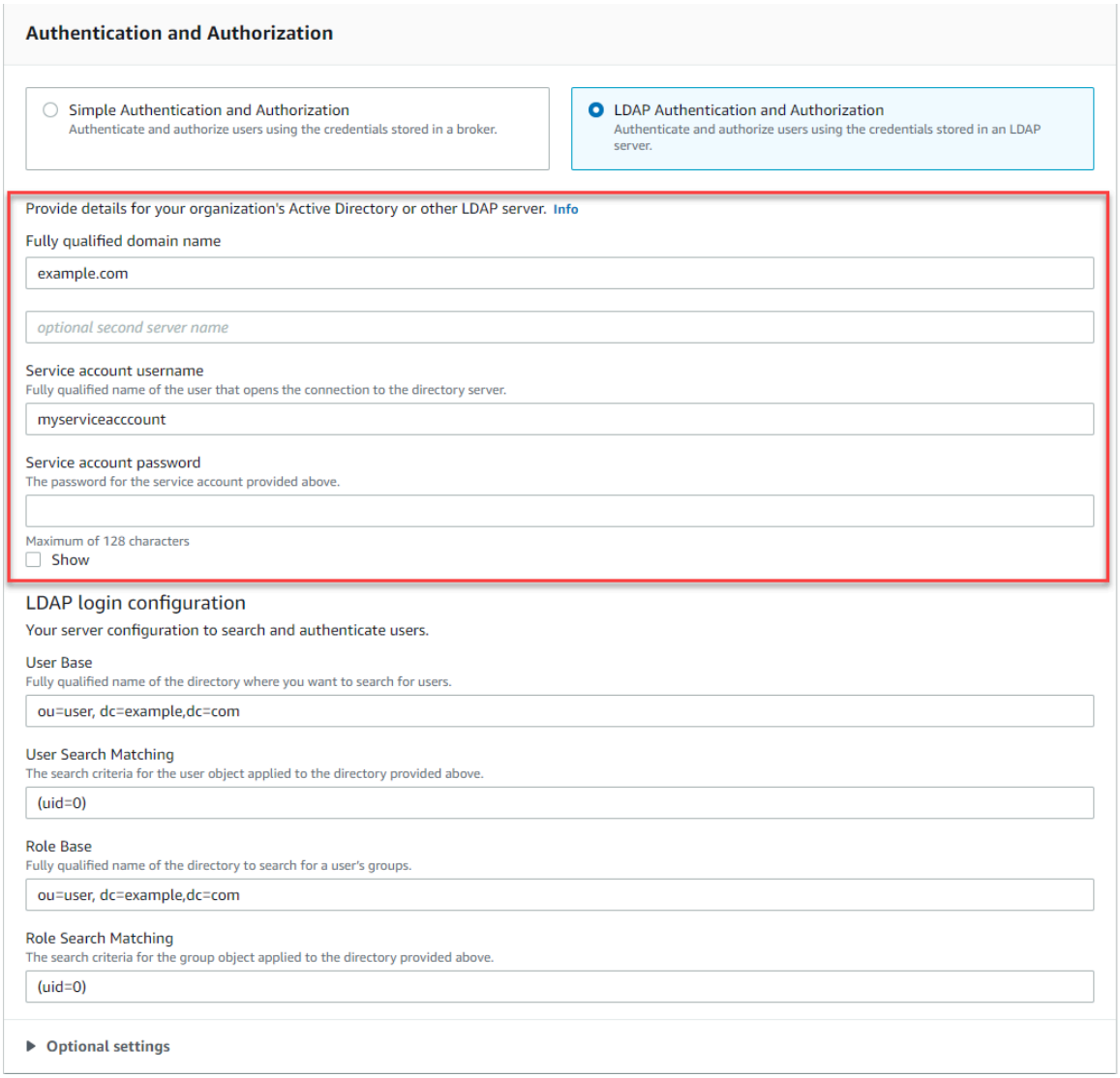
서비스 계정에 대한 다음 정보를 제공합니다.

- 정규화된 도메인 이름: 인증 및 권한 부여 요청을 실행할 LDAP 서버의 위치입니다.

> **Note**
>
> 제공하는 LDAP 서버의 정규화된 도메인 이름에는 프로토콜 또는 포트 번호가 포함되지 않아야 합니다. Amazon MQ에서는 정규화된 도메인 이름 앞에 프로토콜 `ldaps`를 추가하고 뒤에 포트 번호 `636`을 추가합니다.
> 예를 들어 `example.com`과 같은 정규화된 도메인을 제공하는 경우 Amazon MQ는 URL `ldaps://example.com:636`을 사용하여 LDAP 서버에 액세스합니다.
> 브로커 호스트가 LDAP 서버와 통신할 수 있으려면 정규화된 도메인 이름을 공개적으로 확인할 수 있어야 합니다. LDAP 서버를 안전하게 프라이빗으로 유지하려면 브로커의 VPC 내에서 시작된 트래픽만 허용하도록 서버의 인바운드 규칙에서 인바운드 트래픽을 제한합니다.

- 서비스 계정 사용자 이름: LDAP 서버에 대한 초기 바인딩을 수행하는 데 사용할 사용자의 고유 이름입니다.
- 서비스 계정 암호: 초기 바인딩을 수행하는 사용자의 암호입니다.

다음 이미지에서는 이러한 세부 정보를 제공하는 위치를 강조 표시합니다.

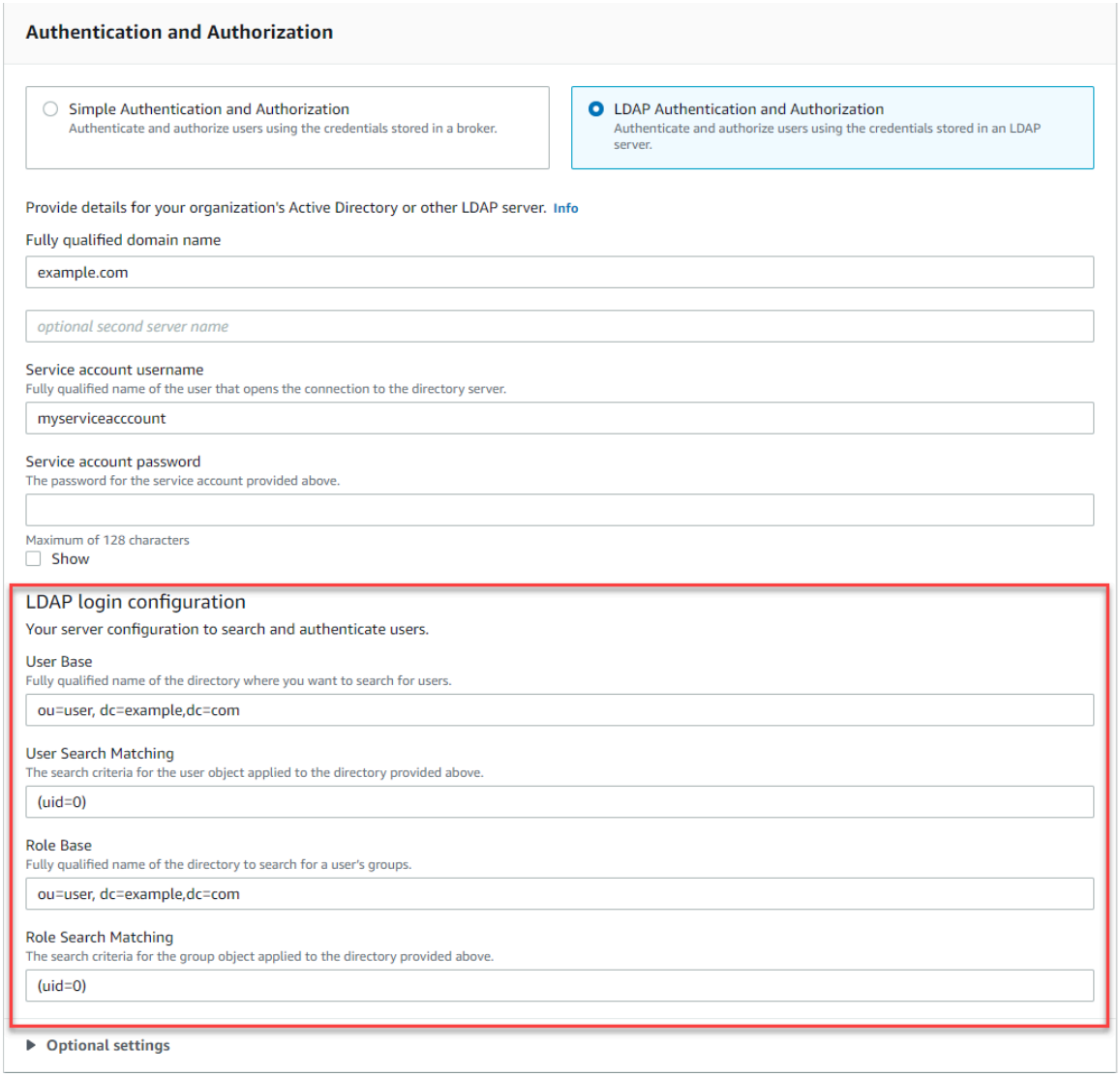


LDAP login configuration(LDAP 로그인 구성) 섹션에서 다음 필수 정보를 제공합니다.

- 사용자 기반: 사용자를 검색할 디렉터리 정보 트리(DIT)에 있는 노드의 고유 이름입니다.
- 사용자 검색 일치: `userBase` 내에서 사용자를 찾는 데 사용되는 LDAP 검색 필터입니다. 클라이언트의 사용자 이름이 검색 필터의 `{0}` 자리 표시자로 대체됩니다. 자세한 내용은 Authentication 및 권한 부여 단원을 참조하세요.
- 역할 기반: 역할을 검색할 DIT에 있는 노드의 고유 이름입니다. 역할은 디렉터리에 있는 명시적 LDAP 그룹 항목으로 구성될 수 있습니다. 일반적인 역할 항목은 일반 이름(CN)과 같은 역할 이름에 대한 하나의 속성 및 역할 그룹에 속한 사용자의 고유 이름 또는 사용자 이름을 나타내는 값이 포함된 또 하나의 속성(예: `member`)으로 구성될 수 있습니다. 예를 들어 조직 구성 단위 `group`의 경우 `ou=group,dc=example,dc=com`과 같은 고유 이름을 제공할 수 있습니다.

- 역할 검색 일치: `roleBase` 내에서 역할을 찾는 데 사용되는 LDAP 검색 필터입니다. `userSearchMatching`과 일치하는 사용자의 고유 이름은 검색 필터의 `{0}` 자리 표시자로 대체됩니다. 클라이언트의 사용자 이름은 `{1}` 자리 표시자로 대체됩니다. 예를 들어 디렉터리의 역할 항목에 해당 역할의 모든 사용자의 사용자 이름이 포함된 `member`라는 속성이 포함된 경우 `(member:=uid={1})`과 같은 검색 필터를 제공할 수 있습니다.

다음 이미지에서는 이러한 세부 정보를 지정하는 위치를 강조 표시합니다.



Optional settings(선택적 설정) 섹션에서 다음과 같은 선택적 정보를 제공할 수 있습니다.

- 사용자 역할 이름: 사용자의 그룹 구성원에 대한 사용자의 디렉토리 항목에 있는 LDAP 속성의 이름입니다. 경우에 따라 사용자 역할은 사용자의 디렉터리 항목에 있는 특성 값으로 식별될 수 있습니

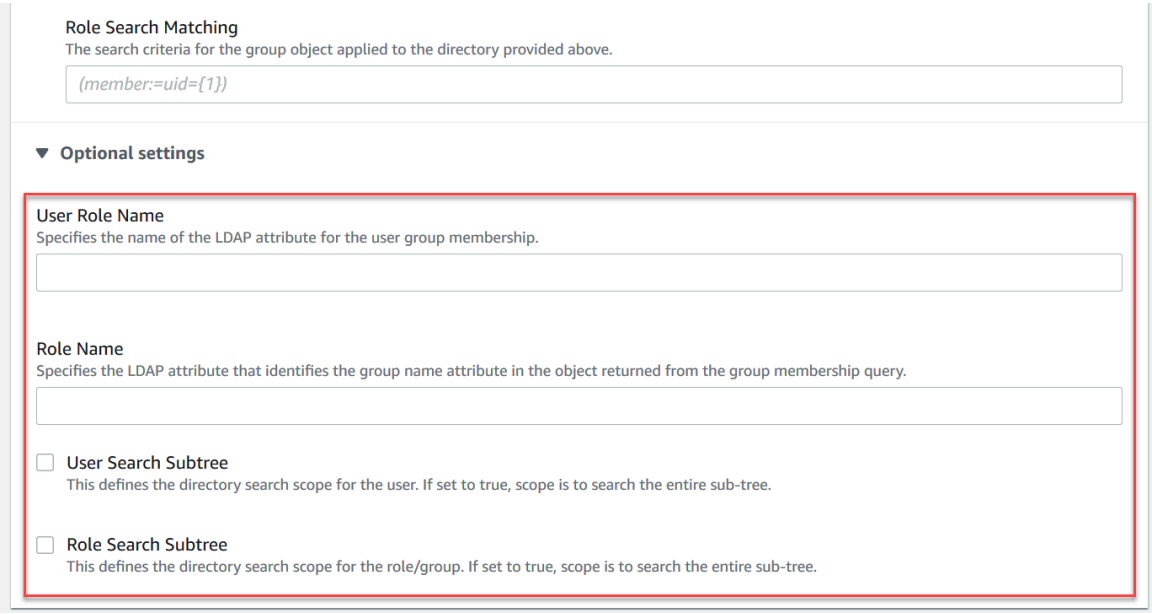
다. 이 userRoleName 옵션을 사용하면 이 속성의 이름을 제공할 수 있습니다. 예를 들어 다음 사용자 항목을 고려해 보겠습니다.

```
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
```

위 예제의 올바른 userRoleName을 제공하려면 memberOf 속성을 지정합니다. 인증이 성공하면 사용자에게 role1 역할이 할당됩니다.

- 역할 이름: 값이 해당 역할의 이름인 역할 항목의 그룹 이름 속성입니다. 예를 들어 그룹 항목의 일반 이름에 대한 cn을 지정할 수 있습니다. 인증이 성공하면 멤버로 속해 있는 각 역할 항목의 cn 속성 값이 사용자에게 할당됩니다.
- 사용자 검색 하위 트리: LDAP 사용자 검색 쿼리의 범위를 정의합니다. True인 경우, userBase에 정의된 노드 아래에 있는 전체 하위 트리를 검색하도록 범위가 설정됩니다.
- 역할 검색 하위 트리: LDAP 역할 검색 쿼리의 범위를 정의합니다. True인 경우, roleBase에 정의된 노드 아래에 있는 전체 하위 트리를 검색하도록 범위가 설정됩니다.

다음 이미지에서는 이러한 선택적 설정을 지정할 위치를 강조 표시합니다.



LDAP 통합의 작동 방식

통합은 인증을 위한 구조와 권한 부여를 위한 구조의 두 가지 기본 범주로 고려할 수 있습니다.

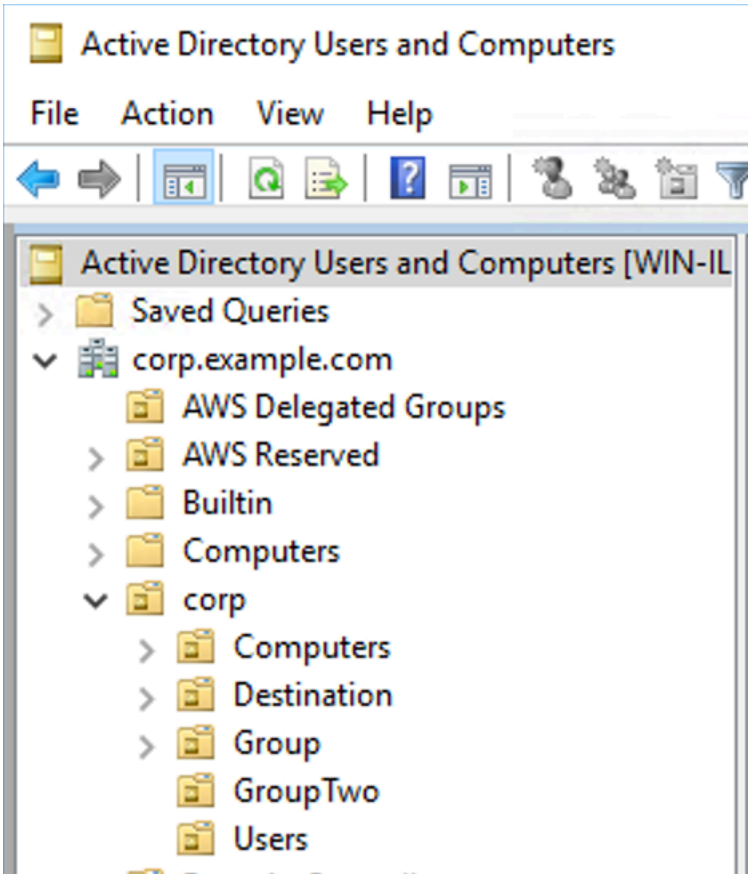
Authentication

인증을 위해서는 클라이언트 자격 증명이 유효해야 합니다. 이러한 자격 증명은 LDAP 서버에 있는 사용자 기반의 사용자에 대해 검증됩니다.

ActiveMQ 브로커에 제공된 사용자 기반은 LDAP 서버에서 사용자가 저장되는 DIT의 노드를 가리켜야 합니다. 예를 들어를 사용 중 AWS Managed Microsoft AD이고 도메인 구성 요소 `corp`, `example`및이 `com`있고 조직 단위 `corp` 및이 있는 경우 다음을 사용자 기반으로 `Users`사용합니다.

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

ActiveMQ 브로커는 브로커에 대한 클라이언트 연결 요청을 인증하기 위해 DIT의 이 위치에서 사용자를 검색합니다.



ActiveMQ 소스 코드는 사용자의 속성 이름을 uid에 하드코딩하므로 각 사용자에게 해당 속성 집합이 있는지 확인해야 합니다. 편의상 사용자의 연결 사용자 이름을 사용할 수 있습니다. 자세한 내용은 activemq 소스 코드 및 Windows Server 2016 이상 버전의 Active Directory 사용자 및 컴퓨터에서 ID 매핑 구성을 참조하세요.

특정 사용자의 ActiveMQ 콘솔 액세스를 활성화하려면 해당 사용자가 amazonmq-console-admins 그룹에 속해 있는지 확인합니다.

권한 부여

권한 부여의 경우 브로커 구성에서 권한 검색 기반을 지정합니다. 권한 부여는 브로커의 activemq.xml 구성 파일에 있는 cachedLdapAuthorizationMap 요소를 통해 대상(또는 와일드카드, 대상 집합)별로 수행됩니다. 자세한 내용은 캐시된 LDAP 권한 부여 모듈을 참조하세요.

> **Note**
>
> 브로커의 activemq.xml 구성 파일에서 cachedLDAPAuthorizationMap 요소를 사용하려면를 통해 구성을 생성할 때 LDAP 인증 및 권한 부여 옵션을 선택하거나,를 통해 구성 생성을 AWS Management Console 설정하거나, Amazon MQ API를 사용하여 새 구성을 생성할 LDAP 때 authenticationStrategy 속성을 로 설정해야 합니다. AWS Management Console

cachedLDAPAuthorizationMap 요소의 일부로 다음 세 가지 속성을 제공해야 합니다.

- queueSearchBase
- topicSearchBase
- tempSearchBase

> **Important**
>
> 중요한 정보가 브로커의 구성 파일에 직접 저장되지 않도록 하기 위해 Amazon MQ는 cachedLdapAuthorizationMap에서 다음 속성을 사용하지 못하도록 차단합니다.
>
> - connectionURL
> - connectionUsername
> - connectionPassword

> 브로커를 생성할 때 Amazon MQ는를 통해 AWS Management Console또는 API 요청의 `ldapServerMetadata` 속성에서 제공하는 값을 위의 속성으로 대체합니다.

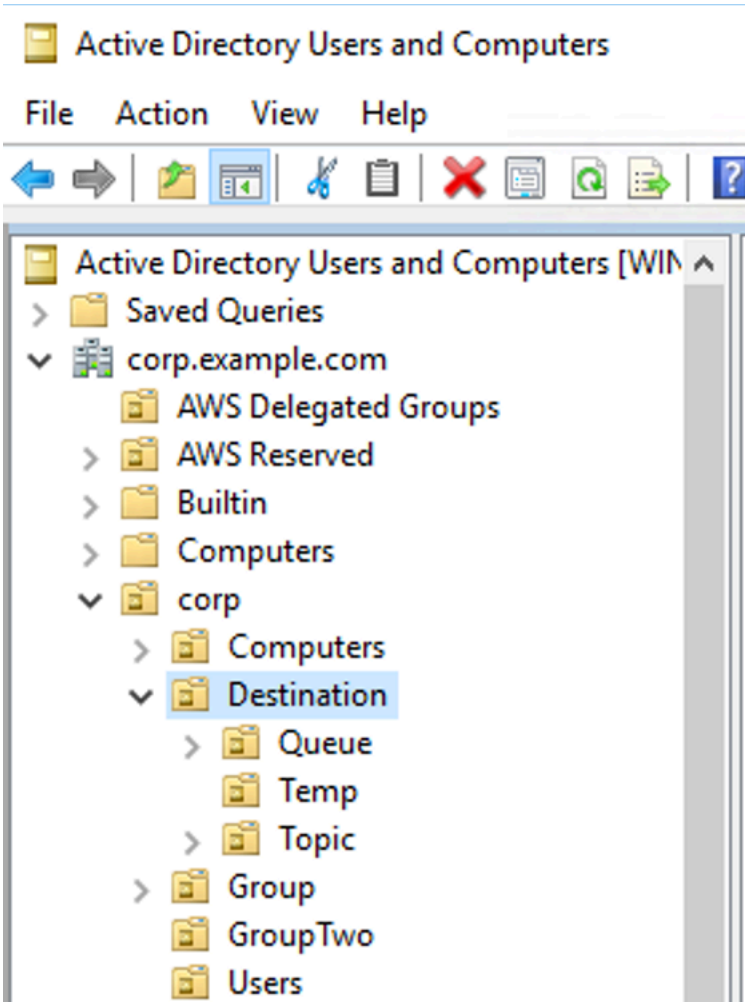
다음은 `cachedLdapAuthorizationMap`의 작동 예제를 보여줍니다.

```
<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>
```

이러한 값은 각 대상 유형의 사용 권한이 지정된 DIT 내의 위치를 식별합니다. 따라서 위의 예제에서 AWS Managed Microsoft AD, `corp` `example`및의 동일한 도메인 구성 요소를 사용하는 `com`경우 모든 대상 유형을 포함하도록 라는 조직 단위`destination`를 지정합니다. 해당 OU 내에서 `queues`, `topics` 및 `temp` 대상에 대해 OU를 하나씩 생성할 수 있습니다.

따라서 대기열 유형 대상의 권한 부여 정보를 제공하는 대기열 검색 기반은 DIT에서 다음 위치에 있게 됩니다.

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



마찬가지로 주제 및 임시 대상의 사용 권한 규칙은 DIT에서 동일한 수준에 있습니다.

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

각 대상 유형(대기열, 주제, 임시)의 OU 내에서 와일드카드 또는 특정 대상 이름을 제공할 수 있습니다. 예를 들어 접두사 DEMO.EVENTS.$.으로 시작하는 모든 대기열의 권한 부여 규칙을 제공하려면 다음 OU를 생성할 수 있습니다.

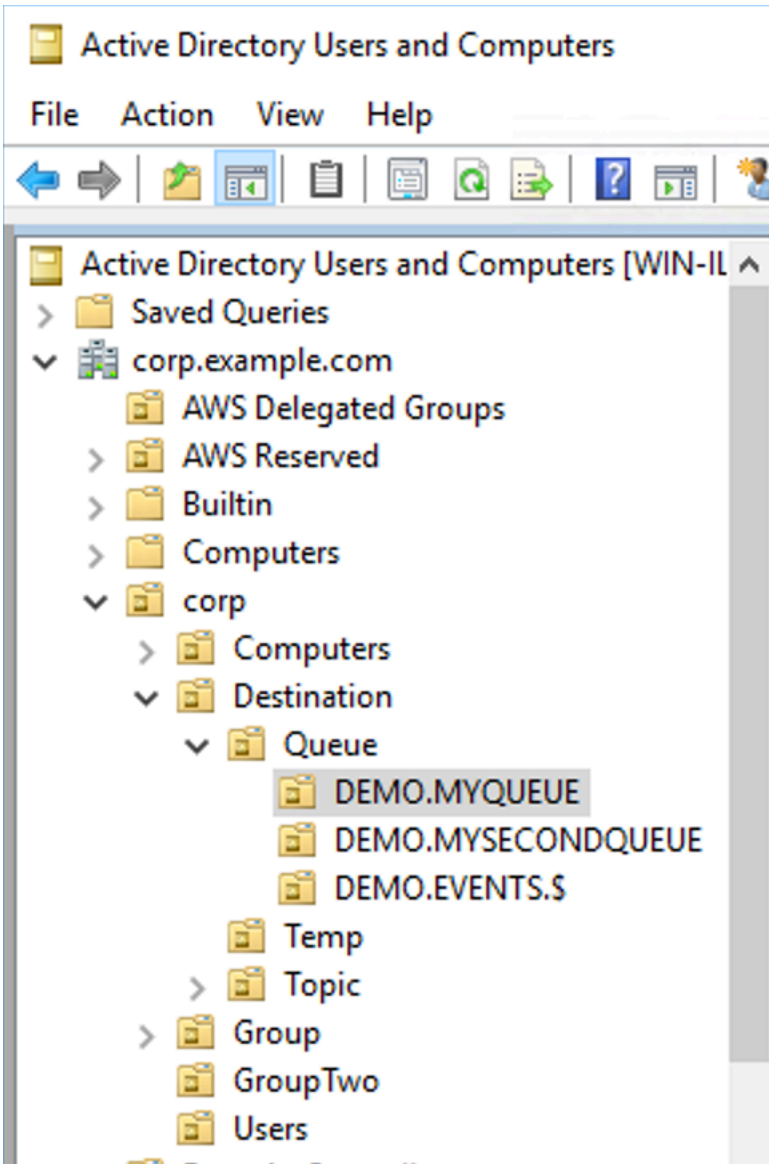
```
OU=DEMO.EVENTS.$,OU=Queue,OU=Destination,OU=corp,DC=corp,DC=example,DC=com
```

> **Note**
>
> DEMO.EVENTS.$ OU는 Queue OU 내에 있습니다.

ActiveMQ의 와일드카드에 대한 자세한 내용은 와일드카드를 참조하세요.

DEMO.MYQUE와 같은 특정 대기열의 권한 부여 규칙을 제공하려면 다음과 같이 지정합니다.

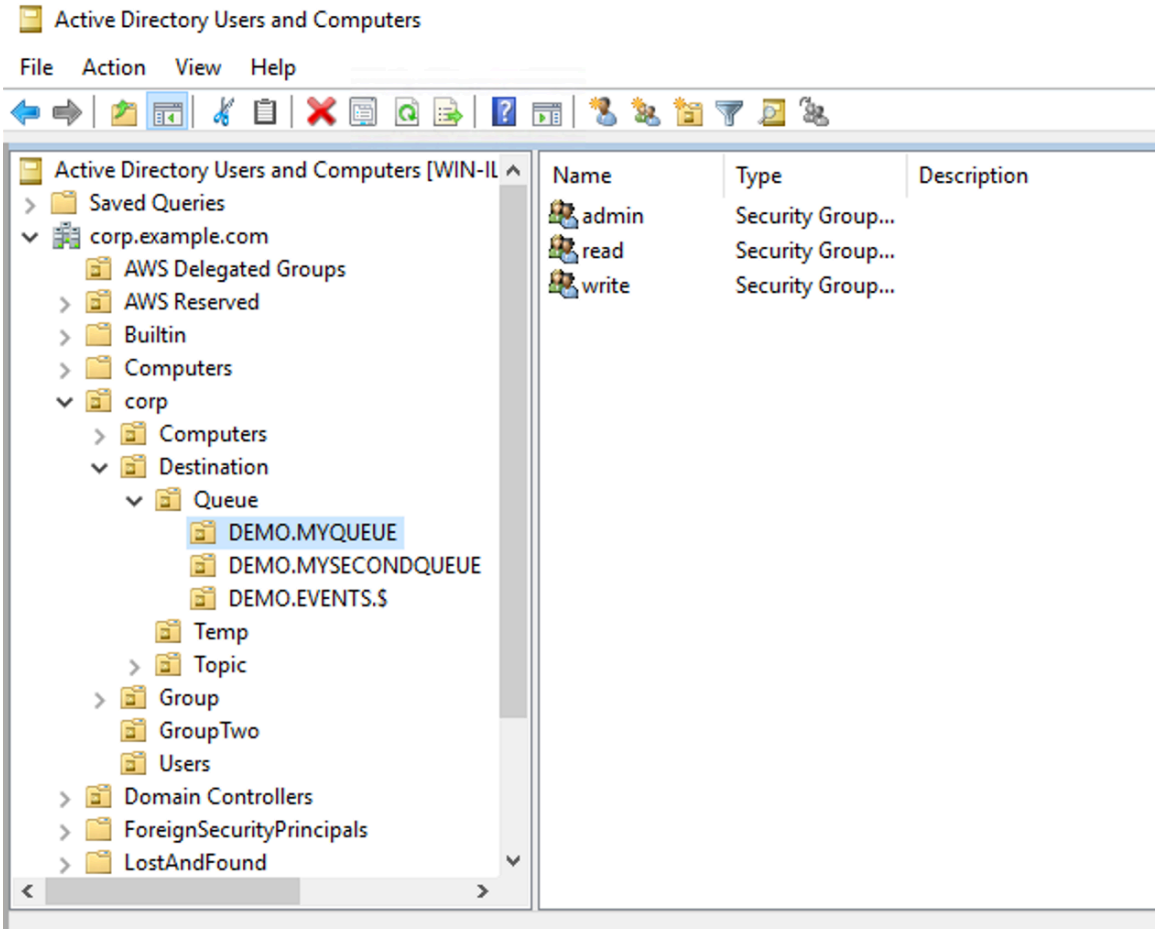
```
OU=DEMO.MYQUEUE,OU=Queue,OU=Destination,OU=corp,DC=corp,DC=example,DC=com
```



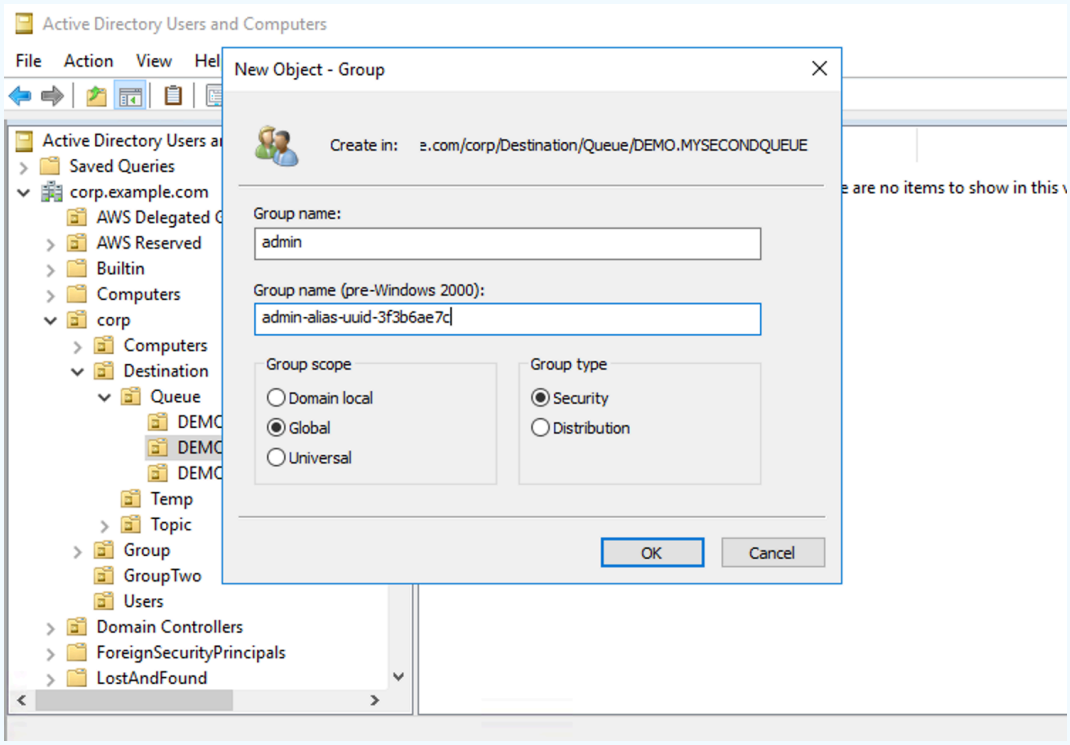
보안 그룹

대상 또는 와일드카드를 나타내는 각 OU 내에서 세 개의 보안 그룹을 생성해야 합니다. ActiveMQ의 모든 권한과 마찬가지로 읽기/쓰기/관리자 권한입니다. 이러한 각 사용 권한에 대한 자세한 내용은 ActiveMQ 설명서에서 보안을 참조하세요.

이러한 보안 그룹 `read` , `write` 및 `admin`의 이름을 지정해야 합니다. 이러한 각 보안 그룹 내에서 사용자 또는 그룹을 추가할 수 있으며, 해당 사용자 또는 그룹은 관련 작업을 수행할 권한을 가집니다. 각 와일드카드 대상 집합 또는 개별 대상에 이러한 보안 그룹이 필요합니다.



> **Note**
>
> 관리자 그룹을 생성하면 그룹 이름에서 충돌이 발생합니다. 이 충돌은 그룹이 DIT의 다른 위치에 있더라도 레거시 Windows 2000 이전 규칙에서 그룹이 같은 이름을 공유하도록 허용하지 않기 때문에 발생합니다. Windows 2000 이전 버전 텍스트 상자의 값은 설정에 영향을 주지 않지만 전역적으로 고유해야 합니다. 이 충돌을 피하기 위해 각 `admin` 그룹에 `uuid` 접미사를 추가할 수 있습니다.



특정 대상의 `admin` 보안 그룹에 사용자를 추가하면 사용자가 해당 주제를 생성 및 삭제할 수 있습니다. `read` 보안 그룹에 추가하면 대상에서 읽을 수 있고 `write` 그룹에 추가하면 대상에 쓸 수 있습니다.

보안 그룹 권한에 개별 사용자를 추가하는 것 외에 전체 그룹을 추가할 수도 있습니다. 그러나 ActiveMQ에서 그룹의 속성 이름을 다시 하드코딩하므로 activemq 소스 코드에 나온 대로 추가할 그룹에 개체 클래스 `groupOfNames`이 있는지 확인해야 합니다.

이렇게 하려면 사용자 `uid`의 경우와 동일한 프로세스를 수행합니다. Windows Server 2016 이상 버전의 Active Directory 사용자 및 컴퓨터에서 ID 매핑 구성을 참조하세요.

3단계: (선택 사항) AWS Lambda 함수에 연결

AWS Lambda 는 Amazon MQ 브로커의 메시지에 연결하고 사용할 수 있습니다. 브로커를 Lambda에 연결할 때는 대기열에서 메시지를 읽고 함수를 동기식으로 호출하는 이벤트 소스 매핑을 생성합니다. 생성하는 이벤트 소스 매핑은 메시지를 브로커에서 배치로 읽고 JSON 객체 형식의 Lambda 페이로드로 변환합니다.

브로커를 Lambda 함수에 연결하려면

1. Lambda 함수 실행 역할에 다음 IAM 역할 권한을 추가합니다.

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

> **Note**
>
> 필요한 IAM 권한이 없으면 함수가 Amazon MQ 리소스의 레코드를 성공적으로 읽을 수 없습니다.

2. (선택 사항) 퍼블릭 액세스 가능성이 없는 브로커를 생성한 경우 다음 중 하나를 수행하여 Lambda가 브로커에 연결하도록 허용해야 합니다.

   - 퍼블릭 서브넷당 하나의 NAT 게이트웨이를 구성합니다. 자세한 내용은 AWS Lambda 개발자 안내서에서 VPC 연결 함수의 인터넷 및 서비스 액세스를 참조하세요.
   - VPC 엔드포인트를 사용하여 Amazon Virtual Private Cloud(Amazon VPC)와 Lambda 간의 연결을 생성합니다. Amazon VPC는 AWS Security Token Service (AWS STS) 및 Secrets Manager 엔드포인트에도 연결해야 합니다. 자세한 내용은 AWS Lambda 개발자 안내서에서 Lambda에 대한 인터페이스 VPC 엔드포인트 구성을 참조하세요.

3. AWS Management Console을 사용하여 Lambda 함수에 대해 브로커를 이벤트 소스로 구성합니다. `create-event-source-mapping` AWS Command Line Interface 명령을 사용할 수도 있습니다.

4. 브로커에서 소비한 메시지를 Lambda 함수가 처리하는 코드를 작성합니다. 이벤트 소스 매핑에서 검색되는 Lambda 페이로드는 브로커의 엔진 유형에 따라 다릅니다. 다음은 ActiveMQ용 Amazon MQ 대기열의 Lambda 페이로드 예제입니다.

> **Note**
>
> 이 예제에서 `testQueue`는 대기열의 이름입니다.

```
{
  "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 MQ를 Lambda에 연결하는 방법, Amazon MQ 이벤트 소스에 대해 Lambda가 지원하는 옵션 및 이벤트 소스 매핑 오류에 대한 자세한 내용은 AWS Lambda 개발자 안내서에서 Amazon MQ에서 Lambda 사용을 참조하세요.

ActiveMQ 브로커 사용자 생성

ActiveMQ 사용자는 ActiveMQ 브로커의 대기열 및 주제에 액세스할 수 있는 사람 또는 애플리케이션입니다. 사용자가 특정 권한을 갖도록 구성할 수 있습니다. 예를 들어 일부 사용자가 ActiveMQ 웹 콘솔에 액세스하도록 허용할 수 있습니다.

그룹을 의미 체계 레이블입니다. 사용자에 그룹을 할당하고 그룹에 대해 특정 대기열 및 주제에 보내고 대기열 및 주제에서 받고 대기열 및 주제를 관리할 권한을 구성할 수 있습니다.

> **Note**
>
> 그룹을 사용자와 독립적으로 구성할 수는 없습니다. 그룹 레이블은 그룹에 하나 이상의 사용자를 추가할 때 생성되고 그룹에서 모든 사용자를 제거하면 삭제됩니다.

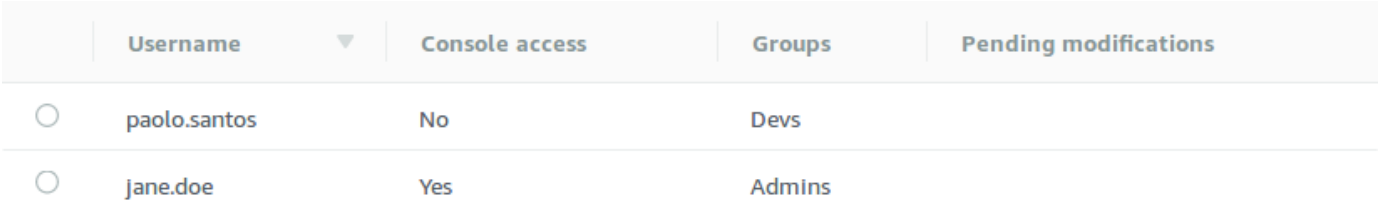
> **Note**
>
> Amazon MQ 기반 ActiveMQ의 `activemq-webconsole` 그룹은 모든 대기열 및 주제에 대한 관리자 권한을 보유하고 있습니다. 이 그룹의 모든 사용자도 관리자 액세스 권한을 갖습니다.

다음 예제에서는 AWS Management Console을 사용하여 Amazon MQ 브로커 사용자를 생성, 편집 및 삭제하는 방법을 보여줍니다.

새 ActiveMQ 브로커 사용자 생성

1. Amazon MQ 콘솔에 로그인합니다.
2. 브로커 목록에서 브로커 이름(예: MyBroker)을 선택한 다음 View details(세부 정보 보기)를 선택합니다.

   MyBroker 페이지의 사용자 섹션에 이 브로커에 속한 모든 사용자가 나열됩니다.



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

3. 사용자 생성을 선택합니다.
4. Create user(사용자 생성) 대화 상자에 Username(사용자 이름) 및 Password(암호)를 입력합니다.
5. (선택 사항) 사용자가 속하게 될 그룹 이름을 쉼표로 구분하여 입력합니다(예: `Devs, Admins`).
6. (선택 사항) 사용자가 ActiveMQ 웹 콘솔에 액세스할 수 있게 하려면 ActiveMQ Web Console(ActiveMQ 웹 콘솔)을 선택합니다.
7. 사용자 생성을 선택합니다.

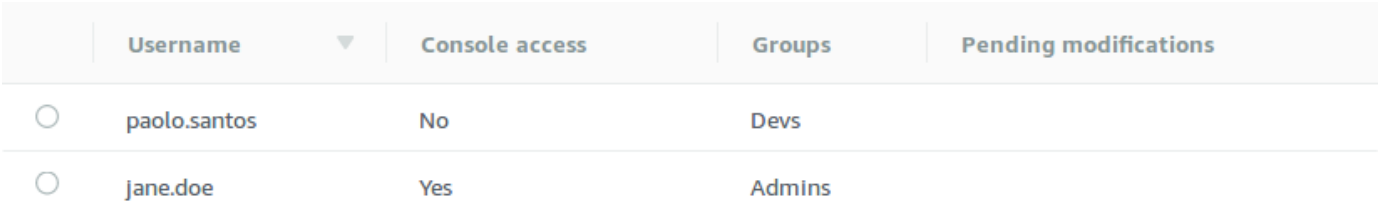
> **Important**
>
> 사용자에 대한 변경 사항이 있어도 즉시 사용자에게 변경 사항이 적용되지는 않습니다. 변경 내용을 적용하려면 다음 유지 관리 기간을 기다리거나 브로커를 재부팅해야 합니다.

ActiveMQ 브로커 사용자 편집

기존 사용자를 편집하려면 다음을 수행합니다.

1. Amazon MQ 콘솔에 로그인합니다.
2. 브로커 목록에서 브로커 이름(예: MyBroker)을 선택한 다음 View details(세부 정보 보기)를 선택합니다.

   MyBroker 페이지의 사용자 섹션에 이 브로커에 속한 모든 사용자가 나열됩니다.



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

3. 로그인 보안 인증 정보를 선택하고 편집을 선택합니다.

   Edit user(사용자 편집) 대화 상자가 표시됩니다.
4. (선택 사항) 새 Password(암호)를 입력합니다.

5. (선택 사항) 사용자가 속하게 될 그룹 이름을 쉼표로 구분하여 추가하거나 제거합니다(예: `Managers, Admins`).
6. (선택 사항) 사용자가 ActiveMQ 웹 콘솔에 액세스할 수 있게 하려면 ActiveMQ Web Console(ActiveMQ 웹 콘솔)을 선택합니다.
7. 사용자에 대한 변경 사항을 저장하려면 Done(완료)을 선택합니다.

> **Important**
>
> 사용자에 대한 변경 사항이 있어도 즉시 사용자에게 변경 사항이 적용되지는 않습니다. 변경 내용을 적용하려면 다음 유지 관리 기간을 기다리거나 브로커를 재부팅해야 합니다.

ActiveMQ 브로커 사용자 삭제

더 이상 필요하지 않은 사용자는 삭제할 수 있습니다.

1. Amazon MQ 콘솔에 로그인합니다.
2. 브로커 목록에서 브로커 이름(예: MyBroker)을 선택한 다음 View details(세부 정보 보기)를 선택합니다.

   MyBroker 페이지의 사용자 섹션에 이 브로커에 속한 모든 사용자가 나열됩니다.

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

3. 로그인 보안 인증 정보(예: ***MyUser***)를 선택한 다음 삭제를 선택합니다.
4. 사용자 삭제를 확인하려면 Delete ***MyUser***(MyUser를 삭제하시겠습니까)? 대화 상자에서 Delete(삭제)를 선택합니다.

> **Important**
>
> 사용자에 대한 변경 사항이 있어도 즉시 사용자에게 변경 사항이 적용되지는 않습니다. 변경 내용을 적용하려면 다음 유지 관리 기간을 기다리거나 브로커를 재부팅해야 합니다.

ActiveMQ를 통한 JMS(Java Message Service) 사용 예제

다음 예제에서는 ActiveMQ를 프로그래밍 방식으로 사용하는 방법을 보여 줍니다.

- OpenWire 예시 Java 코드는 브로커에 연결하고, 대기열을 생성하고, 메시지를 보내고 받습니다. 자세한 설명과 분석은 Connecting a Java application to your broker 단원을 참조하세요.
- MQTT 예제 Java 코드는 브로커에 연결하고, 주제를 생성하고, 메시지를 게시하고 받습니다.
- STOMP+WSS 예제 Java 코드는 브로커에 연결하고, 대기열을 생성하고, 메시지를 게시하고 받습니다.

사전 조건

VPC 속성 활성화

VPC 내에서 브로커에 액세스할 수 있도록 하려면 `enableDnsHostnames` 및 `enableDnsSupport` VPC 속성을 활성화해야 합니다. 자세한 내용은 Amazon VPC 사용 설명서의 VPC에서 DNS 지원을 참조하세요.

인바운드 연결 활성화

Amazon MQ를 프로그래밍 방식으로 사용하려면 인바운드 연결을 사용해야 합니다.

1. Amazon MQ 콘솔에 로그인합니다.
2. 브로커 목록에서 브로커 이름(예: MyBroker)을 선택합니다.
3. ***MyBroker*** 페이지의 Connections(연결) 섹션에서 브로커의 웹 콘솔 URL 및 와이어 레벨 프로토콜의 주소 및 포트를 적어둡니다.
4. 세부 정보 섹션의 보안 및 네트워크에서 보안 그룹의 이름 또는  을 선택합니다.

   EC2 Dashboard의 보안 그룹 페이지가 표시됩니다.
5. 보안 그룹 목록에서 보안 그룹을 선택합니다.
6. 페이지 하단에서 인바운드를 선택한 후 편집을 선택합니다.
7. Edit inbound rules(인바운드 규칙 편집) 대화 상자에서 공개적으로 액세스하고자 하는 모든 URL 또는 엔드포인트에 대한 규칙을 추가합니다. 다음 예제에서는 브로커 웹 콘솔에 대해 이 작업을 수행하는 방법을 보여줍니다.

   a. 규칙 추가(Add Rule)를 선택합니다.

b. 유형에서 Custom TCP(사용자 지정 TCP)를 선택합니다.

c. Port Range(포트 범위)에 웹 콘솔 포트(8162)를 입력합니다.

d. Source(소스)에서 Custom(사용자 지정)을 선택한 상태에서 웹 콘솔에 액세스하는 데 사용할 시스템의 IP 주소(예: `192.0.2.1`)를 입력합니다.

e. 저장을 선택합니다.

이제 브로커가 인바운드 연결을 허용할 수 있습니다.

Java 종속성 추가

OpenWire

`activemq-client.jar` 및 `activemq-pool.jar` 패키지를 Java 클래스 경로에 추가합니다. 다음 예제는 Maven 프로젝트의 `pom.xml` 파일 내에 존재하는 이러한 종속성을 보여줍니다.

```
<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`에 대한 자세한 정보는 Apache ActiveMQ 설명서의 Initial Configuration을 참조하세요.

MQTT

`org.eclipse.paho.client.mqttv3.jar` 패키지를 Java 클래스 경로에 추가합니다. 다음 예제는 Maven 프로젝트 `pom.xml` 파일의 이 종속성을 보여줍니다.

```
<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`에 대한 자세한 내용은 Eclipse Paho Java Client를 참조하세요.

STOMP+WSS

다음 패키지를 Java 클래스 경로에 추가합니다.

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

다음 예제는 Maven 프로젝트의 `pom.xml` 파일 내에 존재하는 이러한 종속성을 보여줍니다.

```
<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>
```

자세한 내용은 Spring Framework 설명서의 STOMP Support를 참조하세요.

AmazonMQExample.java

> **Important**
>
> 다음 예제 코드에서 생성자와 소비자는 단일 스레드에서 실행됩니다. 프로덕션 시스템(또는 브로커 인스턴스 장애 조치 테스트)의 경우 생산자와 소비자가 별도의 호스트 또는 스레드에서 실행되는지 확인합니다.

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);
                        }
                    }
                    }
```

ActiveMQ용 Amazon MQ 엔진 버전 관리

Apache ActiveMQ는 의미 체계 버전 관리 사양에 따라 버전 번호를 `X.Y.Z`로 구성합니다. ActiveMQ용 Amazon MQ 구현에서 `X`는 메이저 버전, `Y`는 마이너 버전, `Z`는 패치 버전 번호를 나타냅니다. Amazon MQ는 메이저 버전 번호가 변경된 경우 메이저 버전 변경으로 간주됩니다. 예를 들어 버전 5.17에서 6.0으로의 업그레이드는 메이저 버전 업그레이드로 간주됩니다. 마이너 버전 또는 패치 버전 번호만 변경된 경우 마이너 버전 변경으로 간주됩니다. 예를 들어 버전 5.18에서 5.19로 업그레이드하는 것은 마이너 버전 업그레이드로 간주됩니다. `autoMinorVersionUpgrade`가 켜지면 Amazon MQ는 브로커를 사용 가능한 최신 패치 버전으로 업그레이드합니다.

ActiveMQ용 Amazon MQ에서는 모든 브로커에 지원되는 최신 마이너 버전을 사용할 것을 권장합니다. 브로커 엔진 버전을 업그레이드하는 방법에 대한 지침은 Amazon MQ 브로커 엔진 버전 업그레이드를 참조하세요.

ActiveMQ용 Amazon MQ에서 지원되는 엔진 버전

Amazon MQ 버전 지원 캘린더에는 브로커 엔진 버전의 지원 종료 시기가 표시되어 있습니다. 버전 지원이 종료되면 Amazon MQ는 해당 버전의 모든 브로커를 다음 지원 버전으로 자동 업그레이드합니다. 이 업그레이드는 지원 종료일 전 45일 이내에 브로커의 예정된 유지보수 기간 동안 진행됩니다.

Amazon MQ는 버전 지원이 종료되기 최소 90일 전에 미리 통지합니다. 중단이 발생하지 않도록 지원 종료일 전에 브로커를 업그레이드하는 것이 좋습니다. 또한 지원 종료일 전 30일 이내에는 지원 종료가 예정된 버전에서 새 브로커를 생성할 수 없습니다.

Apache ActiveMQ 버전	Amazon MQ 지원 종료
ActiveMQ 5.19(권장)	
ActiveMQ 5.18	
ActiveMQ 5.17	2025년 6월 16일
ActiveMQ 5.16	2024년 11월 15일
ActiveMQ 5.15	2024년 9월 16일

새 ActiveMQ용 Amazon MQ 브로커를 생성할 때 지원되는 모든 ActiveMQ 엔진 버전을 지정할 수 있습니다. 브로커를 생성할 때 엔진 버전 번호를 지정하지 않으면 Amazon MQ는 자동으로 최신 엔진 버전 번호로 기본 설정됩니다.

엔진 버전 업그레이드

언제든지 브로커를 지원되는 다음 메이저 또는 마이너 버전으로 수동으로 업그레이드할 수 있습니다. 자동 마이너 버전 업그레이드를 활성화하면 유지 관리 기간 동안 Amazon MQ에서 브로커를 지원되는 최신 패치 버전으로 업그레이드합니다.

브로커 수동 업그레이드에 대한 자세한 내용은 the section called “엔진 버전 업그레이드” 단원을 참조하세요.

지원되는 엔진 버전 목록 표시

`describe-broker-instance-options` AWS CLI 명령을 사용하여 지원되는 모든 마이너 및 메이저 엔진 버전을 나열할 수 있습니다.

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

엔진 및 인스턴스 유형별로 결과를 필터링하려면 다음에 나온 `--engine-type` 및 `--host-instance-type` 옵션을 사용합니다.

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

예를 들어 결과를 ActiveMQ 및 `mq.m5.large` 인스턴스 유형에 대해 필터링하려면 *`engine-type`*을 `ACTIVEMQ`로, *`instance-type`*을 `mq.m5.large`로 대체합니다.

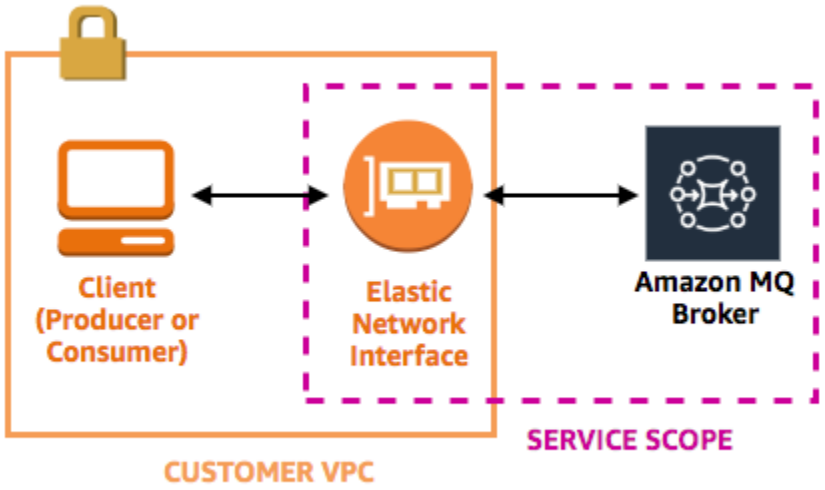
ActiveMQ용 Amazon MQ 모범 사례

이 단원을 참조하여 Amazon MQ에서 ActiveMQ를 이용할 때 성능을 극대화하고 처리량 비용을 최소화할 수 있는 권장 방법을 신속히 찾으세요.

Amazon MQ 탄력적 네트워크 인터페이스를 수정하거나 삭제하지 않음

처음으로 Amazon MQ 브로커 생성할 때 Amazon MQ는 Virtual Private Cloud(VPC)에서 사용자 계정 아래에 탄력적 네트워크 인터페이스를 프로비저닝하므로 많은 EC2 권한이 필요합니다. 네트워크 인터페이스를 통해 클라이언트(생산자 또는 소비자)가 Amazon MQ 브로커와 통신할 수 있습니다. 네트워크 인터페이스는 사용자 계정의 VPC에 속해 있음에도 불구하고 Amazon MQ의 서비스 범위 내에 있는 것으로 간주됩니다.

> **Warning**
>
> 이 네트워크 인터페이스는 수정하거나 삭제하면 안 됩니다. 네트워크 인터페이스를 수정하거나 삭제하면 VPC와 브로커 간의 연결이 영구적으로 손실될 수 있습니다.



항상 연결 풀 사용

단일 생산자와 단일 소비자가 있는 시나리오(예: 시작하기: ActiveMQ 브로커 생성 및 연결 자습서)에서는 모든 생산자와 소비자에 대해 단일 `ActiveMQConnectionFactory` 클래스를 사용할 수 있습니다. 예를 들면 다음과 같습니다.

```
// 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();
```

하지만 생산자와 소비자가가 여러 개인 현실적인 시나리오에서는 여러 생산자에 대해 많은 수의 연결을 생성하려면 비용이 많이 들고 비효율적일 수 있습니다. 이러한 시나리오에서는 `PooledConnectionFactory` 클래스를 사용하여 여러 개의 생산자 요청을 그룹화해야 합니다. 예를 들면 다음과 같습니다.

> **Note**
>
> 메시지 소비자는 PooledConnectionFactory 클래스를 사용하면 안 됩니다.

```
// 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();
```

항상 Failover Transport를 사용하여 여러 브로커 엔드포인트에 연결

애플리케이션이 여러 브로커 엔드포인트에 연결해야 하는 경우(예: 활성/대기 배포 모드를 사용하는 경우나 온프레미스 메시지 브로커에서 Amazon MQ로 마이그레이션하는 경우 장애 조치 전송을 사용하여 소비자가 둘 중 하나에 임의로 연결하도록 허용할 수 있습니다. 예제:

```
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**
>
> 다중 가용 영역 브로커는 유지 관리 기간 및 브로커 재시작 중에 장애 조치를 경험할 수 있습니다. 장애 조치 전송을 사용하여 브로커 가용성을 보장합니다.

메시지 선택기 사용하지 않기

JMS 선택기를 사용하여 필터를 주제 구독에 연결할 수 있습니다(그러면 콘텐츠를 기반으로 메시지를 소비자에게 라우팅할 수 있음). 하지만 JMS 선택기를 사용하면 Amazon MQ 브로커의 필터 버퍼가 가득 차서 메시지를 필터링할 수 없습니다.

일반적으로 소비자가 메시지를 라우팅하도록 두지 마세요. 소비자와 생산자를 최적으로 결합 해제하려면 소비자와 생산자가 모두 한시적이어야 하기 때문입니다.

지속 구독보다 가상 대상을 선호

지속 구독은 주제에 게시된 모든 메시지를 소비자가 수신하도록 보장할 수 있습니다. 예를 들어 끊어진 연결이 복원된 후 메시지를 모두 수신할 수 있습니다. 하지만 지속 구독을 사용하면 상충되는 소비자의 사용을 제외하고 대규모의 성능 문제가 발생할 수 있습니다. 그 대신 가상 대상 사용을 고려하세요.

Amazon VPC 피어링을 사용하는 경우 CIDR 범위 **10.0.0.0/16**의 클라이언트 IP 사용 안 함

온프레미스 인프라와 Amazon MQ 브로커 간에 Amazon VPC 피어링을 설정하는 경우 CIDR 범위 `10.0.0.0/16`의 IP와 클라이언트 연결을 구성해서는 안 됩니다.

느린 소비자를 통해 대기열 동시 저장 및 디스패치 비활성화

기본적으로 Amazon MQ는 빠른 소비자가 있는 대기열에 맞게 최적화됩니다.

- 소비자가 생산자가 생성한 메시지의 속도를 따라갈 수 있는 경우 소비자는 빠른 것으로 간주됩니다.
- 대기열이 승인되지 않은 메시지의 백로그를 작성하여 생산자 처리량의 감소를 유발할 수 있는 경우 소비자는 느린 것으로 간주됩니다.

느린 소비자가 있는 대기열에 맞게 최적화하도록 Amazon MQ에 지시하려면 `concurrentStoreAndDispatchQueues` 속성을 `false`로 설정합니다. 구성 예제는 를 참조하세요`concurrentStoreAndDispatchQueues`

처리량을 최대화하기 위해 올바른 브로커 인스턴스 유형 선택

브로커 인스턴스 유형의 메시지 처리량은 애플리케이션의 사용 사례와 다음 요인에 따라 다릅니다.

- 지속 모드에서 ActiveMQ 사용

- 메시지 크기
- 생산자 수와 소비자 수
- 대상 수

메시지 크기, 대기 시간, 처리량 간의 관계 이해

용도에 따라 더 큰 브로커 인스턴스 유형이 반드시 시스템 처리량을 높여 주는 것이 아닐 수도 있습니다. ActiveMQ가 지속 가능한 스토리지에 메시지를 기록할 때 메시지 크기에 따라 시스템의 제한 요인이 결정됩니다.

- 메시지 크기가 100KB보다 작으면 영구 스토리지 지연 시간이 제한 요인입니다.
- 메시지 크기가 100KB보다 크면 영구 스토리지 처리량이 제한 요인입니다.

ActiveMQ가 지속 모드일 경우 소비자 수가 거의 없거나 소비자 속도가 느릴 때 스토리지에 대한 쓰기가 정상적으로 수행됩니다. 비지속 모드에서는 브로커 인스턴스의 힙 메모리가 가득 찰 경우 소비자 속도가 느려도 스토리지에 대한 쓰기가 수행됩니다.

애플리케이션에 가장 적합한 브로커 인스턴스를 확인하려면 다양한 브로커 인스턴스 유형을 검사하는 것이 좋습니다. 자세한 내용은 Broker instance types 단원 및 JMS Benchmark를 사용하여 Amazon MQ의 처리량 측정을 참조하세요.

더 큰 브로커 인스턴스 유형의 사용 사례

더 큰 브로커 인스턴스 유형이 처리량을 향상시킬 수 있는 세 가지 일반 사용 사례가 있습니다.

- 비지속 모드 - 애플리케이션이 브로커 인스턴스 장애 조치 중 발생하는 메시지 손실에 비교적 민감하지 않은 경우(예: 스포츠 경기 득점 결과를 방송하는 경우) 일반적으로 ActiveMQ의 비지속 모드를 사용할 수 있습니다. 이 모드에서 ActiveMQ는 브로커 인스턴스의 힙 메모리가 가득 찰 경우에만 영구 스토리지에 메시지를 기록합니다. 비지속 모드를 사용하는 시스템은 더 큰 브로커 인스턴스 유형에서 제공하는 메모리 양, CPU 속도 및 빠른 네트워크 속도를 활용할 수 있습니다.
- 빠른 소비자 - 활성 소비자가 사용 가능하고 `concurrentStoreAndDispatchQueues` 플래그가 활성화되면 ActiveMQ는 메시지를 스토리지로 전송하지 않고 메시지가 생산자에서 소비자로 직접 흐르도록 허용합니다(지속 모드에서도 허용). 애플리케이션이 메시지를 빨리 소비할 수 있으면(소비자가 이렇게 실행하도록 설계할 수 있으면) 애플리케이션은 더 큰 브로커 인스턴스 유형의 이점을 활용할 수 있습니다. 애플리케이션에서 메시지를 더 빠르게 소비할 수 있게 하려면 애플리케이션 인스턴스에 소비자 스레드를 추가하거나, 애플리케이션 인스턴스를 수직 또는 수평으로 확장합니다.

- 배치 트랜잭션 - 지속 모드를 사용하며 트랜잭션당 여러 개의 메시지를 전송하는 경우 더 큰 브로커 인스턴스 유형을 사용하면 전체적으로 더 많은 메시지 처리량을 달성할 수 있습니다. 자세한 내용은 ActiveMQ 설명서의 트랜잭션을 사용해야 합니까?를 참조하세요.

처리량을 최대화하기 위해 올바른 브로커 스토리지 유형 선택

다중 가용 영역에 걸친 높은 내구성 및 복제를 활용하려면 Amazon EFS를 사용하세요. 짧은 대기 시간 및 높은 처리량을 활용하려면 Amazon EBS를 사용하세요. 자세한 정보는 Storage을 참조하세요.

브로커 네트워크를 올바로 구성

네트워크 브로커를 생성할 때 애플리케이션에 올바르게 구성합니다.

- 지속 모드를 활성화 - (피어에 비해) 각 브로커 인스턴스는 생산자 또는 소비자처럼 행동하기 때문에 브로커 네트워크는 분산된 메시지 복제를 제공하지 않습니다. 소비자로 행동하는 첫 번째 브로커가 메시지를 수신하고 스토리지에 지속시킵니다. 이 브로커는 생산자에게 승인을 전송하고 메시지를 다음 브로커에 전달합니다. 두 번째 브로커가 메시지의 지속성을 승인하고 첫 번째 브로커가 메시지를 삭제합니다.

  지속 모드가 비활성화된 경우, 첫 번째 브로커가 메시지를 스토리지에 지속시키지 않고 생산자를 승인합니다. 자세한 내용은 Apache ActiveMQ 설명서에서 복제된 메시지 스토어 및 지속적 전송과 비지속적 전송의 차이점은 무엇입니까?를 참조하세요.
- 브로커 인스턴스에 대해 자문 메시지를 비활성화하지 않음 - 자세한 내용은 Apache ActiveMQ 설명서에서 자문 메시지를 참조하세요.
- 멀티캐스트 브로커 검색을 사용하지 않음 - Amazon MQ는 멀티캐스트를 사용한 브로커 검색을 지원하지 않습니다. 자세한 내용은 Apache ActiveMQ 설명서에서 검색, 멀티캐스트, zeroconf의 차이점은 무엇입니까?를 참조하세요.

준비된 XA 트랜잭션을 복구하여 느린 재시작 방지

ActiveMQ는 분산(XA) 트랜잭션을 지원합니다. ActiveMQ에서 XA 트랜잭션을 처리하는 방식을 알면 Amazon MQ에서의 브로커 재시작 및 장애 조치에 대한 복구 시간이 느려지는 것을 방지할 수 있습니다.

해결되지 않은 준비된 XA 트랜잭션은 재시작할 때마다 재실행됩니다. 이러한 트랜잭션이 해결되지 않은 상태로 남아 있으면 시간이 지남에 따라 이들 수가 증가하여 브로커 시작에 필요한 시간이 크게 늘

어납니다. 이는 재시작 및 장애 조치 시간에 영향을 미칩니다. 이러한 트랜잭션은 시간이 지남에 따라 성능이 저하되지 않도록 `commit()` 또는 `rollback()`으로 해결해야 합니다.

해결되지 않은 준비된 XA 트랜잭션을 모니터링하기 위해 Amazon CloudWatch Logs의 `JournalFilesForFastRecovery` 지표를 사용할 수 있습니다. 이 숫자가 증가하고 있거나 1보다 일관되게 높은 경우, 다음 예제와 유사한 코드를 사용하여 해결되지 않은 트랜잭션을 복구해야 합니다. 자세한 정보는 Amazon MQ의 할당량을 참조하세요.

다음 예제 코드는 준비된 XA 트랜잭션을 모두 거치고 `rollback()`을 사용하여 닫습니다.

