分布式消息服务 RabbitMQ

用户指南

天翼云科技有限公司

目　录

1 产品简介

1.1 什么是分布式消息服务 RabbitMQ

分布式消息服务 RabbitMQ 完全兼容开源 RabbitMQ，为您提供即开即用、消息特性丰富、灵活路由、高可用、监控和告警等特性，广泛应用于秒杀、流控、系统解耦等场景。

- 即开即用

  分布式消息服务 RabbitMQ 提供单机和集群的消息实例，拥有丰富内存规格，您可以通过控制台直接下单购买并创建，无需单独准备服务器资源。

- 消息特性丰富

  支持 AMQP 协议，支持普通消息、广播消息、死信、延迟消息等特性。

- 灵活路由

  在 RabbitMQ 中，生产者将消息发送到交换器，由交换器将消息路由到队列中。交换器支持 Direct、Topic、Headers 和 Fanout 四种路由方式，同时支持交换机组合和自定义。

- 高可用

  RabbitMQ 集群提供镜像队列，可通过镜像在其他节点同步数据，单节点宕机时，仍可通过唯一的访问地址对外提供服务，数据不丢失。

- 监控和告警

  支持对 RabbitMQ 实例状态进行监控，支持对集群每个代理的内存、CPU、网络流量等进行监控。如果集群或节点状态异常，将触发告警。

1.2 产品优势

分布式消息服务 RabbitMQ 完全兼容开源社区版本，旨在为您提供便捷高效的消息队列。业务无需改动即可快速迁移上云，为您节省维护和使用成本。

- 一键式部署，免去集群搭建烦恼

  只需要在实例管理界面选好规格配置，提交订单，后台将自动创建部署完成一整套 RabbitMQ 实例。

- 兼容开源，业务零改动迁移上云

  兼容社区版 RabbitMQ 的 API，具备原生 RabbitMQ 的所有消息处理特性。

  业务系统基于开源的 RabbitMQ 进行开发，只需加入少量认证安全配置，即可使用分布式消息服务 RabbitMQ，做到无缝迁移。

说明

RabbitMQ 实例兼容开源社区 RabbitMQ 3.8.35 版本。

- 独占式体验

  RabbitMQ 实例采用物理隔离的方式部署，租户独占 RabbitMQ 实例，每个 RabbitMQ 之间互不影响。

- 高性能

  单队列性能最高可达 10 万 TPS（默认配置），增加队列可获得更高性能。

- 数据安全

  独有的安全加固体系，提供业务操作云端审计，消息存储加密等有效安全措施。

  在网络通信方面，除了提供 SSL 认证，还借助虚拟私有云（VPC）和安全组等加强网络访问控制。

- 无忧运维

  云服务平台提供一整套完整的监控告警等运维服务，故障自动发现和告警，避免 7*24 小时人工值守。RabbitMQ 实例自动上报相关监控指标，如分区数、主题数、堆积消息数等，并支持配置监控数据发送规则，您可以在第一时间通过短信、邮件等获得业务消息队列的运行使用和负载状态。

- 支持多语言客户端

  RabbitMQ 是一款基于 AMQP 协议的开源服务，用于在分布式系统中存储转发消息，服务器端用 Erlang 语言（支持高并发、分布式以及健壮的容错能力等特点）编写，支持多种语言的客户端，如：Python、Ruby、.NET、Java、JMS、C、PHP、ActionScript、XMPP、STOMP 和 AJAX 等。

1.3 典型应用场景

RabbitMQ 作为一款热门的消息队列中间件，具备高效可靠的消息异步传递机制，主要用于不同系统间的数据交流和传递，在企业解决方案、金融支付、电信、电子商务、社交、即时通信、视频、物联网、车联网等众多领域都有广泛应用。

异步通信

将业务中属于非核心或不重要的流程部分，使用消息异步通知的方式发给目标系统，这样主业务流程无需同步等待其他系统的处理结果，从而达到系统快速响应的目的。

如网站的用户注册场景，在用户注册成功后，还需要发送注册邮件与注册短信，这两个流程使用 RabbitMQ 消息服务通知邮件发送系统与短信发送系统，从而提升注册流程的响应速度。

图1-1 串行发送注册邮件与短信流程

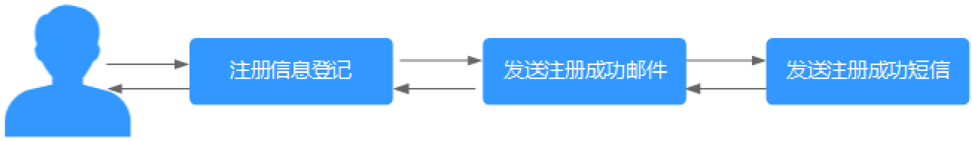
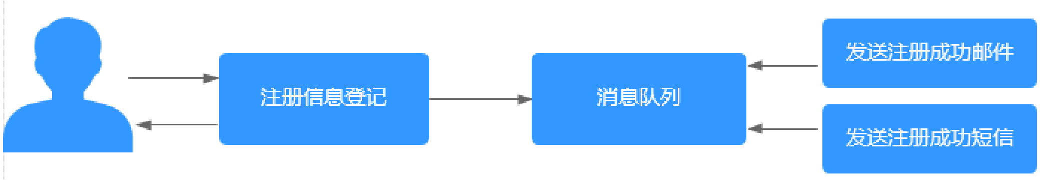


图1-2 借助消息队列异步发送注册邮件与短信流程

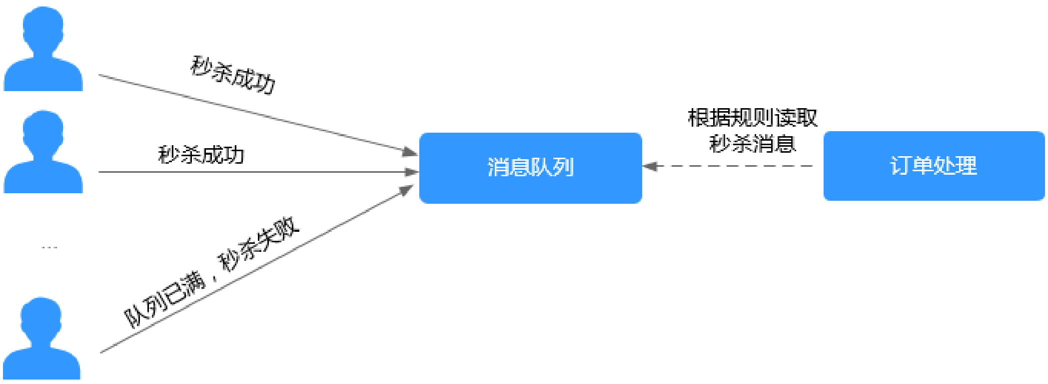


错峰流控与流量削峰

在电子商务系统或大型网站中，上下游系统处理能力存在差异，处理能力高的上游系统的突发流量可能会对处理能力低的某些下游系统造成冲击，需要提高系统的可用性的同时降低系统实现的复杂性。电商大促销等流量洪流突然来袭时，可以通过队列服务堆积缓存订单等信息，在下游系统有能力处理消息的时候再处理，避免下游订阅系统因突发流量崩溃。消息队列提供亿级消息堆积能力，3 天的默认保留时长，消息消费系统可以错峰进行消息处理。

另外，在商品秒杀、抢购等流量短时间内暴增场景中，为了防止后端应用被压垮，可在前后端系统间使用 RabbitMQ 消息队列传递请求。

图1-3 消息队列应对秒杀大流量场景

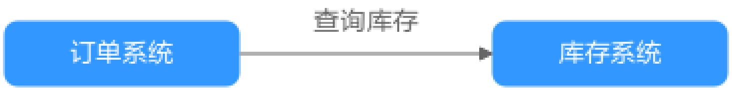


系统解耦

以电商秒杀、抢购等流量短时间内暴增场景为例，传统做法是，用户下单后，订单系统发送查询请求到库存系统，等待库存系统返回请求结果给订单系统。如果库存系统

发生故障，订单系统获取不到数据，订单失败。这种情况下，订单系统和库存系统两个子系统高耦合。

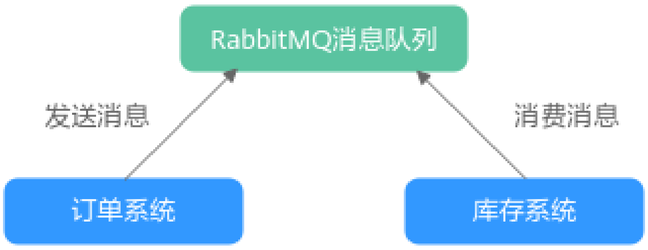
图1-4 系统高耦合



引入 RabbitMQ 消息队列，当用户下单后，将消息写入到 RabbitMQ 消息队列中，然后返回用户下单成功。

库存系统订阅下单的消息，消费下单消息，然后进行库操作。即使库存系统出现故障，也不影响用户下单。

图1-5 系统解耦



高可用

普通队列，由于队列以及队列内容仅存储在单代理上，当该代理故障后，对应的队列不可用。

RabbitMQ 引入镜像队列机制，镜像队列是开源 RabbitMQ 2.6.0 版本新增的一个功能，允许集群将队列镜像到其他代理上，当集群某一代理宕机后，队列能自动切换到镜像中的其他代理，保证服务的可用性。

RabbitMQ 引入仲裁队列机制，仲裁队列是开源 RabbitMQ 3.8 版本新增的一个功能，提供队列复制的能力，当集群某一代理宕机后，队列依旧可以正常运行，保证服务的可用性。

1.4 产品规格

RabbitMQ 实例规格

RabbitMQ 实例兼容开源 RabbitMQ 3.8.35，实例类型包括单机和集群，实例规格请参考表 1-1。

说明

- 为了保证稳定性，**服务端限制了单条消息的最大长度为 50MB，请勿发送大于此长度的消息。**
- 下表中 TPS，是指以 2K 大小的消息为例的每秒处理消息条数，测试场景为不开启持久化的非镜像队列，实时生产实时消费，队列无积压。此数据仅供参考，生产使用需要以实际压测性能为准。
- 服务端的性能主要跟以下因素相关：队列数、消息堆积、连接数、channel、消费者数、镜像队列、优先级队列、消息持久化和 exchange 类型等，在选择实例规格时，请根据业务模型压测结果选择。
- 一条连接最多可以开启 2047 个 channel。

表1-1 RabbitMQ 实例规格

型号	代理数	存储空间范围	TPS 参考值	单个代理最大消费者数	单个代理建议队列数	单个代理最大连接数
rabbitmq.2u4g.single	1	100GB~30000GB	10000	20000	200	3000
rabbitmq.4u8g.single	1	100GB~30000GB	20000	30000	400	4500
rabbitmq.8u16g.single	1	100GB~30000GB	35000	50000	800	7500
rabbitmq.16u32g.single	1	100GB~30000GB	45000	80000	1600	12000
rabbitmq.24u48g.single	1	100GB~30000GB	50000	100000	2400	15000
rabbitmq.2u4g.cluster	3/5/7	3/5/7*100GB~30000GB	30000~70000	20000	200	3000
rabbitmq.4u8g.cluster	3/5/7	3/5/7*100GB~30000GB	45000~80000	30000	400	4500
rabbitmq.8u16g.cluster	3/5/7	3/5/7*100GB~30000GB	85000~120000	50000	800	7500
rabbitmq.12u24g.cluster	3/5/7	3/5/7*100GB~30000GB	100000~150000	60000	1200	10000
rabbitmq.16u32g.cluster	3/5/7	3/5/7*100GB~30000GB	130000~180000	80000	1600	12000
rabbitmq.24u48g.cluster	3/5/7	3/5/7*100GB~30000GB	150000~200000	100000	2400	15000

RabbitMQ 实例的存储空间估算参考

在集群模式中，RabbitMQ 需要对消息持久化写入到磁盘中，因此，您在创建 RabbitMQ 实例选择存储空间时，建议根据业务消息体积预估以及镜像队列副本数量选择合适的存储空间。镜像队列副本数最大为集群的代理数。

例如：业务消息体积预估 100GB，则磁盘容量最少应为 100GB*镜像队列副本数+预留磁盘大小 100GB。

如果是单机实例，则是计算业务消息体积+预留磁盘大小即可。

当前 RabbitMQ 实例支持修改集群实例的代理个数，您可以根据业务情况，随时更改集群代理个数。单机实例暂不支持变更规格。

1.5 与 Kafka、RocketMQ 的差异

功能项	RocketMQ	Kafka	RabbitMQ
优先级队列	不支持	不支持	支持。建议优先级大小设置在 0-10 之间。
延迟队列	支持	不支持	支持
死信队列	支持	不支持	支持
消息重试	支持	不支持	不支持
消费模式	支持客户端主动拉取和服务端推送两种方式	客户端主动拉取	支持客户端主动拉取以及服务端推送两种模式
广播消费	支持	支持	支持
消息回溯	支持	支持。Kafka 支持按照 offset 和 timestamp 两种维度进行消息回溯。	不支持。RabbitMQ 中消息一旦被确认消费就会被标记删除。
消息堆积	支持	支持。考虑吞吐因素，Kafka 的堆积效率比 RabbitMQ 总体上要高。	支持
持久化	支持	支持	支持
消息追踪	支持	不支持	支持。RabbitMQ 中可以采用 Firehose 或者 rabbitmq_tracing 插件实现，但开启 rabbitmq_tracing 插件会影响性能，建议只在定位问题过程中开启。
消息过滤	支持	支持	不支持，但可以自行封装。
多租户	支持	不支持	支持

功能项	RocketMQ	Kafka	RabbitMQ
多协议支持	兼容 RocketMQ 协议	只支持 Kafka 自定义协议。	RabbitMQ 基于 AMQP 协议实现，同时支持 MQTT、STOMP 等协议。
跨语言支持	支持多语言的客户端	采用 Scala 和 Java 编写，支持多种语言的客户端。	采用 Erlang 编写，支持多种语言的客户端。
流量控制	待规划	支持 client 和 user 级别，通过主动设置可将流控作用于生产者或消费者。	RabbitMQ 的流控基于 Credit-Based 算法，是内部被动触发的保护机制，作用于生产者层面。
消息顺序性	单队列（queue）内有序	支持单分区（partition）级别的顺序性。	不支持。需要单线程发送、单线程消费并且不采用延迟队列、优先级队列等一些高级功能整体配合，才能实现消息有序。
安全机制	支持 SSL 认证	支持 SSL、SASL 身份认证和读写权限控制。	与 Kafka 相似
事务性消息	支持	支持	支持

1.6 与其他云服务的关系

- 弹性云主机（Elastic Cloud Server）

  弹性云主机是由 CPU、内存、操作系统、云硬盘组成的基础的计算组件。RabbitMQ 实例运行在弹性云主机上，一个代理对应一台弹性云主机。

- 云硬盘（Elastic Volume Service）

  云硬盘为云服务器提供块存储服务，RabbitMQ 的所有数据（如消息和日志等）都保存在云硬盘中。

- 云审计（Cloud Trace Service）

  云审计为您提供云服务资源的操作记录，记录内容包括您从云服务平台管理控制台或者开放 API 发起的云服务资源操作请求以及每次请求的结果，供您查询、审计和回溯使用。

- 虚拟私有云

  RabbitMQ 实例运行于虚拟私有云，需要使用虚拟私有云创建的 IP 和带宽。通过虚拟私有云安全组的功能可以增强访问 RabbitMQ 实例的安全性。

- 云监控（Cloud Eye）

  云监控是一个开放性的监控平台，提供资源的实时监控、告警、通知等服务。

说明

RabbitMQ 实例向 Cloud Eye 上报监控数据的更新周期为 1 分钟。

- 弹性公网 IP（Elastic IP）

  弹性公网 IP 提供独立的公网 IP 资源，包括公网 IP 地址与公网出口带宽服务。RabbitMQ 实例绑定弹性公网 IP 后，可以通过公网访问 RabbitMQ 实例。
- 标签管理服务（Tag Management Service）

  标签管理服务是一种快速便捷将标签集中管理的可视化服务，提供跨区域、跨服务的集中标签管理和资源分类功能。

  为 RabbitMQ 实例添加标签，可以方便用户识别和管理拥有的实例资源。

1.7 约束与限制

分布式消息服务 RabbitMQ 在某些功能做了约束和限制，如表 1-2 所示。

表1-2 RabbitMQ 使用约束和限制

限制项	约束和限制	描述
版本	当前服务端版本为 3.8.35	兼容 AMQP 0-9-1 协议的客户端版本。
连接数	RabbitMQ 单机和集群实例，不同实例规格的连接数上限不一致，具体限制，请参考产品规格。	-
通道数	<= 2047	单条连接可以建立的通道数。
消息大小	单条消息的最大长度为 50MB	服务端限制了单条消息的最大长度为 50MB，请勿发送大于此长度的消息，否则生产失败。
内存高水位阈值	<= 40%	内存使用率超过 40%会触发内存高水位，生产者流程被阻塞
磁盘高水位阈值	>= 5GB	磁盘剩余空间低于 5GB 会触发磁盘高水位，生产者流程被阻塞
cluster_partition_handling	pause_minority	当集群发生网络分区时，代理会检查自己是否处于“少数派”（存储分区的代理数小于等于总代理数的一半称为少数派）。少数派中的代理将会自动关闭服务并定期检测网络状态，待分区恢复之后重新启动服务。如果未开启镜像队列，发生分区时少数派上的队列将无法生产消费。 此策略相当于放弃了可用性而选

限制项	约束和限制	描述
		择了数据一致性。
rabbitmq_delayed_message_exchange	插件延迟时间存在 1%左右的误差，可能提前或者推迟发送消息给消费者。	实例是否开启消息延迟功能。
RabbitMQ 插件	RabbitMQ 插件功能可用于测试和迁移业务等场景，不建议用于生产业务。	RabbitMQ 实例主要提供 AMQP 0-9-1 业务消息的功能，兼容相关协议的 Vhost、Exchange 等组件。插件相关内容为非核心功能，不建议用于生产业务。

1.8 RabbitMQ 相关概念

云服务平台使用 RabbitMQ 作为消息引擎，RabbitMQ 是一个生产者和消费者模型，主要负责接收、存储和转发消息。以下概念基于 RabbitMQ 进行描述。

消息

消息一般分为两部分，消息体和标签，标签主要用来描述这条消息，消息体是消息的内容，是一个 JSON 体或者数据等。

生产者发送消息，消费者消费消息，生产者与消费者彼此并无直接关系。

生产者（Producer）

即向队列发送消息的一方。发布消息的最终目的在于将消息内容传递给其他系统/模块，使对方按照约定处理该消息。

消费者（Consumer）

接收消息的一方。消费者订阅 RabbitMQ 的队列，当消费者消费一条消息时，只是消费消息的消息体。在消息路由的过程中，会丢弃标签，存入到队列中的只有消息体。

队列（Queue）

队列是用于存储消息的，生产者将消息送到队列，消费者从队列中获取和消费消息。多个消费者可以同时订阅同一个队列，队列里的消息分配给不同的消费者。

代理（Broker）

消息中间件的服务节点。

1.9 权限管理

如果您需要对云服务平台上购买的 DMS for RabbitMQ 资源，给企业中的员工设置不同的访问权限，以达到不同员工之间的权限隔离，您可以使用统一身份认证服务（Identity and Access Management，简称 IAM）进行精细的权限管理。该服务提供用户身份认证、权限分配、访问控制等功能，可以帮助您安全的控制资源的访问。

通过 IAM，您可以在帐号中给员工创建 IAM 用户，并使用策略来控制他们对资源的访问范围。例如您的员工中有负责软件开发的人员，您希望他们拥有 DMS for RabbitMQ 的使用权限，但是不希望他们拥有删除 RabbitMQ 实例等高危操作的权限，那么您可以使用 IAM 为开发人员创建用户，通过授予仅能使用 DMS for RabbitMQ，但是不允许删除 RabbitMQ 实例的权限策略，控制他们对 DMS for RabbitMQ 资源的使用范围。

如果帐号已经能满足您的要求，不需要创建独立的 IAM 用户进行权限管理，您可以跳过本章节，不影响您使用 DMS for RabbitMQ 服务的其它功能。

IAM 是云服务平台提供权限管理的基础服务，无需付费即可使用，您只需要为您帐号中的资源进行付费。关于 IAM 的详细介绍，请参见《IAM 产品介绍》。

DMS for RabbitMQ 权限

默认情况下，管理员创建的 IAM 用户没有任何权限，需要将其加入用户组，并给用户组授予策略或角色，才能使得用户组中的用户获得对应的权限，这一过程称为授权。授权后，用户就可以基于被授予的权限对云服务进行操作。

DMS for RabbitMQ 部署时通过物理区域划分，为项目级服务。授权时，“作用范围”需要选择“区域级项目”，然后在指定区域对应的项目中设置相关权限，并且该权限仅对此项目生效；如果在“所有项目”中设置权限，则该权限在所有区域项目中都生效。访问 DMS for RabbitMQ 时，需要先切换至授权区域。

权限根据授权精细程度分为角色和策略。

- 角色：IAM 最初提供的一种根据用户的工作职能定义权限的粗粒度授权机制。该机制以服务为粒度，提供有限的服务相关角色用于授权。由于云服务平台各服务之间存在业务依赖关系，因此给用户授予角色时，可能需要一并授予依赖的其他角色，才能正确完成业务。角色并不能满足用户对精细化授权的要求，无法完全达到企业对权限最小化的安全管控要求。
- 策略：IAM 最新提供的一种细粒度授权的能力，可以精确到具体服务的操作、资源以及请求条件等。基于策略的授权是一种更加灵活的授权方式，能够满足企业对权限最小化的安全管控要求。例如：针对 DMS for RabbitMQ 服务，管理员能够控制 IAM 用户仅能对实例进行指定的管理操作。

说明

DMS for RabbitMQ 的权限与策略基于分布式消息服务 DMS，因此在 IAM 服务中为 RabbitMQ 分配用户与权限时，请选择并使用“DMS”的权限与策略。

如表 1-3 所示，包括了 DMS for RabbitMQ 的所有系统权限。

表1-3 DMS for RabbitMQ 系统权限

系统角色/策略名称	描述	类别	依赖关系
DMS FullAccess	分布式消息服务管理员权限，拥有该权限的用户可以操作所有分布式消息服务的功能。	系统策略	无
DMS UserAccess	分布式消息服务普通用户权限（没有实例创建、修改、删除、扩容）。	系统策略	无
DMS ReadOnlyAccess	分布式消息服务的只读权限，拥有该权限的用户仅能查看分布式消息服务数据。	系统策略	无
DMS Administrator	分布式消息服务的管理员权限。	系统角色	依赖 Tenant Guest 和 VPC Administrator。

表 2 列出了 DMS for RabbitMQ 常用操作与系统策略的授权关系，您可以参照该表选择合适的系统策略。

表1-4 常用操作与系统策略的关系

操作	DMS FullAccess	DMS UserAccess	DMS ReadOnlyAccess
创建实例	√	×	×
按需转包周期	√	×	×
修改实例	√	×	×
删除实例	√	×	×
变更实例规格	√	×	×
重启实例	√	√	×
查询实例信息	√	√	√

2 快速入门

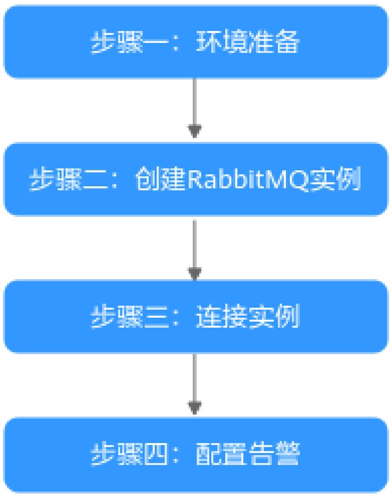
2.1 入门指导

本文将为您介绍分布式消息服务 RabbitMQ 版入门的基本流程，主要包括控制台创建 RabbitMQ 实例、使用弹性云主机连接实例的操作，帮助您快速上手 RabbitMQ。

您还可以通过 API 方式创建 RabbitMQ 实例。

操作流程

图2-1 RabbitMQ 使用流程



1. 环境准备

   RabbitMQ 实例运行于虚拟私有云中，在创建实例前需要确保有可用的虚拟私有云。

2. 创建 RabbitMQ 实例

   在创建实例时，您可以选择是否开启 SSL 访问，开启后，数据加密传输，安全性更高。同时，SSL 开关只能在创建实例时设置，实例创建成功后，不支持修改。

3. 连接实例

客户端连接实例，根据实例是否开启 SSL 开关，存在以下两种场景：不使用 SSL 证书连接和使用 SSL 证书连接。

4. 配置告警

   配置 RabbitMQ 实例监控告警策略，监控实际业务运行状态。

说明

关于 RabbitMQ 的相关概念，请参考 RabbitMQ 相关概念。

2.2 步骤一：准备环境

虚拟私有云

虚拟私有云（Virtual Private Cloud，以下简称 VPC）为 RabbitMQ 实例提供一个隔离的、用户自主配置和管理的虚拟网络环境。

步骤 1 在创建 RabbitMQ 实例前，确保已存在可用的虚拟私有云和子网。

创建方法，请参考《虚拟私有云-用户手册》的“创建虚拟私有云和子网”。如果您已有虚拟私有云和子网，可重复使用，不需要多次创建。

在创建 VPC 和子网时应注意如下要求：

- 创建的 VPC 与使用的 RabbitMQ 服务应在相同的区域。
- 创建 VPC 和子网时，建议配置参数使用默认配置。

步骤 2 在创建 RabbitMQ 实例前，确保已存在可用的安全组。

创建方法，请参考《虚拟私有云-用户手册》的“创建安全组”。如果您已有安全组，可重复使用，不需要多次创建。

使用 RabbitMQ 实例前，添加表 2-1 所示安全组规则，其他规则请根据实际需要添加。

表2-1 安全组规则

方向	协议	端口	源地址	说明
入方向	TCP	5672	0.0.0.0/0	访问 RabbitMQ 实例（关闭 SSL 加密）
入方向	TCP	5671	0.0.0.0/0	访问 RabbitMQ 实例（开启 SSL 加密）
入方向	TCP	15672	0.0.0.0/0	访问 Web 界面 UI 地址（关闭 SSL 加密）
入方向	TCP	15671	0.0.0.0/0	访问 Web 界面 UI 地址（开启 SSL 加密）

说明

创建安全组后，系统默认添加入方向“允许安全组内的弹性云主机彼此通信”规则和出方向“放通全部流量”规则，此时使用内网通过同一个 VPC 访问 RabbitMQ 实例，无需添加表 2-1 的规则。

----结束

（可选）弹性 IP 地址

如果需要通过公网访问 RabbitMQ 实例，请提前准备弹性 IP 地址。

创建方法，请参考申请弹性公网 IP。

在创建弹性 IP 地址时应注意如下要求：创建的弹性 IP 地址与 RabbitMQ 实例在相同的区域。

弹性云主机

在连接 RabbitMQ 实例之前，需要先购买弹性云主机（Elastic Cloud Server，以下简称 ECS），JDK 安装以及环境变量配置。本文以 Linux 系统的 ECS 为例，Windows 系统 ECS 的 JDK 安装与环境变量配置可自行在互联网查找相关帮助。

步骤 1 登录管理控制台，选择“计算 > 弹性云主机”，创建一个 ECS 实例。

具体购买操作，请参考《天翼云弹性云主机用户使用指南》的“创建弹性云主机”章节。如果您已有可用的 ECS，可重复使用，不需要再次购买。

步骤 2 登录弹性云主机。

步骤 3 安装 Java JDK 或 JRE，并配置 JAVA_HOME 与 PATH 环境变量，使用执行用户在用户家目录下修改.bash_profile，添加如下行。其中“/opt/java/jdk1.8.0_151”为 JDK 的安装路径，请根据实际情况修改。

```
export JAVA_HOME=/opt/java/jdk1.8.0_151
export PATH=$JAVA_HOME/bin:$PATH
```

执行 **source .bash_profile** 命令使修改生效。

说明

ECS 默认自带的 JDK 可能不符合要求，例如 OpenJDK，需要配置为 Oracle 的 JDK，可至 Oracle 官方下载页面下载 Java Development Kit 1.8.111 及以上版本。

----结束

2.3 步骤二：创建 RabbitMQ 实例

RabbitMQ 是一款基于 AMQP 协议的开源服务，用于在分布式系统中存储转发消息，服务器端用 Erlang 语言（支持高并发、分布式以及健壮的容错能力等特点）编写，支持多种语言的客户端，如：Python、Ruby、.NET、Java、JMS、C、PHP、ActionScript、XMPP、STOMP、AJAX 等。

AMQP，即 Advanced Message Queuing Protocol，高级消息队列协议，是应用层的一个开放标准协议。

前提条件

在创建 RabbitMQ 实例前，需要保证存在可使用的虚拟私有云。创建方法，请参考《虚拟私有云-用户手册》的“创建虚拟私有云和子网”。

如果您已有虚拟私有云，可重复使用，不需要多次创建。

操作步骤

步骤 1 登录分布式消息服务 RabbitMQ 控制台，单击页面右上方的“购买 RabbitMQ 实例”。

步骤 2 选择“计费模式”、“区域”、“项目”和“可用区”。

步骤 3 设置“实例名称”和“企业项目”。

步骤 4 设置实例信息。

1. 版本：RabbitMQ 的版本号，当前仅支持 RabbitMQ 3.8.35。
2. 实例类型：支持“单机”和“集群”。
   - 单机：表示部署一个 RabbitMQ 代理。
   - 集群：表示部署多个 RabbitMQ 代理，实现高可靠的消息存储。
3. CPU 架构：当前仅支持“x86 计算”，保持默认值即可。
4. 代理规格：根据实际情况选择规格。

说明

为了保证服务的稳定可靠，RabbitMQ 采用了默认的 40%高水位配置。当内存占用率到达 40%高水位后，会触发流控，生产者发送消息会被阻塞。为了避免高水位的产生，请及时消费积压在队列中的消息。

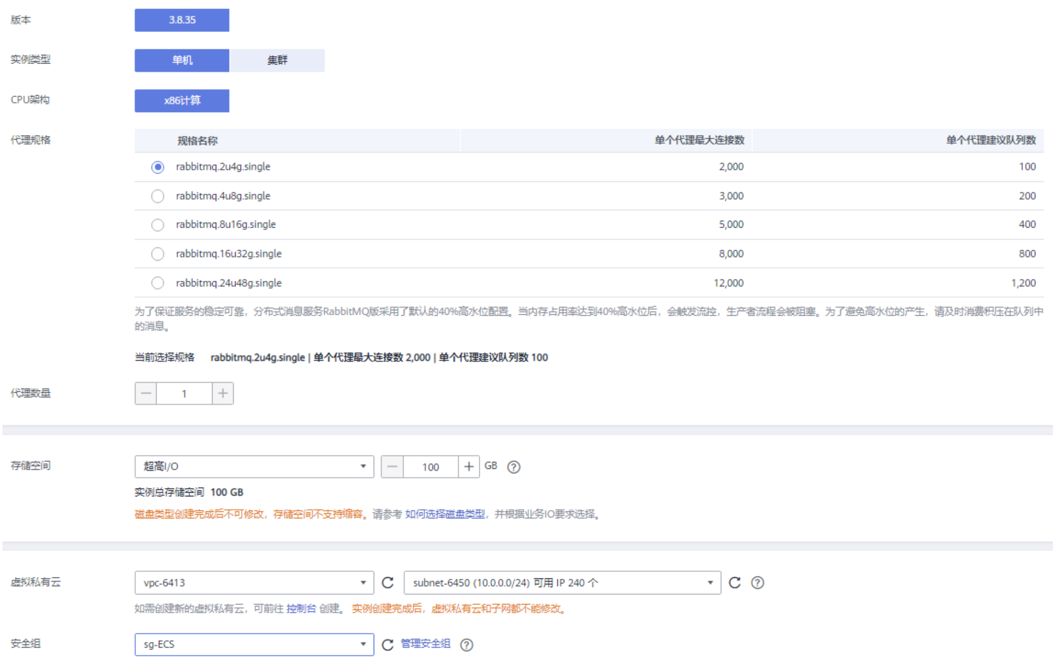
5. 代理个数：根据实例情况选择代理个数。
6. 存储空间：选择 RabbitMQ 实例的磁盘类型和储存空间总量。

