9.4

规划 *IBM MQ*

注

在使用本资料及其支持的产品之前，请阅读第 183 页的『声明』中的信息。

内容

规划 IBM MQ 体系结构

规划 IBM MQ 环境时，请考虑 IBM MQ 为单个和多个队列管理器体系结构以及点到点和发布/预订消息传递样式提供的支持。 还要规划资源需求以及日志记录和备份工具的使用。

关于此任务

在规划 IBM MQ 体系结构之前，请熟悉基本 IBM MQ 概念。 请参阅 IBM MQ 技术概述。

IBM MQ 体系结构从使用单个队列管理器的简单体系结构到相互连接的队列管理器的更复杂网络。 使用分布式排队技术将多个队列管理器连接在一起。 有关规划单个队列管理器和多个队列管理器体系结构的更多信息，请参阅以下主题:

- 第 15 页的『基于单个队列管理器的体系结构』
- 第 16 页的『基于多个队列管理器的体系结构』
  - 第 17 页的『规划分布式队列和集群』
  - 第 58 页的『规划分布式发布/预订网络』

z/OS 在 IBM MQ for z/OS 上，您可以使用共享队列和队列共享组来实现工作负载均衡，并使 IBM MQ 应用程序具有可伸缩性和高可用性。 有关共享队列和队列共享组的信息，请参阅 共享队列和队列共享组。

IBM MQ 提供了两种不同的发行版模型:

- Long Term Support (LTS) 发行版最适合需要长期部署和最大稳定性的系统。
- Continuous Delivery (CD) 发行版适用于需要快速利用 IBM MQ 的最新功能增强功能的系统。

这两种发行版类型的安装方式都相同，但您需要了解与支持和迁移相关的注意事项。 有关更多信息，请参阅 IBM MQ 发行版类型和版本控制。

有关规划多个安装，存储和性能需求以及使用客户机的信息，请参阅其他子主题。

相关概念

IBM MQ 发行版类型和版本控制

第 119 页的『Planning your IBM MQ environment on z/OS』

When planning your IBM MQ environment, you must consider the resource requirements for data sets, page sets, Db2, Coupling Facilities, and the need for logging, and backup facilities. Use this topic to plan the environment where IBM MQ runs.

可用性、恢复和重新启动

相关任务

检查需求

确保消息不丢失 (日志记录)

IBM MQ 发行版类型: 规划注意事项

IBM MQ 的两种主要发行版类型是 Long Term Support (LTS) 和 Continuous Delivery (CD)。 对于每个受支持的平台，您选择的发行版类型会影响订购，安装，维护和迁移。

有关发行版类型的详细信息，请参阅 IBM MQ 发行版类型和版本控制。

IBM MQ for Multiplatforms 的注意事项

Multi

顺序

在 Passport Advantage 中， IBM MQ 9.4 有两个单独的 eAssemblies 。 一个包含 IBM MQ 9.4.0 Long Term Support 发行版的安装映像，另一个包含 IBM MQ 9.4.x Continuous Delivery 发行版的安装映像。 根据您选择的发行版，从 eAssembly 下载安装映像。

所有 IBM MQ 版本以及 IBM MQ 9.4 的 LTS 发行版和 CD 发行版都属于同一产品标识。

使用 IBM MQ 的权利在整个产品 (PID) 中扩展，受许可组件和定价指标的约束。 这意味着您可以在 IBM MQ 9.4 的 LTS 发行版和 CD 发行版安装映像之间自由选择。

安装

从 Passport Advantage 下载安装映像后，应仅为安装选择已购买权利的组件。 请参阅 IBM MQ 许可证信息 ，以获取有关针对每个可收费组件包含哪些可安装组件的更多信息。

可以在同一操作系统映像上安装 IBM MQ 9.4.0 LTS 发行版和 IBM MQ 9.4.x CD 发行版。 如果执行此操作，那么这些组件将显示为 IBM MQ 多版本支持所支持的单独安装。 每个版本都有与该版本关联的不同队列管理器集。

每个新的 CD 发行版都作为安装映像提供。 新的 CD 发行版可以与现有发行版一起安装，也可以由安装程序将较早的 CD 发行版更新为新发行版。

CD 发行版包含功能增强以及最新的一组缺陷修订和安全性更新。 每个 CD 发行版都是累积发行版，并且完全替换该版本的 IBM MQ 的所有先前发行版。 因此，如果特定 CD 发行版不包含任何与企业相关的功能，那么可以跳过该发行版。

维护

LTS 发行版由提供缺陷修订的修订包和提供安全补丁的累积安全性更新 (CSU) 的应用程序提供服务。 修订包和 CSU 定期可用，并且是累积的。

对于 CD，仅针对最新的 CD 发行版 (可能在后续版本上) 生成 CSU。

有时， IBM 支持团队会指示您应用临时修订。 临时修订也称为紧急修订或测试修订，用于应用无法等待下一次维护交付的紧急更新。

LTS 发行版与 CD 发行版之间的迁移

存在一些约束和限制，但通常可以将单个队列管理器从使用 LTS 发行版代码迁移到 CD 发行版代码，或者从使用 CD 发行版代码迁移到 LTS 发行版代码，前提是目标发行版高于迁移之前使用的发行版。

可以采用两种方法:

- 安装新的代码发行版，以便更新 IBM MQ 的现有安装。 与安装关联的任何队列管理器都将在启动时使用新的代码发行版。
- 将新的代码发行版作为新安装安装，然后使用 setmqm 命令将各个队列管理器实例移至新安装。

当队列管理器开始运行代码的 CD 发行版时，将更新队列管理器命令级别以指示新的发行版级别。 这意味着将启用发行版中提供的任何新功能，并且您无法再使用具有较低 VRM 数字的代码发行版来重新启动队列管理器。

IBM MQ for z/OS 的注意事项

z/OS

顺序

订购 IBM MQ for z/OS 9.4 时， ShopZ 上提供了两个单独的功能部件。 这些功能部件对应于 LTS 发行版和 CD 发行版。 这两个功能部件都适用于相同的产品标识 (PID)。 这是已获得许可的产品标识，因此，如果一个功能部件已获得许可，那么将有权使用备用功能部件 (如果需要)。 订购时，选择与 LTS 发行版或 CD 发行版对应的功能部件。

如果要选择要包含在 ServerPac 中的产品，那么不能按同一 ServerPac 顺序同时选择 LTS 发行版和 CD 发行版，因为 SMP/E 无法将这些产品安装在同一目标区域中。

安装

LTS 和 CD 发行版在单独的 FMID 集合中提供。 请注意，这些 FMID 不能安装在同一个 SMP/E 目标区域中。 如果需要 LTS 和 CD 发行版:

- 将 LTS 发行版和 CD 发行版安装在不同的目标区域中。
- 为两个发行版维护单独的目标库和分发库。

如果队列管理器位于队列共享组中，那么在升级到最新 CD 版本时，必须升级该组中的所有队列管理器。

队列管理器的命令级别是三位数的 VRM 级别。 IBM MQ 程序可以通过传递 MQIA_COMMAND_LEVEL 选择器来调用 MQINQ，以获取它所连接到的队列管理器的命令级别。

由于发行版使用不同的 FMID，因此您无法通过维护 LTS 发行版或其他方法来更新 CD 发行版。 同样，无法将产品代码版本从 LTS 发行版切换到 CD 发行版或其他方式。 但是，您可以在发布模型之间切换队列管理器。 请参阅 LTS 发行版与 CD 发行版之间的迁移。

注:

IBM MQ 9.0.x 和 IBM MQ 9.1.x CD 发行版具有独立的版本和发行版从属 FMID。 因此，从 9.0.x CD 移动到 9.1.x CD 需要至少一个完整的 SMP/E 安装。

从 IBM MQ for z/OS 9.2.0 开始， CD 发行版使用一组 FMID，对于版本号为 9 的所有 IBM MQ for z/OS 发行版，这些 FMID 保持不变。 由于每个新版本的 IBM MQ 都作为 CD 和 LTS 发行版提供，因此您可以通过将 PTF 应用于单个 SMP/E 安装来升级 CD 发行版，即使在跨越主要版本边界时也是如此。 例如，您可以从 IBM MQ for z/OS 9.2.0 CD，到 IBM MQ for z/OS 9.2.2 CD，到 IBM MQ for z/OS 9.2.4 CD，到 IBM MQ for z/OS 9.3.0 CD，只需应用 PTF 即可。

您可以通过查看队列管理器作业日志中的 CSQY000I 消息来区分具有相同 VRM 级别的 LTS 和 CD 发行版。

维护

IBM MQ for z/OS 使用 PTF 进行维护。

LTS PTF 特定于对应于特定发行版级别的一组特定库。 对于 UNIX 系统服务功能部件 (即， JMS 和 WEB UI，连接器包和 Managed File Transfer)， z/OS PTF 与多平台修订包和累积安全性更新 (CSU) 直接对齐。 这些修订是累积的，并且与等效的多平台修订包或 CSU 同时可用。

CD CD CSU 通常在 CD 发行版之间不可用，但包含在下一个 IBM MQ for z/OS CD 发行版中。 您还可以联系支持人员以请求 + + USERMOD。

IBM MQ for z/OS 上的其他修订是特定部件上的不同修订。 这些修订可解决非累积的特定问题，并在产生这些修订时提供这些修订。

LTS 发行版与 CD 发行版之间的迁移

存在一些约束和限制，但通常可以将单个队列管理器从使用 LTS 发行版代码迁移到 CD 发行版代码，或者从使用 CD 发行版代码迁移到 LTS 发行版代码，前提是目标发行版高于迁移之前使用的发行版。

从 IBM MQ for z/OS 9.2.0 开始，您可以根据需要在具有相同 VRM 的 CD 和 LTS 发行版之间来回迁移多次，而不会影响向后迁移的能力。 例如，可以在 IBM MQ for z/OS 9.3.0 LTS 上启动队列管理器，然后在 IBM MQ for z/OS 9.3.0 CD 上关闭并启动队列管理器，然后在 IBM MQ for z/OS 9.3.0 LTS 上关闭并启动队列管理器。

IBM MQ for z/OS 传统上提供了回退功能 (向后迁移)，以便在迁移后运行一段时间后，可以回退到先前发行版。 对于 LTS 发行版以及修饰符为 0 (例如 9.3.0 CD) 的 CD 发行版，将保留此功能，但当迁移的源或目标是具有非零修饰符号 (例如 9.2.5 或 9.3.1) 的 CD 发行版时，无法保留此功能。

以下是有效的迁移方案，并说明此原则如何工作:

源发行版	目标发行版	注意
9.1.0 LTS	9.4.0 LTS 或 9.4.0 CD	不支持向后迁移，因为 9.1.0 LTS 超出标准支持范围。
9.2.0 LTS	9.4.0 LTS 或 9.4.0 CD	支持向后迁移。
9.3.0 LTS	9.4.0 LTS 或 9.4.0 CD	支持向后迁移。
9.3.5 CD	9.4.0 LTS 或 9.4.0 CD	不支持向后迁移，因为源发行版是具有非零修饰符的 CD。

源发行版	目标发行版	注意
9.4.0 LTS 或 9.4.0 CD	9.4.1 CD	不支持向后迁移，因为目标发行版是具有非零修饰符的 CD。 发出 `Write to operator with reply` CSQY041D 以确认迁移。

相关任务

z/OS 在 z/OS 上应用和除去维护

相关信息

下载 IBM MQ 9.4

针对 GDPR 就绪状态的 IBM MQ 和 IBM MQ Appliance 本地注意事项

适用于下列 PID：

分布式

- IBM MQ/IBM MQ Advanced - 5724-H72
- IBM MQ for HPE NonStop - 5724-A39

z/OS

- IBM MQ for z/OS - 5655-MQ9
- IBM MQ for z/OS Value Unit Edition - 5655-VU9
- IBM MQ Advanced for z/OS - 5655-AV9
- IBM MQ Advanced for z/OS Value Unit Edition - 5655-AV1

IBM MQ Appliance

- IBM MQ Appliance M2003 -5900-ALJ
- IBM MQ Appliance M2002 - 5737-H47

声明：

此文档旨在帮助您做好 GDPR 准备工作。 它提供的信息涉及您可以配置的 IBM MQ 功能、该产品的各方面使用，以及为了帮助组织针对 GDPR 做好准备而应该考虑的事情。 由于客户选择和配置功能部件的方式有很多种，并且产品单独使用以及与第三方应用程序和系统配合使用的方式也多种多样，因此这些信息并不是一份详尽的列表。

客户负责确保自己遵守各种法律法规，包括欧盟一般数据保护条例。 客户须自行负责从合格的法律顾问那里，就可能会影响客户业务和客户为了遵守此类法律和法规需要采取的任何行动，获得关于任何相关法律和法规的认定和解释的意见。

本文描述的产品、服务和其他功能不适用于所有客户情况，可能具有受限可用性。 IBM 不提供法律，会计或审计建议，也不表示或保证其服务或产品将确保客户遵守任何法律或法规。

目录

7. 数据处理
8. 数据删除
9. 数据监视
10. 限制使用个人数据的功能
11. 文件处理

欧洲通用数据保护条例

欧盟（“EU”）已采用了《一般数据保护条例》(GDPR) 并于 2018 年 5 月 25 日起实施。

为什么 GDPR 很重要?

GDPR 建立了用于处理个人数据的更强大的数据保护监管框架。 GDPR 实现:

- 新增及增强的个人权利
- 更广泛的个人数据定义
- 新增的处理商义务
- 可能会因不合规而遭受重大经济处罚
- 强制性数据违规通知

请阅读关于 GDPR 的更多信息:

- 欧盟 GDPR 信息门户网站
- ibm.com/GDPR Web 站点

产品配置 - 为 GDPR 做准备时的注意事项

以下部分提供配置 IBM MQ 以帮助贵组织准备 GDPR 就绪的注意事项。

数据生命周期

IBM MQ 是面向事务性消息的中间件产品，使应用程序能够异步交换应用程序提供的数据。 IBM MQ 支持一系列用于连接应用程序的消息传递 API，协议和网桥。 因此，可以使用 IBM MQ 来交换多种形式的数据，其中一些数据可能受 GDPR 的约束。 IBM MQ 可能会与多种第三方产品交换数据。 其中一些是 IBM 拥有的，但其他许多是由其他技术供应商提供的。 软件产品兼容性报告 Web 站点 提供关联软件的列表。 有关第三方产品的 GDPR 就绪性的注意事项，请参阅该产品的文档。 IBM MQ 管理员通过队列，主题和预订的定义来控制 IBM MQ 与通过它传递的数据进行交互的方式。

哪些类型的数据流经 IBM MQ?

由于 IBM MQ 为应用程序数据提供异步消息传递服务，因此此问题没有一个确定的答案，因为用例因应用程序部署而异。 应用程序消息数据持久存储在队列文件 (页集或 z/OS 上的耦合设施)，日志和归档中，并且消息本身可能包含由 GDPR 管理的数据。 应用程序提供的消息数据也可能包含在为问题确定目的而收集的文件中，例如错误日志，跟踪文件和 FFST。 在 z/OS 应用程序上，提供的消息数据也可能包含在地址空间或耦合设施转储中。

以下是可使用 IBM MQ 交换的个人数据的一些典型示例:

- 客户的员工 (例如， IBM MQ 可能用于连接客户的工资单或 HR 系统)
- 客户自己的客户个人数据 (例如， IBM MQ 可能由客户用于在与其客户相关的应用程序之间交换数据，例如，获取销售线索并将数据存储在其 CRM 系统中)。
- 客户自己的客户的敏感个人数据 (例如， IBM MQ 可能在需要交换个人数据的行业环境中使用，例如，集成临床应用程序时基于 HL7-based 医疗保健记录)。

除了应用程序提供的消息数据外， IBM MQ 还会处理以下类型的数据:

- 认证凭证 (例如用户名和密码， API 密钥等)
- 技术上可识别的个人信息（例如，设备标识、基于使用情况的标识、IP 地址等 - 当链接到个人时）

用于与 IBM 在线联系的个人数据

IBM MQ 客户可以通过多种方式提交在线评论/反馈/请求，以联系 IBM IBM MQ 主题，主要包括:

- IBM Developer 上的 IBM MQ 区域 中的页面上的公共评论区域
- IBM Documentation 中的 IBM MQ 产品信息 页面上的公共评论区域
- IBM 支持论坛 中的公共评论
- IBM Integration Ideas 中的公共评论

通常，只有客户姓名和电子邮件地址用来启用联系主题的个人回复，而且个人数据的使用符合 IBM 在线隐私声明。

数据 收集

IBM MQ 可用于收集个人数据。 在评估您对 IBM MQ 的使用以及满足 GDPR 需求的需求时，应考虑在您的环境中传递 IBM MQ 的个人数据类型。 您可能想要考虑如下几个方面:

- 数据如何到达队列管理器? (跨哪些协议? 数据是否已加密？ 数据是否已签名？)
- 如何从队列管理器发送数据? (跨哪些协议? 数据是否已加密？ 数据是否已签名？)
- 数据在通过队列管理器时如何存储? (任何消息传递应用程序都有可能将消息数据写入有状态介质，即使消息是非持久的。 您是否知道消息传递功能可能如何公开通过产品传递的应用程序消息数据的各个方面?)
- 在 IBM MQ 需要访问第三方应用程序时，如何收集和存储凭证?

IBM MQ 可能需要与需要认证的其他系统和服务 (例如 LDAP) 进行通信。 需要时，认证数据 (用户标识和密码) 由 IBM MQ 配置和存储，以便在此类通信中使用。 尽可能避免使用个人凭证进行 IBM MQ 认证。 请考虑保护用于认证数据的存储器。 (请参阅下面的 "数据存储"。)

数据存储

当消息数据通过队列管理器传输时， IBM MQ 会将该数据直接持久存储 (可能有多个副本) 到有状态介质。 IBM MQ 用户可能希望考虑在消息数据处于静态状态时对其进行保护。

以下项目突出显示了 IBM MQ 持久存储应用程序提供的数据的区域，用户在确保符合 GDPR 时可能需要考虑这些区域。

- 应用程序消息队列:

  IBM MQ 提供消息队列以允许应用程序之间进行异步数据交换。 存储在队列上的非持久和持久消息将写入有状态介质。
- 文件传输代理队列:

  IBM MQ Managed File Transfer 利用消息队列来协调文件数据的可靠传输，包含个人数据和传输记录的文件存储在这些队列上。
- 传输队列:

  为了在队列管理器之间可靠地传输消息，将消息临时存储在传输队列上。
- 死信队列:

  在某些情况下，如果在队列管理器上配置了死信队列，那么无法将消息放入目标队列并将其存储在死信队列上。
- 回退队列:

  JMS 和 XMS 消息传递接口提供了一种功能，允许在发生多次回退后将有害消息移至回退队列，以允许处理其他有效消息。
- AMS 错误队列:

  IBM MQ Advanced Message Security 会将不符合安全策略的消息移动到 SYSTEM.PROTECTION.ERROR.QUEUE 错误队列与死信队列相似。
- 保留出版物:

  IBM MQ 提供了保留发布功能，以允许预订应用程序重新调用先前的发布。
- 延迟交付:

IBM MQ 支持 JMS 2.0 和 Jakarta Messaging 3.0 传递延迟功能，该功能支持将来将消息传递到其目标。尚未传递的消息存储在 SYSTEM.DDELAY.LOCAL.QUEUE 队列。

阅读更多：

- 日志记录: 确保消息不会丢失
- MFT 代理队列设置
- 使用死信队列
- 在 IBM MQ classes for JMS 中处理有害消息
- AMS 错误处理
- 保留的发布内容
- JMS 2.0 交付延迟

以下项目突出显示了 IBM MQ 可能间接持久存储应用程序提供的数据的区域，用户在确保遵守 GDPR 时也可能希望考虑这些数据。

- 跟踪路由消息传递:

  IBM MQ 提供跟踪路由功能，用于记录消息在应用程序之间采用的路由。 生成的事件消息可能包括技术上可识别的个人信息，例如 IP 地址。
- 应用程序活动跟踪:

  IBM MQ 提供了应用程序活动跟踪，用于记录应用程序和通道的消息传递 API 活动，应用程序活动跟踪可以将应用程序提供的消息数据的内容记录到事件消息中。
- 服务跟踪:

  IBM MQ 提供了服务跟踪功能，用于记录消息数据流所通过的内部代码路径。 作为这些功能的一部分，IBM MQ 可以记录应用程序提供的消息数据的内容，以跟踪存储在磁盘上的文件。
- 队列管理器事件:

  IBM MQ 可以生成可包含个人数据的事件消息，例如权限，命令和配置事件。

阅读更多：

- 跟踪路由消息传递
- 使用跟踪
- 事件监视
- 队列管理器事件

要保护对应用程序提供的消息数据副本的访问权，请考虑以下操作：

- 限制特权用户对文件系统中 IBM MQ 数据的访问权，例如限制 UNIX and Linux® 平台上 "mqm" 组的用户成员资格。
- 通过专用队列和访问控制限制应用程序对 IBM MQ 数据的访问。 在适当情况下，避免不必要的资源 (例如应用程序之间的队列) 共享，并提供对队列和主题资源的细粒度访问控制。
- 限制对高可用性 (HA) 或灾难恢复 (DR) 配置中 IBM MQ 数据的复制副本的访问，并保护用于复制的连接。
- 使用 IBM MQ Advanced Message Security 提供消息数据的端到端签名和/或加密。
- 使用文件或卷级别加密来保护可能包含 IBM MQ 数据，跟踪或日志的目录或文件系统。
- 将服务跟踪上载到 IBM 后，如果您关注可能包含个人数据的内容，那么可以删除服务跟踪文件和 FFST 数据。

阅读更多：

- 特权用户
- 在 Multiplatforms 版上规划文件系统支持
- IBM MQ Appliance 上的文件系统加密

IBM MQ 管理员可以使用凭证 (用户名和密码， API 密钥等) 配置队列管理器 针对第三方服务 (例如 LDAP)。此数据通常存储在通过文件系统许可权保护的队列管理器数据目录中。

创建 IBM MQ 队列管理器时，将使用基于组的访问控制来设置数据目录，以便 IBM MQ 可以读取配置文件并使用凭证连接到这些系统。 IBM MQ 管理员被视为特权用户，并且是此组的成员，因此具有对文件的读访问权。 某些文件已加密，但未加密。 因此，要完全保护对凭证的访问权，您应该考虑以下操作:

- 限制特权用户对 IBM MQ 数据的访问权，例如限制 UNIX and Linux 平台上 "mqm" 组的成员资格。
- 使用文件或卷级别加密来保护队列管理器数据目录的内容。
- 对生产配置目录的备份进行加密，并使用相应的访问控制来存储这些备份。
- 考虑通过安全性，命令和配置事件为认证失败，访问控制和配置更改提供审计跟踪。

阅读更多:

- 保护 IBM MQ

数据访问

可以通过以下产品接口访问 IBM MQ 队列管理器数据，其中一些设计用于通过远程连接进行访问，其他设计用于通过本地连接进行访问。

- IBM MQ 控制台 [仅远程]
- IBM MQ 管理 REST API [仅远程]
- IBM MQ 消息传递 REST API [仅远程]
- MQI [本地和远程]
- JMS [本地和远程]
- XMS [本地和远程]
- IBM MQ 遥测 (MQTT) [仅远程]
- IBM MQ Light (AMQP) [仅远程]
- IBM MQ IMS 网桥 [仅本地]
- IBM MQ CICS 网桥 [仅本地]
- IBM MQ MFT 协议网桥 [仅远程]
- IBM MQ Connect:Direct 网桥 [仅远程]
- IBM MQ MQAI [本地和远程]
- IBM MQ PCF 命令 [本地和远程]
- IBM MQ MQSC 命令 [本地和远程]
- IBM MQ Explorer [本地和远程]
- IBM MQ 用户出口 [仅本地]
- IBM MQ Internet Pass-Thru [仅远程]
- Red Hat® OpenShift® 监视 (Prometheus) 度量 (这些度量是有关队列管理器统计信息的数字数据)
- IBM MQ Appliance 串行控制台 [仅本地]
- IBM MQ Appliance SSH [仅远程]
- IBM MQ Appliance REST API [仅远程]
- IBM MQ Appliance Web UI [仅远程]
- V 9.4.0 IBM MQ Kafka 连接器 (Kafka Connect) [本地和远程]

这些接口旨在允许用户对 IBM MQ 队列管理器以及存储在其中的消息进行更改。 管理和消息传递操作是安全的，因此在发出请求时涉及三个阶段;

- 认证
- 角色映射
- 授权

认证:

如果从本地连接请求了消息或管理操作，那么此连接的源是同一系统上正在运行的进程。 运行该进程的用户必须已通过操作系统提供的任何认证步骤。 将从中建立连接的进程的所有者的用户名声明为身份。 例如，这可能是运行已从中启动应用程序的 shell 的用户的名称。 本地连接的可能认证形式为:

1. 已声明的用户名 (本地操作系统)
2. 可选用户名和密码 (操作系统， LDAP 或定制 3rd 存储库)
3. 仅限安全性令牌 (JWT) IBM MQ

如果从远程连接请求了管理操作，那么将通过网络接口与 IBM MQ 进行通信。 以下形式的身份可以通过网络连接进行认证;

1. 已声明的用户名 (来自远程操作系统)
2. 用户名和密码 (操作系统， LDAP 或定制 3rd 存储库)
3. 源网络地址 (例如 IP 地址)
4. X.509 数字证书 (相互 SSL/TLS 认证)
5. 安全性令牌 (例如 LTPA2 令牌或 JWT 令牌)
6. 其他定制安全性 (3rd 参与方出口提供的功能)
7. SSH 密钥

IBM MQ 与 IBM Cloud Pak for Integration 的集成为 IBM MQ Console 添加了新的认证类型: 使用 Cloud Pak 进行单点登录。 (仅限 CP4I)

角色映射:

在角色映射阶段，可以将认证阶段中提供的凭证映射到备用用户标识。 如果允许映射的用户标识继续 (例如，管理用户可能被通道认证规则阻止) ，那么在针对 IBM MQ 资源授权活动时，映射的用户标识将转入最终阶段。

授权:

IBM MQ 使不同用户能够对不同的消息传递资源 (例如队列，主题和其他队列管理器对象) 具有不同的权限。

日志记录活动:

IBM MQ 的某些用户可能需要创建访问 MQ 资源的审计记录。 理想审计日志的示例可能包括配置更改，其中包含有关更改及其请求者的信息。

以下信息来源可用于实现此要求:

1. 可以将 IBM MQ 队列管理器配置为在成功运行管理命令时生成命令事件。
2. 可以将 IBM MQ 队列管理器配置为在创建，变更或删除队列管理器资源时生成配置事件。
3. 可以将 IBM MQ 队列管理器配置为在资源的授权检查失败时生成权限事件。
4. 指示授权检查失败的错误消息将写入队列管理器错误日志。
5. 当认证，授权检查失败或者创建，启动，停止或删除队列管理器时， IBM MQ 控制台会将审计消息写入其日志。
6. IBM MQ Appliance 会将审计消息写入其日志以记录用户登录和系统更改。

在考虑这些类型的解决方案时， IBM MQ 用户可能希望考虑以下几点:

- 事件消息是非持久的，因此当队列管理器重新启动时，信息将丢失。 应将任何事件监视器配置为持续使用任何可用消息并将内容传输到持久介质。
- IBM MQ 特权用户具有足够的特权来禁用事件，清除日志或删除队列管理器。

有关保护对 IBM MQ 数据的访问并提供审计跟踪的更多信息，请参阅以下主题:

- IBM MQ 安全性机制
- 配置事件
- 命令事件
- 使用错误日志

数据处理

使用公共密钥基础结构进行加密：

您可以通过指定连接使用 TLS 来保护与 IBM MQ 的网络连接，这还可以提供连接起始端的相互认证。

使用传输机制所提供的 PKI 安全工具是保护 IBM MQ 数据处理的第一步。 但是，在不启用进一步的安全功能的情况下，消费应用程序的行为是处理传递给它的所有消息，而不验证消息的来源或在传输过程中是否发生了更改。

获得许可使用 Advanced Message Security (AMS) 功能的 IBM MQ 用户可以通过定义和配置安全策略来控制应用程序处理消息中保存的个人数据的方式。 安全策略允许将数字签名和/或加密应用于应用程序之间的消息数据。

在使用消息时，可以使用安全策略来要求和验证数字签名，以确保消息是真实的。 AMS 加密提供了一种方法，通过该方法将消息数据从可读格式转换为编码版本，只有当它是预期的接收方或消息并且有权访问正确的解密密钥时，才能由另一个应用程序进行解码。

有关使用 SSL 和证书来保护网络连接的更多信息，请参阅 IBM MQ 产品文档中的以下主题:

- 为 IBM MQ
- AMS 概述

数据删除

IBM MQ 提供了用于删除已提供给产品的数据的命令和用户界面操作。 这意味着如果需要， IBM MQ 的用户可以删除与特定个人相关的数据。

- 要考虑遵守 GDPR 客户机数据删除的 IBM MQ 行为区域
  - 通过以下方法删除存储在应用程序队列上的消息数据:
    - 使用消息传递 API 或工具或使用消息到期来除去个别消息。
    - 指定消息是非持久消息，保留在非持久消息类正常的队列上，并重新启动队列管理器。
    - 以管理方式清除队列。
    - 正在删除队列。
  - 通过以下方法删除存储在主题上的保留发布数据:
    - 指定消息是非持久消息，并重新启动队列管理器。
    - 将保留的数据替换为新数据或使用消息到期。
    - 以管理方式清除主题字符串。
  - 通过删除整个队列管理器以及用于高可用性或灾难恢复的任何复制副本，删除存储在队列管理器上的数据。
  - 通过删除跟踪目录中的文件来删除服务跟踪命令所存储的数据。
  - 删除通过删除 errors 目录中的文件存储的 FFST 数据。
  - 删除地址空间和耦合设施转储 (在 z/OS 上)。
  - 删除此类数据的归档，备份或其他副本。
- 要考虑遵守 GDPR 帐户数据删除的 IBM MQ 行为区域
  - 您可以通过删除 (包括归档，备份或其他复制副本) 来删除 IBM MQ 存储的用于连接到队列管理器和 3rd 服务的帐户数据和首选项:
    - 用于存储凭证的队列管理器认证信息对象。
    - 队列管理器权限记录引用用户标识。
    - 用于映射或阻止特定 IP 地址，证书 DN 或用户标识的队列管理器通道认证规则。
    - 由 IBM MQ Managed File Transfer 代理程序，记录器和 MQ Explorer MFT 插件用于向队列管理器和文件服务器进行认证的凭证文件。

- X.509 数字证书，用于表示或包含有关可由 SSL/TLS 连接或 IBM MQ Advanced Message Security (AMS) 使用的密钥库中个人的信息。
- 来自 IBM MQ Appliance 的个人用户帐户，包括系统日志文件中对这些帐户的引用。
- IBM MQ Explorer 工作空间元数据和 Eclipse 设置。
- IBM MQ Explorer 密码存储，如 密码首选项中所指定。
- IBM MQ 控制台和 mqweb 服务器配置文件。
- IBM MQ Internet Pass-Thru 配置文件和密钥库。

阅读更多：

- MFT 和 IBM MQ 连接认证
- 使用 ProtocolBridgeCredentials.xml 文件映射文件服务器的凭证
- 配置 IBM MQ Console 用户和角色

数据监视

IBM MQ 提供了一系列监视功能，用户可以利用这些功能来更好地了解应用程序和队列管理器的执行方式。

IBM MQ 还提供了一些帮助管理队列管理器错误日志的功能。

阅读更多：

- 监视 IBM MQ 网络
- 诊断消息服务
- QMErrorLog 服务
- IBM MQ Appliance 监视和报告

限制使用个人数据的功能

通过使用本文档中概述的工具，IBM MQ 允许最终用户限制对其个人数据的使用。

IBM MQ 消息队列不应以与数据库相同的方式用作永久数据存储器，在处理受 GDPR 约束的应用程序数据时尤其如此。

与可以通过搜索查询找到数据的数据库不同，除非您知道消息的队列，消息和相关标识，否则很难找到消息数据。

提供的包含个人数据的消息可以随时识别和定位，可以使用标准 IBM MQ 消息传递功能来访问或修改消息数据。

文件处理

1. IBM MQ Managed File Transfer 不会对传输的文件执行恶意软件扫描。 按原样传输文件，并执行完整性检查以确保在传输期间不修改文件数据。 源和目标校验和作为传输状态发布的一部分发布。 建议最终用户在 MFT 将文件传输到远程端点之前以及在 MFT 将文件传递到远程端点之后，针对其环境实施相应的恶意软件扫描。
2. IBM MQ Managed File Transfer 不会根据 MIME 类型或文件扩展名执行操作。 MFT 读取文件并传输与从输入文件读取的字节完全相同的字节。

基于单个队列管理器的体系结构

最简单的 IBM MQ 体系结构涉及单个队列管理器的配置和使用。

在规划 IBM MQ 体系结构之前，请熟悉基本 IBM MQ 概念。 请参阅 IBM MQ 技术概述。

以下部分描述了使用单个队列管理器的许多可能的体系结构：

- 第 16 页的『具有访问服务的本地应用程序的单个队列管理器』
- 第 16 页的『具有作为客户机访问服务的远程应用程序的单个队列管理器』

- 第 16 页的『具有发布/预订配置的单个队列管理器』

具有访问服务的本地应用程序的单个队列管理器

基于单个队列管理器的第一个体系结构是访问服务的应用程序与提供服务的应用程序在同一系统上运行。 IBM MQ 队列管理器提供请求服务的应用程序与提供服务的应用程序之间的异步相互通信。 这意味着即使其中一个应用程序处于脱机状态很长一段时间，应用程序之间的通信也可以继续。

具有作为客户机访问服务的远程应用程序的单个队列管理器

基于单个队列管理器的第二个体系结构具有从提供服务的应用程序远程运行的应用程序。 远程应用程序在服务的不同系统上运行。 应用程序作为客户机连接到单个队列管理器。 这意味着可以通过单个队列管理器向多个系统提供对服务的访问权。

此体系结构的一个限制是网络连接必须可供应用程序运行。 应用程序与队列管理器之间通过网络连接进行的交互是同步的。

具有发布/预订配置的单个队列管理器

使用单个队列管理器的备用体系结构是使用发布/预订配置。 在发布/预订消息传递中，可以将信息提供者与该信息的使用者分离。 这与先前描述的体系结构中的点到点消息传递样式不同，其中应用程序必须知道有关目标应用程序的信息，例如要将消息放入的队列名称。 通过使用 IBM MQ 发布/预订，发送应用程序将根据信息主题发布具有指定主题的消息。 IBM MQ 处理将消息分发到已通过预订在该主题中注册兴趣的应用程序。 接收应用程序也不需要知道要接收这些应用程序的消息源的任何信息。 有关更多信息，请参阅 发布/预订消息传递 和 单个队列管理器发布/预订配置的示例。

相关概念

IBM MQ 简介

相关任务

第 5 页的『规划 IBM MQ 体系结构』

规划 IBM MQ 环境时，请考虑 IBM MQ 为单个和多个队列管理器体系结构以及点到点和发布/预订消息传递样式提供的支持。 还要规划资源需求以及日志记录和备份工具的使用。

在 Multiplatforms 版上创建和管理队列管理器

基于多个队列管理器的体系结构

您可以使用分布式消息排队技术来创建涉及配置和使用多个队列管理器的 IBM MQ 体系结构。

在规划 IBM MQ 体系结构之前，请熟悉基本 IBM MQ 概念。 请参阅 IBM MQ 技术概述。

可以通过添加其他队列管理器来更改 IBM MQ 体系结构，而无需更改提供服务的应用程序。

应用程序可以与队列管理器托管在同一台机器上，然后与另一个系统上的另一个队列管理器上托管的服务进行异步通信。 或者，访问服务的应用程序可以作为客户机连接到队列管理器，然后在另一个队列管理器上提供对服务的异步访问。

用于连接不同队列管理器及其队列的路由是使用分布式排队技术定义的。 体系结构中的队列管理器使用通道进行连接。 通道用于根据队列管理器的配置在一个方向上自动将消息从一个队列管理器移动到另一个队列管理器。

有关规划 IBM MQ 网络的高级概述，请参阅 第 18 页的『设计分布式队列管理器网络』。

有关如何为 IBM MQ 体系结构规划通道的信息，请参阅 IBM MQ 分布式排队技术。

分布式队列管理使您能够创建和监视队列管理器之间的通信。 有关分布式队列管理的更多信息，请参阅 分布式队列管理简介。

相关任务

第 5 页的『规划 IBM MQ 体系结构』

规划 IBM MQ 环境时，请考虑 IBM MQ 为单个和多个队列管理器体系结构以及点到点和发布/预订消息传递样式提供的支持。 还要规划资源需求以及日志记录和备份工具的使用。

在 Multiplatforms 版上创建和管理队列管理器

规划分布式队列和集群

您可以手动连接分布式队列管理器上托管的队列，也可以创建队列管理器集群并让产品为您连接队列管理器。 要为分布式消息传递网络选择合适的拓扑，需要考虑您对手动控制，网络大小，更改频率，可用性和可伸缩性的需求。

开始之前

此任务假定您了解分布式消息传递网络是什么以及它们的工作方式。 有关技术概述，请参阅 分布式排队和集群。

关于此任务

要创建分布式消息传递网络，您可以手动配置通道以连接在不同队列管理器上托管的队列，也可以创建队列管理器集群。 集群使队列管理器能够相互通信，而无需设置额外的通道定义或远程队列定义，从而简化其配置和管理。

要为分布式发布/预订网络选择合适的拓扑，您需要考虑以下广泛问题:

- 您需要对网络中的连接进行多少手动控制?
- 您的网络将有多大?
- 它将具有多大的动态性?
- 您的可用性和可伸缩性需求是什么?

过程

- 请考虑您需要对网络中的连接进行多少手动控制。

  如果只需要几个连接，或者需要非常精确地定义各个连接，那么您可能应该手动创建网络。

  如果您需要多个逻辑相关的队列管理器，并且需要共享数据和应用程序，那么应考虑将它们分组到队列管理器集群中。
- 估算网络需要的大小。

  a) 估算需要多少队列管理器。 请记住，可以在多个队列管理器上托管队列。

  b) 如果您正在考虑使用集群，请添加两个额外的队列管理器以充当完整存储库。

  对于较大的网络，手动配置和维护连接可能非常耗时，您应该考虑使用集群。
- 请考虑网络活动的动态程度。

  规划要在性能队列管理器上托管的繁忙队列。

  如果您期望频繁创建和删除队列，请考虑使用集群。
- 考虑您的可用性和可伸缩性需求。

  a) 决定是否需要保证队列管理器的高可用性。 如果是这样，请估算此需求适用于多少队列管理器。

  b) 请考虑某些队列管理器的能力是否低于其他队列管理器。

  c) 请考虑到某些队列管理器的通信链路是否比其他队列管理器更脆弱。

  d) 请考虑在多个队列管理器上托管队列。

  手动配置的网络和集群都可以配置为高可用性和可伸缩性。 如果使用集群，那么需要将两个额外的队列管理器定义为完整存储库。 拥有两个完整存储库可确保集群在其中一个完整存储库变为不可用时继续运行。 确保完整的存储库队列管理器功能强大，性能良好，并且具有良好的网络连接。 请勿计划将完整存储库队列管理器用于任何其他工作。
- 根据这些计算，使用提供的链接来帮助您决定是手动配置队列管理器之间的连接，还是使用集群。

下一步做什么

现在，您已准备好配置分布式消息传递网络。

相关任务

配置分布式队列

配置队列管理器集群

设计分布式队列管理器网络

IBM MQ 使用队列管理器和通道在应用程序之间以及通过网络发送和接收数据。 网络规划涉及定义需求，以创建用于通过网络连接这些系统的框架。

可以在您的系统与需要进行通信的任何其他系统之间创建通道。 可以创建多中继段通道以连接到没有直接连接的系统。 方案中描述的消息通道连接在 第 18 页的图 1 中显示为网络图。

如果需要在不同物理网络上的系统之间创建通道，或者需要在通过防火墙进行通信的通道之间创建通道，那么使用 IBM MQ Internet Pass-Thru 可以简化配置。 有关更多信息，请参阅 IBM MQ Internet Pass-Thru。

通道和传输队列名称

可以为传输队列提供任何名称。 但是为了避免混淆，您可以为它们提供与目标队列管理器名称或队列管理器别名相同的名称 (视情况而定)。 这使传输队列与它们使用的路由相关联，提供了通过中间 (多跳跃) 队列管理器创建的并行路由的清晰概述。

对于通道名称而言，它并不是那么清晰。 例如， 第 18 页的图 1 中针对 QM2 的通道名称对于入局和出局通道必须不同。 所有通道名称仍可包含其传输队列名称，但必须对其进行限定以使其唯一。

例如，在 QM2 上，有一个 QM3 通道来自 QM1，还有一个 QM3 通道转至 QM3。 要使名称唯一，第一个名称可以命名为 QM3_from_QM1，第二个名称可以命名为 QM3_from_QM2。 这样，通道名称在名称的第一部分中显示传输队列名称。 方向和相邻队列管理器名称显示在名称的第二部分中。

第 18 页的表 1 中给出了 第 18 页的图 1 的建议通道名称表。

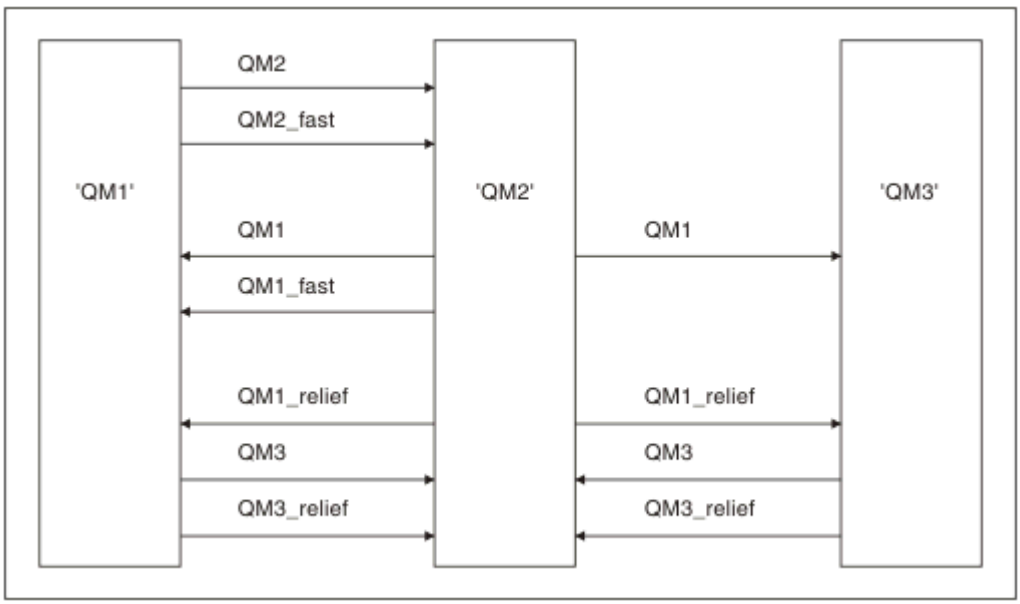


图 *1:* 显示所有通道的网络图

表 *1:* 通道名称示例

路由名称	队列管理器托管通道	传输队列的名称	建议的通道名称
QM1	QM1 & QM2	QM1 (位于 QM2)	QM1.from.QM2
QM1	QM2 & QM3	QM1 (位于 QM3)	QM1.from.QM3

表 1: 通道名称示例 (继续)			
路由名称	**队列管理器托管通道**	**传输队列的名称**	**建议的通道名称**
QM1_fast	QM1 & QM2	QM1_fast (位于 QM2)	QM1_fast.from.QM2
QM1_relief	QM1 & QM2	QM1_relief (位于 QM2)	QM1_relief.from.QM2
QM1_relief	QM2 & QM3	QM1_relief (位于 QM3)	QM1_relief.from.QM3
QM2	QM1 & QM2	QM2 (位于 QM1)	QM2.from.QM1
QM2_fast	QM1 & QM2	QM2_fast (位于 QM1)	QM2_fast.from.QM1
QM3	QM1 & QM2	QM3 (位于 QM1)	QM3.from.QM1
QM3	QM2 & QM3	QM3 (位于 QM2)	QM3.from.QM2
QM3_relief	QM1 & QM2	QM3_relief (位于 QM1)	QM3_relief.from.QM1
QM3_relief	QM2 & QM3	QM3_relief (位于 QM2)	QM3_relief.from.QM2

注:

1. z/OS 在 IBM MQ for z/OS 上，队列管理器名称限制为 4 个字符。
2. 对网络中的所有通道进行唯一命名。 如 第 18 页的表 1 中所示，在通道名称中包含源队列管理器名称和目标队列管理器名称是一种很好的方法。

网络策划员

创建网络假定有另一个更高级别的 网络规划员 功能，其计划由团队的其他成员实施。

对于广泛使用的应用程序，使用本地访问站点之间的宽带链路进行消息流量集中的本地访问站点方面的思考更为经济，如 第 20 页的图 2 所示。

在这个例子中，有两个主要系统和一些卫星系统。 实际配置将取决于业务注意事项。 有两个集中器队列管理器位于方便的中心。 每个 QM 集中器都具有到本地队列管理器的消息通道:

- QM 集中器 1 具有到三个本地队列管理器 (QM1， QM2 和 QM3) 中每一个的消息通道。 使用这些队列管理器的应用程序可以通过 QM 集中器相互通信。
- QM 集中器 2 具有到三个本地队列管理器 (QM4， QM5 和 QM6) 的消息通道。 使用这些队列管理器的应用程序可以通过 QM 集中器相互通信。
- QM 集中器之间有消息通道，因此允许队列管理器中的任何应用程序与另一个队列管理器中的任何其他应用程序交换消息。

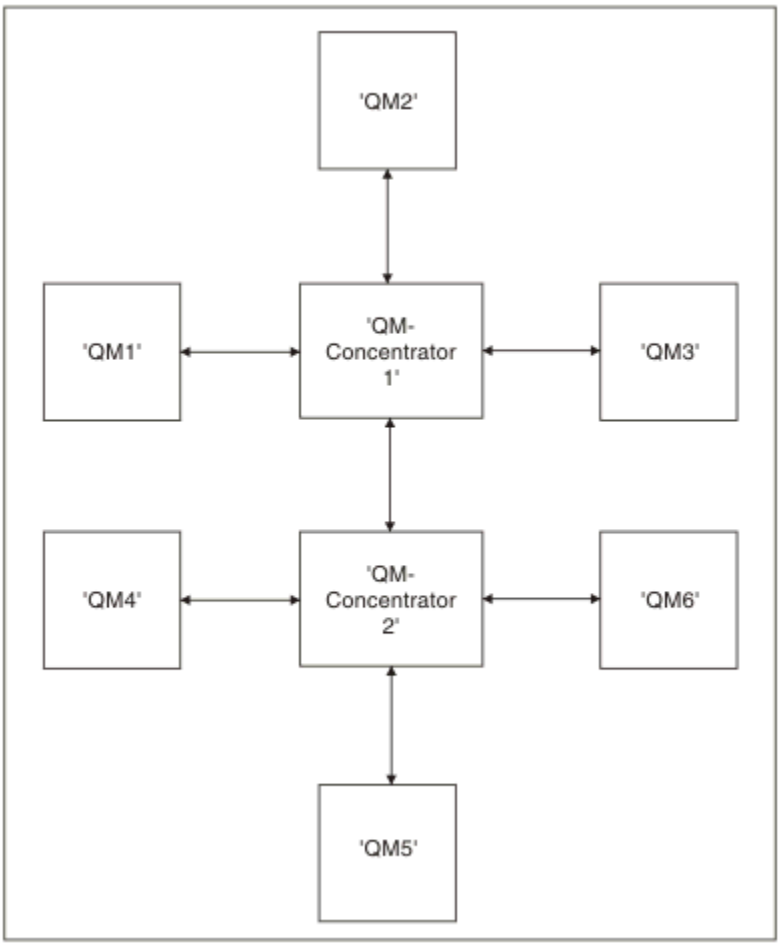


图 *2:* 显示 *QM* 集中器的网络图

设计集群

集群提供了一种用于以简化初始配置和持续管理的方式互连队列管理器的机制。 必须对集群进行精心设计，以确保它们正常运行，并确保它们达到所需的可用性和响应能力级别。

开始之前

有关集群概念的简介，请参阅以下主题:

- 分布式队列和集群
- 第 25 页的『集群与分布式排队的比较』
- 集群的组件

在设计队列管理器集群时，必须做出一些决策。 您必须首先决定集群中的哪些队列管理器将保存集群信息的完整存储库。 您创建的任何队列管理器都可以在集群中工作。 您可以为此目的选择任意数量的队列管理器，但理想数量为两个。 有关选择队列管理器以保存完整存储库的信息，请参阅 第 27 页的『如何选择集群队列管理器以保存完整存储库』。

请参阅以下主题以获取有关设计集群的更多信息:

- 第 32 页的『示例集群』
- 第 28 页的『组织集群』
- 第 28 页的『集群命名约定』
- z/OS 第 29 页的『Queue sharing groups and clusters』
- 第 30 页的『重叠集群』

下一步做什么

有关配置和使用集群的更多信息，请参阅以下主题:

- 在集群中建立通信
- 配置队列管理器集群
- 将消息路由到集群以及从集群路由消息
- 使用集群进行工作负载管理

有关帮助您配置集群的更多信息，请参阅第 30 页的『集群提示』。

规划如何使用多个集群传输队列

您可以显式定义传输队列，或者让系统为您生成传输队列。 如果您自己定义传输队列，那么可以对队列定义进行更多控制。 z/OS 在 z/OS 上，您还可以更多地控制保存消息的页集。

定义传输队列

定义传输队列有两种方法:

- 自动使用队列管理器属性 DEFCLXQ ，如下所示:

  ```
  ALTER QMGR DEFCLXQ(SCTQ | CHANNEL)
  ```

  DEFCLXQ (SCTQ) 指示所有集群发送方通道的缺省传输队列为 SYSTEM.CLUSTER.TRANSMIT.QUEUE。 这是缺省值。

  DEFCLXQ (CHANNEL) 指示缺省情况下，每个集群发送方通道都使用名为 SYSTEM.CLUSTER.TRANSMIT.*channel name*。 每个传输队列由队列管理器自动定义。 有关更多信息，请参阅第 22 页的『自动定义的集群传输队列』。
- 通过使用为 CLCHNAME 属性指定的值来手动定义传输队列。 CLCHNAME 属性指示哪些集群发送方通道应使用传输队列。 z/OS 如果要在 z/OS 上手动定义传输队列，请参阅第 23 页的『规划手动定义的集群传输队列』以获取更多信息。

我需要什么安全?

要自动或手动启动交换机，您需要启动通道的权限。

要定义用作传输队列的队列，您需要标准 IBM MQ 权限来定义队列。

什么时候是实施变革的合适时机?

在更改集群发送方通道所使用的传输队列时，您需要分配进行更新的时间，请考虑以下几点:

- 通道切换传输队列所需的时间取决于旧传输队列上的消息总数，需要移动的消息数以及消息大小。
- 应用程序可以在发生更改时继续将消息放入传输队列。 这可能导致过渡时间增加。
- 您可以随时更改任何传输队列或 DEFCLXQ 的 CLCHNAME 参数，最好是在工作负载较低时。

  请注意，不会立即发生任何情况。
- 仅当通道启动或重新启动时才会发生更改。 当通道启动时，它将检查当前配置并在需要时切换到新的传输队列。
- 有几个更改可能会改变集群发送方通道与传输队列的关联:
  - 更改传输队列的 CLCHNAME 属性的值，使 CLCHNAME 不那么具体或为空。
  - 更改传输队列的 CLCHNAME 属性的值，使 CLCHNAME 更具体。
  - 正在删除指定了 CLCHNAME 的队列。
  - 正在变更队列管理器属性 DEFCLXQ。

交换机需要多长时间?

在过渡期间，通道的任何消息都将从一个传输队列移至另一个传输队列。 通道切换传输队列所需的时间取决于旧传输队列上的消息总数以及需要移动的消息数。

对于包含几千条消息的队列，移动消息应该需要不到一秒时间。 实际时间取决于消息的数量和大小。 您的队列管理器应该能够以每秒兆字节数移动消息。

应用程序可以在发生更改时继续将消息放入传输队列。 这可能导致过渡时间增加。

必须重新启动每个受影响的集群发送方通道才能使更改生效。 因此，最好在队列管理器不忙时更改传输队列配置，很少消息存储在集群传输队列上。

runswchl 命令 z/OS 或 z/OS 上的 CSQUTIL 中的 SWITCH CHANNEL (*) STATUS 命令 可用于查询集群发送方通道的状态以及对其传输队列配置未完成的暂挂更改。

如何实施更改

请参阅 使用多个集群传输队列实现系统，以获取有关如何自动或手动更改多个集群传输队列的详细信息。

正在撤销更改

z/OS

请参阅 撤销对 z/OS 上传输队列的更改，以获取有关在迁到问题时如何回退更改的详细信息。

自动定义的集群传输队列
您可以让系统为您生成传输队列。

开始之前

z/OS 要在 z/OS 上手动设置集群传输队列，请参阅 第 23 页的『规划手动定义的集群传输队列』。

关于此任务

如果通道没有与之关联的手动定义的集群传输队列，并且您指定 DEFCLXQ (CHANNEL)，那么当通道启动时，队列管理器会自动为集群发送方通道定义永久动态队列。 模型队列 SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE 用于自动定义名为 SYSTEM.CLUSTER.TRANSMIT.ChannelName。

要点: z/OS 在 IBM MQ 8.0 中，队列管理器没有 SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE。不能直接从 IBM MQ 8.0 迁移到此版本。 有关将 SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE 添加到从 IBM MQ 8.0 迁移的队列管理器的信息，请参阅用于迁移队列管理器的临时版本的文档中的此主题。

过程

1. 使用 *DEFCLXQ* 队列管理器属性。
   有关此属性的更多信息，请参阅 ALTER QMGR。

   有两个选项:

   Sctq
   此选项是缺省值，表示您使用单个 SYSTEM.CLUSTER.TRANSMIT.QUEUE。

   通道
   表示您使用多个集群传输队列。
2. 要切换到新关联:
   - 停止并重新启动通道。
   - 通道使用新的传输队列定义。
   - 通过过渡切换进程将消息从旧队列传输到新的传输队列。

请注意，任何应用程序消息都将放入旧定义中。

当旧队列上的消息数达到零时，新消息将直接放在新传输队列上。

3. 要监视切换过程何时完成:
   a) 由通道启动的传输队列切换在后台运行，管理员可以监视队列管理器作业日志以确定它何时完成。
   b) 监视作业记录上的消息以显示交换机的进度。
   c) 要确保仅希望使用此传输队列的通道，请发出命令 DIS CLUSQMGR (*) ，其中，例如，定义传输队列的传输队列属性为 APPQMGR.CLUSTER1.XMITQ。
   d) z/OS
   在 CSQUTIL 下使用 SWITCH CHANNEL (*) STATUS 命令。
   此选项告诉您哪些暂挂更改未完成，以及需要在传输队列之间移动多少条消息。

结果

您已设置集群传输队列。

相关任务

第 23 页的『规划手动定义的集群传输队列』
在 IBM MQ for z/OS，如果您自己定义传输队列，您就可以更好地控制定义以及保存消息的页面集。

相关参考

ALTER QMGR
DISPLAY CLUSQMGR

z/OS 规划手动定义的集群传输队列

在 IBM MQ for z/OS，如果您自己定义传输队列，您就可以更好地控制定义以及保存消息的页面集。

开始之前

要自动设置集群传输队列，请参阅第 22 页的『自动定义的集群传输队列』。

关于此任务

您的管理员手动定义传输队列并使用队列属性 CLCHNAME 来定义哪个或哪些集群发送方通道将使用此队列作为其传输队列。

请注意， CLCHNAME 可以在开头或结尾包含通配符，以允许将单个队列用于多个通道。

过程

1. 例如，输入以下内容:

```
DEFINE QLOCAL(APPQMGR.CLUSTER1.XMITQ)
CLCHNAME(CLUSTER1.TO.APPQMGR)
USAGE(XMITQ) STGCLASS(STG1)
INDXTYPE( CORRELID ) SHARE

DEFINE STGCLASS(STG1) PSID(3)
DEFINE PSID(3) BUFFERPOOL(4)
```

   提示: 您需要规划用于传输队列的页集 (和缓冲池)。 您可以为不同的队列设置不同的页面集，并在它们之间提供隔离，因此一个页面集填满不会影响其他页面集中的传输队列。

   请参阅 使用集群传输队列和集群发送方通道 ，以获取有关每个通道如何选择相应队列的信息。

   当通道启动时，它会将其关联切换到新的传输队列。 为了确保不丢失消息，队列管理器会自动按顺序将消息从旧的集群传输队列传输到新的传输队列。

2. 使用 CSQUTIL SWITCH 函数来更改为新的关联。
   请参阅 切换与集群发送方通道 (SWITCH) 关联的传输队列 以获取更多信息。

a) 停止要更改其传输队列的通道，以使其处于 STOPPED 状态。
例如：

```
STOP CHANNEL(CLUSTER1.TO.APPQMGR)
```

b) 更改传输队列上的 CLCHNAME(XXXX) 属性。

c) 使用 SWITCH 函数来切换消息或监视正在发生的情况。
使用命令

```
SWITCH CHANNEL(*) MOVEMSGS(YES)
```

以在不启动通道的情况下移动消息。

d) 启动一个或多个通道，并检查该通道是否正在使用正确的队列。
例如：

```
DIS CHS(CLUSTER1.TO.APPQMGR)
DIS CHS(*) where(XMITQ eq APPQMGR.CLUSTER1.XMITQ)
```

提示: 以下过程使用 CSQUTIL SWITCH 函数。 有关详细信息，请参阅 For more information, see 切换与集群发送方通道关联的传输队列（SWITCH） 。

您不必使用此函数，但使用此函数会提供更多选项:

- 使用 SWITCH CHANNEL (*) STATUS 可轻松识别集群发送方通道的切换状态。 它允许管理员查看当前正在切换的通道，以及那些具有暂挂切换的通道在这些通道下次启动时生效。

  如果没有此功能，那么管理员需要使用多个 DISPLAY 命令，然后处理生成的输出以确定此信息。 管理员还可以确认配置更改是否具有必需的结果。
- 如果使用 CSQUTIL 来启动交换机，那么 CSQUTIL 将继续监视此操作的进度，并且仅在交换机完成时结束。

  这样可以更轻松地批量执行这些操作。 此外，如果运行 CSQUTIL 以切换多个通道，那么 CSQUTIL 会按顺序执行这些操作; 与并行运行的多个交换机相比，这对您的企业的影响较小。

结果

您已在上设置集群传输队列（或多个队列）z/OS 。

访问控制和多个集群传输队列

在应用程序将消息放入远程集群队列时选择三种检查方式。 这些方式是针对集群队列进行远程检查，针对 SYSTEM.CLUSTER.TRANSMIT.QUEUE 进行本地检查，或者针对集群队列或集群队列管理器的本地概要文件进行检查。

IBM MQ 允许您选择在本地或本地和远程检查用户是否有权将消息放入远程队列。 典型的 IBM MQ 应用程序仅使用本地检查，并且依赖于远程队列管理器信任对本地队列管理器进行的访问检查。 如果未使用远程检查，那么将使用远程消息通道进程的权限将消息放入目标队列。 要使用远程检查，必须将接收通道的放置权限设置为上下文安全性。

将针对应用程序打开的队列进行本地检查。 在分布式排队中，应用程序通常会打开远程队列定义，并且会针对远程队列定义进行访问检查。 如果将消息与完整路由头放在一起，那么将针对传输队列进行检查。 如果应用程序打开不在本地队列管理器上的集群队列，那么没有要检查的本地对象。 将针对集群传输队列 SYSTEM.CLUSTER.TRANSMIT.QUEUE 执行访问控制检查。 即使有多个集群传输队列，也会针对 SYSTEM.CLUSTER.TRANSMIT.QUEUE 对远程集群队列进行本地访问控制检查。

选择本地或远程检查是两个极端之间的选择。 远程检查是细颗粒度的。 每个用户都必须在集群中的每个队列管理器上具有访问控制概要文件才能放入任何集群队列。 本地检查是粗粒度的。 每个用户在其连接到的队列管理器上只需要一个集群传输队列的访问控制概要文件。 通过该概要文件，他们可以将消息放入任何集群中任何队列管理器上的任何集群队列。

管理员还有另一种方法来设置集群队列的访问控制。 您可以使用 **setmqaut** 命令在集群中的任何队列管理器上为集群队列创建安全概要文件。 如果在本地打开远程集群队列 (仅指定队列名称)，那么概要文件将生效。 您还可以为远程队列管理器设置概要文件。 如果执行此操作，那么队列管理器可以通过提供标准名称来检查打开集群队列的用户的概要文件。

仅当将队列管理器节 **ClusterQueueAccessControl** 更改为 RQMName 时，新概要文件才有效。 缺省值为 Xmitq。 您必须为现有应用程序使用集群队列的所有集群队列创建概要文件。 如果将节更改为 RQMName 而不创建概要文件，那么应用程序可能会失败。

提示: 集群队列访问检查不适用于远程排队。 仍会针对本地定义进行访问检查。 这些更改意味着您可以遵循相同的方法来配置对集群队列和集群主题的访问检查。 z/OS 这些更改还使集群队列的访问检查方法与 z/OS 更紧密地对齐。 用于在 z/OS 上设置访问权检查的命令有所不同，但这两个命令都检查对概要文件的访问权，而不是对对象本身的访问权。

相关概念

第 39 页的『集群: 使用多个集群传输队列进行应用程序隔离』
您可以在集群中的队列管理器之间隔离消息流。 您可以将由不同集群发送方通道传输的消息放入不同的集群传输队列中。 您可以在单个集群中使用此方法，也可以将此方法与重叠的集群配合使用。 本主题提供了一些示例和一些最佳实践，以指导您选择使用方法。

相关任务

设置 ClusterQueueAccessControl

集群与分布式排队的比较

比较需要定义的组件，以使用分布式排队和集群来连接队列管理器。

如果不使用集群，那么队列管理器是独立的，并使用分布式排队进行通信。 如果一个队列管理器需要将消息发送到另一个队列管理器，那么必须定义:

- 传输队列
- 到远程队列管理器的通道

第 25 页的图 3 显示了分布式排队所需的组件。

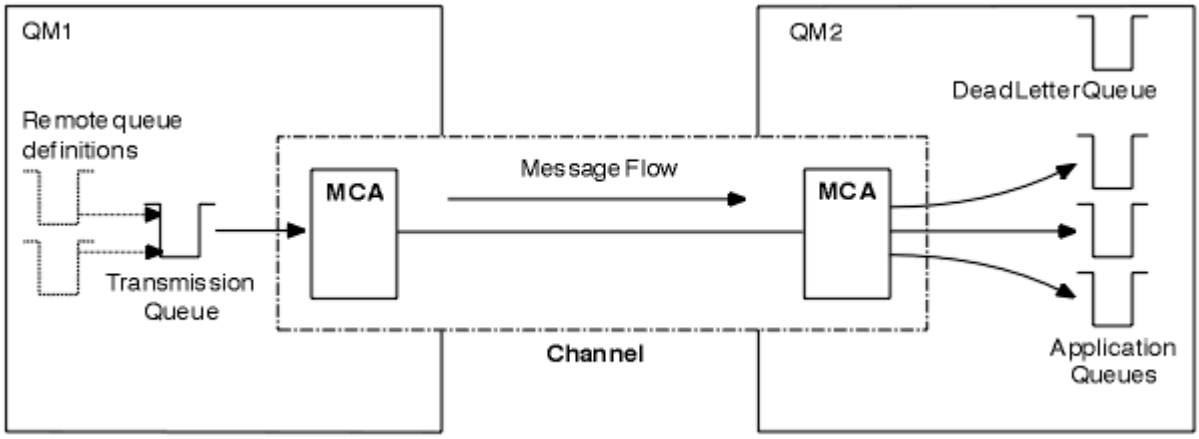


图 *3:* 分布式 排队

如果对集群中的队列管理器进行分组，那么任何队列管理器上的队列都可供集群中的任何其他队列管理器使用。 任何队列管理器都可以在没有显式定义的情况下向同一集群中的任何其他队列管理器发送消息。 您不会为每个目标提供通道定义，远程队列定义或传输队列。 集群中的每个队列管理器都有一个传输队列，可以从中将消息传输到集群中的任何其他队列管理器。 集群中的每个队列管理器仅需要定义:

- 要在其上接收消息的一个集群接收方通道
- 一个集群发送方通道，通过该通道引入自身并了解集群

用于设置集群与分布式排队的定义

查看 第 26 页的图 4，其中显示了四个队列管理器，每个队列管理器具有两个队列。 请考虑使用分布式排队连接这些队列管理器所需的定义数。 比较将同一网络设置为集群所需的定义数。

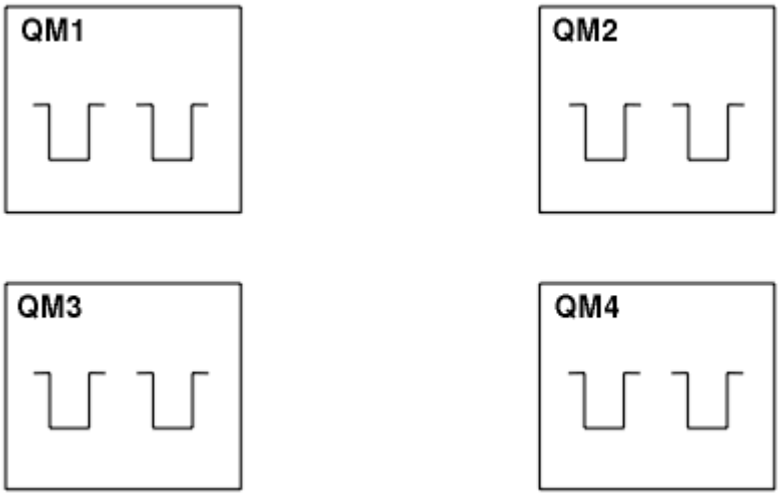


图 *4:* 由四个队列管理器组成的网络

用于使用分布式排队来设置网络的定义

要使用分布式排队来设置 第 25 页的图 3 中显示的网络，您可能具有以下定义:

表 *2:* 分布式排队的定义

描述	每个队列管理器的数目	总数
要将消息发送到其他每个队列管理器的通道的发送方通道定义	3	12
要在其上从每个其他队列管理器接收消息的通道的接收方通道定义	3	12
传输队列到每个其他队列管理器的传输队列定义	3	12
每个本地队列的本地队列定义	2	8
此队列管理器要将消息放入的每个远程队列的远程队列定义	6	24

您可以使用通用接收方通道定义来减少此数目的定义。 每个队列管理器上的最大定义数可能多达 17 个，此网络的定义总数为 68 个。

用于使用集群设置网络的定义

要使用集群设置 第 25 页的图 3 中显示的网络，您需要以下定义:

表 *3:* 集群的定义

描述	每个队列管理器的数目	总数
要将消息发送到存储库队列管理器的通道的集群发送方通道定义	1	4
要在其上从集群中的其他队列管理器接收消息的通道的集群接收方通道定义	1	4
每个本地队列的本地队列定义	2	8

要设置此队列管理器集群 (具有两个完整存储库) ，每个队列管理器上需要四个定义，总共需要十六个定义。 您还需要更改两个队列管理器的队列管理器定义，以使它们成为集群的完整存储库队列管理器。

仅需要一个 CLUSSDR 和一个 CLUSRCVR 通道定义。 定义集群时，您可以添加或除去队列管理器 (存储库队列管理器除外)，而不会对其他队列管理器造成任何干扰。

使用集群可减少设置包含许多队列管理器的网络所需的定义数。

定义较少，因此发生错误的风险较低:

- 对象名始终匹配，例如发送方/接收方对中的通道名称。
- 通道定义中指定的传输队列名称始终与正确的传输队列定义或远程队列定义中指定的传输队列名称相匹配。
- QREMOTE 定义始终指向远程队列管理器上的正确队列。

设置集群后，您可以将集群队列从一个队列管理器移至集群中的另一个队列管理器，而不必对任何其他队列管理器执行任何系统管理工作。 没有可能忘记删除或修改通道，远程队列或传输队列定义。 您可以将新的队列管理器添加到集群中，而不会对现有网络造成任何干扰。

如何选择集群队列管理器以保存完整存储库

在每个集群中，必须至少选择一个队列管理器，最好选择两个队列管理器以保存完整存储库。 除了最例外的情况外，两个完整的存储库足以应对所有情况。 如果可能，请选择在稳健且永久连接的平台上托管的队列管理器，这些队列管理器不会发生重合的中断，并且在地理位置上处于中心位置。 另外，请考虑将系统专用为完整存储库主机，而不要将这些系统用于任何其他任务。

完整存储库 是保持集群状态完整的队列管理器。 要共享此信息，每个完整存储库都由 CLUSSDR 通道 (及其相应的 CLUSRCVR 定义) 连接到集群中的每个其他完整存储库。 您必须手动定义这些通道。

图 *5:* 两个已连接的完整存储库。

集群中的每个其他队列管理器都会在 部分存储库中维护其当前了解的集群状态的图片。 这些队列管理器使用任何两个可用的完整存储库发布有关自身的信息，并请求有关其他队列管理器的信息。 如果所选完整存储库不可用，那么将使用另一个完整存储库。 当所选的完整存储库再次变为可用时，它将从其他存储库收集最新的新信息和已更改的信息，以便它们保持同步。 如果所有完整存储库都不起作用，那么其他队列管理器将使用其部分存储库中的信息。 但是，它们仅限于使用它们所拥有的信息; 无法处理新信息和更新请求。 当完整存储库重新连接到网络时，将交换消息以更新所有存储库 (包括完整存储库和部分存储库)。

规划完整存储库的分配时，请包括以下注意事项:

- 选择用于保存完整存储库的队列管理器需要可靠且受管。 选择在健壮且永久连接的平台上托管的队列管理器。
- 请考虑托管完整存储库的系统的计划中断，并确保这些系统不会发生重合的中断。
- 考虑网络性能: 选择在地理位置上处于中心位置的队列管理器，或者选择与集群中的其他队列管理器共享同一系统的队列管理器。
- 请考虑队列管理器是否是多个集群的成员。 使用同一队列管理器来托管多个集群的完整存储库在管理上可能很方便，前提是此优势与您期望队列管理器的繁忙程度相平衡。
- 请考虑将某些系统专用于仅包含完整存储库，而不将这些系统用于任何其他任务。 这将确保这些系统只需要维护队列管理器配置，并且不会从维护其他业务应用程序的服务中除去。 它还确保维护存储库的任务不会与系统资源的应用程序竞争。 这在大型集群 (例如，包含超过 1000 个队列管理器的集群) 中特别有用，在这些集群中，完整存储库在维护集群状态方面的工作负载要高得多。

可能有两个以上完整存储库，但很少建议使用。 虽然对象定义 (即，队列，主题和通道) 流向所有可用的完整存储库，但请求仅从部分存储库流向最多两个完整存储库。 这意味着当定义了两个以上的完整存储库，并且任何两个完整存储库变为不可用时，某些部分存储库可能不会收到它们期望的更新。 请参阅 MQ 集群: 为什么只有两个完整存储库?