```
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) {
```

```
        }
    }
}
```

실제 시나리오에서는 XA Transaction Manager와 비교하여 준비된 XA 트랜잭션을 확인할 수 있습니다. 그런 다음 각각의 준비된 트랜잭션을 `rollback()` 또는 `commit()` 중에서 어떤 것으로 처리할지 결정할 수 있습니다.

RabbitMQ용 Amazon MQ 사용

Amazon MQ를 사용하면 필요에 맞는 컴퓨팅 및 스토리지 리소스를 사용하여 메시지 브로커를 쉽게 생성할 수 있습니다. , AWS Management Console Amazon MQ REST API 또는를 사용하여 브로커를 생성, 관리 및 삭제할 수 있습니다 AWS Command Line Interface.

이 단원에서는 ActiveMQ 및 RabbitMQ 엔진 유형용 메시지 브로커의 기본 요소를 설명하고, 사용할 수 있는 Amazon MQ 브로커 인스턴스 유형 및 해당 상태를 나열하며, 브로커 아키텍처 및 구성 옵션에 대한 개요를 제공합니다.

Amazon MQ REST API에 대한 자세한 내용은 Amazon MQ REST API 참조를 참조하세요.

RabbitMQ용 Amazon MQ 브로커란?

브로커는 Amazon MQ에서 실행하는 메시지 브로커 환경입니다. 이 인스턴스는 Amazon MQ의 기본 빌딩 블록입니다. 브로커 인스턴스 클래스(`m7g`) 및 크기(`large`, `medium`)의 설명 조합은 브로커 인스턴스 유형(예: `mq.m7g.large`)입니다.

- 단일 인스턴스 브로커는 Network Load Balancer(NLB) 뒤의 가용 영역 하나에 있는 브로커 하나로 구성됩니다. 브로커는 애플리케이션 및 Amazon EBS 스토리지 볼륨과 통신합니다.
- 클러스터 배포는 네트워크 로드 밸런서 뒤에 있는 3개의 RabbitMQ 브로커 노드(각각 사용자, 대기열 및 여러 가용 영역(AZ) 간에 분산된 상태 공유)로 이루어진 논리적 그룹입니다.

자세한 내용은 RabbitMQ 브로커 배포를 참조하세요.

리스너 포트

Amazon MQ 관리형 RabbitMQ 브로커는를 통한 애플리케이션 수준 연결을 위해 다음과 같은 리스너 포트를 지원합니다`amqps`. RabbitMQ 웹 콘솔 및 관리 API를 사용하여 클라이언트 연결에 이러한 포트를 사용할 수도 있습니다. 모든 연결은 보안을 위해 TLS 암호화를 사용합니다.

- 리스너 포트 5671 - 보안 AMQP URL을 통해 이루어진 보안 AMQP 연결에 사용됩니다. 이 포트는 RabbitMQ 4에서 AMQP 0-9-1 및 AMQP 1.0 프로토콜을 모두 지원합니다. 예를 들어, 브로커 ID가 `b-c8352341-ec91-4a78-ad9c-a43f23d325bb`이고 `us-west-2` 리전에 배포된 브로커의 경우 브로커의 전체 `amqps` URL은 `b-c8352341-ec91-4a78-ad9c-a43f23d325bb.mq.us-west-2.amazonaws.com:5671`입니다.

- 리스너 포트 443 및 15671 - 두 리스너 포트를 서로 바꿔서 사용하여 RabbitMQ 웹 콘솔 또는 관리 API를 통해 브로커에 액세스할 수 있습니다. 포트 443은 표준 HTTPS 액세스를 제공하는 반면, 포트 15671은 TLS 암호화를 사용하는 기존 RabbitMQ 관리 포트입니다.

속성

RabbitMQ 브로커에는 다음과 같은 여러 속성이 있습니다.

- 이름. 예를 들어 `MyBroker`입니다.
- ID. 예를 들어 `b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9`입니다.
- Amazon 리소스 이름(ARN). 예를 들어 `arn:aws:mq:us-east-2:123456789012:broker:MyBroker:b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9`입니다.
- RabbitMQ 웹 콘솔 URL. 예를 들어 `https://b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-east-2.amazonaws.com`입니다.

  자세한 내용은 RabbitMQ 설명서의 RabbitMQ 웹 콘솔을 참조하세요.
- 보안 AMQP 엔드포인트. 예를 들어 `amqps://b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-east-2.amazonaws.com`입니다.

브로커 속성의 전체 목록은 Amazon MQ REST API 참조의 다음 단원을 참조하세요.

- REST 작업 ID: 브로커
- REST 작업 ID: 브로커
- REST 작업 ID: 브로커 재부팅

RabbitMQ용 Amazon MQ 엔진 버전 관리

RabbitMQ는 의미 체계 버전 관리 사양에 따라 버전 번호를 `X.Y.Z`로 구성합니다. RabbitMQ용 Amazon MQ 구현에서 `X`는 메이저 버전, `Y`는 마이너 버전, `Z`는 패치 버전 번호를 나타냅니다. Amazon MQ는 메이저 버전 번호가 변경된 경우 메이저 버전 변경으로 간주됩니다. 예를 들어 버전 3.13에서 4.0으로 업그레이드하는 것은 메이저 버전 업그레이드로 간주됩니다. 마이너 버전 또는 패치 버전 번호만 변경된 경우 마이너 버전 변경으로 간주됩니다. 예를 들어 버전 3.11.28에서 3.12.13으로 업그레이드하는 것은 마이너 버전 업그레이드로 간주됩니다.

RabbitMQ용 Amazon MQ는 모든 브로커가 지원되는 최신 버전의 RabbitMQ 4.2를 사용할 것을 권장합니다. 브로커 엔진 버전을 업그레이드하는 방법에 대한 지침은 Amazon MQ 브로커 엔진 버전 업그레이드를 참조하세요.

RabbitMQ용 Amazon MQ 브로커를 새로 생성할 때는 메이저 및 마이너 버전 번호만 지정하면 됩니다. RabbitMQ 4.2를 예로 들 수 있습니다. 브로커를 생성할 때 엔진 버전을 지정하지 않으면 Amazon MQ는 자동으로 최신 엔진 버전으로 기본 설정됩니다.

> **Important**
>
> Amazon MQ에서는 스트림이 지원되지 않습니다. 스트림을 생성하면 데이터가 손실됩니다. Amazon MQ는 JSON에서 구조화된 로깅 사용을 지원하지 않습니다.

Amazon MQ는 RabbitMQ의 두 가지 메이저 버전 릴리스를 지원합니다.

- RabbitMQ 4

  Amazon MQ는 RabbitMQ 모든 인스턴스 크기에서 mq.m7g 인스턴스 유형에 대해서만 RabbitMQ 4 릴리스 시리즈에서 RabbitMQ 4.2를 지원합니다.

- RabbitMQ 3

  Amazon MQ는 지원되는 모든 인스턴스 크기에서 RabbitMQ mq.t3, mq.m5 및 mq.m7g 인스턴스 유형의 RabbitMQ 3 릴리스 시리즈에서 RabbitMQ 3.13을 지원합니다.

지원되는 엔진 버전 목록 표시

`describe-broker-instance-options` AWS CLI 명령을 사용하여 지원되는 모든 마이너 및 메이저 엔진 버전을 나열할 수 있습니다.

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

엔진 및 인스턴스 유형별로 결과를 필터링하려면 다음에 나온 `--engine-type` 및 `--host-instance-type` 옵션을 사용합니다.

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

예를 들어 결과를 RabbitMQ 및 `mq.m7g.large` 인스턴스 유형에 대해 필터링하려면 *`engine-type`*을 `RABBITMQ`로, *`instance-type`*을 `mq.m7g.large`로 대체합니다.

RabbitMQ 4

Amazon MQ는 RabbitMQ 모든 인스턴스 크기에서 mq.m7g 인스턴스 유형에 대해서만 RabbitMQ 4 릴리스 시리즈에서 RabbitMQ 4.2를 지원합니다.

Amazon MQ는 RabbitMQ 3.13에서 RabbitMQ 4.2로의 현재 위치 업그레이드를 지원합니다. 자세한 내용은 RabbitMQ 3에서 4로 업그레이드를 참조하세요.

> **Important**
>
> RabbitMQ 4.2용 Amazon MQ 브로커의 기본 대기열 유형은 “쿼럼”입니다. RabbitMQ 대기열 생성 중에 대기열 유형 인수를 지정하지 않으면 쿼럼 대기열이 생성됩니다.
> RabbitMQ 4의 쿼럼 대기열을 내구성 요구 사항에 사용하는 것이 좋습니다. 클래식 대기열은 어떤 경우에도 내구성을 보장할 수 없기 때문입니다.

Amazon MQ의 RabbitMQ 4에 다음과 같은 변경 사항이 도입되었습니다.

- AMQP 1.0을 코어 프로토콜로 사용: 자세한 내용은 프로토콜을 참조하세요.
- 로컬 셔블: 셔블은 이제 AMQP 0-9-1 및 AMQP 1.0 외에도 "local"이라는 새 프로토콜을 지원합니다. 로컬 셔블은 내부적으로 AMQP 1.0을 기반으로 하지만 별도의 TCP 연결을 사용하는 대신 클러스터 노드와 내부 APIs 간의 클러스터 내 연결을 사용하여 메시지를 게시하고 소비합니다. 이는 동일한 클러스터 내에서 소비하고 게시하는 데에만 사용할 수 있으며 AMQP 0-9-1 및 AMQP 1.0보다 적은 리소스를 사용하면서 더 높은 처리량을 제공할 수 있습니다.
- 쿼럼 대기열은 메시지 우선 순위를 지원합니다. 쿼럼 대기열 메시지 우선 순위는 항상 활성 상태이며 작동하는 데 정책이 필요하지 않습니다. 쿼럼 대기열이 우선 순위가 설정된 메시지를 수신하면 우선 순위가 활성화됩니다. 쿼럼 대기열은 내부적으로 높음과 정상이라는 두 가지 우선 순위만 지원합니다. 우선순위가 설정되지 않은 메시지는 우선순위 0~4와 마찬가지로 정상으로 매핑됩니다. 우선 순위가 4보다 높은 메시지는 높음으로 매핑됩니다. 우선순위가 높은 메시지는 일반 우선순위 메시지보다 2:1의 비율로 우선 적용됩니다. 즉, 우선순위가 높은 메시지 2개마다 대기열은 일반 우선순위 메시지 1개(사용 가능한 경우)를 전달합니다. 따라서 쿼럼 대기열은 일종의 엄격한 "공평 공유" 우선 순위 처리를 구현합니다. 이렇게 하면 항상 일반적인 우선 순위 메시지에서 진행이 이루어지지만 높은 우선 순위는 2:1의 비율로 선호됩니다.
- Khepri: Khepri는 RabbitMQ 4 브로커의 기본 메타데이터 스토어로 사용됩니다.

- 상호 TLS(mTLS): Amazon MQ는 RabbitMQ 브로커에 상호 TLS(mTLS)를 지원하므로 클라이언트가 인증서를 사용하여 인증할 수 있습니다. 자세한 내용은 mTLS 구성을 참조하세요.
- SSL 인증서 인증 플러그인: SSL 인증 플러그인은 mTLS 연결의 클라이언트 인증서를 사용하여 사용자를 인증하므로 사용자 이름 및 암호 자격 증명 대신 X.509 클라이언트 인증서를 사용한 인증을 허용합니다. 자세한 내용은 SSL 인증서 인증을 참조하세요.
- HTTP 인증 플러그인: HTTP 인증 백엔드 플러그인을 사용하면 외부 HTTP 서비스에 인증 및 권한 부여를 위임할 수 있습니다. 자세한 내용은 HTTP 인증 및 권한 부여를 참조하세요.
- JMS 지원: 브로커는 이제 JMS 주제 교환 플러그인이 활성화된 JMS 워크로드를 지원하므로 JMS 애플리케이션이 RabbitMQ JMS 클라이언트를 사용하여 연결할 수 있습니다.

다음 기능은 Amazon MQ의 RabbitMQ 4에서 더 이상 사용되지 않습니다.

- 클래식 대기열 미러링: 클라이언트 라이브러리 및 애플리케이션에 대한 중단 변경 없이 클래식 대기열이 계속 지원되지만 이제는 복제되지 않은 대기열 유형입니다. 클라이언트는 복제되지 않은 클래식 대기열에 게시하고 사용할 노드에 연결할 수 있습니다. 복제 및 데이터 안전을 위해 쿼럼 대기열을 사용하는 것이 좋습니다.
- 글로벌 QoS 제거: 고객은 전체 채널에 단일 공유 프리페치를 사용하는 글로벌 QoS 대신 소비자당 QoS(비글로벌)를 설정하는 것이 좋습니다.
- 임시 비독점 대기열 지원: 임시 대기열은 선언된 노드의 가동 시간에 수명이 연결된 대기열입니다. 단일 인스턴스 브로커에서는 노드가 다시 시작될 때 제거됩니다. 클러스터 배포에서는 호스팅되는 노드가 다시 시작될 때 제거됩니다. 일정 시간 동안 사용하지 않은 유휴 대기열을 자동 삭제하려면 대기열 TTL을 사용하는 것이 좋습니다. 전용 대기열은 계속 지원되며 대기열에 대한 모든 연결이 제거되면 삭제됩니다.

Amazon MQ에서 RabbitMQ 4.2로 업그레이드할 때 다음과 같은 주요 변경 사항이 애플리케이션에 영향을 미칠 수 있습니다.

- 기본 대기열 유형: RabbitMQ 4 브로커의 기본 대기열 유형은 쿼럼으로 설정됩니다. 대기열 생성 중에 대기열 유형 인수를 지정하지 않으면 쿼럼 대기열이 생성됩니다.
- 쿼럼 대기열의 기본 재전송 제한은 20으로 설정됩니다. 20회 이상 재전송되는 메시지는 배달 못한 편지 또는 삭제(제거)됩니다. 메시지당 20개의 전송이 대기열의 일반적인 시나리오인 경우 데이터 손실을 방지하려면 해당 대기열에 대해 배달 못한 편지 대상 또는 더 높은 제한을 구성해야 합니다. 이를 위한 권장 방법은 정책을 통하는 것입니다.
- amqplib: 0.10.7 이전 노드 JS 클라이언트 amqplib 버전 또는 frame_max < 8192를 사용하는 AMQP 클라이언트 라이브러리는 RabbitMQ에 연결할 수 없습니다.

- 기본 리소스 제한: RabbitMQ용 Amazon MQ에는 연결, 채널, 채널당 소비자, 대기열, vhost, 셔블, 교환 및 최대 메시지 크기에 대한 기본 리소스 사용 제한이 도입되었습니다. RabbitMQ 이는 브로커 가용성을 보호하기 위한 가드레일 역할을 하며 특정 요구 사항에 맞게 구성을 사용하여 사용자 지정할 수 있습니다.

다음 기능은 Amazon MQ의 RabbitMQ 4에서 지원되지 않습니다.

- 로컬 임의 교환: Amazon MQ 노드가 네트워크 로드 밸런서 뒤에 있으므로 Amazon MQ에서는 로컬 임의 교환이 지원되지 않습니다. Amazon MQ
- 메시지 인터셉터: RabbitMQ 메시지 인터셉터는 Amazon MQ에서 지원되지 않습니다.
- 대기열당 지표: Amazon MQ는 AWS CloudWatch를 통해 RabbitMQ 4 브로커에 대한 RabbitMQ 대기열 지표를 벤딩하지 않습니다. Amazon MQ는 AWS CloudWatch를 통해 브로커 수준 지표를 제공합니다. RabbitMQ 관리 API를 사용하여 대기열 지표를 쿼리할 수 있습니다. 1분 이상의 간격으로 특정 대기열에 대한 지표를 쿼리하는 것이 좋습니다.

RabbitMQ 버전 지원

아래 Amazon MQ 버전 지원 일정은 브로커 엔진 버전이 지원 종료에 도달하는 시기를 나타냅니다. 버전 지원이 종료되면 Amazon MQ는 해당 버전의 모든 브로커를 다음 지원 버전으로 자동 업그레이드합니다. 이 업그레이드는 브로커의 예약된 유지 관리 기간 동안 end-of-support 후 45일 이내에 수행됩니다.

Amazon MQ는 버전 지원이 종료되기 최소 90일 전에 미리 통지합니다. 중단이 발생하지 않도록 지원 종료일 전에 브로커를 업그레이드하는 것이 좋습니다. 또한 지원 종료일 전 30일 이내에는 지원 종료가 예정된 버전에서 새 브로커를 생성할 수 없습니다.

RabbitMQ 버전	Amazon MQ 지원 종료
4.2(권장)	
3.13	
3.12	2025년 3월 17일

버전 업그레이드

언제든지 브로커를 지원되는 다음 메이저 또는 마이너 버전으로 수동으로 업그레이드할 수 있습니다. 브로커를 수동으로 업그레이드하는 방법에 대한 자세한 내용은 Amazon MQ 브로커 엔진 버전 업그레이드를 참조하세요.

Amazon MQ는 버전 3.13 이상을 사용하여 모든 RabbitMQ 브로커에 대해 지원되는 최신 패치 버전으로의 업그레이드를 관리합니다. 수동 및 자동 버전 업그레이드는 모두 예약된 유지 관리 기간 중이나 브로커 재부팅 후에 발생합니다.

> **Important**
>
> RabbitMQ는 증분 버전 업데이트(예: 3.9.x에서 3.10.x로)만 허용합니다. 업데이트할 때 마이너 버전은 건너뛸 수 없습니다(예: 3.8.x에서 3.11.x).

단일 인스턴스 브로커는 재부팅되는 동안 오프라인 상태가 됩니다. 클러스터 브로커의 경우 재부팅 중에 미러링 대기열을 동기화해야 합니다. 대기열이 길어지면 대기열 동기화 프로세스가 더 오래 걸릴 수 있습니다. 대기열 동기화 프로세스 중에는 소비자와 생산자가 대기열을 사용할 수 없습니다. 대기열 동기화 프로세스가 완료되면 브로커가 다시 사용 가능해집니다. 영향을 최소화하려면 트래픽이 적은 시간에 업그레이드하는 것이 좋습니다. 버전 업그레이드 모범 사례에 대한 자세한 내용은 RabbitMQ용 Amazon MQ 모범 사례 단원을 참조하세요.

RabbitMQ 3용 Amazon MQ 브로커를 4로 업그레이드 RabbitMQ

Amazon MQ는 RabbitMQ 3.13에서 RabbitMQ 4.2로의 현재 위치 업그레이드를 지원합니다. 현재 위치 업그레이드는 애플리케이션 코드를 변경할 필요가 없습니다. 업그레이드 중에 Amazon MQ는 브로커에 대한 모든 연결을 차단합니다.

Amazon MQ는 관리형 블루-그린 배포 옵션을 제공하지 않습니다. 블루-그린 배포를 독립적으로 수행하도록 선택한 경우 블루-그린 배포를 참조하세요.

> **Important**
>
> 업그레이드 후 원활한 작업을 위해 업그레이드하기 전에 RabbitMQ 4에 도입된 기능 사용 중단, 주요 변경 사항 및 새로운 기능을 검토하세요.

다음 표에서는 두 업그레이드 접근 방식을 비교합니다.

업그레이드 접근 방식 비교

고려 사항	현재 위치 업그레이드(권장)	블루/그린 배포
다운타임	예, Amazon MQ는 업그레이드 중에 브로커에 대한 모든 연결을 차단합니다. 가동 중지 시간은 대기열 깊이에 따라 달라집니다. 대기열을 짧게 유지하면 가동 중지 시간이 줄어듭니다.	아니요. 가동 중지 없이 생산자와 소비자를 새 브로커로 마이그레이션할 수 있습니다.
애플리케이션 코드 변경	변경할 필요가 없습니다. 브로커 엔드포인트는 업그레이드 후에도 동일하게 유지됩니다.	예, 생산자와 소비자를 새 브로커로 리디렉션하려면 애플리케이션 코드를 업데이트해야 합니다.

RabbitMQ 3에서 4로 현재 위치 업그레이드

Amazon MQ는 RabbitMQ 3.13에서 RabbitMQ 4.2로 인플레이스 메이저 버전 업그레이드를 지원합니다. RabbitMQ 4.2는 지원되는 모든 `mq.m7g` 인스턴스 크기에서 인스턴스 유형에서만 지원됩니다.

> **Note**
>
> RabbitMQ 3용 Amazon MQ 브로커에 Khepri가 활성화된 경우 RabbitMQ 4에 대한 현재 위치 업그레이드 경로가 없습니다. RabbitMQ 자세한 내용은 RabbitMQ 버전 업그레이드 가능성을 참조하세요. 이 경우 블루-그린 배포를 고려하세요.

> **Important**
>
> 업그레이드 기간은 대기열 수와 대기열 깊이에 따라 달라집니다. 대기열과 메시지가 많은 브로커는 가동 중지 시간이 더 길어집니다. 가동 중지 시간을 최소화하려면 대기열을 짧게 유지하세요.

1단계: 브로커 인스턴스 유형 업그레이드

RabbitMQ 4.2에는 `mq.m7g` 인스턴스 유형이 필요합니다. 브로커가 이미 `mq.m7g` 인스턴스 유형에서 실행 중인 경우 로 이동합니다the section called “2단계: 클래식 대기열을 쿼럼 대기열로 마이그레이션”.

UpdateBroker API 작업을 사용하여 브로커의 인스턴스 유형을 로 수정합니다mq.m7g.

RebootBroker API를 사용하여 브로커를 재부팅하여 인스턴스 유형 변경을 적용하거나 예약된 다음 유지 관리 기간을 기다립니다.

자세한 내용은 Amazon MQ 브로커 인스턴스 유형 업그레이드를 참조하세요.

2단계: 클래식 대기열을 쿼럼 대기열로 마이그레이션

RabbitMQ 4에서는 클래식 미러링 대기열이 지원되지 않습니다. 브로커에 클래식 대기열 또는 클래식 미러링 대기열이 있는 경우 Amazon MQ는 RabbitMQ 4로의 현재 위치 업그레이드를 방지합니다.

Amazon MQ는 클래식 대기열을 쿼럼 대기열로 마이그레이션하는 대기열 마이그레이션 도구를 제공합니다. 이 도구는 RabbitMQ 웹 콘솔의 관리자 > 대기열 마이그레이션 또는 HTTP API를 통해 액세스할 수 있습니다.

도구를 사용하려면 Amazon MQ 대기열 마이그레이션 도구를 참조하세요.

3단계: 엔진 버전을 RabbitMQ 3.13에서 4.2로 업그레이드

> **Note**
>
> Amazon MQ는 업그레이드 중에 브로커에 대한 모든 외부 트래픽을 차단합니다.

> **Important**
>
> RabbitMQ 3.13 클러스터 브로커가 암호화에 고객 관리형 키(CMK)를 사용하는 경우를 호출UpdateBroker하여 버전 4.2로 업그레이드하는 데 사용되는 IAM 역할에는 브로커의 암호화 키에 대한 다음 AWS KMS 권한이 있어야 합니다.
>
> - kms:CreateGrant
> - kms:DescribeKey
>
> 호출 역할에 이러한 권한이 없는 경우 UpdateBroker API는 AWS KMS 키에 권한 부여가 필요함을 나타내는 403 오류를 반환합니다. 이 오류를 해결하려면 브로커의 AWS KMS 키 ARN에 대한 IAM 역할의 정책에 kms:CreateGrant 및 kms:DescribeKey 권한을 추가한 다음 업그레이드를 다시 시도하세요.

UpdateBroker API를 사용하여 RabbitMQ 3.13 브로커에 대해 보류 중인 엔진 버전을 4.2로 설정합니다.

브로커를 재부팅하여 변경 사항을 적용하거나 예약된 다음 유지 관리 기간을 기다립니다.

업그레이드 진행 상황 모니터링

DescribeBroker API 또는 Amazon MQ 콘솔의 브로커 격리 상태를 사용하여 업그레이드 진행 상황을 모니터링할 수 있습니다.

Amazon MQ는 업그레이드 시작 시 업그레이드 자격 확인을 실행합니다. 클래식 대기열을 식별하거나 Khepri가 활성화된 경우 Amazon MQ는 작업 필수 코드와 함께 브로커를 `CRITICAL_ACTION_REQUIRED` 상태로 전환합니다`RABBITMQ_BROKER_NOT_UPGRADEABLE_TO_V4`. Amazon MQ는 메이저 버전 업그레이드를 적용하지 않으며 브로커를 게시하고 사용할 수 있도록 합니다.

업그레이드를 계속하려면 기본 문제를 해결합니다. 자세한 내용은 RABBITMQ_BROKER_NOT_UPGRADEABLE_TO_V4를 참조하세요.

업그레이드 후 리소스 제한 구성 업데이트

RabbitMQ 4용 Amazon MQ에는 연결, 채널, 채널당 소비자, 대기열, vhost, 셔블, 교환 및 최대 메시지 크기에 대한 기본 리소스 제한이 도입되었습니다. RabbitMQ 3 브로커에서 이러한 리소스는 최대 리소스 제한으로 구성됩니다. RabbitMQ 4.2로 현재 위치 업그레이드 후 Amazon MQ는 RabbitMQ 3에 사용되는 최대 리소스 제한보다 낮은 RabbitMQ 4 기본 리소스 제한을 적용합니다.

> **Important**
>
> RabbitMQ 3 브로커가 RabbitMQ 4 기본 제한보다 높은 수의 리소스를 사용하는 경우 브로커는 업그레이드 후 새 제한을 초과하는 새 연결, 채널 또는 대기열 선언을 거부할 수 있습니다. 업그레이드하기 전에 인스턴스 유형 및 배포 모드의 기본 리소스 제한을 검토합니다. 업그레이드가 완료되면 브로커 구성을 업데이트하여 워크로드 요구 사항에 맞게 리소스 제한을 조정합니다. 자세한 내용은 리소스 제한 구성을 참조하세요.

RabbitMQ 3에서 4로의 블루-그린 배포

Amazon MQ는 RabbitMQ 3.13에서 RabbitMQ 4.2로 업그레이드하기 위한 관리형 블루-그린 배포 옵션을 제공하지 않습니다. RabbitMQ 블루-그린 배포를 독립적으로 수행하도록 선택한 경우이 접근 방식

을 사용하려면 생산자와 소비자를 새 브로커로 리디렉션하기 위해 애플리케이션 코드를 변경해야 합니다.

자세한 지침은 RabbitMQ 설명서의 블루-그린 배포를 참조하세요.

RabbitMQ용 Amazon MQ 브로커의 배포 옵션

RabbitMQ 브로커는 단일 인스턴스 브로커나 클러스터 배포로 생성할 수 있습니다. 두 배포 모드 모두에서 Amazon MQ는 데이터를 중복 저장하여 높은 내구성을 제공합니다.

RabbitMQ가 지원하는 모든 프로그래밍 언어를 사용하고 다음 프로토콜에 대해 TLS를 활성화하여 RabbitMQ 브로커에 액세스할 수 있습니다.

- AMQP(0-9-1)

주제

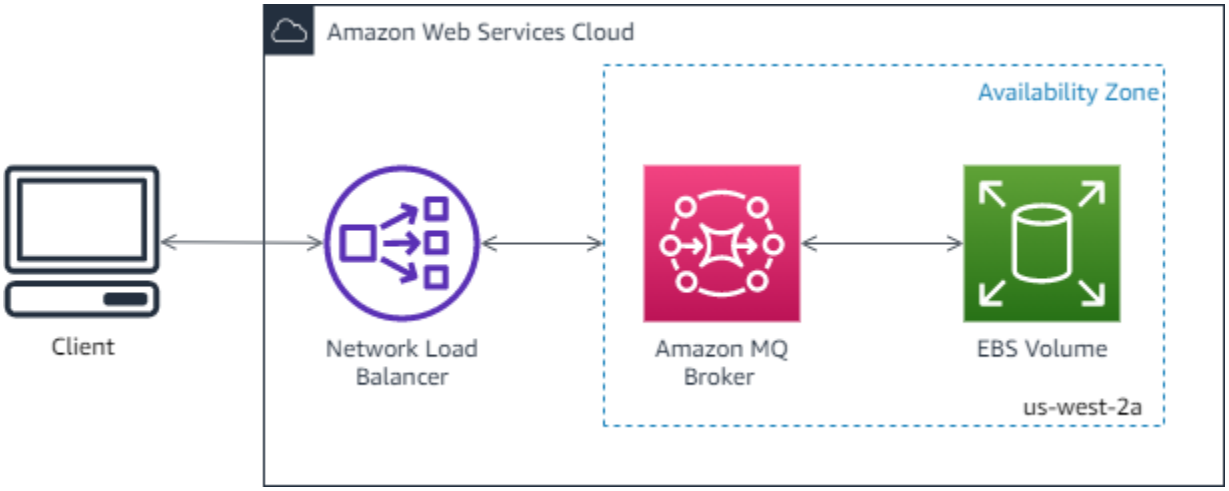
- 옵션 1: RabbitMQ용 Amazon MQ 단일 인스턴스 브로커
- 옵션 2: RabbitMQ용 Amazon MQ 클러스터 배포

옵션 1: RabbitMQ용 Amazon MQ 단일 인스턴스 브로커

단일 인스턴스 브로커는 네트워크 로드 밸런서(NLB) 뒤에서 하나의 가용 영역에 있는 하나의 브로커로 구성됩니다. 브로커는 애플리케이션 및 Amazon EBS 스토리지 볼륨과 통신합니다. Amazon EBS는 짧은 대기 시간과 높은 처리량에 최적화된 블록 수준 스토리지를 제공합니다.

네트워크 로드 밸런서를 사용하면 유지 관리 기간 중이나 기본 Amazon EC2 하드웨어 실패로 인해 브로커 인스턴스가 교체되어도 RabbitMQ용 Amazon MQ 브로커 엔드포인트가 변경되지 않습니다. 네트워크 로드 밸런서를 통해 애플리케이션과 사용자가 계속 같은 엔드포인트를 사용하여 브로커에 연결할 수 있습니다.

다음 다이어그램은 RabbitMQ용 Amazon MQ 단일 인스턴스 브로커를 보여줍니다.



옵션 2: RabbitMQ용 Amazon MQ 클러스터 배포

클러스터 배포는 네트워크 로드 밸런서 뒤에 있는 3개의 RabbitMQ 브로커 노드(각각 사용자, 대기열 및 여러 가용 영역(AZ) 간에 분산된 상태 공유)로 이루어진 논리적 그룹입니다.

클러스터 배포에서 Amazon MQ는 브로커 정책을 자동으로 관리하여 모든 노드에서 클래식 미러링을 활성화하므로 고가용성(HA)을 보장합니다. 미러링된 각 대기열은 하나의 기본 노드와 하나 이상의 미러로 구성됩니다. 대기열마다 자체 기본 노드가 있습니다. 지정된 대기열에 대한 모든 작업은 먼저 대기열의 기본 노드에 적용된 다음 미러로 전파됩니다. Amazon MQ는 `ha-mode` 를 `all`로, `ha-sync-mode`를 `automatic`으로 설정하는 기본 시스템 정책을 생성합니다. 따라서 데이터가 다른 가용 영역에 있는 클러스터의 모든 노드에 복제되어 내구성이 높아집니다.

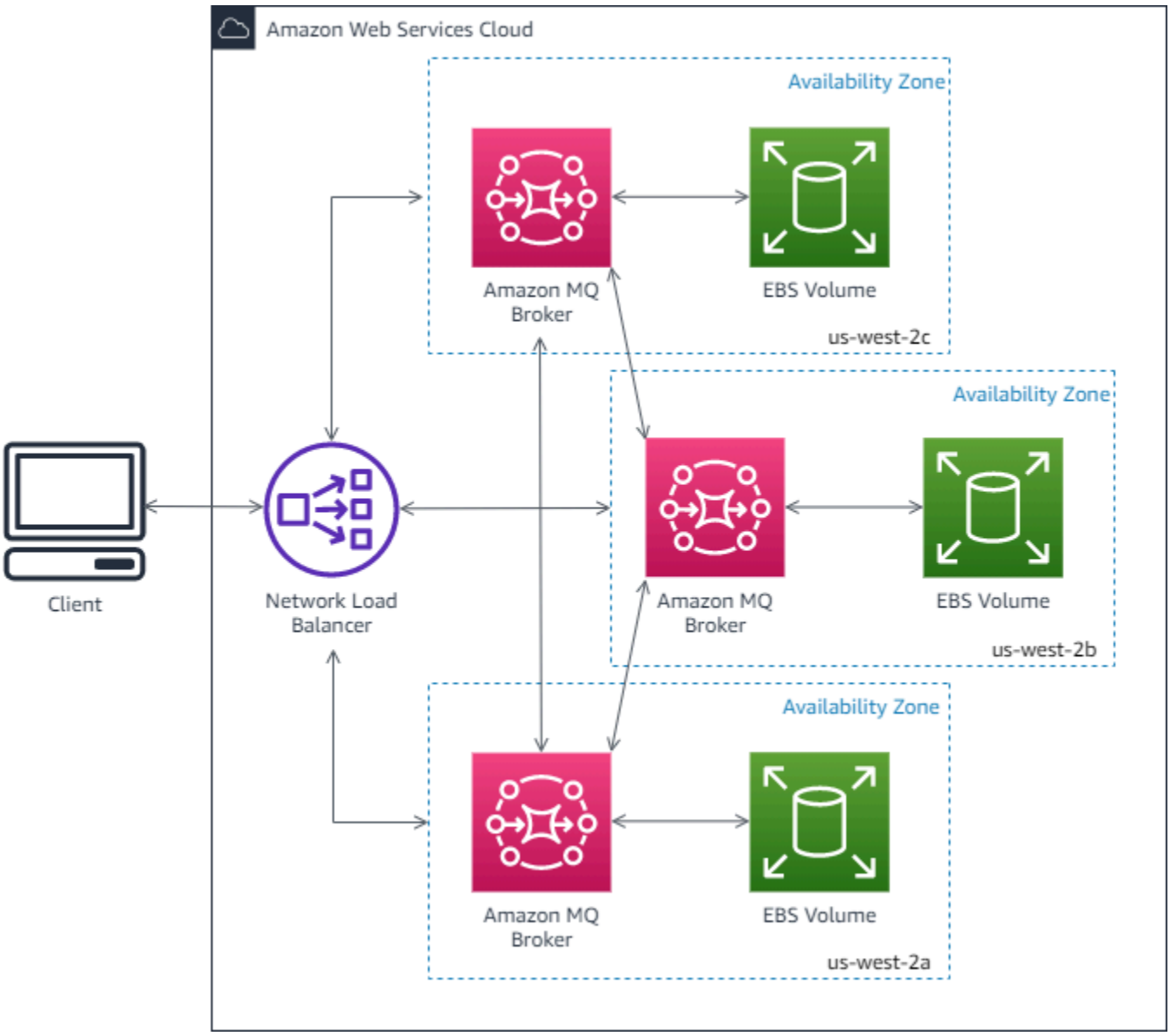
> **Note**
>
> 클러스터 배포에서 가용 영역 중단이 발생하면 Amazon MQ는 클러스터 크기를 유지하기 위해 영향을 받는 RabbitMQ 노드를 다른 AZ로 자동으로 재배치하려고 시도합니다. 중단이 해결되면 클러스터가 AZ 간에 자동으로 재조정됩니다.

> **Note**
>
> 유지 관리 기간 중 클러스터의 모든 유지 관리 작업은 한 번에 한 노드씩 수행되므로 항상 두 개 이상의 노드가 실행 중인 상태로 유지됩니다. 노드가 중단될 때마다 해당 노드의 클라이언트 연결이 끊어지고 다시 설정해야 합니다. 클러스터에 자동으로 다시 연결하도록 클라이언트 코드가 설계되어 있는지 확인해야 합니다. 연결 복구에 대한 자세한 내용은 the section called “1단계: 네트워크 실패 자동 복구” 단원을 참조하세요.

Amazon MQ는 `ha-sync-mode: automatic`을 설정하므로 유지 관리 기간 중 각 노드가 클러스터에 다시 조인할 때 대기열이 동기화됩니다. 대기열 동기화 중에는 다른 모든 대기열 작업이 차단됩니다. 대기열을 짧게 유지하면 유지 관리 기간 중 대기열 동기화의 영향을 완화할 수 있습니다.

기본 정책은 삭제하면 안 됩니다. 이 정책을 삭제하면 Amazon MQ에서 자동으로 다시 생성합니다. 또한 Amazon MQ는 사용자가 클러스터링된 브로커에서 생성하는 다른 모든 정책에도 HA 속성이 적용되도록 합니다. HA 속성 없이 정책을 추가하면 Amazon MQ에서 해당 속성을 자동으로 추가합니다. 고가용성 속성이 다른 정책을 추가하면 Amazon MQ가 해당 정책을 대체합니다. 클래식 미러링에 대한 자세한 내용은 클래식 미러링된 대기열을 참조하세요.

다음 다이어그램은 각각 자체 Amazon EBS 볼륨이 있고 상태를 공유하는 3개 가용 영역(AZ)의 3개 노드가 포함된 RabbitMQ 클러스터 브로커 배포를 보여줍니다. Amazon EBS는 짧은 대기 시간과 높은 처리량에 최적화된 블록 수준 스토리지를 제공합니다.



RabbitMQ용 Amazon MQ 브로커 인스턴스 유형

브로커 인스턴스 클래스(m7g)와 크기(large, medium)의 결합된 설명을 브로커 인스턴스 유형(예: mq.m7g.large)이라고 합니다.

클러스터 및 단일 인스턴스 배포 모두에 mq.m7g 인스턴스 유형을 사용하는 것이 좋습니다.

Amazon MQ는 인스턴스 유형 지원이 종료되기 최소 90일 전에 미리 통지합니다. 중단이 발생하지 않도록 지원 종료일 전에 브로커를 새 인스턴스 유형으로 업그레이드하는 것이 좋습니다.

> **Important**
>
> `mq.m7g` 또는 `mq.m5` 인스턴스 유형에서 `mq.t3.micro` 인스턴스 유형으로 브로커를 다운그레이드할 수 없습니다.
> `mq.t3.micro` 인스턴스 유형은 클러스터 배포를 지원하지 않습니다.

m7g 클러스터 배포를 위한 인스턴스 유형

클러스터 배포와 함께 `mq.m7g.x` 인스턴스 유형을 사용하는 것이 좋습니다. 다음 표에는 클러스터 배포에 사용할 수 있는 `mq.m7g.x` 인스턴스 유형이 나와 있습니다.

인스턴스 유형	vCPU	메모리 (GiB)	네트워크 기준/버스트 대역폭 (Gbps)	권장 사용	스토리지	노드당 디스크 볼륨 크기(GB)
mq.m7g.medium	1	4	0.52/12.5	평가	EBS	5
mq.m7g.large	2	8	0.937 / 12.5	프로덕션	EBS	15
mq.m7g.xlarge	4	16	1.876/12.5	프로덕션	EBS	25
mq.m7g.2xlarge	8	32	3.75/15.0	프로덕션	EBS	45
mq.m7g.4xlarge	16	64	7.5/15.0	프로덕션	EBS	90
mq.m7g.8xlarge	32	128	15기가비트	프로덕션	EBS	175
mq.m7g.12xlarge	48	192	22.5기가비트	프로덕션	EBS	260

인스턴스 유형	vCPU	메모리 (GiB)	네트워크 기준/버스트 대역폭 (Gbps)	권장 사용	스토리지	노드당 디스크 볼륨 크기(GB)
mq.m7g.16xlarge	64	256	30기가비트	프로덕션	EBS	345

m7g 단일 인스턴스 배포를 위한 인스턴스 유형

다음 표에는 단일 인스턴스 배포에 사용할 수 있는 `mq.m7g.x` 인스턴스 유형이 나와 있습니다.

인스턴스 유형	vCPU	메모리 (GiB)	네트워크 기준/버스트 대역폭 (Gbps)	권장 사용	스토리지	노드당 디스크 볼륨 크기(GB)
mq.m7g.medium	1	4	0.52/12.5	평가	EBS	200
mq.m7g.large	2	8	0.937 / 12.5	프로덕션	EBS	200
mq.m7g.xlarge	4	16	1.876/12.5	프로덕션	EBS	200
mq.m7g.2xlarge	8	32	3.75/15.0	프로덕션	EBS	200
mq.m7g.4xlarge	16	64	7.5/15.0	프로덕션	EBS	200
mq.m7g.8xlarge	32	128	15기가비트	프로덕션	EBS	200
mq.m7g.12xlarge	48	192	22.5기가비트	프로덕션	EBS	200

인스턴스 유형	vCPU	메모리 (GiB)	네트워크 기준/버스트 대역폭 (Gbps)	권장 사용	스토리지	노드당 디스크 볼륨 크기(GB)
mq.m7g.16xlarge	64	256	39기가비트	프로덕션	EBS	200

mq.m5 단일 인스턴스 배포의 인스턴스 유형

다음 표에는 단일 인스턴스 배포에 사용할 수 있는 mq.m5.x 인스턴스 유형이 나와 있습니다.

인스턴스 유형	vCPU	메모리 (GiB)	네트워크 기준/버스트 대역폭 (Gbps)	권장 사용	스토리지	노드당 디스크 볼륨 크기(GB)
mq.t3.micro	2	1	0.064/5.0	평가	EBS	20
mq.m5.large	2	8	0.75 / 10.0	프로덕션	EBS	200
mq.m5.xlarge	4	16	1.25 / 10.0	프로덕션	EBS	200
mq.m5.2xlarge	8	32	2.5 / 10.0	프로덕션	EBS	200
mq.m5.4xlarge	16	64	5.0 / 10.0	프로덕션	EBS	200

mq.m5 클러스터 배포를 위한 인스턴스 유형

다음 표에는 클러스터 배포에 사용할 수 있는 mq.m5.x 인스턴스 유형이 나와 있습니다.

인스턴스 유형	vCPU	메모리 (GiB)	네트워크 기준/버스트 대역폭 (Gbps)	권장 사용	스토리지	노드당 디스크 볼륨 크기(GB)
mq.m5.large	2	8	0.75 / 10.0	프로덕션	EBS	200
mq.m5.xlarge	4	16	1.25 / 10.0	프로덕션	EBS	200
mq.m5.2xlarge	8	32	2.5 / 10.0	프로덕션	EBS	200
mq.m5.4xlarge	16	64	5.0 / 10.0	프로덕션	EBS	200

메모리 및 디스크 경보

Amazon MQ는 리소스 소진으로부터 보호하기 위해 각 RabbitMQ 브로커에 메모리 및 디스크 임계값을 구성합니다. 임계값이 초과되면 RabbitMQ는 경보를 트리거하고 게시자가 메시지를 전송하지 못하도록 차단합니다. 별도의 연결에 있는 소비자는 계속 정상적으로 작동합니다. 그러나 게시자와 소비자가 동일한 연결을 공유하는 경우 소비자도 차단됩니다.

> **Important**
>
> Amazon MQ는 이러한 임계값을 관리하므로 수정할 수 없습니다. 경보 조건이 지워지면 게시자가 자동으로 차단 해제됩니다. 문제 해결 정보는 the section called “ RABBITMQ_MEMORY_ALARM ” 및 단원을 참조하십시오the section called “RABBITMQ_DISK_ALARM”.

메모리 경보

`vm_memory_high_watermark` 파라미터는 게시자의 메시지 전송을 차단하기 전에 RabbitMQ 브로커가 사용할 수 있는 최대 메모리 양을 정의합니다. 메모리 사용량이이 임계값을 초과하면 RabbitMQ가 메모리 경보를 트리거합니다. 자세한 내용은 RabbitMQ 웹 사이트의 메모리 경보를 참조하세요.

mq.m7g 인스턴스 유형의 경우 Amazon MQ는 다음과 같은 절대 메모리 하이 워터마크 값을 설정합니다.

인스턴스 유형	메모리 하이 워터마크(GiB)
mq.m7g.medium	1.8
mq.m7g.large	4.3
mq.m7g.xlarge	9.3
mq.m7g.2xlarge	19.3
mq.m7g.4xlarge	39.4
mq.m7g.8xlarge	79.7
mq.m7g.12xlarge	119.8
mq.m7g.16xlarge	160.1

mq.m5 인스턴스 유형의 경우 Amazon MQ는 상대 메모리 하이 워터마크를 0.4(사용 가능한 메모리의 40%)로 설정합니다.

mq.m7g 인스턴스의 메모리 임계값이 높을수록 RabbitMQ는 경보를 트리거하기 전에 사용 가능한 메모리를 더 많이 사용할 수 있습니다. mq.m7g 인스턴스의 성능 개선에 대한 자세한 내용은 AWS 블로그의 AWS Graviton3-based M7g 인스턴스를 사용하여 Amazon MQ에서 RabbitMQ 성능 개선을 참조하세요.

디스크 경보

disk_free_limit 파라미터는 RabbitMQ 노드에 필요한 최소 여유 디스크 공간을 정의합니다. 노드의 여유 디스크 공간이이 제한 아래로 떨어지면 RabbitMQ는 디스크 경보를 트리거하고 게시자가 메시지를 전송하지 못하도록 차단합니다. 자세한 내용은 RabbitMQ 웹 사이트의 디스크 경보를 참조하세요.

mq.m7g 인스턴스 유형의 경우 Amazon MQ는 다음과 같은 디스크 여유 제한을 설정합니다. 단일 인스턴스 브로커는 디스크 공간이 소진된 경우 트래픽을 처리할 다른 노드가 없기 때문에 추가 보호를 제공하기 위해 디스크 여유 한도가 더 높습니다.

배포 모드	디스크 여유 제한(GiB)
단일 인스턴스	10
Cluster	2

mq.m5 인스턴스 유형의 경우 Amazon MQ는 다음과 같은 디스크 여유 제한을 설정합니다. 이러한 값은 단일 인스턴스 및 클러스터 배포 모두에 적용됩니다.

인스턴스 유형	디스크 여유 제한(GiB)
mq.m5.large	12
mq.m5.xlarge	20
mq.m5.2xlarge	36
mq.m5.4xlarge	69

mq.m7g 인스턴스의 디스크 여유 한도가 더 낮기 때문에 동일한 mq.m5 인스턴스에 비해 메시지 스토리지에 프로비저닝된 디스크 볼륨을 더 많이 사용할 수 있습니다.

RabbitMQ용 Amazon MQ 크기 조정 지침

사용 중인 애플리케이션을 가장 잘 지원하는 브로커 인스턴스 유형을 선택할 수 있습니다. 인스턴스 유형을 선택할 때는 브로커 성능에 영향을 미치는 요인을 고려합니다.

- 클라이언트 및 대기열 수
- 전송된 메시지의 양
- 메모리에 보관된 메시지
- 중복 메시지

더 작은 m7g.medium 브로커 인스턴스 유형은 애플리케이션 성능 테스트에만 권장됩니다. 클라이언트 및 대기열의 더 큰 브로커 인스턴스 유형 m7g.large 이상 또는 프로덕션 수준, 높은 처리량, 메모리 내 메시지 및 중복 메시지를 사용하는 것이 좋습니다.

> **Important**
>
> `mq.m5` 또는 `mq.m7g` 인스턴스 유형에서 `mq.t3.micro` 인스턴스 유형으로 브로커를 다운그레이드할 수 없습니다.

브로커를 테스트하여 워크로드 메시징 요구 사항에 적합한 인스턴스 유형과 크기를 결정하는 것이 중요합니다.

항상 RabbitMQ 4 브로커의 기본 리소스 제한을 사용하여 Amazon MQ 모범 사례에 따라 애플리케이션에 적합한 인스턴스 크기를 결정합니다. 이러한 기본 리소스 제한은 유형 `m7g` 인스턴스 유형 및 쿼럼 대기열을 기반으로 합니다.

- m7g 단일 인스턴스 배포에 대한 기본 리소스 제한
- m7g 클러스터 배포에 대한 기본 리소스 제한

인스턴스 유형 및 배포 모드에 정의된 최대 값까지 제한 값을 늘릴 수 있습니다. 그러나 프로덕션 환경에서를 사용하기 전에 증가된 값으로 브로커 성능을 테스트하는 것이 좋습니다.

- m7g 단일 인스턴스 배포에 대한 최대 리소스 제한
- m7g 클러스터 배포에 대한 최대 리소스 제한
- m5 단일 인스턴스 배포에 대한 최대 리소스 제한
- m5 클러스터 배포에 대한 최대 리소스 제한
- 오류 메시지

> **Note**
>
> RabbitMQ 3.13 브로커에는 기본 리소스 제한이 제공되지 않지만 제안된 기본값을 사용하는 것이 좋습니다.

기본 리소스 제한

RabbitMQ용 Amazon MQ는 RabbitMQ 4 이상에서 브로커 리소스 제한 구성을 지원합니다. 브로커를 생성하면 Amazon MQ는 이러한 리소스 제한에 기본값을 자동으로 적용합니다. 이러한 기본값은 일반

적인 고객 사용 패턴을 수용하면서 브로커 가용성을 보호하는 가드레일 역할을 합니다. 특정 워크로드 요구 사항에 더 잘 맞게 제한 구성 값을 변경하여 브로커 동작을 사용자 지정할 수 있습니다.

변경하기 전에 다음 사항에 유의하세요.

> **Important**
>
> 1. 구성 변경은 브로커 성능 및 가용성에 영향을 미칠 수 있습니다.
> 2. 기본 구성 옵션을 변경하기 전에 영향을 이해합니다.
> 3. 먼저 비프로덕션 환경에서 구성 변경 사항 테스트
> 4. 변경 사항 적용 후 브로커 상태 모니터링

> **Important**
>
> 이러한 구성의 기본값과 지원되는 범위는 RabbitMQ 버전, 인스턴스 유형 및 브로커 배포 모드에 따라 다릅니다.

> **Important**
>
> 참고: 지원되는 범위를 벗어난 구성 값과 브로커를 연결하거나 생성하면 오류 응답이 발생합니다.

브로커에 대한 이러한 기본 리소스 제한을 사용자 지정하는 방법은 섹션을 참조하세요the section called “리소스 제한 구성”.

RabbitMQ 4.2 브로커에 적용되는 기본 리소스 제한은 입니다.

- m7g 단일 인스턴스 배포에 대한 기본 리소스 제한
- m7g 클러스터 배포에 대한 기본 리소스 제한

기본 리소스 제한

> **Important**
>
> RabbitMQ 3용 Amazon MQ 브로커, 기본값은 최대 리소스 제한으로 구성되며 Amazon MQ는 리소스 제한 구성을 재정의할 수 있는 기능을 제공하지 않습니다. RabbitMQ

단일 인스턴스 브로커의 기본값

인스턴스 유형	노드당 연결 수	노드당 채널 수	채널당 소비자	Queues	vhost	Shovel	교환	바이트 단위의 메시지 크기
mq.m7g.medium	100	500	10	500	10	30	500	524288
mq.m7g.large	1,500	4,500	10	1,000	50	50	1,000	524288
mq.m7g.xarge	3,000	9,000	10	2,000	100	100	2,000	524288
mq.m7g.2large	6,000	18,000	10	4,000	150	200	4,000	524288
mq.m7g.4large	12,000	36,000	10	8,000	200	400	8,000	524288
mq.m7g.8large	24,000	72,000	10	16,000	250	800	16,000	524288
mq.m7g.1xlarge	36,000	108,000	10	24,000	300	1,200	24,000	524288
mq.m7g.1xlarge	48,000	144,000	10	32,000	350	1,600	32,000	524288

클러스터 브로커의 기본값

인스턴스 유형	노드당 연결 수	노드당 채널 수	채널당 소비자	Queues	vhost	Shovel	교환	바이트 단위의 메시지 크기
mq.m7g.medium	100	300	10	100	10	10	100	524288
mq.m7g.large	500	1500	10	1,000	50	30	1,000	524288
mq.m7g.xlarge	1000	3000	10	2,000	100	60	2,000	524288
mq.m7g.2large	2000	6000	10	4,000	150	120	4,000	524288
mq.m7g.4large	4000	12,000	10	8,000	200	240	8,000	524288
mq.m7g.8large	8000	24,000	10	16,000	250	480	16,000	524288
mq.m7g.1xlarge	12000	36,000	10	24,000	300	720	24000	524288
mq.m7g.1xlarge	16,000	48,000	10	32,000	350	960	32,000	524288

RabbitMQ용 Amazon MQ 최대 리소스 제한 RabbitMQ

다음 표에 표시된 최대값까지 리소스 제한을 구성할 수 있습니다. 브로커의 리소스 제한을 업데이트하는 방법은 섹션을 참조하세요the section called “리소스 제한 구성”.

단일 인스턴스 배포에 대한 쿼럼 대기열이 있는 m7g의 크기 조정 지침

다음 표에는 단일 인스턴스 브로커의 각 인스턴스 유형별 최대 한도 값이 나와 있습니다.

인스턴스 유형	연결	채널	채널당 소비자	Queues	Vhosts	Shovel	교환	바이트 단위의 메시지 크기
mq.m7g.medium	300	900	1,000	2,500	10	150	12500	134217728
mq.m7g.large	5,000	15,000	1,000	20,000건	1500	250	100,000건	134217728
mq.m7g.xlarge	10,000	30,000개	1,000	30,000개	1,500	500	150,000	134217728
mq.m7g.2large	20,000건	60,000	1,000	40,000	1,500	1,000	200,000	134217728
mq.m7g.4large	40,000	120,000	1,000	60,000	1,500	2000	300,000	134217728
mq.m7g.8large	80,000	240,000	1,000	80,000	1,500	4000	400,000	134217728
mq.m7g.12xlarge	120,000	360,000	1,000	100,000건	1,500	6,000	500,000	134217728
mq.m7g.16xlarge	160,000	480,000	1,000	120,000	1,500	8,000	600,000	134217728

클러스터 배포를 위한 쿼럼 대기열이 있는 m7g의 크기 조정 지침

다음 표에는 클러스터 브로커의 각 인스턴스 유형별 최대 한도 값이 나와 있습니다.

인스턴스 유형	노드당 연결 수	노드당 채널 수	채널당 소비자	Queues	Vhosts	Shovel	교환	바이트 단위의 메시지 크기
mq.m7g.medium	300	900	1,000	500	10	50	500	134217728
mq.m7g.large	5,000	15,000	1,000	10,000	1,500	150	50,000	134217728
mq.m7g.xlarge	10,000	30,000개	1,000	15,000	1,500	300	75,000	134217728
mq.m7g.2large	20,000건	60,000	1,000	20,000건	1,500	600	100,000건	134217728
mq.m7g.4large	40,000	120,000	1,000	30,000개	1,500	1200	150,000	134217728
mq.m7g.8large	80,000	240,000	1,000	40,000	1,500	2,400	200,000	134217728
mq.m7g.1xlarge	120,000	360,000	1,000	50,000	1,500	3,600	250,000	134217728
mq.m7g.1xlarge	160,000	480,000	1,000	60,000	1,500	4,800	300,000	134217728

M5 단일 인스턴스 배포에 대한 최대 리소스 제한

다음 표에는 단일 인스턴스 브로커의 각 인스턴스 유형별 최대 한도 값이 나와 있습니다.

인스턴스 유형	연결	채널	채널당 소비자	Queues	Vhosts	Shovel
m5.large	5,000	15,000	1,000	30,000개	1500	250

인스턴스 유형	연결	채널	채널당 소비자	Queues	Vhosts	Shovel
m5.xlarge	10,000	30,000개	1,000	60,000	1500	500
m5.2xlarge	20,000건	60,000	1,000	120,000	1500	1,000
m5.4xlarge	40,000	120,000	1000	240,000	1,000	2,000

m5 클러스터 배포에 대한 최대 리소스 제한

다음 표에는 클러스터 브로커의 각 인스턴스 유형별 최대 한도 값이 나와 있습니다.

인스턴스 유형	Queues	채널당 소비자	Shovel
m5.large	10,000	1,000	150
m5.xlarge	15,000	1,000	300
m5.2xlarge	20,000건	1,000	600
m5.4xlarge	30,000개	1,000	1200

노드당 다음과 같은 연결 및 채널 한도가 적용됩니다.

인스턴스 유형	연결	채널
m5.large	5000	15,000
m5.xlarge	10,000	30,000개
m5.2xlarge	20,000건	60,000
m5.4xlarge	40,000	120,000

클러스터 브로커의 정확한 한도 값은 사용 가능한 노드 수와 RabbitMQ가 사용 가능한 노드 간에 리소스를 분배하는 방식에 따라 표시된 값보다 낮을 수 있습니다. ㅂ 한도 값을 초과하는 경우 다른 노드에

대한 연결을 새로 생성하고 다시 시도하거나 인스턴스 크기를 업그레이드하여 최대 한도를 늘릴 수 있습니다.

오류 메시지

한도를 초과하면 다음 오류 메시지가 반환됩니다. 모든 값은 **m7.large** 단일 인스턴스 한도를 기준으로 합니다.

> **Note**
>
> 다음 메시지의 오류 코드는 사용 중인 클라이언트 라이브러리에 따라 변경될 수 있습니다.

Connection

```
ConnectionClosedByBroker 500 "NOT_ALLOWED - connection refused: node
connection limit (5000) is reached"
```

Channel

```
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)"
```

소비자

```
ConnectionClosedByBroker: (530, 'NOT_ALLOWED - reached maximum (1,000) of
consumers per channel')
```

최대 메시지 크기

```
(406, 'PRECONDITION_FAILED - message size 524289 is larger than configured
max size 524288')
```

교환

```
(406, "PRECONDITION_FAILED - cannot declare exchange 'limit_test_3' in
vhost '/': exchange limit of 10 is reached")
```

> **Note**
>
> 다음 오류 메시지는 HTTP Management API 형식을 사용합니다.

대기열

```
{"error":"bad_request","reason":"cannot declare queue 'my_queue': queue
limit in cluster (10,000) is reached"}]
```

Shovel

```
{"error":"bad_request","reason":"Validation failed\n\ncomponent shovel is
limited to 150 per node\n"}
```

Vhost

```
{"error":"bad_request","reason":"cannot create vhost 'my_vhost': vhost
limit of 1500 is reached"}
```

RabbitMQ용 Amazon MQ 브로커 기본값

RabbitMQ용 Amazon MQ 브로커를 생성할 때 Amazon MQ는 브로커 성능을 최적화하기 위해 기본 집합의 브로커 정책 및 vhost 제한을 적용합니다. Amazon MQ는 vhost 제한을 기본(/) vhost에만 적용합니다. Amazon MQ는 새로 생성된 vhost에 대해 기본 정책을 적용하지 않습니다. 새 브로커와 기존 브로커 모두에 대해 이러한 기본값을 유지하는 것이 좋습니다. 하지만 언제든 이러한 기본값을 수정, 재정의 또는 삭제할 수 있습니다.

Amazon MQ는 RabbitMQ 3 및 RabbitMQ 4용 Amazon MQ에 대해 서로 다른 브로커 정책 및 vhost 제한을 생성합니다. RabbitMQ 차이점은 다음 하위 섹션에서 자세히 설명합니다.

Amazon MQ는 브로커를 생성할 때 선택하는 인스턴스 유형 및 브로커 배포 모드에 따라 정책과 제한을 생성합니다. 기본 정책은 다음과 같이 배포 모드에 따라 이름이 지정됩니다.

RabbitMQ 3용 Amazon MQ: RabbitMQ

- 단일 인스턴스 – `AWS-DEFAULT-POLICY-SINGLE-INSTANCE`
- 클러스터 배포 - `AWS-DEFAULT-POLICY-CLUSTER-MULTI-AZ` && `AWS-DEFAULT-QUORUM-QUEUES-POLICY-CLUSTER-MULTI-AZ`

RabbitMQ 4용 Amazon MQ: RabbitMQ

- 단일 인스턴스 – `AWS-DEFAULT-POLICY-SINGLE-INSTANCE`
- 클러스터 배포 - `AWS-DEFAULT-POLICY-CLUSTER` && `AWS-DEFAULT-QUORUM-QUEUES-POLICY-CLUSTER-MULTI-AZ`

단일 인스턴스 브로커의 경우 Amazon MQ는 정책 우선 순위 값을 0으로 설정합니다. 기본 우선 순위 값을 재정의하려면 높은 우선 순위 값으로 사용자 지정 정책을 직접 생성할 수 있습니다. 클러스터 배포의 경우 Amazon MQ는 브로커 기본값의 우선 순위 값을 1로 설정합니다. 클러스터의 고유한 사용자 지정 정책을 생성하려면 1보다 큰 우선 순위 값을 할당합니다.

> **Note**
>
> 클러스터 배포에서 `ha-mode` 및 `ha-sync-mode` 브로커 정책은 클래식 미러링 및 고가용성(HA)에 필요합니다. 이러한 설정은 RabbitMQ 3용 Amazon MQ에만 적용되며 RabbitMQ 4에는 구성되지 않습니다.
>
> 기본 `AWS-DEFAULT-POLICY-CLUSTER-MULTI-AZ` 정책을 삭제하면 Amazon MQ는 우선 순위 값이 `0`인 `ha-all-AWS-OWNED-DO-NOT-DELETE` 정책을 사용합니다. 그러면 필수 `ha-mode` 및 `ha-sync-mode` 정책이 계속 적용됩니다. 고유한 사용자 지정 정책을 생성하는 경우 Amazon MQ는 정책 정의에 `ha-mode` 및 `ha-sync-mode`를 자동으로 추가합니다.

주제

- 정책 및 제한 설명
- 권장 기본값

정책 및 제한 설명

다음 목록에서는 Amazon MQ가 새로 생성되는 브로커에 적용하는 기본 정책 및 제한에 대해 설명합니다. `max-length` , `max-queues` 및 `max-connections`의 값은 브로커의 인스턴스 유형 및 배포 모드에 따라 다릅니다. 해당 값은 권장 기본값 단원에 나와 있습니다.

RabbitMQ 3 및 RabbitMQ 4 브로커 모두에 대한 설정

- **`queue-mode: lazy`**(정책) - 지연 대기열을 활성화합니다. 기본적으로 대기열은 메시지의 인 메모리 캐시를 유지하여 브로커가 메시지를 소비자에게 가능한 한 빨리 전달할 수 있도록 합니다. 이로 인해 브로커의 메모리가 부족해지고 고용량 메모리 경보가 발생할 수 있습니다. 지연 대기열은 가능한 쉽게 메시지를 디스크로 이동하려고 합니다. 따라서 정상 작동 조건에서 메모리에 유지되는 메시지 수가 줄어듭니다. 지연 대기열을 사용하면 RabbitMQ용 Amazon MQ에서 더 큰 메시징 로드와 더 긴 대기열을 지원할 수 있습니다. 특정 사용 사례에서는 지연 대기열을 사용하는 브로커의 성능이 약간 느려질 수 있습니다. 이는 인 메모리 캐시에서 메시지를 전달하는 것이 아니라 메시지가 디스크에서 브로커로 이동되기 때문입니다.

> 배포 모드
>
> 단일 인스턴스, 클러스터

- **`max-length: number-of-messages`**(정책) - 대기열의 메시지 수에 대한 제한을 설정합니다. 클러스터 배포에서 이 제한은 브로커 재부팅과 같은 경우에나 유지 관리 기간 이후에 대기열 동기화 일시 중지를 방지합니다.

> 배포 모드
>
> Cluster

- **`overflow: reject-publish`**(정책) - `max-length` 정책을 사용하는 대기열에서 대기열의 메시지 수가 `max-length` 값에 도달한 후 새 메시지를 거부하도록 합니다. 대기열이 오버플로 상태이더라도 메시지가 손실되지 않도록 하가 위해 브로커에 메시지를 게시하는 클라이언트 애플리케이션은 게시자 확인을 구현해야 합니다. 게시자 확인을 구현하는 방법에 대한 자세한 내용은 RabbitMQ 웹 사이트에서 게시자 확인을 참조하세요.

> 배포 모드
>
> Cluster

RabbitMQ 3 관련 설정

- **`max-queues: number-of-queues-per-vhost`**(vhost 제한) - 브로커의 대기열 수에 대한 제한을 설정합니다. `max-length` 정책 정의와 마찬가지로 클러스터 배포에서 대기열 수를 제한하면 브로커 재부팅이나 유지 관리 기간 이후에 대기열 동기화 일시 중지가 방지됩니다. 대기열을 제한하면 대기열을 유지하기 위해 과도한 양의 CPU가 사용되는 것도 방지됩니다.

> 배포 모드
>
> 단일 인스턴스, 클러스터

- **`max-connections: number-of-connections-per-vhost`**(vhost 제한) - 브로커에 대한 클라이언트 연결 수의 제한을 설정합니다. 권장 값에 따라 연결 수를 제한하면 과도한 브로커 메모리 사용으로 브로커에서 고용량 메모리 경보가 발생하고 작동이 일시 중지될 수 있는 문제가 방지됩니다.

배포 모드

단일 인스턴스, 클러스터

권장 기본값

Important

`max-queues` 및 `max-connections`는 RabbitMQ 3용 Amazon MQ에만 적용됩니다. RabbitMQ

Note

`max-length` 및 `max-queue` 기본 제한은 평균 메시지 크기 5kB를 기준으로 테스트 및 평가됩니다. 메시지가 5kB보다 훨씬 큰 경우 `max-length` 및 `max-queue` 제한을 조정하고 줄여야 합니다.

다음 표에는 새로 생성된 브로커의 기본 제한 값이 나와 있습니다. Amazon MQ는 브로커의 인스턴스 유형 및 배포 모드에 따라 이러한 값을 적용합니다.

인스턴스 유형	Deployment mode(배포 모드)	**max-length**	**max-queues**	**max-connections**
mq.m7g.medium	단일 인스턴스	해당 사항 없음	2,500	100
	Cluster	500,000	100	100
mq.m7g.large	단일 인스턴스	해당 사항 없음	20,000건	5,000
	Cluster	8,000,000	10,000	5,000
mq.m7g.xlarge	단일 인스턴스	해당 사항 없음	30,000개	10,000
	Cluster	9,000,000	15,000	10,000

인스턴스 유형	Deployment mode(배포 모드)	**max-length**	**max-queues**	**max-connections**
mq.m7g.2xlarge	단일 인스턴스	해당 사항 없음	40,000	20,000건
	Cluster	10,000,000	40,000	20,000건
mq.m7g.4xlarge	단일 인스턴스	해당 사항 없음	60,000	40,000
	Cluster	12,000,000	30,000개	40,000
mq.m7g.8xlarge	단일 인스턴스	해당 사항 없음	80,000	80,000
	Cluster	20,000,000	40,000	80,000
mq.m7g.12xlarge	단일 인스턴스	해당 사항 없음	100,000건	120,000
	Cluster	30,000,000	20,000건	120,000
mq.m7g.16xlarge	단일 인스턴스	해당 사항 없음	120,000	160,000
	Cluster	40,000,000	50,000	160,000

인스턴스 유형	Deployment mode(배포 모드)	**max-length**	**max-queues**	**max-connections**
t3.micro	단일 인스턴스	해당 사항 없음	500	500
m5.large	단일 인스턴스	해당 사항 없음	20,000건	4,000
m5.large	Cluster	8,000,000	10,000	15,000
m5.xlarge	단일 인스턴스	해당 사항 없음	30,000개	8,000
m5.xlarge	Cluster	9,000,000	10,000	20,000건
m5.2xlarge	단일 인스턴스	해당 사항 없음	60,000	15,000
m5.2xlarge	Cluster	10,000,000	10,000	40,000

인스턴스 유형	Deployment mode(배포 모드)	`max-length`	`max-queues`	`max-connections`
m5.4xlarge	단일 인스턴스	해당 사항 없음	150,000	30,000개
m5.4xlarge	Cluster	12,000,000	10,000	100,000건

RabbitMQ 브로커 구성

구성에는 Cuttlefish 형식의 RabbitMQ 브로커에 대한 모든 설정이 포함됩니다. 브로커를 생성하기 전에 구성을 생성할 수 있습니다. 그런 다음 구성을 하나 이상의 브로커에 적용할 수 있습니다.

속성

브로커 구성에는 여러 속성이 있습니다. 예시는 다음과 같습니다.

- 이름(MyConfiguration)
- ID(c-1234a5b6-78cd-901e-2fgh-3i45j6k178l9)
- Amazon 리소스 이름(ARN)(arn:aws:mq:us-east-2:123456789012:configuration:c-1234a5b678cd-901e-2fgh-3i45j6k178l9)

구성 속성의 전체 목록은 Amazon MQ REST API 참조의 다음 단원을 참조하세요.

- REST 작업 ID: 구성
- REST 작업 ID: 구성

구성 개정 속성의 전체 목록은 다음 단원을 참조하세요.

- REST 작업 ID: 구성 개정
- REST 작업 ID: 구성 개정

주제

- RabbitMQ 브로커 구성 생성 및 적용
- RabbitMQ용 Amazon MQ 구성 개정 편집 RabbitMQ

- Amazon MQ의 RabbitMQ에 대해 구성 가능한 값
- RabbitMQ 구성의 ARN 지원

RabbitMQ 브로커 구성 생성 및 적용

구성에는 RabbitMQ 브로커에 대한 모든 설정이 Cuttlefish 형식으로 포함됩니다. 브로커를 생성하기 전에 구성을 생성할 수 있습니다. 그런 다음 구성을 하나 이상의 브로커에 적용할 수 있습니다.

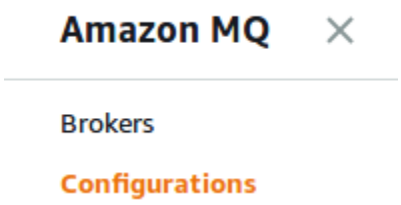
다음 예제에서는 AWS Management Console을 사용하여 RabbitMQ 브로커 구성을 생성하고 적용하는 방법을 보여줍니다.

> **Important**
>
> `DeleteConfiguration` API를 사용해서만 구성을 삭제할 수 있습니다. 자세한 내용은 Amazon MQ API 참조의 구성을 참조하세요.

새 구성 생성

브로커에 구성을 적용하려면 먼저 구성을 생성해야 합니다.

1. Amazon MQ 콘솔에 로그인합니다.
2. 왼쪽에서 탐색 창을 확장하고 Configurations(구성)를 선택합니다.



3. Configurations(구성) 페이지에서 Create configuration(구성 생성)을 선택합니다.
4. 구성 생성 페이지의 세부 정보(Details) 섹션에서 구성 이름(Configuration name)(예: `MyConfiguration`)을 입력하고 브로커 엔진(Broker engine) 버전을 선택합니다.

   RabbitMQ용 Amazon MQ에서 지원하는 RabbitMQ 엔진 버전에 대한 자세한 내용은 the section called “버전 관리” 섹션을 참조하세요.
5. 구성 생성을 선택합니다.

새로운 구성 개정 생성

구성을 생성한 후에는 구성 개정을 사용하여 구성을 편집해야 합니다.

1. 구성 목록에서 ***MyConfiguration***을 선택합니다.

   > **Note**
   > 첫 번째 구성 개정은 Amazon MQ가 구성을 생성할 때 항상 생성됩니다.

   MyConfiguration 페이지에서 새로운 구성 개정에 사용되는 브로커 엔진 유형 및 버전(예: RabbitMQ 3.xx.xx)을 볼 수 있습니다.
2. 구성 세부 정보 탭에 구성 개정 번호, 설명 및 Cuttlefish 형식의 브로커 구성이 표시됩니다.

   > **Note**
   > 현재 구성을 편집하면 새 구성 개정이 생성됩니다.
3. 구성 편집을 선택하고 Cuttlefish 구성을 변경합니다.
4. 저장을 선택합니다.

   Save revision(개정 버전 저장) 대화 상자가 표시됩니다.
5. (선택 사항) `A description of the changes in this revision`을 입력합니다.
6. 저장을 선택합니다.

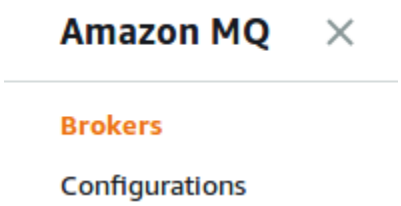
   구성의 새 개정 버전이 저장됩니다.

   > **Important**
   > 구성에 대한 변경 사항이 있어도 즉시 브로커에 변경 사항이 적용되지 않습니다. 변경 내용을 적용하려면 다음 유지 관리 기간을 기다리거나 브로커를 재부팅해야 합니다.
   > 현재 구성을 삭제할 수는 없습니다.

브로커에 구성 개정 적용

구성 개정을 생성한 후에는 구성 개정을 브로커에 적용할 수 있습니다.

1. 왼쪽에서 탐색 창을 확장하고 Brokers(브로커)를 선택합니다.



2. 브로커 목록에서 브로커(예: MyBroker)를 선택한 다음 Edit(편집)을 선택합니다.
3. [Edit ***MyBroker***] 페이지의 [Configuration] 섹션에서 [Configuration] 및 [Revision]을 선택하고 [Schedule Modifications]를 선택합니다.
4. Schedule broker modifications(브로커 수정 예약) 섹션에서 During the next scheduled maintenance window(예약된 다음 유지 관리 기간 동안) 또는 Immediately(즉시) 중에 수정을 적용할 시점을 선택합니다.

> **Important**
>
> 단일 인스턴스 브로커는 재부팅되는 동안 오프라인 상태입니다. 클러스터 브로커의 경우 브로커가 재부팅되는 동안 한 번에 하나의 노드만 다운됩니다.

5. 적용을 선택합니다.

   지정된 시간에 구성 개정이 브로커에 적용됩니다.

RabbitMQ용 Amazon MQ 구성 개정 편집

다음 지침은 브로커의 구성 개정을 편집하는 방법에 대해 설명합니다.

1. Amazon MQ 콘솔에 로그인합니다.
2. 브로커 목록에서 브로커(예: MyBroker)를 선택한 다음 Edit(편집)을 선택합니다.
3. ***MyBroker*** 페이지에서 Edit(편집)를 선택합니다.
4. Edit ***MyBroker***(MyBroker 편집) 페이지의 구성 섹션에서 구성 및 개정을 선택하고 편집을 선택합니다.

> **Note**
>
> 브로커를 생성할 때 구성을 선택하지 않는 한 첫 번째 구성 개정은 Amazon MQ가 브로커를 생성할 때 항상 생성됩니다.

MyBroker 페이지에서 구성에 사용되는 브로커 엔진 유형 및 버전(예: RabbitMQ 3.xx.xx)을 볼 수 있습니다.

5. 구성 세부 정보 탭에 구성 개정 번호, 설명 및 Cuttlefish 형식의 브로커 구성이 표시됩니다.

> **Note**
>
> 현재 구성을 편집하면 새 구성 개정이 생성됩니다.

6. 구성 편집을 선택하고 Cuttlefish 구성을 변경합니다.
7. 저장을 선택합니다.

   Save revision(개정 버전 저장) 대화 상자가 표시됩니다.
8. (선택 사항) `A description of the changes in this revision`을 입력합니다.
9. 저장을 선택합니다.

   구성의 새 개정 버전이 저장됩니다.

> **Important**
>
> 구성에 대한 변경 사항이 있어도 즉시 브로커에 변경 사항이 적용되지 않습니다. 변경 내용을 적용하려면 다음 유지 관리 기간을 기다리거나 브로커를 재부팅해야 합니다.
> 현재 구성을 삭제할 수는 없습니다.

구성 가능한 값

AWS Management Console에서 브로커 구성 파일을 수정하여 다음과 같은 브로커 구성 옵션 값을 설정할 수 있습니다.

다음 표에 설명된 값 외에도 Amazon MQ는 인증 및 권한 부여와 관련된 추가 브로커 구성 옵션과 리소스 제한을 지원합니다. 이러한 구성 옵션에 대한 자세한 내용은 섹션을 참조하세요.

- OAuth 2.0 구성
- LDAP 구성
- HTTP 구성
- SSL 구성

- mTLS 구성
- ARN 지원
- 리소스 제한
- AMQP 클라이언트 SSL 구성

구성	기본 값	권장 값	값	적용 가능한 버전	설명
consumer_timeout	1800000밀리초(30분)	1800000밀리초(30분)	0~2,147,483,647ms. Amazon MQ는 '무한'을 의미하는 값 0도 지원합니다.	모든 버전	소비자가 전송을 확인하지 않는 경우를 감지하기 위한 소비자 전송 승인 제한 시간입니다.
heartbeat	60초	60초	60~3600초	모든 버전	RabbitMQ에서 연결을 사용할 수 없는 것으로 간주하기까지의 시간을 정의합니다.
management.restrictions.operator_policy_changes.disabled	true	true	true, false	모든 버전	운영자 정책 변경 기능을 끕니다. 이렇게 변경하는 경우 자체 운영자 정책에 HA 속성을 포함하는 것이 좋습니다.

구성	기본 값	권장 값	값	적용 가능한 버전	설명
quorum_queue.property_equivalence.relaxed_checks_on_redeclaration	true	true	true, false	모든 버전	TRUE로 설정하면 애플리케이션에서 쿼럼 대기열을 다시 선언할 때 채널 예외를 방지합니다.
secure.management.http.headers.enabled	true	true	true, false	모든 버전	수정할 수 없는 HTTP 보안 헤더를 켭니다.

소비자 전송 승인 구성

소비자가 전송을 승인하지 않는 경우를 감지하도록 consumer_timeout을 구성할 수 있습니다. 소비자가 제한 시간 값 내에 확인을 보내지 않으면 채널이 닫힙니다. 예를 들어 기본값 1800000밀리초를 사용하는 경우 소비자가 1800000밀리초 이내에 전송 승인을 보내지 않으면 채널이 닫힙니다. Amazon MQ는 '무한'을 의미하는 값 0도 지원합니다.

하트비트 구성

하트비트 제한 시간을 구성하여 연결이 중단되거나 실패한 시점을 확인할 수 있습니다. 하트비트 값은 연결이 끊긴 것으로 간주되기까지의 시간 제한을 정의합니다.

운영자 정책 구성

각 가상 호스트의 기본 운영자 정책에는 다음과 같은 권장 HA 속성이 있습니다.

```
{
"name": "default_operator_policy_AWS_managed",
"pattern": ".*",
"apply-to": "all",
"priority": 0,
"definition": {
```

```
"ha-mode": "all",
"ha-sync-mode": "automatic"
}
}
```

AWS Management Console 또는 관리 API를 통한 연산자 정책 변경은 기본적으로 사용할 수 없습니다. 브로커 구성에 다음 줄을 추가하여 변경을 구현할 수 있습니다.

```
management.restrictions.operator_policy_changes.disabled=false
```

이렇게 변경하는 경우 자체 운영자 정책에 HA 속성을 포함하는 것이 좋습니다.

대기열 선언에 대한 느슨한 검사 구성

클래식 대기열을 쿼럼 대기열로 마이그레이션했지만 클라이언트 코드를 업데이트하지 않은 경우 quorum_queue.property_equivalence.relaxed_checks_on_redeclaration을 true로 구성하여 쿼럼 대기열을 다시 선언할 때 채널 예외를 방지할 수 있습니다.

HTTP 보안 헤더 구성

secure.management.http.headers.enabled 구성은 다음 HTTP 보안 헤더를 활성화합니다.

- X-Content-Type-Options: nosniff: 브라우저가 웹 사이트의 파일 형식을 추론하는 데 사용되는 알고리즘인 콘텐츠 스니핑을 수행하지 못하도록 합니다.
- X-Frame-Options: DENY: 다른 사람이 관리 플러그인을 자신의 웹 사이트의 프레임에 임베딩하여 사용자를 속이는 것을 방지합니다.
- Strict-Transport-Security: max-age=47304000; includeSubDomains: 브라우저가 이후에 웹 사이트 및 하위 도메인에 장기간(1.5년) 연결할 때 HTTPS를 사용하도록 강제합니다.

버전 3.10 이상에서 생성된 RabbitMQ용 Amazon MQ 브로커에는 기본적으로 secure.management.http.headers.enabled가 true로 설정됩니다. secure.management.http.headers.enabled를 true로 설정하여 이러한 HTTP 보안 헤더를 켤 수 있습니다. 이러한 HTTP 보안 헤더를 옵트아웃하려면 secure.management.http.headers.enabled를 false로 설정합니다.

OAuth 2.0 인증 및 권한 부여 구성

OAuth 2.0 구성 옵션 및 브로커에 대한 OAuth 2.0 인증 설정에 대한 자세한 내용은 지원되는 OAuth 2.0 구성 및 OAuth 2.0 인증 및 권한 부여 사용을 참조하세요.

LDAP 인증 및 권한 부여 구성

LDAP 구성 옵션 및 브로커에 대한 LDAP 인증 설정에 대한 자세한 내용은 지원되는 LDAP 구성 및 섹션을 참조하세요LDAP 인증 및 권한 부여 사용.

HTTP 인증 및 권한 부여 구성

HTTP 인증 구성 값 및 브로커에 대한 HTTP 인증 설정에 대한 자세한 내용은 HTTP 인증 및 권한 부여 및 섹션을 참조하세요HTTP 인증 및 권한 부여 사용.

> **Note**
>
> HTTP 인증 플러그인은 RabbitMQ용 Amazon MQ 버전 4 이상에서만 사용할 수 있습니다. RabbitMQ

SSL 인증서 인증 구성

SSL 인증서 인증 구성 값 및 브로커에 대한 SSL 인증서 인증 설정에 대한 자세한 내용은 SSL 인증서 인증 및 섹션을 참조하세요SSL 인증서 인증 사용.

> **Note**
>
> SSL 인증서 인증 플러그인은 RabbitMQ용 Amazon MQ 버전 4 이상에서만 사용할 수 있습니다. RabbitMQ

mTLS 구성

RabbitMQ용 Amazon MQ는 다양한 엔드포인트 및 외부 서비스에 대한 보안 연결을 위해 상호 TLS(mTLS)를 지원합니다. mTLS는 클라이언트와 서버 모두 인증서를 사용하여 인증하도록 요구하여 향상된 보안을 제공합니다.

> **Note**
>
> mTLS에 대한 프라이빗 인증 기관 사용은 RabbitMQ용 Amazon MQ 버전 4 이상에서만 사용할 수 있습니다. RabbitMQ

> **Important**
>
> RabbitMQ용 Amazon MQ는 인증서 및 프라이빗 키 파일에 AWS ARNs 사용을 적용합니다. 자세한 내용은 RabbitMQ 구성의 ARN 지원을 참조하세요.

이 페이지의 내용

AMQP 엔드포인트

AMQP 엔드포인트에 대한 클라이언트 연결을 위해 mTLS를 구성합니다. SSL 인증서 인증에 사용됩니다. 지원되는 구성은 섹션을 참조하세요SSL 인증서 인증.

RabbitMQ 관리 플러그인

RabbitMQ 관리 인터페이스에 연결하도록 mTLS를 구성합니다.

> **Note**
>
> 관리 API에는 엄격한 mTLS가 지원되지 않습니다.

지원되는 구성

`aws.arns.management.ssl.cacertfile`

관리 인터페이스에 연결하는 클라이언트 인증서를 검증하기 위한 인증 기관 파일입니다.

`management.ssl.verify`

피어 확인 모드. 지원되는 값: `verify_none`, `verify_peer`

`management.ssl.depth`

확인을 위한 최대 인증서 체인 깊이입니다.

`management.ssl.hostname_verification`

호스트 이름 확인 모드. 지원되는 값: `wildcard`, `none`

지원되지 않는 SSL 옵션

다음 SSL 구성 값은 지원되지 않습니다.

전체 목록 보기

- `management.ssl.cert`
- `management.ssl.client_renegotiation`
- `management.ssl.dh`
- `management.ssl.dhfile`
- `management.ssl.fail_if_no_peer_cert`
- `management.ssl.honor_cipher_order`
- `management.ssl.honor_ecc_order`
- `management.ssl.key.RSAPrivateKey`
- `management.ssl.key.DSAPrivateKey`
- `management.ssl.key.PrivateKeyInfo`
- `management.ssl.log_alert`
- `management.ssl.password`
- `management.ssl.psk_identity`
- `management.ssl.reuse_sessions`
- `management.ssl.secure_renegotiate`
- `management.ssl.versions.$version`
- `management.ssl.sni`

RabbitMQ OAuth 2.0 플러그인

Amazon MQ에서 OAuth 2.0 자격 증명 공급자로의 연결을 위해 mTLS를 구성합니다. 지원되는 구성은 섹션을 참조하세요OAuth 2.0 인증 및 권한 부여.

RabbitMQ HTTP 인증 플러그인

Amazon MQ에서 HTTP 인증 서버로의 연결을 위해 mTLS를 구성합니다. 지원되는 구성은 섹션을 참조하세요HTTP 인증 및 권한 부여.

RabbitMQ LDAP 플러그인

Amazon MQ에서 LDAP 서버로의 연결을 위해 mTLS를 구성합니다. 지원되는 구성은 섹션을 참조하세요LDAP 인증 및 권한 부여.

AMQP 클라이언트 연결

페더레이션 및 셔블에서 사용하는 AMQP 클라이언트 연결에 대한 TLS 피어 확인을 구성합니다. 자세한 내용은 AMQP 클라이언트 SSL 구성을 참조하세요.

> **Important**
>
> Amazon MQ는 현재 AMQP 클라이언트 연결을 위한 클라이언트 인증서 구성을 지원하지 않습니다. 따라서 페더레이션 및 셔블은 클라이언트 인증서 인증이 필요한 mTLS 지원 브로커에 연결할 수 없습니다.

리소스 제한 구성

RabbitMQ용 Amazon MQ는 RabbitMQ 4 이상에서 브로커 리소스 제한 구성을 지원합니다. 브로커를 생성하면 Amazon MQ는 이러한 리소스 제한에 기본값을 자동으로 적용합니다. 이러한 기본값은 일반적인 고객 사용 패턴을 수용하면서 브로커 가용성을 보호하는 가드레일 역할을 합니다. 특정 워크로드 요구 사항에 더 잘 맞게 제한 구성 값을 변경하여 브로커 동작을 사용자 지정할 수 있습니다. 기본 및 최대 허용 값에 대한 자세한 내용은 섹션을 참조하세요the section called “크기 조정 지침”.

리소스 이름 및 구성 키

리소스 이름	구성 키
Connection	`connection_max`
채널	`channel_max_per_node`
대기열	`cluster_queue_limit`

리소스 이름	구성 키
Vhost	`vhost_max`
Shovel	`runtime_parameters.limits.shovel`
Exchange	`cluster_exchange_limit`
채널당 소비자	`consumer_max_per_channel`
최대 메시지 크기	`max_message_size`

리소스 제한을 재정의하는 방법

Amazon MQ API 및 Amazon MQ 콘솔을 사용하여 리소스 제한을 재정의할 수 있습니다.

다음 예제에서는를 사용하여 대기열 수 기본 제한을 재정의하는 방법을 보여줍니다 AWS CLI.