   如何选择磁盘类型请参考《天翼云云硬盘用户使用指南》的“简介 > 磁盘类型及性能介绍”章节。
   - 单机实例的取值范围：100GB～30000GB
   - 集群实例的取值范围为：100GB*代理数 ～30000GB*代理数
7. 虚拟私有云：选择已经创建好的虚拟私有云和子网。

   虚拟私有云可以为您的 RabbitMQ 实例构建隔离的、能自主配置和管理的虚拟网络环境。
8. 安全组：选择已经创建好的安全组。

   安全组是一组对 RabbitMQ 实例的访问规则的集合。您可以单击右侧的“管理安全组”，跳转到网络控制台的安全组页面，查看或创建安全组。

图2-2 设置实例信息



步骤 5 设置连接 RabbitMQ 实例的用户名和密码。

步骤 6 设置实例购买时长。

当选择了“包年/包月”计费模式时，页面才显示“购买时长”参数，您需要根据业务需要选择。

步骤 7 单击“更多配置”，设置更多相关信息。

1. 设置“公网访问”。

   您可以选择是否打开公网访问开关。

   如果选择了开启，表示访问 RabbitMQ 实例可以通过弹性 IP 访问。这时页面会显示“弹性 IP 地址”，在“弹性 IP 地址”区域，您可下拉选择已有的弹性 IP。另外，您可单击右侧的“创建弹性 IP”，跳转到网络控制台的弹性公网 IP 页面，购买弹性公网 IP。

   图2-3 购买 RabbitMQ 实例时公网访问设置

说明

- 公网访问与 VPC 内访问相比，可能存在网络丢包和抖动等情况，且访问时延有所增加，因此建议仅在业务开发测试阶段开启公网访问进行调试。
- 如果用户在虚拟私有云的服务页面手动解绑定或删除 EIP，相应 RabbitMQ 实例的公网访问功能会自动关闭。

2. 设置“SSL”。

   客户端连接实例时 SSL 认证的开关。开启 SSL，则数据加密传输，安全性更高。

   SSL 开关在实例创建完成后不支持修改，请明确是否需要开启。

3. 设置“标签”。

   标签用于标识云资源，当您拥有相同类型的许多云资源时，可以使用标签按各种维度（例如用途、所有者或环境）对云资源进行分类。

   – 如果您已经预定义了标签，在“标签键”和“标签值”中选择已经定义的标签键值对。另外，您可以单击右侧的“查看预定义标签”，系统会跳转到标签管理服务页面，查看已经预定义的标签，或者创建新的标签。
   – 您也可以直接在“标签键”和“标签值”中设置标签。

   当前每个 RabbitMQ 实例最多支持设置 20 个不同标签。

4. 设置实例的描述信息。

步骤 8 填写完上述信息后，单击“立即购买”，进入“规格确认”页面。

步骤 9 确认实例信息无误后，提交请求。

步骤 10 在实例列表页面查看实例是否创建成功。

创建实例大约需要 3 到 15 分钟，此时实例的“状态”为“创建中”。

- 当实例的“状态”变为“运行中”时，说明实例创建成功。
- 如果创建实例失败，在信息栏的“创建失败任务”中查看创建失败的实例。请删除创建失败的 RabbitMQ 实例，然后重新创建。如果重新创建仍然失败，请联系客服。

----结束

2.4 步骤三：连接实例生产消费消息

2.4.1 不使用 SSL 证书连接

本节以 demo 为例，介绍 VPC 内访问与使用 RabbitMQ 的方法，假设 RabbitMQ 客户端部署在弹性云主机上。

RabbitMQ 实例兼容开源协议，如果在业务代码中连接 RabbitMQ 实例，请参考 RabbitMQ 官网提供的不同语言的连接和使用向导：https://www.rabbitmq.com/getstarted.html

前提条件

- 参考创建实例章节创建 RabbitMQ 实例，并记录创建时输入的用户名和密码。
- 创建完成后，单击实例名称，查看并记录实例详情中的“内网连接地址/公网连接地址”。
- 已创建弹性云主机，并且弹性云主机的 VPC、子网、安全组与 RabbitMQ 实例的 VPC、子网、安全组保持一致。
- 已完成 JDK 安装以及环境变量配置，具体操作请参考准备环境。

命令行模式连接实例

步骤 1 下载 RabbitMQ-Tutorial.zip 示例工程代码。

```
$ wget https://obs.ctyun.cn/dms-demo/RabbitMQ-Tutorial.zip
```

步骤 2 解压 RabbitMQ-Tutorial.zip 压缩包。

```
$ unzip RabbitMQ-Tutorial.zip
```

步骤 3 进入 RabbitMQ-Tutorial 目录，该目录下包含预编译好的 jar 文件。

```
$ cd RabbitMQ-Tutorial
```

步骤 4 运行生产消息示例。

```
$ java -cp .:rabbitmq-tutorial.jar Send host port user password
```

其中，host 表示 RabbitMQ 实例的连接地址，port 为 RabbitMQ 实例的监听端口（默认为 5672），user 表示 RabbitMQ 用户名，password 表示用户名对应的密码。

图2-4 运行生产消息示例

```
[root@rabbitmq-0004 RabbitMQ-Tutorial]# java .:rabbitmq-tutorial.jar Send 192.168.0.37 5672 admin admin
[root@rabbitmq-0004 RabbitMQ-Tutorial]# java -cp .:rabbitmq-tutorial.jar Send 192.168.0.37 5672 admin admin
 [x] Sent 'Hello World!'
[root@rabbitmq-0004 RabbitMQ-Tutorial]# java -cp .:rabbitmq-tutorial.jar Send 192.168.0.37 5672 admin admin
 [x] Sent 'Hello World!'
[root@rabbitmq-0004 RabbitMQ-Tutorial]# java -cp .:rabbitmq-tutorial.jar Send 192.168.0.37 5672 admin admin
 [x] Sent 'Hello World!'
[root@rabbitmq-0004 RabbitMQ-Tutorial]# java -cp .:rabbitmq-tutorial.jar Send 192.168.0.37 5672 admin admin
 [x] Sent 'Hello World!'
```

使用 **Ctrl+C** 命令退出。

步骤 5 运行消费消息示例。

```
$ java -cp .:rabbitmq-tutorial.jar Recv host port user password
```

其中，host 表示 RabbitMQ 实例的连接地址，port 为 RabbitMQ 实例的监听端口（默认为 5672），user 表示 RabbitMQ 用户名，password 表示用户名对应的密码。

图2-5 运行消费消息示例

```
[root@rabbitmq-0004 RabbitMQ-Tutorial]# java -cp .:rabbitmq-tutorial.jar Recv 192.168.0.37 5672 admin admin
 [*] Waiting for messages. To exit press CTRL+C
 [x] Received 'Hello World!'
 [x] Received 'Hello World!'
 [x] Received 'Hello World!'
 [x] Received 'Hello World!'
```

如需停止消费使用 **Ctrl+C** 命令退出。

----结束

示例代码（Java）

连接实例并生产消息

```
ConnectionFactory factory = new ConnectionFactory();
 factory.setHost(host);
 factory.setPort(port);

 factory.setUsername(user);
 factory.setPassword(password);
 Connection connection = factory.newConnection();
 Channel channel = connection.createChannel();

 channel.queueDeclare(QUEUE_NAME, false, false, false, null);

 String message = "Hello World!";
 channel.basicPublish("", QUEUE_NAME, null, message.getBytes("UTF-8"));
 System.out.println(" [x] Sent '" + message + "'");

 channel.close();
 connection.close();
```

连接实例并消费消息

```
ConnectionFactory factory = new ConnectionFactory();
 factory.setHost(host);
 factory.setPort(port);
 factory.setUsername(user);
 factory.setPassword(password);
 Connection connection = factory.newConnection();
 Channel channel = connection.createChannel();

 channel.queueDeclare(QUEUE_NAME, false, false, false, null);
 System.out.println(" [*] Waiting for messages. To exit press CTRL+C");

 Consumer consumer = new DefaultConsumer(channel)
 {
    @Override
    public void handleDelivery(String consumerTag, Envelope envelope,
AMQP.BasicProperties properties,
           byte[] body)
           throws IOException
    {
        String message = new String(body, "UTF-8");
        System.out.println(" [x] Received '" + message + "'");
    }
 };
 channel.basicConsume(QUEUE_NAME, true, consumer);
```

2.4.2 使用 SSL 证书连接

创建实例时开启 SSL 访问，则数据加密传输，安全性更高。

本节以 demo 为例，介绍 VPC 内访问与使用 RabbitMQ 的方法，假设 RabbitMQ 客户端部署在弹性云主机上。

RabbitMQ 实例兼容开源协议，如果在业务代码中连接 RabbitMQ 实例，请参考 RabbitMQ 官网提供的不同语言的连接和使用向导：https://www.rabbitmq.com/getstarted.html

前提条件

- 参考创建实例章节创建 RabbitMQ 实例，并记录创建时输入的用户名和密码。
- 创建完成后，单击实例名称，查看并记录实例详情中的“内网连接地址/公网连接地址”。
- 已创建弹性云主机，并且弹性云主机的 VPC、子网、安全组与 RabbitMQ 实例的 VPC、子网、安全组保持一致。
- 已完成 JDK 安装以及环境变量配置，具体操作请参考准备环境。

命令行模式连接实例

步骤 1 下载 RabbitMQ-Tutorial-SSL.zip 示例工程代码。

```
$ wget https://obs.ctyun.cn/dms-demo/RabbitMQ-Tutorial.zip
```

步骤 2 解压 RabbitMQ-Tutorial-SSL.zip 压缩包。

```
$ unzip RabbitMQ-Tutorial-SSL.zip
```

步骤 3 进入 RabbitMQ-Tutorial-SSL 目录，该目录下包含预编译好的 jar 文件。

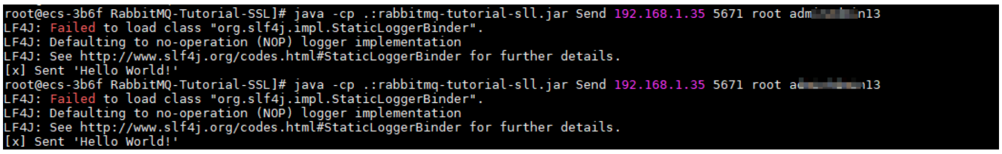
```
$ cd RabbitMQ-Tutorial-SSL
```

步骤 4 运行生产消息示例。

```
$ java -cp .:rabbitmq-tutorial-sll.jar Send host port user password
```

其中，host 表示 RabbitMQ 实例的连接地址，port 为 RabbitMQ 实例的监听端口（默认为 5671），user 表示 RabbitMQ 用户名，password 表示用户名对应的密码。

图2-6 生产消息示例



```
root@ecs-3b6f RabbitMQ-Tutorial-SSL]# java -cp .:rabbitmq-tutorial-sll.jar Send 192.168.1.35 5671 root adm[redacted]n13
LF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
LF4J: Defaulting to no-operation (NOP) logger implementation
LF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
[x] Sent 'Hello World!'
root@ecs-3b6f RabbitMQ-Tutorial-SSL]# java -cp .:rabbitmq-tutorial-sll.jar Send 192.168.1.35 5671 root a[redacted]n13
LF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
LF4J: Defaulting to no-operation (NOP) logger implementation
LF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
[x] Sent 'Hello World!'
```

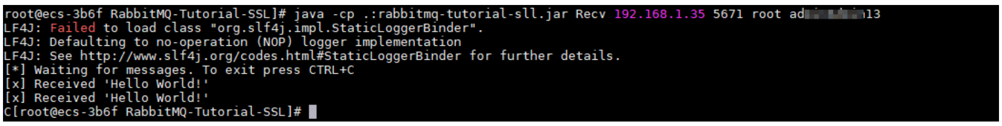
使用 **Ctrl+C** 命令退出。

步骤 5 运行消费消息示例。

```
$ java -cp .:rabbitmq-tutorial-sll.jar Recv host port user password
```

其中，host 表示 RabbitMQ 实例的连接地址，port 为 RabbitMQ 实例的监听端口（默认为 5671），user 表示 RabbitMQ 用户名，password 表示用户名对应的密码。

图2-7 消费消息示例



```
root@ecs-3b6f RabbitMQ-Tutorial-SSL]# java -cp .:rabbitmq-tutorial-sll.jar Recv 192.168.1.35 5671 root ad[redacted]13
LF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
LF4J: Defaulting to no-operation (NOP) logger implementation
LF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
[*] Waiting for messages. To exit press CTRL+C
[x] Received 'Hello World!'
[x] Received 'Hello World!'
C[root@ecs-3b6f RabbitMQ-Tutorial-SSL]#
```

如需停止消费使用 **Ctrl+C** 命令退出。

----结束

示例代码（Java）

连接实例并生产消息

```
ConnectionFactory factory = new ConnectionFactory();
 factory.setHost(host);
 factory.setPort(port);

 factory.setUsername(user);
 factory.setPassword(password);
 factory.useSslProtocol();
 Connection connection = factory.newConnection();
 Channel channel = connection.createChannel();

 channel.queueDeclare(QUEUE_NAME, false, false, false, null);

 String message = "Hello World!";
 channel.basicPublish("", QUEUE_NAME, null, message.getBytes("UTF-8"));
 System.out.println(" [x] Sent '" + message + "'");

 channel.close();
 connection.close();
```

连接实例并消费消息

```
ConnectionFactory factory = new ConnectionFactory();
 factory.setHost(host);
 factory.setPort(port);
 factory.setUsername(user);
 factory.setPassword(password);
 factory.useSslProtocol();
 Connection connection = factory.newConnection();
 Channel channel = connection.createChannel();

 channel.queueDeclare(QUEUE_NAME, false, false, false, null);
 System.out.println(" [*] Waiting for messages. To exit press CTRL+C");

 Consumer consumer = new DefaultConsumer(channel)
 {
     @Override
     public void handleDelivery(String consumerTag, Envelope envelope,
AMQP.BasicProperties properties,
             byte[] body)
             throws IOException
     {
         String message = new String(body, "UTF-8");
         System.out.println(" [x] Received '" + message + "'");
     }
 };
 channel.basicConsume(QUEUE_NAME, true, consumer);
```

2.5 步骤四：配置必须的监控告警

本章节主要介绍部分监控指标的告警策略，以及配置操作。在实际业务中，建议按照以下告警策略，配置监控指标的告警规则。

表2-2 RabbitMQ 实例配置告警的指标

指标名称	告警策略	指标说明	解决方案
内存高水位状态	告警阈值：原始值>=1 连续触发次数：1 告警级别：致命	告警阈值为 1 表示触发内存高水位，会阻塞消息生产	• 加快消费 • 采用生产者确认的发送模式，并监控生产端消息生产速度和时长，当消息生产时长有明显增加时进行流控措施
磁盘高水位状态	告警阈值：原始值>=1 连续触发次数：1 告警级别：致命	告警阈值为 1 表示触发磁盘高水位，会阻塞消息生产	• 减少惰性队列的消息堆积 • 减少持久化队列的消息堆积 • 删除队列
内存使用率	告警阈值：原始值>业务预期使用率（推荐 30%） 连续触发次数：连续 3~5 个周期 告警级别：重要	该指标需要分别为每个节点设置内存使用率告警，避免触发内存高水位阻塞生产	• 加快消费 • 采用生产者确认的发送模式，并监控生产端消息生产速度和时长，当消息生产时长有明显增加时进行流控措施
CPU 使用率	告警阈值：原始值>业务预期使用率（推荐 70%） 连续触发次数：连续 3~5 个周期 告警级别：重要	该指标需要分别为每个节点设置 CPU 使用率告警，CPU 使用率过高可能会影响生产速度	• 减少镜像队列个数 • 对于集群实例，建议扩容节点个数，然后进行节点间重平衡
可消费消息数	告警阈值：原始值>业务预期可消费消息数 连续触发次数：1 告警级别：重要	可消费消息数过多表示消息堆积	请参考消息堆积的解决办法
未确认消息数	告警阈值：原始值>业务预期未确认消息数 连续触发次数：1	未确认消息数过多可能会导致消息堆积	• 检查消费者是否异常 • 检查消费者逻辑是否消耗时间过长

指标名称	告警策略	指标说明	解决方案
	告警级别：重要		
连接数	告警阈值：原始值>业务预期连接数 连续触发次数：1 告警级别：重要	连接数突增可能是流量变大的预警	需检查业务是否正常，可参考其他告警
通道数	告警阈值：原始值>业务预期通道数 连续触发次数：1 告警级别：重要	通道数突增可能是流量变大的预警	需检查业务是否正常，可参考其他告警
Erlang 进程数	告警阈值：原始值>业务预期进程数 连续触发次数：1 告警级别：重要	进程数突增可能是流量变大的预警	需检查业务是否正常，可参考其他告警

说明

- 告警阈值请根据业务预期数设置。例如，业务预期使用率 35%，则告警阈值设置 35%。
- 连续触发次数和告警级别可根据业务逻辑自行调整。

操作步骤

步骤 1 登录管理控制台。

步骤 2 在管理控制台左上角单击 ，选择区域。

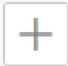
说明

此处请选择 RabbitMQ 实例所在的区域。

步骤 3 在管理控制台左上角单击 ，选择“应用服务 > 分布式消息服务 RabbitMQ”，进入分布式消息服务 RabbitMQ 专享版页面。

步骤 4 通过以下任意一种方法，查看监控数据。

- 在 RabbitMQ 实例名称后，单击 。跳转到云监控页面，查看实例、节点和队列的监控数据，数据更新周期为 1 分钟。
- 单击 RabbitMQ 实例名称，进入实例详情页。在左侧导航栏单击“监控”，进入监控页面，查看实例、节点和队列的监控数据，数据更新周期为 1 分钟。

步骤 5 在实例监控指标页面中，找到需要创建告警的指标项，鼠标移动到指标区域，然后单击指标右上角的，进入“创建告警规则”页面。

步骤 6 在告警规则页面，设置告警信息。

创建告警规则操作，请查看《天翼云云监控服务用户指南》。

1. 设置告警名称和告警的描述。
2. 设置告警策略和告警级别。

   例如，在进行指标监控时，如果连续 3 个周期，连接数原始值超过设置的值，则产生告警，如果未及时处理，则每一天发送一次告警通知。
3. 设置“发送通知”开关。当开启时，设置告警生效时间、产生告警时通知的对象以及触发的条件。
4. 单击“立即创建”，等待创建告警规则成功。

----结束

3 权限管理

3.1 创建用户并授权使用 DMS for RabbitMQ

如果您需要对您所拥有的 DMS for RabbitMQ 服务进行精细的权限管理，您可以使用统一身份认证服务（Identity and Access Management，简称 IAM），通过 IAM，您可以：

- 根据企业的业务组织，在您的帐号中，给企业中不同职能部门的员工创建 IAM 用户，让员工拥有唯一安全凭证，并使用 DMS for RabbitMQ 资源。
- 根据企业用户的职能，设置不同的访问权限，以达到用户之间的权限隔离。
- 将 DMS for RabbitMQ 资源委托给更专业、高效的其他帐号或者云服务，这些帐号或者云服务可以根据权限进行代运维。

如果帐号已经能满足您的要求，不需要创建独立的 IAM 用户，您可以跳过本章节，不影响您使用 DMS for RabbitMQ 服务的其它功能。

本章节为您介绍对用户授权的方法，操作流程如图 3-1 所示。

说明

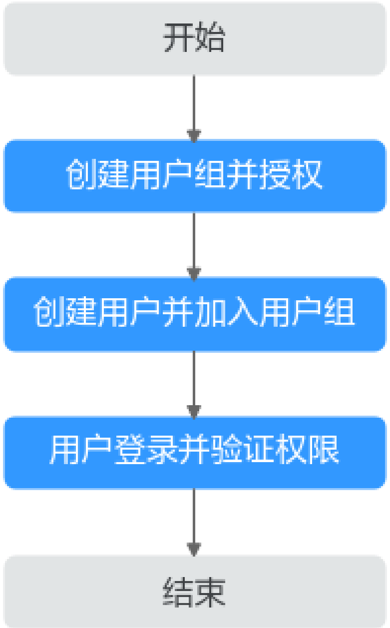
DMS for RabbitMQ 服务的权限与策略基于分布式消息服务 DMS，因此在 IAM 服务中为 RabbitMQ 分配用户与权限时，请选择并使用“DMS”的权限与策略。

前提条件

给用户组授权之前，请您了解用户组可以添加的 DMS for RabbitMQ 系统策略，并结合实际需求进行选择，DMS for RabbitMQ 支持的系统策略及策略间的对比，请参见：DMS for RabbitMQ 系统策略。

示例流程

图3-1 给用户授权 DMS for RabbitMQ 权限流程



1. 创建用户组并授权

   在 IAM 控制台创建用户组，并授予 DMS for RabbitMQ 的只读权限“DMS ReadOnlyAccess”。

2. 创建用户并加入用户组

   在 IAM 控制台创建用户，并将其加入 1 中创建的用户组。

3. 用户登录并验证权限

   新创建的用户登录控制台，切换至授权区域，验证权限：

   - 在“服务列表”中选择分布式消息服务 RabbitMQ，进入 RabbitMQ 实例主界面，单击右上角“购买 RabbitMQ 实例”，尝试购买 RabbitMQ 实例，如果无法购买 RabbitMQ 实例（假设当前权限仅包含 DMS ReadOnlyAccess），表示“DMS ReadOnlyAccess”已生效。
   - 在“服务列表”中选择云硬盘（假设当前策略仅包含 DMS ReadOnlyAccess），若提示权限不足，表示“DMS ReadOnlyAccess”已生效。

3.2 DMS for RabbitMQ 自定义策略

如果系统预置的 DMS for RabbitMQ 权限，不满足您的授权要求，可以创建自定义策略。

目前云服务平台支持以下两种方式创建自定义策略：

- 可视化视图创建自定义策略：无需了解策略语法，按可视化视图导航栏选择云服务、操作、资源、条件等策略内容，可自动生成策略。

- JSON 视图创建自定义策略：可以在选择策略模板后，根据具体需求编辑策略内容；也可以直接在编辑框内编写 JSON 格式的策略内容。

具体创建步骤请参见：《天翼云统一身份认证服务用户指南》的“创建自定义策略”章节。本章为您介绍常用的 DMS for RabbitMQ 自定义策略样例。

说明

DMS for RabbitMQ 服务的权限与策略基于分布式消息服务 DMS，因此在 IAM 服务中为 RabbitMQ 分配用户与权限时，请选择并使用“DMS”的权限与策略。

DMS for RabbitMQ 自定义策略样例

- 示例 1：授权用户删除实例和重启实例

```
{
    "Version": "1.1",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dms:instance:modifyStatus",
                "dms:instance:delete"
            ]
        }
    ]
}
```

- 示例 2：拒绝用户删除实例

  拒绝策略需要同时配合其他策略使用，否则没有实际作用。用户被授予的策略中，一个授权项的作用如果同时存在 Allow 和 Deny，则遵循 Deny 优先。

  如果您给用户授予 DMS FullAccess 的系统策略，但不希望用户拥有 DMS FullAccess 中定义的删除实例权限，您可以创建一条拒绝删除实例的自定义策略，然后同时将 DMS FullAccess 和拒绝策略授予用户，根据 Deny 优先原则，则用户可以对 DMS for RabbitMQ 执行除了删除实例外的所有操作。拒绝策略示例如下：

```
{
        "Version": "1.1",
        "Statement": [
                {
                        "Effect": "Deny",
                        "Action": [
                                "dms:instance:delete"
                        ]
                }
        ]
}
```

3.3 DMS for RabbitMQ 资源

资源是服务中存在的对象。在 DMS for RabbitMQ 中，资源包括：rabbitmq，您可以在创建自定义策略时，通过指定资源路径来选择特定资源。

表3-1 DMS for RabbitMQ 的指定资源与对应路径

指定资源	资源名称	资源路径
rabbitmq	实例	【格式】 DMS:*:*:rabbitmq:*实例 ID* 【说明】 对于实例资源，IAM 自动生成资源路径前缀 **DMS:*:*:rabbitmq:** 通过*实例 ID* 指定具体的资源路径，支持通配符*。例如： DMS:*:*:rabbitmq:*表示任意 RabbitMQ 实例。

3.4 DMS for RabbitMQ 请求条件

您可以在创建自定义策略时，通过添加“请求条件”（Condition 元素）来控制策略何时生效。请求条件包括条件键和运算符，条件键表示策略语句的 Condition 元素，分为全局级条件键和服务级条件键。全局级条件键（前缀为 g:）适用于所有操作，服务级条件键（前缀为服务缩写，如 dms:）仅适用于对应服务的操作。运算符与条件键一起使用，构成完整的条件判断语句。

DMS for RabbitMQ 通过 IAM 预置了一组条件键，例如，您可以先使用 dms:ssl 条件键检查 RabbitMQ 实例是否开启 SSL，然后再允许执行操作。下表显示了适用于 DMS for RabbitMQ 服务特定的条件键。

表3-2 DMS for RabbitMQ 请求条件

DMS for RabbitMQ 条件键	运算符	描述
dms:publicIP	Bool IsNullOrEmpty BoolIfExists	是否开启公网
dms:ssl	Bool IsNullOrEmpty BoolIfExists	是否开启 SSL

4 环境准备

创建 RabbitMQ 实例前，您需要创建虚拟私有云（Virtual Private Cloud，以下简称 VPC），并且已配置好安全组与子网。VPC 为 RabbitMQ 实例提供一个隔离的、用户自主配置和管理的虚拟网络环境，提升云服务资源的安全性，简化用户的网络部署。

如果用户已有 VPC，可重复使用，不需要多次创建。

创建 VPC

步骤 1 登录管理控制台。

步骤 2 在管理控制台左上角单击 ，选择区域。

说明

此处请选择与 RabbitMQ 实例相同的区域。

步骤 3 在管理控制台左上角单击 ，选择“网络 > 虚拟私有云”。

步骤 4 单击“创建虚拟私有云”，进入“创建虚拟私有云”界面。

步骤 5 根据界面提示创建虚拟私有云。关于创建 VPC 的详细信息可以参考《虚拟私有云-用户手册》。

创建虚拟私有云时，会同时创建子网，若需要额外创建子网，请参考步骤 6。如果不需要额外创建子网，请执行步骤 7。

步骤 6 在左侧导航栏，单击“子网”，进入“子网”界面。单击“创建子网”。根据界面提示创建子网。

关于创建子网的详细信息可以参考《虚拟私有云-用户手册》。

步骤 7 在左侧导航栏，选择“访问控制 > 安全组”，进入“安全组”界面。根据界面提示创建安全组。

关于创建安全组的详细信息可以参考《虚拟私有云-用户手册》。

使用 RabbitMQ 实例前，添加表 4-1 所示安全组规则，其他规则请根据实际需要添加。

说明

创建安全组后，系统默认添加入方向“允许安全组内的弹性云主机彼此通信”规则和出方向“放通全部流量”规则，此时使用内网通过同一个 VPC 访问 RabbitMQ 实例，无需添加表 4-1 的规则。

表4-1 安全组规则

方向	协议	端口	源地址	说明
入方向	TCP	5672	0.0.0.0/0	访问 RabbitMQ 实例（关闭 SSL 加密）
入方向	TCP	5671	0.0.0.0/0	访问 RabbitMQ 实例（开启 SSL 加密）
入方向	TCP	15672	0.0.0.0/0	访问 Web 界面 UI 地址（关闭 SSL 加密）
入方向	TCP	15671	0.0.0.0/0	访问 Web 界面 UI 地址（开启 SSL 加密）

----结束

5 购买实例

操作场景

RabbitMQ 实例采用物理隔离的方式部署，租户独占 RabbitMQ 实例。支持用户自定义规格和自定义特性，您可以根据业务需要定制相应计算能力和存储空间的 RabbitMQ 实例。

RabbitMQ 是一款基于 AMQP 协议的开源服务，用于在分布式系统中存储转发消息，服务器端用 Erlang 语言（支持高并发、分布式以及健壮的容错能力等特点）编写，支持多种语言的客户端，如：Python、Ruby、.NET、Java、JMS、C、PHP、ActionScript、XMPP、STOMP、AJAX 等。

AMQP，即 Advanced Message Queuing Protocol，高级消息队列协议，是应用层的一个开放标准协议。

前提条件

RabbitMQ 实例运行于虚拟私有云，购买实例前，需保证有可用的虚拟私有云，并且已配置好安全组与子网。

操作步骤

步骤 1 登录管理控制台。

步骤 2 在管理控制台左上角单击，选择区域。

说明

此处请选择与您的应用服务相同的区域。

步骤 3 在管理控制台左上角单击，选择“应用服务 > 分布式消息服务 RabbitMQ”，进入分布式消息服务 RabbitMQ 专享版页面。

步骤 4 单击页面右上方的“购买 RabbitMQ 实例”。

步骤 5 选择“计费模式”、“区域”、“项目”和“可用区”。

步骤 6 设置“实例名称”和“企业项目”。

步骤 7 设置实例信息。

1. 版本：RabbitMQ 的版本号，当前仅支持 RabbitMQ 3.8.35。
2. 实例类型：支持“单机”和“集群”。
   - 单机：表示部署一个 RabbitMQ 代理。
   - 集群：表示部署多个 RabbitMQ 代理，实现高可靠的消息存储。
3. CPU 架构：当前仅支持“x86 计算”，保持默认值即可。
4. 代理规格：根据实际情况选择规格。

说明

为了保证服务的稳定可靠，RabbitMQ 采用了默认的 40%高水位配置。当内存占用率到达 40%高水位后，会触发流控，生产者发送消息会被阻塞。为了避免高水位的产生，请及时消费积压在队列中的消息。