您可能发现定义两个以上完整存储库很有用的一种情况是在将现有完整存储库迁移到新硬件或新队列管理器时。 在这种情况下，您应该先引入替换完整存储库，并确认它们已完全填充，然后再移除先前的完整存储库。 每当添加完整存储库时，请记住必须使用 CLUSSDR 通道将其直接连接到其他所有完整存储库。

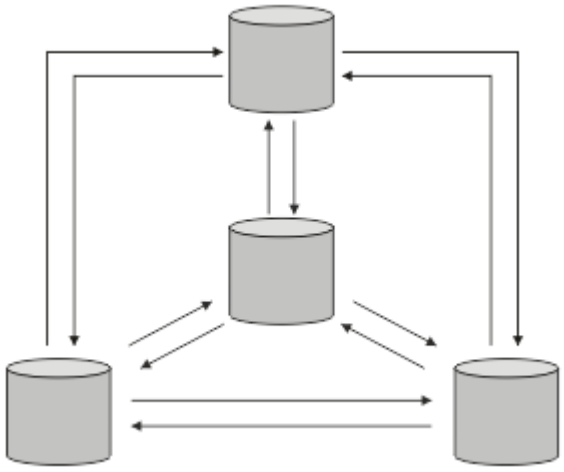


图 *6:* 两个以上已连接的完整存储库

相关信息

MQ 集群: 为什么只有两个完整存储库?

MQ 集群有多大?

组织集群

选择要链接到哪个完整存储库的队列管理器。 请考虑性能影响，队列管理器版本以及是否需要多个 CLUSSDR 通道。

在选择队列管理器以保存完整存储库之后，您需要决定要链接到哪个完整存储库的队列管理器。 CLUSSDR 通道定义将队列管理器链接到完整存储库，从中可以发现集群中的其他完整存储库。 从此，队列管理器将消息发送到任意两个完整存储库。 它始终首先尝试使用其具有 CLUSSDR 通道定义的通道。 您可以选择将队列管理器链接到任一完整存储库。 在选择时，请考虑配置的拓扑以及队列管理器的物理或地理位置。

由于所有集群信息都将发送到两个完整存储库，因此可能存在您想要生成第二个 CLUSSDR 通道定义的情况。 您可以在集群中定义第二个 CLUSSDR 通道，该集群具有分布在广泛区域中的许多完整存储库。 然后，您可以控制将信息发送到哪个两个完整存储库。

集群命名约定

请考虑使用用于标识队列管理器所属集群的命名约定来命名同一集群中的队列管理器。 对通道名称使用类似的命名约定，并对其进行扩展以描述通道特征。

命名 MQ 集群时的最佳实践

虽然集群名称最多可以有 48 个字符，但在对其他对象应用命名约定时，相对较短的集群名称很有用。 请参阅 第 29 页的『选择集群通道名称时的最佳实践』。

在选择集群名称时，通常有助于表示集群的 "用途" (可能是长期存在) 而不是 "内容"。 例如， "B2BPROD" 或 "ACTTEST" 而不是 "QM1_QM2_QM3_CLUS"。

选择集群队列管理器名称时的最佳实践

如果要从头开始创建新集群及其成员，请考虑队列管理器的命名约定，以反映其集群使用情况。 每个队列管理器都必须具有不同的名称。 但是，您可以为集群中的队列管理器提供一组类似的名称，以帮助识别和记住逻辑分组 (例如， "ACTTQM1， ACTTQM2")。

相对较短的队列管理器名称 (例如，少于 8 个字符) 可帮助您选择对通道名称使用下一节中描述的约定或类似的约定。

选择集群通道名称时的最佳实践

由于队列管理器和集群的名称最多可包含 48 个字符，并且通道名称限制为 20 个字符，因此在首次命名对象时请注意避免在项目中途更改命名约定 (请参阅上一部分)。

定义通道时，请记住，在集群中的任何队列管理器上自动创建的集群发送方通道会从集群中的接收队列管理器上配置的相应集群接收方通道中获取其名称，因此这些通道必须是唯一的，并且 在集群中的远程队列管理器上 有意义。

一种常见方法是使用以集群名称开头的队列管理器名称。 例如，如果集群名称为 CLUSTER1 ，队列管理器为 QM1， QM2，那么集群接收方通道为 CLUSTER1.QM1， CLUSTER1.QM2。

如果通道具有不同的优先级或使用不同的协议，那么可以扩展此约定。 例如：

- CLUSTER1.QM1.S1
- CLUSTER1.QM1.N3
- CLUSTER1.QM1.T4

在此示例中， S1 可能是第一个 SNA 通道， N3 可能是网络优先级为 3 的 NetBIOS 通道， T4 可能是使用 IPV4 网络的 TCP IP。

命名共享通道定义

可以跨多个集群共享单个通道定义，在这种情况下，此处建议的命名约定需要修改。 但是，如 管理通道定义 中所述，在任何情况下，通常最好为每个集群定义离散通道。

较旧的通道命名约定

在集群环境之外，使用 "FROMQM.TO.TARGETQM" 命名约定在历史上是常见的，因此您可能会发现现有集群使用了类似的内容 (例如 CLUSTER.TO.TARGET)。 建议不要将此作为新集群命名方案的一部分，因为这将进一步减少可用字符以在通道名称中传递 "有用" 信息。

z/OS IBM MQ for z/OS 上的通道名称

您可以定义 VTAM 通用资源或 *Dynamic Domain Name Server* (DDNS) 通用名称。 您可以使用通用名称来定义连接名称。 但是，创建集群接收方定义时，请勿使用通用连接名称。

对集群接收方定义使用通用连接名称的问题如下: 如果使用通用 CONNAME 定义 CLUSRCVR ，那么不保证 CLUSSDR 通道指向您期望的队列管理器。 初始 CLUSSDR 可能最终指向队列共享组中的任何队列管理器，而不一定是托管完整存储库的队列管理器。 如果通道再次开始尝试连接，那么它可能会重新连接到具有相同通用名称的其他队列管理器，从而中断消息流。

z/OS *Queue sharing groups and clusters*

Shared queues can be cluster queues and queue managers in a queue sharing group can also be cluster queue managers.

On IBM MQ for z/OS you can group queue managers into queue sharing groups. A queue manager in a queue sharing group can define a local queue that is to be shared by up to 32 queue managers.

Shared queues can also be cluster queues. Furthermore, the queue managers in a queue sharing group can also be in one or more clusters.

您可以定义 VTAM 通用资源或 *Dynamic Domain Name Server* (DDNS) 通用名称。 您可以使用通用名称来定义连接名称。 但是，创建集群接收方定义时，请勿使用通用连接名称。

对集群接收方定义使用通用连接名称的问题如下: 如果使用通用 CONNAME 定义 CLUSRCVR ，那么不保证 CLUSSDR 通道指向您期望的队列管理器。 初始 CLUSSDR 可能最终指向队列共享组中的任何队列管理器，而不一定是托管完整存储库的队列管理器。 如果通道再次开始尝试连接，那么它可能会重新连接到具有相同通用名称的其他队列管理器，从而中断消息流。

A CLUSRCVR channel that uses the group listener port can not be started because, if this were the case, it would not be possible to tell which queue manager the CLUSRCVR would connect to each time. The cluster system queues on which information is kept about the cluster are not shared. Each queue manager has its own.

Cluster channels are used not only to transfer application messages but internal system messages about the setup of the cluster. Each queue manager in the cluster must receive these internal system messages to participate properly in clustering, so needs its own unique CLUSRCVR channel on which to receive them.

A shared CLUSRCVR could start on any queue manager in the queue sharing group (QSG) and so lead to an inconsistent supply of the internal system messages to the QSG queue managers, meaning none can properly participate in the cluster. To ensure no shared CLUSRCVR channels can be used, any attempt fails with the CSQX502E message.

重叠集群

重叠的集群提供了额外的管理功能。 使用名称列表来减少管理重叠集群所需的命令数。

您可以创建重叠的集群。 您可以定义重叠集群的原因有很多; 例如:

- 让不同的组织有自己的管理。
- 允许单独管理独立应用程序。
- 创建服务类。

在 第 30 页的图 7 中，队列管理器 STF2 是这两个集群的成员。 当队列管理器是多个集群的成员时，您可以利用名称列表来减少所需的定义数。 名称列表包含名称列表，例如，集群名称。 您可以创建命名集群的名称列表。 在 STF2 的 ALTER QMGR 命令上指定名称列表，以使其成为两个集群的完整存储库队列管理器。

如果网络中有多个集群，那么必须为它们提供不同的名称。 如果两个具有相同名称的集群曾经合并，那么无法再次将它们分开。 给集群和通道不同的名称也是一个好主意。 当您查看 DISPLAY 命令的输出时，可以更轻松地对它们进行区分。 队列管理器名称在集群中必须是唯一的，才能使其正常工作。

定义服务类

想象一下，有一个大学，每个工作人员和每个学生都有一个队列管理器。 工作人员之间的消息是在具有高优先级和高带宽的通道上传输。 学生之间的信息是在更便宜，更慢的渠道上旅行。 您可以使用传统的分布式排队技术来设置此网络。 IBM MQ 通过查看目标队列名称和队列管理器名称来选择要使用的通道。

要明确区分职员和学生，您可以将他们的队列管理器分组到两个集群中，如 第 30 页的图 7 中所示。 IBM MQ 仅通过人员集群中定义的通道将消息移至该集群中的会议队列。 学生集群中 gossip 队列的消息将通过该集群中定义的通道，并接收相应的服务类。

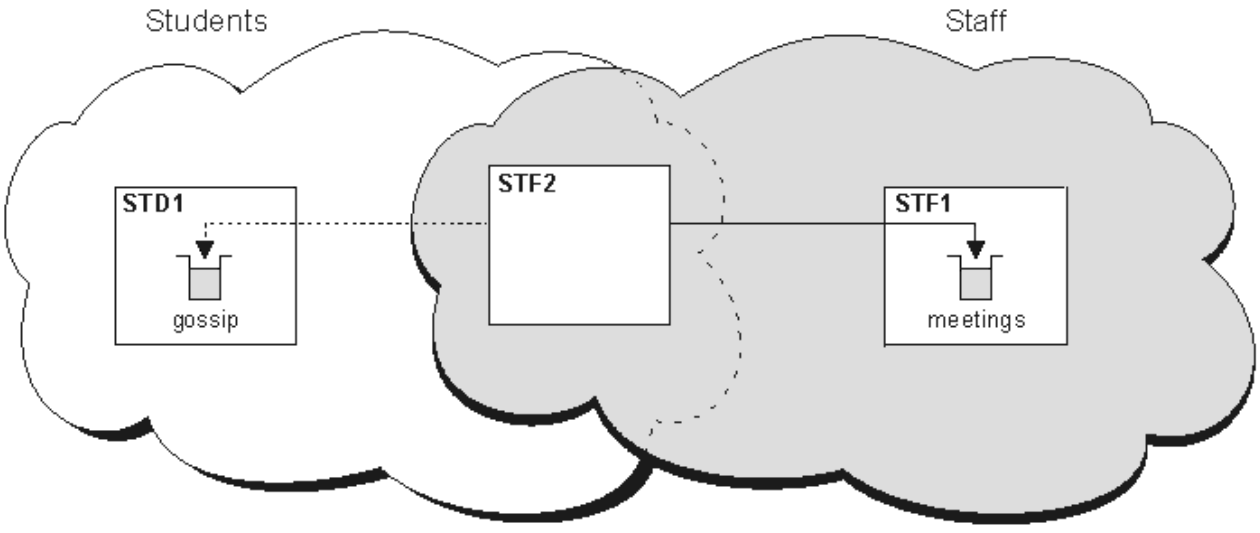


图 7: 服务等级

集群提示

在使用集群之前，您可能需要对系统或应用程序进行一些更改。 与分布式排队的行为既有相似之处，也有不同之处。

- 必须将手动配置定义添加到集群外部的队列管理器，以便它们访问集群队列。
- 如果合并具有相同名称的两个集群，那么不能再次将其分隔开。 因此，建议为所有集群提供唯一名称。
- 如果消息到达队列管理器，但那里没有要接收该消息的队列，那么会将该消息放在死信队列上。 如果没有死信队列，那么通道将失败并重试。 死信队列的使用与分布式排队相同。
- 维护持久消息的完整性。 消息不会因使用集群而重复或丢失。
- 使用集群可减少系统管理。 集群使您能够轻松地将更大的网络与更多的队列管理器连接起来，而不需要您考虑使用分布式排队。 如果尝试在集群中的每个队列管理器之间启用通信，那么可能会消耗过多的网络资源。
- 如果使用以树结构显示队列管理器的 IBM MQ Explorer，那么大型集群的视图可能很繁琐。
- Multi 分发列表的目的是使用单个 MQPUT 命令将同一消息发送到多个目标。 IBM MQ for Multiplatforms 上支持分发列表。 您可以将分发列表与队列管理器集群配合使用。 在集群中，所有消息在 MQPUT 时展开。 在网络流量方面的优势，并不像在非集群环境中那么大。 分发列表的优点是不需要手动定义众多通道和传输队列。
- 如果要使用集群来平衡工作负载，请检查应用程序。 查看它们是要求消息由特定队列管理器处理还是按特定顺序处理。 此类应用程序据说具有消息亲缘关系。 您可能需要先修改应用程序，然后才能在复杂集群中使用这些应用程序。
- 您可以选择在 MQOPEN 上使用 MQOO_BIND_ON_OPEN 选项，以强制将消息发送到特定目标。 如果目标队列管理器不可用，那么直到队列管理器再次可用之后才会传递消息。 由于存在重复的风险，因此不会将消息路由到另一个队列管理器。
- 如果队列管理器要托管集群存储库，那么您需要知道其主机名或 IP 地址。 在加入集群的其他队列管理器上创建 CLUSSDR 定义时，必须在 CONNAME 参数中指定此信息。 如果使用 DHCP，那么 IP 地址可能会更改，因为 DHCP 可以在每次重新启动系统时分配新的 IP 地址。 因此，不得在 CLUSSDR 定义中指定 IP 地址。 即使所有 CLUSSDR 定义都指定了主机名而不是 IP 地址，这些定义仍然不可靠。 DHCP 不一定使用新地址更新主机的 DNS 目录条目。 如果必须将队列管理器指定为使用 DHCP 的系统上的完整存储库，请安装保证 DNS 目录保持最新的软件。
- 请勿使用通用名称，例如 VTAM 通用资源或动态域名服务器 (DDNS) 通用名称作为通道的连接名称。 如果您这样做，那么您的通道可能会连接到与预期不同的队列管理器。
- 只能从本地集群队列获取消息，但可以将消息放入集群中的任何队列。 如果打开队列以使用 MQGET 命令，那么队列管理器将打开本地队列。
- 如果设置了简单 IBM MQ 集群，那么不需要更改任何应用程序。 应用程序可以在 MQOPEN 调用上命名目标队列，并且不需要了解队列管理器的位置。 如果为工作负载管理设置集群，那么必须复审应用程序并根据需要对其进行修改。
- 您可以使用 DISPLAY CHSTATUS 和 DISPLAY QSTATUS **runmqsc** 命令来查看通道或队列的当前监视和状态数据。 监控信息可用于帮助评估系统的性能和运行状况。 监视由队列管理器，队列和通道属性控制。 可以使用 MONACLS 队列管理器属性来监视自动定义的集群发送方通道。

相关概念

集群

第 25 页的『集群与分布式排队的比较』
比较需要定义的组件，以使用分布式排队和集群来连接队列管理器。

集群的组件

相关任务

配置队列管理器集群

设置新集群

队列管理器存储库保留信息的时间长度?

队列管理器存储库将信息保留 30 天。 自动进程高效地刷新正在使用的信息。

当队列管理器发出有关自身的一些信息时，完整和部分存储库队列管理器会将该信息存储 30 天。 例如，当队列管理器发布创建新队列的广告时，将发送信息。 为了防止此信息到期，队列管理器将在 27 天后自动重新发送有关自身的所有信息。 如果部分存储库在 30 天生存期内发送新的信息部分请求，那么到期时间仍为原来的 30 天。

当信息到期时，不会立即将其从存储库中除去。 而是举行 60 天的宽限期。 如果在宽限期内未收到任何更新，那么将除去该信息。 宽限期允许队列管理器可能在到期日期暂时停止服务。 如果队列管理器与集群断开连接超过 90 天，那么它将停止成为集群的一部分。 但是，如果它重新连接到网络，那么它将再次成为集群的一部分。 完整存储库不使用已到期的信息来满足来自其他队列管理器的新请求。

同样，当队列管理器从完整存储库发送对最新信息的请求时，该请求将持续 30 天。 27 天后， IBM MQ 将检查请求。 如果在 27 天内引用了该参数，那么将自动刷新该参数。 如果没有，那么它将保留为到期，并由队列管理器刷新 (如果再次需要)。 到期请求会阻止来自休眠队列管理器的信息请求的累积。

注: 您应该下载并安装 APAR PH43191 的 PTF ，这将修订计算预订到期时间时发生的系统错误。 这些错误可能导致预订提前到期 (导致发出消息 CSQX456I) ，或在对象到期后到期 (导致错误 MQRC 2085 (MQRC_UNKNOWN_OBJECT) 错误)。

对于大型集群，如果许多队列管理器同时自动重新发送有关其自身的所有信息，那么这可能具有破坏性。 请参阅在大型集群中刷新可能会影响集群的性能和可用性。

相关概念

第 56 页的『集群：使用 REFRESH CLUSTER 最佳实践』
使用 **REFRESH CLUSTER** 命令可废弃有关集群的所有本地保存的信息，并从集群中的完整存储库重建该信息。 您不应该使用此命令，除非在特殊情况下。 如果确实需要使用它，那么有关如何使用它有特殊注意事项。 此信息是基于测试和客户反馈的指南。

示例集群

第一个示例显示了两个队列管理器的最小可能集群。 第二个和第三个示例显示了三个队列管理器集群的两个版本。

最小的可能集群仅包含两个队列管理器。 在这种情况下，两个队列管理器都包含完整存储库。 您只需要几个定义即可设置集群，然而每个队列管理器都有高度的自主性。

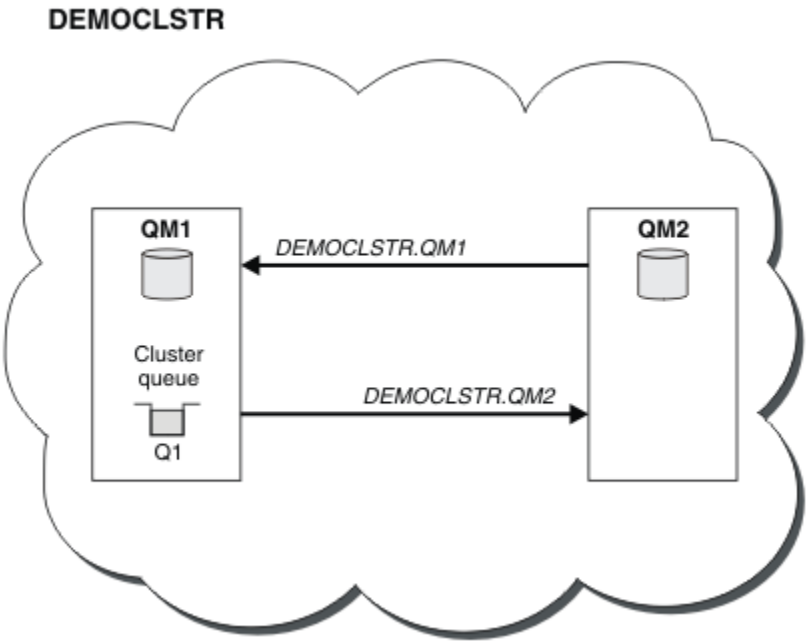


图 *8:* 两个队列管理器的小型集群

- 队列管理器可以具有长名称，例如 LONDON 和 NEWYORK。 z/OS 在 IBM MQ for z/OS 上，队列管理器名称限制为四个字符。
- 通常在单独的机器上配置每个队列管理器。 但是，您可以在同一机器上具有多个队列管理器。

有关设置类似示例集群的指示信息，请参阅 设置新集群。

第 33 页的图 9 显示了名为 CLSTR1 的集群的组件。

- 在此集群中，有三个队列管理器: QM1， QM2 和 QM3。
- 有关集群中所有队列管理器和集群相关对象的信息的 QM1 和 QM2 主机存储库。 它们称为 完整存储库队列管理器。 存储库在图中由阴影柱面表示。
- QM2 和 QM3 托管一些可供集群中任何其他队列管理器访问的队列。 可供集群中任何其他队列管理器访问的队列称为 集群队列。 在图中，集群队列由阴影队列表示。 可以从集群中的任何位置访问集群队列。 IBM MQ 集群代码确保在引用集群队列的任何队列管理器上创建集群队列的远程队列定义。

  与分布式排队一样，应用程序使用 MQPUT 调用将消息放在集群中任何队列管理器上的集群队列上。 应用程序使用 MQGET 调用仅在队列所在的队列管理器上从集群队列检索消息。
- 每个队列管理器都有一个手动创建的定义，用于接收名为 *cluster_name. queue_manager_name* 的通道的接收端，在该通道上可以接收消息。 在接收队列管理器上， *cluster_name. queue_manager_name* 是集群接收方通道。 集群接收方通道类似于分布式排队中使用的接收方通道; 它接收队列管理器的消息。 此外，它还接收有关集群的信息。
-

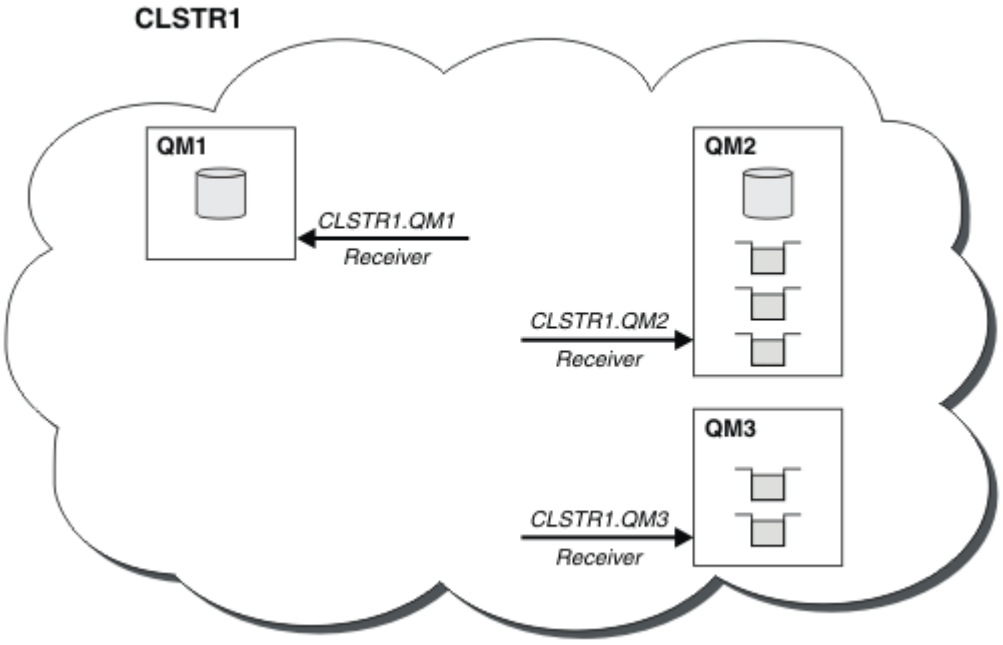


图 *9:* 队列管理器集群

- 在 第 34 页的图 10 中，每个队列管理器还具有通道发送端的定义。 它连接到其中一个完整存储库队列管理器的集群接收方通道。 在发送队列管理器上， *cluster_name. queue_manager_name* 是集群发送方通道。 QM1 和 QM3 将集群发送方通道连接到 CLSTR1.QM2，请参阅虚线 “2”。

  QM2 具有连接到 CLSTR1.QM1 的集群发送方通道，请参阅虚线 “3”。 集群发送方通道类似于分布式排队中使用的发送方通道; 它将消息发送到接收队列管理器。 此外，它还会发送有关集群的信息。

  一旦定义了通道的集群接收方端和集群发送方端，该通道将自动启动。

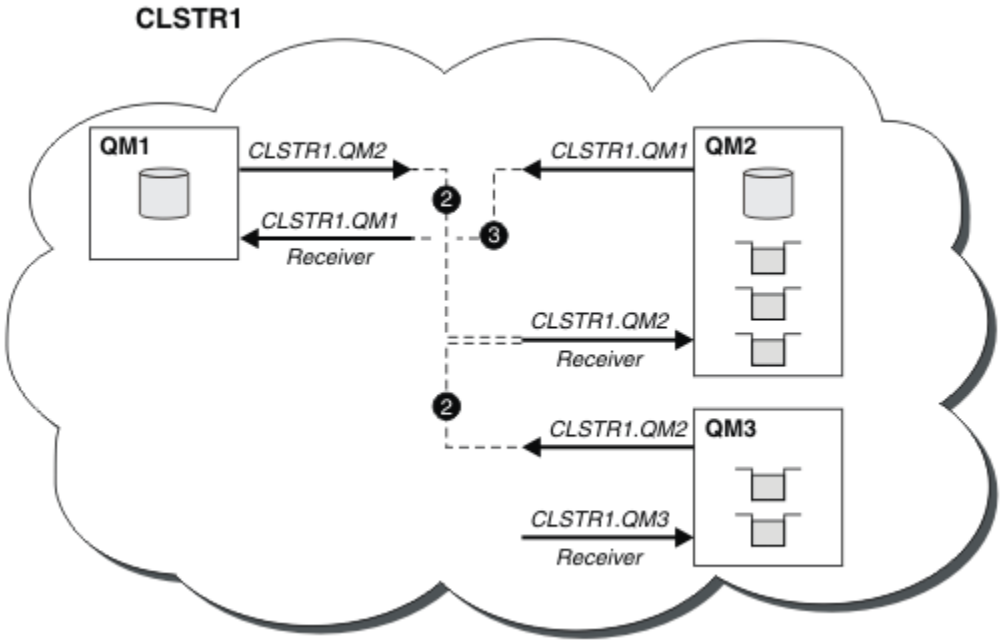


图 *10:* 具有发送方通道的队列管理器集群

在本地队列管理器上定义集群发送方通道会将该队列管理器引入到其中一个完整存储库队列管理器。 完整存储库队列管理器相应地更新其完整存储库中的信息。 然后，它会自动将集群发送方通道创建回原始队列管理器，并发送有关集群的队列管理器信息。 因此，队列管理器学习有关集群的信息，而集群学习有关队列管理器的信息。

再次查看 第 33 页的图 9。 假设连接到队列管理器 QM3 的应用程序想要将一些消息发送到位于 QM2 的队列。 QM3 首次必须访问这些队列时，它会通过查阅完整存储库来发现这些队列。 在此情况下，完整存储库为 QM2，可使用发送方通道 CLSTR1.QM2 进行访问。 通过存储库中的信息，它可以自动为这些队列创建远程定义。 如果队列位于 QM1 上，那么此机制仍有效，因为 QM2 是完整存储库。 完整存储库具有集群中所有对象的完整记录。 在后一种情况下， QM3 还将自动创建与 QM1 上的集群接收方通道对应的集群发送方通道，从而允许两者之间进行直接通信。

第 35 页的图 11 显示了同一集群，以及自动创建的两个集群发送方通道。 集群发送方通道由与集群接收方通道 CLSTR1.QM3 连接的两条虚线表示。 它还显示集群传输队列 SYSTEM.CLUSTER.TRANSMIT.QUEUE，QM1 使用该队列来发送其消息。 集群中的所有队列管理器都有一个集群传输队列，可以从该队列将消息发送到同一集群中的任何其他队列管理器。

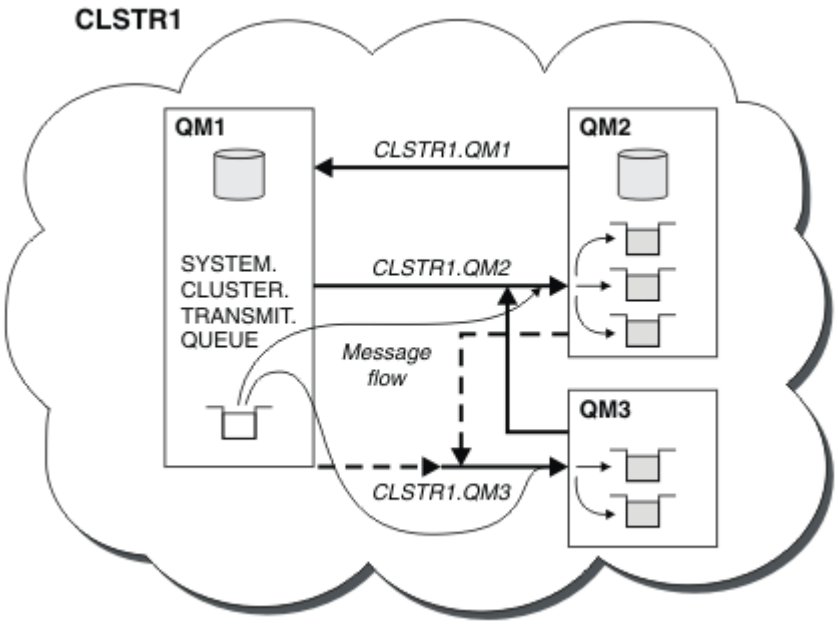


图 *11:* 队列管理器集群，显示自动定义的通道

注: 其他图仅显示您为其进行手动定义的通道的接收端。 由于发送端大多在需要时自动定义，因此省略了发送端。 大多数集群发送方通道的自动定义对于集群的功能和效率至关重要。

相关概念

第 25 页的『集群与分布式排队的比较』
比较需要定义的组件，以使用分布式排队和集群来连接队列管理器。

集群的组件

相关任务

配置队列管理器集群

设置新集群

集群: 最佳实践

集群提供了一种用于互连队列管理器的机制。 本节中描述的最佳实践基于客户的测试和反馈。

成功的集群设置依赖于良好的规划和对 IBM MQ 基础知识 (例如，良好的应用程序管理和网络设计) 的全面了解。 请确保您熟悉相关主题中的信息，然后再继续。

相关概念

分布式排队和集群

集群

相关任务

第 20 页的『设计集群』
集群提供了一种用于以简化初始配置和持续管理的方式互连队列管理器的机制。 必须对集群进行精心设计，以确保它们正常运行，并确保它们达到所需的可用性和响应能力级别。

监视集群

集群: 重叠集群的特殊注意事项

本主题提供有关规划和管理 IBM MQ 集群的指导。 此信息是基于测试和客户反馈的指南。

集群所有权

请先熟悉重叠的集群，然后再阅读以下信息。 请参阅 第 30 页的『重叠集群』 和 配置集群之间的消息路径 以获取必要的信息。

在配置和管理由重叠集群组成的系统时，最好遵循以下内容:

- 尽管如前所述， IBM MQ 集群 "松散耦合" ，但将集群视为单个管理单元很有用。 使用此概念是因为各个队列管理器上的定义之间的交互对于集群的顺利运行至关重要。 例如: 使用工作负载均衡的集群队列时，单个管理员或团队必须了解消息的完整可能目标集，这取决于分布在整个集群中的定义。 更重要的是，集群发送方/接收方通道对必须自始至终兼容。
- 考虑到先前的这一概念; 当多个集群满足 (将由单独的团队/个人管理) 时，重要的是要有明确的策略来控制网关队列管理器的管理。
- 将重叠集群视为单个名称空间很有用: 通道名称和队列管理器名称在单个集群中必须唯一。 在整个拓扑中唯一时，管理更容易。 最好遵循合适的命名约定， 第 28 页的『集群命名约定』 中描述了可能的约定。
- 有时，行政和系统管理合作至关重要。 例如，拥有需要重叠的不同集群的组织之间的合作。 清楚了解谁拥有什么以及可执行的规则和约定，有助于集群在重叠集群时顺利运行。

重叠集群: 网关

通常，单个集群比多个集群更易于管理。 因此，创建大量小型集群 (例如，针对每个应用程序创建一个集群) 通常是要避免的。

但是，要提供服务类，您可以实现重叠集群。 例如:

- 如果您有同心集群，其中较小的集群用于发布/预订。 请参阅 如何调整系统大小 以获取更多信息。
- 如果某些队列管理器将由不同的团队管理。 请参阅上一部分 第 36 页的『集群所有权』 以获取更多信息。
- 如果从组织或地理角度讲是有意义的。
- 如果等价集群使用名称解析，例如在现有集群中实现 TLS 时。

重叠集群没有安全优势; 允许由两个不同团队管理的集群重叠，有效加入团队以及拓扑:

- 在此类集群中公布的任何名称都可供其他集群访问。
- 在一个集群中公布的任何名称都可以在另一个集群中公布，以提取符合条件的消息。
- 可以从网关所属的任何集群中解析与网关相邻的队列管理器上的任何非广告对象。

名称空间是两个集群的并集，必须视为单个名称空间。 因此，重叠集群的所有权在两个集群的所有管理员之间共享。

当系统包含多个集群时，可能需要将消息从一个集群中的队列管理器路由到另一个集群中的队列管理器上的队列。 在此情况下，必须以某种方式互连多个集群: 要遵循的良好模式是在集群之间使用网关队列管理器。 这种安排避免了建立起一个难以管理的点对点通道网格，并为管理安全策略等问题提供了一个好的场所。 实现这一安排有两种不同的方法:

1. 使用第二个集群接收方定义在两个集群中放置一个 (或多个) 队列管理器。 这种安排涉及较少的管理定义，但如前所述，意味着重叠集群的所有权由两个集群的所有管理员共享。
2. 使用传统的点到点通道将集群 1 中的队列管理器与集群 2 中的队列管理器配对。

在任何一种情况下，都可以使用各种工具来适当路由流量。 特别是，可以使用队列或队列管理器别名来路由到其他集群，并且具有空白 **RQMNAME** 属性的队列管理器别名会在需要时重新驱动工作负载均衡。

相关概念

第 28 页的『集群命名约定』
请考虑使用用于标识队列管理器所属集群的命名约定来命名同一集群中的队列管理器。 对通道名称使用类似的命名约定，并对其进行扩展以描述通道特征。

集群: 拓扑设计注意事项
本主题提供有关规划和管理 IBM MQ 集群的指导。 此信息是基于测试和客户反馈的指南。

通过提前思考用户应用程序和内部管理流程的位置，可以避免许多问题，或者在稍后的日期将其最小化。 本主题包含有关设计决策的信息，这些决策可以提高性能，并随着集群的扩展而简化维护任务。

集群基础结构的性能

当应用程序尝试在集群中的队列管理器上打开队列时，队列管理器会向该队列的完整存储库注册其兴趣，以便它可以了解该队列在集群中的位置。 队列位置或配置的任何更新都将由完整存储库自动发送到相关队列管理器。 此相关注册在内部称为预订 (这些预订与 IBM MQ 中用于发布/预订消息传递的 IBM MQ 预订不同)

有关集群的所有信息都将通过每个完整存储库。 因此，完整存储库始终在集群中用于管理消息流量。 在管理这些预订时，系统资源的高使用率，以及这些预订的传输和生成的配置消息，都可能对集群基础结构造成相当大的负载。 在确保尽可能了解并最小化此负载时，需要考虑许多事项:

- 使用集群队列的单个队列管理器越多，系统中的预订就越多，因此发生更改时的管理开销越大，需要通知感兴趣的订户，尤其是在完整存储库队列管理器上。 将不必要的流量和完整存储库负载最小化的一种方法是将类似的应用程序 (即，那些使用相同队列的应用程序) 连接到较少数量的队列管理器。
- 除了系统中影响性能的预订数之外，集群对象配置的更改率也会影响性能，例如频繁更改集群队列配置。
- 当队列管理器是多个集群的成员 (即，它是重叠集群系统的一部分) 时，队列中的任何兴趣都会导致对它是其成员的每个集群进行预订，即使相同的队列管理器是多个集群的完整存储库也是如此。 此安排会增加系统上的负载，并且是考虑是否需要多个重叠集群 (而不是单个集群) 的一个原因。
- 应用程序消息流量 (即， IBM MQ 应用程序发送到集群队列的消息) 不会通过完整存储库到达目标队列管理器。 此消息流量直接在消息进入集群的队列管理器与集群队列所在的队列管理器之间发送。 因此，不必适应与完整存储库队列管理器相关的应用程序消息流量的高速率，除非完整存储库队列管理器恰好是提及的这两个队列管理器中的任一个。 因此，建议不要将完整存储库队列管理器用于集群基础结构负载较大的集群中的应用程序消息流量。

完整存储库

存储库是有关作为集群成员的队列管理器的信息集合。 托管集群中每个队列管理器的完整信息集的队列管理器具有完整的存储库。 有关完整存储库和部分存储库的更多信息，请参阅 集群存储库。

完整存储库必须保存在可靠且尽可能高可用性的服务器上，并且必须避免单点故障。 集群设计必须始终具有两个完整存储库。 如果完整存储库发生故障，那么集群仍可运行。

集群中的队列管理器对集群资源进行的任何更新的详细信息; 例如，集群队列从该队列管理器发送到该集群中最多两个完整存储库 (如果集群中只有一个完整存储库队列管理器，那么发送到一个完整存储库)。 这些完整存储库保存信息，并将其传播到集群中对其感兴趣的任何队列管理器 (即，他们预订该信息)。 要确保集群的每个成员都具有其中的集群资源的最新视图，每个队列管理器必须能够在任何时候与至少一个完整存储库队列管理器进行通信。

如果由于任何原因，队列管理器无法与任何完整存储库通信，那么它可以根据其已高速缓存的信息级别在一段时间内继续在集群中运行，但没有新的更新或对先前未使用的集群资源的访问权。

因此，您必须始终保持这两个完整存储库可用。 但是，这一安排并不意味着必须采取极端措施，因为集群在没有完整存储库的短时间内充分运作。

还有一个原因是集群必须具有两个完整存储库队列管理器 (集群信息的可用性除外): 此原因是为了确保保存在完整存储库高速缓存中的集群信息存在于两个位置以进行恢复。 如果只有一个完整存储库，并且它丢失了有关集群的信息，那么需要对集群中的所有队列管理器进行手动干预才能使集群再次工作。 但是，如果有两个完整存储库，那么由于始终将信息发布到两个完整存储库并从两个完整存储库预订信息，因此可以尽最大努力恢复失败的完整存储库。

- 可以在两个完整存储库集群设计中对完整存储库队列管理器执行维护，而不影响该集群的用户: 该集群仅使用一个存储库继续运行，因此在可能的情况下，请关闭存储库，应用维护，然后一次备份一个存储库。 即使在第二个完整存储库上发生中断，运行中的应用程序也至少不受影响三天。

- 除非有充分的理由使用第三个存储库，例如出于地理原因使用地理上的本地完整存储库，否则请使用两个存储库设计。 具有三个完整存储库意味着您永远不知道当前正在使用的两个存储库，并且可能存在由多个工作负载管理参数之间的交互所导致的管理问题。 建议不要有两个以上的完整存储库。
- 如果仍需要更好的可用性，请考虑将完整存储库队列管理器作为多实例队列管理器托管，或者使用特定于平台的高可用性支持来提高其可用性。
- 您必须将所有完整存储库队列管理器与手动定义的集群发送方通道完全互连。 当由于某种合理原因，集群具有两个以上完整存储库时，必须特别小心。 在这种情况下往往可能错过一个或多个渠道，为的是不至于立竿见影。 当完全互联不发生时，往往会出现难以诊断的问题。 它们很难诊断，因为某些完整存储库未保存所有存储库数据，因此导致集群中的队列管理器具有不同的集群视图，具体取决于它们所连接的完整存储库。

应用程序是否应该在完整存储库上使用队列?

完整存储库在大多数方面与任何其他队列管理器完全相同，因此可以在完整存储库上托管应用程序队列，并将应用程序直接连接到这些队列管理器。 应用程序是否应该在完整存储库上使用队列?

普遍接受的答案是 "否?"。 虽然可以进行此配置，但许多客户希望保留这些队列管理器以专用于维护完整的存储库集群高速缓存。 此处描述了在决定任一选项时要考虑的点，但最终集群体系结构必须适合环境的特定需求。

- 升级: 通常，为了在 IBM MQ 的新发行版中使用新的集群功能，必须首先升级该集群的完整存储库队列管理器。 当集群中的应用程序想要使用新功能时，能够更新完整存储库 (以及部分存储库的部分子集) 而无需测试多个共存的应用程序可能很有用。
- 维护: 以类似方式，如果必须对完整存储库应用紧急维护，那么可以使用 **REFRESH** 命令重新启动或刷新这些存储库，而无需接触应用程序。
- 性能: 随着集群的增长以及对完整存储库集群高速缓存维护的需求越来越大，使应用程序保持独立会降低通过争用系统资源而影响应用程序性能的风险。
- 硬件需求: 通常，完整存储库不需要强大功能; 例如，具有良好可用性期望的简单 UNIX 服务器就足够了。 或者，对于非常大或不断变化的集群，必须考虑完整存储库计算机的性能。
- 软件需求: 需求通常是选择在完整存储库上托管应用程序队列的主要原因。 在小型集群中，并置可能意味着需要减少所有队列管理器/服务器的数量。

管理通道定义

即使在单个集群中，也可以存在多个通道定义，从而在两个队列管理器之间提供多个路由。

有时在单个集群中具有并行通道的优势，但必须彻底考虑此设计决策; 除了增加复杂性外，此设计可能导致通道使用不足，从而降低性能。 发生此情况是因为测试通常涉及以恒定速率发送大量消息，因此完全使用并行通道。 但在非恒定消息流的现实世界条件下，随着消息流从通道切换到通道，工作负载均衡算法会导致性能下降。

当队列管理器是多个集群的成员时，存在将单个通道定义与集群名称列表配合使用的选项，而不是为每个集群定义单独的 CLUSRCVR 通道。 但是，此设置可能会在以后导致管理困难; 例如，考虑将 TLS 应用于一个集群而不是另一个集群的情况。 因此，最好创建单独的定义，并且 第 28 页的『集群命名约定』 中建议的命名约定支持此操作。

多个通道上的工作负载均衡

此信息旨在作为对主题的高级理解。 有关此主题的基本说明 (在使用此处的信息之前必须了解)，请参阅 使用集群进行工作负载管理， 集群中的工作负载均衡和 集群工作负载管理算法。

集群工作负载管理算法提供了一组庞大的工具，但如果不充分了解它们的工作和交互方式，它们就不能全部相互配合使用。 对于工作负载均衡过程，通道的重要性可能并不是很明显: 工作负载管理循环法算法的行为就像拥有集群队列的队列管理器的多个集群通道被视为该队列的多个实例一样。 以下示例更详细地说明了此过程:

1. 集群中有两个托管队列的队列管理器: QM1 和 QM2。
2. 有五个集群接收方通道到 QM1。

3. 只有一个集群接收方通道到 QM2。
4. 当 QM3 上的 **MQPUT** 或 **MQOPEN** 选择实例时，算法将消息发送到 QM1 的可能性比发送到 QM2 的可能性高五倍。
5. 发生步骤 4 中的情况是因为算法看到六个选项可供选择 (5 + 1)，并在所有五个通道之间循环至 QM1 和单个通道至 QM2。

另一个微妙的行为是，即使将消息放入正好在本地队列管理器上配置了一个实例的集群队列， IBM MQ 也会使用本地集群接收方通道的状态来决定是将消息放入队列的本地实例还是队列的远程实例。 在此场景中:

1. 在放置消息时，工作负载管理算法不会查看个别集群队列，而是会查看可到达这些目标的集群通道。
2. 要到达本地目标，本地接收方通道将包括在此列表中 (尽管它们不用于发送消息)。
3. 当本地接收方通道停止时，工作负载管理算法在缺省情况下首选备用实例 (如果其 CLUSRCVR 未停止)。 如果目标有多个本地 CLUSRCVR 实例，并且至少有一个实例未停止，那么本地实例仍符合条件。

集群: 使用多个集群传输队列进行应用程序隔离

您可以在集群中的队列管理器之间隔离消息流。 您可以将由不同集群发送方通道传输的消息放入不同的集群传输队列中。 您可以在单个集群中使用此方法，也可以将此方法与重叠的集群配合使用。 本主题提供了一些示例和一些最佳实践，以指导您选择使用方法。

部署应用程序时，您可以选择它与其他应用程序共享哪些 IBM MQ 资源以及它不共享哪些资源。 可以共享多种类型的资源，主要资源是服务器本身，队列管理器，通道和队列。 您可以选择配置共享资源较少的应用程序; 将单独的队列，通道，队列管理器甚至服务器分配给各个应用程序。 如果执行此操作，那么整体系统配置将变得更大且更复杂。 使用 IBM MQ 集群可降低管理更多服务器，队列管理器，队列和通道的复杂性，但它会引入另一个共享资源，即集群传输队列 SYSTEM.CLUSTER.TRANSMIT.QUEUE。

第 40 页的图 12 是大型 IBM MQ 部署中的一个切片，用于说明共享 SYSTEM.CLUSTER.TRANSMIT.QUEUE 的重要性。 在图中，应用程序 Client App 已连接到集群 CL1 中的队列管理器 QM2 。 来自 Client App 的消息由应用程序 Server App 处理。 此消息由 Server App 从 CLUSTER2 中队列管理器 QM3 上的集群队列 Q1 中检索。 由于客户机和服务器应用程序不在同一集群中，因此消息由网关队列管理器 QM1 传输。

配置集群网关的正常方法是使网关队列管理器成为所有集群的成员。 在网关队列管理器上，为所有集群中的集群队列定义了集群别名队列。 集群队列别名在所有集群中都可用。 放入集群队列别名的消息通过网关队列管理器路由到其正确目标。 网关队列管理器将发送到集群别名队列的消息放到 QM1 上的公共 SYSTEM.CLUSTER.TRANSMIT.QUEUE 上。

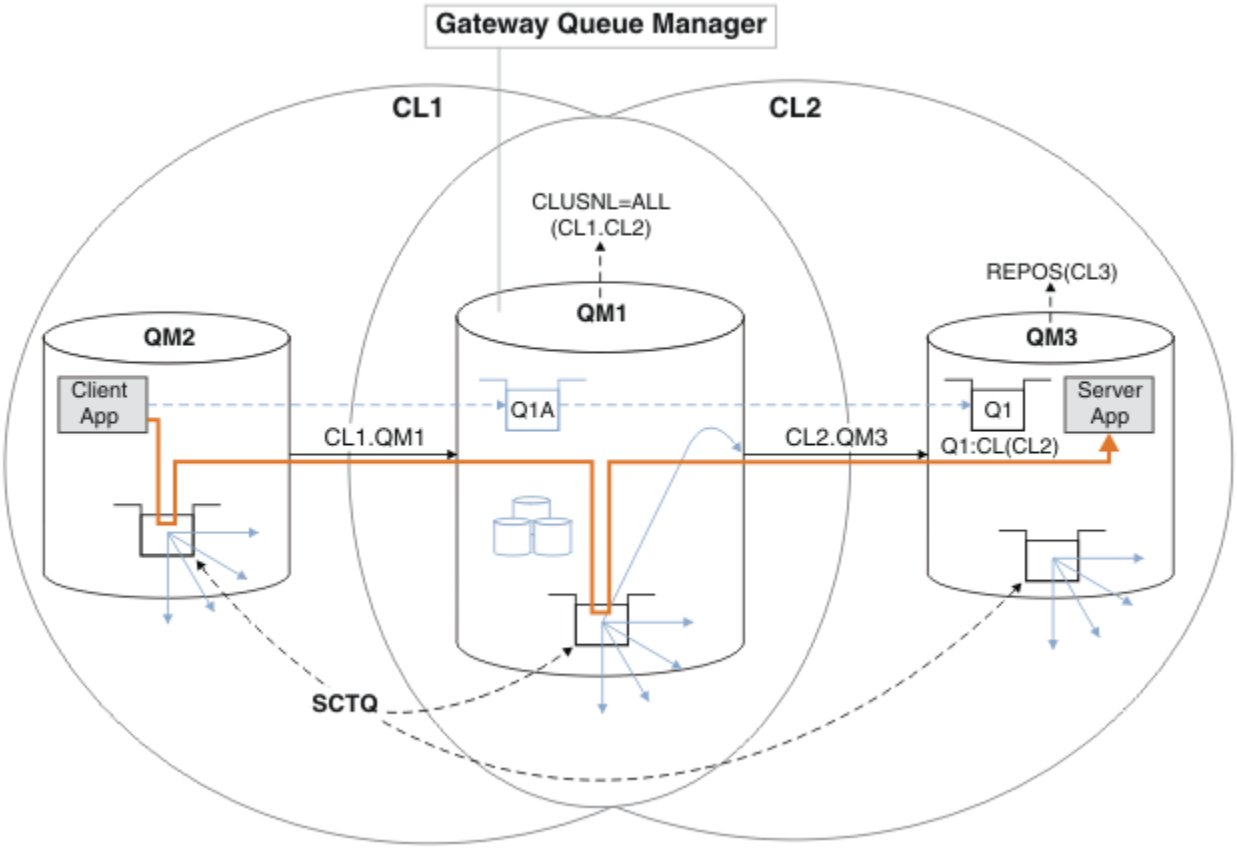
中心和辐射体系结构要求集群之间的所有消息都通过网关队列管理器。 结果是所有消息都流经 QM1 SYSTEM.CLUSTER.TRANSMIT.QUEUE 上的单集群传输队列。

从性能角度来看，单个队列不是问题。 公共传输队列通常不表示性能瓶颈。 网关上的消息吞吐量在很大程度上取决于与其连接的通道的性能。 吞吐量通常不受队列数或使用通道的队列上的消息数的影响。

从其他一些角度来看，将单个传输队列用于多个应用程序有缺点:

- 不能将消息流到一个目标与消息流到另一个目标隔离。 即使目标位于不同队列管理器上的不同集群中，也无法在转发消息之前将其存储分开。

  如果一个集群目标变为不可用，那么将在单个传输队列中构建该目标的消息，并最终填充该消息。 一旦传输队列已满，它就会停止将消息放置到任何集群目标的传输队列上。
- 监视消息到不同集群目标的传输并不容易。 所有消息都在单个传输队列上。 显示传输队列的深度几乎不会指示是否将消息传输到所有目标。



注: 第 40 页的图 12 和下图中的箭头具有不同类型。 实心箭头表示消息流。 实线箭头上的标签是消息通道名称。 灰色实线箭头是从 SYSTEM.CLUSTER.TRANSMIT.QUEUE 到集群发送方通道的潜在消息流。 黑色虚线将标签连接到其目标。 灰色虚线箭头是引用; 例如，从 MQOPEN 调用 Client App 到集群别名队列定义 Q1A。

图 *12:* 使用 *IBM MQ* 集群将客户机/服务器应用程序部署到中心和辐射体系结构

在 第 40 页的图 12 中， Server App 的客户机打开队列 Q1A。 消息将放置到 QM2 上的 SYSTEM.CLUSTER.TRANSMIT.QUEUE， 传输到 QM1 上的 SYSTEM.CLUSTER.TRANSMIT.QUEUE， 然后传输到 QM3 上的 Q1， Server App 应用程序将在此接收到这些消息。

来自 Client App 的消息通过 QM2 和 QM1 上的系统集群传输队列传递。 在 第 40 页的图 12 中，目标是将网关队列管理器上的消息流与客户机应用程序隔离，以便其消息不会存储在 SYSTEM.CLUSTER.TRANSMIT.QUEUE 上。 您可以隔离任何其他集群队列管理器上的流。 您还可以将其他方向的流隔离回客户机。 为了使解决方案的描述简短，这些描述仅考虑来自客户机应用程序的单个流。

用于隔离集群网关队列管理器上的集群消息流量的解决方案

解决问题的一种方法是使用队列管理器别名或远程队列定义在集群之间进行桥接。 创建集群远程队列定义，传输队列和通道，以分隔网关队列管理器上的每个消息流; 请参阅 添加远程队列定义以隔离从网关队列管理器发送的消息。

从 IBM WebSphere MQ 7.5 开始，集群队列管理器不限于单个集群传输队列。 您有两种选择：

1. 手动定义其他集群传输队列，并定义从每个传输队列传输消息的集群发送方通道; 请参阅 添加集群传输队列以隔离从网关队列管理器发送的集群消息流量。
2. 允许队列管理器自动创建和管理其他集群传输队列。 它为每个集群发送方通道定义不同的集群传输队列; 请参阅 更改缺省值以分隔集群传输队列以隔离消息流量。

您可以将某些集群发送方通道的手动定义的集群传输队列与队列管理器一起管理其余的集群传输队列。 传输队列的组合是在 添加集群传输队列以隔离从网关队列管理器发送的集群消息流量中采用的方法。 在该解决

方案中，集群之间的大多数消息都使用公共 SYSTEM.CLUSTER.TRANSMIT.QUEUE。 一个应用程序是关键的，它的所有消息流都通过使用一个手动定义的集群传输队列与其他流隔离。

添加集群传输队列以隔离从网关队列管理器发送的集群消息流量 中的配置受到限制。 它不会将消息流量与另一个集群队列分隔到同一集群中同一队列管理器上的集群队列。 您可以使用属于分布式排队的远程队列定义来分隔到各个队列的消息流量。 借助集群，通过使用多个集群传输队列，您可以将进入不同集群发送方通道的消息流量分开。 同一集群中的多个集群队列在同一队列管理器上共享一个集群发送方通道。 这些队列的消息存储在同一传输队列上，然后再从网关队列管理器转发。 在 添加集群和集群传输队列以隔离从网关队列管理器发送的集群消息流量中的配置中，通过添加另一个集群并使队列管理器和集群队列成为新集群的成员来避免此限制。 新的队列管理器可能是集群中唯一的队列管理器。 您可以向集群添加更多队列管理器，并使用同一集群来隔离这些队列管理器上的集群队列。

相关概念

第 24 页的『访问控制和多个集群传输队列』
在应用程序将消息放入远程集群队列时选择三种检查方式。 这些方式是针对集群队列进行远程检查，针对 SYSTEM.CLUSTER.TRANSMIT.QUEUE 进行本地检查，或者针对集群队列或集群队列管理器的本地概要文件进行检查。

使用集群传输队列和集群发送方通道

第 30 页的『重叠集群』
重叠的集群提供了额外的管理功能。 使用名称列表来减少管理重叠集群所需的命令数。

相关任务

授权将消息放入远程集群队列

添加远程队列定义以隔离从网关队列管理器发送的消息

添加集群传输队列以隔离从网关队列管理器发送的集群消息流量

添加集群和集群传输队列以隔离从网关队列管理器发送的集群消息流量

将缺省值更改为单独的集群传输队列以隔离消息流量

使用网关队列管理器创建两个重叠的集群

配置集群之间的消息路径

保护

相关参考

塞特 MQaut

集群*:* 规划如何配置集群传输队列

将指导您选择集群传输队列。 您可以配置一个公共缺省队列，单独的缺省队列或手动定义的队列。

开始之前

复审 第 43 页的『如何选择要使用的集群传输队列类型』。

关于此任务

当您计划如何配置队列管理器以选择集群传输队列时，可以选择一些选项。

1. 什么是集群消息传输的缺省集群传输队列?
   a. 公共集群传输队列 SYSTEM.CLUSTER.TRANSMIT.QUEUE。
   b. 单独的集群传输队列。 队列管理器管理单独的集群传输队列。 它会将它们创建为模型队列 SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE 中的永久动态队列。 它为它使用的每个集群发送方通道创建一个集群传输队列。
2. 对于您决定手动创建的集群传输队列，还有两个选项:
   a. 为您决定手动配置的每个集群发送方通道定义单独的传输队列。 在这种情况下，请将传输队列的 **CLCHNAME** 队列属性设置为集群发送方通道的名称。 选择要从此传输队列传输消息的集群发送方通道。
   b. 将一组集群发送方通道的消息流量组合到同一集群传输队列上; 请参阅 第 42 页的图 13。 在这种情况下，请将每个公共传输队列的 **CLCHNAME** 队列属性设置为通用集群发送方通道名称。 通用集群发送

方通道名称是用于对集群发送方通道名称进行分组的过滤器。 例如， SALES.* 对名称以 SALES. 开头的所有集群发送方通道进行分组。 可以在 filter-string 中的任何位置放置多个通配符。 通配符为星号 “*”。 它表示从零到任意数目的字符。

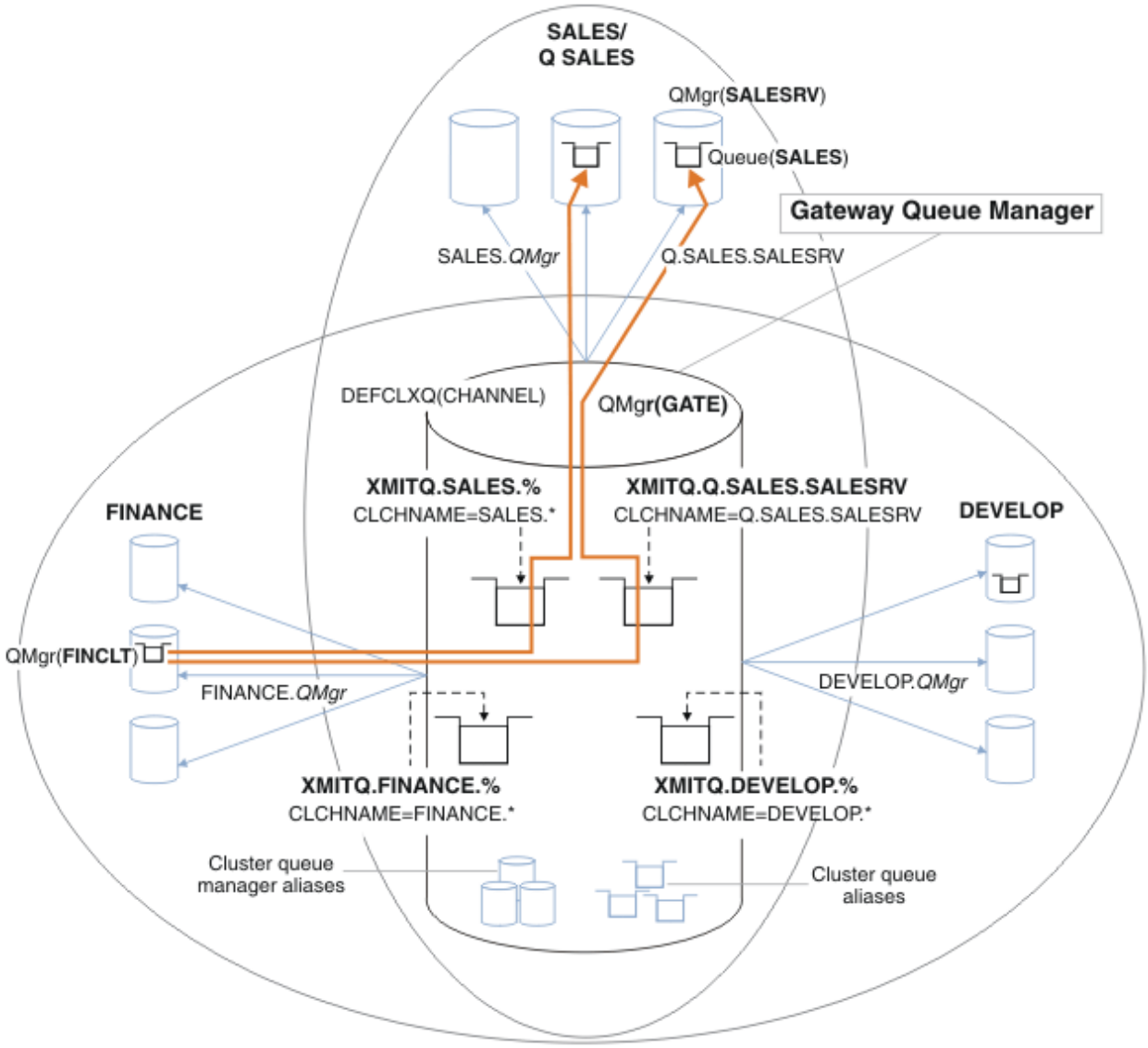


图 *13:* 不同部门 *IBM MQ* 集群的特定传输队列示例

过程

1. 选择要使用的缺省集群传输队列类型。
   - 选择单个集群传输队列，或者为每个集群连接选择单独的队列。

   保留缺省设置或运行 **MQSC** 命令:

   ```
   ALTER QMGR DEFCLXQ(CHANNEL)
   ```

2. 隔离不得与其他流共享集群传输队列的任何消息流。
   - 请参阅第 45 页的『集群: 多个集群传输队列的示例配置』。 在此示例中，必须隔离的 SALES 队列是 SALESRV 上 SALES 集群的成员。 要隔离 SALES 队列，请创建新集群 Q.SALES，使 SALESRV 队列管理器成为成员，并修改 SALES 队列以属于 Q.SALES。
   - 向 SALES 发送消息的队列管理器也必须是新集群的成员。 如果使用集群队列别名和网关队列管理器，例如，在许多情况下，可以将更改限制为使网关队列管理器成为新集群的成员。

- 但是，将流从网关分离到目标不会将流从源队列管理器分离到网关。 但有时事实证明，这足以将流与网关分开，而不是流向网关。 如果不够，请将源队列管理器添加到新集群中。 如果希望消息通过网关传递，请将集群别名移动到新集群，并继续将消息发送到网关上的集群别名，而不是直接发送到目标队列管理器。

执行以下步骤以隔离消息流:

a) 配置流的目标，以便每个目标队列都是该队列管理器上特定集群中的唯一队列。

b) 为遵循系统命名约定创建的任何新集群创建集群发送方通道和集群接收方通道。

- 请参阅第 35 页的『集群: 重叠集群的特殊注意事项』。

c) 为每个将消息发送到目标队列的队列管理器上的每个隔离目标定义集群传输队列。

- 集群传输队列的命名约定是使用以 XMITQ. 为前缀的集群通道名称属性 CLCHNAME 的值

3. 创建集群传输队列以满足监管或监视需求。

- 典型的监管和监视需求会导致每个集群的传输队列或每个队列管理器的传输队列。 如果遵循集群通道 *ClusterName. QueueManagerName* 的命名约定，那么可以轻松创建用于选择队列管理器集群或队列管理器所属的所有集群的通用通道名称; 请参阅 第 45 页的『集群: 多个集群传输队列的示例配置』。
- 通过将星号替换为百分号，扩展集群传输队列的命名约定以迎合通用通道名称。 例如

```
DEFINE QLOCAL(XMITQ.SALES.%) USAGE(XMITQ) CLCHNAME(SALES.*)
```

相关概念

使用集群传输队列和集群发送方通道

第 24 页的『访问控制和多个集群传输队列』
在应用程序将消息放入远程集群队列时选择三种检查方式。 这些方式是针对集群队列进行远程检查，针对 SYSTEM.CLUSTER.TRANSMIT.QUEUE 进行本地检查，或者针对集群队列或集群队列管理器的本地概要文件进行检查。

第 30 页的『重叠集群』
重叠的集群提供了额外的管理功能。 使用名称列表来减少管理重叠集群所需的命令数。

相关任务

添加远程队列定义以隔离从网关队列管理器发送的消息

添加集群传输队列以隔离从网关队列管理器发送的集群消息流量

添加集群和集群传输队列以隔离从网关队列管理器发送的集群消息流量

将缺省值更改为单独的集群传输队列以隔离消息流量

使用网关队列管理器创建两个重叠的集群

配置集群之间的消息路径

如何选择要使用的集群传输队列类型
如何在不同的集群传输队列配置选项之间进行选择。

您可以选择哪个集群传输队列与集群发送方通道相关联。

1. 您可以使所有集群发送方通道与单个缺省集群传输队列 SYSTEM.CLUSTER.TRANSMIT.QUEUE 相关联; 此选项是缺省选项。
2. 您可以将所有集群发送方通道设置为自动与单独的集群传输队列相关联。 队列由队列管理器从模型队列 SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE 创建，命名为 SYSTEM.CLUSTER.TRANSMIT. *ChannelName*。 如果队列管理器属性 **DEFCLXQ** 设置为 CHANNEL，那么通道将使用其唯一命名的集群传输队列。
3. 您可以设置要由单个集群传输队列提供服务的特定集群发送方通道。 通过创建传输队列并将其 **CLCHNAME** 属性设置为集群发送方通道的名称来选择此选项。
4. 您可以选择要由单个集群传输队列提供服务的集群发送方通道组。 通过创建传输队列并将其 **CLCHNAME** 属性设置为通用通道名称 (例如 *ClusterName.**) 来选择此选项。 如果通过遵循 第 35 页的『集群: 重叠集群的特殊注意事项』中的命名约定来命名集群通道，那么此名称将选择连接到集群 *ClusterName* 中的队列管理器的所有集群通道。

您可以将某些集群发送方通道的任一缺省集群传输队列选项与任意数量的特定和通用集群传输队列配置组合在一起。

最佳实践

在大多数情况下，对于现有 IBM MQ 安装，缺省配置是最佳选择。 集群队列管理器将集群消息存储在单个集群传输队列 SYSTEM.CLUSTER.TRANSMIT.QUEUE 上。 您可以选择将缺省值更改为将不同队列管理器和不同集群的消息存储在不同的传输队列上，或者定义您自己的传输队列。

在大多数情况下，对于新的 IBM MQ 安装，缺省配置也是最佳选择。 从缺省配置切换到针对每个集群发送方通道具有一个传输队列的备用缺省值的过程是自动的。 切换回也是自动的。 一个或另一个的选择并不关键，你可以逆转它。

选择其他配置的原因更多是与监管和管理有关，而不是与功能或性能有关。 如果有几个异常，那么配置多个集群传输队列对队列管理器的行为没有好处。 这将生成更多队列，并要求您修改已设置的监视和管理过程，这些过程将引用单个传输队列。 这就是为什么在平衡上，保留缺省配置是最佳选择，除非您有强大的治理或管理原因进行不同的选择。

这些异常都与 SYSTEM.CLUSTER.TRANSMIT.QUEUE 上存储的消息数增加时发生的情况相关。 如果您执行每个步骤来将一个目标的消息与另一个目标的消息分开，那么一个目标的通道和传递问题不应影响到另一个目标的传递。 但是，由于无法以足够快的速度将消息传递到一个目标，因此存储在 SYSTEM.CLUSTER.TRANSMIT.QUEUE 上的消息数可能会增加。 SYSTEM.CLUSTER.TRANSMIT.QUEUE 上针对一个目标的消息数可能会影响将消息传递到其他目标。

要避免由于填充单个传输队列而导致的问题，请将足够的容量构建到配置中。 然后，如果目标失败并且消息积压开始构建，那么您有时间来解决该问题。

如果通过中心队列管理器 (例如集群网关) 路由消息，那么它们共享公共传输队列 SYSTEM.CLUSTER.TRANSMIT.QUEUE。 如果网关队列管理器上存储在 SYSTEM.CLUSTER.TRANSMIT.QUEUE 上的消息数达到其最大深度，那么队列管理器将开始拒绝传输队列的新消息，直到其深度减小为止。 拥塞会影响通过网关路由的所有目标的消息。 消息备份正在向网关发送消息的其他队列管理器的传输队列。 该问题表现在写入队列管理器错误日志的消息中，消息吞吐量下降，以及从发送消息到消息到达其目标的时间之间的耗用时间较长。

拥堵对单个传输队列的影响会变得明显，即使在它已满之前也是如此。 如果您有混合的消息流量，有一些大的非持久消息和一些小的消息，那么随着传输队列的填满，传递小消息的时间会增加。 延迟是由于将通常不会写入磁盘的大型非持久消息写入磁盘。 如果您有时间关键消息流，与其他混合消息流共享集群传输队列，那么可能需要配置特殊消息路径以将其与其他消息流隔离; 请参阅 添加集群和集群传输队列以隔离从网关队列管理器发送的集群消息流量。

配置单独的集群传输队列的其他原因是为了满足监管要求，或者为了简化发送到不同集群目标的监视消息。 例如，您可能必须证明一个目标的消息从不与另一个目标的消息共享传输队列。

更改用于控制缺省集群传输队列的队列管理器属性 **DEFCLXQ**，以便为每个集群发送方通道创建不同的集群传输队列。 多个目标可以共享一个集群发送方通道，因此您必须规划集群以完全满足此目标。 将方法 添加集群和集群传输队列以隔离从网关队列管理器发送的集群消息流量 系统地应用于所有集群队列。 您的目标结果是没有集群目标与另一个集群目标共享集群发送方通道。 因此，集群目标的任何消息都不会与另一个目标的消息共享其集群传输队列。

为某些特定消息流创建单独的集群传输队列，可以轻松监视到该目标的消息流。 要使用新的集群传输队列，请定义该队列，将其与集群发送方通道相关联，然后停止并启动该通道。 此更改不必是永久性的。 您可以将消息流隔离一段时间，以监视传输队列，然后再次还原为使用缺省传输队列。

相关任务

集群: 多个集群传输队列的示例配置
在此任务中，您将应用步骤以将多个集群传输队列规划到三个重叠的集群。 这些要求是将消息流与所有其他消息流分离到一个集群队列，并将不同集群的消息存储在不同的集群传输队列上。

集群: 切换集群传输队列
规划如何使对现有生产队列管理器的集群传输队列所作的更改生效。

集群: 多个集群传输队列的示例配置
在此任务中，您将应用步骤以将多个集群传输队列规划到三个重叠的集群。 这些要求是将消息流与所有其他消息流分离到一个集群队列，并将不同集群的消息存储在不同的集群传输队列上。

关于此任务

此任务中的步骤显示如何应用 第 41 页的『集群: 规划如何配置集群传输队列』 中的过程并得出 第 45 页的图 14 中显示的配置。 它是具有网关队列管理器的三个重叠集群的示例，该集群配置有单独的集群传输队列。 第 47 页的『创建示例集群』 中描述了用于定义集群的 MQSC 命令。

对于此示例，有两个需求。 一是将消息流从网关队列管理器分离到记录销售的销售应用程序。 二是查询有多少消息等待在任何时间点发送到不同的部门地区。 已定义 SALES， FINANCE 和 DEVELOP 集群。 集群消息当前从 SYSTEM.CLUSTER.TRANSMIT.QUEUE 转发。

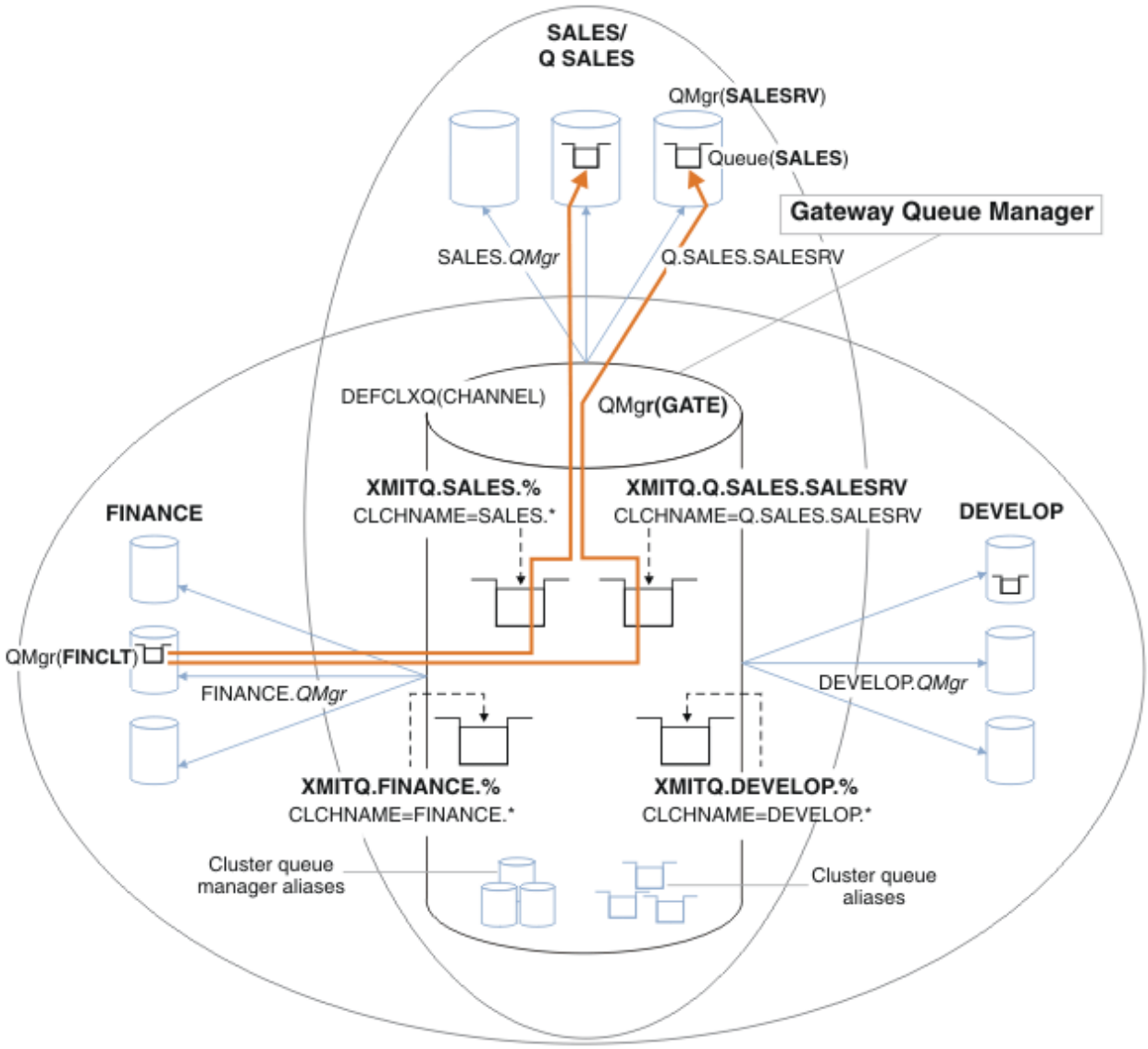


图 *14:* 不同部门 *IBM MQ* 集群的特定传输队列示例

修改集群的步骤如下所示。 有关定义，请参阅 更改以隔离新集群中的销售队列并分隔网关集群传输队列。

过程

1. 第一个配置步骤是 “ 选择要使用的缺省集群传输队列类型 ”。

决定是通过在 GATE 队列管理器上运行以下 **MQSC** 命令来创建单独的缺省集群传输队列。

```
ALTER QMGR DEFCLXQ(CHANNEL)
```

没有强烈的理由选择此缺省值，因为目的是手动定义集群传输队列。 该选项确实具有弱诊断值。 如果手动定义错误，并且消息在缺省集群传输队列中流下，那么会在创建永久动态集群传输队列时显示该消息。

2. 第二个配置步骤是 “ 隔离不得与其他流共享集群传输队列的任何消息流 ”。

   在这种情况下，从 SALESRV 上的队列 SALES 接收消息的销售应用程序需要隔离。 仅需要隔离来自网关队列管理器的消息。 三个分步骤是:

   a) “ 配置流的目标，以便每个目标队列都是该队列管理器上特定集群中的唯一队列 ”.