```
aws mq update-configuration --configuration-id <config-id> --data "$(echo
 "cluster_queue_limit=500" | base64 --wrap=0)"
```

호출이 성공하면 구성 개정이 생성됩니다. 구성을 RabbitMQ 브로커에 연결하고 브로커를 재부팅하여 재정의를 적용해야 합니다. 자세한 내용은 섹션을 참조하세요. RabbitMQ Broker Configurations

구성에서 인스턴스별 섹션 지원

RabbitMQ 4에서는 Amazon MQ가 구성 데이터의 섹션을 지원합니다. 섹션을 사용하면 단일 구성 내에서 인스턴스별 리소스 제한을 정의할 수 있습니다. 각 섹션은 특정 인스턴스 유형 및 배포 모드 조합에 해당합니다. 구성을 브로커와 연결하면 Amazon MQ는 브로커의 인스턴스 유형 및 배포 모드에 일치하는 섹션을 자동으로 적용합니다.

> **Important**
>
> 섹션 지원은 RabbitMQ 4에서만 사용할 수 있습니다. 섹션이 포함된 구성을 RabbitMQ 3 브로커에 적용하려고 하면 API가를 반환합니다`BadRequestException`.

섹션 구문

섹션은 다음 형식의 이중 중괄호로 구분됩니다.

```
{{<host-instance-family>.<size>.<mode>}}
```

mode 값은 배포 모드를 나타냅니다.

- 1 - 단일 인스턴스 브로커
- 3 - 클러스터 브로커

다른 모드 값이 유효하지 않고 API가 오류를 반환합니다.

다음 예제에서는 두 가지 인스턴스 유형에 대한 섹션이 있는 구성 데이터를 보여줍니다.

```
connection_max = 1000

{{m7g.large.3}}
connection_max = 2000
{{m7g.large.3}}

{{m7g.xlarge.3}}
connection_max = 4000
{{m7g.xlarge.3}}
```

섹션에서 허용되는 구성 키

섹션 내에서는 다음 리소스 제한 구성 키만 지원됩니다. 섹션 내에 다른 구성 키를 추가하면 API 오류가 발생합니다.

- max_message_size
- channel_max_per_node
- connection_max
- cluster_queue_limit
- vhost_max
- consumer_max_per_channel
- runtime_parameters.limits.shovel

- `cluster_exchange_limit`

섹션 우선 순위 규칙

일반(상위 수준) 섹션과 인스턴스별 섹션 모두에 구성 키가 나타나면 구성 데이터의 뒷부분에 나타나는 값이 우선합니다. 예를 들어 `m7g.large` 클러스터 브로커에 다음 구성을 적용하면가 `connection_max`로 설정됩니다`2000`.

```
connection_max = 1000

{{m7g.large.3}}
connection_max = 2000
{{m7g.large.3}}
```

일반 값이 마지막으로 오기 `1000`때문에 주문 세트를 `connection_max`로 되돌립니다.

```
{{m7g.large.3}}
connection_max = 2000
{{m7g.large.3}}

connection_max = 1000
```

> **Note**
>
> 구성 데이터가 특정 인스턴스 유형에 대한 값을 정의하지 않는 경우 Amazon MQ는 기본값을 적용합니다.

예시

다음 예제에서는 섹션을 사용하여 구성을 생성하고를 사용하여 브로커와 연결하는 방법을 보여줍니다 AWS CLI.

섹션으로 구성을 업데이트하려면

다음 명령을 실행하여 여러 인스턴스 유형에 대한 인스턴스별 리소스 제한으로 구성을 업데이트합니다.

```
aws mq update-configuration \
    --configuration-id <config-id> \
    --data "$(echo -e "connection_max = 1000\nchannel_max_per_node =
 64\n\n{{m7g.large.3}}\nconnection_max = 2000\nchannel_max_per_node =
 128\n{{m7g.large.3}}\n\n{{m7g.xlarge.3}}\nconnection_max = 4000\nchannel_max_per_node
 = 256\n{{m7g.xlarge.3}}" | base64 --wrap=0)"
```

이 구성은 다음 값을 정의합니다.

- 일반 기본값: `connection_max = 1000` 및 `channel_max_per_node = 64`
- `m7g.large` 클러스터 브로커: `connection_max = 2000` 및 `channel_max_per_node = 128`
- `m7g.xlarge` 클러스터 브로커: `connection_max = 4000` 및 `channel_max_per_node = 256`

구성을 브로커와 연결하려면

구성을 업데이트한 후 브로커와 연결하고 브로커를 재부팅하여 변경 사항을 적용합니다. 다음 명령을 실행합니다.

```
aws mq update-broker \
    --broker-id <broker-id> \
    --configuration id=<config-id>,revision=<revision-number>
```

리소스 제한 재정의 오류

지원되는 범위를 벗어나는 구성 값과 브로커를 연결하거나 생성하면 다음과 유사한 오류 응답이 발생합니다.

```
Configuration Revision N for configuration:cluster_queue_limit has limit: of value:
 100000000 larger than maximum allowed limit:5000
```

인스턴스 유형 및 배포 모드별 기본값 및 지원되는 최대 범위는 the section called “기본 리소스 제한” 및 섹션을 참조하세요the section called “최대 리소스 제한”.

RabbitMQ 구성의 ARN 지원

RabbitMQ용 Amazon MQ는 일부 RabbitMQ 구성 설정의 값에 대해 AWS ARNs을 지원합니다. 이는 RabbitMQ 커뮤니티 플러그인 rabbitmq-aws에 의해 활성화됩니다. 이 플러그인은 Amazon MQ에서 개발 및 유지 관리하며 Amazon MQ에서 관리하지 않는 자체 호스팅 RabbitMQ Amazon MQ 브로커에서도 사용할 수 있습니다.

⚠ 중요 고려 사항

- aws 플러그인에서 검색한 확인된 ARN 값은 런타임 시 RabbitMQ 프로세스로 직접 전달됩니다. RabbitMQ 노드의 다른 위치에 저장되지 않습니다.
- RabbitMQ용 Amazon MQ에는 구성된 ARNs에 액세스하기 위해 Amazon MQ가 수임할 수 있는 IAM 역할이 필요합니다. 이는를 설정하여 구성됩니다`aws.arns.assume_role_arn`.
- IAM 역할이 포함된 브로커 구성으로 CreateBroker 또는 UpdateBroker APIs를 호출하는 사용자는 해당 역할에 대한 `iam:PassRole` 권한이 있어야 합니다.
- IAM 역할은 RabbitMQ 브로커와 동일한 AWS 계정에 있어야 합니다. 구성의 모든 ARNs은 RabbitMQ 브로커와 동일한 AWS 리전에 있어야 합니다.
- Amazon MQ는 IAM 역할을 수임할 `aws:SourceArn` 때 IAM 전역 조건 키 `aws:SourceAccount` 및를 추가합니다. 이러한 값은 혼동된 대리자 보호를 위해 역할에 연결된 IAM 정책에 사용해야 합니다.

이 페이지의 내용

지원되는 키

필수 IAM 역할

`aws.arns.assume_role_arn`

Amazon MQ가 다른 AWS 리소스에 액세스하기 위해 수임하는 IAM 역할 ARN입니다. 다른 ARN 구성을 사용할 때 필요합니다.

AMQP 엔드포인트

구성 키	설명
`aws.arns.ssl_options.cacertfile`	SSL/TLS 클라이언트 연결을 위한 인증 기관 파일입니다. Amazon MQ에서 인증서를 저장하려면 Amazon S3 또는를 사용해야 합니다.

RabbitMQ 관리 플러그인

구성 키	설명
`aws.arns.management.ssl.cacertfile`	관리 인터페이스 SSL/TLS 연결을 위한 인증 기관 파일입니다. Amazon MQ에서 인증서를 저장하려면 Amazon S3 또는를 사용해야 합니다.

RabbitMQ OAuth 2.0 플러그인

구성 키	설명
`aws.arns.auth_oauth2.https.cacertfile`	OAuth 2.0 HTTPS 연결을 위한 인증 기관 파일입니다. Amazon MQ에서 인증서를 저장하려면 Amazon S3 또는를 사용해야 합니다.

RabbitMQ HTTP 인증 플러그인

구성 키	설명
`aws.arns.auth_http.ssl_options.cacertfile`	HTTP 인증 SSL/TLS 연결을 위한 인증 기관 파일입니다. Amazon MQ에서 인증서를 저장하려면 Amazon S3 또는를 사용해야 합니다.
`aws.arns.auth_http.ssl_options.certfile`	Amazon MQ와 HTTP 인증 서버 간의 상호 TLS 연결을 위한 인증서 파일입니다. Amazon MQ에서 인증서를 저장하려면 Amazon S3 또는를 사용해야 합니다.
`aws.arns.auth_http.ssl_options.keyfile`	Amazon MQ와 HTTP 인증 서버 간의 상호 TLS 연결을 위한 프라이빗 키 파일입니다. Amazon MQ에서는를 사용하여 프라이빗 키를 저장 AWS Secrets Manager 해야 합니다.

RabbitMQ LDAP 플러그인

구성 키	설명
`aws.arns.auth_ldap.ssl_options.cacertfile`	LDAP SSL/TLS 연결을 위한 인증 기관 파일입니다. Amazon MQ에서 인증서를 저장하려면 Amazon S3 또는를 사용해야 합니다.
`aws.arns.auth_ldap.ssl_options.certfile`	Amazon MQ와 LDAP 서버 간의 상호 TLS 연결을 위한 인증서 파일입니다. Amazon MQ에서 인증서를 저장하려면 Amazon S3 또는를 사용해야 합니다.
`aws.arns.auth_ldap.ssl_options.keyfile`	Amazon MQ와 LDAP 서버 간의 상호 TLS 연결을 위한 프라이빗 키 파일입니다. Amazon MQ에서는를 사용하여 프라이빗 키를 저장 AWS Secrets Manager 해야 합니다.
`aws.arns.auth_ldap.dn_lookup_bind.password`	LDAP DN 조회 바인드의 암호입니다. Amazon MQ에서는를 사용하여 암호를 일반 텍스트 값으로 저장 AWS Secrets Manager 해야 합니다.

구성 키	설명
`aws.arns.auth_ldap.other_bind.password`	LDAP 기타 바인드의 암호입니다. Amazon MQ에서는를 사용하여 암호를 일반 텍스트 값으로 저장 AWS Secrets Manager 해야 합니다.

IAM 정책 샘플

수임 역할 정책 문서 및 역할 정책 문서를 포함한 IAM 정책 예제는 CDK 샘플 구현을 참조하세요.

AWS Secrets Manager 및 Amazon S3 리소스를 설정하는 방법에 대한 단계는 LDAP 인증 및 권한 부여 사용 섹션을 참조하세요.

액세스 검증

ARN 값을 가져올 수 없는 시나리오의 문제를 해결하기 위해 aws 플러그인은 Amazon MQ가 역할을 성공적으로 수임하고 ARN을 해결할 수 있는지 확인하기 위해 호출할 수 있는 RabbitMQ 관리 API 엔드포인트를 지원합니다. Amazon MQ AWS ARNs 따라서 브로커 구성을 업데이트하고, 브로커를 새 구성 개정으로 업데이트하고, 브로커를 재부팅하여 구성 변경을 테스트할 필요가 없습니다.

> **Note**
>
> 이 API를 사용하려면 기존 RabbitMQ 관리자 사용자가 필요합니다. Amazon MQ는 다른 액세스 방법 외에도 내부 사용자를 사용하여 테스트 브로커를 생성할 것을 권장합니다. OAuth 2.0 및 단순(내부) 인증 활성화를 참조하세요. 그런 다음이 사용자를 사용하여 검증 API에 액세스할 수 있습니다.

> **Note**
>
> aws 플러그인은 새 역할을 검증 API에 대한 입력으로 전달하는 것을 지원하지만 이 파라미터는 Amazon MQ에서 지원되지 않습니다. 검증에 사용되는 IAM 역할은 `aws.arns.assume_role_arn`브로커 구성의 값과 일치해야 합니다.

관련 브로커 격리 상태

ARN 지원 문제와 관련된 브로커 격리 상태에 대한 자세한 내용은 다음을 참조하세요.

- RABBITMQ_INVALID_ASSUMEROLE
- RABBITMQ_INVALID_ARN_LDAP
- RABBITMQ_INVALID_ARN

예제 시나리오

- 브로커b-f0fc695e-2f9c-486b-845a-988023a3e55b가 IAM 역할을 사용하여 AWS Secrets Manager 보안 암호<role>에 액세스하도록 구성되었습니다. <arn>
- Amazon MQ에 제공된 역할에 AWS Secrets Manager 보안 암호에 대한 읽기 권한이 없는 경우 RabbitMQ 로그에 다음 오류가 표시됩니다.

```
[error] <0.254.0> aws_arn_config: {handle_assume_role,{error,{assume_role_failed,"AWS
 service is unavailable"}}}
```

  또한 브로커는 INVALID_ASSUMEROLE 격리 상태로 전환됩니다. 자세한 내용은 INVALID_ASSUMEROLE을 참조하세요.
- LDAP 인증 시도가 실패하고 다음 오류가 발생합니다.

```
[error] <0.254.0> LDAP bind failed: invalid_credentials
```

- 적절한 권한으로 IAM 역할 수정
- 검증 엔드포인트를 호출하여 RabbitMQ가 이제 보안 암호에 액세스할 수 있는지 확인합니다.

```
curl -4su 'guest:guest' -XPUT -H 'content-type: application/json' <broker-endpoint>/
api/aws/arn/validate -d '{"assume_role_arn":"arn:aws:iam::<account-id>:role/<role-
name>","arns":["arn:aws:secretsmanager:<region>:<account-id>:secret:<secret-name>"]}'
 | jq '.'
```

AMQP 클라이언트 SSL 구성

페더레이션 및 셔블은 업스트림 브로커와 다운스트림 브로커 간의 통신에 AMQP를 사용합니다. 기본적으로 TLS 피어 확인은 RabbitMQ 4용 Amazon MQ 클라이언트에 대해 활성화됩니다. 이 설정을 사용하면 Amazon MQ 브로커에서 실행되는 페더레이션 및 셔블 AMQP 클라이언트가 업스트림 브로커와 연결을 설정할 때 피어 확인을 수행합니다.

Amazon MQ 브로커에서 실행되는 AMQP 클라이언트는 Mozilla와 동일한 인증 기관을 지원합니다. ACM을 사용하지 않는 경우 Mozilla 포함 CA 인증서 목록에서 CA가 발급한 인증서를 사용합니다. 온프

레미스 브로커가 다른 인증 기관의 인증서를 사용하는 경우 SSL 확인이 실패합니다. 이러한 사용 사례에 대해 TLS 피어 확인을 비활성화할 수 있습니다.

> **Important**
>
> Amazon MQ는 현재 AMQP 클라이언트 연결을 위한 클라이언트 인증서 구성을 지원하지 않습니다. 따라서 페더레이션 및 셔블은 클라이언트 인증서 인증이 필요한 mTLS 지원 브로커에 연결할 수 없습니다.

> **Important**
>
> RabbitMQ용 Amazon MQ에서 AMQP 클라이언트의 SSL 속성은 RabbitMQ 기본값(verify_none)으로 구성됩니다. RabbitMQ 3용 Amazon MQ는 이러한 기본값 재정의를 지원하지 않습니다. RabbitMQ

> **Note**
>
> 기본 `verify_peer` 설정을 사용하면 2개의 Amazon MQ 브로커 간에 페더레이션 및 셔블 연결을 설정할 수 있지만, 이는 Amazon MQ 브로커와 프라이빗 브로커 또는 비 Amazon MQ CA 인증서로 실행되는 온프레미스 브로커 간의 연결 설정을 지원하지 않습니다. 프라이빗 또는 온프레미스 브로커에 연결하려면 다운스트림 Amazon MQ 브로커에서 피어 확인을 비활성화해야 합니다.

AMQP 클라이언트 SSL 구성 키

구성	구성 키	지원되는 값
AMQP 클라이언트 SSL 피어 확인	`amqp_client.ssl_options.verify`	`verify_none` , `verify_peer`

AMQP 클라이언트 SSL 피어 확인을 재정의하는 방법

RabbitMQ 4 브로커의 Amazon MQ API 및 Amazon MQ 콘솔을 사용하여 AMQP 클라이언트 SSL 피어 확인을 재정의할 수 있습니다.

다음 예제에서는를 사용하여 AMQP 클라이언트 SSL 피어 확인을 재정의하는 방법을 보여줍니다 AWS CLI.

```
aws mq update-configuration --configuration-id <config-id> --data "$(echo
 "amqp_client.ssl_options.verify=verify_none" | base64 --wrap=0)"
```

호출이 성공하면 구성 개정이 생성됩니다. 구성을 RabbitMQ 브로커에 연결하고 브로커를 재부팅하여 재정의를 적용해야 합니다. 자세한 내용은 섹션을 참조하세요. Creating and applying broker configurations

> **Important**
>
> 를 사용할 때 `verify_none`SSL 암호화는 여전히 활성 상태이지만 피어의 자격 증명은 확인되지 않습니다. 필요한 경우에만이 설정을 사용하고 대상 브로커에 대한 네트워크 경로를 신뢰해야 합니다.

RabbitMQ 인증 및 권한 부여를 위한 Amazon MQ RabbitMQ

RabbitMQ용 Amazon MQ는 다음과 같은 인증 및 권한 부여 방법을 지원합니다.

간편한 인증 및 권한 부여

이 방법에서 브로커 사용자는 RabbitMQ 브로커에 내부적으로 저장되고 웹 콘솔 또는 관리 API를 통해 관리됩니다. vhost, Exchange, 대기열 및 주제에 대한 권한은 RabbitMQ에서 직접 구성됩니다. 기본 메서드입니다. 자세한 내용은 단순 인증 및 권한 부여를 참조하세요.

OAuth 2.0 인증 및 권한 부여

이 메서드에서 브로커 사용자와 해당 권한은 외부 OAuth 2.0 ID 제공업체(IdP)에서 관리합니다. vhost, Exchange, 대기열 및 주제에 대한 사용자 인증 및 리소스 권한은 OAuth 2.0 공급자의 범위 시스템을 통해 중앙 집중화됩니다. 이를 통해 사용자 관리를 간소화하고 기존 자격 증명 시스템과 통합할 수 있습니다. 자세한 내용은 OAuth 2.0 인증 및 권한 부여를 참조하세요.

IAM 인증 및 권한 부여

이 방법에서 브로커 사용자는 AWS IAM 아웃바운드 페더레이션을 통해 IAM 자격 증명을 사용하여 인증합니다. IAM 자격 증명은 AWS Security Token Service(STS)에서 JWT 토큰을 얻는 데 사용되며, 이

러한 JWT 토큰은 인증을 위한 OAuth 2.0 토큰 역할을 합니다. 이 방법은 RabbitMQ용 Amazon MQ에서 기존 OAuth 2.0 지원을 활용합니다. 여기서 AWS 는 OAuth 2.0 자격 증명 공급자 역할을 합니다. 사용자 인증은 AWS IAM에서 처리되는 반면, vhost, exchange, queue 및 topics에 대한 리소스 권한은 RabbitMQ에 구성된 IAM 정책 및 범위 별칭을 통해 관리됩니다. 자세한 내용은 IAM 인증 및 권한 부여를 참조하세요.

LDAP 인증 및 권한 부여

이 방법에서는 브로커 사용자와 해당 권한이 외부 LDAP 디렉터리 서비스에 의해 관리됩니다. 사용자 인증 및 리소스 권한은 LDAP 서버를 통해 중앙 집중화되므로 사용자는 기존 디렉터리 서비스 자격 증명을 사용하여 RabbitMQ에 액세스할 수 있습니다. 자세한 내용은 LDAP 인증 및 권한 부여를 참조하세요.

HTTP 인증 및 권한 부여

이 방법에서는 브로커 사용자와 해당 권한이 외부 HTTP 서버에서 관리됩니다. 사용자 인증 및 리소스 권한은 HTTP 서버를 통해 중앙 집중화되므로 사용자는 자체 인증 및 권한 부여 공급자를 사용하여 RabbitMQ에 액세스할 수 있습니다. 이 방법에 대한 자세한 내용은 HTTP 인증 및 권한 부여를 참조하세요.

SSL 인증서 인증

Amazon MQ는 RabbitMQ 브로커에 대해 상호 TLS(mTLS)를 지원합니다. SSL 인증 플러그인은 mTLS 연결의 클라이언트 인증서를 사용하여 사용자를 인증합니다. 이 방법에서 브로커 사용자는 사용자 이름 및 암호 자격 증명 대신 X.509 클라이언트 인증서를 사용하여 인증됩니다. 클라이언트의 인증서는 신뢰할 수 있는 인증 기관(CA)에 대해 검증되며, 사용자 이름은 일반 이름(CN) 또는 주체 대체 이름(SAN)과 같은 인증서의 필드에서 추출됩니다. 이 방법은 네트워크를 통해 자격 증명을 전송하지 않고 강력한 인증을 제공합니다. 자세한 내용은 SSL 인증서 인증을 참조하세요.

> **Note**
>
> RabbitMQ는 동시에 사용할 수 있는 여러 인증 및 권한 부여 방법을 지원합니다. 예를 들어 OAuth 2.0과 단순(내부) 인증을 모두 활성화할 수 있습니다. 자세한 내용은 OAuth 2.0 및 단순(내부) 인증을 모두 활성화하는 방법에 대한 OAuth 2.0 자습서 섹션과 RabbitMQ 액세스 제어 설명서를 참조하세요.
>
> Amazon MQ는 인증 구성을 테스트할 때 내부 사용자를 생성할 것을 권장합니다. 이렇게 하면 RabbitMQ 관리 API를 사용하여 액세스 구성을 검증할 수 있습니다. 자세한 내용은 액세스 검증을 참조하세요.

간편한 인증 및 권한 부여

RabbitMQ용 Amazon MQ 브로커 사용자

> **Note**
>
> 이 주제에서는 RabbitMQ의 기본 내부 인증 및 권한 부여 메커니즘을 사용하여 브로커 사용자를 관리하는 방법을 설명합니다. 지원되는 모든 인증 및 권한 부여 방법에 대한 자세한 내용은 RabbitMQ 인증 및 권한 부여를 위한 Amazon MQ RabbitMQ를 참조하세요.

모든 AMQP 0-9-1 클라이언트 연결에는 연결된 사용자가 있습니다. 이 사용자는 인증되어야 합니다. 각 클라이언트 연결은 가상 호스트(vhost)도 대상으로 합니다. 사용자에게이 vhost에 대한 권한 집합이 있어야 합니다. 사용자는 vhost의 대기열 및 교환에 대해 구성, 쓰기 및 읽기 권한이 있을 수 있습니다. 연결이 설정되면 사용자 자격 증명과 대상 vhost를 지정합니다.

RabbitMQ용 Amazon MQ 브로커를 처음 생성할 때 Amazon MQ는 사용자가 제공하는 로그인 보안 인증 정보를 사용하여 `administrator` 태그가 있는 RabbitMQ 사용자를 생성합니다. 그런 다음 RabbitMQ 관리 API 또는 RabbitMQ 웹 콘솔을 통해 사용자를 추가하고 관리할 수 있습니다. RabbitMQ 웹 콘솔 또는 관리 API를 사용하여 사용자 권한 및 태그를 설정하거나 수정할 수도 있습니다.

> **Note**
>
> RabbitMQ 사용자는 Amazon MQ 사용자 API를 통해 저장되거나 표시되지 않습니다.

> **Important**
>
> RabbitMQ용 Amazon MQ에서는 사용자 이름으로 "guest"를 지원하지 않으므로 새 브로커를 생성하면 기본 게스트 계정이 삭제됩니다. Amazon MQ는 또한 고객이 생성한 "guest"라는 계정도 주기적으로 삭제합니다.

RabbitMQ 관리 API를 사용하여 새 사용자를 생성하려면 다음 API 엔드포인트 및 요청 본문을 사용합니다. *### ##*과 *####*를 새 로그인 보안 인증 정보로 바꾸세요.

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

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

Important

- 개인 식별 정보(PII)나 기타 기밀 정보 또는 민감한 정보를 브로커 사용자 이름에 추가하지 마십시오. 브로커 사용자 이름은 CloudWatch Logs를 포함한 다른 AWS 서비스에서 액세스할 수 있습니다. 브로커 사용자 이름은 개인 데이터나 민감한 데이터에 사용하기 위한 것이 아닙니다.
- 모든 관리자 계정에 대한 액세스 권한을 잃은 경우 복구를 위해 IAM 인증을 사용하도록 브로커 액세스 복구를 참조하세요.

`tags` 키는 필수이며 사용자에 대한 쉼표로 구분된 태그 목록입니다. Amazon MQ는 `administrator`, `management`, `monitoring` 및 `policymaker` 사용자 태그를 지원합니다.

다음 API 끝점 및 요청 본문을 사용하여 개별 사용자의 권한을 설정할 수 있습니다. *`vhost`*와 *`username`*을 사용자 정보로 바꿉니다. 기본 vhost `/`의 경우 `%2F`를 사용합니다.

```
PUT /api/permissions/vhost/username HTTP/1.1

    {"configure":".*","write":".*","read":".*"}
```

Note

`configure`, `read` 및 `write` 키는 모두 필수입니다.

와일드카드 `.*` 값을 사용하면 이 작업으로 지정된 vhost의 모든 대기열에 대한 읽기, 쓰기 및 구성 권한이 사용자에게 부여됩니다. RabbitMQ 관리 API를 통한 사용자 관리에 대한 자세한 내용은 Amazon RabbitMQ 관리 HTTP API를 참조하세요.

RabbitMQ용 Amazon MQ에 대한 OAuth 2.0 인증 및 권한 부여

RabbitMQ용 Amazon MQ는 여러 인증 및 권한 부여 방법을 지원합니다. 지원되는 모든 메서드에 대한 자세한 내용은 RabbitMQ용 Amazon MQ 브로커에 대한 인증 및 권한 부여를 참조하세요.

OAuth 2.0 인증 및 권한 부여에서 브로커 사용자와 해당 권한은 외부 OAuth 2.0 ID 제공업체(IdP)에서 관리합니다. vhost, Exchange, 대기열 및 주제에 대한 사용자 인증 및 리소스 권한은 OAuth 2.0 공급자

의 범위 시스템을 통해 중앙 집중화됩니다. 이를 통해 사용자 관리를 간소화하고 기존 자격 증명 시스템과 통합할 수 있습니다.

⚠ 중요 고려 사항

- Amazon MQ for ActiveMQ 브로커에서는 OAuth 2.0 통합이 지원되지 않습니다.
- Amazon MQ for RabbitMQ는 프라이빗 CA에서 발급한 서버 인증서를 지원하지 않습니다.
- RabbitMQ OAuth 2.0 플러그인은 토큰 내부 검사 엔드포인트 및 불투명 액세스 토큰을 지원하지 않습니다. 또한 토큰 해지 검사를 수행하지 않습니다.
- 기존 브로커에서 OAuth 2.0을 활성화하려면 IAM 권한 `mq:UpdateBrokerAccessConfiguration`을 포함해야 합니다.
- Amazon MQ는 모니터링 전용 권한이 있는 `monitoring-AWS-OWNED-DO-NOT-DELETE`라는 시스템 사용자를 자동으로 생성합니다. 이 사용자는 OAuth 2.0 지원 브로커에서도 RabbitMQ의 내부 인증 시스템을 사용하며 루프백 인터페이스 액세스로만 제한됩니다.

RabbitMQ용 Amazon MQ 브로커에 OAuth 2.0을 구성하는 방법에 대한 자세한 내용은 섹션을 참조하세요OAuth 2.0 인증 및 권한 부여 사용.

이 페이지의 내용

지원되는 OAuth 2.0 구성

RabbitMQ용 Amazon MQ는 다음을 제외하고 RabbitMQ OAuth 2.0 플러그인에서 구성 가능한 모든 변수를 지원합니다.

- `auth_oauth2.https.cacertfile`
- `auth_oauth2.oauth_providers.{id/index}.https.cacertfile`
- `management.oauth_client_secret`

  Amazon MQ는 이 키를 지원하지 않으므로 UAA를 IdP로 지원하지 않습니다.
- `management.oauth_resource_servers.{id/index}.oauth_client_secret`
- `auth_oauth2.signing_keys.{id/index}`

OAuth 2.0 인증에 대한 추가 검증

또한 Amazon MQ는 OAuth 2.0 인증에 대해 다음과 같은 추가 검증을 적용합니다.

- 모든 URL은 `https://`로 시작해야 합니다.
- 지원되는 서명 알고리즘: `Ed25519`, `Ed25519ph`, `Ed448`, `Ed448ph`, `EdDSA`, `ES256K`, `ES256`, `ES384`, `ES512`, `HS256`, `HS384`, `HS512`, `PS256`, `PS384`, `PS512`, `RS256`, `RS384` 및 `RS512`.

RabbitMQ용 Amazon MQ에 대한 IAM 인증 및 권한 부여

RabbitMQ용 Amazon MQ는 여러 인증 및 권한 부여 방법을 지원합니다. 지원되는 모든 메서드에 대한 자세한 내용은 RabbitMQ용 Amazon MQ 브로커에 대한 인증 및 권한 부여를 참조하세요.

IAM 인증 및 권한 부여를 통해 브로커 사용자는 AWS IAM 아웃바운드 페더레이션을 통해 IAM 자격 증명을 사용하여 인증할 수 있습니다. 이 방법에서는 IAM 자격 증명을 사용하여 AWS Security Token Service(STS)에서 JWT 토큰을 가져옵니다. 이러한 JWT 토큰은 인증을 위한 OAuth 2.0 토큰 역할을 하며, RabbitMQ용 Amazon MQ에서 기존 OAuth 2.0 지원을 활용합니다. 여기서는 OAuth 2.0 자격 증명 공급자 역할을 AWS 합니다. AWS IAM은 사용자 인증을 처리하는 반면 가상 호스트, 교환, 대기열 및 주제에 대한 리소스 권한은 RabbitMQ에 구성된 IAM 정책 및 범위 별칭을 통해 관리됩니다.

⚠ 중요 고려 사항

- IAM 인증은 RabbitMQ 버전 3.13, 4.2 이상에서 지원됩니다. ActiveMQ용 Amazon MQ 브로커에서는 지원되지 않습니다.
- IAM 인증을 사용하려면 AWS 계정에서 IAM 아웃바운드 페더레이션을 구성하고 사용할 수 있어야 합니다.
- 이 방법은 RabbitMQ용 Amazon MQ의 기존 OAuth 2.0 인프라를 기반으로 하며 OAuth 2.0 자격 증명 공급자 역할을 AWS 합니다.
- Amazon MQ는 모니터링 전용 권한이 있는 `monitoring-AWS-OWNED-DO-NOT-DELETE`라는 시스템 사용자를 자동으로 생성합니다. 이 사용자는 IAM 지원 브로커에서도 RabbitMQ의 내부 인증 시스템을 사용하며 루프백 인터페이스 액세스로만 제한됩니다.

이 페이지의 내용

IAM 인증 작동 방식

RabbitMQ용 Amazon MQ에 대한 IAM 인증은 IAM 아웃바운드 페더레이션을 사용하여 AWS IAM 자격 증명이 RabbitMQ 브로커로 인증할 수 있도록 합니다. IAM 자격 증명은 AWS Security Token Service(STS)에서 JWT 토큰을 얻는 데 사용되며, 이러한 JWT 토큰은 RabbitMQ 브로커를 사용한 인증을 위한 OAuth 2.0 토큰 역할을 합니다.

제한 사항

RabbitMQ용 Amazon MQ에 대한 IAM 인증에는 다음과 같은 제한이 있습니다.

- 범위 클레임 구성 - STS의 JWT 토큰이 중첩되어 있으므로 범위 클레임을 직접 사용할 수 없습니다. 키는 이며`sts.amazonaws.com`, RabbitMQ 구성에서 범위 별칭을 사용하여 IAM 역할을 RabbitMQ 권한에 매핑해야 합니다. 또한이 제한으로 인해 권한 부여에 IAM 정책을 완전히 사용할 수 없으므로 대신 권한 부여에 RabbitMQ 구성이 필요합니다.

RabbitMQ용 Amazon MQ 브로커에 대한 IAM 인증 및 권한 부여를 구성하는 방법에 대한 자세한 내용은 섹션을 참조하세요IAM 인증 및 권한 부여 사용.

RabbitMQ용 Amazon MQ에 대한 HTTP 인증 및 권한 부여

RabbitMQ용 Amazon MQ는 외부 HTTP 서버를 사용하는 브로커 사용자의 인증 및 권한 부여를 지원합니다. 지원되는 다른 방법은 RabbitMQ용 Amazon MQ 브로커에 대한 인증 및 권한 부여를 참조하세요.

> **Note**
>
> HTTP 인증 플러그인은 RabbitMQ용 Amazon MQ 버전 4 이상에서만 사용할 수 있습니다. RabbitMQ

> **중요 고려 사항**
>
> - HTTP 서버는 퍼블릭 인터넷을 통해 액세스할 수 있어야 합니다. RabbitMQ용 Amazon MQ는 상호 TLS를 사용하여 HTTP 서버에 인증하도록 구성할 수 있습니다.
> - RabbitMQ용 Amazon MQ는 로컬 파일 시스템에 액세스해야 하는 설정에 AWS ARNs 사용을 적용합니다. 자세한 내용은 RabbitMQ 구성의 ARN 지원을 참조하세요.

- 기존 브로커에서 HTTP 인증을 활성화`mq:UpdateBrokerAccessConfiguration`하려면 IAM 권한를 포함해야 합니다.
- Amazon MQ는 모니터링 전용 권한이 있는 `monitoring-AWS-OWNED-DO-NOT-DELETE`라는 시스템 사용자를 자동으로 생성합니다. 이 사용자는 HTTP 지원 브로커에서도 RabbitMQ의 내부 인증 시스템을 사용하며 루프백 인터페이스 액세스로만 제한됩니다. Amazon MQ는 보호된 사용자 태그를 추가하여이 사용자의 삭제를 방지합니다.

RabbitMQ용 Amazon MQ 브로커에 대한 HTTP 인증을 구성하는 방법에 대한 자세한 내용은 섹션을 참조하세요HTTP 인증 및 권한 부여 사용.

이 페이지의 내용

지원되는 HTTP 구성

RabbitMQ용 Amazon MQ는 AWS ARNs이 필요한 다음 예외를 제외하고 RabbitMQ HTTP 인증 플러그인에서 구성 가능한 모든 변수를 지원합니다. RabbitMQ ARN 지원에 대한 자세한 내용은 RabbitMQ 구성의 ARN 지원을 참조하세요.

ARNs이 필요한 구성

`auth_http.ssl_options.cacertfile`

대신 `aws.arns.auth_http.ssl_options.cacertfile` 사용

`auth_http.ssl_options.certfile`

대신 `aws.arns.auth_http.ssl_options.certfile` 사용

`auth_http.ssl_options.keyfile`

대신 `aws.arns.auth_http.ssl_options.keyfile` 사용

지원되지 않는 SSL 옵션

다음 SSL 구성 옵션도 지원되지 않습니다.

전체 목록 보기

- auth_http.ssl_options.cert
- auth_http.ssl_options.client_renegotiation
- auth_http.ssl_options.dh
- auth_http.ssl_options.dhfile
- auth_http.ssl_options.honor_cipher_order
- auth_http.ssl_options.honor_ecc_order
- auth_http.ssl_options.key.RSAPrivateKey
- auth_http.ssl_options.key.DSAPrivateKey
- auth_http.ssl_options.key.PrivateKeyInfo
- auth_http.ssl_options.log_alert
- auth_http.ssl_options.password
- auth_http.ssl_options.psk_identity
- auth_http.ssl_options.reuse_sessions
- auth_http.ssl_options.secure_renegotiate
- auth_http.ssl_options.versions.$version
- auth_http.ssl_options.sni
- auth_http.ssl_options.crl_check

Amazon MQ의 HTTP 구성에 대한 추가 검증

또한 Amazon MQ는 HTTP 인증 및 권한 부여에 대해 다음과 같은 추가 검증을 적용합니다.

- auth_http.http_method는 get 또는 중 하나여야 합니다. post
- 다음 경로 구성은 HTTPS URLs 사용해야 합니다.
  - auth_http.user_path
  - auth_http.vhost_path
  - auth_http.resource_path
  - auth_http.topic_path
- AWS ARN을 사용해야 하는 설정이 있는 경우를 제공해야 aws.arns.assume_role_arn 합니다.

RabbitMQ용 Amazon MQ에 대한 SSL 인증서 인증

RabbitMQ용 Amazon MQ는 X.509 클라이언트 인증서를 사용하여 브로커 사용자의 인증을 지원합니다. 지원되는 다른 방법은 RabbitMQ용 Amazon MQ 브로커에 대한 인증 및 권한 부여를 참조하세요.

> **Note**
>
> SSL 인증서 인증 플러그인은 RabbitMQ용 Amazon MQ 버전 4 이상에서만 사용할 수 있습니다. RabbitMQ

> **중요 고려 사항**
>
> - 클라이언트 인증서는 신뢰할 수 있는 인증 기관(CA)에서 서명해야 합니다. RabbitMQ용 Amazon MQ는 인증 중에 인증서 체인을 검증합니다.
> - RabbitMQ용 Amazon MQ는 CA 인증서와 같은 인증서 관련 설정과 로컬 파일 시스템에 액세스해야 하는 설정에 AWS ARNs 사용을 적용합니다. 자세한 내용은 RabbitMQ 구성의 ARN 지원을 참조하세요.
> - Amazon MQ는 모니터링 전용 권한이 있는 `monitoring-AWS-OWNED-DO-NOT-DELETE`라는 시스템 사용자를 자동으로 생성합니다. 이 사용자는 SSL 인증서가 활성화된 브로커에서도 RabbitMQ의 내부 인증 시스템을 사용하며 루프백 인터페이스 액세스로만 제한됩니다. Amazon MQ는 보호된 사용자 태그를 추가하여이 사용자의 삭제를 방지합니다.

RabbitMQ용 Amazon MQ 브로커에 대한 SSL 인증서 인증을 구성하는 방법에 대한 자세한 내용은 섹션을 참조하세요SSL 인증서 인증 사용.

이 페이지의 내용

지원되는 SSL 구성

RabbitMQ용 Amazon MQ는 클라이언트 연결을 위한 SSL/TLS 구성을 지원합니다. ARN 지원에 대한 자세한 내용은 RabbitMQ 구성의 ARN 지원을 참조하세요.

ARNs이 필요한 구성

`ssl_options.cacertfile`

대신 `aws.arns.ssl_options.cacertfile` 사용

SSL 인증서 로그인 구성

다음 구성은 클라이언트 인증서에서 사용자 이름을 추출하는 방법을 제어합니다.

`ssl_cert_login_from`

사용자 이름 추출에 사용할 인증서 필드를 지정합니다. 지원되는 값:

- `distinguished_name` - 전체 고유 이름 사용
- `common_name` - 일반 이름(CN) 필드 사용
- `subject_alternative_name` 또는 `subject_alt_name` - 주체 대체 이름 사용

`ssl_cert_login_san_type`

주체 대체 이름을 사용할 때는 SAN 유형을 지정합니다. 지원되는 값: `dns`, `ip`, `email`, `uri`, `other_name`

`ssl_cert_login_san_index`

주체 대체 이름을 사용할 때 사용할 SAN 항목의 인덱스를 지정합니다(0 기반). 음수가 아닌 정수여야 합니다.

클라이언트 연결을 위한 SSL 옵션

클라이언트 연결에는 다음 SSL 옵션이 적용됩니다.

`ssl_options.verify`

피어 확인 모드. 지원되는 값: `verify_none`, `verify_peer`

`ssl_options.fail_if_no_peer_cert`

클라이언트가 인증서를 제공하지 않는 경우 연결을 거부할지 여부입니다. 부울 값입니다.

`ssl_options.depth`

확인을 위한 최대 인증서 체인 깊이입니다.

`ssl_options.hostname_verification`

호스트 이름 확인 모드. 지원되는 값: `wildcard`, `none`

지원되지 않는 SSL 옵션

다음 SSL 구성 옵션은 지원되지 않습니다.

전체 목록 보기

- `ssl_options.cert`
- `ssl_options.client_renegotiation`
- `ssl_options.dh`
- `ssl_options.dhfile`
- `ssl_options.honor_cipher_order`
- `ssl_options.honor_ecc_order`
- `ssl_options.key.RSAPrivateKey`
- `ssl_options.key.DSAPrivateKey`
- `ssl_options.key.PrivateKeyInfo`
- `ssl_options.log_alert`
- `ssl_options.password`
- `ssl_options.psk_identity`
- `ssl_options.reuse_sessions`
- `ssl_options.secure_renegotiate`
- `ssl_options.versions.$version`
- `ssl_options.sni`
- `ssl_options.crl_check`

Amazon MQ의 SSL 구성에 대한 추가 검증

또한 Amazon MQ는 SSL 인증서 인증에 대해 다음과 같은 추가 검증을 적용합니다.

- AWS ARN을 사용해야 하는 설정이 있는 경우를 제공해야 `aws.arns.assume_role_arn` 합니다.

RabbitMQ용 Amazon MQ에 대한 LDAP 인증 및 권한 부여

RabbitMQ용 Amazon MQ는 외부 LDAP 서버를 사용하는 브로커 사용자의 인증 및 권한 부여를 지원합니다. 지원되는 다른 방법은 RabbitMQ용 Amazon MQ 브로커에 대한 인증 및 권한 부여를 참조하세요.

⚠ 중요 고려 사항

- 퍼블릭 인터넷을 통해 LDAP 서버에 액세스할 수 있어야 합니다. RabbitMQ용 Amazon MQ는 상호 TLS를 사용하여 LDAP 서버에 인증하도록 구성할 수 있습니다.
- RabbitMQ용 Amazon MQ는 암호와 같은 민감한 LDAP 설정과 로컬 파일 시스템에 액세스해야 하는 설정에 AWS ARNs 사용을 적용합니다. 자세한 내용은 RabbitMQ 구성의 ARN 지원을 참조하세요.
- 기존 브로커에서 LDAP를 활성화하려면 IAM 권한 `mq:UpdateBrokerAccessConfiguration`를 포함해야 합니다.
- Amazon MQ는 모니터링 전용 권한이 있는 `monitoring-AWS-OWNED-DO-NOT-DELETE`라는 시스템 사용자를 자동으로 생성합니다. 이 사용자는 LDAP 지원 브로커에서도 RabbitMQ의 내부 인증 시스템을 사용하며 루프백 인터페이스 액세스로만 제한됩니다. Amazon MQ는 보호된 사용자 태그를 추가하여이 사용자의 삭제를 방지합니다.

RabbitMQ용 Amazon MQ 브로커에 LDAP를 구성하는 방법에 대한 자세한 내용은 섹션을 참조하세요LDAP 인증 및 권한 부여 사용.

이 페이지의 내용

지원되는 LDAP 구성

RabbitMQ용 Amazon MQ는 AWS ARNs이 필요한 다음 예외를 제외하고 RabbitMQ LDAP 플러그인에서 구성 가능한 모든 변수를 지원합니다. RabbitMQ ARN 지원에 대한 자세한 내용은 RabbitMQ 구성의 ARN 지원을 참조하세요.

ARNs이 필요한 구성

`auth_ldap.dn_lookup_bind.password`

대신 `aws.arns.auth_ldap.dn_lookup_bind.password` 사용

`auth_ldap.other_bind.password`

대신 `aws.arns.auth_ldap.other_bind.password` 사용

`auth_ldap.ssl_options.cacertfile`

대신 `aws.arns.auth_ldap.ssl_options.cacertfile` 사용

`auth_ldap.ssl_options.certfile`

대신 `aws.arns.auth_ldap.ssl_options.certfile` 사용

`auth_ldap.ssl_options.keyfile`

대신 `aws.arns.auth_ldap.ssl_options.keyfile` 사용

지원되지 않는 SSL 옵션

다음 SSL 구성 옵션도 지원되지 않습니다.

전체 목록 보기

- `auth_ldap.ssl_options.cert`
- `auth_ldap.ssl_options.client_renegotiation`
- `auth_ldap.ssl_options.dh`
- `auth_ldap.ssl_options.dhfile`
- `auth_ldap.ssl_options.honor_cipher_order`
- `auth_ldap.ssl_options.honor_ecc_order`
- `auth_ldap.ssl_options.key.RSAPrivateKey`
- `auth_ldap.ssl_options.key.DSAPrivateKey`
- `auth_ldap.ssl_options.key.PrivateKeyInfo`
- `auth_ldap.ssl_options.log_alert`
- `auth_ldap.ssl_options.password`
- `auth_ldap.ssl_options.psk_identity`
- `auth_ldap.ssl_options.reuse_sessions`

- `auth_ldap.ssl_options.secure_renegotiate`
- `auth_ldap.ssl_options.versions.$version`
- `auth_ldap.ssl_options.sni`

Amazon MQ의 LDAP 구성에 대한 추가 검증

또한 Amazon MQ는 LDAP 인증 및 권한 부여에 대해 다음과 같은 추가 검증을 적용합니다.

- `auth_ldap.log`는 로 설정할 수 없습니다. `network_unsafe`
- LDAP 서버는 LDAPS를 사용해야 합니다. `auth_ldap.use_ssl` 또는를 명시적으로 활성화해야 `auth_ldap.use_starttls` 합니다.
- AWS ARN을 사용해야 하는 설정이 있는 경우를 제공해야 `aws.arns.assume_role_arn` 합니다.
- `auth_ldap.servers`는 유효한 IP 주소 또는 유효한 FQDN이어야 합니다.
- 다음 키는 유효한 LDAP 고유 이름이어야 합니다.
  - `auth_ldap.dn_lookup_base`
  - `auth_ldap.dn_lookup_bind.user_dn`
  - `auth_ldap.other_bind.user_dn`
  - `auth_ldap.group_lookup_base`

플러그인

RabbitMQ용 Amazon MQ는 다음 플러그인도 지원합니다. RabbitMQ

- RabbitMQ 관리 플러그인
- 셔블 플러그인
- 페더레이션 플러그인
- 일관된 해시 교환 플러그인
- OAuth 2 플러그인
- LDAP 플러그인
- HTTP 플러그인
- SSL 인증서 플러그인
- aws 플러그인
- JMS Topic Exchange 플러그인

- Prometheus 플러그인

RabbitMQ 관리 플러그인

RabbitMQ용 Amazon MQ는 RabbitMQ 웹 콘솔용 브라우저 기반 UI와 함께 HTTP 기반 관리 API를 제공하는 RabbitMQ 관리 플러그인을 지원합니다. RabbitMQ 웹 콘솔 및 관리 API를 사용하여 브로커 사용자 및 정책을 생성하고 관리할 수 있습니다.

Shovel 플러그 인

RabbitMQ용 Amazon MQ는 RabbitMQ 셔블 플러그인을 지원하므로 한 브로커의 대기열 및 교환에서 다른 브로커로 메시지를 이동할 수 있습니다. RabbitMQ Shovel을 사용하여 느슨하게 결합된 브로커를 연결하고 메시지 로드가 많은 노드의 메시지를 분산할 수 있습니다.

> **Important**
>
> shovel 대상이 프라이빗 브로커인 경우 대기열 또는 교환 간 shovel을 구성할 수 없습니다. Amazon MQ는 정적 shovel 사용은 지원하지 않습니다.

동적 셔블만 지원됩니다. 동적 셔블은 런타임 파라미터를 사용하여 구성되며 클라이언트 연결을 통해 프로그래밍 방식으로 언제든지 시작하고 중지할 수 있습니다. 예를 들어 RabbitMQ 관리 API를 사용하여 다음 API 엔드포인트에 대한 PUT 요청을 생성하여 동적 셔블을 구성할 수 있습니다. 이 예제에서는 {vhost}를 브로커의 vhost 이름으로 바꾸고 {name}을 새 동적 셔블의 이름으로 바꿀 수 있습니다.

```
/api/parameters/shovel/{vhost}/{name}
```

요청 본문에 대기열이나 교환 중 하나만 지정해야 합니다. 아래 예제에서는 src-queue에 지정된 로컬 대기열과 dest-queue에 정의된 원격 대기열 간에 동적 셔블을 구성합니다. 마찬가지로 src-exchange 및 dest-exchange 파라미터를 사용하여 두 교환 간에 셔블을 구성할 수 있습니다.

```
{
"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-
west2.amazonaws.com:5671",
```

```
"dest-queue": "destination-queue-name"
}
}
```

Federation 플러그 인

Amazon MQ는 RabbitMQ 페더레이션 플러그인을 사용하여 페더레이션 교환 및 대기열을 지원합니다. Federation을 사용하면 개별 브로커에서 대기열, 교환 및 소비자 간의 메시지 흐름을 복제할 수 있습니다. 연동 대기열 및 교환은 지점 간 링크를 사용하여 다른 브로커의 피어에 연결합니다. 기본적으로 연동 교환은 메시지를 한 번 라우팅하는 반면 연동 대기열은 소비자가 필요한 만큼 몇 번이든 메시지를 이동할 수 있습니다.

Federation을 사용하면 다운스트림 브로커가 업스트림에 있는 교환 또는 대기열의 메시지를 소비하도록 할 수 있습니다. RabbitMQ 웹 콘솔 또는 관리 API를 사용하여 다운스트림 브로커에서 federation을 활성화할 수 있습니다.

> **Important**
>
> 업스트림 대기열 또는 교환이 프라이빗 브로커에 있는 경우 페더레이션을 구성할 수 없습니다. 퍼블릭 브로커의 대기열 또는 교환 간이나 퍼블릭 브로커의 업스트림 대기열 또는 교환과 프라이빗 브로커의 다운스트림 대기열 또는 교환 간에만 페더레이션을 구성할 수 있습니다.

예를 들어 관리 API를 사용하면 다음을 수행하여 federation을 구성할 수 있습니다.

- 다른 노드에 대한 federation 연결을 정의하는 하나 이상의 업스트림을 구성합니다. RabbitMQ 웹 콘솔 또는 관리 API를 사용하여 federation 연결을 정의할 수 있습니다. 관리 API를 사용하여 다음 요청 본문을 사용하여 /api/parameters/federation-upstream/%2f/myupstream에 대한 POST 요청을 생성할 수 있습니다.

  ```
  {"value":{"uri":"amqp://server-name","expires":3600000}}
  ```

- 대기열 또는 교환이 연동될 수 있도록 정책을 구성합니다. RabbitMQ 웹 콘솔 또는 관리 API를 사용하여 정책을 구성할 수 있습니다. 관리 API를 사용하여 다음 요청 본문을 사용하여 /api/policies/%2f/federate-me에 대한 POST 요청을 생성할 수 있습니다.

  ```
  {"pattern":"^amq\.", "definition":{"federation-upstream-set":"all"}, "apply-
  to":"exchanges"}
  ```

> **Note**
>
> 요청 본문은 서버의 교환 이름이 amq로 시작한다고 가정합니다. 정규식 ^amq\를 사용합니다.는 이름이 "amq"로 시작하는 모든 교환에 대해 페더레이션이 활성화되도록 합니다. RabbitMQ 서버의 교환은 이름을 다르게 지정할 수 있습니다.

일관적 해시 교환 플러그 인

RabbitMQ용 Amazon MQ는 RabbitMQ 일관된 해시 교환 유형 플러그인을 지원합니다. 일관적 해시 교환은 메시지의 라우팅 키에서 계산된 해시 값을 기반으로 메시지를 대기열로 라우팅합니다. 라우팅 키가 적절히 균등하면 일관적 해시 교환은 메시지를 대기열 간에 비교적 균등하게 분산할 수 있습니다.

대기열이 일관적 해시 교환에 바인딩된 경우 바인딩 키는 각 대기열의 바인딩 가중치를 결정하는 문자열인 숫자입니다. 바인딩 가중치가 높은 대기열은 바인딩된 일관적 해시 교환에서 비례하여 높은 분포의 메시지를 받습니다. 일관적 해시 교환 토폴로지에서 게시자는 단순히 메시지를 교환에 게시할 수 있지만 소비자는 특정 대기열의 메시지를 소비하도록 명시적으로 구성해야 합니다.

OAuth 2.0 플러그인

RabbitMQ용 Amazon MQ는 OAuth 2 인증 백엔드 플러그인을 지원합니다. 이 플러그인은 브로커 구성에 따라 조건부로 활성화됩니다. 활성화되면이 플러그인은 중앙 집중식 사용자 관리 및 액세스 제어를 위해 외부 OAuth 2.0 자격 증명 공급자와의 통합과 함께 OAuth 2.0 인증 및 권한 부여를 제공합니다. OAuth 2.0 인증에 대한 자세한 내용은 섹션을 참조하세요OAuth 2.0 인증 및 권한 부여.

LDAP 플러그인

RabbitMQ용 Amazon MQ는 LDAP 인증 백엔드 플러그인을 지원합니다. 이 플러그인은 브로커 구성에 따라 조건부로 활성화됩니다. 활성화되면이 플러그인은 중앙 집중식 사용자 인증 및 권한 부여를 위해 외부 LDAP 디렉터리 서비스에 대한 통합과 함께 LDAP 인증 및 권한 부여를 제공합니다. LDAP 인증에 대한 자세한 내용은 섹션을 참조하세요LDAP 인증 및 권한 부여.

HTTP 플러그인

RabbitMQ용 Amazon MQ는 HTTP 인증 백엔드 플러그인을 지원합니다. RabbitMQ 이 플러그인은 브로커 구성에 따라 조건부로 활성화됩니다. 활성화되면이 플러그인은 중앙 집중식 사용자 인증 및 권한 부여를 위해 외부 HTTP 서버에 대한 통합과 함께 HTTP 인증 및 권한 부여를 제공합니다. HTTP 인증에 대한 자세한 내용은 섹션을 참조하세요HTTP 인증 및 권한 부여.

> **Note**
>
> HTTP 인증 플러그인은 RabbitMQ용 Amazon MQ 버전 4 이상에서만 사용할 수 있습니다. RabbitMQ

SSL 인증서 플러그인

Amazon MQ는 RabbitMQ 브로커에 대해 상호 TLS(mTLS)를 지원합니다. SSL 인증 플러그인은 mTLS 연결의 클라이언트 인증서를 사용하여 사용자를 인증합니다. 이 플러그인은 브로커 구성에 따라 조건부로 활성화됩니다. 활성화하면 네트워크를 통해 자격 증명을 전송하지 않고 강력한 인증을 위해 X.509 클라이언트 인증서를 사용하여 인증서 기반 인증을 제공합니다. SSL 인증서 인증에 대한 자세한 내용은 섹션을 참조하세요SSL 인증서 인증.

> **Note**
>
> SSL 인증서 인증 플러그인은 RabbitMQ용 Amazon MQ 버전 4 이상에서만 사용할 수 있습니다. RabbitMQ

aws 플러그인

aws 플러그인은 브로커 구성에 따라 RabbitMQ용 Amazon MQ에서 조건부로 활성화됩니다. RabbitMQ Amazon MQ에서 개발 및 유지 관리하는이 커뮤니티 플러그인은 RabbitMQ 구성 설정에서 AWS ARNs 사용하여 AWS 서비스에서 자격 증명 및 인증서를 안전하게 검색할 수 있습니다. ARN 지원에 대한 자세한 내용은 섹션을 참조하세요ARN support in RabbitMQ configuration.

JMS Topic Exchange 플러그인

JMS Topic Exchange 플러그인은 항상 RabbitMQ용 Amazon MQ에서 활성화됩니다. RabbitMQ JMS 클라이언트와 함께 작동하여 신규 및 기존 JMS 애플리케이션이 RabbitMQ용 Amazon MQ에 연결할 수 있도록 합니다.

> **Note**
>
> JMS Topic Exchange 플러그인은 RabbitMQ용 Amazon MQ 버전 4 이상에서만 사용할 수 있습니다. RabbitMQ 기본적으로 활성화되어 있지만 RabbitMQ JMS 클라이언트를 사용하여 JMS 워크로드를 실행하는 경우에만 활성화됩니다.

Prometheus 플러그인

Prometheus 플러그인은 RabbitMQ에 내장되어 있으며 RabbitMQ 브로커용 Amazon MQ에서 항상 활성화됩니다. RabbitMQ 4.2부터 Amazon MQ는 장기 모니터링을 위한 관리 플러그인 대신 노드별 지표에 Prometheus를 사용하는 RabbitMQ 오픈 소스 전략과 일치합니다.

플러그인은 `/metrics`, `/metrics/detailed`및 `/metrics/memory-breakdown` 엔드포인트를 통해 Prometheus 텍스트 형식으로 지표를 노출합니다. Prometheus 또는 호환되는 모니터링 도구를 사용하여 이러한 엔드포인트를 스크레이프하여 대시보드를 구축하고 알림을 설정할 수 있습니다.

또한 Amazon MQ는 이러한 Prometheus 지표의 선별된 하위 집합을 CloudWatch에 게시합니다. CloudWatch 지표에 대한 자세한 내용은 RabbitMQ용 Amazon MQ 브로커에 사용 가능한 CloudWatch 지표 섹션을 참조하세요.

엔드포인트 세부 정보, 인증 및 스크레이핑 구성은 섹션을 참조하세요Prometheus 지표 액세스.

지원되는 프로토콜

RabbitMQ가 지원하는 프로그래밍 언어를 사용하고 다음 프로토콜 사양에 대해 TLS를 활성화하여 RabbitMQ 브로커에 액세스할 수 있습니다.

- AMQP(0-9-1)
- AMQP 1.0
- JMS 1.1
- JMS 2.0
- JMS 3.1

RabbitMQ용 Amazon MQ JMS 지원 RabbitMQ

이제 RabbitMQ JMS 클라이언트를 사용하여 RabbitMQ 4용 Amazon MQ에서 JMS 1.1, 2.0 및 3.1 워크로드를 실행할 수 RabbitMQ.

RabbitMQ JMS 클라이언트

RabbitMQ JMS 클라이언트는 JMS 애플리케이션을 Amazon MQ RabbitMQ 브로커에 연결하는 데 필요한 오픈 소스 JMS 클라이언트 라이브러리입니다. 자세한 내용은 공식 GitHub 리포지토리를 참조하세요.

지원되는 JMS 1.1, 2.0 및 3.1 APIs

RabbitMQ 4용 Amazon MQ부터는 플러그인`jms-topic-exchange`이 항상 활성화됩니다. RabbitMQ 따라서 RabbitMQ 4용 Amazon MQ와 JMS 워크로드용 RabbitMQ JMS 클라이언트를 사용할 수 있습니다. JMS 1.1에 정의된 모든 JMS APIs는 다음을 제외하고 지원됩니다.

- 서버 세션 APIs는 지원되지 않습니다.
- XA 트랜잭션 APIs는 지원되지 않습니다.
- JMS 대기열 대상에 대한 JMS 선택기는 지원되지 않습니다.
- JMS `NoLocal` 구독 속성은 지원되지 않습니다.

다음을 제외하고 JMS 2.0 및 JMS 3.1에서 새로 추가된 모든 APIs가 지원됩니다. https://javaee.github.io/jms-spec/pages/JMS20FinalRelease#reference-implementation

- `JMSProducer.setDeliveryDelay` API는 지원되지 않습니다.

JMS 애플리케이션을 RabbitMQ용 Amazon MQ 브로커에 연결하는 방법에 대한 자세한 내용은 JMS 애플리케이션을 RabbitMQ용 Amazon MQ 브로커에 연결하는 자습서를 참조하세요.

인증 및 권한 부여

이 섹션에 나열된 모든 인증 및 권한 부여 메커니즘이 지원됩니다. JMS 클라이언트를 사용하여 브로커에 연결하는 데 사용되는 자격 증명은 AMQP Java 클라이언트를 사용하여 RabbitMQ 브로커에 연결하는 것과 동일합니다.

RabbitMQ에서 AMQP 대기열과의 상호 운용성

RabbitMQ JMS 클라이언트를 사용하여 JMS 메시지를 AMQP 교환으로 보내고 AMQP 대기열의 메시지를 사용할 수 있습니다(이 기능은 JMS 주제를 지원하지 않음). 이를 통해 특정 JMS 워크로드를 AMQP 워크로드로 상호 운용하거나 마이그레이션할 수 있습니다. 자세한 내용은 공식 클라이언트 설명서를 참조하세요.

RabbitMQ용 Amazon MQ에 정책 적용

Amazon MQ 권장 기본값을 사용하여 사용자 지정 정책과 제한을 적용할 수 있습니다. 권장되는 기본 정책 및 제한을 삭제한 후 다시 생성하거나 추가 vhost를 생성한 후 새 vhost에 기본 정책과 제한을 적용하려는 경우 다음 단계를 사용할 수 있습니다.

Important

RabbitMQ용 Amazon MQ 엔진 버전 3.13 이하에서 현재 기본 연산자 정책은 다음과 같습니다. RabbitMQ

```
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
```

버전 4.0 이상에서는 기본 연산자 정책이 다음과 같이 변경되었습니다.

```
vhost name pattern apply-to definition priority/
 default_operator_policy_AWS_managed .* classic_queues {"queue-version":2} 0
```

RabbitMQ 4에서는 클래식 대기열 미러링 및 HA 정책 설정이 지원되지 않으므로이 변경이 필요합니다.
클래식 미러링 대기열과 쿼럼 대기열에 동시에 적용되는 정책은 생성할 수 없습니다. 정책이 쿼럼 대기열에만 적용되도록 하려면 `--apply-to`를 `quorum_queues`로 설정해야 합니다.
클래식 미러링 대기열과 쿼럼 대기열을 함께 사용하는 경우 쿼럼 대기열 정책뿐만 아니라 `--apply-to:classic_queues`로 설정한 별도의 정책을 생성해야 합니다.

Important

다음 단계를 수행하려면 관리자 권한이 있는 RabbitMQ용 Amazon MQ 브로커 사용자가 있어야 합니다. 브로커를 처음 생성할 때 만든 관리자 사용자를 사용하거나 나중에 만들었을 수 있는 다른 사용자를 사용할 수 있습니다. 다음 표에는 필요한 관리자 사용자 태그 및 권한이 정규식(regexp) 패턴으로 나와 있습니다.

Tags	regexp 읽기	regexp 구성	regexp 쓰기
`administrator`	`.*`	`.*`	`.*`

RabbitMQ 사용자를 생성하고 사용자 태그 및 권한을 관리하는 방법에 대한 자세한 내용은 RabbitMQ용 Amazon MQ 브로커 사용자 단원을 참조하세요.

RabbitMQ 웹 콘솔을 사용하여 기본 정책 및 가상 호스트 제한을 적용하려면

1. Amazon MQ 콘솔에 로그인합니다.
2. 왼쪽 탐색 창에서 Brokers(브로커)를 선택합니다.
3. 브로커 목록에서 새 정책을 적용할 브로커의 이름을 선택합니다.
4. 브로커 세부 정보 페이지의 Connections(연결) 섹션에서 RabbitMQ 웹 콘솔 URL을 선택합니다. RabbitMQ 웹 콘솔이 새 브라우저 탭 또는 창에 열립니다.
5. 브로커 관리자 사용자 이름 및 암호를 사용하여 RabbitMQ 웹 콘솔에 로그인합니다.
6. RabbitMQ 웹 콘솔의 페이지 상단에서 Admin(관리자)을 선택합니다.
7. Admin(관리자) 페이지의 오른쪽 탐색 창에서 Policies(정책)를 선택합니다.
8. Policies(정책) 페이지에서 브로커의 현재 User policies(사용자 정책) 목록을 볼 수 있습니다. User policies(사용자 정책) 아래에서 Add / update a policy(정책 추가/업데이트)를 확장합니다.
9. 새 브로커 정책을 생성하려면 Add / update a policy(정책 추가/업데이트)에서 다음을 수행합니다.

    a. Virtual host(가상 호스트)의 드롭다운 목록에서 정책을 연결할 vhost의 이름을 선택합니다. 기본 vhost를 선택하려면 /를 선택합니다.

    > **Note**
    >
    > 추가 vhost를 생성하지 않은 경우 Virtual host(가상 호스트) 옵션이 RabbitMQ 콘솔에 표시되지 않으며 정책은 기본 vhost에만 적용됩니다.

    b. Name(이름)에 정책의 이름을 입력합니다(예: **`policy-defaults`**).

    c. Pattern(패턴)에 regexp 패턴 **`.*`**를 입력합니다. 이 경우 정책이 브로커의 모든 대기열과 일치합니다.

    d. Apply to(적용 대상)의 드롭다운 목록에서 Exchanges and queues(교환 및 대기열)를 선택합니다.

    e. Priority(우선 순위)에 vhost에 적용된 다른 모든 정책보다 큰 정수를 입력합니다. 지정된 시간에 RabbitMQ 대기열 및 교환에 정확히 하나의 정책 정의 집합을 적용할 수 있습니다. RabbitMQ는 가장 높은 우선 순위 값과 일치하는 정책을 선택합니다. 정책 우선 순위 및 정책을 결합하는 방법에 대한 자세한 내용은 RabbitMQ Server 설명서에서 정책을 참조하세요.

    f. Definition(정의)에 다음 키-값 페어를 추가합니다.

    - **`queue-mode=lazy`**. 드롭다운 목록에서 String(문자열)을 선택합니다.
    - **`overflow=reject-publish`**. 드롭다운 목록에서 String(문자열)을 선택합니다.

> **Note**
>
> 단일 인스턴스 브로커에는 적용하지 마세요.

- **`max-length=`*`number-of-messages`***. 브로커의 인스턴스 크기 및 배포 모드에 따라 *`number-of-messages`*를 Amazon MQ 권장 값으로 바꿉니다(예: `mq.m7g.large` 클러스터의 경우 **`8000000`**). 드롭다운 목록에서 Number(숫자)를 선택합니다.

> **Note**
>
> 단일 인스턴스 브로커에는 적용하지 마세요.

g. Add / update policy(정책 추가/업데이트)를 선택합니다.

10. 새 정책이 User policies(사용자 정책)의 목록에 표시되는지 확인합니다.

> **Note**
>
> 클러스터 브로커의 경우 Amazon MQ는 `ha-mode: all` 및 `ha-sync-mode: automatic` 정책 정의를 자동으로 적용합니다.

11. 오른쪽 탐색 창에서 Limits(제한)를 선택합니다.
12. Limits(제한) 페이지에서 브로커의 현재 Virtual host limits(가상 호스트 제한) 목록을 볼 수 있습니다. Virtual host limits(가상 호스트 제한) 아래에서 Set / update a virtual host limit(가상 호스트 제한 설정/업데이트)를 확장합니다.
13. 새 vhost 제한을 생성하려면 Set / update a virtual host limit(가상 호스트 제한 설정/업데이트) 아래에서 다음을 수행합니다.

    a. Virtual host(가상 호스트)의 드롭다운 목록에서 정책을 연결할 vhost의 이름을 선택합니다. 기본 vhost를 선택하려면 /를 선택합니다.

    b. Limit(제한)의 드롭다운 옵션에서 max-connections를 선택합니다.

    c. Value(값)에 브로커의 인스턴스 크기 및 배포 모드에 따라 Amazon MQ 권장 값을 입력합니다(예: `mq.m5.large` 클러스터의 경우 **`15000`**).

    d. Set / update limit(제한 설정/업데이트)를 선택합니다.

    e. 위 단계를 반복하고 Limit(제한)의 드롭다운에서 max-queues를 선택합니다.

14. 새 제한이 Virtual host limits(가상 호스트 제한)의 목록에 표시되는지 확인합니다.

RabbitMQ 관리 API를 사용하여 기본 정책 및 가상 호스트 제한을 적용하려면

1. Amazon MQ 콘솔에 로그인합니다.
2. 왼쪽 탐색 창에서 Brokers(브로커)를 선택합니다.
3. 브로커 목록에서 새 정책을 적용할 브로커의 이름을 선택합니다.
4. 브로커 페이지의 Connections(연결) 섹션에서 RabbitMQ 웹 콘솔 URL을 기록합니다. 이는 HTTP 요청에서 사용하는 브로커 엔드포인트입니다.
5. 새 터미널 또는 원하는 명령줄 창을 엽니다.
6. 새 브로커 정책을 생성하려면 다음 `curl` 명령을 입력합니다. 이 명령에서는 대기열이 `%2F`로 인코딩된 기본 `/` vhost에 있다고 가정합니다. 정책을 다른 vhost에 적용하려면 `%2F`를 vhost의 이름으로 바꿉니다.

   > **Note**
   >
   > *`###`* *`##`*과 *`##`*를 관리자 로그인 보안 인증 정보로 바꾸세요. 브로커의 인스턴스 크기 및 배포 모드에 따라 *`number-of-messages`*를 Amazon MQ 권장 값으로 바꿉니다. *`policy-name`*을 정책의 이름으로 바꿉니다. *`broker-endpoint`*를 이전에 기록한 URL로 바꿉니다.

   ```
   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. 브로커의 사용자 정책에 새 정책이 추가되었는지 확인하려면 다음 `curl` 명령을 입력하여 모든 브로커 정책을 나열합니다.

   ```
   curl -i -u username:password broker-endpoint/api/policies
   ```

8. 새 `max-connections` 가상 호스트 제한을 생성하려면 다음 `curl` 명령을 입력합니다. 이 명령에서는 대기열이 `%2F`로 인코딩된 기본 `/` vhost에 있다고 가정합니다. 정책을 다른 vhost에 적용하려면 `%2F`를 vhost의 이름으로 바꿉니다.

> **Note**
>
> *### ##*과 *##*를 관리자 로그인 보안 인증 정보로 바꾸세요. 브로커의 인스턴스 크기 및 배포 모드에 따라 *max-connections*를 Amazon MQ 권장 값으로 바꿉니다. 브로커 엔드포인트를 이전에 기록한 URL로 바꿉니다.

```
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. 새 `max-queues` 가상 호스트 제한을 생성하려면 이전 단계를 반복하되 다음과 같이 curl 명령을 수정합니다.

```
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. 브로커의 가상 호스트 제한에 새 제한이 추가되었는지 확인하려면 다음 `curl` 명령을 입력하여 모든 브로커 가상 호스트 제한을 나열합니다.

```
curl -i -u username:password broker-endpoint/api/vhost-limits
```

Amazon MQ 기반 RabbitMQ의 쿼럼 대기열

쿼럼 대기열은 리더(기본 복제본)와 팔로워(기타 복제본)로 구성된 복제 대기열 유형입니다. 리더를 사용할 수 없게 되면 쿼럼 대기열은 Raft 합의 알고리즘을 사용하여 과반수 득표로 새로운 리더 노드를 선출하고, 이전 리더는 동일한 클러스터에서 팔로워 노드로 강등됩니다. 나머지 팔로워는 이전처럼 계속 복제됩니다. 각 노드는 서로 다른 가용 영역에 있기 때문에 한 노드를 일시적으로 사용할 수 없는 경우에도 다른 가용 영역에서 새로 선출된 리더 복제본을 통해 메시지 전송이 계속됩니다.

쿼럼 대기열은 메시지가 실패하여 여러 번 대기열에 추가될 때 발생하는 유해 메시지를 처리하는 데 유용합니다.

다음과 같은 경우에는 쿼럼 대기열을 사용해서는 안 됩니다.

- 임시 대기열을 사용하는 경우
- 대기열 백로그가 긴 경우
- 짧은 대기 시간을 우선시하는 경우

쿼럼 대기열을 선언하려면 `x-queue-type` 헤더를 `quorum`으로 설정합니다.

주제

- RabbitMQ용 Amazon MQ의 클래식 대기열에서 쿼럼 대기열로 마이그레이션
- RabbitMQ용 Amazon MQ의 쿼럼 대기열에 대한 정책 구성
- RabbitMQ용 Amazon MQ의 쿼럼 대기열 모범 사례

RabbitMQ용 Amazon MQ의 클래식 대기열에서 쿼럼 대기열로 마이그레이션

동일한 클러스터에서 새 가상 호스트를 생성하거나 인플레이스 마이그레이션을 수행하여 Amazon MQ 브로커 버전 3.13 이상에서 클래식 미러링 대기열을 쿼럼 대기열로 마이그레이션할 수 있습니다.

옵션 1: RabbitMQ용 Amazon MQ 대기열 마이그레이션 도구를 사용하여 클래식 미러링 대기열에서 쿼럼 대기열로 마이그레이션

Amazon MQ는 클래식 대기열을 쿼럼 대기열로 마이그레이션하는 대기열 마이그레이션 도구를 제공합니다. 이 도구는 RabbitMQ 웹 콘솔의 관리자 > 대기열 마이그레이션 또는 HTTP API를 통해 액세스할 수 있습니다.

도구를 사용하려면 Amazon MQ 대기열 마이그레이션 도구를 참조하세요.

옵션 2: 새 가상 호스트를 사용하여 클래식 미러링 대기열에서 쿼럼 대기열로 마이그레이션

동일한 클러스터에서 새 가상 호스트를 생성하여 Amazon MQ 브로커 버전 3.13 이상에서 클래식 미러링 대기열을 쿼럼 대기열로 마이그레이션할 수 있습니다.

1. 기존 클러스터에서 기본 대기열 유형을 쿼럼으로 설정하여 새 가상 호스트(vhost)를 생성합니다.
2. URI가 클래식 미러링 대기열을 사용하는 이전 가상 호스트를 가리키도록 설정하여 새 vhost에서 Federation 플러그 인을 생성합니다.
3. `rabbitmqadmin`을 사용하여 이전 vhost의 정의를 새 파일로 내보냅니다. 스키마 파일이 쿼럼 대기열과 호환되도록 변경해야 합니다. 파일에 적용해야 할 전체 변경 사항 목록은 RabbitMQ 쿼

럼 대기열 설명서에서 정의 이동을 참조하세요. 필요한 변경 사항을 파일에 적용한 후 정의를 새 vhost로 다시 가져옵니다.

4. 새 vhost에서 새 정책을 생성합니다. 쿼럼 대기열의 Amazon MQ 정책 구성에 대한 권장 사항은 RabbitMQ용 Amazon MQ의 쿼럼 대기열에 대한 정책 구성 단원을 참조하세요. 그런 다음 앞에서 생성한 페더레이션을 이전 vhost에서 새 vhost로 시작합니다.
5. 소비자와 생산자가 새 vhost를 사용하도록 설정합니다.
6. Shovel 플러그인을 구성하여 나머지 메시지를 이동합니다. 대기열이 비워지면 Shovel을 삭제합니다.

클래식 미러링 대기열에서 쿼럼 대기열로의 인플레이스 마이그레이션

인플레이스 마이그레이션을 수행하여 Amazon MQ 브로커 버전 3.13 이상에서 클래식 미러링 대기열을 쿼럼 대기열로 마이그레이션할 수 있습니다.