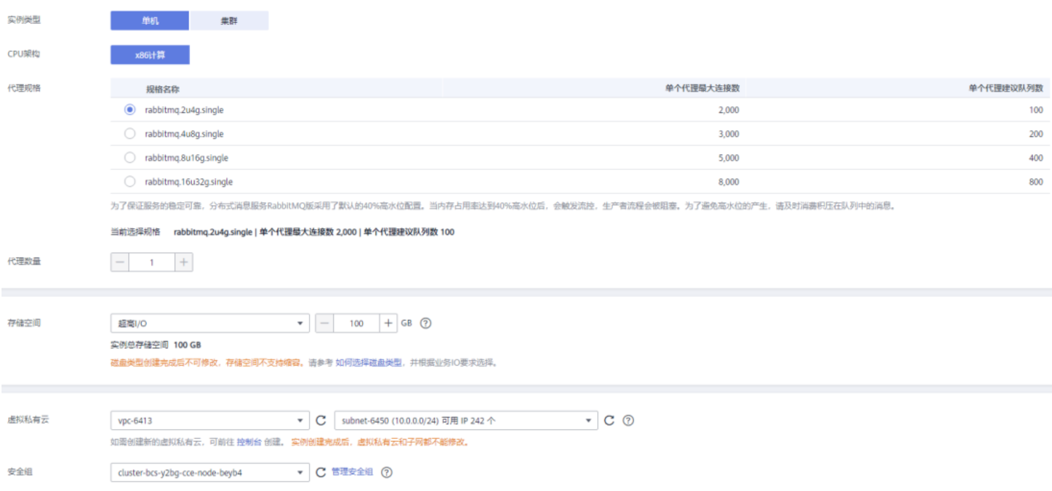
5. 代理数量：根据实例情况选择代理个数。
6. 存储空间：选择 RabbitMQ 实例的磁盘类型和储存空间总量。

   如何选择磁盘类型请参考《天翼云云硬盘用户使用指南》的“简介 > 磁盘类型及性能介绍”章节。
   - 单机实例的取值范围 200GB～90000GB。
   - 集群实例的取值范围为：100GB*代理数 ～90000GB、200GB*代理数 ～90000GB、300GB*代理数 ～90000GB。
7. 虚拟私有云：选择已经创建好的虚拟私有云和子网。

   虚拟私有云可以为您的 RabbitMQ 实例构建隔离的、能自主配置和管理的虚拟网络环境。
8. 安全组：选择已经创建好的安全组。

   安全组是一组对 RabbitMQ 实例的访问规则的集合。您可以单击右侧的“管理安全组”，跳转到网络控制台的安全组页面，查看或创建安全组。

图5-1 设置实例信息



步骤 8 设置连接 RabbitMQ 实例的用户名和密码。

步骤 9 设置实例购买时长。

当选择了“包年/包月”计费模式时，页面才显示“购买时长”参数，您需要根据业务需要选择。

步骤 10 单击“更多配置”，设置更多相关信息。

1. 设置“公网访问”。

   您可以选择是否打开公网访问开关。

   如果选择了开启，表示访问 RabbitMQ 实例可以通过弹性 IP 访问。这时页面会显示“弹性 IP 地址”，在“弹性 IP 地址”区域，您可下拉选择已有的弹性 IP。另外，您可单击右侧的“创建弹性 IP”，跳转到网络控制台的弹性公网 IP 页面，购买弹性公网 IP。

   图5-2 购买 RabbitMQ 实例时公网访问设置

   说明

   - 公网访问与 VPC 内访问相比，可能存在网络丢包和抖动等情况，且访问时延有所增加，因此建议仅在业务开发测试阶段开启公网访问进行调试。
   - 如果用户在虚拟私有云的服务页面手动解绑定或删除 EIP，相应 RabbitMQ 实例的公网访问功能会自动关闭。

2. 设置“SSL”。

   客户端连接实例时 SSL 认证的开关。开启 SSL，则数据加密传输，安全性更高。

   SSL 开关在实例创建完成后不支持修改，请明确是否需要开启。

3. 设置“标签”。

   标签用于标识云资源，当您拥有相同类型的许多云资源时，可以使用标签按各种维度（例如用途、所有者或环境）对云资源进行分类。

   – 如果您已经预定义了标签，在“标签键”和“标签值”中选择已经定义的标签键值对。另外，您可以单击右侧的“查看预定义标签”，系统会跳转到标签管理服务页面，查看已经预定义的标签，或者创建新的标签。
   – 您也可以直接在“标签键”和“标签值”中设置标签。

   当前每个 RabbitMQ 实例最多支持设置 20 个不同标签，标签的命名规格，请参考管理实例标签章节。

4. 设置实例的描述信息。

步骤 11 填写完上述信息后，单击“立即购买”，进入“规格确认”页面。

步骤 12 确认实例信息无误后，单击“提交”。

步骤 13 在实例列表页面查看实例是否创建成功。

创建实例大约需要 3 到 15 分钟，此时实例的“状态”为“创建中”。

- 当实例的“状态”变为“运行中”时，说明实例创建成功。

- 如果创建实例失败，在信息栏的“创建失败任务”中查看创建失败的实例。请参考删除实例，删除创建失败的 RabbitMQ 实例，然后重新购买。如果重新购买仍然失败，请联系客服。

----结束

6 连接实例

6.1 连接未开启 SSL 方式的 RabbitMQ 实例

RabbitMQ 实例兼容开源协议，请参考 RabbitMQ 官网提供的不同语言的连接和使用向导：https://www.rabbitmq.com/getstarted.html。

本节以分布式消息服务 RabbitMQ 提供的 demo 为例，介绍 VPC 内访问与使用 RabbitMQ 的方法，假设 RabbitMQ 客户端部署在弹性云主机上。

如果 RabbitMQ 实例开启了 SSL 认证开关，连接方式请参考连接已开启 SSL 方式的 RabbitMQ 实例。

前提条件

- 参考购买实例章节创建 RabbitMQ 实例，并记录创建时输入的用户名和密码。
- 创建完成后，单击实例名称，查看并记录实例详情中的“内网连接地址/公网连接地址”。
- 已创建弹性云主机，并且弹性云主机的 VPC、子网、安全组与 RabbitMQ 实例的 VPC、子网、安全组保持一致。

命令行模式连接实例

步骤 1 登录弹性云主机。

步骤 2 安装 Java JDK 或 JRE，并配置 JAVA_HOME 与 PATH 环境变量。在用户家目录下修改.bash_profile，添加如下行，路径以实际为准。

```
export JAVA_HOME=/opt/java/jdk1.8.0_151
export PATH=$JAVA_HOME/bin:$PATH
```

执行 source .bash_profile 命令使修改生效。

说明

ECS 虚拟机默认自带的 JDK 可能不符合要求，例如 OpenJDK，需要配置为 Oracle 的 JDK，可至 Oracle 官方下载页面下载 Java Development Kit 1.8.111 及以上版本。

步骤 3 下载 RabbitMQ-Tutorial.zip 示例工程代码。

```
$ wget https://obs.ctyun.cn/dms-demo/RabbitMQ-Tutorial.zip
```

步骤 4 解压 RabbitMQ-Tutorial.zip 压缩包。

```
$ unzip RabbitMQ-Tutorial.zip
```

步骤 5 进入 RabbitMQ-Tutorial 目录，该目录下包含预编译好的 jar 文件。

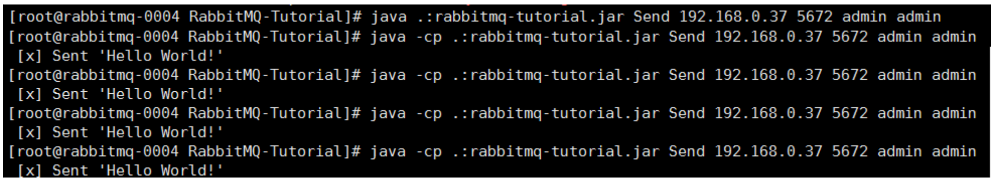
```
$ cd RabbitMQ-Tutorial
```

步骤 6 运行生产消息示例。

```
$ java -cp .:rabbitmq-tutorial.jar Send host port user password
```

其中，host 表示 RabbitMQ 实例的连接地址，port 为 RabbitMQ 实例的监听端口（默认为 5672），user 表示 RabbitMQ 用户名，password 表示用户名对应的密码。

图6-1 运行生产消息示例



```
[root@rabbitmq-0004 RabbitMQ-Tutorial]# java .:rabbitmq-tutorial.jar Send 192.168.0.37 5672 admin admin
[root@rabbitmq-0004 RabbitMQ-Tutorial]# java -cp .:rabbitmq-tutorial.jar Send 192.168.0.37 5672 admin admin
 [x] Sent 'Hello World!'
[root@rabbitmq-0004 RabbitMQ-Tutorial]# java -cp .:rabbitmq-tutorial.jar Send 192.168.0.37 5672 admin admin
 [x] Sent 'Hello World!'
[root@rabbitmq-0004 RabbitMQ-Tutorial]# java -cp .:rabbitmq-tutorial.jar Send 192.168.0.37 5672 admin admin
 [x] Sent 'Hello World!'
[root@rabbitmq-0004 RabbitMQ-Tutorial]# java -cp .:rabbitmq-tutorial.jar Send 192.168.0.37 5672 admin admin
 [x] Sent 'Hello World!'
```

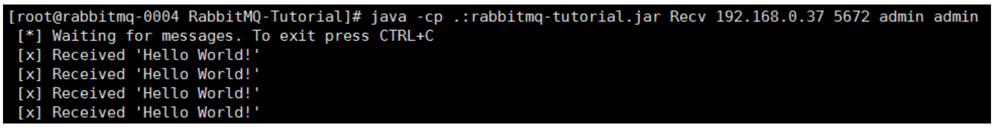
使用 **Ctrl+C** 命令退出。

步骤 7 运行消费消息示例。

```
$ java -cp .:rabbitmq-tutorial.jar Recv host port user password
```

其中，host 表示 RabbitMQ 实例的连接地址，port 为 RabbitMQ 实例的监听端口（默认为 5672），user 表示 RabbitMQ 用户名，password 表示用户名对应的密码。

图6-2 运行消费消息示例



```
[root@rabbitmq-0004 RabbitMQ-Tutorial]# java -cp .:rabbitmq-tutorial.jar Recv 192.168.0.37 5672 admin admin
 [*] Waiting for messages. To exit press CTRL+C
 [x] Received 'Hello World!'
 [x] Received 'Hello World!'
 [x] Received 'Hello World!'
 [x] Received 'Hello World!'
```

如需停止消费使用 **Ctrl+C** 命令退出。

----结束

示例代码（Java）

连接实例并生产消息

```
ConnectionFactory factory = new ConnectionFactory();
 factory.setHost(host);
 factory.setPort(port);

 factory.setUsername(user);
 factory.setPassword(password);
 Connection connection = factory.newConnection();
```

```
Channel channel = connection.createChannel();

channel.queueDeclare(QUEUE_NAME, false, false, false, null);

String message = "Hello World!";
channel.basicPublish("", QUEUE_NAME, null, message.getBytes("UTF-8"));
System.out.println(" [x] Sent '" + message + "'");

channel.close();
connection.close();
```

连接实例并消费消息

```
ConnectionFactory factory = new ConnectionFactory();
factory.setHost(host);
factory.setPort(port);
factory.setUsername(user);
factory.setPassword(password);
Connection connection = factory.newConnection();
Channel channel = connection.createChannel();

channel.queueDeclare(QUEUE_NAME, false, false, false, null);
System.out.println(" [*] Waiting for messages. To exit press CTRL+C");

Consumer consumer = new DefaultConsumer(channel)
{
    @Override
    public void handleDelivery(String consumerTag, Envelope envelope,
AMQP.BasicProperties properties,
            byte[] body)
            throws IOException
    {
        String message = new String(body, "UTF-8");
        System.out.println(" [x] Received '" + message + "'");
    }
};
channel.basicConsume(QUEUE_NAME, true, consumer);
```

6.2 连接已开启 SSL 方式的 RabbitMQ 实例

创建实例时开启 SSL 访问，则数据加密传输，安全性更高。

本节介绍 VPC 内访问开启 SSL 的 RabbitMQ 实例的方法。

前提条件

- 参考购买实例章节创建 RabbitMQ 实例，并记录创建时输入的用户名和密码。
- 创建完成后，单击实例名称，查看并记录实例详情中的“内网连接地址/公网连接地址”。
- 已创建弹性云主机，并且弹性云主机的 VPC、子网、安全组与 RabbitMQ 实例的 VPC、子网、安全组保持一致。

命令行模式连接实例

步骤 1 登录弹性云主机，如开启公网访问，则直接登录执行主机。

步骤 2 安装 Java JDK 或 JRE，并配置 JAVA_HOME 与 PATH 环境变量，使用执行用户在用户家目录下修改.bash_profile，添加如下行，路径以实际为准。

```
export JAVA_HOME=/opt/java/jdk1.8.0_151
export PATH=$JAVA_HOME/bin:$PATH
```

执行 source .bash_profile 命令使修改生效。

说明

ECS 虚拟机默认自带的 JDK 可能不符合要求，例如 OpenJDK，需要配置为 Oracle 的 JDK，可至 Oracle 官方下载页面下载 Java Development Kit 1.8.111 及以上版本。

步骤 3 下载 RabbitMQ-Tutorial-SSL.zip 示例工程代码。

```
$ wget https://obs.ctyun.cn/dms-demo/RabbitMQ-Tutorial.zip
```

步骤 4 解压 RabbitMQ-Tutorial-SSL.zip 压缩包。

```
$ unzip RabbitMQ-Tutorial-SSL.zip
```

步骤 5 进入 RabbitMQ-Tutorial-SSL 目录，该目录下包含预编译好的 jar 文件。

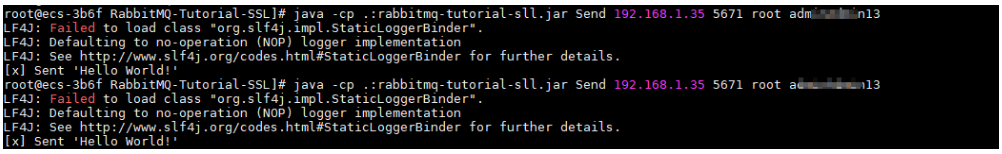
```
$ cd RabbitMQ-Tutorial-SSL
```

步骤 6 运行生产消息示例。

```
$ java -cp .:rabbitmq-tutorial-sll.jar Send host port user password
```

其中，host 表示 RabbitMQ 实例的连接地址，port 为 RabbitMQ 实例的监听端口（默认为 5671），user 表示 RabbitMQ 用户名，password 表示用户名对应的密码。

图6-3 生产消息示例



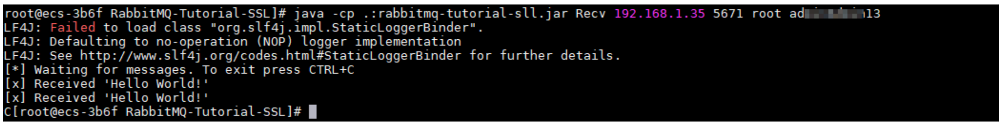
使用 **Ctrl+C** 命令退出。

步骤 7 运行消费消息示例。

```
$ java -cp .:rabbitmq-tutorial-sll.jar Recv host port user password
```

其中，host 表示 RabbitMQ 实例的连接地址，port 为 RabbitMQ 实例的监听端口（默认为 5671），user 表示 RabbitMQ 用户名，password 表示用户名对应的密码。

图6-4 消费消息示例



如需停止消费使用 **Ctrl+C** 命令退出。

----结束

示例代码（Java）

连接实例并生产消息

```
ConnectionFactory factory = new ConnectionFactory();
 factory.setHost(host);
 factory.setPort(port);

 factory.setUsername(user);
 factory.setPassword(password);
 factory.useSslProtocol();
 Connection connection = factory.newConnection();
 Channel channel = connection.createChannel();

 channel.queueDeclare(QUEUE_NAME, false, false, false, null);

 String message = "Hello World!";
 channel.basicPublish("", QUEUE_NAME, null, message.getBytes("UTF-8"));
 System.out.println(" [x] Sent '" + message + "'");

 channel.close();
 connection.close();
```

连接实例并消费消息

```
ConnectionFactory factory = new ConnectionFactory();
 factory.setHost(host);
 factory.setPort(port);
 factory.setUsername(user);
 factory.setPassword(password);
 factory.useSslProtocol();
 Connection connection = factory.newConnection();
 Channel channel = connection.createChannel();

 channel.queueDeclare(QUEUE_NAME, false, false, false, null);
 System.out.println(" [*] Waiting for messages. To exit press CTRL+C");

 Consumer consumer = new DefaultConsumer(channel)
 {
    @Override
    public void handleDelivery(String consumerTag, Envelope envelope,
AMQP.BasicProperties properties,
          byte[] body)
          throws IOException
```

```
    {
        String message = new String(body, "UTF-8");
        System.out.println(" [x] Received '" + message + "'");
    }
};
channel.basicConsume(QUEUE_NAME, true, consumer);
```

6.3 连接 RabbitMQ 管理地址

在浏览器中输入 RabbitMQ 管理地址，访问开源 RabbitMQ 的集群管理工具。

操作步骤

步骤 1 获取实例管理地址。

1. 登录管理控制台。
2. 在管理控制台左上角单击 ，选择区域。

说明

此处请选择与您的应用服务相同的区域。

3. 在管理控制台左上角单击 ，选择“应用服务 > 分布式消息服务 RabbitMQ”，进入分布式消息服务 RabbitMQ 专享版页面。
4. 单击实例名称，进入实例详情页面，获取 Web 界面 UI 地址和用户名。

说明

用户名和密码为创建 RabbitMQ 实例时自定义的内容。

步骤 2 确认实例安全组规则是否配置正确。

1. 在实例详情页面的“基本信息 > 网络”，单击安全组名称，跳转到安全组页面。
2. 选择“入方向规则”，查看安全组入方向规则。
   - 实例未开启 SSL 开关
     - 如果是 VPC 内访问，实例安全组入方向规则，需要允许端口 5672 的访问。
     - 如果是公网访问，需要允许端口 15672 的访问。
   - 实例已开启 SSL 开关
     - 如果是 VPC 内访问，实例安全组入方向规则，需要允许端口 5671 的访问。
     - 如果是公网访问，需要运行端口 15671 的访问。

步骤 3 在浏览器中打开 Web 界面 UI 地址，进入 Web 登录页面。

说明

- 如果 RabbitMQ 实例开启了公网访问，可直接在公网环境下的浏览器访问 Web 页面。
- 如果 RabbitMQ 实例未开启公网访问，您需要购买一台与 RabbitMQ 实例网络相通的 Windows 弹性云主机，然后登录弹性云主机访问 Web 页面。

图6-5 登录实例 Web 页面

步骤 4 单击“Login”，登录完成。

----结束

6.4 开启心跳

客户端连接 RabbitMQ 集群实例时，如果存在消息收发时间间隔大于 90 秒的场景，请在客户端开启心跳并设置小于 90 秒的心跳超时时间，防止断连。

什么是心跳

RabbitMQ 实例提供了心跳功能，以确保应用程序层及时发现中断的连接和完全无响应的对端。心跳还可以防止某些网络设备在一段时间内由于没有活动而中断 TCP 连接。**开启心跳的方法为在连接上指定心跳超时时间。**

心跳超时时间定义了对等 TCP 连接在多长时间后被服务端和客户端视为关闭。服务端和客户端会对配置的心跳超时时间进行协商，客户端必须配置该值来发送心跳。RabbitMQ 官方团队维护的 3 个客户端（Java、.NET、Erlang 语言）的心跳超时时间协商逻辑如下：

- 服务端和客户端设置的心跳超时时间都不为 0 时，两者间较小的值生效。
- 服务端和客户端任意一端设置的心跳超时时间为 0，另一端不为 0 时，非 0 的值生效。
- 服务端和客户端的心跳超时时间都设置为 0 时，表示禁用心跳。

配置心跳超时时间后，RabbitMQ 服务端和客户端都会向对方发送 AMQP 心跳帧作为心跳，发送的时间间隔为心跳超时时间的一半。客户端在两次错过心跳后，会被认为是不可达的，TCP 连接将被关闭。当客户端检测到服务端由于心跳而无法访问时，需要重新连接。

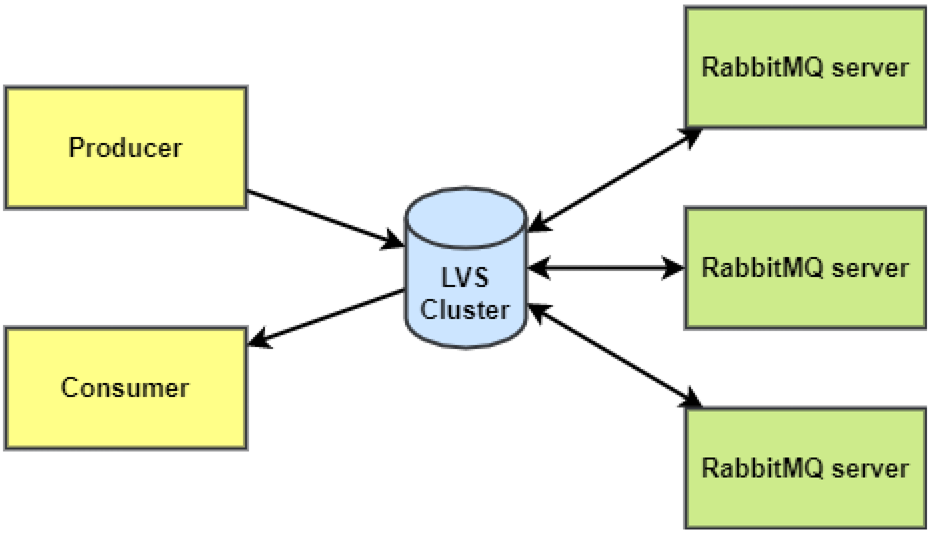
须知

一些客户端（如 C 语言客户端）没有发送心跳的逻辑，即使配置了心跳超时时间，开启了心跳，仍然无法发送心跳。此时需要额外启动一个线程，编写发送心跳的逻辑。

LVS 的心跳超时时间

RabbitMQ 集群实例使用 LVS 进行负载均衡，如图 6-6 所示，单节点实例不涉及 LVS。

图6-6 集群实例的负载均衡



LVS 对客户端连接设置了心跳超时时间，默认为 90 秒。如果客户端在 90 秒内没有向 LVS 发送心跳（AMQP 心跳帧或消息收发），LVS 会主动断开与客户端的连接，此时客户端需要重新连接。

如果存在消息收发时间间隔大于 90 秒的场景，请在客户端开启心跳并设置小于 90 秒的心跳超时时间。

客户端如何配置心跳超时时间

- 在 Java 客户端配置心跳超时时间。

  在创建连接前使用 **ConnectionFactory#setRequestedHeartbeat** 进行设置，示例如下：

  ```
  ConnectionFactory cf = new ConnectionFactory();
  // 将心跳超时时间设置为 60 秒
  cf.setRequestedHeartbeat(60);
  ```

- 在.NET 客户端配置心跳超时时间，示例如下。

  ```
  var cf = new ConnectionFactory();
  // 将心跳超时设置为 60 秒
  cf.RequestedHeartbeat = TimeSpan.FromSeconds(60);
  ```

- 在 Python pika 客户端配置心跳超时时间，示例如下。

```
# 设置心跳时间为 60 秒
params = pika.ConnectionParameters(host='host', heartbeat=60,
credentials=pika.PlainCredentials('username', 'passwd'))
connection = pika.BlockingConnection(params)

while True:
    channel.basic_publish(exchange='', routing_key='hello', body='Hello
World!')
    print(" [x] Sent 'Hello World!'")
    # 生产者需要使用 connection.sleep()才能触发心跳，使用 time.sleep()不会触发心跳
    connection.sleep(200)
```

6.5 查看客户端连接地址

分布式消息服务 RabbitMQ 版支持通过 Web 界面查看客户端连接地址。

说明

客户端处于连接 RabbitMQ 实例时，才可以查看客户端连接地址。

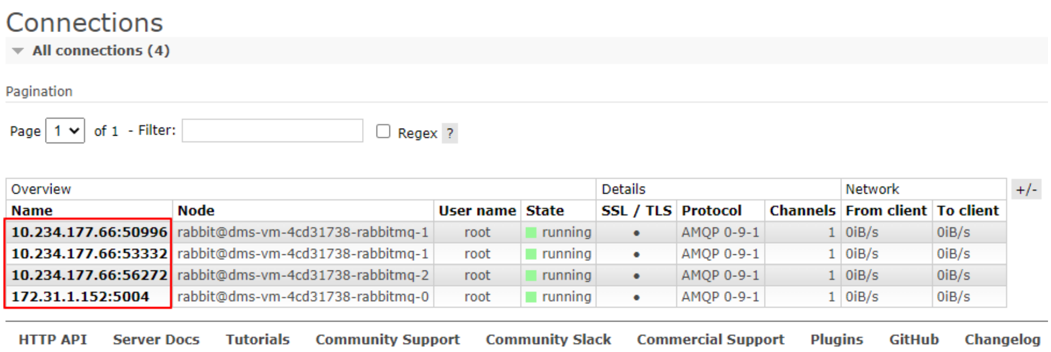
操作步骤

步骤 1 登录 RabbitMQ Web 界面。

步骤 2 在导航栏单击“Connections”，进入“Connections”页面。

步骤 3 查看客户端连接地址，如图 6-7 所示。

图6-7 客户端连接地址



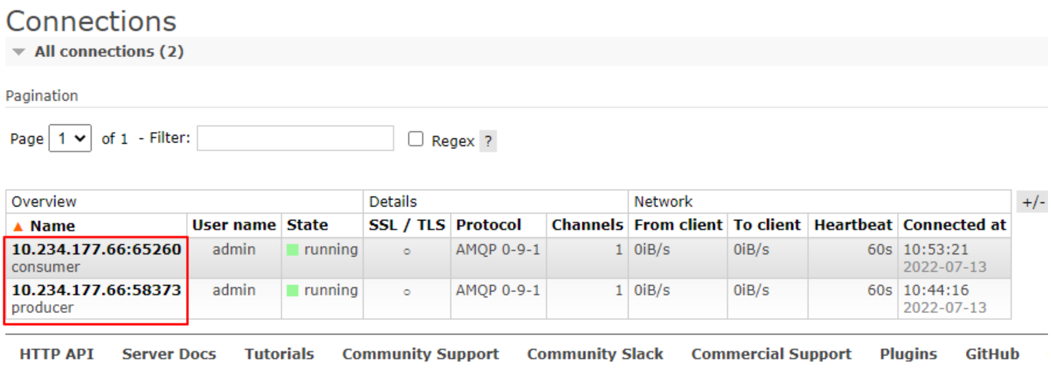
同一个客户端可以作为生产者生产消息，也可以作为消费者消费消息，连接 IP 地址是相同的，如图 6-7 所示，此时我们无法区分哪个是生产者 IP 地址，哪个是消费者 IP 地址。如果想要直观体现生产者/消费者 IP 地址，您可以在客户端中设置“clientProperties”参数，通过此参数来标明生产者/消费者 IP 地址，示例如下。

```
//配置客户端连接参数
HashMap<String, Object> clientProperties = new HashMap<>();
clientProperties.put("connection_name", "producer");
connectionFactory.setClientProperties(clientProperties);
```

```
//创建连接
Connection connection = connectionFactory.newConnection();
```

设置“clientProperties”参数后，连接地址显示如图 6-8 所示。

图6-8 客户端连接地址（分区生产者/消费者 IP 地址）



----结束

7 实例日常操作

7.1 查看实例

操作场景

本节介绍如何在控制台查看 RabbitMQ 实例的详细信息。例如，连接 RabbitMQ 时，需要获取连接 IP 和端口。

前提条件

已创建 RabbitMQ 实例。

操作步骤

步骤 1 登录管理控制台。

步骤 2 在管理控制台左上角单击，选择区域。

> **说明**
>
> 此处请选择 RabbitMQ 实例所在的区域。

步骤 3 在管理控制台左上角单击，选择“应用服务 > 分布式消息服务 RabbitMQ”，进入分布式消息服务 RabbitMQ 专享版页面。

步骤 4 RabbitMQ 实例支持通过筛选来查询对应的 RabbitMQ 实例。当前支持的筛选条件为“标签”、“状态”、“名称”、“连接地址”和“ID”。RabbitMQ 实例状态请参见表 7-1。

表7-1 RabbitMQ 实例状态说明

状态	说明
创建中	创建 RabbitMQ 实例后，在 RabbitMQ 实例状态进入运行中之前的状态。
运行中	RabbitMQ 实例正常运行状态。

状态	说明
	在这个状态的实例可以运行您的业务。
故障	RabbitMQ 实例处于故障的状态。
启动中	RabbitMQ 实例从已冻结到运行中的中间状态。
重启中	RabbitMQ 实例正在进行重启操作。
变更中	RabbitMQ 实例正在进行规格变更操作。
变更失败	RabbitMQ 实例处于规格变更操作失败的状态。
已冻结	RabbitMQ 实例处于已冻结状态。
冻结中	RabbitMQ 实例从运行中到已冻结的中间状态。
升级中	RabbitMQ 实例正在进行升级操作。
回滚中	RabbitMQ 实例正在进行回滚操作。

步骤 5 单击 RabbitMQ 实例的名称，进入该 RabbitMQ 实例的基本信息页面，查看 RabbitMQ 实例的详细信息。

表 7-2 为连接实例的相关参数，其他参数，请查看页面显示。

表7-2 连接参数说明

参数	说明
内网连接地址	未开启公网访问时，连接实例的地址。
公网访问	是否开启公网访问开关。
公网连接地址	开启公网访问后，连接实例的地址。

----结束

7.2 重启实例

操作场景

分布式消息服务 RabbitMQ 管理控制台支持重启运行中的 RabbitMQ 实例，且可实现批量重启 RabbitMQ 实例。

须知

在 RabbitMQ 实例重启过程中，客户端的生产与消费消息等请求会被拒绝。

前提条件

只有当 RabbitMQ 实例处于“运行中”或“故障”状态，才能执行重启操作。

操作步骤

步骤 1 登录管理控制台。

步骤 2 在管理控制台左上角单击，选择区域。

说明

此处请选择 RabbitMQ 实例所在的区域。

步骤 3 在管理控制台左上角单击☰，选择“应用服务 > 分布式消息服务 RabbitMQ”，进入分布式消息服务 RabbitMQ 专享版页面。

步骤 4 通过以下任意一种方法，重启 RabbitMQ 实例。

- 勾选 RabbitMQ 实例名称左侧的方框，可选一个或多个，单击信息栏左上侧的“重启”。
- 在待重启 RabbitMQ 实例所在行，单击“重启”。
- 单击 RabbitMQ 实例名称，进入实例详情页面。单击右上角的“重启”。

步骤 5 单击“是”，完成重启 RabbitMQ 实例。

重启 RabbitMQ 实例大约需要 3 到 15 分钟。RabbitMQ 实例重启成功后，RabbitMQ 实例状态切换为“运行中”。

说明

重启 RabbitMQ 只会重启实例进程，不会重启实例所在虚拟机。

如果只需要重启单个 RabbitMQ 实例，也可以在“RabbitMQ 专享版”界面，单击指定 RabbitMQ 实例右侧“操作”栏下的“重启”。

----结束

7.3 删除实例

操作场景

分布式消息服务 RabbitMQ 管理控制台支持删除 RabbitMQ 实例，且可实现批量删除 RabbitMQ 实例、一键式删除创建失败的 RabbitMQ 实例。