      此示例需要将队列管理器 SALESRV 添加到销售部门中的新集群。 如果您有几个需要隔离的队列，那么可以决定为 SALES 队列创建特定集群。 集群名称的可能命名约定是将此类集群命名为 Q. *QueueName*，例如 Q.SALES。 如果要隔离大量队列，那么可能更实际的替代方法是在需要时创建隔离队列的集群。 集群名称可能为 QUEUES. *n*。

      在此示例中，新集群称为 Q.SALES。 要添加新集群，请参阅 更改以隔离新集群中的销售队列并分隔网关集群传输队列中的定义。 定义更改的摘要如下:

      i) 将 Q.SALES 添加到存储库队列管理器上的集群的名称列表。 在队列管理器 **REPOSNL** 参数中引用了名称列表。

      ii) 将 Q.SALES 添加到网关队列管理器上的集群的名称列表。 在网关队列管理器上的所有集群队列别名和集群队列管理器别名定义中都引用了名称列表。

      iii) 在队列管理器 SALESRV 上为其所属的两个集群创建名称列表，并更改 SALES 队列的集群成员资格:

      ```
      DEFINE NAMELIST(CLUSTERS) NAMES(SALES, Q.SALES) REPLACE
      ALTER QLOCAL(SALES) CLUSTER(' ') CLUSNL(SALESRV.CLUSTERS)
      ```

      SALES 队列是两个集群的成员，仅用于转换。 新配置运行后，将从 SALES 集群中除去 SALES 队列; 请参阅 第 50 页的图 15。

   b) “ 为遵循系统命名约定创建的任何新集群创建集群发送方通道和集群接收方通道 ”.

      i) 将集群接收方通道 Q.SALES. *RepositoryQMgr* 添加到每个存储库队列管理器

      ii) 将集群发送方通道 Q.SALES. *OtherRepositoryQMgr* 添加到每个存储库队列管理器，以连接到其他存储库管理器。 启动这些通道。

      iii) 将集群接收方通道 Q.SALES.SALESRV 和 Q.SALES.GATE 添加到正在运行的任一存储库队列管理器。

      iv) 将集群发送方通道 Q.SALES.SALESRV 和 Q.SALES.GATE 添加到 SALESRV 和 GATE 队列管理器。 将集群发送方通道连接到您在其中创建集群接收方通道的存储库队列管理器。

   c) “ 为每个将消息发送到目标队列的队列管理器上的每个隔离目标定义集群传输队列 ”.

      在网关队列管理器上，为 Q.SALES.SALESRV 集群发送方通道定义集群传输队列 XMITQ.Q.SALES.SALESRV :

      ```
      DEFINE QLOCAL(XMITQ.Q.SALES.SALESRV) USAGE(XMITQ) CLCHNAME(Q.SALES.SALESRV) REPLACE
      ```

3. 第三个配置步骤是 “ 创建集群传输队列以满足监管或监视需求 ”。

   在网关队列管理器上定义集群传输队列:

   ```
   DEFINE QLOCAL(XMITQ.SALES)  USAGE(XMITQ) CLCHNAME(SALES.*)  REPLACE
   DEFINE QLOCAL(XMITQ.DEVELOP) USAGE(XMITQ) CLCHNAME(DEVELOP.*) REPLACE
   DEFINE QLOCAL(XMITQ.FINANCE) USAGE(XMITQ) CLCHNAME(SALES.*)  REPLACE
   ```

下一步做什么

切换到网关队列管理器上的新配置。

通过启动新通道并重新启动现在与不同传输队列相关联的通道来触发交换机。 或者，可以停止并启动网关队列管理器。

1. 停止网关队列管理器上的以下通道:

```
SALES. Qmgr
DEVELOP. Qmgr
FINANCE. Qmgr
```

2. 在网关队列管理器上启动以下通道:

```
SALES. Qmgr
DEVELOP. Qmgr
FINANCE. Qmgr
Q.SALES.SAVESRV
```

交换机完成后，从 SALES 集群中除去 SALES 队列; 请参阅 第 50 页的图 15。

相关概念

如何选择要使用的集群传输队列类型
如何在不同的集群传输队列配置选项之间进行选择。

相关任务

集群: 切换集群传输队列
规划如何使对现有生产队列管理器的集群传输队列所作的更改生效。

创建示例集群
用于创建示例集群并对其进行修改以隔离 SALES 队列和网关队列管理器上的单独消息的定义和指示信息。

关于此任务

用于创建 FINANCE， SALES 和 Q.SALES 集群的完整 **MQSC** 命令在 基本集群的定义， 用于隔离新集群中的销售队列并分隔网关集群传输队列的更改和 从销售集群中除去队列管理器 SALESRV 上的销售队列中提供。 定义中省略了 DEVELOP 集群，以保持定义较短。

过程

1. 创建 SALES 和 FINANCE 集群以及网关队列管理器。
   a) 创建队列管理器。

   对 第 47 页的表 4 中的每个队列管理器名称运行命令 `crtmqm -sax -u SYSTEM.DEAD.LETTER.QUEUE QmgrName` 。

表 *4:* 队列管理器名称和端口号

描述	队列管理器名称	端口号
财务存储库	FINR1	1414
财务存储库	FINR2	1415
财务客户	FINCLT	1418
销售存储库	SALER1	1416
销售存储库	SALER2	1417
销售服务器	SALESRV	1419
网关	GATE	1420

b) 启动所有队列管理器

对 第 47 页的表 4 中的每个队列管理器名称运行命令 `strmqm` *QmgrName* 。

c) 为每个队列管理器创建定义

Run the command: `runmqsc` *QmgrName* < *filename* where the files are listed in 基本集群的定义, and the file name matches the queue manager name.

基本集群的定义

finr1.txt

```
DEFINE LISTENER(1414) TRPTYPE(TCP) IPADDR(localhost) CONTROL(QMGR) PORT(1414) REPLACE
START LISTENER(1414)
ALTER QMGR REPOS(FINANCE)
DEFINE CHANNEL(FINANCE.FINR2) CHLTYPE(CLUSSDR) CONNAME('localhost(1415)')
CLUSTER(FINANCE) REPLACE
DEFINE CHANNEL(FINANCE.FINR1) CHLTYPE(CLUSRCVR) CONNAME('localhost(1414)')
CLUSTER(FINANCE) REPLACE
```

finr2.txt

```
DEFINE LISTENER(1415) TRPTYPE(TCP) IPADDR(localhost) CONTROL(QMGR) PORT(1415) REPLACE
START LISTENER(1415)
ALTER QMGR REPOS(FINANCE)
DEFINE CHANNEL(FINANCE.FINR1) CHLTYPE(CLUSSDR) CONNAME('localhost(1414)')
CLUSTER(FINANCE) REPLACE
DEFINE CHANNEL(FINANCE.FINR2) CHLTYPE(CLUSRCVR) CONNAME('localhost(1415)')
CLUSTER(FINANCE) REPLACE
```

finclt.txt

```
DEFINE LISTENER(1418) TRPTYPE(TCP) IPADDR(localhost) CONTROL(QMGR) PORT(1418) REPLACE
START LISTENER(1418)
DEFINE CHANNEL(FINANCE.FINR1) CHLTYPE(CLUSSDR)  CONNAME('localhost(1414)')
CLUSTER(FINANCE) REPLACE
DEFINE CHANNEL(FINANCE.FINCLT) CHLTYPE(CLUSRCVR) CONNAME('localhost(1418)')
CLUSTER(FINANCE) REPLACE
DEFINE QMODEL(SYSTEM.SAMPLE.REPLY) REPLACE
```

saler1.txt

```
DEFINE LISTENER(1416) TRPTYPE(TCP) IPADDR(localhost) CONTROL(QMGR) PORT(1416) REPLACE
START LISTENER(1416)
ALTER QMGR REPOS(SALES)
DEFINE CHANNEL(SALES.SALER2) CHLTYPE(CLUSSDR) CONNAME('localhost(1417)')
CLUSTER(SALES) REPLACE
DEFINE CHANNEL(SALES.SALER1) CHLTYPE(CLUSRCVR) CONNAME('localhost(1416)')
CLUSTER(SALES) REPLACE
```

saler2.txt

```
DEFINE LISTENER(1417) TRPTYPE(TCP) IPADDR(localhost) CONTROL(QMGR) PORT(1417) REPLACE
START LISTENER(1417)
ALTER QMGR REPOS(SALES)
DEFINE CHANNEL(SALES.SALER1) CHLTYPE(CLUSSDR) CONNAME('localhost(1416)')
CLUSTER(SALES) REPLACE
DEFINE CHANNEL(SALES.SALER2) CHLTYPE(CLUSRCVR) CONNAME('localhost(1417)')
CLUSTER(SALES) REPLACE
```

salesrv.txt

```
DEFINE LISTENER(1419) TRPTYPE(TCP) IPADDR(localhost) CONTROL(QMGR) PORT(1419) REPLACE
START LISTENER(1419)
DEFINE CHANNEL(SALES.SALER1) CHLTYPE(CLUSSDR) CONNAME('localhost(1416)')
CLUSTER(SALES) REPLACE
DEFINE CHANNEL(SALES.SALESRV) CHLTYPE(CLUSRCVR) CONNAME('localhost(1419)')
CLUSTER(SALES) REPLACE
DEFINE QLOCAL(SALES) CLUSTER(SALES) TRIGGER INITQ(SYSTEM.DEFAULT.INITIATION.QUEUE)
```

```
PROCESS(ECHO) REPLACE
DEFINE PROCESS(ECHO) APPLICID(AMQSECH) REPLACE
```

gate.txt

```
DEFINE LISTENER(1420) TRPTYPE(TCP) IPADDR(LOCALHOST) CONTROL(QMGR) PORT(1420) REPLACE
START LISTENER(1420)
DEFINE NAMELIST(ALL) NAMES(SALES, FINANCE)
DEFINE CHANNEL(FINANCE.FINR1) CHLTYPE(CLUSSDR) CONNAME('LOCALHOST(1414)')
CLUSTER(FINANCE) REPLACE
DEFINE CHANNEL(FINANCE.GATE) CHLTYPE(CLUSRCVR) CONNAME('LOCALHOST(1420)')
CLUSTER(FINANCE) REPLACE
DEFINE CHANNEL(SALES.SALER1) CHLTYPE(CLUSSDR) CONNAME('LOCALHOST(1416)')
CLUSTER(SALES) REPLACE
DEFINE CHANNEL(SALES.GATE)  CHLTYPE(CLUSRCVR) CONNAME('LOCALHOST(1420)')
CLUSTER(SALES) REPLACE
DEFINE QALIAS(A.SALES) CLUSNL(ALL) TARGET(SALES) TARGTYPE(QUEUE) DEFBIND(NOTFIXED)
REPLACE
DEFINE QREMOTE(FINCLT) RNAME(' ') RQMNAME(FINCLT) CLUSNL(ALL) REPLACE
DEFINE QREMOTE(SALESRV) RNAME(' ') RQMNAME(SALESRV) CLUSNL(ALL) REPLACE
```

2. 通过运行样本请求程序来测试配置。
   a) 在 SALESRV 队列管理器上启动触发器监视器程序

      在 Windows 上，打开命令窗口并运行命令 `runmqtrm -m SALESRV`

   b) 运行样本请求程序，然后发送请求。

      在 Windows 上，打开命令窗口并运行命令 `amqsreq A.SALES FINCLT`

      将回传请求消息，并在 15 秒后完成样本程序。
3. 创建定义以隔离 Q.SALES 集群中的 SALES 队列，并在网关队列管理器上为 SALES 和 FINANCE 集群单独创建集群消息。

   Run the command: `runmqsc` *QmgrName* < *filename* where the files are listed in the following list, and the file name almost matches the queue manager name.

用于隔离新集群中的销售队列并分隔网关集群传输队列的更改
chgsaler1.txt

```
DEFINE NAMELIST(CLUSTERS) NAMES(SALES, Q.SALES)
ALTER QMGR REPOS(' ') REPOSNL(CLUSTERS)
DEFINE CHANNEL(Q.SALES.SALER2) CHLTYPE(CLUSSDR) CONNAME('localhost(1417)')
CLUSTER(Q.SALES) REPLACE
DEFINE CHANNEL(Q.SALES.SALER1) CHLTYPE(CLUSRCVR) CONNAME('localhost(1416)')
CLUSTER(Q.SALES) REPLACE
```

chgsaler2.txt

```
DEFINE NAMELIST(CLUSTERS) NAMES(SALES, Q.SALES)
ALTER QMGR REPOS(' ') REPOSNL(CLUSTERS)
DEFINE CHANNEL(Q.SALES.SALER1) CHLTYPE(CLUSSDR) CONNAME('localhost(1416)')
CLUSTER(Q.SALES) REPLACE
DEFINE CHANNEL(Q.SALES.SALER2) CHLTYPE(CLUSRCVR) CONNAME('localhost(1417)')
CLUSTER(Q.SALES) REPLACE
```

chgsalesrv.txt

```
DEFINE NAMELIST (CLUSTERS) NAMES(SALES, Q.SALES)
DEFINE CHANNEL(Q.SALES.SALER1) CHLTYPE(CLUSSDR) CONNAME('localhost(1416)')
CLUSTER(Q.SALES) REPLACE
DEFINE CHANNEL(Q.SALES.SAVESRV) CHLTYPE(CLUSRCVR) CONNAME('localhost(1419)')
CLUSTER(Q.SALES) REPLACE
ALTER QLOCAL (SALES) CLUSTER(' ') CLUSNL(CLUSTERS)
```

chggate.txt

```
ALTER NAMELIST(ALL) NAMES(SALES, FINANCE, Q.SALES)
ALTER QMGR DEFCLXQ(CHANNEL)
```

```
DEFINE CHANNEL(Q.SALES.SALER1) CHLTYPE(CLUSSDR)  CONNAME('localhost(1416)')
CLUSTER(Q.SALES) REPLACE
DEFINE CHANNEL(Q.SALES.GATE)  CHLTYPE(CLUSRCVR) CONNAME('localhost(1420)')
CLUSTER(Q.SALES) REPLACE
DEFINE QLOCAL (XMITQ.Q.SALES.SALESRV) USAGE(XMITQ) CLCHNAME(Q.SALES.SALESRV) REPLACE
DEFINE QLOCAL (XMITQ.SALES)     USAGE(XMITQ) CLCHNAME(SALES.*)  REPLACE
DEFINE QLOCAL (XMITQ.FINANCE)    USAGE(XMITQ) CLCHNAME(FINANCE.*) REPLACE
```

4. 从 SALES 集群中除去 SALES 队列。

   在 第 50 页的图 15 中运行 **MQSC** 命令:

```
ALTER QLOCAL(SALES) CLUSTER('Q.SALES') CLUSNL(' ')
```

   图 *15:* 从销售集群中除去队列管理器 *SALESRV* 上的销售队列

5. 将通道切换到新的传输队列。

   需要停止并启动 GATE 队列管理器正在使用的所有通道。 要使用最少数量的命令执行此操作，请停止并启动队列管理器

```
endmqm -i GATE
strmqm GATE
```

下一步做什么

1. 重新运行样本请求程序以验证新配置是否有效; 请参阅步骤 第 49 页的『2』
2. 监视流经 GATE 队列管理器上所有集群传输队列的消息:
   a. 更改要开启队列监视的每个集群传输队列的定义。

```
ALTER QLOCAL(SYSTEM.CLUSTER.TRANSMIT.
name) STATQ(ON)
```

   b. 检查队列管理器统计信息监视是否为 OFF，以最小化输出，并将监视时间间隔设置为较低的值以方便地执行多个测试。

```
ALTER QMGR STATINT(60) STATCHL(OFF) STATQ(OFF) STATMQI(OFF) STATACLS(OFF)
```

   c. 重新启动 GATE 队列管理器。
   d. 运行样本请求程序几次以验证是否有相同数量的消息流经 SYSTEM.CLUSTER.TRANSMIT.Q.SALES.SALESRV 和 SYSTEM.CLUSTER.TRANSMIT.QUEUE。 请求流经 SYSTEM.CLUSTER.TRANSMIT.Q.SALES.SALESRV，应答流经 SYSTEM.CLUSTER.TRANSMIT.QUEUE。

```
amqsmon -m GATE -t statistics
```

   e. 几个时间间隔内的结果如下所示:

```
C:\Documents and Settings\Admin>amqsmon -m GATE -t statistics
MonitoringType: QueueStatistics
QueueManager: 'GATE'
IntervalStartDate: '2012-02-27'
IntervalStartTime: '14.59.20'
IntervalEndDate: '2012-02-27'
IntervalEndTime: '15.00.20'
CommandLevel: 700
ObjectCount: 2
QueueStatistics: 0
```

```
QueueName: 'SYSTEM.CLUSTER.TRANSMIT.QUEUE'
CreateDate: '2012-02-24'
CreateTime: '15.58.15'
...
Put1Count: [0, 0]
Put1FailCount: 0
PutBytes: [435, 0]
GetCount: [1, 0]
GetBytes: [435, 0]
...
QueueStatistics: 1
QueueName: 'SYSTEM.CLUSTER.TRANSMIT.Q.SALES.SAVESRV'
CreateDate: '2012-02-24'
CreateTime: '16.37.43'
...
PutCount: [1, 0]
PutFailCount: 0
Put1Count: [0, 0]
Put1FailCount: 0
PutBytes: [435, 0]
GetCount: [1, 0]
GetBytes: [435, 0]
...
MonitoringType: QueueStatistics
QueueManager: 'GATE'
IntervalStartDate: '2012-02-27'
IntervalStartTime: '15.00.20'
IntervalEndDate: '2012-02-27'
IntervalEndTime: '15.01.20'
CommandLevel: 700
ObjectCount: 2
QueueStatistics: 0
QueueName: 'SYSTEM.CLUSTER.TRANSMIT.QUEUE'
CreateDate: '2012-02-24'
CreateTime: '15.58.15'
...
PutCount: [2, 0]
PutFailCount: 0
Put1Count: [0, 0]
Put1FailCount: 0
PutBytes: [863, 0]
GetCount: [2, 0]
GetBytes: [863, 0]
...
QueueStatistics: 1
QueueName: 'SYSTEM.CLUSTER.TRANSMIT.Q.SALES.SAVESRV'
CreateDate: '2012-02-24'
CreateTime: '16.37.43'
...
PutCount: [2, 0]
PutFailCount: 0
Put1Count: [0, 0]
Put1FailCount: 0
PutBytes: [863, 0]
GetCount: [2, 0]
GetBytes: [863, 0]
...
2 Records Processed.
```

在第一个时间间隔内发送了一个请求和应答消息，在第二个时间间隔内发送了两个请求和应答消息。您可以推断请求消息已放置在 SYSTEM.CLUSTER.TRANSMIT.Q.SALES.SAVESRV 上，应答消息已放置在 SYSTEM.CLUSTER.TRANSMIT.QUEUE 上。

集群: 切换集群传输队列
规划如何使对现有生产队列管理器的集群传输队列所作的更改生效。

开始之前

如果减少切换过程必须传输到新传输队列的消息数，那么切换将更快地完成。 请阅读 将集群发送方通道切换到其他传输队列的过程如何工作，以了解在继续操作之前尝试清空传输队列的原因。

关于此任务

您可以选择两种方法使对集群传输队列的更改生效。

1. 让队列管理器自动进行更改。 这是缺省值。 当集群发送方通道下次启动时，队列管理器会切换具有暂挂传输队列更改的集群发送方通道。
2. 手动进行更改。 您可以在集群发送方通道停止时对其进行更改。 在集群发送方通道启动之前，可以将其从一个集群传输队列切换到另一个集群传输队列。

在决定选择两个选项中的哪一个时，您会考虑哪些因素，以及如何管理交换机?

过程

- 选项 1: 让队列管理器自动进行更改; 请参阅 第 53 页的『将活动集群发送方通道切换到另一组集群传输队列』。

  如果希望队列管理器为您进行切换，请选择此选项。

  描述此选项的另一种方法是，队列管理器在不强制通道停止的情况下切换集群发送方通道。 您可以选择强制通道停止，然后启动通道，以使交换机更快发生。 交换机在通道启动时启动，并在通道运行时运行，这与选项 2 不同。 在选项 2 中，交换机在通道停止时发生。

  如果通过让交换机自动发生来选择此选项，那么切换过程将在集群发送方通道启动时启动。 如果通道未停止，那么它将在变为不活动状态后启动 (如果有要处理的消息)。 如果通道已停止，请使用 START CHANNEL 命令将其启动。
  一旦在通道所服务的传输队列上没有针对集群发送方通道的剩余消息，交换机进程就会完成。 一旦出现这种情况，集群发送方通道的新到达消息将直接存储在新的传输队列上。 在此之前，消息存储在旧的传输队列上，切换过程将消息从旧的传输队列传输到新的传输队列。 集群发送方通道在整个切换过程中转发来自新集群传输队列的消息。
  交换机进程何时完成取决于系统的状态。 如果要在维护窗口中进行更改，请事先评估切换过程是否会及时完成。 它是否会及时完成取决于等待从旧传输队列传输的消息数是否达到零。

  第一种方法的优点是它是自动的。 一个缺点是，如果进行配置更改的时间仅限于维护窗口，那么您必须确信可以控制系统在维护窗口内完成切换过程。 如果无法确定，那么选项 2 可能是更好的选择。

- 选项 2: 手动进行更改; 请参阅 第 54 页的『将已停止的集群发送方通道切换到另一个集群传输队列』。

  如果要手动控制整个切换过程，或者要切换已停止或不活动的通道，请选择此选项。 如果要切换几个集群发送方通道，并且要在维护窗口期间执行切换，那么这是一个不错的选择。

  此选项的替代描述是表示在集群发送方通道停止时切换集群发送方通道。

  如果选择此选项，那么您可以完全控制交换机的发生时间。
  您可以确定在固定时间内，在维护窗口内完成切换过程。 切换所花费的时间取决于必须从一个传输队列传输到另一个传输队列的消息数。 如果消息不断到达，那么进程可能需要一段时间才能传输所有消息。
  您可以选择在不从旧传输队列传输消息的情况下切换通道。 交换机为 “即时”。
  当您重新启动集群发送方通道时，它将开始处理新分配给它的传输队列上的消息。

  第二种方法的优点是您可以控制切换过程。 缺点是您必须识别要切换的集群发送方通道，运行必要的命令，并解决可能阻止集群发送方通道停止的任何不确定通道。

相关概念

如何选择要使用的集群传输队列类型
如何在不同的集群传输队列配置选项之间进行选择。

将集群发送方通道切换到其他传输队列的过程如何工作

相关任务

集群: 多个集群传输队列的示例配置
在此任务中，您将应用步骤以将多个集群传输队列规划到三个重叠的集群。 这些要求是将消息流与所有其他消息流分离到一个集群队列，并将不同集群的消息存储在不同的集群传输队列上。

将活动集群发送方通道切换到另一组集群传输队列
此任务为您提供了三个用于切换活动集群发送方通道的选项。 一个选项是让队列管理器自动进行切换，这不会影响正在运行的应用程序。 其他选项包括手动停止和启动通道，或者重新启动队列管理器。

开始之前

更改集群传输队列配置。 您可以更改 **DEFCLXQ** 队列管理器属性，或者添加或修改传输队列的 **CLCHNAME** 属性。

如果减少切换过程必须传输到新传输队列的消息数，那么切换将更快地完成。 请阅读 将集群发送方通道切换到其他传输队列的过程如何工作 ，以了解在继续操作之前尝试清空传输队列的原因。

关于此任务

使用任务中的步骤作为制定您自己的计划以进行集群传输队列配置更改的基础。

过程

1. 可选： 记录当前通道状态

   记录正在为集群传输队列提供服务的当前通道和已保存通道的状态。 以下命令显示与系统集群传输队列关联的状态。 添加您自己的命令以显示与已定义的集群传输队列相关联的状态。 使用约定 (例如 `XMITQ.ChannelName`) 来命名您定义的集群传输队列，以便于显示这些传输队列的通道状态。

   ```
   DISPLAY CHSTATUS(*) WHERE(XMITQ LK 'SYSTEM.CLUSTER.TRANSMIT.*')
   DISPLAY CHSTATUS(*) SAVED WHERE(XMITQ LK 'SYSTEM.CLUSTER.TRANSMIT.*')
   ```

2. 交换机传输队列。

   - 不执行任何操作。 当集群发送方通道在停止或不活动后重新启动时，队列管理器会切换这些集群发送方通道。

     如果您没有更改队列管理器配置的规则或顾虑，请选择此选项。 正在运行的应用程序不受更改影响。
   - 重新启动队列管理器。 将根据需要停止并自动重新启动所有集群发送方通道。

     选择此选项以立即启动所有更改。 在队列管理器关闭并重新启动时，正在运行的应用程序将被队列管理器中断。
   - 停止各个集群发送方通道并将其重新启动。

     选择此选项可立即更改几个通道。 正在运行的应用程序在停止消息通道和再次启动消息通道之间的消息传输中迁到短暂延迟。 集群发送方通道保持运行，但在您停止该通道期间除外。 在切换过程中，将消息传递到旧的传输队列，通过切换过程传输到新的传输队列，并通过集群发送方通道从新的传输队列转发。

3. 可选： 在通道切换时监视通道

   显示交换机期间的通道状态和传输队列深度。 以下示例显示系统集群传输队列的状态。

   ```
   DISPLAY CHSTATUS(*) WHERE(XMITQ LK 'SYSTEM.CLUSTER.TRANSMIT.*')
   DISPLAY CHSTATUS(*) SAVED WHERE(XMITQ LK 'SYSTEM.CLUSTER.TRANSMIT.*')
   DISPLAY QUEUE('SYSTEM.CLUSTER.TRANSMIT.*') CURDEPTH
   ```

4. 可选： 监视消息 AMQ7341 **通道** *ChannelName* **的传输队列已从队列** *QueueName* **切换到写入队列管理器错误日志的** *QueueName* 。

将已停止的集群发送方通道切换到另一个集群传输队列
如果选择手动进行更改，那么在集群发送方通道停止时对其进行更改，并在集群发送方通道启动前将其从一个集群传输队列切换到另一个集群传输队列。

开始之前

您可以进行一些配置更改，现在希望使这些更改生效，而不启动受影响的集群发送方通道。 或者，作为任务中的其中一个步骤进行所需的配置更改。

如果减少切换过程必须传输到新传输队列的消息数，那么切换将更快地完成。 请阅读 将集群发送方通道切换到其他传输队列的过程如何工作 ，以了解在继续操作之前尝试清空传输队列的原因。

关于此任务

此任务切换由已停止或不活动的集群发送方通道提供服务的传输队列。 您可以执行此任务，因为集群发送方通道已停止，并且您希望立即切换其传输队列。 例如，由于某种原因，集群发送方通道未启动，或者存在一些其他配置问题。 要解决此问题，您决定创建集群发送方通道，并将旧集群发送方通道的传输队列与您定义的新集群发送方通道相关联。

更有可能的情况是，您希望控制何时执行集群传输队列的重新配置。 要完全控制重新配置，请停止通道，更改配置，然后切换传输队列。

过程

1. 停止要切换的通道
   a) 停止要切换的任何正在运行或不活动的通道。 停止不活动的集群发送方通道会阻止它在您进行配置更改时启动。

```
STOP CHANNEL(ChannelName) MODE(QUIESCSE) STATUS(STOPPED)
```

2. 可选： 进行配置更改。

   例如，请参阅第 45 页的『集群: 多个集群传输队列的示例配置』。
3. 将集群发送方通道切换到新的集群传输队列。

   Multi 在 多平台上，发出以下命令:

```
runswchl -m QmgrName -c ChannelName
```

   z/OS 在 z/OS 上，使用 CSQUTIL 命令的 SWITCH 函数来切换消息或监视正在发生的情况。 使用以下命令。

```
SWITCH CHANNEL(channel_name) MOVEMSGS(YES)
```

   有关更多信息，请参阅 SWITCH 函数。

   runswchl 或 CSQUTIL SWITCH 命令将旧传输队列上的任何消息传输到新的传输队列。 当此通道的旧传输队列上的消息数达到零时，交换机将完成。 该命令是同步的。 此命令在切换过程中将进度消息写入窗口。

   在传输阶段，将以集群发送方通道为目标的现有消息和新消息传输到新的传输队列。

   由于集群发送方通道已停止，因此消息会在新的传输队列上构建。 将已停止的集群发送方通道与 第 53 页的『将活动集群发送方通道切换到另一组集群传输队列』中的步骤 第 53 页的『2』 进行对比。 在该步骤中，集群发送方通道正在运行，因此消息不一定会在新的传输队列上构建。
4. 可选： 在通道切换时监视通道

在另一个命令窗口中，显示切换期间的传输队列深度。 以下示例显示系统集群传输队列的状态。

```
DISPLAY QUEUE('SYSTEM.CLUSTER.TRANSMIT.*') CURDEPTH
```

5. 可选： 监视消息 AMQ7341 **通道** *ChannelName* **的传输队列已从队列** *QueueName* **切换到写入队列管理器错误日志的** *QueueName* 。
6. 重新启动已停止的集群发送方通道。

   通道不会自动启动，因为您已停止这些通道，并将其置于 STOPPED 状态。

```
START CHANNEL(ChannelName)
```

相关参考
伦斯湖
解析通道
停止通道

集群*:* 迁移和修改最佳实践
本主题提供有关规划和管理 IBM MQ 集群的指导。 此信息是基于测试和客户反馈的指南。

1. 第 55 页的『在集群中移动对象』 (在集群内移动对象的最佳实践，而不安装任何修订包或新版本的 IBM MQ)。
2. 第 56 页的『升级和维护安装』 (在应用维护或升级以及测试新体系结构时保持工作集群体系结构正常运行的最佳实践)。

在集群中移动对象

应用程序及其队列

当您必须将托管在一个队列管理器上的队列实例移动到另一个队列管理器上时，可以使用工作负载均衡参数来确保平稳过渡。

创建要在其中进行新托管的队列实例，但使用集群工作负载均衡设置继续向原始实例发送消息，直到应用程序准备好切换为止。 这是通过以下步骤实现的:

1. 将现有队列的 **CLWLRANK** 属性设置为高值，例如 5。
2. 创建队列的新实例，并将其 **CLWLRANK** 属性设置为零。
3. 完成新系统的任何进一步配置，例如，针对队列的新实例部署和启动使用应用程序。
4. 将新队列实例的 **CLWLRANK** 属性设置为高于原始实例，例如 9。
5. 允许原始队列实例处理系统中的任何排队消息，然后删除该队列。

移动整个队列管理器

如果队列管理器位于同一主机上，但 IP 地址正在更改，那么此过程如下所示:

- 当正确使用 DNS 时，可以帮助简化该过程。 有关通过设置 连接名称 (CONNAME) 通道属性使用 DNS 的信息，请参阅 ALTER CHANNEL。
- 如果移动完整存储库，请确保在进行更改之前至少有一个其他完整存储库正在平稳运行 (例如，通道状态没有问题)。
- 使用 SUSPEND QMGR 命令暂挂队列管理器以避免流量堆积。
- 修改计算机的 IP 地址。 如果 CLUSRCVR 通道定义使用 CONNAME 字段中的 IP 地址，请修改此 IP 地址项。 可能需要清空 DNS 高速缓存以确保到处都有可用的更新。
- 当队列管理器重新连接到完整存储库时，通道自动定义会自动解析自身。
- 如果队列管理器托管了完整存储库，并且 IP 地址发生更改，请务必确保尽快切换分区以将任何手动定义的 CLUSSDR 通道指向新位置。 在执行此切换之前，这些队列管理器可能只能与剩余的 (未更改的) 完整存储库联系，并且可能会看到有关不正确通道定义的警告消息。
- 使用 RESUME QMGR 命令恢复队列管理器。

如果必须将队列管理器移至新主机，那么可以复制队列管理器数据并从备份复原。 但是，除非没有其他选项，否则建议不要执行此过程; 最好在新机器上创建队列管理器，并如上一节中所述复制队列和应用程序。 这种情况提供了一种平滑的翻转/回滚机制。

如果您决心使用备份来移动完整的队列管理器，请遵循以下最佳实践:

- 将整个进程视为来自备份的队列管理器复原，应用您通常用于操作系统环境的系统恢复的任何进程。
- 迁移后使用 **REFRESH CLUSTER** 命令来废弃所有本地持有的集群信息 (包括任何不确定的自动定义通道) ，并强制对其进行重建。

  注: 对于大型集群，使用 **REFRESH CLUSTER** 命令可能会对正在运行的集群造成干扰，此外每隔 27 天在集群对象向所有相关队列管理器自动发送状态更新时也可能有干扰。 请参阅在大型集群中刷新可能会影响集群的性能和可用性。

创建队列管理器并从集群中的现有队列管理器复制设置 (如本主题中所述) 时，切勿将两个不同的队列管理器视为实际相同。 特别是，不要为新队列管理器提供相同的队列管理器名称和 IP 地址。 尝试 "删除" 替换队列管理器是导致 IBM MQ 集群中出现问题的常见原因。 高速缓存期望接收包括 **QMID** 属性在内的更新，并且状态可能已损坏。

如果意外创建了两个同名的不同队列管理器，那么建议使用 RESET CLUSTER **QMID** 命令从集群中弹出不正确的条目。

升级和维护安装

避免所谓的大爆炸场景 (例如，停止所有集群和队列管理器活动，对所有队列管理器应用所有升级和维护，然后同时启动所有内容)。 集群设计为仍与多个版本的队列管理器共存，因此建议采用规划良好的分阶段维护方法。

具有备份计划:

- 您是否进行了备份?
- 避免立即使用新的集群功能: 请等待您确定所有队列管理器都已升级到新级别，并确定您不会将它们中的任何一个回滚。 在某些队列管理器仍处于较早级别的集群中使用新的集群功能可能会导致未定义的行为。

  存储库将其接收到的记录存储在其自己的版本中。 如果它接收到的记录处于更高版本，那么在存储记录时将废弃更高版本的属性。 接收有关 IBM MQ 9.4 队列管理器的信息的 IBM MQ 9.3 队列管理器仅存储 IBM MQ 9.3 信息。 接收 IBM MQ 9.3 记录的 IBM MQ 9.4 存储库将存储更高版本中引入的属性的缺省值。 缺省值定义未包含在其接收的记录中的属性值。

首先迁移完整存储库。 虽然他们可以转发他们不了解的信息，但他们不能坚持下去，因此除非绝对必要，否则并不是建议的方法。 有关更多信息，请参阅 队列管理器集群迁移。

集群：使用 *REFRESH CLUSTER* 最佳实践
使用 **REFRESH CLUSTER** 命令可废弃有关集群的所有本地保存的信息，并从集群中的完整存储库重建该信息。 您不应该使用此命令，除非在特殊情况下。 如果确实需要使用它，那么有关如何使用它有特殊注意事项。 此信息是基于测试和客户反馈的指南。

仅在确实需要时运行 REFRESH CLUSTER

IBM MQ 集群技术可确保对集群配置的任何更改 (例如对集群队列的更改) 自动为需要了解信息的集群成员所知。 没有必要采取进一步的行政步骤来实现这种信息传播。

如果此类信息未到达需要该信息的集群中的队列管理器，例如，当应用程序首次尝试将其打开时，集群中的另一个队列管理器不知道集群队列，那么意味着集群基础结构中存在问题。 例如，可能无法在队列管理器与完整存储库队列管理器之间启动通道。 因此，必须对观察到的任何不一致情况进行调查。 如果可能，请在不使用 **REFRESH CLUSTER** 命令的情况下解决此情况。

在此产品文档中其他位置记录的极少数情况下，或者当 IBM 支持人员请求时，您可以使用 **REFRESH CLUSTER** 命令来废弃有关集群的所有本地保存的信息，并从集群中的完整存储库重建该信息。

在大型集群中刷新可能会影响集群的性能和可用性

使用 **REFRESH CLUSTER** 命令可能会对正在进行的集群造成干扰，例如，在处理队列管理器集群资源的重新传播时，为完整存储库创建突然增加的工作。 如果在大型集群 (即，数百个队列管理器) 中刷新，那么应避免在日常工作中使用该命令 (如果可能)，并使用替代方法来更正特定不一致情况。 例如，如果未在集群中正确传播集群队列，那么更新集群队列定义的初始调查技术 (例如，更改其描述) 将在集群中重新传播队列配置。 此过程可帮助确定问题并可能解决临时不一致问题。

如果无法使用替代方法，并且必须在大型集群中运行 **REFRESH CLUSTER**，那么应在非高峰时间或维护时段执行此操作，以避免影响用户工作负载。 您还应避免在单个批处理中刷新大型集群，而是按 第 57 页的『避免集群对象发送自动更新时出现性能和可用性问题』中所述错开活动。

避免集群对象发送自动更新时出现性能和可用性问题

在队列管理器上定义新集群对象之后，将从定义开始每 27 天生成此对象的更新，并将此更新发送到集群中的每个完整存储库，然后再发送到任何其他相关队列管理器。 向队列管理器发出 **REFRESH CLUSTER** 命令时，将在指定集群中本地定义的所有对象上重置此自动更新的时钟。

如果在单个批处理中刷新大型集群 (即，数百个队列管理器)，或者在其他情况下 (例如，从配置备份重新创建系统)，那么在 27 天后，所有这些队列管理器都将同时向完整存储库重新发布其所有对象定义。 这可能再次导致系统运行速度显着降低，甚至变为不可用，直到所有更新都完成为止。 因此，当您必须在大型集群中刷新或重新创建多个队列管理器时，应该将活动错开几个小时或几天，以便后续自动更新不会定期影响系统性能。

系统集群历史记录队列

执行 **REFRESH CLUSTER** 时，队列管理器将在刷新之前生成集群状态的快照，并将其存储在 SYSTEM.CLUSTER.HISTORY.QUEUE (SCHQ) (如果在队列管理器上定义) 上。 此快照仅用于 IBM 服务，以防以后发生系统问题。

缺省情况下，在启动时在分布式队列管理器上定义 SCHQ。 对于 z/OS 迁移，必须手动定义 SCHQ。

SCHQ 上的消息将在三个月后到期。

相关概念

第 86 页的『针对发布/预订集群的 REFRESH CLUSTER 注意事项』
发出 **REFRESH CLUSTER** 命令会导致队列管理器临时废弃本地保存的有关集群的信息，包括任何集群主题及其关联的代理预订。

相关参考

运行 REFRESH CLUSTER 时发现的应用程序问题
MQSC 命令参考：REFRESH CLUSTER

集群: 可用性，多实例和灾难恢复
本主题提供有关规划和管理 IBM MQ 集群的指导。 此信息是基于测试和客户反馈的指南。

IBM MQ 集群本身不是高可用性解决方案，但在某些情况下，可以使用它来提高使用 IBM MQ 的服务的可用性，例如，在不同的队列管理器上具有多个队列实例。 本部分提供了有关确保 IBM MQ 基础结构尽可能高可用性的指导信息，以便可以在此类体系结构中使用该基础结构。

注: 其他高可用性和灾难恢复解决方案可用于 IBM MQ，请参阅 配置高可用性，恢复和重新启动。

集群资源的可用性
通常建议维护两个完整存储库的原因是，丢失一个存储库对于集群的顺利运行并不重要。 即使两者都变得不可用，对于部分存储库持有的现有知识也有 60 天的宽限期，尽管在此事件中没有新的或先前未访问的资源 (例如，队列) 可用。

使用集群来提高应用程序可用性
通过使用队列和应用程序的多个实例，集群可以帮助设计高可用性应用程序 (例如请求/响应类型服务器应用程序)。 如果需要，优先级属性可以优先使用 "活动" 应用程序，除非例如队列管理器或通道变为不可用。 这对于在发生问题时快速切换以继续处理新消息非常强大。

但是，传递到集群中特定队列管理器的消息仅保留在该队列实例上，并且在恢复该队列管理器之前不可用于处理。 因此，为了实现真正的数据高可用性，您可能需要考虑其他技术，例如多实例队列管理器。

多实例队列管理器

软件高可用性 (多实例) 是用于保持现有消息可用的内置产品。 请参阅 将 IBM MQ 与高可用性配置配合使用， 创建多实例队列管理器以及以下部分以获取更多信息。 只要集群中的所有队列管理器至少正在运行 IBM WebSphere MQ 7.0.1，就可以使用此技术使集群中的任何队列管理器具有高可用性。 如果集群中的任何队列管理器处于先前级别，那么当它们故障转移到辅助 IP 时，可能会丢失与多实例队列管理器的连接。

正如本主题中先前所讨论的那样，只要配置了两个完整的存储库，它们几乎按其性质高度可用。 如果需要，可以将 IBM MQ 软件高可用性/多实例队列管理器用于完整存储库。 没有强烈的理由使用这些方法，事实上，对于临时中断，这些方法可能会在故障转移期间造成额外的性能成本。 建议不要使用软件 HA 而不是运行两个完整存储库，因为例如，在发生单个通道中断时，它不一定会故障转移，但可能会导致部分存储库无法查询集群资源。

灾难恢复

灾难恢复 (例如，从存储队列管理器数据的磁盘损坏时进行恢复) 很难很好地执行; IBM MQ 可以提供帮助，但它无法自动执行此操作。 IBM MQ 中唯一的 "true" 灾难恢复选项 (不包括任何操作系统或其他底层复制技术) 是从备份复原。 在以下情况下，需要考虑一些特定于集群的点:

- 测试灾难恢复方案时请小心。 例如，如果测试备份队列管理器的操作，那么在使它们在同一网络中联机时要小心，因为有可能意外地加入实时集群，并通过托管与实时集群队列管理器中相同的指定队列来开始 "窃取" 消息。
- 灾难恢复测试不得干扰正在运行的实时集群。 避免干扰的方法包括:
  - 在防火墙级别完成网络分离或分离。
  - z/OS 未启动通道启动或 z/OS **chinit** 地址空间。
  - 在发生实际灾难恢复方案之前，或者除非发生实际灾难恢复方案，否则不会向灾难恢复系统发放实时 TLS 证书。
- 在集群中复原队列管理器的备份时，备份可能与集群的其余部分不同步。 **REFRESH CLUSTER** 命令可以解析更新并与集群同步，但 **REFRESH CLUSTER** 命令必须用作最后的手段。 请参阅第 56 页的『集群：使用 REFRESH CLUSTER 最佳实践』。 查看任何内部流程文档和 IBM MQ 文档，以了解在使用该命令之前是否错过了简单的步骤。
- 对于任何恢复，应用程序都必须处理重放和数据丢失。 必须决定是将队列清除到已知状态，还是在其他地方有足够的信息来管理重放。

规划分布式发布/预订网络

您可以创建队列管理器网络，在该网络中，在一个队列管理器上创建的预订将接收由连接到网络中另一个队列管理器的应用程序发布的匹配消息。 要选择合适的拓扑，需要考虑您对手动控制，网络大小，更改频率，可用性和可伸缩性的需求。

开始之前

此任务假定您了解分布式发布/预订网络是什么以及它们的工作方式。 有关技术概述，请参阅 分布式发布/预订网络。

关于此任务

发布/预订网络有三种基本拓扑:

- 直接路由集群
- 主题主机路由集群
- 层次结构

对于前两个拓扑，起始点是 IBM MQ 集群配置。 可以使用或不使用集群创建第三个拓扑。 请参阅第 17 页的『规划分布式队列和集群』，以获取有关规划底层队列管理器网络的信息。

直接路由集群 是在集群已存在时要配置的最简单拓扑。 您在任何队列管理器上定义的任何主题都将自动在集群中的每个队列管理器上可用，发布将直接从发布应用程序连接的任何队列管理器路由到存在匹配预订的每个队列管理器。 这种简单的配置依赖于 IBM MQ 在集群中的每个队列管理器之间保持高级别的信息共享和连接。 对于小型和简单的网络 (即，少量队列管理器以及一组相当静态的发布者和订户) ，这是可以接受的。 但是，在更大或更动态的环境中使用时，开销可能会令人望而却步。 请参阅第 62 页的『发布/预订集群中的直接路由』。

主题主机路由集群 通过使集群中的任何队列管理器上定义的任何主题自动在集群中的每个队列管理器上可用，提供与直接路由集群相同的优势。 但是，主题主机路由集群要求您仔细选择托管每个主题的队列管理器，因为该主题的所有信息和发布都通过这些主题主机队列管理器传递。 这意味着系统不必维护所有队列管理器之间的通道和信息流。 但是，这也意味着发布可能不再直接发送到订户，而是可以通过主题主机队列管理器进行路由。 由于这些原因，系统上可能会有额外的负载，特别是在托管主题的队列管理器上，因此需要仔细规划拓扑。 对于包含许多队列管理器或托管一组动态发布者和订户 (即，频繁添加或除去的发布者或订户) 的网络，此拓扑特别有效。 可以定义其他主题主机以提高路径可用性并水平缩放发布工作负载。 请参阅第 67 页的『发布/预订集群中的主题主机路由』。

层次结构 需要设置最手动的配置，并且是要修改的最难的拓扑。 您必须手动配置层次结构中每个队列管理器与其直接关系之间的关系。 配置关系后，将发布 (与前两种拓扑一样) 路由到层次结构中其他队列管理器上的预订。 发布使用层次结构关系进行路由。 这允许将非常特定的拓扑配置为满足不同的需求，但这也可能导致发布需要许多 “中继段” 通过中间队列管理器来访问预订。 对于发布，始终只有一个通过层次结构的路由，因此每个队列管理器的可用性至关重要。 通常只有在无法配置单个集群的情况下，才会首选层次结构; 例如，跨多个组织时。 请参阅第 87 页的『在发布/预订层次结构中进行路由』。

必要时，可将上述三种拓扑组合起来，以解决特定的地形要求。 有关示例，请参阅 组合多个集群的主题空间。

要为分布式发布/预订网络选择合适的拓扑，您需要考虑以下广泛问题:

- 您的网络将有多大?
- 您需要对其配置进行多少手动控制?
- 无论在主题和预订方面，还是在队列管理器方面，系统都将具有多大的动态?
- 您的可用性和可伸缩性需求是什么?
- 所有队列管理器可以直接相互连接吗?

过程

- 估算网络需要的大小。
  a) 估算需要多少主题。
  b) 估算您期望拥有的发布者和订户数量。
  c) 估算将有多少队列管理器参与发布/预订活动。

  另请参阅第 76 页的『发布/预订集群: 最佳实践』，特别是以下部分:

  – 如何调整系统大小
  – 限制发布/预订活动中涉及的集群队列管理器数的原因
  – 如何决定要集群的主题

  如果您的网络将具有许多队列管理器，并处理许多发布者和订户，那么您可能需要使用主题主机路由集群或层次结构。 直接路由集群几乎无需手动配置，可以成为小型或静态网络的良好解决方案。
- 请考虑您需要对每个主题，发布者或订户托管的队列管理器进行多少手动控制。
  a) 请考虑某些队列管理器的能力是否低于其他队列管理器。
  b) 请考虑到某些队列管理器的通信链路是否比其他队列管理器更脆弱。
  c) 确定您期望主题具有许多出版物和少量订户的案例。
  d) 确定您期望主题具有许多订户和少量出版物的案例。

  在所有拓扑中，发布将传递到其他队列管理器上的预订。 在直接路由的集群中，这些发布将采用最短的预订路径。 在主题主机路由集群或层次结构中，您可以控制发布所采用的路径。 如果队列管理器的功能

不同，或者具有不同的可用性和连接级别，那么您可能希望将特定工作负载分配给特定队列管理器。 您可以使用主题主机路由集群或层次结构来执行此操作。

在所有拓扑中，尽可能将发布应用程序与预订放在同一队列管理器上，从而最大程度地减少开销并实现性能最大化。 对于主题主机路由集群，请考虑将发布者或订户放在托管该主题的队列管理器上。 这将除去队列管理器之间的任何额外“中继段”，以将发布传递给订户。 在主题具有许多发布者和很少的订户，或者具有许多订户和很少的发布者的情况下，此方法特别有效。 例如，请参阅 使用集中发布程序或订户进行主题主机路由。

另请参阅第 76 页的『发布/预订集群: 最佳实践』，特别是以下部分:

– 如何决定要集群的主题
– 发布程序和预订位置

- 请考虑网络活动的动态程度。
  a) 估算在不同主题上添加和除去订户的频率。

  每当从队列管理器添加或除去预订，并且该预订是该特定主题字符串的第一个或最后一个预订时，都会将该信息传达给拓扑中的其他队列管理器。 在直接路由的集群和层次结构中，此预订信息将传播到拓扑中的每个队列管理器，无论它们是否具有主题上的发布程序。 如果拓扑由许多队列管理器组成，那么这可能是显着的性能开销。 在主题主机路由集群中，此信息仅传播到那些托管映射到预订的主题字符串的集群主题的队列管理器。

  另请参阅第 76 页的『发布/预订集群: 最佳实践』的 预订更改和动态主题字符串 部分。

  注: 在非常动态的系统中，许多唯一主题字符串的集合正在快速且持续地进行更改，因此最好将模型切换到“随时随地发布”方式。 请参阅 发布/预订网络中的预订性能。

  b) 请考虑队列管理器在拓扑中的动态程度。

  层次结构要求将拓扑中队列管理器中的每个更改手动插入或从层次结构中除去，并在层次结构中的较高级别更改队列管理器时小心谨慎。 层次结构中的队列管理器通常也使用手动配置的通道连接。 您必须维护这些连接，在从层次结构中添加和除去队列管理器时添加和除去通道。

  在发布/预订集群中，队列管理器会自动连接到首次加入集群时所需的任何其他队列管理器，并自动了解主题和预订。
- 考虑您的路由可用性和发布流量可伸缩性需求。
  a) 决定是否需要始终具有从发布队列管理器到预订队列管理器的可用路由，即使队列管理器不可用时也是如此。
  b) 请考虑您需要网络的可伸缩性。 确定发布流量级别是否过高，无法通过单个队列管理器或通道进行路由，以及该级别的发布流量是必须由单个主题分支处理，还是可以跨多个主题分支进行传播。
  c) 请考虑是否需要维护消息排序。

  由于直接路由的集群将消息从发布队列管理器直接发送到预订队列管理器，因此您无需考虑沿路由的中间队列管理器的可用性。 同样，不考虑对中间队列管理器进行缩放。 但是，如前所述，自动维护集群中所有队列管理器之间的通道和信息流的开销会显着影响性能，尤其是在大型或动态环境中。

  可以针对各个主题调整主题主机路由集群。 您可以确保在不同的队列管理器上定义具有大量发布工作负载的主题树的每个分支，并且确保每个队列管理器具有足够的性能，并且可用于该主题树的该分支的预期工作负载。 您还可以通过在多个队列管理器上定义每个主题来进一步提高可用性和水平缩放。 这允许系统围绕不可用的主题主机队列管理器进行路由，并使它们之间的工作负载均衡发布流量。 但是，在多个队列管理器上定义给定主题时，还会引入以下约束:

  – 在发布之间丢失消息排序。
  – 不能使用保留发布。 请参阅第 85 页的『发布/预订集群中保留的发布的设计注意事项』。

  无法通过多个路径配置层次结构中的路由的高可用性或可伸缩性。

  另请参阅第 76 页的『发布/预订集群: 最佳实践』的 发布流量 部分。
- 根据这些计算，使用提供的链接来帮助您决定是使用主题主机路由集群，直接路由集群，层次结构还是这些拓扑的混合。

下一步做什么

您现在已准备好配置分布式发布/预订网络。

相关任务

配置队列管理器集群

配置分布式队列

配置发布/预订集群

将队列管理器连接到发布/预订层次结构

设计发布/预订集群

有两种基本发布/预订集群拓扑: 直接路由 和 主题主机路由。 每一个都有不同的好处。 设计发布/预订集群时，请选择最符合预期网络需求的拓扑。

有关两个发布/预订集群拓扑的概述，请参阅 发布/预订集群。 要帮助您评估网络需求，请参阅 第 58 页的『规划分布式发布/预订网络』 和 第 76 页的『发布/预订集群: 最佳实践』。

通常，这两种集群拓扑都具有以下优点:

- 基于点到点集群拓扑的简单配置。
- 自动处理加入和离开集群的队列管理器。
- 通过添加额外的队列管理器并在这些队列管理器和发布程序之间分发额外的预订和发布程序，简化其他预订和发布程序的扩展。

但是，随着需求变得更加具体，这两种拓扑具有不同的优点。

直接路由的发布/预订集群

通过直接路由，集群中的任何队列管理器都会将发布从已连接的应用程序直接发送到具有匹配预订的集群中的任何其他队列管理器。

直接路由的发布/预订集群提供以下优势:

- 发往同一集群中特定队列管理器上的预订的消息将直接传输到该队列管理器，并且不需要通过中间队列管理器。 与主题主机路由拓扑或分层拓扑相比，这可以提高性能。
- 由于所有队列管理器都直接相互连接，因此此拓扑的路由基础结构中没有单点故障。 如果一个队列管理器不可用，那么集群中其他队列管理器上的预订仍能够从可用队列管理器上的发布程序接收消息。
- 配置非常简单，尤其是在现有集群上。

使用直接路由的发布/预订集群时要考虑的事项:

- 集群中的所有队列管理器都可以识别集群中的所有其他队列管理器。
- 集群中用于托管集群主题的一个或多个预订的队列管理器会自动创建集群中所有其他队列管理器的集群发送方通道，即使这些队列管理器未在任何集群主题上发布消息也是如此。
- 队列管理器上对集群主题下的主题字符串的第一个预订会导致将消息发送到集群中的每个其他队列管理器。 同样，要删除的主题字符串上的最后一个预订也会生成一条消息。 在集群主题下使用的单个主题字符串越多，预订的更改速率越高，发生的队列间管理器通信越多。
- 即使队列管理器既不发布也不预订这些主题，集群中的每个队列管理器也会保留其已通知的已预订主题字符串的知识。

由于上述原因，定义了直接路由主题的集群中的所有队列管理器都将产生额外的开销。 集群中的队列管理器越多，开销越大。 同样，预订的主题字符串越多，其更改速率越大，开销越大。 这可能会导致在大型或动态直接路由的发布/预订集群中的小型系统上运行的队列管理器负载过多。 请参阅 直接路由发布/预订性能以获取更多信息。

当您知道集群无法容纳直接路由的集群发布/预订的开销时，可以改为使用 主题主机路由的发布/预订。 或者，在极端情况下，可以通过在集群中的每个队列管理器上将队列管理器属性 **PSCLUS** 设置为 DISABLED 来完全禁用集群发布/预订功能。 请参阅第 83 页的『禁止集群发布/预订』。 这将阻止创建任何集群主题，从而确保网络不会产生与集群发布/预订相关联的任何开销。

主题主机路由的发布/预订集群

通过主题主机路由，以管理方式定义集群主题的队列管理器将成为发布的路由器。 来自集群中非托管队列管理器的发布通过托管队列管理器路由到具有匹配预订的集群中的任何队列管理器。

主题主机路由的发布/预订集群在直接路由的发布/预订集群上提供以下额外优势:

- 只有定义了主题主机路由主题的队列管理器才会了解集群中的所有其他队列管理器。
- 仅主题主机队列管理器需要能够连接到集群中的所有其他队列管理器，并且通常仅连接到存在预订的那些队列管理器。 因此，在队列管理器之间运行的通道明显较少。
- 托管一个或多个集群主题预订的集群队列管理器将自动创建集群发送方通道，仅针对托管映射到预订主题字符串的集群主题的队列管理器。
- 队列管理器上对集群主题下主题字符串的第一个预订会导致将消息发送到集群主题所在的集群中的队列管理器。 同样，要删除的主题字符串上的最后一个预订也会生成一条消息。 在集群主题下使用的单个主题字符串越多，预订的更改速率越高，队列管理器之间的通信就越多，但仅在预订主机和主题主机之间进行。
- 对物理配置进行更多控制。 通过直接路由，所有队列管理器都必须参与发布/预订集群，从而增加其开销。 通过主题主机路由，只有主题主机队列管理器知道其他队列管理器及其预订。 您可以显式选择主题主机队列管理器，因此可以确保这些队列管理器在足够的设备上运行，并且可以将功能较差的系统用于其他队列管理器。

使用主题主机路由发布/预订集群时要考虑的事项:

- 当发布者或订户不在主题主管队列管理器上时，将在发布队列管理器与预订队列管理器之间引入额外的“中继段”。 额外的“中继段”导致的等待时间可能意味着主题主机路由的效率低于直接路由。
- 在大型集群上，主题主机路由可缓解直接路由的显着性能和扩展问题。
- 您可以选择在单个队列管理器上定义所有主题，或者在极少数队列管理器上定义所有主题。 如果执行此操作，请确保主题主机队列管理器托管在具有良好连接的强大系统上。
- 可以在多个队列管理器上定义同一主题。 这会提高主题的可用性，还会提高可伸缩性，因为 IBM MQ 工作负载会在该主题的所有主机上均衡某个主题的出版物。 但是，请注意，在多个队列管理器上定义同一主题会丢失该主题的消息顺序。
- 通过在不同的队列管理器上托管不同的主题，可以提高可伸缩性，而不会丢失消息顺序。

相关任务

方案: 创建发布/预订集群
配置发布/预订集群
调整分布式发布/预订网络
对分布式发布/预订问题进行故障诊断

发布/预订集群中的直接路由

来自任何发布队列管理器的发布将直接路由到具有匹配预订的集群中的任何其他队列管理器。

有关如何在发布/预订层次结构和集群中的队列管理器之间路由消息的简介，请参阅 分布式发布/预订网络。

直接路由的发布/预订集群的行为如下所示:

- 所有队列管理器自动知道所有其他队列管理器。
- 所有具有集群主题预订的队列管理器都会创建到集群中所有其他队列管理器的通道，并将其预订通知给这些队列管理器。
- 应用程序发布的消息将从其连接到的队列管理器路由到存在匹配预订的每个队列管理器。

下图显示了当前未用于发布/预订或点到点活动的队列管理器集群。 请注意，集群中的每个队列管理器仅与完整存储库队列管理器连接。

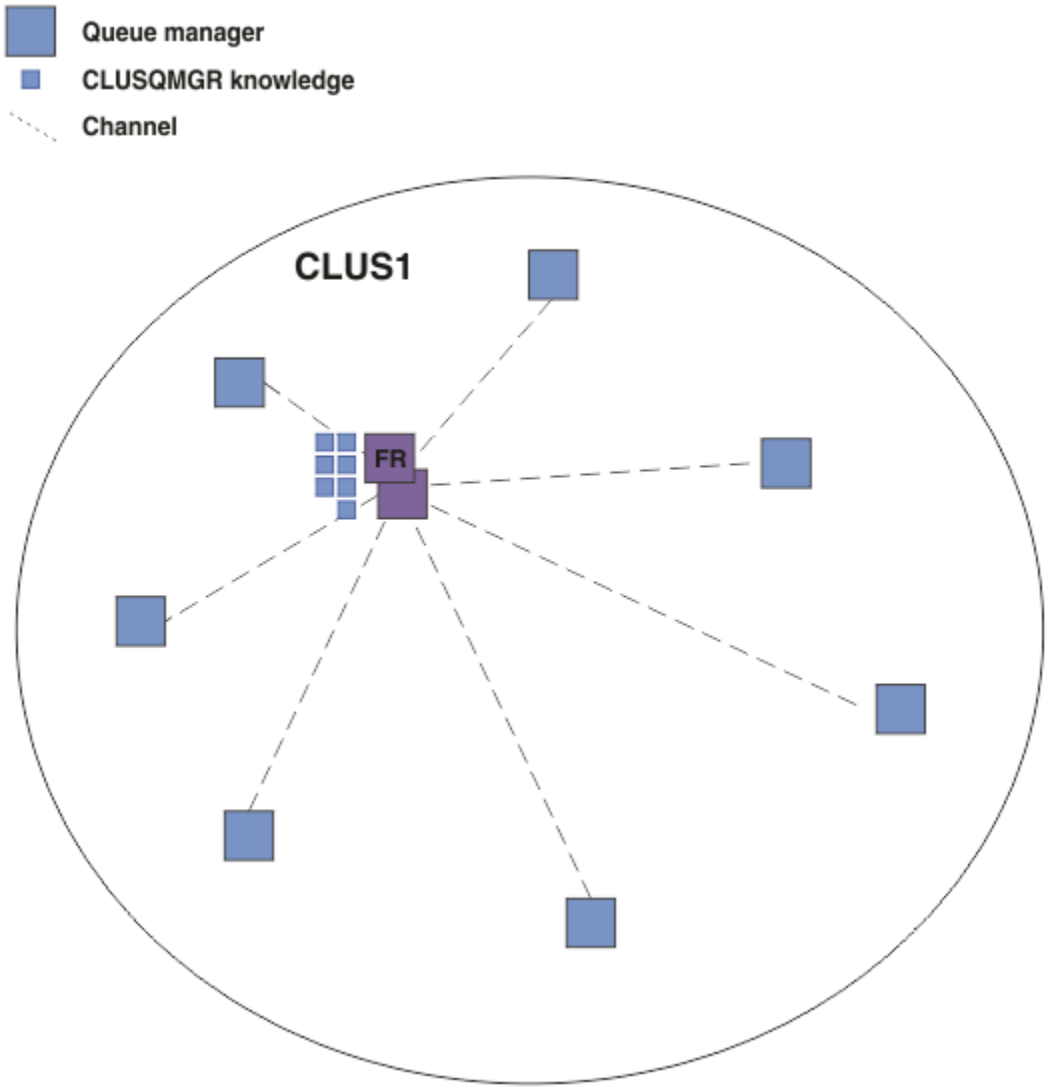


图 *16:* 队列管理器集群

要使发布在直接路由集群中的队列管理器之间流动，请按 配置发布/预订集群中所述对主题树的分支进行集群，并指定 直接路由 (缺省值)。

在直接路由的发布/预订集群中，在集群中的任何队列管理器上定义主题对象。 执行此操作时，完整存储库队列管理器会自动将对象的知识以及集群中所有其他队列管理器的知识推送到集群中的所有队列管理器。 这在任何队列管理器引用主题之前发生:

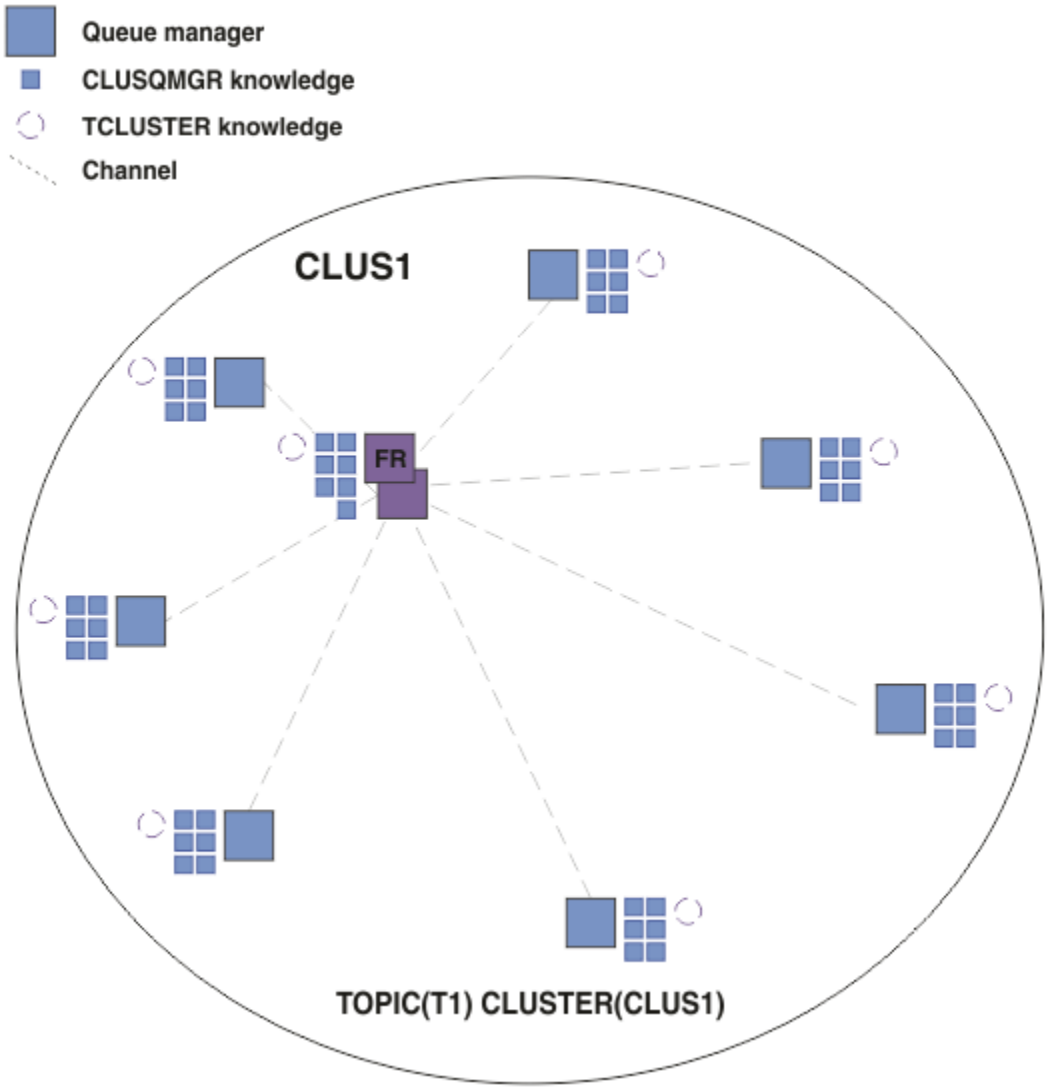


图 *17:* 直接路由的发布/预订集群

创建预订时，托管该预订的队列管理器会向集群中的每个队列管理器建立通道，并发送该预订的详细信息。此分布式预订知识由每个队列管理器上的 代理预订 表示。 在集群中与该代理预订的主题字符串相匹配的任何队列管理器上生成发布时，将建立从发布者队列管理器到托管预订的每个队列管理器的集群通道，并将消息发送到每个队列管理器。