1. 소비자와 생산자를 중지합니다.
2. 임시 쿼럼 대기열을 새로 생성합니다.
3. Shovel 플러그인을 구성하여 이전 클래식 미러링 대기열에서 새 임시 쿼럼 대기열로 메시지를 이동합니다. 모든 메시지가 임시 쿼럼 대기열로 이동하면 Shovel을 삭제합니다.
4. 소스 클래식 미러링 대기열을 삭제합니다. 그런 다음 소스 클래식 미러링 대기열과 동일한 이름과 바인딩을 사용하는 쿼럼 대기열을 다시 생성합니다.
5. 새 Shovel을 생성하여 임시 쿼럼 대기열에서 새 쿼럼 대기열로 메시지를 이동합니다.

RabbitMQ용 Amazon MQ의 쿼럼 대기열에 대한 정책 구성

Amazon MQ에서 RabbitMQ 브로커의 쿼럼 대기열에 특정 정책 구성을 추가할 수 있습니다.

쿼럼 대기열에 대한 정책을 생성할 때 다음을 수행해야 합니다.

- `ha`로 시작하는 정책 속성을 모두 제거합니다(예: `ha-mode`, `ha-params`, `ha-sync-mode`, `ha-sync-batch-size`, `ha-promote-on-shutdown`, `ha-promote-on-failure`).
- `queue-mode`를 제거합니다.
- 오버플로가 `reject-publish-dlx`로 설정된 경우 오버플로를 변경합니다.

> **Important**
>
> RabbitMQ용 Amazon MQ는 정책 내의 속성을 모두 적용하거나 아예 적용하지 않습니다. 클래식 미러링 대기열과 쿼럼 대기열에 동시에 적용되는 정책은 생성할 수 없습니다. 정책이 쿼럼 대기열에만 적용되도록 하려면 `--apply-to`를 `quorum_queues`로 설정해야 합니다. 클래식 미러링 대기열과 쿼럼 대기열을 함께 사용하는 경우 쿼럼 대기열 정책뿐만 아니라 `--apply-to:classic_queues`로 설정한 별도의 정책을 생성해야 합니다.

`AWS-DEFAULT` 정책은 "applies to" 파라미터에서 새 대기열 유형을 자동으로 채택하므로 수정할 필요가 없습니다. RabbitMQ용 Amazon MQ의 기본 정책에 대한 자세한 내용은 운영자 정책 구성 단원을 참조하세요.

RabbitMQ용 Amazon MQ의 쿼럼 대기열 모범 사례

쿼럼 대기열 작업 시 성능을 개선하려면 다음 모범 사례를 사용하는 것이 좋습니다.

전송 한도를 설정하여 유해 메시지 처리

유해 메시지는 메시지가 실패하여 여러 번 다시 전달될 때 발생합니다. `delivery-limit` 정책 인수를 사용하여 메시지 전송 한도를 설정하면 여러 번 재전송되는 메시지를 삭제할 수 있습니다. 메시지가 전송 한도의 허용 횟수보다 더 많이 재전송되면 RabbitMQ에서 해당 메시지를 제거하고 삭제합니다. 전송 한도를 설정하면 메시지가 대기열 맨 앞에 다시 추가됩니다.

쿼럼 대기열의 메시지 우선 순위

쿼럼 대기열에는 메시지 우선 순위가 없습니다. 메시지 우선 순위가 필요한 경우 쿼럼 대기열을 여러 개 생성해야 합니다. 여러 쿼럼 대기열이 있는 메시지의 우선 순위를 정하는 방법은 RabbitMQ 설명서에서 메시지 우선 순위를 참조하세요.

기본 복제 인수 사용

RabbitMQ용 Amazon MQ는 쿼럼 대기열을 사용하는 클러스터 브로커에 대해 기본적으로 복제 인수를 3개 노드로 설정합니다. `x-quorum-initial-group-size`를 변경하면 Amazon MQ에서는 기본값을 다시 복제 인수 3으로 설정합니다.

RabbitMQ용 Amazon MQ 모범 사례

다음 프로덕션 준비 지침에 따라 RabbitMQ용 Amazon MQ 브로커로 작업할 때 브로커 성능을 극대화하고 메시지 처리량 효율성을 최적화합니다.

> **Important**
>
> Amazon MQ는 현재 스트림 또는 RabbitMQ 3.9.x에서 도입된 구조화된 JSON 로깅을 지원하지 않습니다.

주제

- RabbitMQ용 Amazon MQ의 브로커 설정 및 연결 관리 모범 사례
- RabbitMQ용 Amazon MQ의 메시지 내구성 및 신뢰성 모범 사례
- RabbitMQ용 Amazon MQ의 성능 최적화 및 효율성 모범 사례
- RabbitMQ용 Amazon MQ의 네트워크 복원력 및 모니터링 모범 사례

RabbitMQ용 Amazon MQ의 브로커 설정 및 연결 관리 모범 사례

브로커 설정 및 연결 관리는 브로커 메시지 처리량, 리소스 사용률 및 프로덕션 워크로드 처리 기능 문제를 방지하는 첫 번째 단계입니다. RabbitMQ용 Amazon MQ 브로커를 생성하고 구성할 때, 브로커의 성능을 극대화하기 위해 적절한 인스턴스 유형을 선택하고, 연결을 효율적으로 관리하고, 메시지 미리 가져오기를 구성하는 다음 모범 사례를 완료하세요.

> **Important**
>
> RabbitMQ용 Amazon MQ에서는 사용자 이름으로 "guest"를 지원하지 않으므로 새 브로커를 생성하면 기본 게스트 계정이 삭제됩니다. Amazon MQ는 또한 고객이 생성한 "guest"라는 계정도 주기적으로 삭제합니다.

1단계: 클러스터 배포 사용

프로덕션 워크로드의 경우 고가용성과 메시지 복원력을 보장하기 위해 단일 인스턴스 브로커 대신 클러스터 배포를 사용하는 것이 좋습니다. 클러스터 배포는 단일 장애 지점을 제거하고 더 나은 내결함성을 제공합니다.

클러스터 배포는 3개의 가용 영역에 분산된 3개의 RabbitMQ 브로커 노드로 구성되어, 자동 장애 조치를 제공하고 가용 영역 하나가 완전히 사용할 수 없게 되더라도 작업이 계속되도록 합니다. Amazon MQ는 모든 노드에 메시지를 자동으로 복제하여 노드 장애 또는 유지 관리 중에 가용성을 보장합니다.

클러스터 배포는 프로덕션 환경에 필수적이며 Amazon MQ 서비스 수준 계약에서 지원됩니다.

자세한 내용은 RabbitMQ용 Amazon MQ의 클러스터 배포를 참조하세요.

2단계: 올바른 브로커 인스턴스 유형 선택

브로커 인스턴스 유형의 메시지 처리량은 애플리케이션 사용 사례에 따라 다릅니다. `M7g.medium`은 애플리케이션 성능 테스트에만 사용해야 합니다. 프로덕션 환경에서 더 큰 인스턴스를 사용하기 전에 이 작은 인스턴스를 사용하면 애플리케이션 성능이 향상될 수 있습니다. 인스턴스 유형 `m7g.large` 이상에서는 고가용성 및 메시지 내구성을 위해 클러스터 배포를 사용할 수 있습니다. 더 큰 브로커 인스턴스 유형은 프로덕션 수준의 클라이언트 및 대기열, 높은 처리량, 메모리 내 메시지 및 중복 메시지를 처리할 수 있습니다.

올바른 인스턴스 유형 선택에 대한 자세한 내용은 RabbitMQ용 Amazon MQ의 크기 조정 지침을 참조하세요.

3단계: 쿼럼 대기열 사용

클러스터 배포를 사용하는 쿼럼 대기열은 3.13 이상 RabbitMQ 브로커의 프로덕션 환경에서 복제된 대기열 유형에 대한 기본 선택 사항이어야 합니다. 쿼럼 대기열은 높은 안정성, 높은 처리량 및 안정적인 지연 시간을 제공하는 최신 복제 대기열 유형입니다.

쿼럼 대기열은 Raft 합의 알고리즘을 사용하여 내결함성을 개선합니다. 리더 노드를 사용할 수 없게 되면 쿼럼 대기열은 과반수 투표로 새 리더를 자동으로 선출하므로 중단 없이 메시지 전송이 계속됩니다. 각 노드는 서로 다른 가용 영역에 있으므로 가용 영역 하나를 일시적으로 완전히 사용할 수 없게 되더라도 메시징 시스템은 계속 사용할 수 있습니다.

쿼럼 대기열을 선언하려면 대기열을 생성할 때 `x-queue-type` 헤더를 `quorum`으로 설정합니다.

마이그레이션 전략 및 모범 사례를 포함하여 쿼럼 대기열에 대한 자세한 내용은 RabbitMQ용 Amazon MQ의 쿼럼 대기열을 참조하세요.

4단계: 다중 채널 사용

연결 이탈을 방지하려면 단일 연결을 통해 여러 채널을 사용하세요. 애플리케이션은 연결 대 채널 비율이 1:1이 되지 않도록 해야 합니다. 프로세스당 하나의 연결을 사용하고 스레드당 하나의 채널을 사용하는 것이 좋습니다. 채널 유출을 방지하기 위해 과도한 채널 사용을 피하세요.

RabbitMQ용 Amazon MQ의 메시지 내구성 및 신뢰성 모범 사례

애플리케이션을 프로덕션으로 이동하기 전에 메시지 손실 및 리소스 과다 활용을 방지하기 위한 다음 모범 사례를 완료하세요.

1단계: 영구 메시지 및 지속형 대기열 사용

영구 메시지는 브로커가 중단되거나 다시 시작하는 상황에서 데이터 지속성을 보호할 수 있습니다. 영구 메시지는 도착하는 즉시 디스크에 기록됩니다. 그러나 지연 대기열과 달리 영구 메시지는 브로커에 추가 메모리가 필요하지 않는 한 메모리와 디스크 모두에 캐시됩니다. 추가 메모리가 필요한 경우 메시지는 디스크에 대한 메시지 저장을 관리하는 RabbitMQ 브로커 메커니즘(일반적으로 지속성 계층이라고 함)에 의해 메모리에서 제거됩니다.

메시지 지속성을 활성화하려면 대기열을 `durable`로 선언하고 메시지 전달 모드를 `persistent`로 설정할 수 있습니다. 다음의 예제에서는 RabbitMQ Java 클라이언트 라이브러리를 사용하여 지속형 대기열을 선언하는 방법을 보여줍니다. AMQP 0-9-1로 작업할 때 전송 모드를 "2"로 설정하여 메시지를 영구 메시지로 표시할 수 있습니다.

```
boolean durable = true;
channel.queueDeclare("my_queue", durable, false, false, null);
```

대기열을 지속형으로 구성하면 다음 예제와 같이 `MessageProperties`를 `PERSISTENT_TEXT_PLAIN`으로 설정하여 영구 메시지를 대기열에 보낼 수 있습니다.

```
import com.rabbitmq.client.MessageProperties;

channel.basicPublish("", "my_queue",
            MessageProperties.PERSISTENT_TEXT_PLAIN,
            message.getBytes());
```

2단계: 게시자 확인 및 소비자 전송 승인 구성

메시지가 브로커에 전송되었는지 확인하는 프로세스를 게시자 확인이라고 합니다. 게시자 확인은 메시지가 안정적으로 저장된 시점을 애플리케이션에 알립니다. 게시자 확인은 브로커에 저장되는 메시지의 속도를 제어하는 데도 도움이 됩니다. 게시자 확인이 없으면 메시지가 성공적으로 처리되었는지 확인할 수 없으며, 브로커가 처리할 수 없는 메시지를 삭제할 수 있습니다.

마찬가지로 클라이언트 애플리케이션에서 메시지 전송 및 사용에 대한 확인을 브로커에게 다시 보내는 것을 소비자 전송 승인이라고 합니다. 확인 및 승인은 모두 RabbitMQ 브로커를 사용할 때 데이터 안전을 보장하기 위해 꼭 필요합니다.

소비자 배달 승인은 일반적으로 클라이언트 애플리케이션에서 구성합니다. AMQP 0-9-1을 사용할 경우 `basic.consume` 메서드를 구성하여 승인을 활성화할 수 있습니다. AMQP 0-9-1 클라이언트는 `confirm.select` 메서드를 전송하여 게시자 확인을 구성할 수도 있습니다.

일반적으로 배달 승인은 채널에서 활성화합니다. 예를 들어 RabbitMQ Java 클라이언트 라이브러리를 사용할 때 다음 예제와 같이 `Channel#basicAck`를 사용하여 간단한 `basic.ack` 긍정 승인을 설정할 수 있습니다.

```
// 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**
>
> 승인되지 않은 메시지는 메모리에 캐시되어야 합니다. 클라이언트 애플리케이션의 미리 가져오기 설정을 구성하여 소비자가 미리 가져오는 메시지 수를 제한할 수 있습니다.

소비자가 전송을 확인하지 않는 경우를 감지하도록 `consumer_timeout`을 구성할 수 있습니다. 소비자가 제한 시간 값 내에 확인을 보내지 않으면 채널이 닫히고 `PRECONDITION_FAILED`를 수신하게 됩니다. 오류를 진단하려면 UpdateConfiguration API를 사용하여 `consumer_timeout` 값을 늘리세요.

3단계: 대기열을 짧게 유지

클러스터 배포에서 대기열에 메시지 수가 많으면 리소스가 과도하게 사용될 수 있습니다. 브로커가 과도하게 사용되면 RabbitMQ용 Amazon MQ가 재부팅하여 성능이 더욱 저하될 수 있습니다. 재부팅되면 과도하게 사용된 브로커가 `REBOOT_IN_PROGRESS` 상태에서 응답하지 않게 될 수 있습니다.

유지 관리 기간 중 Amazon MQ는 모든 유지 관리 작업을 한 번에 한 노드에서 수행하여 브로커가 계속 작동하도록 합니다. 따라서 각 노드의 작동이 다시 시작할 때 대기열을 동기화해야 할 수 있습니다. 동

기화하는 동안 미러에 복제되어야 하는 메시지는 해당하는 Amazon Elastic Block Store(Amazon EBS) 볼륨에서 메모리로 로드되어 배치로 처리됩니다. 메시지를 배치로 처리하면 대기열을 더 빠르게 동기화할 수 있습니다.

대기열이 짧게 유지되고 메시지가 작으면 대기열이 성공적으로 동기화되고 예상대로 작업이 재개됩니다. 그러나 배치의 데이터 양이 노드의 메모리 한도에 가까워지면 노드에서 고용량 메모리 경보가 발생하여 대기열 동기화가 일시 중지됩니다. 메모리 사용량은 CloudWatch의 브로커 노드 지표 `RabbitMemUsed`와 `RabbitMqMemLimit`를 비교하여 확인할 수 있습니다. 메시지가 사용 또는 삭제되거나 배치의 메시지 수가 줄지 않으면 동기화가 완료될 수 없습니다.

대기열 동기화가 클러스터 배포를 위해 일시 중지된 경우 메시지를 사용하거나 삭제하여 대기열의 메시지 수를 줄이는 것이 좋습니다. 대기열 깊이가 줄고 대기열 동기화가 완료되면 브로커 상태가 `RUNNING`으로 변경됩니다. 일시 중지된 대기열 동기화를 해결하기 위해 대기열 동기화 배치 크기를 줄이는 정책을 적용할 수도 있습니다.

또한 자동 삭제 및 TTL 정책을 정의하여 리소스 사용량을 선제적으로 줄이고 소비자의 NACK을 최소화할 수 있습니다. 브로커에서 메시지를 다시 대기열에 넣는 작업은 CPU를 많이 사용하므로 많은 수의 NACK이 발생하면 브로커 성능에 영향을 줄 수 있습니다.

RabbitMQ용 Amazon MQ의 성능 최적화 및 효율성 모범 사례

처리량을 극대화하고, 지연 시간을 최소화하고, 효율적인 리소스 사용률을 보장하여 RabbitMQ용 Amazon MQ 브로커 성능을 최적화할 수 있습니다. 애플리케이션 성능을 최적화하려면 다음 모범 사례를 완료하세요.

1단계: 메시지 크기를 1MB 미만으로 유지

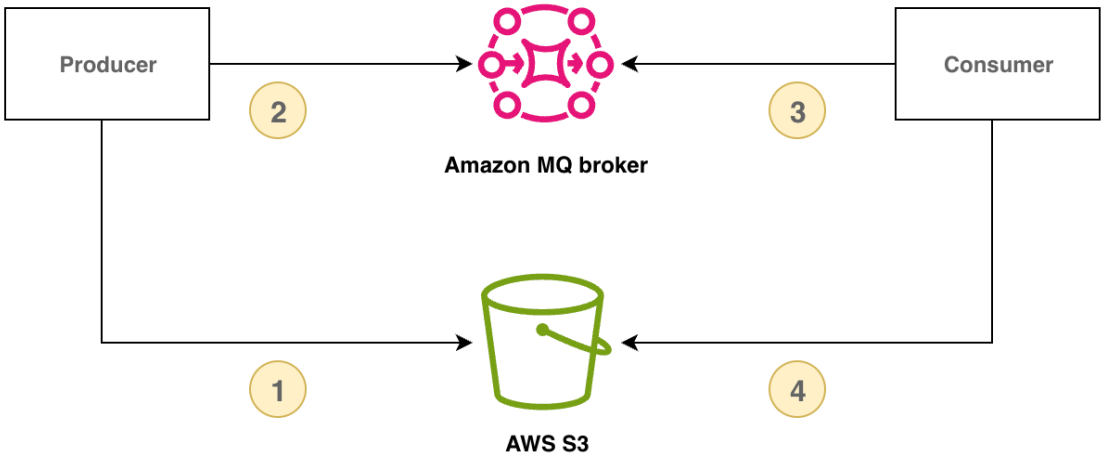
최적의 성능과 신뢰성을 위해 메시지를 1MB(메가바이트) 미만으로 유지하는 것이 좋습니다.

RabbitMQ 3.13은 기본적으로 최대 128MB의 메시지 크기를 지원하지만 대용량 메시지는 노드 간에 메시지를 복제하는 동안 게시를 차단하고 잠재적으로 높은 메모리 압력을 생성하는 예측할 수 없는 메모리 경보를 트리거할 수 있습니다. 메시지가 너무 크면 브로커 재시작 및 복구 프로세스에도 영향을 미칠 수 있으며, 이로 인해 서비스 연속성에 대한 위험이 증가하고 성능 저하가 발생할 수 있습니다.

클레임 검사 패턴을 사용하여 대용량 페이로드 저장 및 검색

대용량 메시지를 관리하려면 메시지 페이로드를 외부 스토리지에 저장하고 RabbitMQ를 통해 페이로드 참조 식별자만 전송하여 클레임 검사 패턴을 구현할 수 있습니다. 소비자는 페이로드 참조 식별자를 사용하여 대용량 메시지를 검색하고 처리합니다.

다음 다이어그램은 RabbitMQ용 Amazon MQ 및 Amazon S3를 사용하여 클레임 확인 패턴을 구현하는 방법을 보여줍니다.



다음 예제에서는 Amazon MQ, AWS SDK for Java 2.x 및 Amazon S3를 사용하여 이 패턴을 보여줍니다.

1. 먼저 Amazon S3 참조 식별자를 포함할 메시지 클래스를 정의합니다.

```
class Message {
    // Other data fields of the message...

    public String s3Key;
    public String s3Bucket;
}
```

2. Amazon S3에 페이로드를 저장하고 RabbitMQ를 통해 참조 메시지를 전송하는 게시자 메서드를 생성합니다.

```
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 S3에서 페이로드를 검색하고, 페이로드를 처리하고, Amazon S3 객체를 삭제하는 소비자 방법을 구현합니다.

```
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());
}
```

2단계: `basic.consume` 및 수명이 긴 소비자 사용

수명이 긴 소비자와 `basic.consume`을 함께 사용하면 `basic.get`을 사용하여 개별 메시지를 폴링하는 것보다 더 효율적입니다. 자세한 내용은 Polling for individual messages를 참조하세요.

3단계: 미리 가져오기 구성

RabbitMQ 미리 가져오기 값을 사용하여 소비자가 메시지를 소비하는 방식을 최적화할 수 있습니다. RabbitMQ는 채널이 아닌 소비자에 미리 가져오기 수를 적용하여 AMQP 0-9-1에서 제공하는 채널 미리 가져오기 메커니즘을 구현합니다. 미리 가져오기 값을 사용하여 지정된 시간에 소비자에게 전송되는 메시지 수를 지정합니다. 기본적으로 RabbitMQ는 클라이언트 애플리케이션에 무제한 버퍼 크기를 설정합니다.

RabbitMQ 소비자의 미리 가져오기 수를 설정할 때는 다양한 요소를 고려해야 합니다. 먼저, 소비자의 환경 및 구성을 고려합니다. 소비자는 처리 중인 모든 메시지를 메모리에 유지해야 하므로 미리 가져오기 값이 높으면 소비자의 성능에 부정적인 영향을 줄 수 있으며 경우에 따라 소비자가 모두 함께 중단될 수도 있습니다. 마찬가지로, RabbitMQ 브로커 자체는 보내는 모든 메시지를 소비자 승인을 받을 때까지 메모리에 캐시된 상태로 유지합니다. 미리 가져오기 값이 높으면 소비자의 자동 승인 구성되어 있

지 않은 경우 및 소비자가 메시지를 처리하는 데 비교적 오래 걸리는 경우 RabbitMQ 서버의 메모리가 빠르게 부족해질 수 있습니다.

위의 고려 사항에 유의하여 RabbitMQ 브로커나 해당 소비자가 많은 수의 처리되지 않거나 승인되지 않은 메시지로 인해 메모리가 부족해지는 상황을 방지하기 위해 항상 미리 가져오기 값을 설정하는 것이 좋습니다. 대량의 메시지를 처리하도록 브로커를 최적화해야 하는 경우 다양한 미리 가져오기 수로 브로커와 소비자를 테스트하여 소비자가 메시지를 처리하는 데 걸리는 시간에 비해 네트워크 오버헤드가 크게 중요하지 않게 되는 값을 결정할 수 있습니다.

> **Note**
>
> - 클라이언트 애플리케이션이 소비자에게 메시지 배달을 자동으로 승인하도록 구성한 경우 미리 가져오기 값을 설정해도 아무런 효과가 없습니다.
> - 미리 가져온 메시지가 모두 대기열에서 제거됩니다.

다음 예제에서는 RabbitMQ Java 클라이언트 라이브러리를 사용하여 단일 소비자의 미리 가져오기 값을 `10`으로 설정하는 방법을 보여줍니다.

```
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 자바 클라이언트 라이브러리에서 `global` 플래그의 기본값은 `false`로 설정되므로 위의 예제는 간단히 `channel.basicQos(10)`으로 작성할 수 있습니다.

4단계: 쿼럼 대기열에서 Celery 5.5 이상 사용

분산 작업 대기열 시스템인 Python Celery는 작업 로드가 높을 때 중요하지 않은 메시지를 많이 생성할 수 있습니다. 이 추가 브로커 활동이 the section called “ `RABBITMQ_MEMORY_ALARM` ”를 트리거하여

브로커를 사용할 수 없게 될 수 있습니다. 메모리 경보를 트리거할 가능성을 줄이려면 다음을 수행합니다.

모든 Celery 버전

1. 대기열 이탈을 완화하려면 `task_create_missing_queues`을 끕니다.
2. 그런 다음 `worker_enable_remote_control`을 꺼서 `celery@...pidbox` 대기열의 동적 생성을 중지합니다. 이렇게 하면 브로커의 대기열 이탈이 줄어듭니다.

```
worker_enable_remote_control = false
```

3. 중요하지 않은 메시지 활동을 더욱 줄이려면 Celery 애플리케이션을 시작할 때 `-E` 또는 `--task-events` 플래그를 포함하지 않아 Celery worker-send-task-events를 끕니다.
4. 다음 파라미터를 사용하여 Celery 애플리케이션을 시작합니다.

```
celery -A app_name worker --without-heartbeat --without-gossip --without-mingle
```

Celery 버전 5.5 이상의 경우

1. 쿼럼 대기열을 지원하는 최소 버전인 Celery 버전 5.5 또는 그 이후 버전으로 업그레이드합니다. 사용 중인 Celery 버전을 확인하려면 `celery --version`을 사용합니다. 쿼럼에 대한 자세한 내용은 the section called “쿼럼 대기열” 섹션을 참조하세요.
2. Celery 5.5 이상으로 업그레이드한 후 `task_default_queue_type`을 "quorum"으로 구성합니다.
3. 그런 다음 브로커 전송 옵션에서 게시 확인도 켜야 합니다.

```
broker_transport_options = {"confirm_publish": True}
```

RabbitMQ용 Amazon MQ의 네트워크 복원력 및 모니터링 모범 사례

네트워크 복원력 및 모니터링 브로커 지표는 안정적인 메시징 애플리케이션을 유지하는 데 필수적입니다. 다음 모범 사례를 완료하여 자동 복구 메커니즘 및 리소스 모니터링 전략을 구현합니다.

1단계: 네트워크 실패 자동 복구

RabbitMQ 노드에 대한 클라이언트 연결이 실패할 경우 상당한 가동 중지를 방지하기 위해 자동 네트워크 복구를 항상 활성화하는 것이 좋습니다. RabbitMQ Java 클라이언트 라이브러리는 `4.0.0` 버전부터 자동 네트워크 복구를 기본적으로 지원합니다.

연결의 I/O 루프에서 처리되지 않은 예외가 발생하거나, 소켓 읽기 작업 시간 초과가 감지되거나, 서버가 하트비트를 놓치는 경우 자동 연결 복구가 트리거됩니다.

클라이언트와 RabbitMQ 노드 간의 초기 연결이 실패하는 경우에는 자동 복구가 트리거되지 않습니다. 연결을 다시 시도하여 초기 연결 실패를 해결하도록 애플리케이션 코드를 작성하는 것이 좋습니다. 다음 예제에서는 RabbitMQ Java 클라이언트 라이브러리를 사용하여 초기 네트워크 실패를 다시 시도하는 방법을 보여 줍니다.

```
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**
>
> 애플리케이션에서 `Connection.Close` 메서드를 사용하여 연결을 종료하면 자동 네트워크 복구가 활성화되거나 트리거되지 않습니다.

2단계: 브로커 지표 및 경보 모니터링

RabbitMQ용 Amazon MQ 브로커에 대한 CloudWatch 지표 및 경보를 정기적으로 모니터링하여 메시징 애플리케이션에 영향을 미치기 전에 잠재적 문제를 식별하고 해결하는 것이 좋습니다. 탄력적 메시징 애플리케이션을 유지하고 최적의 성능을 보장하려면 사전 예방적 모니터링이 필수적입니다.

RabbitMQ용 Amazon MQ는 브로커 성능, 리소스 사용률 및 메시지 흐름에 대한 인사이트를 제공하는 지표를 CloudWatch에 게시합니다. 모니터링할 주요 지표에는 메모리 사용량과 디스크 사용량이 포함

됩니다. 브로커가 리소스 제한에 가까워지거나 성능 저하가 발생할 때 CloudWatch 경보를 설정할 수 있습니다.

다음 필수 지표를 모니터링합니다.

`RabbitMQMemUsed` 및 `RabbitMQMemLimit`

메모리 사용량을 모니터링하여 메시지 게시를 차단할 수 있는 메모리 경보를 방지합니다.

`RabbitMQDiskFree` 및 `RabbitMQDiskFreeLimit`

디스크 사용량을 모니터링하여 브로커 실패를 일으킬 수 있는 디스크 공간 문제를 방지합니다.

클러스터 배포의 경우 노드별 지표도 모니터링하여 노드별 문제를 식별합니다.

> **Note**
>
> 고용량 메모리 경보를 방지하는 방법에 대한 자세한 내용은 높은 메모리 경보 해결 및 방지를 참조하세요.

RabbitMQ 자습서

다음 자습서는 Amazon MQ에서 RabbitMQ를 구성하고 사용하는 방법을 보여줍니다. Node.js, Python, .NET 등과 같은 다양한 프로그래밍 언어로 지원되는 클라이언트 라이브러리를 사용하는 방법에 대한 자세한 내용은 RabbitMQ 시작 안내서에서 RabbitMQ 자습서를 참조하세요.

주제

- 브로커 기본 설정 편집
- RabbitMQ용 Amazon MQ와 함께 Python Pika 사용
- RabbitMQ 일시 중지된 대기열 동기화 문제 해결
- 연결 및 채널 수 축소
- 2단계: 브로커에 JVM 기반 애플리케이션 연결
- 3단계: (선택 사항) AWS Lambda 함수에 연결
- RabbitMQ용 Amazon MQ에 OAuth 2.0 인증 및 권한 부여 사용
- RabbitMQ용 Amazon MQ에 대한 IAM 인증 및 권한 부여 사용

- RabbitMQ용 Amazon MQ에 LDAP 인증 및 권한 부여 사용
- RabbitMQ용 Amazon MQ에 HTTP 인증 및 권한 부여 사용
- RabbitMQ용 Amazon MQ에 SSL 인증서 인증 사용
- AMQP 및 관리 엔드포인트에 mTLS 사용
- JMS 애플리케이션 연결

브로커 기본 설정 편집

를 사용하여 CloudWatch 로그 활성화 또는 비활성화와 같은 브로커 기본 설정을 편집할 수 있습니다 AWS Management Console.

RabbitMQ 브로커 옵션 편집

1. Amazon MQ 콘솔에 로그인합니다.
2. 브로커 목록에서 브로커(예: MyBroker)를 선택한 다음 Edit(편집)을 선택합니다.
3. Edit ***MyBroker***(MyBroker 편집) 페이지의 Specifications(사양) 섹션에서 Broker engine version(브로커 엔진 버전) 또는 Broker Instance type(브로커 인스턴스 유형)을 선택합니다.
4. CloudWatch Logs 섹션에서 토글 버튼을 클릭하여 일반 로그를 활성화하거나 비활성화합니다. 다른 단계는 필요하지 않습니다.

   > **Note**
   >
   > - RabbitMQ 브로커의 경우 Amazon MQ는 자동으로 서비스 연결 역할(SLR)을 사용하여 CloudWatch 에 일반 로그를 게시합니다. 자세한 내용은 the section called “서비스 연결 역할 사용” 단원을 참조하세요.
   > - Amazon MQ는 RabbitMQ 브로커에 대한 감사 로깅을 지원하지 않습니다.

5. Maintenance(유지 관리) 섹션에서 브로커의 유지 관리 일정을 구성합니다.

   AWS 릴리스 시 브로커를 새 버전으로 업그레이드하려면 자동 마이너 버전 업그레이드 활성화를 선택합니다. 자동 업그레이드는 요일, 시간(24시간 형식) 및 시간대(기본값은 UTC)에 의해 정의된 유지 관리 기간 도중에 발생합니다.
6. Schedule modifications(수정 예약)를 선택합니다.

> **Note**
>
> Enable automatic minor version upgrades(마이너 버전 자동 업그레이드 활성화)만 선택하는 경우 브로커 재부팅이 필요하지 않기 때문에 버튼이 Save(저장)로 바뀝니다.

지정된 시간에 기본 설정이 브로커에 적용됩니다.

RabbitMQ용 Amazon MQ와 함께 Python Pika 사용

다음 자습서에서는 RabbitMQ용 Amazon MQ 브로커에 연결하도록 구성된 TLS를 사용하여 Python Pika 클라이언트를 설정하는 방법을 보여줍니다. Pika는 RabbitMQ용 AMQP 0-9-1 프로토콜의 Python 구현입니다. 이 자습서에서는 Pika 설치, 대기열 선언, 브로커의 기본 교환으로 메시지를 전송하도록 게시자 설정, 대기열에서 메시지를 수신하도록 소비자 설정을 안내합니다.

주제

사전 조건

이 자습서를 완료하려면 다음과 같은 사전 조건이 필요합니다.

- RabbitMQ용 Amazon MQ 브로커 자세한 내용은 RabbitMQ용 Amazon MQ 브로커 생성을 참조하세요.
- 운영 체제에 맞춰 설치된 Python 3
- Python `pip`를 사용하여 설치된 Pika Pika를 설치하려면 새 터미널 창을 열고 다음을 실행합니다.

```
$ python3 -m pip install pika
```

권한

이 자습서를 진행하려면 vhost에 대한 쓰기 및 읽기 권한이 있는 RabbitMQ용 Amazon MQ 브로커 사용자가 한 명 이상 필요합니다. 다음 표에서는 필요한 최소 권한을 정규식(regexp) 패턴으로 설명합니다.

Tags	regexp 구성	regexp 쓰기	regexp 읽기
none		`.*`	`.*`

나열된 사용자 권한은 사용자에게 읽기 및 쓰기 권한만 제공합니다. 브로커에 대한 관리 작업을 수행할 수 있는 관리 플러그 인에 대한 액세스 권한은 부여되지 않습니다. 사용자의 액세스를 지정된 대기열로 제한하는 정규 표현식 패턴을 사용하여 권한을 추가적으로 제한할 수 있습니다. 예를 들어 읽기 정규 표현식 패턴을 `^[hello world].*`로 변경하면 해당 사용자에게는 `hello world`로 시작하는 대기열에서 읽을 수 있는 권한만 부여됩니다.

RabbitMQ 사용자를 생성하고 사용자 태그 및 권한을 관리하는 방법에 대한 자세한 내용은 RabbitMQ용 Amazon MQ 브로커 사용자 단원을 참조하세요.

1단계: 기본 Python Pika 클라이언트 생성

RabbitMQ용 Amazon MQ 브로커와 상호 작용할 때 생성자를 정의하고 TLS 구성에 필요한 SSL 컨텍스트를 제공하는 Python Pika 클라이언트 기본 클래스를 생성하려면 다음을 수행합니다.

1. 새 터미널 창을 열고 프로젝트에 대한 새 디렉터리를 생성한 다음, 해당 디렉터리로 이동합니다.

```
$ mkdir pika-tutorial
$ cd pika-tutorial
```

2. 다음의 Python 코드가 포함된 `basicClient.py`라는 이름의 파일을 생성합니다.

```
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()
```

이제 `BasicPikaClient`에서 상속된 게시자 및 소비자에 대한 추가 클래스를 정의할 수 있습니다.

2단계: 게시자 생성 및 메시지 전송

대기열을 선언하는 게시자를 생성하고 단일 메시지를 전송하려면 다음을 수행합니다.

1. 다음 코드 샘플의 내용을 복사하고 로컬의 이전 단계에서 생성한 것과 동일한 디렉터리에 `publisher.py`로 저장합니다.

```
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 클래스는 BasicPikaClient에서 상속하여 대기열을 선언하고, 대기열에 메시지를 전송하고, 연결을 닫는 추가 메서드를 구현합니다. 코드 샘플은 대기열 이름과 동일한 라우팅 키를 사용하여 메시지를 기본 교환으로 라우팅합니다.

2. `if __name__ == "__main__":` 아래에서 다음 정보가 포함된 BasicMessageSender 생성자 문으로 전달되는 파라미터를 교체합니다.

   - **<broker-id>** - Amazon MQ가 브로커에 대해 생성하는 고유한 ID입니다. 브로커 ARN에서 ID를 구문 분석할 수 있습니다. 예를 들어 ARN이 arn:aws:mq:us-east-2:123456789012:broker:MyBroker:b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9인 경우 브로커 ID는 b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9입니다.
   - **<username>** - 브로커에 메시지를 쓸 수 있는 충분한 권한이 있는 브로커 사용자의 사용자 이름입니다.
   - **<password>** - 브로커에 메시지를 쓸 수 있는 충분한 권한이 있는 브로커 사용자의 암호입니다.
   - **<region>** - RabbitMQ용 Amazon MQ 브로커를 생성한 AWS 리전입니다. RabbitMQ 예를 들어 us-west-2입니다.

3. publisher.py를 생성한 것과 동일한 디렉터리에서 다음 명령을 실행합니다.

```
$ python3 publisher.py
```

코드가 성공적으로 실행되면 터미널 창에 다음과 같이 출력됩니다.

```
Trying to declare queue(hello world queue)...
Sent message. Exchange: , Routing Key: hello world queue, Body: b'Hello World!'
```

3단계: 소비자 생성 및 메시지 수신

대기열에서 단일 메시지를 수신하는 소비자를 생성하려면 다음을 수행합니다.

1. 다음 코드 샘플의 내용을 복사하고 로컬의 동일한 디렉터리에 `consumer.py`로 저장합니다.

```
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()
```

이전 단계에서 생성한 게시자와 마찬가지로는에서 BasicMessageReceiver 상속BasicPikaClient하고 단일 메시지를 수신하고 연결을 닫기 위한 추가 방법을 구현합니다.

2. if __name__ == "__main__": 문 아래에서 BasicMessageReceiver 생성자로 전달되는 파라미터를 사용자의 정보로 교체합니다.
3. 프로젝트 디렉터리에서 다음 명령을 실행합니다.

```
$ python3 consumer.py
```

코드가 성공적으로 실행되면 메시지 본문과 라우팅 키를 포함한 헤더가 터미널 창에 표시됩니다.

```
<Basic.GetOk(['delivery_tag=1', 'exchange=', 'message_count=0',
 'redelivered=False', 'routing_key=hello world queue'])> <BasicProperties> b'Hello
 World!'
```

4단계: (선택 사항) 이벤트 루프 설정 및 메시지 소비

대기열에서 여러 메시지를 소비하려면 다음과 같이 Pika의 basic_consume 메서드 및 콜백 함수를 사용합니다.

1. consumer.py에서 BasicMessageReceiver 클래스에 다음 메서드 정의를 추가합니다.

```
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.py의 if __name__ == "__main__": 아래에서 이전 단계에서 정의한 consume_messages 메서드를 호출합니다.

```
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`를 다시 실행하고, 성공하면 대기 중인 메시지가 터미널 창에 표시됩니다.

```
[*] Waiting for messages. To exit press CTRL+C
[x] Received b'Hello World!'
[x] Received b'Hello World!'
...
```

다음 단계

- 지원되는 기타 RabbitMQ 클라이언트 라이브러리에 대한 자세한 내용은 RabbitMQ 웹 사이트에서 RabbitMQ 클라이언트 설명서를 참조하세요.

RabbitMQ 일시 중지된 대기열 동기화 문제 해결

RabbitMQ용 Amazon MQ 클러스터 배포에서 각 대기열에 게시된 메시지는 세 개의 브로커 노드 간에 복제됩니다. 미러링이라고 하는 이 복제는 RabbitMQ 브로커에 고가용성(HA)을 제공합니다. 클러스터 배포의 대기열은 한 노드의 기본 복제본과 하나 이상의 미러로 구성됩니다. 미러링된 대기열에 적용되는 메시지 대기열에 넣기를 비롯한 모든 작업은 먼저 기본 대기열에 적용된 다음 해당 미러 전체에 복제됩니다.

예를 들어 미러링된 대기열이 기본 노드(`main`)와 두 개 미러(`mirror-1` 및 `mirror-2`)의 세 개 노드에서 복제된다고 가정합니다. 이 미러링된 대기열의 모든 메시지가 모든 미러에 전파되면 대기열이 동기화됩니다. 노드(`mirror-1`)가 일정 시간 사용할 수 없게 되더라도 대기열은 작동하고 계속 메시지를 대기열에 넣을 수 있습니다. 그러나 대기열을 동기화하려면 `mirror-1`을 사용할 수 없는 동안 `main`에 게시된 메시지를 `mirror-1`에 복제해야 합니다.

미러링에 대한 자세한 내용은 RabbitMQ 웹 사이트에서 클래식 미러링된 대기열을 참조하세요.

유지 관리 및 대기열 동기화

유지 관리 기간 중 Amazon MQ는 모든 유지 관리 작업을 한 번에 한 노드에서 수행하여 브로커가 계속 작동하도록 합니다. 따라서 각 노드의 작동이 다시 시작할 때 대기열을 동기화해야 할 수 있습니다. 동기화하는 동안 미러에 복제되어야 하는 메시지는 해당하는 Amazon Elastic Block Store(Amazon EBS) 볼륨에서 메모리로 로드되어 배치로 처리됩니다. 메시지를 배치로 처리하면 대기열을 더 빠르게 동기화할 수 있습니다.

대기열이 짧게 유지되고 메시지가 작으면 대기열이 성공적으로 동기화되고 예상대로 작업이 재개됩니다. 그러나 배치의 데이터 양이 노드의 메모리 한도에 가까워지면 노드에서 고용량 메모리 경보가 발생하여 대기열 동기화가 일시 중지됩니다. 메모리 사용량은 CloudWatch의 브로커 노드 지표 `RabbitMemUsed`와 `RabbitMqMemLimit`를 비교하여 확인할 수 있습니다. 메시지가 사용 또는 삭제되거나 배치의 메시지 수가 줄지 않으면 동기화가 완료될 수 없습니다.

> **Note**
>
> 대기열 동기화 배치 크기를 줄이면 복제 트랜잭션 수가 더 많아질 수 있습니다.

일시 중지된 대기열 동기화 문제를 해결하려면 `ha-sync-batch-size` 정책을 적용하고 대기열 동기화를 다시 시작하는 방법을 보여주는 이 자습서의 단계를 따릅니다.

주제

- 사전 조건
- 1단계: ha-sync-batch-size 정책 적용
- 2단계: 대기열 동기화 다시 시작
- 다음 단계
- 관련 리소스

사전 조건

이 자습서를 사용하려면 관리자 권한이 있는 RabbitMQ용 Amazon MQ 브로커 사용자가 있어야 합니다. 브로커를 처음 생성할 때 만든 관리자 사용자를 사용하거나 나중에 만들었을 수 있는 다른 사용자를 사용할 수 있습니다. 다음 표에는 필요한 관리자 사용자 태그 및 권한이 정규식(regexp) 패턴으로 나와 있습니다.

Tags	regexp 읽기	regexp 구성	regexp 쓰기
`administrator`	`.*`	`.*`	`.*`

RabbitMQ 사용자를 생성하고 사용자 태그 및 권한을 관리하는 방법에 대한 자세한 내용은 RabbitMQ용 Amazon MQ 브로커 사용자 단원을 참조하세요.

1단계: **`ha-sync-batch-size`** 정책 적용

다음 절차에서는 브로커에서 생성되는 모든 대기열에 적용되는 정책을 추가하는 방법을 보여줍니다. RabbitMQ 웹 콘솔이나 RabbitMQ 관리 API를 사용할 수 있습니다. 자세한 내용은 RabbitMQ 웹 사이트에서 관리 플러그 인을 참조하세요.

RabbitMQ 웹 콘솔을 사용하여 **`ha-sync-batch-size`** 정책을 적용하려면

1. Amazon MQ 콘솔에 로그인합니다.
2. 왼쪽 탐색 창에서 Brokers(브로커)를 선택합니다.
3. 브로커 목록에서 새 정책을 적용할 브로커의 이름을 선택합니다.
4. 브로커 페이지의 Connections(연결) 섹션에서 RabbitMQ 웹 콘솔 URL을 선택합니다. RabbitMQ 웹 콘솔이 새 브라우저 탭 또는 창에 열립니다.
5. 브로커 관리자 로그인 보안 인증 정보를 사용하여 RabbitMQ 웹 콘솔에 로그인합니다.
6. RabbitMQ 웹 콘솔의 페이지 상단에서 Admin(관리자)을 선택합니다.
7. Admin(관리자) 페이지의 오른쪽 탐색 창에서 Policies(정책)를 선택합니다.
8. Policies(정책) 페이지에서 브로커의 현재 User policies(사용자 정책) 목록을 볼 수 있습니다. User policies(사용자 정책) 아래에서 Add / update a policy(정책 추가/업데이트)를 확장합니다.

> **Note**
>
> 기본적으로 RabbitMQ용 Amazon MQ 클러스터는 `ha-all-AWS-OWNED-DO-NOT-DELETE`라는 초기 브로커 정책과 함께 생성됩니다. Amazon MQ는 브로커의 모든 대기열이 세 노드 모두에 복제되고 대기열이 자동으로 동기화되도록 이 정책을 관리합니다.

9. 새 브로커 정책을 생성하려면 Add / update a policy(정책 추가/업데이트)에서 다음을 수행합니다.

   a. Name(이름)에 정책의 이름을 입력합니다(예: **`batch-size-policy`**).

   b. Pattern(패턴)에 regexp 패턴 **`.*`**를 입력합니다. 이 경우 정책이 브로커의 모든 대기열과 일치합니다.

   c. Apply to(적용 대상)의 드롭다운 목록에서 Exchanges and queues(교환 및 대기열)를 선택합니다.

   d. Priority(우선 순위)에 vhost에 적용된 다른 모든 정책보다 큰 정수를 입력합니다. 지정된 시간에 RabbitMQ 대기열 및 교환에 정확히 하나의 정책 정의 집합을 적용할 수 있습니다. RabbitMQ는 가장 높은 우선 순위 값과 일치하는 정책을 선택합니다. 정책 우선 순위 및 정책을 결합하는 방법에 대한 자세한 내용은 RabbitMQ Server 설명서에서 정책을 참조하세요.

   e. Definition(정의)에 다음 키-값 페어를 추가합니다.

      - **`ha-sync-batch-size`**=*`100`*. 드롭다운 목록에서 Number(숫자)를 선택합니다.

        > **Note**
        >
        > 대기열에 있는 동기화되지 않은 메시지 수와 크기를 기준으로 `ha-sync-batch-size` 값을 조정하고 보정해야 할 수 있습니다.

      - **`ha-mode=all`**. 드롭다운 목록에서 String(문자열)을 선택합니다.

        > **Important**
        >
        > `ha-mode` 정의는 모든 HA 관련 정책에 필요합니다. 생략하면 검증이 실패합니다.

      - **`ha-sync-mode=automatic`**. 드롭다운 목록에서 String(문자열)을 선택합니다.

> **Note**
>
> ha-sync-mode 정의는 모든 사용자 지정 정책에 필요합니다. 생략하면 Amazon MQ가 자동으로 정의를 추가합니다.

f. Add / update policy(정책 추가/업데이트)를 선택합니다.

10. 새 정책이 User policies(사용자 정책)의 목록에 표시되는지 확인합니다.

RabbitMQ 관리 API를 사용하여 **ha-sync-batch-size** 정책을 적용하려면

1. Amazon MQ 콘솔에 로그인합니다.
2. 왼쪽 탐색 창에서 Brokers(브로커)를 선택합니다.
3. 브로커 목록에서 새 정책을 적용할 브로커의 이름을 선택합니다.
4. 브로커 페이지의 Connections(연결) 섹션에서 RabbitMQ 웹 콘솔 URL을 기록합니다. 이는 HTTP 요청에서 사용하는 브로커 엔드포인트입니다.
5. 새 터미널 또는 원하는 명령줄 창을 엽니다.
6. 새 브로커 정책을 생성하려면 다음 curl 명령을 입력합니다. 이 명령에서는 대기열이 %2F로 인코딩된 기본 / vhost에 있다고 가정합니다.

> **Note**
>
> *### ##*과 *####*를 브로커 관리자 로그인 보안 인증 정보로 바꾸세요. 대기열에 있는 동기화되지 않은 메시지 수와 크기를 기준으로 ha-sync-batch-size 값(*100*)을 조정하고 보정해야 할 수 있습니다. 브로커 엔드포인트를 이전에 기록한 URL로 바꿉니다.

```
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. 브로커의 사용자 정책에 새 정책이 추가되었는지 확인하려면 다음 curl 명령을 입력하여 모든 브로커 정책을 나열합니다.

```
curl -i -u username:password https://b-589c045f-f8ln-4ab0-a89c-co62e1c32ef8.mq.us-west-2.amazonaws.com/api/policies
```

2단계: 대기열 동기화 다시 시작

브로커에 새 `ha-sync-batch-size` 정책을 적용한 후 대기열 동기화를 다시 시작합니다.

RabbitMQ 웹 콘솔을 사용하여 대기열 동기화를 다시 시작하려면

> **Note**
>
> RabbitMQ 웹 콘솔을 열려면 이 자습서 1단계의 이전 지침을 참조하세요.

1. RabbitMQ 웹 콘솔의 페이지 상단에서 Queues(대기열)를 선택합니다.
2. Queues(대기열) 페이지의 All queues(모든 대기열)에서 일시 중지된 대기열을 찾습니다. 정책 열에서 생성한 새 정책의 이름(예: `batch-size-policy`)이 대기열에 표시되어야 합니다.
3. 줄어든 배치 크기로 동기화 프로세스를 다시 시작하려면 먼저 대기열 동기화를 취소합니다. 그런 다음 대기열 동기화를 다시 시작합니다.

> **Note**
>
> 동기화가 일시 중지되고 성공적으로 완료되지 않으면 `ha-sync-batch-size` 값을 줄이고 대기열 동기화를 다시 시작해 보세요.

다음 단계

- 대기열이 동기화되면 Amazon CloudWatch 지표 `RabbitMQMemUsed`를 확인하여 RabbitMQ 노드에서 사용하는 메모리 양을 모니터링할 수 있습니다. 또한 `RabbitMQMemLimit` 지표를 확인하여 노드의 메모리 제한을 모니터링할 수 있습니다. 자세한 내용은 Amazon MQ의 CloudWatch 지표 액세스 및 RabbitMQ용 Amazon MQ 브로커에 사용 가능한 CloudWatch 지표 단원을 참조하세요.
- 대기열 동기화 일시 중지를 방지하려면 대기열을 짧게 유지하고 메시지를 처리하는 것이 좋습니다. 메시지 크기가 더 큰 워크로드의 경우 브로커 인스턴스 유형을 메모리가 더 많은 더 큰 인스턴스 크기로 업그레이드하는 것이 좋습니다. 브로커 인스턴스 유형 및 브로커 기본 설정 편집에 대한 자세한 내용은 브로커 기본 설정 편집 섹션을 참조하세요.

- 새 RabbitMQ용 Amazon MQ 브로커를 생성할 때 Amazon MQ는 브로커 성능을 최적화하기 위해 기본 집합의 브로커 정책 및 가상 호스트 제한을 적용합니다. 브로커에 권장되는 기본 정책 및 제한이 없는 경우 직접 생성하는 것이 좋습니다. 기본 정책 및 vhost 제한을 생성하는 방법에 대한 자세한 내용은 https://docs.aws.amazon.com//amazon-mq/latest/developer-guide/rabbitmq-defaults.html 단원을 참조하세요.

관련 리소스

- UpdateBrokerInput - Amazon MQ API를 사용하여 브로커 인스턴스 유형을 업데이트하려면 이 브로커 속성을 사용합니다.
- 파라미터 및 정책(RabbitMQ Server 설명서) - RabbitMQ 웹 사이트에서 RabbitMQ 파라미터 및 정책에 대해 자세히 알아봅니다.
- RabbitMQ 관리 HTTP API - RabbitMQ 관리 API에 대해 자세히 알아봅니다.

연결 및 채널 수 축소

Amazon MQ 브로커용 RabbitMQ와의 연결은 클라이언트 애플리케이션에서 닫거나 RabbitMQ 웹 콘솔을 사용하여 수동으로 닫을 수 있습니다. RabbitMQ 웹 콘솔을 사용하여 연결을 닫으려면 다음을 수행하세요.

1. 에 로그인 AWS Management Console 하고 브로커의 RabbitMQ 웹 콘솔을 엽니다.
2. RabbitMQ 콘솔에서 연결(Connections) 탭을 선택합니다.
3. 연결 페이지의 모든 연결(All connections) 목록에서 닫을 연결의 이름을 선택합니다.
4. 연결 세부 정보 페이지에서 이 연결 닫기(Close this connection)를 선택하여 섹션을 확장한 다음 강제 닫기(Force Close)를 선택합니다. 필요에 따라 직접 입력한 설명을 추가하여 이유(Reason)의 기본 텍스트를 변경할 수 있습니다. RabbitMQ용 Amazon MQ는 연결을 닫을 때 사용자가 지정한 이유를 클라이언트에 반환합니다.
5. 대화 상자에서 확인(OK)을 선택하여 연결을 확인하고 닫습니다.

연결을 닫으면 닫힌 연결과 연결된 모든 채널도 닫힙니다.

> **Note**
>
> 클라이언트 애플리케이션이 닫힌 후 브로커에 대한 연결을 자동으로 다시 설정하도록 구성할 수 있습니다. 이 경우 브로커 웹 콘솔에서 연결을 닫는 것만으로는 연결 또는 채널 수를 줄이는 데 충분하지 않습니다.

퍼블릭 액세스가 없는 브로커의 경우, 적절한 메시지 프로토콜 포트(예: AMQP 연결용 5671 포트)에서 인바운드 트래픽을 거부하여 연결을 일시적으로 차단할 수 있습니다. 브로커를 생성할 때 Amazon MQ에 제공한 보안 그룹의 포트를 차단할 수 있습니다. 보안 그룹 수정에 대한 자세한 내용은 Amazon VPC 사용 설명서의 보안 그룹에 규칙 추가를 참조하세요.

2단계: 브로커에 JVM 기반 애플리케이션 연결

RabbitMQ 브로커를 생성한 후 애플리케이션을 브로커에 연결할 수 있습니다. 다음 예제에서는 RabbitMQ Java 클라이언트 라이브러리를 사용하여 브로커에 대한 연결을 생성하고, 대기열을 생성하고, 메시지를 전송하는 방법을 보여줍니다. 다양한 언어로 지원되는 RabbitMQ 클라이언트 라이브러리를 사용하여 RabbitMQ 브로커에 연결할 수 있습니다. 지원되는 RabbitMQ 클라이언트 라이브러리에 대한 자세한 내용은 RabbitMQ 클라이언트 라이브러리 및 개발자 도구를 참조하세요.

사전 조건

> **Note**
>
> 다음의 사전 필수 단계는 퍼블릭 액세스 가능성 없이 생성된 RabbitMQ 브로커에만 적용됩니다. 퍼블릭 액세스 가능성이 있는 브로커를 생성하는 경우에는 건너뛸 수 있습니다.

VPC 속성 활성화

VPC 내에서 브로커에 액세스할 수 있도록 하려면 `enableDnsHostnames` 및 `enableDnsSupport` VPC 속성을 활성화해야 합니다. 자세한 내용은 Amazon VPC 사용 설명서의 VPC에서 DNS 지원을 참조하세요.

인바운드 연결 활성화

1. Amazon MQ 콘솔에 로그인합니다.
2. 브로커 목록에서 브로커 이름(예: MyBroker)을 선택합니다.

3. ***MyBroker*** 페이지의 Connections(연결) 섹션에서 브로커의 웹 콘솔 URL 및 와이어 레벨 프로토콜의 주소 및 포트를 적어둡니다.
4. 세부 정보 섹션의 보안 및 네트워크에서 보안 그룹의 이름 또는  을 선택합니다.

   EC2 Dashboard의 보안 그룹 페이지가 표시됩니다.
5. 보안 그룹 목록에서 보안 그룹을 선택합니다.
6. 페이지 하단에서 인바운드를 선택한 후 편집을 선택합니다.
7. Edit inbound rules(인바운드 규칙 편집) 대화 상자에서 공개적으로 액세스하고자 하는 모든 URL 또는 엔드포인트에 대한 규칙을 추가합니다. 다음 예제에서는 브로커 웹 콘솔에 대해 이 작업을 수행하는 방법을 보여줍니다.

   a. 규칙 추가(Add Rule)를 선택합니다.
   b. 유형에서 Custom TCP(사용자 지정 TCP)를 선택합니다.
   c. Source(소스)에서 Custom(사용자 지정)을 선택한 상태에서 웹 콘솔에 액세스하는 데 사용할 시스템의 IP 주소(예: `192.0.2.1`)를 입력합니다.
   d. 저장을 선택합니다.

      이제 브로커가 인바운드 연결을 허용할 수 있습니다.

Java 종속성 추가

빌드 자동화를 위해 Apache Maven을 사용하는 경우 `pom.xml` 파일에 다음 종속성을 추가합니다. Apache Maven의 Project Object Model 파일에 대한 자세한 내용은 POM 소개를 참조하세요.

```
<dependency>
    <groupId>com.rabbitmq</groupId>
    <artifactId>amqp-client</artifactId>
    <version>5.9.0</version>
</dependency>
```

빌드 자동화에 Gradle을 사용하는 경우 다음 종속성을 선언합니다.

```
dependencies {
    compile 'com.rabbitmq:amqp-client:5.9.0'
}
```

Connection 및 **Channel** 클래스 가져오기

RabbitMQ Java 클라이언트는 각각 AMQP 0-9-1 연결 및 채널을 나타내는 Connection 및 Channel API 클래스가 포함된 com.rabbitmq.client를 최상위 패키지로 사용합니다. Connection 및 Channel 클래스를 사용하려면 다음 예제와 같이 먼저 가져옵니다.

```
import com.rabbitmq.client.Connection;
import com.rabbitmq.client.Channel;
```

ConnectionFactory 생성 및 브로커에 연결

다음 예제에 따라 지정된 파라미터를 사용하여 ConnectionFactory 클래스의 인스턴스를 생성합니다. setHost 메서드를 사용하여 앞서 기록한 브로커 엔드포인트를 구성합니다. AMQPS 와이어 레벨 연결에는 포트 5671을 사용합니다.

```
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();
```

교환에 메시지 게시

Channel.basicPublish를 사용하여 메시지를 교환에 게시할 수 있습니다. 다음 예제에서는 AMQP Builder 클래스를 사용하여 콘텐츠 유형이 plain/text인 메시지 속성 객체를 빌드합니다.

```
byte[] messageBodyBytes = "Hello, world!".getBytes();
channel.basicPublish(exchangeName, routingKey,
             new AMQP.BasicProperties.Builder()
               .contentType("text/plain")
```

```
                .userId("userId")
                .build(),
                messageBodyBytes);
```

> **Note**
>
> BasicProperties는 자동 생성된 소유자 클래스의 내부 클래스인 AMQP입니다.

대기열 구독 및 메시지 수신

Consumer 인터페이스를 통해 대기열을 구독하여 메시지를 수신할 수 있습니다. 구독하면 도착하는 메시지가 자동으로 배달됩니다.

Consumer를 구현하는 가장 쉬운 방법은 하위 클래스 DefaultConsumer를 사용하는 것입니다. 다음 예제와 같이 DefaultConsumer 객체를 basicConsume 호출의 일부로 전달하여 구독을 설정할 수 있습니다.

```
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 = false를 지정했으므로 Consumer에 배달된 메시지를 승인해야 하며, 예제와 같이 handleDelivery 메서드에서 수행하면 가장 간편합니다.

연결 닫기 및 브로커와 연결 끊기

RabbitMQ 브로커와 연결을 끊으려면 다음과 같이 채널과 연결을 모두 닫습니다.

```
channel.close();
conn.close();
```

> **Note**
>
> RabbitMQ Java 클라이언트 라이브러리를 사용하는 방법에 대한 자세한 내용은 RabbitMQ Java 클라이언트 API 가이드를 참조하세요.

3단계: (선택 사항) AWS Lambda 함수에 연결

AWS Lambda 는 Amazon MQ 브로커의 메시지에 연결하고 사용할 수 있습니다. 브로커를 Lambda에 연결할 때는 대기열에서 메시지를 읽고 함수를 동기식으로 호출하는 이벤트 소스 매핑을 생성합니다. 생성하는 이벤트 소스 매핑은 메시지를 브로커에서 배치로 읽고 JSON 객체 형식의 Lambda 페이로드로 변환합니다.

브로커를 Lambda 함수에 연결하려면

1. Lambda 함수 실행 역할에 다음 IAM 역할 권한을 추가합니다.

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

> **Note**
>
> 필요한 IAM 권한이 없으면 함수가 Amazon MQ 리소스의 레코드를 성공적으로 읽을 수 없습니다.

2. (선택 사항) 퍼블릭 액세스 가능성이 없는 브로커를 생성한 경우 다음 중 하나를 수행하여 Lambda가 브로커에 연결하도록 허용해야 합니다.

   - 퍼블릭 서브넷당 하나의 NAT 게이트웨이를 구성합니다. 자세한 내용은 AWS Lambda 개발자 안내서에서 VPC 연결 함수의 인터넷 및 서비스 액세스를 참조하세요.
   - VPC 엔드포인트를 사용하여 Amazon Virtual Private Cloud(Amazon VPC)와 Lambda 간의 연결을 생성합니다. Amazon VPC는 AWS Security Token Service (AWS STS) 및 Secrets Manager 엔드포인트에도 연결해야 합니다. 자세한 내용은 AWS Lambda 개발자 안내서에서 Lambda에 대한 인터페이스 VPC 엔드포인트 구성을 참조하세요.

3. AWS Management Console을 사용하여 Lambda 함수에 대해 브로커를 이벤트 소스로 구성합니다. `create-event-source-mapping` AWS Command Line Interface 명령을 사용할 수도 있습니다.

4. 브로커에서 소비한 메시지를 Lambda 함수가 처리하는 코드를 작성합니다. 이벤트 소스 매핑에서 검색되는 Lambda 페이로드는 브로커의 엔진 유형에 따라 다릅니다. 다음은 RabbitMQ용 Amazon MQ 대기열의 Lambda 페이로드 예제입니다.

> **Note**
>
> 예제에서 `test`는 RabbitMQ 대기열의 이름이고 `/`는 기본 가상 호스트의 이름입니다. 메시지를 받을 때 이벤트 소스는 `test::/`에 메시지를 나열합니다.

```
{
  "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 MQ를 Lambda에 연결하는 방법, Amazon MQ 이벤트 소스에 대해 Lambda가 지원하는 옵션 및 이벤트 소스 매핑 오류에 대한 자세한 내용은 AWS Lambda 개발자 안내서에서 Amazon MQ에서 Lambda 사용을 참조하세요.

RabbitMQ용 Amazon MQ에 OAuth 2.0 인증 및 권한 부여 사용

이 자습서에서는 Amazon Cognito를 OAuth 2.0 공급자로 사용하여 RabbitMQ용 Amazon MQ 브로커에 대한 OAuth 2.0 인증을 구성하는 방법을 설명합니다.

> **Note**
>
> 중국(베이징) 및 중국(닝샤) 리전에서 Amazon Cognito를 사용할 수 없습니다.

> **Important**
>
> 이 자습서는 Amazon Cognito에만 적용되지만 다른 ID 제공업체(IdP)를 사용할 수 있습니다. 자세한 내용은 OAuth 2.0 인증 예제를 참조하세요.

이 페이지의 내용

OAuth 2.0 인증을 구성하기 위한 사전 조건

AWS CDK 스택, RabbitMQ용 Amazon Cognito OAuth 2 플러그인을 배포하여 이 자습서에 필요한 Amazon Cognito 리소스를 설정할 수 있습니다. Amazon Cognito를 수동으로 설정하는 경우 RabbitMQ용 Amazon MQ 브로커에서 OAuth 2.0을 구성하기 전에 다음 사전 조건을 충족해야 합니다.

Amazon Cognito를 설정하기 위한 사전 조건

- 사용자 풀을 생성하여 Amazon Cognito 엔드포인트를 설정합니다. 이렇게 하려면 Amazon Cognito에서 OAuth 2.0을 사용하는 방법: 다양한 OAuth 2.0 권한 부여에 대해 알아보기라는 제목의 블로그를 참조하세요.
- 범위가 정의된 사용자 풀 `read:all`, `write:all`, `configure:all` 및 `tag:administrator`에서 `rabbitmq`라는 리소스 서버를 생성합니다. 이러한 범위는 RabbitMQ 권한과 연결됩니다.

리소스 서버 생성에 대한 자세한 내용은 Amazon Cognito 개발자 안내서의 사용자 풀(AWS Management Console)에 대한 리소스 서버 정의를 참조하세요.

- 다음 애플리케이션 클라이언트를 생성합니다.
  - `Machine-to-Machine application` 유형의 사용자 풀에 대한 애플리케이션 클라이언트. RabbitMQ AMQP 클라이언트에 사용할 클라이언트 보안 암호가 있는 기밀 클라이언트입니다. 애플리케이션 클라이언트 및 생성에 대한 자세한 내용은 앱 클라이언트 유형 및 앱 클라이언트 생성을 참조하세요.
  - `Single-page application` 유형의 사용자 풀에 대한 애플리케이션 클라이언트. RabbitMQ 관리 콘솔에 사용자를 로그인하는 데 사용되는 퍼블릭 클라이언트입니다. 다음 절차에서 생성할 RabbitMQ용 Amazon MQ 브로커의 엔드포인트를 허용된 콜백 URL로 포함하도록 이 애플리케이션 클라이언트를 업데이트해야 합니다. 자세한 내용은 Amazon Cognito 콘솔을 사용하여 관리형 로그인 설정을 참조하세요.

Amazon MQ를 설정하기 위한 사전 조건

- OAuth 2.0 설정의 성공 여부를 확인하는 bash 스크립트를 실행하는 작업 Docker 설치.
- AWS CLI 버전 >= `2.28.23` - 브로커 생성 중에 사용자 이름과 암호를 선택적으로 추가할 수 있습니다.

AWS CLI를 사용하여 Amazon Cognito로 OAuth 2.0 인증 구성

다음 절차에서는 Amazon Cognito를 IdP로 사용하여 RabbitMQ용 Amazon MQ 브로커에 대한 OAuth 2.0 인증을 설정하는 방법을 보여줍니다. 이 절차에서는 AWS CLI를 사용하여 필요한 리소스를 생성하고 구성합니다.

다음 절차에서는 configurationID 및 Revision, *<c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca>* 및 *<2>*와 같은 자리 표시자 값을 실제 값으로 바꿔야 합니다.

1. 다음 예제와 같이 create-configuration AWS CLI 명령을 사용하여 새 구성을 생성합니다.

```
aws mq create-configuration \
  --name "rabbitmq-oauth2-config" \
  --engine-type "RABBITMQ" \
  --engine-version "3.13"
```

이 명령은 다음 예제와 유사한 응답을 반환합니다.

```
{
    "Arn": "arn:aws:mq:us-west-2:123456789012:configuration:c-fa3390a5-7e01-4559-
ae0c-eb15b38b22ca",
    "AuthenticationStrategy": "simple",
    "Created": "2025-07-17T16:03:01.759943+00:00",
    "Id": "c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca",
    "LatestRevision": {
    "Created": "2025-07-17T16:03:01.759000+00:00",
    "Description": "Auto-generated default for rabbitmq-oauth2-config on RabbitMQ
 3.13",
    "Revision": 1
    },
    "Name": "rabbitmq-oauth2-config"
}
```

2. 다음 예제와 같이 OAuth 2.0을 인증 및 권한 부여 방법으로 사용하도록 **rabbitmq.conf**라는 구성 파일을 생성합니다.

```
auth_backends.1 = oauth2

# FIXME: Update this value with the token signing key URL of your Amazon Cognito
 user pool.
# If you used the AWS CDK stack to deploy Amazon Cognito, this is one of the stack
 outputs.
auth_oauth2.jwks_url = ${RabbitMqOAuth2TestStack.JwksUri}
auth_oauth2.resource_server_id = rabbitmq
# Amazon Cognito does not include an audience field in access tokens
auth_oauth2.verify_aud = false

# Amazon Cognito does not allow * in its custom scopes. Use aliases to translate
 between Amazon Cognito and RabbitMQ.
auth_oauth2.scope_prefix = rabbitmq/
auth_oauth2.scope_aliases.1.alias = rabbitmq/read:all
auth_oauth2.scope_aliases.1.scope = rabbitmq/read:*/*
auth_oauth2.scope_aliases.2.alias = rabbitmq/write:all
auth_oauth2.scope_aliases.2.scope = rabbitmq/write:*/*
auth_oauth2.scope_aliases.3.alias = rabbitmq/configure:all
auth_oauth2.scope_aliases.3.scope = rabbitmq/configure:*/*

# Allow OAuth 2.0 login for RabbitMQ management console
management.oauth_enabled = true
# FIXME: Update this value with the client ID of your public application client
```

```
management.oauth_client_id
 = ${RabbitMqOAuth2TestStack.ManagementConsoleAppClientId}
# FIXME: Update this value with the base JWKS URI (without /.well-known/jwks.json)
auth_oauth2.issuer = ${RabbitMqOAuth2TestStack.Issuer}
management.oauth_scopes = rabbitmq/tag:administrator
```

이 구성은 범위 별칭을 사용하여 Amazon Cognito에 정의된 범위를 RabbitMQ 호환 범위에 매핑합니다.

3. 다음 예제와 같이 update-configuration AWS CLI 명령을 사용하여 구성을 업데이트합니다. 이 명령에서 이 절차의 1단계에 대한 응답으로 받은 구성 ID를 추가합니다. 예를 들어 **c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca**입니다.

```
aws mq update-configuration \
  --configuration-id "<c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca>" \
  --data "$(cat rabbitmq.conf | base64 --wrap=0)"
```

이 명령은 다음 예제와 유사한 응답을 반환합니다.

```
{
    "Arn": "arn:aws:mq:us-west-2:123456789012:configuration:c-b600ac8e-8183-4f74-
a713-983e59f30e3d",
    "Created": "2025-07-17T16:57:04.520931+00:00",
    "Id": "c-b600ac8e-8183-4f74-a713-983e59f30e3d",
    "LatestRevision": {
        "Created": "2025-07-17T16:57:39.172000+00:00",
        "Revision": 2
    },
    "Name": "rabbitmq-oauth2-config",
    "Warnings": []
}
```

4. 이 절차의 2단계에서 생성한 OAuth 2.0 구성으로 브로커를 생성합니다. 이렇게 하려면 다음 예제에서와 같이 create-broker AWS CLI 명령을 사용하면 됩니다. 이 명령에서 1단계와 2단계의 응답에서 얻은 구성 ID와 개정 번호를 각각 입력합니다. 예: **c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca** 및 **2**.

```
aws mq create-broker \
 --broker-name "rabbitmq-oauth2-broker" \
 --engine-type "RABBITMQ" \
 --engine-version "3.13" \
```

```
 --host-instance-type "mq.m7g.large" \
 --deployment-mode "CLUSTER_MULTI_AZ" \
 --logs '{"General": true}' \
 --publicly-accessible \
 --configuration '{"Id": "<c-fa3390a5-7e01-4559-ae0c-
eb15b38b22ca>","Revision": <2>}' \
```

이 명령은 다음 예제와 유사한 응답을 반환합니다.

```
{
    "BrokerArn": "arn:aws:mq:us-west-2:123456789012:broker:rabbitmq-oauth2-
broker:b-2a1b5133-a10c-49d2-879b-8c176c34cf73",
    "BrokerId": "b-2a1b5133-a10c-49d2-879b-8c176c34cf73"
}
```

5. 다음 예제와 같이 describe-broker AWS CLI 명령을 사용하여 브로커의 상태가 CREATION_IN_PROGRESS에서 RUNNING로 전환되는지 확인합니다. 이 명령에서 이전 단계의 결과에서 얻은 브로커 ID를 입력합니다. 예: **b-2a1b5133-a10c-49d2-879b-8c176c34cf73**.

```
aws mq describe-broker \
 --broker-id "<b-2a1b5133-a10c-49d2-879b-8c176c34cf73>"
```

이 명령은 다음 예제와 유사한 응답을 반환합니다. 다음 응답은 describe-broker 명령이 반환하는 전체 출력의 약식 버전입니다. 이 응답은 브로커 상태와 브로커를 보호하는 데 사용되는 인증 전략을 보여줍니다. 이 경우 config_managed 인증 전략은 브로커가 OAuth 2 인증 방법을 사용함을 나타냅니다.

```
{
    "AuthenticationStrategy": "config_managed",
    ...,
    "BrokerState": "RUNNING",
    ...
}
```

OAuth2를 사용하여 RabbitMQ 관리 콘솔에 로그인하려면 브로커 엔드포인트를 해당 Amazon Cognito 앱 클라이언트에 유효한 콜백 URL로 추가해야 합니다. 자세한 내용은 샘플 Amazon Cognito CDK 스택 설정의 5단계를 참조하세요.

6. 다음 perf-test.sh 스크립트를 사용하여 OAuth 2.0 인증 및 권한 부여를 확인합니다.

이 bash 스크립트를 사용하여 RabbitMQ용 Amazon MQ 브로커에 대한 연결을 테스트합니다. 이 스크립트는 Amazon Cognito에서 토큰을 가져와 연결이 제대로 구성되었는지 확인합니다. 성공적으로 구성되면 브로커가 메시지를 게시하고 소비하는 것을 볼 수 있습니다.

ACCESS_REFUSED 오류가 발생하면 브로커에 대한 CloudWatch 로그를 사용하여 구성 설정 문제를 해결할 수 있습니다. Amazon MQ 콘솔에서 브로커의 CloudWatch 로그 그룹에 대한 링크를 찾을 수 있습니다.

이 스크립트에서는 다음 값을 제공해야 합니다.

- CLIENT_ID 및 CLIENT_SECRET: Amazon Cognito 콘솔의 앱 클라이언트 페이지에서 이러한 값을 찾을 수 있습니다.
- Cognito 도메인: Amazon Cognito 콘솔에서 찾을 수 있습니다. 브랜딩에서 도메인을 선택합니다. 도메인 페이지의 리소스 서버 섹션에서 이 값을 찾을 수 있습니다.
- Amazon MQ 브로커 엔드포인트: Amazon MQ 콘솔의 브로커 세부 정보 페이지에 있는 연결에서 이 값을 찾을 수 있습니다.

```
#! /bin/bash
set -e

# Client information
## FIXME: Update this value with the client ID and secret of your confidential
 application client
CLIENT_ID=${RabbitMqOAuth2TestStack.AmqpAppClientId}
CLIENT_SECRET=${RabbitMqOAuth2TestStack.AmqpAppClientSecret}

# FIXME: Update this value with the domain of your Amazon Cognito user pool
RESPONSE=$(curl -X POST ${RabbitMqOAuth2TestStack.TokenEndpoint} \
                -H "Content-Type: application/x-www-form-urlencoded" \
                -d
 "grant_type=client_credentials&client_id=${CLIENT_ID}&client_secret=${CLIENT_SECRET}&scope
configure:all rabbitmq/read:all rabbitmq/tag:administrator rabbitmq/write:all")


# Extract the access_token from the response.
# This token will be passed in the password field when connecting to the broker.
# Note that the username is left blank, the field is ignored by the plugin.
BROKER_PASSWORD=$(echo ${RESPONSE} | jq -r '.access_token')
```

```
# FIXME: Update this value with the endpoint of your broker. For
 example, b-89424106-7e0e-4abe-8e98-8de0dada7630.mq.us-east-1.on.aws.
BROKER_DNS=<broker_dns>
CONNECTION_STRING=amqps://:${BROKER_PASSWORD}@${BROKER_DNS}:5671

# Produce/consume messages using the above connection string
QUEUES_COUNT=1
PRODUCERS_COUNT=1
CONSUMERS_COUNT=1
PRODUCER_RATE=1

docker run -it --rm --ulimit nofile=40960:40960 pivotalrabbitmq/perf-test:latest \
    --queue-pattern 'test-queue-%d' --queue-pattern-from 1 --queue-pattern-to
 $QUEUES_COUNT \
    --producers $PRODUCERS_COUNT --consumers $CONSUMERS_COUNT \
    --id "test${QUEUES_COUNT}q${PRODUCERS_COUNT}p${CONSUMERS_COUNT}c
${PRODUCER_RATE}r" \
    --uri ${CONNECTION_STRING} \
    --flag persistent --rate $PRODUCER_RATE
```

Amazon Cognito를 사용한 OAuth 2.0 및 단순 인증 구성

OAuth 2.0 인증을 사용하여 브로커를 생성할 때 다음 인증 방법 중 하나를 지정할 수 있습니다.