须知

RabbitMQ 实例删除后，实例中原有的数据将被删除，且没有备份，请谨慎操作。

前提条件

- RabbitMQ 实例状态为运行中、故障、已冻结的按需付费实例才能执行删除操作。
- 包年/包月类型的 RabbitMQ 实例，不支持进行删除和批量删除操作。若不再使用，可单击“操作”栏下的“更多 > 退订”进行退订。

删除 RabbitMQ 实例

步骤 1 登录管理控制台。

步骤 2 在管理控制台左上角单击，选择区域。

说明

此处请选择 RabbitMQ 实例所在的区域。

步骤 3 在管理控制台左上角单击，选择“应用服务 > 分布式消息服务 RabbitMQ”，进入分布式消息服务 RabbitMQ 专享版页面。

步骤 4 通过以下任意一种方法，删除 RabbitMQ 实例。

- 勾选 RabbitMQ 实例名称左侧的方框，可选一个或多个，单击信息栏左上侧的“删除”。
- 在待删除 RabbitMQ 实例所在行，单击“更多 > 删除”。
- 单击 RabbitMQ 实例名称，进入实例详情页面。单击右上角的“更多 > 删除”。

说明

RabbitMQ 实例状态为创建中、启动中、变更中、变更失败、重启中时不允许执行删除操作。

步骤 5 单击“是”，完成删除 RabbitMQ 实例。

删除 RabbitMQ 实例大约需要 1 到 60 秒。

----结束

删除创建失败的 RabbitMQ 实例

步骤 1 登录管理控制台。

步骤 2 在管理控制台左上角单击，选择区域。

说明

此处请选择 RabbitMQ 实例所在的区域。

步骤 3 在管理控制台左上角单击，选择“应用服务 > 分布式消息服务 RabbitMQ”，进入分布式消息服务 RabbitMQ 专享版页面。

步骤 4 若当前存在创建失败的 RabbitMQ 实例，界面信息栏会显示“创建失败任务”及失败数量信息。单击“创建失败任务”后的图标或者数量，弹出“创建失败任务”对话框。

步骤 5 在“创建失败任务”界面删除创建失败的 RabbitMQ 实例。

- 单击“清理失败任务”，一键式删除所有创建失败的 RabbitMQ 实例。
- 单击需要删除的 RabbitMQ 实例右侧的“删除任务”，依次删除创建失败的 RabbitMQ 实例。

----结束

7.4 修改实例信息

创建 RabbitMQ 实例成功后，您可以根据自己的业务情况对 RabbitMQ 实例的部分参数进行调整。

操作步骤

步骤 1 登录管理控制台。

步骤 2 在管理控制台左上角单击 ，选择区域。

说明

此处请选择 RabbitMQ 实例所在的区域。

步骤 3 在管理控制台左上角单击 ，选择“应用服务 > 分布式消息服务 RabbitMQ”，进入分布式消息服务 RabbitMQ 专享版页面。

步骤 4 单击 RabbitMQ 实例的名称，进入实例详情页面。

步骤 5 以下参数支持修改。

- 实例名称
- 描述
- 企业项目
- 安全组
- 公网访问（公网访问的修改方法，请参考设置实例的公网访问。）

参数修改完成后，通过以下方式查看修改结果。

- 修改“公网访问”，系统跳转到“后台任务管理”页签，并显示当前任务的操作进度和结果。
- 修改“实例名称”、“描述”、“企业项目”和“安全组”后，右上角直接提示修改结果。

----结束

7.5 重置实例密码

操作场景

如果您忘记了创建实例时设置的密码，通过重置密码功能，重新设置一个新的密码，使用新密码连接 RabbitMQ 实例。

说明

只有处于“运行中”状态的 RabbitMQ 实例支持重置密码。

操作步骤

步骤 1 登录管理控制台。

步骤 2 在管理控制台左上角单击 ，选择区域。

说明

此处请选择 RabbitMQ 实例所在的区域。

步骤 3 在管理控制台左上角单击 ，选择“应用服务 > 分布式消息服务 RabbitMQ”，进入分布式消息服务 RabbitMQ 专享版页面。

步骤 4 通过以下任意一种方法，重置实例密码。

- 在待重置密码的实例所在行，单击“更多 > 重置密码”。
- 单击 RabbitMQ 实例名称，进入实例详情页面。单击右上角的“更多 > 重置密码”。

步骤 5 输入“新密码”和“确认密码”，单击“确定”完成密码重置。

- 重置密码成功，界面提示重置实例的密码成功。
- 重置密码失败，界面提示重置实例的密码失败，请重新尝试重置密码操作。如果多次重置失败，请联系客服处理。

说明

只有所有代理都重置密码成功，才会提示重置密码成功，否则会提示重置失败。

----结束

7.6 变更实例规格

操作场景

RabbitMQ 实例创建成功后，您可以根据业务需要，扩容或者缩容，RabbitMQ 实例支持的变更配置如表 7-3 所示。

表7-3 变更配置列表

实例类型	变更配置类型	是否支持扩容	是否支持缩容
集群	代理个数	√	×
	存储空间	√	×
	代理规格	√	√
单机	代理个数	×	×
	存储空间	√	×
	代理规格	√	√

约束与限制

- 为了实例运行正常，变更规格过程中，请勿对实例进行其他操作。
- 实例变更规格后，配置费用将发生变化。

前提条件

已创建 RabbitMQ 实例，且实例状态为“运行中”。

操作步骤

步骤 1 登录管理控制台。

步骤 2 在管理控制台左上角单击，选择区域。

说明

此处请选择 RabbitMQ 实例所在的区域。

步骤 3 在管理控制台左上角单击，选择“应用服务 > 分布式消息服务 RabbitMQ”，进入分布式消息服务 RabbitMQ 专享版页面。

步骤 4 通过以下任意一种方法，变更实例规格。

- 在待变更规格的实例所在行，单击“更多 > 变更规格”。
- 单击 RabbitMQ 实例名称，进入实例详情页面。单击右上角的“更多 > 变更规格”。

步骤 5 根据实际情况选择扩容存储空间、代理个数或者扩容/缩容代理规格。

- 扩容存储空间

  在“变更配置”中，选择“存储空间”，在“单个代理存储空间”中，选择扩容后的单个代理的存储空间大小，单击“下一步”。确认扩容信息无误后，单击“提交”。

  在实例列表页面的“可用存储空间”中查看扩容后的总存储空间大小（即扩容后的单个代理的存储空间*代理个数）。

说明

可用存储空间=实际存储空间-用于存储日志的存储空间-格式化磁盘的损耗。

例如，实际扩容存储空间到 700GB，用于存储日志的数据的存储空间为 100GB，格式化磁盘损耗 7GB，那么扩容后的可用存储空间为 593GB。

- 扩容代理个数

  在“变更配置”中，选择“代理数量”，在“代理数量”中，选择扩容后的代理个数，单击“下一步”。确认扩容信息无误后，单击“提交”。

  在实例列表页面的“规格”中查看扩容后的代理个数。

说明

变更规格过程中会有秒级业务中断，客户端需要支持自动重连，建议在业务低峰时进行变更。

- 扩容/缩容代理规格

  在“变更配置”中，选择“代理规格”，在“代理规格”中，选择扩容/缩容后的规格，单击“下一步”。确认扩容/缩容信息无误后，单击“提交”。

  在实例列表页面的“规格”中查看扩容/缩容后的代理规格。

说明

- 单机实例和没有配置镜像/仲裁队列的集群实例在变更规格过程中会有分钟级业务中断，客户端需要支持自动重连，建议在业务低峰时进行变更。
- 配置了镜像/仲裁队列的集群实例在变更规格过程中会有秒级业务中断，客户端需要支持自动重连，建议在业务低峰时进行变更。

----结束

7.7 设置实例的公网访问

操作场景

当您需要通过公网访问 RabbitMQ 实例时，需要开启该实例的公网访问功能。当业务不再使用公网访问功能时，也可以关闭实例的公网访问功能。

须知

公网访问与 VPC 内访问相比，可能存在网络丢包和抖动等情况，且访问时延有所增加，因此建议仅在业务开发测试阶段开启公网访问 RabbitMQ 实例。

前提条件

仅状态为“运行中”的实例，可以开启公网访问功能。

开启公网访问

步骤 1 登录管理控制台。

步骤 2 在管理控制台左上角单击，选择区域。

> **说明**
>
> 此处请选择 RabbitMQ 实例所在的区域。

步骤 3 在管理控制台左上角单击，选择“应用服务 > 分布式消息服务 RabbitMQ”，进入分布式消息服务 RabbitMQ 专享版页面。

步骤 4 单击待开启公网访问的实例名称，进入实例详情页面。

步骤 5 单击“公网访问”右侧的，打开公网访问开关。

步骤 6 从“弹性 IP 地址”下拉列表中选择一个弹性 IP，然后单击，开启公网访问功能。

如果“弹性 IP 地址”下拉列表没有值，可单击“创建弹性 IP”，跳转到弹性公网 IP 页面，您可以申请一个新的弹性 IP。弹性 IP 申请完后，返回 RabbitMQ 控制台，单击“弹性 IP 地址”后的，然后在下拉列表中选择新申请的弹性 IP。

开启公网访问功能大约需要 10~30 秒，请耐心等待。开启公网访问后，页面会自动跳转到“后台任务管理”页签，当任务状态为“成功”时，表示开启公网访问成功。

> **说明**
>
> 开启公网访问后，有如下注意事项：
>
> - 如果实例未开启 SSL，修改实例的安全组策略，增加入方向规则，允许端口 5672 和 15672 的访问。
>
>   访问 RabbitMQ 管理面：输入地址 http://*{RabbitMQ 实例公网 IP 地址}*:15672，然后输入自己配置的用户名和密码。
>
>   Client 方式：请使用 5672 端口。
>
> - 如果实例开启 SSL，修改实例的安全组策略，增加入方向规则，允许端口 5671 和 15671 的访问。
>
>   访问 RabbitMQ 管理面：输入地址 https://*{RabbitMQ 实例公网 IP 地址}*:15671，然后输入自己配置的用户名和密码。
>
>   Client 方式：请使用 5671 端口。

----结束

关闭公网访问

步骤 1 登录管理控制台。

步骤 2 在管理控制台左上角单击，选择区域。

> **说明**
>
> 此处请选择 RabbitMQ 实例所在的区域。

步骤 3 在管理控制台左上角单击，选择“应用服务 > 分布式消息服务 RabbitMQ”，进入分布式消息服务 RabbitMQ 专享版页面。

步骤 4 单击待关闭公网访问的实例名称，进入实例详情页面。

步骤 5 单击“公网访问”右侧的开关图标，关闭公网访问开关。

步骤 6 单击 ✓，关闭公网访问功能。

关闭公网访问功能大约需要 10~30 秒，请耐心等待。关闭公网访问后，页面会自动跳转到“后台任务管理”页签，当任务状态为“成功”时，表示关闭公网访问成功。

----结束

7.8 设置实例镜像队列

镜像队列，允许集群将队列镜像到其他节点上，当集群某一节点宕机后，队列能自动切换到镜像中的其他节点，保证服务的可用性。

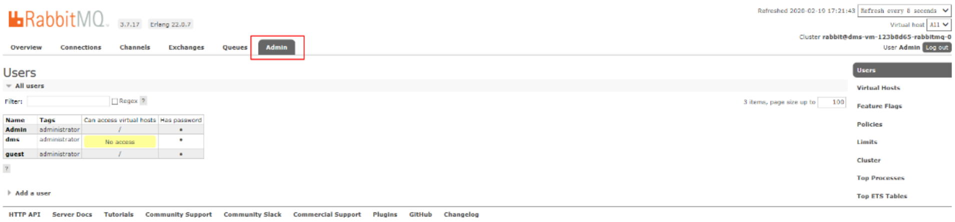
如果您需要了解 RabbitMQ Web UI 相关功能和概念，请自行查阅 RabbitMQ 官网。本章节仅介绍登录 RabbitMQ 实例的 Web 页面设置镜像队列的操作步骤。

操作步骤

步骤 1 登录 RabbitMQ 实例的 Web UI。

步骤 2 在菜单栏，选择“Admin”。

图7-1 选择 Admin 菜单



步骤 3 （可选）选择右侧导航栏“Virtual Hosts”，然后输入“Name”，单击“Add virtual host”，创建 Vhost。

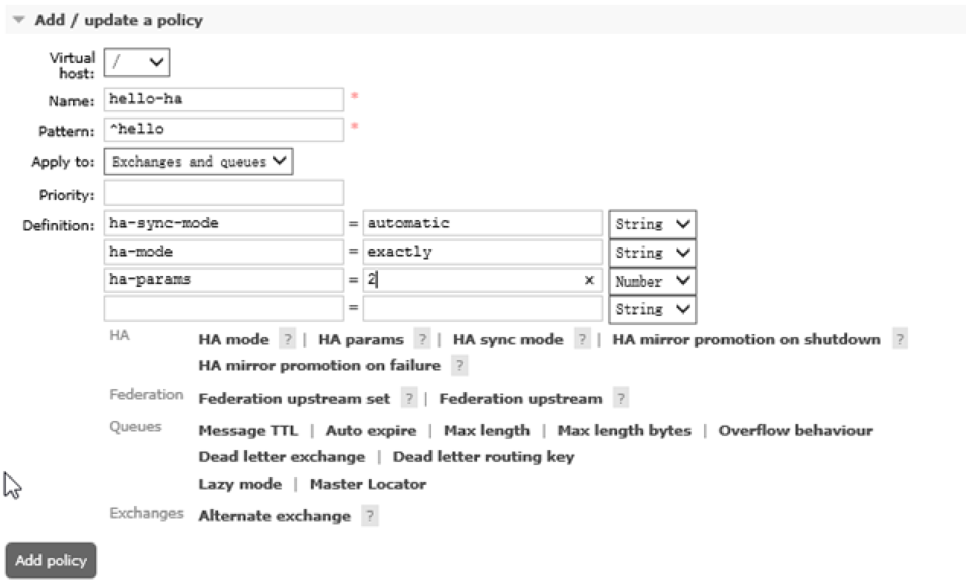
如果您需要设置指定 Vhost，请执行本步骤；如果不需要，请直接执行步骤 4。

图7-2 创建 Vhost

步骤 4 选择右侧导航栏“Policies”，为 Vhost 设置规则。

如果为指定的 Vhost 设置，请在“Virtual Host”选择步骤 3 创建的 Vhost；如果没有，则默认为“/”。

图7-3 设置 Vhost 规则



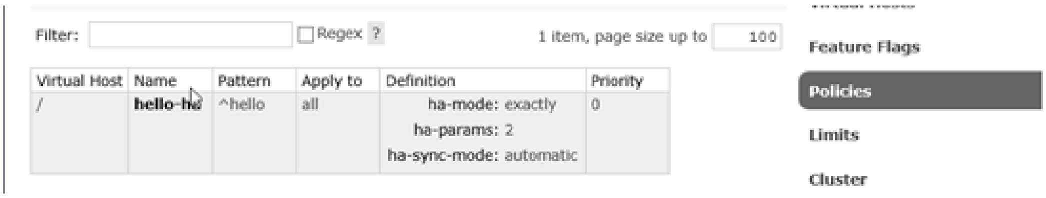
参数解释如下：

- Name: policy 的名称，用户自定义。
- Pattern: queue 的匹配模式（正则表达式）。
- Definition: 镜像定义，包括三个部分 ha-sync-mode、ha-mode、ha-params。
  - ha-sync-mode: 表示镜像队列中消息的同步方式，有效取值范围为：automatic 和 manually。
    - automatic：表示自动向 master 同步数据。
    - manually：表示手动向 master 同步数据。
  - ha-mode: 指明镜像队列的模式，有效取值范围为：all、exactly 和 nodes。
    - all：表示在集群所有的节点上进行镜像。
    - exactly：表示在指定个数的节点上进行镜像，节点的个数由 ha-params 指定。
    - nodes：表示在指定的节点上进行镜像，节点名称通过 ha-params 指定。
  - ha-params: ha-mode 模式需要用到的参数。
- Priority: 可选参数，policy 的优先级。

步骤 5 单击“Add policy”。

规则添加成功后如下图所示。

图7-4 Vhost 规则



----结束

7.9 管理实例标签

标签是 RabbitMQ 实例的标识，为 RabbitMQ 实例添加标签，方便您识别和管理拥有的 RabbitMQ 实例资源。

您可以在创建 RabbitMQ 实例时添加标签，也可以在 RabbitMQ 实例创建完成后，在“标签”页面添加标签，最多可以给实例添加 20 个标签。另外，您还可以进行修改和删除标签。

标签共由两部分组成：“标签键”和“标签值”，其中，“标签键”和“标签值”的命名规则如表 7-4 所示。

表7-4 标签命名规则

<table>
<tr><th>参数名称</th><th>规则</th></tr>
<tr><td>标签键</td><td>不能为空。 对于同一个实例，Key 值唯一。 长度不超过 36 个字符。 不能包含“=”,“*”,“<”,“>”,“\”,“,”,“|”,“/”。 首尾字符不能为空格。</td></tr>
<tr><td>标签值</td><td>不能为空。 长度不超过 43 个字符。 不能包含“=”,“*”,“<”,“>”,“\”,“,”,“|”,“/”。 首尾字符不能为空格。</td></tr>
</table>

操作步骤

步骤 1 登录管理控制台。

步骤 2 在管理控制台左上角单击，选择区域。

说明

此处请选择 RabbitMQ 实例所在的区域。

步骤 3 在管理控制台左上角单击☰，选择“应用服务 > 分布式消息服务 RabbitMQ”，进入分布式消息服务 RabbitMQ 专享版页面。

步骤 4 单击待设置标签的实例名称，进入实例详情页面。

步骤 5 单击“标签”页签，进入标签管理页面，页面显示该实例的标签列表。

步骤 6 您可以根据实际需要，执行以下操作：

- 添加标签
  a. 单击“创建/删除标签”，弹出“创建/删除标签”对话框。
  b. 在“标签键”和“标签值”中，输入标签的键/值，单击“添加”。
     如果您已经预定义了标签，在“标签键”和“标签值”中选择已经定义的标签键/值，单击“添加”。
  c. 单击“确定”，成功为实例添加标签。
- 删除标签
  通过以下任意一种方法，删除标签。
  – 在待删除的标签所在行，单击“删除”，弹出“删除标签”对话框。单击“是”，完成标签的删除。
  – 单击“创建/删除标签”，弹出“创建/删除标签”对话框。在待删除的标签后，单击⊗，然后单击“确定”，完成标签的删除。

----结束

7.10 按需转包周期

按需付费的用户可以选择“转包周期”，变更实例计费模式为包年/包月。

操作步骤

步骤 1 登录管理控制台。

步骤 2 在管理控制台左上角单击⊙，选择区域。

说明

此处请选择 RabbitMQ 实例所在的区域。

步骤 3 在管理控制台左上角单击☰，选择“应用中间件 > 分布式消息服务 RabbitMQ 版”，进入分布式消息服务 RabbitMQ 专享版页面。

步骤 4 通过以下任意一种方法，实现按需实例转包周期。

- 勾选 RabbitMQ 实例名称左侧的方框，可选一个或多个，单击信息栏左上侧的“转包周期”。

- 在待转包周期的 RabbitMQ 实例所在行，单击“更多 > 转包周期”。
- 单击 RabbitMQ 实例名称，进入实例详情页面。单击右上角的“更多 > 转包周期”。

步骤 5 单击“确定”，页面跳转到“按需转包周期”页面。

步骤 6 选择续费时长，然后单击“去支付”，根据界面提示信息，支付费用，完成实例转包周期操作。

----结束

7.11 删除队列

本章节指导您通过 RabbitMQ WebUI 页面或调用 API 方式删除队列。

- 方法一：在 WebUI 页面删除单个队列：在 WebUI 页面的“Queues”页签中，删除单个队列。
- 方法二：通过 Policy 批量删除队列：新增与待删除队列的前缀名称相同、且队列过期时间（TTL）为 1 毫秒的策略，通过此策略实现批量删除队列。
- 方法三：调用 API 删除单个队列：在 RabbitMQ 实例未开启 SSL 时，通过调用 API 删除单个队列。
- 方法四：调用 API 批量删除队列：在 RabbitMQ 实例未开启 SSL 时，通过编写 Shell 脚本循环调用 API 执行删除命令，实现批量删除队列。

须知

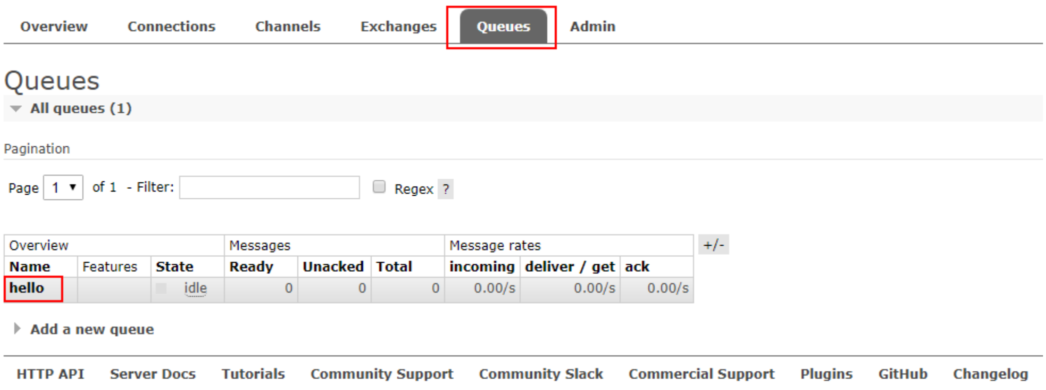
删除队列前，请确保队列中的消息已经全部被消费，否则未消费的消息将和队列一起被删除。

方法一：在 WebUI 页面删除单个队列

步骤 1 登录 RabbitMQ WebUI 页面。

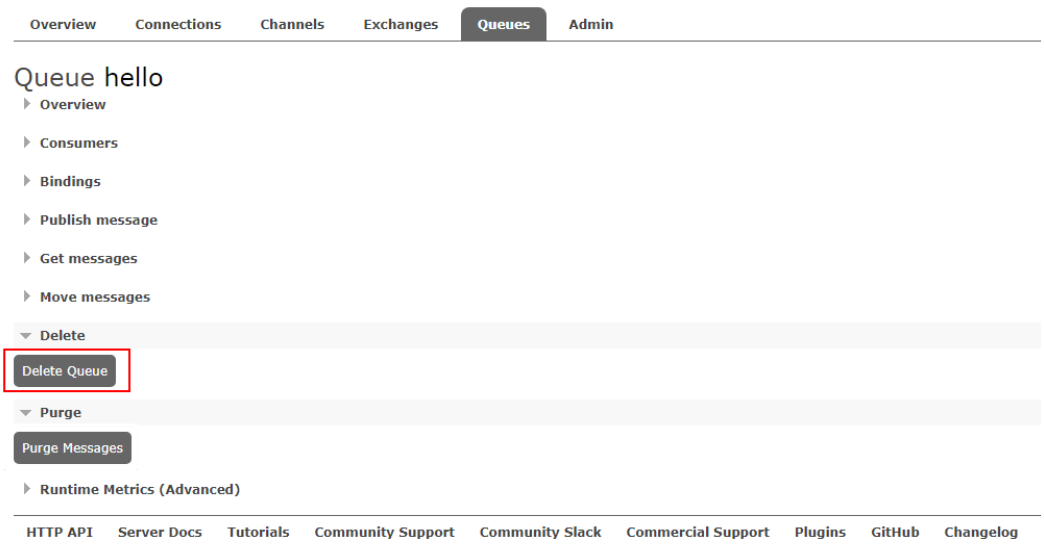
步骤 2 在“Queues”页签，单击需要删除的队列名称，进入队列详情页面。

图7-5 队列列表



步骤 3 单击“Delete Queue”，删除单个队列。

图7-6 删除单个队列



----结束

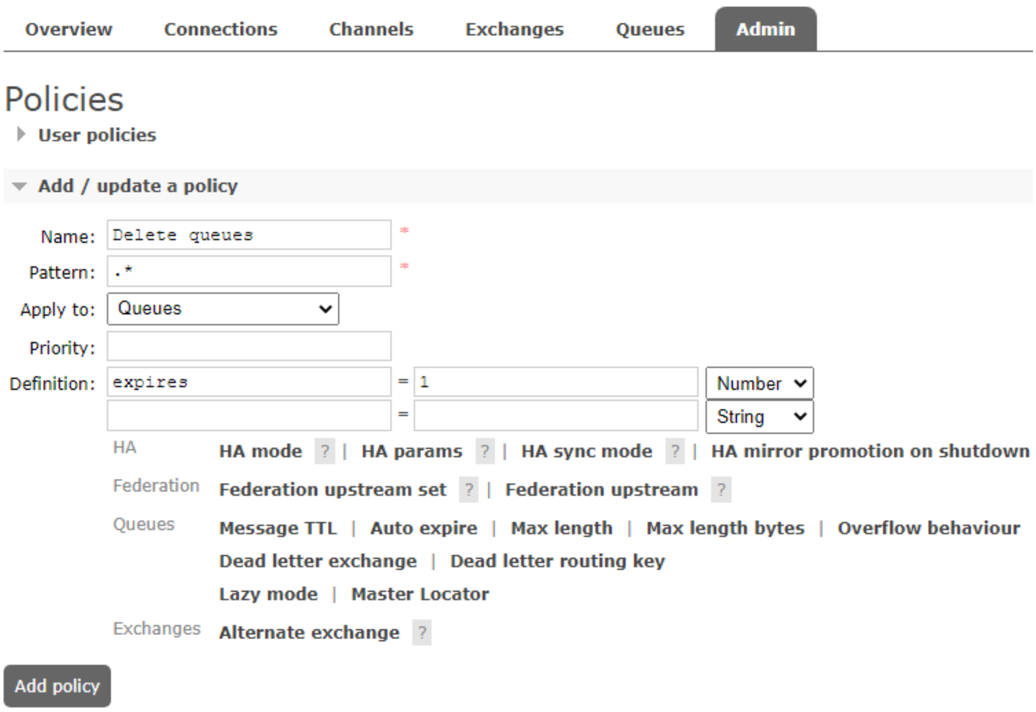
方法二：通过 Policy 批量删除队列

新增与待删除队列的前缀名称相同、且队列 TTL 为 1 毫秒的策略，通过此策略实现批量删除队列。

步骤 1 登录 RabbitMQ WebUI 页面。

步骤 2 在“Admin > Policies”页面中，新增一条策略。

图7-7 通过 Policy 批量删除队列



- Name：填写策略名称。
- Pattern：队列匹配模式，填写队列名称，会匹配前缀同名的队列。例如：设置为“.*”时，表示匹配所有队列。设置为“.*queue-name”时，表示匹配队列名前缀为 queue-name 的所有队列。
- Apply to：选择“Queues”。
- Priority：可选参数，策略优先级，数字越大，优先级越高。
- Definition：定义 TTL，单位为毫秒。填写“expires”参数，值设置为“1”，表示队列过期时间为 1 毫秒。

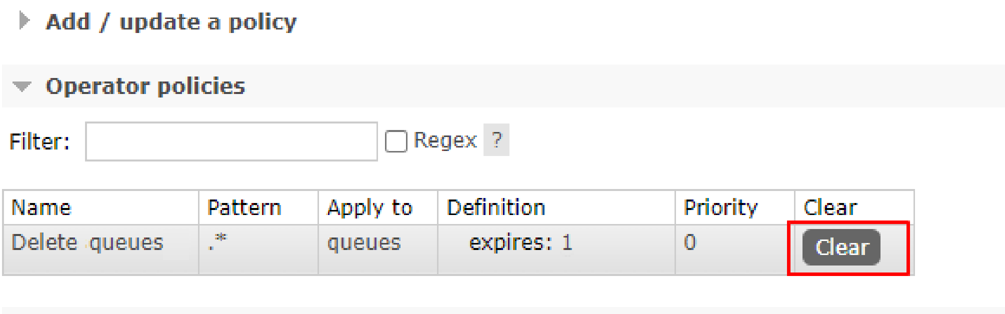
步骤 3 单击“Add policy”。

在“Queues”页签，查看队列是否成功删除。

步骤 4 队列成功删除后，在“Admin > Policies”页面中，在步骤 2 中新增的策略后，单击“Clear”，删除策略。

如果保留此策略，它对后续新建的队列依然生效，可能会出现误删除队列的情况。

图7-8 删除策略



----结束

方法三：调用 API 删除单个队列

在 RabbitMQ 实例未开启 SSL 时，通过调用 API 删除单个队列。

步骤 1 在 Linux 系统中连接 RabbitMQ 实例，具体步骤请参考连接未开启 SSL 方式的 RabbitMQ 实例。

步骤 2 执行以下命令删除单个队列。

```
curl -i -XDELETE
http://${USERNAME}:${PASSWORD}@${HOST}:${PORT}/api/queues/${VHOST_NAME}/${QUEUE_NAM
E}
```

参数说明如下：

- USERNAME：创建实例时设置的用户名。
- PASSWORD：创建实例时设置的密码，如果忘记密码，参考重置实例密码，重新设置密码。
- HOST：在实例详情页，查看 Web 界面 UI 地址。
- PORT：在实例详情页，查看 Web 界面 UI 端口号。
- VHOST_NAME：Vhost 名称，默认为“/”，在命令中设置为“%2F”。
- QUEUE_NAME：待删除队列的名称。

示例如下：

```
curl-i -XDELETE http://test:Zsxxxdx@192.168.0.241:15672/api/queues/%2F/hello
```

删除成功后，回显如下：

图7-9 删除队列成功

```
HTTP/1.1 204 No Content
content-security-policy: default-src 'self'
date: Tue, 14 Jun 2022 02:52:23 GMT
server: Cowboy
vary: accept, accept-encoding, origin
```

您还可以在 WebUI 页面的“Queues”页签，查看队列是否成功删除。

----结束

方法四：调用 API 批量删除队列

在 RabbitMQ 实例未开启 SSL 时，通过编写 Shell 脚本循环调用 API 执行删除命令，实现批量删除队列。

步骤 1 在 Linux 系统中连接 RabbitMQ 实例，具体步骤请参考连接未开启 SSL 方式的 RabbitMQ 实例。

步骤 2 创建“delete_queues.sh”脚本文件。

```
touch delete_queues.sh
```

步骤 3 执行以下命令，编辑脚本。

```
vim delete_queues.sh
```

将以下内容复制到脚本中，其中 USERNAME、PASSWORD、HOST 和 QUEUES_LIST 的值，请根据实际情况修改。

```
#!/usr/bin/env bash

USERNAME=root
PASSWORD=Zsxxxdx
HOST=192.168.0.241
PORT=15672
VHOST='%2F'


QUEUES_LIST="test1 test2 test3";
for QUEUE_NAME in $QUEUES_LIST :
do
   curl -i -XDELETE
http://$USERNAME:$PASSWORD@$HOST:$PORT/api/queues/$VHOST/$QUEUE_NAME
done
```

参数说明如下：

- USERNAME：创建实例时设置的用户名。
- PASSWORD：创建实例时设置的密码，如果忘记密码，参考重置实例密码，重新设置密码。
- HOST：在实例详情页，查看 Web 界面 UI 地址。