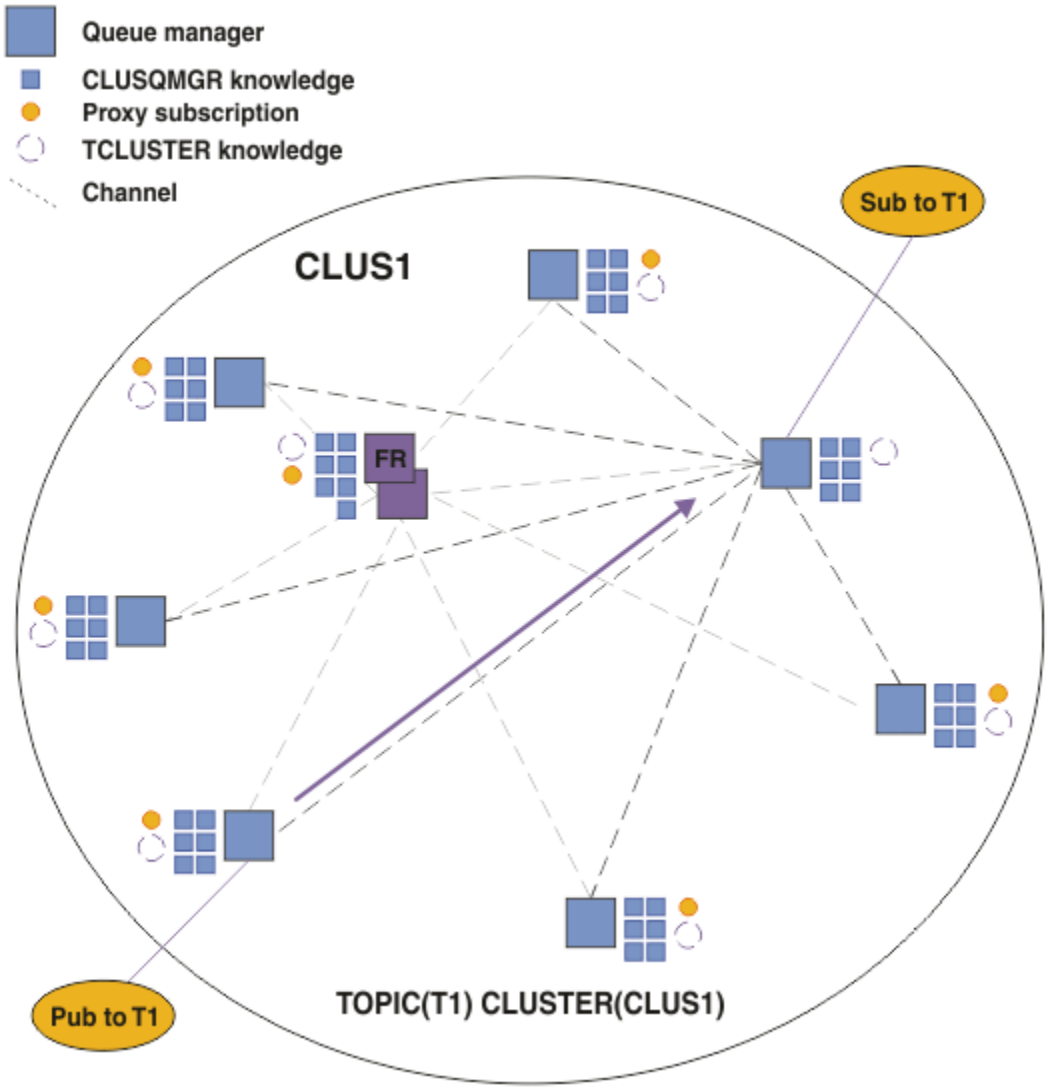


图 *18:* 具有集群主题的发布者和订户的直接路由发布/预订集群

将发布直接路由到预订托管队列管理器可简化配置，并最大程度地缩短将发布交付到预订的等待时间。

但是，根据预订和发布程序的位置，集群可以快速完全互连，每个队列管理器都与其他每个队列管理器直接连接。 这在您的环境中可能是可接受的，也可能是不可接受的。 同样，如果要预订的主题字符串集频繁更改，那么在所有队列管理器之间传播该信息的开销也会很大。 直接路由的发布/预订集群中的所有队列管理器都必须能够处理这些开销。

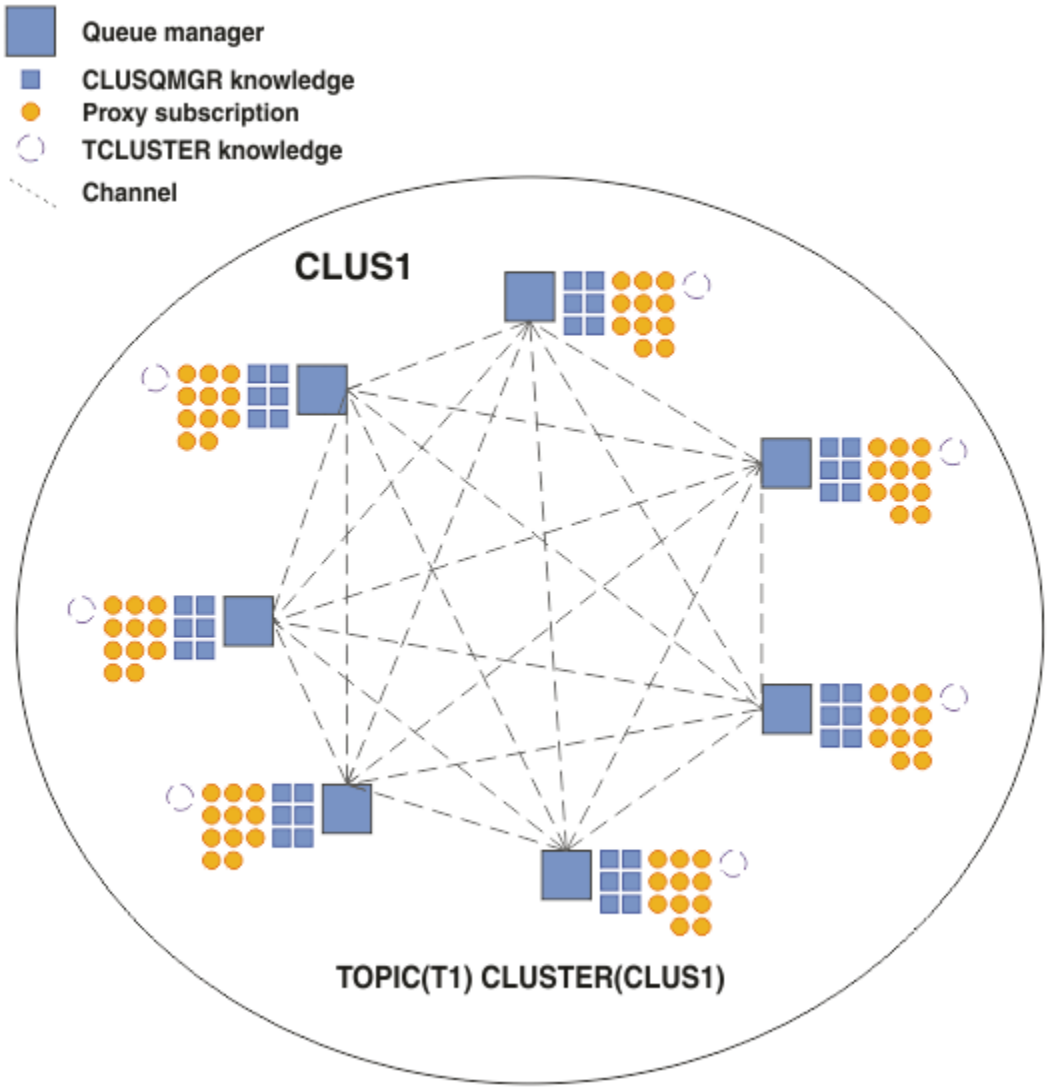


图 *19:* 完全互连的直接路由发布/预订集群

摘要和其他注意事项

直接路由的发布/预订集群无需手动干预即可创建或管理，并在发布者和订户之间提供直接路由。 对于某些配置，它通常是最合适的拓扑，尤其是具有很少队列管理器的集群，或者具有可接受的高队列管理器连接且预订不经常更改的集群。 但是，这也会对您的系统施加某些约束:

- 每个队列管理器上的负载与集群中的队列管理器总数成比例。 因此，在更大的集群中，单个队列管理器和整个系统可能会迁到性能问题。
- 缺省情况下，预订的所有集群主题字符串都将在整个集群中传播，并且发布仅传播到预订了关联主题的远程队列管理器。 因此，对预订集的快速更改可能成为限制因素。 您可以更改此缺省行为，而是将所有发布传播到所有队列管理器，这将除去对代理预订的需求。 这将减少预订知识流量，但可能会增加发布流量以及每个队列管理器建立的通道数。 请参阅 发布/预订网络中的预订性能。

  注: 类似的限制也适用于层次结构。
- 由于发布/预订队列管理器的互连性质，代理预订在网络中的所有节点之间传播需要时间。 远程发布不一定会立即开始预订，因此在预订新主题字符串之后可能不会发送早期发布。 您可以通过将所有发布传播到所有队列管理器来除去预订延迟所导致的问题，这将除去对代理预订的需求。 请参阅 发布/预订网络中的预订性能。

  注: 此限制也适用于层次结构。

在使用直接路由之前，请探索 第 67 页的『发布/预订集群中的主题主机路由』和 第 87 页的『在发布/预订层次结构中进行路由』中详细描述的替代方法。

发布/预订集群中的主题主机路由

来自集群中非托管队列管理器的发布通过托管队列管理器路由到具有匹配预订的集群中的任何队列管理器。

有关如何在发布/预订层次结构和集群中的队列管理器之间路由消息的简介，请参阅 分布式发布/预订网络。

要了解主题主机路由的行为和优点，最好首先了解 第 62 页的『发布/预订集群中的直接路由』。

主题主机路由的发布/预订集群的行为如下所示:

- 集群受管主题对象是在集群中的各个队列管理器上手动定义的。 这些称为 主题主机队列管理器。
- 在集群队列管理器上进行预订时，将创建从预订主机队列管理器到主题主机队列管理器的通道，并且仅在托管主题的队列管理器上创建代理预订。
- 当应用程序将信息发布到主题时，已连接的队列管理器始终将该发布转发到主管该主题的一个队列管理器，该队列管理器将该主题传递到集群中具有与该主题的匹配预订的所有队列管理器。

以下示例更详细地说明了此过程。

使用单个主题主机进行主题主机路由

要使发布在主题主机路由集群中的队列管理器之间流动，请按 配置发布/预订集群中所述对主题树的分支进行集群，并指定 主题主机路由。

在集群中的多个队列管理器上定义主题主机路由主题对象有多种原因。 但是，为了简单起见，我们从单个主题主机开始。

下图显示了当前未用于发布/预订或点到点活动的队列管理器集群。 请注意，集群中的每个队列管理器仅与完整存储库队列管理器连接。

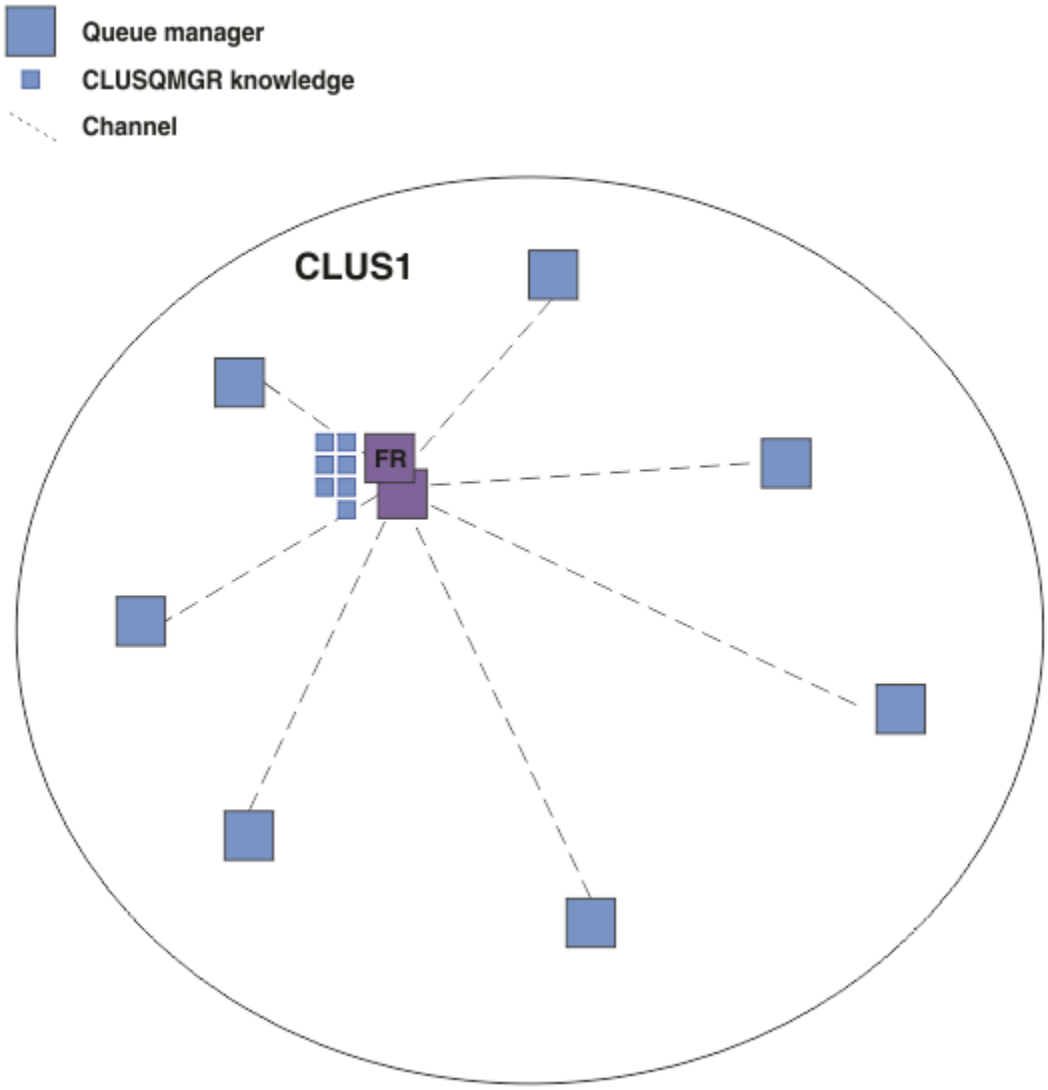


图 *20:* 队列管理器集群

在主题主机路由的发布/预订集群中，您在集群中的特定队列管理器上定义主题对象。 然后，发布/预订流量流经该队列管理器，使其成为集群中的关键队列管理器并增加其工作负载。 由于这些原因，建议不要使用完整存储库队列管理器，而要使用集群中的另一个队列管理器。 在主机队列管理器上定义主题对象时，完整存储库队列管理器会将该对象及其主机的知识自动推送到集群中的所有其他队列管理器。 请注意，与 直接路由不同，不会向每个队列管理器讲述集群中的每个其他队列管理器。

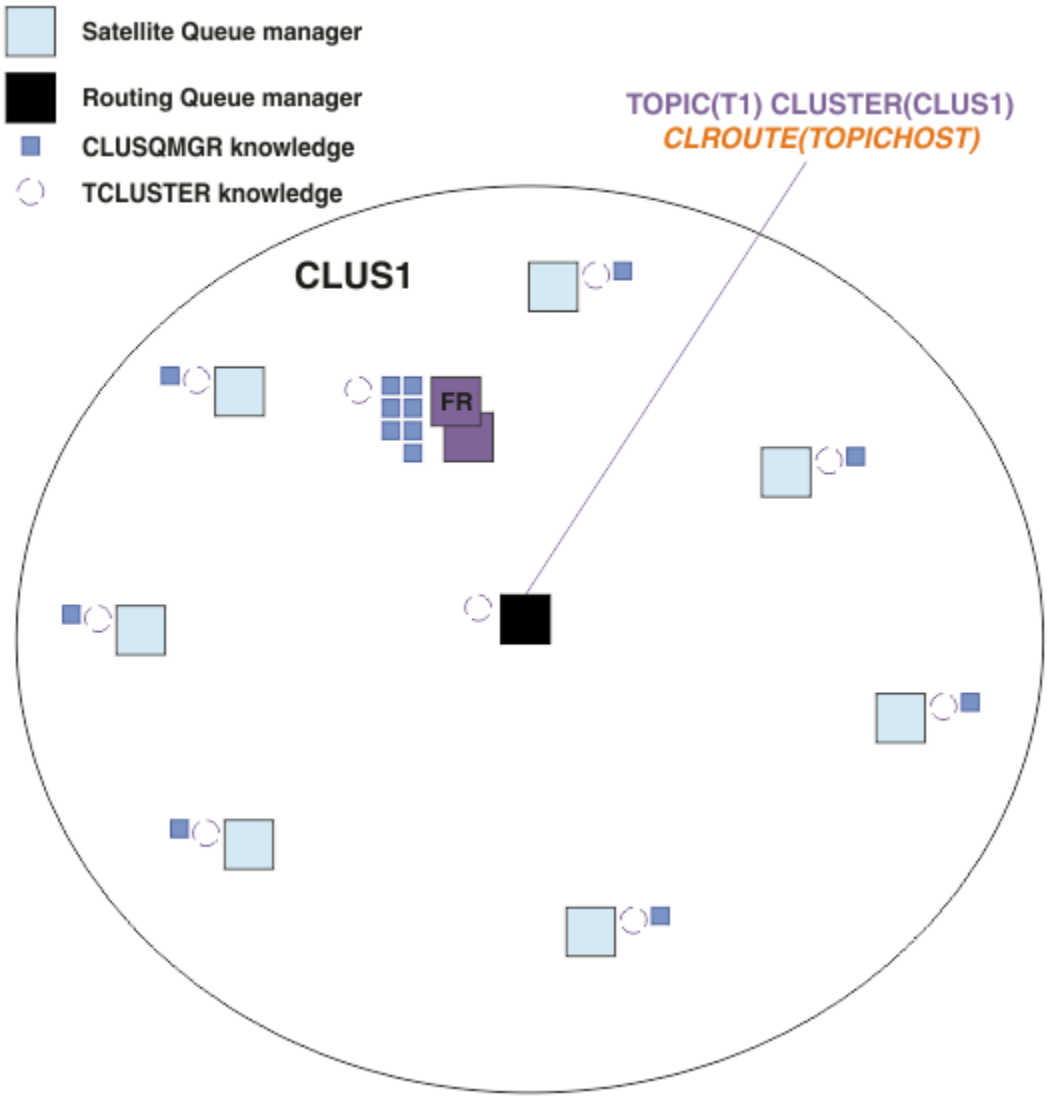


图 *21:* 在一个主题主机上定义了一个主题的主题主机路由发布/预订集群

在队列管理器上创建预订时，将在预订队列管理器与主题主机队列管理器之间创建通道。 预订队列管理器仅连接到主题主机队列管理器，并发送预订的详细信息 (以 代理预订的形式)。 主题主机队列管理器不会将此预订信息转发到集群中的任何其他队列管理器。

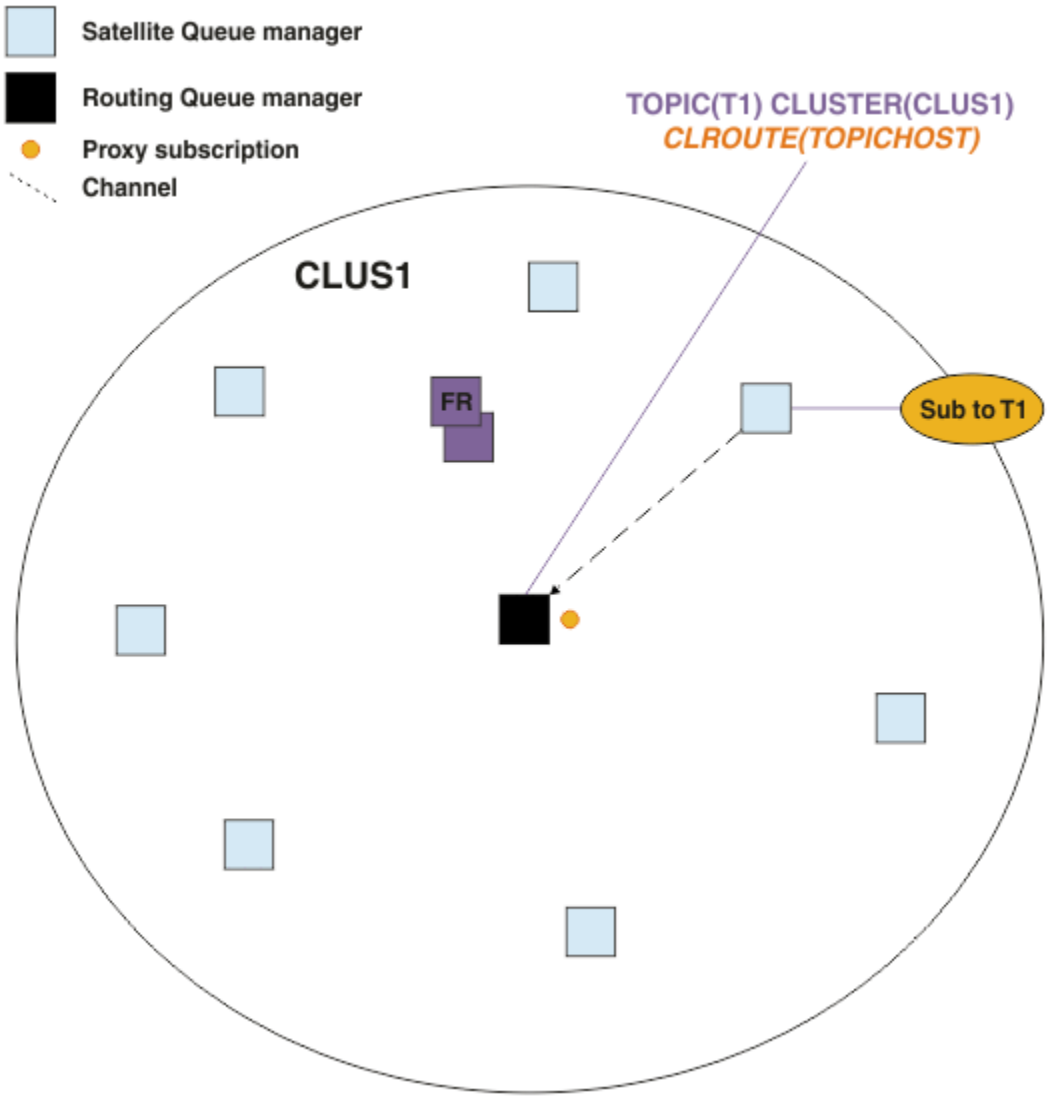


图 *22:* 主题主机路由的发布/预订集群，其中一个主题主机上定义了一个主题，一个订户

当发布应用程序连接到另一个队列管理器并发布消息时，将在发布队列管理器与主题主机队列管理器之间创建通道，并将消息转发到该队列管理器。 发布队列管理器不知道集群中其他队列管理器上的任何预订，因此即使集群中没有该主题的订户，也会将消息转发到主题主机队列管理器。 发布队列管理器仅连接到主题主机队列管理器。 发布通过主题主机路由到预订队列管理器 (如果存在)。

将直接满足与发布程序相同的队列管理器上的预订，而不首先将消息发送到主题主机队列管理器。

请注意，由于每个主题主机队列管理器都具有关键作用，因此您必须选择可处理主题托管的负载，可用性和连接需求的队列管理器。

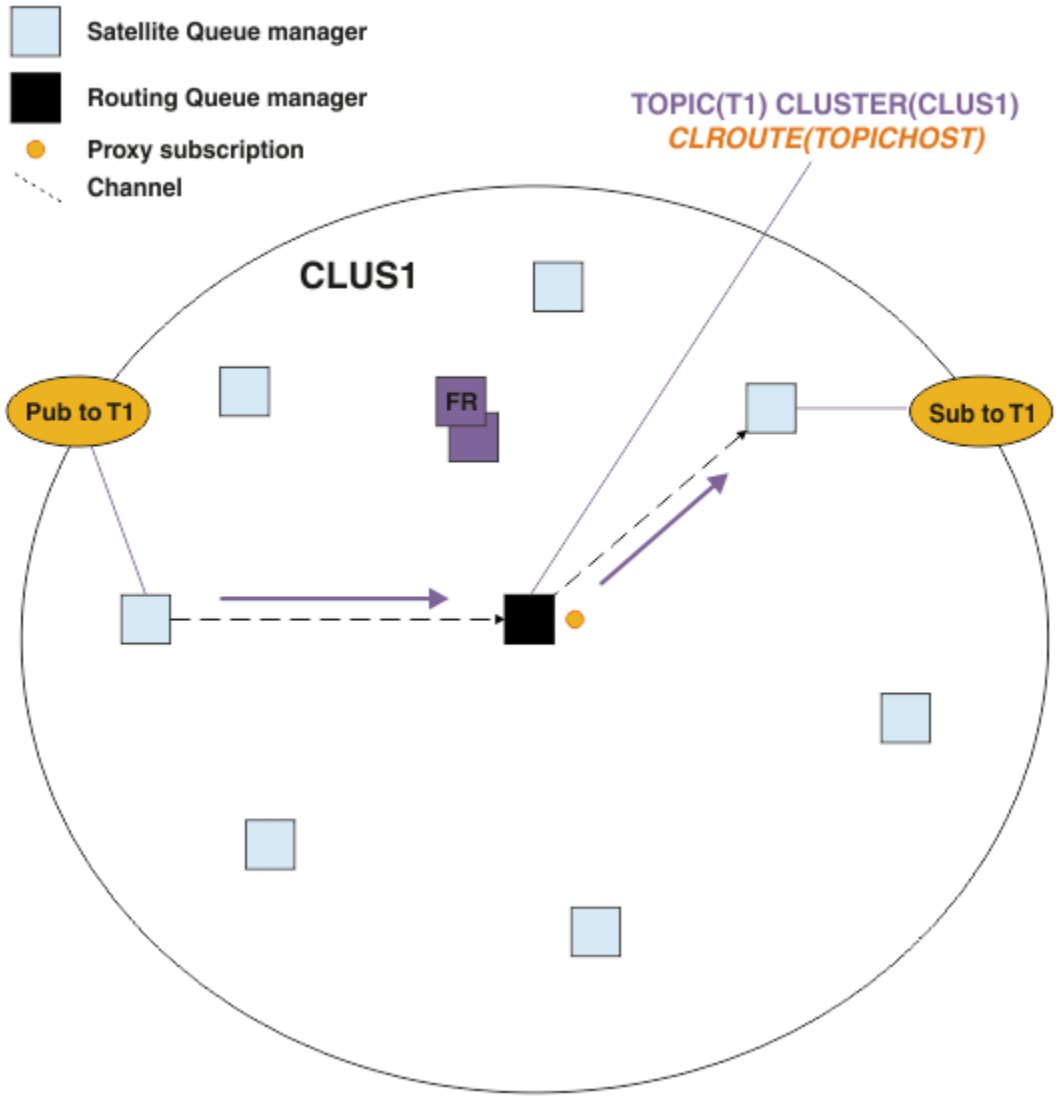


图 *23:* 具有一个主题，一个订户和一个发布者的主题主机路由发布/预订集群

在多个队列管理器之间划分主题树

路由主题主管队列管理器仅负责与配置其受管主题对象的主题树分支相关的预订知识和发布消息。 如果集群中的不同发布/预订应用程序使用不同的主题，那么可以配置不同的队列管理器来托管主题树的不同集群分支。 这允许通过减少集群中每个主题主机队列管理器上的发布流量，预订知识和通道来进行扩展。 您应该将此方法用于主题树的不同高容量分支:

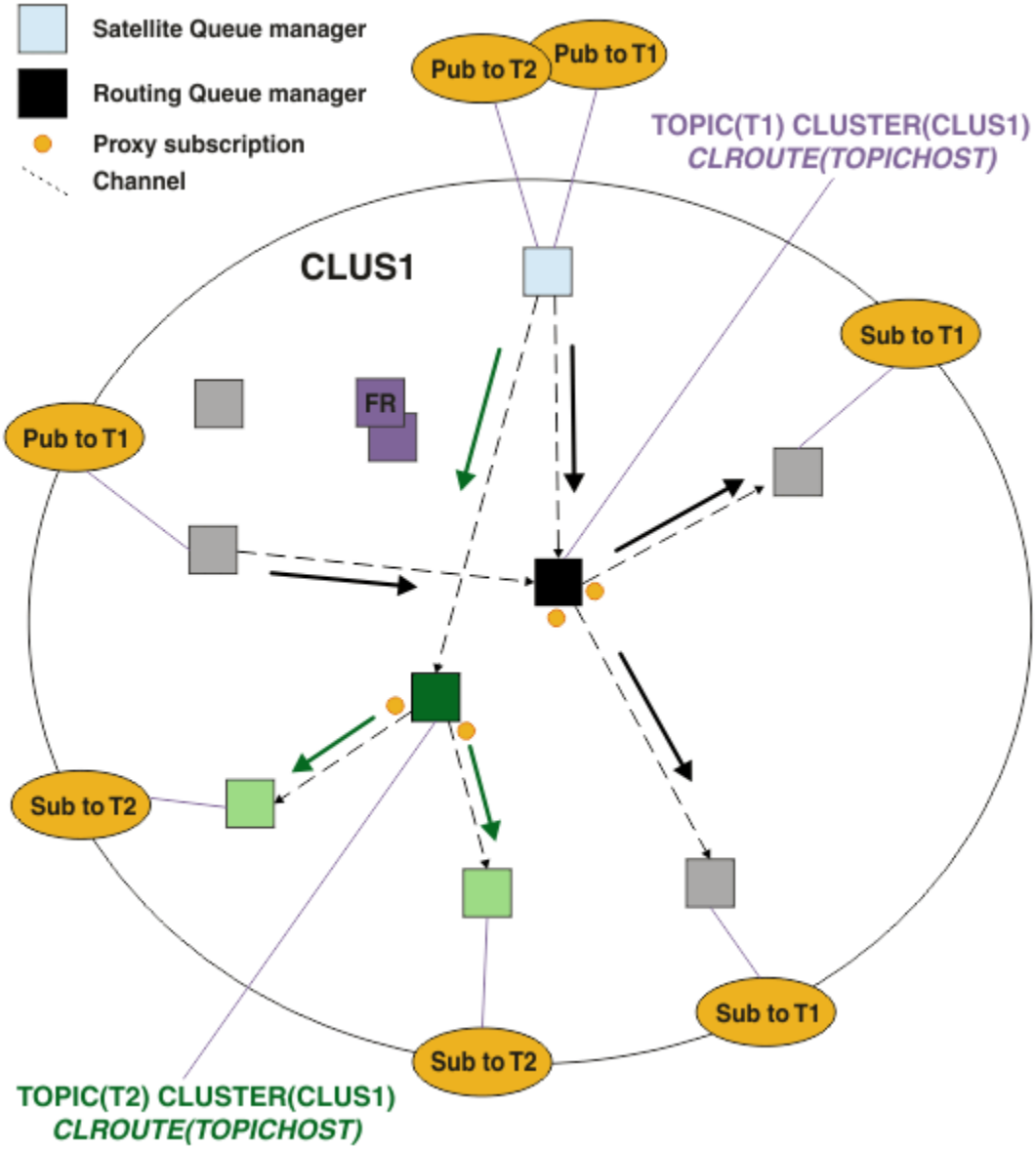


图 *24:* 主题主机路由的发布/预订集群，包含两个主题，每个主题在一个主题主机上定义

例如，使用 主题树中描述的主题，如果主题 T1 配置了主题字符串 /USA/Alabama，而主题 T2 配置了主题字符串 /USA/Alaska，那么将通过托管 T1 的队列管理器来路由发布到 /USA/Alabama/Mobile 的消息。并且将通过托管 T2 的队列管理器来路由发布到 /USA/Alaska/Juneau 的消息。

注: 通过在主题树中使用比集群点更高的通配符，无法使单个预订跨主题树的多个集群分支。 请参阅 通配符预订。

使用单个主题的多个主题主机进行主题主机路由

如果单个队列管理器负责主题的路由，并且该队列管理器变得不可用或无法处理工作负载，那么发布将不会迅速流向预订。

如果您需要比仅在一个队列管理器上定义主题时获得更大的弹性，可伸缩性和工作负载均衡，那么可以在多个队列管理器上定义主题。 发布的每条消息都通过单个主题主机进行路由。 存在多个匹配的主题主机定义时，将选择其中一个主题主机。 所作的选择与对集群队列的选择相同。 这允许将消息路由到可用的主题主机，从而避免任何不可用的消息，并允许在多个主题主机队列管理器和通道之间均衡消息负载。 但是，当您将多个主题主机用于集群中的同一主题时，不会维护跨多条消息的排序。

下图显示了在两个队列管理器上定义了相同主题的主题主机路由集群。 在此示例中，预订队列管理器以代理预订的形式向两个主题主机队列管理器发送有关预订主题的信息:

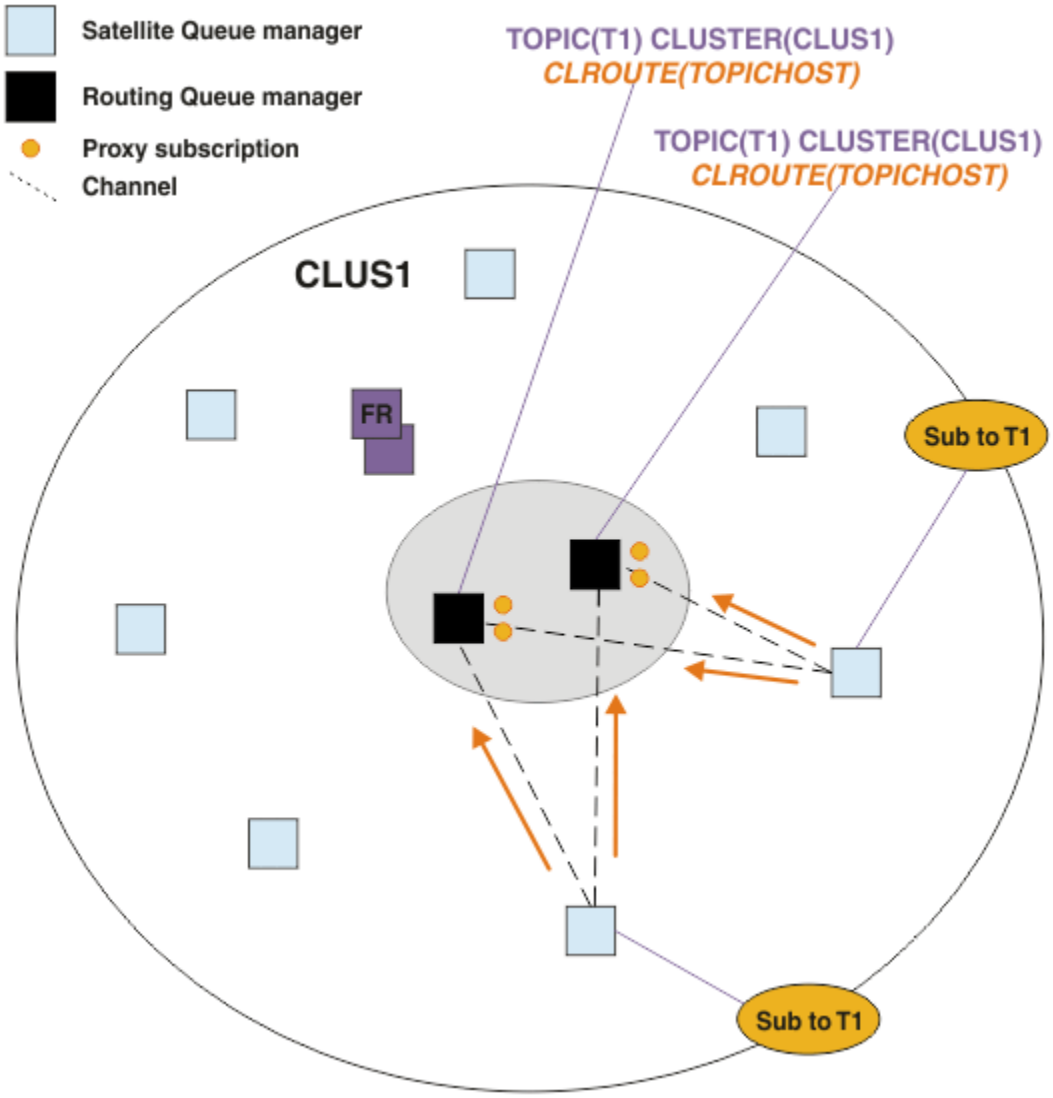


图 *25:* 在多主题主机发布/预订集群中创建代理预订

从非主管队列管理器进行发布时，队列管理器会将发布的副本发送到该主题的主题主机队列管理器的 一个 。 系统根据 集群工作负载管理算法的缺省行为来选择主机。 在典型系统中，这近似于每个主题主机队列管理器之间的循环分布。 来自同一发布应用程序的消息之间没有亲缘关系; 这等同于使用集群绑定类型 NOTFIXED。

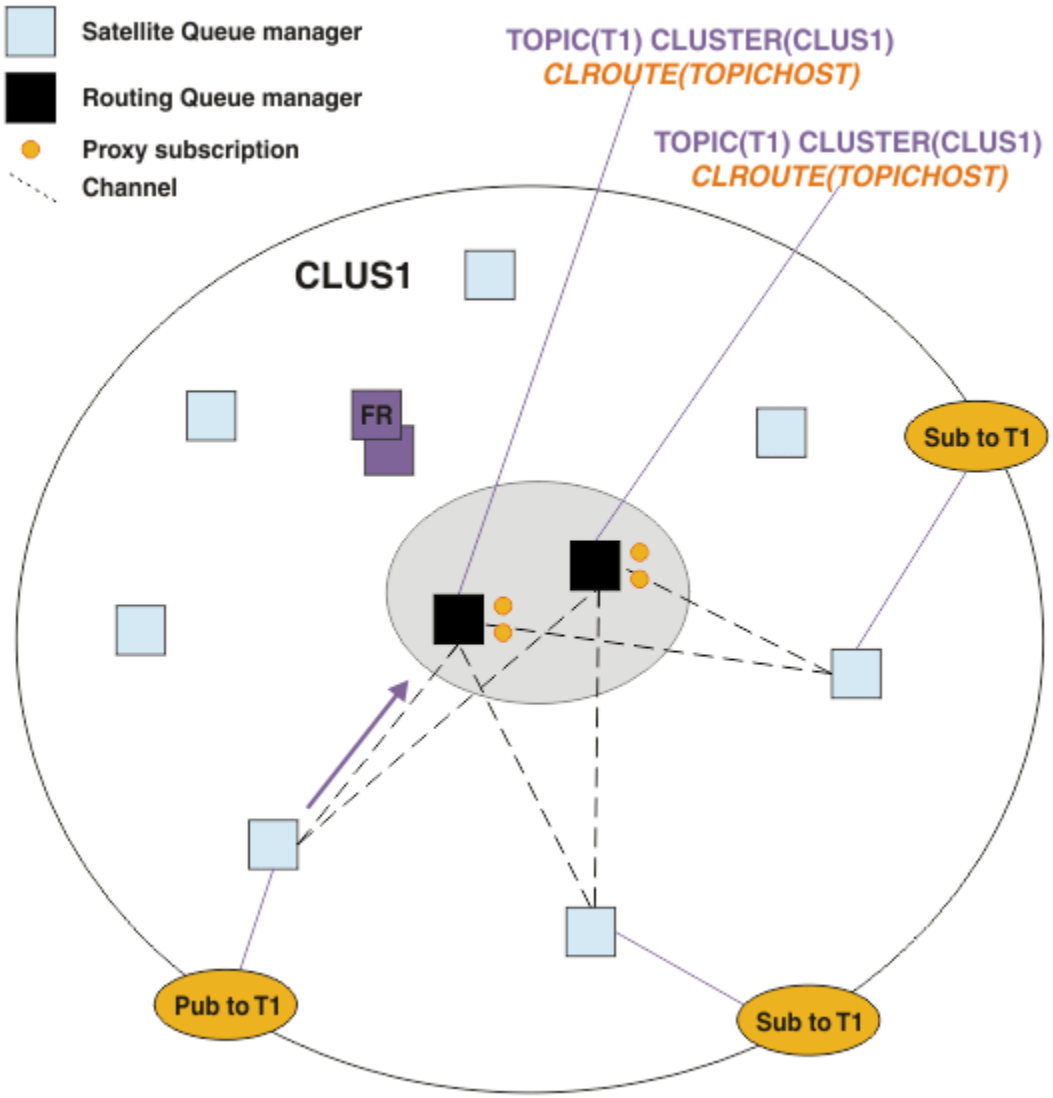


图 *26:* 在多主题主机发布/预订集群中接收发布

然后，将所选主题主机队列管理器的入站发布转发到已注册匹配代理预订的所有队列管理器:

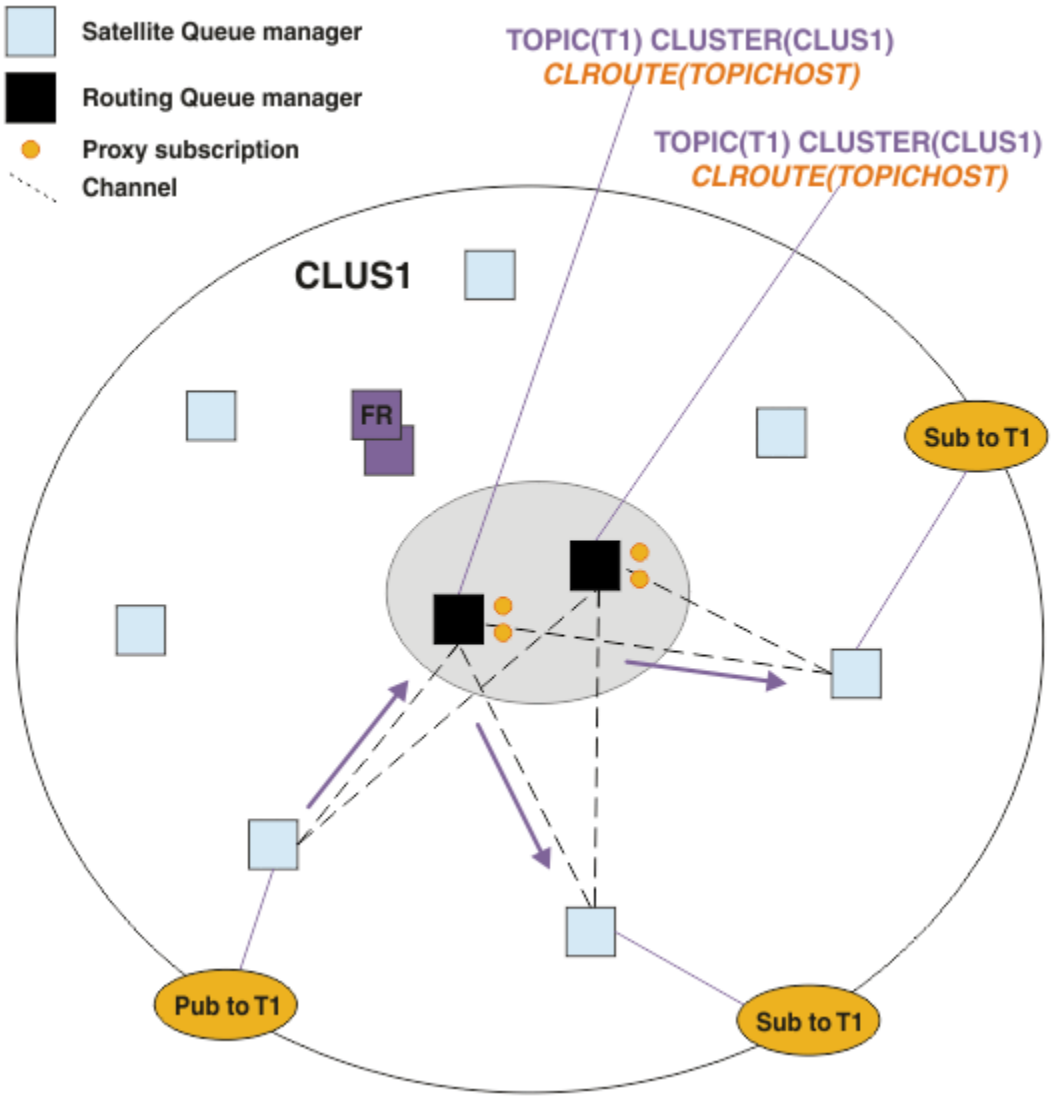


图 *27:* 将发布路由到多主题主机发布/预订集群中的订户

使预订和发布程序成为主题主机队列管理器的本地预订和发布程序

以上示例显示了未托管受管路由主题对象的队列管理器上的发布者和订户之间的路由。 在这些拓扑中，消息需要多个 中继段 才能访问预订。

如果不需要额外的中继段，那么可能需要将密钥发布程序连接到主题托管队列管理器。 但是，如果一个主题有多个主题主机，并且只有一个发布程序，那么所有发布程序流量都将通过发布程序所连接的主题主机队列管理器进行路由。

同样，如果存在密钥预订，那么这些预订可能位于主题主机队列管理器上。 但是，如果路由主题有多个主机，那么只有一部分发布内容将避免额外的中继段，其余部分将首先通过其他主题主机队列管理器进行路由。

此处进一步描述了这些拓扑: 使用集中发布程序或订户进行主题主机路由。

注: 如果在将发布程序或预订与路由主题主机共存时更改路由主题配置，那么需要进行特殊规划。 例如，请参阅 向主题主机路由集群添加额外主题主机。

摘要和其他注意事项

主题主机路由的发布/预订集群使您能够精确控制哪些队列管理器托管每个主题，这些队列管理器将成为主题树的该分支的 路由 队列管理器。 此外，没有预订或发布者的队列管理器无需与主题主机队列管理器连接，

而有预订的队列管理器无需与不托管主题的队列管理器连接。 此配置可以显着减少集群中队列管理器之间的连接数以及在队列管理器之间传递的信息量。 在只有一部分队列管理器在执行发布/预订工作的大型集群中尤其如此。 此配置还使您能够对集群中各个队列管理器上的负载进行一些控制，因此 (例如) 您可以选择在功能更强大且弹性更强的系统上托管高度活动的主题。 对于某些配置 (尤其是较大的集群)，它通常是比 直接路由更合适的拓扑。

但是，主题主机路由还将对您的系统施加一些约束：

- 系统配置和维护需要比直接路由更多的规划。 您需要决定在主题树中要对哪些点建立集群，以及集群中主题定义的位置。
- 与直接路由主题一样，在定义新的主题主机路由主题时，该信息会推送到完整存储库队列管理器，并从中定向到集群的所有成员。 此事件将导致通道从完整存储库启动到集群的每个成员（如果尚未启动）。
- 发布始终会从非主机队列管理器发送至主机队列管理器，即使集群中没有预订也是如此。 因此，通常期望存在预订时，或全局连接和知晓的开销大于额外发布流量的风险时，应使用路由的主题。

  注: 如前所述，使发布者成为主题主机的本地发布者可以降低此风险。
- 在非主机队列管理器上发布的消息不会直接进入托管预订的队列管理器，而是会始终通过主题主机队列管理器进行路由。 此方法可能会增加集群的总开销，并且可能会增加消息等待时间而降低性能。

  注: 如前所述，使预订或发布程序成为主题主机的本地预订或发布程序可降低此风险。
- 使用单一主题主机队列管理器将为发布到某个主题的所有消息引入单一故障点。 您可以通过定义多个主题主机来移除此单一故障点。 然而，拥有多个主机将影响预订收到已发布消息的顺序。
- 主题主机队列管理器将产生额外消息负载，因为它们需要处理来自多个队列管理器的发布流量。 可以减轻此负载：将多个主题主机用于单个主题（在此情况下，不会维护消息顺序），或使用不同的队列管理器来托管主题树的不同分支的路由主题。

在使用主题主机路由之前，请探索 第 62 页的『发布/预订集群中的直接路由』和 第 87 页的『在发布/预订层次结构中进行路由』中详细描述的替代方法。

发布/预订集群: 最佳实践

使用集群主题使在队列管理器之间扩展发布/预订域变得简单，但如果无法完全理解机制和影响，可能会导致问题。 有两个模型用于信息共享和发布路由。 实施最能满足个人业务需求的模型，并在所选集群上执行最佳操作。

以下部分中的最佳实践信息并不提供一刀切的解决方案，而是共享用于解决常见问题的常见方法。 它假定您基本了解 IBM MQ 集群以及发布/预订消息传递，并且您熟悉 分布式发布/预订网络 和 第 61 页的『设计发布/预订集群』中的信息。

当您将集群用于点到点消息传递时，集群中的每个队列管理器都将以需要知道的方式工作。 即，仅当连接到集群的应用程序请求使用其他集群资源 (例如，集群中的其他队列管理器和集群队列) 时，它才会发现这些资源。 将发布/预订消息传递添加到集群时，将引入更多级别的信息共享以及集群队列管理器之间的连接。 为了能够遵循发布/预订集群的最佳实践，您需要充分了解这种行为变化的影响。

为了使您能够根据您的精确需求构建最佳体系结构，在发布/预订集群中有两个用于信息共享和发布路由的模型: 直接路由 和 主题主机路由。 要做出正确的选择，您需要了解两个模型，以及每个模型满足的不同需求。 以下部分与 第 58 页的『规划分布式发布/预订网络』一起讨论了这些需求:

- 第 76 页的『限制发布/预订活动中涉及的集群队列管理器数的原因』
- 第 77 页的『如何确定要集群的主题』
- 第 77 页的『如何调整系统大小』
- 第 78 页的『发布者和预订位置』
- 第 78 页的『发布流量』
- 第 79 页的『预订更改和动态主题字符串』

限制发布/预订活动中涉及的集群队列管理器数的原因

在集群中使用发布/预订消息传递时，存在容量和性能注意事项。 因此，最好仔细考虑跨队列管理器的发布/预订活动的需求，并将其限制为仅需要该活动的队列管理器的数量。 在确定需要发布和预订主题的最小队列管理器集合后，可以使这些队列管理器成为仅包含它们且不包含其他队列管理器的集群的成员。

如果已建立的集群已正常用于点到点消息传递，那么此方法特别有用。 将现有大型集群转换为发布/预订集群时，最好先为发布/预订工作创建单独的集群，在此工作中可以尝试使用应用程序，而不是使用当前集群。 您可以使用已存在于一个或多个点到点集群中的现有队列管理器子集，并使此子集成为新的发布/预订集群的成员。 但是，新集群的完整存储库队列管理器不得是任何其他集群的成员; 这将隔离现有集群完整存储库中的额外负载。

如果无法创建新集群，并且必须将现有大型集群转换为发布/预订集群，请不要使用直接路由模型。 主题主机路由模型通常在更大的集群中表现更好，因为它通常限制发布/预订信息共享以及与正在主动执行发布/预订工作的队列管理器集合的连接，集中在托管主题的队列管理器上。 此情况的例外情况是，在托管主题定义的队列管理器上调用了预订信息的手动刷新，此时主题主机队列管理器将连接到集群中的每个队列管理器。 请参阅 代理预订的再同步。

如果您确定集群由于其大小或当前负载而无法用于发布/预订，那么最好防止此集群意外变为发布/预订集群。 使用 **PSCLUS** 队列管理器属性可阻止任何人在集群中的任何队列管理器上添加集群主题。 请参阅 第 83 页的『禁止集群发布/预订』。

如何确定要集群的主题

请务必仔细选择将哪些主题添加到集群中: 这些主题的主题树越高，其使用范围就越广。 这可能会导致传播的预订信息和发布数超过必需的数量。 如果主题树有多个不同的分支 (其中一些需要集群，而另一些不需要集群) ，请在需要集群的每个分支的根上创建受管主题对象，并将这些对象添加到集群中。 例如，如果分支 /A， /B 和 /C 需要集群，请为每个分支定义单独的集群主题对象。

注: 系统会阻止您在主题树中嵌套集群主题定义。 仅允许您在每个子分支的主题树中的一个点上对主题进行聚类。 例如，无法为 /A 和 /A/B 定义集群主题对象。 嵌套集群主题可能会导致混淆哪个集群对象适用于哪个预订，尤其是在预订使用通配符时。 这在使用主题主机路由时更为重要，其中路由决策由主题主机的分配精确定义。

如果必须将集群主题添加到主题树的高处，但集群点以下的树的某些分支不需要集群行为，那么可以使用预订和发布范围属性来降低预订和发布共享级别以用于更多主题。

如果不考虑所看到的行为，那么不应将主题根节点放入集群中。 尽可能使全局主题变得明显，例如，在主题字符串中使用高级限定符: /global 或 /cluster。

还有一个原因是不想使根主题节点集群化。 这是因为每个队列管理器都具有根节点 SYSTEM.BASE.TOPIC 主题对象的本地定义。 当此对象在集群中的一个队列管理器上进行集群时，将使所有其他队列管理器了解此对象。 但是，当存在同一对象的本地定义时，其属性将覆盖集群对象。 这将导致这些队列管理器像未对主题进行集群一样工作。 要解决此问题，您需要对 SYSTEM.BASE.TOPIC 的每个定义进行集群。 您可以对直接路由定义执行此操作，但不能对主题主机路由定义执行此操作，因为这会导致每个队列管理器都成为主题主机。

如何调整系统大小

发布/预订集群通常导致集群通道模式与集群中的点到点消息传递模式不同。 点对点模型是 "选择性加入" 模型，但发布/预订集群具有更滥用的预订扇出性质，尤其是在使用直接路由主题时。 因此，请务必确定发布/预订集群中的哪些队列管理器将使用集群通道连接到其他队列管理器，以及在何种情况下连接到其他队列管理器。

下表列出了在正常运行的发布/预订集群中每个队列管理器期望的集群发送方和接收方通道的典型集合，这取决于发布/预订集群中的队列管理器角色。

表 *5:* 每个路由方法的集群发送方和接收方通道。

队列管理器角色	直接集群接收方	直接集群发送方	主题集群接收方	主题集群发送方
完整存储库	AllQmgrs	AllQmgrs	AllQmgrs	AllQMgrs
主题定义的主机	不适用	不适用	AllSubs+AllPubs (1)	AllSubs (1)
已创建预订	AllPubs (1)	AllQMgrs	AllHosts	AllHosts
已连接发布程序	AllSubs (1)	AllSubs (1)	AllHosts	AllHosts
无发布者或订户	AllSubs (1)	无 (1)	无 (2)	无 (2)

关键字：

AllQmgrs
这是与集群中的每个队列管理器之间的通道。

AllSubs
与已创建预订的每个队列管理器之间的通道。

AllPubs
与已连接发布应用程序的每个队列管理器之间的通道。

AllHosts
在其中配置了集群主题对象定义的每个队列管理器之间的通道。

无
仅出于发布/预订消息传递的目的，集群中没有与其他队列管理器之间的通道。

注意：

1. 如果从此队列管理器刷新代理预订，那么可能会自动创建与集群中所有其他队列管理器之间的通道。
2. 如果从此队列管理器刷新代理预订，那么可能会自动创建用于托管集群主题定义的集群中任何其他队列管理器的通道。

上表显示，与直接路由相比，主题主机路由通常使用的集群发送方和接收方通道要少得多。 如果通道连接是集群中某些队列管理器所关注的问题，那么出于容量或建立特定通道 (例如，通过防火墙) 的能力的原因，主题主机路由是首选解决方案。

发布者和预订位置

集群发布/预订使在一个队列管理器上发布的消息能够传递到集群中任何其他队列管理器上的预订。 对于点到点消息传递，在队列管理器之间传输消息的成本可能对性能不利。 因此，您应该考虑在发布消息的队列管理器上创建对主题的预订。

在集群中使用主题主机路由时，还应考虑与主题托管队列管理器相关的预订和发布程序的位置。 当发布者未连接到作为集群主题的主机的队列管理器时，发布的消息将始终发送到主题托管队列管理器。 同样，在不是集群主题的主题主机的队列管理器上创建预订时，从集群中的其他队列管理器发布的消息总是首先发送到主题主管队列管理器。 更具体地说，如果预订位于托管该主题的队列管理器上，但有一个或多个其他队列管理器也托管同一主题，那么来自其他队列管理器的部分发布将通过这些其他主题托管队列管理器进行路由。 请参阅 使用集中发布程序或订户的主题主机路由 ，以获取有关设计主题主机路由发布/预订集群以最大限度缩短发布程序与预订之间的距离的更多信息。

发布流量

由连接到集群中的一个队列管理器的应用程序发布的消息将使用集群发送方通道传输到其他队列管理器上的预订。

使用直接路由时，发布的消息在队列管理器之间采用最短路径。 即，它们直接从发布队列管理器到具有预订的每个队列管理器。 不会将消息传输到没有主题预订的队列管理器。 请参阅 发布/预订网络中的代理预订。

如果集群中任何一个队列管理器与另一个队列管理器之间的发布消息速率较高，那么这两个点之间的集群通道基础结构必须能够保持该速率。 这可能涉及调整正在使用的通道和传输队列。

当您使用主题主机路由时，在不是主题主机的队列管理器上发布的每条消息都会传输到主题主机队列管理器。 这与集群中其他位置是否存在一个或多个预订无关。 这引入了在规划时要考虑的其他因素:

- 首次向主题主机队列管理器发送每个发布的额外等待时间是否可接受?
- 每个主题主机队列管理器能否维持入站和出站发布速率? 请考虑一个具有许多不同队列管理器上的发布程序的系统。 如果它们都将其消息发送到一组非常小的主题托管队列管理器，那么这些主题主机可能会成为处理这些消息并将其路由到预订队列管理器的瓶颈。
- 是否预期相当一部分已发布的消息将没有匹配的订户? 如果是这样，并且发布此类消息的速率很高，那么最好使发布者的队列管理器成为主题主机。 在此情况下，不会将集群中不存在任何预订的任何已发布消息传输到任何其他队列管理器。

还可以通过引入多个主题主机来缓解这些问题，以在它们之间传播发布负载:

- 如果有多个不同的主题 (每个主题都有一定比例的发布流量) ，请考虑将它们托管在不同的队列管理器上。
- 如果无法将主题分隔到不同的主题主机上，请考虑在多个队列管理器上定义相同的主题对象。 这将导致出版物的工作负载在每个出版物之间进行均衡，以便进行路由。 但是，仅当不需要发布消息排序时，此操作才适用。

预订更改和动态主题字符串

另一个考虑因素是对传播代理预订的系统性能的影响。 通常，当在该队列管理器上创建特定集群主题字符串 (而不仅仅是配置的主题对象) 的第一个预订时，队列管理器会将代理预订消息发送到集群中的某些其他队列管理器。 同样，在删除特定集群主题字符串的最后一个预订时，将发送代理预订删除消息。

对于直接路由，具有预订的每个队列管理器都会将这些代理预订发送到集群中的每个其他队列管理器。 对于主题主机路由，具有预订的每个队列管理器仅将代理预订发送到主管该集群主题的定义的每个队列管理器。 因此，使用直接路由时，集群中的队列管理器越多，跨这些队列管理器维护代理预订的开销就越高。 而对于主题主机路由，集群中的队列管理器数不是一个因素。

在这两种路由模型中，如果发布/预订解决方案由许多要预订的唯一主题字符串组成，或者集群中的队列管理器上的主题经常被预订和取消预订，那么将在该队列管理器上看到由于不断生成消息分发和删除代理预订而导致的大量开销。 通过直接路由，由于需要将这些消息发送到集群中的每个队列管理器，这就更加复杂了。

如果预订的更改速率过高而无法容纳 (即使在主题主机路由系统中) ，请参阅 发布/预订网络中的预订性能 以获取有关减少代理预订开销的方法的信息。

定义集群主题

集群主题是定义了 **cluster** 属性的管理主题。 有关集群主题的信息会推送给集群的所有成员，并与本地主题相结合以创建横跨多个队列管理器的主题空间的一些部分。 这支持将一个队列管理器中的某个主题上发布的消息传递给集群中其他队列管理器的预订。

在队列管理器上定义集群主题时，会将集群主题定义发送到完整存储库队列管理器。 然后，完整存储库会将集群主题定义传播到集群中的所有队列管理器，从而使相同的集群主题可用于集群中任何队列管理器处的发布者和订户。 您在其中创建集群主题的队列管理器称为集群主题主机。 集群主题可供集群中的任何队列管理器使用，但是对集群主题的任何修改必须在定义该主题的队列管理器（主机）上执行，此时会通过完整存储库将修改传播到集群的所有成员。

使用直接路由时，集群主题定义的位置不会直接影响系统的行为，因为集群中的所有队列管理器都以相同方式使用主题定义。 因此，您应该在任何将成为集群成员的队列管理器上定义主题，只要需要该主题，并且该主题位于足够可靠的系统上，以便定期与完整存储库队列管理器联系。

使用主题主机路由时，集群主题定义的位置非常重要，因为集群中的其他队列管理器会创建此队列管理器的通道，并向其发送预订信息和发布。 要选择托管主题定义的最佳队列管理器，您需要了解主题主机路由。 请参阅第 67 页的『发布/预订集群中的主题主机路由』。

如果您具有集群主题和本地主题对象，那么本地主题优先。 请参阅第 81 页的『具有相同名称的多个集群主题定义』。

有关用于显示集群主题的命令的信息，请参阅相关信息。

集群主题继承

通常，在集群发布/预订拓扑中发布和预订应用程序期望工作方式相同，无论它们连接到集群中的哪个队列管理器。 这就是集群管理的主题对象传播到集群中每个队列管理器的原因。

受管主题对象从主题树中更高的其他受管主题对象继承其行为。 当没有为主题参数设置显式值时，将发生此继承。

在集群发布/预订的情况下，考虑此类继承很重要，因为这会引入发布者和订户根据其连接到的队列管理器的不同行为方式的可能性。 如果集群主题对象保留从更高主题对象继承的任何参数，那么该主题在集群中的不同队列管理器上的行为可能不同。 类似地，在主题树中的集群主题对象下定义的本地定义主题对象将意味着那些较低级别的主题仍处于集群状态，但本地对象可能会以与集群中的其他队列管理器不同的方式更改其行为。

通配符预订

当对在集群主题对象处或下方解析的主题字符串进行本地预订时，将创建代理预订。 如果在主题层次结构中进行的通配符预订高于任何集群主题，那么不会在集群周围为匹配的集群主题发送代理预订，因此不会从集群的其他成员接收任何发布。 但是，它会从本地队列管理器接收发布。

但是，如果另一个应用程序预订解析为集群主题或低于集群主题的主题字符串，那么将生成代理预订并将发布内容传播到此队列管理器。 原件到达时，更高的通配符预订将被视为这些出版物的合法接收方并接收副本。 如果不需要此行为，请在集群主题上设置 **WILDCARD(BLOCK)** 。 这将使原始通配符不被视为合法预订，并使其停止接收集群主题或其子主题上的任何发布 (本地发布或来自集群中的其他位置)。

相关概念

处理管理主题

使用预订

相关参考

显示 TOPIC

显示主题状态

显示

集群主题属性

当主题对象设置了集群名称属性时，将在集群中的所有队列管理器之间传播主题定义。 每个队列管理器都使用传播的主题属性来控制发布/预订应用程序的行为。

主题对象具有许多适用于发布/预订集群的属性。 部分控制发布和预订应用程序的一般行为，部分控制如何在集群中使用主题。

必须以集群中所有队列管理器都可以正确使用集群主题对象定义的方式进行配置。

例如，如果要用于受管预订 (MDURMDL 和 MNDURMDL) 的模型队列 设置为非缺省队列名称，必须在将创建受管预订的所有队列管理器上定义指定的模型队列。

同样，如果任何属性设置为 ASPARENT，那么主题的行为将依赖于主题树中的较高节点 (请参阅 管理主题对象) 在集群中的每个单独队列管理器上。 在从不同的队列管理器发布或预订时，这可能会导致不同的行为。

与集群中的发布/预订行为直接相关的主要属性如下所示:

CLROUTE

此参数控制连接了发布程序的队列管理器与存在匹配预订的队列管理器之间的消息路由。

- 您可以将路由配置为在这些队列管理器之间直接执行，或者通过托管集群主题定义的队列管理器执行。 有关更多详细信息，请参阅 发布/预订集群 。
- 在设置 **CLUSTER** 参数时，无法更改 **CLROUTE** 。 要更改 **CLROUTE**，请首先将 **CLUSTER** 属性设置为空白。 这将阻止使用该主题的应用程序以集群方式执行操作。 这反过来会导致将发布传递到预订的中断，因此您在进行更改时也应该停顿发布/预订消息传递。

PROXYSUB

此参数控制何时进行代理预订。

- FIRSTUSE 是缺省值，会导致发送代理预订以响应分布式发布/预订拓扑中队列管理器上的本地预订，并在不再需要时取消。 有关您可能希望将此属性从缺省值 FIRSTUSE 更改为缺省值的原因的详细信息，请参阅 各个代理预订转发和 处处发布 。
- 要启用 所有位置发布，请将高级主题对象的 **PROXYSUB** 参数设置为 FORCE 。 这将导致单个通配符代理预订与主题树中此主题对象下的所有主题匹配。

注: 在大型或繁忙的发布/预订集群中设置 **PROXYSUB(FORCE)** 属性可能会导致系统资源负载过高。 **PROXYSUB(FORCE)** 属性将传播到每个队列管理器，而不仅仅是定义主题的队列管理器。 这将导致集群中的每个队列管理器创建通配符代理预订。

在集群中的任何队列管理器上发布的此主题的消息副本将直接或通过主题主机队列管理器 (具体取决于 **CLROUTE** 设置) 发送到集群中的每个队列管理器。

当直接路由主题时，每个队列管理器都会创建集群发送方通道到其他每个队列管理器。 当主题是主题主机路由时，将从集群中的每个队列管理器创建到每个主题主机队列管理器的通道。

有关在集群中使用 **PROXYSUB** 参数的更多信息，请参阅 直接路由发布/预订性能。

PUBSCOB 和 SUBSCOPE

这些参数确定此队列管理器是将发布传播到拓扑 (发布/预订集群或层次结构) 中的队列管理器，还是将作用域限制为仅其本地队列管理器。 您可以使用 MQPMO_SCOPE_QMGR 和 MQSO_SCOPE_QMGR 以编程方式执行等效作业。

PUBSCOPE

如果使用 **PUBSCOPE(QMGR)**定义了集群主题对象，那么该定义将与集群共享，但基于该主题的发布的作用域仅为本地，并且不会将其发送到集群中的其他队列管理器。

SUBSCOPE

如果使用 **SUBSCOPE(QMGR)**定义了集群主题对象，那么该定义将与集群共享，但基于该主题的预订的作用域仅为本地，因此不会将代理预订发送到集群中的其他队列管理器。

这两个属性通常一起用于隔离队列管理器，使其不与特定主题上的集群其他成员进行交互。 队列管理器既不向集群的其他成员发布这些主题的出版物，也不从集群的其他成员接收这些主题的出版物。 如果在子主题上定义了主题对象，那么此情境不会阻止发布或预订。

在主题的本地定义上将 **SUBSCOPE** 设置为 QMGR 不会阻止集群中的其他队列管理器将其代理预订传播到队列管理器 (如果他们正在将该主题的集群版本与 **SUBSCOPE(ALL)**配合使用)。 但是，如果本地定义还将 **PUBSCOPE** 设置为 QMGR ，那么不会从此队列管理器向这些代理预订发送发布。

相关概念

发布作用域

预订作用域

具有相同名称的多个集群主题定义

您可以在集群中的多个队列管理器上定义相同的指定集群主题对象，在某些情况下，这将启用特定行为。 当存在多个同名的集群主题定义时，大多数属性应该匹配。 如果不匹配，那么将根据不匹配的重要性来报告错误或警告。

通常，如果多个集群主题定义的属性不匹配，那么将发出警告，并且集群中的每个队列管理器将使用其中一个主题对象定义。 每个队列管理器使用的定义不是确定性的，或者在集群中的队列管理器之间是一致的。 这种不匹配应该尽快解决。

在集群设置或维护期间，有时需要创建多个不相同的集群主题定义。 但是，这仅作为临时措施有用，因此被视为潜在的错误情况。

检测到不匹配时，会将以下警告消息写入每个队列管理器的错误日志:

- Multi 在 多平台 上，AMQ9465 和 AMQ9466。
- z/OS 在 z/OS 上，CSQX465I 和 CSQX466I。

可以通过查看主题状态而不是主题对象定义 (例如，使用 **DISPLAY TPSTATUS**) 来确定每个队列管理器上任何主题字符串的所选属性。

在某些情况下，配置属性中的冲突严重到足以停止正在创建的主题对象，或者导致不匹配的对象被标记为无效且未在集群中传播 (请参阅 DISPLAY TOPIC 中的 **CLSTATE**)。 当集群路由属性 (**CLROUTE**) 中存在冲突时，会发生这些情况 主题定义。 此外，由于主题主机路由定义之间一致性的重要性，将拒绝进一步的不一致，如本文的后续部分中所述。