- OAuth 2.0 전용: 이 방법을 사용하려면 브로커를 생성하는 동안 사용자 이름과 암호를 제공하지 마십시오. 이전 절차에서는 OAuth 2.0 인증 방법만 사용하는 방법을 보여줍니다.
- OAuth 2.0 및 단순 인증: 이 방법을 사용하려면 브로커를 생성하는 동안 사용자 이름과 암호를 제공합니다. 또한 다음 절차와 같이 `auth_backends.2 = internal`을 브로커 구성에 추가합니다.

다음 절차에서는 *<ConfigurationId>* 및 *<Revision>*과 같은 자리 표시자 값을 실제 값으로 바꿔야 합니다.

1. 두 인증 방법을 모두 사용하려면 다음 예제와 같이 브로커 구성을 생성합니다.

```
auth_backends.1 = oauth2
auth_backends.2 = internal

# FIXME: Update this value with the token signing key URL of your Amazon Cognito
 user pool
auth_oauth2.jwks_url = ${RabbitMqOAuth2TestStack.JwksUri}
```

```
auth_oauth2.resource_server_id = rabbitmq
auth_oauth2.verify_aud = false

auth_oauth2.scope_prefix = rabbitmq/
auth_oauth2.scope_aliases.1.alias = rabbitmq/read:all
auth_oauth2.scope_aliases.1.scope = rabbitmq/read:*/*
auth_oauth2.scope_aliases.2.alias = rabbitmq/write:all
auth_oauth2.scope_aliases.2.scope = rabbitmq/write:*/*
auth_oauth2.scope_aliases.3.alias = rabbitmq/configure:all
auth_oauth2.scope_aliases.3.scope = rabbitmq/configure:*/*
```

이 구성은 범위 별칭을 사용하여 Amazon Cognito에 정의된 범위를 RabbitMQ 호환 범위에 매핑합니다.

2. 다음 예제와 같이 두 인증 방법을 모두 사용하는 브로커를 생성합니다.

```
aws mq create-broker \
 --broker-name "rabbitmq-oauth2-broker-with-internal-user" \
 --engine-type "RABBITMQ" \
 --engine-version "3.13" \
 --host-instance-type "mq.m7g.large" \
 --deployment-mode "CLUSTER_MULTI_AZ" \
 --logs '{"General": true}' \
 --publicly-accessible \
 --configuration '{"Id": "<ConfigurationId>","Revision": <Revision>}' \
 --users '[{"Username":"<myUser>","Password":"<myPassword11>"}]'
```

3. Amazon Cognito를 사용하여 OAuth 2.0 인증 구성 절차의 5단계와 6단계에 설명된 대로 브로커 상태와 인증 방법 설정 구성이 성공했는지 확인합니다.

RabbitMQ용 Amazon MQ에 대한 IAM 인증 및 권한 부여 사용

다음 절차에서는 RabbitMQ용 Amazon MQ 브로커에 대해 AWS IAM 인증 및 권한 부여를 활성화하는 방법을 보여줍니다. IAM을 활성화한 후 사용자는 AWS IAM 자격 증명을 사용하여 RabbitMQ 관리 API에 액세스하고 AMQP를 통해 연결할 수 있습니다. IAM 인증이 RabbitMQ용 Amazon MQ에서 작동하는 방식에 대한 자세한 내용은 섹션을 참조하세요the section called “IAM 인증 및 권한 부여”.

사전 조건

- AWS RabbitMQ용 Amazon MQ 브로커를 소유한 AWS 계정의 관리자 자격 증명
- 이러한 관리자 자격 증명으로 구성된 셸 환경(AWS CLI 프로필 또는 환경 변수 사용)

- AWS CLI 설치 및 구성
- `jq` 명령줄 JSON 프로세서 설치됨
- `curl` 명령줄 도구 설치됨

를 사용하여 IAM 인증 및 권한 부여 구성 AWS CLI

1. 환경 변수 설정

   브로커에 필요한 환경 변수를 설정합니다.

   ```
   export AWS_DEFAULT_REGION=<region>
   export BROKER_ID=<broker-id>
   ```

2. 아웃바운드 JWT 토큰 활성화

   AWS 계정에 대해 아웃바운드 웹 자격 증명 페더레이션을 활성화합니다.

   ```
   ISSUER_IDENTIFIER=$(aws iam enable-outbound-web-identity-federation --query
    'IssuerIdentifier' --output text)
   echo $ISSUER_IDENTIFIER
   ```

   출력에는 계정의 고유한 발급자 식별자 URL이 형식으로 표시됩니다`https://<id>.tokens.sts.global.api.aws`.

3. IAM 정책 문서 생성

   웹 자격 증명 토큰을 가져올 수 있는 권한을 부여하는 정책 문서를 생성합니다.

   ```
   cat > policy.json << 'EOF'
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Sid": "VisualEditor0",
               "Effect": "Allow",
               "Action": [
   ```

```
                "sts:GetWebIdentityToken",
                "sts:TagGetWebIdentityToken"
            ],
            "Resource": "*"
        }
    ]
}
EOF
```

4. 신뢰 정책 생성

   호출자 자격 증명을 검색하고 신뢰 정책 문서를 생성합니다.

```
CALLER_ARN=$(aws sts get-caller-identity --query Arn --output text)
cat > trust-policy.json << EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "$CALLER_ARN"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
EOF
```

5. IAM 역할 생성

   IAM 역할을 생성하고 정책을 연결합니다.

```
aws iam create-role --role-name RabbitMqAdminRole --assume-role-policy-document
 file://trust-policy.json
aws iam put-role-policy --role-name RabbitMqAdminRole --policy-name
 RabbitMqAdminRolePolicy --policy-document file://policy.json
```

6. RabbitMQ OAuth2 설정 구성

   OAuth2 인증 및 권한 부여 설정을 사용하여 RabbitMQ 구성 파일을 생성합니다.

```
cat > rabbitmq.conf << EOF
auth_backends.1 = oauth2
auth_backends.2 = internal

auth_oauth2.jwks_url = ${ISSUER_IDENTIFIER}/.well-known/jwks.json
auth_oauth2.resource_server_id = rabbitmq
auth_oauth2.scope_prefix = rabbitmq/

auth_oauth2.additional_scopes_key = sub
auth_oauth2.scope_aliases.1.alias = arn:aws:iam::$(aws sts get-caller-identity --
query Account --output text):role/RabbitMqAdminRole
auth_oauth2.scope_aliases.1.scope = rabbitmq/tag:administrator rabbitmq/read:*/*
 rabbitmq/write:*/* rabbitmq/configure:*/*
auth_oauth2.https.hostname_verification = wildcard

management.oauth_enabled = true
EOF
```

7. 브로커 구성 업데이트

   브로커에 새 구성을 적용합니다.

```
# Retrieve the configuration ID
CONFIG_ID=$(aws mq describe-broker --broker-id $BROKER_ID --query
 'Configurations[0].Id' --output text)

# Create a new configuration revision
REVISION=$(aws mq update-configuration --configuration-id $CONFIG_ID --data "$(cat
 rabbitmq.conf | base64 --wrap=0)" --query 'LatestRevision.Revision' --output text)

# Apply the configuration to the broker
aws mq update-broker --broker-id $BROKER_ID --configuration Id=$CONFIG_ID,Revision=
$REVISION

# Reboot the broker to apply changes
aws mq reboot-broker --broker-id $BROKER_ID
```

다음 단계로 진행하기 RUNNING 전에 브로커 상태가 로 돌아갈 때까지 기다립니다.

8. JWT 토큰 가져오기

   IAM 역할을 수임하고 웹 자격 증명 토큰을 얻습니다.

```
# Assume the RabbitMqAdminRole
ROLE_CREDS=$(aws sts assume-role --role-arn arn:aws:iam::$(aws sts get-caller-
identity --query Account --output text):role/RabbitMqAdminRole --role-session-name
 rabbitmq-session)

# Configure the session with temporary credentials
export AWS_ACCESS_KEY_ID=$(echo "$ROLE_CREDS" | jq -r '.Credentials.AccessKeyId')
export AWS_SECRET_ACCESS_KEY=$(echo "$ROLE_CREDS" | jq -r
 '.Credentials.SecretAccessKey')
export AWS_SESSION_TOKEN=$(echo "$ROLE_CREDS" | jq -r '.Credentials.SessionToken')

# Obtain the web identity token
TOKEN_RESPONSE=$(aws sts get-web-identity-token \
    --audience "rabbitmq" \
    --signing-algorithm ES384 \
    --duration-seconds 300 \
    --tags Key=scope,Value="rabbitmq/tag:administrator")

# Extract the token
TOKEN=$(echo "$TOKEN_RESPONSE" | jq -r '.WebIdentityToken')
```

9. RabbitMQ 관리 API에 액세스

   JWT 토큰을 사용하여 RabbitMQ 관리 API에 액세스합니다.

```
BROKER_URL=<broker-id>.mq.<region>.on.aws

curl -u ":$TOKEN" \
    -X GET https://${BROKER_URL}/api/overview \
    -H "Content-Type: application/json"
```

응답이 성공하면 IAM 인증이 올바르게 작동하는지 확인합니다. 응답에는 JSON 형식의 브로커 개요 정보가 포함되어 있습니다.

10. JWT 토큰을 사용하여 AMQP를 통해 연결

    성능 테스트 도구와 함께 JWT 토큰을 사용하여 AMQP 연결을 테스트합니다.

```
BROKER_DNS=<broker-endpoint>
CONNECTION_STRING=amqps://:${TOKEN}@${BROKER_DNS}:5671

docker run -it --rm --ulimit nofile=40960:40960 pivotalrabbitmq/perf-test:latest \
    --queue-pattern 'test-queue-%d' --queue-pattern-from 1 --queue-pattern-to 1 \
    --producers 1 --consumers 1 \
    --uri ${CONNECTION_STRING} \
    --flag persistent --rate 1
```

    ACCESS_REFUSED 오류가 발생하면 브로커에 대한 CloudWatch 로그를 사용하여 구성 설정 문제를 해결할 수 있습니다. Amazon MQ 콘솔에서 브로커의 CloudWatch Logs 로그 그룹에 대한 링크를 찾을 수 있습니다.

RabbitMQ용 Amazon MQ에 LDAP 인증 및 권한 부여 사용

이 자습서에서는를 사용하여 RabbitMQ용 Amazon MQ 브로커에 대한 LDAP 인증 및 권한 부여를 구성하는 방법을 설명합니다 AWS Managed Microsoft AD.

이 페이지의 내용

LDAP 인증 및 권한 부여를 구성하기 위한 사전 조건

RabbitMQ LDAP 통합용 Amazon MQ용 CDK 스택을 배포하여이 자습서에 필요한 AWS 리소스를 설정할 수 있습니다. AWS Amazon MQ RabbitMQ AWS Managed Microsoft AD

이 CDK 스택은 LDAP 사용자 및 그룹 AWS Managed Microsoft AD, Network Load Balancer, 인증서 및 IAM 역할을 포함하여 필요한 모든 AWS 리소스를 자동으로 생성합니다. 스택에서 생성한 리소스의 전체 목록은 패키지 README를 참조하세요.

CDK 스택을 사용하는 대신 리소스를 수동으로 설정하는 경우 RabbitMQ용 Amazon MQ 브로커에 LDAP를 구성하기 전에 동등한 인프라가 있는지 확인합니다.

Amazon MQ를 설정하기 위한 사전 조건

AWS CLI 버전 >= 2.28.23: 브로커 생성 중에 사용자 이름과 암호를 선택적으로 추가할 수 있습니다.

AWS CLI를 사용하여 RabbitMQ에서 LDAP 구성

이 절차에서는 AWS CLI를 사용하여 필요한 리소스를 생성하고 구성합니다. 다음 절차에서는 configurationID 및 개정, <c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca>와 같은 자리 표시자 값을 실제 값으로 바꿔야 <2>합니다.

1. 다음 예제와 같이 `create-configuration` AWS CLI 명령을 사용하여 새 구성을 생성합니다.

```
aws mq create-configuration \
  --name "rabbitmq-ldap-config" \
  --engine-type "RABBITMQ" \
  --engine-version "3.13"
```

이 명령은 다음 예제와 유사한 응답을 반환합니다.

```
{
"Arn": "arn:aws:mq:us-west-2:123456789012:configuration:c-fa3390a5-7e01-4559-ae0c-
eb15b38b22ca",
    "AuthenticationStrategy": "simple",
    "Created": "2025-07-17T16:03:01.759943+00:00",
    "Id": "c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca",
    "LatestRevision": {
"Created": "2025-07-17T16:03:01.759000+00:00",
    "Description": "Auto-generated default for rabbitmq-ldap-config on RabbitMQ
 3.13",
    "Revision": 1
    },
    "Name": "rabbitmq-ldap-config"
}
```

2. 다음 예제와 같이 라는 구성 파일을 생성`rabbitmq.conf`하여 LDAP를 인증 및 권한 부여 방법으로 사용합니다. 템플릿의 모든 자리 표시자 값(로 표시됨`${RabbitMqLdapTestStack.*}`)을 배포된 AWS CDK 사전 조건 스택 출력 또는 이에 상응하는 인프라의 실제 값으로 바꿉니다.

```
auth_backends.1 = ldap

# LDAP authentication settings - For more information,
# see https://www.rabbitmq.com/docs/ldap#basic

# FIXME: Replace the ${RabbitMqLdapTestStack.*} placeholders with actual values
# from your deployed prerequisite CDK stack outputs.
auth_ldap.servers.1 = ${RabbitMqLdapTestStack.NlbDnsName}
auth_ldap.dn_lookup_bind.user_dn = ${RabbitMqLdapTestStack.DnLookupUserDn}
auth_ldap.dn_lookup_base = ${RabbitMqLdapTestStack.DnLookupBase}
auth_ldap.dn_lookup_attribute = ${RabbitMqLdapTestStack.DnLookupAttribute}
auth_ldap.port = 636
auth_ldap.use_ssl = true
auth_ldap.ssl_options.verify = verify_peer
auth_ldap.log = network

# AWS integration for secure credential retrieval
# - see: https://github.com/amazon-mq/rabbitmq-aws
# The aws plugin allows RabbitMQ to securely retrieve credentials and certificates
# from AWS services.

# Replace the ${RabbitMqLdapTestStack.*} placeholders with actual ARN values
# from your deployed prerequisite CDK stack outputs.
aws.arns.auth_ldap.ssl_options.cacertfile = ${RabbitMqLdapTestStack.CaCertArn}
aws.arns.auth_ldap.dn_lookup_bind.password =
 ${RabbitMqLdapTestStack.DnLookupUserPasswordArn}
aws.arns.assume_role_arn = ${RabbitMqLdapTestStack.AmazonMqAssumeRoleArn}

# LDAP authorization queries - For more information,
# see: https://www.rabbitmq.com/docs/ldap#authorisation

# FIXME: Replace the ${RabbitMqLdapTestStack.*} placeholders with actual group DN
# values from your deployed prerequisite CDK stack outputs
# Uses Active Directory groups created by the prerequisite CDK stack
auth_ldap.queries.tags = '''
[{administrator, {in_group,
 "${RabbitMqLdapTestStack.RabbitMqAdministratorsGroupDn}"}},
```

```
{management,    {in_group,
 "${RabbitMqLdapTestStack.RabbitMqMonitoringUsersGroupDn}"}}]
'''

# FIXME: This provides all authenticated users access to all vhosts
# - update to restrict access as required
auth_ldap.queries.vhost_access = '''
{constant, true}
'''

# FIXME: This provides all authenticated users full access to all
# queues and exchanges - update to restrict access as required
auth_ldap.queries.resource_access = '''
{for, [    {permission, configure, {constant, true}},
     {permission, write,
      {for, [{resource, queue,    {constant, true}},
             {resource, exchange, {constant, true}}]}},
     {permission, read,
      {for, [{resource, exchange, {constant, true}},
             {resource, queue,    {constant, true}}]}}
    ]
}
'''

# FIXME: This provides all authenticated users access to all topics
# - update to restrict access as required
auth_ldap.queries.topic_access = '''
{for, [{permission, write, {constant, true}},
     {permission, read,  {constant, true}}
    ]
}
'''
```

3. 다음 예제와 같이 `update-configuration` AWS CLI 명령을 사용하여 구성을 업데이트합니다. 이 명령에서 이 절차의 1단계에 대한 응답으로 받은 구성 ID를 추가합니다. 예를 들어 `c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca`입니다.

```
aws mq update-configuration \
  --configuration-id "<c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca>" \
  --data "$(cat rabbitmq.conf | base64 --wrap=0)"
```

이 명령은 다음 예제와 유사한 응답을 반환합니다.

```
{
    "Arn": "arn:aws:mq:us-west-2:123456789012:configuration:c-b600ac8e-8183-4f74-
a713-983e59f30e3d",
    "Created": "2025-07-17T16:57:04.520931+00:00",
    "Id": "c-b600ac8e-8183-4f74-a713-983e59f30e3d",
    "LatestRevision": {
        "Created": "2025-07-17T16:57:39.172000+00:00",
        "Revision": 2
    },
    "Name": "rabbitmq-ldap-config",
    "Warnings": []
}
```

4. 이 절차의 2단계에서 생성한 LDAP 구성을 사용하여 브로커를 생성합니다. 이렇게 하려면 다음 예제와 같이 `create-broker` AWS CLI 명령을 사용합니다. 이 명령에서 1단계와 2단계의 응답에서 얻은 구성 ID와 개정 번호를 각각 입력합니다. 예: `c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca` 및 2.

```
aws mq create-broker \
 --broker-name "rabbitmq-ldap-test-1" \
 --engine-type "RABBITMQ" \
 --engine-version "3.13" \
 --host-instance-type "mq.m7g.large" \
 --deployment-mode "CLUSTER_MULTI_AZ" \
 --logs '{"General": true}' \
 --publicly-accessible \
 --configuration '{"Id": "<c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca>","Revision":
 <2>}'
```

이 명령은 다음 예제와 유사한 응답을 반환합니다.

```
{
    "BrokerArn": "arn:aws:mq:us-west-2:123456789012:broker:rabbitmq-ldap-
broker:b-2a1b5133-a10c-49d2-879b-8c176c34cf73",
```

```
    "BrokerId": "b-2a1b5133-a10c-49d2-879b-8c176c34cf73"
}
```

> ⚠ 브로커 이름 지정 제한
>
> 필수 CDK 스택에서 생성한 IAM 역할은 브로커 이름을 로 시작하도록 제한합니다rabbitmq-ldap-test. 브로커 이름이이 패턴을 따르는지 확인합니다. 그렇지 않으면 IAM 역할에 ARN 확인을 위해 역할을 수임할 권한이 없습니다.

5. 다음 예제와 같이 describe-broker AWS CLI 명령을 RUNNING사용하여 브로커의 상태가 CREATION_IN_PROGRESS에서 로 전환되는지 확인합니다. 이 명령에서 이전 단계의 결과에서 얻은 브로커 ID를 입력합니다. 예: b-2a1b5133-a10c-49d2-879b-8c176c34cf73.

```
aws mq describe-broker \
 --broker-id "<b-2a1b5133-a10c-49d2-879b-8c176c34cf73>"
```

이 명령은 다음 예제와 유사한 응답을 반환합니다. 다음 응답은 describe-broker 명령이 반환하는 전체 출력의 약식 버전입니다. 이 응답은 브로커 상태와 브로커를 보호하는 데 사용되는 인증 전략을 보여줍니다. 이 경우 config_managed 인증 전략은 브로커가 LDAP 인증 방법을 사용함을 나타냅니다.

```
{
"AuthenticationStrategy": "config_managed",
    ...,
    "BrokerState": "RUNNING",
    ...
}
```

6. 필수 CDK 스택에서 생성한 테스트 사용자 중 하나를 사용하여 RabbitMQ 액세스 검증

```
# FIXME: Replace ${RabbitMqLdapTestStack.ConsoleUserPasswordArn} with the actual
 ARN from your deployed prerequisite CDK stack outputs
CONSOLE_PASSWORD=$(aws secretsmanager get-secret-value \
  --secret-id ${RabbitMqLdapTestStack.ConsoleUserPasswordArn} \
```

```
  --query 'SecretString' --output text)

# FIXME: Replace BrokerConsoleURL with the actual ConsoleURL retrieved by
# calling describe-broker for the broker created above
# Call management API /api/overview (should succeed)
curl -u RabbitMqConsoleUser:$CONSOLE_PASSWORD \
  https://${BrokerConsoleURL}/api/overview

# Try to create a user (should fail - console user only has monitoring permissions)
curl -u RabbitMqConsoleUser:$CONSOLE_PASSWORD \
  -X PUT https://${BrokerConsoleURL}/api/users/testuser \
  -H "Content-Type: application/json" \
  -d '{"password":"testpass","tags":"management"}'
```

RabbitMQ용 Amazon MQ에 HTTP 인증 및 권한 부여 사용

이 자습서에서는 외부 HTTP 서버를 사용하여 RabbitMQ용 Amazon MQ 브로커에 대한 HTTP 인증 및 권한 부여를 구성하는 방법을 설명합니다.

> **Note**
>
> HTTP 인증 플러그인은 RabbitMQ용 Amazon MQ 버전 4 이상에서만 사용할 수 있습니다. RabbitMQ

이 페이지의 내용

HTTP 인증 및 권한 부여를 구성하기 위한 사전 조건

AWS RabbitMQ용 Amazon MQ HTTP 인증 통합을 위한 CDK 스택을 배포하여이 자습서에 필요한 AWS 리소스를 설정할 수 있습니다.

이 CDK 스택은 HTTP 인증 서버, 인증서 및 IAM 역할을 포함하여 필요한 모든 AWS 리소스를 자동으로 생성합니다. 스택에서 생성한 리소스의 전체 목록은 패키지 README를 참조하세요.

CDK 스택을 사용하는 대신 리소스를 수동으로 설정하는 경우 RabbitMQ용 Amazon MQ 브로커에 HTTP 인증을 구성하기 전에 동등한 인프라가 있는지 확인합니다.

Amazon MQ를 설정하기 위한 사전 조건

AWS CLI 버전 >= 2.28.23: 브로커 생성 중에 사용자 이름과 암호를 선택적으로 추가할 수 있습니다.

AWS CLI를 사용하여 RabbitMQ에서 HTTP 인증 구성

이 절차에서는 AWS CLI를 사용하여 필요한 리소스를 생성하고 구성합니다. 다음 절차에서는 자리 표시자 값을 실제 값으로 바꿔야 합니다.

1. 다음 예제와 같이 `create-configuration` AWS CLI 명령을 사용하여 새 구성을 생성합니다.

```
aws mq create-configuration \
  --name "rabbitmq-http-config" \
  --engine-type "RABBITMQ" \
  --engine-version "4.2"
```

이 명령은 다음 예제와 유사한 응답을 반환합니다.

```
{
    "Arn": "arn:aws:mq:us-west-2:123456789012:configuration:c-fa3390a5-7e01-4559-
ae0c-eb15b38b22ca",
    "AuthenticationStrategy": "simple",
    "Created": "2025-07-17T16:03:01.759943+00:00",
    "Id": "c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca",
    "LatestRevision": {
        "Created": "2025-07-17T16:03:01.759000+00:00",
        "Description": "Auto-generated default for rabbitmq-http-config on RabbitMQ
 4.2",
        "Revision": 1
    },
    "Name": "rabbitmq-http-config"
}
```

2. 다음 예제와 같이 HTTP를 인증 및 권한 부여 방법으로 `rabbitmq.conf` 사용하도록 라는 구성 파일을 생성합니다. 템플릿의 모든 자리 표시자 값(로 표시됨`${...}`)을 배포된 AWS CDK 사전 조건 스택 출력 또는 이에 상응하는 인프라의 실제 값으로 바꿉니다.

```
auth_backends.1 = cache
auth_backends.2 = http
auth_cache.cached_backend = http

# HTTP authentication settings
# For more information, see https://github.com/rabbitmq/rabbitmq-auth-backend-http

# FIXME: Replace the ${...} placeholders with actual values
# from your deployed prerequisite CDK stack outputs.
auth_http.http_method = post
auth_http.user_path = ${HttpServerUserPath}
auth_http.vhost_path = ${HttpServerVhostPath}
auth_http.resource_path = ${HttpServerResourcePath}
auth_http.topic_path = ${HttpServerTopicPath}

# TLS/HTTPS configuration
auth_http.ssl_options.verify = verify_peer
auth_http.ssl_options.sni = test.amazonaws.com

# AWS integration for secure credential retrieval
# For more information, see https://github.com/amazon-mq/rabbitmq-aws

# Replace the ${...} placeholders with actual ARN values
# from your deployed prerequisite CDK stack outputs.
aws.arns.assume_role_arn = ${AmazonMqAssumeRoleArn}
aws.arns.auth_http.ssl_options.cacertfile = ${CaCertArn}
```

3. `update-configuration` AWS CLI 명령을 사용하여 구성을 업데이트합니다. 3단계의 구성 ID 를 사용합니다.

```
aws mq update-configuration \
  --configuration-id "<c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca>" \
  --data "$(cat rabbitmq.conf | base64 --wrap=0)"
```

이 명령은 다음 예제와 유사한 응답을 반환합니다.

```
{
    "Arn": "arn:aws:mq:us-west-2:123456789012:configuration:c-fa3390a5-7e01-4559-
ae0c-eb15b38b22ca",
    "Created": "2025-07-17T16:57:04.520931+00:00",
    "Id": "c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca",
    "LatestRevision": {
        "Created": "2025-07-17T16:57:39.172000+00:00",
        "Revision": 2
    },
    "Name": "rabbitmq-http-config",
    "Warnings": []
}
```

4. HTTP 구성을 사용하여 브로커를 생성합니다. 이전 단계의 구성 ID와 개정 번호를 사용합니다.

```
aws mq create-broker \
  --broker-name "rabbitmq-http-test-1" \
  --engine-type "RABBITMQ" \
  --engine-version "4.2" \
  --host-instance-type "mq.m7g.large" \
  --deployment-mode "SINGLE_INSTANCE" \
  --logs '{"General": true}' \
  --publicly-accessible \
  --configuration '{"Id": "<c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca>","Revision":
 <2>}'
```

이 명령은 다음 예제와 유사한 응답을 반환합니다.

```
{
    "BrokerArn": "arn:aws:mq:us-west-2:123456789012:broker:rabbitmq-http-
test-1:b-2a1b5133-a10c-49d2-879b-8c176c34cf73",
    "BrokerId": "b-2a1b5133-a10c-49d2-879b-8c176c34cf73"
}
```

5. describe-broker AWS CLI 명령을 RUNNING사용하여 브로커의 상태가에서 CREATION_IN_PROGRESS로 전환되는지 확인합니다.

```
aws mq describe-broker \
  --broker-id "<b-2a1b5133-a10c-49d2-879b-8c176c34cf73>"
```

이 명령은 다음 예제와 유사한 응답을 반환합니다. config_managed 인증 전략은 브로커가 HTTP 인증 방법을 사용함을 나타냅니다.

```
{
    "AuthenticationStrategy": "config_managed",
    ...,
    "BrokerState": "RUNNING",
    ...
}
```

6. 필수 CDK 스택에서 생성한 테스트 사용자 중 하나를 사용하여 RabbitMQ 액세스 검증

```
# FIXME: Replace ${RabbitMqHttpAuthElbStack.ConsoleUserPasswordArn} with the actual
 ARN from your deployed prerequisite CDK stack outputs
CONSOLE_PASSWORD=$(aws secretsmanager get-secret-value \
  --secret-id ${RabbitMqHttpAuthElbStack.ConsoleUserPasswordArn} \
  --query 'SecretString' --output text)

# FIXME: Replace BrokerConsoleURL with the actual ConsoleURL retrieved by
# calling describe-broker for the broker created above
# Call management API /api/overview (should succeed)
curl -u RabbitMqConsoleUser:$CONSOLE_PASSWORD \
  https://${BrokerConsoleURL}/api/overview

# Try to create a vhost (should fail - console user only has management
 permissions)
curl -u RabbitMqConsoleUser:$CONSOLE_PASSWORD \
  -X PUT https://${BrokerConsoleURL}/api/vhosts/test-vhost \
  -H "Content-Type: application/json" \
  -d '{}'
```

RabbitMQ용 Amazon MQ에 SSL 인증서 인증 사용

이 자습서에서는 프라이빗 인증 기관을 사용하여 RabbitMQ용 Amazon MQ 브로커에 대한 SSL 인증서 인증을 구성하는 방법을 설명합니다.

> **Note**
>
> SSL 인증서 인증 플러그인은 RabbitMQ용 Amazon MQ 버전 4 이상에서만 사용할 수 있습니다. RabbitMQ

이 페이지의 내용

SSL 인증서 인증을 구성하기 위한 사전 조건

SSL 인증서 인증은 상호 TLS(mTLS)를 사용하여 X.509 인증서를 사용하여 클라이언트를 인증합니다. AWS RabbitMQ용 Amazon MQ mTLS 통합을 위한 CDK 스택을 배포하여이 자습서에 필요한 AWS 리소스를 설정할 수 있습니다.

이 CDK 스택은 인증 기관, 클라이언트 인증서 및 IAM 역할을 포함하여 필요한 모든 AWS 리소스를 자동으로 생성합니다. 스택에서 생성한 리소스의 전체 목록은 패키지 README를 참조하세요.

> **Note**
>
> CDK 스택을 배포하기 전에 `RABBITMQ_TEST_USER_NAME` 환경 변수를 설정합니다. 이 값은 클라이언트 인증서의 일반 이름(CN)으로 사용되며 자습서 단계에서 사용하는 사용자 이름과 일치해야 합니다. 예: `export RABBITMQ_TEST_USER_NAME="myuser"`

CDK 스택을 사용하는 대신 리소스를 수동으로 설정하는 경우 RabbitMQ용 Amazon MQ 브로커에서 SSL 인증서 인증을 구성하기 전에 동등한 인프라가 있는지 확인합니다.

Amazon MQ를 설정하기 위한 사전 조건

AWS CLI 버전 >= 2.28.23: 브로커 생성 중에 사용자 이름과 암호를 선택적으로 추가할 수 있습니다.

AWS CLI를 사용하여 RabbitMQ에서 SSL 인증서 인증 구성

이 절차에서는 AWS CLI를 사용하여 필요한 리소스를 생성하고 구성합니다. 다음 절차에서는 configurationID, Revision 및 <c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca>와 같은 자리 표시자 값을 실제 값으로 바꿔야 <2>합니다.

1. 다음 예제와 같이 `create-configuration` AWS CLI 명령을 사용하여 새 구성을 생성합니다.

```
aws mq create-configuration \
  --name "rabbitmq-ssl-config" \
  --engine-type "RABBITMQ" \
  --engine-version "4.2"
```

이 명령은 다음 예제와 유사한 응답을 반환합니다.

```
{
    "Arn": "arn:aws:mq:us-west-2:123456789012:configuration:c-fa3390a5-7e01-4559-
ae0c-eb15b38b22ca",
    "AuthenticationStrategy": "simple",
    "Created": "2025-07-17T16:03:01.759943+00:00",
    "Id": "c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca",
    "LatestRevision": {
        "Created": "2025-07-17T16:03:01.759000+00:00",
        "Description": "Auto-generated default for rabbitmq-ssl-config on RabbitMQ
 4.2",
        "Revision": 1
    },
    "Name": "rabbitmq-ssl-config"
}
```

2. 다음 예제와 같이 라는 구성 파일을 생성`rabbitmq.conf`하여 SSL 인증서 인증을 사용합니다. 템플릿의 모든 자리 표시자 값(로 표시됨`${...}`)을 배포된 AWS CDK 사전 조건 스택 출력 또는 이에 상응하는 인프라의 실제 값으로 바꿉니다.

```
auth_mechanisms.1 = EXTERNAL
ssl_cert_login_from = common_name

auth_backends.1 = internal

# Reject if no client cert
ssl_options.verify = verify_peer
ssl_options.fail_if_no_peer_cert = true

# AWS integration for secure credential retrieval
# For more information, see https://github.com/amazon-mq/rabbitmq-aws

# FIXME: Replace the ${...} placeholders with actual ARN values
# from your deployed prerequisite CDK stack outputs.
aws.arns.assume_role_arn = ${AmazonMqAssumeRoleArn}
aws.arns.ssl_options.cacertfile = ${CaCertArn}
```

3. 다음 예제와 같이 update-configuration AWS CLI 명령을 사용하여 구성을 업데이트합니다. 이 명령에서 이 절차의 1단계에 대한 응답으로 받은 구성 ID를 추가합니다. 예를 들어 c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca입니다.

```
aws mq update-configuration \
  --configuration-id "<c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca>" \
  --data "$(cat rabbitmq.conf | base64 --wrap=0)"
```

이 명령은 다음 예제와 유사한 응답을 반환합니다.

```
{
    "Arn": "arn:aws:mq:us-west-2:123456789012:configuration:c-fa3390a5-7e01-4559-
ae0c-eb15b38b22ca",
    "Created": "2025-07-17T16:57:04.520931+00:00",
    "Id": "c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca",
    "LatestRevision": {
        "Created": "2025-07-17T16:57:39.172000+00:00",
        "Revision": 2
    },
```

```
    "Name": "rabbitmq-ssl-config",
    "Warnings": []
}
```

4. 이 절차의 2단계에서 생성한 SSL 인증서 인증 구성을 사용하여 브로커를 생성합니다. 이렇게 하려면 다음 예제와 같이 `create-broker` AWS CLI 명령을 사용합니다. 이 명령에서 1단계와 2단계의 응답에서 얻은 구성 ID와 개정 번호를 각각 입력합니다. 예: `c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca` 및 2.

```
aws mq create-broker \
  --broker-name "rabbitmq-ssl-test-1" \
  --engine-type "RABBITMQ" \
  --engine-version "4.2" \
  --host-instance-type "mq.m7g.large" \
  --deployment-mode "SINGLE_INSTANCE" \
  --logs '{"General": true}' \
  --publicly-accessible \
  --configuration '{"Id": "<c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca>","Revision":
 <2>}' \
  --users '[{"Username":"testuser","Password":"testpassword"}]'
```

이 명령은 다음 예제와 유사한 응답을 반환합니다.

```
{
    "BrokerArn": "arn:aws:mq:us-west-2:123456789012:broker:rabbitmq-ssl-
test-1:b-2a1b5133-a10c-49d2-879b-8c176c34cf73",
    "BrokerId": "b-2a1b5133-a10c-49d2-879b-8c176c34cf73"
}
```

5. 다음 예제와 같이 `describe-broker` AWS CLI 명령을 `RUNNING`사용하여 브로커의 상태가 `CREATION_IN_PROGRESS`에서 로 전환되는지 확인합니다. 이 명령에서 이전 단계의 결과에서 얻은 브로커 ID를 제공합니다. 예를 들어 `b-2a1b5133-a10c-49d2-879b-8c176c34cf73`입니다.

```
aws mq describe-broker \
```

```
  --broker-id "<b-2a1b5133-a10c-49d2-879b-8c176c34cf73>"
```

이 명령은 다음 예제와 유사한 응답을 반환합니다. 다음 응답은 `describe-broker` 명령이 반환하는 전체 출력의 약식 버전입니다. 이 응답은 브로커 상태와 브로커를 보호하는 데 사용되는 인증 전략을 보여줍니다. 이 경우 `config_managed` 인증 전략은 브로커가 SSL 인증서 인증 방법을 사용함을 나타냅니다.

```
{
    "AuthenticationStrategy": "config_managed",
    ...,
    "BrokerState": "RUNNING",
    ...
}
```

6. 다음 `ssl.sh` 스크립트를 사용하여 SSL 인증서 인증을 확인합니다.

   이 bash 스크립트를 사용하여 RabbitMQ용 Amazon MQ 브로커에 대한 연결을 테스트합니다. 이 스크립트는 클라이언트 인증서를 인증에 사용하고 연결이 제대로 구성되었는지 확인합니다. 성공적으로 구성되면 브로커가 메시지를 게시하고 소비하는 것을 볼 수 있습니다.

   `ACCESS_REFUSED` 오류가 발생하면 브로커에 대한 CloudWatch 로그를 사용하여 구성 설정 문제를 해결할 수 있습니다. Amazon MQ 콘솔에서 브로커의 CloudWatch 로그 그룹에 대한 링크를 찾을 수 있습니다.

   이 스크립트에서는 다음 값을 제공해야 합니다.

   - `USERNAME`: 클라이언트 인증서의 일반 이름(CN)입니다.
   - `CLIENT_KEYSTORE`: 클라이언트 키 스토어 파일(PKCS12 형식)의 경로입니다. 필수 CDK 스택을 사용한 경우 기본 경로는 입니다`$(pwd)/certs/client-keystore.p12`.
   - `KEYSTORE_PASSWORD`: 클라이언트 키 스토어의 암호입니다. 필수 CDK 스택을 사용한 경우 기본 암호는 입니다`changeit`.
   - `BROKER_DNS`: Amazon MQ 콘솔의 브로커 세부 정보 페이지에 있는 연결에서이 값을 찾을 수 있습니다.

```
#! /bin/bash
set -e
```

```
# Client information
## FIXME: Update this value with the client ID and secret of your confidential
 application client
USERNAME=<client_cert_common_name>
CLIENT_KEYSTORE=$(pwd)/certs/client-keystore.p12
KEYSTORE_PASSWORD=changeit

BROKER_DNS=<broker_dns>
CONNECTION_STRING=amqps://${BROKER_DNS}:5671

# Produce/consume messages using the above connection string
QUEUES_COUNT=1
PRODUCERS_COUNT=1
CONSUMERS_COUNT=1
PRODUCER_RATE=1

finch run --rm --ulimit nofile=40960:40960 \
    -v ${CLIENT_KEYSTORE}:/certs/client-keystore.p12:ro \
    -e JAVA_TOOL_OPTIONS="-Djavax.net.ssl.keyStore=/certs/client-
keystore.p12 -Djavax.net.ssl.keyStorePassword=${KEYSTORE_PASSWORD} -
Djavax.net.ssl.keyStoreType=PKCS12" \
    pivotalrabbitmq/perf-test:latest \
    --queue-pattern 'test-queue-cert-%d' --queue-pattern-from 1 --queue-pattern-to
 $QUEUES_COUNT \
    --producers $PRODUCERS_COUNT --consumers $CONSUMERS_COUNT \
    --id "cert-test${QUEUES_COUNT}q${PRODUCERS_COUNT}p${CONSUMERS_COUNT}c
${PRODUCER_RATE}r" \
    --uri ${CONNECTION_STRING} \
    --sasl-external \
    --use-default-ssl-context \
    --flag persistent --rate $PRODUCER_RATE
```

AMQP 및 관리 엔드포인트에 mTLS 사용

이 자습서에서는 프라이빗 인증 기관을 사용하여 AMQP 클라이언트 연결 및 RabbitMQ 관리 인터페이스에 대해 상호 TLS(mTLS)를 구성하는 방법을 설명합니다.

> **Note**
>
> mTLS에 대한 프라이빗 인증 기관 사용은 RabbitMQ용 Amazon MQ 버전 4 이상에서만 사용할 수 있습니다. RabbitMQ

이 페이지의 내용

mTLS를 구성하기 위한 사전 조건

와의 RabbitMQ mTLS 통합을 위한 Amazon MQ용 CDK 스택을 배포하여이 자습서에 필요한 AWS 리소스를 설정할 수 있습니다. AWS Amazon MQ RabbitMQ

이 CDK 스택은 인증 기관, 클라이언트 인증서 및 IAM 역할을 포함하여 필요한 모든 AWS 리소스를 자동으로 생성합니다. 스택에서 생성한 리소스의 전체 목록은 패키지 README를 참조하세요.

CDK 스택을 사용하는 대신 리소스를 수동으로 설정하는 경우 RabbitMQ용 Amazon MQ 브로커에서 mTLS를 구성하기 전에 동등한 인프라가 있는지 확인합니다.

Amazon MQ를 설정하기 위한 사전 조건

AWS CLI 버전 >= 2.28.23: 브로커 생성 중에 사용자 이름과 암호를 선택적으로 추가할 수 있습니다.

AWS CLI를 사용하여 RabbitMQ에서 mTLS 구성

이 절차에서는 AWS CLI를 사용하여 필요한 리소스를 생성하고 구성합니다. 다음 절차에서는 configurationID 및 개정, <c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca>와 같은 자리 표시자 값을 실제 값으로 바꿔야 <2>합니다.

1. 다음 예제와 같이 `create-configuration` AWS CLI 명령을 사용하여 새 구성을 생성합니다.

```
aws mq create-configuration \
  --name "rabbitmq-mtls-config" \
  --engine-type "RABBITMQ" \
  --engine-version "4.2"
```

이 명령은 다음 예제와 유사한 응답을 반환합니다.

```
{
    "Arn": "arn:aws:mq:us-west-2:123456789012:configuration:c-fa3390a5-7e01-4559-
ae0c-eb15b38b22ca",
    "AuthenticationStrategy": "simple",
    "Created": "2025-07-17T16:03:01.759943+00:00",
    "Id": "c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca",
    "LatestRevision": {
        "Created": "2025-07-17T16:03:01.759000+00:00",
        "Description": "Auto-generated default for rabbitmq-mtls-config on RabbitMQ
 4.2",
        "Revision": 1
    },
    "Name": "rabbitmq-mtls-config"
}
```

2. 다음 예제와 같이 라는 구성 파일을 생성`rabbitmq.conf`하여 AMQP 및 관리 엔드포인트에 대한 mTLS를 구성합니다. 템플릿의 모든 자리 표시자 값(로 표시됨`${...}`)을 배포된 AWS CDK 사전 조건 스택 출력 또는 이에 상응하는 인프라의 실제 값으로 바꿉니다.

```
auth_backends.1 = internal

# TLS configuration
ssl_options.verify = verify_peer
ssl_options.fail_if_no_peer_cert = true
management.ssl.verify = verify_peer

# AWS integration for secure credential retrieval
# For more information, see https://github.com/amazon-mq/rabbitmq-aws

# FIXME: Replace the ${...} placeholders with actual ARN values
# from your deployed prerequisite CDK stack outputs.
aws.arns.assume_role_arn = ${AmazonMqAssumeRoleArn}
aws.arns.ssl_options.cacertfile = ${CaCertArn}
aws.arns.management.ssl.cacertfile = ${CaCertArn}
```

3. 다음 예제와 같이 update-configuration AWS CLI 명령을 사용하여 구성을 업데이트합니다. 이 명령에서 이 절차의 1단계에 대한 응답으로 받은 구성 ID를 추가합니다. 예를 들어 c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca입니다.

```
aws mq update-configuration \
  --configuration-id "<c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca>" \
  --data "$(cat rabbitmq.conf | base64 --wrap=0)"
```

이 명령은 다음 예제와 유사한 응답을 반환합니다.

```
{
    "Arn": "arn:aws:mq:us-west-2:123456789012:configuration:c-fa3390a5-7e01-4559-
ae0c-eb15b38b22ca",
    "Created": "2025-07-17T16:57:04.520931+00:00",
    "Id": "c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca",
    "LatestRevision": {
        "Created": "2025-07-17T16:57:39.172000+00:00",
        "Revision": 2
    },
    "Name": "rabbitmq-mtls-config",
    "Warnings": []
}
```

4. 이 절차의 2단계에서 생성한 mTLS 구성으로 브로커를 생성합니다. 이렇게 하려면 다음 예제와 같이 create-broker AWS CLI 명령을 사용합니다. 이 명령에서 1단계와 2단계의 응답에서 얻은 구성 ID와 개정 번호를 각각 입력합니다. 예: c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca 및 2.

```
aws mq create-broker \
  --broker-name "rabbitmq-mtls-test-1" \
  --engine-type "RABBITMQ" \
  --engine-version "4.2" \
  --host-instance-type "mq.m7g.large" \
  --deployment-mode "SINGLE_INSTANCE" \
  --logs '{"General": true}' \
  --publicly-accessible \
```

```
  --configuration '{"Id": "<c-fa3390a5-7e01-4559-ae0c-eb15b38b22ca>","Revision":
 <2>}' \
  --users '[{"Username":"testuser","Password":"testpassword"}]'
```

이 명령은 다음 예제와 유사한 응답을 반환합니다.

```
{
    "BrokerArn": "arn:aws:mq:us-west-2:123456789012:broker:rabbitmq-mtls-
test-1:b-2a1b5133-a10c-49d2-879b-8c176c34cf73",
    "BrokerId": "b-2a1b5133-a10c-49d2-879b-8c176c34cf73"
}
```

5. 다음 예제와 같이 describe-broker AWS CLI 명령을 RUNNING사용하여 브로커의 상태가 CREATION_IN_PROGRESS에서 로 전환되는지 확인합니다. 이 명령에서 이전 단계의 결과에서 얻은 브로커 ID를 제공합니다. 예를 들어 b-2a1b5133-a10c-49d2-879b-8c176c34cf73입니다.

```
aws mq describe-broker \
  --broker-id "<b-2a1b5133-a10c-49d2-879b-8c176c34cf73>"
```

이 명령은 다음 예제와 유사한 응답을 반환합니다. 다음 응답은 describe-broker 명령이 반환하는 전체 출력의 약식 버전입니다.

```
{
    "AuthenticationStrategy": "simple",
    ...,
    "BrokerState": "RUNNING",
    ...
}
```

6. 다음 스크립트를 사용하여 mtls.sh mTLS 인증을 확인합니다.

이 bash 스크립트를 사용하여 RabbitMQ용 Amazon MQ 브로커에 대한 연결을 테스트합니다. 이 스크립트는 클라이언트 인증서를 사용하여 연결을 인증하고 연결이 제대로 구성되었는지 확인합니다. 성공적으로 구성되면 브로커가 메시지를 게시하고 소비하는 것을 볼 수 있습니다.

ACCESS_REFUSED 오류가 발생하면 브로커에 대한 CloudWatch 로그를 사용하여 구성 설정 문제를 해결할 수 있습니다. Amazon MQ 콘솔에서 브로커의 CloudWatch 로그 그룹에 대한 링크를 찾을 수 있습니다.

이 스크립트에서는 다음 값을 제공해야 합니다.

- USERNAME 및 PASSWORD: 브로커로 생성한 RabbitMQ 사용자 자격 증명입니다.
- CLIENT_KEYSTORE: 클라이언트 키 스토어 파일(PKCS12 형식)의 경로입니다. 필수 CDK 스택을 사용한 경우 기본 경로는 입니다`$(pwd)/certs/client-keystore.p12`.
- KEYSTORE_PASSWORD: 클라이언트 키 스토어의 암호입니다. 필수 CDK 스택을 사용한 경우 기본 암호는 입니다`changeit`.
- BROKER_DNS: Amazon MQ 콘솔의 브로커 세부 정보 페이지에 있는 연결에서이 값을 찾을 수 있습니다.

```
#! /bin/bash
set -e

# Client information
## FIXME: Update this value with the client ID and secret of your confidential
 application client
USERNAME=<testuser>
PASSWORD=<testpassword>
CLIENT_KEYSTORE=$(pwd)/certs/client-keystore.p12
KEYSTORE_PASSWORD=changeit

BROKER_DNS=<broker_dns>
CONNECTION_STRING=amqps://${USERNAME}:${PASSWORD}@${BROKER_DNS}:5671

# Produce/consume messages using the above connection string
QUEUES_COUNT=1
PRODUCERS_COUNT=1
CONSUMERS_COUNT=1
PRODUCER_RATE=1

finch run --rm --ulimit nofile=40960:40960 \
```

```
    -v ${CLIENT_KEYSTORE}:/certs/client-keystore.p12:ro \
    -e JAVA_TOOL_OPTIONS="-Djavax.net.ssl.keyStore=/certs/client-
keystore.p12 -Djavax.net.ssl.keyStorePassword=${KEYSTORE_PASSWORD} -
Djavax.net.ssl.keyStoreType=PKCS12" \
    pivotalrabbitmq/perf-test:latest \
    --queue-pattern 'test-queue-cert-%d' --queue-pattern-from 1 --queue-pattern-to
 $QUEUES_COUNT \
    --producers $PRODUCERS_COUNT --consumers $CONSUMERS_COUNT \
    --id "cert-test${QUEUES_COUNT}q${PRODUCERS_COUNT}p${CONSUMERS_COUNT}c
${PRODUCER_RATE}r" \
    --uri ${CONNECTION_STRING} \
    --use-default-ssl-context \
    --flag persistent --rate $PRODUCER_RATE
```

JMS 애플리케이션 연결

이 자습서에서는 RabbitMQ Amazon MQ RabbitMQ 브로커에 연결하는 방법을 보여줍니다. 메시지를 보낼 생산자와 RabbitMQ 대기열에서 메시지를 받을 소비자를 생성하는 방법을 알아봅니다.

시작하기 전에 Maven 프로젝트에 적절한 RabbitMQ JMS 종속성을 추가합니다.

JMS 1.1 및 2.0의 경우:

```
<dependencies>

  <dependency>
    <groupId>com.rabbitmq.jms</groupId>
    <artifactId>rabbitmq-jms</artifactId>
    <version>2.12.0</version>
  </dependency>

</dependencies>
```

JMS 3.1의 경우:

```
<dependencies>

  <dependency>
    <groupId>com.rabbitmq.jms</groupId>
    <artifactId>rabbitmq-jms</artifactId>
    <version>3.5.0</version>
  </dependency>
```

```
</dependencies>
```

생산자 생성

다음 코드 예제에서는 JMS를 사용하여 RabbitMQ 대기열에 쓰는 방법을 보여줍니다.

```
import jakarta.jms.*;
import com.rabbitmq.jms.admin.*;

// Setting the connection factory
RMQConnectionFactory factory = new RMQConnectionFactory();
factory.setHost(envProps.getProperty("RABBITMQ_HOST", "localhost"));
factory.setPort(Integer.parseInt(envProps.getProperty("RABBITMQ_PORT", "5672")));
factory.setUsername(envProps.getProperty("RABBITMQ_USERNAME", "guest"));
factory.setPassword(envProps.getProperty("RABBITMQ_PASSWORD", "guest"));
factory.setVirtualHost(envProps.getProperty("RABBITMQ_VIRTUAL_HOST", "/"));
factory.useSslProtocol();

connection = factory.createConnection();
connection.start();

String queueName = "test-queue-jms";
Session session = connection.createSession(false, Session.AUTO_ACKNOWLEDGE);

RMQDestination destination = new RMQDestination(queueName, true, false);

// Send the message to the queue
MessageProducer producer = session.createProducer(destination);
producer.setDeliveryMode(DeliveryMode.PERSISTENT);

String msg_content = "Hello World!!";
TextMessage textMessage = session.createTextMessage(msg_content);
producer.send(textMessage);

System.out.printf("Published to AMQP queue '%s': %s", queueName, msg_content);
```

소비자 생성

다음 코드 예제에서는 JMS를 사용하여 RabbitMQ 대기열에서 읽는 방법을 보여줍니다.

```
import jakarta.jms.*;
import com.rabbitmq.jms.admin.*;
```

```
// Setting the connection factory
RMQConnectionFactory factory = new RMQConnectionFactory();
factory.setHost(envProps.getProperty("RABBITMQ_HOST", "localhost"));
factory.setPort(Integer.parseInt(envProps.getProperty("RABBITMQ_PORT", "5672")));
factory.setUsername(envProps.getProperty("RABBITMQ_USERNAME", "guest"));
factory.setPassword(envProps.getProperty("RABBITMQ_PASSWORD", "guest"));
factory.setVirtualHost(envProps.getProperty("RABBITMQ_VIRTUAL_HOST", "/"));
factory.useSslProtocol();

// Establish the connection and session
jakarta.jms.Connection connection = factory.createConnection();

String queueName = "test-queue-jms";
Session session = connection.createSession(false, Session.AUTO_ACKNOWLEDGE);

RMQDestination destination = new RMQDestination();
destination.setDestinationName(queueName);
destination.setAmqp(true);
destination.setAmqpQueueName(queueName);

// Initialize consumer
MessageConsumer consumer = session.createConsumer(destination);
consumer.setMessageListener(message -> {
    try {
        if (message instanceof TextMessage) {
            TextMessage textMessage = (TextMessage) message;
            System.out.printf("Message: %s%n", textMessage.getText());
        } else if (message instanceof BytesMessage) {
            BytesMessage bytesMessage = (BytesMessage) message;
            byte[] bytes = new byte[(int) bytesMessage.getBodyLength()];
            bytesMessage.readBytes(bytes);
            String content = new String(bytes);
            System.out.printf("Message: %s%n", content);
        } else {
            System.out.printf("Message: [%s]%n", message.getClass().getSimpleName());
        }
    } catch (JMSException e) {
        System.err.printf("Error processing message: %s%n", e.getMessage());
    }
});

connection.start();
```

Amazon MQ의 보안

의 클라우드 보안 AWS 이 최우선 순위입니다. AWS 고객은 보안에 가장 민감한 조직의 요구 사항을 충족하도록 구축된 데이터 센터 및 네트워크 아키텍처의 이점을 누릴 수 있습니다.

보안은 AWS 와 사용자 간의 공동 책임입니다. 공동 책임 모델은 이 사항을 클라우드의 보안 및 클라우드 내 보안으로 설명합니다.

- 클라우드 보안 - AWS 는 AWS 클라우드에서 AWS 서비스를 실행하는 인프라를 보호할 책임이 있습니다. AWS 또한는 안전하게 사용할 수 있는 서비스를 제공합니다. 타사 감사자는 AWS 규정 준수 프로그램 일환으로 보안의 효과를 정기적으로 테스트하고 확인합니다. Amazon MQ에 적용되는 규정 준수 프로그램에 대한 자세한 내용은 규정 준수 프로그램 AWS 제공 범위 내 서비스규정 준수 프로그램 .
- 클라우드의 보안 - 사용자의 책임은 사용하는 AWS 서비스에 따라 결정됩니다. 또한 귀하는 귀사의 데이터 민감도, 귀사의 요구 사항, 관련 법률 및 규정을 비롯한 기타 요소에 대해서도 책임이 있습니다.

이 설명서는 Amazon MQ를 사용할 때 공동 책임 모델을 적용하는 방법을 이해하는 데 도움이 됩니다. 다음 주제에서는 보안 및 규정 준수 목적에 맞게 Amazon MQ를 구성하는 방법을 보여줍니다. 또한 Amazon MQ 리소스를 모니터링하고 보호하는 데 도움이 되는 다른 AWS 서비스를 사용하는 방법을 알아봅니다.

주제

- Amazon MQ의 데이터 보호
- Amazon MQ의 Identity and Access Management
- Amazon MQ에 대한 규정 준수 확인
- Amazon MQ의 복원성
- Amazon MQ의 인프라 보안
- Amazon MQ에 대한 보안 모범 사례

Amazon MQ의 데이터 보호

AWS 공동 책임 모델 Amazon MQ의 데이터 보호에 적용됩니다. 이 모델에 설명된 대로 AWS 는 모든 를 실행하는 글로벌 인프라를 보호할 책임이 있습니다 AWS 클라우드. 사용자는 이 인프라에 호스팅되는 콘텐츠에 대한 통제 권한을 유지할 책임이 있습니다. 사용하는 AWS 서비스 의 보안 구성과 관리 태

스크에 대한 책임도 사용자에게 있습니다. 데이터 프라이버시에 관한 자세한 내용은 데이터 프라이버시 FAQ를 참조하세요. 유럽의 데이터 보호에 대한 자세한 내용은 AWS 보안 블로그의 AWS 공동 책임 모델 및 GDPR 블로그 게시물을 참조하세요.

데이터 보호를 위해 자격 증명을 보호하고 AWS 계정 AWS IAM Identity Center 또는 AWS Identity and Access Management (IAM)를 사용하여 개별 사용자를 설정하는 것이 좋습니다. 이렇게 하면 개별 사용자에게 자신의 직무를 충실히 이행하는 데 필요한 권한만 부여됩니다. 또한 다음과 같은 방법으로 데이터를 보호하는 것이 좋습니다.

- 각 계정에 다중 인증(MFA)을 사용합니다.
- SSL/TLS를 사용하여 AWS 리소스와 통신합니다. TLS 1.2는 필수이며 TLS 1.3을 권장합니다.
- 를 사용하여 API 및 사용자 활동 로깅을 설정합니다 AWS CloudTrail. CloudTrail 추적을 사용하여 AWS 활동을 캡처하는 방법에 대한 자세한 내용은 AWS CloudTrail 사용 설명서의 CloudTrail 추적 작업을 참조하세요.
- 내의 모든 기본 보안 제어와 함께 AWS 암호화 솔루션을 사용합니다 AWS 서비스.
- Amazon S3에 저장된 민감한 데이터를 검색하고 보호하는 데 도움이 되는 Amazon Macie와 같은 고급 관리형 보안 서비스를 사용합니다.
- 명령줄 인터페이스 또는 API를 AWS 통해에 액세스할 때 FIPS 140-3 검증 암호화 모듈이 필요한 경우 FIPS 엔드포인트를 사용합니다. 사용 가능한 FIPS 엔드포인트에 대한 자세한 내용은 연방 정보 처리 표준(FIPS) 140-3을 참조하세요.

고객의 이메일 주소와 같은 기밀 정보나 중요한 정보는 태그나 이름 필드와 같은 자유 형식 텍스트 필드에 입력하지 않는 것이 좋습니다. 여기에는 Amazon MQ 또는 기타 AWS 서비스 에서 콘솔 AWS CLI, API 또는 AWS SDKs를 사용하여 작업하는 경우가 포함됩니다. 이름에 사용되는 태그 또는 자유 형식 텍스트 필드에 입력하는 모든 데이터는 청구 또는 진단 로그에 사용될 수 있습니다. 외부 서버에 URL을 제공할 때 해당 서버에 대한 요청을 검증하기 위해 자격 증명을 URL에 포함해서는 안 됩니다.

Amazon MQ for ActiveMQ와 Amazon MQ for RabbitMQ 브로커용의 경우 브로커 웹 콘솔 또는 Amazon MQ API를 통해 리소스를 생성할 때 브로커 이름이나 사용자 이름에 개인 식별 정보(PII) 또는 기타 기밀 정보 또는 민감한 정보를 사용하지 마십시오. 브로커 이름과 사용자 이름은 CloudWatch Logs를 포함한 다른 AWS 서비스에서 액세스할 수 있습니다. 브로커 사용자 이름은 개인 데이터나 민감한 데이터에 사용하기 위한 것이 아닙니다.

> **Important**
>
> RabbitMQ 브로커에는 TLS 1.3을 사용할 수 없습니다.

암호화(Encryption)

Amazon MQ에 저장된 사용자 데이터는 유휴 시 암호화됩니다. Amazon MQ 유휴 시 암호화는 AWS Key Management Service (KMS)에 저장된 암호화 키로 데이터를 암호화하여 향상된 보안을 제공합니다. 이 서비스를 사용하면 중요한 데이터 보호와 관련된 운영 부담 및 복잡성을 줄일 수 있습니다. 유휴 시 암호화를 사용하면 암호화 규정 준수 및 규제 요구 사항이 필요한, 보안에 민감한 애플리케이션을 구축할 수 있습니다.

Amazon MQ 브로커 사이의 모든 연결에서는 전송 계층 보안(TLS)을 사용하여 전송 중 암호화를 제공합니다.

Amazon MQ는 유휴 상태이거나 전송 중인 메시지를 암호화 키로 암호화하여 안전하게 관리 및 저장합니다. 자세한 내용은 AWS Encryption SDK 개발자 안내서를 참조하세요.

저장 시 암호화

Amazon MQ는 AWS Key Management Service (KMS)와 통합되어 투명한 서버 측 암호화를 제공합니다. Amazon MQ는 유휴 시 데이터를 항상 암호화합니다.

ActiveMQ용 Amazon MQ 브로커 또는 RabbitMQ용 Amazon MQ 브로커를 생성할 때 Amazon MQ AWS KMS key 가 저장 데이터를 암호화하는 데 사용할를 지정할 수 있습니다. KMS 키를 지정하지 않으면 Amazon MQ는 사용자를 대신하여 AWS 소유 KMS 키를 생성하고 사용합니다. Amazon MQ는 현재 대칭 KMS 키를 지원합니다. KMS 키에 대한 자세한 내용은 AWS KMS keys 단원을 참조하세요.

브로커를 생성할 때 다음 중 하나를 선택하여 Amazon MQ가 암호화 키로 사용할 항목을 구성할 수 있습니다.

- Amazon MQ 소유 KMS 키(Amazon MQ owned KMS key)(기본값) - 이 키는 Amazon MQ가 소유하고 관리하며 사용자 계정에 없습니다.
- AWS 관리형 KMS 키 - AWS 관리형 KMS 키(`aws/mq`)는 Amazon MQ에서 사용자를 대신하여 생성, 관리 및 사용하는 계정의 KMS 키입니다.
- 기존 고객 관리형 KMS 키 선택 - 고객 관리형 KMS 키는 사용자가 AWS Key Management Service (KMS)에서 생성하고 관리합니다.

> ⚠ Important
>
> - 권한 부여 취소는 실행 취소할 수 없습니다. 브로커를 삭제하여 액세스 권한을 취소합니다.

- Amazon Elastic File System(EFS)을 사용하여 메시지 데이터를 저장하는 ActiveMQ용 Amazon MQ 브로커의 경우 필요한 작업을 수행한 후 계정에서 KMS 키를 사용하는 권한이 취소되는 데 몇 시간이 걸릴 수 있습니다.
- EBS를 사용하여 메시지 데이터를 저장하는 RabbitMQ용 Amazon MQ 및 ActiveMQ용 Amazon MQ 브로커의 경우 Amazon EBS에 사용자 계정의 KMS 키를 사용하도록 부여한 권한을 비활성화하거나 삭제 예약하거나 취소하면 Amazon MQ가 브로커를 유지하지 못하고 성능 저하 상태로 변경될 수 있습니다.
- 키를 비활성화했거나 키 삭제를 예약한 경우 키를 다시 활성화하거나 키 삭제를 취소하고 브로커를 유지할 수 있습니다.
- 필요한 작업을 수행한 후 키를 비활성화하거나 권한 부여를 취소하는 데 몇 시간이 걸릴 수 있습니다.
- CloudWatch Logs 암호화 또는 복호화의 경우 Amazon MQ가 암호화 키에 사용하는 항목을 구성할 수 없습니다. CloudWatch Logs는 암호화를 사용하여 저장 데이터를 보호하고 로그 그룹은 암호화됩니다. CloudWatch Logs 서비스는 기본적으로 서버 측 암호화를 관리합니다. 로그 그룹을 암호화하는 방법에 대한 자세한 내용은 Amazon CloudWatch Logs 사용 설명서를 참조하세요.

RabbitMQ용 KMS 키로 단일 인스턴스 브로커를 생성하면 두 개의 `CreateGrant` 이벤트가 AWS CloudTrail에 로깅된 것을 볼 수 있습니다. 첫 번째 이벤트는 Amazon MQ에서 KMS 키에 대한 권한을 생성하는 이벤트입니다. 두 번째 이벤트는 EBS가 EBS에서 사용할 권한을 생성하는 이벤트입니다.

CreateGrant AWS CloudTrail 로그 항목: 단일 인스턴스 브로커

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 권한 생성에 대한 이벤트가 하나 표시됩니다.

```
                                {
    "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"
}
```

RabbitMQ용 KMS 키를 사용하여 클러스터 배포를 생성하면 5개의 `CreateGrant` 이벤트가 AWS CloudTrail에 로깅된 것을 볼 수 있습니다. 처음 두 가지 이벤트는 Amazon MQ에 대한 권한을 생성하는 이벤트입니다. 다음 세 가지 이벤트는 EBS가 EBS에서 사용할 권한을 생성하는 이벤트입니다.

CreateGrant AWS CloudTrail 로그 항목: 클러스터 배포

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 권한 생성에 대한 이벤트가 3개 표시됩니다.

```
                                    {
    "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 키에 대한 자세한 내용은 AWS Key Management Service 개발자 안내서에서 AWS KMS keys 단원을 참조하세요.

전송 중 암호화

ActiveMQ용 Amazon MQ: ActiveMQ용 Amazon MQ는 강력한 전송 계층 보안(TLS)을 요구하며 사용자의 Amazon MQ 배포 브로커 간에 전송 중 데이터를 암호화합니다. Amazon MQ 브로커 사이를 통과하는 모든 데이터는 강력한 전송 계층 보안(TLS)을 사용하여 암호화됩니다. 이것은 사용 가능한 모든 프로토콜에 적용됩니다.

RabbitMQ용 Amazon MQ: RabbitMQ용 Amazon MQ는 모든 클라이언트 연결에 강력한 전송 계층 보안(TLS) 암호화를 요구합니다. RabbitMQ 클러스터 복제 트래픽은 브로커의 VPC만 전송하며 AWS 데이터 센터 간의 모든 네트워크 트래픽은 물리적 계층에서 투명하게 암호화됩니다. RabbitMQ용 Amazon MQ 클러스터 브로커는 현재 클러스터 복제에 노드 간 암호화를 지원하지 않습니다. 전송 중 데이터에 대한 자세한 내용은 저장 데이터 및 전송 중 데이터 암호화를 참조하세요.

ActiveMQ용 Amazon MQ 프로토콜

TLS가 활성화된 상태에서 다음 프로토콜을 사용하여 ActiveMQ 브로커에 액세스할 수 있습니다.

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

ActiveMQ에 대해 지원되는 TLS 암호 제품군

Amazon MQ에서 ActiveMQ는 다음 암호 제품군을 지원합니다.

- 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

RabbitMQ용 Amazon MQ 프로토콜

TLS가 활성화된 상태에서 다음 프로토콜을 사용하여 RabbitMQ 브로커에 액세스할 수 있습니다.

- AMQP(0-9-1)

RabbitMQ에 대해 지원되는 TLS 암호 제품군

Amazon MQ에서 RabbitMQ는 다음 암호 제품군을 지원합니다.

- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384

- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256

Amazon MQ의 Identity and Access Management

AWS Identity and Access Management (IAM)는 관리자가 AWS 리소스에 대한 액세스를 안전하게 제어하는 데 도움이 AWS 서비스 되는 입니다. IAM 관리자는 어떤 사용자가 Amazon MQ 리소스를 사용할 수 있는 인증(로그인) 및 권한(권한 있음)을 받을 수 있는지 제어합니다. IAM은 추가 비용 없이 사용할 수 AWS 서비스 있는 입니다.

주제

대상

AWS Identity and Access Management (IAM)를 사용하는 방법은 역할에 따라 다릅니다.

- 서비스 사용자 - 기능에 액세스할 수 없는 경우 관리자에게 권한 요청(참조Amazon MQ 자격 증명 및 액세스 문제 해결)
- 서비스 관리자 - 사용자 액세스 결정 및 권한 요청 제출(Amazon MQ에서 IAM을 사용하는 방법 참조)
- IAM 관리자 - 액세스를 관리하기 위한 정책 작성(Amazon MQ 자격 증명 기반 정책 예제 참조)

ID를 통한 인증

인증은 자격 증명 자격 증명을 AWS 사용하여에 로그인하는 방법입니다. AWS 계정 루트 사용자, IAM 사용자 또는 IAM 역할을 수임하여 인증되어야 합니다.

AWS IAM Identity Center (IAM Identity Center), Single Sign-On 인증 또는 Google/Facebook 자격 증명과 같은 자격 증명 소스의 자격 증명을 사용하여 페더레이션 자격 증명으로 로그인할 수 있습니다. 로그인하는 방법에 대한 자세한 내용은 AWS 로그인 사용 설명서의 AWS 계정에 로그인하는 방법 섹션을 참조하세요.

프로그래밍 방식 액세스를 위해는 요청에 암호화 방식으로 서명할 수 있는 SDK 및 CLI를 AWS 제공합니다. 자세한 내용은 IAM 사용 설명서의 API 요청용AWS Signature Version 4 섹션을 참조하세요.

AWS 계정 루트 사용자

를 생성할 때 모든 AWS 서비스 및 리소스에 대한 완전한 액세스 권한이 있는 AWS 계정 theroot 사용자라는 하나의 로그인 자격 증명으로 AWS 계정시작합니다. 일상적인 태스크에 루트 사용자를 사용하지 않을 것을 강력히 권장합니다. 루트 사용자 자격 증명이 필요한 작업은 IAM 사용 설명서의 루트 사용자 자격 증명이 필요한 작업 섹션을 참조하세요.

사용자 및 그룹

IAM 사용자는 단일 개인 또는 애플리케이션에 대한 특정 권한을 가진 ID입니다. 장기 자격 증명이 있는 IAM 사용자 대신 임시 자격 증명을 사용하는 것이 좋습니다. 자세한 내용은 IAM 사용 설명서의 자격 증명 공급자와의 페더레이션을 사용하여 임시 자격 증명을 AWS 사용하여에 액세스하도록 인간 사용자에게 요구 섹션을 참조하세요.

IAM 그룹은 IAM 사용자 모음을 지정하고 대규모 사용자 집합에 대한 관리 권한을 더 쉽게 만듭니다. 자세한 내용은 IAM 사용 설명서의 IAM 사용자 사용 사례 섹션을 참조하세요.

IAM 역할

IAM 역할은 임시 자격 증명을 제공하는 특정 권한이 있는 자격 증명입니다. 사용자에서 IAM 역할(콘솔)로 전환하거나 또는 API 작업을 호출하여 역할을 수임할 수 있습니다. AWS CLI AWS 자세한 내용은 IAM 사용 설명서의 역할 수임 방법을 참조하세요.

IAM 역할은 페더레이션 사용자 액세스, 임시 IAM 사용자 권한, 교차 계정 액세스, 교차 서비스 액세스 및 Amazon EC2에서 실행되는 애플리케이션에 유용합니다. 자세한 내용은 IAM 사용 설명서의 교차 계정 리소스 액세스를 참조하세요.

정책을 사용하여 액세스 관리

정책을 AWS 생성하고 자격 증명 또는 리소스에 연결하여 AWS 에서 액세스를 제어합니다. 정책은 자격 증명 또는 리소스와 연결될 때 권한을 정의합니다.는 보안 주체가 요청할 때 이러한 정책을 AWS 평가합니다. 대부분의 정책은에 JSON 문서 AWS 로 저장됩니다. JSON 정책 문서에 대한 자세한 내용은 IAM 사용 설명서의 JSON 정책 개요 섹션을 참조하세요.

정책을 사용하여 관리자는 어떤 보안 주체가 어떤 리소스에 대해 어떤 조건에서 작업을 수행할 수 있는지 정의하여 누가 무엇을 액세스할 수 있는지 지정합니다.

기본적으로 사용자 및 역할에는 어떠한 권한도 없습니다. IAM 관리자는 IAM 정책을 생성하고 사용자가 수임할 수 있는 역할에 추가합니다. IAM 정책은 작업을 수행하기 위해 사용하는 방법과 관계없이 작업에 대한 권한을 정의합니다.

ID 기반 정책

ID 기반 정책은 ID(사용자, 사용자 그룹 또는 역할)에 연결하는 JSON 권한 정책 문서입니다. 이러한 정책은 자격 증명이 수행할 수 있는 작업, 대상 리소스 및 이에 관한 조건을 제어합니다. ID 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서에서 고객 관리형 정책으로 사용자 지정 IAM 권한 정의를 참조하세요.

ID 기반 정책은 인라인 정책(단일 ID에 직접 포함) 또는 관리형 정책(여러 ID에 연결된 독립 실행형 정책)일 수 있습니다. 관리형 정책 또는 인라인 정책을 선택하는 방법을 알아보려면 IAM 사용 설명서의 관리형 정책 및 인라인 정책 중에서 선택 섹션을 참조하세요.

리소스 기반 정책

리소스 기반 정책은 리소스에 연결하는 JSON 정책 설명서입니다. 예를 들어 IAM 역할 신뢰 정책 및 Amazon S3 버킷 정책이 있습니다. 리소스 기반 정책을 지원하는 서비스에서 서비스 관리자는 이러한 정책을 사용하여 특정 리소스에 대한 액세스를 통제할 수 있습니다. 리소스 기반 정책에서 보안 주체를 지정해야 합니다.

리소스 기반 정책은 해당 서비스에 있는 인라인 정책입니다. 리소스 기반 정책에서는 IAM의 AWS 관리형 정책을 사용할 수 없습니다.

액세스 제어 목록(ACL)

액세스 제어 목록(ACL)은 어떤 위탁자(계정 멤버, 사용자 또는 역할)가 리소스에 액세스할 수 있는 권한을 가지고 있는지를 제어합니다. ACL은 JSON 정책 문서 형식을 사용하지 않지만 리소스 기반 정책과 유사합니다.

Amazon S3 AWS WAF및 Amazon VPC는 ACLs. ACL에 관한 자세한 내용은 Amazon Simple Storage Service 개발자 가이드의 액세스 제어 목록(ACL) 개요를 참조하세요.

기타 정책 타입

AWS 는 보다 일반적인 정책 유형에서 부여한 최대 권한을 설정할 수 있는 추가 정책 유형을 지원합니다.

- 권한 경계 - ID 기반 정책에서 IAM 엔터티에 부여할 수 있는 최대 권한을 설정합니다. 자세한 정보는 IAM 사용 설명서의 IAM 엔터티의 권한 범위를 참조하세요.
- 서비스 제어 정책(SCP) - AWS Organizations내 조직 또는 조직 단위에 대한 최대 권한을 지정합니다. 자세한 내용은AWS Organizations 사용 설명서의 서비스 제어 정책을 참조하세요.
- 리소스 제어 정책(RCP) – 계정의 리소스에 사용할 수 있는 최대 권한을 설정합니다. 자세한 내용은 AWS Organizations 사용 설명서의 리소스 제어 정책(RCP)을 참조하세요.
- 세션 정책 – 역할 또는 페더레이션 사용자에 대해 임시 세션을 프로그래밍 방식으로 생성할 때 파라미터로 전달하는 고급 정책입니다. 자세한 내용은 IAM 사용 설명서의 세션 정책을 참조하세요.

여러 정책 유형

여러 정책 유형이 요청에 적용되는 경우, 결과 권한은 이해하기가 더 복잡합니다. 에서 여러 정책 유형이 관련될 때 요청을 허용할지 여부를 AWS 결정하는 방법을 알아보려면 IAM 사용 설명서의 정책 평가 로직을 참조하세요.

Amazon MQ에서 IAM을 사용하는 방법

IAM을 사용하여 Amazon MQ에 대한 액세스를 관리하기 전에 Amazon MQ에서 사용할 수 있는 IAM 기능을 이해해야 합니다. Amazon MQ 및 기타 AWS 서비스에서 IAM을 사용하는 방법을 전체적으로 알아보려면 IAM 사용 설명서의 AWS IAM으로 작업하는 서비스를 참조하세요.

Amazon MQ는 Amazon MQ API 작업에 IAM을 사용하여 브로커를 생성, 업데이트, 삭제 및 나열합니다. 메시지 게시 및 구독을 위한 브로커 액세스의 경우 ActiveMQ용 Amazon MQ는 기본 ActiveMQ 인증 및 LDAP를 지원하는 반면 RabbitMQ용 Amazon MQ는 IAM 인증 및 기타 방법을 지원합니다. ActiveMQ 자세한 내용은 the section called “브로커 인증 및 권한 부여” 단원을 참조하십시오.

주제

- Amazon MQ 자격 증명 기반 정책
- Amazon MQ 리소스 기반 정책
- Amazon MQ 태그 기반 권한 부여
- Amazon MQ IAM 역할

Amazon MQ 자격 증명 기반 정책

IAM ID 기반 정책을 사용하면 허용되거나 거부되는 작업과 리소스뿐 아니라 작업이 허용되거나 거부되는 조건을 지정할 수 있습니다. Amazon MQ는 특정 작업, 리소스 및 조건 키를 지원합니다. JSON 정

책에서 사용하는 모든 요소에 대해 알고 싶다면 IAM 사용 설명서의 IAM JSON 정책 요소 참조를 참조하세요.

작업

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

JSON 정책의 `Action`요소는 정책에서 액세스를 허용하거나 거부하는 데 사용할 수 있는 작업을 설명합니다. 연결된 작업을 수행할 수 있는 권한을 부여하기 위한 정책에 작업을 포함하세요.

Amazon MQ의 정책 작업은 작업 앞에 `mq:` 접두사를 사용합니다. 예를 들어 누군가에게 Amazon MQ `CreateBroker` API 작업을 통해Amazon MQ 인스턴스를 실행하는 권한을 부여하려면 해당 정책에 `mq:CreateBroker` 작업을 포함합니다. 정책 문에는 `Action` 또는 `NotAction` 요소가 포함되어야 합니다. Amazon MQ는 이 서비스로 수행할 수 있는 태스크를 설명하는 고유한 작업 세트를 정의합니다.

명령문 하나에 여러 태스크를 지정하려면 다음과 같이 쉼표로 구분합니다.