- PORT：在实例详情页，查看 Web 界面 UI 端口号。
- VHOST：Vhost 名称，默认为“/”，在命令中设置为“%2F”。
- QUEUES_LIST：待删除队列的名称，队列名称之间使用空格隔开。

步骤 4 保存脚本内容。

步骤 5 对脚本进行授权。

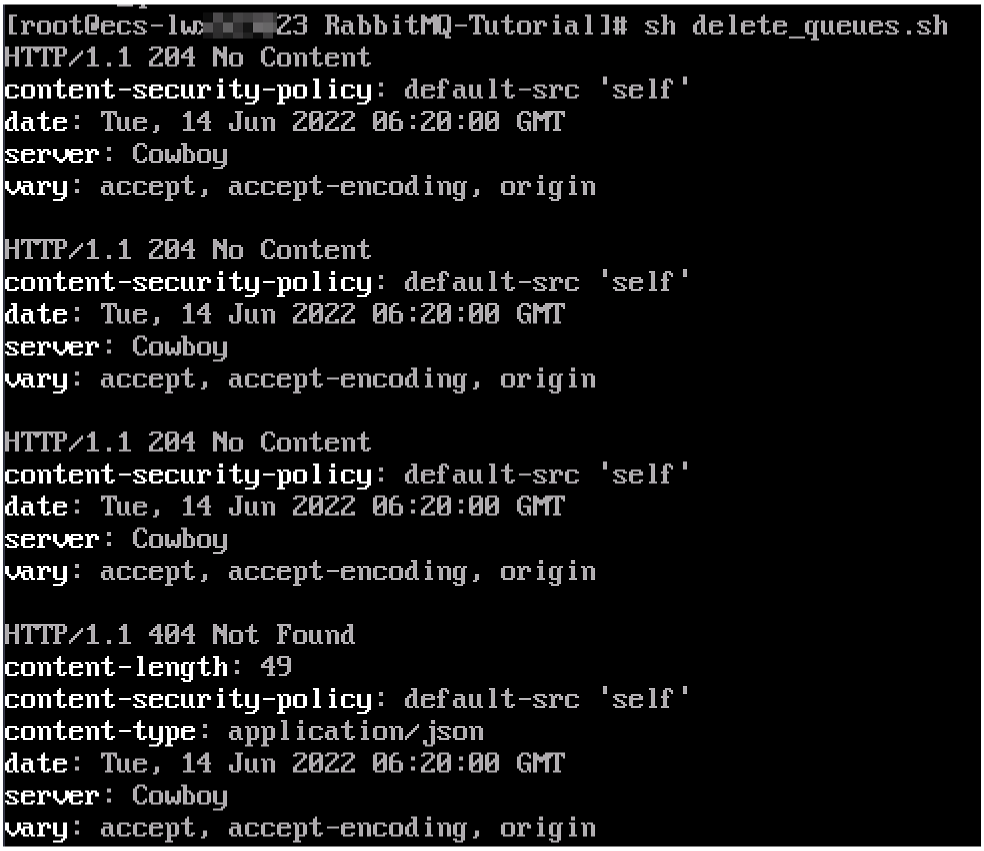
```
chmod 777 delete_queues.sh
```

步骤 6 执行脚本。

```
sh delete_queues.sh
```

删除成功后，回显如下：

图7-10 批量删除队列成功



```
[root@ecs-lw[redacted]23 RabbitMQ-Tutorial]# sh delete_queues.sh
HTTP/1.1 204 No Content
content-security-policy: default-src 'self'
date: Tue, 14 Jun 2022 06:20:00 GMT
server: Cowboy
vary: accept, accept-encoding, origin

HTTP/1.1 204 No Content
content-security-policy: default-src 'self'
date: Tue, 14 Jun 2022 06:20:00 GMT
server: Cowboy
vary: accept, accept-encoding, origin

HTTP/1.1 204 No Content
content-security-policy: default-src 'self'
date: Tue, 14 Jun 2022 06:20:00 GMT
server: Cowboy
vary: accept, accept-encoding, origin

HTTP/1.1 404 Not Found
content-length: 49
content-security-policy: default-src 'self'
content-type: application/json
date: Tue, 14 Jun 2022 06:20:00 GMT
server: Cowboy
vary: accept, accept-encoding, origin
```

您还可以在 WebUI 页面的“Queues”页签，查看队列是否成功删除。

----结束

8 插件管理

8.1 开启实例插件

RabbitMQ 实例创建后，支持开启如下插件，实例创建后默认都是关闭状态。

RabbitMQ 插件功能可用于测试和迁移业务等场景，不建议用于生产业务。详情请参考约束与限制。

说明

开启插件过程中，不会重启实例，但是以下插件（rabbitmq_mqtt、rabbitmq_web_mqtt、rabbitmq_stomp、rabbitmq_web_stomp）会重启 keepalived，导致连接断开。连接断开后，是否会自动重连依赖于用户自身的业务逻辑。

表8-1 支持修改状态的实例插件

插件名称	功能描述	端口号
rabbitmq_amqp1_0	表示实例是否支持 AMQP1.0 协议。	-
rabbitmq_delayed_message_exchange	表示实例是否开启消息延迟功能。 插件延迟时间存在 1%左右的误差，可能提前或者推迟发送消息给消费者。	-
rabbitmq_federation	表示实例是否开启消息同步功能。	-
rabbitmq_sharding	表示实例是否开启消息分片功能。	-
rabbitmq_shovel	表示实例是否开启消息迁移功能。	-
rabbitmq_tracing	表示实例是否开启消息追踪功能。	-
rabbitmq_mqtt	表示实例是否支持 MQTT 协议（TCP 方式）。	1883
rabbitmq_web_mqtt	表示实例是否支持 MQTT 协议（WebSocket 方式）。	15675
rabbitmq_stomp	表示实例是否支持 STOMP 协议	61613

插件名称	功能描述	端口号
	（TCP 方式）。	
rabbitmq_web_stomp	表示实例是否支持 STOMP 协议（WebSocket 方式）。	15674
rabbitmq_consistent_hash_exchange	表示实例是否支持 x-consistent-hash。	-

说明

插件端口号不支持修改。

操作步骤

步骤 1 登录管理控制台。

步骤 2 在管理控制台左上角单击，选择区域。

说明

此处请选择 RabbitMQ 实例所在的区域。

步骤 3 在管理控制台左上角单击，选择“应用服务 > 分布式消息服务 RabbitMQ”，进入分布式消息服务 RabbitMQ 专享版页面。

步骤 4 单击待开启插件的实例名称，进入实例详情页面。

步骤 5 在“插件管理”页签，单击待开启插件后的“开启”。

确认开启后，等待实例插件开启成功。

----结束

8.2 使用 rabbitmq_tracing 插件

操作场景

rabbitmq_tracing 插件提供消息追踪功能，它能追踪流入流出 RabbitMQ 的消息，并对其进行封装，将封装后的消息日志存入相应的 trace 文件中。

前提条件

已购买实例。

操作步骤

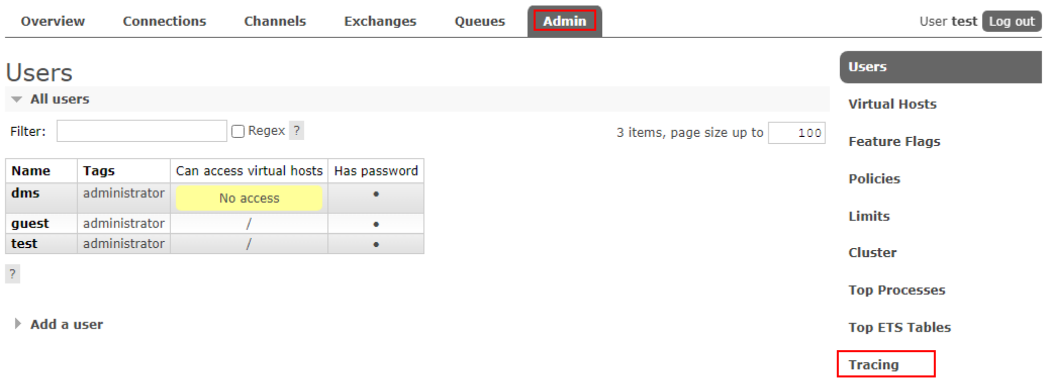
步骤 1 开启 rabbitmq_tracing 插件，具体步骤请参考开启实例插件。

步骤 2 登录 RabbitMQ WebUI 页面。

步骤 3 在顶部导航栏选择“Admin”，进入 Admin 页面。

步骤 4 在右侧导航栏选择“Tracing”，进入 Tracing 页面。

图8-1 Admin 页面



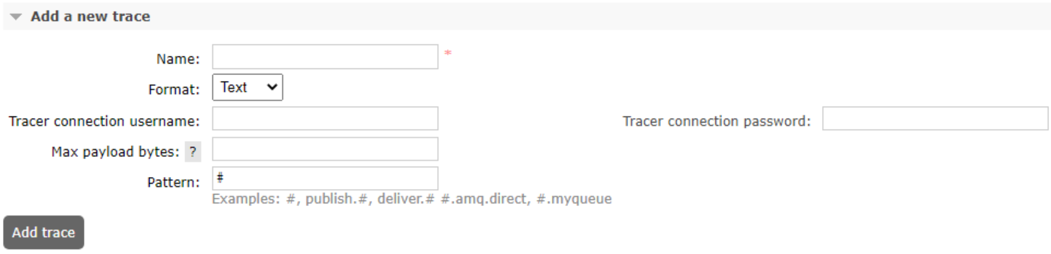
步骤 5 在“Add a new trace”区域，输入以下参数，单击“Add trace”，新增一个 trace。

表8-2 trace 参数说明

参数	说明
Name	自定义 trace 的名称，用于区分不同的 trace。
Format	输出消息日志的格式。支持“Text”和“JSON”两种格式，“Text”格式方便阅读，“JSON”格式方便解析。
Tracer connection username	指定创建 trace 的用户名。
Tracer connection password	指定创建 trace 的密码。
Max payload bytes	每条消息的最大限制，单位为 B。 假设“Max payload bytes”设置为“10”，当有超过 10B 的消息经过 RabbitMQ 流转时就会被载断，例如“trace test payload”会被载断成“trace test”。
Pattern	设置匹配的模式。取值示例如下： • #：追踪所有进入和离开 RabbitMQ 的消息 • publish.#：追踪所有进入 RabbitMQ 的消息 • deliver.#：追踪所有离开 RabbitMQ 的消息 • publish.delay_exchange：追踪进入指定交换机的信息，delay_exchange 为交换机名称，请根据实际情况修改。 • deliver.delay_queue：追踪离开指定队列的消息，

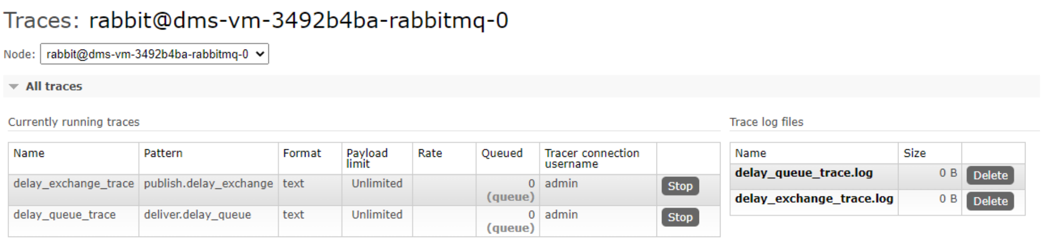
参数	说明
	delay_queue 为队列名称，请根据实际情况修改。

图8-2 新增一个 trace



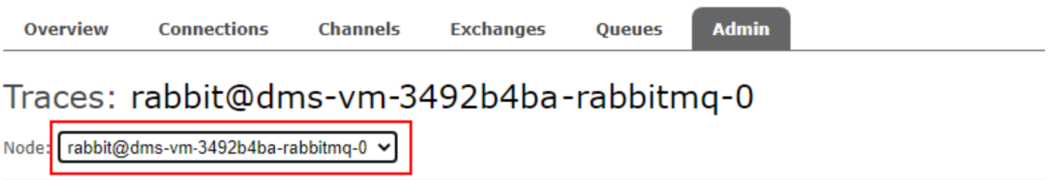
trace 创建成功后，在“All traces”区域，显示已创建的 trace 列表。

图8-3 trace 列表



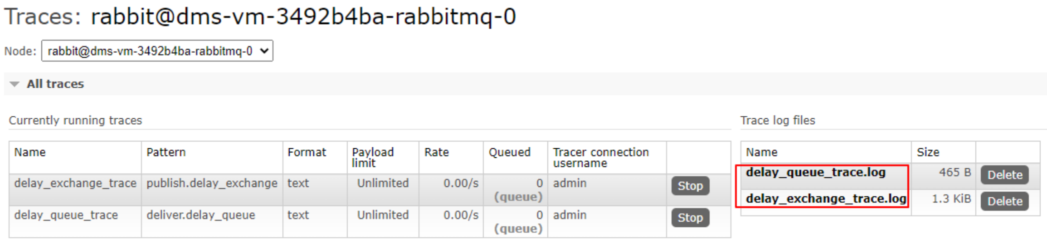
步骤 6 （可选）如果 RabbitMQ 实例为集群，在“Node”中切换到其他节点，重复步骤 5，为其他所有节点创建 trace。

图8-4 切换节点



步骤 7 当 trace 日志文件中存入消息日志后，单击 trace 日志文件名称，查看日志内容。

图8-5 trace 日志文件



“delay_exchange_trace.log”的日志内容如图 8-6 所示。

图8-6 delay_exchange_trace.log

```
================================================================================
2022-07-20 3:22:32:837: Message published

Node:          rabbit@dms-vm-3492b4ba-rabbitmq-0
Connection:    <rabbit@dms-vm-3492b4ba-rabbitmq-0.1657790484.10274.7>
Virtual host:  /
User:          admin
Channel:       1
Exchange:      delay_exchange
Routing keys:  [<<>>]
Routed queues: []
Properties:    [{<<"delivery_mode">>,signedint,2},{<<"headers">>,table,[]}]
Payload:
hello world
```

“delay_queue_trace.log”的日志内容如图 8-7 所示。

图8-7 delay_queue_trace.log

```
================================================================================
2022-07-20 3:23:22:468: Message received

Node:          rabbit@dms-vm-3492b4ba-rabbitmq-0
Connection:    <rabbit@dms-vm-3492b4ba-rabbitmq-0.1657790484.10565.7>
Virtual host:  /
User:          admin
Channel:       1
Exchange:
Routing keys:  [<<"delay_queue">>]
Queue:         delay_queue
Properties:    [{<<"delivery_mode">>,signedint,1},{<<"headers">>,table,[]}]
Payload:
hello world
```

----结束

9 Vhost 管理

9.1 创建 Vhost

操作场景

每个 Vhost（Virtual Hosts）相当于一个相对独立的 RabbitMQ 服务器。Vhost 用作逻辑隔离，分别管理 Exchange、Queue 和 Binding，使得应用安全地运行在不同的 Vhost 上，相互之间不会干扰。一个实例下可以有多个 Vhost，一个 Vhost 里可以有若干个 Exchange 和 Queue。生产者和消费者连接 RabbitMQ 实例时，需要指定一个 Vhost。Vhost 的相关介绍，请参考官网文档 Virtual Hosts。

本章节主要介绍创建 Vhost 的操作，有以下几种方式，您可以根据实际情况选择任意一种方式：

- 方式一：在控制台创建
- 方式二：使用 RabbitMQ WebUI 创建
- 方式三：调用 API 创建

方式一：在控制台创建

步骤 1 登录管理控制台。

步骤 2 在管理控制台左上角单击 ，选择区域。

说明

此处请选择 RabbitMQ 实例所在的区域。

步骤 3 在管理控制台左上角单击 ，选择“应用服务 > 分布式消息服务 RabbitMQ”，进入分布式消息服务 RabbitMQ 专享版页面。

步骤 4 单击实例名称，进入实例详情页面。

步骤 5 在左侧导航栏选择“Vhost 列表”，进入 Vhost 列表页面。

步骤 6 单击“创建 Vhost”，弹出“创建 Vhost”对话框。

步骤 7 设置 Vhost 的名称，单击“确定”。

创建成功后，在 Vhost 列表页面显示创建成功的 Vhost。

图9-1 Vhost 列表（控制台）

名称	tracing	操作
/	否	删除
test-vhost	否	删除

“tracing”表示是否开启消息追踪功能。开启消息追踪后，您可以跟踪消息的转发路径。

说明

- Vhost 创建成功后，无法修改名称。
- 实例创建后，会自动创建一个名为“/”的 Vhost。

----结束

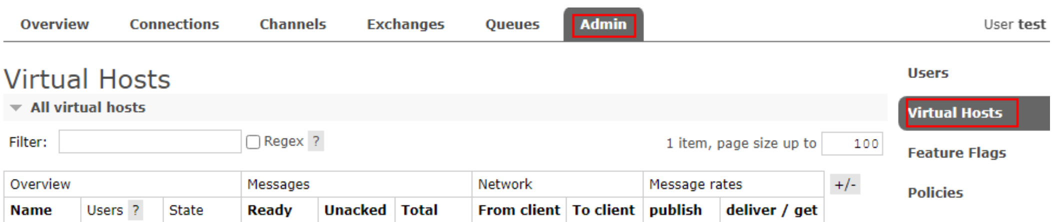
方式二：使用 RabbitMQ WebUI 创建

步骤 1 登录 RabbitMQ WebUI。

步骤 2 在顶部导航栏选择“Admin”，进入 Admin 页面。

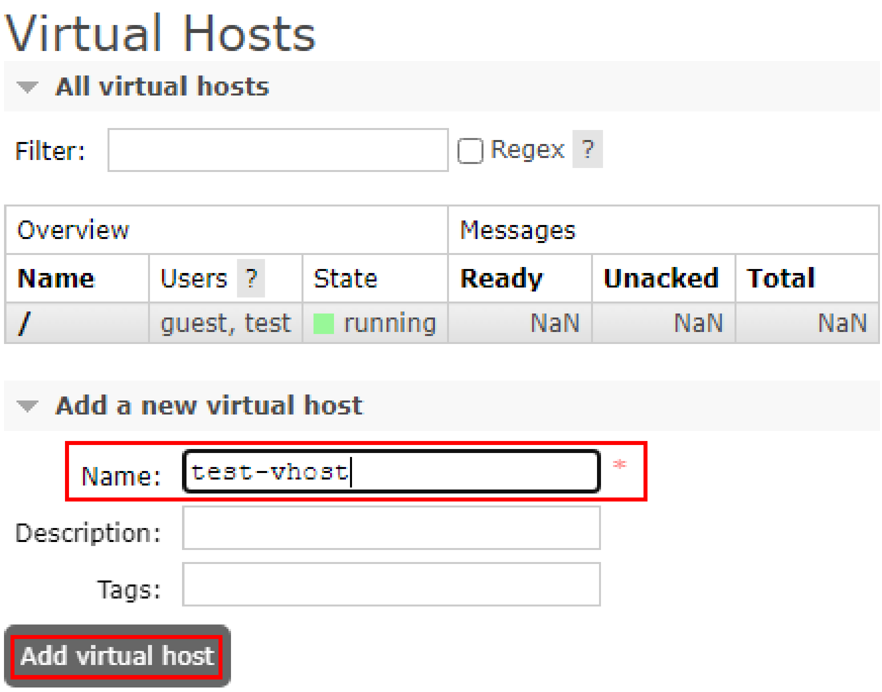
步骤 3 在右侧导航栏选择“Virtual Hosts”，进入 Virtual Hosts 页面。

图9-2 Virtual Hosts



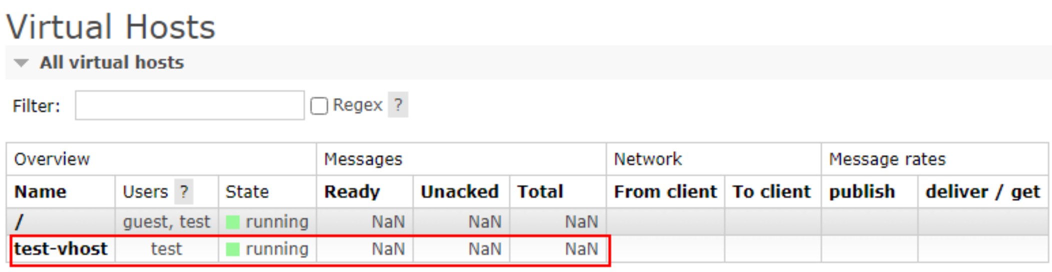
步骤 4 在“Add a new virtual host”区域，输入 Vhost 名称，单击“Add virtual host”。

图9-3 创建 Vhost（WebUI）



创建成功后，在“All virtual hosts”区域，显示创建成功的 Vhost。

图9-4 Vhost 列表（WebUI）



----结束

方式三：调用 API 创建

步骤 1 在 Linux 中，连接 RabbitMQ 实例。

步骤 2 执行以下命令，创建 Vhost。

```
curl -i -X PUT
http://${USERNAME}:${PASSWORD}@${HOST}:${PORT}/api/vhosts/${VHOST_NAME}
```

参数说明如下：

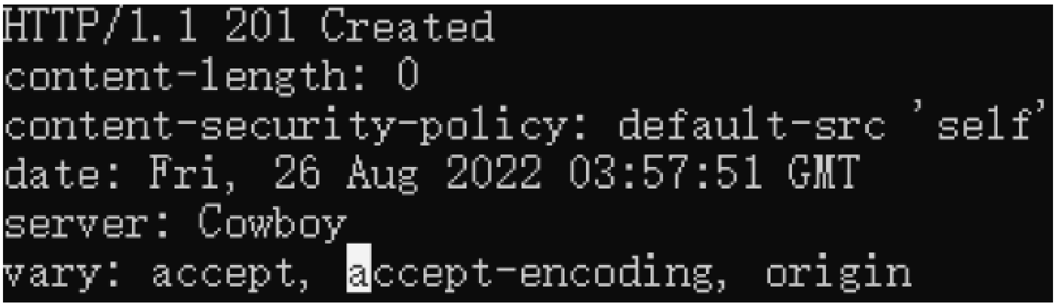
- USERNAME：创建 RabbitMQ 实例时，设置的用户名。在实例详情页的“连接信息”区域，查看用户名。
- PASSWORD：创建 RabbitMQ 实例时，设置的密码。如果忘记密码，参考重置实例密码，重新设置密码。
- HOST：WebUI 的地址。在实例详情页的“连接信息”区域，查看 Web 界面 UI 地址。
- PORT：WebUI 的端口号。在实例详情页的“连接信息”区域，查看 Web 界面 UI 地址中的端口号。
- VHOST_NAME：待创建 Vhost 的名称。

示例如下：

```
curl -i -X PUT http://root:txxxt@192.168.1.3:15672/api/vhosts/vhost-demo
```

创建成功后，回显如下所示。

图9-5 Vhost 创建成功



----结束

9.2 删除 Vhost

操作场景

本章节主要介绍删除 Vhost 的操作，有以下几种方式，您可以根据实际情况选择任意一种方式：

- 方式一：在控制台删除
- 方式二：使用 RabbitMQ WebUI 删除
- 方式三：调用 API 删除

方式一：在控制台删除

步骤 1 登录管理控制台。

步骤 2 在管理控制台左上角单击，选择区域。

说明

此处请选择 RabbitMQ 实例所在的区域。

步骤 3 在管理控制台左上角单击，选择“应用服务 > 分布式消息服务 RabbitMQ”，进入分布式消息服务 RabbitMQ 专享版页面。

步骤 4 单击实例名称，进入实例详情页面。

步骤 5 在左侧导航栏选择“Vhost 列表”，进入 Vhost 列表页面。

步骤 6 通过以下任意一种方法，删除 Vhost。

- 勾选 Vhost 名称左侧的方框，可选一个或多个，单击信息栏左上侧的“删除 Vhost”。
- 在待删除的 Vhost 所在行，单击“删除”。

步骤 7 在弹出的确认删除对话框中，单击“是”。

----结束

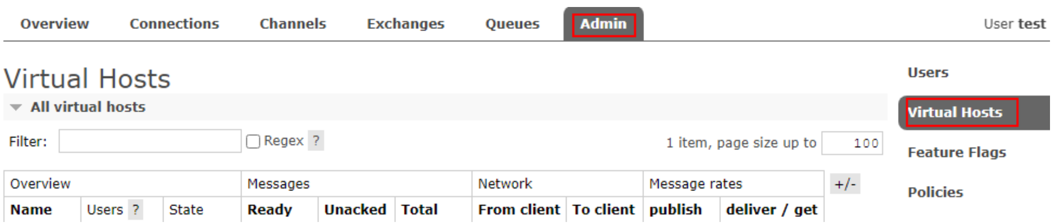
方式二：使用 RabbitMQ WebUI 删除

步骤 1 登录 RabbitMQ WebUI。

步骤 2 在顶部导航栏选择“Admin”，进入 Admin 页面。

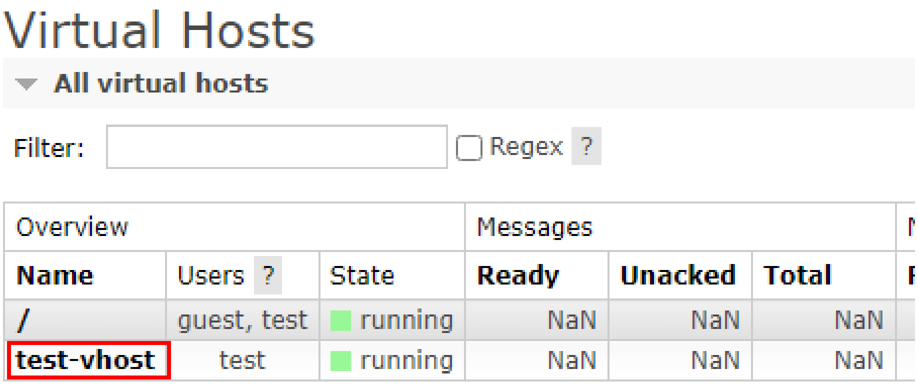
步骤 3 在右侧导航栏选择“Virtual Hosts”，进入 Virtual Hosts 页面。

图9-6 Virtual Hosts 页面



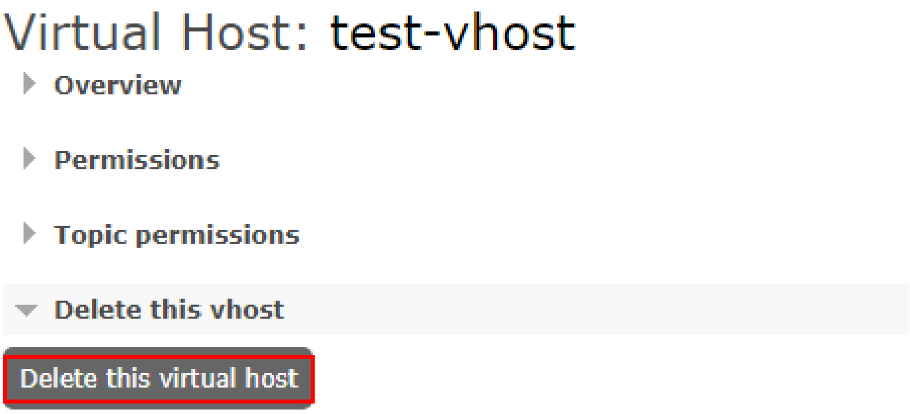
步骤 4 单击待删除的 Vhost 名称，进入 Vhost 详情页。

图9-7 待删除的 Vhost



步骤 5 在“Delete this vhost”区域，单击“Delete this virtual host”，弹出确认删除对话框。

图9-8 删除 Vhost



步骤 6 单击“确定”，完成 Vhost 的删除。

----结束

方式三：调用 API 删除

步骤 1 在 Linux 中，连接 RabbitMQ 实例。

步骤 2 执行以下命令，删除 Vhost。

```
curl -i -X DELETE
http://${USERNAME}:${PASSWORD}@${HOST}:${PORT}/api/vhosts/${VHOST_NAME}
```

参数说明如下：

- USERNAME：创建 RabbitMQ 实例时，设置的用户名。在实例详情页的“连接信息”区域，查看用户名。

- PASSWORD：创建 RabbitMQ 实例时，设置的密码。如果忘记密码，参考重置实例密码，重新设置密码。
- HOST：WebUI 的地址。在实例详情页的“连接信息”区域，查看 Web 界面 UI 地址。
- PORT：WebUI 的端口号。在实例详情页的“连接信息”区域，查看 Web 界面 UI 地址中的端口号。
- VHOST_NAME：待删除 Vhost 的名称。

示例如下：

```
curl -i -X DELETE http://root:txxxt@192.168.1.3:15672/api/vhosts/vhost-demo
```

删除成功后，回显如下所示。

图9-9 Vhost 删除成功

```
HTTP/1.1 204 No Content
content-security-policy: default-src 'self'
date: Fri, 26 Aug 2022 04:26:50 GMT
server: Cowboy
vary: accept, accept-encoding, origin
```

----结束

10 高级特性

10.1 惰性队列

使用场景

默认情况下，RabbitMQ 生产者生产的消息存储在内存中，当需要释放内存时，会将内存中的消息换页至磁盘中。换页操作会消耗较长的时间，且换页过程中队列无法处理消息。

如果生产速度过快（例如执行批处理任务），或者消费者由于各种原因（例如消费者下线、宕机）长时间内无法消费消息，导致消息大量堆积，使得内存使用率过高，换页频繁，可能会影响其他队列的消息收发。这种场景下，建议您启用惰性队列。

惰性队列（Lazy Queue）会尽可能的将消息存入磁盘中，在消费者消费到相应的消息时才会被加载到内存中，这样可以减少内存的消耗，但是会增加 I/O 的使用，影响单个队列的吞吐量。惰性队列的一个重要的设计目标是能够支持更长的队列，即支持更多的消息存储/消息堆积。

在以下情况下，推荐使用惰性队列：

- 队列可能会产生消息堆积
- 队列对性能（吞吐量）的要求不是非常高，例如 TPS 1 万以下的场景
- 希望队列有稳定的生产消费性能，不受内存影响而波动

处于以下情况时，无需使用惰性队列：

- RabbitMQ 需要高性能的场景
- 队列总是很短（即队列中没有消息堆积）
- 设置了最大长度策略

更多关于惰性队列的说明，请参考 Lazy Queues。

设置惰性队列

队列具备两种模式：default 和 lazy，默认模式为 default。lazy 模式即为惰性队列的模式，可以通过调用 **channel.queueDeclare** 方法的时候在参数中设置，也可以通过 Policy

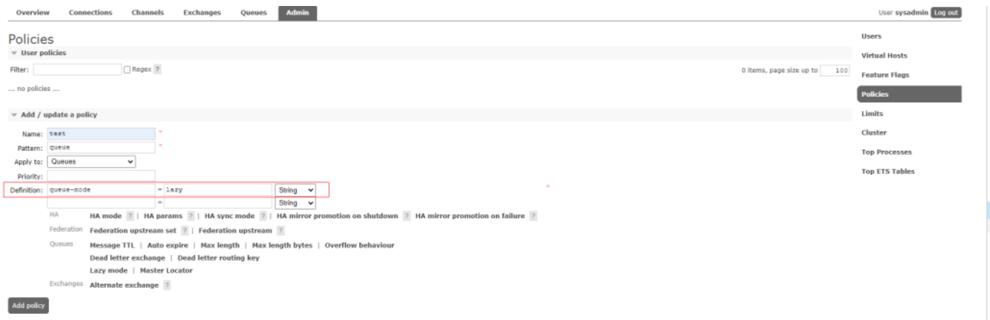
的方式设置。如果一个队列同时使用这两种方式设置的话，Policy 的方式具备更高的优先级。