如果在定义对象时检测到冲突，那么将拒绝配置更改。 如果稍后由完整存储库队列管理器检测到，那么会将以下警告消息写入队列管理器错误日志:

- Multi 在 多平台上: AMQ9879
- z/OS 在 z/OS 上: CSQX879E。

在集群中定义同一主题对象的多个定义时，本地定义的定义优先于任何远程定义的定义。 因此，如果定义中存在任何差异，那么托管多个定义的队列管理器的行为将彼此不同。

定义与另一个队列管理器中的集群主题同名的非集群主题的影响

可以定义未在集群中的队列管理器上集群的受管主题对象，并同时定义与其他队列管理器上的集群主题定义相同的指定主题对象。 在这种情况下，本地定义的主题对象优先于所有同名的远程定义。

这将在从此队列管理器中使用时阻止主题的集群行为。 即，预订可能不会接收来自远程发布程序的发布，并且来自发布程序的消息可能不会传播到集群中的远程预订。

在配置此类系统之前应仔细考虑，因为这可能会导致混淆行为。

注: 如果单个队列管理器需要阻止发布和预订在集群中传播，即使主题已在其他位置进行集群，另一种方法是将发布和预订作用域设置为仅本地队列管理器。 请参阅第 80 页的『集群主题属性』。

直接路由集群中的多个集群主题定义

对于直接路由，通常不会在多个集群队列管理器上定义相同的集群主题。 这是因为直接路由使该主题在集群中的所有队列管理器上都可用，无论该主题是在哪个队列管理器上定义的。 此外，添加多个集群主题定义会显着增加系统活动和管理复杂性，随着复杂性的增加，出现人为错误的可能性会更大:

- 每个定义都会导致将另一个集群主题对象推送到集群中的其他队列管理器 (包括其他集群主题主机队列管理器)。
- 集群中特定主题的所有定义必须相同，否则难以确定队列管理器正在使用哪个主题定义。

由于集群主题定义由完整的存储库队列管理器以及其部分集群存储库中的所有其他队列管理器进行高速缓存，因此唯一的主机队列管理器对于该主题在整个集群中正常运行并不重要。 有关更多信息，请参阅 使用直接路由的主题主机队列管理器的可用性。

对于可能需要在第二个队列管理器上临时定义集群主题的情况，例如，要从集群中除去该主题的现有主机时，请参阅 将集群主题定义移至其他队列管理器。

如果您需要更改集群主题定义，请在定义该集群主题定义的队列管理器上谨慎进行修改。 尝试从另一个队列管理器对其进行修改时，可能会意外地创建具有冲突主题属性的主题的第二个定义。

主题主机路由集群中的多个集群主题定义

当使用 主题主机的集群路由定义集群主题时，该主题将在集群中的所有队列管理器之间传播，就像针对 直接 路由的主题一样。 此外，该主题的所有发布/预订消息传递都将通过定义该主题的队列管理器进行路由。 因此，集群中主题的位置和定义数变得很重要 (请参阅 第 67 页的『发布/预订集群中的主题主机路由』)。

为了确保足够的可用性和可伸缩性，如果可能，具有多个主题定义是很有用的。 请参阅 使用主题主机路由的主题主机队列管理器的可用性。

在集群中添加或除去 主题主机 路由主题的其他定义时，应考虑配置更改时的消息流。 如果在更改时集群中正在将消息发布到主题，那么需要一个登台过程来添加或除去主题定义。 请参阅 将集群主题定义移动到其他队列管理器 和 将额外的主题主机添加到主题主机路由的集群。

如前所述，多个定义的属性应该匹配，但 **PUB** 参数可能例外，如下一节中所述。 当通过主题主机队列管理器路由发布时，更重要的是使多个定义保持一致。 因此，如果为主题主机集群路由配置了一个或多个主题定义，那么将拒绝在主题字符串或集群名称中检测到的不一致。

注: 如果尝试在主题树中的另一个主题上方或下方配置集群主题定义，其中配置了现有集群主题定义以进行主题主机路由，那么也会拒绝集群主题定义。 这可防止在与通配符预订相关的发布路由中出现岐义。

PUB 参数的特殊处理

PUB 参数用于控制应用程序何时可以发布到主题。 对于集群中的主题主机路由，它还可以控制使用哪些主题主机队列管理器来路由发布。 因此，允许它在集群中具有同一主题对象的多个定义，以及 PUB 参数的不同设置。

如果主题的多个远程集群定义具有此参数的不同设置，那么当满足以下条件时，该主题允许将发布发送并传递到预订:

- 在将发布程序连接到的队列管理器上未定义匹配的主题对象，该对象设置为 PUB(DISABLED)。

- 集群中的一个或多个多主题定义设置为 PUB(ENABLED)，或者一个或多个多主题定义设置为 PUB(ASPARENT)，并且在主题树中的较高位置将连接发布程序和定义的预订的本地队列管理器设置为 PUB(ENABLED)。

对于主题主机路由，当消息由连接到不是主题主机的队列管理器的应用程序发布时，仅将消息路由到未显式将 **PUB** 参数设置为 DISABLED 的主题托管队列管理器。 因此，您可以使用 PUB(DISABLED) 设置来停顿通过特定主题主机的消息流量。 您可能希望执行此操作以准备维护或除去队列管理器，或者出于 向主题主机路由集群添加额外主题主机中描述的原因。

集群主题主机队列管理器的可用性
设计发布/预订集群以将主题主机队列管理器变为不可用时集群将无法处理主题流量的风险降至最低。 主题主机队列管理器变为不可用的影响取决于集群是使用主题主机路由还是直接路由。

使用直接路由的主题主机队列管理器的可用性

对于直接路由，通常不会在多个集群队列管理器上定义相同的集群主题。 这是因为直接路由使该主题在集群中的所有队列管理器上都可用，无论该主题是在哪个队列管理器上定义的。请参阅 直接路由集群中的多个集群主题定义。

在集群中，每当集群对象 (例如，集群队列或集群主题) 的主机长时间不可用时，集群的其他成员将最终使这些对象的知识到期。 对于集群主题，如果集群主题主机队列管理器变为不可用，那么其他队列管理器将继续以直接集群方式处理主题的发布/预订请求 (即，向远程队列管理器上的预订发送发布)，至少从主题托管队列管理器上次与完整存储库队列管理器进行通信的时间开始 60 天。 如果定义了集群主题对象的队列管理器不再可用，那么最终将删除其他队列管理器上的高速缓存主题对象，并且该主题将还原为本地主题，在这种情况下，预订将停止接收来自连接到远程队列管理器的应用程序的发布。

使用 60 天时间段来恢复您在其中定义集群主题对象的队列管理器，几乎不需要采取特殊措施来保证集群主题主机保持可用状态 (但是请注意，在不可用的集群主题主机上定义的任何预订都不会保持可用状态)。 60 天的时间足以迎合技术问题，仅因行政错误就有可能被超过。 为了降低这种可能性，如果集群主题主机不可用，那么集群的所有成员将每小时写入错误日志消息，并声明其高速缓存的集群主题对象未刷新。 通过确保定义了集群主题对象的队列管理器正在运行来响应这些消息。 如果无法使集群主题主机队列管理器再次可用，请在集群中的另一个队列管理器上定义具有完全相同的属性的同一集群主题定义。

使用主题主机路由的主题主机队列管理器的可用性

对于主题主机路由，主题的所有发布/预订消息传递都通过定义该主题的队列管理器进行路由。 因此，考虑集群中这些队列管理器的持续可用性非常重要。 如果主题主机变为不可用，并且该主题不存在其他主机，那么从发布者到集群中不同队列管理器上的订户的流量将立即停止该主题。 如果其他主题主机可用，那么集群队列管理器将通过这些主题主机来路由新的发布流量，从而提供消息路由的持续可用性。

对于直接主题，在 60 天后，如果第一个主题主机仍然不可用，那么将从集群中除去对该主题主机的主题的了解。 如果这是集群中此主题的最后一个剩余定义，那么所有其他队列管理器将停止将发布转发到任何主题主机以进行路由。

因此，为了确保足够的可用性和可伸缩性，如果可能，可以在至少两个集群队列管理器上定义每个主题。 这提供了防止任何给定主题主机队列管理器变为不可用的保护。 另请参阅 主题主机路由集群中的多个集群主题定义。

如果无法配置多个主题主机 (例如，因为需要保留消息排序)，并且无法仅配置一个主题主机 (因为单个队列管理器的可用性不得影响发布到集群中所有队列管理器中的预订的流)，请考虑将该主题配置为直接路由的主题。 这避免了对整个集群的单个队列管理器的依赖，但仍要求每个单独的队列管理器都可用，以便它处理本地托管的预订和发布程序。

禁止集群发布/预订

将第一个直接路由的集群主题引入到集群中会强制集群中的每个队列管理器了解其他每个队列管理器，并可能导致它们相互创建通道。 如果不需要这样做，那么您应该改为配置主题主机路由发布/预订。 如果存在直接路由的集群主题可能会危及集群的稳定性，那么由于每个队列管理器的缩放问题，您可以通过在集群中的每个队列管理器上将 **PSCLUS** 设置为 DISABLED 来完全禁用集群发布/预订功能。

如第 62 页的『发布/预订集群中的直接路由』中所述，当您将直接路由的集群主题引入到集群中时，将自动通知所有部分存储库该集群的所有其他成员。 集群主题还可能在所有其他节点 (例如，指定了

PROXYSUB(FORCE)) 创建预订，并导致从队列管理器启动大量通道，即使没有本地预订也是如此。 这会使集群中的每个队列管理器立即产生额外负载。 对于包含许多队列管理器的集群，这可能会导致性能显着降低。 因此，必须仔细规划对集群的直接路由发布/预订的引入。

当您知道集群无法容纳直接路由的发布/预订的开销时，可以改为使用主题主机路由的发布/预订。 有关差异的概述，请参阅 第 61 页的『设计发布/预订集群』。

如果您希望完全禁用集群的发布/预订功能，那么可以通过在集群中的每个队列管理器上将队列管理器属性 **PSCLUS** 设置为 DISABLED 来执行此操作。 此设置通过修改队列管理器功能的三个方面来禁用集群中的直接路由和主题主机路由发布/预订:

- 此队列管理器的管理员不再能够将 Topic 对象定义为集群对象。
- 将拒绝来自其他队列管理器的入局主题定义或代理预订，并且将记录一条警告消息以通知管理员不正确的配置。
- 当完整存储库接收到主题定义时，它们不再自动与所有其他部分存储库共享有关每个队列管理器的信息。

虽然 **PSCLUS** 是集群中每个单独队列管理器的参数，但它并非旨在选择性地禁用集群中队列管理器子集中的发布/预订。 如果以这种方式选择性地禁用，那么您将看到频繁的错误消息。 这是因为如果在启用了 **PSCLUS** 的队列管理器上集群某个主题，那么会不断看到和拒绝代理预订和主题定义。

因此，您应该在集群中的每个队列管理器上将 **PSCLUS** 设置为 DISABLED 。 但是，在实践中，此状态可能难以实现和维护，例如，队列管理器可以随时加入和离开集群。 至少，必须确保在所有完整存储库队列管理器上将 **PSCLUS** 设置为 DISABLED 。 如果执行此操作，并且随后在集群中的 ENABLED 队列管理器上定义了集群主题，那么这不会导致完整存储库通知每个其他队列管理器的每个队列管理器，因此将保护集群免受所有队列管理器中的潜在缩放问题影响。 在此场景中，将在完整存储库队列管理器的错误日志中报告集群主题的源。

如果队列管理器参与一个或多个发布/预订集群以及一个或多个点到点集群，那么必须在该队列管理器上将 **PSCLUS** 设置为 ENABLED 。 因此，将点到点集群与发布预订集群重叠时，应该在每个集群中使用一组单独的完整存储库。 此方法允许有关每个队列管理器的主题定义和信息仅在发布/预订集群中流动。

在将 **PSCLUS** 从 ENABLED 更改为 DISABLED 时，为了避免配置不一致，此队列管理器所属的任何集群中都不能存在任何集群主题对象。 在将 **PSCLUS** 更改为 DISABLED 之前，必须删除任何此类主题 (即使是远程定义的主题)。

有关 **PSCLUS** 的更多信息，请参阅 ALTER QMGR (PSCLUS)。

相关概念

直接路由的发布/预订集群性能

发布/预订和多个集群

单个队列管理器可以是多个集群的成员。 此排列有时称为 重叠集群。 通过这种重叠，可以从多个集群访问集群队列，并且可以将点到点消息流量从一个集群中的队列管理器路由到另一个集群中的队列管理器。 发布/预订集群中的集群主题不提供相同的功能。 因此，在使用多个集群时，必须清楚地了解它们的行为。

与队列不同，不能将主题定义与多个集群相关联。 集群主题的作用域仅限于为其定义主题的同一集群中的那些队列管理器。 这允许将发布仅传播到同一集群中的那些队列管理器上的预订。

队列管理器的主题树

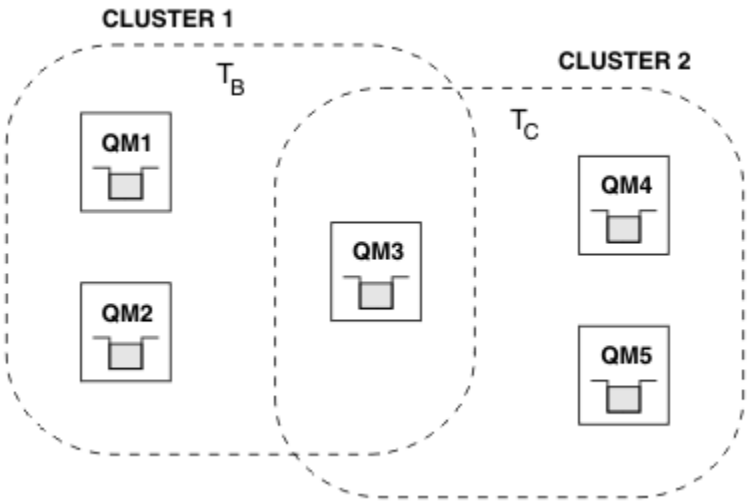


图 *28:* 重叠集群*:* 两个集群各预订不同的主题

当队列管理器是多个集群的成员时，将使其了解其中每个集群中定义的所有集群主题。 例如，在上图中，QM3 同时识别 T_B 和 T_C 受管集群主题对象，而 QM1 仅识别 T_B。 QM3 将这两个主题定义应用于其本地主题，因此对于某些主题具有不同的行为 QM1 。 因此，不同集群中的集群主题互不干扰非常重要。 当一个集群主题在另一个集群中的另一个集群主题 (例如，它们的主题字符串为 `/Sport` 和 `/Sport/Football`) 的上方或下方定义时，甚至对于这两个集群中的同一主题字符串，都可能发生干扰。 另一种干扰形式是在不同集群中使用相同的对象名定义受管集群主题对象时，但针对不同的主题字符串。

如果进行了此类配置，那么将发布传递到匹配的预订将非常依赖于发布者和订户相对于集群的相对位置。 因此，您不能依赖此类配置，您应该对其进行更改以除去干扰主题。

使用发布/预订消息传递来规划重叠集群拓扑时，可以通过将主题树和集群主题对象名称视为跨拓扑中所有重叠集群来避免任何干扰。

集成多个发布/预订集群

如果需要发布/预订消息传递以跨不同集群中的队列管理器，那么有两个选项可用:

- 通过使用发布/预订层次结构配置将集群连接在一起。 请参阅 组合多个集群的主题空间。
- 创建覆盖现有集群的其他集群，并包含需要发布或预订特定主题的所有队列管理器。

使用后一个选项时，您应该仔细考虑集群的大小以及最有效的集群路由机制。 请参阅第 61 页的『设计发布/预订集群』。

发布/预订集群中保留的发布的设计注意事项

在设计发布/预订集群以使用保留发布时，需要考虑一些限制。

注意事项

注意事项 *1:* 以下集群队列管理器始终存储保留发布的最新版本:

- 发布者的队列管理器
- 在主题主机路由集群中，主题主机 (前提是该主题只有一个主题主机，如本文下一节所述)
- 具有与保留发布的主题字符串匹配的预订的所有队列管理器

注意事项 *2:* 队列管理器在没有预订时不会接收更新的保留发布。 因此，存储在不再预订该主题的队列管理器上的任何保留发布将变得过时。

注意事项 *3:* 在创建任何预订时，如果存在主题字符串的保留发布的本地副本，那么本地副本将传递到预订。 如果您是任何给定主题字符串的第一个订户，那么还会从下列其中一个集群成员交付匹配的保留发布内容:

- 在直接路由的集群中，发布者的队列管理器
- 在主题主机路由集群中，给定主题的主题主机

将保留发布从主题主机或发布队列管理器传递到预订队列管理器与 MQSUB 调用异步。 因此，如果您使用 MQSUBRQ 调用，那么可能会错过最新的保留发布，直到后续调用 MQSUBRQ 为止。

影响

对于任何发布/预订集群，在进行第一个预订时，本地队列管理器可能正在存储保留发布的旧副本，这是传递到新预订的副本。 本地队列管理器上存在预订意味着这将在下次更新保留发布时解决。

对于主题主机路由的发布/预订集群，如果您为给定主题配置了多个主题主机，那么新订户可能会从主题主机接收最新保留发布，或者他们可能会从另一个主题主机接收旧的保留发布 (最新的已丢失)。 对于主题主机路由，通常为给定主题配置多个主题主机。 但是，如果您期望应用程序使用保留发布，那么应该仅为每个主题配置一个主题主机。

对于任何给定的主题字符串，您应该只使用单个发布程序，并确保发布程序始终使用相同的队列管理器。 如果不执行此操作，那么不同的保留发布可能在同一主题的不同队列管理器上处于活动状态，从而导致意外行为。 由于分发了多个代理预订，因此可能会接收到多个保留发布。

如果您仍然关注使用旧发布的订户，请考虑在创建每个保留发布时设置消息到期时间。

您可以使用 **CLEAR TOPICSTR** 命令从发布/预订集群中除去保留发布。 在某些情况下，您可能需要在发布/预订集群的多个成员上发出该命令，如 **CLEAR TOPICSTR** 中所述。

通配符预订和保留发布

如果您使用的是通配符预订，那么传递到发布/预订集群的其他成员的相应代理预订将在第一个通配符之前从主题分隔符进行通配符。 请参阅 通配符和集群主题。

因此，所使用的通配符可能与更多主题字符串和更多保留发布相匹配，而不是与预订应用程序相匹配。

这将增加保留发布所需的存储量，因此您需要确保主管队列管理器具有足够的存储容量。

相关概念
保留的发布内容
个人代理订阅转发和发布随处可见

针对发布/预订集群的 *REFRESH CLUSTER* 注意事项

发出 **REFRESH CLUSTER** 命令会导致队列管理器临时废弃本地保存的有关集群的信息，包括任何集群主题及其关联的代理预订。

从发出 **REFRESH CLUSTER** 命令到队列管理器完全了解集群发布/预订的必要信息所花费的时间取决于集群的大小，可用性以及完整存储库队列管理器的响应能力。

在刷新处理期间，会中断发布/预订集群中的发布/预订流量。 对于大型集群，使用 **REFRESH CLUSTER** 命令可能会在集群进行中时中断集群，并且在集群对象自动向所有相关队列管理器发送状态更新后的 27 天时间间隔内再次中断集群。 请参阅在大型集群中刷新可能会影响集群的性能和可用性。 由于这些原因，只有在 IBM 支持中心的指导下，才必须在发布/预订集群中使用 **REFRESH CLUSTER** 命令。

对集群的中断可能会在外部显示为以下症状:

- 此队列管理器上集群主题的预订未从连接到集群中其他队列管理器的发布程序接收发布。
- 发布到此队列管理器上的集群主题的消息不会传播到其他队列管理器上的预订。
- 在此期间创建的此队列管理器上的集群主题预订未一致地将代理预订发送给集群的其他成员。
- 在此期间删除的此队列管理器上的集群主题预订无法一致地从集群的其他成员中除去代理预订。
- 在消息传递中暂停 10 秒或更长时间。
- **MQPUT** 失败，例如， MQRC_PUBLICATION_FAILURE。
- 放置在死信队列上的发布，原因为 MQRC_UNKNOWN_REMOTE_Q_MGR

由于这些原因，需要先停顿发布/预订应用程序，然后再发出 **REFRESH CLUSTER** 命令。

在发布/预订集群中的队列管理器上发出 **REFRESH CLUSTER** 命令后，请等待直到成功刷新所有集群队列管理器和集群主题，然后再同步代理预订，如 代理预订再同步中所述。 当所有代理预订都已正确再同步时，请重新启动发布/预订应用程序。

如果 **REFRESH CLUSTER** 命令需要很长时间才能完成，请通过查看 SYSTEM.CLUSTER.COMMAND.QUEUE 的 CURDEPTH 来对其进行监视。

相关概念

第 56 页的『集群：使用 REFRESH CLUSTER 最佳实践』

使用 **REFRESH CLUSTER** 命令可废弃有关集群的所有本地保存的信息，并从集群中的完整存储库重建该信息。 您不应该使用此命令，除非在特殊情况下。 如果确实需要使用它，那么有关如何使用它有特殊注意事项。 此信息是基于测试和客户反馈的指南。

相关参考

运行 REFRESH CLUSTER 时发现的应用程序问题

MQSC 命令参考：REFRESH CLUSTER

在发布/预订层次结构中进行路由

如果分布式队列管理器拓扑是发布/预订层次结构，并且在队列管理器上进行了预订，那么缺省情况下将在层次结构中的每个队列管理器上创建代理预订。 然后，在任何队列管理器上接收到的发布将通过层次结构路由到托管匹配预订的每个队列管理器。

有关如何在发布/预订层次结构和集群中的队列管理器之间路由消息的简介，请参阅 分布式发布/预订网络。

在分布式发布/预订层次结构中的队列管理器上预订主题时，队列管理器将管理将预订传播到已连接的队列管理器的过程。 代理预订 流至网络中的所有队列管理器。 代理预订向队列管理器提供将发布转发到托管该主题的预订的那些队列管理器所需的信息。 发布/预订层次结构中的每个队列管理器仅知道其直接关系。 放入一个队列管理器的发布通过其直接关系发送给那些具有预订的队列管理器。 下图对此进行了说明，其中 订户 *1* 在 亚洲 队列管理器 (1) 上注册特定主题的预订。 *Asia* 队列管理器上此预订的代理预订将转发到网络中的所有其他队列管理器 (2,3, 4)。

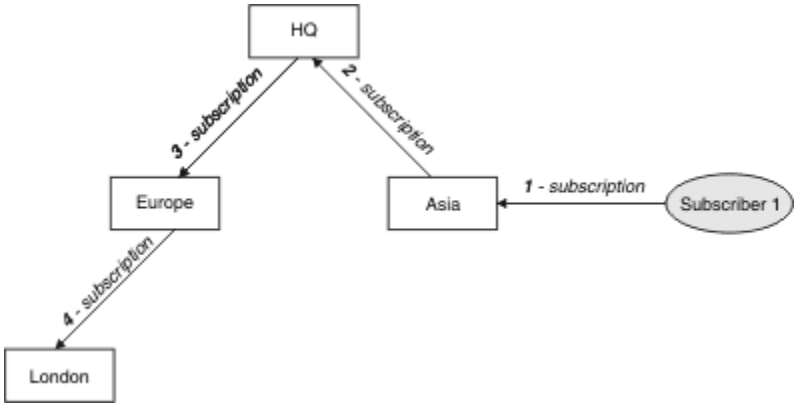


图 *29:* 通过队列管理器网络传播预订

队列管理器将合并在其上创建的所有预订，无论是从本地应用程序还是从远程队列管理器。 除非代理预订已存在，否则它将为具有其邻居的预订的主题创建代理预订。 下图对此进行了说明，其中 *Subscriber 2* 在 *HQ* 队列管理器 (5) 上注册了与 第 87 页的图 29 中相同的主题的预订。 此主题的预订将转发到 亚洲 队列管理器，以便它知道网络 (6) 上的其他位置存在预订。 不会将预订转发到 欧洲 队列管理器，因为已注册此主题的预订; 请参阅 第 87 页的图 29 中的步骤 3。

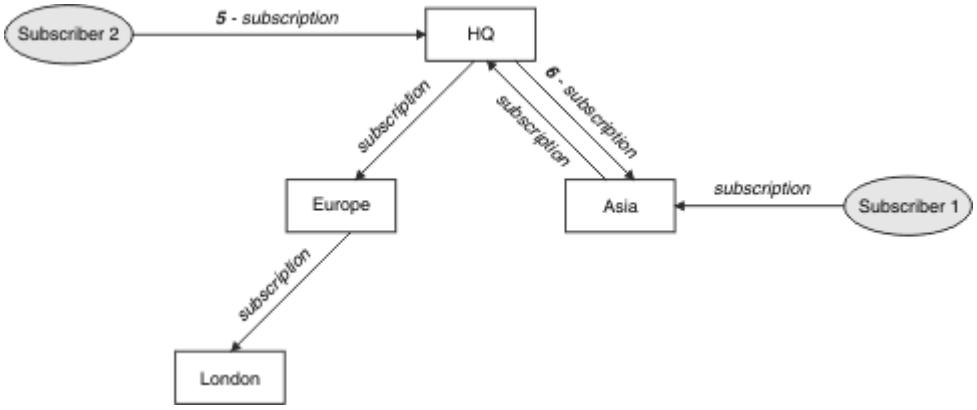


图 *30:* 多个预订

当应用程序将信息发布到主题时，缺省情况下，接收队列管理器会将其转发到具有该主题的有效预订的所有队列管理器。 它可以通过一个或多个中间队列管理器将其转发。 下图对此进行了说明，在该图中，发布者将发布内容 (与 第 88 页的图 30 中的主题相同) 发送到 欧洲 队列管理器 (7)。 此主题的预订从 *HQ* 到 欧洲，因此该出版物将转发到 *HQ* 队列管理器 (8)。 但是，不存在从 伦敦 到 欧洲 (仅从 欧洲 到 伦敦) 的预订，因此不会将该出版物转发到 伦敦 队列管理器。 *HQ* 队列管理器将发布直接发送到 *Subscriber 2* 和 *Asia* 队列管理器 (9)。 该出版物将从 亚洲 (10) 转发到 订户 *1* 。

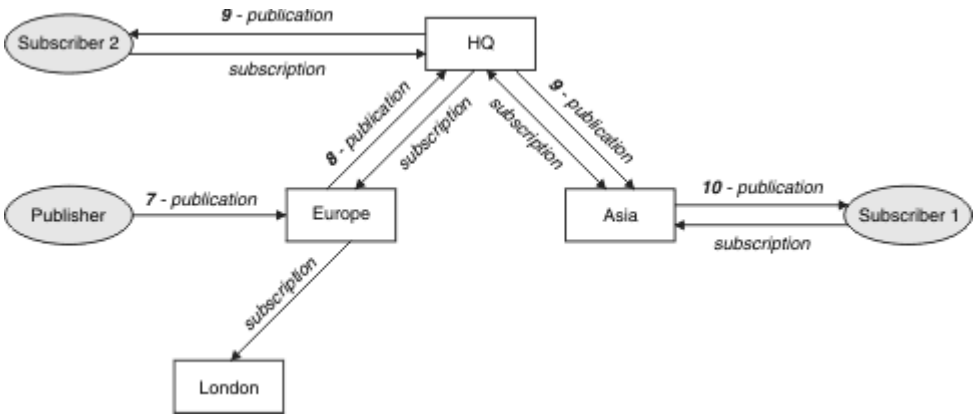


图 *31:* 通过队列管理器网络传播发布

当队列管理器将任何发布或预订发送到另一个队列管理器时，它会在消息中设置自己的用户标识。 如果您正在使用发布/预订层次结构，并且如果将入局通道设置为在消息中放入具有用户标识权限的消息，那么必须对发送队列管理器的用户标识进行授权。 请参阅 将缺省用户标识与队列管理器层次结构配合使用。

注: 如果改为使用发布/预订集群，那么授权由集群处理。

摘要和其他注意事项

发布/预订层次结构使您能够精确控制队列管理器之间的关系。 创建后，无需手动干预即可进行管理。 但是，这也会对您的系统施加某些约束:

- 层次结构中的较高节点 (尤其是根节点) 必须托管在强大，高可用性和高性能的设备上。 这是因为预期会有更多发布流量流经这些节点。
- 层次结构中每个非叶队列管理器的可用性会影响网络将消息从发布者流向其他队列管理器上的订户的能力。
- 缺省情况下，预订的所有主题字符串都在整个层次结构中传播，发布仅传播到预订了关联主题的远程队列管理器。 因此，对预订集的快速更改可能成为限制因素。 您可以更改此缺省行为，而是将所有发布传播到所有队列管理器，这将除去对代理预订的需求。 请参阅 发布/预订网络中的预订性能。

注: 类似的限制也适用于直接路由的集群。

- 由于发布/预订队列管理器的互连性质，代理预订在网络中的所有节点之间传播需要时间。 远程发布不一定会立即开始预订，因此在预订新主题字符串之后可能不会发送早期发布。 您可以通过将所有发布传播到所有队列管理器来除去预订延迟所导致的问题，这将除去对代理预订的需求。 请参阅 发布/预订网络中的预订性能。

  注: 此限制也适用于直接路由集群。
- 对于发布/预订层次结构，添加或除去队列管理器需要对该层次结构进行手动配置，并仔细考虑这些队列管理器的位置及其对其他队列管理器的依赖。 除非要添加或除去位于层次结构底部的队列管理器，因此在它们下面没有更多分支，否则还必须在层次结构中配置其他队列管理器。

在使用发布/预订层次结构作为路由机制之前，请探索 第 62 页的『发布/预订集群中的直接路由』 和 第 67 页的『发布/预订集群中的主题主机路由』 中详细描述的替代方法。

分布式发布/预订系统队列

队列管理器使用四个系统队列进行发布/预订消息传递。 仅出于问题确定和容量规划目的，您需要了解它们是否存在。

请参阅 在发布/预订网络中均衡生产者和使用者 ，以获取有关如何监视这些队列的指导信息。

表 *6: Multiplatforms* 版上的发布/预订系统队列

系统队列	**用途**
SYSTEM.INTER.QMGR.CONTROL	IBM MQ 分布式发布/预订控制队列
SYSTEM.INTER.QMGR.FANREQ	IBM MQ 分布式发布/预订内部代理预订扇出进程输入队列
SYSTEM.INTER.QMGR.PUBS	IBM MQ 分布式发布/预订发布
SYSTEM.HIERARCHY.STATE	IBM MQ 分布式发布/预订层次结构关系状态

z/OS 在 z/OS 上，通过在 CSQINP2 初始化输入数据集中包含 CSQ4INSX， CSQ4INSR 和 CSQ4INSG 样本，在创建队列管理器时设置必要的系统对象。 有关更多信息，请参阅 任务 13: 定制初始化输入数据集。

发布/预订系统队列的属性显示在 第 89 页的表 7 中。

表 *7:* 发布/预订系统队列的属性

属性	**缺省值**
DEFPSIST	Yes
DEFSOPT	SHARED
MAXMSGL	Multi 在 多平台上 :ALTER QMGR 命令的 MAXMSGL 参数值 z/OS 在 z/OS 上: 4194304 (即， 4 MB)
MAXDEPTH	999999999
SHARE	不适用
z/OS z/OS STGCLASS	此属性仅在 z/OS 平台上使用

注: 包含应用程序放入的消息的唯一队列是 SYSTEM.INTER.QMGR.PUBS。 **MAXDEPTH** 设置为此队列的最大值，以允许在中断或过度装入期间临时构建已发布的消息。 如果队列管理器正在无法包含该队列深度的系统上运行，那么应该对其进行调整。

相关任务

对分布式发布/预订问题进行故障诊断

分布式发布/预订系统队列错误

当分布式发布/预订队列管理器队列不可用时，可能会发生错误。 这将影响在发布/预订网络中传播预订知识，并将其发布到远程队列管理器上的预订。

如果扇出请求队列 SYSTEM.INTER.QMGR.FANREQ 不可用，那么创建预订可能会生成错误，当需要将代理预订传递到直接连接的队列管理器时，会将错误消息写入队列管理器错误日志。

如果层次结构关系状态队列 SYSTEM.HIERARCHY.STATE 不可用，那么会将错误消息写入队列管理器错误日志，并将发布/预订引擎置于 COMPAT 方式。 要查看发布/预订方式，请使用命令 DISPLAY QMGR PSMODE。

如果任何其他 SYSTEM.INTER.QMGR 队列不可用，那么会将一条错误消息写入队列管理器错误日志，并且尽管功能未禁用，但发布/预订消息可能会在此队列管理器或远程队列管理器上的队列上构建。

如果发布/预订系统队列或到父，子或发布/预订集群队列管理器的必需传输队列不可用，那么将发生以下结果:

- 未交付发布，发布应用程序可能会接收到错误。 有关发布应用程序何时接收到错误的详细信息，请参阅 **DEFINE TOPIC** 命令的以下参数: **PMSGDLV**， **NPMSGDLV** 和 **USEDLQ**。
- 接收到的队列间管理器发布将回退到输入队列，然后重新尝试。 如果达到回退阈值，那么会将未交付的发布放在死信队列上。 队列管理器错误日志将包含问题的详细信息。
- 未交付的代理预订将回退到扇出请求队列，然后再次尝试。 如果达到回退阈值，那么未交付的代理预订不会交付到任何已连接的队列管理器，并且会放置在死信队列上。 队列管理器错误日志将包含问题的详细信息，包括所需的任何必需纠正管理操作的详细信息。
- 层次结构关系协议消息失败，连接状态标记为 ERROR。 要查看连接状态，请使用命令 **DISPLAY PUBSUB**。

相关任务

对分布式发布/预订问题进行故障诊断

Multi 规划 Multiplatforms 版上的存储和性能需求

您必须为 IBM MQ 系统设置切合实际且可实现的存储和性能目标。 使用这些链接可了解影响平台上的存储和性能的因素。

根据您要在其中使用 IBM MQ 的系统以及要使用的组件不同，需求也会有所不同。

有关受支持的硬件和软件环境的最新信息，请参阅 IBM MQ 的系统需求。

IBM MQ 将队列管理器数据存储在文件系统中。 使用以下链接来了解有关规划和配置目录结构以与 IBM MQ 配合使用的信息:

- 第 93 页的『多平台上的规划文件系统支持』
- 第 93 页的『Multiplatforms 版上共享文件系统的需求』
- 第 102 页的『在 Multiplatforms 版上共享 IBM MQ 文件』
- Linux AIX 第 103 页的『AIX and Linux 系统上的目录结构』
- Windows 第 112 页的『Windows 系统上的目录结构』
- IBM i 第 115 页的『IBM i 上的目录结构』

使用以下链接以获取有关 AIX and Linux 上的系统资源，共享内存和进程优先级的信息:

- Linux AIX 第 119 页的『IBM MQ 和 UNIX System V IPC 资源』
- AIX 第 119 页的『AIX 上的共享内存』
- Linux AIX 第 119 页的『IBM MQ 和 UNIX 进程优先级』

使用以下链接以获取有关日志文件的信息:

- 第 118 页的『在 Multiplatforms 版上选择循环或线性日志记录』
- 计算日志大小

相关概念

第 119 页的『Planning your IBM MQ environment on z/OS』
When planning your IBM MQ environment, you must consider the resource requirements for data sets, page sets, Db2, Coupling Facilities, and the need for logging, and backup facilities. Use this topic to plan the environment where IBM MQ runs.

相关任务

第 5 页的『规划 IBM MQ 体系结构』
规划 IBM MQ 环境时，请考虑 IBM MQ 为单个和多个队列管理器体系结构以及点到点和发布/预订消息传递样式提供的支持。 还要规划资源需求以及日志记录和备份工具的使用。

相关参考

AIX and Linux 上的硬件和软件需求
Windows 上的硬件和软件需求

Multi Multiplatforms 版上的磁盘空间需求

IBM MQ 的存储需求取决于您安装的组件以及所需的工作空间量。

您选择安装的可选组件 (包括所需的任何必备组件) 需要磁盘存储器。 总存储需求还取决于您使用的队列数，队列上消息的数量和大小以及消息是否持久。 您还需要磁盘，磁带或其他介质上的归档容量以及您自己的应用程序的空间。

下表显示在不同平台上安装产品的各种组合时所需的大致磁盘空间。 (值向上取整为最接近的 5 MB，其中 MB 为 1,048,576 字节。)

- LTS 第 91 页的『Long Term Support 的磁盘空间需求』
- CD 第 92 页的『Continuous Delivery 的磁盘空间需求』

Long Term Support 的磁盘空间需求

V 9.4.0 LTS

表 8: *IBM MQ Multiplatforms for Long Term Support* 的磁盘空间需求

平台	客户端安装第 92 页的『1』	服务器安装第 92 页的『2』	完全安装 第 92 页的『3』
AIX AIX	335 MB	375 MB	1810 MB
IBM i IBM i (请参阅 IBM i 的其他说明)	485 MB	845 MB	1965 MB
Linux Linux for x86-64	270 MB	295 MB	2010 MB
Linux Linux on Power Systems - Little Endian	170 MB	190 MB	1400 MB
Linux Linux for IBM Z	255 MB	290 MB	1485 MB
Windows Windows (64 位安装) 第 92 页的『4』	295 MB	425 MB	2310 MB

注意:

1. 客户机安装包含以下组件:
   - 运行时
   - 客户机
2. 服务器安装包含以下组件:
   - 运行时
   - 服务器
3. 完整安装包含所有可用组件。
4. Windows 并非此处列出的所有组件都是 Windows 系统上的可安装功能部件; 它们的功能有时包含在其他功能部件中。 请参阅 Windows 系统的 IBM MQ 功能部件。

IBM i 的其他说明: IBM i

1. 在 IBM i 上，无法将本机客户机与服务器分开。 表中的服务器图用于不带 Java 的 5724H72*BASE 以及英语语言装入 (2924)。 有 22 种可能的唯一语言负载。
2. 该表中的图适用于没有 Java 的本机客户机 5725A49 *BASE。
3. 可以将 Java 和 JMS 类添加到服务器和客户机绑定。 如果要包含这些功能部件，请添加 110 MB。
4. 将样本源添加到客户机或服务器将额外添加 10 MB。
5. 向 Java 和 JMS 类添加样本将添加额外的 5 MB。

Continuous Delivery 的磁盘空间需求

CD V 9.4.0

表 *9: IBM MQ Multiplatforms for Continuous Delivery* 的磁盘空间需求

平台/CD 发行版	客户端安装第 93 页的『1』	服务器安装第 93 页的『2』	完全安装 第 93 页的『3』
AIX **AIX**			
V 9.4.0 IBM MQ 9.4.0	355 MB	390 MB	1440 MB
Linux **Linux for x86-64 (64 位)**			
V 9.4.0 IBM MQ 9.4.0	280 MB	295 MB	1195 MB
Linux **Linux on Power Systems - Little Endian**			
V 9.4.0 IBM MQ 9.4.0	170 MB	195 MB	1075 MB
Linux **Linux for IBM Z**			
V 9.4.0 IBM MQ 9.4.0	260 MB	290 MB	1160 MB
Windows **Windows (64 位安装)** 第 93 页的『4』			
V 9.4.0 IBM MQ 9.4.0	300 MB	425 MB	1785 MB

注意:

1. 客户机安装包含以下组件:
   - 运行时
   - 客户机
2. 服务器安装包含以下组件:
   - 运行时
   - 服务器
3. 完整安装包含所有可用组件。
4. Windows 并非此处列出的所有组件都是 Windows 系统上的可安装功能部件; 它们的功能有时包含在其他功能部件中。 请参阅 Windows 系统的 IBM MQ 功能部件。

相关概念

IBM MQ 组件和功能

Multi 多平台上的规划文件系统支持

队列管理器数据存储在文件系统中。 队列管理器使用文件系统锁定来防止多实例队列管理器的多个实例同时处于活动状态。

共享文件系统

共享文件系统使多个系统能够同时访问同一物理存储设备。 如果多个系统直接访问同一物理存储设备，而没有某种强制锁定和并行控制的方法，那么将发生损坏。 操作系统为本地文件系统提供对本地进程的锁定和并行控制; 网络文件系统为分布式系统提供锁定和并行控制。

历史上，网络文件系统的执行速度不够快，或者提供了足够的锁定和并行控制，无法满足日志记录消息的要求。 如今，联网的文件系统可以提供良好的性能，实现可靠的网络文件系统协议，如 *RFC 3530* ，网络文件系统 *(NFS)* 版本 *4* 协议，满足可靠记录消息的需求。

共享文件系统和 IBM MQ

多实例队列管理器的队列管理器数据存储在共享网络文件系统中。 在 AIX, Linux, and Windows 系统上，队列管理器的数据文件和日志文件必须放在共享网络文件系统中。 IBM i 在 IBM i 上，将使用日记帐而不是日志文件，并且无法共享日记帐。 IBM i 上的多实例队列管理器使用日志复制或可切换日志，以使日志在不同队列管理器实例之间可用。

IBM MQ 使用锁定来防止同一多实例队列管理器的多个实例同时处于活动状态。 同一锁定还可确保两个单独的队列管理器不能无意中使用同一组队列管理器数据文件。 一次只能有一个队列管理器实例具有其锁定。 因此， IBM MQ 支持存储在作为共享文件系统访问的联网存储器上的队列管理器数据。

由于并非网络文件系统的所有锁定协议都是健全的，并且由于可能针对性能而非数据完整性配置了文件系统，因此您必须运行 **amqmfsck** 命令来测试网络文件系统是否将正确控制对队列管理器数据和日志的访问。 此命令仅适用于 UNIX， Linux 和 IBM i 系统。 在 Windows 上，只有一个受支持的网络文件系统，不需要 **amqmfsck** 命令。

相关任务

第 95 页的『在 Multiplatforms 版上验证共享文件系统行为』

运行 **amqmfsck** 以检查 AIX， Linux 或 IBM i 上的共享文件系统是否满足存储多实例队列管理器的队列管理器数据的需求。 (Windows 配置的唯一要求是它将 SMB 3 用于共享存储器供应。)

Multi Multiplatforms 版上共享文件系统的需求

共享文件系统必须提供数据写完整性，保证对文件的独占访问权，并在无法可靠地使用 IBM MQ 时释放锁定。

共享文件系统必须满足的需求

共享文件系统必须满足三个基本要求才能可靠地使用 IBM MQ:

1. 数据写入完整性

   数据写入完整性有时称为 清空时写入磁盘。 队列管理器必须能够与成功落实到物理设备的数据同步。 在事务系统中，您需要确保在继续其他处理之前已安全地落实某些写操作。

   更具体地说， IBM MQ for AIX or Linux 平台使用 *O_SYNC* 打开选项和 `fsync()` 系统调用来显式强制写入可恢复介质，并且写操作依赖于正确运行的这些选项。

   注意: Linux 您应该使用 `async` 选项安装文件系统，该选项仍然支持同步写入选项，并提供比 `sync` 选项更好的性能。

   但是，请注意，如果已从 Linux 导出文件系统，那么仍必须使用 `sync` 选项导出文件系统。

2. 保证对文件的独占访问权

   为了使多个队列管理器同步，需要有一种队列管理器获取对文件的互斥锁定的机制。

3. 发生故障时释放锁定

   如果队列管理器发生故障，或者如果与文件系统发生通信故障，那么队列管理器锁定的文件需要解锁并可供其他进程使用，而无需等待队列管理器重新连接到文件系统。

共享文件系统必须满足这些要求才能使 IBM MQ 可靠运行。 如果没有，那么在多实例队列管理器配置中使用共享文件系统时，队列管理器数据和日志会损坏。

对于 Microsoft Windows 上的多实例队列管理器，联网存储器必须由 Microsoft Windows 网络所使用的服务器消息块 (SMB) 协议访问。 服务器消息块 (SMB) 客户机不满足 IBM MQ 在 Microsoft Windows 以外的平台上锁定语义的要求，因此在 Microsoft Windows 以外的平台上运行的多实例队列管理器不得使用服务器消息块 (SMB) 作为其共享文件系统。

对于其他受支持平台上的多实例队列管理器，必须通过符合 Posix 的网络文件系统协议来访问存储器，并支持基于租赁的锁定。 网络文件系统 4 满足此要求。 不能将较旧的文件系统 (例如，网络文件系统版本 3) 与多实例队列管理器配合使用，这些系统没有在发生故障后释放锁定的可靠机制。

检查共享文件系统是否满足需求

您必须检查计划使用的共享文件系统是否满足这些需求。 您还必须检查是否正确配置了文件系统以实现可靠性。 共享文件系统有时会提供配置选项，以牺牲可靠性来提高性能。

有关更多信息，请参阅 测试 IBM MQ 多实例队列管理器文件系统的语句。

在正常情况下， IBM MQ 使用属性高速缓存正常运行，不需要禁用高速缓存，例如通过在 NFS 安装上设置 NOAC。 当多个文件系统客户机争用对文件系统服务器上同一文件的写访问权时，属性高速缓存可能会导致问题，因为每个客户机使用的高速缓存属性可能与服务器上的那些属性不同。 以此方式访问的文件的一个示例是多实例队列管理器的队列管理器错误日志。 队列管理器错误日志可能由活动队列管理器实例和备用队列管理器实例写入，并且高速缓存的文件属性可能导致在发生文件回滚之前，错误日志的大小大于预期。

要帮助检查文件系统，请运行任务 验证共享文件系统行为。 此任务检查共享文件系统是否满足需求 2 和 3。 您需要验证共享文件系统文档中的需求 1，或者通过尝试将数据记录到磁盘。

在写入磁盘时，磁盘故障会导致错误， IBM MQ 会将这些错误报告为 "首次故障数据捕获" 错误。 您可以针对操作系统运行文件系统检查程序，以检查共享文件系统是否存在任何磁盘故障。 例如:

- Linux AIX 在 AIX and Linux 上，文件系统检查程序称为 fsck。
- Windows 在 Windows 平台上，文件系统检查程序称为 CHKDSK 或 SCANDISK。

NFS 服务器安全性

注意:

- 不能对用于存放 IBM MQ 安装目录的安装点使用 **nosuid** 或 **noexec** 选项。 这是因为 IBM MQ 包含 setuid/setgid 可执行程序，并且不能阻止这些程序正常运行。

- 将队列管理器数据仅放在网络文件系统 (NFS) 服务器上时，可以将以下三个选项与 mount 命令配合使用，以确保系统安全，而不会对队列管理器的运行造成有害影响:

  诺埃克
  通过使用此选项，可以阻止二进制文件在 NFS 上运行，这将阻止远程用户在系统上运行不需要的代码。

  诺苏伊德
  通过使用此选项，可防止使用 set-user-identifier 和 set-group-identifier 位，这将阻止远程用户获取更高的特权。

  节点 v
  通过使用此选项，可以停止字符并阻止使用或定义特殊设备，这将阻止远程用户走出 chroot 监狱。

IBM i Linux AIX 在 *Multiplatforms* 版上验证共享文件系统行为

运行 **amqmfsck** 以检查 AIX， Linux 或 IBM i 上的共享文件系统是否满足存储多实例队列管理器的队列管理器数据的需求。 (Windows 配置的唯一要求是它将 SMB 3 用于共享存储器供应。)

开始之前

您需要一个具有联网存储器的服务器，以及另外两个已安装 IBM MQ 的连接到该服务器的服务器。 您必须具有管理员 (root) 权限才能配置文件系统，并且必须是 IBM MQ 管理员才能运行 **amqmfsck**。

关于此任务

第 93 页的『Multiplatforms 版上共享文件系统的需求』 描述了将共享文件系统与多实例队列管理器配合使用的文件系统需求。 IBM MQ 技术说明 针对 IBM MQ 多实例队列管理器文件系统的测试语句 列出了 IBM 已使用其进行测试的共享文件系统。 本任务中的过程描述了如何测试文件系统以帮助您评估未列出的文件系统是否保持数据完整性。

多实例队列管理器的故障转移可由硬件或软件故障触发，包括阻止队列管理器写入其数据或日志文件的网络问题。 主要是您有兴趣在文件服务器上引起故障。 但是，您还必须导致 IBM MQ 服务器失败，以测试是否成功释放了任何锁定。 要对共享文件系统充满信心，请测试以下所有故障以及特定于您的环境的任何其他故障:

1. 在文件服务器上关闭操作系统，包括同步磁盘。
2. 正在停止文件服务器上的操作系统而不同步磁盘。
3. 按每个服务器上的重置按钮。
4. 从每个服务器拔出网络电缆。
5. 从每个服务器拔出电源线。
6. 关闭每个服务器。

在要用于共享队列管理器数据和日志的网络存储器上创建目录。 目录所有者必须是 IBM MQ 管理员，换言之，是 AIX and Linux 上 mqm 组的成员。 运行测试的用户必须具有 IBM MQ 管理员权限。

使用 在 Linux 上创建多实例队列管理器 或 在 IBM i 上使用日志镜像和 NetServer 创建多实例队列管理器 中导出和安装文件系统的示例来帮助您配置文件系统。 不同的文件系统要求执行不同的配置步骤。 请阅读文件系统文档。

注: 与 **amqmfsck** 并行运行 IBM MQ MQI client 样本程序 **amqsfhac** 以演示队列管理器在故障期间维护消息完整性。

过程

在每个检查中，当文件系统检查程序正在运行时，将导致先前列表中的所有故障。 如果您打算与 **amqmfsck** 同时运行 **amqsfhac** ，请执行与此任务并行的任务 第 100 页的『运行 amqsfhac 以测试消息完整性』 。

1. 将导出的目录安装在两个 IBM MQ 服务器上。

   在文件系统服务器上，创建共享目录 shared 和子目录以保存多实例队列管理器 qmdata 的数据。 有关在 Linux 上为多实例队列管理器设置共享目录的示例，请参阅 在 Linux 上创建多实例队列管理器
2. 检查基本文件系统行为。

在一个 IBM MQ 服务器上，运行不带参数的文件系统检查程序。

在 IBM MQ 服务器 1 上:

```
amqmfsck /shared/qmdata
```

3. 检查是否同时从两个 IBM MQ 服务器写入同一目录。
   在两个 IBM MQ 服务器上，使用 -c 选项同时运行文件系统检查程序。

   在 IBM MQ 服务器 1 上:

   ```
   amqmfsck -c /shared/qmdata
   ```

   在 IBM MQ 服务器 2 上:

   ```
   amqmfsck -c /shared/qmdata
   ```

4. 检查是否正在等待和释放两个 IBM MQ 服务器上的锁定。
   在两个 IBM MQ 服务器上，使用 -w 选项同时运行文件系统检查程序。

   在 IBM MQ 服务器 1 上:

   ```
   amqmfsck -w /shared/qmdata
   ```

   在 IBM MQ 服务器 2 上:

   ```
   amqmfsck -w /shared/qmdata
   ```

5. 检查数据完整性。

   a) 格式化测试文件。

   在要测试的目录中创建大文件。 文件已格式化，以便后续阶段可以成功完成。 该文件必须足够大，以便有足够的时间来中断第二阶段以模拟故障转移。 尝试缺省值 262144 页 (1 GB)。 程序会自动减少慢速文件系统上的此缺省值，以便在大约 60 秒内完成格式化

   在 IBM MQ 服务器 1 上:

   ```
   amqmfsck -f /shared/qmdata
   ```

   服务器通过以下消息进行响应:

   ```
   Formatting test file for data integrity test.

   Test file formatted with 262144 pages of data.
   ```

   b) 在导致失败时，使用文件系统检查程序将数据写入测试文件。

   同时在两台服务器上运行测试程序。 在将要经历故障的服务器上启动测试程序，然后在将要经历故障的服务器上启动测试程序。 导致您正在调查的故障。

   第一个测试程序停止，并显示错误消息。 第二个测试程序获取对测试文件的锁定，并从第一个测试程序离开的位置开始将数据写入测试文件。 让第二个测试程序运行到完成。

表 *10:* 同时在两台服务器上运行数据完整性检查

IBM MQ 服务器 1	**IBM MQ 服务器 2**
`amqmfsck -a /shared/qmdata`	

表 *10:* 同时在两台服务器上运行数据完整性检查 *(继续)*

IBM MQ 服务器 1	**IBM MQ 服务器 2**
	`amqmfsck -a /shared/qmdata`

`Please start this program on a second machine`
`with the same parameters.`
 	
`File lock acquired.`
 	
`Start a second copy of this program with the same parameters on another server.`
 	
`Writing data into test file.`
 	
`To increase the effectiveness of the test, interrupt the writing by ending the process,`
`temporarily breaking the network connection`
`to the networked storage,`
`rebooting the server or turning off the power.`
	`Waiting for lock...`
	
	`Waiting for lock...`
	
	`Waiting for lock...`
	
	`Waiting for lock...`
	
	`Waiting for lock...`
	
	`Waiting for lock...`

Turn the power off here.	

	`File lock acquired.`
	
	`Reading test file`
	
	`Checking the integrity of the data read.`
	
	`Appending data into the test file after data already found.`
	
	`The test file is full of data.`
	`It is ready to be inspected for data integrity.`

测试的时间取决于文件系统的行为。 例如，在断电后，文件系统通常需要 30 到 90 秒来释放由第一个程序获取的文件锁定。 如果在第一个测试程序填充该文件之前，您没有太多时间来引入该故障，请使用 **amqmfsck** 的 -x 选项来删除该测试文件。 使用更大的测试文件从头开始尝试测试。

c) 验证测试文件中数据的完整性。

在 IBM MQ 服务器 2 上:

```
amqmfsck -i /shared/qmdata
```

服务器通过以下消息进行响应:

```
File lock acquired

Reading test file checking the integrity of the data read.

The data read was consistent.
```

```
The tests on the directory completed successfully.
```

6. 删除测试文件。

   在 IBM MQ 服务器 2 上:

```
amqmfsck -x /shared/qmdata

Test files deleted.
```

   服务器通过以下消息进行响应:

```
Test files deleted.
```

结果

如果测试成功完成，那么程序返回的退出代码为零，否则返回非零。

示例

第一组三个示例显示了生成最小输出的命令。

在一台服务器上成功测试基本文件锁定

```
> amqmfsck /shared/qmdata
The tests on the directory completed successfully.
```

在一个服务器上测试基本文件锁定失败

```
> amqmfsck /shared/qmdata
AMQ6245: Error Calling 'write()[2]' on file '/shared/qmdata/amqmfsck.lck' error '2'.
```

在两台服务器上成功进行锁定测试

表 *11:* 在两台服务器上成功锁定

IBM MQ 服务器 1	**IBM MQ 服务器 2**

`> amqmfsck -w /shared/qmdata`
`Please start this program on a second`
`machine with the same parameters.`
`Lock acquired.`
`Press Return`
`or terminate the program to release the lock.`	

	`> amqmfsck -w /shared/qmdata` `Waiting for lock...`
`[ Return pressed ]` `Lock released.`	

	`Lock acquired.`
	`The tests on the directory completed`
	`successfully`

第二组三个示例显示了使用详细方式的相同命令。

在一台服务器上成功测试基本文件锁定

```
> amqmfsck -v /shared/qmdata
System call: stat("/shared/qmdata")'
```

```
System call: fd = open("/shared/qmdata/amqmfsck.lck", O_RDWR, 0666)
System call: fchmod(fd, 0666)
System call: fstat(fd)
System call: fcntl(fd, F_SETLK, F_WRLCK)
System call: write(fd)
System call: close(fd)
System call: fd = open("/shared/qmdata/amqmfsck.lck", O_RDWR, 0666)
System call: fcntl(fd, F_SETLK, F_WRLCK)
System call: close(fd)
System call: fd1 = open("/shared/qmdata/amqmfsck.lck", O_RDWR, 0666)
System call: fcntl(fd1, F_SETLK, F_RDLCK)
System call: fd2 = open("/shared/qmdata/amqmfsck.lck", O_RDWR, 0666)
System call: fcntl(fd2, F_SETLK, F_RDLCK)
System call: close(fd2)
System call: write(fd1)
System call: close(fd1)
The tests on the directory completed successfully.
```

在一个服务器上测试基本文件锁定失败

```
> amqmfsck -v /shared/qmdata
System call: stat("/shared/qmdata")
System call: fd = open("/shared/qmdata/amqmfsck.lck", O_RDWR, 0666)
System call: fchmod(fd, 0666)
System call: fstat(fd)
System call: fcntl(fd, F_SETLK, F_WRLCK)
System call: write(fd)
System call: close(fd)
System call: fd = open("/shared/qmdata/amqmfsck.lck", O_RDWR, 0666)
System call: fcntl(fd, F_SETLK, F_WRLCK)
System call: close(fd)
System call: fd = open("/shared/qmdata/amqmfsck.lck", O_RDWR, 0666)
System call: fcntl(fd, F_SETLK, F_RDLCK)
System call: fdSameFile = open("/shared/qmdata/amqmfsck.lck", O_RDWR, 0666)
System call: fcntl(fdSameFile, F_SETLK, F_RDLCK)
System call: close(fdSameFile)
System call: write(fd)
AMQxxxx: Error calling 'write()[2]' on file '/shared/qmdata/amqmfsck.lck', errno 2
(Permission denied).
```

在两台服务器上成功进行锁定测试

表 *12:* 在两个服务器上成功锁定-详细方式

IBM MQ 服务器 1	**IBM MQ 服务器 2**

`> amqmfsck -wv /shared/qmdata`
`Calling 'stat("/shared/qmdata")'`
`Calling 'fd = open("/shared/qmdata/`
`amqmfsck.lkw",`
`O_EXCL | O_CREAT | O_RDWR, 0666)'`
`Calling 'fchmod(fd, 0666)'`
`Calling 'fstat(fd)'`
`Please start this program on a second`
`machine with the same parameters.`
`Calling 'fcntl(fd, F_SETLK, F_WRLCK)'`
`Lock acquired.`
`Press Return`
`or terminate the program to release the lock.`	

	`> amqmfsck -wv /shared/qmdata`
	`Calling 'stat("/shared/qmdata")'`
	`Calling 'fd = open("/shared/qmdata/`
	`amqmfsck.lkw",`
	`O_EXCL | O_CREAT | O_RDWR,0666)'`
	`Calling 'fd = open("/shared/qmdata/amqmfsck.lkw,`
	`O_RDWR, 0666)'`
	`Calling 'fcntl(fd, F_SETLK, F_WRLCK)`
	`'Waiting for lock...`

`[ Return pressed ]`
`Calling 'close(fd)'`
`Lock released.`	

表 *12:* 在两个服务器上成功锁定-详细方式 *(*继续*)*	
IBM MQ 服务器 1	**IBM MQ 服务器 2**

	`Calling 'fcntl(fd, F_SETLK, F_WRLCK)'`
	`Lock acquired.`
	`The tests on the directory completed successfully`

相关参考

"高可用性"样本程序

Multi

运行 *amqsfhac* 以测试消息完整性

与 **amqmfsck** 并行运行 IBM MQ MQI client 样本程序 **amqsfhac** 以演示队列管理器在故障期间维护消息完整性。

开始之前

此测试需要四个服务器。 两个服务器用于多实例队列管理器，一个用于文件系统，另一个用于将 **amqsfhac** 作为 IBM MQ MQI client 应用程序运行。

遵循 第 95 页的『在 Multiplatforms 版上验证共享文件系统行为』 中的步骤 第 95 页的『1』，为多实例队列管理器设置文件系统。

关于此任务

IBM MQ MQI client 样本程序 **amqsfhac** 检查使用联网存储器的队列管理器在发生故障后是否保持数据完整性。 与 **amqmfsck** 并行运行 **amqsfhac** 以演示队列管理器在故障期间维护消息完整性。

过程

1. 使用您在 过程中的步骤 第 95 页的『1』 中创建的文件系统，在另一服务器 QM1 上创建多实例队列管理器。

   请参阅 创建多实例队列管理器。
2. 在两个服务器上启动队列管理器，使其具有高可用性。

   在服务器 1 上:

   ```
   strmqm -x QM1
   ```

   在服务器 2 上:

   ```
   strmqm -x QM1
   ```
3. 设置客户机连接以运行 **amqsfhac**。

   a) 对于您的企业用于设置客户机连接的平台或平台，请使用 验证 *IBM MQ* 安装 中的过程，或者使用 可重新连接的客户机样本中的示例脚本。

   b) 修改客户机通道以具有两个 IP 地址，对应于运行 QM1 的两个服务器。

   在示例脚本中，修改:

   ```
   DEFINE CHANNEL(CHANNEL1) CHLTYPE(CLNTCONN) TRPTYPE(TCP) +
   CONNAME('LOCALHOST(2345)') QMNAME(QM1) REPLACE
   ```

   收件人:

```
DEFINE CHANNEL(CHANNEL1) CHLTYPE(CLNTCONN) TRPTYPE(TCP) +
CONNAME('server1(2345),server2(2345)') QMNAME(QM1) REPLACE
```

其中 server1 和 server2 是两个服务器的主机名， 2345 是通道侦听器正在侦听的端口。 通常，此值缺省为 1414。 您可以将 1414 与缺省侦听器配置配合使用。

4. 在 QM1 上为测试创建两个本地队列。
   运行以下 MQSC 脚本:

```
DEFINE QLOCAL(TARGETQ) REPLACE
DEFINE QLOCAL(SIDEQ) REPLACE
```

5. 使用 **amqsfhac** 测试配置

```
amqsfhac QM1 TARGETQ SIDEQ 2 2 2
```

6. 测试文件系统完整性时测试消息完整性。

   在 第 95 页的『在 Multiplatforms 版上验证共享文件系统行为』的步骤 第 96 页的『5』 期间运行 **amqsfhac** 。

```
amqsfhac QM1 TARGETQ SIDEQ 10 20 0
```

如果停止活动队列管理器实例，那么 **amqsfhac** 将在另一个队列管理器实例处于活动状态后重新连接到该实例。 再次重新启动已停止的队列管理器实例，以便您可以在下一次测试中反转故障。 您可能需要根据对环境的试验增加迭代次数，以便测试程序运行足够的时间进行故障转移。

结果

以下示例显示了在步骤 第 101 页的『6』 中运行 **amqsfhac** 的示例。 在此示例中，测试成功。

```
Sample AMQSFHAC start
qmname = QM1
qname = TARGETQ
sidename = SIDEQ
transize = 10
iterations = 20
verbose = 0
Iteration 0
Iteration 1
Iteration 2
Iteration 3
Iteration 4
Iteration 5
Iteration 6
Resolving MQRC_CALL_INTERRUPTED
MQGET browse side tranid=14 pSideinfo->tranid=14
Resolving to committed
Iteration 7
Iteration 8
Iteration 9
Iteration 10
Iteration 11
Iteration 12
Iteration 13
Iteration 14
Iteration 15
Iteration 16
Iteration 17
Iteration 18
Iteration 19
Sample AMQSFHAC end
```

如果测试检测到问题，那么输出将报告故障。 在某些测试中， MQRC_CALL_INTERRUPTED 可能会报告 “Resolving to backed out”。 它与结果没有区别。 结果取决于是在发生故障之前还是之后由联网文件存储器落实对磁盘的写入。

相关参考

amqmfsck（文件系统检查）

“高可用性”样本程序

Multi 在 Multiplatforms 版上共享 IBM MQ 文件

某些 IBM MQ 文件由活动队列管理器独占访问，其他文件共享。

IBM MQ 文件拆分为程序文件和数据文件。 程序文件通常以本地方式安装在运行 IBM MQ 的每个服务器上。 队列管理器共享对缺省数据目录中的数据文件和目录的访问权。 他们需要对包含在 第 102 页的图 32 中显示的每个 qmgrs 和 log 目录中的自己的队列管理器目录树进行独占访问。

第 102 页的图 32 是 IBM MQ 目录结构的高级视图。 它显示了可以在队列管理器之间共享并使其成为远程目录的目录。 详细信息因平台而异。 虚线指示可配置路径。

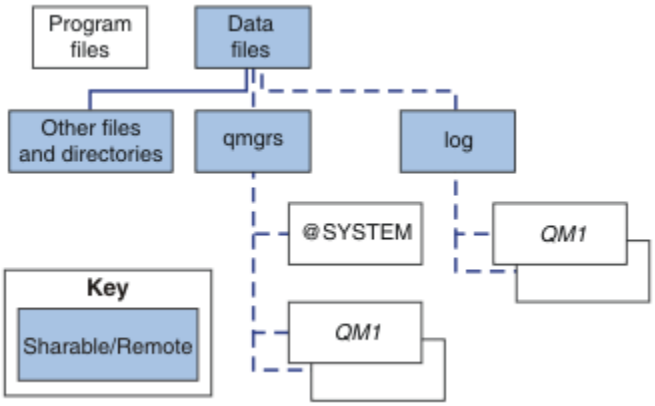


图 *32: IBM MQ* 目录结构的整体视图

程序文件

程序文件目录通常保留在缺省位置，是本地的，并且由服务器上的所有队列管理器共享。

数据文件

数据文件目录通常在缺省位置 /var/mqm (在 AIX and Linux 系统上) 本地，并可在 Windows 上安装时进行配置。 它在队列管理器之间共享。 您可以使缺省位置成为远程位置，但不要在 IBM MQ 的不同安装之间共享该位置。 IBM MQ 配置中的 DefaultPrefix 属性指向此路径。

qmgrs

有两种替代方法可指定队列管理器数据的位置。

使用 Prefix 属性

Prefix 属性指定 qmgrs 目录的位置。 IBM MQ 根据队列管理器名称构造队列管理器目录名称，并将其创建为 qmgrs 目录的子目录。

Prefix 属性位于 mqs.ini 文件的 QueueManager 节 中，并且继承自 所有队列管理器节的 **DefaultPrefix** 属性中的值。 缺省情况下，为了简化管理，队列管理器通常共享同一 qmgrs 目录。

如果更改任何队列管理器的 qmgrs 目录的位置，那么必须更改其 **Prefix** 属性的值。

AIX and Linux 平台的 第 102 页的图 32 中 QM1 目录的 **Prefix** 属性如下所示:

```
Prefix=/var/mqm
```

使用 DataPath 属性

DataPath 属性指定队列管理器数据目录的位置。

DataPath 属性指定完整路径，包括队列管理器数据目录的名称。 **DataPath** 属性与 **Prefix** 属性不同，后者指定队列管理器数据目录的不完整路径。

DataPath 属性 (如果已指定) 位于 `mqs.ini` 文件的 QueueManager 节 节中。 如果已指定此属性，那么它优先于 **Prefix** 属性中的任何值。

如果更改任何队列管理器的队列管理器数据目录的位置，那么必须更改 `DataPath` 属性的值。

对于 Linux 或 AIX 平台， 第 102 页的图 32 中 `QM1` 目录的 `DataPath` 属性如下所示:

```
DataPath=/var/mqm/qmgrs/QM1
```

log

针对队列管理器配置中的 日志节 中的每个队列管理器单独指定日志目录。 队列管理器配置在 `qm.ini` 中。

***DataPath*/*QmgrName*/@IPCC 子目录**

DataPath/*QmgrName*/@IPCC 子目录位于共享目录路径中。 它们用于构造 IPC 文件系统对象的目录路径。 当在系统之间共享队列管理器时，它们需要区分队列管理器的名称空间。

IPC 文件系统对象必须由系统进行区分。 对于运行队列管理器的每个系统，会将一个子目录添加到目录路径中，请参阅 第 103 页的图 33。

```
DataPath/QmgrName/@IPCC/esem/myHostName/
```

图 *33:* 示例 *IPC* 子目录

myHostName 最多包含操作系统返回的主机名的前 20 个字符。 在某些系统上，在截断之前，主机名的长度可能最多为 64 个字符。 生成的 *myHostName* 值可能由于以下两个原因导致问题:

1. 前 20 个字符不唯一。
2. 主机名由 DHCP 算法生成，该算法并不总是将相同的主机名分配给系统。

在这些情况下，请使用环境变量 **MQS_IPC_HOST** 设置 *myHostName* ; 请参阅 第 103 页的图 34。

```
export MQS_IPC_HOST= myHostName
```

图 *34:* 示例: 设置 ***MQS_IPC_HOST***

其他文件和目录

其他文件和目录 (例如包含跟踪文件的目录和公共错误日志) 通常共享并保留在本地文件系统上。

通过支持共享文件系统， IBM MQ 使用文件系统锁定来管理对这些文件的互斥访问。 文件系统锁定一次仅允许特定队列管理器的一个实例处于活动状态。

当您启动特定队列管理器的第一个实例时，它将获取其队列管理器目录的所有权。 如果启动第二个实例，那么仅当第一个实例已停止时，才能获取所有权。 如果第一个队列管理器仍在运行，那么第二个实例无法启动，并报告队列管理器正在其他位置运行。 如果第一个队列管理器已停止，那么第二个队列管理器将接管队列管理器文件的所有权并成为正在运行的队列管理器。

您可以自动执行第二个队列管理器从第一个队列管理器接管的过程。 使用 `strmqm -x` 选项启动第一个队列管理器，该选项允许另一个队列管理器从该队列管理器接管。 然后，第二个队列管理器将等待队列管理器文件解锁，然后再尝试接管队列管理器文件的所有权，然后启动。

Linux AIX AIX and Linux 系统上的目录结构

AIX and Linux 系统上的 IBM MQ 目录结构可以映射到不同的文件系统，以便于管理，提高性能和提高可靠性。

使用 IBM MQ 的灵活目录结构以利用共享文件系统来运行多实例队列管理器。

使用命令 **crtmqm** *QM1* 来创建 第 104 页的图 35 中显示的目录结构，其中 R 是产品的发行版。 它是在 IBM MQ 系统上创建的队列管理器的典型目录结构。 为了清晰起见，省略了某些目录，文件和 .ini 属性设置，另一个队列管理器名称可能会被修改。 文件系统的名称在不同的系统上会有所不同。

在典型安装中，您创建的每个队列管理器都指向本地文件系统上的公共 log 和 qmgrs 目录。 在多实例配置中， log 和 qmgrs 目录位于与另一个 IBM MQ 安装共享的网络文件系统上。

第 104 页的图 35 显示了 AIX 上 IBM MQ v7.R 的缺省配置，其中 R 是产品的发行版。 有关备用多实例配置的示例，请参阅第 108 页的『AIX and Linux 系统上的示例目录配置』。