```
"Action": [
      "mq:action1",
      "mq:action2"
```

와일드카드(*)를 사용하여 여러 작업을 지정할 수 있습니다. 예를 들어, `Describe`라는 단어로 시작하는 모든 작업을 지정하려면 다음 작업을 포함합니다.

```
"Action": "mq:Describe*"
```

Amazon MQ 작업의 목록을 보려면 IAM 사용 설명서의 Amazon MQ에서 정의한 작업을 참조하세요.

리소스

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

`Resource` JSON 정책 요소는 작업이 적용되는 하나 이상의 객체를 지정합니다. 모범 사례에 따라 Amazon 리소스 이름(ARN)을 사용하여 리소스를 지정합니다. 리소스 수준 권한을 지원하지 않는 작업의 경우, 와일드카드(*)를 사용하여 해당 문이 모든 리소스에 적용됨을 나타냅니다.

```
"Resource": "*"
```

Amazon MQ에서 기본 AWS 리소스는 Amazon MQ 메시지 브로커 및 해당 구성입니다. 다음 표와 같이 Amazon MQ 브로커 및 구성에는 고유한 Amazon 리소스 이름(ARN)이 연결되어 있습니다.

리소스 유형	ARN	조건 키
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 형식에 대한 자세한 내용은 Amazon 리소스 이름(ARNs) 및 AWS 서비스 네임스페이스를 참조하세요.

예를 들어 명령문에서 이름이 `MyBroker`이고 brokerId가 `b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9`인 브로커를 지정하려면 다음 ARN을 사용하세요.

```
"Resource": "arn:aws:mq:us-
east-1:123456789012:broker:MyBroker:b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9"
```

특정 계정에 속하는 모든 브로커 및 구성을 지정하려면 와일드카드(*)를 사용합니다.

```
"Resource": "arn:aws:mq:us-east-1:123456789012:*"
```

리소스 생성 작업과 같은 일부 Amazon MQ 작업은 특정 리소스에서 수행할 수 없습니다. 이러한 경우, 와일드카드(*)를 사용해야 합니다.

```
"Resource": "*"
```

API 작업 `CreateTags`에는 브로커와 구성이 모두 필요합니다. 단일 문에서 여러 리소스를 지정하려면 ARN을 쉼표로 구분합니다.

```
"Resource": [
      "resource1",
      "resource2"
```

Amazon MQ 리소스 유형 및 해당 ARN의 목록을 보려면 IAM 사용 설명서의 Amazon MQ에서 정의한 리소스를 참조하세요. 각 리소스의 ARN을 지정할 수 있는 작업을 알아보려면 Amazon MQ에서 정의한 작업을 참조하세요.

조건 키

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

`Condition` 요소는 정의된 기준에 따라 문이 실행되는 시기를 지정합니다. 같음(equals) 또는 미만(less than)과 같은 조건 연산자를 사용하여 정책의 조건을 요청의 값과 일치시키는 조건식을 생성할 수 있습니다. 모든 AWS 전역 조건 키를 보려면 IAM 사용 설명서의 AWS 전역 조건 컨텍스트 키를 참조하세요.

Amazon MQ는 서비스별 조건 키를 정의하지 않지만, 일부 전역 조건 키 사용을 지원합니다. Amazon MQ 조건 키 목록을 보려면 IAM 사용 설명서의 Amazon MQ의 조건 키를 참조하세요. 조건 키를 사용할 수 있는 작업과 리소스를 알아보려면 Amazon MQ에서 정의한 작업을 참조하세요.

조건 키	설명	형식
aws:RequestTag/${TagKey}	요청에서 전달되는 태그를 기준으로 작업을 필터링합니다.	문자열
aws:ResourceTag/${TagKey}	리소스와 연결된 태그를 기준으로 작업을 필터링합니다.	문자열
aws:TagKeys	요청에서 전달되는 태그 키를 기준으로 작업을 필터링합니다.	문자열

예제

Amazon MQ 자격 증명 기반 정책 예제를 보려면 Amazon MQ 자격 증명 기반 정책 예제 단원을 참조하세요.

Amazon MQ 리소스 기반 정책

현재 Amazon MQ는 리소스 기반 권한 또는 리소스 기반 정책을 사용한 IAM 인증을 지원하지 않습니다.

Amazon MQ 태그 기반 권한 부여

Amazon MQ 리소스에 태그를 연결하거나 Amazon MQ에 대한 요청에서 태그를 전달할 수 있습니다. 태그에 근거하여 액세스를 제어하려면 `mq:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`*또는 `aws:TagKeys`조건 키를 사용하여 정책의 조건 요소에 태그 정보를 제공합니다.

Amazon MQ는 태그를 기반으로 하는 정책을 지원합니다. 예를 들어, `environment` 키 및 `production` 값의 태그를 포함하는 Amazon MQ 리소스에 대한 액세스를 거부할 수 있습니다.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "mq:DeleteBroker",
                "mq:RebootBroker",
                "mq:DeleteTags"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/environment": "production"
                }
            }
        }
    ]
}
```

이 정책은 태그 `Deny`이 포함된 Amazon MQ 브로커를 삭제하거나 재부팅할 수 있는 기능을 `environment/production`합니다.

태그 지정에 대한 자세한 내용은 다음을 참조하세요.

- Amazon MQ 리소스에 태그 추가
- IAM 태그를 사용한 액세스 제어

Amazon MQ IAM 역할

IAM 역할은 특정 권한이 있는 AWS 계정 내 엔터티입니다.

Amazon MQ에서 임시 자격 증명 사용

임시 보안 인증을 사용하여 페더레이션을 통해 로그인하거나, IAM 역할을 맡거나, 교차 계정 역할을 맡을 수 있습니다. AssumeRole 또는 GetFederationToken과 같은 AWS STS API 작업을 호출하여 임시 보안 자격 증명을 얻습니다.

Amazon MQ는 임시 자격 증명 사용을 지원합니다.

서비스 역할

이 기능을 사용하면 서비스가 사용자를 대신하여 서비스 역할을 수임할 수 있습니다. 이 역할을 사용하면 서비스가 다른 서비스의 리소스에 액세스해 사용자를 대신해 작업을 완료할 수 있습니다. 서비스 역할은 IAM 계정에 나타나고, 해당 계정이 소유합니다. 즉, IAM 관리자가 이 역할에 대한 권한을 변경할 수 있습니다. 그러나 권한을 변경하면 서비스의 기능이 손상될 수 있습니다.

Amazon MQ는 서비스 역할을 지원합니다.

Amazon MQ 자격 증명 기반 정책 예제

기본적으로 사용자 및 역할은 Amazon MQ 리소스를 생성하거나 수정할 수 있는 권한이 없습니다. 또한 AWS Management Console AWS CLI또는 AWS API를 사용하여 작업을 수행할 수 없습니다. IAM 관리자는 지정된 리소스에서 특정 API 작업을 수행할 수 있는 권한을 사용자와 역할에게 부여하는 IAM 정책을 생성해야 합니다. 그런 다음 관리자는 해당 권한이 필요한 IAM 사용자 또는 그룹에 이러한 정책을 연결해야 합니다.

이러한 예제 JSON 정책 문서를 사용하여 IAM ID 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서의 JSON 탭에서 정책 생성을 참조하세요.

주제

- 정책 모범 사례
- Amazon MQ 콘솔 사용
- 사용자가 자신이 권한을 볼 수 있도록 허용

정책 모범 사례

ID 기반 정책에 따라 계정에서 사용자가 Amazon MQ 리소스를 생성, 액세스 또는 삭제할 수 있는지 여부가 결정됩니다. 이 작업으로 인해 AWS 계정에 비용이 발생할 수 있습니다. ID 기반 정책을 생성하거나 편집할 때는 다음 지침과 권장 사항을 따르세요.

- AWS 관리형 정책을 시작하고 최소 권한으로 전환 - 사용자 및 워크로드에 권한 부여를 시작하려면 많은 일반적인 사용 사례에 대한 권한을 부여하는 AWS 관리형 정책을 사용합니다. 에서 사용할 수 있습니다 AWS 계정. 사용 사례에 맞는 AWS 고객 관리형 정책을 정의하여 권한을 추가로 줄이는 것이 좋습니다. 자세한 내용은 IAM 사용 설명서의 AWS 관리형 정책 또는 AWS 직무에 대한 관리형 정책을 참조하세요.
- 최소 권한 적용 – IAM 정책을 사용하여 권한을 설정하는 경우, 작업을 수행하는 데 필요한 권한만 부여합니다. 이렇게 하려면 최소 권한으로 알려진 특정 조건에서 특정 리소스에 대해 수행할 수 있는 작업을 정의합니다. IAM을 사용하여 권한을 적용하는 방법에 대한 자세한 정보는 IAM 사용 설명서에 있는 IAM의 정책 및 권한을 참조하세요.
- IAM 정책의 조건을 사용하여 액세스 추가 제한 – 정책에 조건을 추가하여 작업 및 리소스에 대한 액세스를 제한할 수 있습니다. 예를 들어, SSL을 사용하여 모든 요청을 전송해야 한다고 지정하는 정책 조건을 작성할 수 있습니다. AWS 서비스와 같은 특정를 통해 사용되는 경우 조건을 사용하여 서비스 작업에 대한 액세스 권한을 부여할 수도 있습니다 CloudFormation. 자세한 내용은 IAM 사용 설명서의 IAM JSON 정책 요소: 조건을 참조하세요.
- IAM Access Analyzer를 통해 IAM 정책을 확인하여 안전하고 기능적인 권한 보장 - IAM Access Analyzer에서는 IAM 정책 언어(JSON)와 모범 사례가 정책에서 준수되도록 새로운 및 기존 정책을 확인합니다. IAM Access Analyzer는 100개 이상의 정책 확인 항목과 실행 가능한 추천을 제공하여 안전하고 기능적인 정책을 작성하도록 돕습니다. 자세한 내용은 IAM 사용 설명서의 IAM Access Analyzer에서 정책 검증을 참조하세요.
- 다중 인증(MFA) 필요 -에서 IAM 사용자 또는 루트 사용자가 필요한 시나리오가 있는 경우 추가 보안을 위해 MFA를 AWS 계정켭니다. API 작업을 직접적으로 호출할 때 MFA가 필요하면 정책에 MFA 조건을 추가합니다. 자세한 내용은 IAM 사용 설명서의 MFA를 통한 보안 API 액세스를 참조하세요.

IAM의 모범 사례에 대한 자세한 내용은 IAM 사용 설명서의 IAM의 보안 모범 사례를 참조하세요.

Amazon MQ 콘솔 사용

Amazon MQ 콘솔에 액세스하려면 최소한의 권한 집합이 있어야 합니다. 이러한 권한은 AWS 계정의 Amazon MQ 리소스에 대한 세부 정보를 나열하고 볼 수 있도록 허용해야 합니다. 최소 필수 권한보다 더 제한적인 보안 인증 기반 정책을 만들면 콘솔이 해당 정책에 연결된 개체(IAM 사용자 또는 역할)에 대해 의도대로 작동하지 않습니다.

이러한 엔터티가 Amazon MQ 콘솔을 계속 사용할 수 있도록 하려면 다음 AWS 관리형 정책도 엔터티에 연결합니다. 자세한 내용은 IAM 사용 설명서의 사용자에게 권한 추가를 참조하세요.

```
AmazonMQReadOnlyAccess
```

AWS CLI 또는 AWS API만 호출하는 사용자에게는 최소 콘솔 권한을 허용할 필요가 없습니다. 그 대신, 수행하려는 API 작업과 일치하는 작업에만 액세스할 수 있도록 합니다.

사용자가 자신이 권한을 볼 수 있도록 허용

이 예제는 IAM 사용자가 자신의 사용자 ID에 연결된 인라인 및 관리형 정책을 볼 수 있도록 허용하는 정책을 생성하는 방법을 보여줍니다. 이 정책에는 콘솔에서 또는 AWS CLI 또는 AWS API를 사용하여 프로그래밍 방식으로이 작업을 완료할 수 있는 권한이 포함됩니다.

```
{
    "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 MQ에 대한 API 인증 및 권한 부여

Amazon MQ는 API 인증에 표준 AWS 요청 서명을 사용합니다. 자세한 내용은 AWS 일반 참조의 AWS API 요청 서명을 참조하십시오.

> **Note**
>
> 현재 Amazon MQ는 리소스 기반 권한 또는 리소스 기반 정책을 사용한 IAM 인증을 지원하지 않습니다.

AWS 사용자에게 브로커, 구성 및 사용자 작업을 승인하려면 IAM 정책 권한을 편집해야 합니다.

주제

- Amazon MQ 브로커 생성에 필요한 IAM 권한
- Amazon MQ REST API 권한 참조
- Amazon MQ 추가 권한 참조
- Amazon MQ API 작업에 대한 리소스 수준 권한

Amazon MQ 브로커 생성에 필요한 IAM 권한

브로커를 생성하려면 `AmazonMQFullAccess` IAM 정책을 사용하거나 IAM 정책에 다음 EC2 권한을 포함해야 합니다.

다음 사용자 지정 정책은 Amazon MQ에서 ActiveMQ 브로커를 생성하는 데 필요한 리소스를 조작할 수 있는 권한을 부여하는 두 개의 문(한 개는 조건문)으로 구성됩니다.

> **Important**
>
> - `ec2:CreateNetworkInterface` 작업은 Amazon MQ가 사용자를 대신하여 사용자 계정에서 탄력적 네트워크 인터페이스(ENI)를 생성하도록 허용하는 데 필요합니다.
> - `ec2:CreateNetworkInterfacePermission` 작업은 Amazon MQ가 ENI를 ActiveMQ 브로커에 연결하도록 승인합니다.

- `ec2:AuthorizedService` 조건 키는 Amazon MQ 서비스 계정에만 ENI 권한을 부여할 수 있도록 보장합니다.

JSON

```
{
    "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"
            }
        }
    }]
}
```

자세한 내용은 2단계: 사용자 생성 및 자격 AWS 증명 가져오기 및 Amazon MQ 탄력적 네트워크 인터페이스를 수정하거나 삭제하지 않음 단원을 참조하세요.

Amazon MQ REST API 권한 참조

다음 표에는 Amazon MQ REST API 및 해당 IAM 권한이 나열되어 있습니다.

Amazon MQ REST API 및 필요한 권한

Amazon MQ REST API	필수 권한
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
ListBrokers	mq:ListBrokers
ListConfigurationRevisions	mq:ListConfigurationRevisions
ListConfigurations	mq:ListConfigurations
ListTags	mq:ListTags
ListUsers	mq:ListUsers
RebootBroker	mq:RebootBroker

Amazon MQ REST API	필수 권한
UpdateBroker	mq:UpdateBroker
UpdateConfiguration	mq:UpdateConfiguration
UpdateUser	mq:UpdateUser

Amazon MQ 추가 권한 참조

다음 표에는 OAuth 2.0 인증과 같은 특정 기능에 필요한 Amazon MQ API 및 추가 IAM 권한이 나열되어 있습니다.

Amazon MQ REST API	권한	설명
UpdateBroker	mq:UpdateBrokerAccessConfiguration	연결된 브로커 구성에서 인증 및 권한 부여 옵션을 업데이트하려면 이 권한이 필요합니다. 자세한 내용은 RabbitMQ용 Amazon MQ에 대한 OAuth 2.0 인증 및 권한 부여 단원을 참조하십시오.

Amazon MQ API 작업에 대한 리소스 수준 권한

리소스 수준 권한이란 사용자가 작업을 수행할 수 있는 리소스를 지정하는 기능을 말합니다. Amazon MQ는 리소스 수준 권한을 부분적으로 지원합니다. 특정 Amazon MQ 작업의 경우, 이행해야 하는 조건 또는 사용자가 사용할 수 있는 특정 리소스를 기반으로 사용자가 해당 작업을 언제 사용할 수 있는지를 제어할 수 있습니다.

다음 표에서는 현재 리소스 수준 권한을 지원하는 Amazon MQ API 작업과 각 작업에 지원되는 리소스, 리소스 ARN 및 조건 키를 설명합니다.

> **Important**
>
> 이 표에 표시되지 않은 Amazon MQ API 작업은 리소스 수준 권한을 지원하지 않습니다. Amazon MQ API 작업이 리소스 수준 권한을 지원하지 않는 경우, 사용자에게 이 작업을 사용할 권한을 부여할 수 있지만 정책 설명의 리소스 요소에 * 와일드카드를 지정해야 합니다.

API 작업	리소스 유형(*필수)
CreateConfiguration	구성*
CreateTags	브로커, 구성
CreateUser	브로커*
DeleteBroker	브로커*
DeleteUser	브로커*
DescribeBroker	브로커*
DescribeConfiguration	구성*
DescribeConfigurationRevision	구성*
DescribeUser	브로커*
ListConfigurationRevisions	구성*
ListConfigurationRevisions	구성*
ListTags	브로커, 구성
ListUsers	브로커*
RebootBroker	브로커*

API 작업	리소스 유형(*필수)
UpdateBroker	브로커*
UpdateConfiguration	구성*
UpdateUser	브로커*

브로커 인증 및 권한 부여

Amazon MQ는 브로커 엔진 유형에 따라 다양한 인증 및 권한 부여 방법을 제공합니다.

ActiveMQ용 Amazon MQ에 대한 인증 및 권한 부여

ActiveMQ용 Amazon MQ는 다음과 같은 인증 및 권한 부여 방법을 지원합니다.

간편한 인증 및 권한 부여

이 메서드에서 브로커 사용자는 Amazon MQ 콘솔 또는 API를 통해 생성되고 관리됩니다. 대기열, 주제 및 ActiveMQ 웹 콘솔에 액세스할 수 있는 특정 권한으로 사용자를 구성할 수 있습니다. 이 방법에 대한 자세한 내용은 ActiveMQ 브로커 사용자 생성을 참조하세요.

LDAP 인증 및 권한 부여

이 방법에서 브로커 사용자는 LDAP 서버에 저장된 자격 증명을 통해 인증합니다. LDAP 서버를 통해 사용자를 추가, 삭제 및 수정하고 주제 및 대기열에 권한을 할당하여 중앙 집중식 인증 및 권한 부여를 제공할 수 있습니다. 이 방법에 대한 자세한 내용은 ActiveMQ 브로커와 LDAP 통합을 참조하세요.

RabbitMQ용 Amazon MQ에 대한 인증 및 권한 부여

RabbitMQ용 Amazon MQ는 다음과 같은 인증 및 권한 부여 방법을 지원합니다.

간편한 인증 및 권한 부여

이 방법에서 브로커 사용자는 RabbitMQ 브로커에 내부적으로 저장되고 웹 콘솔 또는 관리 API를 통해 관리됩니다. vhost, Exchange, 대기열 및 주제에 대한 권한은 RabbitMQ에서 직접 구성됩니다. 기본 메서드입니다. 자세한 내용은 단순 인증 및 권한 부여를 참조하세요.

OAuth 2.0 인증 및 권한 부여

이 메서드에서 브로커 사용자와 해당 권한은 외부 OAuth 2.0 ID 제공업체(IdP)에서 관리합니다. vhost, Exchange, 대기열 및 주제에 대한 사용자 인증 및 리소스 권한은 OAuth 2.0 공급자의 범위 시스템을

통해 중앙 집중화됩니다. 이를 통해 사용자 관리를 간소화하고 기존 자격 증명 시스템과 통합할 수 있습니다. 자세한 내용은 OAuth 2.0 인증 및 권한 부여를 참조하세요.

IAM 인증 및 권한 부여

이 방법에서 브로커 사용자는 AWS IAM 아웃바운드 페더레이션을 통해 IAM 자격 증명을 사용하여 인증합니다. IAM 자격 증명은 AWS Security Token Service(STS)에서 JWT 토큰을 얻는 데 사용되며, 이러한 JWT 토큰은 인증을 위한 OAuth 2.0 토큰 역할을 합니다. 이 방법은 RabbitMQ용 Amazon MQ에서 기존 OAuth 2.0 지원을 활용합니다. 여기서 AWS 는 OAuth 2.0 자격 증명 공급자 역할을 합니다. 사용자 인증은 AWS IAM에서 처리되는 반면, vhost, Exchange, 대기열 및 주제에 대한 리소스 권한은 RabbitMQ에 구성된 IAM 정책 및 범위 별칭을 통해 관리됩니다. 자세한 내용은 IAM 인증 및 권한 부여를 참조하세요.

LDAP 인증 및 권한 부여

이 방법에서는 브로커 사용자와 해당 권한이 외부 LDAP 디렉터리 서비스에 의해 관리됩니다. 사용자 인증 및 리소스 권한은 LDAP 서버를 통해 중앙 집중화되므로 사용자는 기존 디렉터리 서비스 자격 증명을 사용하여 RabbitMQ에 액세스할 수 있습니다. 자세한 내용은 LDAP 인증 및 권한 부여를 참조하세요.

HTTP 인증 및 권한 부여

이 방법에서는 브로커 사용자와 해당 권한이 외부 HTTP 서버에서 관리됩니다. 사용자 인증 및 리소스 권한은 HTTP 서버를 통해 중앙 집중화되므로 사용자는 자체 인증 및 권한 부여 공급자를 사용하여 RabbitMQ에 액세스할 수 있습니다. 이 방법에 대한 자세한 내용은 HTTP 인증 및 권한 부여를 참조하세요.

SSL 인증서 인증

Amazon MQ는 RabbitMQ 브로커에 대해 상호 TLS(mTLS)를 지원합니다. SSL 인증 플러그인은 mTLS 연결의 클라이언트 인증서를 사용하여 사용자를 인증합니다. 이 방법에서 브로커 사용자는 사용자 이름 및 암호 자격 증명 대신 X.509 클라이언트 인증서를 사용하여 인증됩니다. 클라이언트의 인증서는 신뢰할 수 있는 인증 기관(CA)에 대해 검증되고 사용자 이름은 일반 이름(CN) 또는 주체 대체 이름(SAN)과 같은 인증서의 필드에서 추출됩니다. 이 방법은 네트워크를 통해 자격 증명을 전송하지 않고 강력한 인증을 제공합니다. 자세한 내용은 SSL 인증서 인증을 참조하세요.

> **Note**
>
> RabbitMQ는 동시에 사용할 수 있는 여러 인증 및 권한 부여 방법을 지원합니다. 예를 들어 OAuth 2.0과 단순(내부) 인증을 모두 활성화할 수 있습니다. 자세한 내용은 OAuth 2.0 및 단순

(내부) 인증 활성화에 대한 OAuth 2.0 자습서 섹션과 RabbitMQ 액세스 제어 설명서를 참조하세요.

Amazon MQ는 인증 구성을 테스트할 때 내부 사용자를 생성할 것을 권장합니다. 이렇게 하면 RabbitMQ 관리 API를 사용하여 액세스 구성을 검증할 수 있습니다. 자세한 내용은 액세스 검증을 참조하세요.

AWS Amazon MQ에 대한 관리형 정책

AWS 관리형 정책은에서 생성하고 관리하는 독립 실행형 정책입니다 AWS. AWS 관리형 정책은 사용자, 그룹 및 역할에 권한 할당을 시작할 수 있도록 많은 일반적인 사용 사례에 대한 권한을 제공하도록 설계되었습니다.

AWS 관리형 정책은 모든 AWS 고객이 사용할 수 있으므로 특정 사용 사례에 대해 최소 권한을 부여하지 않을 수 있습니다. 사용 사례에 고유한 고객 관리형 정책을 정의하여 권한을 줄이는 것이 좋습니다.

AWS 관리형 정책에 정의된 권한은 변경할 수 없습니다. 가 관리형 정책에 정의된 권한을 AWS 업데이트하는 AWS 경우 업데이트는 정책이 연결된 모든 보안 주체 자격 증명(사용자, 그룹 및 역할)에 영향을 줍니다. AWS AWS 서비스 는 새가 시작되거나 기존 서비스에 새 API 작업을 사용할 수 있게 될 때 AWS 관리형 정책을 업데이트할 가능성이 높습니다.

자세한 내용은 IAM 사용자 가이드의 AWS 관리형 정책을 참조하세요.

Amazon MQ는 다음과 같은 AWS 관리형 정책을 지원합니다.

- AmazonMQApiFullAccess
- AmazonMQApiReadOnlyAccess
- AmazonMQFullAccess
- AmazonMQReadOnlyAccess
- AmazonMQServiceRolePolicy

AWS 관리형 정책: AmazonMQServiceRolePolicy

`AmazonMQServiceRolePolicy`를 IAM 엔티티에 연결할 수 없습니다. 이 정책은 서비스 연결 역할에 연결하면 Amazon MQ가 사용자를 대신하여 작업을 수행할 수 있습니다. 이 권한 정책 및 Amazon MQ

가 수행할 수 있는 작업에 대한 자세한 내용은 the section called “Amazon MQ에 대한 서비스 연결 역할 권한” 단원을 참조하세요.

AWS 관리형 정책에 대한 Amazon MQ 업데이트

이 서비스가 이러한 변경 사항을 추적하기 시작한 이후부터 Amazon MQ의 AWS 관리형 정책 업데이트에 대한 세부 정보를 봅니다. 이 페이지의 변경 사항에 대한 자동 알림을 받아보려면 Amazon MQ 문서 기록 페이지에서 RSS 피드를 구독하세요.

변경	설명	Date
Amazon MQ 변경 사항 추적 시작	Amazon MQ는 AWS 관리형 정책에 대한 변경 사항 추적을 시작했습니다.	2021년 5월 5일

Amazon MQ에 대해 서비스 연결 역할 사용

Amazon MQ는 AWS Identity and Access Management (IAM) 서비스 연결 역할을 사용합니다. 서비스 연결 역할은 Amazon MQ에 직접 연결된 고유한 유형의 IAM 역할입니다. 서비스 연결 역할은 Amazon MQ에서 사전 정의하며 서비스가 사용자를 대신하여 다른 AWS 서비스를 호출하는 데 필요한 모든 권한을 포함합니다.

필요한 권한을 수동으로 추가할 필요가 없으므로 서비스 연결 역할은 Amazon MQ를 더 쉽게 설정할 수 있습니다. Amazon MQ에서 서비스 연결 역할의 권한을 정의하므로 다르게 정의되지 않은 한, Amazon MQ만 해당 역할을 수임할 수 있습니다. 정의된 권한에는 신뢰 정책과 권한 정책이 포함되며 이 권한 정책은 다른 IAM 엔터티에 연결할 수 없습니다.

먼저 관련 리소스를 삭제한 후에만 서비스 연결 역할을 삭제할 수 있습니다. 이렇게 하면 리소스에 대한 액세스 권한을 실수로 삭제할 수 없기 때문에 Amazon MQ 리소스가 보호됩니다.

서비스 연결 역할을 지원하는 기타 서비스에 대한 자세한 내용은 IAM으로 작업하는AWS 서비스를 참조하고 서비스 연결 역할 열에 예가 표시된 서비스를 찾으십시오. 해당 서비스에 대한 서비스 연결 역할 설명서를 보려면 예(Yes) 링크를 선택합니다.

Amazon MQ에 대한 서비스 연결 역할 권한

Amazon MQ는 AWSServiceRoleForAmazonMQ라는 서비스 연결 역할을 사용합니다. Amazon MQ는 이 서비스 연결 역할을 사용하여 사용자를 대신하여 AWS 서비스를 호출합니다.

AWSServiceRoleForAmazonMQ 서비스 연결 역할은 역할을 수임하기 위해 다음 서비스를 신뢰합니다.

- `mq.amazonaws.com`

Amazon MQ는 AWSServiceRoleForAmazonMQ 서비스 연결 역할에 연결된 권한 정책 `AmazonMQServiceRolePolicy`를 사용하여 지정된 리소스에서 다음 작업을 완료합니다.

- 작업: `vpc` 리소스에 대한 `ec2:CreateVpcEndpoint`.
- 작업: `subnet` 리소스에 대한 `ec2:CreateVpcEndpoint`.
- 작업: `security-group` 리소스에 대한 `ec2:CreateVpcEndpoint`.
- 작업: `vpc-endpoint` 리소스에 대한 `ec2:CreateVpcEndpoint`.
- 작업: `vpc` 리소스에 대한 `ec2:DescribeVpcEndpoints`.
- 작업: `subnet` 리소스에 대한 `ec2:DescribeVpcEndpoints`.
- 작업: `vpc-endpoint` 리소스에 대한 `ec2:CreateTags`.
- 작업: `log-group` 리소스에 대한 `logs:PutLogEvents`.
- 작업: `log-group` 리소스에 대한 `logs:DescribeLogStreams`.
- 작업: `log-group` 리소스에 대한 `logs:DescribeLogGroups`.
- 작업: `log-group` 리소스에 대한 `CreateLogStream`.
- 작업: `log-group` 리소스에 대한 `CreateLogGroup`.

RabbitMQ용 Amazon MQ 브로커를 생성할 때 `AmazonMQServiceRolePolicy` 권한 정책을 사용하여 Amazon MQ가 사용자 대신 다음 작업을 수행할 수 있습니다.

- 지정한 Amazon VPC, 서브넷 및 보안 그룹을 사용하여 브로커용 Amazon VPC 엔드포인트를 생성합니다. 브로커용으로 생성된 엔드포인트를 사용하여 RabbitMQ 관리 콘솔이나 관리 API를 통해 또는 프로그래밍 방식으로 브로커에 연결할 수 있습니다.
- 로그 그룹을 생성하고 Amazon CloudWatch Logs에 브로커 로그를 게시합니다.

JSON

```
{
    "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/*"
            ]
        }
    ]
}
```

IAM 엔터티(사용자, 그룹, 역할 등)가 서비스 연결 역할을 생성하고 편집하거나 삭제할 수 있도록 권한을 구성할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 권한 섹션을 참조하세요.

Amazon MQ에 대한 서비스 연결 역할 생성

서비스 연결 역할은 수동으로 생성할 필요가 없습니다. 브로커를 처음 생성할 때 Amazon MQ는 사용자를 대신하여 서비스를 호출하는 AWS 서비스 연결 역할을 생성합니다. 이후에 생성하는 모든 브로커는 동일한 역할을 사용하며 새 역할이 생성되지 않습니다.

> **Important**
>
> 이러한 서비스 연결 역할은 해당 역할이 지원하는 기능을 사용하는 다른 서비스에서 작업을 완료했을 경우 계정에 나타날 수 있습니다. 자세한 내용은 내 IAM 계정에 표시되는 새 역할을 참조하세요.

이 서비스 연결 역할을 삭제했다가 다시 생성해야 하는 경우 동일한 프로세스를 사용하여 계정에서 역할을 다시 생성할 수 있습니다.

또한 IAM 콘솔을 사용해 Amazon MQ 사용 사례로 서비스 연결 역할을 생성할 수도 있습니다. AWS CLI 또는 AWS API에서 서비스 이름을 사용하여 `mq.amazonaws.com` 서비스 연결 역할을 생성합니다. 자세한 내용은 IAM 사용자 설명서의 서비스 연결 역할 생성 섹션을 참조하세요. 이 서비스 연결 역할을 삭제하면 동일한 프로세스를 사용하여 역할을 다시 생성할 수 있습니다.

> **Important**
>
> 서비스 연결 역할은 RabbitMQ용 Amazon MQ에 대해서만 생성됩니다.

Amazon MQ에 대한 서비스 연결 역할 편집

Amazon MQ는 AWSServiceRoleForAmazonMQ 서비스 연결 역할을 편집하도록 허용하지 않습니다. 하지만 IAM을 사용하여 역할의 설명을 편집할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 편집을 참조하세요.

Amazon MQ에 대한 서비스 연결 역할 삭제

서비스 연결 역할이 필요한 기능 또는 서비스가 더 이상 필요 없는 경우에는 해당 역할을 삭제하는 것이 좋습니다. 따라서 적극적으로 모니터링하거나 유지하지 않는 미사용 엔터티가 없도록 합니다. 단, 서비스 연결 역할에 대한 리소스를 먼저 정리해야 수동으로 삭제할 수 있습니다.

> **Note**
>
> 리소스를 삭제하려고 할 때 Amazon MQ 서비스가 역할을 사용 중이면 삭제에 실패할 수 있습니다. 이 문제가 발생하면 몇 분 기다렸다가 작업을 다시 시도하세요.

AWSServiceRoleForAmazonMQ에서 사용하는 Amazon MQ 리소스를 삭제하려면

- AWS Management Console, Amazon MQ CLI 또는 Amazon MQ API를 사용하여 Amazon MQ 브로커를 삭제합니다. 브로커를 삭제하는 방법에 대한 자세한 내용은 ??? 단원을 참조하세요.

IAM을 사용하여 수동으로 서비스 연결 역할을 삭제하려면 다음을 수행하세요.

IAM 콘솔 AWS CLI, 또는 AWS API를 사용하여 AWSServiceRoleForAmazonMQ 서비스 연결 역할을 삭제합니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 삭제를 참조하세요.

Amazon MQ 서비스 연결 역할을 지원하는 리전

Amazon MQ는 서비스가 제공되는 모든 리전에서 서비스 연결 역할 사용을 지원합니다. 자세한 설명은 AWS 리전 및 엔드포인트를 참조하세요.

리전 이름	리전 자격 증명	Amazon MQ에서 지원
미국 동부(버지니아 북부)	us-east-1	예
미국 동부(오하이오)	us-east-2	예
미국 서부(캘리포니아 북부)	us-west-1	예
미국 서부(오리건)	us-west-2	예
아시아 태평양(뭄바이)	ap-south-1	예
아시아 태평양(오사카)	ap-northeast-3	예
아시아 태평양(서울)	ap-northeast-2	예
아시아 태평양(싱가포르)	ap-southeast-1	예
아시아 태평양(시드니)	ap-southeast-2	예

리전 이름	리전 자격 증명	Amazon MQ에서 지원
아시아 태평양(도쿄)	ap-northeast-1	예
캐나다(중부)	ca-central-1	예
유럽(프랑크푸르트)	eu-central-1	예
유럽(아일랜드)	eu-west-1	예
유럽(런던)	eu-west-2	예
유럽(파리)	eu-west-3	예
남아메리카(상파울루)	sa-east-1	예
AWS GovCloud (US)	us-gov-west-1	아니요

Amazon MQ 자격 증명 및 액세스 문제 해결

다음 정보를 사용하여 Amazon MQ 및 IAM으로 작업할 때 발생할 수 있는 일반적인 문제를 진단하고 수정할 수 있습니다.

주제

- Amazon MQ에서 작업을 수행할 권한이 없음
- iam:PassRole을 수행하도록 인증되지 않음
- 내 AWS 계정 외부의 사람이 내 Amazon MQ 리소스에 액세스하도록 허용하고 싶습니다.

Amazon MQ에서 작업을 수행할 권한이 없음

에서 작업을 수행할 권한이 없다는 AWS Management Console 메시지가 표시되면 관리자에게 문의하여 도움을 받아야 합니다. 관리자는 로그인 보안 인증 정보를 제공한 사람입니다.

다음 예제 오류는 `mateojackson` 사용자가 콘솔을 사용하여 *##*에 대한 세부 정보를 보려고 하지만 `mq:GetWidget` 권한이 없는 경우에 발생합니다.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 mq:GetWidget on resource: my-example-widget
```

이 경우, Mateo는 *my-example-widget* 작업을 사용하여 mq:*GetWidget* 리소스에 액세스하도록 허용하는 정책을 업데이트하라고 관리자에게 요청합니다.

iam:PassRole을 수행하도록 인증되지 않음

iam:PassRole 작업을 수행할 수 있는 권한이 없다는 오류가 수신되면 Amazon MQ에 역할을 전달할 수 있도록 정책을 업데이트해야 합니다.

일부 AWS 서비스 에서는 새 서비스 역할 또는 서비스 연결 역할을 생성하는 대신 기존 역할을 해당 서비스에 전달할 수 있습니다. 이렇게 하려면 역할을 서비스에 전달할 권한이 있어야 합니다.

다음 예제 오류는 marymajor라는 IAM 사용자가 콘솔을 사용하여 Amazon MQ에서 작업을 수행하려고 하는 경우에 발생합니다. 하지만 작업을 수행하려면 서비스 역할이 부여한 권한이 서비스에 있어야 합니다. Mary는 서비스에 역할을 전달할 권한이 없습니다.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

이 경우, Mary가 iam:PassRole작업을 수행할 수 있도록 Mary의 정책을 업데이트해야 합니다.

도움이 필요한 경우 AWS 관리자에게 문의하세요. 관리자는 로그인 자격 증명을 제공한 사람입니다.

내 AWS 계정 외부의 사람이 내 Amazon MQ 리소스에 액세스하도록 허용하고 싶습니다.

다른 계정의 사용자 또는 조직 외부의 사람이 리소스에 액세스할 때 사용할 수 있는 역할을 생성할 수 있습니다. 역할을 수임할 신뢰할 수 있는 사람을 지정할 수 있습니다. 리소스 기반 정책 또는 액세스 제어 목록(ACL)을 지원하는 서비스의 경우, 이러한 정책을 사용하여 다른 사람에게 리소스에 대한 액세스 권한을 부여할 수 있습니다.

자세한 내용은 다음을 참조하세요.

- Amazon MQ에서 이러한 기능을 지원하는지 여부를 알아보려면 Amazon MQ에서 IAM을 사용하는 방법 단원을 참조하세요.
- 소유 AWS 계정 한의 리소스에 대한 액세스 권한을 제공하는 방법을 알아보려면 IAM 사용 설명서의 소유한 다른의 IAM 사용자에게 액세스 권한 제공을 참조 AWS 계정 하세요.
- 타사에 리소스에 대한 액세스 권한을 제공하는 방법을 알아보려면 IAM 사용 설명서의 타사가 AWS 계정 소유한에 대한 액세스 권한 제공을 AWS 계정참조하세요.

- ID 페더레이션을 통해 액세스 권한을 제공하는 방법을 알아보려면 IAM 사용 설명서의 외부에서 인증된 사용자에게 액세스 권한 제공(ID 페더레이션)을 참조하세요.
- 크로스 계정 액세스에 대한 역할과 리소스 기반 정책 사용의 차이점을 알아보려면 IAM 사용 설명서의 IAM의 크로스 계정 리소스 액세스를 참조하세요.

Amazon MQ에 대한 규정 준수 확인

타사 감사자는 여러 규정 준수 프로그램의 일환으로 Amazon MQ의 보안 및 AWS 규정 준수를 평가합니다. 여기에는 SOC, PCI, HIPAA 등이 포함됩니다.

AWS 서비스 가 특정 규정 준수 프로그램의 범위 내에 있는지 알아보려면 AWS 서비스 규정 준수 프로그램 제공 범위 내를 참조하고 관심 있는 규정 준수 프로그램을 선택합니다. 일반 정보는 AWS 규정 준수 프로그램.

를 사용하여 타사 감사 보고서를 다운로드할 수 있습니다 AWS Artifact. 자세한 내용은 Downloading Reports inDownloading AWS Artifact을 참조하세요.

사용 시 규정 준수 책임은 데이터의 민감도, 회사의 규정 준수 목표, 관련 법률 및 규정에 따라 AWS 서비스 결정됩니다. 사용 시 규정 준수 책임에 대한 자세한 내용은 AWS 보안 설명서를 AWS 서비스참조하세요.

Amazon MQ의 복원성

AWS 글로벌 인프라는 AWS 리전 및 가용 영역을 중심으로 구축됩니다. AWS 리전은 물리적으로 분리되고 격리된 다수의 가용 리전을 제공하며 이러한 가용 리전은 짧은 지연 시간, 높은 처리량 및 높은 중복성을 갖춘 네트워크에 연결되어 있습니다. 가용 영역을 사용하면 중단 없이 영역 간에 자동으로 장애 극복 조치가 이루어지는 애플리케이션 및 데이터베이스를 설계하고 운영할 수 있습니다. 가용 영역은 기존의 단일 또는 다중 데이터 센터 인프라보다 가용성, 내결함성, 확장성이 뛰어납니다.

AWS 리전 및 가용 영역에 대한 자세한 정보는 AWS 글로벌 인프라를 참조하세요.

Amazon MQ의 인프라 보안

관리형 서비스인 Amazon MQ는 AWS 글로벌 네트워크 보안으로 보호됩니다. AWS 보안 서비스 및가 인프라를 AWS 보호하는 방법에 대한 자세한 내용은 AWS 클라우드 보안을 참조하세요. 인프라 보안 모범 사례를 사용하여 환경을 설계하려면 보안 원칙 AWS Well-Architected Framework의 인프라 보호를 참조하세요 AWS .

AWS 에서 게시한 API 호출을 사용하여 네트워크를 통해 Amazon MQ에 액세스합니다. 클라이언트는 다음을 지원해야 합니다.

- Transport Layer Security(TLS). TLS 1.2는 필수이며 TLS 1.3을 권장합니다.
- DHE(Ephemeral Diffie-Hellman) 또는 ECDHE(Elliptic Curve Ephemeral Diffie-Hellman)와 같은 완전 전송 보안(PFS)이 포함된 암호 제품군. Java 7 이상의 최신 시스템은 대부분 이러한 모드를 지원합니다.

Amazon MQ에 대한 보안 모범 사례

다음 설계 패턴은 Amazon MQ 브로커의 보안을 개선할 수 있습니다.

주제

Amazon MQ가 데이터를 암호화하는 방법과 지원되는 프로토콜 목록에 대한 자세한 내용은 데이터 보호를 참조하세요.

퍼블릭 액세스 가능성이 없는 브로커 선호

퍼블릭 액세스 기능 없이 생성된 브로커는 VPC 외부에서 액세스할 수 없습니다. 따라서 브로커가 퍼블릭 인터넷에서 분산 서비스 거부(DDoS) 공격을 받을 가능성이 대폭 감소합니다. 자세한 내용은 AWS 보안 블로그의 공격 표면을 줄여 DDoS 공격에 대비하는 방법을 참조하세요.

항상 권한 부여 맵 구성

ActiveMQ에서는 기본적으로 권한 부여 맵이 구성되지 않기 때문에 모든 인증된 사용자가 브로커에서 모든 작업을 수행할 수 있습니다. 따라서 그룹별로 권한을 제한하는 것이 모범 사례입니다. 자세한 정보는 `authorizationEntry`을 참조하세요.

> **Important**
>
> `activemq-webconsole` 그룹을 포함하지 않는 권한 부여 맵을 지정하는 경우, 그룹이 Amazon MQ 브로커에 메시지를 보내거나 브로커에서 메시지를 수신할 권한이 없기 때문에 ActiveMQ 웹 콘솔을 사용할 수 없습니다.

VPC 보안 그룹으로 불필요한 프로토콜 차단

프라이빗 브로커의 보안을 강화하려면 Amazon VPC 보안 그룹을 적절하게 구성하여 불필요한 프로토콜 및 포트의 연결을 제한해야 합니다. 예를 들어, 대부분의 프로토콜에 대한 액세스는 제한하면서 OpenWire 및 웹 콘솔에 대한 액세스는 허용하기 위해 61617 및 8162에 대한 액세스만 허용할 수 있습니다. 이렇게 하면 OpenWire 및 웹 콘솔이 제대로 작동하도록 허용하면서 사용하지 않는 프로토콜은 차단하여 노출이 제한됩니다.

사용 중인 프로토콜 포트만 허용합니다.

- AMQP: 5671
- MQTT: 8883
- OpenWire: 61617
- STOMP: 61614
- WebSocket: 61619

자세한 내용은 다음을 참조하세요.

- VPC의 보안 그룹
- VPC의 기본 보안 그룹
- 보안 그룹 작업

RabbitMQ용 Amazon MQ 브로커 로깅 및 모니터링

모니터링은 AWS 솔루션의 안정성, 가용성 및 성능을 유지하는 데 중요한 부분입니다. 다중 지점 장애가 발생할 경우 보다 쉽게 디버깅할 수 있도록 AWS 솔루션의 모든 부분으로부터 모니터링 데이터를 수집해야 합니다.는 Amazon MQ 리소스를 모니터링하고 잠재적 인시던트에 대응하기 위한 몇 가지 도구를 AWS 제공합니다.

CloudWatch를 사용하여 Amazon MQ 브로커의 지표를 확인하고 분석할 수 있습니다. CloudWatch 콘솔 AWS CLI, 또는 CloudWatch AWS CLI에서 브로커 지표를 보고 분석할 수 있습니다. Amazon MQ의 CloudWatch 지표는 브로커에서 자동으로 폴링되며, 이후 1분마다 CloudWatch에 푸시됩니다. ActiveMQ 브로커의 경우 CloudWatch는 처음 1,000개 대상만 모니터링합니다. RabbitMQ 브로커의 경우 CloudWatch는 소비자 수에 따라 정렬된 대상 중 처음 500개만 모니터링합니다.

전체 Amazon MQ 지표 목록은 ActiveMQ용 Amazon MQ 브로커에 사용 가능한 CloudWatch 지표 단원을 참조하세요.

지표의 CloudWatch 경보를 생성하는 방법에 대한 자세한 내용은 Amazon CloudWatch 사용 설명서의 CloudWatch 경보 생성 또는 편집을 참조하세요.

Amazon MQ의 CloudWatch 지표 액세스

AWS Management Console AWS CLI및 API를 사용하여 CloudWatch 지표에 액세스할 수 있습니다.

를 사용하지 않고 CloudWatch 지표에 액세스할 수 있습니다 AWS Management Console.

를 사용하여 Amazon MQ 지표에 액세스하려면 `get-metric-statistics` 명령을 AWS CLI사용합니다. 자세한 내용은 Amazon CloudWatch 사용 설명서의 지표에 대한 통계 얻기를 참조하세요.

CloudWatch API를 사용하여 Amazon MQ 지표에 액세스하려면 `GetMetricStatistics` 작업을 사용합니다. 자세한 내용은 Amazon CloudWatch 사용 설명서의 지표에 대한 통계 얻기를 참조하세요.

를 사용하여 CloudWatch 지표에 액세스 AWS Management Console

다음 예제에서는 AWS Management Console을 사용하여 Amazon MQ에 대한 CloudWatch 지표에 액세스하는 방법을 보여줍니다. Amazon MQ 콘솔에 이미 로그인한 경우 브로커 세부 정보 페이지에서 작업, CloudWatch 지표 보기를 선택합니다.

1. CloudWatch 콘솔에 로그인합니다.

2. 탐색 창에서 [Metrics]를 선택합니다.
3. AmazonMQ 지표 네임스페이스를 선택합니다.
4. 다음 지표 차원 중 하나를 선택합니다.

   - 브로커 지표
   - Queue Metrics by Broker
   - Topic Metrics by Broker

   이 예에서는 Broker Metrics(브로커 지표)를 선택합니다.
5. 이제 Amazon MQ 지표를 검토할 수 있습니다.

   - 지표를 정렬하려면 열 머리글을 사용합니다.

   - 지표를 그래프로 표시하려면 지표 옆에 있는 확인란을 선택합니다.
   - 지표로 필터링하려면 지표 이름을 선택한 후 검색에 추가를 선택합니다.

Prometheus 지표 액세스

> **Note**
>
> Prometheus 지표는 RabbitMQ 4.2 이상에서만 사용할 수 있습니다. ActiveMQ 브로커는 Prometheus 지표를 지원하지 않습니다.

이제 Amazon MQ는 RabbitMQ용 Amazon MQ 브로커에 대한 Prometheus 지표를 지원합니다. Prometheus 지표를 사용하면 브로커 관찰성을 기존 모니터링 인프라에 통합하여 다른 서비스와 함께 브로커 성능을 통합적으로 볼 수 있습니다. Prometheus 지표를 사용하면 세분화된 알림 및 대시보드를 설정하여 메시징 워크로드의 문제를 사전에 감지하고 대응할 수 있습니다.

RabbitMQ 4.2부터 RabbitMQ용 Amazon MQ는 Prometheus 지표를 지원하므로 Prometheus 모니터링 시스템을 사용하여 브로커 지표를 스크레이프할 수 있습니다. RabbitMQ 지원되는 엔드포인트는 다음과 같습니다.

- `/metrics`
- `/metrics/detailed`

- `/metrics/memory-breakdown`

`/metrics/per-object` 엔드포인트는 지원되지 않습니다.

각 엔드포인트에서 노출되는 지표에 대한 자세한 내용은 RabbitMQ 설명서의 Prometheus 지표를 참조하세요.

Prometheus 지표와 CloudWatch 지표 비교

RabbitMQ용 Amazon MQ는 Prometheus 엔드포인트와 CloudWatch를 통해 지표를 노출합니다. 둘 다 브로커 상태에 대한 가시성을 제공하지만 범위와 사용량은 다릅니다.

Prometheus 엔드포인트는 연결 이탈, 채널 활동, 대기열 및 교환 통계, Raft 합의 지표와 같은 광범위한 브로커 내부를 포함하는 RabbitMQ 브로커 상태에 대한 더 풍부한 집계 지표 세트를 제공합니다. 이는 기존 Prometheus 기반 모니터링 인프라 및 세분화된 알림과 통합하는 데 적합합니다.

CloudWatch 지표는 Prometheus 엔드포인트에서 얻은 브로커 지표의 선별된 하위 집합입니다. 사용 가능한 CloudWatch 지표의 전체 목록은 섹션을 참조하세요RabbitMQ용 Amazon MQ 브로커에 사용 가능한 CloudWatch 지표.

CloudWatch에서 지표는 항상 시각화 전 최소 60초 간격으로 집계됩니다. 반대로 Prometheus는 원시 지표 데이터 포인트를 노출하고 Grafana와 같은 대시보드 솔루션은 기본적으로 집계 없이 개별 데이터 포인트를 시각화합니다. 따라서 동일한 지표의 시각화는 CloudWatch에 사용되는 통계에 따라 CloudWatch와 Prometheus 간에 다를 수 있습니다.

> **Note**
>
> RabbitMQ용 Amazon MQ 운영 지표의 집계되지 않은 모니터링에는 Prometheus를 사용하는 것이 좋습니다.

Prometheus 엔드포인트 가져오기 및 액세스

AWS Management Console 또는를 사용하여 RabbitMQ용 Amazon MQ 브로커의 Prometheus 엔드포인트를 얻을 수 있습니다 AWS CLI.

- AWS Management Console - Amazon MQ 콘솔로 이동하여 브로커의 세부 정보 페이지를 열고 연결 섹션에서 Prometheus 엔드포인트를 찾습니다.
- AWS CLI - `describe-broker` 명령을 사용합니다.

```
aws mq describe-broker --broker-id <broker-id>
```

Prometheus 엔드포인트는의 응답으로 반환됩니다`BrokerInstances.Endpoints`.

RabbitMQ Prometheus용 Amazon MQ 지원은 브로커와 동일한 인증 체계를 사용합니다. RabbitMQ 지원되는 인증 방법에 대한 자세한 내용은 섹션을 참조하세요RabbitMQ 인증 및 권한 부여를 위한 Amazon MQ RabbitMQ. Prometheus에서 인증을 구성하는 방법을 알아보려면 Prometheus 설명서의 http_config를 참조하세요.

Prometheus 구성 모범 사례

- 60초 이상의 스크레이핑 기간을 구성합니다. 이는 운영 안전을 위해 권장됩니다.

샘플 스크레이핑 구성

다음 섹션에서는 RabbitMQ용 Amazon MQ에 대한 샘플 Prometheus 스크레이핑 구성을 제공합니다. `<broker-prometheus-endpoint>`를 브로커의 Prometheus 엔드포인트 호스트 이름으로 바꾸고 `<username>`를 브로커 자격 증명`<password>`으로 바꿉니다.

권장 구성

대부분의 사용 사례에는 다음 구성을 사용하는 것이 좋습니다. `/metrics` 엔드포인트를 스크래핑하면 전체 클러스터 상태에 대한 잘 집계된 지표가 제공되므로 세부 지표 수집의 오버헤드 없이 브로커 성능을 명확하게 파악할 수 있습니다.

```
global:
  scrape_interval: 60s

scrape_configs:
  - job_name: 'rabbitmq-aws-cluster'
    scheme: https
    basic_auth:
      username: <username>
      password: <password>
    metrics_path: '/metrics'
    static_configs:
      - targets:
        - '<broker-prometheus-endpoint>:16001'
        - '<broker-prometheus-endpoint>:16002'
```

```
        - '<broker-prometheus-endpoint>:16003'
```

세부 지표 구성

다음 구성은 특정 브로커 구성 요소에 대한 심층적인 관찰성을 위해 추가 세부 지표 패밀리를 스크레이프합니다.

```
global:
  scrape_interval: 60s

scrape_configs:
  - job_name: 'rabbitmq-connection-churn'
    scheme: https
    basic_auth:
      username: <username>
      password: <password>
    metrics_path: '/metrics/detailed'
    params:
      family: ['connection_churn_metrics']
    static_configs:
      - targets:
        - '<broker-prometheus-endpoint>:16001'
        - '<broker-prometheus-endpoint>:16002'
        - '<broker-prometheus-endpoint>:16003'
  - job_name: 'rabbitmq-ra'
    scheme: https
    basic_auth:
      username: <username>
      password: <password>
    metrics_path: '/metrics/detailed'
    params:
      family: ['ra_metrics']
    static_configs:
      - targets:
        - '<broker-prometheus-endpoint>:16001'
        - '<broker-prometheus-endpoint>:16002'
        - '<broker-prometheus-endpoint>:16003'
  - job_name: 'rabbitmq-queue'
    scheme: https
    basic_auth:
      username: <username>
      password: <password>
    metrics_path: '/metrics/detailed'
```

```
    params:
      family: ['queue_metrics']
    static_configs:
      - targets:
        - '<broker-prometheus-endpoint>:16001'
        - '<broker-prometheus-endpoint>:16002'
        - '<broker-prometheus-endpoint>:16003'
  - job_name: 'rabbitmq-exchange'
    scheme: https
    basic_auth:
      username: <username>
      password: <password>
    metrics_path: '/metrics/detailed'
    params:
      family: ['exchange_metrics']
    static_configs:
      - targets:
        - '<broker-prometheus-endpoint>:16001'
        - '<broker-prometheus-endpoint>:16002'
        - '<broker-prometheus-endpoint>:16003'
  - job_name: 'rabbitmq-connection'
    scheme: https
    basic_auth:
      username: <username>
      password: <password>
    metrics_path: '/metrics/detailed'
    params:
      family: ['connection_metrics']
    static_configs:
      - targets:
        - '<broker-prometheus-endpoint>:16001'
        - '<broker-prometheus-endpoint>:16002'
        - '<broker-prometheus-endpoint>:16003'
  - job_name: 'rabbitmq-channel'
    scheme: https
    basic_auth:
      username: <username>
      password: <password>
    metrics_path: '/metrics/detailed'
    params:
      family: ['channel_metrics']
    static_configs:
      - targets:
        - '<broker-prometheus-endpoint>:16001'
```

```
        - '<broker-prometheus-endpoint>:16002'
        - '<broker-prometheus-endpoint>:16003'
  - job_name: 'rabbitmq-exchange-count'
    scheme: https
    basic_auth:
      username: <username>
      password: <password>
    metrics_path: '/metrics/detailed'
    params:
      family: ['exchange_names']
    static_configs:
      - targets:
        - '<broker-prometheus-endpoint>:16001'
        - '<broker-prometheus-endpoint>:16002'
        - '<broker-prometheus-endpoint>:16003'
```

ActiveMQ용 Amazon MQ 브로커에 사용 가능한 CloudWatch 지표

ActiveMQ용 Amazon MQ 지표

지표	단위	설명
AmqpMaximumConnections	개수	AMQP를 사용하여 브로커에 연결할 수 있는 최대 클라이언트 수입니다. 연결 할당량에 대한 자세한 내용은 Quotas in Amazon MQ 섹션을 참조하세요.
BurstBalance	%	처리량이 최적화된 브로커의 메시지 데이터를 유지하는 데 사용되는 Amazon EBS 볼륨에 남아 있는 버스트 크레딧 비율입니다. 이 밸런스가 0에 도달하면 버스트 밸런스가 다시 채워질 때까지 Amazon EBS 볼륨에서 제공하는 IOPS가 감소합니다. Amazon EBS에서 버스트 밸런스 작동 방식에 대한

지표	단위	설명
		자세한 내용은 I/O 크레딧 및 버스트 성능을 참조하세요.
CpuCreditBalance	크레딧(vCPU-분)	**Important** 이 지표는 mq.t2.micro 브로커 인스턴스 유형에 대해서만 사용할 수 있습니다. CPU 크레딧 지표는 5분 간격으로만 사용할 수 있습니다. 시작된 후 인스턴스가 누적한 획득 CPU 크레딧 수입니다(시작 크레딧 수 포함). 크레딧 밸런스는 브로커 인스턴스가 기준 CPU 사용률을 초과하여 버스트를 소비하는 경우 사용할 수 있습니다. 크레딧은 획득 후 크레딧 밸런스에 누적되고, 소비 후 크레딧 밸런스에서 제거됩니다. 크레딧 밸런스에는 최대 한도가 있습니다. 이 한도에 도달하면 새로 획득한 크레딧은 폐기됩니다.
CpuUtilization	%	브로커가 현재 사용하는 할당된 Amazon EC2 컴퓨팅 유닛(ECU)의 비율입니다.
CurrentConnectionsCount	개수	현재 브로커의 현재 활성 연결 수입니다.

지표	단위	설명
EstablishedConnectionsCount	개수	브로커에 설정되어 있는 활성 및 비활성 상태의 총 연결 수입니다.
HeapUsage	%	브로커가 현재 사용하는 ActiveMQ JVM 메모리 제한의 비율입니다.
InactiveDurableTopicSubscribersCount	개수	비활성 상태인 영구적인 주제 가입자의 수(최대 2000)입니다.
JobSchedulerStorePercentUsage	%	작업 스케줄러 스토어에 사용되는 디스크 공간의 백분율입니다.
JournalFilesForFastRecovery	개수	완전한 종료 후 재생될 저널 파일 수입니다.
JournalFilesForFullRecovery	개수	불완전한 종료 후 재생될 저널 파일 수입니다.
MqttMaximumConnections	개수	MQTT를 사용하여 브로커에 연결할 수 있는 최대 클라이언트 수입니다. 연결 할당량에 대한 자세한 내용은 Quotas in Amazon MQ 섹션을 참조하세요.
NetworkConnectorConnectionCount	개수	NetworkConnector를 사용하여 브로커 네트워크의 브로커에 연결된 노드 수입니다.
NetworkIn	바이트	브로커의 수신 트래픽 볼륨입니다.

지표	단위	설명
NetworkOut	바이트	브로커의 발신 트래픽 볼륨입니다.
OpenTransactionCount	개수	진행 중인 총 트랜잭션 수입니다.
OpenwireMaximumConnections	개수	OpenWire를 사용하여 브로커에 연결할 수 있는 최대 클라이언트 수입니다. 연결 할당량에 대한 자세한 내용은 Quotas in Amazon MQ 섹션을 참조하세요.
StompMaximumConnections	개수	STOMP를 사용하여 브로커에 연결할 수 있는 최대 클라이언트 수입니다. 연결 할당량에 대한 자세한 내용은 Quotas in Amazon MQ 섹션을 참조하세요.
StorePercentUsage	%	스토리지 제한에서 사용된 비율입니다. 이 비율이 100에 도달하면 브로커가 메시지를 거부합니다.
TempPercentUsage	%	비영구 메시지에 사용되는 사용 가능한 임시 스토리지의 백분율입니다.
TotalConsumerCount	개수	현재 브로커의 대상에 구독한 메시지 소비자 수입니다.
TotalMessageCount	개수	브로커에 저장된 메시지 수.
TotalProducerCount	개수	현재 브로커의 대상에 활성화된 메시지 생산자 수입니다.

지표	단위	설명
VolumeReadOps	개수	Amazon EBS 볼륨에서 수행된 읽기 작업 수입니다.
VolumeWriteOps	개수	Amazon EBS 볼륨에서 수행된 쓰기 작업 수입니다.
WsMaximumConnections	개수	WebSocket을 사용하여 브로커에 연결할 수 있는 최대 클라이언트 수입니다. 연결 할당량에 대한 자세한 내용은 Quotas in Amazon MQ 섹션을 참조하세요.

ActiveMQ 브로커 지표 차원

차원	설명
Broker	브로커의 이름입니다. **Note** 단일 인스턴스 브로커의 접미사는 -1입니다. 고가용성을 위한 활성/대기 브로커의 접미사는 해당 중복 페어인 -1 및 -2입니다.

ActiveMQ 대상(대기열 및 주제) 지표

Important

다음 지표에는 CloudWatch 폴링 기간에 대한 분당 카운트가 포함됩니다.

- EnqueueCount
- ExpiredCount

- DequeueCount
- DispatchCount
- InFlightCount

예를 들어, 5분 CloudWatch 기간에서 EnqueueCount에는 각 기간의 1분 부분에 대해 5개의 카운트 값이 있습니다. Minimum 및 Maximum 통계는 지정된 기간 동안 최저 분당 값과 최고 분당 값을 제공합니다.

지표	단위	설명
ConsumerCount	개수	대상에 구독한 소비자 수입니다.
EnqueueCount	개수	분당 대상에 전송된 메시지 수입니다.
EnqueueTime	시간(밀리초)	메시지가 브로커에 도달하고 소비자에게 전달되는 종단 간 지연 시간입니다. **Note** EnqueueTime 은 생산자가 메시지를 보낼 때부터 메시지가 브로커에 도달 할 때까지 종단 간 대기 시간도, 브로커가 메시지를 수신할 때부터 브로커에서 메시지를 승인할 때까지 대기 시간도 측정하지 않습니다. 대신 EnqueueTime 은 브로커가 메시지를 수시한 순간부터 소비자에

지표	단위	설명
		게 성공적으로 배달될 때까지 시간(밀리초)입니다.
ExpiredCount	개수	만료되어 전달할 수 없는 분당 메시지 수입니다.
DispatchCount	개수	소비자에게 전송된 분당 메시지 수입니다.
DequeueCount	개수	소비자가 승인한 분당 메시지 수입니다.
InFlightCount	개수	확인되지 않은 소비자에게 보낸 메시지 수.
ReceiveCount	개수	전이중 네트워크 커넥터에 대해 원격 브로커로부터 받은 메시지 수입니다.
MemoryUsage	%	대상이 현재 사용하는 메모리 제한의 비율(%).
ProducerCount	개수	대상의 생산자 수.
QueueSize	개수	대기열에 있는 메시지의 수입니다. **Important** 이 지표는 대기열에만 적용됩니다.
TotalEnqueueCount	개수	브로커에게 전송된 총 메시지 수입니다.

지표	단위	설명
TotalDequeueCount	개수	클라이언트가 사용한 총 메시지 수입니다.

> **Note**
>
> TotalEnqueueCount 및 TotalDequeueCount 지표에는 자문 주제에 대한 메시지가 포함됩니다. 공지 주제 메시지에 대한 자세한 내용은 ActiveMQ 설명서를 참조하세요.

ActiveMQ 대상(대기열 및 주제) 지표의 차원

차원	설명
Broker	브로커의 이름입니다. **Note** 단일 인스턴스 브로커의 접미사는 -1입니다. 고가용성을 위한 활성/대기 브로커의 접미사는 해당 중복 페어인 -1 및 -2입니다.
Topic 또는 Queue	주제 또는 대기열의 이름입니다.
NetworkConnector	네트워크 커넥터의 이름입니다.

RabbitMQ용 Amazon MQ 브로커에 사용 가능한 CloudWatch 지표

> **Warning**
>
> RabbitMQ 4.2부터 RabbitMQIOReadAverageTime 및 RabbitMQIOWriteAverageTime는 더 이상 사용되지 않으며 의미 있는 값을 게시하지 않습니다. 이러한 지표는 다음 주요 RabbitMQ 릴리스에서 CloudWatch에서 제거됩니다.

RabbitMQ 브로커 지표

> **Note**
>
> 관리 플러그인은 오픈 소스 RabbitMQ의 프로덕션 또는 장기 모니터링에는 권장되지 않습니다. Prometheus를 사용하여 RabbitMQ 4.2 이상에서 노드별 지표를 쿼리하는 것이 좋습니다.

지표	단위	설명
`ExchangeCount`	개수	브로커에 구성된 총 교환 수입니다.
`QueueCount`	개수	브로커에 구성된 총 대기열 수입니다.
`ConnectionCount`	개수	브로커에서 설정된 총 연결 수입니다.
`ChannelCount`	개수	브로커에서 설정된 총 채널 수입니다. **Important** 채널의 개념은 AMQP 0-9-1에만 해당됩니다.
`ConsumerCount`	개수	브로커에 연결된 총 소비자 수입니다.
`MessageCount`	개수	대기열에 있는 총 메시지 수입니다. **Note** 생성된 수는 브로커에서 준비되어 승인되지

지표	단위	설명
		않은 메시지의 총 합계입니다.
MessageReadyCount	개수	대기열에 있는 준비된 메시지의 총 수입니다.
MessageUnacknowledgedCount	개수	대기열에 있는 승인되지 않은 메시지의 총 수입니다.
PublishRate	개수	메시지가 브로커에 게시되는 비율입니다. 생성된 수는 샘플링 시 초당 메시지 수를 나타냅니다. **Important** 이 지표는 AMQP 0-9-1 프로토콜 활동만 반영합니다. AMQP 1.0 지표는 섹션을 참조하세요Prometheus 지표 액세스.

지표	단위	설명
ConfirmRate	개수	RabbitMQ 서버가 게시된 메시지를 확인하는 비율입니다. 이 지표를 PublishRate 와 비교하여 브로커 성능을 보다 잘 파악할 수 있습니다. 생성된 수는 샘플링 시 초당 메시지 수를 나타냅니다. **Important** 이 지표는 AMQP 0-9-1 프로토콜 활동만 반영합니다. AMQP 1.0 지표는 섹션을 참조하세요Prometheus 지표 액세스.
AckRate	개수	소비자가 메시지를 승인하는 비율입니다. 생성된 수는 샘플링 시 초당 메시지 수를 나타냅니다. **Important** 이 지표는 AMQP 0-9-1 프로토콜 활동만 반영합니다. AMQP 1.0 지표는 섹션을 참조하세요Prometheus 지표 액세스.

지표	단위	설명
SystemCpuUtilization	%	브로커가 현재 사용하는 할당된 Amazon EC2 컴퓨팅 유닛(ECU)의 비율입니다. 클러스터 배포의 경우 이 값은 세 개의 RabbitMQ 노드의 해당 지표 값 모두의 집계를 나타냅니다.
RabbitMQMemLimit	바이트	RabbitMQ 브로커의 RAM 한도입니다. 이 지표는 인스턴스 유형에 따라 다릅니다. 자세한 내용은 the section called “메모리 및 디스크 경보” 단원을 참조하십시오. 클러스터 배포의 경우 이 값은 세 개의 RabbitMQ 노드의 해당 지표 값 모두의 집계를 나타냅니다.
RabbitMQMemUsed	바이트	RabbitMQ 브로커가 사용하는 RAM의 양입니다. 클러스터 배포의 경우 이 값은 세 개의 RabbitMQ 노드의 해당 지표 값 모두의 집계를 나타냅니다.
RabbitMQDiskFreeLimit	바이트	RabbitMQ 브로커의 디스크 한도입니다. 클러스터 배포의 경우 이 값은 세 개의 RabbitMQ 노드의 해당 지표 값 모두의 집계를 나타냅니다. 이 지표는 인스턴스 유형 및 배포 모드에 따라 달라집니다. 자세한 내용은 the section called “메모리 및 디스크 경보” 단원을 참조하십시오.

지표	단위	설명
RabbitMQDiskFree	바이트	RabbitMQ 브로커에서 사용할 수 있는 사용 가능한 디스크 공간의 총 볼륨입니다. 디스크 사용량이 한도를 초과하면 클러스터는 모든 생산자 연결을 차단합니다. 클러스터 배포의 경우 이 값은 세 개의 RabbitMQ 노드의 해당 지표 값 모두의 집계를 나타냅니다.
RabbitMQFdUsed	개수	사용된 파일 설명자 수입니다. 클러스터 배포의 경우 이 값은 세 개의 RabbitMQ 노드의 해당 지표 값 모두의 집계를 나타냅니다.
RabbitMQIOReadAverageTime	개수	RabbitMQ가 한 번의 읽기 작업을 수행하는 데 걸리는 평균 시간(밀리초)입니다. 값은 메시지 크기에 비례합니다.
RabbitMQIOWriteAverageTime	개수	RabbitMQ가 한 번의 쓰기 작업을 수행하는 데 걸리는 평균 시간(밀리초)입니다. 값은 메시지 크기에 비례합니다.

RabbitMQ 브로커 지표의 차원

차원	설명
Broker	브로커의 이름입니다.

RabbitMQ 노드 지표

지표	단위	설명
SystemCpuUtilization	%	브로커가 현재 사용하는 할당된 Amazon EC2 컴퓨팅 유닛(ECU)의 비율입니다.
RabbitMQMemLimit	바이트	RabbitMQ 노드의 RAM 한도입니다. 이 지표는 인스턴스 유형에 따라 다릅니다. 자세한 내용은 the section called “메모리 및 디스크 경보” 단원을 참조하십시오.
RabbitMQMemUsed	바이트	RabbitMQ 노드가 사용하는 RAM의 양입니다. 메모리 사용이 한도를 초과하면 클러스터는 모든 생산자 연결을 차단합니다.
RabbitMQDiskFreeLimit	바이트	RabbitMQ 노드의 디스크 한도입니다. 이 지표는 인스턴스 유형 및 배포 모드에 따라 달라집니다. 자세한 내용은 the section called “메모리 및 디스크 경보” 단원을 참조하십시오.
RabbitMQDiskFree	바이트	RabbitMQ 노드에서 사용할 수 있는 사용 가능한 디스크 공간의 총 볼륨입니다. 디스크 사용량이 한도를 초과하면 클러스터는 모든 생산자 연결을 차단합니다.
RabbitMQFdUsed	개수	사용된 파일 설명자 수입니다.

지표	단위	설명
ExchangeCount	개수	노드에 구성된 총 교환 수입니다. RabbitMQ 4.2 이상에서 사용할 수 있습니다.
QueueCount	개수	노드에 구성된 총 대기열 수입니다. RabbitMQ 4.2 이상에서 사용할 수 있습니다.
ConnectionCount	개수	노드에 설정된 총 연결 수입니다. RabbitMQ 4.2 이상에서 사용할 수 있습니다.
ChannelCount	개수	노드에 설정된 총 채널 수입니다. RabbitMQ 4.2 이상에서 사용할 수 있습니다.
ConsumerCount	개수	노드에 연결된 총 소비자 수입니다. RabbitMQ 4.2 이상에서 사용할 수 있습니다.
MessageCount	개수	노드의 대기열에 있는 총 메시지 수입니다. RabbitMQ 4.2 이상에서 사용할 수 있습니다.
MessageReadyCount	개수	노드의 대기열에 있는 준비된 메시지의 총 수입니다. RabbitMQ 4.2 이상에서 사용할 수 있습니다.
MessageUnacknowledgedCount	개수	노드의 대기열에 있는 승인되지 않은 메시지의 총 수입니다. RabbitMQ 4.2 이상에서 사용할 수 있습니다.

RabbitMQ 노드 지표에서 클러스터 전체 지표 집계

집계된 클러스터 전체 지표를 가져오려면 브로커 이름과 지표 이름을 기준으로 필터링하여 CloudWatch 콘솔에서 해당 노드별 지표를 찾을 수 있습니다. 그런 다음 확인란을 클릭하여 해당 지표를 선택하고 수학 추가 > 공통 > 합계를 선택합니다.

RabbitMQ 노드 지표의 차원

차원	설명
`Node`	노드의 이름입니다. **Note** 노드 이름은 접두사(일반적으로 `rabbit`) 및 호스트 이름의 두 부분으로 구성됩니다. 예를 들어 `rabbit@ip-10-0-0-230.us-west-2.compute.internal` 은 접두사가 `rabbit`이고 호스트 이름이 `ip-10-0-0-230.us-west-2.compute.internal` 인 노드 이름입니다.
`Broker`	브로커의 이름입니다.

RabbitMQ 대기열 지표

지표	단위	설명
`ConsumerCount`	개수	대기열을 구독하는 소비자 수입니다.
`MessageReadyCount`	개수	현재 배달할 수 있는 메시지 수입니다.

지표	단위	설명
MessageUnacknowledgedCount	개수	서버가 승인을 대기 중인 메시지 수입니다.
MessageCount	개수	MessageReadyCount 및 MessageUnacknowledgedCount 의 총 수(대기열 깊이라고도 함)입니다.

RabbitMQ 대기열 지표의 차원

> **Note**
>
> RabbitMQ용 Amazon MQ에서는 이름에 공백, 탭 또는 기타 ASCII가 아닌 문자가 포함된 가상 호스트 및 대기열의 지표를 게시할 수 없습니다.
> 차원 이름에 대한 자세한 내용은 Amazon CloudWatch API 참조의 차원을 참조하세요.

차원	설명
Queue	대기열의 이름입니다.
VirtualHost	가상 호스트의 이름입니다.
Broker	브로커의 이름입니다.

RabbitMQ 네트워크 지표

지표	단위	설명
NetworkOut	바이트	모든 네트워크 인터페이스에서 인스턴스가 보낸 바이트 수입니다. 이 측정치는 단일 인스턴스에서 나가는 네트워크 트래픽의 볼륨을 식별합니다. 보고된 숫자는 해당 기간에 전송된 바이트 수입니다. 기본(5분) 모니터링을 사용하고

지표	단위	설명
		통계가 합계인 경우, 이 숫자를 300으로 나누어 바이트/초를 찾을 수 있습니다. 세부(1분) 모니터링으로 설정되어 있고 통계가 합계인 경우 60으로 나눕니다. CloudWatch 지표 수학 함수 `DIFF_TIME` 을 사용하여 초당 바이트 수를 찾을 수도 있습니다. 예를 들어 CloudWatch에서 NetworkOut을 `m1`으로 그래프로 표시한 경우 지표 수학 공식 `m1/(DIFF_TIME(m1))` 은 지표(바이트/초)를 반환합니다. `DIFF_TIME` 및 지표 수학 함수에 대한 자세한 설명은 지표 수학 사용을 참조하세요. 유용한 통계: 합계, 평균, 최소, 최대
`NetworkIn`	바이트	모든 네트워크 인터페이스에서 인스턴스가 받은 바이트 수입니다. 이 측정치는 단일 인스턴스로 들어오는 네트워크 트래픽의 볼륨을 식별합니다. 보고된 숫자는 해당 기간에 수신된 바이트 수입니다. 기본(5분) 모니터링을 사용하고 통계가 합계인 경우, 이 숫자를 300으로 나누어 바이트/초를 찾을 수 있습니다. 세부(1분) 모니터링으로 설정되어 있고 통계가 합계인 경우 60으로 나눕니다. CloudWatch 지표 수학 함수 `DIFF_TIME` 을 사용하여 초당 바이트 수를 찾을 수도 있습니다. 예를 들어 CloudWatch에서 NetworkIn을 `m1`으로 그래프로 표시한 경우 지표 수학 공식 `m1/(DIFF_TIME(m1))` 은 지표(바이트/초)를 반환합니다. `DIFF_TIME` 및 지표 수학 함수에 대한 자세한 설명은 지표 수학 사용을 참조하세요. 유용한 통계: 합계, 평균, 최소, 최대

RabbitMQ 브로커 지표의 차원

차원	설명
`BrokerId`	브로커의 ID

RabbitMQ Amazon MQ 로그 구성

RabbitMQ 브로커에 대해 CloudWatch 로깅을 활성화하면 Amazon MQ는 서비스 연결 역할을 사용하여 일반 로그를 CloudWatch에 게시합니다. 브로커를 처음 생성할 때 Amazon MQ 서비스 연결 역할이 없는 경우 Amazon MQ에서 자동으로 생성합니다. 이후의 모든 RabbitMQ 브로커는 동일한 서비스 연결 역할을 사용하여 CloudWatch에 로그를 게시합니다.