- **以下示例演示在 Java 客户端通过调用 channel.queueDeclare 设置惰性队列。**

```
Map<String, Object> args = new HashMap<String, Object>();
args.put("x-queue-mode", "lazy");
channel.queueDeclare("myqueue", false, false, false, args);
```

- **以下示例演示在** RabbitMQ WebUI 页面**通过 Policy 的方式设置惰性队列。**

图10-1 通过 Policy 的方式设置惰性队列



10.2 消息持久化

使用场景

默认情况下，RabbitMQ 生产者生产的消息存储在内存中，当节点宕机或重启时，如何确保消息不丢失呢？RabbitMQ 通过持久化机制实现，持久化包括 Exchange 持久化、Queue 持久化和 Message 持久化。**持久化是将内存中的消息写入到磁盘中，以防异常情况导致内存中的消息丢失。但是磁盘的读写速度远不如内存，开启消息持久化后，RabbitMQ 的性能会下降。**与惰性队列不同，持久化消息会在磁盘和内存中各存储一份，只有在内存空间不够时，才会将内存中的消息删除，存储到磁盘中。

须知

- 非持久化 Queue、Exchange 在重启之后会丢失。
- 非持久化 Message 在重启之后会丢失（经过持久化 Queue/Exchange 的消息不会自动变为持久化消息）。
- 持久化消息在尚未完成持久化时，如果服务器重启，消息会丢失。

设置 Exchange 持久化

在 RabbitMQ WebUI 页面创建 Exchange 时，设置“durable”为“true”，如图 10-2 所示，设置成功后如图 10-3 所示。

图10-2 设置 Exchange 持久化

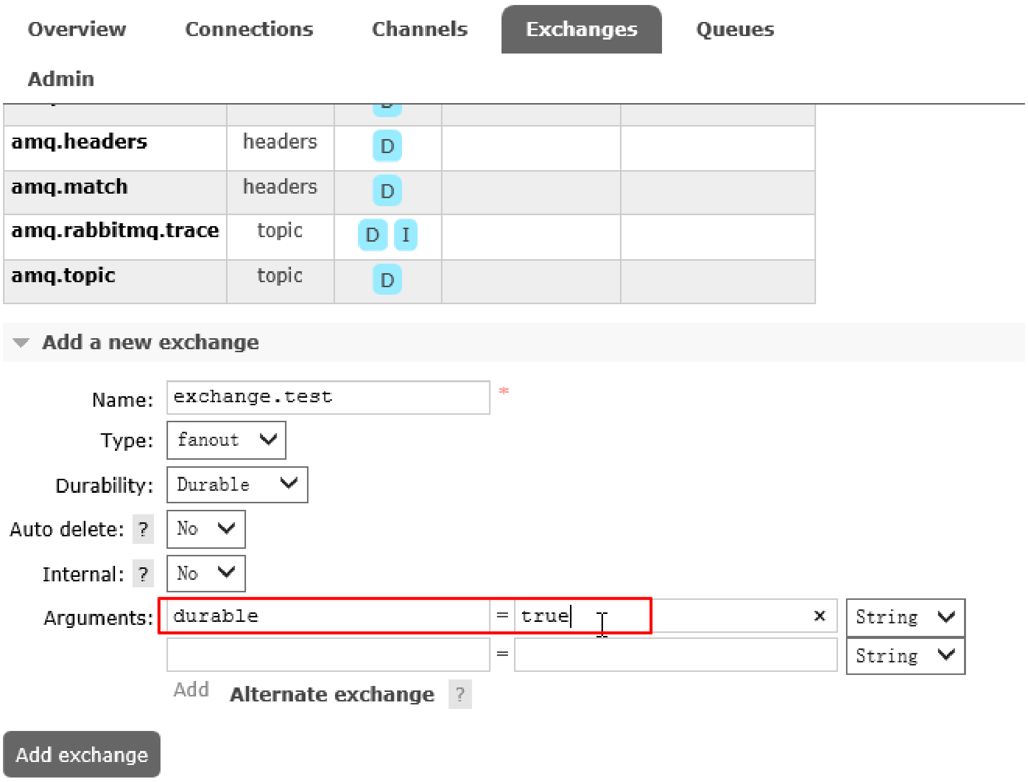
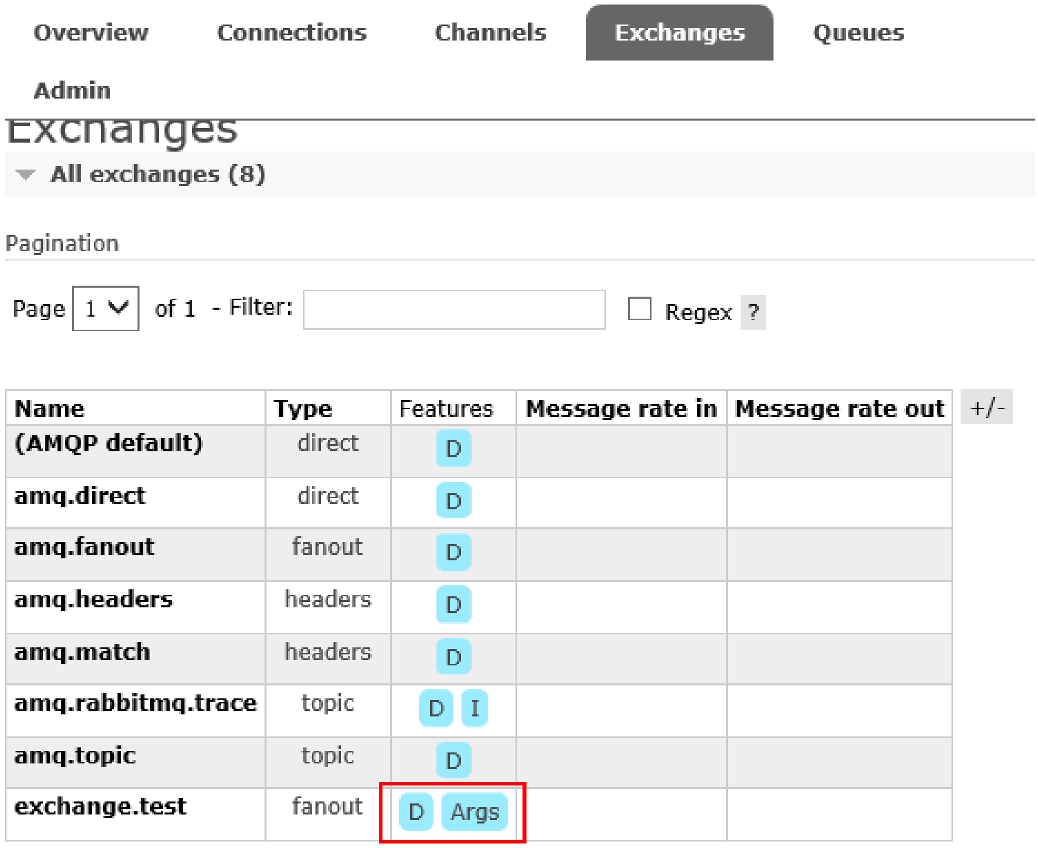


图10-3 持久化的 Exchange



Name	Type	Features	Message rate in	Message rate out	+/-
(AMQP default)	direct	D			
amq.direct	direct	D			
amq.fanout	fanout	D			
amq.headers	headers	D			
amq.match	headers	D			
amq.rabbitmq.trace	topic	D I			
amq.topic	topic	D			
exchange.test	fanout	D Args			

设置 Queue 持久化

在 RabbitMQ WebUI 页面创建 Queue 时，设置“durable”为“true”，如图 10-4 所示，设置成功后如图 10-5 所示。

图10-4 设置 Queue 持久化

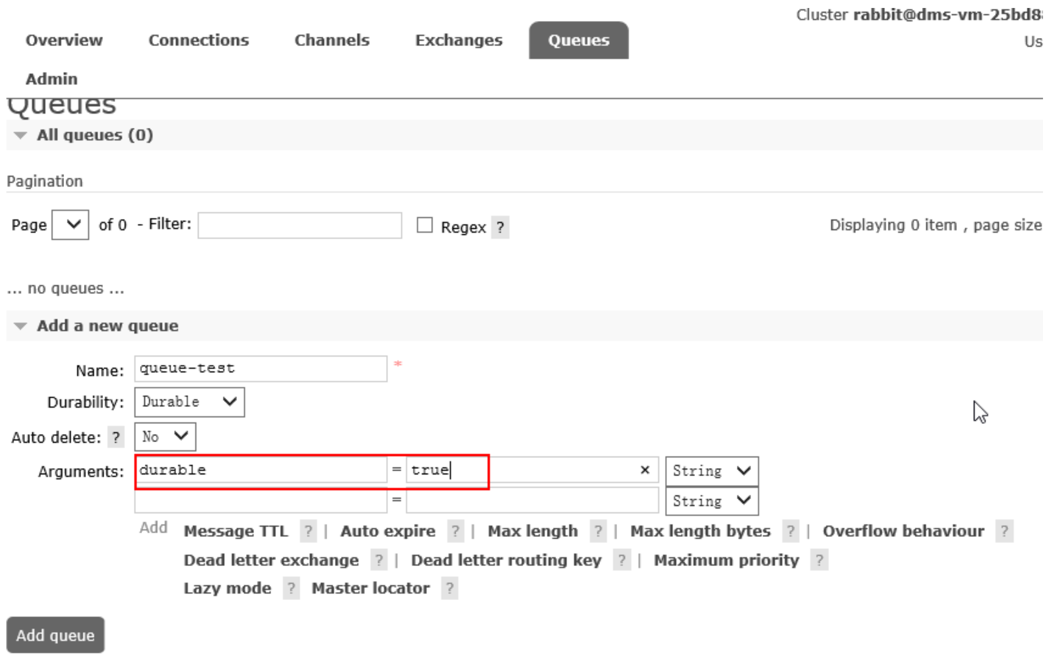
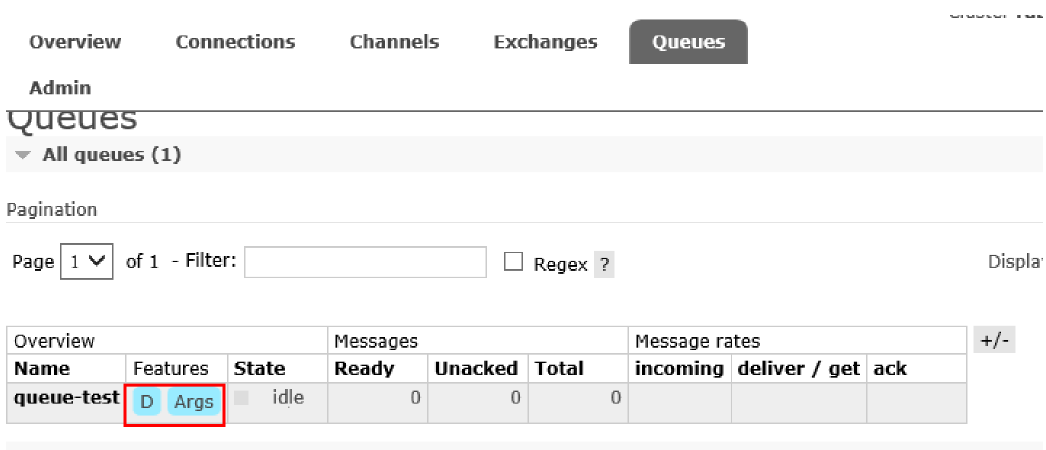


图10-5 持久化的 Queue



设置 Message 持久化

Queue 设置为持久化后，通过设置“MessageProperties”为“PERSISTENT_TEXT_PLAIN”来向 Queue 发送持久消息。

以下示例演示在 Java 客户端设置 Message 持久化

```
import com.rabbitmq.client.MessageProperties;
channel.basicPublish("", "my_queue",MessageProperties.PERSISTENT_TEXT_PLAIN,
message.getBytes());
```

10.3 死信和 TTL

死信和 TTL（Time To Live）是 RabbitMQ 中需要慎用的 2 个特性，它们可能会对性能产生负面影响。

死信

死信是 RabbitMQ 中的一种消息机制，在消费消息时，如果队列里的消息满足以下任意一种情况，那么该消息将成为“死信”。

- “requeue”被设置为“false”，消费者使用“basic.reject”或“basic.nack”否定应答（NACK）消息。
- 消息在队列的存活时间超过设置的 TTL 时间。
- 队列的消息数量已经超过最大队列长度。

死信消息会被 RabbitMQ 进行特殊处理，如果配置了死信队列信息，该消息将会被存储到死信队列中，如果没有配置，该消息将会被丢弃。

更多关于死信的说明，请参考 Dead Letter Exchanges。

使用队列参数配置死信交换机和路由

为队列配置死信交换机，并在申明队列时指定“x-dead-letter-exchange”和“x-dead-letter-routing-key”参数。队列根据“x-dead-letter-exchange”将死信消息发送到死信交换机中，并根据“x-dead-letter-routing-key”为死信消息设置死信路由 Key。

以下示例演示在 Java 客户端配置死信交换机和路由

```
channel.exchangeDeclare("some.exchange.name", "direct");

Map<String, Object> args = new HashMap<String, Object>();
args.put("x-dead-letter-exchange", "some.exchange.name");
args.put("x-dead-letter-routing-key", "some-routing-key");
channel.queueDeclare("myqueue", false, false, false, args);
```

TTL

TTL 即过期时间。RabbitMQ 支持设置消息和队列的 TTL，消息的 TTL 可以通过以下两种方法设置：

- 通过队列属性设置：队列中所有消息的具有相同的过期时间。
- 对消息本身单独设置：每条消息可以设置不同的 TTL。

如果两种方法同时使用，以较小的 TTL 为准。

消息在队列中的生存时间超过了 TTL 后，消息会被丢弃，如果队列设置了死信交换机，丢弃的消息会被转发到死信交换机，由死信交换机将其路由到死信队列。

更多关于 TTL 的说明，请参考 TTL。

设置队列 TTL

通过 **channel.queueDeclare** 方法中的“x-expires”参数控制队列被自动删除前处于未使用状态的时间。未使用是指队列中没有任何消费者，也没有被重新声明，并且在过期

时间段内也未调用过 **Basic.Get** 命令。“x-expires”参数的值必须为非零整数，单位为毫秒。

以下示例演示在 Java 客户端设置队列 TTL。

```
Map<String, Object> args = new HashMap<String, Object>();
args.put("x-expires", 1800000);
channel.queueDeclare("myqueue", false, false, false, args);
```

设置消息 TTL

通过队列属性设置消息 TTL：在 **channel.queueDeclare** 方法中加入“x-message-ttl”参数，此参数的值必须为非零整数，单位为毫秒。

以下示例演示在 Java 客户端通过队列属性设置消息 TTL。

```
Map<String,Object> arg = new HashMap<String, Object>();
arg.put("x-message-ttl",6000);
channel.queueDeclare("normalQueue",true,false,false,arg);
```

对消息本身单独设置 TTL：在 **channel.basicPublish** 方法中加入“expiration”参数，此参数的值必须为非零整数，单位为毫秒。

以下示例演示在 Java 客户端对消息本身单独设置 TTL。

```
byte[] messageBodyBytes = "Hello, world!".getBytes();
AMQP.BasicProperties properties = new AMQP.BasicProperties.Builder()
                             .expiration("60000")
                             .build();
channel.basicPublish("my-exchange", "routing-key", properties, messageBodyBytes);
```

10.4 RabbitMQ 消息确认机制

使用场景

生产者在发布消息后，怎么确认消息已正确发布到服务器端？服务器端又怎么确认消息已成功被消费？RabbitMQ 提供的消息确认机制可以解决此问题。

在使用 RabbitMQ 时，生产者确认和消费者确认对于确保数据可靠性至关重要。如果连接失败，传输中的消息可能会丢失，需要重新传输。确认机制可以让服务器和客户端知道何时重新传输消息。客户端可以在收到消息时确认消息，也可以在客户端完全处理完消息后确认。**生产者确认会影响性能，如果需要很高的吞吐量，应禁用生产者确认。注意，不使用生产者确认会导致可靠性下降。**

更多关于消息确认机制的说明，请参考 Consumer Acknowledgements and Publisher Confirms。

生产者确认

生产者确认，即服务端在收到来自生产者的消息时进行确认。

以下示例演示在 Java 客户端配置生产者确认：

```
try {
    channel.confirmSelect() ; //将信道置为publisher confirm模式
    //之后正常发送消息
    channel . basicPublish( "exchange " , " routingKey" , null , "publisher confirm
test " .getBytes());
    if (!channel.waitForConfirms()) {
        System.out.println( "send message failed " ) ;
        // do something else....
    }
} catch (InterruptedException e) {
        e.printStackTrace() ;
}
```

调用 **channel .waitForConfirms** 方法之后，会等待服务端确认，这是一种同步等待的方式，会对性能产生影响。如果生产者要满足 at least once，就必须使用同步等待方式。

消费者确认

消费者确认是指服务端通过确认消息是否成功被消费者接收，来判断是否删除队列中的此消息。

消费者确认对数据可靠性十分重要，接收重要消息的消费应用程序在未处理完消息前不应确认消息，以便消费者有足够的时间处理消息，无需担心消息处理过程中由于消费者进程异常（如工作程序崩溃、重启等）导致消息丢失。

消费者确认在客户端上配置，通过配置 **basicConsume** 方法启用确认。在 channel 中启用消费者确认适用于大多数场景。

以下示例演示在 Java 客户端配置消费者确认（使用 **Channel#basicAck** 设置 **basic.ack** 为肯定）：

```
// 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);
       }
    });
```

未确认的消息缓存在内存中，如果未确认的消息过多，会导致内存使用率过高，此时可以在客户端配置预取值来限制消费者预取的消息数量，具体方法请参见预取值。

10.5 预取值

使用场景

设置预取值可以限制未被确认的消息个数，一旦消费者中未被确认的消息数量达到设置的预取值，服务端将不再向此消费者发送消息，除非至少有一个未被确认的消息被确认。设置预取值本质上是一种对消费者进行流控的方法。

设置预取值时，需要考虑多种因素：

- 预取值设置太小可能会损害性能，RabbitMQ 会一直在等待获得发送消息的权限。
- 预取值设置太大可能会导致从队列中取出大量消息传递给一个消费者，而使其他消费者处于空闲状态。另外还需要考虑消费者的配置，消费者在处理消息时会将所有消息保存在内存中，太大的预取值会对消费者的性能产生负面影响，甚至可能会导致消费者崩溃。

更多关于预取值的说明，请参考 Consumer Prefetch。

如何设置合适的预取值？

- 如果您只有一个或很少几个消费者在处理消息，建议一次预取多条消息，尽量让客户端保持忙碌。如果您的处理时间和网络状态稳定，则只需将总往返时间除以每条消息在客户端的处理时间即可获得估计的预取值。
- 在消费者多且处理时间短的情况下，建议使用较低的预取值。过低的预取值会使消费者闲置，因为消费者在处理完消息后需要等待下一批的消息到达。过高的值可能会使单个消费者忙碌，其他消费者处于空闲状态。
- 在消费者多且处理时间很长的情况下，建议您将预取值设置为 1，以便消息在所有消费者间均匀分布。

须知

如果客户端配置的消息确认机制为自动确认，则设置的预取值无效，已确认的消息会从队列中删除。

设置预取值

以下示例演示在 Java 客户端为单个消费者设置预取值为 10。

```
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);
```

在 Java 客户端中，**global** 的默认值为 **false**，因此以上示例可以简单地写为 **channel.basicQos(10)**。

global 取值的含义如下：

表10-1 global 取值说明

global 取值	说明
false	分别作用于通道上的每个新消费者。
true	在通道上的所有消费者之间所共享。

10.6 心跳检测

RabbitMQ 实例提供了心跳功能，以确保应用程序层及时发现中断的连接和完全无响应的对端。心跳还可以防止某些网络设备在一段时间内由于没有活动而中断 TCP 连接。

心跳超时时间

心跳超时时间定义了对等 TCP 连接在多长时间后被服务端和客户端视为关闭。

在 RabbitMQ 服务端和客户端分别设置心跳超时时间，服务端和客户端会对配置的心跳超时时间进行协商，客户端必须配置该值来发送心跳。RabbitMQ 官方团队维护的 3 个客户端（Java、.NET、Erlang 语言）的心跳超时时间协商逻辑如下：

- 服务端和客户端设置的心跳超时时间都不为 0 时，两者间较小的值生效。
- 服务端和客户端任意一端设置的心跳超时时间为 0，另一端不为 0 时，非 0 的值生效。
- 服务端和客户端的心跳超时时间都设置为 0 时，表示禁用心跳。

更多关于心跳检测的说明，请参考 Detecting Dead TCP Connections with Heartbeats and TCP Keepalives。

心跳帧

心跳帧发送时间间隔为心跳超时时间/2，该值有时也被称为心跳间隔。客户端在两次错过心跳后，会被认为是不可达的。不同的客户端会以不同的方式显示这一点，但 TCP 连接将被关闭。当客户端检测到服务端由于心跳而无法访问时，需要重新连接。

任何流量（如协议操作、消息发布、消息确认、心跳帧等）都会被认为是有效的心跳。如果连接上有其他流量，客户端可以选择发送心跳帧，也可以选择不发送。如果连接上没有其他流量，客户端必须发送心跳帧。

在客户端设置心跳超时时间

如果心跳超时时间设置过低，在短暂的网络拥塞或短暂的服务器流量控制等情况下可能会产生误报。对于大多数环境，超时时间设置为 5-20 秒最佳。

- 使用 Java 客户端启动心跳。

  在创建连接前使用 **ConnectionFactory#setRequestedHeartbeat** 进行设置，示例如下：

```
ConnectionFactory cf = new ConnectionFactory();
// 将心跳超时时间设置为 15 秒
cf.setRequestedHeartbeat(15);
```

- 使用.NET 客户端启用心跳。

```
var cf = new ConnectionFactory();
// 将心跳超时时间设置为 15 秒
cf.RequestedHeartbeat = TimeSpan.FromSeconds(15);
```

10.7 单一活跃消费者

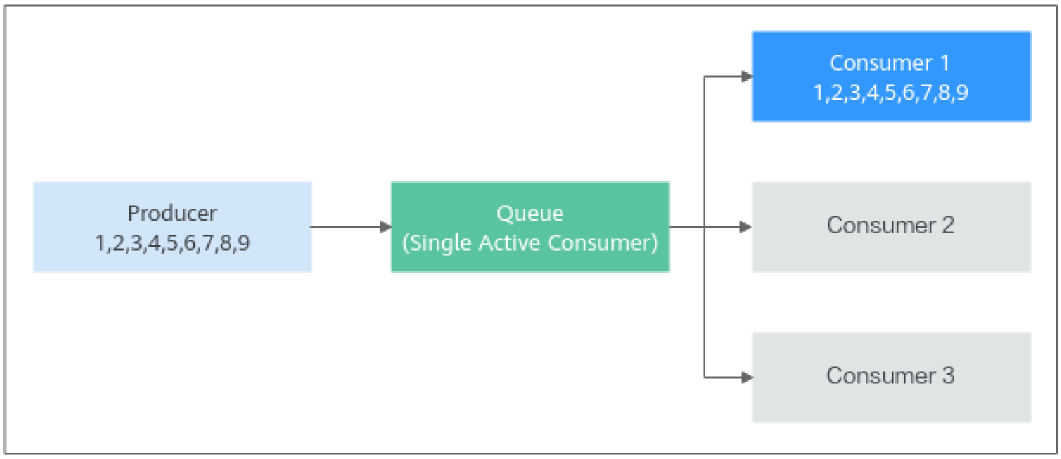
使用场景

单一活跃消费者（Single Active Consumer）表示队列中可以注册多个消费者，但是只允许一个消费者消费消息，只有在此消费者出现异常时，才会自动转移到另一个消费者进行消费。单一活跃消费者适用于需要保证消息消费顺序性，同时提供高可靠能力的场景。

须知

分布式消息服务 RabbitMQ 版 3.8.35 版本才提供单一活跃消费者特性。

图10-6 单一活跃消费者消费流程



如图 10-6 所示，Producer 生产 9 条消息，由于队列设置了单一活跃消费者特性，只有 Consumer 1 在消费消息。

更多关于单一活跃消费者的说明，请参考 Single Active Consumer。

配置方法

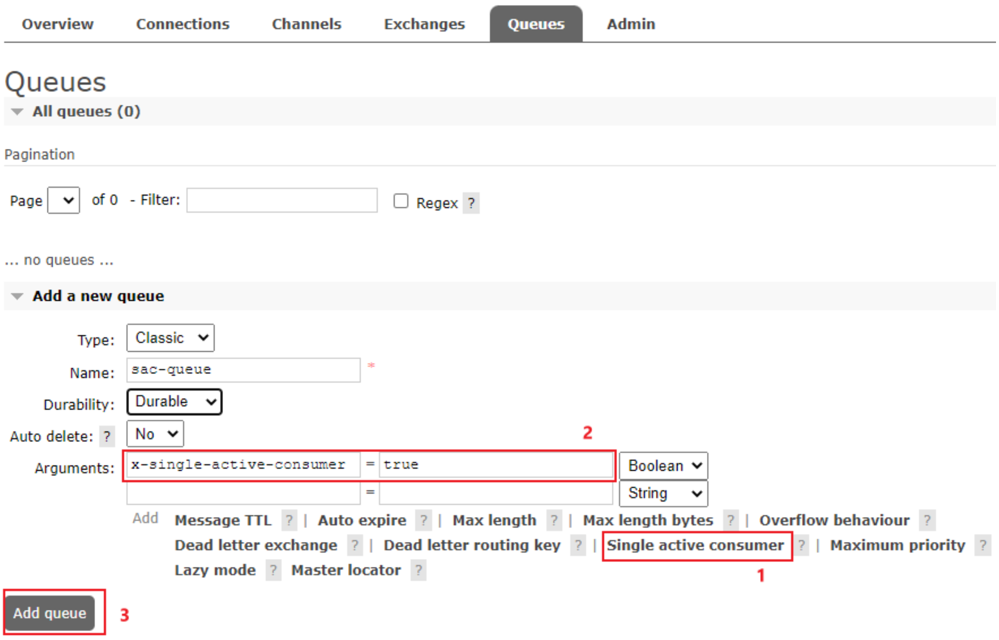
在声明队列时，可以配置单一活跃消费者，只需要将队列的“x-single-active-consumer”参数设置为“true”。

- **以下示例演示在 Java 客户端设置单一活跃消费者。**

```
Channel ch = ...;
Map<String, Object> arguments = newHashMap<String, Object>();
arguments.put("x-single-active-consumer", true);
ch.queueDeclare("my-queue", false, false, false, arguments);
```

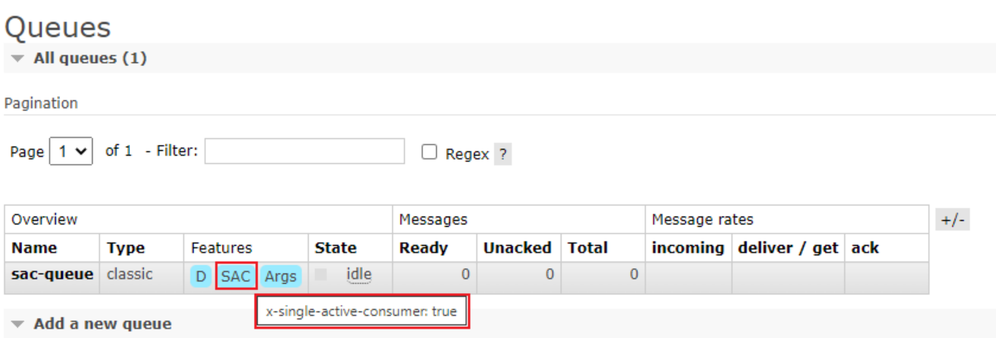
- **以下示例演示在 RabbitMQ WebUI 页面设置单一活跃消费者。**

图10-7 设置单一活跃消费者



设置完成后，在“Queues”页面查看队列特性是否包含单一活跃消费者。如图 10-8 所示，“SAC”即单一活跃消费者。

图10-8 查看队列特性



10.8 仲裁队列

使用场景

仲裁队列（Quorum Queues）提供队列复制的能力，保障数据的高可用和安全性。使用仲裁队列可以在 RabbitMQ 节点间进行队列数据的复制，在一个节点宕机时，队列依旧可以正常运行。

仲裁队列适用于队列长时间存在，对队列容错和数据安全要求高，对延迟和队列特性要求相对低的场景。在可能出现消息大量堆积的场景，不推荐使用仲裁队列，因为仲裁队列的写入放大会造成成倍的磁盘占用。

仲裁队列的消息会优先保存在内存中，使用仲裁队列时，建议定义队列最大长度和最大内存占用，在消息堆积超过阈值时从内存转移到磁盘，以免造成内存高水位。

更多关于仲裁队列的说明，请参考 Quorum Queues。

须知

分布式消息服务 RabbitMQ 版 3.8.35 版本才提供仲裁队列特性。

仲裁队列与镜像队列的差异

仲裁队列是 RabbitMQ 3.8 版本引入的队列类型，它与镜像队列拥有类似的功能，为 RabbitMQ 提供高可用的队列。镜像队列有一些设计上的缺陷，这也是 RabbitMQ 提供仲裁队列的原因。

镜像队列主要的缺陷在于消息同步的性能低。

- 镜像队列包含一个主队列和多个从队列，当生产者向主队列发送一条消息，主队列会将消息同步给从队列，所有的从队列都保存消息后，主队列才会向生产者发送确认。
- RabbitMQ 使用集群部署时，如果其中一个节点故障下线，待它消除故障重新上线后，它保存的所有从队列的数据都会丢失。此时运维人员需要选择是否同步主队列的数据到从队列中，如果不同步数据，会增加消息丢失的风险。如果同步数据，同步时队列是阻塞的，无法对其进行操作。当队列中存在大量堆积消息时，同步会导致队列几分钟、几小时或者更长时间不可用。

仲裁队列解决了镜像队列的性能和同步问题。

- 仲裁队列的算法是基于 Raft 共识算法的一个变种，提供更好的消息吞吐量。仲裁队列包含一个主副本和多个从副本，当生产者向主副本发送一条消息，主副本会将消息同步给从副本，超过半数的副本保存消息后，主副本才会向生产者发送确认。这意味着少部分比较慢的从副本不会影响整个队列的性能。同样地，主副本的选举也需要超过半数的副本同意，这会避免出现网络分区时，队列存在 2 个主副本。由此可见，仲裁队列相对于可用性更看重一致性。
- RabbitMQ 使用集群部署时，如果其中一个节点故障下线，待它消除故障重新上线后，它保存的数据不会丢失，主副本会直接从从副本中断的地方开始复制消息。复制的过程是非阻塞的，整个队列不会因为新的副本加入而受到影响。