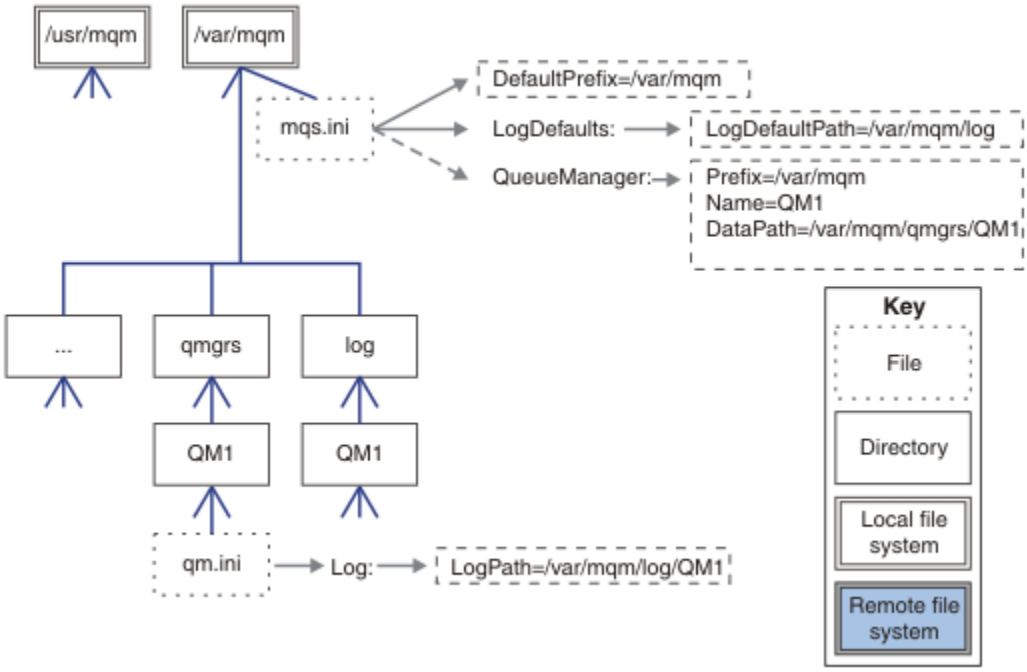


图 *35: AIX and Linux* 系统的缺省 *IBM MQ* 目录结构示例

缺省情况下，该产品安装到 AIX 上的 /usr/mqm 和其他系统上的 /opt/mqm 中。 工作目录将安装到 /var/mqm 目录中。

注: 如果在安装 IBM MQ 之前创建了 /var/mqm 文件系统，请确保 mqm 用户具有完整目录许可权，例如，文件方式 755。

注: /var/mqm/errors 目录应该是单独的文件系统，以防止队列管理器生成的 FFDC 填充包含 /var/mqm 的文件系统。

请参阅 在 AIX and Linux 系统上创建文件系统 以获取更多信息。

log 和 qmgrs 目录显示在其缺省位置中，如 mqs.ini 文件中的 LogDefault **路径** 和 DefaultPrefix 属性的缺省值所定义。 创建队列管理器时，缺省情况下将在 *DefaultPrefix*/qmgrs 中创建队列管理器数据目录，并在 *LogDefaultPath*/log 中创建日志文件目录。 LogDefault **路径** 和 DefaultPrefix 仅影响缺省情况下创建队列管理器和日志文件的情况。 队列管理器目录的实际位置保存在 mqs.ini 文件中，日志文件目录的位置保存在 qm.ini 文件中。

队列管理器的日志文件目录在 LogPath 属性的 qm.ini 文件中定义。 在 **crtmqm** 命令上使用 -ld 选项来设置队列管理器的 LogPath 属性; 例如， **crtmqm** -ld *LogPath* QM1 。 如果省略 ld 参数，那么将改为使用 LogDefaultPath 的值。

队列管理器数据目录在 mqs.ini 文件的 QueueManager 节中的 DataPath 属性中定义。 在 **crtmqm** 命令上使用 -md 选项为队列管理器设置 DataPath ; 例如， **crtmqm** - md *DataPath* QM1 。 如果省略 md 参数，那么将改为使用 DefaultPrefix 或 Prefix 属性的值。 **前缀** 优先于 DefaultPrefix。

通常，在单个命令中同时指定日志和数据目录来创建 QM1 。

```
crtmqm
```

```
-md DataPath -ld
LogPath QM1
```

当队列管理器停止时，您可以通过编辑 qm.ini 文件中的 DataPath 和 LogPath 属性来修改现有队列管理器的队列管理器日志和数据目录的位置。

errors 目录的路径 (与 /var/mqm 中所有其他目录的路径一样) 不可修改。 但是，这些目录可以安装在不同的文件系统上，也可以象征性地链接到不同的目录。

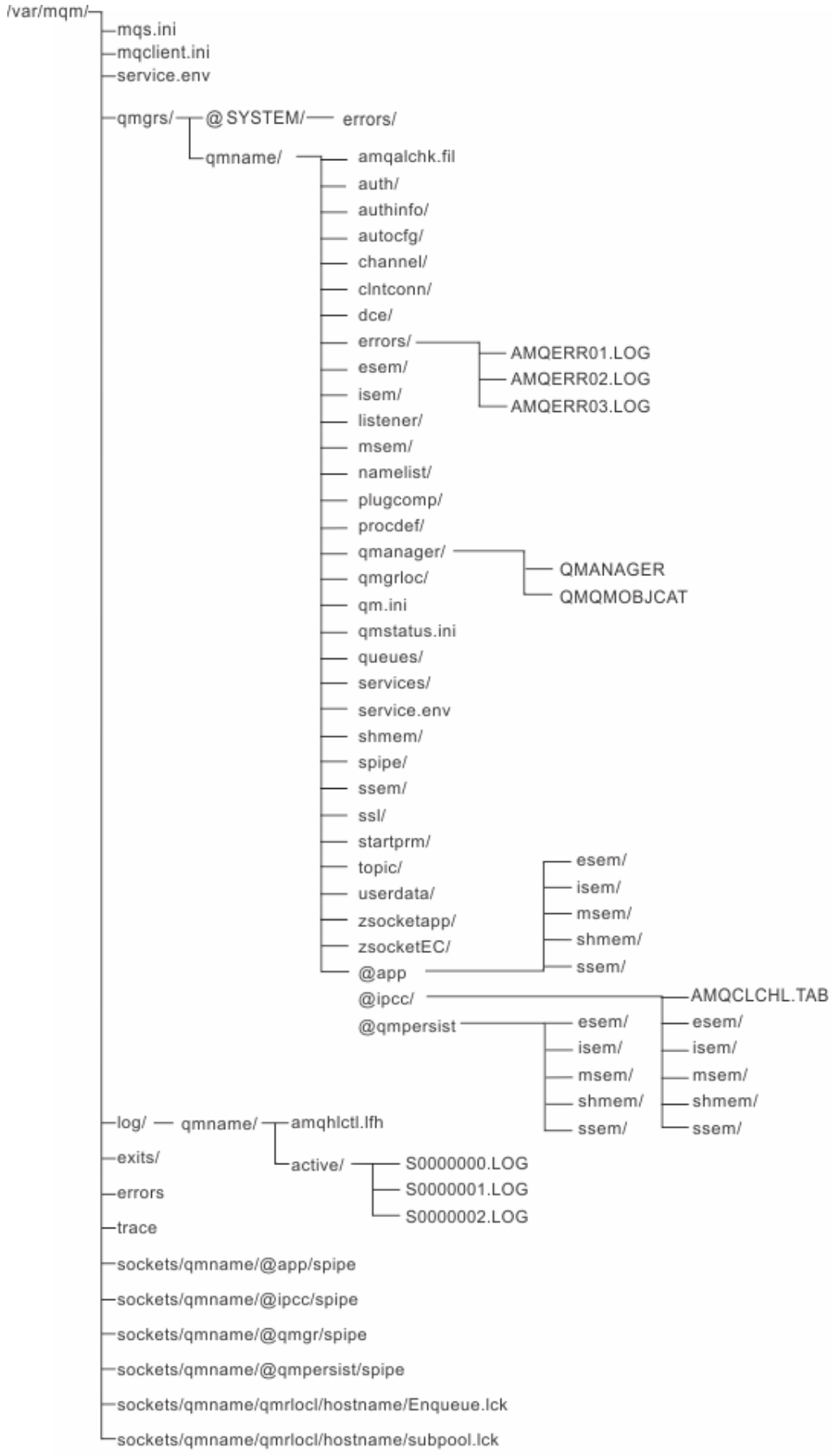
Linux AIX *AIX and Linux* 系统上的目录内容

与队列管理器关联的目录的内容。

有关产品文件位置的信息，请参阅 选择安装位置

有关备用目录配置的信息，请参阅 第 93 页的『多平台上的规划文件系统支持』。

在队列管理器已使用一段时间之后，以下目录结构代表 IBM MQ 。 您拥有的实际结构取决于在队列管理器上执行的操作。


/var/mqm/
mqs.ini
mqclient.ini
service.env
qmgrs/
@SYSTEM/
errors/
qmname/
amqalchk.fil
auth/
authinfo/
autocfg/
channel/
clntconn/
dce/
errors/
AMQERR01.LOG
AMQERR02.LOG
AMQERR03.LOG
esem/
isem/
listener/
msem/
namelist/
plugcomp/
procdef/
qmanager/
QMANAGER
QMQMOBJCAT
qmgrloc/
qm.ini
qmstatus.ini
queues/
services/
service.env
shmem/
spipe/
ssem/
ssl/
startprm/
topic/
userdata/
zsocketapp/
zsocketEC/
@app
esem/
isem/
msem/
shmem/
ssem/
@ipcc/
AMQCLCHL.TAB
esem/
isem/
msem/
shmem/
ssem/
@qmpersist
esem/
isem/
msem/
shmem/
ssem/
log/
qmname/
amqhlctl.lfh
active/
S0000000.LOG
S0000001.LOG
S0000002.LOG
exits/
errors
trace
sockets/qmname/@app/spipe
sockets/qmname/@ipcc/spipe
sockets/qmname/@qmgr/spipe
sockets/qmname/@qmpersist/spipe
sockets/qmname/qmrlocl/hostname/Enqueue.lck
sockets/qmname/qmrlocl/hostname/subpool.lck

/var/mqm/

/var/mqm 目录包含适用于整个 IBM MQ 安装而不是个别队列管理器的配置文件和输出目录。

表 *13: AIX and Linux* 上 */var/mqm* 目录的已记录内容	
目录或文件名	**内容**
mqs.ini	IBM MQ 安装范围配置文件，在队列管理器启动时读取。 可使用 **AMQ_MQS_INI_LOCATION** 环境变量修改文件路径。 确保在运行 **strmqm** 命令的 shell 中设置并导出此命令。
mqclient.ini	IBM MQ MQI client 程序读取的缺省客户机配置文件。 可使用 **MQCLNTCF** 环境变量修改文件路径。
service.env	包含服务进程的机器作用域环境变量。 文件路径已固定。
个错误/	机器作用域错误日志和 FFST 文件。 目录路径已修正。 另请参阅 FFST: IBM MQ for UNIX 和 Linux 系统。
套接字/	包含每个队列管理器的信息，仅供系统使用。
跟踪/	跟踪文件。 目录路径已修正。
Web/	mqweb 服务器目录。
个出口/	包含用户通道出口程序的缺省目录。 可在 mqs.ini 文件的 ApiExit 节中修改位置。
exits64/	

/var/mqm/qmgrs/*qmname*/

/var/mqm/qmgrs/*qmname*/ 包含队列管理器的目录和文件。 该目录已锁定，以供活动队列管理器实例独占访问。 可以在 mqs.ini 文件中直接修改目录路径，也可以使用 **crtmqm** 命令的 **md** 选项进行修改。

表 *14: AIX and Linux* 上 */var/mqm/qmgrs/qmname* 目录的已记录内容	
目录或文件名	**内容**
qm.ini	队列管理器配置文件，在队列管理器启动时读取。
个错误/	队列管理器作用域错误日志。 *qmname* = @system 包含未知或不可用队列管理器的通道相关消息。
@ipcc/ AMQCLCHL.TAB	缺省客户机通道控制表，由 IBM MQ 服务器创建，并由 IBM MQ MQI client 程序读取。 可使用 **MQCHLLIB** 和 **MQCHLTAB** 环境变量修改文件路径。
QMANAGER	队列管理器对象文件: QMANAGER 队列管理器对象目录: QMQMOBJCAT

表 14: AIX and Linux 上 /var/mqm/qmgrs/qmname 目录的已记录内容 (继续)	
目录或文件名	**内容**
authinfo/	队列管理器中定义的每个对象都与这些目录中的一个文件相关联。 文件名与定义名称大致匹配; 请参阅 了解 IBM MQ 文件名。
通道/	
clntconn/	
侦听器/	
名称列表/	
进程/	
队列/	
服务/	
主题/	
...	IBM MQ 使用的其他目录 (例如 @ipcc) 将仅由 IBM MQ 修改。
用户数据/	可用于存储应用程序的持久状态 (当将队列管理器移动到不同节点时， RDQM 可以使用此状态-请参阅 存储持久应用程序状态。)
***DataPath*\autocfg**	用于自动配置

/var/mqm/log/*qmname*/

/var/mqm/log/*qmname*/ 包含队列管理器日志文件。 该目录已锁定，以供活动队列管理器实例独占访问。 可以在 qm.ini 文件中修改目录路径，也可以使用 **crtmqm** 命令的 **ld** 选项进行修改。

表 15: AIX and Linux 上 /var/mqm/log/qmname 目录的已记录内容	
目录或文件名	**内容**
amqhlctl.lfh	日志控制文件。
活动/	此目录包含编号为 S0000000.LOG， S0000001.LOG， S0000002.LOG，依此类推。

/opt/mqm

缺省情况下， /opt/mqm 是大多数平台上的安装目录。 请参阅 第 91 页的『Multiplatforms 版上的磁盘空间需求』，以获取有关企业使用的一个或多个平台上的安装目录所需的空间量的更多信息。

Linux AIX *AIX and Linux* 系统上的示例目录配置

AIX and Linux 系统上备用文件系统配置的示例。

您可以通过各种方式定制 IBM MQ 目录结构，以实现许多不同的目标。

- 将 qmgrs 和 log 目录放在远程共享文件系统上以配置多实例队列管理器。
- 将单独的文件系统用于数据和日志目录，并将这些目录分配给不同的磁盘，以通过减少 I/O 争用来提高性能。
- 将更快的存储设备用于对性能影响更大的目录。 物理设备等待时间通常是持久消息传递性能中比本地还是远程安装设备更重要的因素。 以下列表显示哪些目录最敏感和最不敏感的性能。

  1. log
  2. qmgrs
  3. 其他目录，包括 /usr/mqm

- 在分配给具有良好弹性的存储器 (例如冗余磁盘阵列) 的文件系统上创建 `qmgrs` 和 `log` 目录。
- 最好将公共错误日志存储在 `var/mqm/errors` 本地，而不是存储在网络文件系统上，以便可以记录与网络文件系统相关的错误。

第 109 页的图 36 是从中派生备用 IBM MQ 目录结构的模板。 在模板中，虚线表示可配置的路径。 在示例中，虚线将替换为与 AMQ_MQS_INI_LOCATION 环境变量以及 `mqs.ini` 和 `qm.ini` 文件中存储的配置信息相对应的实线。

注: 路径信息显示在 `mqs.ini` 或 `qm.ini` 文件中。 如果在 **crtmqm** 命令中提供路径参数，请省略队列管理器目录的名称: 队列管理器名称由 IBM MQ 添加到路径中。

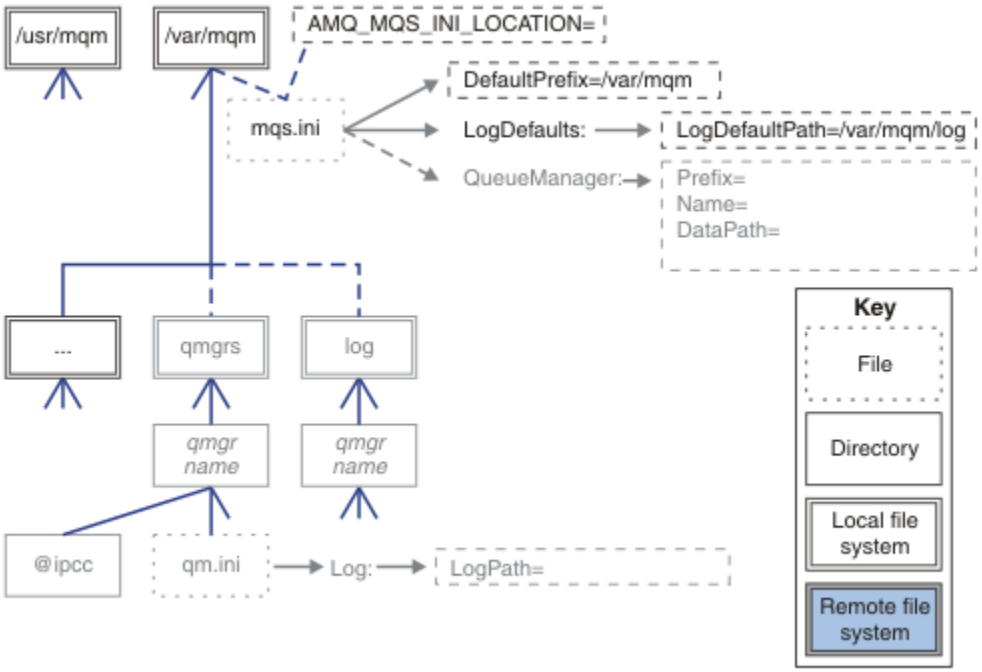


图 *36:* 目录结构模式模板

IBM MQ 的典型目录结构

第 110 页的图 37 是通过发出命令 **crtmqm** *QM1* 在 IBM MQ 中创建的缺省目录结构。

`mqs.ini` 文件具有 QM1 队列管理器的节，此节是通过引用 `DefaultPrefix` 的值创建的。 `qm.ini` 文件中的 `Log` 节具有 `LogPath` 的值，通过引用 `mqs.ini` 中的 `LogDefaultPath` 进行设置。

使用可选 **crtmqm** 参数可覆盖 `DataPath` 和 `LogPath` 的缺省值。

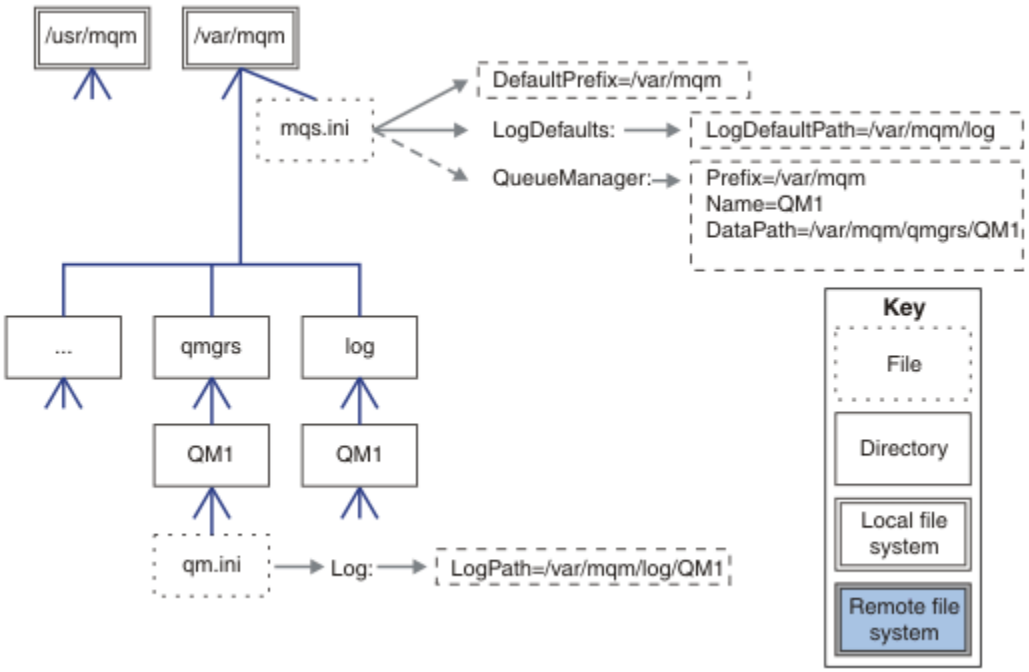


图 *37: AIX and Linux* 系统的缺省 *IBM MQ* 目录结构示例

共享缺省 qmgrs 和 log 目录

第 111 页的『共享所有内容』的替代方法是单独共享 qmgrs 和 log 目录 (第 110 页的图 38)。 在此配置中，无需设置 AMQ_MQS_INI_LOCATION ，因为缺省 mqs.ini 存储在本地 /var/mqm 文件系统中。 文件和目录 (例如 mqclient.ini 和 mqserver.ini) 也不会共享。

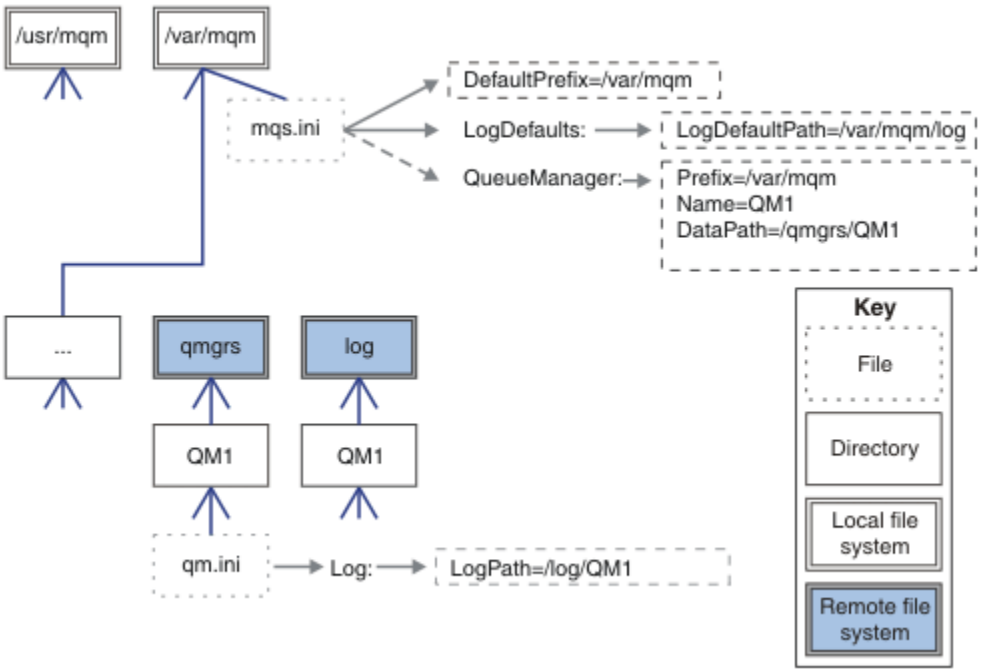


图 *38:* 共享 *qmgrs* 和 *log* 目录

共享名为 qmgrs 和 log 的目录

第 111 页的图 39 中的配置将 log 和 qmgrs 放置在名为 /ha 的公共命名远程共享文件系统中。 可以通过两种不同的方式创建相同的物理配置。

1. 设置 LogDefaultPath=/ha，然后运行命令 **crtmqm** - md */ha/qmgrs* QM1。 结果与 第 111 页的图 39 中的说明完全相同。
2. 保留缺省路径不变，然后运行命令 **crtmqm** - ld */ha/log* - md */ha/qmgrs* QM1。

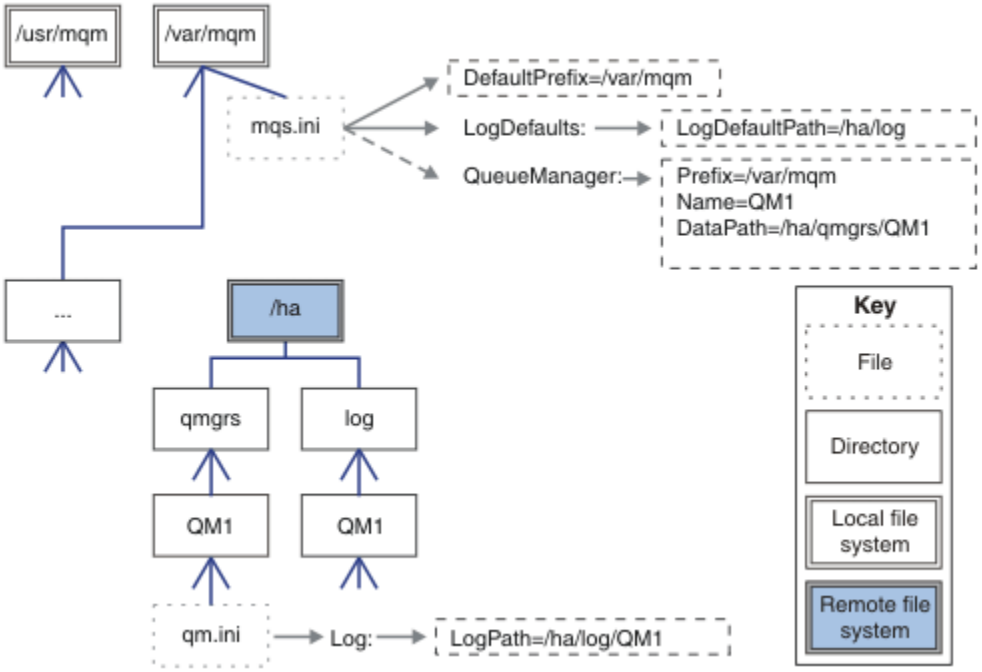


图 *39:* 共享名为 *qmgrs* 和 *log* 的目录

共享所有内容

第 112 页的图 40 是具有快速联网文件存储器的系统的简单配置。

将 /var/mqm 安装为远程共享文件系统。 缺省情况下，当您启动 QM1 时，它将查找 /var/mqm，在共享文件系统上查找并读取 /var/mqm 中的 mqs.ini 文件。 您可以将每个服务器上的 AMQ_MQS_INI_LOCATION 环境变量设置为指向不同的 mqs.ini 文件，而不是将单个 /var/mqm/mqs.ini 文件用于所有服务器上的队列管理器。

注: /var/mqm/errors/ 中通用错误文件的内容在不同服务器上的队列管理器之间共享。

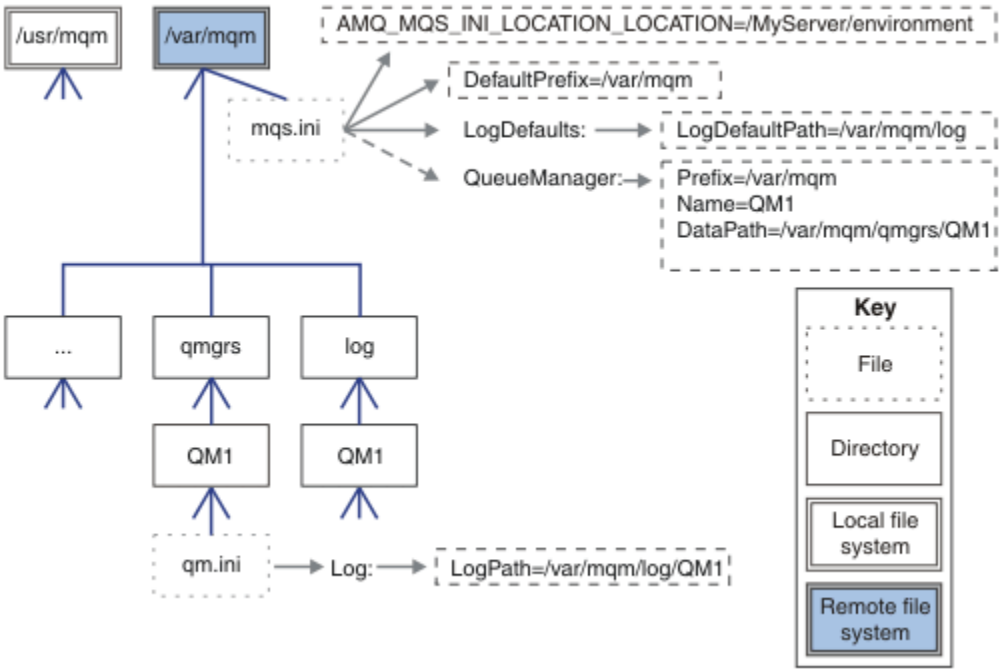


图 *40:* 共享所有内容

请注意，不能将此用于多实例队列管理器。 原因是多实例队列管理器中的每个主机都必须具有其自己的 `/var/mqm` 本地副本，以跟踪本地数据 (例如信号量和共享内存)。 无法在主机之间共享这些实体。

Windows Windows 系统上的目录结构

如何在 Windows 上查找队列管理器配置信息和目录。

IBM MQ for Windows 安装的缺省目录为:

程序目录
`C:\Program Files\IBM\MQ`

数据目录
`C:\ProgramData\IBM\MQ`

要点: Windows 对于 Windows 安装，会按照上述内容使用目录，除非存在先前的产品安装，该安装仍包含注册表项和/或队列管理器。 在此情况下，新安装将使用原有的数据目录位置。 有关更多信息，请参阅程序和数据目录位置。

如果要知道使用的是哪个安装目录和哪个数据目录，请运行 dspmqver 命令。

安装目录列示在 **InstPath** 字段中，数据目录列示在 **DataPath** 字段中。

例如，运行 **dspmqver** 命令将显示以下信息:

```
>dspmqver
Name:        IBM MQ
Version:     9.0.0.0
Level:       p900-L160512.4
BuildType:   IKAP - (Production)
Platform:    IBM MQ for Windows (x64 platform)
Mode:        64-bit
O/S:         Windows 7 Professional x64 Edition, Build 7601: SP1
InstName:    Installation1
InstDesc:
Primary:     Yes
InstPath:    C:\Program Files\IBM\MQ
DataPath:    C:\ProgramData\IBM\MQ
MaxCmdLevel: 900
LicenseType: Production
```

多实例队列管理器

要配置多实例队列管理器，必须将日志和数据目录放在联网存储器上，最好是放在与正在运行队列管理器实例的任何服务器不同的服务器上。

在 **crtmqm** 命令 **-md** 和 **-ld** 上提供了两个参数，以便更轻松地指定队列管理器数据和日志目录的位置。 指定 **-md** 参数的效果是四重:

1. mqs.ini 节 QueueManager*QmgrName* 包含指向队列管理器数据目录的新变量 *DataPath*。 与 *Prefix* 变量不同，路径包含队列管理器目录的名称。
2. 存储在 mqs.ini 文件中的队列管理器配置信息将缩减为 *Name*， *Prefix*， *Directory* 和 *DataPath*。

Windows 目录内容

列出 IBM MQ 目录的位置和内容。

IBM MQ 配置具有三组主要文件和目录:

1. 可执行文件以及仅在应用维护时更新的其他只读文件。 例如:
   - 自述文件
   - IBM MQ Explorer 插件和帮助文件
   - 许可证文件

   第 113 页的表 16 中描述了这些文件。
2. 可能可修改的文件和目录并非特定于特定队列管理器。 第 114 页的表 17 中描述了这些文件和目录。
3. 特定于服务器上每个队列管理器的文件和目录。 第 114 页的表 18 中描述了这些文件和目录。

资源目录和文件

资源目录和文件包含用于运行队列管理器的所有可执行代码和资源。 特定于安装的 IBM MQ 配置注册表键中的变量 *FilePath* 包含资源目录的路径。

表 *16: FilePath* 目录中的目录和文件

文件路径	**内容**
FilePath\bin	命令和 DLL
FilePath\bin64	命令和 DLL (64 位)
FilePath\conv	数据转换表
FilePath\doc	向导帮助文件
FilePath\MQExplorer	Explorer 和 Explorer 帮助 Eclipse 插件
FilePath\gskit8	Global Security Kit
FilePath\java	Java 资源，包括 JRE
FilePath\licenses	许可证信息
FilePath\Non_IBM_License	许可证信息
FilePath\properties	内部使用
FilePath\Tivoli	
FilePath\tools	开发资源和样本
FilePath\web	在 IBM MQ Console 和 REST API 不可编辑文件的安装组件文件结构中进行了描述。
FilePath\Uninst	内部使用
FilePath\README.TXT	自述文件

不特定于队列管理器的目录

某些目录包含并非特定于特定队列管理器的文件，例如跟踪文件和错误日志。 *DefaultPrefix* 变量包含这些目录的路径。 *DefaultPrefix* 是 AllQueueManagers 节的一部分。

表 *17: DefaultPrefix* 目录中的目录和文件	
文件路径	**内容**
DefaultPrefix\config	内部使用
DefaultPrefix\conv	ccsid_part2.tbl 和 ccsid.tbl data 转换控制文件，如数据转换 中所述
DefaultPrefix\errors	非队列管理器错误日志， AMQERR *nn*.LOG
DefaultPrefix\exits	通道出口程序
DefaultPrefix\exits64	通道出口程序 (64 位)
DefaultPrefix\ipc	未使用
DefaultPrefix\qmgrs	描述位置 第 114 页的表 18
DefaultPrefix\trace	跟踪文件
DefaultPrefix\web	在用户可编辑文件的 IBM MQ Console 和 REST API 安装组件文件结构中描述
DefaultPrefix\amqmjpse.txt	内部使用

队列管理器目录

创建队列管理器时，将创建一组特定于队列管理器的新目录。

如果使用 **-md** *filepath* 参数创建队列管理器，那么路径将存储在 mqs.ini 文件的队列管理器节中的 *DataPath* 变量中。 如果在不设置 **-md** *filepath* 参数的情况下创建队列管理器，那么将在存储在 *DefaultPrefix* 中的路径中创建队列管理器目录，并将该路径复制到 mqs.ini 文件的队列管理器节中的 *Prefix* 变量中。

表 *18: DataPath* 和 *Prefix**qmgrs**QmgrName* 目录中的目录和文件	
文件路径	**内容**
DataPath\@ipcc	AMQCLCHL.TAB(客户机连接表) 的缺省位置。
DataPath\authinfo	内部使用
DataPath\channel	
DataPath\clntconn	
DataPath\errors	错误日志， AMQERR *nn*.LOG

表 18: *DataPath* 和 *Prefix*\qmgrs*QmgrName* 目录中的目录和文件 (继续)	
文件路径	**内容**
DataPath\listener	内部使用
DataPath\namelist	
DataPath\plugcomp	
DataPath\procdef	
DataPath\qmanager	
DataPath\queues	
DataPath\services	
DataPath\ssl	
DataPath\startprm	
DataPath\topic	
DataPath\active	
DataPath\active.dat	
DataPath\amqalchk.fil	
DataPath\master	
DataPath\master.dat	
DataPath\qm.ini	队列管理器配置
DataPath\qmstatus.ini	队列管理器状态
DataPath\userdata	可用于存储应用程序的持久状态。
Prefix\qmgrs*QmgrName*	内部使用
Prefix\qmgrs\@SYSTEM	未使用
Prefix\qmgrs\@SYSTEM\errors	
*DataPath**autocfg*	用于自动配置

IBM i IBM i 上的目录结构

提供了 IFS 的描述，并且针对服务器，客户机和 Java 描述了 IBM MQ IFS 目录结构。

集成文件系统 (IFS) 是 IBM i 的一部分，支持类似于个人计算机 AIX and Linux 操作系统的流输入/输出和存储管理，同时针对存储在服务器中的所有信息提供集成结构。

在 IBM i 上，目录名称以字符 & （ampersand）(而不是字符 @ （at）) 开头。 例如， IBM i 上的 @system 是 &system。

IBM MQ 服务器的 IFS 根文件系统

安装 IBM MQ Server for IBM i 时，将在 IFS 根文件系统中创建以下目录。

ProdData:

概述

```
QIBM
   '-- ProdData
      '-- mqm
```

```
        '-- doc
        '-- inc
        '-- lib
        '-- samp
        '-- licenses
        '-- LicenseDoc
        '-- 5724H72_V8R0M0
```

/QIBM/ProdData/mqm
下面的子目录包含所有产品数据，例如 C++ 类，跟踪格式文件和许可证文件。 每次安装产品时，都会删除并替换此目录中的数据。

/QIBM/ProdData/mqm/doc
CL 命令的 "命令参考" 以 HTML 格式提供，并安装在此处。

/QIBM/ProdData/mqm/inc
用于编译 C 或 C++ 程序的头文件。

/QIBM/ProdData/mqm/lib
MQ 使用的辅助文件。

/QIBM/ProdData/mqm/samp
更多样本。

/QIBM/ProdData/mqm/License
许可证文件。 每种语言的两个文件命名为类似于 LA_ *xx* 和 LI_ *xx* ，其中 *xx* 是所提供的每种语言的 2 字符语言标识。

另外，以下目录存储许可协议文件:

/QIBM/ProdData/LicenseDoc/5724H72_V8R0M0
许可证文件。 这些文件的名称类似于 5724H72_V8R0M0_ *xx* ，其中 *xx* 是所提供的每种语言的 2 或 5 字符语言标识。

UserData:

概述

```
   QIBM
      '-- UserData
```

```
        '-- mqm
        '-- errors
        '-- trace
        '-- qmgrs
        '-- &system
        '-- qmgrname1
        '-- qmgrname2
        '-- and so on
```

/QIBM/UserData/mqm
此目录下的子目录包含与队列管理器相关的所有用户数据。

安装产品时，将在目录 /QIBM/UserData/mqm/ 中创建 mqs.ini 文件 (除非先前安装已有此文件)。

创建队列管理器时，将在目录 /QIBM/UserData/mqm/qmgrs/ *QMGRNAME* /(其中 *QMGRNAME* 是队列管理器的名称) 中创建 qm.ini 文件。

删除产品时，将保留目录中的数据。

IBM MQ MQI client 的 IFS 根文件系统

安装 IBM MQ MQI client for IBM i 时，将在 IFS 根文件系统中创建以下目录:

ProdData:

概述

```
QIBM
   '-- ProdData
      '-- mqm
      '-- lib
```

/QIBM/ProdData/mqm
此目录下的子目录包含所有产品数据。 每次替换产品时，都会删除并替换此目录中的数据。

UserData:

概述

```
QIBM
   '-- UserData
      '-- mqm
      '-- errors
      '-- trace
```

/QIBM/UserData/mqm
此目录下的子目录包含所有用户数据。

IBM MQ Java 的 IFS 根文件系统

在 IBM i 上安装 IBM MQ Java 时，将在 IFS 根文件系统中创建以下目录:

ProdData:

概述

```
QIBM
   '-- ProdData
      '-- mqm
      '-- java
      '--samples
      '-- bin
      '-- lib
```

/QIBM/ProdData/mqm/java
下面的子目录包含所有产品数据，包括 Java 类。 每次替换产品时，都会删除并替换此目录中的数据。

/QIBM/ProdData/mqm/java/samples
下面的子目录包含所有样本 Java 类和数据。

由服务器和客户机安装创建的库

安装 IBM MQ 服务器或客户机将创建以下库:

- QMQM

  产品库。

- QMQMSAMP

  样本库 (如果选择安装样本)。

- QMxxxx

  仅服务器。

  每次创建队列管理器时， IBM MQ 都会自动创建一个相关联的库，其名称类似于 QMxxxx ，其中 xxxx 派生自队列管理器名称。 此库包含特定于队列管理器的对象，包括日志和关联的接收方。 缺省情况下，此库的名称派生自以 QM 字符作为前缀的队列管理器的名称。 例如，对于名为 TEST 的队列管理器，库将被称为 QMTEST。

注: 创建队列管理器时，可以根据需要指定其库的名称。 例如：

```
CRTMQM MQMNAME(TEST) MQMLIB(TESTLIB)
```

可以使用 WRKLIB 命令来列示 IBM MQ for IBM i 已创建的所有库。 针对队列管理器库，您将看到文本 `QMGR： QMGRNAME`。 命令格式为：

```
WRKLIB LIB(QM*)
```

删除产品时，将保留这些与队列管理器关联的库。

Multi 规划 MFT on Multiplatforms 的文件系统支持

IBM MQ Managed File Transfer MFT 代理程序可用于将数据传输到文件系统上的文件以及从中传输数据。 除此之外，可以将代理程序中运行的资源监视器配置为监视文件系统上的文件。

MFT 要求这些文件存储在支持锁定的文件系统上。 原因有二：

- 代理程序锁定文件以确保它在开始从中读取数据或向其写入数据后不会更改。
- 资源监视器锁定文件以检查当前是否没有其他进程在使用这些文件。

代理程序和资源监视器使用 Java 方法 **`FileChannel.tryLock()`** 来执行锁定，并且当使用此调用要求文件系统锁定文件时，该文件系统必须能够锁定文件。

要点: 以下文件系统不受支持，因为它们不满足 MFT 的技术要求：

- GlusterFS
- NFS V3

Multi 在 Multiplatforms 版上选择循环或线性日志记录

在 IBM MQ 中，可以选择循环或线性日志记录。 以下信息提供了这两种类型的概述。

循环日志记录的优点

循环日志记录的主要优点是：

- 易于管理。

  为工作负载正确配置循环日志记录后，无需进一步管理。 而对于线性日志记录，需要记录介质图像，并且需要归档或删除不再需要的日志扩展数据块。
- 性能更佳

  循环日志记录的性能优于线性日志记录，因为循环日志记录能够复用已格式化的日志扩展数据块。 而线性日志记录必须分配新的日志扩展数据块并对其进行格式化。

请参阅 管理日志 以获取更多信息。

线性日志记录的优点

线性日志记录的主要优点是，线性日志记录提供了防止更多故障的保护。

循环或线性日志记录都无法防止已损坏或已删除的日志，或者应用程序或管理员已删除的消息或队列。

线性日志记录 (但不是循环) 使受损对象能够恢复。 因此，线性日志记录提供了防止队列文件损坏或删除的保护，因为这些损坏的队列可以从线性日志中恢复。

循环和线性保护以防止电源丢失和通信故障，如 从电源丢失或通信故障中恢复中所述。

其他注意事项

选择线性还是循环取决于需要多少冗余。

选择更多冗余 (即线性日志记录) 的成本由性能成本和管理成本引起。

请参阅 日志记录类型 以获取更多信息。

AIX AIX 上的共享内存

如果某些应用程序类型由于 AIX 内存限制而无法连接，那么在大多数情况下，可以通过设置环境变量 EXTSHM=ON 来解决此问题。

AIX 上的某些 32 位进程可能会迁到操作系统限制，这会影响它们连接到 IBM MQ 队列管理器的能力。 到 IBM MQ 的每个标准连接都使用共享内存，但与其他 UNIX 平台不同， AIX 仅允许 32 位进程连接 11 个共享内存集。

大多数 32 位进程不会迁到此限制，但是具有高内存需求的应用程序可能无法连接到 IBM MQ ，原因码为 2102: MQRC_RESOURCE_PROBLEM。 以下应用程序类型可能会看到此错误:

- 在 32 位 Java 虚拟机中运行的程序
- 使用大型或超大型内存模型的程序
- 连接到多个队列管理器或数据库的程序
- 独立连接到共享内存集的程序

AIX 为 32 位进程提供了扩展共享内存功能，允许它们连接更多共享内存。 要使用此功能运行应用程序，请在启动队列管理器和程序之前导出环境变量 EXTSHM=ON。 EXTSHM=ON 功能部件在大多数情况下可防止此错误，但它与使用 shmctl 函数的 SHM_SIZE 选项的程序不兼容。

IBM MQ MQI client 应用程序和所有 64 位进程不受此限制影响。 无论是否已设置 EXTSHM ，它们都可以连接到 IBM MQ 队列管理器。

Linux AIX IBM MQ 和 UNIX System V IPC 资源

队列管理器使用一些 IPC 资源。 使用 **ipcs -a** 来查找正在使用的资源。

此信息仅适用于在 AIX and Linux 系统上运行的 IBM MQ 。

IBM MQ 使用 System V 进程间通信 (IPC) 资源 (*semaphores* 和 共享内存段) 以在系统组件之间存储和传递数据。 这些资源由连接到队列管理器的队列管理器进程和应用程序使用。 IBM MQ MQI clients 不使用 IPC 资源，但 IBM MQ 跟踪控制除外。 使用 UNIX 命令 **ipcs -a** 来获取有关机器上当前正在使用的 IPC 资源的数量和大小的完整信息。

Linux AIX IBM MQ 和 UNIX 进程优先级

设置进程优先级 *nice* 值时的良好实践。

此信息仅适用于在 AIX and Linux 系统上运行的 IBM MQ 。

如果在后台运行进程，那么调用 shell 可以为该进程提供更高的 *nice* 值 (从而降低优先级)。 这可能具有一般的 IBM MQ 性能影响。 在高度紧张的情况下，如果有许多就绪可运行的线程处于较高优先级，而有些线程处于较低优先级，那么操作系统调度特性会使较低优先级的线程失去处理器时间。

与队列管理器 (例如 **runmqlsr**) 关联的独立启动的进程具有与它们关联的队列管理器相同的 *nice* 值是好的做法。 确保 shell 未将更高的 *nice* 值分配给这些后台进程。 例如，在 ksh 中，使用设置 “`set +o bgnice`” 来阻止 ksh 提高后台进程的 *nice* 值。 您可以通过检查 “ps -efl” 列表的 *NI* 列来验证正在运行的进程的 *nice* 值。

此外，使用与队列管理器相同的 *nice* 值来启动 IBM MQ 应用程序进程。 如果它们使用不同的 *nice* 值运行，那么应用程序线程可能会阻止队列管理器线程，反之亦然，从而导致性能下降。

z/OS Planning your IBM MQ environment on z/OS

When planning your IBM MQ environment, you must consider the resource requirements for data sets, page sets, Db2, Coupling Facilities, and the need for logging, and backup facilities. Use this topic to plan the environment where IBM MQ runs.

Before you plan your IBM MQ architecture, familiarize yourself with the basic IBM MQ for z/OS concepts, see the topics in IBM MQ for z/OS concepts.

When planning your queue manager, you might need to work with different people in your organization. It is usually a good idea to involve those people early, as change control procedures can take a long time. They might also be able to tell you what parameters you need to configure IBM MQ for z/OS.

For example you might need to work with the:

- Storage administrator, to determine the high level qualifier of queue manager data sets, and to allocate enough space for queue manager data sets.
- z/OS system programmer to define the IBM MQ subsystem to z/OS and APF authorize the IBM MQ for z/OS libraries.
- Network administrator to determine which TCP/IP stack and ports should be used for IBM MQ for z/OS.
- Security administrator to set up access to queue manager data sets, security profiles for IBM MQ for z/OS resources, and TLS certificates.
- Db2 administrator to set up Db2 tables when configuring a queue sharing group.

Related concepts

IBM MQ Technical overview

Related tasks

“规划 IBM MQ 体系结构” on page 5

规划 IBM MQ 环境时，请考虑 IBM MQ 为单个和多个队列管理器体系结构以及点到点和发布/预订消息传递样式提供的支持。 还要规划资源需求以及日志记录和备份工具的使用。

Configuring z/OS

Administering IBM MQ for z/OS

z/OS Planning for your queue manager

When you are setting up a queue manager, your planning should allow for the queue manager to grow, so that the queue manager meets the needs of your enterprise.

The best way to configure a queue manager is in steps:

1. Configure the base queue manager
2. Configure the channel initiator which does queue manager to queue manager communications, and remote client application communication
3. If you want to encrypt and protect messages, configure Advanced Message Security
4. If you want to use File Transfer over IBM MQ, configure Managed File Transfer for z/OS.
5. If you want to use the administrative or messaging REST API, or the IBM MQ Console to manage IBM MQ from a web browser, configure the mqweb server.

Some enterprises have hundreds of thousands of queue managers in their environment. You need to consider your IBM MQ network now, and in five years time.

On z/OS, some queue managers process thousands of messages a second, and log over 100 MB a second. If you expect very high volumes you may need to consider having more than one queue manager.

On z/OS, IBM MQ can run as part of a queue sharing group (QSG) where messages are stored in the Coupling Facility, and any queue manager in the queue sharing group can access the messages. If you want to run in a queue sharing group you need to consider how many queue managers you need. Typically, there is one queue manager for each LPAR. You might also have one queue manager to backup CF structures regularly.

Some changes to configuration are easy to do, such as defining a new queue. Some are harder, such as making logs and page sets bigger; and some configuration cannot be changed, such as the name of a queue manager or the queue sharing group name.

There is performance and tuning information available in the MP16 performance SupportPac .

Naming conventions

You need to have a naming convention for the queue manager data sets.

Many enterprises use the release number in the name of the load libraries, and so on. You might want to consider having an alias of `MQM.SCSQAUTH` pointing to the version currently in use, such as `MQM.V930.SCSQAUTH`, so you do not have to change CICS®, Batch, and IMS JCL when you migrate to a new version of IBM MQ.

You can use a symbolic link in z/OS UNIX System Services to reference the installation directory for the version of IBM MQ currently in use.

The data sets used by the queue manager (logs, page sets, JCL libraries) need a naming convention to simplify the creation of security profiles, and the mapping of data sets to SMS storage classes that control where the data sets are placed on disk, and the attributes they have.

Note, that putting the version of IBM MQ into the name of the page sets or logs, is not a good idea. One day you might migrate to a new version, and the data set will have the "wrong" names.

Applications

You need to understand the business applications and the best way to configure IBM MQ. For example if applications have logic to provide recovery and repeat capability, then non persistent messages might be good enough. If you want IBM MQ to handle the recovery, then you need to use persistent messages and put and get messages in syncpoint.

You need to isolate queues from different business transactions. If a queue for one business application fills up, you do not want this impacting other business applications. Isolate the queues in different page sets and buffer pools, or structures, if possible.

You need to understand the profile of messages. For many applications the queues have only a few messages. Other applications can have queues build up during the day, and be processed overnight. A queue which normally has only a few messages on it, might need to hold many hours worth of messages if there is a problem and messages are not processed. You need to size the CF structures and page sets to allow for your expected peak capacity.

Post configuration

Once you have configured your queue manager (and components) you need to plan for:

- Backing up page sets.
- Backing up definitions of objects.
- Automating the backup of any CF structures.
- Monitoring IBM MQ messages, and taking action when a problem is detected.
- Collecting the IBM MQ statistics data.
- Monitoring resource usage, such as virtual storage, and amount of data logged per hour. With this you can see if your resource usage is increasing and if you need to take actions, such as setting up a new queue manager

z/OS Planning your storage and performance requirements on z/OS

You must set realistic and achievable storage, and performance goals for your IBM MQ system. Use this topic help you understand the factors which affect storage, and performance.

This topic contains information about the storage and performance requirements for IBM MQ for z/OS. It contains the following sections:

- z/OS performance options for IBM MQ
- Determining z/OS workload management importance and velocity goals
- “Library storage” on page 122

- “System LX usage” on page 122
- “Storage configuration” on page 123
- “Disk storage” on page 128

See, “Where to find more information about storage and performance requirements” on page 128 for more information.

z/OS performance options for IBM MQ

With workload management, you define performance goals and assign a business importance to each goal. You define the goals for work in business terms, and the system decides how much resource, such as processor and storage, should be given to the work to meet its goal. Workload management controls the dispatching priority based on the goals you supply. Workload management raises or lowers the priority as needed to meet the specified goal. Thus, you need not fine-tune the exact priorities of every piece of work in the system and can focus instead on business objectives.

The three kinds of goals are:

Response time
How quickly you want the work to be processed

Execution velocity
How fast the work should be run when ready, without being delayed for processor, storage, I/O access, and queue delay

Discretionary
A category for low priority work for which there are no performance goals

Response time goals are appropriate for end-user applications. For example, CICS users might set workload goals as response time goals. For IBM MQ address spaces, velocity goals are more appropriate. A small amount of the work done in the queue manager is counted toward this velocity goal but this work is critical for performance. Most of the work done by the queue manager counts toward the performance goal of the end-user application. Most of the work done by the channel initiator address space counts toward its own velocity goal. The receiving and sending of IBM MQ messages, which the channel initiator accomplishes, is typically important for the performance of business applications using them.

Determining z/OS workload management importance and velocity goals

See “Determining z/OS workload management importance” on page 123 for more information.

Library storage

You must allocate disk storage for the product libraries. The exact figures depend on your configuration, and should include both the target and distribution libraries, as well as the SMP/E libraries.

The target libraries used by IBM MQ for z/OS use PDSE formats. Ensure that any PDSE target libraries are not shared outside a sysplex. For more information about the required libraries and their sizes and the required format, see the Program Directory. 有关程序目录的下载链接，请参阅 IBM MQ for z/OS 程序目录 PDF 文件。

System LX usage

Each defined IBM MQ subsystem reserves one system linkage index (LX) at IPL time, and a number of non-system linkage indexes when the queue manager is started. The system linkage index is reused when the queue manager is stopped and restarted. Similarly, distributed queuing reserves one non-system linkage index. In the unlikely event of your z/OS system having inadequate system LXs defined, you might need to take these reserved system LXs into account.

If required, the number of system LXs can be increased by setting the *NSYSLX* parameter in SYS1.PARMLIB member IEASYSxx.

z/OS *Determining z/OS workload management importance*

For full information about workload management and defining goals through the service definition, see the .z/OS product documentation.

This topic suggests how to set the z/OS workload management importance and velocity goals relative to other important work in your system. See *z/OS MVS Planning: Workload Management* for more information.

The queue manager address space needs to be defined with high priority as it provides subsystem services. The channel initiator is an application address space, but is usually given a high priority to ensure that messages being sent to a remote queue manager are not delayed. Advanced Message Security (AMS) also provides subsystem services and needs to be defined with high priority.

Use the following service classes:

The default SYSSTC service class

- VTAM and TCP/IP address spaces
- IRLM address space (IRLMPROC)

Note: The VTAM, TCP/IP, and IRLM address spaces must have a higher dispatching priority than all the DBMS address spaces, their attached address spaces, and their subordinate address spaces. Do not allow workload management to reduce the priority of VTAM, TCP/IP, or IRLM to (or below) that of the other DBMS address spaces

A high velocity goal and importance of 1 for a service class with a name that you define, such as PRODREGN, for the following:

- IBM MQ queue manager, channel initiator and AMS address spaces
- Db2 (all address spaces, except for the Db2-established stored procedures address space)
- CICS (all region types)
- IMS (all region types except BMPs)

A high velocity goal is good for ensuring that startups and restarts are performed as quickly as possible for all these address spaces.

The velocity goals for CICS and IMS regions are only important during startup or restart. After transactions begin running, workload management ignores the CICS or IMS velocity goals and assigns priorities based on the response time goals of the transactions that are running in the regions. These transaction goals should reflect the relative priority of the business applications they implement. They might typically have an importance value of 2. Any batch applications using IBM MQ should similarly have velocity goals and importance reflecting the relative priority of the business applications they implement. Typically the importance and velocity goals will be less than those for PRODREGN.

z/OS *Storage configuration*

V 9.4.0 In a 64 bit address space, there is a virtual line called “the bar” that marks the 2GB address. The bar separates storage below the 2GB address, called “below the bar”, from storage above the 2GB address, called “above the bar”. Storage below the bar uses 31 bit addressability, storage above the bar uses 64 bit addressability.

V 9.4.0

You can specify the limit of 31-bit storage by using the JCL REGION parameter, and the limit of 64-bit storage by using the MEMLIMIT parameter. These specified values can be overridden by z/OS exits.

Suggested storage configuration

The following table shows suggested **REGION** and **MEMLIMIT** values for the queue manager, channel initiator, and AMS address spaces. These suggestions should be used as a starting point and adjusted using the information in:

- “Queue manager storage configuration” on page 124
- “Channel initiator storage configuration from IBM MQ 9.4.0” on page 126

Table 19. Suggested definitions for REGION and MEMLIMIT	
Address space	**Storage configuration**
Queue manager	REGION=0M, MEMLIMIT=3G
V 9.4.0 Channel initiator from IBM MQ 9.4.0	REGION=0M, MEMLIMIT=2G
AMS address space	REGION=0M

Managing the MEMLIMIT and REGION size

Other mechanisms, for example the **MEMLIMIT** parameter in the SMFPRMxx member of SYS1.PARMLIB or the IEFUSI exit might be used at your installation to provide a default amount of virtual storage above the bar for z/OS address spaces. See Memory management above the bar for full details about limiting storage above the bar.

z/OS *Queue manager storage configuration*

The queue manager address space is likely to be the major user of 64-bit storage in an IBM MQ installation. Each connection to the queue manager requires common storage to be allocated as described in the following text. In addition to 64-bit storage, you should allow the queue manager to use all available 31-bit storage by specifying REGION=0M on the queue manager JCL.

Common storage

Each IBM MQ for z/OS subsystem has the following approximate storage requirements:

- CSA 4KB
- ECSA 800KB, plus the size of the trace table that is specified in the **TRACTBL** parameter of the CSQ6SYSP system parameter macro. For more information, see Using CSQ6SYSP.

In addition, each concurrent logical connection to the queue manager requires about 5 KB of ECSA. When a task ends, other IBM MQ tasks can reuse this storage.

IBM MQ does not release the storage until the queue manager is shut down, so you can calculate the maximum amount of ECSA required by multiplying the maximum number of concurrent connections by 5KB. The number of concurrent logical connections is the sum of the number of:

- Tasks (TCBs) in Batch, TSO, z/OS UNIX System Services, IMS, and Db2 stored procedure address space (SPAS) regions that are connected to IBM MQ, but not disconnected.
- CICS transactions that have issued an IBM MQ request, but have not terminated
- JMS Connections, Sessions, TopicSessions or QueueSessions that have been created (for bindings connection), but not yet destroyed or garbage collected.
- Active IBM MQ channels

You can set a limit to the common storage, used by logical connections to the queue manager, with the **ACELIM** configuration parameter. The **ACELIM** control is primarily of interest to sites where Db2 stored procedures cause operations on IBM MQ queues.

When driven from a stored procedure, each IBM MQ operation can result in a new logical connection to the queue manager. Large Db2 units of work, for example due to table load, can result in an excessive demand for common storage.

ACELIM is intended to limit common storage use and to protect the z/OS system, by limiting the number of connections in the system. You should only set **ACELIM** on queue managers that have been identified

as using excessive quantities of ECSA storage. See the **ACELIM** section in *Using CSQ6SYSP* for more information.

To set a value for **ACELIM**, firstly determine the amount of storage currently in the subpool controlled by the **ACELIM** value. This information is in the SMF 115 subtype 5 records produced by statistics CLASS(3) trace.

IBM MQ SMF data can be formatted using SupportPac MP1B. The number of bytes in use in the subpool controlled by **ACELIM** is displayed in the STGPOOL DD, on the line titled *ACE/PEB*.

For more information about SMF 115 statistics records, see Interpreting IBM MQ for z/OS performance statistics.

Increase the normal value by a sufficient margin to provide space for growth and workload spikes. Divide the new value by 1024 to yield a maximum storage size in KB for use in the **ACELIM** configuration.

Private storage

The queue manager address space uses 64-bit storage for many internal control blocks. The **MEMLIMIT** parameter of the queue manager JCL defines the maximum amount of 64-bit storage available. 3GB of storage, `MEMLIMIT=3G`, is the minimum you should use, however, depending on your configuration significantly more might be required.

You should specify a specific **MEMLIMIT** value rather than `MEMLIMIT=NOLIMIT` to prevent potential problems. If you specify `NOLIMIT` or a very large value, then there is the potential to use up all of the available z/OS virtual storage, which leads to paging in your system. When increasing the value of **MEMLIMIT** you should discuss the new setting with your z/OS system programmer in case there is a system-wide limit on the amount of on storage that can be used.

If you have a large value for **MEMLIMIT** you might need to increase the size of your dump data sets as more data is captured in a dump.

You can monitor the address space storage usage from the CSQY220I message that indicates the amount of 31 and 64-bit private storage in use, and the remaining free amount.

Buffer pools

Buffer pools are a significant user of private storage in the queue manager address space. Each buffer pool size is determined at queue manager initialization time, and storage is allocated for the buffer pool when a page set that is using that buffer pool is connected. The parameter **LOCATION (ABOVE|BELOW)** is used to specify where the buffers are allocated. You can use the ALTER BUFFPOOL command to dynamically change the size of buffer pools.

When calculating a value for **MEMLIMIT** it is critical that you take into account the buffer pool sizes if they are configured with **LOCATION(ABOVE)**. You should perform the calculation as follows.

Calculate the value of **MEMLIMIT** as 2GB plus the size of the buffer pools configured with **LOCATION(ABOVE)**, rounded up to the nearest GB. Set MEMLIMIT to a minimum of 3GB and increase this as necessary when you need to increase the size of your buffer pools.

For example, for three buffer pools configured with **LOCATION(ABOVE)**, buffer pool one has 10,000 buffers, and buffer pools two and three have 50,000 buffers each. Memory usage above the bar equals 110,000 (total number of buffers) * 4096 = 450,560,000 bytes = 430MB.

All buffer pools regardless of **LOCATION** make use of 64-bit storage for control structures. As the number of buffer pools and number of buffers in those pools increase this can become significant. Each buffer requires around an additional 200 bytes of 64-bit storage. For the preceding configuration that would require: 200 * 110,000 = 22,000,000 bytes = 21MB.

Therefore, in this scenario 3GB can be used for the **MEMLIMIT**, which allows scope for growth: 21MB + 430MB + 2GB which rounds up to 3GB.

For some configurations there can be significant performance benefits to using buffer pools that have their buffers permanently backed by real storage. You can achieve this by specifying the FIXED4KB value

for the **PAGECLAS** attribute of the buffer pool. However, you should only do this if there is sufficient real storage available on the LPAR, otherwise other address spaces might be affected. For information about when you should use the FIXED4KB value for **PAGECLAS**, see IBM MQ Support Pac MP16: IBM MQ for z/OS - Capacity planning & tuning.

Making the buffer pools so large that there is MVS™ paging might adversely affect performance. You might consider using a smaller buffer pool that does not page, with IBM MQ moving the message to and from the page set.

Indexed queues

On z/OS, local queues are indexed if the queue has an **INDXTYPE** attribute that has not been set to NONE. The indexes for shared queues are held in a coupling facility, but for private queues the index is held in 64 bit storage. For each message on an indexed queue 136 bytes of data are used to index the message. For very deep queues this can result in a significant amount of 64 bit storage being allocated. For example, 10 million messages on an indexed queue will use 1.27 GB of 64 bit storage in order to maintain the index.

If you expect to have a large number of messages on indexed queues you should allow for this when setting **MEMLIMIT**. To calculate an upper limit for the amount of storage required for indexes, multiply the **MAXDEPTH** attribute for each indexed queue by 136 and sum the value. This value should be added to your existing **MEMLIMIT**.

V 9.4.0 RECOVER CFSTRUCT

From IBM MQ 9.4.0 the **RECOVER CFSTRUCT** command makes greater use of 64-bit storage. In many cases there should be spare 64-bit storage available and so use of the command does not require an increase in the value of **MEMLIMIT**. However, if you are likely to have large structure backups, containing more than a few million messages, you should increase the **MEMLIMIT** for all queue managers which might process the **RECOVER CFSTRUCT** command by 500MB.

For example if you had MEMLIMIT=3G already, you should consider using MEMLIMIT=4G as the **MEMLIMIT** parameter does not allow for decimal points.

Shared Message Data Set (SMDS) buffers and MEMLIMIT

When running messaging workloads using shared message data sets, there are two levels of optimizations that can be achieved by adjusting the DSBUFS and DSBLOCK attributes.

The amount of above bar queue manager storage used by the SMDS buffer is DSBUFS x DSBLOCK. This means that by default, 100 x 256KB (25MB) is used for each CFLEVEL(5) structure in the queue manager.

Although this value is not too high, if your enterprise, or enterprises have many CFSTRUCTs, some of them might allocate a high value of MEMLIMIT for buffer pools, and sometimes they have deep indexed queues, so in total, they might run out of storage above the bar.

V 9.4.0 z/OS *Channel initiator storage configuration from IBM MQ 9.4.0*

The channel initiator typically uses much less 64-bit storage than the queue manager. However, from IBM MQ 9.4.0 the usage has increased. In addition to 64-bit storage, you should allow the channel initiator to use all available 31-bit storage by specifying REGION=0M on the queue manager JCL.

Common storage

The channel initiator typically requires ECSA usage of up to 160KB.

31-bit private storage

The 31-bit storage available to the channel initiator limits the number of concurrent connections the CHINIT can have.

Every channel uses approximately 170KB of extended private region in the channel initiator address space. For message channels, for example, sender or receiver channels, storage is increased by message size if messages larger than 32KB are transmitted. This increased storage is freed when:

- A sending or client channel requires less than half the current buffer size for 10 consecutive messages.
- A heartbeat is sent or received.

The storage is freed for reuse within the Language Environment, however, the storage is not seen as free by the z/OS virtual storage manager. This means that the upper limit for the number of channels is dependent on message size and arrival patterns, and on limitations of individual user systems on extended private region size.

The upper limit on the number of channels is likely to be approximately 9000 on many systems because the extended region size is unlikely to exceed 1.6GB.

The channel initiator trace is written to a data space. The size of the data space storage, is controlled by the **TRAXTBL** parameter. See ALTER QMGR.

64-bit private storage

The MEMLIMIT parameter of the channel initiator JCL defines the maximum amount of 64-bit storage available. 2 GB of storage, MEMLIMIT=2 GB, is the minimum value you should use. Depending on your configuration significantly more might be required.

You should specify a sensible MEMLIMIT value rather than MEMLIMIT=NOLIMIT to prevent potential problems. If you specify NOLIMIT or a very large value, then there is the potential to use up all of the available z/OS virtual storage, leading to paging in your system. When increasing the value of MEMLIMIT you should discuss the new setting with your z/OS system programmer in case there is a system-wide limit on the amount of on storage that can be used.

If you have a large value for MEMLIMIT you might need to increase the size of your dump data sets as more data is captured in a dump.

There are two users of 64-bit storage in the channel initiator: SMF and server-connection channels.

SMF

If enabled, SMF class 4 accounting, or statistics, require 64-bit storage. A minimum of 256MB storage is required. If sufficient storage is not available, the channel initiator issues the CSQX124E message and class 4 accounting and statistics are not available.

Server-connection channels

From IBM MQ 9.4.0 server-connection channels allocate message buffers in 64-bit storage, if they are transferring messages larger than 32 KB in size.

These buffers are freed if the channels require less than half the current buffer size for 10 consecutive messages, or a heartbeat is sent or received.

The value of MEMLIMIT sets an upper limit on how many concurrent server-connection channels can run. You should use a minimum value of MEMLIMIT=2G to ensure that the same number of channels can run as in earlier versions of IBM MQ, as well as providing some capacity for growth.

You can calculate an approximate value for MEMLIMIT by working out the peak maximum number of concurrently active server-connection channels, and for those channels the maximum message size you expect them to transfer. You should use MEMLIMIT=2GB as a starting point and round up.

For example, if you set the maximum number of concurrent server-connection channels to be 2,000 and each channel to have a maximum message size of 1MB, then server-connection channels are using a maximum of just under 2GB of 64-bit storage. As this is very close to 2GB then you should round up to MEMLIMIT=3G.

z/OS *Disk storage*

Use this topic when planning your disk storage requirements for log data sets, Db2 storage, coupling facility storage, and page data sets.

Work with your storage administrator to determine where to put the queue manager data sets. For example, your storage administrator may give you specific DASD volumes, or SMS storage classes, data classes, and management classes for the different data set types.

- Log data sets must be on DASD. These logs can have high I/O activity with a small response time and do not need to be backed up.
- Archive logs can be on DASD or tape. After they have been created, they might never be read again except in an abnormal situation, such as recovering a page set from a backup. They should have a long retention date.
- Page sets might have low to medium activity and should be backed up regularly. On a high use system, they should be backed up twice a day.
- BSDS data sets should be backed up daily; they do not have high I/O activity.

All data sets are similar to those used by Db2, and similar maintenance procedures can be used for IBM MQ.

See the following sections for details of how to plan your data storage:

- **Logs and archive storage**

  "How long do I need to keep archive logs" on page 146 describes how to determine how much storage your active log and archive data sets require, depending on the volume of messages that your IBM MQ system handles and how often the active logs are offloaded to your archive data sets.
- **Db2 storage**

  "Db2 storage" on page 163 describes how to determine how much storage Db2 requires for the IBM MQ data.
- **coupling facility storage**

  "Defining coupling facility resources" on page 154 describes how to determine how large to make your coupling facility structures.
- **Page set and message storage**

  "Planning your page sets and buffer pools" on page 129 describes how to determine how much storage your page data sets require, depending on the sizes of the messages that your applications exchange, on the numbers of these messages, and on the rate at which they are created or exchanged.

z/OS *Where to find more information about storage and performance requirements*

Use this topic as a reference to find more information about storage and performance requirements.

You can find more information from the following sources:

Table 20. Where to find more information about storage requirements

Topic	Where to look
System parameters	Using CSQ6SYSP and Customizing your queue managers
Storage required to install IBM MQ	Program Directory. 有关程序目录的下载链接，请参阅 IBM MQ for z/OS 程序目录 PDF 文件。
IEALIMIT and IEFUSI exits	See IEALIMIT and IEFUSI in the *z/OS:MVS Installation Exits* documentation.
Latest information	IBM MQ SupportPac website IBM MQ 和其他项目区域的 SupportPacs.

Table 20. Where to find more information about storage requirements (continued)	
Topic	**Where to look**
Workload management and defining goals through the service definition	*z/OS MVS Planning: Workload Management*

z/OS Planning your page sets and buffer pools

Information to help you with planning the initial number, and sizes of your page data sets, and buffer pools.

This topic contains the following sections:

Plan your page sets

Page set usage

For short-lived messages, few pages are normally used on the page set and there is little or no I/O to the data sets except at startup, during a checkpoint, or at shutdown.

For long-lived messages, those pages containing messages are normally written out to disk. This operation is performed by the queue manager in order to reduce restart time.

Separate short-lived messages from long-lived messages by placing them on different page sets and in different buffer pools.

Number of page sets

Using several large page sets can make the role of the IBM MQ administrator easier because it means that you need fewer page sets, making the mapping of queues to page sets simpler.

Using multiple, smaller page sets has a number of advantages. For example, they take less time to back up, and I/O can be carried out in parallel during backup and restart. However, consider that this adds a significant performance cost to the role of the IBM MQ administrator, who is required to map each queue to one of a much greater number of page sets.