서비스 연결 역할에 대한 자세한 내용은 AWS Identity and Access Management 사용 설명서에서 서비스 연결 역할 사용을 참조하세요 Amazon MQ가 서비스 연결 역할을 사용하는 방법에 대한 자세한 내용은 the section called “서비스 연결 역할 사용” 단원을 참조하세요.

를 사용하여 Amazon MQ API 호출 로깅 AWS CloudTrail

Amazon MQ는 사용자 AWS CloudTrail, 역할 또는 서비스가 수행하는 Amazon MQ 호출의 레코드를 제공하는 AWS 서비스인와 통합됩니다. CloudTrail은 Amazon MQ 콘솔을 통한 호출과 Amazon MQ API를 통한 코드 호출을 포함하여 Amazon MQ 블로커 및 구성과 관련된 API 호출을 이벤트로 캡처합니다. CloudTrail에 대한 자세한 내용은 AWS CloudTrail 사용자 안내서를 참조하세요.

> **Note**
>
> CloudTrail은 ActiveMQ 작업(예: 메시지 송수신) 또는 ActiveMQ 웹 콘솔과 관련된 API 호출은 로깅하지 않습니다. ActiveMQ 작업과 관련된 정보를 로깅하려면 일반 및 감사 로그를 Amazon CloudWatch Logs에 게시하도록 Amazon MQ를 구성할 수 있습니다.

CloudTrail에서 수집하는 정보를 사용하여 Amazon MQ API에 대한 특정 요청, 요청자의 IP 주소, 요청자의 자격 증명, 요청의 날짜 및 시간 등을 식별할 수 있습니다. 추적을 구성하면 CloudTrail 이벤트를 지속적으로 Amazon S3 버킷에 배포할 수 있습니다. 추적을 구성하지 않는 경우 CloudTrail 콘솔의 이벤트 기록에서 최신 이벤트를 볼 수 있습니다. 자세한 내용은 AWS CloudTrail 사용 설명서에서 추적 생성 개요를 참조하세요.

CloudTrail의 Amazon MQ 정보

AWS 계정을 생성하면 CloudTrail이 활성화됩니다. 지원되는 Amazon MQ 이벤트 활동이 발생하면 이벤트 기록에 다른 AWS 서비스 이벤트와 함께 CloudTrail 이벤트에 기록됩니다. AWS 계정에 대한 최신 이벤트를 확인, 검색 및 다운로드할 수 있습니다. 자세한 내용은AWS CloudTrail 사용 설명서에서 CloudTrail 이벤트 기록을 사용하여 이벤트 보기를 참조하세요.

CloudTrail은 추적을 사용하여 Amazon S3 버킷으로 로그 파일을 전송할 수 있습니다. AWS 계정에 이벤트를 지속적으로 기록하는 추적을 생성할 수 있습니다. 기본적으로를 사용하여 추적을 생성하면 추적 AWS Management Console이 모든 AWS 리전에 적용됩니다. 추적은 모든 AWS 리전의 이벤트를 로깅하고 지정된 Amazon S3 버킷으로 로그 파일을 전송합니다. CloudTrail 로그에서 수집된 이벤트 데이터를 추가로 분석하고 처리하도록 다른 AWS 서비스를 구성할 수도 있습니다. 자세한 내용은AWS CloudTrail 사용 설명서에서 다음 주제를 참조하세요.

- CloudTrail 지원 서비스 및 통합
- CloudTrail에서 Amazon SNS 알림 구성
- 여러 리전에서 CloudTrail 로그 파일 수신
- 여러 계정에서 CloudTrail 로그 파일 수신

Amazon MQ는 다음 API에 대한 요청 파라미터와 응답을 CloudTrail 로그 파일에 이벤트로 로깅할 수 있습니다.

- `CreateConfiguration`
- `DeleteBroker`
- `DeleteUser`
- `RebootBroker`
- `UpdateBroker`

> **Note**
>
> RebootBroker 로그 파일은 브로커를 재부팅할 때 기록됩니다. 유지 관리 기간 중에는 서비스가 자동으로 재부팅되고 RebootBroker 로그 파일은 기록되지 않습니다.

> **Important**
>
> 다음 API의 `GET` 방법의 경우 요청 파라미터는 로깅되지만 응답은 수정됩니다.
>
> - `DescribeBroker`
> - `DescribeConfiguration`
> - `DescribeConfigurationRevision`
> - `DescribeUser`

- ListBrokers
- ListConfigurationRevisions
- ListConfigurations
- ListUsers

다음 API의 경우 data 및 password 요청 파라미터는 별표(***)로 숨겨집니다.

- CreateBroker (POST)
- CreateUser (POST)
- UpdateConfiguration (PUT)
- UpdateUser (PUT)

모든 이벤트 또는 로그 항목에는 요청자에 대한 정보가 들어 있습니다. 이 정보를 이용하면 다음을 쉽게 판단할 수 있습니다.

- 요청이 루트를 통해 이루어졌습니까? 아니면 사용자 자격 증명을 통해 이루어졌습니까?
- 역할 또는 연합된 사용자에 대한 임시 보안 자격 증명을 사용하여 요청이 이루어졌습니까?
- 요청이 다른 AWS 서비스에서 이루어졌습니까?

자세한 내용은 AWS CloudTrail 사용 설명서의 CloudTrail userIdentity 요소를 참조하세요.

Amazon MQ 로그 파일 항목 예

추적이란 지정한 Amazon S3 버킷에 이벤트를 로그 파일로 전달하도록 허용하는 구성입니다. CloudTrail 로그 파일에는 하나 이상의 로그 항목이 포함될 수 있습니다.

이벤트는 특정 소스의 단일 요청을 나타내며 Amazon MQ API에 대한 요청, 요청자의 IP 주소, 요청자의 자격 증명, 요청의 날짜 및 시간 등에 대한 정보를 포함합니다.

다음 예제에서는 CreateBroker API 호출의 CloudTrail 로그 항목을 보여줍니다.

> **Note**
>
> CloudTrail 로그 파일은 퍼블릭 API의 순서가 지정된 스택 추적이 아니므로 특정 순서로 정보를 나열하지 않습니다.

```
{
    "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"
}
```


ActiveMQ 로그용 Amazon MQ 구성

Amazon MQ가 CloudWatch Logs로 로그를 게시하도록 허용하려면 Amazon MQ 사용자에게 권한을 추가하고 브로커를 생성하거나 다시 시작하기 전에 Amazon MQ에 대한 리소스 기반 정책도 구성해야 합니다.

> **Note**
>
> ActiveMQ 웹 콘솔에서 로그를 켜고 메시지를 게시하면 메시지 내용이 CloudWatch로 전송되고 로그에 표시됩니다.

다음 내용에서는 ActiveMQ 브로커에 대한 CloudWatch Logs를 구성하는 단계를 설명합니다.

주제

- CloudWatch Logs에서 로깅의 구조 이해

- Amazon MQ 사용자에게 CreateLogGroup 권한 추가
- Amazon MQ에 대한 리소스 기반 정책 구성
- 교차 서비스 혼동된 대리인 방지

CloudWatch Logs에서 로깅의 구조 이해

브로커를 생성할 때나 브로커를 편집할 때 고급 브로커 설정을 구성하여 일반 및 감사 로깅을 활성화할 수 있습니다.

일반 로깅은 기본 `INFO` 로깅 수준(`DEBUG` 로깅이 지원되지 않음)을 활성화하고 `activemq.log`를 CloudWatch 계정의 로그 그룹에 게시합니다. 로그 그룹의 형식은 다음과 같습니다.

```
/aws/amazonmq/broker/b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9/general
```

감사 로깅은 JMX를 사용하거나 ActiveMQ 웹 콘솔을 사용하여 수행된 관리 작업의 로깅을 활성화하고 `audit.log`를 CloudWatch 계정의 로그 그룹에 게시합니다. 로그 그룹의 형식은 다음과 같습니다.

```
/aws/amazonmq/broker/b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9/audit
```

단일 인스턴스 브로커가 있는지 또는 활성/대기 브로커가 있는지에 따라 Amazon MQ는 각 로그 그룹 내에 1개 또는 2개의 로그 스트림을 생성합니다. 로그 스트림의 형식은 다음과 같습니다.

```
activemq-b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.log
                activemq-b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-2.log
```

`-1` 및 `-2` 접미사는 개별 브로커 인스턴스를 나타냅니다. 자세한 내용은 Amazon CloudWatch Logs 사용 설명서의 로그 그룹 및 로그 스트림 작업을 참조하세요.

Amazon MQ 사용자에게 **`CreateLogGroup`** 권한 추가

Amazon MQ가 CloudWatch Logs 로그 그룹을 생성하도록 허용하려면 브로커를 생성하거나 재부팅하는 사용자에게 `logs:CreateLogGroup` 권한이 있는지 확인해야 합니다.

> **Important**
>
> 사용자가 브로커를 생성하거나 재부팅하기 전에 `CreateLogGroup` 권한을 Amazon MQ 사용자에게 추가하지 않으면 Amazon MQ가 로그 그룹을 생성하지 않습니다.

다음 IAM 기반 정책 예제에서는 이 정책이 연결된 사용자에 대해 `logs:CreateLogGroup`에 대한 권한을 부여합니다.

JSON

```
{
                    "Version":"2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "logs:CreateLogGroup",
                            "Resource": "arn:aws:logs:*:*:log-group:/aws/
amazonmq/*"
                        }
                    ]
                    }
```

> **Note**
>
> 여기서, 사용자는 새 브로커를 구성할 때 생성되는 사용자를 나타내며 Amazon MQ 사용자가 아닙니다. 사용자 설정 및 IAM 정책 구성에 대한 자세한 내용은 IAM 사용 설명서의 보안 인증 정보 관리 개요 섹션을 참조하세요.

자세한 내용은 Amazon CloudWatch Logs API 참조의 `CreateLogGroup` 단원을 참조하세요.

Amazon MQ에 대한 리소스 기반 정책 구성

> **Important**
>
> Amazon MQ에 대한 리소스 기반 정책을 구성하지 않으면 브로커가 CloudWatch Logs에 로그를 게시할 수 없습니다.

Amazon MQ가 CloudWatch Logs 로그 그룹에 로그를 게시하도록 허용하려면 Amazon MQ에 다음 CloudWatch Logs API 작업에 대한 액세스 권한을 부여하도록 리소스 기반 정책을 구성합니다.

- `CreateLogStream` - 지정된 로그 그룹에 대한 CloudWatch Logs 로그 스트림을 생성합니다.

- PutLogEvents - 지정된 CloudWatch Logs 로그 스트림에 이벤트를 전달합니다.

다음 리소스 기반 정책은 logs:CreateLogStream 및 logs:PutLogEvents에 대한 권한을 부여합니다 AWS.

JSON

```
{
                        "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/*"
                            }
                        ]
                    }
```

이 리소스 기반 정책은 다음 명령과 AWS CLI 같이를 사용하여 구성해야 합니다. 이 예제에서 *us-east-1*은 사용자 고유의 정보로 바꿉니다.

```
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**
>
> 이 예제에서는 /aws/amazonmq/ 접두사를 사용하기 때문에 리전별로 AWS 계정당 한 번만 리소스 기반 정책을 구성해야 합니다.

교차 서비스 혼동된 대리인 방지

혼동된 대리자 문제는 작업을 수행할 권한이 없는 엔터티가 권한이 더 많은 엔터티에게 작업을 수행하도록 강요할 수 있는 보안 문제입니다. 에서 AWS교차 서비스 가장은 혼동된 대리자 문제를 초래할 수 있습니다. 교차 서비스 가장은 한 서비스(직접 호출하는 서비스)가 다른 서비스(직접 호출되는 서비스)를 직접 호출할 때 발생할 수 있습니다. 직접 호출하는 서비스는 다른 고객의 리소스에 대해 액세스 권한이 없는 방식으로 작동하게 권한을 사용하도록 조작될 수 있습니다. 이를 방지하기 위해는 계정의 리소스에 대한 액세스 권한이 부여된 서비스 보안 주체를 사용하여 모든 서비스에 대한 데이터를 보호하는 데 도움이 되는 도구를 AWS 제공합니다.

Amazon MQ 리소스 기반 정책의 전역 조건 컨텍스트 키인 `aws:SourceArn` 및 `aws:SourceAccount`를 사용하여 하나 이상의 지정된 브로커에 대한 CloudWatch Logs 액세스를 제한하는 것이 좋습니다.

> **Note**
>
> 두 전역 조건 컨텍스트 키를 모두 사용하는 경우 `aws:SourceAccount` 값과 `aws:SourceArn` 값의 계정은 동일한 정책 문에서 사용할 경우 동일한 계정 ID를 사용해야 합니다.

다음 예제에서는 CloudWatch Logs 액세스를 단일 Amazon MQ 브로커로 제한하는 리소스 기반 정책을 보여줍니다.

JSON

```
{
                        "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-
west-1:123456789012:broker:my-broker:123456789012"
                                }
                            }
                            }
                        ]
                        }
```

또한 다음과 같이 계정의 모든 브로커에 대한 CloudWatch Logs 액세스를 제한하도록 리소스 기반 정책을 구성할 수 있습니다.

JSON

```
{
                        "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"
                            }
                            }
```

```
            }
            ]
        }
```

혼동된 대리자 보안 문제에 관한 자세한 내용은 사용 설명서의 혼동된 대리자 문제를 참조하세요.

Amazon MQ로 CloudWatch 로그 구성 문제 해결

경우에 따라서는 CloudWatch Logs가 예상한 대로 동작하지 않을 수도 있습니다. 이 단원에서는 일반적인 문제에 대한 개요를 제공하고 해결 방법을 알려줍니다.

CloudWatch에 로그 그룹이 표시되지 않음

Amazon MQ 사용자에게 `CreateLogGroup` 권한을 추가하고 브로커를 재부팅합니다. 이를 통해 Amazon MQ가 로그 그룹을 생성할 수 있습니다.

CloudWatch Logs 그룹에 로그 스트림이 표시되지 않음

Amazon MQ에 대한 리소스 기반 정책을 구성합니다. 이를 통해 브로커가 로그를 게시할 수 있습니다.

Amazon MQ의 할당량

이 주제에서는 Amazon MQ 내의 할당량에 대해 설명합니다. 특정 AWS 계정에 대해 다음과 같은 많은 제한을 변경할 수 있습니다. 제한 증가를 요청하려면 Amazon Web Services 일반 참조의 AWS 서비스 할당량을 참조하세요. 업데이트된 한도는 한도 증가가 적용된 후에도 표시되지 않습니다. Amazon CloudWatch의 현재 연결 한도를 확인하는 방법에 대한 자세한 내용은 Amazon CloudWatch를 사용하여 Amazon MQ 브로커 모니터링을 참조하세요.

주제

브로커

다음 표에는 Amazon MQ 브로커와 관련된 할당량이 나열되어 있습니다.

Limit	설명
브로커 이름	• AWS 계정에서 고유해야 합니다. • 1-50자여야 합니다. • 인쇄 가능한 ASCII 문자 집합에 지정된 문자만을 포함해야 합니다. • 영숫자, 대시, 마침표, 밑줄 및 물결 기호(- . _ ~)만 포함할 수 있습니다.
리전당 브로커 수	200

Limit	설명
소형 브로커의 프로토콜당 와이어 레벨 연결	**Important** RabbitMQ 브로커에는 적용되지 않습니다. `mq.*.micro` 인스턴스 유형 브로커의 경우 300
대형 브로커의 프로토콜당 와이어 레벨 연결	**Important** RabbitMQ 브로커에는 적용되지 않습니다. `mq.*.*large` 인스턴스 유형 브로커의 경우 2,000
브로커당 보안 그룹	5
CloudWatch에서 모니터링되는 ActiveMQ 대상(대기열 및 주제)	CloudWatch는 처음 1,000개 대상만 모니터링합니다.
CloudWatch에서 모니터링되는 RabbitMQ 대상(대기열)	CloudWatch는 소비자 수에 따라 정렬된 대상 중 처음 500개만 모니터링합니다.
브로커당 태그	50

구성

다음 표에는 Amazon MQ 구성과 관련된 할당량이 나열되어 있습니다.

Limit	설명
구성 이름	• 1-150자여야 합니다.

Limit	설명
	• 인쇄 가능한 ASCII 문자 집합에 지정된 문자만을 포함해야 합니다. • 영숫자, 대시, 마침표, 밑줄 및 물결 기호(- . _ ~)만 포함할 수 있습니다.
구성당 개정	300

Users

다음 표에는 Amazon MQ ActiveMQ 브로커와 관련된 할당량이 나열되어 있습니다.

Limit	설명
사용자 이름	• 1-100자여야 합니다. • 인쇄 가능한 ASCII 문자 집합에 지정된 문자만을 포함해야 합니다. • 영숫자, 대시, 마침표, 밑줄 및 물결 기호(- . _ ~)만 포함할 수 있습니다. • 쉼표(,)를 포함하면 안 됩니다.
암호	• 12-250자여야 합니다. • 인쇄 가능한 ASCII 문자 집합에 지정된 문자만을 포함해야 합니다. • 최소 4개 이상의 고유 문자를 포함해야 합니다. • 쉼표(,)를 포함하면 안 됩니다.

Limit	설명
브로커당 사용자(단순 인증)	250
사용자당 그룹(단순 인증)	20

데이터 저장

다음 표에는 Amazon MQ 데이터 스토리지와 관련된 할당량이 나열되어 있습니다.

Limit	설명
소형 브로커당 스토리지 용량	`mq.*.micro` 인스턴스 유형 브로커의 경우 20GB. Amazon MQ 인스턴스 유형에 관한 자세한 내용은 Broker instance types 단원을 참조하세요.
대형 브로커당 스토리지 용량	`mq.m5.*` 인스턴스 유형 브로커의 경우 200GB. Amazon MQ 인스턴스 유형에 관한 자세한 내용은 Broker instance types 단원을 참조하세요.
Amazon EBS가 지원하는 브로커당 작업 스케줄러 사용 한도	**Important** RabbitMQ 브로커에는 적용되지 않습니다. 50GB. 작업 스케줄러 사용에 대한 자세한 내용은 Apache ActiveMQ API 설명서에서 `JobSchedulerUsage` 단원을 참조하세요.
소형 브로커당 임시 스토리지 용량	**Important** RabbitMQ 브로커에는 적용되지 않습니다.

Limit	설명
	`mq.*.micro` 인스턴스 유형 브로커의 경우 5GB
대형 브로커당 임시 스토리지 용량	**Important** RabbitMQ 브로커에는 적용되지 않습니다. `mq.m5.*` 인스턴스 유형 브로커의 경우 50GB

API 조절

다음 제한 할당량은 서비스 대역폭을 유지하기 위해 모든 Amazon MQ APIs에서 AWS 계정별로 집계됩니다. Amazon MQ API에 대한 자세한 내용은 Amazon MQ REST API 참조를 참조하세요.

Important

이러한 할당량은 ActiveMQ용 Amazon MQ 또는 RabbitMQ용 Amazon MQ 브로커 메시징 API에 적용되지 않습니다. 예를 들어 Amazon MQ가 메시지 전송 또는 수신을 조절하지 않습니다.

API 순간 확장 한도	API 속도 한도
100	15

Amazon MQ 문제 해결

이 단원에서는 Amazon MQ 브로커를 사용할 때 발생할 수 있는 일반적인 문제와 해당 문제를 해결하기 위해 수행할 수 있는 단계에 대해 설명합니다. 일반적인 문제 해결은 섹션을 참조하세요the section called “문제 해결: 일반 Amazon MQ”. 특정 엔진 버전의 문제를 해결하려면 다음 섹션을 참조하세요.

Amazon MQ용 ActiveMQ 문제 해결

문제 해결 주제	설명
일반 문제 해결	이 섹션의 정보를 사용하여 Amazon MQ용 ActiveMQ 브로커를 사용할 때 발생할 수 있는 일반적인 문제를 진단하고 해결할 수 있습니다.
BROKER_ENI_DELETED	Amazon MQ용 ActiveMQ는 브로커의 탄력적 네트워크 인터페이스(ENI)를 삭제하면 `BROKER_ENI_DELETED` 경보를 발생시킵니다.
BROKER_OOM	Amazon MQ용 ActiveMQ는 메모리 용량이 부족하여 브로커가 재시작 루프를 거칠 때 BROKER_OOM 경보를 발생시킵니다.

Amazon MQ용 RabbitMQ 문제 해결

문제 해결 주제	설명
일반 문제 해결	RabbitMQ 브로커 작업 시 발생할 수 있는 일반적인 문제를 진단합니다.
RABBITMQ_MEMORY_ALARM	RabbitMQ는 CloudWatch 지표 `RabbitMQMemUsed` 로 식별되는 브로커의 메모리 사용량이 `RabbitMQMemLimit` 으로

문제 해결 주제	설명
	식별되는 메모리 제한을 초과할 때 높은 메모리 경보를 발생시킵니다.
RABBITMQ_INVALID_KMS_KEY	Amazon MQ의 RabbitMQ는 고객 관리형 AWS KMS key(CMK)으로 생성된 브로커가 AWS Key Management Service (KMS) 키가 비활성화된 것을 감지하면 INVALID_KMS_KEY 중요 작업 필수 코드를 생성합니다. Amazon MQ
RABBITMQ_INVALID_ASSUMEROLE	Amazon MQ에서에 지정된 IAM 역할 ARN을 수임할 수 없는 경우 Amazon MQ의 RabbitMQ는 INVALID_ASSUMEROLE 중요 작업 필수 코드를 생성합니다. `aws.arns.assume_role_arn` Amazon MQ
RABBITMQ_INVALID_ARN_LDAP	Amazon MQ의 RabbitMQ는 LDAP 서비스 계정 암호 ARN이 유효하지 않거나 액세스할 수 없는 경우 INVALID_ARN_LDAP 중요 작업 필수 코드를 생성합니다. Amazon MQ

문제 해결 주제	설명
RABBITMQ_INVALID_ARN_HTTP	Amazon MQ의 RabbitMQ는 HTTP auth_backend에 대한 SSL 인증서 또는 키 파일의 ARNs이 하나 이상 유효하지 않거나 액세스할 수 없는 경우 INVALID_ARN_HTTP 중요 작업 필수 코드를 발생시킵니다. Amazon MQ
RABBITMQ_INVALID_ARN_SSL	Amazon MQ의 RabbitMQ는 EXTERNAL auth_mechanism용 CA 인증서 트러스트 스토어의 ARNs이 하나 이상 유효하지 않거나 액세스할 수 없는 경우 INVALID_ARN_SSL 중요 작업 필수 코드를 생성합니다. Amazon MQ
RABBITMQ_INVALID_ARN	브로커 구성에서 하나 이상의 ARN이 유효하지 않거나 액세스할 수 없는 경우 Amazon MQ의 RabbitMQ는 INVALID_ARN 중요 작업 필수 코드를 생성합니다. Amazon MQ ARNs
RABBITMQ_DISK_ALARM	디스크 제한 경보는 새 메시지가 추가되는 동안 소비되지 않은 메시지 수가 많아 RabbitMQ 노드에서 사용하는 디스크 볼륨이 감소했음을 나타냅니다.

문제 해결 주제	설명
RABBITMQ_BROKER_NOT_UPGRADEABLE_TO_V4	클래식 대기열 또는 Khepri가 활성화된 브로커에서 RabbitMQ 4로 업그레이드를 시도하면 Amazon MQ의 RabbitMQ_BROKER_NOT_UPGRADEABLE_TO_V4 중요 작업 필수 코드를 생성합니다.

문제 해결: 일반 Amazon MQ

이 단원의 정보를 사용하여 Amazon MQ 브로커를 사용할 때 발생할 수 있는 브로커 연결 문제, 브로커 재부팅 등과 같은 일반적인 문제를 진단할 수 있습니다.

목차

브로커 웹 콘솔 또는 엔드포인트에 연결할 수 없습니다.

웹 콘솔 또는 와이어 레벨 엔드포인터를 사용하여 브로커에 연결할 대 문제가 발생하는 경우 다음 단계를 수행하는 것이 좋습니다.

1. 방화벽 뒤에 있는 브로커에 연결하려고 하는지 확인합니다. 브로커에 대한 액세스를 허용하도록 방화벽을 구성해야 할 수도 있습니다.
2. 원하는 작업이 FIPS 엔드포인트를 사용하여 브로커에 연결하는 것이 맞는지 확인합니다. Amazon MQ는 API 작업을 사용할 때 FIPS 엔드포인트만 지원하며, 브로커 인스턴스 자체에 대한 와이어 수준 연결에 대해서는 지원하지 않습니다.
3. 브로커의 Public Accessibility(퍼블릭 액세스 가능성) 옵션이 Yes(예)로 설정되어 있는지 확인합니다. 해당 옵션이 No(아니요)로 설정된 경우 서브넷의 네트워크 액세스 제어 목록(ACL) 규칙을 확인

합니다. 사용자 지정 네트워크 ACL을 생성한 경우 브로커에 대한 액세스를 제공하기 위해 네트워크 ACL 규칙을 변경해야 할 수도 있습니다. Amazon VPC 네트워킹에 대한 자세한 내용은 Amazon VPC 사용 설명서에서 인터넷 액세스 활성화를 참조하세요.

4. 브로커의 보안 그룹 규칙을 확인합니다. 다음 포트에 대한 연결을 허용하고 있는지 확인합니다.

> **Note**
>
> Amazon MQ용 ActiveMQ와 Amazon MQ용 RabbitMQ에서 연결에 서로 다른 포트를 사용하므로 다음 포트는 엔진 유형에 따라 분류됩니다.

Amazon MQ용 ActiveMQ

- 웹 콘솔 - 포트 8162
- OpenWire - 포트 61617
- AMQP - 포트 5671
- STOMP - 포트 61614
- MQTT - 포트 8883
- WSS - 포트 61619

Amazon MQ용 RabbitMQ

- 웹 콘솔 및 관리 API - 포트 443 및 15671
- AMQP - 포트 5671

5. 브로커 엔진 유형에 대해 다음 네트워크 연결 테스트를 실행합니다.

> **Note**
>
> 퍼블릭 액세스 가능성이 없는 브로커의 경우 Amazon MQ 브로커와 동일한 Amazon VPC 내의 Amazon EC2 인스턴스에서 테스트를 실행하고 응답을 평가합니다.

ActiveMQ on Amazon MQ

Amazon MQ용 ActiveMQ 브로커의 네트워크 연결을 테스트하려면

1. 새 터미널 또는 명령줄 창을 엽니다.

2. 다음 nslookup 명령을 실행하여 브로커 DNS 레코드를 쿼리합니다. 활성/대기 배포의 경우 활성 및 대기 엔드포인트를 모두 테스트합니다. 활성/대기 엔드포인트는 고유한 브로커 ID에 추가된 접미사 -1 또는 -2로 식별됩니다. 엔드포인트를 사용자 정보로 대체합니다.

```
$ nslookup b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-west-2.amazonaws.com
```

쿼리가 성공하면 다음과 비슷한 출력이 표시됩니다.

```
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
```

확인된 IP 주소는 Amazon MQ 콘솔에 제공된 IP 주소와 일치해야 합니다. 이는 DNS 서버에서 도메인 이름이 올바르게 확인되고 다음 단계로 이동할 수 있음을 나타냅니다.

3. 다음 telnet 명령을 실행하여 브로커의 네트워크 경로를 테스트합니다. 엔드포인트를 사용자 정보로 대체합니다. *port*를 웹 콘솔의 경우 포트 번호 8162로 대체하거나 필요한 경우 추가 포로토콜을 테스트하려면 다른 와이어 레벨 포트로 대체합니다.

> **Note**
>
> 활성/대기 배포의 경우 telnet을 대기 엔드포인트로 실행하면 Connect failed 오류 메시지를 받습니다. 대기 인스턴스 자체가 실행 중이므로 이는 정상적인 현상이지만 ActiveMQ 프로세스가 실행 중이 아니고 브로커의 Amazon EFS 스토리지 볼륨에 액세스할 수 없습니다. 두 -1 및 -2 엔드포인트 모두에 대해 명령을 실행하여 활성 및 대기 인스턴스를 모두 테스트해야 합니다.

```
$ telnet b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-
west-2.amazonaws.com port
```

활성 인스턴스의 경우 다음과 비슷한 출력이 표시됩니다.

```
Connected to b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-
west-2.amazonaws.com.
Escape character is '^]'.
```

4. 다음 중 하나를 수행하세요.

   - `telnet` 명령이 성공하면 `EstablishedConnectionsCount` 지표를 확인하고 브로커가 최대 와이어 레벨 연결 한도에 도달하지 않았는지 확인합니다. 브로커 `General` 로그를 검토하여 한도에 도달했는지 확인할 수도 있습니다. 이 지표가 0보다 크면 현재 하나 이상의 클라이언트가 브로커에 연결되어 있는 것입니다. 지표에 연결이 0으로 표시되는 경우 브로커 지표가 매분 게시되므로 `telnet` 경로 테스트를 다시 실행하고 연결을 끊기 전 1분 이상 기다립니다.
   - `telnet` 명령이 실패하면 브로커의 탄력적 네트워크 인터페이스 상태를 확인하고 상태가 `in-use`인지 확인합니다. 각 인스턴스의 네트워크 인터페이스에 대해 Amazon VPC 흐름 로그를 생성하고 생성된 흐름 로그를 검토합니다. `telnet` 명령을 실행할 때 브로커의 IP 주소를 찾고 반환 패킷을 포함하여 연결 패킷이 `ACCEPTED`인지 확인합니다. 자세한 내용 및 흐름 로그 예제를 확인하려면 Amazon VPC 개발자 안내서의 흐름 로그 레코드 예를 참조하세요.

5. 다음 `curl` 명령을 실행하여 ActiveMQ 관리 웹 콘솔에 대한 연결을 확인합니다.

```
$ curl https://b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-
west-2.amazonaws.com:8162/index.html
```

명령이 성공하면 출력은 다음과 비슷한 HTML 문서여야 합니다.

```
<!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 MQ용 RabbitMQ 브로커의 네트워크 연결을 테스트하려면

1. 새 터미널 또는 명령줄 창을 엽니다.
2. 다음 nslookup 명령을 실행하여 브로커 DNS 레코드를 쿼리합니다. 엔드포인트를 사용자 정보로 대체합니다.

```
$ nslookup b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-west-2.amazonaws.com
```

쿼리가 성공하면 다음과 비슷한 출력이 표시됩니다.

```
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. 다음 telnet 명령을 실행하여 브로커의 네트워크 경로를 테스트합니다. 엔드포인트를 사용자 정보로 대체합니다. *port*를 웹 콘솔의 경우 포트 443으로 대체하거나 와이어 레벨 AMQP 연결을 테스트하려는 경우 5671로 대체할 수 있습니다.

```
$ telnet b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-
west-2.amazonaws.com port
```

명령이 성공하면 다음과 비슷한 출력이 표시됩니다.

```
Connected to b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-
west-2.amazonaws.com.
Escape character is '^]'.
```

> **Note**
>
> telnet 연결은 몇 초 후 자동으로 닫힙니다.

4. 다음 중 하나를 수행하세요.

   - `telnet` 명령이 성공하면 `ConnectionCount` 지표를 확인하고 브로커가 `max-connections` 기본 정책에 설정된 값에 도달하지 않았음을 확인합니다. 브로커 `Connection.log` 로그 그룹을 검토하여 한도에 도달했는지 확인할 수도 있습니다. 이 지표가 0보다 크면 현재 하나 이상의 클라이언트가 브로커에 연결되어 있는 것입니다. 지표에 연결이 0으로 표시되는 경우 `telnet` 경로 테스트를 다시 실행합니다. 브로커가 CloudWatch에 새 연결 지표를 게시하기 전에 연결이 종료되는 경우 이 프로세스를 반복해야 할 수 있습니다. 지표는 매분 게시됩니다.
   - 퍼블릭 액세스 가능성이 없는 브로커의 경우 `telnet` 명령이 실패하면 브로커의 탄력적 네트워크 인터페이스 상태를 확인하고 상태가 `in-use`인지 확인합니다. 각 네트워크 인터페이스에 대해 Amazon VPC 흐름 로그를 생성하고 생성된 흐름 로그를 검토합니다. `telnet` 명령을 호출할 때 브로커의 IP 주소를 찾고 반환 패킷을 포함하여 연결 패킷이 `ACCEPTED`인지 확인합니다. 자세한 내용 및 흐름 로그 예제를 확인하려면 Amazon VPC 개발자 안내서의 흐름 로그 레코드 예를 참조하세요.

   > **Note**
   >
   > 퍼블릭 액세스 가능성이 있는 Amazon MQ용 RabbitMQ 브로커에는 이 단계가 적용되지 않습니다.

5. 다음 `curl` 명령을 실행하여 RabbitMQ 관리 웹 콘솔에 대한 연결을 확인합니다.

```
$ curl https://b-1234a5b6-78cd-901e-2fgh-3i45j6k178l9-1.mq.us-west-2.amazonaws.com:443/index.html
```

명령이 성공하면 출력은 다음과 비슷한 HTML 문서여야 합니다.

```
<!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>
```

```
...
```

브로커가 실행 중이고 **telnet**을 사용하여 연결을 확인할 수 있지만, 클라이언트 측에서는 연결할 수 없고 SSL 예외를 반환하고 있습니다.

브로커 유지 관리 기간 중에 브로커 엔드포인트 인증서가 업데이트되었을 수 있습니다. Amazon MQ 브로커 인증서는 브로커의 지속적인 가용성과 보안을 유지하기 위해 주기적으로 교체됩니다.

클라이언트의 트러스트 스토어에 대해 인증하기 위해 Amazon Trust Services에서 Amazon 루트 인증 기관(CA)을 이용하는 것이 좋습니다. 모든 Amazon MQ 브로커 인증서는 이 루트 CA로 서명됩니다. Amazon 루트 CA를 사용하면 브로커에 인증서 업데이트가 있을 때마다 새 Amazon MQ 브로커 인증서를 더 이상 다운로드할 필요가 없습니다.

브로커를 생성했지만 브로커 생성에 실패했습니다.

브로커가 `CREATION_FAILED` 상태인 경우 다음을 수행합니다.

- IAM 권한을 확인합니다. 브로커를 생성하려면 AWS 관리형 IAM 정책을 사용하거나 사용자 지정 IAM 정책에 `AmazonMQFullAccess` 올바른 Amazon EC2 권한 집합이 있어야 합니다. 필요한 Amazon EC2 권한에 대한 자세한 내용은 Amazon MQ 브로커를 생성하는 데 필요한 IAM 권한을 참조하세요.
- 브로커에 대해 선택하는 서브넷이 공유 Amazon Virtual Private Cloud(VPC)에 있는지 확인합니다. 공유 Amazon VPC에서 Amazon MQ 브로커를 생성하려면 Amazon VPC를 소유하는 계정에서 생성해야 합니다.

브로커가 다시 시작되었는데 이유를 잘 모르겠습니다.

브로커가 자동으로 다시 시작한 경우 다음 이유 중 하나 때문일 수 있습니다.

- 예약한 주별 유지 관리 기간 때문에 브로커가 다시 시작되었을 수 있습니다. 주기적으로 Amazon MQ는 메시지 브로커의 하드웨어, 운영 체제 또는 엔진 소프트웨어에 대한 유지 관리를 수행합니다. 유지 관리 기간은 다양하지만 메시지 브로커에 대해 예약된 작업에 따라 최대 2시간까지 지속될 수 있습니다. 브로커는 2시간의 유지 관리 기간 중 언제든 다시 시작될 수 있습니다. 브로커 유지 관리 기간에 대한 자세한 내용은 the section called “브로커 유지 관리 예약” 섹션을 참조하세요.
- 브로커 인스턴스 유형이 애플리케이션 워크로드에 적합하지 않을 수 있습니다. 예를 들어, `mq.t3.micro`에서 프로덕션 워크로드를 실행하면 브로커 리소스가 부족해질 수 있습니다. CPU 사용률이 높거나 브로커 메모리 사용량이 높으면 브로커가 예기치 않게 다시 시작될 수 있습니다. 브로

커에서 사용 중인 CPU 및 메모리 양을 확인하려면 엔진 유형에 대한 다음 CloudWatch 지표를 사용합니다.

- Amazon MQ용 ActiveMQ - 할당된 Amazon EC2 컴퓨팅 유닛(ECU) 중 브로커가 현재 사용하는 비율은 `CpuUtilization`을 확인합니다. ActiveMQ JVM 메모리 한도 중 브로커가 현재 사용하는 비율은 `HeapUsage`를 확인합니다.
- Amazon MQ용 RabbitMQ - 할당된 Amazon EC2 컴퓨팅 유닛(ECU) 중 브로커가 현재 사용하는 비율은 `SystemCpuUtilization`을 확인합니다. 사용된 RAM 볼륨(바이트)은 `RabbitMQMemUsed`을 확인하고 RabbitMQ 노드에서 사용하는 메모리 비율은 해당 값을 `RabbitMQMemLimit`로 나눕니다.

브로커 인스턴스 유형 및 워크로드에 적합한 인스턴스 유형을 선택하는 방법에 대한 자세한 내용은 Broker instance types 섹션을 참조하세요.

Amazon MQ용 ActiveMQ 문제 해결

이 섹션의 정보를 사용하여 Amazon MQ용 ActiveMQ 브로커를 사용할 때 발생할 수 있는 일반적인 문제를 진단하고 해결할 수 있습니다.

목차

로깅을 활성화했는데도 CloudWatch Logs에서 브로커에 대한 일반 로그나 감사 로그를 볼 수 없습니다.

CloudWatch Logs에서 브로커에 대한 로그를 볼 수 없는 경우 다음을 수행합니다.

1. 브로커를 생성하거나 재부팅하는 사용자에게 `logs:CreateLogGroup` 권한이 있는지 확인합니다. 사용자가 브로커를 생성하거나 재부팅하기 전에 `CreateLogGroup` 권한을 사용자에게 추가하지 않으면 Amazon MQ가 로그 그룹을 생성하지 않습니다.

2. Amazon MQ가 CloudWatch Logs에 로그에 로그를 게시할 수 있도록 리소스 기반 정책을 구성했는지 확인합니다. Amazon MQ가 CloudWatch Logs 로그 그룹에 로그를 게시하도록 허용하려면 Amazon MQ에 다음 CloudWatch Logs API 작업에 대한 액세스 권한을 부여하도록 리소스 기반 정책을 구성합니다.
   - `CreateLogStream` - 지정된 로그 그룹에 대한 CloudWatch Logs 로그 스트림을 생성합니다.
   - `PutLogEvents` - 지정된 CloudWatch Logs 로그 스트림에 이벤트를 전달합니다.

CloudWatch Logs에 로그를 게시하도록 Amazon MQ용 ActiveMQ를 구성하는 방법에 대한 자세한 내용은 로깅 구성을 참조하세요.

브로커 재시작 또는 유지 관리 기간이 지나면 상태가 **RUNNING**과 같더라도 브로커에 연결할 수 없습니다. 이유?

브로커를 다시 시작했거나, 예약된 유지 관리 기간이 완료된 후 또는 대기 인스턴스가 활성화되는 장애 이벤트로 인해 연결 문제가 발생할 수 있습니다. 두 경우 모두 브로커 재시작 후 연결 문제는 브로커의 Amazon EFS 또는 Amazon EBS 스토리지 볼륨에 비정상적으로 많은 수의 메시지가 존재하여 발생할 수 있습니다. 다시 시작하는 동안 Amazon MQ는 지속적 메시지를 스토리지에서 브로커 메모리로 이동합니다. 이 진단을 확인하기 위해 Amazon MQ for ActiveMQ 브로커에 대한 CloudWatch에서 다음 지표를 모니터링할 수 있습니다.

- **`StoragePercentUsage`** — 100%이거나 100%에 근접한 높은 비율로 인해 브로커가 연결을 거부할 수 있습니다.
- **`JournalFilesForFullRecovery`** — 불완전하게 종료하고 다시 시작한 후 재생될 저널 파일 수를 나타냅니다. 값이 증가하거나 지속적으로 1보다 높으면 다시 시작한 후 연결 문제가 발생할 수 있는 해결되지 않은 트랜잭션이 표시됩니다.
- **`OpenTransactionCount`** — 재시작 후 0보다 큰 숫자는 브로커가 이전에 사용한 메시지를 저장하려고 시도하므로 연결 문제가 발생합니다.

이 문제를 해결하려면 `rollback()` 또는 `commit()` 중 하나를 사용하여 XA 트랜잭션을 해결하는 것이 좋습니다. 자세한 내용을 알려보고, `rollback()`을(를) 사용하여 XA 트랜잭션을 해결하는 코드 예제를 보려면 XA 트랜잭션 복구 단원을 참조하십시오.

일부 클라이언트는 브로커에 연결되는 반면 다른 클라이언트는 연결할 수 없습니다.

브로커가 RUNNING 상태이며 일부 클라이언트는 브로커에 성공적으로 연결할 수 있지만 다른 클라이언트는 그렇게 할 수 없는 경우, 브로커에 대한 와이어 레벨 연결 한도에 도달했을 수 있습니다. 와이어 레벨 연결 한도에 도달했는지 확인하려면 다음을 수행합니다.

- CloudWatch Logs의 Amazon MQ용 ActiveMQ 브로커에 대한 일반 브로커 로그를 확인합니다. 한도에 도달한 경우, 브로커 로그의 `Reached Maximum Connections`을 참조하십시오. Amazon MQ용 ActiveMQ 브로커를 위한 CloudWatch Logs에 대한 자세한 내용은 the section called "CloudWatch Logs에서 로깅의 구조 이해" 섹션을 참조하세요.

와이어 레벨 연결 제한에 도달하면 브로커는 추가 수신 연결을 적극적으로 거부합니다. 이 문제를 해결하려면 브로커 인스턴스 유형을 업그레이드하는 것이 좋습니다. 워크로드에 가장 적합한 인스턴스 유형을 선택하는 방법에 대한 자세한 내용은 Broker instance types 단원을 참조하십시오.

와이어 레벨 연결 수가 브로커 연결 한도보다 작다는 것을 확인한 경우 클라이언트 재부팅과 관련된 문제가 발생할 수 있습니다. 브로커 로그에서 다수의 자주 발생하는 `... Inactive for longer than 600000 ms - removing ...` 항목이 있는지 확인하십시오. 로그 항목은 클라이언트 재부팅 또는 연결 문제를 나타냅니다. 이 효과는 브로커와 자주 연결을 끊고 다시 연결하는 클라이언트를 포함하는 Network Load Balancer(NLB)를 통해 클라이언트가 브로커에 연결할 때 더 분명합니다. 이는 컨테이너 기반 클라이언트에서 더 일반적으로 관찰됩니다.

자세한 내용은 클라이언트 측 로그를 확인하십시오. 브로커는 600000ms 후에 비활성 TCP 연결을 정리하고 연결 소켓을 비웁니다.

작업을 수행할 때 ActiveMQ 콘솔에서 예외 **`org.apache.jasper.JasperException: An exception occurred processing JSP page`**을(를) 볼 수 있습니다.

간단한 인증을 사용하고 대기열 및 주제 승인에 대한 `AuthorizationPlugin`을 구성하는 경우, XML 구성 파일에 있는 `AuthorizationEntries` 요소를 사용하고 모든 대기열 및 주제에 대한 `activemq-webconsole` 그룹 사용 권한을 허용하십시오. 이렇게 하면 ActiveMQ 웹 콘솔이 ActiveMQ 브로커와 통신할 수 있습니다.

다음 예제 `AuthorizationEntry`은 모든 대기열 및 주제에 대한 읽기 및 쓰기 권한을 `activemq-webconsole` 그룹에 부여합니다.

```
<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>
```

마찬가지로 브로커와 LDAP를 통합할 때, `amazonmq-console-admins` 그룹에 대한 권한을 부여하십시오. LDAP 통합에 대한 자세한 내용은 the section called “LDAP 통합의 작동 방식” 섹션을 참조하세요.

문제 해결: Amazon MQ용 RabbitMQ

이 섹션의 정보를 사용하여 Amazon MQ용 RabbitMQ 브로커를 사용할 때 발생할 수 있는 일반적인 문제를 진단하고 해결할 수 있습니다.

목차

CloudWatch에서 대기열 또는 가상 호스트에 대한 지표를 볼 수 없습니다.

CloudWatch에서 대기열 또는 가상 호스트를 볼 수 없는 경우 대기열 또는 가상 호스트 이름에 공백, 탭 또는 기타 ASCII가 아닌 문자가 포함되어 있는지 확인합니다.

Amazon MQ에서는 이름에 공백, 탭 또는 기타 ASCII가 아닌 문자가 포함된 가상 호스트 및 대기열의 지표를 게시할 수 없습니다.

차원 이름에 대한 자세한 내용은 Amazon CloudWatch API 참조의 차원을 참조하세요.

Amazon MQ용 RabbitMQ에서 플러그인을 활성화하려면 어떻게 해야 합니까?

Amazon MQ용 RabbitMQ는 현재 기본적으로 활성화되어 있는 RabbitMQ 관리, 쇼벨, 페더레이션, 일관성 있는 해시 교환 플러그인만 지원합니다. 지원되는 플러그인 사용에 대한 자세한 내용은 the section called “플러그인” 단원을 참조하십시오.

브로커에 대한 Amazon VPC 구성을 변경할 수 없습니다.

Amazon MQ는 브로커가 생성된 후 Amazon VPC 구성 변경을 지원하지 않습니다. 새 Amazon VPC 구성을 사용하여 새 브로커를 생성하고 클라이언트 연결 URL을 새 브로커 연결 URL로 업데이트해야 합니다.

클러스터 배포에서 대기열 동기화가 일시 중지됨

RabbitMQ의 높은 메모리 사용량 경보 문제를 해결하는 중에 하나 이상의 대기열 메시지를 사용할 수 없는 문제가 발생할 수도 있습니다. 해당 대기열은 노드 간에 메시지를 동기화하는 진행하는 중일 수 있으며, 이 시간 동안 각 대기열은 게시 및 사용을 할 수 없게 됩니다. 높은 메모리 사용량 경보 때문에 대기열 동기화가 일시 중지될 수 있으며, 메모리 경보의 원인이 될 수도 있습니다.

일시 중지된 대기열 동기화 중지 및 재시도에 대한 자세한 내용은 the section called “일시 중지된 대기열 동기화 문제 해결” 섹션을 참조하세요.

RabbitMQ용 Amazon MQ 단일 인스턴스 브로커가 재시작 루프에 있습니다.

높은 메모리 사용량 경보를 발생시킨 RabbitMQ용 Amazon MQ 단일 인스턴스 브로커를 메모리가 시작하기에 충분하지 않은 상태에서 재시작하면 브로커를 사용할 수 없게 될 위험이 있습니다. 이로 인해 RabbitMQ가 재시작 루프에 들어가서 문제가 해결 될 때까지 브로커와의 추가 상호 작용이 불가능하게 될 수 있습니다. 브로커가 재시작 루프에 있는 경우 Amazon MQ 권장 모범 사례를 적용하여 높은 메모리 경보를 해결할 수 없습니다.

브로커를 복구하려면 더 많은 메모리를 가진 더 큰 인스턴스 유형으로 업그레이드하는 것이 좋습니다. 클러스터 배포에서와 달리, 단일 인스턴스 브로커는 다시 시작하는 동안 노드 간에 수행할 대기열 동기화가 없기 때문에 높은 메모리 사용량 경보가 발생한 상태에서 업그레이드할 수 있습니다.

브로커의 모든 관리자 계정에 대한 액세스 권한을 잃었습니다.

IAM 인증을 사용하여 액세스를 복구할 수 있습니다. AWS 계정에 대한 아웃바운드 웹 자격 증명 연동을 활성화하고, 웹 자격 증명 토큰을 가져올 권한이 있는 IAM 역할을 생성하고, OAuth 2.0을 통해 IAM 인증을 수락하도록 브로커를 구성한 다음, IAM 자격 증명을 사용하여 JWT 토큰을 가져오고 새 관리자

사용자를 생성합니다. 자세한 지침은 the section called “IAM 인증 및 권한 부여 사용” 섹션을 참조하세요.

Amazon MQ용 ActiveMQ: 탄력적 네트워크 인터페이스가 삭제됨 경보

Amazon MQ용 ActiveMQ는 브로커의 탄력적 네트워크 인터페이스(ENI)를 삭제하면 BROKER_ENI_DELETED 경보를 발생시킵니다. 처음으로 Amazon MQ 브로커 생성할 때 Amazon MQ는 Virtual Private Cloud(VPC)에서 사용자 계정 아래에 탄력적 네트워크 인터페이스를 프로비저닝하므로 많은 EC2 권한이 필요합니다.

이 네트워크 인터페이스는 수정하거나 삭제하면 안 됩니다. 네트워크 인터페이스를 수정하거나 삭제하면 VPC와 브로커 간의 연결이 영구적으로 손실될 수 있습니다. 네트워크 인터페이스를 삭제하려면 먼저 브로커를 삭제해야 합니다.

Amazon MQ용 ActiveMQ: 브로커 메모리 부족 경보

Amazon MQ용 ActiveMQ는 메모리 용량이 부족하여 브로커가 재시작 루프를 거칠 때 BROKER_OOM 경보를 발생시킵니다. 브로커가 바운스 루프라고도 하는 재시작 루프에 있을 때 브로커는 짧은 시간 내에 반복적인 복구 시도를 시작합니다. 메모리 용량이 부족하여 시작을 완료할 수 없는 브로커는 재시작 루프에 들어갈 수 있으며, 이 시간 동안 브로커와의 상호 작용이 제한됩니다.

Amazon MQ는 기본적으로 브로커에 대한 지표를 활성화합니다. Amazon CloudWatch 콘솔에 액세스하거나 CloudWatch API를 사용하여 브로커 지표를 볼 수 있습니다. 다음 지표는 ActiveMQ BROKER_OOM 경보를 진단할 때 유용합니다.

Amazon MQ CloudWatch 지표	메모리 사용량이 많은 이유	
`TotalMessageCount`	메시지는 사용되거나 폐기될 때까지 메모리에 저장됩니다. 메시지 수가 많으면 리소스가 과도하게 사용됨을 나타내며 높은 메모리 사용량 경보가 발생할 수 있습니다.	
`HeapUsage`	브로커가 현재 사용하는 ActiveMQ JVM 메모리 제한의	

Amazon MQ CloudWatch 지표	메모리 사용량이 많은 이유	
	비율입니다. 비율이 높을수록 브로커가 리소스를 많이 사용하고 있음을 나타내며 OOM 경보가 발생할 수 있습니다.	
`ConnectionCount`	클라이언트 연결은 메모리를 사용하며 동시 연결이 너무 많으면 높은 메모리 사용량 경보가 발생할 수 있습니다.	
`CpuUtilization`	브로커가 현재 사용하는 할당된 EC2 컴퓨팅 유닛(ECU)의 비율(%)입니다.	
`TotalConsumerCount`	브로커에 연결된 모든 소비자에 대해 설정된 수의 메시지가 소비자에게 전달되기 전에 스토리지에서 메모리로 로드됩니다. 소비자 연결 수가 많을 경우 메모리 사용량이 증가하여 높은 메모리 사용량 경보가 발생할 수 있습니다.	

재시작 루프와 BROKER_OOM 경보를 방지하려면 메시지가 빠르게 소비되도록 해야 합니다. 가장 효과적인 브로커 인스턴스 유형을 선택하고 DLQ(Dead Letter Queue)를 정리하여 전송할 수 없거나 만료된 메시지를 삭제하면 됩니다. Amazon MQ용 ActiveMQ 모범 사례에서 효과적인 성능을 유지하는 방법에 대해 자세히 알아볼 수 있습니다.

RabbitMQ용 Amazon MQ: 높은 메모리 사용량 경보

RabbitMQ용 Amazon MQ는 CloudWatch 지표 `RabbitMQMemUsed`로 식별되는 브로커의 메모리 사용량이 `RabbitMQMemLimit`으로 식별되는 메모리 제한을 초과할 때 높은 메모리 경보를 발생시킵니다.

높은 메모리 사용량 경보를 발생한 RabbitMQ 브로커는 메시지를 게시하는 모든 클라이언트를 차단합니다. 브로커가 재시작 루프에 들어가거나, 대기열 동기화가 일시 중지되거나, 경보 진단 및 해결을 복잡하게 만드는 기타 문제가 발생할 수 있습니다.

높은 메모리 경보를 진단하고 해결하려면 먼저 RabbitMQ의 모든 모범 사례를 따른 후 다음 단계를 완료합니다.

> **Important**
>
> - `RabbitMQMemLimit`은 Amazon MQ에 의해 설정되며 각 호스트 인스턴스 유형에 사용할 수 있는 메모리를 고려하여 특별히 조정됩니다. 자세한 내용은 the section called "메모리 및 디스크 경보" 단원을 참조하십시오.
> - Amazon MQ는 높은 메모리 사용량 경보가 발생하는 브로커를 다시 시작하지 않으며, 브로커가 계속해서 경보를 올리는 한 `RebootBroker` API 작업에 대한 예외를 반환합니다

1단계: 높은 메모리 경보 진단

RabbitMQ용 Amazon MQ 브로커에서 높은 메모리 경보를 진단하는 방법에는 두 가지가 있습니다. CloudWatch에서 RabbitMQ 웹 콘솔과 Amazon MQ 지표를 모두 확인하는 것이 좋습니다.

RabbitMQ 웹 콘솔을 사용하여 높은 메모리 사용량 경보 진단

RabbitMQ 웹 콘솔은 각 노드에 대한 자세한 메모리 사용 정보를 생성하고 표시할 수 있습니다. 다음을 수행하여 이 정보를 확인할 수 있습니다.

1. 에 로그인 AWS Management Console 하고 브로커의 RabbitMQ 웹 콘솔을 엽니다.
2. RabbitMQ 콘솔에 있는 개요(Overview) 페이지의 노드(Nodes) 목록에서 노드 이름을 선택합니다.
3. 노드 세부 정보 페이지에서 메모리 세부 정보(Memory details)를 선택하고 섹션을 확장하여 노드의 메모리 사용 정보를 볼 수 있습니다.

RabbitMQ가 웹 콘솔에서 제공하는 메모리 사용 정보는 어느 리소스가 메모리를 많이 소비하고 있고 높은 메모리 사용량 경보의 원인일지를 알아내는 데 도움이 될 수 있습니다. RabbitMQ 웹 콘솔을 통해 사용할 수 있는 메모리 사용량 세부 정보에 대한 자세한 내용은 RabbitMQ 서버 문서 웹 사이트의 메모리 사용에 관한 추론을 참조하세요.

Amazon MQ 지표를 사용한 높은 메모리 사용량 경보 진단

Amazon MQ는 기본적으로 브로커에 대한 지표를 활성화합니다. CloudWatch 콘솔에 액세스하거나 CloudWatch API를 사용하여 브로커 지표를 볼 수 있습니다. 다음 지표는 RabbitMQ 높은 메모리 사용량 경보를 진단할 때 유용합니다.

Amazon MQ CloudWatch 지표	메모리 사용량이 많은 이유
MessageCount	메시지는 사용되거나 폐기될 때까지 메모리에 저장됩니다. 메시지 수가 많으면 리소스가 과도하게 사용됨을 나타내며 높은 메모리 사용량 경보가 발생할 수 있습니다.
QueueCount	대기열은 메모리에 저장되며 대기열 수가 많으면 높은 메모리 사용량 경보가 발생할 수 있습니다.
ConnectionCount	클라이언트 연결은 메모리를 사용하며 동시 연결이 너무 많으면 높은 메모리 사용량 경보가 발생할 수 있습니다.
ChannelCount	연결과 마찬가지로, 각 연결을 사용하여 설정된 채널도 노드 메모리에 저장되며 채널 수가 많으면 높은 메모리 사용량 경보가 발생할 수 있습니다.
ConsumerCount	브로커에 연결된 모든 소비자에 대해 설정된 수의 메시지가 소비자에게 전달되기 전에 스토리지에서 메모리로 로드됩니다. 소비자 연결 수가 많을 경우 메모리 사용량이 증가하여 높은 메모리 사용량 경보가 발생할 수 있습니다.
PublishRate	메시지 게시는 브로커 메모리를 사용합니다. 브로커에 메시지가 게시되는 비율이 너무 높고 브로커가 소비자에게 메시

Amazon MQ CloudWatch 지표	메모리 사용량이 많은 이유	
	지를 전달하는 비율을 크게 능가하는 경우, 브로커가 높은 메모리 사용량 경보를 발생시킬 수 있습니다.	

2단계: 높은 메모리 경보 해결 및 방지

> **Note**
>
> 필요한 조치를 취한 후 RABBITMQ_MEMORY_ALARM 상태가 지워지는 데 최대 몇 시간이 걸릴 수 있습니다.

일반적인 예방 방법으로 RabbitMQ에 대한 모든 모범 사례를 따릅니다. RabbitMQ의 높은 메모리 사용량 경보 발생을 해결하고 방지하려면 식별된 원인에 따라 아래와 같은 조치를 취하는 것이 좋습니다.

메모리 사용량이 많은 소스	해결을 위한 Amazon MQ 권장 사항	방지를 위한 Amazon MQ 권장 사항
메시지 수	대기열에 게시된 메시지를 사용하거나, 대기열에서 메시지를 제거하거나, 브로커에서 대기열을 삭제합니다.	지연 대기열을 활성화하고 대기열 깊이 제한을 설정하거나 줄입니다.
대기열 수	대기열 수를 줄입니다.	대기열 수 제한을 설정 또는 축소합니다.
연결 수	연결 수를 줄입니다.	연결 수 제한을 설정 또는 축소합니다.
채널 수	채널 수를 줄입니다.	클라이언트 애플리케이션에서 연결당 최대 채널 수를 설정합니다.

메모리 사용량이 많은 소스	해결을 위한 Amazon MQ 권장 사항	방지를 위한 Amazon MQ 권장 사항
소비자 수	브로커에 연결된 소비자 수를 줄입니다.	소규모 소비자에 미리 가져오기 제한을 설정합니다.
메시지 게시 속도	게시자가 브로커에 보내는 메시지 빈도를 줄입니다.	게시자 확인을 켭니다.
클라이언트 연결 시도 속도	메시지를 게시 또는 소비하거나 브로커를 구성하기 위해 클라이언트가 브로커에 연결을 시도하는 빈도를 줄입니다.	연결 시도 횟수와 빈도를 줄이려면 수명이 긴 연결을 사용합니다.

브로커의 메모리 경보가 해결되면 호스트 인스턴스 유형을 추가 리소스가 있는 인스턴스로 업그레이드할 수 있습니다. 브로커 인스턴스 유형을 업데이트하는 방법에 대한 자세한 내용은 Amazon MQ REST API 참조의 `UpdateBrokerInput` 섹션을 참조하세요.

> **Note**
>
> `mq.m5.x` 인스턴스 유형에서 `mq.t3.micro` 인스턴스 유형으로 브로커를 다운그레이드할 수는 없습니다. 다운그레이드하려면 브로커를 삭제하고 새 브로커를 생성해야 합니다.

Amazon MQ의 RabbitMQ: 잘못된 AWS Key Management Service 키 Amazon MQ

Amazon MQ의 RabbitMQ는 고객 관리형 AWS KMS key(CMK)으로 생성된 브로커가 AWS Key Management Service (KMS) 키가 비활성화된 것을 감지하면 INVALID_KMS_KEY 중요 작업 필수 코드를 생성합니다. Amazon MQ CMK를 사용하는 RabbitMQ 브로커는 KMS 키가 활성화되어 있고 브로커에 필요한 모든 권한이 있는지 정기적으로 확인합니다. RabbitMQ에서 키가 활성화되었는지 확인할 수 없는 경우 브로커는 격리되고 RabbitMQ는 INVALID_KMS_KEY를 반환합니다.

활성 KMS 키가 없으면 브로커는 고객 관리형 KMS 키에 대한 기본 권한을 갖지 않습니다. 사용자가 키를 다시 활성화하고 브로커가 다시 시작되기 전까지는 브로커가 키를 사용하여 암호화 작업을 수행할 수 없습니다. KMS 키가 비활성화된 RabbitMQ 브로커는 성능 저하를 방지하기 위해 격리됩니다. RabbitMQ에서 KMS 키가 다시 활성화된 것을 확인하면 브로커가 격리 해제됩니다. Amazon MQ는

KMS 키가 비활성화된 브로커를 다시 시작하지 않으며, 브로커가 계속해서 잘못된 KMS 키를 사용하는 한 `RebootBroker` API 작업에 대한 예외를 반환합니다.

INVALID_KMS_KEY 진단 및 해결

INVALID_KMS_KEY 작업 필수 코드를 진단하고 해결하려면 AWS 명령줄 인터페이스(CLI)와 AWS Key Management Service 콘솔을 사용해야 합니다.

KMS 키를 다시 활성화하는 방법

1. `DescribeBroker` 메서드를 호출하여 CMK 브로커의 `kmsKeyId`를 검색합니다.
2. AWS Key Management Service 콘솔에 로그인합니다.
3. 고객 관리형 키 페이지에서 문제가 있는 브로커의 KMS 키 ID를 찾아 상태가 활성인지 확인합니다.
4. KMS 키가 비활성화된 경우 키 작업을 선택한 다음 활성화를 선택하여 키를 다시 활성화합니다. 키를 다시 활성화한 후에는 RabbitMQ가 브로커를 격리 해제할 때까지 기다려야 합니다.

필요한 권한이 브로커의 KMS 키와 여전히 연결되어 있는지 확인하려면 `ListGrantListGrant` 메서드를 호출하여 `mq_rabbit_grant` 및 `mq_grant`가 존재하는지 확인합니다. KMS 권한 또는 키가 삭제된 경우 브로커를 삭제하고 필요한 모든 권한이 포함된 새 브로커를 생성해야 합니다. 브로커 삭제 단계는 브로커 삭제를 참조하세요.

INVALID_KMS_KEY 중요 작업 필요 코드를 방지하려면 KMS 키 또는 CMK 권한을 수동으로 삭제하거나 비활성화하지 마세요. 키를 삭제하려면 먼저 브로커를 삭제하세요.

Amazon MQ의 RabbitMQ: 디스크 제한 경보

디스크 제한 경보는 새 메시지가 추가되는 동안 소비되지 않은 메시지 수가 많아 RabbitMQ 노드에서 사용하는 디스크 볼륨이 감소했음을 나타냅니다. RabbitMQ는 브로커의 사용 가능한 디스크 공간(Amazon CloudWatch 지표 `RabbitMQDiskFree`로 식별)이 디스크 제한(`RabbitMQDiskFreeLimit`로 식별)에 도달하면 디스크 제한 경보를 발생시킵니다. `RabbitMQDiskFreeLimit`는 Amazon MQ에서 설정되며 각 브로커 인스턴스 유형에 사용할 수 있는 디스크 공간을 고려하여 정의되었습니다. 자세한 내용은 the section called “메모리 및 디스크 경보” 단원을 참조하십시오.

디스크 한도 경보가 발생한 Amazon MQ 기반 RabbitMQ 브로커는 새 메시지를 게시하는 데 사용할 수 없게 됩니다. 게시자와 소비자가 같은 연결에 있는 경우 소비자도 메시지를 받을 수 없게 됩니다. 클러

스터에서 RabbitMQ를 실행하는 경우 디스크 경보는 클러스터 전체에서 발생합니다. 한 노드가 제한 미만으로 떨어지면 다른 모든 노드는 들어오는 메시지를 차단합니다. 디스크 공간이 부족하기 때문에 브로커가 경보 진단 및 해결을 복잡하게 만드는 다른 문제가 발생할 수도 있습니다.

Amazon MQ는 높은 디스크 경보가 발생하는 브로커를 다시 시작하지 않으며, 브로커가 계속해서 경보를 울리는 한 `RebootBroker` API 작업에 대한 예외를 반환합니다

> **Note**
>
> `mq.m5` 인스턴스 유형에서 `mq.t3.micro` 인스턴스 유형으로 브로커를 다운그레이드할 수는 없습니다. 브로커를 다운그레이드하려면 브로커를 삭제하고 새 브로커를 생성해야 합니다.

디스크 제한 경보 진단 및 해결

Amazon MQ는 기본적으로 브로커에 대한 지표를 활성화합니다. Amazon CloudWatch 콘솔에 액세스하거나 CloudWatch API를 사용하여 브로커 지표를 볼 수 있습니다. `MessageCount`는 RabbitMQ 디스크 제한 경보를 진단할 때 유용한 지표입니다. 메시지는 사용되거나 폐기될 때까지 메모리에 저장됩니다. 메시지 수가 많으면 디스크 스토리지가 과도하게 사용됨을 나타내며 디스크 경보가 발생할 수 있습니다.

디스크 제한 경보를 진단하려면 Amazon MQ 관리 콘솔을 사용하여 다음을 수행합니다.

- 대기열에 게시된 메시지를 처리할 새 연결을 생성합니다.
- 대기열에서 메시지를 삭제합니다.
- 브로커에서 대기열을 삭제합니다.

> **Note**
>
> 필요한 조치를 취한 후 RABBITMQ_DISK_ALARM 상태가 지워지는 데 최대 몇 시간이 걸릴 수 있습니다.

디스크 제한 경보가 다시 발생하는 것을 방지하기 위해 호스트 인스턴스 유형을 추가 리소스가 있는 인스턴스로 업그레이드할 수 있습니다. 브로커 인스턴스 유형을 업데이트하는 방법에 대한 자세한 내용은 Amazon MQ REST API 참조의 `UpdateBrokerInput` 섹션을 참조하세요. 또한 게시자와 소비자를 서로 다른 연결로 유지하는 것이 좋습니다.

RabbitMQ용 Amazon MQ 인스턴스 유형 변경 경보

RABBITMQ_CLUSTER_DISK_USAGE_TOO_HIGH_FOR_INSTANCE_CHANGE는 현재 RabbitMQ 노드의 디스크 사용량이 높아 요청된 브로커 인스턴스 유형 변경을 진행할 수 없음을 나타냅니다. RabbitMQ용 Amazon MQ는 현재 디스크 사용량이 CloudWatch 지표 RabbitMQDiskFree로 식별되는 요청된 인스턴스 유형에서 사용할 수 있는 양을 초과할 때 이 경보를 발생시킵니다.

RABBITMQ_CLUSTER_DISK_USAGE_TOO_HIGH_FOR_INSTANCE_CHANGE 상태로 전환되는 RabbitMQ 브로커는 애플리케이션에서 계속 사용할 수 있지만 요청된 인스턴스 유형 변경은 진행되지 않습니다. Amazon MQ는 이 상태에서 브로커 재시작을 허용하지만 디스크 사용량이 요청된 인스턴스 유형의 임계값을 초과하는 동안에는 인스턴스 유형을 변경할 수 없습니다. 브로커는 이 상태에 있는 동안 인스턴스 유형 변경을 시도하는 ModifyBroker API 작업에 대한 예외를 반환합니다.

인스턴스 유형 변경 경보 진단 및 해결

Amazon MQ는 기본적으로 브로커에 대한 지표를 활성화합니다. CloudWatch 콘솔에 액세스하거나 CloudWatch API를 사용하여 브로커 지표를 볼 수 있습니다. MessageCount 및 RabbitMQDiskFree 지표를 사용하여 RABBITMQ_CLUSTER_DISK_USAGE_TOO_HIGH_FOR_INSTANCE_CHANGE를 진단할 수 있습니다.

격리 상태를 해결하고 인스턴스 유형 변경을 진행하려면 Amazon MQ 관리 콘솔을 사용하여 다음을 수행합니다.

- 대기열에 게시된 메시지를 처리할 새 연결을 생성합니다.
- 대기열에서 메시지를 삭제합니다.
- 브로커에서 대기열을 삭제합니다.

> **Note**
>
> 필요한 조치를 취한 후 RABBITMQ_CLUSTER_DISK_USAGE_TOO_HIGH_FOR_INSTANCE_CHANGE 상태가 지워지는 데 최대 몇 시간이 걸릴 수 있습니다.

Amazon MQ의 RabbitMQ: 잘못된 IAM 수임 역할 Amazon MQ

Amazon MQ의 RabbitMQ는에 지정된 IAM 역할 ARNaws.arns.assume_role_arn이 유효하지 않거나 Amazon MQ에서 수임할 수 없는 경우 INVALID_ASSUMEROLE 중요 작업 필수 코드

를 생성합니다. Amazon MQ 이는 역할이 존재하지 않거나, 브로커와 다른 AWS 계정에 있거나, mq.amazonaws.com 필요한 신뢰 관계가 없는 경우에 발생할 수 있습니다.

RABBITMQ_INVALID_ASSUMEROLE 격리의 브로커는 LDAP 인증에 필요한 자격 증명 또는 인증서를 검색할 수 없으므로 LDAP 인증을 사용할 수 없게 됩니다. LDAP가 유일하게 구성된 인증 방법인 경우 사용자는 브로커에 연결할 수 없습니다. IAM 역할은 Amazon MQ에서 LDAP 인증에 사용되는 AWS Secrets Manager 보안 암호 또는 Amazon S3 객체와 같은 브로커 구성의 ARNs에서 참조하는 AWS 리소스에 액세스하는 데 필요합니다.

RABBITMQ_INVALID_ASSUMEROLE 진단 및 해결

RABBITMQ_INVALID_ASSUMEROLE 작업 필수 코드를 진단하고 해결하려면 Amazon CloudWatch Logs 및 AWS Identity and Access Management 콘솔을 사용해야 합니다.

잘못된 수임 역할 문제를 해결하려면

1. Amazon CloudWatch Logs Insights로 이동하여 브로커의 로그 그룹에 대해 다음 쿼리를 실행합니다. `/aws/amazonmq/broker/<broker-id>/general`

```
fields @timestamp, @message
| sort @timestamp desc
| filter @message like /error.*aws_arn_config/
| limit 10000
```

2. 다음과 유사한 오류 메시지를 찾습니다.

```
[error] <0.254.0> aws_arn_config: {handle_assume_role,{error,
{assume_role_failed,"AWS service is unavailable"}}}
```

3. IAM 역할 구성을 확인하고 다음과 같은 문제를 해결합니다.
   - 역할이 브로커와 동일한 AWS 계정에 있는지 확인합니다.
   - 신뢰 정책에서 mq.amazonaws.com 역할 수임이 허용되는지 확인
   - 역할에 필요한 AWS 리소스에 액세스할 수 있는 적절한 권한이 있는지 확인합니다.
4. 브로커 구성을 업데이트하기 전에 ARN 액세스 검증 API 엔드포인트를 사용하여 수정 사항을 검증합니다.

5. 브로커 구성을 업데이트하고 브로커를 재부팅합니다.

Amazon MQ의 RabbitMQ: 잘못된 LDAP ARN Amazon MQ

Amazon MQ의 RabbitMQ는 LDAP 서비스 계정 암호에 대해 구성된 ARN이 유효하지 않거나 액세스할 수 없는 경우 INVALID_ARN_LDAP 중요 작업 필수 코드를 생성합니다. Amazon MQ 이는 일반 텍스트 암호가 포함된 AWS Secrets Manager 암호를 참조해야 `aws.arns.auth_ldap.other_bind.password`하는 `aws.arns.auth_ldap.dn_lookup_bind.password` 또는에 지정된 ARNs에 적용됩니다.

RABBITMQ_INVALID_ARN_LDAP 격리의 브로커는 LDAP 서비스 계정으로 인증할 수 없으므로 LDAP 인증을 사용할 수 없습니다. LDAP가 유일하게 구성된 인증 방법인 경우 사용자는 브로커에 연결할 수 없습니다. 잘못된 ARNs 구문, 존재하지 않는 보안 암호에 대한 참조, 브로커와 다른 AWS 리전에 있는 보안 암호 또는 IAM 역할의 Secretsmanager:GetSecretValue 권한 부족으로 인해 잘못된 ARN이 발생할 수 있습니다.

RABBITMQ_INVALID_ARN_LDAP 진단 및 해결

RABBITMQ_INVALID_ARN_LDAP 작업 필수 코드를 진단하고 해결하려면 Amazon CloudWatch Logs 및 콘솔을 사용해야 합니다.

잘못된 LDAP ARN 문제를 해결하려면

1. Amazon CloudWatch Logs Insights로 이동하여 브로커의 로그 그룹에 대해 다음 쿼리를 실행합니다. `/aws/amazonmq/broker/<broker-id>/general`

```
fields @timestamp, @message
| sort @timestamp desc
| filter @message like /error.*aws_arn_config/
| limit 10000
```

2. 다음과 유사한 오류 메시지를 찾습니다.

```
[error] <0.254.0> aws_arn_config: {<<"could not resolve
 ARN 'arn:aws:secretsmanager:xxx' for configuration
 'aws.arns.auth_ldap.dn_lookup_bind.password', error: \"AWS service is unavailable
\"">>,{error,"AWS service is unavailable"}}
```

3. Secrets Manager 보안 암호를 확인하고 다음과 같은 문제를 해결합니다.
   - 보안 암호가 브로커와 동일한 AWS 리전에 있는지 확인
   - ARN 구문이 올바른지 확인
   - IAM 역할에 secretsmanager:GetSecretValue 권한이 있는지 확인합니다.
4. 브로커 구성을 업데이트하기 전에 ARN 액세스 검증 API 엔드포인트를 사용하여 수정 사항을 검증합니다.
5. 브로커 구성을 업데이트하고 브로커를 재부팅합니다.

Amazon MQ의 RabbitMQ: 잘못된 HTTP ARN Amazon MQ

Amazon MQ의 RabbitMQ는 HTTP auth_backend에 대한 SSL 인증서 또는 키 파일의 ARNs이 하나 이상 유효하지 않거나 액세스할 수 없는 경우 INVALID_ARN_HTTP 중요 작업 필수 코드를 발생시킵니다. Amazon MQ 이는 인증서 및 프라이빗 키가 포함된 Amazon S3 객체 및 AWS Secrets Manager 보안 암호를 참조해야 `aws.arns.auth_http.ssl_options.keyfile`하는 `aws.arns.auth_http.ssl_options.cacertfile` `aws.arns.auth_http.ssl_options.certfile` 또는에 지정된 ARNS에 적용됩니다.

RABBITMQ_INVALID_ARN_HTTP 격리의 브로커는 HTTP 서버를 통해 인증할 수 없습니다. HTTP가 유일하게 구성된 인증 방법인 경우 사용자는 브로커에 연결할 수 없습니다. 잘못된 ARNs 구문, 존재하지 않는 보안 암호에 대한 참조, 브로커와 다른 AWS 리전에 있는 보안 암호 또는 IAM 역할의 s3:GetObject/secretsmanager:GetSecretValue 권한 부족으로 인해 잘못된 ARN이 발생할 수 있습니다.

RABBITMQ_INVALID_ARN_HTTP 진단 및 해결

RABBITMQ_INVALID_ARN_HTTP 작업 필수 코드를 진단하고 해결하려면 Amazon CloudWatch Logs 및 콘솔을 사용해야 합니다.

잘못된 HTTP ARN 문제를 해결하려면

1. Amazon CloudWatch Logs Insights로 이동하여 브로커의 로그 그룹에 대해 다음 쿼리를 실행합니다. `/aws/amazonmq/broker/<broker-id>/general`

```
fields @timestamp, @message
```

```
| sort @timestamp desc
| filter @message like /error.*aws_arn_config/
| limit 10000
```

2. 다음과 유사한 오류 메시지를 찾습니다.