仲裁队列相比镜像队列，缺少了一些特性，如表 10-2 所示，且消耗更多的内存和磁盘。

表10-2 特性列表

特性	镜像队列	仲裁队列
非持久化队列	支持	不支持
排他队列	支持	不支持
每条消息的持久化	每条消息	永远
队列重平衡	自动	手动
消息超时时间	支持	不支持
队列超时时间	支持	支持
队列长度限制	支持	支持（除 x-overflow: reject-publish-dlx）
惰性队列	支持	限制队列长度后支持
消息优先级	支持	不支持
消费优先级	支持	支持
死信交换器	支持	支持
动态 Policy	支持	支持
毒药消息（让消费者无限循环消费）处理	不支持	支持
全局消息预取（Qos）	支持	不支持

配置方法

在声明队列时，将队列的“x-queue-type”参数设置为“quorum”。此参数只能在声明队列时设置，不能通过 Policy 设置。

仲裁队列默认的复制因子是 5。

- **以下示例演示在 Java 客户端设置仲裁队列。**

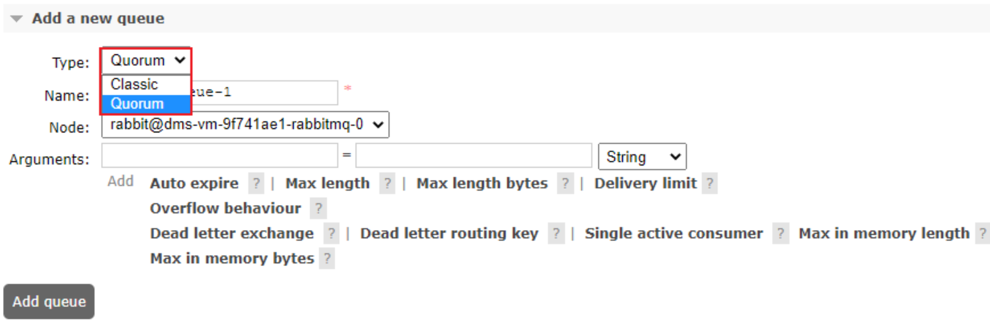
```
ConnectionFactory factory = newConnectionFactory();
factory.setRequestedHeartbeat(30);
factory.setHost(HOST);
factory.setPort(PORT);
factory.setUsername(USERNAME);
factory.setPassword(PASSWORD);

finalConnection connection = factory.newConnection();
finalChannel channel = connection.createChannel();
```

```
// 创建队列参数 Map
Map<String, Object> arguments = newHashMap<>();
arguments.put("x-queue-type", "quorum");
// 声明仲裁队列
channel.queueDeclare("test-quorum-queue", true, false, false, arguments);
```

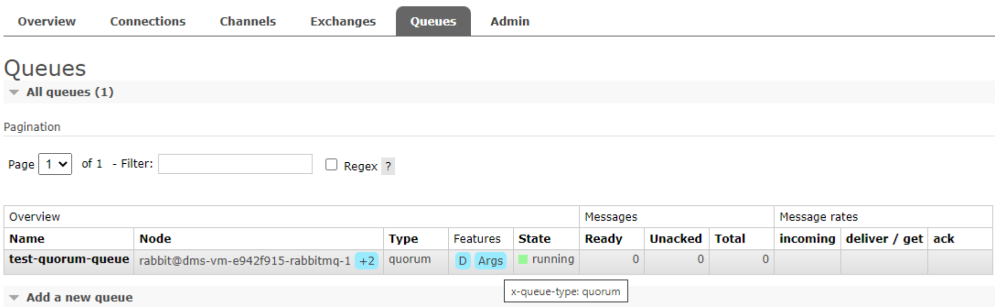
- **以下示例演示在 RabbitMQ WebUI 页面设置仲裁队列。**

图10-9 设置仲裁队列



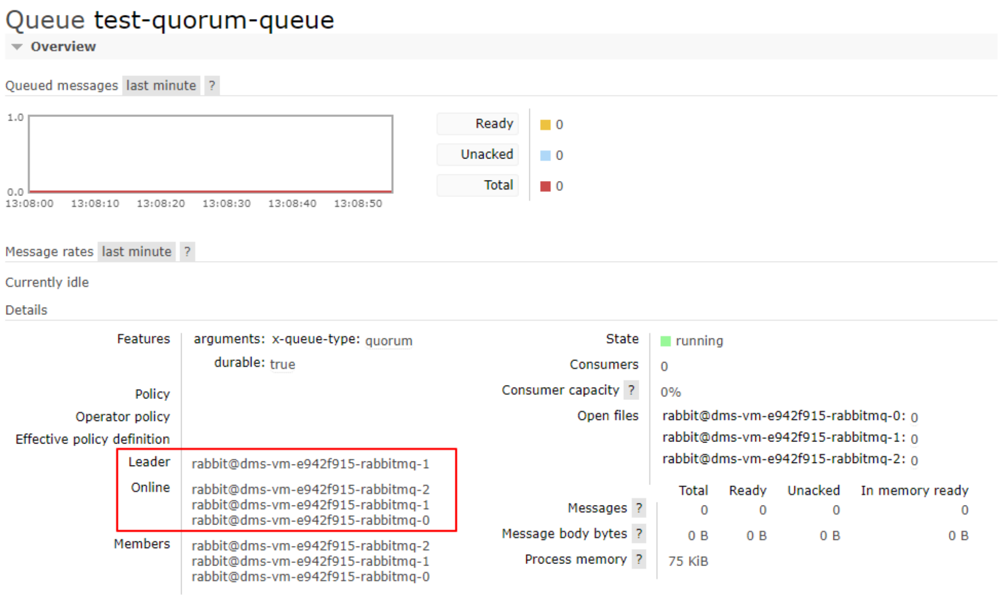
设置完成后，在“Queues”页面查看队列类型是否为“quorum”，如图 10-10 所示。“Node”中的“+2”表示该队列有 2 个副本，蓝色表示这两个副本消息同步已经完成，如果为红色则表示部分消息还未同步。

图10-10 查看队列类型



在“Queues”页面，单击队列名称，进入队列详情页。查看当前仲裁队列主副本所在节点和在线副本所在节点。

图10-11 队列详情页



设置仲裁队列的长度

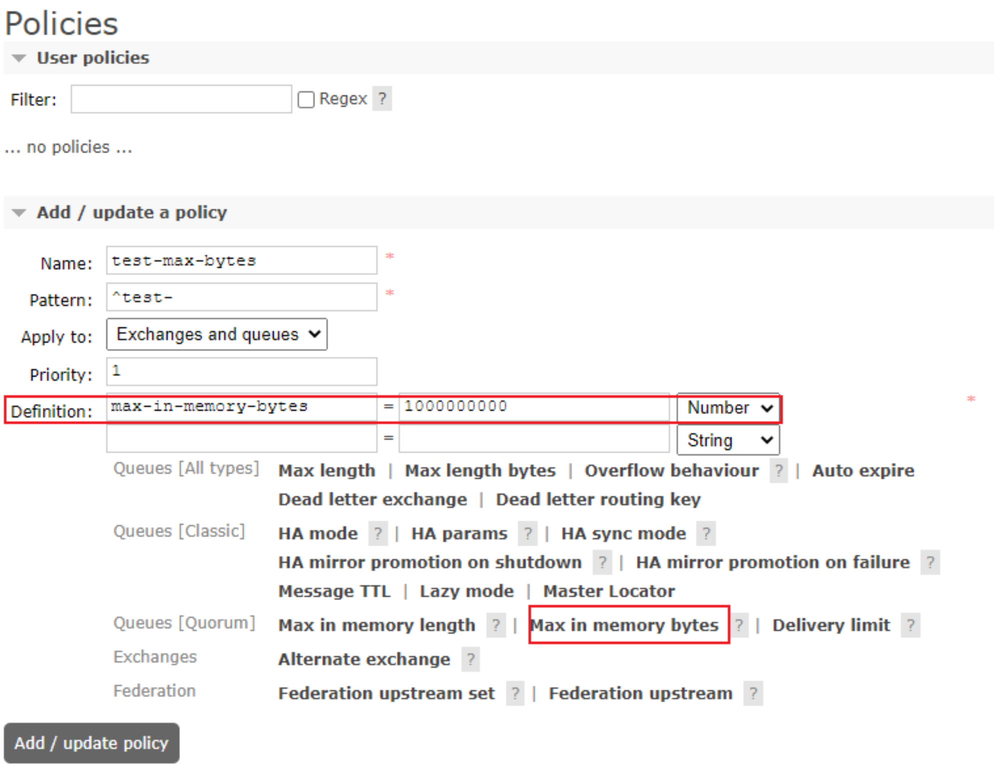
通过配置 Policy 或者队列属性的方式可以限制仲裁队列的长度和在内存中保存的长度。

- x-max-length：仲裁队列最大消息数。如果超过则丢弃消息，或者发送到死信交换器。
- x-max-length-bytes：仲裁队列最大总消息大小（字节数）。如果超过则丢弃消息，或者发送到死信交换器。
- x-max-in-memory-length：限制仲裁队列的内存中最大消息数量。
- x-max-in-memory-bytes：限制仲裁队列的内存中的最大总消息大小（字节数）。

以下举例说明通过配置 Policy 或者队列属性的方式限制内存中保存的仲裁队列长度。

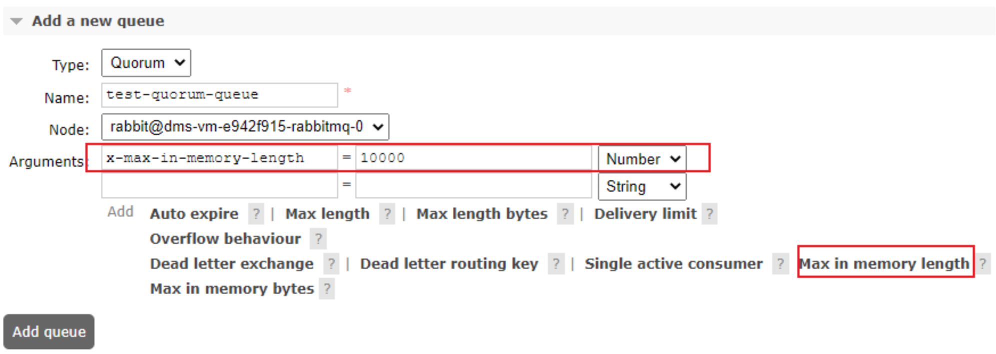
- 配置 Policy 方式，推荐使用此方式。

图10-12 使用 Policy 设置 x-max-in-memory-bytes



- 配置队列属性方式。

图10-13 使用队列属性设置 x-max-in-memory-length



11 调整资源配额

什么是配额？

为防止资源滥用，平台限定了各服务资源的配额，对用户的资源数量和容量做了限制。如您最多可以创建多少个 RabbitMQ 实例。

如果当前资源配额限制无法满足使用需要，您可以申请扩大配额。

怎样查看我的配额？

1. 登录管理控制台。
2. 单击页面右上角的“My Quota”图标 。

   系统进入“服务配额”页面。
3. 您可以在“服务配额”页面，查看各项资源的总配额及使用情况。

   如果当前配额不能满足业务要求，请参考后续操作，申请扩大配额。

如何申请扩大配额？

目前系统暂不支持在线调整配额大小。如您需要调整配额，请拨打热线或发送邮件至客服，客服会及时为您处理配额调整的需求，并以电话或邮件的形式告知您实时进展。

在拨打热线或发送邮件之前，请您准备好以下信息：

- 帐号名，获取方式如下：

  登录云帐户管理控制台，在右上角单击帐户名，选择“我的凭证”，在“我的凭证”页面获取“帐号名”。
- 配额信息，包括：服务名、配额类别、需要的配额值。

12 监控

12.1 支持的监控指标

功能说明

本章节定义了分布式消息服务 RabbitMQ 上报云监控服务的监控指标的命名空间，监控指标列表和维度定义，用户可以通过云监控服务提供的管理控制台来检索 DMS 服务产生的监控指标和告警信息。

命名空间

SYS.DMS

实例监控指标

表12-1 实例支持的监控指标

指标 ID	指标名称	指标含义	取值范围	测量对象	监控周期（原始指标）
connections	连接数	该指标用于统计 RabbitMQ 实例中的总连接数。 单位：Count	>= 0	RabbitMQ 实例	1 分钟
channels	通道数	该指标用于统计 RabbitMQ 实例中的总通道数。 单位：Count	0~2047	RabbitMQ 实例	1 分钟
queues	队列数	该指标用于统计 RabbitMQ 实例中的总队列数。 单位：Count	0~1200	RabbitMQ 实例	1 分钟

指标 ID	指标名称	指标含义	取值范围	测量对象	监控周期（原始指标）
consumers	消费者数	该指标用于统计RabbitMQ 实例中的总消费者数。 单位：Count	0~1200	RabbitMQ 实例	1 分钟
messages_ready	可消费消息数	该指标用于统计RabbitMQ 实例中总可消费消息数量。 单位：Count	0~10000000	RabbitMQ 实例	1 分钟
messages_unacknowledged	未确认消息数	该指标用于统计RabbitMQ 实例中总已经消费但还未确认的消息数量。 单位：Count	0~10000000	RabbitMQ 实例	1 分钟
publish	生产速率	统计 RabbitMQ 实例中实时消息生产速率。 单位：Count/s	0~25000	RabbitMQ 实例	1 分钟
deliver	消费速率（手工确认）	统计 RabbitMQ 实例中实时消息消费速率（手工确认）。 单位：Count/s	0~25000	RabbitMQ 实例	1 分钟
deliver_no_ack	消费速率（自动确认）	统计 RabbitMQ 实例中实时消息消费速率（自动确认）。 单位：Count/s	0~50000	RabbitMQ 实例	1 分钟
connections_states_running	运行状态的连接个数	该指标用于统计整个实例中的connection，状态是starting/tuning/opening/running 状态的总数。 单位：Count	>= 0	RabbitMQ 实例	1 分钟
connections_states_flow	flow 状态的连接数	该指标用于统计整个实例中的connection，状态是flow 状态的总数。 单位：Count	>= 0	RabbitMQ 实例	1 分钟

指标 ID	指标名称	指标含义	取值范围	测量对象	监控周期（原始指标）
connections_states_block	block 状态的连接数	该指标用于统计整个实例中的 connection，状态是 blocking/blocked 状态的总数。 单位：Count	>= 0	RabbitMQ 实例	1 分钟
connections_states_close	close 状态的连接数	该指标用于统计整个实例中的 connection，状态是 closing/closed 状态的总数。 单位：Count	>= 0	RabbitMQ 实例	1 分钟
channels_states_running	运行状态的通道数	该指标用于统计整个实例中的 channel，状态是 starting/tuning/opening/running 状态的总数。 单位：Count	>= 0	RabbitMQ 实例	1 分钟
channels_states_flow	flow 状态的通道数	该指标用于统计整个实例中的 channel，状态是 flow 状态的总数。 单位：Count	>= 0	RabbitMQ 实例	1 分钟
channels_states_block	block 状态的通道数	该指标用于统计整个实例中的 channel，状态是 blocking/blocked 状态的总数。 单位：Count	>= 0	RabbitMQ 实例	1 分钟
channels_states_close	close 状态的通道数	该指标用于统计整个实例中的 channel，状态是 closing/closed 状态的总数。 单位：Count	>= 0	RabbitMQ 实例	1 分钟
queues_states_running	运行状态的队列数	该指标用于统计整个实例中的 queue，状态是 running 状态的总数。 单位：Count	>= 0	RabbitMQ 实例	1 分钟

指标 ID	指标名称	指标含义	取值范围	测量对象	监控周期（原始指标）
queues_states_flow	flow 状态的队列数	该指标用于统计整个实例中的 queue，状态是 flow 状态的总数。 单位：Count	>= 0	RabbitMQ 实例	1 分钟

节点监控指标

表12-2 节点支持的监控指标

指标 ID	指标名称	指标含义	取值范围	测量对象	监控周期（原始指标）
fd_used	文件句柄数	该指标用于统计当前节点 RabbitMQ 所占用的文件句柄数。 单位：Count	0~65535	RabbitMQ 实例节点	1 分钟
socket_used	Socket 连接数	该指标用于统计当前节点 RabbitMQ 所使用的 Socket 连接数。 单位：Count	0~50000	RabbitMQ 实例节点	1 分钟
proc_used	Erlang 进程数	该指标用于统计当前节点 RabbitMQ 所使用的 Erlang 进程数。 单位：Count	0~1048576	RabbitMQ 实例节点	1 分钟
mem_used	内存占用	该指标用于统计当前节点 RabbitMQ 内存占用。 单位：Byte	0~32000000000	RabbitMQ 实例节点	1 分钟
disk_free	可用存储空间	该指标用于统计当前节点可使用的存储空间。 单位：Byte	0~500000000	RabbitMQ 实例节点	1 分钟
rabbitmq_alive	节点存活状态	表示 Rabbitmq 节点是否存活。	1：存活 0：离线	RabbitMQ 实例节点	1 分钟
rabbitmq_disk_usag	磁盘容量使用	统计 Rabbitmq 节点虚拟机的磁盘容量使用	0~100%	RabbitMQ 实例节点	1 分钟

指标 ID	指标名称	指标含义	取值范围	测量对象	监控周期（原始指标）
e	率	率。 单位：%			
rabbitmq_cpu_usage	CPU 使用率	统计 Rabbitmq 节点虚拟机的 CPU 使用率。 单位：%	0~100%	RabbitMQ 实例节点	1 分钟
rabbitmq_cpu_core_load	CPU 核均负载	统计 Rabbitmq 节点虚拟机 CPU 每个核的平均负载。	>0	RabbitMQ 实例节点	1 分钟
rabbitmq_memory_usage	内存使用率	统计 Rabbitmq 节点虚拟机的内存使用率。 单位：%	0~100%	RabbitMQ 实例节点	1 分钟
rabbitmq_disk_read_await	磁盘平均读操作耗时	该指标用于统计磁盘在测量周期内平均每个读 IO 的操作时长。 单位：ms	>0	RabbitMQ 实例节点	1 分钟
rabbitmq_disk_write_await	磁盘平均写操作耗时	该指标用于统计磁盘在测量周期内平均每个写 IO 的操作时长。 单位：ms	>0	RabbitMQ 实例节点	1 分钟
rabbitmq_node_bytes_in_rate	网络入流量	统计 Rabbitmq 节点每秒网络访问流入流量。 单位：Byte/s	>0	RabbitMQ 实例节点	1 分钟
rabbitmq_node_bytes_out_rate	网络出流量	统计 Rabbitmq 节点每秒网络访问流出流量。 单位：Byte/s	>0	RabbitMQ 实例节点	1 分钟
rabbitmq_node_queues	节点队列数	该指标用于统计 Rabbitmq 节点队列个数。 单位：个	>0	RabbitMQ 实例节点	1 分钟
rabbitmq_memory_high_watermark	内存高水位状态	表示 Rabbitmq 节点是否触发内存高水位，如果触发，会阻塞集群的所有生产者。	1：触发 0：没有触发	RabbitMQ 实例节点	1 分钟
rabbitmq_disk_insufficient	磁盘高水位状态	表示 Rabbitmq 节点是否触发磁盘高水位，如果触发，会阻塞集群的	1：触发 0：没有触发	RabbitMQ 实例节点	1 分钟

指标 ID	指标名称	指标含义	取值范围	测量对象	监控周期（原始指标）
		所有生产者。			
rabbitmq_disk_read_rate	磁盘读流量	统计节点磁盘每秒的读字节大小。 单位：KB/s	>= 0	RabbitMQ实例节点	1 分钟
rabbitmq_disk_write_rate	磁盘写流量	统计节点磁盘每秒的写字节大小。 单位：KB/s	>= 0	RabbitMQ实例节点	1 分钟

队列监控指标

表12-3 队列支持的监控指标

指标 ID	指标名称	指标含义	取值范围	测量对象	监控周期（原始指标）
queue_messages_unacknowledged	队列未确认消息数	该指标用于统计队列中已消费未确认消息数。 单位：Count	0~10000000	RabbitMQ实例队列	1 分钟
queue_messages_ready	队列可消费消息数	该指标用于统计队列中可消费的消息数。 单位：Count	0~10000000	RabbitMQ实例队列	1 分钟
queue_consumers	消费者数量	该指标用于统计订阅该队列的消费者个数。 单位：Count	>= 0	RabbitMQ实例队列	1 分钟
queue_messages_publish_rate	生产速率	该指标用于统计每秒该队列的消息流入数。 单位：Count/s	>= 0	RabbitMQ实例队列	1 分钟
queue_messages_ack_rate	消费速率（手工确认）	该指标用于统计该队列每秒传递给客户端并确认的消息数。 单位：Count/s	>= 0	RabbitMQ实例队列	1 分钟
queue_messages_deliver_get_rate	消费速率	该指标用于统计该队列每秒的消息流出数。 单位：Count/s	>= 0	RabbitMQ实例队列	1 分钟

指标 ID	指标名称	指标含义	取值范围	测量对象	监控周期（原始指标）
queue_messages_redeliver_rate	重传速率	该指标用于统计该队列每秒的重传消息数。 单位：Count/s	>= 0	RabbitMQ 实例队列	1 分钟
queue_messages_persistent	消息总数（持久化）	该指标用来统计该队列中持久消息的总数（对于瞬时队列始终为 0）。 单位：Count	>= 0	RabbitMQ 实例队列	1 分钟
queue_messages_ram	消息总数（内存）	该指标用于统计该队列中驻留在内存中的消息总数。 单位：Count	>= 0	RabbitMQ 实例队列	1 分钟
queue_memory	Erlang 进程消耗字节数	该指标用于统计与队列关联的 Erlang 进程消耗的内存字节数，包括堆栈、堆和内部结构。 单位：Byte	>= 0	RabbitMQ 实例队列	1 分钟
queue_message_bytes	消息大小总和	该指标用于统计该队列中所有消息的大小总和（字节）。 单位：Byte	>= 0	RabbitMQ 实例队列	1 分钟

维度

Key	Value
rabbitmq_instance_id	RabbitMQ 实例
rabbitmq_node	RabbitMQ 实例节点
rabbitmq_queue	RabbitMQ 实例队列

12.2 设置 RabbitMQ 告警规则

本章节主要介绍部分监控指标的告警策略，以及配置操作。在实际业务中，建议按照以下告警策略，配置监控指标的告警规则。

表12-4 RabbitMQ 实例配置告警的指标

<table>
<tr><th>指标名称</th><th>告警策略</th><th>指标说明</th><th>解决方案</th></tr>
<tr><td>内存高水位状态</td><td>告警阈值：原始值>=1
连续触发次数：1
告警级别：致命</td><td>告警阈值为 1 表示触发内存高水位，会阻塞消息生产</td><td>• 加快消费
• 采用生产者确认的发送模式，并监控生产端消息生产速度和时长，当消息生产时长有明显增加时进行流控措施</td></tr>
<tr><td>磁盘高水位状态</td><td>告警阈值：原始值>=1
连续触发次数：1
告警级别：致命</td><td>告警阈值为 1 表示触发磁盘高水位，会阻塞消息生产</td><td>• 减少惰性队列的消息堆积
• 减少持久化队列的消息堆积
• 删除队列</td></tr>
<tr><td>内存使用率</td><td>告警阈值：原始值>业务预期使用率（推荐 30%）
连续触发次数：连续 3~5 个周期
告警级别：重要</td><td>该指标需要分别为每个节点设置内存使用率告警，避免触发内存高水位阻塞生产</td><td>• 加快消费
• 采用生产者确认的发送模式，并监控生产端消息生产速度和时长，当消息生产时长有明显增加时进行流控措施</td></tr>
<tr><td>CPU 使用率</td><td>告警阈值：原始值>业务预期使用率（推荐 70%）
连续触发次数：连续 3~5 个周期
告警级别：重要</td><td>该指标需要分别为每个节点设置 CPU 使用率告警，CPU 使用率过高可能会影响生产速度</td><td>• 减少镜像队列个数
• 对于集群实例，建议扩容节点个数，然后进行节点间重平衡</td></tr>
<tr><td>可消费消息数</td><td>告警阈值：原始值>业务预期可消费消息数
连续触发次数：1
告警级别：重要</td><td>可消费消息数过多表示消息堆积</td><td>请参考消息堆积的解决办法</td></tr>
<tr><td>未确认消息数</td><td>告警阈值：原始值>业务预期未确认消息数
连续触发次数：1
告警级别：重要</td><td>未确认消息数过多可能会导致消息堆积</td><td>• 检查消费者是否异常
• 检查消费者逻辑是否消耗时间过长</td></tr>
<tr><td>连接数</td><td>告警阈值：原始值>业务预期连接数</td><td>连接数突增可能是流量变大的预警</td><td>需检查业务是否正常，可参考其他告警</td></tr>
</table>

指标名称	告警策略	指标说明	解决方案
	连续触发次数：1 告警级别：重要		
通道数	告警阈值：原始值>业务预期通道数 连续触发次数：1 告警级别：重要	通道数突增可能是流量变大的预警	需检查业务是否正常，可参考其他告警
Erlang 进程数	告警阈值：原始值>业务预期进程数 连续触发次数：1 告警级别：重要	进程数突增可能是流量变大的预警	需检查业务是否正常，可参考其他告警

说明

- 告警阈值请根据业务预期数设置。例如，业务预期使用率 35%，则告警阈值设置 35%。
- 连续触发次数和告警级别可根据业务逻辑自行调整。

操作步骤

步骤 1 登录管理控制台。

步骤 2 在管理控制台左上角单击 ，选择区域。

说明

此处请选择 RabbitMQ 实例所在的区域。

步骤 3 在管理控制台左上角单击 ，选择“应用服务 > 分布式消息服务 RabbitMQ”，进入分布式消息服务 RabbitMQ 专享版页面。

步骤 4 通过以下任意一种方法，查看监控数据。

- 在 RabbitMQ 实例名称后，单击 。跳转到云监控页面，查看实例、节点和队列的监控数据，数据更新周期为 1 分钟。
- 单击 RabbitMQ 实例名称，进入实例详情页。在左侧导航栏单击“监控”，进入监控页面，查看实例、节点和队列的监控数据，数据更新周期为 1 分钟。

步骤 5 在实例监控指标页面中，找到需要创建告警的指标项，鼠标移动到指标区域，然后单击指标右上角的 ，进入“创建告警规则”页面。

步骤 6 在告警规则页面，设置告警信息。

创建告警规则操作，请查看《天翼云云监控服务用户指南》。

1. 设置告警名称和告警的描述。
2. 设置告警策略和告警级别。

   例如，在进行指标监控时，如果连续 3 个周期，连接数原始值超过设置的值，则产生告警，如果未及时处理，则每一天发送一次告警通知。
3. 设置“发送通知”开关。当开启时，设置告警生效时间、产生告警时通知的对象以及触发的条件。
4. 单击“立即创建”，等待创建告警规则成功。

----结束

12.3 查看监控数据

操作场景

云监控对分布式消息服务 RabbitMQ 的运行状态进行日常监控，可以通过控制台直观的查看分布式消息服务 RabbitMQ 版各项监控指标。

前提条件

已创建 RabbitMQ 实例，且实例中有可消费的消息。

操作步骤

步骤 1 登录管理控制台。

步骤 2 在管理控制台左上角单击，选择区域。

说明

此处请选择 RabbitMQ 实例所在的区域。

步骤 3 在管理控制台左上角单击，选择“应用服务 > 分布式消息服务 RabbitMQ”，进入分布式消息服务 RabbitMQ 专享版页面。

步骤 4 通过以下任意一种方法，查看监控数据。

- 在 RabbitMQ 实例名称后，单击。跳转到云监控页面，查看实例、节点和队列的监控数据，数据更新周期为 1 分钟。
- 单击 RabbitMQ 实例名称，进入实例详情页。在左侧导航栏单击“监控”，进入监控页面，查看实例、节点和队列的监控数据，数据更新周期为 1 分钟。

----结束

13 云审计服务支持的关键操作

13.1 云审计服务支持的 DMS for RabbitMQ 操作列表

通过云审计服务，您可以记录与分布式消息服务 RabbitMQ 相关的操作事件，便于日后的查询、审计和回溯。

表13-1 云审计服务支持的 DMS for RabbitMQ 操作列表

操作名称	资源类型	事件名称
删除后台任务成功	rabbitmq	deleteDMSBackendJobSuccess
删除后台任务失败	rabbitmq	deleteDMSBackendJobFailure
创建 DMS 实例订单成功	rabbitmq	createDMSInstanceOrderSuccess
创建 DMS 实例订单失败	rabbitmq	createDMSInstanceOrderFailure
修改 DMS 实例订单成功	rabbitmq	modifyDMSInstanceOrderSuccess
修改 DMS 实例订单失败	rabbitmq	modifyDMSInstanceOrderFailure
扩容实例成功	rabbitmq	extendDMSInstanceSuccess
扩容实例失败	rabbitmq	extendDMSInstanceFailure
重置 DMS 实例密码成功	rabbitmq	resetDMSInstancePasswordSuccess
重置 DMS 实例密码失败	rabbitmq	resetDMSInstancePasswordFailure
删除创建失败的 DMS 实例成功	rabbitmq	deleteDMSCreateFailureInstancesSuccess
删除创建失败的 DMS	rabbitmq	deleteDMSCreateFailureInstancesFa

操作名称	资源类型	事件名称
实例失败		ilure
重启 DMS 实例成功	rabbitmq	restartDMSInstanceSuccess
重启 DMS 实例失败	rabbitmq	restartDMSInstanceFailure
批量删除 DMS 实例成功	rabbitmq	batchDeleteDMSInstanceSuccess
批量删除 DMS 实例失败	rabbitmq	batchDeleteDMSInstanceFailure
批量重启 DMS 实例成功	rabbitmq	batchRestartDMSInstanceSuccess
批量重启 DMS 实例失败	rabbitmq	batchRestartDMSInstanceFailure
修改 DMS 实例信息成功	rabbitmq	modifyDMSInstanceInfoSuccess
修改 DMS 实例信息失败	rabbitmq	modifyDMSInstanceInfoFailure
批量删除 DMS 实例任务	rabbitmq	batchDeleteDMSInstanceTask
解冻 DMS 实例任务成功	rabbitmq	unfreezeDMSInstanceTaskSuccess
解冻 DMS 实例任务失败	rabbitmq	unfreezeDMSInstanceTaskFailure
冻结 DMS 实例任务成功	rabbitmq	freezeDMSInstanceTaskSuccess
冻结 DMS 实例任务失败	rabbitmq	freezeDMSInstanceTaskFailure
删除 DMS 实例任务成功	rabbitmq	deleteDMSInstanceTaskSuccess
删除 DMS 实例任务失败	rabbitmq	deleteDMSInstanceTaskFailure
创建 DMS 实例任务成功	rabbitmq	createDMSInstanceTaskSuccess
创建 DMS 实例任务失败	rabbitmq	createDMSInstanceTaskFailure
扩容 DMS 实例任务成功	rabbitmq	extendDMSInstanceTaskSuccess

操作名称	资源类型	事件名称
扩容 DMS 实例任务失败	rabbitmq	extendDMSInstanceTaskFailure
重启 DMS 实例任务成功	rabbitmq	restartDMSInstanceTaskSuccess
重启 DMS 实例任务失败	rabbitmq	restartDMSInstanceTaskFailure
批量重启 DMS 实例任务成功	rabbitmq	batchRestartDMSInstanceTaskSuccess
批量重启 DMS 实例任务失败	rabbitmq	batchRestartDMSInstanceTaskFailure
修改 DMS 实例信息任务成功	rabbitmq	modifyDMSInstanceInfoTaskSuccess
修改 DMS 实例信息任务失败	rabbitmq	modifyDMSInstanceInfoTaskFailure

13.2 查看云审计日志

查看 DMS for RabbitMQ 云审计日志，请参考《云审计服务用户指南》的“查看追踪事件”章节。

14 常见问题

14.1 实例问题

14.1.1 RabbitMQ 使用的版本是多少？

服务端 RabbitMQ 的版本是 3.8.35。

14.1.2 RabbitMQ 实例 SSL 连接的协议版本号是多少？

TLS v1.2 版本。

14.1.3 创建实例时为什么无法查看子网和安全组等信息？

创建实例时，如果无法查看虚拟私有云、子网、安全组、弹性 IP，可能原因是该用户无 Server Administrator 和 VPC Administrator 权限，增加权限的详细步骤请参考《天翼云统一身份认证服务用户指南》的“用户指南 > 用户组及授权 > 查看或修改用户组”章节。

14.1.4 重启 RabbitMQ 实例时，若其中一台 RabbitMQ 重启失败，会如何处理？

重启 RabbitMQ 实例时，不会重启实例所在虚拟机，仅重启 RabbitMQ 进程。

重启集群实例时，若其中一台 RabbitMQ 进程重启失败，则重启后实例状态依然为“运行中”，并提示“部分节点故障”。在每台虚拟机上都有 RabbitMQ 的守护进程，定时检查 RabbitMQ 进程是否存在，当进程不存在时会自动拉起 RabbitMQ 进程。

如果 RabbitMQ 实例异常持续超过 1 分钟，会上报告警。

14.1.5 RabbitMQ 集群实例如何均衡分发请求到每个虚拟机？

集群内部使用 LVS 做负载均衡，由 LVS 将请求均衡分发到每个虚拟机节点。

14.1.6 RabbitMQ 实例集群内部的队列是否有冗余备份？

队列是否做镜像（即冗余备份）取决于用户的需要，如果用户设置了镜像，会在集群中多个代理上存储队列的副本，当某个代理故障，集群会从其他正常的代理中选择一个代理，用来同步队列数据。

14.1.7 RabbitMQ 实例是否支持持久化，如何定时备份数据？

RabbitMQ 支持消息数据持久化，可从客户端连接 RabbitMQ 并设置消息持久化，也可在 RabbitMQ 集群管理工具界面创建队列时设置消息持久化。

不支持客户自定义定时备份数据，或从界面触发备份数据。

14.1.8 RabbitMQ 实例开启 SSL 后，证书怎么获取？

RabbitMQ 实例开启 SSL 后只做单向认证，不需要证书。

14.1.9 RabbitMQ 实例的 SSL 开关是否支持修改？

不支持动态修改，即如果实例创建时没有选择开启，创建完成之后，不支持修改，建议在实例创建时将开关打开。

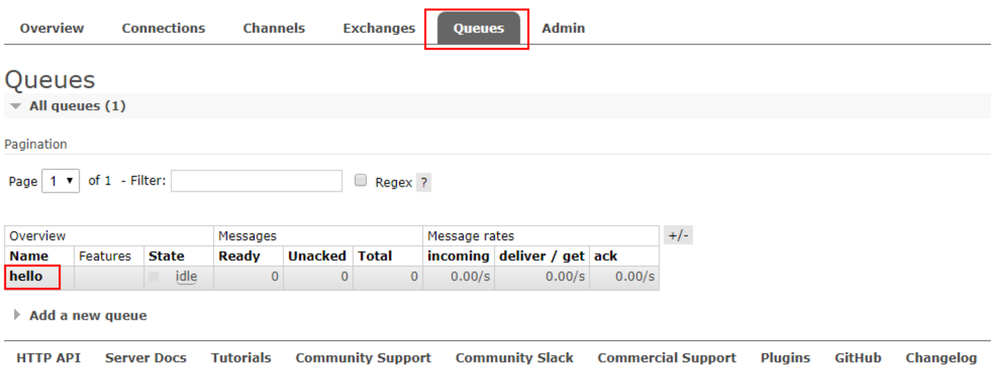
14.1.10 RabbitMQ 实例是否支持扩容？

单机版本的 RabbitMQ 实例支持扩大存储空间，以及扩容/缩容代理规格。

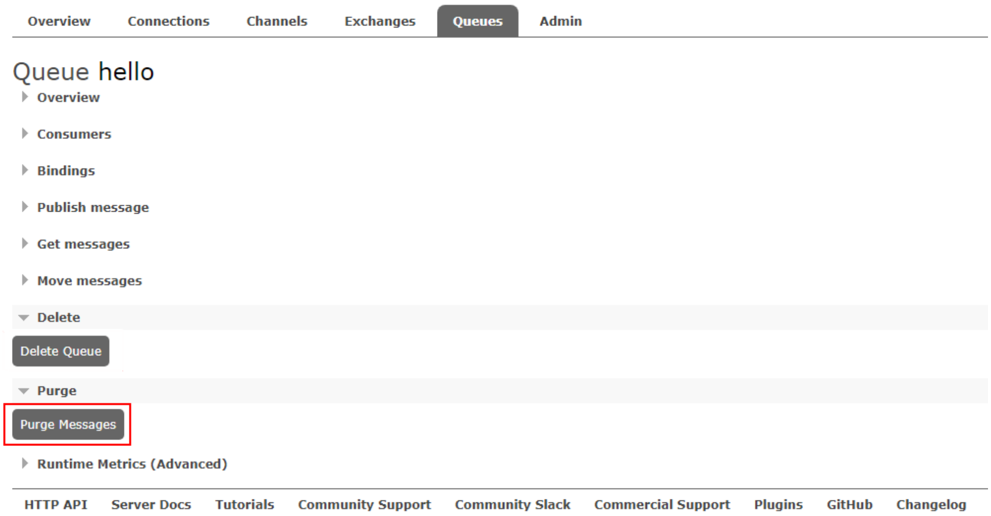
集群版本的 RabbitMQ 实例支持扩大存储空间和代理个数，以及扩容/缩容代理规格。

14.1.11 如何清空队列数据？

1. 登录 Web UI。
2. 在“Queues”页签，单击需要清空数据的队列名称，进入队列详情页面。



3. 单击“Purge Messages”，清空队列数据。



14.1.12 RabbitMQ 支持双向认证吗？

不支持。

14.1.13 RabbitMQ 支持升级 CPU 和内存吗？

RabbitMQ 支持扩容/缩容代理规格，具体请参见变更实例规格。

14.1.14 如何关闭 RabbitMQ 的 WebUI？

创建 RabbitMQ 实例后，如果想要关闭 RabbitMQ 的 WebUI，只要您在安全组入方向中不开放 15672 端口（实例未开启 SSL 时的端口）或者 15671（实例开启 SSL 时的端口），此时就无法登录 WebUI 界面。

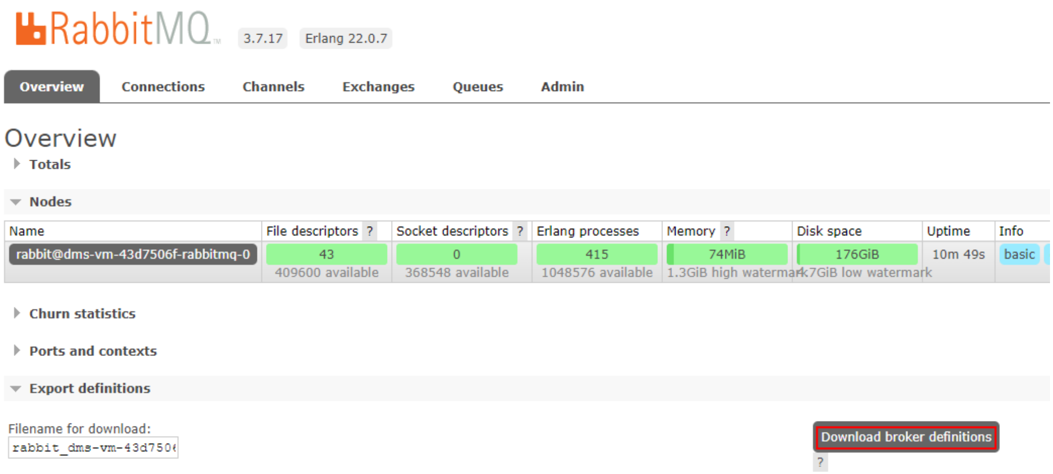
14.1.15 实例是否支持修改可用区？

不支持，您可以重新创建实例，以满足可用区要求，然后进行实例元数据的迁移。

实例元数据的迁移步骤如下：

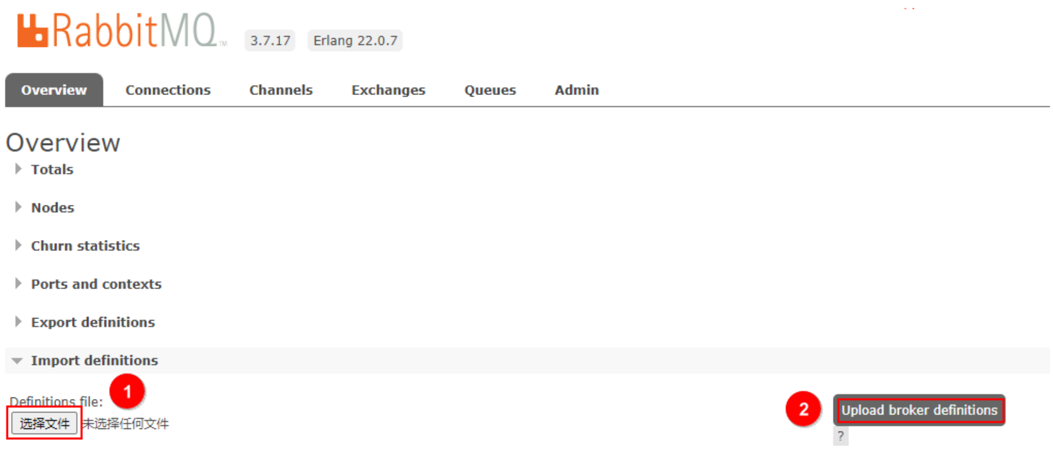
步骤 1 登录重新创建前的 RabbitMQ 实例的 WebUI 页面。

步骤 2 在“Overview”页签中，单击“Download broker definitions”，导出元数据。

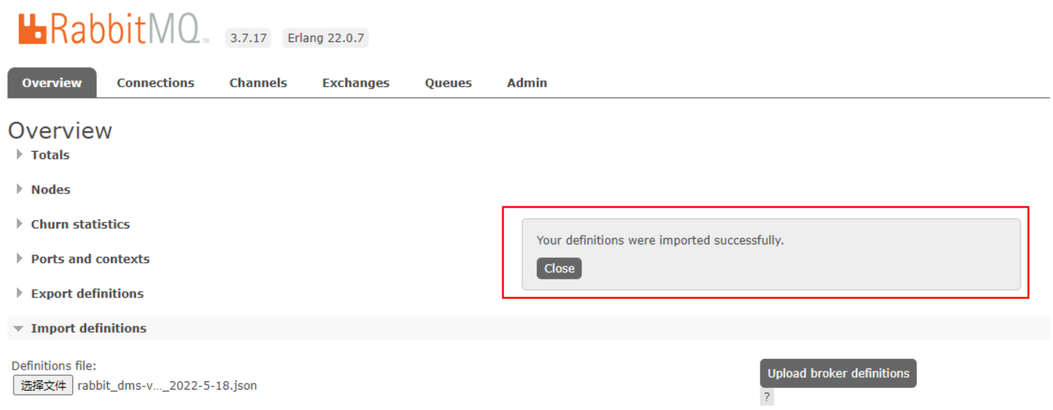


步骤 3 登录重新创建的 RabbitMQ 实例的 WebUI 页面，在“Overview”页签中，单击“选择文件”，选择步骤 2 中导出的元数据。

步骤 4 单击“Upload broker definitions”，上传元数据。



上传成功后，显示如下信息。



----结束

14.1.16 如何获取 region id？

获取 region id 的方法如下：从管理员处获取。

14.2 连接问题

14.2.1 如何配置安全组？

RabbitMQ 实例支持 VPC 内访问和公网访问，配置安全组的方式如下：

- VPC 内访问实例

  客户端只能部署在与 RabbitMQ 实例处于相同虚拟私有云（VPC）的弹性云主机（ECS）上。

  除了 ECS、RabbitMQ 实例必须处于相同 VPC 之外，还需要他们的安全组分别配置了正确的规则，客户端才能访问 RabbitMQ 实例。

  a. 建议 ECS、RabbitMQ 实例配置相同的安全组。安全组创建后，默认包含组内网络访问不受限制的规则。

  b. 如果配置了不同安全组，可参考如下配置方式：

说明

- 假设 ECS、RabbitMQ 实例分别配置了安全组：sg-53d4、sg-RabbitMQ、Default_All。
- 以下规则，远端可使用安全组，也可以使用具体的 IP 地址。

ECS 所在安全组需要增加如下规则，以保证客户端能正常访问 RabbitMQ 实例。

表14-1 安全组规则

方向	协议端口	目的地址
出方向	全部放通	Default_All

RabbitMQ 实例所在安全组需要增加如下规则，以保证能被客户端访问。

表14-2 安全组规则

方向	协议端口	源地址
入方向	全部放通	sg-53d4

- 通过公网访问实例

  RabbitMQ 实例所在安全组需要增加如下规则，以保证能被客户端访问。

表14-3 安全组规则

方向	协议端口	源地址
入方向	TCP:5672	0.0.0.0/0

14.2.2 RabbitMQ 客户端连接报错原因分析

RabbitMQ 客户端连接失败，可能原因包括地址、端口填错、用户名或者密码填错、超过最大连接数。

- **连接地址不正确**

  VPC 内访问场景下，连接地址不正确时，报错如下：

```
[root@ecs-heru RabbitMQ-Tutorial]# java -cp .:rabbitmq-tutorial.jar Send
192.168.125.110 5672 user *******
Exception in thread "main" java.net.NoRouteToHostException: No route to host
(Host unreachable)
at java.net.PlainSocketImpl.socketConnect(Native Method)
at java.net.AbstractPlainSocketImpl.doConnect(AbstractPlainSocketImpl.java:350)
at
java.net.AbstractPlainSocketImpl.connectToAddress(AbstractPlainSocketImpl.java:
206)
```

  公网访问场景下，连接地址不正确时，报错如下：

```
[root@ecs-heru RabbitMQ-Tutorial]# java -cp .:rabbitmq-tutorial.jar Send
139.xxx.178 5672 user *******
Exception in thread "main" java.net.SocketTimeoutException: connect timed out
at java.net.PlainSocketImpl.socketConnect(Native Method)
at java.net.AbstractPlainSocketImpl.doConnect(AbstractPlainSocketImpl.java:350)
```

- **端口不正确**

  VPC 内访问场景下，端口不正确时，报错如下：

```
[root@ecs-heru RabbitMQ-Tutorial]# java -cp .:rabbitmq-tutorial.jar Send
192.168.125.111 5673 user *******
Exception in thread "main" java.net.ConnectException: Connection refused
(Connection refused)
at java.net.PlainSocketImpl.socketConnect(Native Method)
at java.net.AbstractPlainSocketImpl.doConnect(AbstractPlainSocketImpl.java:350)
at
java.net.AbstractPlainSocketImpl.connectToAddress(AbstractPlainSocketImpl.java:
206)
```

  公网访问场景下，端口不正确时，报错如下：

```
[root@ecs-heru RabbitMQ-Tutorial]# java -cp .:rabbitmq-tutorial.jar Send
139.xxx.179 5673 user *******
Exception in thread "main" java.net.SocketTimeoutException: connect timed out
at java.net.PlainSocketImpl.socketConnect(Native Method)
at java.net.AbstractPlainSocketImpl.doConnect(AbstractPlainSocketImpl.java:350)
at
java.net.AbstractPlainSocketImpl.connectToAddress(AbstractPlainSocketImpl.java:
206)
```

- **用户名或密码错误**

```
[root@ecs-heru RabbitMQ-Tutorial]# java -cp .:rabbitmq-tutorial.jar Send
192.168.125.111 5672 user *******
Exception in thread "main" com.rabbitmq.client.AuthenticationFailureException:
ACCESS_REFUSED - Login was refused using authentication mechanism PLAIN. For
details
 see the broker logfile.
at com.rabbitmq.client.impl.AMQConnection.start(AMQConnection.java:351)
at
com.rabbitmq.client.impl.recovery.RecoveryAwareAMQConnectionFactory.newConnecti
on(RecoveryAwareAMQConnectionFactory.java:64)
```

- **超过最大连接数**

14.2.3 RabbitMQ 实例是否支持公网访问？

支持。在创建 RabbitMQ 实例时开启“公网访问”，或创建完后，在实例详情页中将公网访问开关打开。

14.2.4 RabbitMQ 是否支持跨 Region 部署？

当前支持跨 AZ（可用区），不支持跨 Region 部署。

14.2.5 RabbitMQ 实例是否支持跨 VPC 和跨子网访问？

RabbitMQ 实例支持跨 VPC 和子网访问，可以通过创建 VPC 对等连接，将两个 VPC 的网络打通，实现跨 VPC 访问实例。

关于创建和使用 VPC 对等连接，请参考《虚拟私有云-用户手册》的“VPC 对等连接”章节。

14.2.6 RabbitMQ 实例是否支持不同的子网？

支持。

客户端与实例在相同 VPC 内，可以跨子网段访问。同一个 VPC 内的子网默认可以进行通信。

客户端与实例在不同 VPC 时，需建立 VPC 对等连接。

此外，可以为实例绑定公网地址，客户端访问实例公网地址即可。

14.2.7 SSL 方式连接 RabbitMQ 实例失败？

首先排查安全组的入方向规则，是否放开了端口 5671（SSL 方式访问）或 5672（非 SSL 访问）。

其次，参考如下内容配置 SSL 单向认证：

```
ConnectionFactory factory = new ConnectionFactory();
factory.setHost(host);
factory.setPort(port);
factory.setUsername(user);
factory.setPassword(password);
factory.useSslProtocol();
```

```
Connection connection = factory.newConnection();
Channel channel = connection.createChannel();
```

14.2.8 客户端是否可以通过 DNAT 方式访问 RabbitMQ 实例？

不可以。客户端可以使用代理、VPN、专线、FullNAT 或者反向代理等方式访问 RabbitMQ 实例。

14.2.9 RabbitMQ 实例的 Web 管理页面无法打开

可能原因：实例安全组配置不正确

解决方案：重新配置安全组，具体步骤如下。

1. 在实例详情页面的“基本信息 > 网络”，单击安全组名称，跳转到安全组页面。
2. 选择“入方向规则”，查看安全组入方向规则。
   - 实例未开启 SSL 开关
     - 如果是 VPC 内访问，实例安全组入方向规则，需要允许端口 5672 的访问。
     - 如果是公网访问，需要允许端口 15672 的访问。
   - 实例已开启 SSL 开关
     - 如果是 VPC 内访问，实例安全组入方向规则，需要允许端口 5671 的访问。
     - 如果是公网访问，需要运行端口 15671 的访问。

14.2.10 客户端是否可以连接同个 RabbitMQ 下多个 Vhost？

客户端可以连接同个 RabbitMQ 下多个 Vhost。

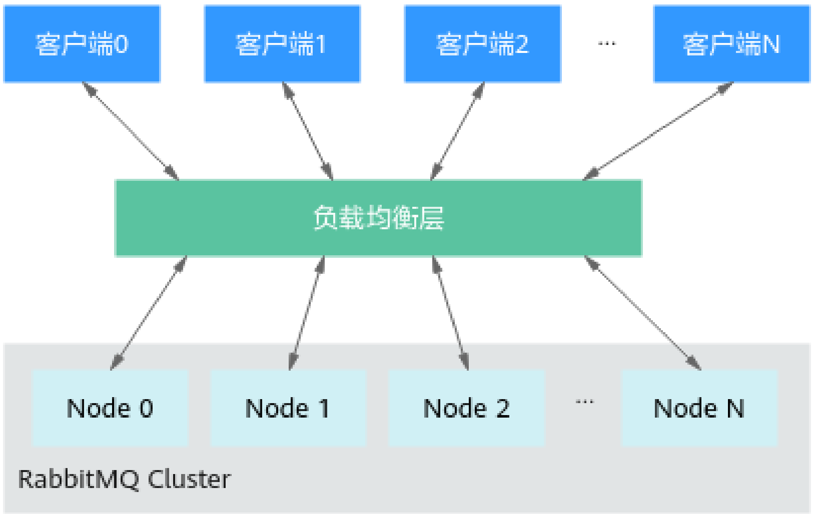
Vhost（Virtual Hosts）是 RabbitMQ 的基本特性，每个 Vhost 相当于一个相对独立的 RabbitMQ 服务器，每个 Vhost 数据目录不同，共用一个进程。性能上，连接多个 Vhost 和单独使用一个 Vhost 差别不大，只是 RabbitMQ 进程多一些对象，建议使用业务模型实测。

Vhost 的相关介绍，请参考官网文档 Virtual Hosts。

14.2.11 为什么 RabbitMQ 集群只有一个连接地址？

RabbitMQ 集群实例的连接地址，实际上是实例的 LVS 节点地址（负载均衡地址），客户端连接实例时，通过负载均衡器将客户端请求分发到集群实例的各个节点。

图14-1 连接示意图



14.3 插件问题

14.3.1 支持的 RabbitMQ 插件有哪些？

支持的 RabbitMQ 插件有哪些？

RabbitMQ 实例购买后，支持的插件如下，其中，方括号中为空的表示还未安装，标记为[E*]的插件是明确安装的，标记为[e*]的插件是隐式安装的，也就是说，这些插件是作为其它的插件的依赖而进行安装的。

```
[  ] rabbitmq_amqp1_0                3.8.35
[  ] rabbitmq_auth_backend_cache       3.8.35
[  ] rabbitmq_auth_backend_http       3.8.35
[  ] rabbitmq_auth_backend_ldap       3.8.35
[  ] rabbitmq_auth_mechanism_ssl       3.8.35
[  ] rabbitmq_consistent_hash_exchange 3.8.35
[  ] rabbitmq_delayed_message_exchange 3.8.9
[  ] rabbitmq_event_exchange          3.8.35
[  ] rabbitmq_federation              3.8.35
[  ] rabbitmq_federation_management    3.8.35
[  ] rabbitmq_jms_topic_exchange       3.8.35
[E*] rabbitmq_management              3.8.35
[e*] rabbitmq_management_agent         3.8.35
[  ] rabbitmq_mqtt                   3.8.35
[  ] rabbitmq_peer_discovery_aws       3.8.35
[  ] rabbitmq_peer_discovery_common    3.8.35
[  ] rabbitmq_peer_discovery_consul    3.8.35
[  ] rabbitmq_peer_discovery_etcd      3.8.35
[  ] rabbitmq_peer_discovery_k8s       3.8.35
[  ] rabbitmq_random_exchange         3.8.35
[  ] rabbitmq_recent_history_exchange  3.8.35
```

```
[  ] rabbitmq_sharding              3.8.35
[  ] rabbitmq_shovel                3.8.35
[  ] rabbitmq_shovel_management     3.8.35
[  ] rabbitmq_stomp                 3.8.35
[E*] rabbitmq_top                   3.8.35
[  ] rabbitmq_tracing               3.8.35
[  ] rabbitmq_trust_store           3.8.35
[e*] rabbitmq_web_dispatch          3.8.35
[  ] rabbitmq_web_mqtt              3.8.35
[  ] rabbitmq_web_mqtt_examples     3.8.35
[  ] rabbitmq_web_stomp             3.8.35
[  ] rabbitmq_web_stomp_examples    3.8.35
```

RabbitMQ 控制台支持安装的插件有：rabbitmq_amqp1_0、rabbitmq_delayed_message_exchange、rabbitmq_federation、rabbitmq_sharding、rabbitmq_shovel、rabbitmq_tracing、rabbitmq_mqtt、rabbitmq_web_mqtt、rabbitmq_stomp、rabbitmq_web_stomp 和 rabbitmq_consistent_hash_exchange，实例创建后，以上插件默认关闭，如需开启，在控制台实例详情的“插件管理”页面开启，具体操作请参见开启实例插件。

RabbitMQ 插件功能可用于测试和迁移业务等场景，不建议用于生产业务。详情请参考约束与限制。

如果需要激活其他未安装的插件，需要联系客服在后台开启插件，开启过程中，对业务没有任何影响。

14.4 消息问题

14.4.1 RabbitMQ 实例支持延时消息队列么？

RabbitMQ 可以通过设置消息的有效期和死信队列来实现延迟消息。同时，也提供安装插件实现延迟消息。当前 RabbitMQ 支持的插件：rabbitmq_amqp1_0、rabbitmq_delayed_message_exchange、rabbitmq_federation、rabbitmq_sharding、rabbitmq_shovel、rabbitmq_tracing、rabbitmq_mqtt、rabbitmq_web_mqtt、rabbitmq_stomp、rabbitmq_web_stomp 和 rabbitmq_consistent_hash_exchange。

14.4.2 消息堆积对业务的影响及解决办法

消息堆积对业务的影响

过量的消息堆积可能会引起内存或磁盘告警，从而造成所有 connection 阻塞，进而影响到其他队列的使用，导致整体服务质量的下降。

消息堆积产生的原因

1. 一般来说消息堆积是由于生产消息的速率远大于消费消息的速率所导致的。比如某个时间段消费端处理消息异常缓慢，发送一条消息只要 3 秒钟，而消费一条消息需要 1 分钟，每分钟发送 20 个消息，只有一个消息被消费端处理，这样队列中就会产生大量的消息堆积。

2. 消费者出现异常，生产者一直在发送消息，但是消费者不能消费，造成消息积压。
3. 消费者没有出现异常，但是消费者与队列间的订阅可能出现了异常，也会导致消息无法被消费从而造成堆积的情况。
4. 消费者正常，与队列间的订阅也正常，但是消费端的代码本身逻辑耗费时间长导致了消费能力降低，这时候就会出现 1 中的情况从而导致消息堆积。

解决消息堆积的办法

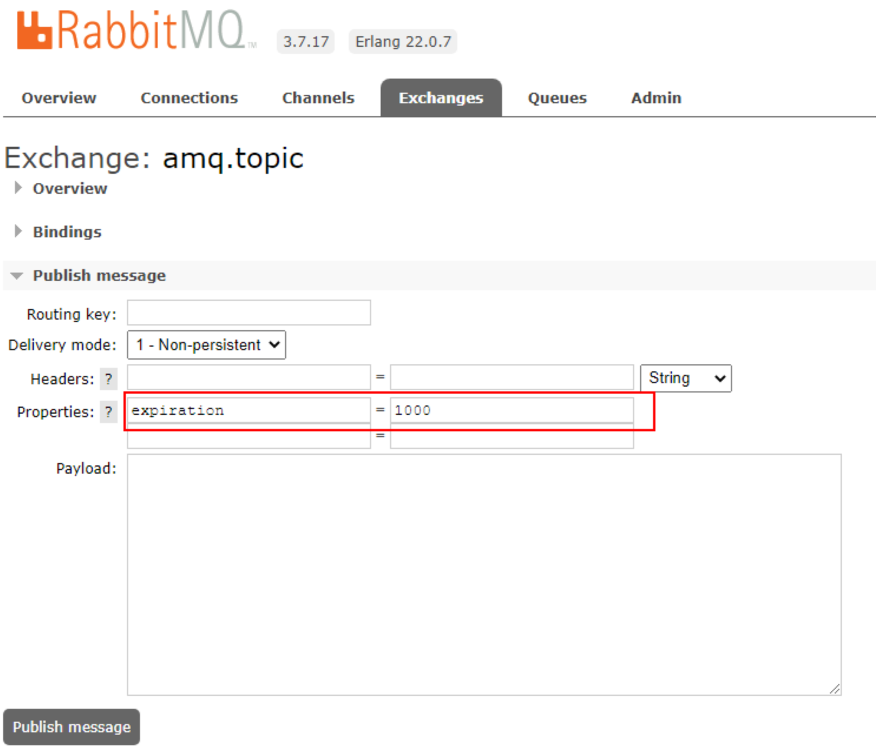
1. **生产速率较快，消费速率较慢**：您可以通过以下方法解决。
   - 增加消费者数量提高消费速率。
   - 采用生产者确认的发送模式，并监控生产端消息生产速度和时长，当消息生产时长有明显增加时进行流控措施。
2. **消费者异常**：建议排查消费者逻辑是不是有问题，优化程序。
3. **消费者与队列间的订阅异常**：建议排查队列和消费者之间的订阅是否正常。
4. **消费端的代码本身逻辑耗费时间长**：建议给消息设置过期时间，设置方法如下：
   - 在生产消息时，设置消息过期时间。消息过期时间以 expiration 值体现。
     - 在 properties 中设置 expiration 值，单位为毫秒(ms)。

```
AMQP.BasicProperties properties = new AMQP.BasicProperties().builder()
    .deliveryMode(2)
    .contentEncoding("UTF-8")
    .expiration("10000")
    .build();

String message = "hello rabbitmq";
channel.basicPublish(exchange, routingKey, properties,
message.getBytes(StandardCharsets.UTF_8));
```

     - 在 Web 界面中设置 expiration 值，单位为毫秒(ms)。

       登录 Web 界面，在“Exchanges”页签，单击 Exchange 名称，进入 Exchange 详情页。在“Publish message”区域，设置 expiration 值，如下图所示。



– 设置队列过期时间。队列过期时间以 x-message-ttl 值体现。从消息进入队列开始计算，超过了配置的队列过期时间，消息会自动被删除。

■ 在客户端代码中设置 x-message-ttl 值，单位为毫秒(ms)。

```
Map<String, Object> arguments = new HashMap<String, Object>();
arguments.put("x-message-ttl", 10000);
channel.queueDeclare(queueName, true, false, false, arguments);
```

■ 在 Web 界面新建队列时，设置 x-message-ttl 值，单位为毫秒(ms)。

登录 Web 界面，在“Exchanges”页签，新建队列时，设置 x-message-ttl 值，如下图所示。

14.4.3 消息的最长保留时间是多久？

一般情况下消息如果未被消费会一直保留，只有被消费后，才会被删除。但是如果设置了过期时间（TTL），则以 TTL 时间为准。

14.4.4 消息创建时间在哪设置？

消息创建时间是由生产客户端在生产消息时设置的。

14.5 监控告警问题

14.5.1 云监控无法展示 RabbitMQ 监控数据

监控数据无法展示，可能原因：队列名称开头包含特殊字符，例如点号“.”、下划线“_”，建议删除带特殊字符的队列。

14.5.2 云监控显示通道数一直上升报警有影响吗？

一个连接最大通道数是 2047，超过后再创建通道数会失败，建议排查是否为资源没有释放导致的。