Define at least five page sets, as follows:

- A page set reserved for object definitions (page set zero)
- A page set for system-related messages
- A page set for performance-critical long-lived messages
- A page set for performance-critical short-lived messages
- A page set for all other messages

"Defining your buffer pools" on page 134 explains the performance advantages of distributing your messages on page sets in this way.

Size of page sets

Define sufficient space in your page sets for the expected peak message capacity. Consider for any unexpected peak capacity, such as when a build-up of messages develops because a queue server program is not running. You can do this by allocating the page set with secondary extents or, alternatively, by enabling dynamic page set expansion. For more information, see "Enabling dynamic page set expansion" on page 132. It is difficult to make a page set smaller, so it is often better to allocate a smaller page set, and allow it to expand when needed.

When planning page set sizes, consider all messages that might be generated, including non-application message data. For example, trigger messages, event messages and any report messages that your application has requested.

The size of the page set determines the time taken to recover a page set when restoring from a backup, because a large page set takes longer to restore.

Note: Recovery of a page set also depends on the time the queue manager takes to process the log records written since the backup was taken; this time period is determined by the backup frequency. For more information, see "Planning for backup and recovery" on page 165.

Note: Page sets larger than 4 GB require the use of SMS extended addressability.

Planning for z/OS data set encryption

You can apply the z/OS data set encryption feature to page sets for queue managers running at IBM MQ for z/OS 9.1.4 or later.

You must allocate these page sets with EXTENDED attributes, and a data set key label that ensures the data is AES encrypted.

See the section, confidentiality for data at rest on IBM MQ for z/OS with data set encryption. for more information.

Calculate the size of your page sets

For queue manager object definitions (for example, queues and processes), it is simple to calculate the storage requirement because these objects are of fixed size and are permanent. For messages however, the calculation is more complex for the following reasons:

- Messages vary in size.
- Messages are transitory.
- Space occupied by messages that have been retrieved is reclaimed periodically by an asynchronous process.

Large page sets of greater than 4 GB that provide extra capacity for messages if the network stops, can be created if required. It is not possible to modify the existing page sets. Instead, new page sets with extended addressability and extended format attributes, must be created. The new page sets must be the same physical size as the old ones, and the old page sets must then be copied to the new ones. If backward migration is required, page set zero must not be changed. If page sets less than 4 GB are adequate, no action is needed.

Page set zero

Page set zero is reserved for object definitions.

For page set zero, the storage required is:

```
   (maximum number of local queue definitions x 1010)
      (excluding shared queues)
+  (maximum number of model queue definitions x 746)
+  (maximum number of alias queue definitions x 338)
+  (maximum number of remote queue definitions x 434)
+  (maximum number of permanent dynamic queue definitions x 1010)
+  (maximum number of process definitions x 674)
+  (maximum number of namelist definitions x 12320)
+  (maximum number of message channel definitions x 2026)
+  (maximum number of client-connection channel definitions x 5170)
+  (maximum number of server-connection channel definitions x 2026)
+  (maximum number of storage class definitions x 266)
+  (maximum number of authentication information definitions x 1010)
+  (maximum number of administrative topic definitions x 15000)
+  (total length of topic strings defined in administrative topic definitions)
```

Divide this value by 4096 to determine the number of records to specify in the cluster for the page set data set.

You do not need to allow for objects that are stored in the shared repository, but you must allow for objects that are stored or copied to page set zero (objects with a disposition of GROUP or QMGR).

The total number of objects that you can create is limited by the capacity of page set zero. The number of local queues that you can define is limited to 524 287.

Page sets 01 - 99

For page sets 01 - 99, the storage required for each page set is determined by the number and size of the messages stored on that page set. (Messages on shared queues are not stored on page sets.)

Divide this value by 4096 to determine the number of records to specify in the cluster for the page set data set.

Calculating the storage requirement for messages

This section describes how messages are stored on pages. Understanding this can help you calculate how much page set storage you must define for your messages. To calculate the approximate space required for all messages on a page set you must consider maximum queue depth of all the queues that map to the page set and the average size of messages on those queues.

Note: The sizes of the structures and control information given in this section are liable to change between major releases. For details specific to your release of IBM MQ, refer to SupportPac MP16 - IBM MQ , 用于 z/OS 容量规划和调整 and IBM MQ 系列-性能报告

You must allow for the possibility that message "gets" might be delayed for reasons outside the control of IBM MQ (for example, because of a problem with your communications protocol). In this case, the "put" rate of messages might far exceed the "get" rate. This can lead to a large increase in the number of messages stored in the page sets and a consequent increase in the storage size demanded.

Each page in the page set is 4096 bytes long. Allowing for fixed header information, each page has 4057 bytes of space available for storing messages.

When calculating the space required for each message, the first thing you must consider is whether the message fits on one page (a short message) or whether it needs to be split over two or more pages (a long message). When messages are split in this way, you must allow for additional control information in your space calculations.

For the purposes of space calculation, a message can be represented as the following:

The message header section contains the message descriptor and other control information, the size of which varies depending on the size of the message. The message data section contains all the actual message data, and any other headers (for example, the transmission header or the IMS bridge header).

A minimum of two pages are required for page set control information which, is typically less than 1% of the total space required for messages.

Short messages

A short message is defined as a message that fits on one page.

Small messages are stored one on each page.

Long messages

If the size of the message data is greater than 3596 bytes, but not greater than 4 MB, the message is classed as a long message. When presented with a long message, IBM MQ stores the message on a series of pages, and stores control information that points to these pages in the same way that it would store a short message. This is shown in Figure 41 on page 132:

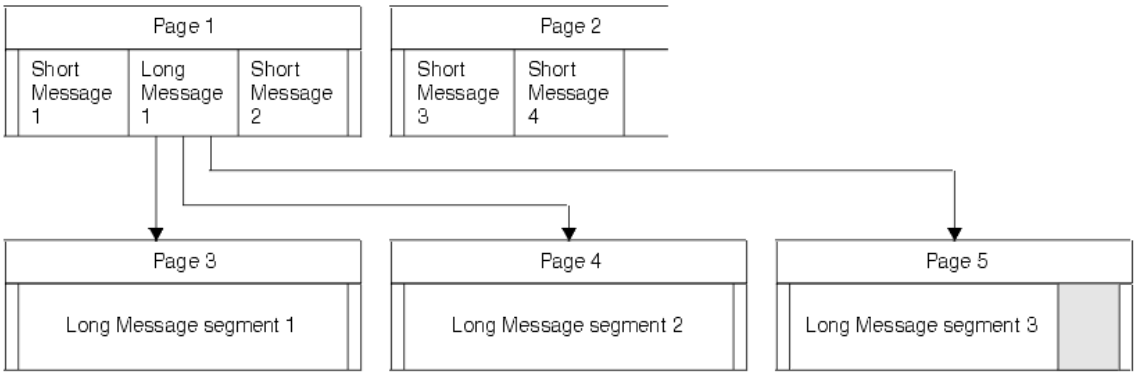


Figure 41. How IBM MQ stores long messages on page sets

Very long messages

Very long messages are messages with a size greater than 4 MB. These are stored so that each 4 MB uses 1037 pages. Any remainder is stored in the same way as a long message, as described above.

z/OS *Enabling dynamic page set expansion*

Page sets can be extended dynamically while the queue manager is running. A page set can have 123 extents, and can be spread over multiple disk volumes.

Each time a page set expands, a new data set extent is used. The queue manager continues to expand a page set when required, until the maximum number of extents has been reached, or until no more storage is available for allocation on eligible volumes.

Once page set expansion fails for one of the reasons above, the queue manager marks the page set for no further expansion attempts. This marking can be reset by altering the page set to EXPAND(SYSTEM).

Page set expansion takes place asynchronously to all other page set activity, when 90% of the existing space in the page set is allocated.

The page set expansion process formats the newly allocated extent and makes it available for use by the queue manager. However, none of the space is available for use, until the entire extent has been formatted. This means that expansion by a large extent is likely to take some time, and putting applications might 'block' if they fill the remaining 10% of the page set before the expansion has completed.

Sample thlqual.SCSQPROC(CSQ4PAGE) shows how to define the secondary extents.

To control the size of new extents, you use one of the following options of the EXPAND keyword of the DEFINE PSID and ALTER PSID commands:

- USER
- SYSTEM
- NONE

USER

Uses the secondary extent size specified when the page set was allocated. If a value was not specified, or if a value of zero was specified, dynamic page set expansion cannot occur.

Page set expansion occurs when the space in the page is 90% used, and is performed asynchronously with other page set activity.

This may lead to expansion by more than a single extent at a time.

Consider the following example: you allocate a page set with a primary extent of 100,000 pages and a secondary extent of 5000 pages. A message is put that requires 9999 pages. If the page set is already using 85,000 pages, writing the message crosses the 90% full boundary (90,000 pages). At this point, a further secondary extent is allocated to the primary extent of 100,000 pages, taking the page set size to 105,000 pages. The remaining 4999 pages of the message continue to be written. When the used page space reaches 94,500 pages, which is 90% of the updated page set size of 105,000 pages, another 5000 page extent is allocated, taking the page set size to 110,000 pages. At the end of the MQPUT, the page set has expanded twice, and 94,500 pages are used. None of the pages in the second page set expansion have been used, although they were allocated.

At restart, if a previously used page set has been replaced with a data set that is smaller, it is expanded until it reaches the size of the previously used data set. Only one extent is required to reach this size.

SYSTEM

Ignores the secondary extent size that was specified when the page set was defined. Instead, the queue manager sets a value that is approximately 10% of the current page set size. The value is rounded up to the nearest cylinder of DASD.

If a value was not specified, or if a value of zero was specified, dynamic page set expansion can still occur. The queue manager sets a value that is approximately 10% of the current page set size. The new value is rounded up depending on the characteristics of the DASD.

Page set expansion occurs when the space in the page set is approximately 90% used, and is performed asynchronously with other page set activity.

At restart, if a previously used page set has been replaced with a data set that is smaller, it is expanded until it reaches the size of the previously used data set.

NONE

No further page set expansion is to take place.

Related reference

ALTER PSID

DEFINE PSID

DISPLAYUSAGE

Defining your buffer pools

Use this topic to help plan the number of buffer pools you should define, and their settings.

This topic is divided into the following sections:

1. “Decide on the number of buffer pools to define” on page 134
2. “Decide on the settings for each buffer pool” on page 135
3. “Monitor the performance of buffer pools under expected load” on page 135
4. “Adjust buffer pool characteristics” on page 135

Decide on the number of buffer pools to define

You should define four buffer pools initially:

Buffer pool 0
Use for object definitions (in page set zero) and performance critical, system related message queues, such as the SYSTEM.CHANNEL.SYNCQ queue and the SYSTEM.CLUSTER.COMMAND.QUEUE and SYSTEM.CLUSTER.REPOSITORY.QUEUE queues.

However it is important to consider point “7” on page 136 in *Adjust buffer pool characteristics* if a large number of channels, or clustering, is to be used.

Use the remaining three buffer pools for user messages.

Buffer pool 1
Use for important long-lived messages.

Long-lived messages are those that remain in the system for longer than two checkpoints, at which time they are written out to the page set. If you have many long-lived messages, this buffer pool should be relatively small, so that page set I/O is evenly distributed (older messages are written out to DASD each time the buffer pool becomes 85% full).

If the buffer pool is too large, and the buffer pool never gets to 85% full, page set I/O is delayed until checkpoint processing. This might affect response times throughout the system.

If you expect few long-lived messages only, define this buffer pool so that it is sufficiently large to hold all these messages.

Buffer pool 2
Use for performance-critical, short-lived messages.

There is normally a high degree of buffer reuse, using few buffers. However, you should make this buffer pool large to allow for unexpected message accumulation, for example, when a server application fails.

Buffer pool 3
Use for all other (typically, performance noncritical) messages.

Queues such as the dead-letter queue, SYSTEM.COMMAND.* queues and SYSTEM.ADMIN.* queues can also be mapped to buffer pool 3.

Where virtual storage constraints exist, and buffer pools need to be smaller, buffer pool 3 is the first candidate for size reduction.

You might need to define additional buffer pools in the following circumstances:

- If a particular queue is known to require isolation, perhaps because it exhibits different behavior at various times.
  - Such a queue might either require the best performance possible under the varying circumstances, or need to be isolated so that it does not adversely affect the other queues in a buffer pool.
  - Each such queue can be isolated into its own buffer pool and page set.
- You want to isolate different sets of queues from each other for class-of-service reasons.

– Each set of queues might then require one, or both, of the two types of buffer pools 1 or 2, as described in Suggested definitions for buffer pool settings, necessitating creation of several buffer pools of a specific type.

Decide on the settings for each buffer pool

If you are using the four buffer pools described in “Decide on the number of buffer pools to define” on page 134, then Suggested definitions for buffer pool settings gives two sets of values for the size of the buffer pools.

The first set is suitable for a test system, the other for a production system or a system that will become a production system eventually. In all cases define your buffer pools with the **LOCATION**(*ABOVE*) attribute

Table 21. Suggested definitions for buffer pool settings

Definition setting	Test system	Production system
BUFFPOOL 0	1 050 buffers	50 000 buffers
BUFFPOOL 1	1 050 buffers	20 000 buffers
BUFFPOOL 2	1 050 buffers	50 000 buffers
BUFFPOOL 3	1 050 buffers	20 000 buffers

If you need more than the four suggested buffer pools, select the buffer pool (1 or 2) that most accurately describes the expected behavior of the queues in the buffer pool, and size it using the information in Suggested definitions for buffer pool settings.

Ensure that your MEMLIMIT is set high enough, so that all the buffer pools can be located above the bar.

Monitor the performance of buffer pools under expected load

You can monitor the usage of buffer pools by analyzing buffer pool performance statistics. In particular, you should ensure that the buffer pools are large enough so that the values of QPSTSOS, QPSTSTLA, and QPSTDMC remain at zero.

For further information, see Buffer manager data records.

Adjust buffer pool characteristics

Use the following points to adjust the buffer pool settings from “Decide on the settings for each buffer pool” on page 135, if required.

Use the performance statistics from “Monitor the performance of buffer pools under expected load” on page 135 as guidance.

1. If you are migrating from an earlier version of IBM MQ, only change your existing settings if you have more real storage available.
2. In general, bigger buffer pools are better for performance, and buffer pools can be much bigger if they are above the bar.

   However, at all times you should have sufficient real storage available so that the buffer pools are resident in real storage. It is better to have smaller buffer pools that do not result in paging, than big ones that do.

   Additionally, there is no point having a buffer pool that is bigger than the total size of the page sets that use it, although you should take into account page set expansion if it is likely to occur.
3. Aim for one page set per buffer pool, as this provides better application isolation.
4. If you have sufficient real storage, such that your buffer pools will never be paged out by the operating system, consider using page-fixed buffers in your buffer pool.

This is particularly important if the buffer pool is likely to undergo much I/O, as it saves the CPU cost associated with page-fixing the buffers before the I/O, and page-unfixing them afterwards.

5. There are several benefits to locating buffer pools above the bar even if they are small enough to fit below the bar. These are:
   - 31 bit virtual storage constraint relief - for example more space for common storage.
   - If the size of a buffer pool needs to be increased unexpectedly while it is being heavily used, there is less impact and risk to the queue manager, and its workload, by adding more buffers to a buffer pool that is already above the bar, than moving the buffer pool to above the bar and then adding more buffers.
6. Tune buffer pool zero and the buffer pool for short-lived messages (buffer pool 2) so that the 15% free threshold is never exceeded (that is, QPSTCBSL divided by QPSTNBUF is always greater than 15%). If more than 15% of buffers remain free, I/O to the page sets using these buffer pools can be largely avoided during normal operation, although messages older than two checkpoints are written to page sets.

   Attention: The optimum value for these parameters is dependent on the characteristics of the individual system. The values given are intended only as a guideline and might not be appropriate for your system.
7. SYSTEM.* queues which get very deep, for example SYSTEM.CHANNEL.SYNCQ, might benefit from being placed in their own buffer pool, if sufficient storage is available.

IBM MQ SupportPac MP16 - IBM MQ，用于 z/OS 容量规划和调整 provides further information about tuning buffer pools.

z/OS Planning your logging environment

Use this topic to plan the number, size and placement of the logs, and log archives used by IBM MQ.

Logs are used to:

- Write recovery information about persistent messages
- Record information about units of work using persistent messages
- Record information about changes to objects, such as define queue
- Backup CF structures

and for other internal information.

The IBM MQ logging environment is established using the system parameter macros to specify options, such as: whether to have single or dual active logs, what media to use for the archive log volumes, and how many log buffers to have.

These macros are described in Create the bootstrap and log data sets and Tailor your system parameter module.

Note: If you are using queue sharing groups, ensure that you define the bootstrap and log data sets with SHAREOPTIONS(2 3).

This section contains information about the following topics:

z/OS *Log data set definitions*

Use this topic to decide on the most appropriate configuration for your log data sets.

This topic contains information to help you answer the following questions:

- Should your installation use single or dual logging?
- How many active log data sets do you need?
- “How large should the active logs be?” on page 138
- Active log placement

- “Active log encryption with z/OS data set encryption” on page 139

Should your installation use single or dual logging?

In general you should use dual logging for production, to minimize the risk of losing data. If you want your test system to reflect production, both should use dual logging, otherwise your test systems can use single logging.

With single logging data is written to one set of log data sets. With dual logging data is written to two sets of log data sets, so in the event of a problem with one log data set, such as the data set being accidentally deleted, the equivalent data set in the other set of logs can be used to recover the data.

With dual logging you require twice as much DASD as with single logging.

If you are using dual logging, then also use dual BSDSs and dual archiving to ensure adequate provision for data recovery.

Dual active logging adds a small performance cost.

Attention: Use of disk mirroring technologies, such as Metro Mirror, are not necessarily a replacement for dual logging and dual BSDS. If a mirrored data set is accidentally deleted, both copies are lost.

If you use persistent messages, single logging can increase maximum capacity by 10-30% and can also improve response times.

Single logging uses 2 - 310 active log data sets, whereas dual logging uses 4 - 620 active log data sets to provide the same number of active logs. Thus single logging reduces the amount of data logged, which might be important if your installation is I/O constrained.

How many active log data sets do you need?

The number of logs depends on the activities of your queue manager. For a test system with low throughput, three active log data sets might be suitable. For a high throughput production system you might want the maximum number of logs available, so, if there is a problem with offloading logs you have more time to resolve the problems.

You must have at least three active log data sets, but it is preferable to define more. For example, if the time taken to fill a log is likely to approach the time taken to archive a log during peak load, define more logs.

Note: Page sets and active log data sets are eligible to reside in the extended addressing space (EAS) part of an extended address volumes (EAV) and an archive log dataset can also reside in the EAS.

You should also define more logs to offset possible delays in log archiving. If you use archive logs on tape, allow for the time required to mount the tape.

Consider having enough active log space to keep a day's worth of data, in case the system is unable to archive because of lack of DASD or because it cannot write to tape. If all the active logs fill up, then IBM MQ is unable to process persistent messages or transactions. It is very important to have enough active log space.

It is possible to dynamically define new active log data sets as a way of minimizing the effect of archive delays or problems. New data sets can be brought online rapidly, using the **DEFINE LOG** command to avoid queue manager 'stall' due to lack of space in the active log.

If you want to define more than 31 active log data sets, you must configure your logging environment to use a version 2 format BSDS. Once a version 2 format BSDS is in use, up to 310 active log data sets can be defined for each log copy ring. See “Planning to increase the maximum addressable log range” on page 148 for information on how you convert to a version 2 format BSDS.

You can tell whether your queue manager is using a version 2 or higher BSDS, either by running the print log map utility (CSQJU004), or from the CSQJ034I message issued during queue manager initialization.

An end of log RBA range of FFFFFFFFFFFFFFFF, in the CSQJ034I message, indicates that a version 2, or higher, format BSDS is in use. An end of log RBA range of 0000FFFFFFFFFFFF, in the CSQJ034I message, indicates that a version 1 format BSDS is in use.

When a queue manager is using a version 2, or higher, format BSDS, it is possible to use the **DEFINE LOG** command to dynamically add more than 31 active log data sets to a log copy ring.

How large should the active logs be?

The maximum supported active log size, when archiving to disk or to tape, is 4 GB.

You should create active logs of at least 1 GB in size for production and test systems.

Important: You need to be careful when allocating data sets, because IDCAMS rounds up the size you allocate.

To allocate a 3 GB log specify one of the following options:

- Cylinders(4369)
- Megabytes(3071)
- TRACKS(65535)
- RECORD(786420)

Any one of these allocates 2.99995 GB.

To allocate a 4GB log specify one of the following options:

- Cylinders(5825)
- Megabytes(4095)
- TRACKS(87375)
- RECORD(1048500)

Any one of these allocates 3.9997 GB.

When using striped data sets, where the data set is spread across multiple volumes, the specified size value is allocated on each DASD volume used for striping. So, if you want to use 4 GB logs and four volumes for striping, you should specify:

- CYLinders(1456)
- Megabytes(1023)

Setting these attributes allocates 4*1456 = 5824 Cylinders or 4 * 1023 = 4092 Megabytes.

Note: Striping is supported when using extended format data sets. This is usually set by the storage manager.

See Increasing the size of the active log for information on carrying out the procedure.

Active log placement

You should work with your storage management team to set up storage pools for the queue managers. You need to consider:

- A naming convention, so the queue managers use the correct SMS definitions.
- Space required for active and archive logs. Your storage pool should have enough space for the active logs from a whole day.
- Performance and resilience to failures.

For performance reasons you should consider striping your active log data sets. The I/O is spread across multiple volumes and reduces the I/O response times, leading to higher throughput. See the preceding text for information about allocating the size of the active logs when using striping.

You should review the I/O statistics using reports from RMF or a similar product. Perform the review of these statistics monthly (or more frequently) for the IBM MQ data sets, to ensure there are no delays due to the location of the data sets.

In some situations, there can be much IBM MQ page set I/O, and this can impact the IBM MQ log performance if they are located on the same DASD.

If you use dual logging, ensure that each set of active and archive logs is kept apart. For example, allocate them on separate DASD subsystems, or on different devices.

This reduces the risk of them both being lost if one of the volumes is corrupted or destroyed. If both copies of the log are lost, the probability of data loss is high.

When you create a new active log data, set you should preformat it using CSQJUFMT. If the log is not preformatted, the queue manager formats the log the first time it is used, which impacts the performance.

With older DASD with large spinning disks, you had to be careful which volumes were used to get the best performance.

With modern DASD, where data is spread over many PC sized disks, you do not need to worry so much about which volumes are used.

Your storage manager should be checking the enterprise DASD to review and resolve any performance problems. For availability, you might want to use one set of logs on one DASD subsystem, and the dual logs on a different DASD subsystem.

Active log encryption with z/OS data set encryption

You can apply the z/OS data set encryption feature to active log data sets for queue managers running at IBM MQ for z/OS 9.1.4 or later.

You must allocate these active log data sets with EXTENDED attributes, and a data set key label that ensures the data is AES encrypted.

See the section, confidentiality for data at rest on IBM MQ for z/OS with data set encryption. for more information.

z/OS *Using MetroMirror with IBM MQ*

IBM Metro Mirror, previously known as Synchronous Peer to Peer Remote Copy (PPRC), is a synchronous replication solution between two storage subsystems, where write operations are completed on both the primary and secondary volumes before the write operation is considered to be complete. Metro Mirror can be used in environments that require no data loss in the event of a storage subsystem failure.

Supported data set types

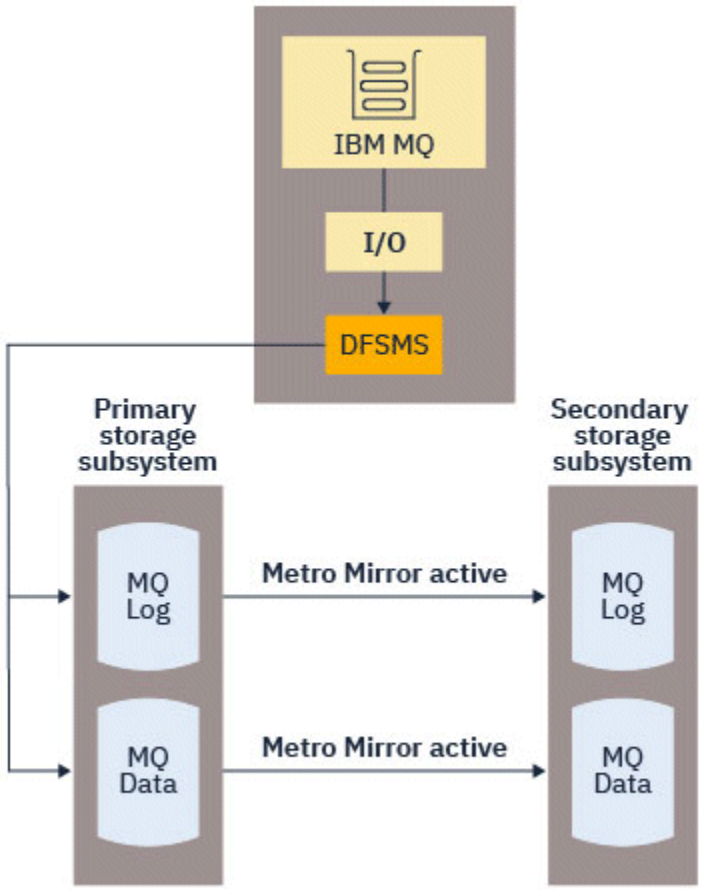
All of the following IBM MQ data set types can be replicated using Metro Mirror. However, exactly which ones are replicated depends on the availability requirements of your enterprise:

- Active logs
- Archive logs
- Bootstrap data set (BSDS)
- Page sets
- Shared message data set (SMDS)
- Data sets used for configuration, for example, in the CSQINP* DD cards on the MSTR JCL

Using zHyperWrite with IBM MQ active logs

When a write is made to a data set that is replicated using Metro Mirror, the write is first made to the primary volume, and then replicated to the secondary volume. This replication is done by the storage subsystem and is transparent to the application that issued the write, for example IBM MQ.

This process is illustrated in the following diagram.

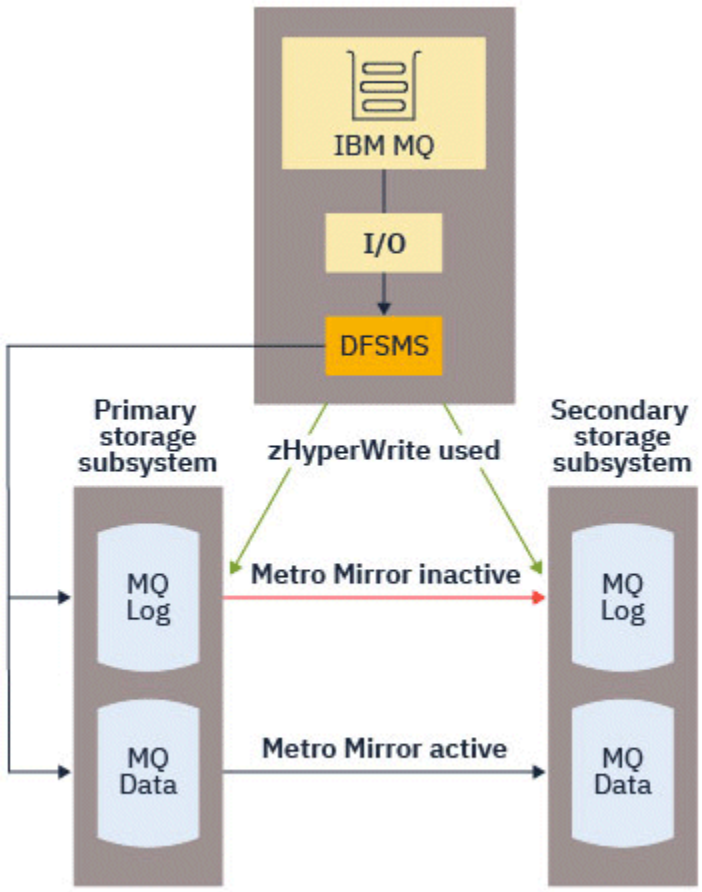


Because both writes to the primary and secondary storage subsystems need to complete before the write returns to IBM MQ, use of Metro Mirror can have a performance impact. You need to balance this performance impact against the availability benefits of using Metro Mirror.

The IBM MQ active logs are most sensitive to the performance impact of using Metro Mirror. IBM MQ allows use of zHyperWrite with the active logs to help reduce this performance impact.

zHyperWrite is a storage subsystem technology that works with z/OS to reduce the performance impact of writes made to data sets that are replicated using Metro Mirror. When zHyperWrite is used, the write to the primary and secondary volumes are issued in parallel at the Data Facility Storage Management Subsystem (DFSMS) level, instead of sequentially at the storage subsystem level, thereby reducing the performance impact.

The following diagram illustrates zHyperWrite being used for the active logs, and Metro Mirror being used for the other IBM MQ data set types. Note that if a zHyperWrite write fails, DFSMS will transparently reissue the write using Metro Mirror.



zHyperWrite on IBM MQ, is supported only on the active log data sets.

In order to use zHyperWrite with the active logs, you need to:

- Configure IBM MQ to use zHyperWrite, and
- The active logs need to be on zHyperWrite capable volumes

You can configure IBM MQ to use zHyperWrite by using one of the following methods:

- Specify ZHYWRITE(YES) in the system parameter module.
- Issue the command SET LOG ZHYWRITE(YES).

Set the following conditions for active log data sets to be on zHyperWrite capable volumes:

- Enable the volumes for Metro Mirror, and the volumes support zHyperWrite
- Ensure that the volumes are HyperSwap enabled
- Specify HYPERWRITE=YES in the IECIOSxx parameter

V 9.4.0 Prior to IBM MQ 9.4.0, if all the preceding conditions are met, then writes to the active logs are enabled for zHyperWrite. If one, or more, of these conditions are not met, IBM MQ writes to the active logs as normal, and Metro Mirror replicates the writes if it is configured.

V 9.4.0 From IBM MQ 9.4.0, if ZHYWRITE(YES) is specified, then IBM MQ always attempts to use zHyperWrite when writing to the active logs, regardless of whether the logs are on zHyperWrite capable volumes. If the logs are not on zHyperWrite capable volumes then Metro Mirror replicates the writes if it is configured. There are no negative effects of attempting to use zHyperWrite if the logs are not on zHyperWrite capable volumes

Notes:

- IBM MQ does not require that all active log data sets are on zHyperWrite capable volumes.

  If IBM MQ detects that some active log data sets are on zHyperWrite capable volumes, and others are not, it issues message CSQJ166E and carries on processing.
- IBM MQ checks whether active log data sets are zHyperWrite capable when the data sets are first opened.

  Log data sets are opened either at queue manager start up, or when dynamically adding using the DEFINE LOG command. If the log data sets are made zHyperWrite capable while a queue manager has them open, the queue manager will not detect this until it has been restarted.

You can use the output of the DISPLAY LOG command to indicate whether the current active log data sets are zHyperWrite capable. The following example shows that both of the data sets are zHyperWrite capable. If the queue manager has been configured with ZHYWRITE(YES), writes to these logs would be enabled for zHyperWrite:

```
Copy %Full zHyperWrite DSName
 1      4  CAPABLE     MQTST.SUBSYS.MQDL.LOGCOPY1.DS001
 2      4  CAPABLE     MQTST.SUBSYS.MQDL.LOGCOPY2.DS001
```

z/OS V 9.4.0 通过 *zHyper* 链路实现更快的日志吞吐量

zHyperLink 技术旨在通过在 CPU 与 I/O 设备之间提供快速，可靠且直接的通信路径，减少输入/输出 (I/O) 等待时间。

zHyper 链接概述

zHyperLink 可提高活动日志吞吐量，并将 IBM MQ 事务时间最多减少 3.5 倍。 要实现此目标，请在 z/OS 主机上安装 zHyper 链路适配器，选择 IBM 存储硬件，然后使用 zHyper 链路电缆连接这些适配器。 这将在 CPU 和 I/O 设备之间创建点对点连接，与 IBM z High Performance FICON ® (zHPF) 相比，最多可将 I/O 响应时间减少 10 倍。 这样低的响应时间是通过使用同步 I/O 请求来实现的。

同步 I/O 比异步 I/O 的优点

IBM MQ 记录器任务包含一个循环，用于等待需要写入日志的下一段数据。 当该数据可用时，记录器会调度写入，等待其完成，然后移至下一个数据段。

传统 I/O 比 CPU 慢，因此以异步方式执行 I/O 以释放 CPU 用于其他任务的效率最高。 因此，传统异步 I/O 需要暂挂记录器任务，直到写操作完成为止。 当写操作完成时，记录器任务必须等待 CPU 变为可用，添加短暂的重新分派延迟，以及由于重新填充 CPU 高速缓存而导致的延迟。

zHyperLink 提供更快的 I/O 时间，这些时间更接近 CPU 速度，因此，通过 zHyperLink ，可以同步执行 I/O ，这意味着记录器任务不会在写操作期间暂挂，从而除去重新分派和高速缓存相关延迟。

在发生写操作时，记录器任务仍在主动使用 CPU ，这将增加与传统 I/O 相比的 CPU 使用率。

如果队列管理器尝试使用 zHyper 链接，并且 zHyper 链接写入失败 (例如，由于配置问题) ，那么队列管理器将透明地回退到传统 I/O。

最低硬件需求

- IBM z14 或更高版本
- DS8880 或更高版本

软件需求

- zHyperLink Express 在 z/OS 2.3 或更高版本上受支持。
- z/OS 映像必须在 LPAR 中运行，而不是在 IBM z/VM®下作为访客运行。
- zHyper 链接需要启用 IBM z High Performance FICON (zHPF)。

使用 zHyper 链接 IBM MQ 活动日志

要将 zHyper 链接与队列管理器的活动日志配合使用，您需要:

- 配置 IBM MQ 以使用 zHyper 链接，以及
- 确保活动日志位于支持 zHyper 链接的卷上。

  请参阅 IBM zHyperLink for z/OS 入门，以获取更多信息。

您可以使用下列其中一种方法将 IBM MQ 配置为使用 zHyper 链接:

- 在日志参数中指定 ZHYLINK(YES)。
- 发出命令 SET LOG ZHYLINK(YES)。

注意:

- zHyper 链接要求开启 zHyper 写。 这意味着为了使用 ZHYLINK，还必须在日志参数中打开 ZHYWRITE。仅当在队列管理器上设置了 ZHYWRITE (NO) 时指定 ZHYLINK (YES) 时， ZHYWRITE 参数会自动覆盖为 YES。
- 显式尝试使用 ZHYWRITE (NO) 设置 ZHYLINK (YES) 会导致 SET LOG 命令异常完成。
- 在 ZPRM 中设置 ZHYLINK=YES 会将 ZHYWRITE 覆盖为 YES。

如果迁到任何问题，请参阅 故障诊断 zHyper 链接 以获取更多信息。

IBM MQ 不要求所有活动日志数据集都位于支持 zHyper 链接的卷上，但建议您这样做。 如果 IBM MQ 检测到某些活动日志数据集位于支持链接的 zHyper 卷上，而其他数据集不在其中，那么它会发出消息 CSQJ601E 并继续处理。

IBM MQ 会在首次打开数据集时检查活动日志数据集是否支持 zHyper 链接。 日志数据集在队列管理器启动时打开，或者在使用 DEFINE LOG 命令动态添加时打开。 如果在队列管理器打开日志数据集时使其具有 zHyper 链接功能，那么队列管理器直到重新启动后才会检测到此数据集。

如果指定了 ZHYLINK (YES)，那么 IBM MQ 在写入活动日志时始终尝试使用 zHyper 链接，而不考虑日志是否位于支持 zHyper 链接的卷上。 如果日志不在支持 zHyper 链接的卷上，那么尝试使用 zHyper 链接不会产生负面影响。

可以使用 DISPLAY LOG 命令的输出来指示当前活动日志数据集的 zHyper 链接的状态:

```
Copy %Full zHyperWrite    Encrypted          DSName
  1     81 YES          NO                 MQTST.SUBSYS.MQDL.LOGCOPY1.DS001
  2     81 YES            NO                 MQTST.SUBSYS.MQDL.LOGCOPY2.DS001
Copy zHyperLink
  1  YES
  2  YES
```

zHyper 链路状态为下列其中一项:

YES
将在队列管理器上启用 zHyper 链接，并且将在所有写操作中尝试此链接。

否
在队列管理器上未启用 zHyper 链接，并且在支持 zHyper 链接的卷上，数据集 **不是** 。

能够
在队列管理器上未启用 zHyper 链接，并且在支持 zHyper 链接的卷上，数据集 **为** 。

还有多个其他 SMF 统计信息用于监视和了解 zHyper 链路性能; 请参阅 zHyper 链路统计信息 以获取详细信息。

写会话

使用 zHyper 链路时，将与 DASD 建立一个或多个 zHyper 链路写入会话。 当前 DASD 最多支持 64 个并发写会话，因此您应该仔细考虑启用 zHyper 链路的队列管理器，以及其他子系统 (例如 Db2) 是否也使用 zHyper 链路来写入同一 DASD。 如果耗尽可用写会话，那么队列管理器会自动切换回使用传统异步 I/O。

您可以计算 zHyper 链接写入会话数，如下所示:

```
Number of log copies (either 1 or 2) * number of stripes per log copy * 2
if Metro Mirror (PPRC) is used.
```

因此，具有一个条带且没有 Metro Mirror 的单个日志记录方式的队列管理器使用单个写会话。 具有两个条带和 PPRC 的双日志记录方式的队列管理器使用 8 写会话。

注: 虽然 Metro Mirror 导致使用的写会话数是使用的写会话数的两倍，但这些写会话在两个镜像的 DASD 之间平均分割。

z/OS *Planning your log archive storage*

Use this topic to understand the different ways of maintaining your archive log data sets.

You can place archive log data sets on standard-label tapes, or DASD, and you can manage them by data facility hierarchical storage manager (DFHSM). Each z/OS logical record in an archive log data set is a VSAM control interval from the active log data set. The block size is a multiple of 4 KB.

Archive log data sets are dynamically allocated, with names chosen by IBM MQ. The data set name prefix, block size, unit name, and DASD sizes needed for such allocations are specified in the system parameter module. You can also choose, at installation time, to have IBM MQ add a date and time to the archive log data set name.

It is not possible to specify with IBM MQ, specific volumes for new archive logs, but you can use Storage Management routines to manage this. If allocation errors occur, offloading is postponed until the next time offloading is triggered.

If you specify dual archive logs at installation time, each log control interval retrieved from the active log is written to two archive log data sets. The log records that are contained in the pair of archive log data sets are identical, but the end-of-volume points are not synchronized for multivolume data sets.

Should your archive logs reside on tape or DASD?

When deciding whether to use tape or DASD for your archive logs, there are a number of factors that you should consider:

- Review your operating procedures before deciding about tape or disk. For example, if you choose to archive to tape, there must be enough tape drive when they are required. After a disaster, all subsystems might want tape drives and you might not have as many free tape drives as you expect.
- During recovery, archive logs on tape are available as soon as the tape is mounted. If DASD archives have been used, and the data sets migrated to tape using hierarchical storage manager (HSM), there is a delay while HSM recalls each data set to disk. You can recall the data sets before the archive log is used. However, it is not always possible to predict the correct order in which they are required.
- When using archive logs on DASD, if many logs are required (which might be the case when recovering a page set after restoring from a backup) you might require a significant quantity of DASD to hold all the archive logs.
- In a low-usage system or test system, it might be more convenient to have archive logs on DASD to eliminate the need for tape mounts.
- Both issuing a RECOVER CFSTRUCT command, and backing out a persistent unit of work, result in the log being read backwards. Tape drives with hardware compression perform badly on operations that read backwards. Plan sufficient log data on DASD to avoid reading backwards from tape.

Archiving to DASD offers faster recoverability but is more expensive than archiving to tape. If you use dual logging, you can specify that the primary copy of the archive log go to DASD and the secondary copy go to tape. This increases recovery speed without using as much DASD, and you can use the tape as a backup.

See “Changing the storage medium for archive logs” on page 146 for details of how you archive your logs from tape to DASD, and how you carry out the reverse process.

Archiving to tape

If you choose to archive to a tape device, IBM MQ can extend to a maximum of 20 volumes.

If you are considering changing the size of the active log data set so that the set fits on one tape volume, note that a copy of the BSDS is placed on the same tape volume as the copy of the active log data set. Adjust the size of the active log data set downward to offset the space required for the BSDS on the tape volume.

If you use dual archive logs on tape, it is typical for one copy to be held locally, and the other copy to be held off-site for use in disaster recovery.

Archiving to DASD volumes

IBM MQ requires that you catalog all archive log data sets allocated on non-tape devices (DASD). If you choose to archive to DASD, the CATALOG parameter of the CSQ6ARVP macro must be YES. If this parameter is NO, and you decide to place archive log data sets on DASD, you receive message CSQJ072E each time an archive log data set is allocated, although IBM MQ still catalogs the data set.

If the archive log data set is held on DASD, the archive log data sets can extend to another volume; multivolume is supported.

If you choose to use DASD, make sure that the primary space allocation (both quantity and block size) is large enough to contain either the data coming from the active log data set, or that from the corresponding BSDS, whichever is the larger of the two.

This minimizes the possibility of unwanted z/OS X'B37' or X'E37' abend codes during the offload process. The primary space allocation is set with the PRIQTY (primary quantity) parameter of the CSQ6ARVP macro.

Archive log data sets can exist on large or extended-format sequential data sets. SMS ACS routines now use DSNTYPE(LARGE) or DSNTYPE(EXT).

IBM MQ supports allocation of archive logs as extended format data sets. When extended format is used, the maximum archive log size is increased from 65535 tracks to the maximum active log size of 4GB. Archive logs are eligible for allocation in the extended addressing space (EAS) of extended address volumes (EAV).

Where the required hardware and software levels are available, allocating archive logs to a data class defined with COMPACTION using zEDC might reduce the disk storage required to hold archive logs. For more information, see IBM MQ for z/OS: Reducing storage occupancy with IBM zEnterprise Data Compression (zEDC) and zEnterprise Data Compression (zEDC) for more information.

The z/OS data set encryption feature can be applied to archive logs for queue managers running on IBM MQ. These archive logs must be allocated through Automatic Class Selection (ACS) routines to a data class defined with EXTENDED attributes, and a data set key label that ensures the data is AES encrypted.

Using SMS with archive log data sets

If you have MVS/DFP storage management subsystem (DFSMS) installed, you can write an Automatic Class Selection (ACS) user-exit filter for your archive log data sets, which helps you convert them for the SMS environment.

Such a filter, for example, can route your output to a DASD data set, which DFSMS can manage. You must exercise caution if you use an ACS filter in this manner. Because SMS requires DASD data sets to be cataloged, you must make sure the CATALOG DATA field of the CSQ6ARVP macro contains YES. If it does not, message CSQJ072E is returned; however, the data set is still cataloged by IBM MQ.

For more information about ACS filters, see Data sets that DFSMShsm dynamically allocates during aggregate backup processing.

z/OS *Changing the storage medium for archive logs*

The procedure for changing the storage medium used by archive logs.

About this task

This task describes how to change the storage medium used for archive logs, for example moving from archiving to tape to archiving to DASD.

You have a choice of how to make the changes:

1. Make the changes only using the CSQ6ARVP macro so that they are applied from the next time the queue manager restarts.
2. Make the changes using the CSQ6ARVP macro, and dynamically using the SET ARCHIVE command. This means that the changes apply from the next time the queue manager archives a log file, and persist after the queue manager restarts.

Procedure

1. Changing so archive logs are stored on DASD instead of tape:
   a) Read the section "Archiving to DASD volumes" on page 145 and review the CSQ6ARVP parameters.
   b) Make changes to the following parameters in CSQ6ARVP
      - Update the UNIT and, if necessary, the UNIT2 parameters.
      - Update the BLKSIZE parameter, as the optimal setting for DASD differs from tape.
      - Set the PRIQTY and SECQTY parameters to be large enough to hold the largest of the active log or BSDS.
      - Set the CATALOG parameter to be YES.
      - Confirm the ALCUNIT setting is what you want. You should use BLK, because it is independent of the device type.
      - Set the ARCWTOR parameter to NO if it is not already.
2. Changing so archive logs are stored on tape instead of DASD:
   a) Read the section "Archiving to tape" on page 145, and review the CSQ6ARVP parameters.
   b) Make changes to the following parameters in CSQ6ARVP:
      - Update the UNIT and, if necessary, the UNIT2 parameters.
      - Update the BLKSIZE parameter, as the optimal setting for tape differs from DASD.
      - Confirm the ALCUNIT setting is what you want. You should use BLK, because it is independent of the device type.
      - Review the setting of the ARCWTOR parameter.

z/OS ***How long do I need to keep archive logs***

Use the information in this section to help you plan your backup strategy.

You specify how long archive logs are kept in days , using the ARCRETN parameter in USING CSQ6ARVP or the SET SYSTEM command. After this period the data sets can be deleted by z/OS.

You can manually delete archive log data sets when they are no longer needed.

- The queue manager might need the archive logs for recovery.

  The queue manager can only keep the most recent 1000 archives in the BSDS, When the archive logs are not in the BSDS they cannot be used for recovery, and are only of use for audit, analysis, or replay type purposes.
- You might want to keep the archive logs so that you can extract information from the logs. For example, extracting messages from the log, and reviewing which user ID put or got the message.

The BSDS contains information on logs and other recovery information. This data set is a fixed size. When the number of archive logs reaches the value of MAXARCH in CSQ6LOGP, or when the BSDS fills up, the oldest archive log information is overwritten.

There are utilities to remove archive log entries from the BSDS, but in general, the BSDS wraps and overlays the oldest archive log record.

When is an archive log needed

You need to back up your page sets regularly. The frequency of backups determines which archive logs are needed in the event of losing a page set.

You need to back up your CF structures regularly. The frequency of backups determines which archive logs are needed in the event of losing data in the CF structure.

The archive log might be needed for recovery. The following information explains when the archive log might be needed, where there are problems with different IBM MQ resources.

Loss of a page set
You must recover your system from your backup and restart the queue manager.

You need the logs from when the backup was taken, as well as up to three log data sets prior to the backup being taken.

All LPARs lose connectivity to a CF structure, or the structure is unavailable
Use the RECOVER CFSTRUCT command to recover the structure.

Structure recovery requires the logs from all queue managers that have accessed the structure since the last backup (back to the time when the backup was taken) plus the structure backup itself in the log of the queue manager that took the backup.

If you have been doing frequent backups of the CF structures, the data should be in active logs, and you should not need archive logs.

If there is no recent backup of the CF structure, you might need archive logs.

Note: All non persistent messages will be lost; all persistent messages will be re-created by performing the following tasks:

1. Reading the last CF structure backup from the log
2. Reading the logs from all queue managers that have used the structure
3. Merging updates since the backup

Administration structure rebuild
If you need to rebuild the administration structure, the information is read from the last checkpoint of the log for each queue manager in the QSG.

If a queue manager is not active, another queue manager in the QSG reads the log.

You should not need archive logs.

Loss of an SMDS data set
If you lose an SMDS data set, or the data set gets corrupted, the data set becomes unusable and the status for it is set to FAILED. The CF structure is unchanged.

In order to restore the SMDS data set, you need to:

1. Redefine the SMDS data set, and
2. Recover the CF structure by issuing the RECOVER CFSTRUCT command.

   Note: All non persistent messages on the CF structure will be lost; all persistent messages will be restored.

The requirement for queue manager logs is the same as for recovering from a structure that is unavailable.

z/OS *Planning to increase the maximum addressable log range*

You can increase the maximum addressable log range by configuring your queue manager to use a larger log relative byte address (RBA).

The log RBA size was increased from IBM MQ for z/OS 8.0. For an overview of this change, see Larger log Relative Byte Address.

Queue managers created at IBM MQ 9.3.0 or later, have 8 byte log RBA enabled by default and, therefore, do not require conversion.

You can convert your queue managers to use 8 byte log RBA values at any time. A queue sharing group can contain some queue managers with 8 byte log RBA enabled, and some queue managers with 6 byte log RBA enabled.

Undoing the change

The change cannot be backed out.

How long does it take?

The change requires a queue manager restart. Stop the queue manager, run the CSQJUCNV utility against the bootstrap data set (BSDS), or data sets, to create new data sets, rename these bootstrap data sets, and restart the queue manager. The CSQJUCNV utility usually takes a few seconds to run.

What impact does this have?

- With 8 byte log RBA in use, every write of data to the log data sets has additional bytes. Therefore, for a workload consisting of persistent messages there is a small increase in the amount of data written to the logs.
- Data written to a page set, or coupling facility (CF) structure, is not affected.

Related tasks

Implementing the larger log Relative Byte Address

z/OS Planning your channel initiator

The channel initiator provides communications between queue managers, and runs in its own address space.

There are two types of connections:

1. Application connections to a queue manager over a network. These are known as client channels.
2. Queue manager to queue manager connections. These are known as MCA channels.

Listeners

A channel listener program listens for incoming network requests and starts the appropriate channel when that channel is needed. To process inbound connections the channel initiator needs at least one IBM MQ listener task configured. A listener can either be a TCP listener, or a LU 6.2 listener.

Each listener requires a TCP port or LU name.

Note that you can have more than one listener for each channel initiator.

TCP/IP

A channel initiator can operate with more than one TCP stack on the same z/OS image. For example, one TCP stack could be for internal connections, and another TCP stack for external connections.

When you define an output channel:

1. You set the destination host and port of the connection. This can be either:
   - an IP address, for example `10.20.4.6`
   - a host name, for example `mvs-prod.myorg.com`

   If you use a host name to specify the destination, IBM MQ uses the Domain Name System (DNS) to resolve the IP address of the destination.
2. If you are using multiple TCP stacks you can specify the **LOCLADDR** parameter on the channel definition, which specifies the IP Stack address to be used.

You should plan to have a highly available DNS server, or DNS servers. If the DNS is not available, outbound channels might not be able to start, and channel authentication rules that map an incoming connection using a host name cannot be processed.

APPC and LU 6.2

If you are using APPC, the channel initiator needs an LU name, and configuration in APPC.

Queue sharing groups

To provide a single system image, and allow an incoming IBM MQ connection request to go to any queue manager in the queue sharing group, you need to do some configuration. For example:

1. A hardware network router. This router has one IP address seen by the enterprise, and can route the initial request to any queue manager connected to this hardware.
2. A Virtual IP address (VIPA). An enterprise wide IP address is specified, and that address can be routed to any one of the TCP stacks in a sysplex. The TCP stack can then route it to any listening queue manager in the sysplex.

Protecting IBM MQ traffic

You can configure IBM MQ to use TLS connections to protect data on the wire. To use TLS you need to use digital certificates and key rings.

You also need to work with the personnel at the remote end of the channel, to ensure that you have compatible IBM MQ definitions and compatible certificates.

You can control which connections can connect to IBM MQ and the user ID, based on

- IP address
- Client user ID
- Remote queue manager, or
- Digital certificate (see Channel Authentication Records)

It is also possible to restrict client applications by ensuring that they supply a valid user ID and password (see Connection Authentication).

You can get the channel initiator working, and then configure each channel to use TLS, one at a time.

Monitoring the channel initiator

There are MQSC commands that give information about the channel initiator and channels:

- The DISPLAY CHINIT command gives information about the channel initiator, and active listeners.
- The DISPLAY CHSTATUS command displays the activity and status of a channel.

The channel initiator can also produce SMF records with information about the channel initiator tasks and channel activity. See “Planning for channel initiator SMF data” on page 151 for more information.

The channel initiator emits messages to the job log when channels start and stop. Automation in your enterprise can use these messages to capture status. As some channels are active for only a few seconds,

many messages can be produced. You can suppress these messages either by using the z/OS message processing facility, or by setting **EXCLMSG** with the SET SYSTEM command.

Configuring your IBM MQ channel definitions

When you have many queue managers connected together it can be hard to manage all the object definitions. Using IBM MQ clustering can simplify this.

You specify two queue managers as full repositories. Other queue managers need one connection to, and one connection from, one of the repositories. When connections to other queue managers are needed, the queue manager creates and starts channels automatically.

If you are planning to have a large number of queue managers in a cluster, you should plan to have queue managers that act as dedicated repositories and have no application traffic.

See “规划分布式队列和集群” on page 17 for more information.

Actions before you configure the channel initiator

1. Decide if you are using TCP/IP or APPC.
2. If you are using TCP, allocate at least one port for IBM MQ.
3. If you need a a DNS server, configure the server to be highly available if required.
4. If you are using APPC, allocate an LU name, and configure APPC.

Actions after you have configured the channel initiator, before you go into production

1. Plan what connections you will have:
   a. Client connections from remote applications.
   b. MCA channels to and from other queue managers. Typically you have a channel to and from each remote queue manager.
2. Set up clustering, or join an existing clustering environment.
3. Consider whether you need to use multiple TCP stacks, VIPA, or an external router for availability in front of the channel initiator.
4. If you are planning on using TLS:
   a. Set up the key ring
   b. Set up certificates
5. If you are planning on using channel authentication:
   a. Decide the criteria for mapping inbound sessions to MCA user IDs
   b. Enable reverse DNS lookup by setting the queue manager parameter **REVDNS**
   c. Review security. For example, delete the default channels, and specify user IDs with only the necessary authority in the **MCAUSER** attribute for a channel.
6. Capture the accounting and statistics SMF records produced by the channel initiator and post process them.
7. Automate the monitoring of job log messages.
8. If necessary, tune your network environment to improve throughput. With TCP, large send and receive buffers improve throughput. You can force MQ to use specific TCP buffer sizes using the commands:

   ```
   RECOVER QMGR(TUNE CHINTCPRBDYNSZ nnnnn)
   RECOVER QMGR(TUNE CHINTCPSBDYNSZ nnnnn)
   ```

   which sets the SO_RCVBUF, and SO_SNDBUF, for the channels to the size in bytes specified in nnnnn.

Related concepts

“Planning for your queue manager” on page 120
When you are setting up a queue manager, your planning should allow for the queue manager to grow, so that the queue manager meets the needs of your enterprise.

z/OS Planning for channel initiator SMF data

You need to plan the implementation of collecting SMF data for the channel initiator.

The channel initiator produces two types of record:

- Statistics data with information about the channel initiator and the tasks within it.
- Channel accounting data with information similar to the DISPLAY CHSTATUS command.

You start collecting statistics data using the command:

```
START TRACE(STAT) CLASS(4)
```

and stop it using the command:

```
STOP TRACE(STAT) CLASS(4)
```

You start collecting accounting data using the command:

```
START TRACE(ACCTG) CLASS(4)
```

and stop it using the command:

```
STOP TRACE(ACCTG) CLASS(4)
```

You can control which channels have accounting data collected for using the **STATCHL** attribute on the channel definition or the queue manager.

- For client channels, you must set **STATCHL** at the queue manager level.
- For automatically defined cluster sender channels, you can control the collection of accounting data with the **STATACLS** queue manager attribute.

The default value of **STATCHL** for the queue manager is OFF. In order to collect channel accounting data you must change the value of **STATCHL** from the default on either the queue manager or channel definition, in addition to starting class 4 accounting trace.

The SMF records are produced when:

- From IBM MQ for z/OS 9.3.0 onwards, the time interval indicated by the CSQ6SYSP **STATIME** or **ACCTIME** parameters has elapsed; or, if **STATIME** or **ACCTIME** is zero on the SMF data collection broadcast. The requests to collect SMF data for the channel initiator and the queue manager are synchronized.
- A `STOP TRACE(ACCTG) CLASS(4)` or `STOP TRACE(STAT) CLASS(4)` command is issued, or
- The channel initiator is shut down. At this point any SMF data is written out.

If a channel stops during the SMF interval, accounting data is written to SMF the next time the SMF processing runs. If a client connects, does some work and disconnects, then reconnects and disconnects, there are two sets of channel accounting data produced.

The statistics data normally fits into one SMF record, however, multiple SMF records might be created if a large number of tasks are in use.

Accounting data is gathered for each channel for which it is enabled, and normally fits into one SMF record. However, multiple SMF records might be created if a large number of channels are active.

The cost of collecting the channel initiator SMF data is small. Typically the increase in CPU usage is under a few percent, and often within measurement error.

Before you use this function you need to work with your z/OS systems programmer to ensure that SMF has the capacity for the additional records, and that they change their processes for extracting SMF records to include the new SMF data.

For channel initiator statistics data, the SMF record type is 115 and sub-type 231.

For channel initiator accounting data, the SMF record type is 116 and sub-type 10.

You can write your own programs to process this data, or use the SupportPac MP1B that contains a program, MQSMF, for printing the data, and creating data in Comma Separated Values (CSV) format suitable for importing into a spread sheet.

If you are experiencing issues with capturing channel initiator SMF data, see Dealing with issues when capturing SMF data for the channel initiator (CHINIT) for further information.

Related tasks

Interpreting IBM MQ performance statistics

Troubleshooting channel accounting data

z/OS Planning your z/OS TCP/IP environment

To get the best throughput through your network, you must use TCP/IP send and receive buffers with a size of 64 KB, or greater. With this size, the system optimizes its buffer sizes.

See What is Dynamic Right Sizing for High Latency Networks? for more information.

You can check your system buffer size by using the following Netstat command, for example:

```
TSO NETSTAT ALL (CLIENT csq1CHIN
```

The results display much information, including the following two values:

```
ReceiveBufferSize: 0000065536
SendBufferSize: 0000065536
```

65536 is 64 KB. If your buffer sizes are less than 65536, you must work with your network team to increase the **TCPSENDBFRSIZE** and **TCPRCVBUFRSIZE** values in the `PROFILE DDName` in the TCPIP procedure. For example, you might use the following command:

```
TCPCONFIG TCPSENDBFRSZE 65536 TCPRCVBUFRSIZE 65536
```

If you are unable to change your system-wide **TCPSENDBFRSIZE** or **TCPRCVBUFRSIZE** settings, contact your IBM Software Support center.

z/OS Planning your queue sharing group (QSG)

The easiest way to implement a shared queuing environment, is to configure a queue manager, add that queue manager to a QSG, then add other queue managers to the QSG.

A queue sharing group uses Db2 tables to store configuration information. There is one set of tables used by all QSGs that share the same Db2 data sharing group.

Shared queue messages are stored in a structure in a coupling facility (CF). Each QSG has its own set of CF structures. You need to configure the structures to meet your needs.

Messages over 63KB in size cannot be stored in the CF. You need to use either Shared Message Data Sets (SMDS) or Db2 for these messages.

Message profiles and capacity planning

You should understand the message profile of your shared queue messages. The following are examples of factors that you need to consider:

- Average, and maximum message size
- The typical queue depth, and exception queue depth. For example, you might need to have enough capacity to hold messages for a whole day, and the typical queue depth is under 100 messages.

If the message profile changes, you can increase the size of the structures, or implement SMDS, at a later date.

If you want to be able to handle a large peak volume of messages, you can configure IBM MQ to offload messages to SMDS when the usage of the structure reaches user specified thresholds.

You need to decide if you want to duplex the CF structures. This is controlled by the CF structure definition in the CFRM policy:

1. A duplexed structure uses two coupling facilities. If there is a problem with one CF, there is no interruption to the service, and the structure can be rebuilt on a third CF, if one is available. Duplexed structures can significantly impact the performance of operations on shared queues.
2. If the structure is not duplexed, then a problem with the CF means that shared queues on structures in that CF will become unavailable until the structure can be rebuilt in another CF.

   IBM MQ can be configured to automatically rebuild structures in another CF in this case. Persistent messages will be recovered from the logs of the queue managers.

Note that it is easy to change the CF definitions.

You can define a structure so that it can hold nonpersistent messages only, or so that it can hold persistent and nonpersistent messages.

Structures that can hold persistent messages need to be backed up periodically. Back up your CF structures at least every hour to minimize the time needed to recover the structure in the event of a failure. The backup is stored in the log data set of the queue manager performing the backup.

If you are expecting to have a high throughput of messages on your shared queues, it is best practice to have a dedicated queue manager for backing up the CF structures. This reduces the time needed to recover the structures, as a less data needs to be read from queue manager logs.

Channels

To provide a single system image for applications connecting into an IBM MQ QSG, you can define shared input channels. If these are set up, then a connection coming into the queue sharing group environment, can go to any queue manager in the QSG.

You might need to set up a network router, or Virtual IP address (VIPA) for these channels.

You can define shared output channels. A shared output channel instance can be started from any queue manager in the QSG.

See Shared channels for more information.

Security

You protect IBM MQ resources using an external security manager. If you are using RACF®, the RACF profiles are prefixed with the queue manager name. For example, a queue named APPLICATION.INPUT would be protected using a profile in the MQQUEUE class named `qmgrName.APPLICATION.INPUT.`

When using a queue sharing group you can continue to protect resources with profiles prefixed with the queue manager name, or you can prefix profiles with the queue sharing group name. For example `qsgName.APPLICATION.INPUT`.

You should aim to use profiles prefix with the queue sharing group name because this means there is a single definition for all queue managers, saving you work, and preventing a mismatch in definitions between queue managers.

Related concepts

"Planning for your queue manager" on page 120
When you are setting up a queue manager, your planning should allow for the queue manager to grow, so that the queue manager meets the needs of your enterprise.

z/OS Planning your coupling facility and offload storage environment

Use this topic when planning the initial sizes, and formats of your coupling facility (CF) structures, and shared message data set (SMDS) environment or Db2 environment.

This section contains information about the following topics:

Defining coupling facility resources

If you intend to use shared queues, you must define the coupling facility structures that IBM MQ will use in your CFRM policy. To do this you must first update your CFRM policy with information about the structures, and then activate the policy.

Your installation probably has an existing CFRM policy that describes the coupling facilities available. The Administrative data utility is used to modify the contents of the policy based on textual statements you provide. You must add statements to the policy that defines the names of the new structures, the coupling facilities that they are defined in, and what size the structures are.

The CFRM policy also determines whether IBM MQ structures are duplexed and how they are reallocated in failure scenarios. Shared queue recovery contains recommendations for configuring CFRM for resilience to failures that affect the coupling facility.

Deciding your offload storage environment

The message data for shared queues can be offloaded from the coupling facility and stored in either a Db2 table or in an IBM MQ managed data set called a *shared message data set* (SMDS). Messages which are too large to store in the coupling facility (that is, larger than 63 KB) must always be offloaded, and smaller messages can optionally be offloaded to reduce coupling facility space usage.

For more information, see Specifying offload options for shared messages.

Planning your structures

A queue sharing group (QSG) requires a minimum of two structures to be defined. The first structure, known as the administrative structure, is used to coordinate IBM MQ internal activity across the queue sharing group. No user data is held in this structure. It has a fixed name of *qsg-name*CSQ_ADMIN (where *qsg-name* is the name of your queue sharing group). Subsequent structures are known as application

structures, and are used to hold the messages on IBM MQ shared queues. Each structure can hold up to 512 shared queues.

An application structure named *qsg-name*CSQSYSAPPL is used for system queues. Defining this structure is optional, but it is required in order to use certain features. By default, the SYSTEM.QSG.CHANNEL.SYNCQ and SYSTEM.QSG.UR.RESOLUTION.QUEUE queues are defined on the *qsg-name*CSQSYAPPL structure.

Using multiple structures

A queue sharing group can connect to up to 64 coupling facility structures. One of these structures must be the administration structure. If it is defined, another of these structures might be the *qsg-name*CSQSYSAPPL structure. You can use up to 63 (62 if *qsg-name*CSQSYSAPPL is defined) structures for message data. You might choose to use multiple application structures for any of the following reasons:

- You have some queues that are likely to hold a large number of messages and so require all the resources of an entire coupling facility.
- You have a requirement for a large number of shared queues, so they must be split across multiple structures because each structure can contain only 512 queues.
- RMF reports on the usage characteristic of a structure suggest that you should distribute the queues it contains across a number of coupling facilities.
- You want some queue data to held in a physically different coupling facility from other queue data for data isolation reasons.
- Recovery of persistent shared messages is performed using structure level attributes and commands, for example BACKUP CFSTRUCT. To simplify backup and recovery, you could assign queues that hold nonpersistent messages to different structures from those structures that hold persistent messages.

When choosing which coupling facilities to allocate the structures in, consider the following points:

- Your data isolation requirements.
- The volatility of the coupling facility (that is, its ability to preserve data through a power outage).
- Failure independence between the accessing systems and the coupling facility, or between coupling facilities.
- The level of coupling facility control code (CFCC) installed on the coupling facility (IBM MQ requires Level 9 or higher).

Planning the size of your structures

The administrative structure

The administrative structure (*qsg-name*CSQ_ADMIN) must be large enough to contain 1000 list entries for each queue manager in the queue sharing group. When a queue manager starts, the structure is checked to see if it is large enough for the number of queue managers currently *defined* to the queue sharing group. Queue managers are considered as being defined to the queue sharing group if they have been added by the CSQ5PQSG utility. You can check which queue managers are defined to the group with the MQSC DISPLAY GROUP command.

Note: When calculating the size of the structure, you should allow for the size of large units of work, in addition to the number of queue managers in the queue sharing group.

Table 22 on page 156 shows the minimum required size for the administrative structure for various numbers of queue managers defined in the queue sharing group. These sizes were established for a CFCC level 14 coupling facility structure; for higher levels of CFCC, they probably need to be larger.

Table 22. Minimum administrative structure sizes

Number of queue managers defined in queue sharing group	Required storage
1	6144 KB
2	6912 KB
3	7976 KB
4	8704 KB
5	9728 KB
6	10496 KB
7	11520 KB
8	12288 KB
9	13056 KB
10	14080 KB
11	14848 KB
12	15616 KB
13	16640 KB
14	17408 KB
15	18176 KB
16	19200 KB
17	19968 KB
18	20736 KB
19	21760 KB
20	22528 KB
21	23296 KB
22	24320 KB
23	25088 KB
24	25856 KB
25	27136 KB
26	27904 KB
27	28672 KB
28	29696 KB
29	30464 KB
30	31232 KB
31	32256 KB

When you add a queue manager to an existing queue sharing group, the storage requirement might have increased beyond the size recommended in Table 22 on page 156. If so, use the following procedure to estimate the required storage for the *qsg-name*CSQ_ADMIN structure:

1. Issue MQSC command **DISPLAY CFSTATUS(CSQ_ADMIN)** on an existing member of the queue sharing group.
2. Extract the ENTSMAX information for the CSQ_ADMIN structure.
3. If this number is less than 1000 times the total number of queue managers you want to define in the queue sharing group, increase the structure size.

Application structures

The size of the application structures required to hold IBM MQ messages depends on the likely number and size of the messages to be held on a structure concurrently.

The graph in Figure 42 on page 157 shows how large you should make your CF structures to hold the messages on your shared queues. To calculate the allocation size you need the following information:

- The average size of messages on your queues.
- The total number of messages likely to be stored in the structure.

Find the number of messages along the horizontal axis. Select the curve that corresponds to your message size and determine the required value from the vertical axis. For example, for 200 000 messages of length 1 KB gives a value in the range 256 through 512 MB.

Table 23 on page 158 provides the same information in tabular form.

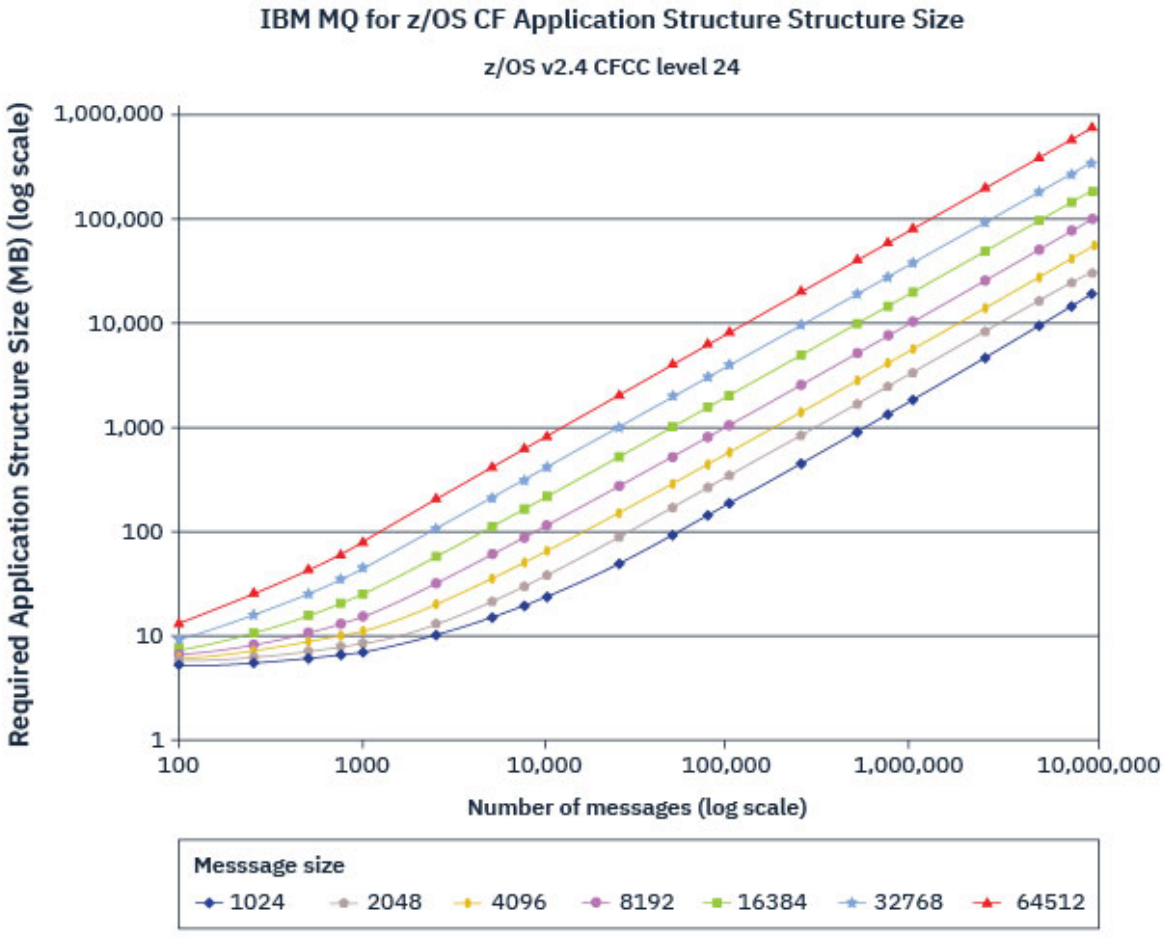


Figure 42. Calculating the size of a coupling facility structure

Use this table to help calculate how large to make your coupling facility structures:

Table 23. Calculating the size of a coupling facility structure							
Number of messages	**1 KB**	**2 KB**	**4 KB**	**8 KB**	**16 KB**	**32 KB**	**63 KB**
100	6 MB	6 MB	7 MB	7 MB	8 MB	10 MB	14 MB
1000	8 MB	9 MB	12 MB	17 MB	27 MB	48 MB	88 MB
10000	25 MB	38 MB	64 MB	115 MB	218 MB	423 MB	821 MB
100000	199 MB	327 MB	584 MB	1097 MB	2124 MB	4177 MB	8156 MB

Your CFRM policy should include the following statements:

- INITSIZE is the size in KB that the structure is allocated with when the first queue manager connects to it.
- SIZE is the maximum size that the structure can attain.
- FULLTHRESHOLD sets the percentage value of the threshold at which z/OS issues message IXC585E to indicate that the structure is getting full.