```
[error] <0.209.0> aws_arn_config: {<<"could not resolve ARN 'arn:aws:s3:::xxxx' for
 configuration 'aws.arns.auth_http.ssl_options.certfile', error: \"AWS service is
 unavailable\"">>,{error,"AWS service is unavailable"}}
```

3. S3 Object/Secrets Manager 보안 암호를 확인하고 다음과 같은 문제를 해결합니다.
   - 리소스가 브로커와 동일한 AWS 리전에 있는지 확인
   - ARN 구문이 올바른지 확인
   - IAM 역할에 s3:GetObject 및 secretsmanager:GetSecretValue 권한이 있는지 확인합니다.
4. 브로커 구성을 업데이트하기 전에 ARN 액세스 검증 API 엔드포인트를 사용하여 수정 사항을 검증합니다.
5. 브로커 구성을 업데이트하고 브로커를 재부팅합니다.

Amazon MQ의 RabbitMQ: 잘못된 SSL ARN Amazon MQ

Amazon MQ의 RabbitMQ는 EXTERNAL auth_mechanism용 CA 인증서 트러스트 스토어의 ARNs 이 하나 이상 유효하지 않거나 액세스할 수 없는 경우 INVALID_ARN_SSL 중요 작업 필수 코드를 생성합니다. Amazon MQ 이는 인증서가 포함된 Amazon S3 또는 ACM PCA 객체를 참조해야 `aws.arns.management.ssl.cacertfile`하는 `aws.arns.ssl_options.cacertfile` 또는에 지정된 ARNS에 적용됩니다.

RABBITMQ_INVALID_ARN_SSL 격리의 브로커는 유효한 트러스트 스토어가 구성되지 않았기 때문에 상호 TLS 핸드셰이크 중에 클라이언트 인증서를 인증할 수 없습니다. EXTERNAL 인증 메커니즘이 유일하게 구성된 인증 방법인 경우 사용자는 브로커에 연결할 수 없습니다. 잘못된 ARN 구문, 존재하지 않는 S3 객체에 대한 참조, 브로커와 다른 AWS 리전에 있는 S3 객체 또는 IAM 역할의 s3:GetObject/acm-pca:GetCertificateAuthorityCertificate 권한이 부족하여 잘못된 ARN이 발생할 ARNs 수 있습니다.

RABBITMQ_INVALID_ARN_SSL 진단 및 해결

RABBITMQ_INVALID_ARN_SSL 작업 필수 코드를 진단하고 해결하려면 Amazon CloudWatch Logs 및 콘솔을 사용해야 합니다.

잘못된 SSL ARN 문제를 해결하려면

1. Amazon CloudWatch Logs Insights로 이동하여 브로커의 로그 그룹에 대해 다음 쿼리를 실행합니다. `/aws/amazonmq/broker/<broker-id>/general`

```
fields @timestamp, @message
| sort @timestamp desc
| filter @message like /error.*aws_arn_config/
| limit 10000
```

2. 다음과 유사한 오류 메시지를 찾습니다.

```
[error] <0.209.0> aws_arn_config: {<<"could not resolve ARN 'arn:aws:acm-pca:xxxx'
 for configuration 'aws.arns.ssl_options.cacertfile', error: \"AWS service is
 unavailable\"">>,{error,"AWS service is unavailable"}}
```

3. S3/ACM-PCA 객체를 확인하고 다음과 같은 문제를 해결합니다.
   - 보안 암호가 브로커와 동일한 AWS 리전에 있는지 확인
   - ARN 구문이 올바른지 확인
   - IAM 역할에 s3:GetObject/acm-pca:GetCertificateAuthorityCertificate 권한이 있는지 확인합니다.
4. 브로커 구성을 업데이트하기 전에 ARN 액세스 검증 API 엔드포인트를 사용하여 수정 사항을 검증합니다.
5. 브로커 구성을 업데이트하고 브로커를 재부팅합니다.

Amazon MQ의 RabbitMQ: 잘못된 ARN Amazon MQ

브로커에 구성된 하나 이상의 ARN이 유효하지 않거나 액세스할 수 없는 경우 Amazon MQ의 RabbitMQ는 INVALID_ARN 중요 작업 필수 코드를 생성합니다. Amazon MQ ARNs 이는 SSL 인증

서, AWS Secrets Manager 보안 암호, Amazon S3 객체 또는 RABBITMQ_INVALID_ARN_LDAP 또는 RABBITMQ_INVALID_ASSUMEROLE과 같은 보다 구체적인 격리 코드에서 다루지 않는 기타 AWS 리소스 참조에 사용되는 ARNs에 적용됩니다.

RABBITMQ_INVALID_ARN 격리의 브로커는 유효하지 않은 ARNs에 따라 기능이 저하될 수 있습니다. 액세스할 수 없는 리소스에 의존하는 기능은 사용할 수 없으며 브로커는 해결에 실패한 ARN을 나타내는 오류를 기록합니다. 브로커 가용성에 미치는 영향은 중요한 브로커 작업에 잘못된 ARN이 필요한지 여부에 따라 달라집니다.

RABBITMQ_INVALID_ARN 진단 및 해결

RABBITMQ_INVALID_ARN 작업 필수 코드를 진단하고 해결하려면 Amazon CloudWatch Logs와 영향을 받는 리소스에 적합한 AWS 서비스 콘솔을 사용해야 합니다.

잘못된 ARN 문제를 해결하려면

1. Amazon CloudWatch Logs Insights로 이동하여 브로커의 로그 그룹에 대해 다음 쿼리를 실행합니다. `/aws/amazonmq/broker/<broker-id>/general`

```
fields @timestamp, @message
| sort @timestamp desc
| filter @message like /error.*aws_arn_config/
| limit 10000
```

2. 다음과 유사한 오류 메시지를 찾습니다.

```
[error] <0.254.0> aws_arn_config: {<<"could not resolve ARN
 'arn:aws:s3:::bucket-name/certificate.pem' for configuration
 'aws.arns.auth_ldap.ssl_options.cacertfile', error: \"AWS service is unavailable
\"">>,{error,"AWS service is unavailable"}}
```

3. AWS 리소스를 확인하고 다음과 같은 문제를 해결합니다.

   - 리소스가 브로커와 동일한 AWS 리전에 있는지 확인
   - ARN 구문이 올바른지 확인
   - IAM 역할에 리소스에 액세스할 수 있는 적절한 권한이 있는지 확인합니다.

4. 브로커 구성을 업데이트하기 전에 ARN 액세스 검증 API 엔드포인트를 사용하여 수정 사항을 검증합니다.
5. 브로커 구성을 업데이트하고 브로커를 재부팅합니다.

RabbitMQ용 Amazon MQ: 버전 4로 업그레이드할 수 없는 브로커 RabbitMQ

RabbitMQ용 Amazon MQ는 RabbitMQ 3 브로커를 RabbitMQ 4로 업그레이드하려고 하고 브로커에 클래식 대기열이 있거나 Khepri 메타데이터 스토어 기능 플래그가 활성화된 경우 `RABBITMQ_BROKER_NOT_UPGRADEABLE_TO_V4` 작업 필수 코드를 생성합니다. RabbitMQ Amazon MQ는 메이저 버전 업그레이드를 적용하지 않으며 브로커를 게시하고 사용할 수 있도록 합니다.

이 작업 필수 코드는 RabbitMQ 3 브로커에만 적용됩니다. 이 상태를 해결하고 업그레이드를 계속하려면 다음 단계를 완료하세요.

RABBITMQ_BROKER_NOT_UPGRADEABLE_TO_V4 진단 및 해결

1. Amazon MQ 대기열 마이그레이션 도구를 사용하여 모든 클래식 대기열을 쿼럼 대기열로 마이그레이션합니다. 이 도구는 RabbitMQ 웹 콘솔(관리자 > 대기열 마이그레이션) 또는 HTTP API를 통해 액세스할 수 있습니다.
2. 브로커에서 Khepri가 활성화된 경우 RabbitMQ 4에 대한 현재 위치 업그레이드 경로가 없습니다. 대신 RabbitMQ 블루-그린 배포를 고려해 보세요.

기본 문제를 해결하면 Amazon MQ가 자동으로 `CRITICAL_ACTION_REQUIRED` 상태를 지웁니다.

> **Note**
>
> UpdateBroker API 작업을 사용하여 브로커 엔진 버전을 3.13으로 다시 업데이트하여 `CRITICAL_ACTION_REQUIRED` 상태를 지울 수 있습니다.

관련 리소스

Amazon MQ 리소스

다음 표에는 Amazon MQ를 이용할 때 유용한 리소스가 나와 있습니다.

리소스	설명
Amazon MQ REST API 참조	REST 리소스, 예제 요청, HTTP 메서드, 스키마, 파라미터 및 서비스에서 반환하는 오류에 대한 설명입니다.
AWS CLI 명령 참조의 Amazon MQ	메시지 브로커 작업에 사용할 수 있는 AWS CLI 명령에 대한 설명입니다.
AWS CloudFormation 사용 설명서의 Amazon MQ	`AWS::Amazon MQ::Broker` 리소스를 사용하면 Amazon MQ 브로커를 생성하고 구성 변경 사항을 추가하거나 지정된 브로커에 대한 사용자를 수정하고 지정된 브로커에 대한 정보를 반환하며 지정된 브로커를 삭제할 수 있습니다. `AWS::Amazon MQ::Configuration` 리소스를 사용하면 Amazon MQ 구성을 생성하고 구성 변경 사항을 추가하거나 사용자를 수정하고 지정된 구성에 대한 정보를 반환할 수 있습니다.
지역 및 엔드포인트	Amazon MQ 리전 및 엔드포인트에 대한 정보
제품 페이지	Amazon MQ에 대한 정보를 제공하는 기본 웹 페이지입니다.
토론 포럼	Amazon MQ와 관련된 기술적 질문에 대해 토론할 수 있는 개발자를 위한 커뮤니티 기반 포럼입니다.
AWS 프리미엄 지원 정보	AWS Premium Support에 대한 정보를 제공하는 기본 웹 페이지입니다. 일대일의 신속한 응대 지

리소스	설명
	원 채널을 통해 AWS 인프라 서비스에서 애플리케이션을 구축하고 실행하도록 지원합니다.

ActiveMQ용 Amazon MQ 리소스

다음 표에는 Apache ActiveMQ를 이용할 때 유용한 리소스가 나와 있습니다.

리소스	설명
Apache ActiveMQ 시작 가이드	Apache ActiveMQ의 공식 설명서입니다.
ActiveMQ의 실제 동작	JMS 메시지, 커넥터, 메시지 지속성, 인증 및 권한 부여의 구조를 설명하는 Apache ActiveMQ에 대한 가이드입니다.
언어 간 클라이언트	프로그래밍 언어 및 해당하는 Apache ActiveMQ 라이브러리의 목록입니다. ActiveMQ 클라이언트 및 QpidJMS 클라이언트도 참조하세요.

RabbitMQ용 Amazon MQ 리소스

다음 표에는 RabbitMQ를 이용할 때 유용한 리소스가 나와 있습니다.

리소스	설명
RabbitMQ 시작 가이드	RabbitMQ에 대한 공식 설명서입니다.
RabbitMQ 클라이언트 라이브러리 및 개발자 도구	다양한 프로그래밍 언어와 플랫폼을 사용한 RabbitMQ 작업을 위해 공식적으로 지원되는 클라이언트 라이브러리 및 개발자 도구에 대한 가이드입니다.
RabbitMQ 모범 사례	RabbitMQ 작업을 위한 모범 사례 및 권장 사항에 대한 CloudAMQP의 가이드입니다.

Amazon MQ 릴리스 정보

다음 표에는 Amazon MQ 기능 릴리스 및 개선 사항이 나열되어 있습니다.

Date	설명서 업데이트
2026년 5월 5일	Amazon MQ는 이제 RabbitMQ 3.13에서 RabbitMQ 4.2로 인플레이스 메이저 버전 업그레이드를 지원합니다. 브로커 엔드포인트는 동일하게 유지되므로 애플리케이션 코드를 변경할 필요가 없습니다. 업그레이드 중에 Amazon MQ는 안전한 업그레이드를 수행하기 위해 모든 연결을 차단하여 브로커 가동 중지 시간을 초래합니다. 자세한 내용은 다음 섹션을 참조하세요. • RabbitMQ 3.x에서 4.x로 업그레이드
2026년 4월 30일	RabbitMQ용 Amazon MQ는 이제 RabbitMQ 4.2 이상을 실행하는 브로커에 대한 Prometheus 지표를 지원하므로 기존 Prometheus 기반 모니터링 인프라에 브로커 관찰성을 통합할 수 있습니다. `/metrics`, `/metrics/detailed` 및 엔드포인트가 지원됩니다`/metrics/memory-breakdown` . 자세한 내용은 RabbitMQ용 Amazon MQ 브로커에 대한 Prometheus 지표 및 사용 가능한 CloudWatch 지표 액세스를 참조하세요. CloudWatch Amazon MQ RabbitMQ
2026년 2월 19일	Amazon MQ Amazon MQ는 이제 새로운 마이너 엔진 버전 릴리스인 ActiveMQ 5.19를 지원합니다. 자세한 내용은 다음 섹션을 참조하세요. • ActiveMQ 5.19 릴리스 페이지 • ActiveMQ용 Amazon MQ 엔진 버전 관리 • Amazon MQ 브로커 엔진 버전 업그레이드 • Spring XML 구성 파일 사용
2026년 1월 22일	Amazon MQ는 이제 RabbitMQ 4.2 이상에서 브로커에 대한 JMS 주제 교환 플러그인을 지원합니다. 공식 RabbitMQ JMS 클라이언트를 사용하여 RabbitMQ

Date	설명서 업데이트
	용 Amazon MQ 브로커에서 JMS 워크로드를 실행할 수 있습니다. JMS 1.1, 2.0 및 3.1을 지원합니다. 자세한 내용은 다음 섹션을 참조하세요. • 공식 JMS 2.0 사양(및 확장 JMS 1.1과 백워드 호환) • 공식 JMS 3.1 사양 • RabbitMQ JMS 클라이언트의 제한 • JMS 애플리케이션을 RabbitMQ용 Amazon MQ 브로커에 연결
2026년 1월 8일	Amazon MQ는 이제 X.509 클라이언트 인증서 및 상호 TLS(mTLS) 구성을 사용하여 RabbitMQ 4.2 이상의 브로커에 대한 SSL 인증서 인증을 지원합니다. Amazon MQ를 사용할 수 AWS 리전 있는 모든 AWS CDK 에서 또는 AWS Management Console AWS CloudFormation AWS CLI를 통해 SSL 인증서 인증 및 mTLS를 구성할 수 있습니다. 자세한 내용은 SSL 인증서 인증 및 mTLS 구성을 참조하세요.
2026년 1월 6일	Amazon MQ는 이제 외부 HTTP 서버가 있는 RabbitMQ 4.2 이상에서 브로커에 대한 HTTP 인증 및 권한 부여를 지원합니다. Amazon MQ를 사용할 수 AWS 리전 있는 모든 AWS CDK 에서 또는 AWS Management Console AWS CloudFormation AWS CLI를 통해 HTTP 인증을 구성할 수 있습니다. 자세한 내용은 HTTP 인증 및 권한 부여를 참조하세요.

Date	설명서 업데이트
2025년 11월 20일	Amazon MQ는 이제 AMQP 1.0 프로토콜에 대한 기본 지원을 도입하는 새로운 메이저 버전 릴리스인 RabbitMQ 4.2, 새로운 Raft 기반 메타데이터 스토어 Khepri, 로컬 셔블 및 쿼럼 대기열에 대한 메시지 우선 순위를 지원합니다. RabbitMQ 4.2에는 처리량 및 메모리 관리를 위한 다양한 버그 수정 및 성능 개선도 포함되어 있습니다. 이 버전에는 새로운 기능이 도입되었지만 몇 가지 주요 변경 사항이 있습니다. 자세한 내용은 다음 섹션을 참조하세요. • RabbitMQ 4 • 오픈 소스 RabbitMQ 릴리스 정보 • 리소스 제한 구성 • 지원되는 프로토콜 • Amazon MQ 버전 업그레이드
2025년 11월 18일	Amazon MQ는 이제 RabbitMQ용 Graviton3 기반 m7g 인스턴스를 아프리카(케이프타운)에서 중간부터 16xlarge까지 다양한 크기로 지원합니다. 자세한 내용은 RabbitMQ용 Amazon MQ 브로커 인스턴스 유형 단원을 참조하십시오.
2025년 11월 17일	Amazon MQ는 이제 외부 LDAP 디렉터리 서비스를 사용하는 RabbitMQ 브로커에 대해 LDAP 인증 및 권한 부여를 지원합니다. Amazon MQ를 사용할 수 AWS 리전 있는 모든 AWS CDK 에서 또는 AWS Management Console AWS CloudFormation AWS CLI를 통해 LDAP를 구성할 수 있습니다. 자세한 내용은 RabbitMQ용 Amazon MQ에 대한 LDAP 인증 및 권한 부여 단원을 참조하십시오.
2025년 10월 22일	아시아 태평양(뉴질랜드) 리전에서 이제 Amazon MQ를 사용할 수 있습니다. 이 사용 가능한 리전에 대한 자세한 내용은 AWS 일반 참조 가이드의 AWS 리전 및 엔드포인트를 참조하세요.

Date	설명서 업데이트
2025년 9월 3일	Amazon MQ는 이제 퍼블릭 ID 제공업체(idP)가 있는 RabbitMQ 브로커에 대한 OAuth 2.0 인증 및 권한 부여를 지원합니다. Amazon MQ를 사용할 수 AWS 리전 있는 모든 AWS CDK 에서 OAuth 2.0~ AWS Management Console AWS CloudFormation AWS CLI, 또는를 구성할 수 있습니다. 자세한 내용은 RabbitMQ용 Amazon MQ에 대한 OAuth 2.0 인증 및 권한 부여 단원을 참조하십시오.
2025년 7월 22일	Amazon MQ는 이제 RabbitMQ용 Graviton3 기반 `m7g` 인스턴스를 중간에서 16xlarge까지 다양한 크기로 지원합니다. `m7g` 인스턴스에서 실행되는 RabbitMQ 클러스터는 `m5` 인스턴스에서 실행되는 비교 가능한 RabbitMQ용 Amazon MQ 클러스터에 비해 최대 50% 더 높은 워크로드 용량과 최대 85% 향상된 처리량을 제공합니다. `M7g` 인스턴스에는 인스턴스 크기에 따라 달라지는 최적화된 디스크 볼륨 크기도 있습니다. 자세한 내용은 Broker instance types 단원을 참조하십시오. Amazon MQ의 `M7g` 인스턴스는 현재 아프리카(케이프타운), 캐나다 서부(캘거리), 유럽(밀라노) 리전을 제외한 일반적으로 사용 가능한 모든 리전에서 사용할 수 있습니다.
2025년 7월 8일	아시아 태평양(타이베이) 리전에서 이제 Amazon MQ를 사용할 수 있습니다. 이 사용 가능한 리전에 대한 자세한 내용은 AWS 일반 참조 가이드의 AWS 리전 및 엔드포인트를 참조하세요.
2025년 4월 22일	이제 `DeleteConfiguration` API를 사용하여 Amazon MQ 브로커 구성을 삭제할 수 있습니다. 자세한 내용은 Amazon MQ API 참조의 구성을 참조하세요.
2025년 4월 16일	RabbitMQ용 Amazon MQ는 이제 듀얼 스택(IPv4 및 IPv6) 엔드포인트를 사용하여 퍼블릭 및 프라이빗 브로커에 연결할 수 있도록 지원합니다. 자세한 내용은 Connecting to Amazon MQ 및 Configuring a private Amazon MQ broker 섹션을 참조하세요.

Date	설명서 업데이트
2025년 4월 7일	이제 Amazon MQ는 아시아 태평양(태국) 및 멕시코(중부) 리전에서 사용할 수 있습니다. 이 사용 가능한 리전에 대한 자세한 내용은 AWS 일반 참조 가이드의 AWS 리전 및 엔드포인트를 참조하세요.
2025년 2월 13일	이제 캐나다(중부) 및 캐나다 서부(캘거리) 리전에서 Amazon MQ API FIPS 엔드포인트를 사용할 수 있습니다. Amazon MQ API에서 FIPS 엔드포인트를 사용하는 방법에 대한 자세한 내용은 Connecting to Amazon MQ 섹션을 참조하세요. 이 사용 가능한 리전에 대한 자세한 내용은 AWS 일반 참조 가이드의 AWS 리전 및 엔드포인트를 참조하세요.
2025년 2월 12일	Amazon MQ는 다음과 같은 인스턴스 유형 지원 종료일을 발표합니다. Broker instance types • ActiveMQ `mq.t2.micro` : 2025년 5월 12일 • ActiveMQ `mq.m4.large` : 2025년 5월 12일 2025년 3월 17일 이후에는 `mq.t2.micro` 또는 `mq.m4.large` 에서 브로커를 생성할 수 없습니다.
2024년 12월 10일	Amazon MQ는 이제 트래픽을 퍼블릭 인터넷에 노출하지 않고 Virtual Private Cloud(VPCs)와 Amazon MQ API 간에 연결하기 위해 AWS PrivateLink 사용을 지원합니다. 자세한 내용은 the section called “AWS PrivateLink를 사용하여 Amazon MQ에 연결” 단원을 참조하십시오.
2024년 11월 18일	아시아 태평양(말레이시아) 리전에서 Amazon MQ를 사용할 수 있습니다. 이 사용 가능한 리전에 대한 자세한 내용은 AWS 일반 참조 가이드의 AWS 리전 및 엔드포인트를 참조하세요.

Date	설명서 업데이트
2024년 11월 14일	Amazon MQ는 다음과 같은 엔진 버전 지원 종료일을 발표합니다. ActiveMQ용 Amazon MQ 엔진 버전 관리 • ActiveMQ 5.17: 2025년 6월 16일 RabbitMQ용 Amazon MQ 엔진 버전 관리 • RabbitMQ 3.11: 2025년 2월 17일 • RabbitMQ 3.12: 2025년 3월 17일 최신 버전으로 업그레이드하는 방법에 대한 자세한 정보는 Amazon MQ 브로커 엔진 버전 업그레이드 섹션을 참조하세요.
2024년 11월 13일	Amazon MQ는 이제 IPv4 또는 IPv6를 사용하여 연결할 수 있는 듀얼 스택 서비스 엔드포인트를 지원합니다. Amazon MQ 듀얼 스택 리전 서비스 엔드포인트는 A 및 AAAA DNS 레코드로 확인할 수 있습니다. 자세한 내용은 ??? 단원을 참조하십시오.
2024년 7월 25일	Amazon MQ가 이제 새로운 마이너 엔진 버전 릴리스인 ActiveMQ 5.18을 지원합니다. 자세한 내용은 다음을 참조하세요. • ActiveMQ 5.18 릴리스 페이지 • ActiveMQ용 Amazon MQ 엔진 버전 관리 • Amazon MQ 브로커 엔진 버전 업그레이드 • Spring XML 구성 파일 사용
2024년 7월 22일	Amazon MQ는 이제 버전 3.13 이상을 사용하는 브로커에서만 쿼럼 대기열을 지원합니다. 쿼럼 대기열은 Raft 합의 알고리즘을 사용하여 데이터 일관성을 유지하는 복제된 FIFO 대기열 유형입니다. 쿼럼 대기열은 유해 메시지 처리 기능을 제공하므로 처리되지 않은 메시지를 관리하는 데 도움이 될 수 있습니다. 쿼럼 대기열을 시작하려면 Amazon MQ 기반 RabbitMQ의 쿼럼 대기열 단원을 참조하세요.

Date	설명서 업데이트
2024년 7월 2일	이제 RabbitMQ용 Amazon MQ가 이제 마이너 버전 릴리스인 RabbitMQ 3.13을 지원합니다. 엔진 버전 3.13 이상을 사용하는 모든 브로커의 경우, 유지 관리 기간 동안 지원되는 최신 패치 버전으로의 업그레이드는 Amazon MQ에서 관리합니다. 자세한 내용은 Amazon MQ 브로커 엔진 버전 업그레이드 단원을 참조하십시오. 엔진 버전 3.13을 사용하는 브로커의 대기열, 채널당 소비자 및 shovel에 대한 새로운 한도가 포함되도록 RabbitMQ용 Amazon MQ 크기 조정 지침이 업데이트되었습니다. 이 릴리스의 수정 사항 및 기능에 대한 자세한 내용은 RabbitMQ 서버 GitHub 리포지토리에서 RabbitMQ 3.13 릴리스 정보를 참조하세요. 지원되는 RabbitMQ용 Amazon MQ 버전 및 브로커 업그레이드에 대한 자세한 내용은 RabbitMQ용 Amazon MQ 엔진 버전 관리 섹션을 참조하세요.
2024년 6월 10일	이제 캐나다 서부(캘거리) 리전에서 Amazon MQ를 사용할 수 있습니다. 이 사용 가능한 리전에 대한 자세한 내용은 AWS 일반 참조 가이드의 AWS 리전 및 엔드포인트를 참조하세요.
2024년 5월 10일	Amazon MQ 버전 지원 캘린더에는 브로커 엔진 버전의 지원 종료 시기가 표시되어 있습니다. 엔진 버전의 지원이 종료되면 Amazon MQ는 해당 버전의 모든 브로커를 지원되는 다음 마이너 버전으로 자동 업데이트합니다. Amazon MQ는 엔진 버전 지원이 종료되기 최소 90일 전에 미리 통지합니다. 버전 지원 캘린더 및 지원 종료를 확인하려면 다음을 참조하세요. • ActiveMQ용 Amazon MQ 엔진 버전 관리 • RabbitMQ용 Amazon MQ 엔진 버전 관리 유지 관리 기간 동안 브로커가 다음 패치 버전으로 업데이트되도록 자동 마이너 버전 업그레이드를 활성화할 수도 있습니다. 자세한 내용은 Amazon MQ 브로커 엔진 버전 업그레이드 섹션을 참조하세요.

Date	설명서 업데이트
2024년 5월 9일	이제 RabbitMQ용 Amazon MQ가 이제 마이너 버전 릴리스인 RabbitMQ 3.12를 지원합니다. 3.12.13 이상의 모든 브로커는 클래식 대기열 버전 2(CQv2)를 사용하며 3.12.13 이상의 모든 대기열은 지연 대기열로 작동합니다. 3.12.13 이전 버전의 브로커의 경우 CQv2 및 지연 대기열을 활성화하거나 최신 버전의 RabbitMQ용 Amazon MQ로 업그레이드할 것을 권장합니다. 이 릴리스의 수정 사항 및 기능에 대한 자세한 내용은 다음을 참조하세요. • RabbitMQ 서버 GitHub 리포지토리의 RabbitMQ 3.12 릴리스 정보 지원되는 RabbitMQ용 Amazon MQ 버전 및 브로커 업그레이드에 대한 자세한 내용은 RabbitMQ용 Amazon MQ 엔진 버전 관리 섹션을 참조하세요.
2024년 3월 4일	RabbitMQ용 Amazon MQ가 이제 RabbitMQ 3.11.28을 지원합니다. 이 릴리스의 수정 사항 및 기능에 대한 자세한 내용은 다음을 참조하세요. • RabbitMQ 서버 GitHub 리포지토리의 RabbitMQ 3.11.28 릴리스 정보 • RabbitMQ changelog 지원되는 RabbitMQ용 Amazon MQ 버전 및 브로커 업그레이드에 대한 자세한 내용은 RabbitMQ용 Amazon MQ 엔진 버전 관리 섹션을 참조하세요.
2024년 1월 19일	RabbitMQ용 Amazon MQ에서는 사용자 이름으로 "guest"를 지원하지 않으므로 새 브로커를 생성하면 기본 게스트 계정이 삭제됩니다. Amazon MQ는 또한 고객이 생성한 "guest"라는 계정도 주기적으로 삭제합니다.
2023년 12월 15일	이제 이스라엘(텔아비브) 리전에서 Amazon MQ를 사용할 수 있습니다. 이 사용 가능한 리전에 대한 자세한 내용은 AWS 일반 참조 가이드의 AWS 리전 및 엔드포인트를 참조하세요.

Date	설명서 업데이트
2023년 12월 11일	RabbitMQ용 Amazon MQ가 이제 RabbitMQ 3.10.25를 지원합니다. 이 릴리스의 수정 사항 및 기능에 대한 자세한 내용은 다음을 참조하세요. • RabbitMQ 서버 GitHub 리포지토리의 RabbitMQ 3.10.25 릴리스 정보 • RabbitMQ changelog 지원되는 RabbitMQ용 Amazon MQ 버전 및 브로커 업그레이드에 대한 자세한 내용은 RabbitMQ용 Amazon MQ 엔진 버전 관리 섹션을 참조하세요.
2023년 10월 26일	Amazon MQ에서 최신 ActiveMQ 마이너 버전 5.15.16, 5.16.7, 5.17.6이 중요 업데이트와 함께 출시되었습니다. ActiveMQ의 이전 마이너 버전은 더 이상 사용되지 않으며 모든 버전의 5.15에서 5.15.16으로, 5.16에서 5.16.7로, 5.17에서 5.17.6으로 모든 브로커를 업데이트할 예정입니다. ActiveMQ 브로커 업데이트에 대한 자세한 내용은 ActiveMQ용 Amazon MQ 엔진 버전 관리 섹션을 참조하세요.
2023년 9월 27일	RabbitMQ용 Amazon MQ가 이제 RabbitMQ 3.11.20을 지원합니다. 이 릴리스의 수정 사항 및 기능에 대한 자세한 내용은 다음을 참조하세요. • RabbitMQ 서버 GitHub 리포지토리의 RabbitMQ 3.11.20 릴리스 정보 • RabbitMQ changelog 지원되는 RabbitMQ용 Amazon MQ 버전 및 브로커 업그레이드에 대한 자세한 내용은 RabbitMQ용 Amazon MQ 엔진 버전 관리 섹션을 참조하세요.

Date	설명서 업데이트
2023년 7월 27일	RabbitMQ용 Amazon MQ가 이제 RabbitMQ 3.11.16을 지원합니다. 이 릴리스의 수정 사항 및 기능에 대한 자세한 내용은 다음을 참조하세요. • RabbitMQ 서버 GitHub 리포지토리의 RabbitMQ 3.11.16 릴리스 정보 • RabbitMQ changelog 지원되는 RabbitMQ용 Amazon MQ 버전 및 브로커 업그레이드에 대한 자세한 내용은 RabbitMQ용 Amazon MQ 엔진 버전 관리 섹션을 참조하세요.
2023년 7월 27일	RabbitMQ용 Amazon MQ는 이제 RabbitMQ 브로커에 대한 구성을 생성하고 적용하는 것을 지원합니다. 브로커에 구성을 추가하는 방법에 대한 자세한 내용은 RabbitMQ Broker Configurations 섹션을 참조하세요. 이 기능에 대한 자세한 내용은 다음을 참조하세요. • 운영자 정책 • 운영자 정책 변경
2023년 6월 23일	Amazon MQ는 이제 새로운 보조 엔진 버전 릴리스인 ActiveMQ 5.17.3을 지원합니다. 이번 릴리스는 Amazon MQ의 새로운 리전 간 데이터 복제(CRDR) 기능을 지원합니다. 자세한 내용은 다음을 참조하세요. • CRDR을 시작하려면 개발자 안내서의 ActiveMQ용 Amazon MQ의 리전 간 데이터 복제 섹션을 참조하세요. • ActiveMQ 5.17.3 릴리스 페이지 • ActiveMQ용 Amazon MQ 엔진 버전 관리 • Amazon MQ 브로커 엔진 버전 업그레이드 • Spring XML 구성 파일 사용

Date	설명서 업데이트
2023년 6월 21일	ActiveMQ용 Amazon MQ는 이제 기본 리전의 기본 브로커에서 복제본 AWS 리전의 복제본 브로커로 비동기식 메시지 복제를 허용하는 리전 간 데이터 복제(CRDR) 기능을 제공합니다. 기본 리전의 기본 브로커에 장애가 발생하면 전환 또는 장애 조치를 시작하여 보조 리전의 복제본 브로커를 기본 브로커로 승격할 수 있습니다. CRDR을 시작하려면 개발자 안내서의 ActiveMQ용 Amazon MQ의 리전 간 데이터 복제 섹션을 참조하세요.
2023년 5월 18일	이제 다음 리전에서 Amazon MQ를 사용할 수 있습니다. • 아시아 태평양(멜버른) • 아시아 태평양(하이데라바드) • 유럽(스페인) • 유럽(취리히) 이 사용 가능한 리전에 대한 자세한 내용은 AWS 일반 참조 가이드의 AWS 리전 및 엔드포인트를 참조하세요.
2023년 4월 14일	RabbitMQ용 Amazon MQ는 이제 RabbitMQ 버전 3.9.27을 지원합니다. 이 릴리스의 수정 사항 및 기능에 대한 자세한 내용은 다음을 참조하세요. • RabbitMQ 서버 GitHub 리포지토리의 RabbitMQ 3.9.27 릴리스 정보 • RabbitMQ changelog 지원되는 RabbitMQ용 Amazon MQ 버전 및 브로커 업그레이드에 대한 자세한 내용은 RabbitMQ용 Amazon MQ 엔진 버전 관리 섹션을 참조하세요.

Date	설명서 업데이트
2023년 4월 14일	RabbitMQ용 Amazon MQ는 이제 RabbitMQ 버전 3.10.20을 지원합니다. 이 릴리스의 수정 사항 및 기능에 대한 자세한 내용은 다음을 참조하세요. • RabbitMQ 서버 GitHub 리포지토리의 RabbitMQ 3.10.20 릴리스 정보 • RabbitMQ changelog 지원되는 RabbitMQ용 Amazon MQ 버전 및 브로커 업그레이드에 대한 자세한 내용은 RabbitMQ용 Amazon MQ 엔진 버전 관리 섹션을 참조하세요.
2023년 3월 31일	RabbitMQ용 Amazon MQ가 RabbitMQ 엔진 버전 3.10.17을 비활성화했습니다. RabbitMQ용 Amazon MQ 팀과 RabbitMQ의 오픈 소스 유지 관리자는 버전 3.10.17에서 RabbitMQ 관리 콘솔 문제를 확인했습니다. Amazon MQ는 이 버전을 철회했습니다. 이 문제의 영향을 줄이려면 RabbitMQ의 새 패치 버전이 준비되는 동안 버전 3.10.10으로 새 브로커를 생성하세요. 최신 버그 수정, 보안 업데이트 및 성능 향상을 자동으로 가져오려면 버전 업그레이드 옵션을 활성화하는 것이 좋습니다. 사용할 수 있는 RabbitMQ용 Amazon MQ 버전에 자세한 내용은 RabbitMQ용 Amazon MQ 엔진 버전을 참조하세요.
2023년 3월 1일	RabbitMQ용 Amazon MQ는 이제 RabbitMQ 버전 3.10.17을 지원합니다. 이 릴리스의 수정 사항 및 기능에 대한 자세한 내용은 다음을 참조하세요. • RabbitMQ 서버 GitHub 리포지토리의 RabbitMQ 3.10.17 릴리스 정보 • RabbitMQ changelog 지원되는 RabbitMQ용 Amazon MQ 버전 및 브로커 업그레이드에 대한 자세한 내용은 RabbitMQ용 Amazon MQ 엔진 버전 관리 섹션을 참조하세요.

Date	설명서 업데이트
2023년 2월 21일	RabbitMQ용 Amazon MQ는 이제 AWS Key Management Service (KMS)와 통합되어 서버 측 암호화를 제공합니다. 이제 자체 고객 관리형 CMK를 선택하거나 AWS KMS 계정에서 AWS 관리형 KMS 키를 사용할 수 있습니다. 자세한 내용은 저장 시 암호화 단원을 참조하십시오. Amazon MQ는 다음과 같은 방법으로 AWS KMS 키 사용을 지원합니다. • Amazon MQ 소유 KMS 키(Amazon MQ owned KMS key)(기본값) - 이 키는 Amazon MQ가 소유하고 관리하며 사용자 계정에 없습니다. • AWS 관리형 KMS 키 - AWS 관리형 KMS 키(`aws/mq`)는 Amazon MQ에서 사용자를 대신하여 생성, 관리 및 사용하는 계정의 KMS 키입니다. • 기존 고객 관리형 KMS 키 선택 - 고객 관리형 KMS 키는 사용자가 AWS Key Management Service (KMS)에서 생성하고 관리합니다.
2023년 1월 13일	RabbitMQ용 Amazon MQ는 이제 RabbitMQ 버전 3.8.34를 지원합니다. 이 릴리스의 수정 사항 및 기능에 대한 자세한 내용은 다음을 참조하세요. • RabbitMQ 서버 GitHub 리포지토리의 RabbitMQ 3.8.34 릴리스 정보 • RabbitMQ changelog 지원되는 RabbitMQ용 Amazon MQ 버전 및 브로커 업그레이드에 대한 자세한 내용은 RabbitMQ용 Amazon MQ 엔진 버전 관리 섹션을 참조하세요.
2022년 12월 15일	RabbitMQ용 Amazon MQ는 이제 RabbitMQ 버전 3.9.24를 지원합니다. 이 릴리스의 수정 사항 및 기능에 대한 자세한 내용은 다음을 참조하세요. • RabbitMQ 서버 GitHub 리포지토리의 RabbitMQ 3.9.24 릴리스 정보 • RabbitMQ changelog 지원되는 RabbitMQ용 Amazon MQ 버전 및 브로커 업그레이드에 대한 자세한 내용은 RabbitMQ용 Amazon MQ 엔진 버전 관리 섹션을 참조하세요.

Date	설명서 업데이트
2022년 12월 13일	중동(UAE) 리전에서 이제 Amazon MQ를 사용할 수 있습니다. 이 사용 가능한 리전에 대한 자세한 내용은 AWS 일반 참조 가이드의 AWS 리전 및 엔드포인트를 참조하세요.
2022년 11월 14일	RabbitMQ용 Amazon MQ는 이제 주 엔진 버전 릴리스인 3.10을 지원합니다. 이제 RabbitMQ 대기열에서 클래식 대기열 버전 2(CQv2)를 활성화할 수 있습니다. 3.8에서 3.10으로의 직접 업데이트는 지원되지 않습니다. 자세한 내용은 다음을 참조하세요. • RabbitMQ 3.10.10 릴리스 정보 • RabbitMQ changelog 지원되는 RabbitMQ용 Amazon MQ 버전 및 브로커 업그레이드에 대한 자세한 내용은 RabbitMQ용 Amazon MQ 엔진 버전 관리 섹션을 참조하세요.
2022년 11월 9일	Amazon MQ는 이제 새로운 보조 엔진 버전 릴리스인 ActiveMQ 5.17.2를 지원합니다. 자세한 내용은 다음을 참조하세요. • ActiveMQ 5.17.2 릴리스 페이지 • ActiveMQ용 Amazon MQ 엔진 버전 관리 • Amazon MQ 브로커 엔진 버전 업그레이드 • Spring XML 구성 파일 사용
2022년 8월 17일	Amazon MQ는 이제 새로운 주 엔진 버전 릴리스인 ActiveMQ 5.17.1을 지원합니다. 자세한 내용은 다음을 참조하세요. • ActiveMQ 5.17.1 릴리스 페이지 • ActiveMQ용 Amazon MQ 엔진 버전 관리 • Amazon MQ 브로커 엔진 버전 업그레이드 • Spring XML 구성 파일 사용

Date	설명서 업데이트
2022년 7월 14일	Amazon MQ는 이제 보조 엔진 버전 릴리스인 ActiveMQ 5.16.5를 지원합니다. 자세한 내용은 다음을 참조하세요. • ActiveMQ 5.16.5 릴리스 페이지 • ActiveMQ용 Amazon MQ 엔진 버전 관리 • Spring XML 구성 파일 사용 • Amazon MQ 브로커 엔진 버전 업그레이드
2022년 5월 4일	Amazon MQ에 브로커 구성에서 `networkConnector` 요소에 대한 포괄적 언어가 추가되었습니다. • 브로커의 Amazon MQ 네트워크 생성 및 구성
2022년 4월 25일	Amazon MQ 이 릴리스에는 `CRITICAL_ACTION_REQUIRED` 브로커 상태와 `ActionRequired` API 속성이 추가되었습니다. `CRITICAL_ACTION_REQUIRED` 는 브로커 성능이 저하되면 알려줍니다. `ActionRequired` 는 개발자 가이드에서 문제 해결 방법에 대한 지침을 찾는 데 사용할 수 있는 코드를 제공합니다. • 문제 해결 • Amazon MQ API 참조의 `ActionRequired` 문서
2022년 4월 20일	Amazon MQ는 이제 보조 엔진 버전 릴리스인 ActiveMQ 5.16.4를 지원합니다. 자세한 내용은 다음을 참조하세요. • ActiveMQ 5.16.4 릴리스 페이지 • ActiveMQ용 Amazon MQ 엔진 버전 관리 • Spring XML 구성 파일 사용 • Amazon MQ 브로커 엔진 버전 업그레이드
2022년 3월 1일	아시아 태평양(자카르타) 리전에서 Amazon MQ를 사용할 수 있습니다. 이 사용 가능한 리전에 대한 자세한 내용은 AWS 일반 참조 가이드의 AWS 리전 및 엔드포인트를 참조하세요.

Date	설명서 업데이트
2022년 2월 25일	RabbitMQ용 Amazon MQ는 이제 RabbitMQ 3.8.27을 지원합니다. 이 릴리스의 수정 사항 및 기능에 대한 자세한 내용은 다음을 참조하세요. • RabbitMQ 서버 GitHub 리포지토리의 RabbitMQ 3.8.27 릴리스 정보 • RabbitMQ changelog 지원되는 RabbitMQ용 Amazon MQ 버전 및 브로커 업그레이드에 대한 자세한 내용은 RabbitMQ용 Amazon MQ 엔진 버전 관리 섹션을 참조하세요.
2022년 2월 16일	이제 아프리카(케이프타운) 리전에서 Amazon MQ를 사용할 수 있습니다. 이 사용 가능한 리전에 대한 자세한 내용은 AWS 일반 참조 가이드의 AWS 리전 및 엔드포인트를 참조하세요.
2022년 2월 14일	RabbitMQ용 Amazon MQ는 이제 RabbitMQ 3.9.13을 지원합니다. 마이너 버전 자동 업그레이드는 래빗 3.8에서 3.9로 업그레이드하는 데 사용할 수 없습니다. 그렇게 하려면 브로커 수동 업그레이드를 실행하십시오. RabbitMQ 3.9에 도입된 새로운 기능에 대한 자세한 내용은 GitHub 웹 사이트의 버전 3.9.0용 릴리스 노트 페이지를 참조하십시오. **Note** 현재 Amazon MQ는 지원하지 않습니다.스트림또는 RabbitMQ 3.9에 도입된 JSON에서 구조화된 로깅을 사용합니다. 이 릴리스의 수정 사항 및 기능에 대한 자세한 내용은 다음을 참조하세요. • RabbitMQ 서버 GitHub 리포지토리의 RabbitMQ 3.9.13 릴리스 정보 • RabbitMQ changelog 지원되는 RabbitMQ용 Amazon MQ 버전 및 브로커 업그레이드에 대한 자세한 내용은 RabbitMQ용 Amazon MQ 엔진 버전 관리 섹션을 참조하세요.

Date	설명서 업데이트
2022년 2월 7일	Amazon MQ용 RabbitMQ는 새로운 브로커 지표를 도입하여 클러스터 배포의 세 노드 모두에서 평균 리소스 사용률을 모니터링할 수 있습니다. 자세한 내용은 다음을 참조하세요. • the section called “RabbitMQ 지표”
2022년 1월 18일	RabbitMQ용 Amazon MQ는 이제 RabbitMQ 3.8.26을 지원합니다. 이 릴리스의 수정 사항 및 기능에 대한 자세한 내용은 다음을 참조하세요. • RabbitMQ 서버 GitHub 리포지토리의 RabbitMQ 3.8.26 릴리스 정보 • RabbitMQ changelog 지원되는 RabbitMQ용 Amazon MQ 버전 및 브로커 업그레이드에 대한 자세한 내용은 RabbitMQ용 Amazon MQ 엔진 버전 관리 섹션을 참조하세요.
2022년 1월 13일	Amazon MQ에 브로커가 높은 메모리 경보를 발생시켰고 비정상 상태일 경우 이를 알려주기 위한 `RABBITMQ_MEMORY_ALARM` 상태 코드가 도입되었습니다. Amazon MQ는 높은 메모리 경보를 진단, 해결 및 방지하는 데 도움이 되는 자세한 정보와 권장 사항을 제공합니다. 추가 정보는 다음을 참조하세요. • the section called “ `RABBITMQ_MEMORY_ALARM` ”
2022년 1월 6일	ActiveMQ 브로커용 Amazon MQ에 대한 CloudWatch Logs를 구성하면, Amazon MQ는 혼동된 대리자 문제를 방지하기 위한 IAM 리소스 기반 정책의 전역 조건 컨텍스트 키인 `aws:SourceArn` 및 `aws:SourceAccount` 사용을 지원합니다. 추가 정보는 다음을 참조하세요. • the section called “교차 서비스 혼동된 대리인 방지”
2021년 12월 20일	ActiveMQ용 Amazon MQ에는 새로운 지표 세트 도입으로 지원되는 다양한 전송 프로토콜을 사용하여 브로커에 연결할 수 있는 최대 연결 수를 모니터링할 수 있는 지표와 브로커 네트워크의 브로커에 연결된 노드 수를 모니터링할 수 있는 새로운 지표도 도입되었습니다. 추가 정보는 다음을 참조하세요. • the section called “ActiveMQ 지표”

Date	설명서 업데이트
2021년 11월 16일	RabbitMQ용 Amazon MQ는 이제 RabbitMQ 3.8.23을 지원합니다. 이 릴리스의 수정 사항 및 기능에 대한 자세한 내용은 다음을 참조하세요. • RabbitMQ 서버 GitHub 리포지토리의 RabbitMQ 3.8.23 릴리스 정보 • RabbitMQ changelog 지원되는 RabbitMQ용 Amazon MQ 버전 및 브로커 업그레이드에 대한 자세한 내용은 RabbitMQ용 Amazon MQ 엔진 버전 관리 섹션을 참조하세요.
2021년 10월 12일	Amazon MQ는 이제 보조 엔진 버전 릴리스인 ActiveMQ 5.16.3을 지원합니다. 자세한 내용은 다음을 참조하세요. • ActiveMQ 5.16.3 릴리스 페이지 • ActiveMQ용 Amazon MQ 엔진 버전 관리 • Amazon MQ 브로커 엔진 버전 업그레이드 • Spring XML 구성 파일 사용
2021년 9월 8일	RabbitMQ용 Amazon MQ는 이제 RabbitMQ 3.8.22를 지원합니다. 이 릴리스에는 이전에 지원되는 버전 RabbitMQ 3.8.17에서 확인된 메시지별 유지 시간(TTL)을 사용하는 대기열 관련 문제에 대한 수정이 포함되어 있습니다. 기존 브로커를 버전 3.8.22로 업그레이드하는 것이 좋습니다. 이 릴리스의 수정 사항 및 기능에 대한 자세한 내용은 다음을 참조하세요. • RabbitMQ 서버 GitHub 리포지토리의 RabbitMQ 3.8.22 릴리스 정보 • RabbitMQ changelog 지원되는 RabbitMQ용 Amazon MQ 버전 및 브로커 업그레이드에 대한 자세한 내용은 RabbitMQ용 Amazon MQ 엔진 버전 관리 단원을 참조하세요.
2021년 8월 25일	RabbitMQ용 Amazon MQ는 메시지별 유지 시간(TTL)을 사용하는 대기열에서 식별된 문제로 인해 RabbitMQ 엔진 버전 3.8.17을 일시적으로 비활성화했습니다. 버전 3.8.11을 사용하는 것이 좋습니다.

Date	설명서 업데이트
2021년 7월 29일	RabbitMQ용 Amazon MQ는 이제 RabbitMQ 3.8.17을 지원합니다. 이 업데이트에 포함된 수정 사항 및 기능에 대한 자세한 내용은 다음을 참조하세요. • RabbitMQ 서버 GitHub 리포지토리의 RabbitMQ 3.8.17 릴리스 정보 • RabbitMQ changelog • RabbitMQ용 Amazon MQ 엔진 버전 관리
2021년 7월 16일	이제 AWS Management Console AWS CLI또는 Amazon MQ API를 사용하여 Amazon MQ 브로커의 유지 관리 기간을 조정할 수 있습니다. 브로커 유지 관리 기간에 대한 자세한 내용은 다음을 참조하세요. • Amazon MQ 브로커의 유지 관리 기간 예약
2021년 7월 6일	RabbitMQ용 Amazon MQ는 일관적 해시 교환 유형에 대한 지원을 도입합니다. 일관적 해시 교환은 메시지의 라우팅 키에서 계산된 해시 값을 기반으로 메시지의 경로를 대기열로 지정합니다. 자세한 내용은 다음을 참조하세요. • 일관적 해시 교환 플러그 인 • RabbitMQ GitHub 리포지토리의 RabbitMQ 일관적 해시 교환 유형
2021년 6월 7일	Amazon MQ는 이제 새로운 주 엔진 버전 릴리스인 ActiveMQ 5.16.2를 지원합니다. 자세한 내용은 다음을 참조하세요. • ActiveMQ 5.16.2 릴리스 페이지 • ActiveMQ용 Amazon MQ 엔진 버전 관리 • Amazon MQ 브로커 엔진 버전 업그레이드 • Spring XML 구성 파일 사용
2021년 5월 26일	이제 중국(베이징) 및 중국(닝샤) 리전에서 RabbitMQ용 Amazon MQ를 사용할 수 있습니다. 사용 가능한 리전에 대한 자세한 내용은 AWS 리전 및 엔드포인트를 참조하세요.

Date	설명서 업데이트
2021년 5월 18일	RabbitMQ용 Amazon MQ는 브로커 기본값을 구현합니다. 브로커를 처음 생성할 때 Amazon MQ는 브로커의 성능을 최적화하기 위해 선택한 인스턴스 유형 및 배포 모드에 따라 일련의 브로커 정책 및 vhost 제한을 생성합니다. 자세한 내용은 다음을 참조하세요. https://docs.aws.amazon.com//amazon-mq/latest/developer-guide/rabbitmq-defaults.html
2021년 5월 5일	Amazon MQ는 이제 ActiveMQ 5.15.15를 지원합니다. 자세한 내용은 다음을 참조하세요. • ActiveMQ 5.15.15 릴리스 페이지 • ActiveMQ용 Amazon MQ 엔진 버전 관리 • Spring XML 구성 파일 사용
2021년 5월 5일	Amazon MQ가 AWS 관리형 정책에 대한 변경 사항 추적을 시작했습니다. 자세한 내용은 다음을 참조하세요. • the section called “AWS 관리형 정책”
2021년 4월 14일	이제 중국(베이징) 및 중국(닝샤) 리전에서 Amazon MQ를 사용할 수 있습니다. 사용 가능한 리전에 대한 자세한 내용은 AWS 리전 및 엔드포인트를 참조하세요.
2021년 4월 7일	Amazon MQ는 이제 RabbitMQ 3.8.11을 지원합니다. 이 업데이트에 포함된 수정 사항 및 기능에 대한 자세한 내용은 다음을 참조하세요. • RabbitMQ 서버 GitHub 리포지토리의 RabbitMQ 3.8.11 릴리스 정보 • RabbitMQ changelog • RabbitMQ용 Amazon MQ 엔진 버전 관리
2021년 4월 1일	이제 아시아 태평양(오사카) 리전에서 Amazon MQ를 사용할 수 있습니다. 사용 가능한 리전에 대한 자세한 내용은 Amazon MQ 리전 및 엔드포인트를 참조하세요.

Date	설명서 업데이트
2020년 12월 21일	Amazon MQ는 이제 ActiveMQ 5.15.14를 지원합니다. 자세한 내용은 다음을 참조하세요. • ActiveMQ 5.15.14 릴리스 정보 • ActiveMQ용 Amazon MQ 엔진 버전 관리 • Spring XML 구성 파일 사용 • **Important** 이 릴리스에서 알려진 Apache ActiveMQ 문제로 인해 ActiveMQ 웹 콘솔에서 새로운 Pause Queue(대기열 일시 중지) 버튼을 ActiveMQ용 Amazon MQ 브로커에 사용할 수 없습니다. 이 문제에 대한 자세한 내용은 AMQ-8104를 참조하세요.

Date	설명서 업데이트
2020년 11월 4일	Amazon MQ는 이제 널리 사용되는 오픈 소스 메시지 브로커인 RabbitMQ를 지원합니다. 이렇게 하면 코드를 다시 작성할 필요 AWS 없이 기존 RabbitMQ 메시지 브로커를 로 마이그레이션할 수 있습니다. RabbitMQ용 Amazon MQ는 개별 메시지 브로커 및 클러스터링된 메시지 브로커를 모두 관리하고 인프라 프로비저닝, 브로커 설정, 소프트웨어 업데이트와 같은 작업을 처리합니다. • Amazon MQ는 RabbitMQ 3.8.6을 지원합니다. 지원되는 엔진 버전에 대한 자세한 내용은 the section called “버전 관리” 단원을 참조하세요. • AWS 프리 티어에는 1년간 매월 최대 750시간의 단일 인스턴스 `mq.t3.micro` 브로커 및 최대 20GB의 스토리지가 포함됩니다. 지원되는 인스턴스 유형에 대한 자세한 내용은 Broker instance types 단원을 참조하세요. • Amazon MQ for RabbitMQ를 사용하면 AMQP 0-9-1을 사용하여 RabbitMQ 클라이언트 라이브러리가 지원하는 모든 언어로 브로커에 액세스할 수 있습니다. 지원되는 프로토콜 및 암호 제품군에 대한 자세한 내용은 the section called “RabbitMQ용 Amazon MQ 프로토콜” 단원을 참조하세요. • RabbitMQ용 Amazon MQ는 Amazon MQ가 현재 제공되는 모든 리전에서 사용할 수 있습니다. 사용 가능한 모든 리전에 대해 자세히 알아보려면 AWS 리전 표를 참조하세요. Amazon MQ 사용을 시작하고, 브로커를 생성하고, JVM 기반 애플리케이션을 RabbitMQ 브로커에 연결하려면 시작하기: RabbitMQ 브로커 생성 및 연결 단원을 참조하세요.
2020년 10월 22일	Amazon MQ는 ActiveMQ 5.15.13을 지원합니다. 자세한 내용은 다음을 참조하세요. • ActiveMQ 5.15.13 릴리스 정보 • ActiveMQ용 Amazon MQ 엔진 버전 관리 • Spring XML 구성 파일 사용
2020년 9월 30일	이제 유럽(밀라노) 리전에서 Amazon MQ를 사용할 수 있습니다. 사용 가능한 리전에 대한 자세한 내용은 Amazon MQ 리전 및 엔드포인트를 참조하세요.

Date	설명서 업데이트
2020년 7월 27일	Active Directory 또는 다른 LDAP 서버에 저장된 자격 증명을 사용하여 Amazon MQ 사용자를 인증할 수 있습니다. 또한 Amazon MQ 사용자를 추가, 삭제 및 수정하고 주제 및 대기열에 대한 권한을 할당할 수도 있습니다. 자세한 정보는 LDAP와 ActiveMQ 통합을 참조하세요.
2020년 7월 17일	Amazon MQ는 이제 `mq.t3.micro` 인스턴스 유형을 지원합니다. 자세한 정보는 Broker instance types을 참조하세요.
2020년 6월 30일	Amazon MQ는 ActiveMQ 5.15.12를 지원합니다. 자세한 내용은 다음을 참조하세요. • ActiveMQ 5.15.12 릴리스 정보 • ActiveMQ용 Amazon MQ 엔진 버전 관리 • Spring XML 구성 파일 사용
2020년 4월 30일	Amazon MQ는 `broker` 요소에서 새 하위 컬렉션 요소 `systemUsage` 를 지원합니다. 자세한 정보는 systemUsage을 참조하세요. Amazon MQ는 `kahaDB` 하위 요소에서 세 가지 새로운 속성도 지원합니다. • `journalDiskSyncInterval` - `journalDiskSyncStrategy=periodic` 의 경우 디스크 동기화를 수행할 때의 간격(ms)입니다. • `journalDiskSyncStrategy` - 디스크 동기화 정책을 구성합니다. • `preallocationStrategy` - 새 저널 파일이 필요한 경우 브로커가 저널 파일을 미리 할당하는 방법을 구성합니다. 자세한 정보는 속성을 참조하세요.

Date	설명서 업데이트
2020년 3월 3일	Amazon MQ는 두 가지 새로운 CloudWatch 지표를 지원합니다. • `TempPercentUsage` - 비영구 메시지에 사용되는 사용 가능한 임시 스토리지의 백분율입니다. • `JobSchedulerStorePercentUsage` - 작업 스케줄러 스토어에 사용되는 디스크 공간의 백분율입니다. 자세한 내용은 Monitoring and logging Amazon MQ brokers 단원을 참조하십시오.
2020년 2월 4일	아시아 태평양(홍콩) 및 중동(바레인) 리전에서 Amazon MQ를 사용할 수 있습니다. 사용 가능한 리전에 대한 자세한 내용은 AWS 리전 및 엔드포인트를 참조하세요.
2020년 1월 22일	Amazon MQ는 ActiveMQ 5.15.10을 지원합니다. 자세한 내용은 다음을 참조하세요. • ActiveMQ 5.15.10 출시 정보 • ActiveMQ용 Amazon MQ 엔진 버전 관리 • Spring XML 구성 파일 사용
2019년 12월 19일	유럽(스톡홀름) 및 남아메리카(상파울루) 리전에서 Amazon MQ를 사용할 수 있습니다. 사용 가능한 리전에 대한 자세한 내용은 AWS 리전 및 엔드포인트를 참조하세요.

Date	설명서 업데이트
2019년 12월 16일	Amazon MQ는 브로커 스토리지에 기본 Amazon Elastic File System(Amazon EFS) 대신 Amazon Elastic Block Store(EBS)를 사용하여 처리량에 최적화된 브로커를 생성하도록 지원합니다. 다중 가용 영역에 걸친 높은 내구성 및 복제를 활용하려면 Amazon EFS를 사용하세요. 짧은 지연 시간 및 높은 처리량을 활용하려면 Amazon EBS를 사용세요. **Important** • Amazon EBS는 `mq.m5` 브로커 인스턴스 유형 패밀리에서만 사용할 수 있습니다. • 브로커를 생성한 후에는 브로커 인스턴스 유형은 변경할 수 있지만, 브로커 스토리지 유형은 변경할 수 없습니다. • Amazon EBS는 단일 가용 영역 내에서 데이터를 복제하며 ActiveMQ 활성/대기 배포 모드를 지원하지 않습니다. 자세한 내용은 다음을 참조하세요. • Storage • 처리량을 최대화하기 위해 올바른 브로커 스토리지 유형 선택 • Amazon MQ REST API 참조에서 `broker-instance-options` 리소스의 `storageType` 속성 • Monitoring and logging Amazon MQ brokers 단원의 `BurstBalance`, `VolumeReadOps` 및 `VolumeWriteOps` 지표입니다.
2019년 10월 18일	사용할 수 있는 두 개의 새로운 Amazon CloudWatch 지표는 `TotalEnqueueCount` 및 `TotalDequeueCount` 입니다. 자세한 내용은 Monitoring and logging Amazon MQ brokers 단원을 참조하세요.

Date	설명서 업데이트
2019년 10월 11일	Amazon MQ는 이제 미국 상용 리전에서 FIPS(Federal Information Processing Standard) 140-2 준수 엔드포인트를 지원합니다. 자세한 내용은 다음을 참조하세요. • FIPS(Federal Information Processing Standard) 140-2 • Amazon MQ 리전 및 엔드포인트
2019년 9월 30일	이제 Amazon MQ에는 호스트 인스턴스 유형을 변경하여 브로커의 크기를 조정하는 기능이 포함되어 있습니다. 자세한 내용은 `hostInstanceType` 의 `UpdateBrokerInput` 특성 및 `pendingHostInstanceType` 의 `DescribeBrokerOutput` 특성을 참조하세요.
2019년 8월 30일	이제 `UpdateBrokerInput` 을 사용하여 콘솔에서 브로커와 연결된 보안 그룹을 업데이트할 수 있습니다.
2019년 7월 22일	Amazon MQ는 AWS Key Management Service (KMS)와 통합되어 서버 측 암호화를 제공합니다. 이제 자체 고객 관리형 CMK를 선택하거나 AWS KMS 계정에서 AWS 관리형 KMS 키를 사용할 수 있습니다. 자세한 내용은 저장 시 암호화 단원을 참조하십시오. Amazon MQ는 다음과 같은 방법으로 AWS KMS 키 사용을 지원합니다. • AWS 소유 KMS 키 - 키는 Amazon MQ 소유이며 계정에 없습니다. • AWS 관리형 KMS 키 - AWS 관리형 KMS 키(aws/mq)는 Amazon MQ에서 사용자를 대신하여 생성, 관리 및 사용하는 계정의 KMS 키입니다. • 기존 고객 관리형 CMK 선택 - 고객 관리형 CMK는 사용자가 AWS Key Management Service (KMS)에서 생성하여 관리합니다.
2019년 6월 19일	유럽(파리) 및 아시아 태평양(뭄바이) 리전에서 Amazon MQ를 사용할 수 있습니다. 사용 가능한 리전에 대한 자세한 내용은 AWS 리전 및 엔드포인트를 참조하세요.
2019년 6월 12일	캐나다(중부) 리전에서 Amazon MQ를 사용할 수 있습니다. 사용 가능한 리전에 대한 자세한 내용은 AWS 리전 및 엔드포인트를 참조하세요.

Date	설명서 업데이트
2019년 6월 3일	사용할 수 있는 두 개의 새로운 Amazon CloudWatch 지표는 `EstablishedConnectionsCount` 및 `InactiveDurableSubscribers` 입니다. 자세한 내용은 다음을 참조하세요. • Monitoring and logging Amazon MQ brokers • Monitoring and logging Amazon MQ brokers
2019년 5월 10일	새로운 `mq.t2.micro` 인스턴스 유형을 위한 데이터 스토리지가 20GB로 제한됩니다. 자세한 내용은 다음을 참조하세요. • the section called “데이터 저장” • Broker instance types
2019년 4월 29일	이제 태그 기반 정책과 리소스 수준 권한을 사용할 수 있습니다. 자세한 내용은 다음을 참조하세요. • Amazon MQ에서 IAM을 사용하는 방법 • Amazon MQ API 작업에 대한 리소스 수준 권한
2019년 4월 16일	이제 REST API를 사용하여 브로커 엔진 및 브로커 인스턴스 옵션에 대한 정보를 검색할 수 있습니다. 자세한 내용은 다음을 참조하세요. • 브로커 인스턴스 옵션 • 브로커 엔진 유형
2019년 4월 8일	Amazon MQ는 ActiveMQ 5.15.9를 지원합니다. 자세한 내용은 다음을 참조하세요. • ActiveMQ 5.15.9 출시 정보 • ActiveMQ용 Amazon MQ 엔진 버전 관리 • Spring XML 구성 파일 사용

Date	설명서 업데이트
2019년 3월 4일	브로커 네트워크의 클라이언트 재분배 및 동적 장애 조치 구성에 대해 설명서가 개선되었습니다. `transportConnectors` 구성 옵션을 `networkConnectors` 구성 옵션과 함께 구성하여 동적 장애 조치를 활성화합니다. 자세한 내용은 다음을 참조하세요. • 전송 커넥터를 사용한 동적 장애 조치 • Amazon MQ 브로커 네트워크 • Amazon MQ Broker Configuration Parameters
2019년 2월 27일	Amazon MQ는 다음 리전 외에 유럽(런던) 리전에서도 사용할 수 있습니다. • 아시아 태평양(싱가포르) • 미국 동부(오하이오) • 미국 동부(버지니아 북부) • 미국 서부(캘리포니아 북부) • 미국 서부(오리건) • 아시아 태평양(도쿄) • 아시아 태평양(서울) • 아시아 태평양(시드니) • 유럽(프랑크푸르트) • 유럽(아일랜드)
2019년 1월 24일	비활성 대상을 제거하는 정책을 기본 구성에 포함했습니다.
2019년 1월 17일	Amazon MQ `mq.t2.micro` 인스턴스 유형은 이제 와이어 레벨 프로토콜당 연결 100개만 지원합니다. 자세한 내용은 Quotas in Amazon MQ 단원을 참조하세요.

Date	설명서 업데이트
2018년 12월 19일	브로커 네트워크에서 일련의 Amazon MQ 브로커를 구성할 수 있습니다. 자세한 내용은 다음 단원을 참조하세요. • Amazon MQ 브로커 네트워크 • Creating and Configuring a Network of Brokers • 브로커 네트워크를 올바로 구성 • `networkConnector` • `networkConnectionStartAsync`

2018년 12월 11일	Amazon MQ는 ActiveMQ 5.15.8, 5.15.6 및 5.15.0을 지원합니다.
	• ActiveMQ에서 해결된 버그 및 개선 사항:
	  • ActiveMQ 5.15.8 출시 정보
	  • ActiveMQ 5.15.7 출시 정보

2018년 12월 5일	AWS 는 비용 할당을 추적하는 데 도움이 되는 리소스 태그 지정을 지원합니다. 리소스를 생성할 때 또는 해당 리소스의 세부 정보를 확인하여 리소스에 태그를 지정할 수 있습니다. 자세한 내용은 리소스에 태그 지정 단원을 참조하세요.
2018년 11월 19일	AWS 는 Amazon MQ를 SOC 준수 서비스로 포함하도록 SOC 규정 준수 프로그램을 확장했습니다.
2018년 10월 15일	• 사용자당 최대 그룹 수는 20개입니다. 자세한 정보는 Users을 참조하세요. • 브로커당, 와이어 레벨 프로토콜당 최대 연결 수는 1,000개입니다. 자세한 정보는 브로커을 참조하세요.
2018년 10월 2일	AWS 는 Amazon MQ를 HIPAA 적격 서비스로 포함하도록 HIPAA 규정 준수 프로그램을 확장했습니다.

Date	설명서 업데이트

2018년 9월 27일	Amazon MQ는 ActiveMQ 5.15.0 외에도 ActiveMQ 5.15.6을 지원합니다. 자세한 내용은 다음을 참조하세요.
	• 시작하기: ActiveMQ 브로커 생성 및 연결
	• ActiveMQ 설명서에서 해결된 버그 및 개선 사항:
	  • ActiveMQ 5.15.6 출시 정보
	  • ActiveMQ 5.15.5 출시 정보
	  • ActiveMQ 5.15.4 출시 정보
	  • ActiveMQ 5.15.3 출시 정보
	  • ActiveMQ 5.15.2 출시 정보
	  • ActiveMQ 5.15.1 출시 정보
	• ActiveMQ 클라이언트 5.15.6

2018년 8월 31일	• 다음과 같은 지표를 사용할 수 있습니다.
	  • `CurrentConnectionsCount`
	  • `TotalConsumerCount`
	  • `TotalProducerCount`
	  자세한 내용은 Monitoring and logging Amazon MQ brokers 단원을 참조하세요.
	• Details(세부 정보) 페이지에 브로커의 IP 주소가 표시됩니다.
	  Note
	  퍼블릭 액세스 기능이 비활성화된 브로커의 경우 내부 IP 주소가 표시됩니다.

Date	설명서 업데이트
2018년 8월 30일	Amazon MQ는 다음 리전 외에 아시아 태평양(싱가포르) 리전에서도 사용할 수 있습니다. • 미국 동부(오하이오) • 미국 동부(버지니아 북부) • 미국 서부(캘리포니아 북부) • 미국 서부(오리건) • 아시아 태평양(도쿄) • 아시아 태평양(서울) • 아시아 태평양(시드니) • 유럽(프랑크푸르트) • 유럽(아일랜드)
2018년 7월 30일	일반 및 감사 로그를 Amazon CloudWatch Logs에 게시하도록 Amazon MQ를 구성할 수 있습니다. 자세한 내용은 Monitoring and logging Amazon MQ brokers 단원을 참조하십시오.
2018년 7월 25일	Amazon MQ는 다음 리전 외에 아시아 태평양(도쿄) 및 아시아 태평양(서울) 리전에서도 사용할 수 있습니다. • 미국 동부(오하이오) • 미국 동부(버지니아 북부) • 미국 서부(캘리포니아 북부) • 미국 서부(오리건) • 아시아 태평양(시드니) • 유럽(프랑크푸르트) • 유럽(아일랜드)
2018년 7월 19일	AWS CloudTrail 를 사용하여 Amazon MQ API 호출을 로깅할 수 있습니다. 자세한 내용은 Logging Amazon MQ API calls using CloudTrail 단원을 참조하십시오.

Date	설명서 업데이트
2018년 6월 29일	많은 처리량이 필요한 일반적인 개발, 테스트 및 프로덕션 워크로드에는 `mq.t2.micro` 및 `mq.m4.large` 외에도 다음과 같은 브로커 인스턴스 유형을 사용할 수 있습니다. • `mq.m5.large` • `mq.m5.xlarge` • `mq.m5.2xlarge` • `mq.m5.4xlarge` 자세한 정보는 Broker instance types을 참조하세요.
2018년 27월 6일	Amazon MQ는 다음 리전 외에 미국 서부(캘리포니아 북부) 리전에서도 사용할 수 있습니다. • 미국 동부(오하이오) • 미국 동부(버지니아 북부) • 미국 서부(오레곤) • 아시아 태평양(시드니) • 유럽(프랑크푸르트) • 유럽(아일랜드)

Date	설명서 업데이트

2018년 6월 14일	• AWS::Amazon MQ::Broker AWS CloudFormation 리소스를 사용하여 다음 작업을 수행할 수 있습니다.
	  • 브로커를 생성합니다.
	  • 구성 변경 사항을 추가하거나 지정된 브로커에 대한 사용자를 수정합니다.
	  • 지정된 브로커에 대한 정보를 반환합니다.
	  • 지정된 브로커를 삭제합니다.
	
	  Note
	  Amazon MQ 브로커 ConfigurationId의 속성 또는 Amazon MQ 브로커 사용자 속성 유형을 변경하면 브로커가 즉시 재부팅됩니다.
	• AWS::Amazon MQ::Configuration AWS CloudFormation 리소스를 사용하여 다음 작업을 수행할 수 있습니다.
	  • 구성을 생성합니다.
	  • 지정된 구성을 업데이트합니다.
	  • 지정된 구성에 대한 정보를 반환합니다.
	
	  Note
	  CloudFormation 를 사용하여 Amazon MQ 구성을 수정할 수 있지만 삭제할 수는 없습니다.

2018년 6월 7일	Amazon MQ 콘솔은 독일어, 브라질 포르투갈어, 스페인어, 이탈리아어 및 중국어 번체를 지원합니다.
2018년 5월 17일	브로커당 사용자 한도는 250명입니다. 자세한 정보는 Users을 참조하세요.
2018년 3월 13일	브로커 생성은 약 15분 정도 소요됩니다. 자세한 내용은 브로커 생성 완료를 참조하세요.

Date	설명서 업데이트
2018년 3월 1일	• `concurrentStoreAndDispatchQueues` 속성을 사용하는 Apache KahaDB에 대한 동시 저장 및 디스패치를 구성할 수 있습니다. • `>CpuCreditBalance` CloudWatch 지표를 `mq.t2.micro` 브로커 인스턴스 유형에 사용할 수 있습니다.
2018년 1월 10일	다음 변경 사항은 Amazon MQ 콘솔에 영향을 줍니다. • 브로커 목록에서 Creation(생성) 열이 기본적으로 숨겨집니다. 페이지 크기와 열을 사용자 지정하려면 을 선택합니다. • ***MyBroker*** 페이지의 연결 섹션에서 보안 그룹의 이름 또는  아이콘을 선택하면 VPC 콘솔 대신 EC2 콘솔이 열립니다. EC2 콘솔을 사용하면 인바운드 및 아웃바운드 규칙을 보다 직관적으로 구성할 수 있습니다. 자세한 내용은 업데이트된 Connecting a Java application to your broker 섹션을 참조하세요.
2018년 1월 9일	• REST 작업 ID `UpdateBroker` 의 권한이 IAM 콘솔에 `mq:UpdateBroker` 로 올바르게 나열됩니다. • 잘못된 `mq:DescribeEngine` 권한이 IAM 콘솔에서 제거됩니다.

Date	설명서 업데이트

2017년 11월 28일	이것은 Amazon MQ 개발자 안내서의 최초 릴리스입니다.
	• 다음 리전에서 Amazon MQ를 사용할 수 있습니다.
	  • 미국 동부(오하이오)
	  • 미국 동부(버지니아 북부)
	  • 미국 서부(오레곤)
	  • 아시아 태평양(시드니)
	  • 유럽(프랑크푸르트)
	  • 유럽(아일랜드)
	  `mq.t2.micro` 인스턴스 유형 사용에는 기준 수준 이상으로 버스트하는 기능이 있는 CPU 크레딧 및 기준 성능이 적용됩니다(자세한 내용은 `CpuCreditBalance` 지표 참조). 애플리케이션에 고정 성능이 필요한 경우 `mq.m5.large` 인스턴스 유형을 사용해 봅니다.
	• `mq.m4.large` 브로커와 `mq.t2.micro` 브로커를 생성할 수 있습니다.
	  `mq.t2.micro` 인스턴스 유형 사용에는 기준 수준 이상으로 버스트하는 기능이 있는 CPU 크레딧 및 기준 성능이 적용됩니다(자세한 내용은 `CpuCreditBalance` 지표 참조). 애플리케이션에 고정 성능이 필요한 경우 `mq.m5.large` 인스턴스 유형을 사용해 봅니다.
	• ActiveMQ 5.15.0 브로커 엔진을 사용할 수 있습니다.
	• Amazon MQ REST API 및 AWS SDKs.
	• ActiveMQ가 지원하는 모든 프로그래밍 언어를 사용하고 다음 프로토콜에 대해 명시적으로 TLS를 활성화하여 브로커에 액세스할 수 있습니다.
	  • AMQP
	  • MQTT
	  • MQTT over WebSocket
	  • OpenWire
	  • STOMP
	  • STOMP over WebSocket

Date	설명서 업데이트
	• 다양한 ActiveMQ 클라이언트를 사용하여 ActiveMQ 브로커에 연결할 수 있습니다. ActiveMQ 클라이언트를 사용하는 것이 좋습니다. 자세한 정보는 Connecting a Java application to your broker을 참조하세요. • 브로커가 어떠한 크기의 메시지도 주고 받을 수 있습니다.

기계 번역으로 제공되는 번역입니다. 제공된 번역과 원본 영어의 내용이 상충하는 경우에는 영어 버전이 우선합니다.