A best practice is to ensure that INITSIZE and SIZE are within a factor of 2. For example, with the figures determined previously, you might include the following statements:

```
STRUCTURE NAME(structure-name)
INITSIZE(value from graph in KB, that is, multiplied by 1024)
SIZE(something larger)
FULLTHRESHOLD(85)
```

```
STRUCTURE NAME(QSG1APPLICATION1)
INITSIZE(262144) /* 256 MB */
SIZE(524288)     /* 512 MB */
FULLTHRESHOLD(85)
```

If the structure use reaches the threshold where warning messages are issued, intervention is required. You might use IBM MQ to inhibit MQPUT operations to some of the queues in the structure to prevent applications from writing more messages, start more applications to get messages from the queues, or quiesce some of the applications that are putting messages to the queue.

Alternatively, you can use z/OS facilities to alter the structure size in place. The following z/OS command:

```
SETXCF START,ALTER,STRNAME=structure-name,SIZE=newsize
```

alters the size of the structure to *newsize*, where *newsize* is a value that is less than the value of SIZE specified on the CFRM policy for the structure, but greater than the current coupling facility size.

You can monitor the use of a coupling facility structure with the MQSC DISPLAY CFSTATUS command.

If no action is taken and a queue structure fills up, an MQRC_STORAGE_MEDIUM_FULL return code is returned to the application. If the administration structure becomes full, the exact symptoms depend on which processes experience the error, but they might include the following problems:

- No responses to commands.
- Queue manager failure as a result of problems during commit processing.

The CSQSYSAPPL structure

The *qsg-name*CSQSYSAPPL structure is an application structure for system queues. Table 3 demonstrates an example of how to estimate the message data sizes for the default queues defined on the *qsg-name*CSQSYSAPPL structure.

Table 24. Table showing CSQSYSAPPL usage against sizing.	
***qsg-name*CSQSYSAPPL usage**	**Sizing**
SYSTEM.QSG.CHANNEL.SYNCQ	2 messages of 500 bytes per active instance of a shared channel
SYSTEM.QSG.UR.RESOLUTION.QUEUE	1000 messages of 2 KB

The suggested initial structure definition values are as follows:

```
STRUCTURE NAME(qsg-nameCSQSYSAPPL)
INITSIZE(20480)        /* 20 MB */
SIZE(30720)            /* 30 MB */
FULLTHRESHOLD(85)
```

These values can be adjusted depending on your use of shared channels and group units of recovery.

Mapping shared queues to structures

To define an application structure to IBM MQ, use the DEFINE CFSTRUCT command. When you define a structure to IBM MQ, do not include the QSG name prefix in the structure name. For example, to define an application structure to IBM MQ that has the name *qsg-name*APPLICATION1 in the CFRM policy, issue the following command:

```
DEFINE CFSTRUCT(APPLICATION1)
```

The CFSTRUCT attribute of the queue definition is used to map the queue to a structure. Specify the name of the CF structure without the QSG name prefix in this attribute. For example, the following command defines a shared queue on the APPLICATION1 structure:

```
DEFINE QLOCAL(myqueue) QSGDISP(SHARED) CFSTRUCT(APPLICATION1)
```

z/OS *Planning your shared message data set (SMDS) environment*

If you are using queue sharing groups with SMDS offloading, IBM MQ needs to connect to a group of shared message data sets. Use this topic to help understand the data set requirements, and configuration required to store IBM MQ message data.

A *shared message data set* (described by the keyword SMDS) is a data set used by a queue manager to store offloaded message data for shared messages stored in a coupling facility structure.

Note: When defining SMDS data sets for a structure, you must have one for each queue manager.

When this form of data offloading is enabled, the **CFSTRUCT** requires an associated group of shared message data sets, one data set for each queue manager in the queue sharing group. The group of shared message data sets is defined to IBM MQ using the **DSGROUP** parameter on the **CFSTRUCT** definition. Additional parameters can be used to supply further optional information, such as the number of buffers to use and expansion attributes for the data sets.

Each queue manager can write to the data set which it owns, to store shared message data for messages written through that queue manager, and can read all of the data sets in the group.

A list describing the status and attributes for each data set associated with the structure is maintained internally as part of the **CFSTRUCT** definition, so each queue manager can check the definition to find out which data sets are currently available.

This data set information can be displayed using the **DISPLAY CFSTATUS TYPE(SMDS)** command to display current status and availability, and the **DISPLAY SMDS** command to display the parameter settings for the data sets associated with a specified **CFSTRUCT**.

Individual shared message data sets are effectively identified by the combination of the owning queue manager name (usually specified using the **SMDS** keyword) and the **CFSTRUCT** structure name.

This section describes the following topics:

- The DSGROUP parameter
- The DSBLOCK parameter
- Shared message data set characteristics
- Shared message data set space management
- Access to shared message data sets
- Creating a shared message data set
- Shared message data set performance and capacity considerations
- Activating a shared message data set

See DEFINE CFSTRUCT for details of these parameters.

For information on managing your shared message data sets, see Managing shared message data sets for further details.

The DSGROUP parameter

The **DSGROUP** parameter on the **CFSTRUCT** definition identifies the group of data sets in which large messages for that structure are to be stored. Additional parameters may be used to specify the logical block size to be used for space allocation purposes and values for the buffer pool size and automatic data set expansion options.

The **DSGROUP** parameter must be set up before offloading to data sets can be enabled.

- If a new **CFSTRUCT** is being defined at **CFLEVEL(5)** and the option **OFFLOAD(SMDS)** is specified or assumed, then the **DSGROUP** parameter must be specified on the same command.
- If an existing **CFSTRUCT** is being altered to increase the **CFLEVEL** to **CFLEVEL(5)** and the option **OFFLOAD(SMDS)** is specified or assumed, then the **DSGROUP** parameter must be specified on the same command if it is not already set.

The DSBLOCK parameter

Space within each data set is allocated to queues as logical blocks of a fixed size (usually 256 KB) specified using the **DSBLOCK** parameter on the **CFSTRUCT** definition, then allocated to individual messages as ranges of pages of 4 KB (corresponding to the physical block size and control interval size) within each logical block. The logical block size also determines the maximum amount of message data that can be read or written in a single I/O operation, which is the same as the buffer size for the SMDS buffer pool.

A larger value of the **DSBLOCK** parameter can improve performance for very large messages by reducing the number of separate I/O operations. However, a smaller value decreases the amount of buffer storage required for each active request. The default value for the **DSBLOCK** parameter is 256 KB, which provides a reasonable balance between these requirements, so specifying this parameter might not normally be necessary.

Shared message data set characteristics

A shared message data set is defined as a VSAM linear data set (LDS). Each offloaded message is stored in one or more blocks in the data set. The stored data is addressed directly by information in the coupling facility entries, like an extended form of virtual storage. There is no separate index or similar control information stored in the data set itself.

The direct addressing scheme means that for messages which fit into one block, only a single I/O operation is needed to read or write the block. When a message spans more than one block, the I/O

operations for each block can be fully overlapped to minimize elapsed time, provided that sufficient buffers are available.

The shared message data set also contains a small amount of general control information, consisting of a header in the first page, which includes recovery and restart status information, and a space map checkpoint area which is used to save the free block space map at queue manager normal termination.

Shared message data set space management

As background information for capacity, performance and operational considerations, it might be useful to understand the concepts of how space in shared message data sets is managed by the queue managers.

Free space in each shared message data set is tracked by its owning queue manager using a space map which indicates the number of pages in use within each logical block. The space map is maintained in main storage while the data set is open and saved in the data set when it is closed normally. (In recovery situations the space map is automatically rebuilt by scanning the messages in the coupling facility structure to find out which data set pages are currently in use).

When a shared message with offloaded message data is being written, the queue manager allocates a range of pages for each message block. If there is a partly used current logical block for the specified queue, the queue manager allocates space starting at the next free page in that block, otherwise it allocates a new logical block. If the whole message does not fit within the current logical block, the queue manager splits the message data at the end of the logical block and allocates a new logical block for the next message block. This is repeated until space has been allocated for the whole message. Any unused space in the last logical block is saved as the new current logical block for the queue. When the data set is closed normally, any unused pages in current logical blocks are returned to the space map before it is saved.

When a shared message with offloaded message data has been read and is ready to be deleted, the queue manager processes the delete request by transferring the coupling facility entry for the message to a clean-up list monitored by the owning queue manager (which may be the same queue manager). When entries arrive on this list, the owning queue manager reads and deletes the entries and returns the freed ranges of pages to the space map. When all used pages in a logical block have been freed the block becomes available for reuse.

Access to shared message data sets

Each shared message data set must be on shared direct access storage which is accessible to all queue managers in the queue sharing group.

During normal running, each queue manager opens its own shared message data set for read/write access, and opens any active shared message data sets for other queue managers for read-only access, so it can read messages stored by those queue managers. This means that each queue manager userid requires at least UPDATE access to its own shared message data set and READ access to all other shared message data sets for the structure.

If it is necessary to recover shared message data sets using **RECOVER CFSTRUCT**, the recovery process can be executed from any queue manager in the queue sharing group. A queue manager which may be used to perform recovery processing requires UPDATE access to all data sets that it may need to recover

Creating a shared message data set

Each shared message data set should normally be created before the corresponding **CFSTRUCT** definition is created or altered to enable the use of this form of message offloading, as the **CFSTRUCT** definition changes will normally take effect immediately, and the data set will be required as soon as a queue manager attempts to access a shared queue which has been assigned to that structure. A sample job to allocate and pre-format a shared message data set is provided in SCSQPROC(CSQ4SMDS). The job must be customized and run to allocate a shared message data set for each queue manager which uses a CFSTRUCT with OFFLOAD(SMDS).

If the queue manager finds that offload support has been enabled and tries to open its shared message data set but it has not yet been created, the shared message data set will be flagged as unavailable. The queue manager will then be unable to store any large messages until the data set has been created and the queue manager has been notified to try again, for example using the **START SMDSCONN** command.

A shared message data set is created as a VSAM linear data set using an Access Method Services **DEFINE CLUSTER** command. The definition must specify **SHAREOPTIONS(2 3)** to allow one queue manager to open it for write access and any number of queue managers to read it at the same time. The default control interval size of 4 KB must be used. If the data set may need to expand beyond 4 GB, it must be defined using an SMS data class which has the VSAM extended addressability attribute. A shared message data set is eligible to reside in the extended addressing space (EAS) part of an extended address volumes (EAV).

Each shared message data set can either be empty or pre-formatted to binary zeros (using **CSQJUFMT** or a similar utility such as the sample job SCSQPROC(CSQ4SMDS)), before its initial use. If it is empty or only partly formatted when it is opened, the queue manager automatically formats the remaining space to binary zeros.

Shared message data set performance and capacity considerations

Each shared message data set is used to store offloaded data for shared messages written to the associated **CFSTRUCT** by the owning queue manager, from regions within the same system. Each message that is offloaded takes up to 768 bytes of CF storage, made up of 256 bytes for the entry and 512 bytes for the two elements of header and descriptor. Each offloaded message is stored in one or more pages (physical blocks of size 4 KB) in the data set.

The data set space required for a given number of offloaded messages can therefore be estimated by rounding up the overall message size (including the descriptor) to the next multiple of 4 KB and then multiplying by the number of messages.

As for a page set, when a shared message data set is almost full, it can optionally be expanded automatically. The default behavior for this automatic expansion can be set using the **DSEXPAND** parameter on the **CFSTRUCT** definition. This setting can be overridden for each queue manager using the **DSEXPAND** parameter on the **ALTER SMDS** command. Automatic expansion is triggered when the data set reaches 90% full and more space is required. If expansion is allowed but an expansion attempt is rejected by VSAM because no secondary space allocation was specified when the data set was defined, expansion is retried using a secondary allocation of 20% of the current size of the data set.

Provided that the shared message data set is defined with the extended addressability attribute, the maximum size is only limited by VSAM considerations to a maximum of 16 TB or 59 volumes. This is significantly larger than the 64 GB maximum size of a local page set.

Activating a shared message data set

When a queue manager has successfully connected to an application coupling facility structure, it checks whether that structure definition specifies offloading using an associated **DSGROUP** parameter. If so, the queue manager allocates and opens its own shared message data set for write access, then it opens for read access any existing shared message data sets owned by other queue managers.

When a shared message data set is opened for the first time (before it has been recorded as active within the queue sharing group), the first page will not yet contain a valid header. The queue manager fills in header information to identify the queue sharing group, the structure name and the owning queue manager.

After the header has been completed, the queue manager registers the new shared message data set as active and broadcasts an event to notify any other active queue managers about the new data set.

Every time a queue manager opens a shared message data set it validates the header information to ensure that the correct data set is still being used and that it has not been damaged.

z/OS *Planning your Db2 environment*

If you are using queue sharing groups, IBM MQ needs to attach to a Db2 subsystem that is a member of a data sharing group. Use this topic to help understand the Db2 requirements used to hold IBM MQ data.

IBM MQ needs to know the name of the data sharing group that it is to connect to, and the name of a Db2 subsystem (or Db2 group) to connect to, to reach this data sharing group. These names are specified in the QSGDATA parameter of the CSQ6SYSP system parameter macro (described in Using CSQ6SYSP).

Within the data sharing group, shared Db2 tables are used to hold:

- Configuration information for the queue sharing group.
- Properties of IBM MQ shared and group objects.
- Optionally, data relating to offloaded IBM MQ messages.

IBM MQ provides a single set of sample jobs for defining the necessary Db2 table spaces, tables, and indexes. These jobs make use of Universal Table Spaces (UTS). Earlier versions of the product had two sets of jobs, one for UTS, and one for older types of table space, which have been deprecated by the most recent versions of Db2.

IBM MQ can still be used with older types of table space, and this might be appropriate if you already have an existing queue sharing group. However, if you are creating a new queue sharing group, it should use UTS.

Db2 V12 Function level 508 provides a non disruptive migration process for migrating multi-table table spaces to universal table spaces. You can use this approach to migrate the multi-table table spaces, used by existing queue sharing groups, to universal table spaces without taking an outage of the whole queue sharing group.

In Db2 V13, use the MOVE TABLE option of the ALTER TABLESPACE statement. See Moving tables from multi-table table spaces to partition-by-growth table spaces for more information.

By default Db2 uses the user ID of the person running the jobs as the owner of the Db2 resources. If this user ID is deleted then the resources associated with it are deleted, and so the table is deleted. Consider using a group ID to own the tables, rather than an individual user ID. You can do this by adding `GROUP=groupname` onto the JOB card, and specifying `SET CURRENT SQLID='groupname'` before any SQL statements.

IBM MQ uses the RRS Attach facility of Db2. This means that you can specify the name of a Db2 group that you want to connect to. The advantage of connecting to a Db2 group attach name (rather than a specific Db2 subsystem), is that IBM MQ can connect (or reconnect) to any available Db2 subsystem on the z/OS image that is a member of that group. There must be a Db2 subsystem that is a member of the data sharing group active on each z/OS image where you are going to run a queue-sharing IBM MQ subsystem, and RRS must be active.

Db2 storage

For most installations, the amount of Db2 storage required is about 20 or 30 cylinders on a 3390 device. However, if you want to calculate your storage requirement, the following table gives some information to help you determine how much storage Db2 requires for the IBM MQ data. The table describes the length of each Db2 row, and when each row is added to or deleted from the relevant Db2 table. Use this information together with the information about calculating the space requirements for the Db2 tables and their indexes in the *Db2 for z/OS Installation Guide*.

Table 25. Planning your Db2 storage requirements

Db2 table name	Length of row	A row is added when:	A row is deleted when:
CSQ.ADMIN_B_QSG	252 bytes	A queue sharing group is added to the table with the ADD QSG function of the CSQ5PQSG utility.	A queue sharing group is removed from the table with the REMOVE QSG function of the CSQ5PQSG utility. (All rows relating to this queue sharing group are deleted automatically from all the other Db2 tables when the queue sharing group record is deleted.)
CSQ.ADMIN_B_QMGR	Up to 3828 bytes	A queue manager is added to the table with the ADD QMGR function of the CSQ5PQSG utility.	A queue manager is removed from the table with the REMOVE QMGR function of the CSQ5PQSG utility.
CSQ.ADMIN_B_STRUCTURE	1454 bytes	The first local queue definition, specifying the QSGDISP(SHARED) attribute, that names a previously unknown structure within the queue sharing group is defined.	The last local queue definition, specifying the QSGDISP(SHARED) attribute, that names a structure within the queue sharing group is deleted.
CSQ.ADMIN_B_SCST	342 bytes	A shared channel is started.	A shared channel becomes inactive.
CSQ.ADMIN_B_SSKT	254 bytes	A shared channel that has the NPMSPEED(NORMAL) attribute is started.	A shared channel that has the NPMSPEED(NORMAL) attribute becomes inactive.
CSQ.ADMIN_B_STRBACKUP	514 bytes	A new row is added to the CSQ.ADMIN_B_STRUCTURE table. Each entry is a dummy entry until the BACKUP CFSTRUCT command is run, which overwrites the dummy entries.	A row is deleted from the CSQ.ADMIN_B_STRUCTURE table.
CSQ.OBJ_B_AUTHINFO	3400 bytes	An authentication information object with QSGDISP(GROUP) is defined.	An authentication information object with QSGDISP(GROUP) is deleted.
CSQ.OBJ_B_QUEUE	Up to 3707 bytes	• A queue with the QSGDISP(GROUP) attribute is defined. • A queue with the QSGDISP(SHARED) attribute is defined. • A model queue with the DEFTYPE(SHAREDYN) attribute is opened.	• A queue with the QSGDISP(GROUP) attribute is deleted. • A queue with the QSGDISP(SHARED) attribute is deleted. • A dynamic queue with the DEFTYPE(SHAREDYN) attribute is closed with the DELETE option.
CSQ.OBJ_B_NAMELIST	Up to 15127 bytes	A namelist with the QSGDISP(GROUP) attribute is defined.	A namelist with the QSGDISP(GROUP) attribute is deleted.

Table 25. Planning your Db2 storage requirements (continued)			
Db2 table name	**Length of row**	**A row is added when:**	**A row is deleted when:**
CSQ.OBJ_B_CHANNEL	Up to 14127 bytes	A channel with the QSGDISP(GROUP) attribute is defined.	A channel with the QSGDISP(GROUP) attribute is deleted.
CSQ.OBJ_B_STGCLASS	Up to 2865 bytes	A storage class with the QSGDISP(GROUP) attribute is defined.	A storage class with the QSGDISP(GROUP) attribute class is deleted.
CSQ.OBJ_B_PROCESS	Up to 3347 bytes	A process with the QSGDISP(GROUP) attribute is defined.	A process with the QSGDISP(GROUP) attribute is deleted.
CSQ.OBJ_B_TOPIC	Up to 14520 bytes	A topic object with QSGDISP(GROUP) attribute is defined.	A topic object with QSGDISP(GROUP) attribute is deleted.
CSQ.EXTEND_B_QMGR	Less than 430 bytes	A queue manager is added to the table with the ADD QMGR function of the CSQ5PQSG utility.	A queue manager is removed from the table with the REMOVE QMGR function of the CSQ5PQSG utility.
CSQ.ADMIN_B_MESSAGES	87 bytes	For large message PUT (1 per BLOB).	For large message GET (1 per BLOB).
CSQ.ADMIN_MSGS_BAUX1 CSQ.ADMIN_MSGS_BAUX2 CSQ.ADMIN_MSGS_BAUX3 CSQ.ADMIN_MSGS_BAUX4		These 4 tables contain message payload for large messages added into one of these 4 tables for each BLOB of the message. BLOBS are up to 511 KB in length, so if the message size is > 711 KB, there will be multiple BLOBs for this message.	

The use of large numbers of shared queue messages of size greater than 63 KB can have significant performance implications on your IBM MQ system. For more information, see SupportPac MP16, Capacity Planning and Tuning for IBM MQ for z/OS, at: SupportPacs for IBM MQ and other project areas.

z/OS Planning for backup and recovery

Developing backup and recovery procedures at your site is vital to avoid costly and time-consuming losses of data. IBM MQ provides means for recovering both queues and messages to their current state after a system failure.

This topic contains the following sections:

- “Example of queue manager backup activity” on page 172

Recovery procedures

Develop the following procedures for IBM MQ:

- Creating a point of recovery.
- Backing up page sets.
- Backing up CF structures.
- Recovering page sets.
- Recovering from out-of-space conditions (IBM MQ logs and page sets).
- Recovering CF structures.

See 管理 IBM MQ for z/OS for information about these.

Become familiar with the procedures used at your site for the following:

- Recovering from a hardware or power failure.
- Recovering from a z/OS component failure.
- Recovering from a site interruption, using off-site recovery.

z/OS Tips for backup and recovery

Use this topic to understand some backup and recovery tasks.

The queue manager restart process recovers your data to a consistent state by applying log information to the page sets. If your page sets are damaged or unavailable, you can resolve the problem using your backup copies of your page sets (if all the logs are available). If your log data sets are damaged or unavailable, it might not be possible to recover completely.

Consider the following points:

- Periodically take backup copies
- Do not discard archive logs you might need
- Do not change the DDname to page set association

Periodically take backup copies

A *point of recovery* is the term used to describe a set of backup copies of IBM MQ page sets and the corresponding log data sets required to recover these page sets. These backup copies provide a potential restart point in the event of page set loss (for example, page set I/O error). If you restart the queue manager using these backup copies, the data in IBM MQ is consistent up to the point that these copies were taken. Provided that all logs are available from this point, IBM MQ can be recovered to the point of failure.

The more recent your backup copies, the quicker IBM MQ can recover the data in the page sets. The recovery of the page sets is dependent on all the necessary log data sets being available.

In planning for recovery, you need to determine how often to take backup copies and how many complete backup cycles to keep. These values tell you how long you must keep your log data sets and backup copies of page sets for IBM MQ recovery.

When deciding how often to take backup copies, consider the time needed to recover a page set. The time needed is determined by the following:

- The amount of log to traverse.
- The time it takes an operator to mount and remove archive tape volumes.

- The time it takes to read the part of the log needed for recovery.
- The time needed to reprocess changed pages.
- The storage medium used for the backup copies.
- The method used to make and restore backup copies.

In general, the more frequently you make backup copies, the less time recovery takes, but the more time is spent making copies.

For each queue manager, you should take backup copies of the following:

- The archive log data sets
- The BSDS copies created at the time of the archive
- The page sets
- Your object definitions
- Your CF structures

To reduce the risk of your backup copies being lost or damaged, consider:

- Storing the backup copies on different storage volumes to the original copies.
- Storing the backup copies at a different site to the original copies.
- Making at least two copies of each backup of your page sets and, if you are using single logging or a single BSDS, two copies of your archive logs and BSDS. If you are using dual logging or BSDS, make a single copy of both archive logs or BSDS.

Before moving IBM MQ to a production environment, fully test and document your backup procedures.

Backing up your page sets

You need to back up page sets regularly. Some enterprises back up the page sets twice a day.

You need the active and archive logs since a backup to be able to recover using the backup. You need enough log data to go back four checkpoints if the backup was taken when the queue manager was running.

You can use ADRDSSU FastReplication to back up page sets, and you can do this while the queue manager is active. Note that you need to ensure there is enough space in the storage pool.

Backing up your object definitions

Create backup copies of your object definitions. To do this, use the MAKEDEF feature of the COMMAND function of the utility program (described in Using the COMMAND function of CSQUTIL).

You should do this whenever you take backup copies of your queue manager data sets, and keep the most current version.

Backing up your coupling facility structures

If you have set up any queue sharing groups, even if you are not using them, you must take periodic backups of your CF structures. To do this, use the IBM MQ BACKUP CFSTRUCT command. You can use this command only on CF structures that are defined with the RECOVER(YES) attribute. If any CF entries for persistent shared messages refer to offloaded message data stored in a shared message data set (SMDS) or Db2, the offloaded data is retrieved and backed up together with the CF entries. Shared message data sets should not be backed up separately.

It is recommended that you take a backup of all your CF structures about every hour, to minimize the time it takes to restore a CF structure.

You could perform all your CF structure backups on a single queue manager, which has the advantage of limiting the increase in log use to a single queue manager. Alternatively, you could perform backups on all the queue managers in the queue sharing group, which has the advantage of spreading the workload across the queue sharing group. Whichever strategy you use, IBM MQ can locate the backup

and perform a RECOVER CFSTRUCT from any queue manager in the queue sharing group. The logs of all the queue managers in the queue sharing group need to be accessed to recover the CF structure.

Backing up your message security policies

If you are using Advanced Message Security to create a backup of your message security policies, create a backup using the message security policy utility (CSQ0UTIL) to run **dspmqspl** with the -export parameter, then save the policy definitions that are output to the EXPORT DD.

You should create a backup of your message security policies whenever you take backup copies of your queue manager data sets, and keep the most current version.

Do not discard archive logs you might need

IBM MQ might need to use archive logs during restart. You must keep sufficient archive logs so that the system can be fully restored. IBM MQ might use an archive log to recover a page set from a restored backup copy. If you have discarded that archive log, IBM MQ cannot restore the page set to its current state. When and how you discard archive logs is described in Discarding archive log data sets.

You can use the `/cpf DIS USAGE TYPE(ALL)` command to display the log RBA, and log range sequence number (LRSN) that you need to recover your queue manager's page sets and the queue sharing group's structures. You should then use the print log map utility (CSQJU004) to print bootstrap data set (BSDS) information for the queue manager to locate the logs containing the log RBA.

For CF structures, you need to run the CSQJU004 utility on each queue manager in the queue sharing group to locate the logs containing the LRSN. You need these logs and any later logs to be able to recover the page sets and structures.

Do not change the DDname to page set association

IBM MQ associates page set number 00 with DDname CSQP0000, page set number 01 with DDname CSQP0001, and so on, up to CSQP0099. IBM MQ writes recovery log records for a page set based on the DDname that the page set is associated with. For this reason, you must not move page sets that have already been associated with a PSID DDname.

z/OS Recovering page sets

Use this topic to understand the factors involved when recovering pages sets, and how to minimize restart times.

A key factor in recovery strategy concerns the time for which you can tolerate a queue manager outage. The total outage time might include the time taken to recover a page set from a backup, or to restart the queue manager after an abnormal termination. Factors affecting restart time include how frequently you back up your page sets, and how much data is written to the log between checkpoints.

To minimize the restart time after an abnormal termination, keep units of work short so that, at most, two active logs are used when the system restarts. For example, if you are designing an IBM MQ application, avoid placing an MQGET call that has a long wait interval between the first in-syncpoint MQI call and the commit point because this might result in a unit of work that has a long duration. Another common cause of long units of work is batch intervals of more than 5 minutes for the channel initiator.

You can use the DISPLAY THREAD command to display the RBA of units of work and to help resolve the old ones.

How often must you back up a page set?

Frequent page set backup is essential if a reasonably short recovery time is required. This applies even when a page set is very small or there is a small amount of activity on queues in that page set.

If you use persistent messages in a page set, the backup frequency should be in hours rather than days. This is also the case for page set zero.

To calculate an approximate backup frequency, start by determining the target total recovery time. This consists of the following:

1. The time taken to react to the problem.
2. The time taken to restore the page set backup copy.

   If you use SnapShot backup/restore, the time taken to perform this task is a few seconds. For information about SnapShot, see the *DFSMSdss Storage Administration Guide*.
3. The time the queue manager requires to restart, including the additional time needed to recover the page set.

   This depends most significantly on the amount of log data that must be read from active and archive logs since that page set was last backed up. All such log data must be read, in addition to that directly associated with the damaged page set.

   Note: When using *fuzzy backup* (where a snapshot is taken of the logs and page sets while a unit of work is active), it might be necessary to read up to three additional checkpoints, and this might result in the need to read one or more additional logs.

When deciding on how long to allow for the recovery of the page set, the factors that you need to consider are:

- The rate at which data is written to the active logs during normal processing depends on how messages arrive in your system, in addition to the message rate.

  Messages received or sent over a channel result in more data logging than messages generated and retrieved locally.
- The rate at which data can be read from the archive and active logs.

  When reading the logs, the achievable data rate depends on the devices used and the total load on your particular DASD subsystem.

  With most tape units, it is possible to achieve higher data rates for archived logs with a large block size. However, if an archive log is required for recovery, all the data on the active logs must be read also.

z/OS Recovering CF structures

Use this topic to understand the recovery process for CF structures.

At least one queue manager in the queue sharing group must be active to process a RECOVER CFSTRUCT command. CF structure recovery does not affect queue manager restart time, because recovery is performed by an already active queue manager.

The recovery process consists of two logical steps that are managed by the RECOVER CFSTRUCT command:

1. Locating and restoring the backup.
2. Merging all the logged updates to persistent messages that are held on the CF structure from the logs of all the queue managers in the queue sharing group that have used the CF structure, and applying the changes to the backup.

The second step is likely to take much longer because a lot of log data might need to be read. You can reduce the time taken if you take frequent backups, or if you recover multiple CF structures at the same time, or both.

The queue manager performing the recovery locates the relevant backups on all the other queue managers' logs using the data in Db2 and the bootstrap data sets. The queue manager replays these backups in the correct time sequence across the queue sharing group, from just before the last backup through to the point of failure.

The time it takes to recover a CF structure depends on the amount of recovery log data that must be replayed, which in turn depends on the frequency of the backups. In the worst case, it takes as long to

read a queue manager's log as it did to write it. So if, for example, you have a queue sharing group containing six queue managers, an hour's worth of log activity could take six hours to replay. In general it takes less time than this, because reading can be done in bulk, and because the different queue manager's logs can be read in parallel. As a starting point, we recommend that you back up your CF structures every hour.

All queue managers can continue working with non-shared queues and queues in other CF structures while there is a failed CF structure. If the administration structure has also failed, at least one of the queue managers in the queue sharing group must be started before you can issue the RECOVER CFSTRUCT command.

Backing up CF structures can require considerable log writing capacity, and can therefore impose a large load on the queue manager doing the backup. Choose a lightly loaded queue manager for doing backups; for busy systems, add an additional queue manager to the queue sharing group and dedicate it exclusively for doing backups.

z/OS Achieving specific recovery targets

Use this topic for guidance on how you can achieve specific recovery target times by adjusting backup frequency.

If you have specific recovery targets to achieve, for example, completion of the queue manager recovery and restart processing in addition to the normal startup time within *xx* seconds, you can use the following calculation to estimate your backup frequency (in hours):

```
Formula (A)
                    Required restart time * System recovery log read rate
                          (in secs)                  (in MB/sec)
Backup frequency  = ------------------------------------------------------
   (in hours)              Application log write rate (in MB/hour)
```

Note: The examples given next are intended to highlight the need to back up your page sets frequently. The calculations assume that most log activity is derived from a large number of persistent messages. However, there are situations where the amount of log activity is not easily calculated. For example, in a queue sharing group environment, a unit of work in which shared queues are updated in addition to other resources might result in UOW records being written to the IBM MQ log. For this reason, the Application log write rate in Formula (A) can be derived accurately only from the observed rate at which the IBM MQ logs fill.

For example, consider a system in which IBM MQ MQI clients generate a total load of 100 persistent messages a second. In this case, all messages are generated locally.

If each message is of user length 1 KB, the amount of data logged each hour is approximately:

```
100 * (1 + 1.3) KB * 3600 = approximately 800 MB

where
      100           = the message rate a second
      (1 + 1.3) KB  = the amount of data logged for
                      each 1 KB of persistent messages
```

Consider an overall target recovery time of 75 minutes. If you have allowed 15 minutes to react to the problem and restore the page set backup copy, queue manager recovery and restart must then complete within 60 minutes (3600 seconds) applying formula (A). Assuming that all required log data is on RVA2-T82 DASD, which has a recovery rate of approximately 2.7 MB a second, this necessitates a page set backup frequency of at least every:

```
3600 seconds * 2.7 MB a second / 800 MB an hour = 12.15 hours
```

If your IBM MQ application day lasts approximately 12 hours, one backup each day is appropriate. However, if the application day lasts 24 hours, two backups each day is more appropriate.

Another example might be a production system in which all the messages are for request-reply applications (that is, a persistent message is received on a receiver channel and a persistent reply message is generated and sent down a sender channel).

In this example, the achieved batch size is one, and so there is one batch for every message. If there are 50 request replies a second, the total load is 100 persistent messages a second. If each message is 1 KB in length, the amount of data logged each hour is approximately:

```
50((2 * (1+1.3) KB) + 1.4 KB + 2.5 KB) * 3600 = approximately 1500 MB

where:
 50                 = the message pair rate a second
 (2 * (1 + 1.3) KB) = the amount of data logged for each message pair
 1.4 KB             = the overhead for each batch of messages
                      received by each channel
 2.5 KB             = the overhead for each batch of messages sent
                      by each channel
```

To achieve the queue manager recovery and restart within 30 minutes (1800 seconds), again assuming that all required log data is on RVA2-T82 DASD, this requires that page set backup is carried out at least every:

```
1800 seconds * 2.7 MB a second / 1500 MB an hour = 3.24 hours
```

Periodic review of backup frequency

Monitor your IBM MQ log usage in terms of MB an hour. Periodically perform this check and amend your page set backup frequency if necessary.

z/OS Backup considerations for other products

If you are using IBM MQ with CICS or IMS then you must also consider the implications for your backup strategy with those products. The data facility hierarchical storage manager (DFHSM) manages data storage, and can interact with the storage used by IBM MQ.

Backup and recovery with DFHSM

The data facility hierarchical storage manager (DFHSM) does automatic space-availability and data-availability management among storage devices in your system. If you use it, you need to know that it moves data to and from the IBM MQ storage automatically.

DFHSM manages your DASD space efficiently by moving data sets that have not been used recently to alternative storage. It also makes your data available for recovery by automatically copying new or changed data sets to tape or DASD backup volumes. It can delete data sets, or move them to another device. Its operations occur daily, at a specified time, and allow for keeping a data set for a predetermined period before deleting or moving it.

You can also perform all DFHSM operations manually. For more information on DFHSM, see the z/OS DFSMS product documentation. If you use DFHSM with IBM MQ, note that DFHSM does the following:

- Uses cataloged data sets.
- Operates on page sets and logs.
- Supports VSAM data sets.

Recovery and CICS

The recovery of CICS resources is not affected by the presence of IBM MQ. CICS recognizes IBM MQ as a non-CICS resource (or external resource manager), and includes IBM MQ as a participant in any syncpoint coordination requests using the CICS resource manager interface (RMI). For more information about CICS recovery and the CICS resource manager interface, see the CICS product documentation.

Recovery and IMS

IMS recognizes IBM MQ as an external subsystem and as a participant in syncpoint coordination. IMS recovery for external subsystem resources is described in the IMS product documentation.

z/OS Preparing for recovery on an alternative site

If a total loss of an IBM MQ computing center, you can recover on another IBM MQ system at a recovery site.

To recover an IBM MQ system at a recovery site, you must regularly back up the page sets and the logs. As with all data recovery operations, the objectives of disaster recovery are to lose as little data, workload processing (updates), and time as possible.

At the recovery site:

- The recovery IBM MQ queue manager **must** have the same name as the lost queue manager.
- Ensure the system parameter module used on the recovery queue manager contains the same parameters as the lost queue manager.

See Administering IBM MQ for z/OS and Troubleshooting IBM MQ for z/OS problems for more information.

z/OS Example of queue manager backup activity

This topic shows as an example of queue manager backup activity.

When you plan your queue manager backup strategy, a key consideration is retention of the correct amount of log data. Managing the logs describes how to determine which log data sets are required, by reference to the system recovery RBA of the queue manager. IBM MQ determines the system recovery RBA using information about the following:

- Currently active units of work.
- Page set updates that have not yet been flushed from the buffer pools to disk.
- CF structure backups, and whether this queue manager's log contains information required in any recovery operation using them.

You must retain sufficient log data to be able to perform media recovery. While the system recovery RBA increases over time, the amount of log data that must be retained only decreases when subsequent backups are taken. CF structure backups are managed by IBM MQ, and so are taken into account when reporting the system recovery RBA. This means that in practice, the amount of log data that must be retained only reduces when page set backups are taken.

Figure 43 on page 173 shows an example of the backup activity on a queue manager that is a member of a queue sharing group, how the recovery RBA varies with each backup, and how that affects the amount of log data that must be retained. In the example the queue manager uses local and shared resources: page sets, and two CF structures, STRUCTURE1 and STRUCTURE2.

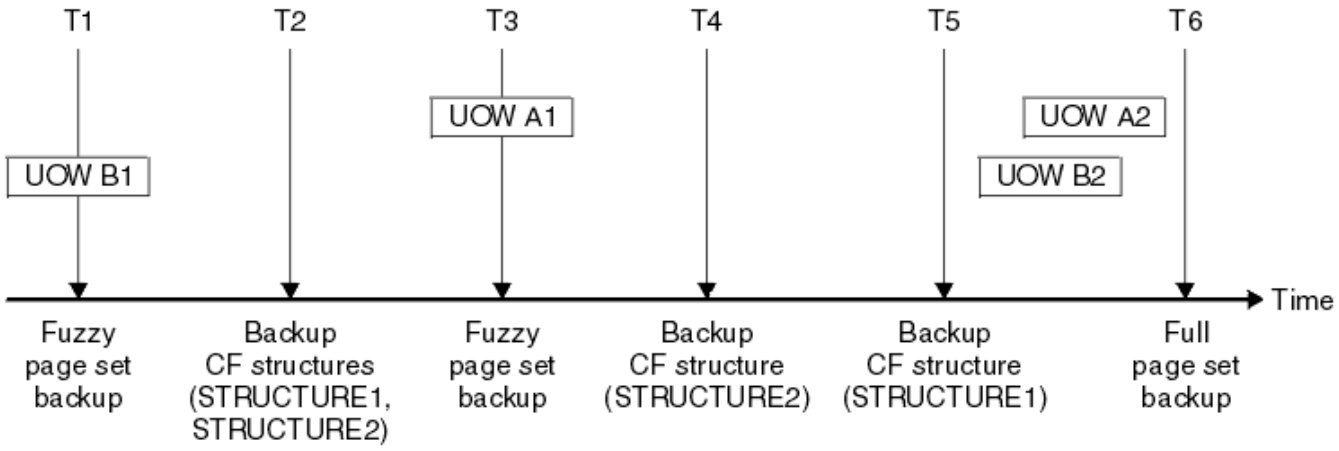


Figure 43. Example of queue manager backup activity

This is what happens at each point in time:

Point in time T1

A fuzzy backup is created of your page sets, as described in How to back up and recover page sets.

The system recovery RBA of the queue manager is the lowest of the following:

- The recovery RBAs of the page sets being backed up at this point.
- The lowest recovery RBA required to recover the CF application structures. This relates to the recovery of backups of STRUCTURE1 and STRUCTURE2 created earlier.
- The recovery RBA for the oldest currently active unit of work within the queue manager (UOWB1).

The system recovery RBA for this point in time is given by messages issued by the DISPLAY USAGE command, which is part of the fuzzy backup process.

Point in time T2

Backups of the CF structures are created. CF structure STRUCTURE1 is backed up first, followed by STRUCTURE2.

The amount of log data that must be retained is unchanged, because the same data as determined from the system recovery RBA at T1 is still required to recover using the page set backups taken at T1.

Point in time T3

Another fuzzy backup is created.

The system recovery RBA of the queue manager is the lowest of the following:

- The recovery RBAs of the page sets being backed up at this point.
- The lowest recovery RBA required to recover CF structure STRUCTURE1, because STRUCTURE1 was backed up before STRUCTURE2.
- The recovery RBA for the oldest currently active unit of work within the queue manager (UOWA1).

The system recovery RBA for this point in time is given by messages issued by the DISPLAY USAGE command, which is part of the fuzzy backup process.

You can now reduce the log data retained, as determined by this new system recovery RBA.

Point in time T4

A backup is taken of CF structure STRUCTURE2. The recovery RBA for the recovery of the oldest required CF structure backup relates to the backup of CF structure STRUCTURE1, which was backed up at time T2.

The creation of this CF structure backup has no effect on the amount of log data that must be retained.

Point in time T5

A backup is taken of CF structure STRUCTURE1. The recovery RBA for recovery of the oldest required CF structure backup now relates to recovery of CF structure STRUCTURE2, which was backed up at time T4.

The creation of this CF structure backup has no effect on amount of log data that must be retained.

Point in time T6

A full backup is taken of your page sets as described in How to back up and recover page sets.

The system recovery RBA of the queue manager is the lowest of the following:

- The recovery RBAs of the page sets being backed up at this point.
- The lowest recovery RBA required to recover the CF structures. This relates to recovery of CF structure STRUCTURE2.
- The recovery RBA for the oldest currently active unit of work within the queue manager. In this case, there are no current units of work.

The system recovery RBA for this point in time is given by messages issued by the DISPLAY USAGE command, which is part of the full backup process.

Again, the log data retained can be reduced, because the system recovery RBA associated with the full backup is more recent.

z/OS Planning your z/OS UNIX environment

Certain processes within the IBM MQ queue manager, channel initiator, and mqweb server use z/OS UNIX System Services (z/OS UNIX) for their normal processing.

The queue manager and channel initiator started task user IDs need an OMVS segment with a UID defined in order to be able to access z/OS UNIX. The user IDs require no special permissions in z/OS UNIX.

Note: Although the queue manager and channel initiator make use of z/OS UNIX facilities (for example, to interface with TCP/IP services), they do not need to access any of the content of the IBM MQ installation directory in the z/OS UNIX file system. As a result, the queue manager and channel initiator do not require any configuration to specify the path for the z/OS UNIX file system.

The mqweb server, which hosts the IBM MQ Console and REST API, makes use of files in the IBM MQ installation directory in the z/OS UNIX file system. It also needs access to another file system which is used to store data such as configuration and log files. The mqweb started task JCL needs to be customized to reference these z/OS UNIX file systems.

The content of the IBM MQ directory in the z/OS UNIX file system is also used by applications connecting to IBM MQ. For example, applications using the IBM MQ classes for Java or IBM MQ classes for JMS interfaces.

See the following topics for the relevant configuration instructions:

- Environment variables relevant to IBM MQ classes for Java
- IBM MQ classes for Java libraries
- Setting environment variables
- Configuring the Java Native Interface (JNI) libraries

z/OS Planning for Advanced Message Security

TLS (or SSL) can be used to encrypt and protect messages flowing on a network, but this does not protect messages when they are on a queue ("at rest"). Advanced Message Security (AMS) protects the messages from the time that they are first put to a queue, until they are got, so that only the intended recipients of the message can read that message. The messages are encrypted and signed during put processing, and unprotected during get processing.

AMS can be configured to protect messages in different ways:

1. A message can be signed. The message is in clear text, but there is a checksum, which is signed. This allows any changes in the message content to be detected. From the signed content, you can identify who signed the data.
2. A message can be encrypted. The contents are not visible to anyone without the decryption key. The decryption key is encrypted for each recipient.
3. A message can be encrypted and signed. They decryption key is encrypted for each recipient, and from the signing you can identify who sent the message.

The encryption and signing use digital certificates and key rings.

You can set up a client to use AMS, so the data is protected before the data is put on the client channel. Protected messages can be sent to a remote queue manager, and you need to configure the remote queue manager to process these messages.

Setting up AMS

An AMS address space is used for doing the AMS work. This has additional security set up, to give access to and protect the use of key rings and certificates.

You configure which queues are to be protected by using a utility program (CSQ0UTIL) to define the security policies for queues.

Once AMS is set up

You need to set up a digital certificate and a key ring for people who put messages, and the people who get messages.

If a user, Alice, on z/OS needs to send a message to Bob, AMS needs a copy of the public certificate for Bob.

If Bob wants to process a message from Alice, AMS needs the public certificate for Alice, or the same certificate authority certificate used by Alice.

Attention: You need to:

- Carefully plan who can put to, or get from, queues
- Identify the people and their certificate names.

It is easy to make mistakes, and problems can be hard to resolve.

Related concepts

“Planning for your queue manager” on page 120
When you are setting up a queue manager, your planning should allow for the queue manager to grow, so that the queue manager meets the needs of your enterprise.

z/OS Planning for Managed File Transfer

Use this section as guidance on how you need to set up your system to run Managed File Transfer (MFT) on z/OS.

z/OS Planning for Managed File Transfer - hardware and software requirements

Use this topic as guidance on how you need to set up hardware and software requirements on your system to run Managed File Transfer (MFT) on z/OS.

Software requirements

Managed File Transfer is written in Java, with some shell scripts and JCL to configure and operate the program.

Important: You must be familiar with z/OS UNIX System Services (z/OS UNIX) in order to configure Managed File Transfer. For example:

- The file directory structure, with names such as `/u/userID/myfile.txt`
- z/OS UNIX commands, for example:
  - `cd` (change directory)
  - `ls` (list)
  - `chmod` (change the file permissions)
  - `chown` (change file ownership or groups which can access the file or directory)

You require the following products in z/OS UNIX to be able to configure and run MFT:

1. Java, for example, in directory `/java/java80_bit64_GA/J8.0_64/`
2. IBM MQ 9.4.0, for example, in directory `/mqm/V9R3M0`
3. If you want to use Db2 for status and history, you need to install Db2 JDBC libraries, for example, in directory `/db2/db2v10/jdbc/libs`.

Product registration

At startup Managed File Transfer checks the registration in `sys1.parmlib`(IFAPRDxx) concatenation. The following code is an example of how you register MFT:

```
PRODUCT OWNER('IBM CORP')
NAME('WS MQ FILE TRANS')
ID(5655-MFT)
VERSION(*) RELEASE(*) MOD(*)
FEATURENAME('WS MQ FILE TRANS')
STATE(ENABLED)
```

Disk space

The IBM MQ for z/OS Program Directory states the DASD and zFS storage requirements for Managed File Transfer. For download links for the Program Directory for IBM MQ for z/OS, see IBM MQ 9.4 PDF files for product documentation and Program Directories.

z/OS Planning for Managed File Transfer - topologies

Use this topic as guidance on what topology you need on your system to run Managed File Transfer (MFT) on z/OS.

Managed File Transfer queue managers

IBM MQ Managed File Transfer topologies consist of:

Agents, and their associated queue managers
The agent uses system queues hosted on their agent queue manager to maintain state information and receive requests for work.

A command queue manager
This acts as a gateway into an MFT topology. It is connected to the agent queue managers through either sender and receiver channels, or clustering. When certain commands are run, they connect directly to the command queue manager, and send a message to the specified agent. This message is routed through the IBM MQ network to the agent queue manager, where it is picked up by the agent and processed.

A coordination queue manager
This is a central hub that has knowledge of the entire topology. The coordination queue manager is connected to all of the agent queue managers in a topology through either sender and receiver

channels, or using clustering. Agents regularly publish status information to the coordination queue manager, and store their transfer templates there.

It is possible for a single queue manager to perform multiple roles within a topology. For example, the same queue manager can be configured as both the coordination queue manager and the command queue manager for a topology.

If you are using multiple queue managers you need to set up channels between the queue managers. You can either do this by using clustering or by using point-to-point connections.

When using IBM MQ Managed File Transfer for z/OS, there are a number of things to consider when determining which queue managers to use for the different roles within a topology.

Agent queue managers

The agent queue manager for an IBM MQ Managed File Transfer for z/OS agent must be running on z/OS.

If:

- The agent is running Managed File Transfer for z/OS on IBM MQ 9.1 or later
- And, the agent queue manager is licensed for IBM MQ Advanced for z/OS Value Unit Edition (Advanced VUE)

the agent can connect to the queue manager using the CLIENT transport.

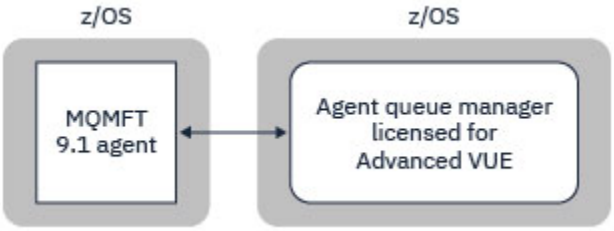


Figure 44. MFT 9.1 agents on z/OS can connect to a queue manager using the CLIENT transport, assuming the queue manager is licensed for Advanced VUE.

If:

- The agent is running Managed File Transfer for z/OS on IBM MQ 9.0 or earlier
- Or, the agent queue manager is running Managed File Transfer for z/OS on IBM MQ 9.0 or later, and the agent queue manager is licensed for either MFT, IBM MQ Advanced for z/OS, or Advanced VUE

the agent must connect to the queue manager using the BINDINGS transport.

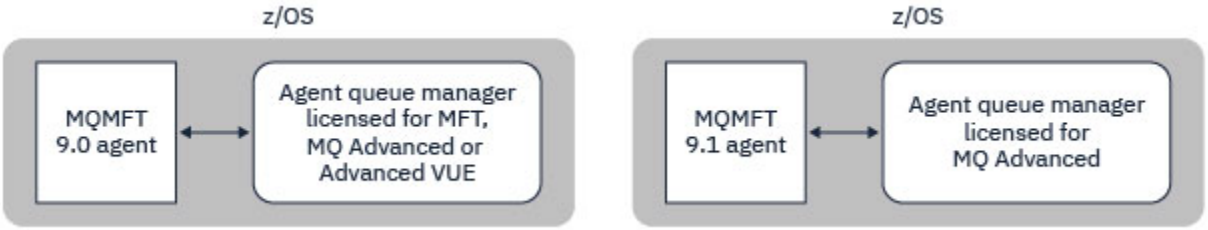


Figure 45. MFT 9.0 agents on z/OS and 9.1 agents that have an agent queue manager licensed for either MFT or IBM MQ Advanced, must connect using the BINDINGS transport.

Command queue managers

The Which MFT commands and processes connect to which queue manager topic shows all of the commands that connect to the command queue manager for a Managed File Transfer topology.

Note: When running these commands on z/OS, the command queue manager must also be on z/OS.

If the command queue manager is licensed for Advanced VUE, the commands can connect to the queue manger using the CLIENT transport. Otherwise, the commands must connect to the command queue manager using the BINDINGS transport.

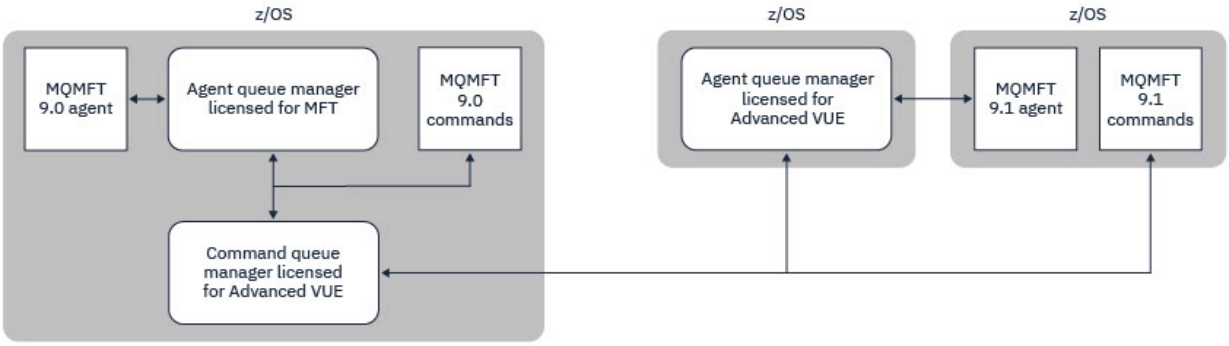


Figure 46. Commands connect to the command queue manager for an MFT topology. When running these commands on z/OS, the command queue manager must also be on z/OS

Coordination queue managers

IBM MQ Managed File Transfer for z/OS agents can be part of a topology where the coordination queue manager is either running on z/OS, or is running on a multiplatform.

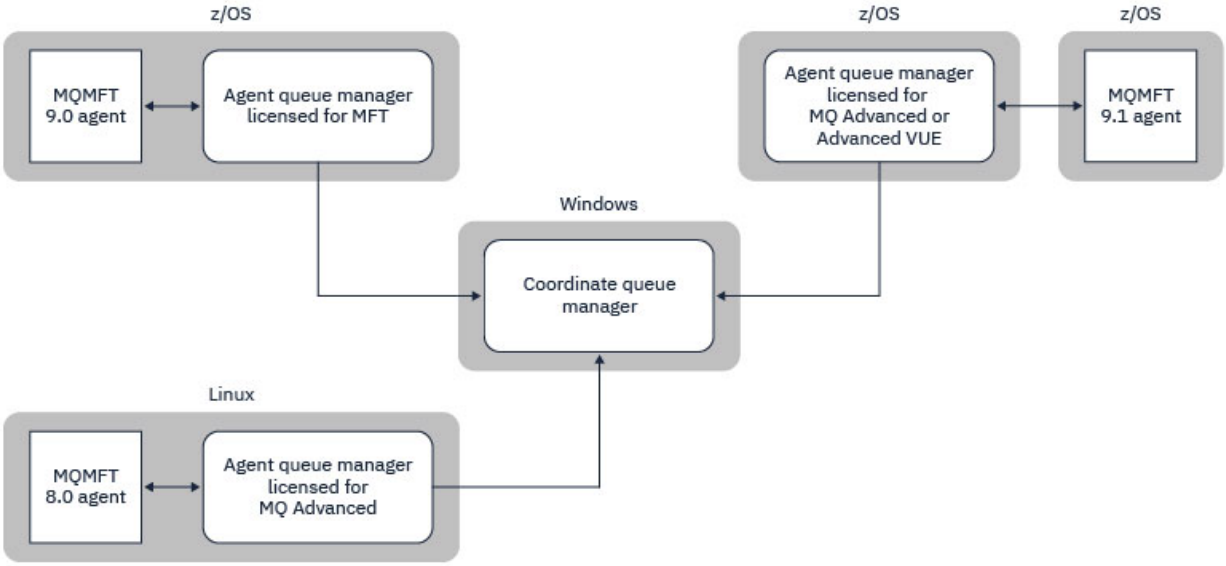


Figure 47. MFT agents running on z/OS can be part of an MFT topology where the coordination queue manager is running on an IBM MQ multiplatform.

The Which MFT commands and processes connect to which queue manager topic shows the commands that connect to the coordination queue manager for a Managed File Transfer topology. It is possible to run these commands on z/OS and have then connect to the coordination queue manager running on a different platform.

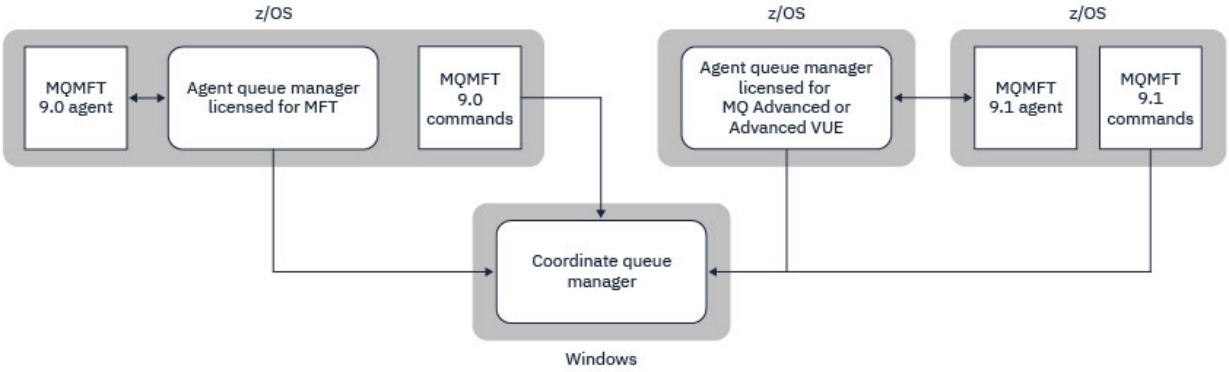


*Figure 48. Certain commands, such as **fteListAgents**, connect directly to the coordination queue manager for an MFT topology.*

How many agents do I need?

The agents do the work in transferring data, and when you make a request to transfer data you specify the name of an agent.

By default an agent can process 25 send and 25 receive requests concurrently. You can configure these processes. See Managed File Transfer configuration options on z/OS for more information.

If the agent is busy then work is queued. The time taken to process a request depends on multiple factors, for example, the amount of data to be sent, the network bandwidth, and the delay on the network.

You might want to have multiple agents to process work in parallel.

You can also control which resources an agent can access, so you might want some agents to work with a limited subset of data.

If you want to process requests with different priority you can use multiple agents and use workload manager to set the priority of the jobs.

Running the agents

Typically the agents are long running processes. The processes can be submitted as jobs that run in batch, or as started tasks.

z/OS Planning for Managed File Transfer - security considerations

Use this topic as guidance on what security considerations you need on your system to run Managed File Transfer (MFT) on z/OS.

Security

You need to identify which user IDs are going to be used for MFT configuration and for MFT operation.

You need to identify the files or queues you transfer, and which user IDs are going to be submitting transfer requests to MFT.

When you customize the agents and logger, you specify the group of users that is allowed to run MFT services, or do MFT administration.

You should set up this group before you start customizing MFT. As MFT uses IBM MQ queues, if you have security enabled in the queue manager, MFT requires access to the following resources:

Table 26. MQADMIN resource class

Name	Access required
QUEUE.SYSTEM.FTE.EVENT.agent_name	Update

Table 26. MQADMIN resource class (continued)	
Name	**Access required**
QUEUE.SYSTEM.FTE.COMMAND.agent_name	Update
CONTEXT.SYSTEM.FTE.COMMAND.agent_name	Update
QUEUE.SYSTEM.FTE.STATE.agent_name	Update
QUEUE.SYSTEM.FTE.DATA.agent_name	Update
QUEUE.SYSTEM.FTE.REPLY.agent_name	Update
QUEUE.SYSTEM.FTE.AUTHAGT1.agent_name	Update
QUEUE.SYSTEM.FTE.AUTHTRN1.agent_name	Update
QUEUE.SYSTEM.FTE.AUTHOPS1.agent_name	Update
QUEUE.SYSTEM.FTE.AUTHSCH1.agent_name	Update
QUEUE.SYSTEM.FTE.AUTHMON1.agent_name	Update
QUEUE.SYSTEM.FTE.AUTHADM1.agent_name	Update

Table 27. MQQUEUE resource class	
Name	**Access required**
SYSTEM.FTE.AUTHAGT1.agent_name	Update
SYSTEM.FTE.AUTHTRN1.agent_name	Update
SYSTEM.FTE.AUTHOPS1.agent_name	Update
SYSTEM.FTE.AUTHSCH1.agent_name	Update
SYSTEM.FTE.AUTHMON1.agent_name	Update

You can use user sandboxing to determine which parts of the file system the user who requests the transfer can access.

To enable user sandboxing, add the `userSandboxes=true` statement to the *agent.properties* file for the agent that you want to restrict, and add appropriate values to the `MQ_DATA_PATH/mqft/config/coordination_qmgr_name/agents/agent_name/UserSandboxes.xml` file.

See Working with user sandboxes for further information.

This user ID is configured in `UserSandboxes.xml` files.

This XML file has information like user ID, or user ID* and a list of resource that can be used (included), or cannot be used (excluded). You need to define specific user IDs that can access which resources: for example:

Table 28. Example user ID together with access to specific resources			
User ID	**Access**	**Include or Exclude**	**Resource**
Admin*	Read	Include	`/home/user/**`
Admin*	Read	Exclude	`/home/user/private/**`
Sysprog	Read	Include	`/home/user/**`
Admin*	Read	Include	Application.reply.queue

Notes:

1. If `type=queue` is specified, the resource is either a queue name, or `queue@qmgr`.
2. If the resource begins with `//`, the resource is a data set; otherwise the resource is a file in z/OS UNIX.
3. The user ID is the user ID from the MQMD structure, so this might not reflect the user ID that actually puts the message.
4. For requests on the local queue manager you can use MQADMIN CONTEXT.* to limit which users can set this value.
5. For requests coming in over a remote queue manager, you have to assume that the distributed queue managers have security enabled to prevent unauthorized setting of the user ID in the MQMD structure.
6. A user ID of SYSPROG1 on a Linux machine, is the same user ID SYSPROG1 for the security checking on z/OS.

z/OS Planning to use the IBM MQ Console and REST API on z/OS

The IBM MQ Console and REST API are applications that run in a WebSphere Liberty (Liberty) server known as mqweb. The mqweb server runs as a started task. The IBM MQ Console allows a web browser to be used to administer queue managers. The REST API provides a simple programmatic interface for applications to do queue manager administration, and to perform messaging.

Installation and configuration files

You need to install the IBM MQ for z/OS UNIX System Services Web Components feature, which will install the files needed to run the mqweb server in z/OS UNIX System Services (z/OS UNIX). You need to be familiar with z/OS UNIX to be able to configure and manage the mqweb server.

See IBM MQ for z/OS Program Directory PDF files for information on installing IBM MQ for z/OS UNIX System Services Components.

The IBM MQ files in z/OS UNIX are installed with various attributes set that are required for the correct operation of the mqweb server. If you need to copy the IBM MQ z/OS UNIX installation files, for example if you have installed IBM MQ on one system, and run IBM MQ on a different system, you should copy the IBM MQ ZFS created during the installation, and mount it read only at the destination. Copying the files in other ways might cause some file attributes to be lost.

You need to decide upon the location for, and create, a Liberty user directory when you create the mqweb server. This directory contains configuration and log files, and the location can be something similar to `/var/mqm/mqweb`.

Using the IBM MQ Console and REST API with queue managers at different levels

The REST API can directly interact only with queue managers that run at the same Version, Release, and Modification (VRM) as the mqweb server which runs the REST API. For example, the IBM MQ 9.4.0 REST API can directly interact only with local queue managers at IBM MQ 9.4.0, and the IBM MQ 9.3.5 REST API can directly interact only with local queue managers at IBM MQ 9.3.5.

You can use the REST API to administer a queue manager at a different version from the mqweb server by configuring a gateway queue manager. However, you need at least one queue manager at the same version as the mqweb server to act as the gateway queue manager. For more information, see Remote administration using the REST API.

The IBM MQ Console can be used to manage local queue managers that run at the same version as the IBM MQ Console. From IBM MQ 9.3.0, you can also use the IBM MQ Console to administer a queue manager running on a remote system, or at a different version to the IBM MQ Console. For more information, see IBM MQ Console: Adding a remote queue manager.

Migration

If you have only one queue manager, you can run the mqweb server as a single started task, and change the libraries it uses when you migrate your queue manager.

If you have more than one queue manager, during migration you can start mqweb servers at different versions by using started tasks with different names. These names can be any name you want. For example, you can start an IBM MQ 9.3.0 mqweb server using a started task named MQWB0930, and an IBM MQ 9.3.5 mqweb server using a started task named MQWB0935.

Then, when you migrate the queue managers from one version to a later version, the queue managers become available in the mqweb server for the later version, and are no longer available in the mqweb server for the earlier version.

After you have migrated all the queue managers to the later version, you can delete the mqweb server for the earlier version.

HTTP ports

The mqweb server uses up to two ports for HTTP:

- One for HTTPS, with a default value of 9443.
- One for HTTP. HTTP is not enabled by default, but if enabled, has a default value of 9080.

If the default port values are in use, you must allocate other ports. If you have more than one mqweb server running simultaneously for more than one version of IBM MQ, you must allocate separate ports for each version. For more information on setting the ports that the mqweb server uses, see Configuring the HTTP and HTTPS ports.

You can use the following TSO command to display information about a port:

```
NETSTAT TCP tcpip (PORT portNumber)
```

where *tcpip* is the name of the TCP/IP address space, and *portNumber* specifies the number of the port to display information about.

Security - starting the mqweb server

The mqweb server user ID needs certain authorities. For more information, see Authority required by the mqweb server started task user ID.

Security - using the IBM MQ Console and REST API

When you use the IBM MQ Console and REST API, you must authenticate as a user that is included in a configured registry. These users are assigned specific roles that determine the actions the users can perform. For example, to use the messaging REST API, a user must be assigned the `MQWebUser` role. For more information about the available roles for the IBM MQ Console and REST API, and the access that these roles grant, see Roles on the IBM MQ Console and REST API.

For more information about configuring security for the IBM MQ Console and REST API, see IBM MQ Console and REST API security.

声明

本信息是为在美国国内供应的产品和服务而编写的。

IBM 可能在其他国家或地区不提供本文档中讨论的产品、服务或功能。 有关您当前所在区域的产品和服务的信息，请向您当地的 IBM 代表咨询。 任何对 IBM 产品、程序或服务的引用并非意在明示或暗示只能使用 IBM 的产品、程序或服务。 只要不侵犯 IBM 的知识产权，任何同等功能的产品、程序或服务，都可以代替 IBM 产品、程序或服务。 但是，评估和验证任何非 IBM 产品、程序或服务，则由用户自行负责。

IBM 可能已拥有或正在申请与本文档内容有关的各项专利。 提供本文档并未授予用户使用这些专利的任何许可。 您可以用书面方式将许可查询寄往：

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

有关双字节（DBCS）信息的许可查询，请与您所在国家或地区的 IBM 知识产权部门联系，或用书面方式将查询寄往：

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

本条款不适用英国或任何这样的条款与当地法律不一致的国家或地区: International Business Machines Corporation “按现状” 提供本出版物，不附有任何种类的 (无论是明示的还是暗含的) 保证，包括但不限于暗含的有关非侵权，适销和适用于某种特定用途的保证。 某些国家或地区在某些交易中不允许免除明示或暗含的保证。 因此本条款可能不适用于您。

本信息中可能包含技术方面不够准确的地方或印刷错误。 此处的信息将定期更改；这些更改将编入本资料的新版本中。 IBM 可以随时对本资料中描述的产品和/或程序进行改进和/或更改，而不另行通知。

本信息中对非 IBM Web 站点的任何引用都只是为了方便起见才提供的，不以任何方式充当对那些 Web 站点的保证。 那些 Web 站点中的资料不是 IBM 产品资料的一部分，使用那些 Web 站点带来的风险将由您自行承担。

IBM 可以按它认为适当的任何方式使用或分发您所提供的任何信息而无须对您承担任何责任。

本程序的被许可方如果要了解有关程序的信息以达到如下目的：（i）允许在独立创建的程序和其他程序（包括本程序）之间进行信息交换，以及（ii）允许对已经交换的信息进行相互使用，请与下列地址联系：

IBM Corporation
软件互操作性协调员，部门 49XA
北纬 3605 号公路
罗切斯特，明尼苏达州 55901
U.S.A.

只要遵守适当的条件和条款，包括某些情形下的一定数量的付费，都可获得这方面的信息。

本资料中描述的许可程序及其所有可用的许可资料均由 IBM 依据 IBM 客户协议、IBM 国际软件许可协议或任何同等协议中的条款提供。

此处包含的任何性能数据都是在受控环境中测得的。 因此，在其他操作环境中获得的数据可能会有明显的不同。 有些测量可能是在开发级的系统上进行的，因此不保证与一般可用系统上进行的测量结果相同。 此外，有些测量是通过推算而估计的， 实际结果可能会有差异。 本文档的用户应当验证其特定环境的适用数据。

涉及非 IBM 产品的信息可从这些产品的供应商、其出版说明或其他可公开获得的资料中获取。 IBM 没有对这些产品进行测试，也无法确认其性能的精确性、兼容性或任何其他关于非 IBM 产品的声明。 有关非 IBM 产品性能的问题应当向这些产品的供应商提出。

所有关于 IBM 未来方向或意向的声明都可随时更改或收回，而不另行通知，它们仅仅表示了目标和意愿而已。

本信息包含日常商业运作所使用的数据和报表的示例。 为了尽可能全面地说明这些数据和报表，这些示例包括个人、公司、品牌和产品的名称。 所有这些名字都是虚构的，若现实生活中实际业务企业使用的名字和地址与此相似，纯属巧合。

版权许可：

本信息包括源语言形式的样本应用程序，这些样本说明不同操作平台上的编程方法。 如果是为按照在编写样本程序的操作平台上的应用程序编程接口（API）进行应用程序的开发、使用、经销或分发为目的，您可以任何形式对这些样本程序进行复制、修改、分发，而无须向 IBM 付费。 这些示例并未在所有条件下作全面测试。 因此，IBM 不能担保或默示这些程序的可靠性、可维护性或功能。

如果您正在查看本信息的软拷贝，图片和彩色图例可能无法显示。

编程接口信息

编程接口信息 (如果提供) 旨在帮助您创建用于此程序的应用软件。

本书包含有关允许客户编写程序以获取 IBM MQ 服务的预期编程接口的信息。

但是，该信息还可能包含诊断、修改和调优信息。 提供诊断、修改和调优信息是为了帮助您调试您的应用程序软件。

要点: 请勿将此诊断，修改和调整信息用作编程接口，因为它可能会发生更改。

商标

IBM IBM 徽标 ibm.com 是 IBM Corporation 在全球许多管辖区域的商标。 当前的 IBM 商标列表可从 Web 上的 “Copyright and trademark information”www.ibm.com/legal/copytrade.shtml 获取。 其他产品和服务名称可能是 IBM 或其他公司的商标。

Microsoft 和 Windows 是 Microsoft Corporation 在美国和/或其他国家或地区的商标。

UNIX 是 The Open Group 在美国和其他国家或地区的注册商标。

Linux 是 Linus Torvalds 在美国和/或其他国家或地区的商标。

此产品包含由 Eclipse 项目 (https://www.eclipse.org/) 开发的软件。

Java 和所有基于 Java 的商标和徽标是 Oracle 和/或其附属公司的商标或注册商标。

部件号:

(1P) P/N: