8.0

Migrating and Upgrading IBM MQ

IBM

Note

Before using this information and the product it supports, read the information in "Notices" on page 283.

This edition applies to version 8 release 0 of IBM® MQ and to all subsequent releases and modifications until otherwise indicated in new editions.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 2007, 2022.
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Migrating and upgrading IBM MQ

Migration is the process of updating queue managers, and other objects, such as applications or administrative procedures. To migrate a queue manager to run on a new level of code, you must first upgrade IBM MQ to install the new code level. When you have verified that the upgrade is successful, migrate the queue manager and all the applications and resources that are associated with it. Before you start this process, create a migration plan, based on the information in this documentation.

Maintenance, migration and upgrading

IBM MQ uses this terminology as follows:

- Migration is the process of updating queue manager data to match a newer level of code. This occurs the first time a queue manager is started with the newer level of code.
- Maintenance is the application of a fix pack, interim fix or program Temporary Fix (PTF). The installation can be restored to its previous level and queue managers or applications continue to work. Migration is not required after applying maintenance. However, you should test applications with the new level of IBM MQ code.
- Upgrading is the process of taking an existing IBM MQ installation and upgrading to a new level of code. Unless the upgrade is applying a fix (and not enabling new function), an upgrade must be followed by migration.
- Once migration has occurred, the queue manager can no longer be started by an earlier code level. On most platforms, queue manager migration is not reversible. z/OS The exception is z/OS®, where you can reverse queue manager migration, but only if you have not enabled new function.

IBM WebSphere MQ / IBM MQ Migration Guides

distributed The *IBM WebSphere MQ / IBM MQ Migration Guide* provides information to help you plan the process of migrating from an older version to a new version of the product on distributed systems.

- dW For an introduction to the guide and its contents, see the IBM Developer blog article IBM WebSphere® MQ / IBM MQ Migration Guide.
- To view the guide in your web browser, click the following link: IBM WebSphere MQ / IBM MQ Migration Guide - HTML version.
- To download the guide as a PDF file, click the following link: IBM WebSphere MQ / IBM MQ Migration Guide - PDF file.

z/OS The *IBM WebSphere MQ/ IBM MQ for z/OS Migration Guide* provides information to help you plan the process of migrating from an older version to a new version of the product on z/OS.

- For an introduction to the guide and its contents, see the Support page IBM WebSphere MQ / IBM MQ for z/OS Migration Guide.
- To view the guide in your web browser, click the following link: IBM WebSphere MQ / IBM MQ for z/OS Migration Guide - HTML version.
- To download the guide as a PDF file, click the following link: IBM WebSphere MQ / IBM MQ for z/OS Migration Guide - PDF file.

Where to find information about system requirements and prerequisites

From IBM MQ 8.0, you can use the Software Product Compatibility Reports (SPCR) tool to find information on supported operating systems, system requirements, prerequisites, and optional supported software.

© Copyright IBM Corp. 2007, 2022

For more information about the SPCR tool and links to reports for each supported platform, see the System Requirements for IBM MQ Version 8.0 web page.

For links to system requirements information for all releases of IBM WebSphere MQ or IBM MQ, see the System Requirements for IBM MQ web page.

For information about limitations and known problems for IBM MQ 8.0 and its maintenance, see the product readme file, which is available from the product readmes web page.

Migrating from a release of IBM WebSphere MQ before Version 7.0

Important: If you are migrating your system from a version of IBM WebSphere MQ before Version 7.0, you must migrate your system to Version 7.0, Version 7.0.1, or Version 7.1 before you migrate to Version 8.0. See the appropriate version of the product documentation for information on how to carry out the task. For links to earlier versions of the product documentation not available in IBM Documentation, see the IBM MQ documentation library page.

Getting started with migrating and upgrading

If you are not familiar with IBM MQ migration, start by reading the following information:

- The "Introduction to IBM MQ migration" on page 19 section: use these topics to find out more about the concepts that you must understand before planning migration tasks, including the difference between maintenance, migration, and upgrading and which migration paths are supported.
- The "IBM WebSphere MQ / IBM MQ Migration Guides" on page 5: use these guides to find more information about planning the migration process for your release and platform.

For information about new features and changes in this release, see the following topics:

- What's new in IBM MQ 8.0
- What's changed in IBM MQ 8.0
- What's new and what's changed in MQ Explorer

Some of the changes have a migration impact, because they affect the behavior of existing applications or the automation of management tasks. This subset of the changes is listed in "Changes that affect migration" on page 173. Study this list of changes to plan what migration tasks you must perform.

If you prefer to use the IBM MQ 8.0 product documentation offline, you can download it through the links on the IBM MQ documentation library page, either as a downloadable package or as a set of PDF files.

Related information

z/OS IBM MQ for z/OS Program Directory

Entities for MQTT clients

MQTT names for client pack, clients, and client sample apps

machine-to-machine (M2M)
Mobile Messaging and M2M
Mobile Messaging and M2M Client Pack
IBM Messaging Telemetry Clients
IBM Messaging Telemetry Clients SupportPac
MQTT client for Java
MQTT messaging client for JavaScript
MQTT client for C
MQTT client sample Java app
MQTT client sample Java app for Android
MQTT messaging client sample JavaScript pages
MQTT client sample C app

MQTT phrases, xrefs, filepaths

The examples are `MQTTV3ASample.c` and `MQTTV3ASSample.c` in *`sdkroot`*`\SDK\clients\c\samples`.

MQTT build options for different platforms

Install a C development environment on the platform on which you are building.

System requirements for IBM Messaging Telemetry Clients SupportPac

The client identifier must be unique across all clients that connect to the server, and must not be the same as the queue manager name on the server.

The configuration gives everyone permission to publish and subscribe to any topic. The configuration of security and access control is minimal and is intended only for a queue manager that is on a secure network with restricted access. To run IBM WebSphere MQ and MQTT in an insecure environment, you must configure security. To configure security for IBM WebSphere MQ and MQTT, see the related links at the end of this task.

If you use the default `MqttConnectOptions`, or set `MqttConnectOptions.cleanSession` to `true` before connecting the client, any old subscriptions for the client are removed when the client connects. Any new subscriptions the client makes during the session are removed when it disconnects.

If you set `MqttConnectOptions.cleanSession` to `false` before connecting, any subscriptions the client creates are added to all the subscriptions that existed for the client before it connected. All the subscriptions remain active when the client disconnects.

Another way of understanding the way the `cleanSession` attribute affects subscriptions is to think of it as a modal attribute. In its default mode, `cleanSession=true`, the client creates subscriptions and receives publications only within the scope of the session. In the alternative mode, `cleanSession=false`, subscriptions are durable. The client can connect and disconnect and its subscriptions remain active. When the client reconnects, it receives any undelivered publications. While it is connected, it can modify the set of subscriptions that are active on its behalf.

You must set the `cleanSession` mode before connecting; the mode lasts for the whole session. To change its setting, you must disconnect and reconnect the client. If you change modes from using `cleanSession=false` to `cleanSession=true`, all previous subscriptions for the client, and any publications that have not been received, are discarded.

Example scripts to configure SSL certificates for Windows

Change *`sdkroot`* to suit your environment.

Install a Java development kit (JDK) at Version 7 or later.

Version 7 is required to run the **`keytool`** command to certify certificates. If you are not going to certify certificates, you do not require the Version 7 JDK.

```
# Build output is produced in the current directory.
# MQTTCLIENT_DIR must point to the base directory containing the MQTT client source code.
# Default MQTTCLIENT_DIR is the current directory
# Default TOOL_DIR is /Applications/Xcode.app/Contents/Developer/Platforms
# Default OPENSSL_DIR is sdkroot/openssl, relative to sdkroot/sdk/clients/c/mqttv3c/src
# OPENSSL_DIR must point to the base directory containing the OpenSSL build.
# Example:  make -f MQTTios.mak MQTTCLIENT_DIR=sdkroot/sdk/clients/c/mqttv3c/src all
ifndef TOOL_DIR
  TOOL_DIR = /Applications/Xcode.app/Contents/Developer/Platforms endif
IPHONE_SDK = iPhoneOS.platform/Developer/SDKs/iPhoneOS6.0.sdk
IPHONESIM_SDK = iPhoneSimulator.platform/Developer/SDKs/iPhoneSimulator6.0.sdk
SDK_ARM = ${TOOL_DIR}/${IPHONE_SDK}
SDK_i386 = ${TOOL_DIR}/${IPHONESIM_SDK}
MQTTLIB = mqttv3c
SOURCE_FILES = ${filter-out ${MQTTCLIENT_DIR}/MQTTAsync.c ${MQTTCLIENT_DIR}/SSLSocket.c, $
{ALL_SOURCE_FILES}}
MQTTLIB_DARWIN = darwin_x86_64/lib${MQTTLIB}.a
MQTTLIB_S = mqttv3cs
SOURCE_FILES_S = ${filter-out ${MQTTCLIENT_DIR}/MQTTAsync.c, ${ALL_SOURCE_FILES}}
MQTTLIB_DARWIN_S = darwin_x86_64/lib${MQTTLIB_S}.a
MQTTLIB_A = mqttv3a
SOURCE_FILES_A = ${filter-out ${MQTTCLIENT_DIR}/MQTTClient.c ${MQTTCLIENT_DIR}/SSLSocket.c,
${ALL_SOURCE_FILES}}
```

```
MQTTLIB_DARWIN_A = darwin_x86_64/lib${MQTTLIB_A}.a
MQTTLIB_AS = mqttv3as
SOURCE_FILES_AS = ${filter-out ${MQTTCLIENT_DIR}/MQTTClient.c, ${ALL_SOURCE_FILES}}
MQTTLIB_DARWIN_AS = darwin_x86_64/lib${MQTTLIB_AS}.a
CC = iPhoneOS.platform/Developer/usr/bin/gcc
CC_armv7 = ${TOOL_DIR}/${CC} -arch armv7
CC_armv7s = ${TOOL_DIR}/${CC} -arch armv7s
CC_i386 = ${TOOL_DIR}/${CC} -arch i386
CCFLAGS = -Os -Wall -fomit-frame-pointer
CCFLAGS_SO_ARM =  ${CCFLAGS} -isysroot ${SDK_ARM} -I${OPENSSL_DIR}/include  -L$
{SDK_ARM}/usr/lib/system
CCFLAGS_SO_i386 = ${CCFLAGS} -isysroot ${SDK_i386} -I${OPENSSL_DIR}/include -L$
{SDK_i386}/usr/lib/system
all: ${MQTTLIB_DARWIN} ${MQTTLIB_DARWIN_A} ${MQTTLIB_DARWIN_AS} ${MQTTLIB_DARWIN_S}
${MQTTLIB_DARWIN}: ${SOURCE_FILES}
   -mkdir darwin_x86_64
   ${CC_armv7} ${CCFLAGS_SO_ARM} -c ${SOURCE_FILES} -DUSE_NAMED_SEMAPHORES
-DNOSIGPIPE
   libtool -static -syslibroot ${SDK_ARM} -o $@.armv7 *.o
   rm *.o
   ${CC_armv7s} ${CCFLAGS_SO_ARM} -c ${SOURCE_FILES} -DUSE_NAMED_SEMAPHORES
-DNOSIGPIPE
   libtool -static -syslibroot ${SDK_ARM} -o $@.armv7s *.o
   rm *.o
   ${CC_i386} ${CCFLAGS_SO_i386} -c ${SOURCE_FILES} -DUSE_NAMED_SEMAPHORES -DNOSIGPIPE
   libtool -static -syslibroot ${SDK_i386} -o $@.i386 *.o
   rm *.o
   lipo -create $@.armv7 $@.armv7s $@.i386 -output $@
${MQTTLIB_DARWIN_A}: ${SOURCE_FILES_A}
   -mkdir darwin_x86_64
   ${CC_armv7} ${CCFLAGS_SO_ARM} -c ${SOURCE_FILES_A} -DMQTT_ASYNC
-DUSE_NAMED_SEMAPHORES -DNOSIGPIPE
   libtool -static -syslibroot ${SDK_ARM} -o $@.armv7 *.o
   rm *.o
   ${CC_armv7s} ${CCFLAGS_SO_ARM} -c ${SOURCE_FILES_A} -DMQTT_ASYNC
-DUSE_NAMED_SEMAPHORES -DNOSIGPIPE
   libtool -static -syslibroot ${SDK_ARM} -o $@.armv7s *.o
   rm *.o
   ${CC_i386} ${CCFLAGS_SO_i386} -c ${SOURCE_FILES_A} -DMQTT_ASYNC
-DUSE_NAMED_SEMAPHORES -DNOSIGPIPE
   libtool -static -syslibroot ${SDK_i386} -o $@.i386 *.o
   rm *.o
   lipo -create $@.armv7 $@.armv7s $@.i386 -output $@
${MQTTLIB_DARWIN_S}: ${SOURCE_FILES_S}
   -mkdir darwin_x86_64
   ${CC_armv7} ${CCFLAGS_SO_ARM} -c ${SOURCE_FILES_S} -DOPENSSL
-DUSE_NAMED_SEMAPHORES -DNOSIGPIPE
   libtool -static -syslibroot ${SDK_ARM} -L${OPENSSL_DIR}/armv7 -lssl -lcrypto -o $@.armv7 *.o
   rm *.o
   ${CC_armv7s} ${CCFLAGS_SO_ARM} -c ${SOURCE_FILES_S} -DOPENSSL
-DUSE_NAMED_SEMAPHORES -DNOSIGPIPE
   libtool -static -syslibroot ${SDK_ARM} -L${OPENSSL_DIR}/armv7s -lssl -lcrypto -o $@.armv7s *.o
   rm *.o
   ${CC_i386} ${CCFLAGS_SO_i386} -c ${SOURCE_FILES_S} -DOPENSSL -DUSE_NAMED_SEMAPHORES
-DNOSIGPIPE
   libtool -static -syslibroot ${SDK_ARM} -L${OPENSSL_DIR}/i386 -lssl -lcrypto -o $@.i386 *.o
   rm *.o
   lipo -create $@.armv7 $@.armv7s $@.i386 -output $@
```

```
${MQTTLIB_DARWIN_AS}: ${SOURCE_FILES_AS}
   -mkdir darwin_x86_64
   ${CC_armv7} ${CCFLAGS_SO_ARM} -c ${SOURCE_FILES_AS} -DMQTT_ASYNC -DOPENSSL
-DUSE_NAMED_SEMAPHORES -DNOSIGPIPE
   libtool -static -syslibroot ${SDK_ARM} -L${OPENSSL_DIR}/armv7 -lssl -lcrypto -o $@.armv7 *.o
   rm *.o
   ${CC_armv7s} ${CCFLAGS_SO_ARM} -c ${SOURCE_FILES_AS} -DMQTT_ASYNC -DOPENSSL
-DUSE_NAMED_SEMAPHORES -DNOSIGPIPE
   libtool -static -syslibroot ${SDK_ARM} -L${OPENSSL_DIR}/armv7s -lssl -lcrypto -o $@.armv7s *.o
   rm *.o
   ${CC_i386} ${CCFLAGS_SO_i386} -c ${SOURCE_FILES_AS} -DMQTT_ASYNC -DOPENSSL
-DUSE_NAMED_SEMAPHORES -DNOSIGPIPE
   libtool -static -syslibroot ${SDK_ARM} -L${OPENSSL_DIR}/i386 -lssl -lcrypto -o $@.i386 *.o
   rm *.o
   lipo -create $@.armv7 $@.armv7s $@.i386 -output $@
.PHONY : clean
clean:
   -rm -f *.obj
   -rm -f -r darwin_x86_64
```

Download and install the iOS development tools.

For links to client API documentation for the MQTT client libraries, see MQTT client programming reference.

You can run an MQTT client for Java app on any platform with JSE 1.5 or above that is "Java Compatible"

Log on with a user ID that has administrative authority to IBM WebSphere MQ.

The server is now ready for you to test your MQTT V3.1 app.

You must have access to an MQTT Version 3.1 server that supports the MQTT protocol over SSL.

If there is a firewall between your client and the server, check that it does not block MQTT traffic.

You can run an MQTT client for Java app on any platform with JSE 1.5 or above that is "Java Compatible". See System requirements for IBM Messaging Telemetry Clients SupportPac.

The SSL channels must be started.

Choose an MQTT server to which you can connect the client app.

The statement `rm *.o` deletes all the object files that are created for each library. `lipo` concatenates all three libraries into one file.

Download the IBM Messaging Telemetry Clients SupportPac.

Download the IBM Messaging Telemetry Clients SupportPac and install the MQTT SDK.

either IBM MessageSight or IBM WebSphere MQ as the MQTT server

The server must support the MQTT Version 3.1 protocol.

The server must support the MQTT Version 3.1 protocol over SSL.

The MQTT Paho sample applications in the SDK are set up to connect to the Eclipse M2M server by default. See MQTT Sandbox Server. You do not have to configure a server to try the Paho sample applications out.

Create and run the scripts to generate key-pairs and certificates, and configure IBM WebSphere MQ as the MQTT server.

Check the SSL channels are running and are set up as you expect.

Connections between the MQTT client and the queue manager are always initiated by the MQTT client. The MQTT client is always the SSL client. Client authentication of the server and server authentication of the MQTT client are both optional.

As an alternative to using SSL, some kinds of Virtual Private Network (VPN), such as IPsec, authenticate the endpoints of a TCP/IP connection. VPN encrypts each IP packet that flows over the network. Once such a VPN connection is established, you have established a trusted network. You can connect MQTT clients to telemetry channels using TCP/IP over the VPN network.

The client always attempts to authenticate the server, unless the client is configured to use a CipherSpec that supports anonymous connection. If the authentication fails, then the connection is not established.

By providing the client with a private signed digital certificate, you can authenticate the MQTT client to WebSphere MQ. The WebSphere MQ Administrator can force MQTT clients to authenticate themselves to the queue manager using SSL. You can only request client authentication as part of mutual authentication.

Client authentication using SSL relies upon the client having a secret. The secret is the private key of the client in the case of a self-signed certificate, or a key provided by a certificate authority. The key is used to sign the digital certificate of the client. Anyone in possession of the corresponding public key can verify the digital certificate. Certificates can be trusted, or if they are chained, traced back through a certificate chain to a trusted root certificate. Client verification sends all the certificates in the certificate chain provided by the client to the server. The server checks the certificate chain until it finds a certificate it trusts. The trusted certificate is either the public certificate generated from a self-signed certificate, or a root certificate typically issued by a certificate authority. As a final, optional, step the trusted certificate can be compared with a "live" certificate revocation list.

The trusted certificate might be issued by a certificate authority and already included in the JRE certificate store. It might be a self-signed certificate, or any certificate that has been added to the telemetry channel keystore as a trusted certificate.

The telemetry channel has a combined keystore/truststore that holds both the private keys to one or more telemetry channels, and any public certificates needed to authenticate clients. Because an SSL channel must have a keystore, and it is the same file as the channel truststore, the JRE certificate store is never referenced. The implication is that if authentication of a client requires a CA root certificate, you must place the root certificate in the keystore for the channel, even if the CA root certificate is already in the JRE certificate store. The JRE certificate store is never referenced.

Think about the threats that client authentication is intended to counter, and the roles the client and server play in countering the threats. Authenticating the client certificate alone is insufficient to prevent unauthorized access to a system. If someone else has got hold of the client device, the client device is not necessarily acting with the authority of the certificate holder. Never rely on a single defense against unwanted attacks. At least use a two-factor authentication approach and supplement possession of a certificate with knowledge of private information. For example, use JAAS, and authenticate the client using a password issued by the server.

The primary threat to the client certificate is that it gets into the wrong hands. The certificate is held in a password protected keystore at the client. How does it get placed in the keystore? How does the MQTT client get the password to the keystore? How secure is the password protection? Telemetry devices are often easy to remove, and then can be hacked in private. Must the device hardware be tamper-proof? Distributing and protecting client-side certificates is recognized to be hard; it is called the key-management problem.

A secondary threat is that the device is misused to access servers in unintended ways. For example, if the MQTT application is tampered with, it might be possible to use a weakness in the server configuration using the authenticated client identity.

To authenticate an MQTT client using SSL, configure the telemetry channel, and the client.

The IBM MQ administrator configures telemetry channels at the server. Each channel is configured to accept a TCP/IP connection on a different port number. SSL channels are configured with passphrase protected access to key files. If an SSL channel is defined with no passphrase or key file, the channel does not accept SSL connections.

The client JVM must use the standard socket factory from JSSE. If you are using Java ME, you must ensure that the JSSE package is loaded. If you are using Java SE, JSSE has been included with the JRE since Java version 1.4.1.

The SSL connection requires a number of SSL properties to be set before connecting. You can set the properties either by passing them to the JVM using the `-D` switch, or you can set the properties using the `MqttConnectionOptions.setSSLProperties` method.

If you load a non-standard socket factory, by calling the method `MqttConnectOptions.setSocketFactory(javax.net.SocketFactory)`, then the way SSL settings are passed to the network socket is application defined.

Server authentication using SSL authenticates the server to which you are about to send confidential information to. The client performs the checks matching the certificates sent from the server, against certificates placed in its truststore, or in its JRE `cacerts` store.

The JRE certificate store is a JKS file, `cacerts`. It is located in `JRE InstallPath\lib\security\`. It is installed with the default password `changeit`. You can either store certificates you trust in the JRE certificate store, or in the client truststore. You cannot use both stores. Use the client truststore if you want to keep the public certificates the client trusts separate from certificates other Java applications use. Use the JRE certificate store if you want to use a common certificate store for all Java applications running on the client. If you decide to use the JRE certificate store review the certificates it contains, to make sure you trust them.

You can modify the JSSE configuration by supplying a different trust provider. You can customize a trust provider to perform different checks on a certificate. In some OGSi environments that have used the

MQTT client, the environment provides a different trust provider.

To authenticate the telemetry channel using SSL, configure the server, and the client.

When `SSLCIPH` is used with a telemetry channel, it means "SSL Cipher Suite".

The SSL cipher suite is the one supported by the JVM that is running the telemetry (MQXR) service.

If the `SSLCIPH` parameter is blank, no attempt is made to use SSL on the channel.

`SSLCIPH` specifies the CipherSpec that is used on the channel. The maximum length is 32 characters.

This parameter is valid on all channel types which use transport type `TRPTYPE(TCP)`.

Specify the name of the CipherSpec you are using. The CipherSpecs that can be used with IBM MQ SSL support are shown in the following table. If a specific named CipherSpec is being used, the **`SSLCIPH`** values at the two ends of a channel must specify the same named CipherSpec.

On z/OS IBM MQ for z/OS; IBM i IBM i; you can also specify the two digit hexadecimal code of a CipherSpec, whether or not it appears in the table. On IBM i, installation of AC3 is a prerequisite for the use of SSL.

If you plan to use SHA-2 cipher suites, see System requirements for using SHA-2 cipher suites with MQTT channels.

The value for this parameter is also used to set the value of SECPROT

Download and redistribution of the OpenSSL package is subject to stringent import and export regulation, and open source licensing conditions. Take careful note of the restrictions and warnings before you decide whether to download the package.

For the C samples, the store is a Privacy-Enhanced Mail (PEM) file. For the Java samples it is a Java keystore (JKS).

Create a script in the client samples directory to compile and run `Sample` on your chosen platform.

Install a Java development kit (JDK) Version 6 or later.

Because you are developing a Java app for Android, the JDK must come from Oracle. You can get the JDK from Java SE Downloads.

Each successive line that defines the implementation of a target must start with a tab character.

The default file persistence class in the Java SE MQTT client supplied with IBM MQ Telemetry creates a folder with the name: *clientIdentifier*-tcp*hostNameport* or *clientIdentifier*-ssl*hostNameport* in the client working directory. The folder name tells you the `hostName` and `port` used in the connection attempt

Open the MQTT client sample Java app for Android.

Connect to an MQTT server.

```
# Build output is produced in the current directory.
# MQTTCLIENT_DIR must point to the base directory containing the MQTT client source code.
# Default MQTTCLIENT_DIR is the current directory
# Default OPENSSL_DIR is sdkroot\openSSL, relative to sdkroot\sdk\clients\c\mqttv3c\src
# OPENSSL_DIR must point to the base directory containing the OpenSSL build.
# Example:  make -f MQTTwin.mak MQTTCLIENT_DIR=sdkroot/sdk/clients/c/mqttv3c/src
# Set the build environment, for example:
#    %comspec% /k ""C:\Program Files\Microsoft Visual Studio 9.0\VC\vcvarsall.bat"" x86
#    set path=%path%;C:\Program Files\GnuWin32\bin;C:\cygwin\bin
ifndef MQTTCLIENT_DIR
  MQTTCLIENT_DIR = ${CURDIR}
endif
VPATH = ${MQTTCLIENT_DIR}
ifndef OPENSSL_DIR
  OPENSSL_DIR = ${MQTTCLIENT_DIR}/../../../../../openssl-1.0.1c
endif
ALL_SOURCE_FILES = ${wildcard ${MQTTCLIENT_DIR}/*.c}
MQTTLIB = mqttv3c
MQTTDLL = windows_ia32/${MQTTLIB}.dll
SOURCE_FILES = ${filter-out ${MQTTCLIENT_DIR}/MQTTAsync.c ${MQTTCLIENT_DIR}/SSLSocket.c, $
{ALL_SOURCE_FILES}}
MANIFEST = mt -manifest ${MQTTDLL}.manifest -outputresource:${MQTTDLL}\;2
MQTTLIB_S = mqttv3cs
MQTTDLL_S  = windows_ia32/${MQTTLIB_S}.dll
SOURCE_FILES_S = ${filter-out ${MQTTCLIENT_DIR}/MQTTAsync.c, ${ALL_SOURCE_FILES}}
MANIFEST_S = mt -manifest ${MQTTDLL_S}.manifest -outputresource:${MQTTDLL_S}\;2
```

```
MQTTLIB_A = mqttv3a
MQTTDLL_A  = windows_ia32/${MQTTLIB_A}.dll
SOURCE_FILES_A = ${filter-out ${MQTTCLIENT_DIR}/MQTTClient.c ${MQTTCLIENT_DIR}/SSLSocket.c,
${ALL_SOURCE_FILES}}
MANIFEST_S = mt -manifest ${MQTTDLL_S}.manifest -outputresource:${MQTTDLL_S}\;2
MQTTLIB_AS = mqttv3as
MQTTDLL_AS  = windows_ia32/${MQTTLIB_AS}.dll
SOURCE_FILES_AS = ${filter-out ${MQTTCLIENT_DIR}/MQTTClient.c, ${ALL_SOURCE_FILES}}
MANIFEST_S = mt -manifest ${MQTTDLL_S}.manifest -outputresource:${MQTTDLL_S}\;2
CC = cl
CPPFLAGS = /D "WIN32" /D "_UNICODE" /D "UNICODE" /D "_CRT_SECURE_NO_WARNINGS"
CFLAGS = /nologo /c /O2 /W3 /Fd /MD /TC
INC =  /I ${MQTTCLIENT_DIR} /I ${MQTTCLIENT_DIR}/..
CPPFLAGS_S = ${CPPFLAGS} /D "OPENSSL"
INC_S = ${INC} /I ${OPENSSL_DIR}/inc32/
LD = link
LINKFLAGS = /nologo /machine:x86 /manifest /dll
WINLIBS = kernel32.lib user32.lib gdi32.lib winspool.lib comdlg32.lib\
        advapi32.lib shell32.lib ole32.lib oleaut32.lib uuid.lib\
        odbc32.lib odbccp32.lib ws2_32.lib
IMP = /implib:${@:.dll=.lib}
LIBPDB = /pdb:${@:.dll=.pdb}
LIBMAP = /map:${@:.dll=.map}
WINLIBS_S = ${WINLIBS} crypt32.lib ssleay32.lib libeay32.lib
LIBPATH_S = /LIBPATH:${OPENSSL_DIR}/lib
${MQTTDLL}: ${SOURCE_FILES}
   -mkdir windows_ia32
   -rm ${CURDIR}/MQTTAsync.obj
   ${CC} ${CPPFLAGS} ${CFLAGS} ${INC} /Fo ${SOURCE_FILES}
   ${LD} ${LINKFLAGS} ${IMP} ${LIBPDB} ${LIBMAP} ${WINLIBS} *.obj /out:${MQTTDLL}
   ${MANIFEST}
${MQTTDLL_A}: ${SOURCE_FILES_A}
   -mkdir windows_ia32
   -rm ${CURDIR}/MQTTClient.obj
   ${CC} ${CPPFLAGS} ${CFLAGS} ${INC} /Fo ${SOURCE_FILES_A}
   ${LD} ${LINKFLAGS} ${IMP} ${LIBPDB} ${LIBMAP} ${WINLIBS} *.obj /out:${MQTTDLL_A}
   ${MANIFEST_A}
${MQTTDLL_S}: ${SOURCE_FILES_S}
   -mkdir windows_ia32
   -rm ${CURDIR}/MQTTAsync.obj
   ${CC} ${CPPFLAGS_S} ${CFLAGS} ${INC_S} /Fo ${SOURCE_FILES}
   ${LD} ${LINKFLAGS} ${IMP} ${LIBPDB} ${LIBMAP} ${WINLIBS_S} ${LIBPATH_S} *.obj /out:$
{MQTTDLL_S}
   ${MANIFEST_S}
${MQTTDLL_AS}: ${SOURCE_FILES_AS}
   -rm ${CURDIR}/MQTTClient.obj
   ${CC} ${CPPFLAGS_S} ${CFLAGS} ${INC_S} /Fo ${SOURCE_FILES_AS}
   ${LD} ${LINKFLAGS} ${IMP} ${LIBPDB} ${LIBMAP} ${WINLIBS_S} ${LIBPATH_S} *.obj /out:$
(MQTTDLL_AS}
   $(MANIFEST_AS}
all: ${MQTTDLL} ${MQTTDLL_A} ${MQTTDLL_AS} ${MQTTDLL_S}
.PHONY : clean
clean:
   -rm -f *.obj
   -rm -f -r windows_ia32
```

This step is optional on Windows because you can administer IBM WebSphere MQ as a Windows administrator.

Set the location of the MQTT source code.

Run the makefile in the same directory as the MQTT source files, or set the `MQTTCLIENT_DIR` command-line parameter:
make -f makefile MQTTCLIENT_DIR=*sdkroot*/SDK/clients/c/mqttv3c/src
Add the following lines to the makefile:
The example sets VPATH to the directory where **make** searches for source files that are not explicitly identified; for example all the header files that are required in the build.
Set the location of the OpenSSL libraries.
This step is required to build the SSL versions of the MQTT client for C libraries.
Set the default path to the OpenSSL libraries to same directory as you expanded the MQTT SDK. Otherwise, set `OPENSSL_DIR` as a command-line parameter.
`OpenSSL` is the OpenSSL directory that contains all the OpenSSL subdirectories. You might have to move the directory tree from where you expanded it, because it contains unnecessary empty parent directories.
Select all the source files that are required to build each MQTT library. Also, set the name and location of the MQTT library to build.
The source files depend on whether you are building a synchronous or asynchronous library, and whether the library includes SSL or not.
Add the following line to the makefile to list all the MQTT source files:

Set the keystore location and attributes with MQ Explorer, or with the **`DEFINE CHANNEL`** command; see DEFINE CHANNEL (MQTT). Multiple channels can share a keystore.

MQTT non-phrases

```
MQTTClient mqttClient = new MqttClient( "ssl://www.example.org:8884", "clientId1");
mqttClient.connect();
```

`SampleAsyncCallBack`

`SampleAsyncCallBack` is in the `org.eclipse.paho.client.mqttv3` package. It calls the asynchronous MQTT API. The asynchronous API does not wait for MQTT to complete processing a call; it returns to the application. The application carries on with other tasks, then waits for the next event to arrive for it to process. MQTT posts an event notification back to the application when it completes processing. The event driven MQTT interface is suited to the service and activity programming model of Android and other event driven operating systems.

As an example, look at how the `mqttExerciser` sample integrates MQTT into Android using the service and activity programming model.

`SampleAsyncWait`

`SampleAsyncWait` is in the `org.eclipse.paho.client.mqttv3` package. It uses the asynchronous MQTT API; it waits on a different thread until an action completes. The main thread can do other work until it synchronizes on the thread that is waiting for the MQTT action to complete.

The example command files create the certificates and certificate stores as described in the steps in the task. In addition, the example sets up the MQTT client queue manager to use the server certificate store. The example deletes and re-creates the queue manager by calling the `SampleMQM.bat` script that is provided with IBM WebSphere MQ.

`initcert.bat`

`initcert.bat` sets the names and paths to certificates and other parameters that are required by the **`keytool`** and **`openSSL`** commands. The settings are described in comments in the script.

```
@echo off
@rem Set the path where you installed the MQTT SDK
@rem and short cuts to the samples directories.
set SDKRoot=C:\MQTT
set jsamppath=%SDKRoot%\sdk\clients\java\samples
set csamppath=%SDKRoot%\sdk\clients\c\samples
```

```
@rem Set the paths to Version 7 of the JDK
@rem and to the directory where you built the openSSL package.
@rem Set short cuts to the tools.
set javapath=C:\Program Files\IBM\Java70
set keytool="%javapath%\jre\bin\keytool.exe"
set ikeyman="%javapath%\jre\bin\ikeyman.exe"
set openssl=%SDKRoot%\openSSL
set runopenssl="%openssl%\bin\openssl"
```

```
@rem Set the path to where certificates are to be stored,
@rem and set global security parameters.
@rem Omit set password, and the security tools prompt you for passwords.
@rem Validity is the expiry time of the certificates in days.
set certpath=%SDKRoot%\Certificates
set password=password
set validity=5000
set algorithm=RSA
```

```
@rem Set the certificate authority (CA) jks keystore and certificate parameters.
@rem Omit this step, unless you are defining your own certificate authority.
@rem The CA keystore contains the key-pair for your own certificate authority.
@rem You must protect the CA keystore.
@rem The CA certificate is the self-signed certificate authority public certificate.
@rem It is commonly known as the CA root certificate.
set caalias=caalias
set cadname="CN=mqttca.ibm.id.com, OU=ID, O=IBM, L=Hursley, S=Hants, C=GB"
set cakeypass=%password%
@rem ca key store
set cajkskeystore=%certpath%\cakeystore.jks
set cajkskeystorepass=%password%
@rem ca certificate (root certificate)
set cacert=%certpath%\cacert.cer
```

```
@rem Set the server jks keystore and certificate parameters.
@rem The server keystore contains the key-pair for the server.
@rem You must protect the server keystore.
@rem If you then export the server certificate it is self-signed.
@rem Alternatively, if you export a certificate signing request (CSR)
@rem from the server keystore for the server key,
@rem and import the signed certificate back into the same keystore,
@rem it forms a certificate chain.
@rem The certificate chain links the server certificate to the CA.
@rem When you now export the server certificate,
@rem the exported certificate includes the certificate chain.
set srvalias=srvalias
set srvdname="CN=mqttserver.ibm.id.com, OU=ID, O=IBM, L=Hursley, S=Hants, C=GB"
set srvkeypass=%password%
@rem server key stores
set srvjkskeystore=%certpath%\srvkeystore.jks
set srvjkskeystorepass=%password%
@rem server certificates
set srvcertreq=%certpath%\srvcertreq.csr
set srvcertcasigned=%certpath%\srvcertcasigned.cer
set srvcertselfsigned=%certpath%\srvcertselfsigned.cer
```

```
@rem Set the client jks keystore and certificate parameters
@rem Omit this step, unless you are authenticating clients.
@rem The client keystore contains the key-pair for the client.
@rem You must protect the client keystore.
@rem If you then export the client certificate it is self-signed.
@rem Alternatively, if you export a certificate signing request (CSR)
@rem from the client keystore for the client key,
@rem and import the signed certificate back into the same keystore,
@rem it forms a certificate chain.
@rem The certificate chain links the client certificate to the CA.
@rem When you now export the client certificate,
@rem the exported certificate includes the certificate chain.
set cltalias=cltalias
set cltdname="CN=mqttclient.ibm.id.com, OU=ID, O=IBM, L=Hursley, S=Hants, C=GB"
set cltkeypass=%password%
@rem client key stores
set cltjkskeystore=%certpath%\cltkeystore.jks
set cltjkskeystorepass=%password%
set cltcertreq=%certpath%\cltcertreq.csr
set cltcertcasigned=%certpath%\cltcacertsigned.cer
set cltcertselfsigned=%certpath%\cltcertselfsigned.cer
```

```
@rem Set the paths to the client truststores signed by CA and signed by server key.
@rem You only need to define one of the trust stores.
@rem A trust store holds certificates that you trust,
@rem which are used to authenticate untrusted certificates.
@rem In this example, when the client authenticates the MQTT server it connects to,
@rem it authenticates the certificate it is sent by the server
@rem with the certificates in its trust store.
@rem For example, the MQTT server sends its server certificate,
@rem and the client authenticates it with either the same server certificate
@rem that you have stored in the cltsrvtruststore.jks trust store,
```

cleancert.bat

The commands in the `cleancert.bat` script delete the MQTT client queue manager to ensure that the server certificate store is not locked, and then delete all the keystores and certificates that are created by the sample security scripts.

```
@rem Delete the MQTT sample queue manager, MQXR_SAMPLE_QM
call "%MQ_FILE_PATH%\bin\setmqenv" -s
endmqm -i %qm%
dltmqm %qm%
```

```
@rem Erase all the certificates and key stores created by the sample scripts.
erase %cajkskeystore%
erase %cacert%
erase %srvjkskeystore%
erase %srvcertreq%
erase %srvcertcasigned%
erase %srvcertselfsigned%
erase %cltjkskeystore%
erase %cltp12keystore%
erase %cltpemkeystore%
erase %cltcertreq%
erase %cltcertcasigned%
erase %cltcertselfsigned%
erase %cltcajkstruststore%
erase %cltcap12truststore%
erase %cltcapemtruststore%
erase %cltsrvjkstruststore%
erase %cltsrvp12truststore%
erase %cltsrvpemtruststore%
erase %mqlog%
@echo Cleared all certificates
dir %certpath%\*.* /b
```

Figure 2. `cleancert.bat`

genkeys.bat

The commands in the `genkeys.bat` script create key-pairs for your private certificate authority, the server, and a client.

```
@rem
@echo ________________________________________________________________________________
@echo Generate %caalias%, %srvalias%, and %cltalias% key-pairs in %cajkskeystore%,
%srvjkskeystore%, and %cltjkskeystore%
@rem
@rem -- Generate a client certificate and a private key pair
@rem Omit this step, unless you are authenticating clients.
%keytool% -genkeypair -noprompt -alias %cltalias% -dname %cltdname% -keystore
%cltjkskeystore% -storepass %cltjkskeystorepass% -keypass %cltkeypass% -keyalg %algorithm%
-validity %validity%
```

```
@rem -- Generate a server certificate and private key pair
%keytool% -genkeypair -noprompt -alias %srvalias% -dname %srvdname% -keystore
%srvjkskeystore% -storepass %srvjkskeystorepass% -keypass %srvkeypass% -keyalg %algorithm%
-validity %validity%
```

```
@rem Create CA, client and server key-pairs
@rem  -- Generate a CA certificate and private key pair - The extension asserts this is a
certificate authority certificate, which is required to import into firefox
%keytool% -genkeypair -noprompt -ext bc=ca:true -alias %caalias%  -dname %cadname%
-keystore %cajkskeystore% -storepass %cajkskeystorepass%  -keypass %cakeypass% -keyalg
%algorithm% -validity %validity%
```

Figure 3. `genkeys.bat`

sscerts.bat

The commands in the `sscerts.bat` script export the client and server self-signed certificates from their keystores, and import the server certificate into the client truststore, and the client certificate into the server keystore. The server does not have a truststore. The commands create a client truststore in PEM format from the client JKS truststore.

```
@rem
@echo ________________________________________________________________________________
@echo Export self-signed certificates:  %srvcertselfsigned% and %cltcertselfsigned%
@rem Export Server public certificate
%keytool% -exportcert -noprompt -rfc -alias %srvalias% -keystore %srvjkskeystore%
-storepass %srvjkskeystorepass% -file %srvcertselfsigned%
@rem Export Client public certificate
@rem Omit this step, unless you are authenticating clients.
%keytool% -exportcert -noprompt -rfc -alias %cltalias% -keystore %cltjkskeystore%
-storepass %cltjkskeystorepass% -file %cltcertselfsigned%
```

```
@rem
@echo ________________________________________________________________________________
@echo Add selfsigned server certificate %srvcertselfsigned% to client trust store:
%cltsrvjkstruststore%
@rem Import the server certificate into the client-server trust store (for server self-
signed authentication)
%keytool% -import -noprompt -alias %srvalias% -file %srvcertselfsigned% -keystore
%cltsrvjkstruststore% -storepass %cltsrvjkstruststorepass%
```

```
@rem
@echo ________________________________________________________________________________
@echo Add selfsigned client certificate %cltcertselfsigned% to server trust store:
%srvjkskeystore%
@rem Import the client certificate into the server trust store (for client self-signed
authentication)
@rem Omit this step, unless you are authenticating clients.
%keytool% -import -noprompt -alias %cltalias% -file %cltcertselfsigned% -keystore
%srvjkskeystore% -storepass %srvjkskeystorepass%
```

```
@rem
@echo ________________________________________________________________________________
@echo Create a pem client-server trust store from the jks client-server trust store:
%cltsrvpemtruststore%
@rem
%keytool% -importkeystore -noprompt -srckeystore %cltsrvjkstruststore% -destkeystore
%cltsrvp12truststore% -srcstoretype jks -deststoretype pkcs12 -srcstorepass
%cltsrvjkstruststorepass% -deststorepass %cltsrvp12truststorepass%
%openssl%\bin\openssl pkcs12 -in %cltsrvp12truststore% -out %cltsrvpemtruststore% -passin
pass:%cltsrvp12truststorepass% -passout pass:%cltsrvpemtruststorepass%@rem
@rem
@echo ________________________________________________________________________________
@echo Create a pem client key store from the jks client keystore
@rem Omit this step, unless you are configuring a C or iOS client.
@rem
%keytool% -importkeystore -noprompt -srckeystore %cltjkskeystore%      -destkeystore
%cltp12keystore%     -srcstoretype jks -deststoretype pkcs12 -srcstorepass
%cltjkskeystorepass%     -deststorepass %cltp12keystorepass%
%openssl%\bin\openssl pkcs12 -in %cltp12keystore%     -out %cltpemkeystore%      -passin
pass:%cltp12keystorepass% -passout pass:%cltpemkeystorepass%
```

Figure 4. `sscerts.bat`

`cacerts.bat`

The script imports the certificate authority root certificate into the private keystores. The CA root certificate is needed to create the keychain between the root certificate and the signed certificate. The `cacerts.bat` script exports the client and server certificate requests from their keystores. The script signs the certificate requests with the key of the private certificate authority in the `cajkskeystore.jks` keystore, then imports the signed certificates back into the same keystores from which the requests came. The import creates the certificate chain with the CA root certificate. The script creates a client truststore in PEM format from the client JKS truststore.

```
@rem
@echo ________________________________________________________________________________
@echo Export self-signed certificates: %cacert%
@rem
@rem Export CA public certificate
%keytool% -exportcert -noprompt -rfc -alias %caalias%  -keystore %cajkskeystore% -storepass 
%cajkskeystorepass%  -file %cacert%
```

```
@rem
@echo ________________________________________________________________________________
@echo Add CA to server key and client key and trust stores: %srvjkskeystore%, 
%cltjkskeystore%, %cltcajkstruststore%,
@rem The CA certificate is necessary to create key chains in the client and server key 
stores,
@rem and to certify key chains in the server key store and the client trust store
@rem
@rem Import the CA root certificate into the server key store
%keytool% -import -noprompt -alias %caalias% -file %cacert% -keystore %srvjkskeystore% 
-storepass %srvjkskeystorepass%
@rem Import the CA root certificate into the client key store
@rem Omit this step, unless you are authenticating clients.
%keytool% -import -noprompt -alias %caalias% -file %cacert% -keystore %cltjkskeystore% 
-storepass %cltjkskeystorepass%
@rem Import the CA root certificate into the client ca-trust store (for ca chained 
authentication)
%keytool% -import -noprompt -alias %caalias% -file %cacert% -keystore %cltcajkstruststore% 
-storepass %cltcajkstruststorepass%
```

```
@rem
@echo ________________________________________________________________________________
@echo Create certificate signing requests: %srvcertreq% and %cltcertreq%
@rem
@rem Create a certificate signing request (CSR) for the server key
%keytool% -certreq -alias %srvalias% -file %srvcertreq% -keypass %srvkeypass% -keystore 
%srvjkskeystore% -storepass %srvjkskeystorepass%
@rem Create a certificate signing request (CSR) for the client key
%keytool% -certreq -alias %cltalias% -file %cltcertreq% -keypass %cltkeypass% -keystore 
%cltjkskeystore% -storepass %cltjkskeystorepass%
```

```
@rem
@echo ________________________________________________________________________________
@echo Sign certificate requests: %srvcertcasigned% and %cltcertcasigned%
@rem The requests are signed with the ca key in the cajkskeystore.jks keystore
@rem
@rem Sign server certificate request
%keytool% -gencert -infile %srvcertreq% -outfile %srvcertcasigned% -alias %caalias% 
-keystore %cajkskeystore% -storepass %cajkskeystorepass% -keypass %cakeypass%
@rem Sign client certificate request
@rem Omit this step, unless you are authenticating clients.
%keytool% -gencert -infile %cltcertreq% -outfile %cltcertcasigned% -alias %caalias% 
-keystore %cajkskeystore% -storepass %cajkskeystorepass% -keypass %cakeypass%
```

```
@rem
@echo ________________________________________________________________________________
@echo Import the signed certificates back into the key stores to create the key chain: 
%srvjkskeystore% and %cltjkskeystore%
@rem
@rem Import the signed server certificate
%keytool% -import -noprompt -alias %srvalias% -file %srvcertcasigned% -keypass %srvkeypass% 
-keystore %srvjkskeystore% -storepass %srvjkskeystorepass%
@rem Import the signed client certificate and key chain back into the client keystore
%keytool% -import -noprompt -alias %cltalias% -file %cltcertcasigned% -keypass %cltkeypass% 
-keystore %cltjkskeystore% -storepass %cltjkskeystorepass%
@rem
@rem The CA certificate is needed in the server key store, and the client trust store
@rem to verify the key chain sent from the client or server
@echo Delete the CA certificate from %cltjkskeystore%: it causes a problem in converting 
keystore to pem
@rem Omit this step, unless you are authenticating clients.
%keytool% -delete -alias %caalias% -keystore %cltjkskeystore% -storepass 
%cltjkskeystorepass%
```

```
@rem
@echo ________________________________________________________________________________
@echo Create a pem client-ca trust store from the jks client-ca trust store: 
%cltcapemtruststore%
```

mqcerts.bat

The script lists the keystores and certificates in the certificate directory. It then creates the MQTT sample queue manager and configures the secure telemetry channels.

```
@echo _______________________________________________________________________________
@echo List keystores and certificates
dir %certpath%\*.* /b
```

```
@rem
@echo Create queue manager and define mqtt channels and certificate stores
call "%MQ_FILE_PATH%\mqxr\Samples\SampleMQM" >> %mqlog%
echo DEFINE CHANNEL(%chlreq%)      CHLTYPE(MQTT) TRPTYPE(TCP) PORT(%portsslreq%)
SSLCAUTH(%authreq%) SSLKEYR('%srvjkskeystore%') SSLKEYP('%srvjkskeystorepass%')
MCAUSER(%mcauser%) | runmqsc %qm% >> %mqlog%
echo DEFINE CHANNEL(%chlopt%)      CHLTYPE(MQTT) TRPTYPE(TCP) PORT(%portsslopt%)
SSLCAUTH(%authopt%) SSLKEYR('%srvjkskeystore%') SSLKEYP('%srvjkskeystorepass%')
MCAUSER(%mcauser%) | runmqsc %qm% >> %mqlog%
echo DEFINE CHANNEL(%chlsslreqws%) CHLTYPE(MQTT) TRPTYPE(TCP) PORT(%portsslreqws%)
SSLCAUTH(%authreq%) SSLKEYR('%srvjkskeystore%') SSLKEYP('%srvjkskeystorepass%')
MCAUSER(%mcauser%) PROTOCOL(HTTP) | runmqsc %qm% >> %mqlog%
echo DEFINE CHANNEL(%chlssloptws%) CHLTYPE(MQTT) TRPTYPE(TCP) PORT(%portssloptws%)
SSLCAUTH(%authopt%) SSLKEYR('%srvjkskeystore%') SSLKEYP('%srvjkskeystorepass%')
MCAUSER(%mcauser%) PROTOCOL(HTTP) | runmqsc %qm% >> %mqlog%
echo DEFINE CHANNEL(%chlws%)       CHLTYPE(MQTT) TRPTYPE(TCP) PORT(%portws%)
MCAUSER(%mcauser%) PROTOCOL(HTTP) | runmqsc %qm% >> %mqlog%
@echo MQ logs saved in %mqlog%echo
```

Figure 6. `mqcerts.bat`

Get a copy of the IBM WebSphere MQ installation materials and a license in one of the following ways:

1. Ask your IBM WebSphere MQ administrator for the installation materials, and to confirm you can accept the license agreement.
2. Get a 90-day evaluation copy of IBM WebSphere MQ. See Evaluate: IBM WebSphere MQ.
3. Buy IBM WebSphere MQ. See IBM WebSphere MQ product page.

Secure the SSL channel with either certificate authority signed keys, or self-signed keys.

There is no installation program, you just expand the downloaded file.

1. Download the IBM Messaging Telemetry Clients SupportPac.
2. Create a folder where you are going to install the SDK.

   You might want to name the folder MQTT. The path to this folder is referred to here as *sdkroot*.
3. Expand the compressed IBM Messaging Telemetry Clients SupportPac file contents into *sdkroot*. The expansion creates a directory tree that starts at *sdkroot*\SDK.

There is a similar limitation for the MQTT client for Java. If the client code is running on a Java 1.6 JRE from IBM, the required SHA-2 cipher suites must be explicitly enabled. In order to use these suites, the client must also set the SSL context to a value that supports Version 1.2 of the Transport Layer Security (TLS) protocol. For example:

```
MqttConnectOptions mqttConnectOptions = new MqttConnectOptions();
java.util.Properties sslClientProps = new java.util.Properties();
sslClientProps.setProperty("com.ibm.ssl.keyStore", sslKeys.clientKeyStore);
sslClientProps.setProperty("com.ibm.ssl.keyStorePassword", sslKeys.clientStorePassword);
sslClientProps.setProperty("com.ibm.ssl.trustStore", sslKeys.clientKeyStore);
sslClientProps.setProperty("com.ibm.ssl.trustStorePassword", sslKeys.clientStorePassword);
sslClientProps.setProperty("com.ibm.ssl.protocol", "TLSv1.2");
sslClientProps.setProperty("com.ibm.ssl.enabledCipherSuites",
"SSL_RSA_WITH_AES_256_CBC_SHA256" );
mqttConnectOptions.setSSLProperties(sslClientProps);
```

On IBM WebSphere MQ, type the following command into a command window:

- Linux

```
echo 'DISPLAY CHSTATUS(SSL*) CHLTYPE(MQTT) ALL' | runmqsc MQXR_SAMPLE_QM
echo 'DISPLAY CHANNEL(SSL*) CHLTYPE(MQTT) ALL' | runmqsc MQXR_SAMPLE_QM
```

- Windows

```
echo DISPLAY CHSTATUS(SSL*) CHLTYPE(MQTT) ALL | runmqsc MQXR_SAMPLE_QM
echo DISPLAY CHANNEL(SSL*) CHLTYPE(MQTT) ALL | runmqsc MQXR_SAMPLE_QM
```

When you request a personal certificate, you specify a key size for the public and private key pair. The key size that is used during the SSL handshake can depend on the size stored in the certificate and on the CipherSpec:

- On z/OS z/OS, Windows, UNIX and Linux® systems, when a CipherSpec name includes _EXPORT, the maximum handshake key size is 512 bits. If either of the certificates exchanged during the SSL handshake has a key size greater than 512 bits, a temporary 512-bit key is generated for use during the handshake.
- On Windows, UNIX and Linux systems, when a CipherSpec name includes _EXPORT1024, the handshake key size is 1024 bits.
- Otherwise the handshake key size is the size stored in the certificate.

Introduction to IBM MQ migration

These topics explain the concepts that you must understand before planning migration tasks, where to find migration topics, and which migration paths are supported.

Migration paths

An overview of the migration paths between different IBM MQ versions. For some IBM MQ for z/OS migration paths, you can revert to the version you were using prior to migration. For other platforms, you cannot easily revert to a previous version.

Note: Before each new version of the product is released, it is tested for migration from earlier versions that are in support at that time. Direct migration from a version that is out of support might also work, but is neither tested nor supported. Therefore, to migrate to the latest version from a version that is out of support, you should first migrate to an interim version that was released before the earlier version went out of support.

Migration paths: IBM MQ on platforms other than z/OS

IBM i distributed

You can migrate from Version 7.0.1 or later direct to Version 8.0.

For IBM MQ for UNIX, Linux, and Windows, Side-by-side and Multi-stage migration were introduced in Version 7.1, and can be used to migrate from Version 7.0.1, Fix Pack 6 or later. Before you use either of these migration methods to migrate from Version 7.0.1, check that your Version 7.0.1 installation is at Fix Pack 6 or later.

Table 1. Migration paths: IBM MQ on platforms other than z/OS

From / To	Version 7.R	Version 8.0
Prior to Version 7.0.1	Follow the instructions for your current version. For more information about where to find the documentation for older versions of the product, see Documentation for older versions of IBM MQ	Direct migration is not possible. You must follow an indirect migration path, which involves migrating IBM MQ more than once.

Table 1. Migration paths: IBM MQ on platforms other than z/OS (continued)

From / To	Version 7.R	Version 8.0
Version 7.0.1 or later	Follow the instructions for your current version. See IBM MQ product information in IBM Documentation.	"Introduction to IBM MQ migration" on page 19 and "IBM MQ migration planning to the latest version on UNIX platforms, Windows, and IBM i" on page 80

For IBM MQ on platforms other than z/OS, you cannot easily revert to a previous version. However, if a queue manager has not been started, you can uninstall the current version and reinstall a different version of IBM MQ. It does not matter what versions of IBM MQ are installed between when a queue manager was last started and when it is next started.

Migration paths: IBM MQ for z/OS

z/OS

You can migrate from Version 7.0.1 or later direct to Version 8.0.

Table 2. Migration paths: IBM MQ for z/OS

From / To	Version 7.1	Version 8.0
Prior to Version 7.0.1	Follow the instructions on the Web for your current version. See the IBM MQ library page for further details.	Direct migration is not possible. You must follow an indirect migration path, which involves migrating IBM MQ more than once. See "Migrating from earlier unsupported releases of IBM MQ for z/OS" on page 143 for further information.
Version 7.0.1 or later	Follow the instructions for your current version. See IBM MQ product information in IBM Documentation.	"z/OS: Migration planning to the latest release" on page 81

To revert to a previous version, see "z/OS: Reverting a queue manager to a previous release" on page 155.

The version naming scheme for IBM MQ for z/OS

On IBM MQ for z/OS, releases have a three-digit Version, Release, and Maintenance (VRM) level code. The code is significant; it identifies the service life of a release. To run a queue manager at a different VRM level, you must migrate the queue manager, its applications, and the environment in which it runs. Depending on the migration path, the migration might require more or less effort.

The release level of IBM MQ for z/OS is described by a three-digit VRM code. The VRM acronym stands for:

```
Version . Release . Modification
```

`6.0.0`, `7.0.0`, `7.0.1`, `7.1.0`, and `8.0.0` are examples of IBM MQ for z/OS release level codes. On z/OS, a release of IBM MQ always has a three-digit VRM code, even if the release is the first release in a version, such as `7.0.0`. IBM MQ for z/OS follows a convention of changing the VRM when the product is installed by SMP/E with a new FMID.

You can modify existing libraries, without changing the FMID, by applying PTFs. You cannot upgrade existing libraries to an FMID or release level by applying PTFs.

The release level of a z/OS queue manager is written to the operator console in the message CSQY000I.

The command level of a queue manager is a three-digit VRM code. You can look at the queue manager command level in the queue manager property panel in IBM MQ Explorer. An IBM MQ program can call `MQINQ`, passing the `MQIA_COMMAND_LEVEL` selector, to obtain the command level of the queue manager it is connected to.

The VRM code, or release level, is significant in two respects. Changing the release level that a queue manager runs at, requires migration of the queue manager. It also requires attention to the PTF level of other queue managers that are in the same queue sharing group. It is also significant because each release level has its own service life, and end of service date.

The service life depends on the VRM. Each release level has its own service end date. So, for example, `7.0.0`, on z/OS, has a different service end date from `7.0.1`. See the Software lifecycle search results for "IBM MQ". Scroll through the search results to find your version of IBM MQ, then follow the link to find the service life and end of service dates.

Related concepts

"Upgrade, migration, and maintenance of IBM MQ on z/OS" on page 49
You can install new releases of IBM MQ to upgrade IBM MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same z/OS instance. Running a queue manager at a higher level requires migration. Maintenance differs from upgrading. To maintain a level of IBM MQ, you apply Program Temporary Fixes (PTFs) to the installed code.

The version naming scheme for IBM MQ (On platforms other than z/OS)

IBM MQ releases have a four-digit Version, Release, Modification, and Fix (VRMF) level code.

The full version of IBM MQ (On platforms other than z/OS) is described by a four-digit VRMF code.

The version and release parts of the code are significant; they identify the service life of a release. To run a queue manager at a different VR level, you must migrate the queue manager, its applications, and the environment in which it runs. Depending on the migration path, the migration might require more or less effort.

The VRMF acronym stands for:

```
Version . Release . Modification . Fix
```

`8.0`, `7.5.0.1`, and `7.0.1.0` are examples of full IBM MQ version codes.

You can find the full version level of an IBM MQ installation by typing the command **DSPMQVER**, or **DSPMQMVER** on IBM i. It returns the full four-digit VRMF code.

Versions and releases of IBM MQ are known by the first two digits of the VRMF code. The two digits are sometimes prefixed by a V, such as V`8.0`. A version of IBM MQ always has a release level, even if it is the first release in a version.

The first release is normally labeled V*x*`.0`, for example, IBM MQ V7.0. Occasionally, the first release of a version on a specific platform is not labeled V *x*`.0`. It is numbered to correspond the command level that has been implemented on the platform.

In documentation, the release level is sometimes dropped from the VRMF code, for example, V7. Dropping the release level can lead to ambiguity, if the context is not clear. For example, V7 might mean the whole of V7, or the release level V7.0, as opposed to the release level V7.2, or V7.3.

The third digit in the VRMF identifies the modification level of a release. A change in the third digit does not change the release. For example, after upgrading IBM MQ to modification level `7.0.1`, the release of IBM MQ remains 7.0. However the command level does change to `7.0.1`.

The significance of the distinction between release and modification level concerns migration, and the service life of a product. Queue manager objects, such as queue managers, channels, queues, and messages do not require migration to upgrade to a new modification level. Nor do they require migration if the modification level is removed [1] . Migration might be required for a version or release level change.

Note: IBM i distributed Backward migration is not possible. To be able to restore an earlier version or release level of a queue manager, you must back it up before upgrading. If you do restore it, you restore the queue manager, and its data, to the state it was in when you backed it up.

A new version or release has a new service end date. New modification levels generally do not result in a new service end date. But if a modification level is announced, then a new service end date might be announced too.

The fourth digit in the VRMF code is the fix level. Fix levels do not affect the command level of the queue manager. No migration is required, and fix levels do not affect the service end date of a release.

Trailing zeros in the VRMF code are never significant, but are sometimes quoted for clarity. For example, you might see `7.0.0` to distinguish it from `7.0.1`, and `7.0.1.0` to distinguish it from `7.0.1.1`. `7.0.0` is no different from `7.0` or `7.0.0.0`, and `7.0.1` and `7.0.1.0` are the same level.

Modification levels and fix levels are known by three and four-digit VRMF codes. `7.0.1` is a modification level and `7.0.1.2` is a fix level. Modification levels are shipped as refresh packs, and fix levels as fix packs.

A refresh or fix pack is named using a two part name that uniquely identifies it. The first part of the name is a truncated VRMF. The second part of the name is the name new refresh or fix pack. So, for example, the name of the fix pack `7.0.1.2` for Windows is `7.0.1-WS-MQ-Windows-FP0002`, and the name of the refresh pack `7.0.1` for Windows is `7.0-WS-MQ-Windows-RP0001`.

Refresh packs and fix packs for a particular version/release are cumulative, from the initial release. You can apply any higher numbered refresh, or fix pack, of the same version/release to upgrade directly to that version level. You do not have to apply the intervening fixes. Refresh packs and fix packs are obtained as service through Fix Central.

The latest modification level is also used to refresh the version of IBM MQ available through Electronic Software Download using Passport Advantage®, or on physical media.

When you order IBM MQ you receive a manufacturing refresh at the latest modification level. The result of installing a manufacturing refresh is almost the same as applying the refresh pack to an earlier fix level of IBM MQ. There is one important difference. Refresh packs are applied using a maintenance procedure, manufacturing refreshes are installed using an installation procedure. You can "unapply" a refresh pack to return to the previous fix level you had installed. You can only uninstall a manufacturing refresh, which removes IBM MQ from your system.

In addition to fixes packaged as refresh packs and fix packs, you can also obtain interim fixes for IBM MQ. You get these from Fix Central. Interim fixes are also known as emergency or test fixes, and are known collectively as interim fixes. The naming scheme for refresh and fix packs extends to interim fixes. Interim fixes are known either by their fix name, or by the list of APARs they fix.

When you apply new fix packs or refresh packs, all interim fixes are removed. The documentation with the fix pack or refresh pack tells you if the APARS associated with the interim fixes you have applied have been fixed. If they have not, check to see if there are new interim fixes, at the new level, for the APARs that concern you. If there are not, consult service. They might either tell you to reapply the interim fix, or supply a new interim fix.

Related concepts

“Upgrade, migration, and maintenance of IBM MQ (On platforms other than z/OS)” on page 51
You can install new releases of IBM MQ to upgrade IBM MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of IBM MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

[1] Applications using new functions introduced in a modification level do not work on an earlier level.

Internet Protocol Version 6 (IPv6) migration

This section deals with using IPv4 and IPv6 when you are thinking of installing IBM MQ

General Introduction

The Internet Protocol Version 6 (IPv6) is designed by the Internet Engineering Task Force (IETF) to replace the current version Internet Protocol, Version 4 (IPv4). IPv4 has been around for over 20 years and is one of the primary methods for machines to communicate to each other over the internet. IPv4 is limited to 32-bit addressing for internet addresses. These addresses are needed by all new machines added to the internet and they are beginning to run out. The IETF is the controlling standards body for the Internet and to meet the growing demand for internet addresses has increased the number of digits used for Internet addresses from 32 to 128 bits. IPv6 offers a far larger number (2^{128}) of internet addresses and should solve the address shortage for the foreseeable future. IPv6 is expected to gradually replace IPv4, with the two protocols coexisting for a number of years while this transition period exists. IPv6 also simplifies header formats and improves support for extensions and options, flow labeling capability, and consolidated authentication and privacy capabilities

IBM MQ has the ability for queue managers to communicate using the IPv6 protocol in addition to the existing, IPv4, protocol.

Further information on IPv6 can be found at IPv6.

IBM MQ platforms that support IPv6

This section lists the IBM MQ platforms that support IPv6.

IPv6 is supported on the following IBM MQ platforms:

- IBM MQ for AIX®
- IBM MQ for Linux
- IBM MQ for Sun Solaris
- IBM MQ for HP-UX
- IBM MQ for Windows
- IBM MQ for IBM i
- IBM MQ for z/OS

Key points in migrating to IPv6 and using IBM MQ

This section lists some key points to be aware of when you are thinking of installing IBM MQ and using IPv6.

- IBM MQ recognizes IPv6 hexadecimal addresses (for example fe80:43e4:0204:acff:fe97:2c34:fde0:3485) as well as IPv4 dotted decimal addresses (for example 9.20.9.30).
- For a system running both IPv4 and IPv6 system, the connection name (CONNAME) you specify for a given channel determines the IP protocol for the channel making a connection.

Considerations when implementing IPv6 in a network

This section lists some things that you should consider when you are thinking of installing IBM MQ on an IPv6 network.

- To ensure consistency across the network, you should plan the introduction of IPv6 for the whole network, especially where clusters are involved. For example, although a queue manager is now IPv6 capable, this doesn't imply that the queue managers it can communicate with are also IPv6 capable.
- When setting the domain name server (DNS) or equivalent, consider whether the system on which the target queue manager is running can resolve to an IPv4 address, an IPv6 address or a dual IPv4 and IPv6 address.

- If the system that you are installing IBM MQ on does not support IPv6, IBM MQ will only be able to connect using IPv4.
- For a queue manager running on an IPv6 enabled system to be able to communicate with a queue manager running on an IPv4 enabled system, the IPv4 enabled system must have a hostname that resolves to an IPv4 address only.
- If there are multiple domain name servers in an IBM MQ network, each hostname used in a channel definition must resolve to the same address (or addresses), regardless of which DNS is used.

Migrating a queue manager to IPv6

This section deals with migrating a queue manager when you are thinking of installing IBM MQ on an IPv6 network.

The IPv6 protocol can only be utilized by IBM WebSphere MQ 6.0 or later. In order to make use of the IPv6 protocol, IBM MQ must be installed on a system that is IPv6 capable.

The preferred IP version that two systems use for communicating (if both IPv4 and IPv6 are available) is determined by a new queue manager attribute IPADDRV. This parameter only has an effect if the hostname resolves ambiguously to both an IPv4 address and an IPv6 address.

To migrate a queue manager to use the IPv6 protocol:

1. Configure dual IPv4 and IPv6 protocols on the system where the queue manager to be migrated resides.
2. Install IBM MQ.
3. Add an entry to the DNS to resolve the hostname of the system that is to be migrated, to both an IPv4 address and an IPv6 address.
4. Set the IPADDRV parameter to IPv6 (or set the LOCLADDR parameter to resolve to an IPv6 address).

> **CAUTION:** Not all IPv6 software can interpret an IPv4 mapped IPv6 address. If the combination of CONNAME and LOCLADDR results in an IPv4 mapped IPv6 address, ensure that the system hosting the target queue manager is capable of handling this.
>
> Using mapped addresses can require protocol translators in the IP network.

Migration scenarios (non-cluster topology)

It is possible to come up with a number of different interconnection possibilities, and the following sections aim to help you understand how IBM MQ will work in each case.

Non-cluster migration scenario 1

Three systems exist that are IPv4 only capable. Each system hosts a queue manager (QM1, QM2, and QM3) and each queue manager connects to the other two. All CONNAMEs in the cluster channel definitions are made using DNS names rather than IP addresses.

Enable QM1 to be able to use channels running over IPv6 as follows

1. Upgrade the host system to have dual IPv4 and IPv6 stacks.

   Important: A listener is required for each IP stack.
2. Install the latest version of IBM MQ.
3. Update the DNS table so that it has two entries for the system running QM1; one entry for its IPv4 address and one for its IPv6 address. This enables a DNS name request to return both IPv4 and IPv6 addresses for this host.
4. Set the queue manager IPADDRV attribute to IPv6.

Note: Even with these changes made to support IPv6 addressing, QM1 will still be able to communicate with queue managers (both existing and new ones) that are only IPv4 capable.

Enable QM2 to be able to use channels running over IPv6 as for QM1 above.

- Communications between QM1 and QM2 will now be over IPv6.
- Communications between QM1 and QM3 will still be over IPv4.
- Communications between QM2 and QM3 will still be over IPv4.

With the queue manager IPADDRV attribute set to IPv6, the preference has been set for the queue manager to connect using the IPv6 protocol. If a channel from QM1 to QM3 has LOCLADDR set to a host name which resolves to an IPv6 address, or both IPv4 and IPv6 addresses (with the IPADDRV attribute set to IPv6, the IPv6 address will be returned as that is the preference), this channel will attempt to use the IPv6 protocol. If the IPv6 protocol installed on the QM1 host system is capable of using a mapped address then QM1 will communicate with QM3 over IPv6. Otherwise, the channel will fail to resolve CONNAME.

While QM3 remains a queue manager on an earlier version of the product, you will need to check that all CONNAMEs used to start a channel to QM3 do not resolve to an IPv6 address or dual IPv4 and IPv6 addresses where the IPv6 address could be returned. This would cause QM1 to attempt to start the channel over IPv6 which would fail, as it would be unable to resolve the CONNAME.

It is possible to upgrade a system to have dual IPv4 and IPv6 capability and still run a queue manager on an earlier version of the product, on the system. While it is not recommended to run this type of configuration, as long as the addresses that are returned to this level of queue manager are either IPv4 or an IPv4 mapped version of an IPv6 address, this should work.

Non-cluster migration scenario 2

Three systems exist that are IPv4 only capable. Each system hosts a queue manager (QM1, QM2, and QM3) and each queue manager connects to the other two. All CONNAMEs in the cluster channel definitions are made using IP addresses.

Because addresses have been specified instead of DNS names, to allow a queue manager to connect to another using the IPv6 protocol you will need to duplicate the definitions that use IPv4 addresses between them and provide them with IPv6 addresses instead. The original definitions that use IPv4 addresses will continue to work, but if you intend to take advantage of the IPv6 protocol, you will need to connect using the new definitions.

Enable QM1 to be able to use channels running over IPv6 as follows

1. Upgrade the host system to have dual IPv4 and IPv6 stacks.

   Important: A listener is required for each IP stack.
2. Install IBM MQ.
3. Duplicate the channel, transmission queue and, where applicable, any process definitions using IPv6 addresses where required.

Note: Even with these changes made to support IPv6 addressing, QM1 will still be able to communicate with existing queue managers that are only IPv4 capable.

Enable QM2 to be able to use channels running over IPv6 as for QM1 above.

1. Upgrade the host system to have dual IPv4 and IPv6 stacks.

   Important: A listener is required for each IP stack.
2. Install IBM MQ.
3. Where necessary amend applications to write to the new remote queue (created above for QM1 with the IPv6 addresses).
4. Verify the channels can be started.

The queue managers can now connect as follows:

- QM1 can now connect with QM2 over either IPv4 or IPv6 depending on the channel the application writes its messages to.
- QM1 still connects with QM3 over IPv4 using the original definitions.

Migrating a cluster to IPv6

This section deals with migrating clusters when you are thinking of installing IBM MQ on an IPv6 capable network.

The following gives an overview of approaches that can be taken when migrating a cluster to the latest version of IBM MQ. Due to the variations that can occur within a cluster, the detail is deliberately general and should only be seen as a guide to the likely course of action you will need to take.

Migration scenarios (cluster topology)

Where an IPv6 capable system is to be added to an IBM MQ cluster, all full repository systems in that cluster must be IPv6 capable.

The following scenarios are seen as the ones most likely to occur in customer installations. They describe the changes that are likely to be required.

Scenario 1

A cluster from an earlier version of the product is installed on IPv4 only capable, systems and you need to connect an IPv6 only capable system into the cluster. All CONNAMEs in cluster channel definitions are made using DNS names rather than IP addresses.

When adding a new IPv6 only system to the cluster, identify those queue managers that your new system will communicate with. These include:

- The queue managers your new system will send messages to.
- The queue managers your new system will receive messages from.
- The full repository queue managers

The systems that you have identified must be upgraded before introducing the new system.

Recommended migration procedure:

- Upgrade each of the systems hosting a full repository queue manager as shown in "Migrating a queue manager to IPv6" non-cluster scenario 1.
- Upgrade the remaining cluster systems which need to be IPv6 capable as shown in "Migrating a queue manager to IPv6" non-cluster scenario 1.

With this configuration:

- The new IPv6 only capable system will communicate with the cluster using IPv6 addressing
- All other IPv4 systems that connect into the cluster will continue to communicate using IPv4 addressing
- The systems in the cluster will be able to connect to each other using either IPv4 or IPv6 addressing. The decision as to which address is used depends on whether you have set IPADDRV to specify IPv4 or IPv6 connections.

Scenario 2

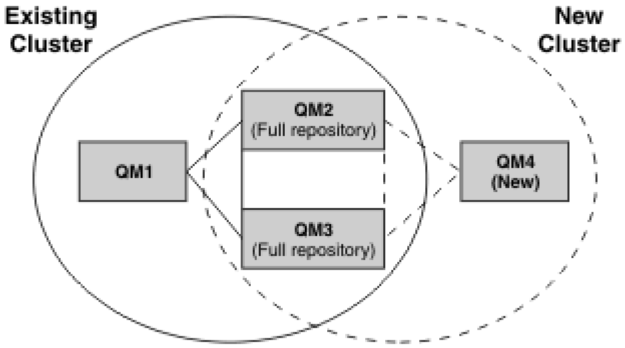
A cluster from an earlier version of the product is installed on IPv4 only capable systems and you need to connect an IPv6 only capable system into the cluster. Your network does not support adding both IPv6 and IPv4 addresses using the same hostname or you are using IP addresses rather than DNS names in the cluster channel CONNAMEs.

The problem here is likely to be that all of the systems cannot be switched to IPv6 simultaneously and some at least must remain only IPv4 capable. The systems that your new IPv6 only system communicates with must be IPv4 and IPv6 capable. We do not recommend simply adding a new set of IPv6 channels into the cluster for the IPv6 system to use, as the IPv4 system would also try to use them, resulting in communication errors.

The recommended approach is:

- Define a new cluster which contains the IPv6 only capable system or systems with new IPv6 addresses and channel definitions. The existing cluster remains, and contains the IPv4 only system definitions. The image below gives a pictorial representation of this. QM1, QM2, and QM3 represent the original IPv4 cluster. QM2, QM3, and QM4 represent the new cluster created to allow the IPv6 only capable system (QM4) to connect into your configuration.

- If you are using DNS names, you can give each of the systems separate DNS names for IPv4 and IPv6 (for example system1_IPv4.ibm.com and system1_IPv6.ibm.com).
- Define a new CLUSRCVR channel and any corresponding CLUSSDR channels using the new IPv6 names or IP addresses on each system in the new cluster. In this way the systems with only IPv4 or IPv6 capability do not see channels which they are not able to use and no communications error will result.



Note: There are both IPv4 and IPv6 definitions connecting the full repositories so that definitions for both new and existing cluster definitions are replicated between them. Also be aware that the queue managers QM1 and QM4 cannot communicate directly because they do not share a common network. They could communicate indirectly, for example by using ALIAS queues defined in the queue managers QM2 and QM3. In the configuration shown above you would need to pay attention to the ordering of application messages flowing between QM2 and QM3 because multiple routes exist, if this is relevant you could use BIND_OPEN to fix the route.

Abbreviated migration scenarios

This section gives some abbreviated scenarios for when you are thinking of installing clusters on IBM MQ

Abbreviated scenarios: Effects of CONNAME and LOCLADDR settings

The following table provides an overview of what will occur for the different TCP/IP stacks (IPv4 only, IPv6 only and dual IPv4 and IPv6 stacks) and given the settings for CONNAME and LOCLADDR the expected connection result.

Note: Using mapped addresses can require protocol translators in the IP network.

Table 3. Effects of CONNAME and LOCLADDR settings

Stack Type	CONNAME setting	LOCLADDR setting	Connection result
IPv4 only stack	IPv4 address		Channel binds to IPv4 stack
	IPv6 address		Channel fails to resolve CONNAME
	Host name resolves to both IPv4 and IPv6 addresses		Channel binds to IPv4 stack
	IPv4 address	IPv4 address	Channel binds to IPv4 stack
	IPv6 address	IPv4 address	Channel fails to resolve CONNAME
	Host name resolves to both IPv4 and IPv6 addresses	IPv4 address	Channel binds to IPv4 stack

Table 3. Effects of CONNAME and LOCLADDR settings (continued)

Stack Type	CONNAME setting	LOCLADDR setting	Connection result
	Any address	IPv6 address	Channel fails to resolve LOCLADDR
	IPv4 address	Host name resolves to both IPv4 and IPv6 addresses	Channel binds to IPv4 stack
	IPv6 address	Host name resolves to both IPv4 and IPv6 addresses	Channel fails to resolve CONNAME
	Host name resolves to both IPv4 and IPv6 addresses	Host name resolves to both IPv4 and IPv6 addresses	Channel binds to IPv4 stack
Dual IPv4 and IPv6 stack	IPv4 address		Channel binds to IPv4 stack
	IPv6 address		Channel binds to IPv6 stack
	Host name resolves to both IPv4 and IPv6 addresses		Channel binds to stack determined by IPADDRV
	IPv4 address	IPv4 address	Channel binds to IPv4 stack
	IPv6 address	IPv4 address	Channel fails to resolve CONNAME
	Host name resolves to both IPv4 and IPv6 addresses	IPv4 address	Channel binds to IPv4 stack
	IPv4 address	IPv6 address	Maps an IPv4 CONNAME to an IPv4 mapped IPv6 address. IPv6 implementations that do not support IPv4 mapped IPv6 addressing fail to resolve CONNAME
	IPv6 address	IPv6 address	Channel binds to IPv6 stack
	Host name resolves to both IPv4 and IPv6 addresses	IPv6 address	Channel binds to IPv6 stack
	IPv4 address	Host name resolves to both IPv4 and IPv6 addresses	Maps an IPv4 CONNAME to an IPv4 mapped IPv6 address. IPv6 implementations that do not support IPv4 mapped IPv6 addressing fail to resolve CONNAME
	IPv6 address	Host name resolves to both IPv4 and IPv6 addresses	Channel binds to IPv6 stack
	Host name resolves to both IPv4 and IPv6 addresses	Host name resolves to both IPv4 and IPv6 addresses	Channel binds to IPv6 stack

Table 3. Effects of CONNAME and LOCLADDR settings (continued)

Stack Type	**CONNAME setting**	**LOCLADDR setting**	**Connection result**
IPv6 only stack	IPv4 address		Maps an IPv4 CONNAME to an IPv4 mapped IPv6 address. IPv6 implementations that do not support IPv4 mapped IPv6 addressing fail to resolve CONNAME
	IPv6 address		Channel binds to IPv6 stack
	Host name resolves to both IPv4 and IPv6 addresses		Channel binds to IPv6 stack
	Any address	IPv4 address	Channel fails to resolve LOCLADDR
	IPv4 address	IPv6 address	Maps an IPv4 CONNAME to an IPv4 mapped IPv6 address. IPv6 implementations that do not support IPv4 mapped IPv6 addressing fail to resolve CONNAME
	IPv6 address	IPv6 address	Channel binds to IPv6 stack
	Host name resolves to both IPv4 and IPv6 addresses	IPv6 address	Channel binds to IPv6 stack
	IPv4 address	Host name resolves to both IPv4 and IPv6 addresses	Maps an IPv4 CONNAME to an IPv4 mapped IPv6 address. IPv6 implementations that do not support IPv4 mapped IPv6 addressing fail to resolve CONNAME
	IPv6 address	Host name resolves to both IPv4 and IPv6 addresses	Channel binds to IPv6 stack
	Host name resolves to both IPv4 and IPv6 addresses	Host name resolves to both IPv4 and IPv6 addresses	Channel binds to IPv6 stack

Abbreviated scenarios: System configurations

Table 5 on page 30 gives a number of abbreviated scenarios based on the configuration of the installed queue managers and the IP configuration they are running on. The list is not intended to be exhaustive, but to give a number of examples of what to expect based on the configurations shown.

The abbreviations are combined in Table 5 on page 30 to give the configuration of the systems involved in trying to establish communication. For example:

- v71 + IPv6: Represents a queue manager from an earlier version of the product on a system with a TCP/IP Version 6 stack
- v8 + Dual: Represents a queue manager from the latest version of the product on system with a dual TCP/IP version 4 and Version 6 stack

Table 4. Abbreviations used in system configurations

Abbreviation	Meaning
v71	queue manager from an earlier version of the product
v8	queue manager from the latest version of the product
IPv4	a system using an IPv4 only stack
IPv6	a system using an IPv6 only stack
Dual	a system using both an IPv4 and an IPv6 stack
IPv4DNS	DNS returns an IPv4 address only for hostname of system holding the responding queue manager
IPv6DNS	DNS returns an IPv6 address only for hostname of system holding the responding queue manager
DualDNS	DNS returns an IPv4 and IPv6 address for hostname of system holding the responding queue manager
LOCLADDR4	The LOCLADDR parameter is set to IPv4 addressing
LOCLADDR6	The LOCLADDR parameter is set to IPv6 addressing
IPADDR4	IPADDRV is set to IPv4 addressing
IPADDR6	IPADDRV is set to IPv6 addressing

Table 5. System configurations

Originating queue manager		Responding queue manager			Result
Queue manager and Stack	**LOCLADDR**	**IPADDRV**	**Queue Manager and Stack**	**DNS Return**	
v71 + IPv6	Any	Not applicable			IP Error
v71 + IPv4 or v71 + Dual	Both LOCLADDR4 & LOCLADDR6	Not applicable	v71 + IPv4 or v71 + Dual	IPv4DNS or DualDNS	IPv4 connection can be established
v71 + IPv4 or v71 + Dual	Blank or LOCLADDR4	Not applicable	v71 + IPv4 or v71 + Dual	IPv4DNS or DualDNS	IPv4 connection can be established
v71 + IPv4 or v71 + Dual	Blank or LOCLADDR4	Not applicable	v71 + Dual	IPv6DNS	Unable to resolve CONNAME
v71 + IPv4 or v71 + Dual	Blank or LOCLADDR4	Not applicable	v71 + Dual or v8 + Dual v8 + IPv4	IPv4DNS or DualDNS	IPv4 connection can be established

Table 5. System configurations (continued)

Originating queue manager		**Responding queue manager**			**Result**
Queue manager and Stack	**LOCLADDR**	**IPADDRV**	**Queue Manager and Stack**	**DNS Return**	
v71 + IPv4 or v71 + Dual	LOCLADDR6	Not applicable			IP Error
v71 + IPv4 or v71 + Dual	Blank or LOCLADDR4 or both LOCLADDR4 & LOCLADDR6	Not applicable	v8 + IPv6	IPv6DNS	Unable to resolve CONNAME
v8 + IPv4	Blank or LOCLADDR4	Not specified	v71 + IPv4 or v71 + Dual or v8 + IPv4	IPv4DNS or DualDNS	IPv4 connection can be established
v8 + IPv4	LOCADD6	Not specified			Unable to resolve LOCLADDR
v8 + IPv4	Blank or LOCLADDR4	Not specified	v8 + IPv6	IPv6DNS	Unable to resolve CONNAME
v8 + IPv6	Blank or LOCLADDR6	Not specified	v71 + Dual	DualDNS	Attempts to start IPv6 channel and fails as there will be no IPv6 listener available
v8 + IPv6	Blank or LOCLADDR6	Not specified	v71 + IPv4	IPv4DNS	Attempts to start IPv6 channel and fails as there will be no IPv6 listener available
v8 + IPv6 or v8 + Dual	LOCLADDR6	Blank or IPADDR6	v8 + IPv6 or v8 + Dual	IPv6DNS or DualDNS	IPv6 connection can be established
v8 + Dual	LOCLADDR6	IPADDR4	v8 + Dual	IPv4DNS or DualDNS	IPv6 connection can be established where mapped addressing can be used
v8 + Dual	Blank or LOCLADDR4	IPADDR4	v71 + Dual	IPv4DNS or DualDNS	IPv4 connection can be established
v8 + Dual	Both LOCLADDR4 & LOCLADDR6	Blank or IPADDR4	v71 + Dual	IPv4DNS or DualDNS	IPv4 connection can be established

Table 5. System configurations (continued)

Originating queue manager		**Responding queue manager**			**Result**
Queue manager and Stack	**LOCLADDR**	**IPADDRV**	**Queue Manager and Stack**	**DNS Return**	
v8 + Dual	LOCLADDR4	IPADDR4			Unable to resolve LOCLADDR
v8 + Dual	LOCLADDR6 or both LOCLADDR4 & LOCLADDR6	Blank or IPADDR6	v8 + IPv6 or v8 + Dual	IPv6DNS or DualDNS	IPv6 connection can be established

Overview of migration methods

An overview of the various methods of migrating IBM MQ from one release to another are explained, together with a brief description of the difference between maintenance, migration, and upgrading.

Maintenance, migration and upgrading

How IBM MQ uses this terminology:

- Migration is the process of updating queue manager data to match a newer level of code. This occurs the first time a queue manager is started with the newer level of code.
- Maintenance is the application of a fix pack, interim fix or program Temporary Fix (PTF). The installation can be restored to its previous level and queue managers or applications continue to work. Migration is not required after applying maintenance. However, you should test applications with the new level of IBM MQ code.
- Upgrading is the process of taking an existing IBM MQ installation and upgrading to a new level of code. Unless the upgrade is applying a fix (and not enabling new function), an upgrade must be followed by migration.
- Once migration has occurred, the queue manager can no longer be started by an earlier code level. Queue manager migration is not reversible z/OS (except on z/OS, with important restrictions).

Notes:

1. The example commands in these topics apply to UNIX and Linux, and Windows platforms only.
2. IBM i Side-by-side migration has a different meaning on IBM i. See "Installation methods on IBM i" on page 113 for further information.
3. IBM i Multiple installations are not applicable to IBM i.

Single stage migration

Single-stage migration is the term used to describe replacing the only installation of IBM MQ on a server, with a later release.

Until IBM WebSphere MQ 7.1, single-stage was the only migration scenario. Single-stage migration preserves existing scripts and procedures for running IBM MQ the most. With other migration scenarios you might change some scripts and procedures, but you can reduce the effect queue manager migration has on users.

See "UNIX, Linux, and Windows: Single-stage migration to a later version" on page 92 for information on carrying out this method of migration.

Side-by-side migration

Side-by-side migration is the term used to describe installing a new version of IBM MQ alongside an older version on the same server.

Queue managers continue running, and remain associated with the older version of IBM MQ, during the installation and verification of the new version of IBM MQ.

When you decide to migrate queue managers to the new version of IBM MQ, you stop all queue managers, migrate them all to the new version, and uninstall the old version of IBM MQ.

The side-by-side migration scenario is less flexible than the multistage migration. The advantage the side-by-side scenario has over the single-stage scenario is that you can install and verify the new IBM MQ installation on the server before switching over to it.

The process is based on the following premise:

- Install additional IBM MQ code alongside existing installation while queue managers are still running.
- Move queue managers one at a time to the new installation.
- Migrate and test applications one at a time.

With the side-by-side approach, you can assign a later version of IBM MQ to be the primary installation, whereas, with the multistage approach, you cannot do so, if you have IBM WebSphere MQ 7.0.1 installed, until you uninstall IBM WebSphere MQ 7.0.1.

With a later version of IBM MQ set as the primary installation, many applications restart without having to re-configure their environment, and IBM MQ commands work without providing a local search path.

See “UNIX, Linux, and Windows: Side-by-side migration to a later version” on page 96 for information on carrying out this method of migration.

Multistage migration

Multistage migration is the term used to describe running a new version of IBM MQ alongside an older version on the same server.

After installing the new version alongside the old, you can create new queue managers to verify the new installation, and develop new applications.

At the same time, you can migrate queue managers and their associated applications from the old version to the new. By migrating queue managers and applications one-by-one, you can reduce the peak workload on your staff managing the migration.

See “UNIX, Linux, and Windows: Multi-stage migration to a later version” on page 100 for information on carrying out this method of migration.

Multiple IBM MQ installations

IBM WebSphere MQ 7.1 introduced support for multiple IBM MQ installations on Windows, AIX, HP-UX, Linux, and Solaris (but not IBM i) and gives you the option to install and select between one or more IBM MQ installations.

z/OS

▶ z/OS

On z/OS multiple releases can exist, but the information within this topic does not apply to z/OS.

Use STEPLIBs to control which level of IBM MQ is used.

For further information, see “Coexistence” on page 174.

Overview

You can select between:

- Simplicity of maintaining a single IBM MQ installation.
- Flexibility, by allowing up to a maximum of 128 IBM MQ installations on a system. The first of these 128 installs is reserved for an (optional) IBM WebSphere MQ 7.0.1 installation of level 7.0.1.6 or above. The other installations must be IBM WebSphere MQ 7.1 or above.

You can install multiple copies of the same code level; this is especially convenient for maintenance purposes.

For example, if you want to upgrade IBM WebSphere MQ 7.1.0.0 to IBM WebSphere MQ 7.1.0.1, you can install a second copy of IBM WebSphere MQ 7.1.0.0, apply the maintenance to bring it to IBM WebSphere MQ 7.1.0.1, and then move the queue managers across to the new installation.

You still have the original installation, so it is a simple matter to move the queue managers back if you encounter any problems.

Notes:

1. Solaris Linux On Linux and Solaris only, you must ensure that each package installed has a unique name.

   You need to use a tool to create a unique set of packages:

   - `$ crtmqpkg PACKAGE_SUFFIX`
   - This takes the IBM MQ installation packages, and repackages them with a new name of your choice. You then install as usual.
2. All installations share a data directory; this is where `mqs.ini` is located for example.
3. All installations share the same namespace for queue managers. This means that you cannot create several queue managers of the same name in different installations.
4. IBM MQ installations are fully relocatable; each installation has a separate installation path. You can choose where you would like to install IBM MQ.
5. IBM MQ resources have installation-scope resource isolation, so operations on one installation do not affect the others.

   This means that the resources created by one installation are isolated from those created by other installations. It enables actions, such as removing an installation of IBM MQ, while queue managers are running under another installation.
6. Queue managers are "associated" with an installation You can move them, but you cannot migrate data back to earlier releases.

Working with multiple installations

To work with a queue manager, you need to use the commands from its installation. If you select the wrong installation, you see:

```
AMQ5691: Queue manager 'MYQM' is associated with a different installation (Inst1)
```

To work with a queue manager, you have to use the control commands from its associated installation. You have a choice of:

- Using the full path to the control commands, for example:

  ```
  $ MQ_INSTALLATION_PATH\bin\strmqm MYQM
  ```

  or
- Setting the environment variables for an installation with one of:

```
$ MQ_INSTALLATION_PATH/bin/setmqenv 's
$ setmqenv -m MYQM
$ setmqenv -n InstallationName
$ setmqenv -p MQ_INSTALLATION_PATH
```

You might consider using a shell script or batch file to set up the environment for each IBM MQ installation. You can use the **setmqenv** or **crtmqenv** commands to help with this.

- setmqenv sets the values of the environment variables, such as PATH, CLASSPATH and LD_LIBRARY_PATH, for use with an IBM MQ installation.
- crtmqenv creates a list of the environment variables and their values for use with a particular IBM MQ installation. You can then use this list to incorporate into a shell script or batch file.

Commands

In general, you must use the commands for the correct installation. However, there are some exceptions:

- Commands that work across installations
  - dspmq
  - dspmqinst
  - dspmqver
  - setmqinst
- New control commands for multiple installations
  - crtmqenv
  - dspmqinst
  - setmqenv
  - setmqinst
  - setmqm

Primary installation

Explanation of the IBM MQ primary installation.

Overview

The primary installation is:

- The installation to which system-wide locations refer
- Optional, but convenient

Note: Prior to IBM WebSphere MQ 7.1, the only installation was the primary installation. However, if you uninstall IBM WebSphere MQ 7.0.1, and then install IBM WebSphere MQ 7.1, the installation is not by default the primary installation.

UNIX systems

The primary installation:

- Has symbolic links in `/usr/lib` and `/usr/bin`

  If you have not set a primary installation there are no symbolic links.
- Must be configured manually using the following command:

  ```
  $ MQ_INSTALLATION_PATH/bin/setmqinst -i -p MQ_INSTALLATION_PATH
  ```

To locate your various installations, you can use the:

- Platform installation tools to query what is installed where on the system
- dspmqver command
- dspmqinst command
- Following command to list the installations:

```
cat /etc/opt/mqm/mqinst.ini
```

Windows

The primary installation is:

- By default the first installation.
- Pointed to by global environment variables.
- Used by some operating system features that require central registration of interface libraries.

  For example, .NET monitor (transactional mode) and COM/ActiveX interface classes.

To locate your various installations, you can use the:

- Platform installation tools to query what is installed where on the system
- dspmqver command
- dspmqinst command
- Following command to query the registry:

```
reg.exe query "HKLM\Software\[Wow6432Node\]IBM\WebSphere MQ\Installation" /s
```

Migration of Windows registry information

IBM WebSphere MQ 7.1 onwards uses `mqs.ini` and `qm.ini`.

However, if IBM WebSphere MQ 7.0.1 is still installed, IBM WebSphere MQ 7.1 uses the registry.

To uninstall IBM WebSphere MQ 7.0.1 with IBM WebSphere MQ 7.1 already installed, you must:

- End all the queue managers and listeners
- Uninstall IBM WebSphere MQ 7.0.1 which migrates the registry information into `mqs.ini` and `qm.ini`.
- IBM WebSphere MQ 7.1 now starts using `mqs.ini` instead.

Note: If for some reason the uninstall is interrupted, and the automatic migration does not run, a tool is provided to migrate the information out of the registry manually.

Multiple installations and application programs

When a local application connects to a queue manager, the application needs to load the libraries from the installation associated with the queue manager. Multiple installations introduce some complexity.

Using the `setmqm` command

When you use setmqm to change the installation associated with a queue manager, the libraries that need to be loaded change.

When an application connects to multiple queue managers owned by different installations, multiple sets of libraries need to be loaded.

Note: If you link your applications to IBM WebSphere MQ 7.1 (or later) libraries, the applications automatically load the appropriate libraries when the application connects to a queue manager.

Loading IBM MQ libraries in a multi-version environment

How libraries are located depends upon your environment.

If IBM WebSphere MQ 7.1 (or later) is installed in the default location, existing applications continue to work as before. Otherwise, you might need to rebuild the application or change your configuration.

The order in which libraries are searched, depends upon the platform you are using:

- Windows
  - The application's directory
  - The current directory
  - The global and your PATH variables
- Other platforms
  - LD_LIBRARY_PATH (or LIBPATH/SHLIB_PATH)
  - An embedded search path (RPath)
  - The default library path

Table 6. Options for loading libraries

Platform	Option	Benefits	Drawbacks
UNIX systems	Set/change the embedded runtime search path (RPath)	The path is explicit in the way the application is built	You need to recompile and link If you move IBM MQ, you must change RPath
UNIX systems	Set LD_LIBRARY_PATH or equivalent using setmqenv	Overrides RPath No changes to existing applications Easy to change if you move IBM MQ	Depends on environment variables Possible impacts on other applications
Windows	Set PATH using setmqenv	No changes to existing applications Easy to change if you move IBM MQ	Depends on environment variables Possible impacts on other applications
All	Set the primary installation to IBM WebSphere MQ 7.1 or later	No changes to existing applications Easy to change the primary installation Similar behavior to previous versions of IBM MQ	While IBM WebSphere MQ 7.0.1 is installed, you cannot make IBM WebSphere MQ 7.1 the primary installation UNIX systems: Relies on `/usr/lib` in the default search path

Maintenance, upgrade, and migration

Maintenance is a reversible change to the code level of IBM MQ. Maintenance requires no migration. Upgrading is the process of changing the code level of IBM MQ. Migration is the process of updating queue managers, and other objects, such as applications or administrative procedures.

Maintenance is the application of a fix pack, interim fix or PTF. It has one main characteristic. Those fixes, whether applied using a maintenance installation tool, or installed using a manufacturing refresh on top of an installation, are at the same command level as the existing code. No migration is required after applying maintenance. The installation can be restored to its previous level and any changed queue managers or applications will continue to work at the restored code level.

Upgrading and migration are related but distinct. Upgrading is the process of taking an existing IBM MQ installation and upgrading to a new level of code. Unless you are upgrading the fix level of IBM MQ, but not its command level, an upgrade must be followed by migration. Migration is the process of converting queue managers, applications, and other objects to run at a new command level.

An upgrade can take four different forms:

1. Application of a fix pack, interim fix, or a program temporary fix (PTF) using maintenance installation tool. Upgrades applied this way might not be called upgrades, but just fixes. Fixes, applied using a maintenance installation tool, can be rolled back completely as long as no queue manager migration has taken place, and IBM MQ is returned to its previous code level.
2. Installation of new code on top of existing code. You might be able to roll back an upgrade applied in this way; it depends on the platform. Generally speaking, you cannot roll back the installation of new code. To restore the old code level, you must retain the old installation media, and any fixes you applied.
3. Removal of the old level of code, followed by installation of the new level. The installers on very few platforms require you to remove an old installation first. Needless to say, to restore the old code level, you must reinstall it and any fixes.
4. Side by side installation. On z/OS you can install different code levels alongside each other on the same server. In the Job Control Language to start a subsystem, you select the code level to use. On UNIX, Linux, and Windows, you associate a queue manager with an installation, and start the queue manager. In IBM MQ, running multiple queue managers at different command levels on the same server is termed queue manager coexistence. You must not infer from this you can select different installations to run a queue manager at different times. Once a queue manager has been run, it is subject to the rules regarding reverting to earlier or later command levels.

Migration always follows an upgrade that changes the queue manager command level, both automatic and manual changes. Migration is the transformation of queue manager data, applications, and the environment that the queue manager runs in. Migration, and maintenance, and upgrading are described in the following topics.

Upgrades can be backed out, as long as no migration has taken place. The process of removing an upgrade varies by platform and how the upgrade was applied. Upgrades that change the command level of IBM MQ require queue manager migration before applications can reconnect.

IBM i distributed Migration cannot be reversed.

z/OS Migration is a two stage process. Only the first stage, called compatibility mode, is reversible.

IBM MQ maintenance

Maintenance is the application of a reversible fix. Any changes to queue manager data are compatible with the previous code level.

IBM i distributed Maintenance is the process of applying interim fixes or fix-packs.

z/OS Maintenance is the process of applying program temporary fixes on z/OS.

An important characteristic of applying maintenance is that it must be reversible. Reversibility implies two things:

1. The previous level of code is fully restored.
2. Changes that are made to IBM MQ objects are compatible. Changes are things like the creation or deletion of persistent messages, changes to queue managers, channels, topics, and queues. New and modified objects continue to work correctly with the restored level of code.

The reversibility of a maintenance package limits the extent of functional changes that are included in a maintenance package. No irreversible changes are included in a maintenance package. But, reversibility has limits. A maintenance package might include new programming and administrative interfaces. If you build new or modified applications to use the new interfaces, those applications do not work, if the maintenance package is removed.

Multi-instance queue managers are a good example. Should you remove the Version 7.0.1 fix pack that upgraded Version 7.0, then multi-instance queue manager functions no longer work. However, the queue managers continue to function correctly as single instance queue managers in Version 7.0.

On a smaller scale, a fix pack or interim fix might introduce a new configuration parameter to solve a problem. If you remove the fix pack or interim fix, although the new interface introduced by the change is not available any more, IBM MQ works with any objects that have been changed by the configuration parameter. For example, a new Java system property might introduce a parameter to set a code page for queue manager data conversion. The fix does not change any existing persistent queue manager state information. It can be removed, and the queue manager continues to work as before, but without the capability introduced in the fix.

Related concepts

"The version naming scheme for IBM MQ for z/OS" on page 20
On IBM MQ for z/OS, releases have a three-digit Version, Release, and Maintenance (VRM) level code. The code is significant; it identifies the service life of a release. To run a queue manager at a different VRM level, you must migrate the queue manager, its applications, and the environment in which it runs. Depending on the migration path, the migration might require more or less effort.

"The version naming scheme for IBM MQ (On platforms other than z/OS)" on page 21
IBM MQ releases have a four-digit Version, Release, Modification, and Fix (VRMF) level code.

IBM MQ upgrades and fixes

The term upgrade applies to changing the version V, release R, or modification `M` of a product. The term fix applies to a change in the F digit.

When you upgrade from one release to another, or apply maintenance refresh packs, fix packs, or interim fixes, the impact of the change depends on the extent of the change in V,R,M,F level. The VRM codes are explained in "The version naming scheme for IBM MQ (On platforms other than z/OS)" on page 21.

At each change of V, R, or `M`, the command level on the queue manager changes, but on a change to F, the command level does not.

distributed On distributed platforms, after an upgrade has been applied, the only way to *back out* a `V.R.M` change is by:

- Uninstalling the product code and reinstalling the code, or
- Installing the old level of code alongside the existing code and using the setmqm command to associate the queue manager with the other installation.

The general rule, is that if you have carried out an install that causes the command level of the new installation to be updated, and started the queue manager, you cannot *back out* the changes.

Characteristics of fixes

Application of a fix pack, interim fix, or a program temporary fix (PTF) using a maintenance installation tool should be called a fix.

Fixes, applied using a maintenance installation tool, can be rolled back completely, as long as no queue manager migration has taken place on:

- AIX
- Windows
- z/OS

and IBM MQ is returned to its previous code level.

On all other platforms you must reinstall the product.

Characteristics of different types of upgrade

An upgrade can take one of three different forms:

1. Installation of new code on top of existing code. You might be able to roll back an upgrade applied in this way; it depends on the platform. Generally speaking, you cannot roll back the installation of new code. To restore the old code level, you must retain the old installation media, and any fixes you applied.
2. Removal of the old level of code, followed by installation of the new level. The installers on very few platforms require you to remove an old installation first. Needless to say, to restore the old code level, you must reinstall it and any fixes.
3. Side by side installation.

   - z/OS On z/OS you can install different code levels alongside each other on the same server. In the Job Control Language to start a subsystem, you select the code level to use.
   - Windows Linux UNIX On UNIX, Linux, and Windows, you associate a queue manager with an installation, and start the queue manager. In IBM MQ, running multiple queue managers at different command levels on the same server is termed queue manager coexistence.

   You must not infer from this, that you can select different installations to run a queue manager at different times. Once a queue manager has been run, it is subject to the rules regarding reverting to earlier or later command levels.

Note: The term upgrade does not imply that an IBM MQ installation can be directly upgraded from one level to another. On some platforms, an upgrade requires that you remove the previous IBM MQ installation. You can retain any queue managers that you have created.

z/OS On z/OS, reversibility of an upgrade has two parts; backout of the installation to the previous code level, and reversion of any queue managers that have been started at the new code level, to work with the previous code level again. See "Upgrade, migration, and maintenance of IBM MQ on z/OS" on page 49 for more information.

The rules regarding the reversibility of an queue manager to run on a previous code level is dependent on the platform.

distributed On IBM i, UNIX, Linux, and Windows, changes in version, release, or modification level are not fully reversible, but changes in fix level are reversible under certain conditions.

An irreversible upgrade implies that you must back up the queue managers, or your system, before upgrading, to be able to restore your queue managers. Taking a backup of a queue manager requires you to stop the queue manager. If you do not take a backup, you are not able to restore IBM MQ to its previous level. Any changes you make on the new level cannot be restored onto the backup system. Changes include the creation or deletion of persistent messages, and changes to queue managers, channels, topics, and queues.

Related concepts

"Upgrade, migration, and maintenance of IBM MQ on z/OS" on page 49
You can install new releases of IBM MQ to upgrade IBM MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same z/OS instance. Running a queue manager at a higher level requires migration. Maintenance differs from upgrading. To maintain a level of IBM MQ, you apply Program Temporary Fixes (PTFs) to the installed code.

"New function in maintenance level upgrades (On platforms other than z/OS)" on page 48

IBM might introduce new functions between releases in maintenance level upgrades such as fix packs. A maintenance level upgrade including new function increases the maximum command level of an installation.

Related reference

"z/OS: OPMODE" on page 272
The availability of new functions and backward migration for IBM MQ for z/OS is controlled by the **OPMODE** parameter in the **CSQ6SYSP** macro. To access V8.0 capabilities, change the value of **OPMODE** to `OPMODE=(NEWFUNC,800)`. To restrict the use of new capabilities, and retain the ability to revert the queue manager to its earlier level, leave **OPMODE** at its default setting, `OPMODE=(COMPAT,800)`.

Windows *Windows - upgrading an IBM MQ installation*

To upgrade an IBM MQ installation on Windows, from one version, release, and modification level to a later one, carry out the tasks detailed in one of the following subtopics.

Before you begin, ensure that you have backed up your data.

Important: If you want to apply maintenance instead, see the Windows topics listed in "Applying and removing maintenance level updates (On platforms other than z/OS)" on page 215.

Related tasks

"Migrating a queue manager to the latest release" on page 107
The procedures for migrating a queue manager to the latest release are detailed in the following topics.

Windows *Upgrading an IBM MQ server installation using the Launchpad*

How you upgrade an IBM MQ server installation on Windows to a newer version, release, or modification, using the Launchpad.

Before you begin

Ensure that you have:

1. Stopped all your IBM MQ applications
2. Shut down your listeners
3. Stopped all your queue managers
4. Backed up your data

Important: If you want to apply maintenance instead, see the Windows topics listed in "Applying and removing maintenance level updates (On platforms other than z/OS)" on page 215.

Procedure

1. Access the IBM MQ installation image. The location might be the mount point of the DVD, a network location, or a local file system directory.
2. Locate `setup.exe` in the base directory of the IBM MQ installation image.
   - From a DVD, this location might be:

     ```
     E:\setup.exe
     ```

   - From a network location, this location might be:

     ```
     m:\instmqs\setup.exe
     ```

   - From a local file system directory, this location might be:

     ```
     C:\instmqs\setup.exe
     ```

3. Double-click the **Setup** icon to start the installation process. It is possible to run either by:

   - Running `setup.exe` from the command prompt. Or
   - Double-clicking `setup.exe` from Windows Explorer.

   If you are installing on a Windows system with UAC enabled, accept the Windows prompt to allow the launchpad to run as elevated. During installation, you might also see **Open File - Security Warning** dialog boxes that list International Business Machines Limited as the publisher. Click **Run** to allow the installation to continue.

   The IBM MQ Installation Launchpad window is displayed.
4. Continue to follow the Launchpad instructions as shown on screen.
5. Select **Installing a new instance**, if you see a panel asking you to choose between installing a new instance, or maintaining or upgrading an existing instance, when you click the **Launch IBM MQ Installer** button.

   You use the other option when adding or removing features from an already installed IBM MQ.
6. On the next panel, choose between **Install leaving the existing installation(s) untouched** or **Upgrade an existing named installation already on the machine**, and click **Next**.

   ⚠ **Attention:** If you do not see this screen, it means that there was no IBM MQ server installation on the machine that could be upgraded by this installer.
7. Follow the installer prompts to upgrade your IBM MQ server installation.

Related tasks

"Upgrading an IBM MQ server installation using msiexec" on page 43
How you upgrade an IBM MQ server installation on Windows to a newer version, release, or modification, using msiexec.

"Upgrading an IBM MQ client installation using the Launchpad" on page 42
How you upgrade an IBM MQ client installation on Windows to a newer version, release, or modification, using the Launchpad.

"Upgrading an IBM MQ client installation using msiexec" on page 44
How you upgrade an IBM MQ client installation on Windows to a newer version, release, or modification, using msiexec.

Windows *Upgrading an IBM MQ client installation using the Launchpad*
How you upgrade an IBM MQ client installation on Windows to a newer version, release, or modification, using the Launchpad.

Before you begin

Ensure that you have:

1. Stopped all your IBM MQ applications
2. Shut down your listeners
3. Stopped all your queue managers
4. Backed up your data

Procedure

1. Access the IBM MQ installation image. The location might be the mount point of the DVD, a network location, or a local file system directory.
2. Locate `setup.exe` in the base directory of the IBM MQ installation image.
   - From a DVD, this location might be:

     ```
     E:\setup.exe
     ```

   - From a network location, this location might be:

```
m:\instmqs\setup.exe
```

- From a local file system directory, this location might be:

  ```
  C:\instmqs\setup.exe
  ```

3. Double-click the **Setup** icon to start the installation process. It is possible to run either by:

   - Running `setup.exe` from the command prompt. Or
   - Double-clicking `setup.exe` from Windows Explorer.

   If you are installing on a Windows system with UAC enabled, accept the Windows prompt to allow the launchpad to run as elevated. During installation, you might also see **Open File - Security Warning** dialog boxes that list International Business Machines Limited as the publisher. Click **Run** to allow the installation to continue.

   The IBM MQ Installation Launchpad window is displayed.
4. Continue to follow the Launchpad instructions as shown on screen.
5. Select **Installing a new instance**, if you see a panel asking you to choose between installing a new instance, or maintaining or upgrading an existing instance, when you click the **Launch IBM MQ Installer** button.

   You use the other option when adding or removing features from an already installed IBM MQ.
6. On the next panel, choose between **Install leaving the existing installation(s) untouched** or **Upgrade an existing named installation already on the machine**, and click **Next**.

   ⚠ **Attention:** If you do not see this screen, it means that there was no IBM MQ client installation on the machine that could be upgraded by this installer.
7. Follow the installer prompts to upgrade your IBM MQ client installation.

Related tasks

“Upgrading an IBM MQ server installation using the Launchpad” on page 41
How you upgrade an IBM MQ server installation on Windows to a newer version, release, or modification, using the Launchpad.

“Upgrading an IBM MQ client installation using the Launchpad” on page 42
How you upgrade an IBM MQ client installation on Windows to a newer version, release, or modification, using the Launchpad.

“Upgrading an IBM MQ client installation using msiexec” on page 44
How you upgrade an IBM MQ client installation on Windows to a newer version, release, or modification, using msiexec.

Windows *Upgrading an IBM MQ server installation using msiexec*
How you upgrade an IBM MQ server installation on Windows to a newer version, release, or modification, using msiexec.

Before you begin

Ensure that you have:

1. Stopped all your IBM MQ applications
2. Shut down your listeners
3. Stopped all your queue managers
4. Backed up your data

Important: if you want to apply maintenance instead, see the Windows topics listed in “Applying and removing maintenance level updates (On platforms other than z/OS)” on page 215.

Procedure

1. Access the IBM MQ installation image. The location might be the mount point of the DVD, a network location, or a local file system directory.
2. Locate `MSI file` in the MSI directory of the IBM MQ installation image.
   - From a DVD, this location might be:

     ```
     E:\\MSI\IBM MQ.msi
     ```
   - From a network location, this location might be:

     ```
     m:\instmqs\\MSI\IBM MQ.msi
     ```
   - From a local file system directory, this location might be:

     ```
     C:\instmqs\\MSI\IBM MQ.msi
     ```
3. Optional: If you are upgrading the only IBM MQ server installation, where the installation has the default value `Installation1` issue the following command:

   ```
   msiexec /i "<InstallationImage>\MSI\IBM MQ.msi" /q AGREETOLICENSE=YES
   INSTALLATIONNAME="Installation1"
   ```
4. Optional: If you are upgrading an installation on a machine that already has one or more IBM MQ server installations of the level you are upgrading to, you must provide additional parameters to select a free MSI instance ID.

   See Choosing MSI Instance IDs for multiple server installations for more information.

   In this case, the command might look something like:

   ```
   msiexec /i "<Installation Image>\MSI\IBM MQ.msi" /q AGREETOLICENSE=YES
   INSTALLATIONNAME="Installation2" NEWINSTANCE=1
   TRANSFORMS=":InstanceId2.mst;1033.mst"
   ```

Related tasks

"Upgrading an IBM MQ server installation using the Launchpad" on page 41
How you upgrade an IBM MQ server installation on Windows to a newer version, release, or modification, using the Launchpad.

"Upgrading an IBM MQ client installation using the Launchpad" on page 42
How you upgrade an IBM MQ client installation on Windows to a newer version, release, or modification, using the Launchpad.

"Upgrading an IBM MQ client installation using msiexec" on page 44
How you upgrade an IBM MQ client installation on Windows to a newer version, release, or modification, using msiexec.

Windows *Upgrading an IBM MQ client installation using msiexec*
How you upgrade an IBM MQ client installation on Windows to a newer version, release, or modification, using msiexec.

Before you begin

Ensure that you have:

1. Stopped all your IBM MQ applications
2. Shut down your listeners
3. Stopped all your queue managers

4. Backed up your data

Procedure

1. Access the IBM MQ installation image. The location might be the mount point of the DVD, a network location, or a local file system directory.
2. Locate `MSI file` in the MSI directory of the IBM MQ installation image.
   - From a DVD, this location might be:

     ```
     E:\\MSI\IBM MQ.msi
     ```
   - From a network location, this location might be:

     ```
     m:\instmqs\\MSI\IBM MQ.msi
     ```
   - From a local file system directory, this location might be:

     ```
     C:\instmqs\\MSI\IBM MQ.msi
     ```
3. Optional: If you are upgrading the only IBM MQ client installation, where the installation has the default value `Installation1` issue the following command:

   ```
   msiexec /i "<InstallationImage>\MSI\IBM MQ.msi" /q AGREETOLICENSE=YES
   INSTALLATIONNAME="Installation1"
   ```
4. Optional: If you are upgrading an installation on a machine that already has one or more IBM MQ client installations of the level you are upgrading to, you must provide additional parameters to select a free MSI instance ID.

   See Choosing MSI Instance IDs for multiple client installations for more information.

   In this case, the command might look something like:

   ```
   msiexec /i "<Installation Image>\MSI\IBM MQ.msi" /q AGREETOLICENSE=YES
   INSTALLATIONNAME="Installation2" NEWINSTANCE=1
   TRANSFORMS=":InstanceId2.mst;1033.mst”
   ```

Related tasks

“Upgrading an IBM MQ server installation using the Launchpad” on page 41
How you upgrade an IBM MQ server installation on Windows to a newer version, release, or modification, using the Launchpad.

“Upgrading an IBM MQ client installation using the Launchpad” on page 42
How you upgrade an IBM MQ client installation on Windows to a newer version, release, or modification, using the Launchpad.

“Upgrading an IBM MQ server installation using msiexec” on page 43
How you upgrade an IBM MQ server installation on Windows to a newer version, release, or modification, using msiexec.

IBM MQ migration

Migration is the conversion of programs and data to work with a new code level of IBM MQ. Some types of migration are required, and some are optional. Queue manager migration is never required after applying a maintenance level update, that does not change the command level. Some types of migration are automatic, and some are manual. Queue manager migration is typically automatic and required after releases and manual and optional after a maintenance level upgrade that introduces a new function. Application migration is typically manual and optional.

Whenever you upgrade IBM MQ to a new release that changes its command level, migration is performed by the queue manager. Whenever you upgrade IBM MQ to a new maintenance or fix level, which introduces a new function using a new command level, you can migrate the queue manager to use the new command level and thereby the new function.

You must read "Changes that affect migration" on page 173 before upgrading your IBM MQ installation or migrating your queue managers, to identify what migration tasks you must plan for.

Using the model in Figure 7 on page 48 you can distinguish different migration questions, which are discussed in the following topics:

Operating environment migration
Upgrading the operating environment, or components in the environment such as installing a new level of JRE; see "IBM MQ operating environment migration" on page 52

Queue manager migration
Migrating a queue manager following an upgrade of the IBM MQ installation to a new command level; see "Queue manager migration" on page 68.

IBM MQ MQI client migration
Migrating a client configuration following installation of a new version or release of the IBM MQ MQI client ; see "IBM MQ MQI client migration" on page 70.

Application migration
Relinking, recompiling, or recoding an IBM MQ server or client application; see "Application migration and interoperation" on page 73. Application migration also includes migrating any API or channel exits

In addition, you must consider the impact of migrating one queue manager, or IBM MQ MQI client client, on other clients or queue managers:

Compatibility, coexistence, and interoperability
See "Coexistence, compatibility, and interoperability" on page 53 for information about the compatibility of IBM MQ applications connected to queue managers and IBM MQ MQI client clients on different command levels. The section also explains the concept of queue manager coexistence, and the interoperability of IBM MQ JMS applications with WebSphere Application Server.

Queue manager clusters
Can a queue manager cluster contain queue managers at different command levels? See "Queue manager cluster migration" on page 74 to answer this question, and how to migrate a cluster of queue managers.

z/OS **Queue-sharing groups**
Queue sharing groups involve multiple queue managers running on z/OS. How do you migrate queue managers that are part of a queue-sharing group to a new command level; see "Queue-sharing group migration" on page 76.

High-availability clusters
How do you migrate queue managers that are part of a high-availability cluster to a new command level, and maintain continuous and reliable service? See "Migrating a queue manager in a high-availability configuration" on page 76, which covers both migration of multi-instance queue managers, and the migration of queue managers operating in high-availability clusters.

IBM MQ migration concepts

Figure 7 on page 48 shows two runtime operating system environments. One environment is called `Server`, and contains an IBM MQ server and server application. The other is called `Client`, and contains an IBM MQ MQI client client application. The server environment has one or more queue managers represented by **QM** using the installation of IBM MQ installed on the server.

The queue manager labeled `QM-n`? coexists on the same server as `QM`, but runs at a different release level. Multiple releases of IBM MQ installed in the same operating environment are called coexistent [2] .

[2] It is not necessary, but it is usual, for coexistent installations to be at different release levels.

The IBM MQ installations for different release levels are not shown. The question-mark in the queue manager name indicates this capability might not be present in your environment.

z/OS Only z/OS supports multiple queue managers coexisting at different release levels in the same operating environment.

Queue manager coexistence is important for migration in two respects:

1. It can be used to reduce the risk involved in migrating to a new command level, and reduce the downtime during the migration process.
2. You must consider any configuration implications of running some applications or clusters on the same server with queue managers at different command levels.

For details, see "Queue manager coexistence in Version 8.0" on page 53

The queue manager, `QM*`, represents queue managers of various levels installed on other servers.

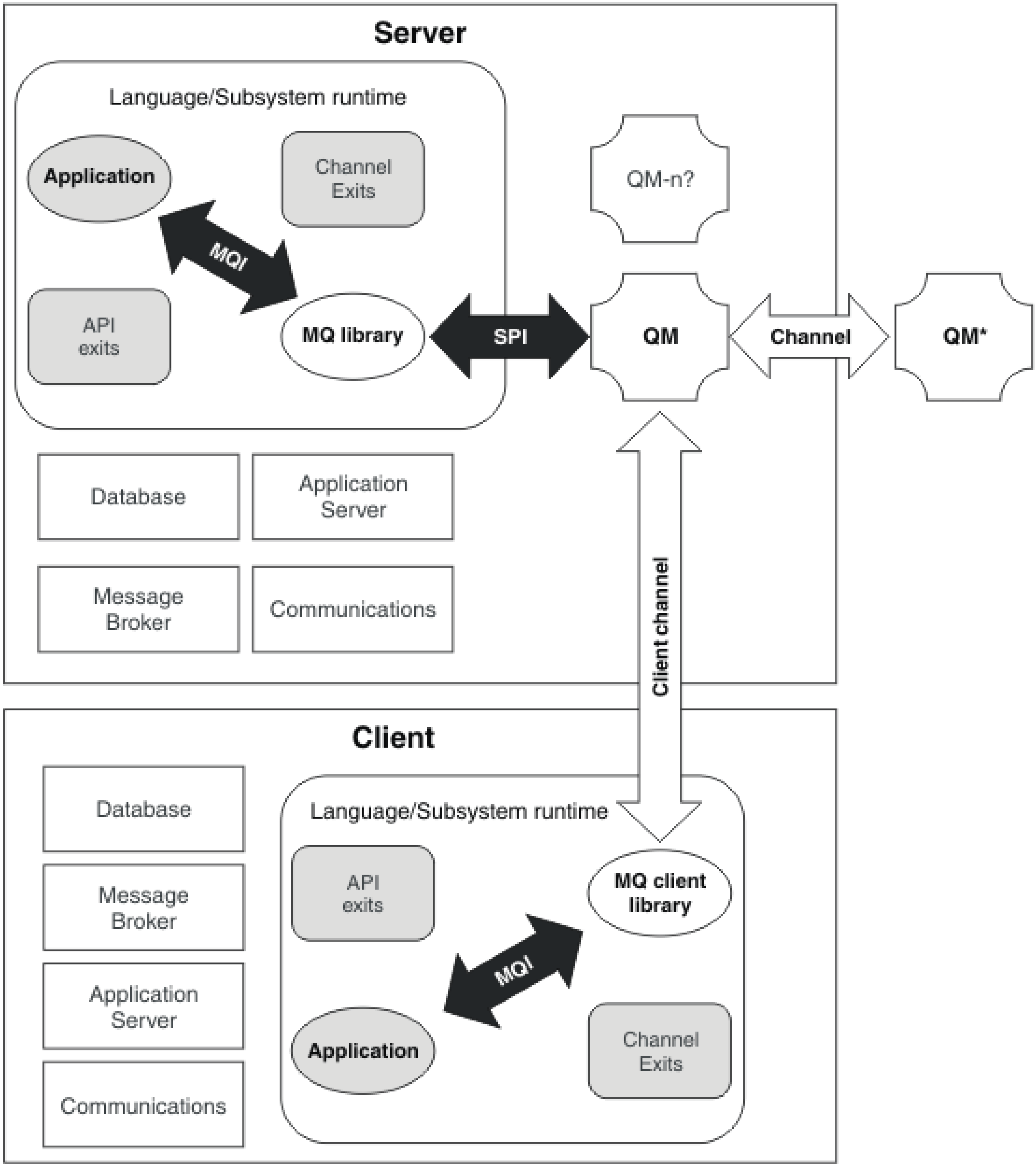


Figure 7. IBM MQ application migration model

New function in maintenance level upgrades (On platforms other than z/OS)

IBM might introduce new functions between releases in maintenance level upgrades such as fix packs. A maintenance level upgrade including new function increases the maximum command level of an installation.

When you apply the maintenance level upgrade, the installation supports the new command level. A queue manager associated with the installation is not able to use the new function until you have run **strmqm** to set the queue manager to the new command level.

A fix pack introducing a new function also includes regular fixes as a normal fix pack. If you do not want to use the function, but apply the fixes, do not enable the new function for any queue manager. The fix pack then is a fully reversible maintenance level upgrade. If you want to use a new function with a queue manager, you must change the command level of the queue manager. If one queue manager associated with an installation enables the new function, the fix pack introducing the new function and all previous fix packs cannot be removed anymore.

By applying a maintenance level upgrade that introduces a new function the IBM MQ installation supports an additional command level. An installation can therefore support multiple command levels and multiple queue managers associated with it can run on different command levels.

To enable a new function for a queue manager, you must set the command level of queue manager to the command level supporting the new function. Because command levels are cumulative, the command level also supports all other functions introduced by previous maintenance level upgrades. By enabling a new function from a maintenance level upgrade you therefore enable all new functions from previous maintenance level upgrades, too.

Example

You have installed two fix packs including new functions:

1. Fix pack 7.1.0.2, introducing the command level 711.
2. Fix pack 7.1.0.15, introducing the command level 712.

You might have also installed other fix packs, but these did not introduce new function.

You want to use the new function from fix pack 7.1.0.15 on one of your queue managers. You enable command level 712 for that queue manager using the **strmqm** command. This enables all of the function introduced in both fix packs. It is not possible to use the function introduced by command level 712 without also enabling the function introduced by command level 711.

Once you have enabled command level 712 for a queue manager, the minimum version of product code to start the queue manager is 7.1.0.15. If you uninstall fix pack 7.1.0.15 from the installation with which the queue manager is associated, you will not be able to start the queue manager. In this situation, the **strmqm** command fails and issues error message AMQ7204 indicating that the queue manager has already been started by a newer release. If you have multiple installations of IBM MQ, you can start the queue manager using another installation which satisfies this minimum version requirement.

Related tasks

“Migrating queue managers to new-function fix packs” on page 250

Upgrade, migration, and maintenance of IBM MQ on z/OS

You can install new releases of IBM MQ to upgrade IBM MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same z/OS instance. Running a queue manager at a higher level requires migration. Maintenance differs from upgrading. To maintain a level of IBM MQ, you apply Program Temporary Fixes (PTFs) to the installed code.

Applying PTFs does not change the version, release, or maintenance level of the code. No queue manager migration is required after applying maintenance. PTFs are grouped into Recommended Service Updates (RSUs) that have been tested together in a Consolidated Service Test (CST); see Consolidated Service Test and the RSU.

When you install a new VRM level of IBM MQ on z/OS using SMP/E, it creates a set of IBM MQ libraries. The libraries for different VRM levels of IBM MQ can coexist on the same instance of z/OS. You can then run different queue managers against different release levels of IBM MQ on the same z/OS instance.

If you start a queue manager running on a later release level, then migration of the queue manager to that release level is required. Even if the difference is only in the modification level, some migration might be required. The migration tasks you must perform to migrate from one version to another are documented in “z/OS: Migration planning to the latest release” on page 81; see also “Changes that affect migration” on page 173.

From Version 7.0.1, after you have fully migrated a queue manager to a new version or release, reverse migration is not possible. Before Version 7.0.1, full migration took place when you started a queue manager for the first time at the new release level. For Version 7.0.1 and later versions, you have control over when migration takes place, using a new **CSQ6SYSP** parameter, **OPMODE** ; see “z/OS: OPMODE” on page 272. If your queue manager is on Version 7.0 or earlier, you can revert to an earlier release. You might have to contact your IBM support center for a backward migration PTF.

Using **OPMODE**, you can migrate all your existing applications to the new release level, and still be able to revert to the previous release level. Once you start changing applications, or adding applications that use new function, you cannot revert to the previous level of IBM MQ. **OPMODE** applies to migration from Version 6.0 to Version 7.0.1, onwards.

OPMODE gives you the option of enforcing a two-stage migration process:

1. Regression test your existing applications.
2. Develop new applications, and change existing applications, to use the new function in the release.

The strategy for upgrading queue managers at Version 6.0 or later is as follows:

1. Apply the coexistence and backward migration PTFs to all the queue managers you are going to upgrade. After applying the PTFs, you can run queue managers of different levels in the same queue sharing groups. You can also reverse the migration of a queue manager back to your current level.
2. Upgrade the first queue manager.
3. Check all your existing applications run correctly on this queue manager.
4. Bring all the queue managers in a queue sharing group up to the new level, and check that existing applications continue to work correctly.
5. Change the setting of **OPMODE** so that applications can use new function on all the queue managers in the queue sharing group.

   Note: Step 5 is the point of no return. You can no longer run that queue manager at the previous level of IBM MQ.
6. To enable new IBM MQ v7.1, or later, function, restart all queue managers within the queue sharing group.

The coexistence and backward migration PTFs have two distinct purposes: [3]

1. To allow queue managers at the earlier release level to coexist with ones at the later release level. In particular for queue managers to coexist in the same queue sharing group.
2. To handle queue manager data and logs formatted using the data definitions of the later release.

On z/OS, maintenance is supplied as Program Temporary Fixes, PTFs, which are applied and removed using SMP/E. PTFs are specific to a particular set of libraries corresponding to specific release level. Apart from any exceptions documented with the PTFs, PTFs do not change the correct operation of IBM MQ. Nonetheless, you must check that the fixes have not changed the operation of critical programs unexpectedly.

PTFs that apply to a category of software fixes might be grouped together and identified using a fix category. For further information, see IBM Fix category values and descriptions.

Characteristics of different types of upgrade on z/OS

When you upgrade from one release to another, or apply maintenance in the form of PTFs, on z/OS, the impact of the change depends on the extent of the change in VRM level. The VRM codes are explained in "The version naming scheme for IBM MQ for z/OS" on page 20.

PTF upgrades do not require migration, and are reversible. From Version 7.0.1, all upgrades from Version 6.0 or later are reversible if the **OPMODE** has not been set to NEWFUNC.

Related concepts

"The version naming scheme for IBM MQ for z/OS" on page 20
On IBM MQ for z/OS, releases have a three-digit Version, Release, and Maintenance (VRM) level code. The code is significant; it identifies the service life of a release. To run a queue manager at a different

[3] Coexistence and backward migration changes might be shipped as a single or multiple fixes.

VRM level, you must migrate the queue manager, its applications, and the environment in which it runs. Depending on the migration path, the migration might require more or less effort.

“Queue manager coexistence in Version 8.0” on page 53
Queue managers, with different names, can coexist on any server as long as they use the same IBM MQ installation. On z/OS z/OS, UNIX, Linux, and Windows, different queue managers can coexist on the same server and be associated with different installations.

Upgrade, migration, and maintenance of IBM MQ (On platforms other than z/OS)

You can install new releases of IBM MQ to upgrade IBM MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of IBM MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

On different platforms, you employ different mechanisms to install and maintain software releases. Installing a release at a new maintenance level, and applying maintenance level upgrades to upgrade an earlier release to the same maintenance level, have different results.

If you start a queue manager running on a later release level, then migration of the queue manager to that release level is required. The migration tasks you must perform to migrate from one release to another are documented in “Migrating a queue manager to the latest release” on page 107 ; see also “Changes that affect migration” on page 173.

When you upgrade the maintenance or fix level of IBM MQ by applying a regular maintenance level upgrade, you can reverse the upgrade by removing the fix. When you upgrade the maintenance or fix level of IBM MQ by applying a maintenance level upgrade containing a new function, you can reverse that upgrade and all previously reversible upgrades until a queue manager associated with the installation enables the new function.

You cannot easily revert to a previous level of IBM MQ after installation. If you install a copy of IBM MQ obtained from Passport Advantage or from physical media, the installer uninstalls IBM MQ, if it is present. It then installs the new level of IBM MQ. To revert to the previous level of IBM MQ, you must keep the earlier installation image and any fixes you applied. Then you must uninstall the new level, reinstall the previous release level, and reapply the required fixes. If you have started any queue managers at the later level, they will not work with the restored level of IBM MQ[4] . To restore IBM MQ to its previous level, after starting any queue managers, you must first back up the queue managers. You can then restore the backup queue managers after restoring the previous level of IBM MQ.

Maintenance levels and fix levels are both supplied from the service site, Fix Central. Fix central has a function to tell you what upgrades you can apply to the current level of your system. If you back out a maintenance level upgrade, it returns IBM MQ code to the same level of code as before applying the maintenance level upgrade.

Characteristics of different types of upgrade

When you upgrade from one release to another, or apply maintenance refresh packs, fix packs, or interim fixes, the impact of the change depends on the extent of the change in VRMF level. The VRM codes are explained in “The version naming scheme for IBM MQ (On platforms other than z/OS)” on page 21.

Table 7 on page 52 describes characteristics of different upgrade paths for IBM MQ. In particular, notice that migration is required only if the version or release number changes. Other types of upgrade do not require migration, and are reversible if the upgrade is applied using a maintenance procedure.

[4] Unless you installed a later maintenance level upgrade, not a new release or version: then you could revert to an earlier maintenance level by reinstalling the earlier maintenance level upgrade. Queue manager data is compatible between maintenance levels.

Table 7. Types of upgrade (Platforms other than z/OS)

Upgrade type	Examples	Suggested testing	Type of installation	Manufacturing refresh	New features	IBM MQ Migration
Version	7.0	Full testing of all applications	Full or upgrade installation	Yes	Yes	Yes
Modification	7.0.1	Regression test critical applications	Full, or upgrade installation, or maintenance package		Minor	No
Enabled new-function fix pack	No example yet	Regression test critical applications	Maintenance package & queue manager migration	No		
No-function or not enabled new-function fix pack	7.0.1.3	Brief test of critical applications	Maintenance package		No	
Interim fix	7.0.0.1-WS-MQ-Windows-LAIZ50784	Test affected applications	Manual			

Related concepts

"Queue manager coexistence in Version 8.0" on page 53
Queue managers, with different names, can coexist on any server as long as they use the same IBM MQ installation. On z/OS z/OS, UNIX, Linux, and Windows, different queue managers can coexist on the same server and be associated with different installations.

"Multi-installation queue manager coexistence on UNIX, Linux, and Windows" on page 57
You can install multiple copies of IBM MQ for UNIX, Linux, and Windows on the same server. The installations must be at Version 7.1 or later, with one exception. One Version 7.0.1 installation, at fix pack level 6, or later, can coexist with multiple Version 7.1, or later installations.

"The version naming scheme for IBM MQ (On platforms other than z/OS)" on page 21
IBM MQ releases have a four-digit Version, Release, Modification, and Fix (VRMF) level code.

Related information

Backing up and restoring a queue manager

IBM MQ operating environment migration

You might perform some migration tasks for IBM MQ as a result of upgrading the operating environment.

To find out what operating environment upgrades you must make before upgrading IBM MQ, compare the requirements for different releases. For more information about system requirements, see System Requirements for IBM MQ.

Note that the System Requirements page for IBM MQ 8.0 uses the Software Product Compatibility Reports (SPCR) tool.

By selecting the appropriate link on the web page, the SPCR tool enables you to go directly to the:

- Supported operating systems
- Prerequisites
- System requirements, and
- Optional supported software

for the specific operating system, or systems, that your enterprise uses.

If an operating environment change directly affects the migration to a new version of IBM MQ, it is listed in "Changes that affect migration" on page 173.

The change might affect IBM MQ migration indirectly. For example, the runtime linkage conventions for applications, or the way memory is allocated, might change.

Coexistence, compatibility, and interoperability

The definitions of the IBM MQ terms coexistence, compatibility, and interoperability.

Coexistence

Is being able to install and run two or more versions of the same program on the same server. For IBM MQ, it normally means installing and running multiple versions of IBM MQ on a server.

Compatibility

Is the ability to run applications from one level of queue manager with an earlier, or previous level, of the queue manager.

If you are using a message channel agent (MCA) channel, any version and release of an IBM MQ queue manager can connect, using an MCA channel, to any version and release of another IBM MQ queue manager.

The MCA channel is automatically configured to the latest version of protocol that is supported by both ends of the channel.

Compatibility is also the ability to run client applications with different versions of the IBM MQ MQI client, and different levels of the queue manager.

Interoperability

Is mainly the ability to exchange messages between different versions of IBM MQ. It can also mean the interoperability between others things, such as publish/subscribe brokers, or between components such as the IBM MQ classes for JMS and WebSphere Application Server.

Maintaining the compatibility, coexistence, and interoperability of IBM MQ is important in order to preserve the investment you make in applications and administrative procedures.

Three areas to which this objective does not apply to as rigidly, are:

- GUI interfaces, such as IBM MQ Explorer.
- Information for service, such as FFST files and traces.
- Error messages. The text in an error message might change, to make the wording clearer or more accurate.

Queue manager coexistence in Version 8.0

Queue managers, with different names, can coexist on any server as long as they use the same IBM MQ installation. On z/OS z/OS, UNIX, Linux, and Windows, different queue managers can coexist on the same server and be associated with different installations.

Single installation queue manager coexistence on all platforms

Single installation queue manager coexistence is useful in development and production environments. In development environments, you can set up different queue manager configurations to support different development activities. You can also work with multiple queue manager configurations on a single server, connected by channels, as if deployed on a network.

In production environments configuring multiple queue manager on a single server is less common. It has no performance or functional advantage over a single queue manager configuration. Sometimes, you must deploy multiple queue managers on server. It might be essential to meet the requirements of a particular software stack, governance, administration, or as a consequence of the consolidation of servers.

Multi-installation queue manager coexistence

Multi-installation [5] queue manager coexistence became available in Version 7.1 on UNIX, Linux, and Windows. Multi-installation queue manager coexistence has always been supported on z/OS.

With multi-installation queue manager coexistence on the same server you can run queue managers at different commands levels on the same server. You can also run multiple queue managers at the same command level, but associate them with different installations.

Multi-installation adds more flexibility to the coexistence of queue managers using a single installation. Any of the reasons behind running multiple queue managers, such as supporting different software stacks, might require different versions of IBM MQ.

The biggest benefit of multi-installation identified by early users, is in upgrading from one version of IBM MQ to another. Multi-installation makes upgrading less risky, less costly, and is more flexible in meeting the migration needs of applications running on a server.

The key to migration flexibility is being able to install a new version alongside an existing installation; see Figure 8 on page 54, which is extracted from "UNIX, Linux, and Windows: Side-by-side migration to a later version" on page 96.

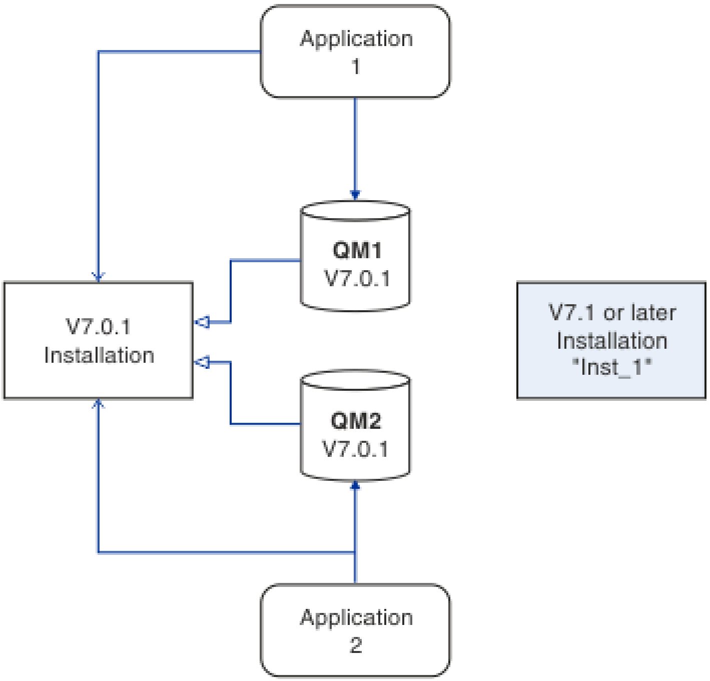


Figure 8. Side-by-side installation - step 2

When the installation is complete, and verified, migrate queue managers and applications to the new installation; see Figure 9 on page 55. When migration is complete, uninstall the old installation.

[5] Do not confuse multi-installation queue manager coexistence with multi-instance queue managers. They are completely different, though they sound similar in English.

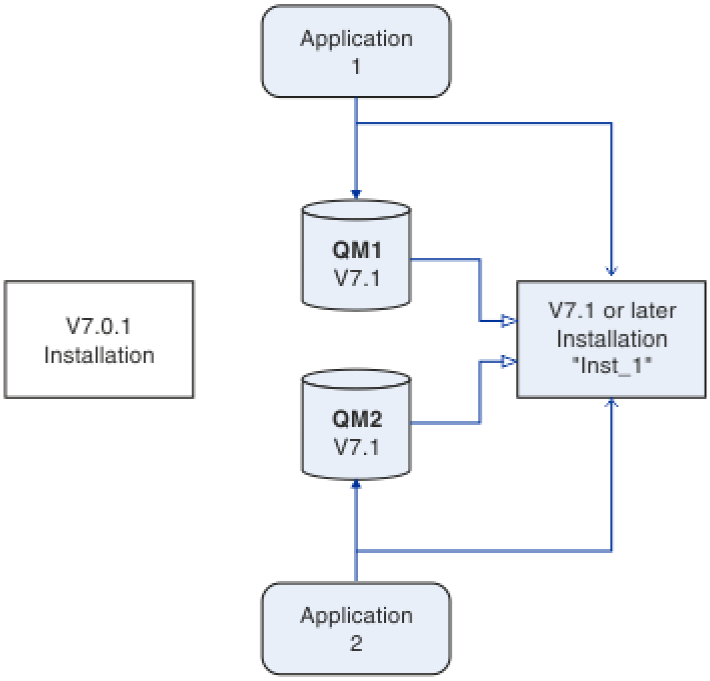


Figure 9. Side-by-side installation - step 4

Think of multi-installation as being the basis for a range of migration strategies. At one end is "Single-stage", in which you only have one installation on a server at a time. At the other end is multi-stage migration, in which you continue to run multiple installations at the same time. In the middle is side-by-side migration. Each of the three strategies is explained in these three tasks:

1. “UNIX, Linux, and Windows: Single-stage migration to a later version” on page 92
2. “UNIX, Linux, and Windows: Side-by-side migration to a later version” on page 96
3. “UNIX, Linux, and Windows: Multi-stage migration to a later version” on page 100

Another similar use of multi-installation is to support the migration of queue managers to a new fix level; see Figure 10 on page 56. You maintain two installations, one of which has the latest fix pack applied, and the other has the previous maintenance levels. When you have moved all queue managers to the latest fix pack level, you can replace the previous fix pack with the next fix pack to be released. The configuration allows you to stage the migrating applications and queue managers to the latest fix pack level. You can switch the primary installation designation to the latest fix pack level.

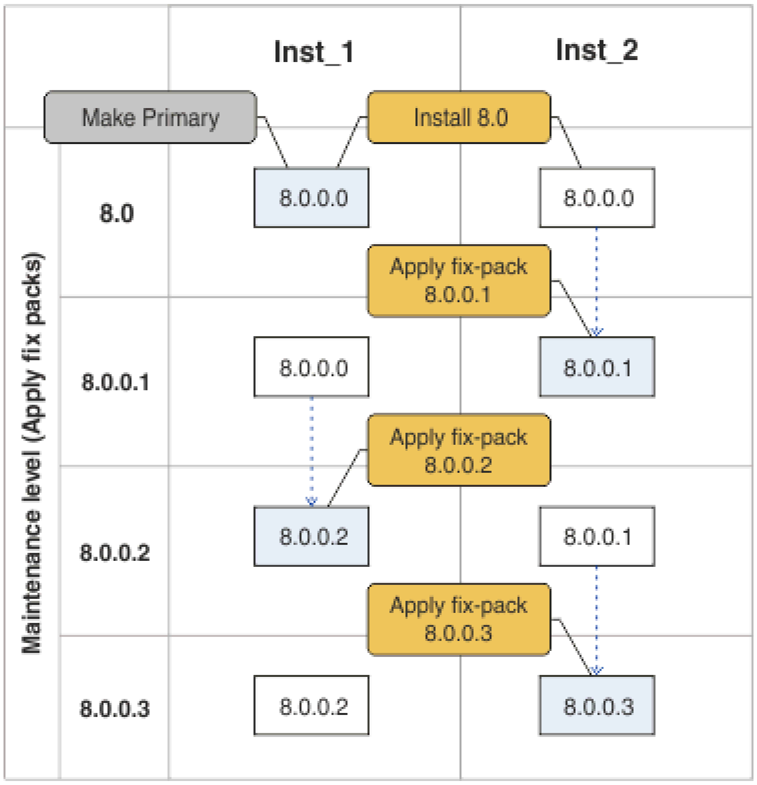


Figure 10. Rolling fix packs

Related concepts

"Multi-installation queue manager coexistence on UNIX, Linux, and Windows" on page 57
You can install multiple copies of IBM MQ for UNIX, Linux, and Windows on the same server. The installations must be at Version 7.1 or later, with one exception. One Version 7.0.1 installation, at fix pack level 6, or later, can coexist with multiple Version 7.1, or later installations.

"Upgrade, migration, and maintenance of IBM MQ on z/OS" on page 49
You can install new releases of IBM MQ to upgrade IBM MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same z/OS instance. Running a queue manager at a higher level requires migration. Maintenance differs from upgrading. To maintain a level of IBM MQ, you apply Program Temporary Fixes (PTFs) to the installed code.

Related tasks

"Migrating IBM MQ library loading from an earlier version of the product to the latest version" on page 131
No change in the way IBM MQ libraries are loaded is normally required if you upgrade from an earlier version of the product to the latest version. You must have followed the instructions on building IBM MQ applications in Version 7.0.1 and you must replace IBM WebSphere MQ 7.0.1 with the latest version of the product. If you choose to take advantage of multi-installation in the latest version of the product, based on the side-by-side or multi-stage migration scenarios, you must modify the environment for the operating system to resolve IBM MQ dependencies for an application. Typically, you can modify the runtime environment, rather than relink the application.

"UNIX: Migrating IBM MQ library loading from Version 7.0.1, or later, to the latest version" on page 136
Investigate whether applications connecting to the latest version of the product are linked to, and load libraries from, the correct installation.

"UNIX, Linux, and Windows: Single-stage migration to a later version" on page 92
Single-stage migration, is the term used to describe replacing the only installation of IBM MQ on a server, with a later release. Single stage migration is also known as `upgrading in place` or `in place upgrade` . Until Version 7.0.1.6, single-stage was the only migration scenario. Single-stage migration preserves existing scripts and procedures for running IBM MQ the most. With other migration scenarios

you might change some scripts and procedures, but you can reduce the effect queue manager migration has on users.

"UNIX, Linux, and Windows: Side-by-side migration to a later version" on page 96

"UNIX, Linux, and Windows: Multi-stage migration to a later version" on page 100

"UNIX, Linux, and Windows: Staging maintenance fixes" on page 264
Use multiple installations of IBM MQ on the same server to control the release of maintenance fixes.

"Windows: Migrating IBM MQ library loading from Version 7.0.1, or later, to the latest version" on page 133
Investigate whether applications connecting to the latest version of the product are linked to, and load libraries from, the correct installation.

Multi-installation queue manager coexistence on UNIX, Linux, and Windows

You can install multiple copies of IBM MQ for UNIX, Linux, and Windows on the same server. The installations must be at Version 7.1 or later, with one exception. One Version 7.0.1 installation, at fix pack level 6, or later, can coexist with multiple Version 7.1, or later installations.

Figure 11 on page 57 shows two IBM MQ installations, two queue managers, and three applications. Applications 2 and 3 are connected to `QM2`, and application 1 is connected to `QM1`. Applications 1 and 3 load IBM MQ libraries from the `Inst_1` installation, and application 2 loads libraries from the Version 7.0.1 installation.

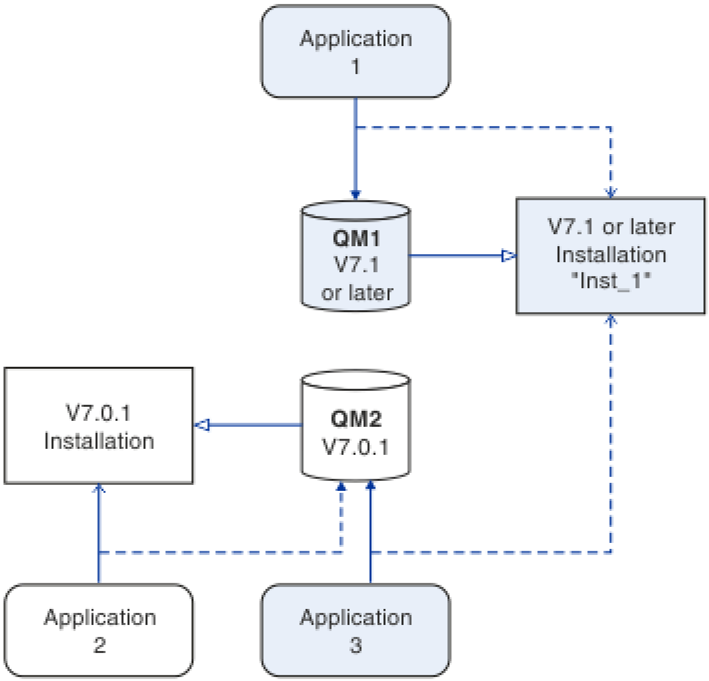


Figure 11. Coexistence of two queue managers using Version 7.0.1 and a later version installations

When you upgrade from Version 7.0.1 to a later version, you can choose to run Version 7.0.1 alongside the later version. The installation, illustrated in Figure 11 on page 57, is called a multi-version installation. You can also install multiple copies of Version 7.1 alongside each other. That would be called multi-installation. Multi-installation is the more general term.

Version 7.0.1 did not support multi-installation on distributed platforms. Before Version 7.1 becoming available, fix pack 7.0.1.6 was shipped with some fixes to make Version 7.0.1 compatible with a later version on the same server. With 7.0.1.6 installed, you can run one copy of Version 7.0.1 alongside multiple copies of the later version. You do not have to apply the fix pack to upgrade Version 7.0.1 to Version 7.1 "in-place" ; see "UNIX, Linux, and Windows: Single-stage migration to a later version" on page 92.

A multi-version installation that includes Version 7.0.1, does not behave the same way as a multi-installation that does not. The differences primarily affect how you might choose to configure how applications load IBM MQ libraries, and run IBM MQ commands. Because of these differences, think of the multi-version support provided in 7.0.1.6, as a migration aid to moving to a later version multi-installation environment. The topics that explain the restrictions in Version 7.0.1 multi-version are listed in related links.

If you run multiple installations of IBM MQ on a server you must consider three questions:

1. Which installation is a queue manager associated with; see "Queue manager association" on page 58 ?
2. Which installation does an application load; see "Loading IBM MQ libraries" on page 58 ?
3. Which installation is an IBM MQ command run from; see "Command association" on page 60 ?

Queue manager association

Before Version 7.1, queue managers on UNIX, Linux, or Windows were associated with the only installation on the server. With Version 7.1, or later, installed on the same server as Version 7.0.1, you can change the association of a queue manager to a later version by running **setmqm** ; see setmqm. You cannot change the association of a queue manager running a release of IBM MQ earlier than Version 7.0.1 because you cannot install a later version of the product on a server with an installation of IBM MQ earlier than Version 7.0.1.

A queue manager is permanently associated with an installation, until you choose to change the association with the **setmqm** command. You cannot associate a queue manager with an installation at a lower command level than the current command level of the queue manager.

In Figure 11 on page 57, QM1 is associated with Inst_1. The association is made by running setmqm -m QM1 -n Inst_1. When QM1 is first started, after running **setmqm**, if QM1 was running Version 7.0.1, it is migrated to a later version. QM2 is associated with Version 7.0.1 because the association has not been changed.

Loading IBM MQ libraries

The application connections to the queue managers are established by calling MQCONN or MQCONNX in the normal way.

Which IBM MQ library an application loads depends on the configuration of the operating system loader and on the IBM MQ installation the queue manager is associated with.

In Figure 11 on page 57, the operating system loads the IBM MQ library from the Inst_1 installation for applications 1 and 3. It loads the IBM WebSphere MQ 7.0.1 library for application 2. The operating system has loaded the wrong library for application 3. Application 3 requires the IBM WebSphere MQ 7.0.1 libraries.

Figure 12 on page 59 shows what happens to application 3. Application 3 is connecting to QM2, and QM2 is associated with the IBM WebSphere MQ 7.0.1 installation. IBM MQ detects that the operating system has loaded the wrong library to process calls from application 3 to QM2. IBM MQ loads the correct library from the IBM WebSphere MQ 7.0.1 installation. It transfers the MQCONN or MQCONNX call to the IBM WebSphere MQ 7.0.1 library. Subsequent MQI calls that use the connection handle returned by MQCONN or MQCONNX, call entry points in the IBM WebSphere MQ 7.0.1 library.

Because IBM WebSphere MQ 7.0.1 libraries cannot load IBM MQ libraries from other installations, there is no corresponding application in Figure 12 on page 59 that loads a IBM WebSphere MQ 7.0.1 library and connects to a queue manager running Version 7.1. If you attempt a connection to QM1 with application 2, IBM MQ returns an error; see 2059 (080B) (RC2059): MQRC_Q_MGR_NOT_AVAILABLE.

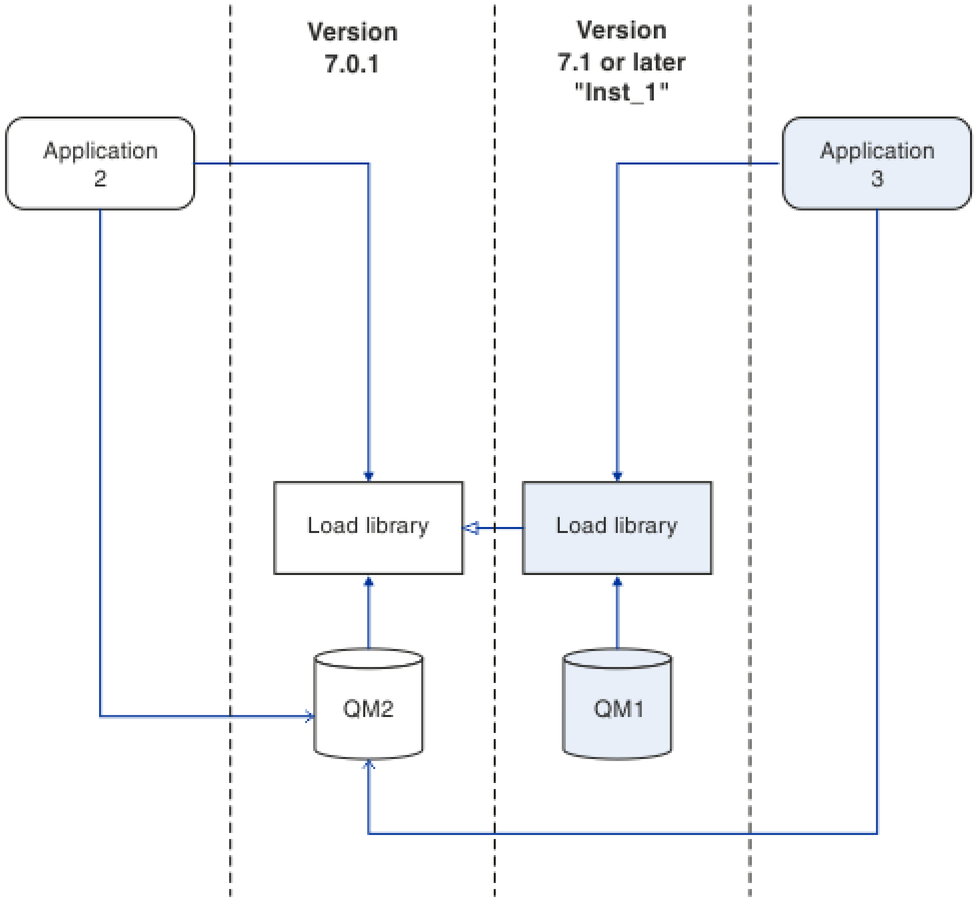


Figure 12. Loading calls in a different library

A Version 7.1, or later, IBM MQ library includes a routing capability that is based on the installation a queue manager is associated with. Earlier IBM MQ libraries do not have a routing capability. The operating system can load a library from any Version 7.1 installation, or later, and IBM MQ transfers MQI calls to the correct library.

The new loading capability of IBM MQ libraries in Version 7.1, or later, does not relax the restriction that an application compiled and linked at a later release level must not directly load an IBM MQ library at an earlier release level. In practice the restriction is of less significance than in earlier releases, because as long as the operating system loads a library at the same or later level than the library the application was compiled and linked with, IBM MQ can call any other level of IBM MQ on the same server from Version 7.0.1 upwards.

For example, suppose you recompile and link an application that is to connect to a Version 7.0.1 queue manager using the libraries shipped with Version 7.1. At run time the operating system must load the Version 7.1 libraries for the application, even though the application connects to a Version 7.0.1 queue manager. IBM WebSphere MQ 7.1 detects the inconsistency and loads the Version 7.0.1 library for the application. The same applies to any future release. If the application is recompiled and linked against a later release, then the application must load an IBM MQ library that matches the later release, even if it continues to connect to a Version 7.1 queue manager.

Your application might not be linked to an IBM MQ library, but instead calls the operating system directly to load an IBM MQ library. If the library that is loaded is from Version 7.1 or later, IBM MQ checks the library is from the installation that is associated with the queue manager. If it is not, IBM MQ loads the correct library.

Special migration considerations involving loading IBM MQ libraries

You might have been asked to modify the installation of an earlier IBM MQ release to satisfy the requirements of a build environment, or the IT standards in your organization. If you copied IBM MQ libraries to other directories, or created symbolic links, you ended up with an unsupported configuration. The requirement to move IBM MQ libraries to other directories was one of the reasons for changing the installation of IBM MQ on UNIX and Linux. You can now install IBM MQ into a directory of your choosing. You can also load IBM MQ libraries from the `/usr/lib` directory, which is normally on the default load path on UNIX and Linux systems.

A common IT standard or build environment requirement is to include IBM MQ libraries in the default load path on UNIX and Linux systems. IBM WebSphere MQ 7.1 has a solution. In Version 8.0 you can install IBM MQ into a directory of your own choosing, and IBM MQ can create symbolic links in `/usr` and its subdirectories. If you make a Version 7.1, or later, installation primary by using the **setmqinst** command, IBM MQ inserts symbolic links to the IBM MQ libraries into `/usr/lib`. As a result, the operating system finds the IBM MQ libraries in the default load path, if that includes `/usr/lib`.

Because IBM WebSphere MQ 7.1, or later, libraries transfer calls to the correct installation, defining that version installation as primary, also results in the correct libraries being loaded for any application that is built with a link to `/usr/lib`, regardless of which queue manager it connects to. Unfortunately, this solution does not work if you have a Version 7.0.1 installation on the server, because then you cannot define a Version 7.1, or later, installation as primary, and the Version 7.0.1 libraries do not load libraries from other installations. As an alternative to setting the later version installation primary, use **setmqenv** with the `-k` or `-l` options to achieve a similar result.

You can find more information in Connecting applications in a multiple installation environment.

Command association

Examples of commands are **dspmqver**, **setmqinst**, **runmqsc**, and **strmqm**. The operating system must find a command in an IBM MQ installation. Many commands also require a queue manager as an argument and assume the default queue manager, if a queue manager name is not provided as a parameter.

Unlike loading libraries, if a command includes a queue manager as a parameter, the command is not switched to the installation that is associated with the queue manager. You must use the **setmqenv** command to set up your environment correctly, so that any commands that you issue are run from the correct installation. You can provide a queue manager as a parameter to **setmqenv**, to set up the command environment for that queue manager; see Figure 13 on page 61.

On Windows, the **setmqinst** command sets global environment variables, and **setmqenv** local environment variables, including the PATH variable to find commands.

On UNIX and Linux, the **setmqinst** command copies symbolic links for a subset of the commands into `/usr/bin` ; see External library and control command links to primary installation on UNIX and Linux . The **setmqenv** command sets up local environment variables, including the search path to the binary folder in the installation directory.

setmqenv must be on the search path in order to run it. One reason for having a later version installation as primary is to be able to run setmqenv without having to configure the search path. If IBM WebSphere MQ 7.0.1 is installed on the server, no Version 7.1, or later, installation can be primary and IBM WebSphere MQ 7.0.1 does not have a setmqenv command. The consequence is, you must provide a path to run the setmqenv command to set up the command environment for any of the later version installations on the server.

Figure 13 on page 61 shows two examples of running **setmqenv** to set up the command environment for the copy of IBM MQ that is associated with the queue manager, `QM1`.

IBM WebSphere MQ for Windows Version 7.1

```
" MQ_INSTALLATION_PATH\bin\setmqenv" -m QM1
```

IBM WebSphere MQ 7.1 for UNIX and Linux

```
. MQ_INSTALLATION_PATH/bin/setmqenv -m QM1
```

*Figure 13. Running **setmqenv***

Related tasks

“Migrating IBM MQ library loading from an earlier version of the product to the latest version” on page 131
No change in the way IBM MQ libraries are loaded is normally required if you upgrade from an earlier version of the product to the latest version. You must have followed the instructions on building IBM MQ applications in Version 7.0.1 and you must replace IBM WebSphere MQ 7.0.1 with the latest version of the product. If you choose to take advantage of multi-installation in the latest version of the product, based on the side-by-side or multi-stage migration scenarios, you must modify the environment for the operating system to resolve IBM MQ dependencies for an application. Typically, you can modify the runtime environment, rather than relink the application.

“UNIX, Linux, and Windows: Single-stage migration to a later version” on page 92
Single-stage migration, is the term used to describe replacing the only installation of IBM MQ on a server, with a later release. Single stage migration is also known as `upgrading in place` or `in place upgrade` . Until Version 7.0.1.6, single-stage was the only migration scenario. Single-stage migration preserves existing scripts and procedures for running IBM MQ the most. With other migration scenarios you might change some scripts and procedures, but you can reduce the effect queue manager migration has on users.

“UNIX, Linux, and Windows: Side-by-side migration to a later version” on page 96

“UNIX, Linux, and Windows: Multi-stage migration to a later version” on page 100

“UNIX, Linux, and Windows: Staging maintenance fixes” on page 264
Use multiple installations of IBM MQ on the same server to control the release of maintenance fixes.

“UNIX: Migrating IBM MQ library loading from Version 7.0.1, or later, to the latest version” on page 136
Investigate whether applications connecting to the latest version of the product are linked to, and load libraries from, the correct installation.

“Windows: Migrating IBM MQ library loading from Version 7.0.1, or later, to the latest version” on page 133
Investigate whether applications connecting to the latest version of the product are linked to, and load libraries from, the correct installation.

Related reference

“Coexistence” on page 174
Queue managers, with different names, can coexist on any server as long as they use the same IBM MQ installation. On z/OS z/OS, UNIX, Linux, and Windows, different queue managers can coexist on the same server and be associated with different installations. In addition to queue managers coexisting on a server, objects, and commands must work correctly with different queue managers running at different command levels.

Related information

Changing the primary installation

Connecting applications in a multiple installation environment

Dynamic-Link Library Search Order

External library and control command links to primary installation on UNIX and Linux

Features that can be used only with the primary installation on Windows

Installation configuration file, mqinst.ini

setmqenv

setmqinst

setmqm

strmqm -e CMDLEVEL

Application compatibility and interoperability with earlier versions of IBM MQ

Connecting an application that is built against libraries shipped with a later version of IBM MQ to an earlier version IBM MQ is not supported. Avoid building applications against a later version, and redeploying them to a queue manager running at an earlier version, although some applications do work in practice.

IBM MQ applications do interoperate with applications running on earlier versions of IBM MQ, as long as they use no new function. IBM MQ clients can connect to queue managers running at an earlier version than the client, as long as the client uses no new functions.

An IBM MQ application that uses only functions provided by an earlier version of a queue manager can continue to send messages to the earlier version. It does not matter what version of IBM MQ an application is built on and connected to. It can exchange messages with an application connected to an earlier version of IBM MQ, as long as it does not use new function.

Consider these four cases; the first two cases are not supported though they might work in practice, the last two cases are supported. The first two cases require compatibility with an earlier version of IBM MQ. The last two cases rely on the interoperability between all versions of IBM MQ

1. Running an IBM MQ server application, built with a later version of IBM MQ, connecting to a queue manager running on a server with an earlier version of IBM MQ installed.
2. Running an IBM MQ client application, built with a later version of IBM MQ, on a client platform with an earlier client installation, connecting to a queue manager running on a server with a later version of IBM MQ installed.
3. Running an IBM MQ client application, built with a later version of IBM MQ, on a client platform with the later client installation, connecting to a queue manager running on a server with an earlier version of IBM MQ installed.
4. Exchanging messages between an IBM MQ client or server application, connected to a queue manager running on a server with a later version of IBM MQ installed, with applications connected to a queue manager running on a server with an earlier version of IBM MQ installed.

Plan to avoid the first two cases, as they are not guaranteed to work all the time. If you are running an incompatible configuration and you encounter a problem, you must rebuild your applications with the correct level of IBM MQ. You can then continue with problem diagnosis.

Multi-installation and application loading

The new loading capability of IBM MQ libraries in Version 7.1, or later, does not relax the restriction that an application compiled and linked at a later release level must not directly load an IBM MQ library at an earlier release level. In practice the restriction is of less significance than in earlier releases, because as long as the operating system loads a library at the same or later level than the library the application was compiled and linked with, IBM MQ can call any other level of IBM MQ on the same server from Version 7.0.1 upwards.

For example, suppose you recompile and link an application that is to connect to a Version 7.0.1 queue manager using the libraries shipped with Version 7.1. At run time the operating system must load the Version 7.1 libraries for the application, even though the application connects to a Version 7.0.1 queue manager. IBM WebSphere MQ 7.1 detects the inconsistency and loads the Version 7.0.1 library for the application. The same applies to any future release. If the application is recompiled and linked against a later release, then the application must load an IBM MQ library that matches the later release, even if it continues to connect to a Version 7.1 queue manager.

Examples

1. You decide to rebuild a client application. Can you deploy it to your production environment that contains some earlier versions of client and server platforms?

   The answer is no, you must upgrade all the client workstations you deploy to, at least to the version of the client you have built. The queue managers running on earlier versions of IBM MQ do not have to be upgraded. In practice all the clients are likely to work, but for maintainability you must avoid running incompatible levels of an application and the IBM MQ client.
2. You deploy some IBM MQ queue managers at a new version level. You have an existing IBM MQ application that you use to send messages between the servers. Do you rebuild the application to deploy it onto the new servers? Can you deploy the old version onto the new servers?

   The answer is, either. You can continue to deploy the existing version of the application onto all your servers, or you can deploy the rebuilt application onto the new servers. Either configuration works. IBM MQ supports running the existing application on later servers and sending messages from later application versions to earlier ones. What you must not do is to rebuild the application on the later version and redeploy it onto both the earlier and newer servers. IBM MQ does not support compatibility with earlier versions.

z/OS application stubs

z/OS

The stub modules that are listed are link-edited with applications and exits. The version 7 stub modules might not work with Version 6.

- `CSQASTUB`
- `CSQBRSSI`
- `CSQBRSTB`
- `CSQBSTUB`
- `CSQCSTUB`
- `CSQQSTUB`
- `CSQXSTUB`

To take advantage of the new APIs introduced in Version 7.0, for example `MQSUB` and `MQCB`, you must link-edit your application with stubs shipped with Version 7.0 or later, for an LE program, the sidedeck, see Building z/OS batch applications using Language Environment . Such an application will not run on an earlier version of the queue manager.

Application compatibility and interoperability with later versions of IBM MQ

IBM MQ applications run against later versions of a queue manager without recoding, recompiling, or relinking. You can connect an application that is built against libraries shipped with an earlier version of IBM MQ to a queue manager running at a later version of IBM MQ.

If you upgrade a queue manager to a later version, existing applications built against its earlier version work without change. Exceptions are noted in “Changes that affect migration” on page 173. Likewise applications connected to the IBM MQ Client, run against later versions of the client without recoding, recompiling, or relinking. You can deploy client applications built against earlier versions of the IBM MQ Client libraries to connect using later versions of the libraries.

All the following four cases are supported. The first two cases rely on the compatibility of later version of IBM MQ with applications built against earlier versions. The last two cases rely on the interoperability between all versions of IBM MQ.

1. Running an IBM MQ server application, built with a later version of IBM MQ, connecting to a queue manager running on a server with an earlier version of IBM MQ installed.

2. Running an IBM MQ client application, built with a later version of IBM MQ, on a client platform with an earlier client installation, connecting to a queue manager running on a server with a later version of IBM MQ installed.
3. Running an IBM MQ client application, built with a later version of IBM MQ, on a client platform with the later client installation, connecting to a queue manager running on a server with an earlier version of IBM MQ installed.
4. Exchanging messages between an IBM MQ client or server application, connected to a queue manager running on a server with a later version of IBM MQ installed, with applications connected to a queue manager running on a server with an earlier version of IBM MQ installed.

You might change the operating environment as a prerequisite of migrating to a new level of queue manager. The operating environment changes, rather than changes in IBM MQ itself, might require application change, recompilation, or relinking. Sometime the operating environment change affects only the development environment, and the operating environment supports applications built at an earlier level. In which case, you might be able to run existing applications built at the older level of the operating environment. You might not be able to build any new applications until the operating environment is upgraded.

In the future, after you have migrated queue managers and clients to the latest release level, consider changing your applications to take advantage of new capabilities.

z/OS application stubs

z/OS

The stub modules that are listed are link-edited with applications and exits. The Version 6.0 stub modules continue to work with Version 8.0.

- CSQASTUB
- CSQBRSSI
- CSQBRSTB
- CSQBSTUB
- CSQCSTUB
- CSQQSTUB
- CSQXSTUB

To take advantage of the new APIs introduced in Version 7.0, for example MQSUB and MQCB, you must link-edit your application with stubs shipped with Version 7.0 or later, for an LE program, the sidedeck, see Building z/OS batch applications using Language Environment . Such an application will not run on an earlier version of the queue manager.

Compatibility between different versions of an IBM MQ MQI client and a queue manager

Any version and release of an IBM MQ MQI client can connect to any version and release of an IBM MQ queue manager. The MQI channel is automatically configured to the latest version that both the client and server support. If the client and server are different versions, the client application must use only the functions in the earlier version.

The compatibility between clients and queue managers applies only to the version and release (V.R) of the product. The statement of compatibility does not necessarily apply to the modification and fix pack level (M.F) of the product.

If there are known problems at a specific V.R.M.F of the product, an upgrade to a more recent fix pack for the same Version.Release is necessary.

When you upgrade a queue manager to a different version, you automatically upgrade IBM MQ libraries. The libraries are used by IBM MQ MQI client and server applications running on the same server as the queue manager. To access new functions from remote clients, you must also upgrade the IBM MQ

MQI client installation on remote workstations. The IBM MQ MQI client includes the IBM MQ MQI client libraries.

Remote clients that have not been upgraded continue to work with an upgraded queue manager. The behavior of the client application might, in rare cases change. You must consult "Changes that affect migration" on page 173, to find out whether changes in the current version affect your client applications.

Remote clients that are connected to upgraded queue managers can use the new functions in the release. If an upgraded remote client is connected to a queue manager that has not been upgraded, it must not use new functions. In rare cases, the behavior of the client might change; see "Changes that affect migration" on page 173.

You can generally assume that upgrading the IBM MQ MQI client does not require you to recompile or relink the client application. You can also continue to use the same connection to the queue manager. If changes are required, they are identified in "Migrating a queue manager to the latest release" on page 107, for the particular migration path and platform you are concerned with.

The Client Channel Definition Table (CCDT) is an interface to customize the connection between an IBM MQ Client and a queue manager. Entries in the tables are client connections, which are defined using a queue manager. The version of a CCDT is the version of the queue manager used to define the client connections. If an IBM MQ MQI client uses CCDT to connect to a queue manager, the CCDT can be at a version greater than, less than, or equal to that of the client.

You can connect to a queue manager with an earlier IBM MQ Client or an earlier CCDT. If you are using a CCDT, and you plan to use new client channel configuration options, such as shared conversations, you must upgrade the CCDT, and therefore the IBM MQ MQI client installation to the new version.

MQI client: Client Channel Definition Table (CCDT)

The client channel definition table has changed from Version 6.0 to Version 7.5. Existing clients must continue to use existing CCDTs. To use a Version 7.5 CCDT, you must update the client.

You can connect an IBM MQ MQI client application to any level of queue manager. If the client uses a CCDT, it must use a CCDT built by the same or earlier version of queue manager. If the client connects without using a CCDT, this restriction does not apply.

In a common migration scenario, if you upgrade a Version 6.0 queue manager to Version 7.5, and you do not create new CCDTs for its clients, the clients connect to the Version 7.5 queue manager without any changes being required. Client behavior might change as a result of changes to the queue manager.

Another common migration scenario is to update some queue managers and some clients to Version 7.5, leaving some queue managers and clients at Version 6.0. In this scenario, you want to update the CCDT for Version 7.5 IBM MQ MQI client connected to Version 7.5 queue managers to Version 7.5, so that the clients can exploit Version 7.5 function fully. The new clients also connect to Version 6.0 queue managers. Existing clients connect to both Version 6.0 and Version 7.5 queue managers. In order that Version 7.5 clients can exploit new Version 7.5 function, you must deploy a Version 7.5 CCDT to Version 7.5 clients. Version 6.0 clients must continue to use the Version 6.0 CCDT. Both sets of clients can connect to both sets of queue managers, regardless of the CCDT they are using.

If the client is an IBM MQ MQI client, the version of the IBM MQ MQI client libraries linked to by the client must be the same or greater than the version of the queue manager that was used to build the CCDT. If the client is a Java or JMS client, then the client must be built with versions of the IBM MQ JAR files that are the same or greater than the queue manager that was used to build the CCDT.

To upgrade a Version 6.0 IBM MQ MQI client to use a Version 7.5 CCDT, you must upgrade the IBM MQ MQI client installation to Version 7.5. Unless you decide to do so for other reasons, do not rebuild the client application.

To upgrade a Version 6.0 Java or JMS client to use a Version 7.5 CCDT, redeploy the IBM MQ JAR files to the client workstation. You do not need to rebuild the Java or JMS client with the new JAR files.

MQI client: Client configuration stanzas moved into a new configuration file

Client configuration information is moved from existing configuration stanzas into a new configuration file, `mqclient.ini`.

Moving client configuration information affects existing settings; for example:

- Set the TCP `KeepAlive` attribute for client connections in `mqclient.ini` ; for example:

  ```
  TCP:
  KeepAlive = Yes
  ```

  An existing setting in `qm.ini` is ignored.
- Set `ClientExitPath` in `mqclient.ini` ; for example:

  ```
  ClientExitPath:
  ExitsDefaultPath=/var/mqm/exits
  ExitsDefaultPath64=/var/mqm/exits64
  ```

  An existing setting in `mqs.ini` is moved to the client configuration file when you upgrade the client. If you add values to `mqs.ini`, they are ignored.
- Set `JavaExitsClasspath` in `mqclient.ini`.

  Do not continue to use the Java system property `com.ibm.mq.exitClasspath`. Existing settings continue to work, but they are deprecated. The setting in `mqclient.ini` has precedence over the Java system property.

Related information

The IBM MQ classes for JMS configuration file

Assigning channel exits for IBM MQ classes for JMS

IBM MQ client configuration file

MQI client: Default behavior of client-connection and server-connection channels

In Version 7.0 the default settings for client and server connection channels changed to use shared conversations. This change affects the behavior of heartbeats and channels exits, and can have an impact on performance.

Before Version 7.0, each conversation was allocated to a different channel instance. From Version 7.0, the default for client and server connections is to share an MQI channel. You use the **SHARECNV** (sharing conversations) parameter to specify the maximum number of conversations that can be shared over a particular TCP/IP client channel instance. The possible values are as follows:

SHARECNV(0)

This value specifies no sharing of conversations over a TCP/IP socket. The channel instance behaves exactly as if it was a Version 6.0 server or client connection channel, and you do not get the extra features such as bi-directional heartbeats that are available when you set **SHARECNV** to 1 or greater. Only use a value of `0` if you have existing client applications that do not run correctly when you set **SHARECNV** to 1 or greater.

SHARECNV(1)

This value specifies no sharing of conversations over a TCP/IP socket. Performance on distributed servers is similar to that for a value of `0`. Client heartbeating (whether in an MQGET call or not) and read ahead are available, and channel quiescing is more controllable. You can usually use this setting with existing Version 6.0 client applications.

SHARECNV(2) to SHARECNV(999999999)

Each of these values specifies the number of shared conversations. If the client-connection **SHARECNV** value does not match the server-connection **SHARECNV** value, then the lowest value is used. The default value is `SHARECNV(10)`, which specifies 10 threads to run up to 10 client conversations per channel instance. However, on distributed servers there are performance issues with SHARECNV channels that can be eased by using `SHARECNV(1)` wherever possible.

For all **SHARECNV** values of 1 or greater, the channel supports the following features:

- Bi-directional heartbeats
- Administrator stop-quiesce
- Read-ahead
- Asynchronous-consume by client applications

You can also set the MQCONNX option, MQCNO_NO_CONV_SHARING and connect the application to a channel with **SHARECNV** set to a value greater than 1. The result is the same as connecting the application to a channel with **SHARECNV** set to 1.

Performance

The change in Version 7.0 to use shared conversations, and further enhancements introduced in Version 8.0, can impact performance on distributed servers. See Tuning client and server connection channels.

Heartbeats

From Version 7.0, heartbeats can flow across the channel at any time in either direction. The SHARECNV(0) and Version 6.0 behavior is for heartbeats to flow only when an MQGET call is waiting.

Channel exits

The behavior of a client or server connection channel exit changes when the channel is sharing conversations (that is, when you set **SHARECNV** to a value greater than 1). It is unlikely, but possible, that the change affects the behavior of existing exits. The change is as follows:

- Send or receive exits can alter the MQCD structure on an MQXR_INIT call. The effect of these exits differs, depending on whether the conversation is shared with other conversations on the same channel:
  - If the MQCXP SharingConversations field passed to the exit instance is set to FALSE, this exit instance is the first, or only, conversation on the channel instance. No other exit can be changing the MQCD at the same time, and changes that are made to the MQCD can affect the way that the channel runs.
  - If the MQCXP SharingConversations field passed to the exit instance is set to TRUE, this exit instance is a subsequent conversation. It is sharing the channel instance with other conversations. Changes made to the MQCD in the exit instance are retained in the MQCD but do not affect the way the channel runs.
- Send, receive, and security exit instances can alter the MQCD, when the MQCXP SharingConversations field is set to TRUE. Exit instances on other conversations might be changing the MQCD at the same time. Updates written by one exit instance can be overwritten by another instance. It might be necessary to serialize access to the MQCD across these different exit instances to maintain the consistency of fields in MQCD.

Updating MQCD when the SharingConversations field is set to TRUE does not affect the way the channel runs. Only alterations made when the MQCXP SharingConversations field is set to FALSE, on an MQXR_INIT call, change channel behavior.

Related information

Using sharing conversations

Channel-exit programs for MQI channels

Using read ahead

Stopping MQI channels

Tuning client and server connection channels

HeartbeatInterval (MQLONG)

SharingConversations (MQLONG)

ALTER CHANNEL

The Asynchronous consume sample program

MQI client: MQPUT1 sync point behavior change

An MQPUT1 call by an IBM MQ MQI client application that failed in IBM WebSphere MQ 6.0 can now sometimes succeed. The failure is returned to the application later, if it calls MQCMIT. For the change in behavior to occur, the MQPUT1 must be in sync point.

In the scenario, "Example call sequence that demonstrates the change in behavior" on page 68, an MQPUT1 call can succeed where it failed in Version 6.0. The result occurs when all the following conditions are met:

- Both client and queue manager are later than Version 6.0.
- The application program is connected to the queue manager as a client application
- MQPMO_SYNCPOINT is set in the Put Message Options structure, MQPMO

You can make the IBM MQ MQI client behave like Version 6.0. Set **Put1DefaultAlwaysSync** to YES in the **CHANNELS** stanza of the client configuration file; see Figure 14 on page 68.

```
Channels:
Put1DefaultAlwaysSync=YES
```

*Figure 14. Add **Put1DefaultAlwaysSync** to mqclient.ini*

Example call sequence that demonstrates the change in behavior

1. MQCONN to queue manager from an IBM MQ MQI client application.
2. MQPUT1 to a nonexistent queue with the MQPMO_SYNCPOINT option
3. MQDISC

In IBM WebSphere MQ 6.0 the MQPUT1 call ends with MQCC_FAILED and MQRC_UNKNOWN_OBJECT_NAME(2085). Running with a client and server later than Version 6.0, the MQPUT1 call ends with MQRC_NONE and MQCC_OK.

Related information

CHANNELS stanza of the client configuration file

Queue manager migration

After upgrading an installation, queue manager migration might be required. Migration takes place when you start a queue manager.

z/OS On z/OS, queue manager migration is required after upgrading to a different version, release, or maintenance level. The upgrade changes the command level. The current command, or VRM, level is shown in the z/OS console log.

On other platforms, queue manager migration is always required for changes in the first two digits of the VRMF code. Changes in the maintenance and fix level, M and F in the VRMF code, never cause automatic queue manager migration. No migration was required for the upgrade from Version 7.0 to Version 7.0.1. The change from Version 7.0 to Version 7.0.1 did change the command level from 700 to 701. From Version 7.1 onwards, a change in the command level always requires queue manager migration, but if the change is shipped in a maintenance or fix pack, you have the choice of whether to increase the command level, and cause queue manager migration.

Command level always increases with a change in version or release. If you decide to use new function introduced in a maintenance level upgrade, you must change the command level. The converse is not the case. You do not have to change the command level when the fix level changes. You can decide to install the fix pack, but not use the new function. Whether or not you use the new function, the installation of the fix pack increases the maximum command level supported by the installation. Run the **dspmqver** command to display the current maximum supported command level.

Queue manager migration is the process of converting persistent queue manager data from one version to another. Persistent queue manager data includes log files and data in the queue manager directory. The data records changes to objects such as messages, subscriptions, publications, queue managers, channels, queues, and topics.

Queue manager migration is required and largely automatic. z/OS On z/OS, you must migrate queue managers manually between compatability mode and new function mode by setting the **OPMODE** parameter.

After migrating to a new release z/OS, or on z/OS, after setting the **OPMODE** to `NEWFUNC`, the queue manager cannot be restored to an earlier release level.

You can reduce the downtime and risk caused by queue manager migration, by verifying the new version first, using a different queue manager. Unless the platform supports queue manager coexistence, you need to perform the verification on a different server, or in a virtualized environment on the same server. If the platform you are upgrading supports queue manager coexistence, you can install the new version of IBM MQ on the same server, verify it, and minimize downtime to the time required to stop, backup, and restart the queue manager.

Note: If you are migrating a queue manager through multiple release levels, one level at a time, you must start the queue manager after each upgrade to migrate it. You must also start all the channels, to ensure they are migrated.

Related concepts

"The version naming scheme for IBM MQ (On platforms other than z/OS)" on page 21
IBM MQ releases have a four-digit Version, Release, Modification, and Fix (VRMF) level code.

"The version naming scheme for IBM MQ for z/OS" on page 20
On IBM MQ for z/OS, releases have a three-digit Version, Release, and Maintenance (VRM) level code. The code is significant; it identifies the service life of a release. To run a queue manager at a different VRM level, you must migrate the queue manager, its applications, and the environment in which it runs. Depending on the migration path, the migration might require more or less effort.

"Upgrade, migration, and maintenance of IBM MQ (On platforms other than z/OS)" on page 51
You can install new releases of IBM MQ to upgrade IBM MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of IBM MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

"Upgrade, migration, and maintenance of IBM MQ on z/OS" on page 49
You can install new releases of IBM MQ to upgrade IBM MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same z/OS instance. Running a queue manager at a higher level requires migration. Maintenance differs from upgrading. To maintain a level of IBM MQ, you apply Program Temporary Fixes (PTFs) to the installed code.

Related tasks

"Migrating a queue manager to the latest release" on page 107
The procedures for migrating a queue manager to the latest release are detailed in the following topics.

Related information

dspmqver

Reverting a queue manager to a previous version

You can remove an upgrade before you have started a queue manager. After a queue manager has been started, if you remove the upgrade, the queue manager will not work.

IBM i distributed You must back up your system before starting migration. You can either back up queue manager data, or use a backup queue manager; see Backing up and restoring IBM MQ. To back up, you must stop the queue manager.

You can reduce the downtime and risk caused by queue manager migration, by verifying the new version first, using a different queue manager. Unless the platform supports queue manager coexistence, you need to perform the verification on a different server, or in a virtualized environment on the same server. If the platform you are upgrading supports queue manager coexistence, you can install the new version of IBM MQ on the same server, verify it, and minimize downtime to the time required to stop, backup, and restart the queue manager.

z/OS On z/OS, before Version 7.0.1, it is possible to revert to a previous level, as long as you have applied the correct PTFs.

z/OS On z/OS, from Version 7.0.1 onwards, it is impossible to revert to an earlier release after running with **OPMODE** *NEWFUNC*. Otherwise, you can backward migrate, as described in Migration PTFs.

z/OS On z/OS, you must migrate queue managers manually between compatibility mode and new function mode by setting the **OPMODE** parameter. If you have never switched a queue manager to new function mode, you can still run it against the earliest release it is compatible with. You must have applied compatibility PTFs to the earlier release before starting a queue manager at the new command level. The compatibility level is shown in the log.

Related concepts

"Upgrade, migration, and maintenance of IBM MQ (On platforms other than z/OS)" on page 51
You can install new releases of IBM MQ to upgrade IBM MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of IBM MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

"Upgrade, migration, and maintenance of IBM MQ on z/OS" on page 49
You can install new releases of IBM MQ to upgrade IBM MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same z/OS instance. Running a queue manager at a higher level requires migration. Maintenance differs from upgrading. To maintain a level of IBM MQ, you apply Program Temporary Fixes (PTFs) to the installed code.

Related reference

"z/OS: OPMODE" on page 272
The availability of new functions and backward migration for IBM MQ for z/OS is controlled by the **OPMODE** parameter in the **CSQ6SYSP** macro. To access V8.0 capabilities, change the value of **OPMODE** to `OPMODE=(NEWFUNC,800)`. To restrict the use of new capabilities, and retain the ability to revert the queue manager to its earlier level, leave **OPMODE** at its default setting, `OPMODE=(COMPAT,800)`.

Related information

Backing up and restoring IBM MQ

IBM MQ MQI client migration

IBM MQ MQI client migration is the process of converting IBM MQ MQI client configurations, and client and server channels from one version to another. Client migration can take place after upgrading the IBM MQ MQI client, and are reversible.

Client migration on the client workstation is optional and manual. Client migration on the server is required and automatic. See "Changes that affect migration" on page 173 for a list of any client changes. You must upgrade an IBM MQ MQI client before migrating a client workstation to make use of new configuration options. You can make configuration changes to client and server connection channels on the server, but they have no effect on a client workstation, until the client is upgraded.

An example of client migration performed at the client workstation is to manually migrate configuration settings to the `mqclient.ini` configuration file.

An example of combined client and server migration is the deployment of a new client connection definition table (CCDT). To use a new version of the CCDT, generate the table on a queue manager that is at the new code level. Deploy the table to clients that are going to use it. To deploy the table to a client, you first must update the client to at least the same level as the queue manager that created the table.

IBM MQ MQI client s are interoperable with earlier and later versions of IBM MQ. Upgrading the IBM MQ MQI client makes new function available to client applications, and is important to maintain the service level. Migrating an IBM MQ MQI client gives it access to new configuration options.

The IBM MQ MQI client libraries, such as `mqic.dll`, are dynamic, and the application linkages to the libraries do not normally change. You do not relink a client application to pick up new IBM MQ client libraries. The client picks up the new library next time the library is loaded by the client application. Do not move libraries from their installed directory. Linking to libraries in anything other than their installed directory is an unsupported configuration.

Related concepts

"Application compatibility and interoperability with earlier versions of IBM MQ" on page 62

"Application compatibility and interoperability with later versions of IBM MQ" on page 63
IBM MQ applications run against later versions of a queue manager without recoding, recompiling, or relinking. You can connect an application that is built against libraries shipped with an earlier version of IBM MQ to a queue manager running at a later version of IBM MQ.

Related reference

"MQI client: Client Channel Definition Table (CCDT)" on page 65
The client channel definition table has changed from Version 6.0 to Version 7.5. Existing clients must continue to use existing CCDTs. To use a Version 7.5 CCDT, you must update the client.

"MQI client: Default behavior of client-connection and server-connection channels" on page 66

"MQI client: MQPUT1 sync point behavior change" on page 68
An `MQPUT1` call by an IBM MQ MQI client application that failed in IBM WebSphere MQ 6.0 can now sometimes succeed. The failure is returned to the application later, if it calls `MQCMIT`. For the change in behavior to occur, the `MQPUT1` must be in sync point.

Migrating an IBM MQ MQI client on UNIX platforms, and Windows to the latest version

Before migrating an IBM MQ MQI client, create a migration plan. Stop all IBM MQ activity on the client workstation. Upgrade the IBM MQ MQI client installation. Make any essential configuration and application changes.

Before you begin

1. Create a migration plan. Use the planning task, "IBM MQ migration planning to the latest version on UNIX platforms, Windows, and IBM i" on page 80, as a guide.

Procedure

1. Review the latest IBM MQ requirements; see Checking requirements.
2. Review all the changes in IBM MQ that affect you; see What's changed in IBM MQ 8.0 and "Changes from Version 7.0.1 or later to Version 8.0" on page 177
3. End all IBM MQ activity on the workstation.
4. Upgrade the client.

- To upgrade a client installation on a workstation; see the appropriate topic for the platform, or platforms, that your enterprise uses:
  - Client installation procedure on an AIX workstation
  - Client installation procedure on an HP-UX workstation
  - Client installation procedure on an Linux workstation
  - Client installation procedure on an Solaris workstation
  - Client installation procedure on an Windows workstation
- To upgrade a client installation on a server; see Installing an IBM MQ MQI client on the same computer as the server

What to do next

After upgrading the IBM MQ MQI client, you must check the client channel configuration, and verify that your IBM MQ MQI client applications work correctly with the latest version of the product.

IBM i: Migrating an IBM MQ classes for JMS and Java client

If you have IBM MQ Java SupportPac MA88 installed, you must uninstall it first.

Before you begin

SupportPac MQ88 is installed.

If you try to install the latest version of IBM MQ classes for Java anyway, the installation fails with a warning requesting you to uninstall the old client. You must follow the steps in this task to uninstall IBM MQ classes for Java and IBM MQ classes for JMS.

A previous version of IBM MQ classes for Java is installed.

Installation of the latest version of IBM MQ classes for Java uninstalls the previous version automatically. Do not follow the steps in this task.

About this task

The steps in this task uninstall the IBM MQ classes for JMS and Java.

Procedure

To uninstall the previous IBM MQ Java client:

1. Delete the QMQMJAVA library and the /QIBM/ProdData/mqm/java directory, by issuing the command:

   ```
   DLTLICPGM LICPGM(5648C60) OPTION(*ALL)
   ```

2. If the previous step failed to delete the IFS directory /QIBM/ProdData/mqm/java and its subdirectories, use the **EDTF** command, for example:

   ```
   EDTF STMF('/QIBM/ProdData/mqm')
   ```

   and select option 9 against the java directory.

Restoring an IBM MQ MQI client and client connection to a previous release

If you restore an IBM MQ MQI client to a previous code level, you must undo the configuration changes manually.

About this task

It is unusual to restore earlier IBM MQ MQI client libraries to a workstation. The principal tasks are listed in the following steps.

Procedure

1. End all IBM MQ activity on the workstation.
2. Uninstall the IBM MQ MQI client code for the current version.
3. Follow the client installation procedure for the appropriate platform to install the IBM MQ MQI client for the previous release.
4. If you configured a Client Connection Definition Table (CCDT) for a queue manager using the current version, revert to using a table created by a queue manager for the previous release.

   The CCDT must always be created by a queue manager on the same, or earlier, release to the client.

Application migration and interoperation

IBM MQ supports running applications compiled and linked against previous versions of IBM MQ, with later levels of IBM MQ.

To migrate an application to run with a new level of IBM MQ, disconnect an application from the queue manager. Reconnect it when the queue manager is running again. However, it takes only one small difference in the interface between IBM MQ and the application to break an application, or make it behave wrongly. Sometimes a problem does not show up for a long time. For this reason, you must always test your applications against a new version of IBM MQ. The suggested extent of testing varies depending on the extent of the changes in IBM MQ ; see z/OS “Characteristics of different types of upgrade on z/OS” on page 50 or “Characteristics of different types of upgrade” on page 51.

Application migration refers to four kinds of changes.

1. Application changes that are consequent to upgrading the operating environment along with the queue manager. Rarely, linkage conventions change. The most likely reason for a linkage change is switching from 32 bit to a 64 bit environment. If you are using SSL or TLS you might have to relink with a new secure library.
2. Changes that you must make to the application in order to run an application against a new level of queue manager. Changes of this sort are uncommon. However, you must check “Changes that affect migration” on page 173 to see if any changes might affect your applications.
3. Changes that are not required, but that you might want to make in future, perhaps because you have a business reason to modify an application.
4. Changes to applications that are supplied by IBM, or other vendors, that require you to run migration utilities. The utilities convert the applications to running on the new version of IBM MQ.

Do not load IBM MQ libraries from an earlier level. IBM MQ does not support connecting server applications loading libraries from the earlier level to connect to a later level of queue manager. On UNIX, Linux, and Windows platforms, the application load path must be set up to the location of the IBM MQ server libraries. You do not have to recompile and relink an application. Applications compiled and linked against an earlier version of IBM MQ can load libraries from a later version.

On IBM i, UNIX, Linux, and Windows, from Version 7.1 onwards, IBM MQ loads the library from the installation the application is connecting to. An application must initially load a library of at least the same level as the application linked to. IBM MQ then loads the correct version of the library from the installation that the queue manager is associated with. If you have two installations of the same version, but at different fix levels, IBM MQ chooses which library to load. The choice is based on the queue manager the application is connected to. If an application is connected to multiple queue managers, it is possible that multiple libraries are loaded.

To help you write applications that can exchange messages with earlier versions of the product, IBM MQ provides data type versioning. Data type versioning assists you in exchanging messages that are compatible with target queue managers. A good programming practice is to set the version number of a data structure explicitly. Do not assume that the default version is the one you require. By setting the version explicitly, you are forced to look up what version to use. The description of the data type version tells you what level of queue manager supports that version.

It is poor practice to set the data type version to the current version. If you recompile your program against a new version of IBM MQ, the data type version might change with unexpected consequences.

Client applications are more likely to connect to different queue managers than applications written for a specific server. Plan carefully when writing an application that is to connect to different versions of a queue manager, and to queue managers on different platforms. The default values of some IBM MQ constants, such as `MQPMO_SYNCPOINT, MQPMO_NO_SYNCPOINT` differ between platforms. Some functions are not available on all platforms.

You must be aware of, and code to, the capabilities of all the queue managers the application interacts with. It requires planning and design to write an application that works with different versions of a queue manager. There is no API provided with IBM MQ to restrict an application to a function subset common to the set of queue managers it interacts with. To improve interoperability, some developers choose to provide an MQI wrapper layer, or use MQI API exits, to control the functions programs use.

Related concepts

“Application compatibility and interoperability with earlier versions of IBM MQ” on page 62

“Application compatibility and interoperability with later versions of IBM MQ” on page 63
IBM MQ applications run against later versions of a queue manager without recoding, recompiling, or relinking. You can connect an application that is built against libraries shipped with an earlier version of IBM MQ to a queue manager running at a later version of IBM MQ.

Queue manager cluster migration

You can migrate queue managers in a cluster all at once, or one at a time, which is called a staged migration. Migrate full repository queue managers in a cluster before partial repository queue managers.

Cluster queue managers can participate in clusters with other queue managers running at different versions, which is why a staged migration is possible. Being able to stage a migration is important, as migrating each queue manager in a cluster takes time. By staging the migration, which leaves other queue managers that are in the cluster running, you reduce the effect of queue manager downtime on applications.

Migrate queue managers with full repositories first. Then migrate the other queue managers, which have partial repositories, one at a time. Complete migration of the entire cluster before starting to use new functions.

If you do have to start using new functions before completing migration of the entire cluster, you might need to refresh the partial repositories. After each migration of a queue manager with a partial repository, issue the **REFRESH CLUSTER** command on the newly migrated queue manager. The command updates the cluster records in the newly migrated queue manager, potentially receiving updates for any new attributes. Do not do this step if you migrated the entire cluster before using new function. The **REFRESH CLUSTER** command takes a long time for all the changes to work through the cluster.

Note: For large clusters, use of the **REFRESH CLUSTER** command can be disruptive to the cluster while it is in progress, and again at 27 day intervals thereafter when the cluster objects automatically send status updates to all interested queue managers. See Refreshing in a large cluster can affect performance and availability of the cluster.

If full repositories are not migrated before partial repositories, the cluster continues to work, but without all the new features in a version working as expected. To work predictably, the full repository queue managers must be at the new command level to be able to store information from the rest of the cluster that arises from using new features.

For example, the information might be a new channel attribute, such as shared conversations, which were introduced in Version 7.0. Information about the shared conversation attribute of a channel between two other Version 7.0.1 queue managers can be stored in a Version 7.0 full repository, but not in a Version 6.0 repository. If information about a channel with the shared conversation attribute is updated from the Version 6.0 full repository, the definition loses its shared conversation attribute. "How mixed version cluster repositories are updated" on page 75 explains how information is updated in a mixed-version cluster.

Note: If a queue manager is a member of a cluster, and is running at a release earlier than Version 6.0, you must migrate the queue manager to Version 7.0.1, before migrating it to the latest release. You must start the queue manager after the first migration step, before proceeding to Version 8.0.

How mixed version cluster repositories are updated

Repositories store records for an object in a cluster in the version of the record format that matches the version of the queue manager hosting the repository. Repository queue managers forward object records, before they are stored, in the format that they are received in. The recipient ignores fields from a newer version, and uses default values for fields that are not present in the record.

Cluster repositories hold records that represent objects, for example, a queue record represents a cluster queue. A full repository holds records for all objects in the cluster. Partial repositories hold records for local objects and remote objects that are used locally. A repository record can hold information only about attributes at the same command level as the queue manager holding that repository. So for example, a Version 6.0 repository contains only Version 6.0 level attribute information. A Version 8.0 repository contains all Version 6.0 records, plus Version 8.0 records containing additional Version 8.0 attributes.

A repository stores a record it receives in its own version. If the record it receives is at a later version, the later version attributes are discarded when the record is stored. A Version 6.0 queue manager receiving information about a Version 8.0 queue manager stores only Version 6.0 information. A Version 8.0 repository receiving a Version 6 record stores default values for attributes introduced in the version 7. The defaults define the values for the attributes that are not included in the record it receives.

A repository normally sends records in its own version format, which is the same as the format it has stored them in. There is one exception to this rule. When a full repository receives a record from a partial repository, it is immediately forwarded in the same format. So if a Version 6.0 full repository were to receive a record from a Version 8.0 partial repository, it would forward the Version 8.0 record. It sends the record to any other full repositories, and any other partial repositories that have subscriptions that match the record.

A partial repository reflects whichever full repository sent it the latest update to a record. As a consequence, you might see the information held by a Version 8.0 partial repository for new Version 8.0 attributes changing unexpectedly. The values might change from actual Version 8.0 information to default values. The changes occur if the full repositories in the cluster are at different levels. Migrate full repositories first to avoid instability.

A partial repository sends information about its objects to a full repository periodically at least once every 27 days. Information is sent about any object when it is altered or defined. See How long do the queue manager repositories retain information?

After migrating all full repositories to Version 8.0, some attributes might hold default values. The attributes might hold default values in place of actual values, if a repository has not received an update. You can refresh the repository in either of two ways:

- Alter the object which the record containing default values represents, for example, using `ALTER QL` for a local queue. The alteration forces the local repository to send the record again.
- Issue the **REFRESH CLUSTER** command on the partial repository which holds the record containing default values. **REFRESH CLUSTER** forces the partial repository to discard the record containing default values and get a new record as required.

Note: For large clusters, use of the **REFRESH CLUSTER** command can be disruptive to the cluster while it is in progress, and again at 27 day intervals thereafter when the cluster objects automatically send status updates to all interested queue managers. See Refreshing in a large cluster can affect performance and availability of the cluster.

In summary, for the most predictable, and fastest migration, when you stage cluster migration do these steps in the following order:

1. Migrate the queue managers with full repositories.
2. Migrate the queue managers with partial repositories.
3. Start using new function in the cluster.

Related information

How long do the queue manager repositories retain information?

z/OS Queue-sharing group migration

You can combine queue managers from different releases in a queue-sharing group. Limit the time you manage a mixed group to only as long as it takes to migrate all the queue managers to the same command level. You cannot combine a queue manager at Version 8.0 or later in the same queue-sharing group as queue managers earlier than Version 7.0.1. You must update all queue managers in a queue-sharing group with a coexistence PTF, before migrating any of them. All queue managers in the queue-sharing group must be at the same version before any queue manager can be migrated to Version 8.0.

When you migrate queue managers in a queue-sharing group, aim to migrate all the queue managers to the new version as soon as you can. Queue-sharing groups can contain queue managers with a restricted mixture of versions. A mixture of queue managers in a queue-sharing group is supported so that you can migrate and test the upgrade of each queue manager. Migrate each queue manager, one at a time, leaving the queue-sharing group running. Mixed groups are harder to administer, than if all the queue managers are at the same version.

Related reference

"z/OS: MQSC commands in a mixed queue-sharing group" on page 176
Existing **MQSC** commands using new keywords and attribute values can be entered for routing to a migrated queue manager. You can enter the commands on any queue manager. Route the commands using **CMDSCOPE**. Commands with new keywords and attribute values, or new commands, routed to a previous version of queue manager, fail.

"z/OS: Properties of objects in a mixed queue-sharing group" on page 176
Attributes that did not exist in earlier versions can be created and altered on queue managers of a later version in a mixed queue-sharing group. The attributes are not available to queue managers in the group that are at an earlier level.

"z/OS: Queue sharing group coexistence" on page 176

Migrating a queue manager in a high-availability configuration

Follow standard procedures to migrate a queue manager that is part of a high-availability configuration. (On platforms other than z/OS.)

High-availability configurations of queue managers can increase the availability of IBM MQ applications. If a queue manager, or server fails, it is restarted automatically on another server. You can arrange for IBM MQ MQI client applications to automatically reconnect to the queue manager. Server applications can be configured to start when the queue manager starts.

IBM i distributed High-availability configurations are implemented either by using a high-availability cluster solution or by using multi-instance queue managers. Red Hat Cluster Suite or Microsoft Cluster Service (MSCS) are examples of high-availability cluster solutions.

z/OS On z/OS there are several alternative techniques to increase queue manager availability; see Availability. Migration considerations on z/OS depend on the availability techniques that are employed, and are not described in this topic. The term high-availability configuration refers only to queue managers in configurations on platforms other than z/OS.

The overall principles involved in queue manager migration in a high availability configuration are the same, whether you are migrating a multi-instance queue manager or a high-availability cluster. In either case, the principles are as follows:

1. You must not restart a queue manager at a lower command level than the one it was previously running.
2. You cannot upgrade the code an active queue manager is running.
3. You cannot back up an active queue manager.

Overall steps to migrate a queue manager in a multi-instance queue manager configuration

The following terms are relevant:

active queue manager instance
A queue manager instance that has been started permitting standby instances, and is running.

standby queue manager instance
A queue manager instance that has been started permitting standby instances, and is in standby. It is ready to take over from the active instance automatically.

Base your migration procedure on the following steps.

1. Before you start the migration process, create a different queue manager on a server on which you have installed the upgrade. Test the upgrade by performing whatever verification checks that your organization requires.
2. If you have a pool of servers that you pick from when starting a queue manager instance, upgrade IBM MQ on the servers that are in the pool and are neither active or acting as a standby.
3. Stop the standby queue manager instance. Make sure that you have no system management procedure running that restarts the instance automatically.
4. If you do not have a pool of servers, upgrade IBM MQ on the server that was running the standby instance.
5. Decide whether downtime or recoverability is more important in the migration:

   Follow these steps if recoverability is more important, and you must take a backup:

   a. Stop the active queue manager instance, without switching to any standby.
   b. Back up the queue manager.
   c. Start a queue manager instance, permitting standbys, on one of the upgraded servers.
   d. If you have a pool of upgraded servers, start another one, permitting standbys.

   If availability is more important, follow this procedure; you do not take a backup.

   a. Start a queue manager instance as a standby on one of the upgraded servers.
   b. Stop the active queue manager instance, switching to the standby.
   c. If you have a pool of upgraded servers, start another one, permitting standbys.
6. Upgrade the IBM MQ code on the server that was the active queue manager instance, and start it as the standby instance if you have not already started a standby.

Overall steps to migrate a queue manager in a high-availability cluster

The following terms are relevant:

active server
The running server or active queue manager instance

passive server
A server that is ready to take over from the active server automatically.

inactive server
A server that is not prepared to take over automatically. The server might have been removed from the cluster, or taken offline in some way.

Base your migration procedure on the following steps. The details depend on the specific commands in the cluster concerned.

1. Before you start the migration process, create a different queue manager on a server on which you have installed the upgrade. Test the upgrade by performing whatever verification checks that your organization requires.
2. If you have four servers available, you can form two cluster pairs.

   With two pairs, the queue manager can continue to run in a cluster-pair at the old command level. When you are ready, you can transfer the queue manager to the pair of servers at the new command level.
3. Remove a passive server from the cluster. Make sure that the cluster cannot automatically restart the server. The server is made inactive.
4. If a high-availability cluster is using a common location for IBM MQ code, you must create a second location for the upgraded code.
5. Install, or upgrade, IBM MQ code using the server that is not now running the queue manager.
6. Verify the upgrade by creating a different queue manager on the server, and performing whatever verification checks that your organization requires.
7. If more than half the servers remain in the cluster, remove a server, upgrade IBM MQ, and verify the upgrade. Each server is made inactive as part of the process. Continue until half the servers are upgraded.
8. If your active server is part of a remaining cluster, deactivate the passive servers so that the cluster cannot reactivate them automatically.
9. Decide whether downtime or recoverability is more important in the migration:

   Follow these steps if recoverability is more important:

   a. Stop the queue manager and remove the server from the cluster.
   b. Back up the queue manager.

   Or this step, if downtime is more important:

   a. Add the migrated servers back into the cluster, as passive servers. Then switch the remaining server in the high-availability server cluster over to one of the passive servers. The switch causes the running queue manager to stop, restarts it on one of the passive servers.
10. Upgrade any remaining high-availability servers, and add them back into the cluster.

Related tasks

“Windows: Migrating an MSCS configuration” on page 165
Migrate queue managers in MSCS configuration one node at a time, following these instructions.

Introduction to changes for Windows on IBM MQ 8.0

A number of changes have been made for IBM MQ 8.0 for Windows. You must understand these changes before planning any migration tasks using IBM MQ 8.0 for Windows.

Installing a single copy of the product

If you have an existing previous version of the product on your system, and want to upgrade to the latest version, you have various options. You can either:

- Uninstall the previous version and then install the latest version,

- Install the new copy alongside the currently installed one and uninstall the original at a later time. See “Installing the product alongside an existing version” on page 79, or
- Perform a migration install, electing to replace the currently installed version when prompted.

After you have installed the product, start each queue manager and its data migration takes place. This includes the migration of queue managers from 32 bit to 64 bit.

Installing the product alongside an existing version

If you want to install another version of the product alongside your existing product you can do so. See “Multiple IBM MQ installations” on page 33 and “UNIX, Linux, and Windows: Side-by-side migration to a later version” on page 96 for further information.

When you install the new version of the product, run the setmqm command to associate the queue managers with the new installation.

Start each queue manager in turn and its data migration takes place.

Upgrading one of a pair (or more) of installations

If you already have, for example, a IBM WebSphere MQ 7.5 installation and an IBM MQ 8.0 installation on a machine, upgrading the Version 7.5 installation to Version 8.0 requires the following additional step.

When you start the Version 8.0 installer, you are asked whether you want to **Install a new instance** or **Maintain or upgrade an existing instance**.

However, only the other Version 8.0 installation, or installations, are displayed; not the Version 7.5 installation in the selection box. At this point, select **Install a new instance**.

After the splash screen has been displayed, a second panel appears, which lists any older installations that you can upgrade to Version 8.0 using the Version 8.0 installer.

On this panel, select **Upgrade 7.5.0.n Installation 'Installation m'**, and then click **Next**.

Change of digital signature algorithm

The IBM MQ programs and installation image are digitally signed on Windows to confirm that they are genuine and unmodified.

In older releases the product was signed using the SHA-1 with RSA algorithm.

In IBM MQ 8.0, the SHA-256 with RSA algorithm is now used. Some old versions of Windows do not support the new digital signature algorithm, but those versions are not supported by IBM MQ 8.0.

See Hardware and software requirements on Windows(tm) systems, and ensure that you install IBM MQ 8.0 on a supported version of Windows.

Existing applications

All applications that were built with previous versions of the product continue to work with a 64 bit queue manager.

All applications using the C++ object interface need to be rebuilt; applications using the C interface are not affected.

Exits

Queue manager exits on Windows 64-bit operating systems must be compiled as 64-bit exits. Any 32-bit queue manager exits must be recompiled before they can be used with a 64-bit queue manager. If you try to use a 32-bit exit with a 64-bit queue manager on IBM MQ 8.0, an AMQ9535 "invalid exit" error message is issued.

Clients

32-bit client applications can connect transparently to queue managers from all supported versions of IBM MQ. This includes 64-bit IBM MQ 8.0.

Samples

The samples for the C and C++ languages are compiled as 64-bit.

Related information

Windows: changes for IBM MQ 8.0

Directory structure on Windows systems

Hardware and software requirements on Windows systems

IBM MQ migration planning to the latest version on UNIX platforms, Windows, and IBM i

Before migrating from one version to another, read this topic. Create your own migration plan based on the outline in the planning topic.

If there are concepts about migration you do not understand, read "Introduction to IBM MQ migration" on page 19 first.

About this task

Use the following steps as a guide to creating a migration plan.

Procedure

1. Review the IBM MQ system requirements for Version 8.0.

   See System Requirements for IBM MQ.
2. Decide whether to run the current version of your product and the latest version on the same server.

   Note that if you require a multi version installation, and you are using a version of the product prior to IBM WebSphere MQ 7.0.1.6, you must install IBM WebSphere MQ 7.0.1.6 first.
3. Review all the changes in IBM MQ that affect you.

   See What's changed in IBM MQ 8.0.
4. Review performance changes.

   Performance reports are published as Supportpacs; see IBM MQ - SupportPacs by Product.
5. Review the latest README file for the product you are working with.

   See IBM MQ and MQSeries® product READMEs.
6. Plan the sequence and timing of queue manager upgrades.

   If the queue manager is part of a queue manager cluster, you must migrate the queue managers that are full repositories first.

   If the queue manager is part of a high availability cluster, plan the migration to minimize downtime and maximize availability; see "Migrating a queue manager in a high-availability configuration" on page 76.
7. Plan to migrate your queue manager to the latest version.

   UNIX platforms - Migrating a queue manager from a previous release to the latest release.

   Windows - Migrating a queue manager from a previous release to the latest release

   IBM i IBM i - Migrating a queue manager from a previous release to the latest release or Migrating a queue manager from a previous release to the latest release, alternative method

   Backing up queue manager data is part of the queue manager migration task.

An alternative approach to backing up queue manager data, is to install and configure a new server. Test the latest version with a new queue manager on the new server. When you are ready to go into production on the latest version, copy the queue manager configuration and data, to the new server.

8. Plan to update any manual or automated procedures you have written with changes to messages and codes.
9. Decide on what regression tests to perform before putting the queue manager into production on the latest version.

   Include the procedures and applications you identified in steps 6 and 7 in your regression tests.
10. Plan to upgrade your IBM MQ MQI client installations to the latest version.
11. Plan to upgrade your client and server applications to use new functions in the latest version.

z/OS: Migration planning to the latest release

Create a migration plan for IBM MQ for z/OS to upgrade to the latest release.

About this task

The goal of this task is for you to create your own plan to migrate your queue managers to the latest release. Incorporate the task to migrate a queue manager, "Migrating IBM MQ for z/OS - order of tasks" on page 139, into your plan.

Migration plan overview for your enterprise	
Migration phase	**Required tasks**
Phase I, before migration. See "Preparing to migrate a single IBM MQ for z/OS queue manager" on page 146 for further information.	Preparing each queue manager in your enterprise for migration.
Phase II, migrate each single queue manager in the order listed. See "Migrating a single IBM MQ z/OS queue manager to the next release of the product" on page 151 for further information.	Carry out this process for each queue manager in the order shown: 1. Queue managers in a full repository on platforms other than z/OS. See "Migrating a queue manager to the latest release" on page 107 for further information. 2. z/OS partial repository queue managers in a QSG. 3. z/OS partial repository queue managers not in a QSG . 4. z/OS queue managers not in a QSG or cluster. 5. Partial repository queue managers on platforms other than z/OS. 6. Queue managers not in a cluster on platforms other than z/OS.
Phase III, post migration. See "Post migration tasks" on page 157 for further information.	Carry out a full regression test, and then explore the new function available to you. Optionally, at any time in the process, upgrade your client libraries if necessary, recompile your clients using the additional features provided by the new version, and deploy the clients.

Procedure

1. Review the IBM MQ system requirements for the latest version.

    See System requirements for IBM MQ.
2. Review all the changes in the product that affect you.

    See What's changed in IBM MQ 8.0 for further information.
3. Review performance changes.

    Performance reports are published as Supportpacs; see IBM MQ - SupportPacs by Product.
4. Review the backward and coexistence (or migration and toleration) PTFs for your current version of the product. See IBM MQ Support, Migration PTFs.

    These PTFs must be applied to your current version of the product to enable you to revert your queue managers to the current version, after the queue managers have been started at the target version.

    Note, that you can have different versions of queue managers coexisting in the same queue-sharing group.

    If you are unsure which migration PTFs you require, run the following command SMP/E:

    ```
    REPORT MISSINGFIX ZONES(mqtgtzone) FIXCAT(IBM.Coexistence.MQ.V8R0M0)
    ```

    See FIXCAT and IBM MQ Migration Installation for further information.

    Attention: If a PTF requires a rebind of Db2® plans, the PTF is shipped with ++HOLD(ACTION), indicating the need for this process. In such a case, see Migrating Db2 tables to bind the plans before starting migration.

    Other FIXCAT categories are listed in IBM Fix Category Values and Descriptions.
5. Plan to install the latest version early code, and activate for all queue managers on the LPAR.

    See Installing early code for more information.

    Note that:

    Before migration, all systems that are running queue managers that you plan to migrate to the latest version must have the early code for that version installed and running. Queue managers in queue-sharing groups that contain queue managers that are to be migrated, must also be running the early code.
    A queue manager must use the early code from the same release level, or a later release level.
6. Consider using aliases for the IBM MQ libraries.

    For example, use the IDCAMS utility with the DEFINE command:

    ```
    DEFINE ALIAS(NAME(MQM.SCSQANLE)RELATE(MQM.V710.SCSQANLE))
    ```

    You can use `MQM.SCSQANLE`, where applicable, in your STEPLIB, and it resolves to the actual data set.

    When you migrate to a new release, change the alias definition, rather than changing all the places in your JCL where the library is referenced.

    This process has the most benefit for your server application programs, because you can get all of the programs to refer to the new libraries at the same time.
7. Plan the sequence and timing of queue manager upgrades.

    - You must install the backward and coexistence (or migration and toleration) PTF to bring the previous version queue managers up to the latest maintenance level for that version.
    - You must install the PTF on all members of a queue-sharing group, before you migrate any queue managers to the latest version. You can install the PTF one member at a time, leaving the other members running.

- If the queue manager is a member of a queue manager cluster, you must consider the order of migration of queue managers in the cluster; see "Queue manager cluster migration" on page 74.
- Check that any products that require the previous version of the product also support the new version.

8. Plan to update any manual or automated procedures you have written with changes to messages and codes.
9. Plan to update applications that might be affected by changes.

   Update the IBM MQ library in the application STEPLIB concatenations to the latest version.

   Consider whether the application must be able to run on both the previous version and the latest version. You might be able to change the application to be compatible with both code levels. If you cannot, you can query the queue manager command level, and make the code conditional on the command level. Call `MQINQ` setting the `MQIA_COMMAND_LEVEL` selector.
10. Decide on what regression tests to perform before enabling the new function in the latest version.

    The **OPMODE** parameter controls a staged migration from the previous version to the latest version.

    Initially set **OPMODE** to `(COMPAT,800)` to make sure that you can go back to using an earlier version of the product. Once you are satisfied with the stability of the latest version, you can start to use the new functions available, which requires setting **OPMODE** to `(NEWFUNC,800)`.

    Include the procedures and applications you identified in steps "8" on page 83 and "9" on page 83 in your regression tests.
11. Review the tasks to customize z/OS, and the queue manager. Plan how to change the queue manager definitions and started task JCL to migrate your queue managers to the latest versions.

    The customization steps for migration are described in "z/OS: Review and modify queue manager customizations from the previous release" on page 84
12. Review the usage of page set 0.

    Issue the operator command **cpf**, `/cpf DISPLAY USAGE PSID(0)` to obtain a report on pageset 0 usage.

    The size of queue definitions increased in IBM WebSphere MQ 7.1. During migration queue definitions stored on pageset 0 are rewritten if you are migrating from a previous release.

    The rewrite is carried out as a single transaction when the queue manager is first migrated to IBM WebSphere MQ 7.1.

    Ensure that there is enough space available on pageset 0 to create a copy of the queue definitions while migration is taking place. Typically, 60% free space on pageset 0 before migration is sufficient. However, the use of `EXPAND(SYSTEM)` on the pageset definition allows for automatic expansion as required.

    If there is insufficient space on pageset 0 during migration, the queue manager abends with completion code `X'5C6'` and reason code `X'00C91900'`
13. Check that you are using a supported level of assembler or compiler.

    You can write IBM MQ applications using any compiler capable of generating standard OS linkage to the IBM MQ stub routines.

    Some of the data types used by IBM MQ API calls are not supported on some older compilers. You might require a more recent compiler. The following limitations are known:

    a. Assembler copy books contain blank lines, which are not tolerated by assemblers earlier than **HLASM**.
    b. Some older releases of PL/I do not support fixed `bin(63)` type. A macro defines such fields as `char(8)` when an earlier compiler is detected.
    c. Some older releases of COBOL do not support function-pointers, which are used by the `MQCB` API.
14. Plan any changes to libraries required by your applications and channel exits.
15. Plan to upgrade your IBM MQ MQI client installations to the latest version.
16. Plan to upgrade your client and server applications to use new functions in the latest version.

17. Plan to migrate other vendor software, such as WebSphere Application Server, or CICS® to use the latest version.

    Update the IBM MQ libraries in the STEPLIB and DFHRPL concatenations of your CICS region JCL and restart CICS.

    Note:

    - CICS

      Update the IBM MQ libraries in the STEPLIB and DFHRPL concatenations of your CICS region JCL and restart CICS.

      Up to, and including CICS 3.2, the connection between IBM MQ and CICS is provided by IBM MQ. You must change the SCSQCICS and SCSQAUTH libraries in the DFHRPL concatenation provided by IBM MQ.

      After CICS 3.2, the connection between IBM MQ and CICS is provided by CICS libraries. Update the libraries, if you are using CICS Transaction Server for z/OS Version 3.2 or later. Without this change, you are not able to use the most recent IBM MQ features. You must change the SCSQCICS library in the DFHRPL concatenation provided by IBM MQ, and also the STEPLIB concatenation.

      Create separate CICS started procedure JCL. For each CICS region that is connected to an IBM MQ queue manager, ensure that there is a separate CICS started procedure JCL.

      This ensures that the modification of reference to a certain version of IBM MQ libraries in the CICS started procedure JCL only has impact for that single CICS region. In this way you can migrate one queue manager, and only the CICS region or regions connected to it, which makes staged migration possible.

      CICS STEPLIB has `thlqual.SCSQAUTH`, and DFHRPL has `thlqual.SCSQCICS`, `thlqual.SCSQLOAD`, and `thlqual.SCSQAUTH`. For more information, see Setting up the CICS-IBM MQ adapter

18. Review any other installed SupportPacs for their applicability to the latest version.

What to do next

Do the task, "z/OS: Review and modify queue manager customizations from the previous release" on page 84 to migrate a queue manager. If you must restore a queue manager to the previous version, see "z/OS: Reverting a queue manager to a previous release" on page 155.

When you are confident existing applications are running without migration problems on the latest version, plan to update **OPMODE** to (NEWFUNC,800) to enable new function.

About the CICS-MQ adapter

IBM MQ Support, Migration PTFs

IBM MQ - SupportPacs by Product

z/OS: Review and modify queue manager customizations from the previous release

Review the z/OS and IBM MQ customization steps, and change any customizations before starting any queue managers with the libraries of the latest version.

Before you begin

You must change the JCL and configuration definitions for the queue manager when you migrate to the latest version. Make copies of the JCL libraries, configuration definitions, and the started task JCL. Modify the copied files to customize the queue manager definitions and JCL for the latest version.

1. Copy the JCL libraries containing the configuration definitions for your queue manager. You must change some of the members, for example to reference the libraries of the new version.

2. Copy the started task JCL for the queue manager and the channel initiator. You might change these members.

You can continue to run the queue manager on the previous version until it is ready to switch to the latest version. Preparing for the switch is a long process. Switching from the earlier version to the latest version is a quick process. The switch to the latest version occurs when you restart the queue manager:

- Switch to using your customized copies of the configuration definitions and started task JCL for the queue manager.

  Create dataset aliases, such as `MQM.MQP1.SCSLOAD`, and reference them in JCL. Map the aliases to the real data sets, such as `MQM.MQV71.SCSLOAD` or `MQM.MQV80.SCSLOAD`.

  Change the aliases to switch between the two sets of target libraries. With the aliases, you can start applications or the queue manager when moving to a new release of IBM MQ without changing the `STEPLIB` JCL; switch the aliases from referring to the earlier version of the product to refer to the latest version of the product.
- Until you have verified the start up, start the queue manager, channel initiator, and listener separately, checking for errors on the console after each component is started. If the start up runs cleanly, combine the start up of all three components in the production environment.

As part of this task you must set **OPMODE** to `(COMPAT,800)`. If you have not set **OPMODE**, the queue manager does not start.

Note that if you use the supplied CSQ4ZPRM sample, the queue manager starts without any problems because the provided default value for **OPMODE** is `COMPAT,800)`.

The `(COMPAT,800)` setting does two things:

1. It checks that you are starting the queue manager that links to the libraries for the latest version of the product.
2. It disables the use of new function in the latest version of the product.

You can revert to running the queue manager with the earlier version libraries, if you have never set **OPMODE** to `(NEWFUNC,800)`

Tip: Create data set aliases such as `MQM.MQP1.SCSLOAD`, and reference them in JCL. Map the aliases to the real data sets, such as `MQM.MQV71.SCSLOAD` or `MQM.MQV80.SCSLOAD`. Change the aliases to switch between the two sets of target libraries. With the aliases, you can start applications or the queue manager when moving to a new release of IBM MQ without changing the `STEPLIB` JCL.

Tip: You can use the z/OS command `D GrS,system,RES=(*,MQM.V701.SCSQLOAD)` to display which jobs are using the specified data set, and so identify which jobs and JCL need to be changed.

About this task

The steps in this task can be performed by restarting a queue manager once. You must restart the queue manager to apply essential maintenance, and to install early code before proceeding with migration.

The steps are based on the setup procedure for new queue managers; see Customizing your queue managers.

Procedure

Steps “1” on page 85 to “10” on page 87 are z/OS customization tasks.

Steps “11” on page 87 to “16” on page 89 are IBM MQ customization tasks.

1. Review that all the load libraries that must be APF authorized are authorized.

   See Task 2: APF authorize the IBM MQ load libraries.
2. Update the LPA libraries with the new version of early code libraries.

   See Task 3: Update the z/OS link list and LPA.

This step is unnecessary if you update the IBM MQ early code in all the LPARs as part of bringing your previous version IBM MQ libraries up to the latest maintenance level; see step "3" on page 86.

You must activate the early code with an IPL, or restart the queue manager after issuing the following command.

```
REFRESH QMGR TYPE(EARLY)
```

Note: As part of the update you are asked to stop the queue manager. While the queue manager is stopped, you must check the *qmgr*.REFRESH.QMGR security profile is set up, refresh the queue manager and restart it.

3. Make your configuration of the previous version ready for migration.

   a) Apply current maintenance to the current version libraries.

      Refer to the Preventive Service Planning (PSP) bucket for the current version of your product; see PSP Buckets - How to find them on Web.

   b) Apply the latest version migration and toleration [6] PTFs to the previous version code; see IBM MQ Support, Migration PTFs.

   c) If required by any of the PTFs, make Db2 configuration changes and rebind Db2.

      See the appropriate section in Task 9: Select and set up your coupling facility offload storage environment.

   d) Install the early code for the latest version.

      Replace the early code.

   e) Make the latest version early code and earlier version target libraries available on all the LPARs that are running queue managers.

      If the queue manager is a member of a queue-sharing group, update all the systems in the group.

   f) Restart IBM MQ systems.

      The early code is activated by an IPL, or a by issuing the command `REFRESH QMGR TYPE(EARLY)`, and restarting the queue manager.

   g) Verify correct function before proceeding and review any special actions detailed in the PTFs.

   If you require fall back at this stage, use normal maintenance procedures to revert to the code for the previous version before PTF application.

4. Update your procedure to start the queue manager.

   Change the `STEPLIB` for the queue manager to reference the libraries of the new version of the product.

   See Task 6: Create procedures for the IBM MQ queue manager.

   IBM MQ uses z/OS memory objects above the bar for some functions. You must allow the queue manager to access storage above the bar.

   Your installation might have customized the `SMFPRMxx` member of `SYS1.PARMLIB`, or the **`IEFUSI`** exit to provide a default limit for jobs using virtual storage above the 2 GB bar. Check these limits give sufficient memory for a queue manager. A reasonable starting allocation is 2GB. The message CSQY220I displays the amount of virtual storage currently used and available.

   If your installation does not have a default limit for storage above the bar, or if you want to use a different limit for your queue manager, you can provide a queue manager-specific restriction on

[6] The "migration and toleration" PTFs are also known as the "backward migration and coexistence" PTFs. They are the same PTFs.

the amount of virtual storage available above the bar for memory objects by coding a **MEMLIMIT** parameter on the JCL of the queue manager stored procedure, xxxxMSTR, for example:

```
//PROCSTEP EXEC PGM=CSQYASCP,REGION=0M,MEMLIMIT=2G
```

MEMLIMIT defines memory available above the bar; see Address space storage for more information.
For specific information on setting MEMLIMIT, when using the accounting and statistics changes in IBM MQ 8.0, see Channel initiator storage usage.

5. Update your procedures for the channel initiator.

   Change the STEPLIB for the channel initiator to reference the libraries of the new version of the product.
   See Task 7: Create procedures for the channel initiator.
6. Review any automated alerts when queue manager and channel initiator messages, and errors, are detected.

   New messages might have been added that cause automated alerts, and some messages might have changed.
7. Review C language channel exits

   Ensure your C language channel exits are using the following statement:

   ```
   #pragma environment(function-name)
   ```

   as defined in the C systems programming environment for system exits, described in the z/OS C/C++ Programming Guide.
8. Update the IBM MQ Db2 configuration.

   If you are using queue-sharing groups, you must update the procedures.

   Customize and run the CSQ4570T and CSQ4571T sample JCL in *hlq*.SCSQPROC.

   Customize and run the CSQ45BPL and CSQ45GEX samples in hlq.SCSQPROC. Tailor these members to your environment, using your Db2 subsystem names and data set names.
   CSQ45BPL of hlq.SCSQPROC contains the plan names required for the latest version of IBM MQ.
   CSQ45GEX of hlq.SCSQPROC contains the authorities required.
   See steps 5 and 6 of Task 9: Select and set up your coupling facility offload storage environment.
9. Review your security controls for queue-sharing groups, the channel initiator, and all queue managers accessing the coupling facility list structures.

   See Task 11: Implement your ESM security controls.

   You must ensure that you have addressed the following points when you migrate your security profiles to the latest version of IBM MQ.

   - The External Security Manager software is at the correct version and level and that all of your prerequisite software is installed.
   - IBM MQ security classes have been updated to include the mixed case classes.
   - Enterprise has migrated to mixed case security; see “z/OS Migrating a queue manager to mixed case security” on page 153.
10. Change SYS1.PARMLIB to ensure that any changes you made dynamically remain in effect after an IPL.

    SYS1.PARMLIB must reference the initialization input data sets that you customize in step “11” on page 87.
    See Task 12: Update SYS1.PARMLIB members.
11. Update the initialization input data sets.

Each IBM MQ queue manager gets its initial definitions from a series of commands contained in the IBM MQ initialization input data sets. These data sets are referenced by the Data Definition (DD) names CSQINP1 and CSQINP2 defined in the queue manager started task procedure.

See Task 13: Customize the initialization input data sets.

The CSQINP1 and CSQINP2 initialization input data sets include more samples and the contents of some samples have been moved to other samples. Particular changes to take note of are the commands to define queues to hold publish/subscribe state information. The commands must be in the right order.

You must review the customization you have made previously to CSQINP1 and CSQINP2, and merge them into the initial definitions provided with the latest version of the product.

Secure the server-connection channels used by clients; see Securing remote connectivity to the queue manager.

DEFINE SUB for SYSTEM.DEFAULT.SUB is no longer permitted in the CSQINP2 input data set. DEFINE SUB commands can instead be issued from the CSQINPT input data set. The CSQINPT input data set is processed each time the publish/subscribe engine is started, either during queue manager startup, or when the publish/subscribe engine is started with the **ALTER QMGR PSMODE(ENABLED)** command. See Issuing commands to IBM MQ for z/OS for more information on using the CSQINPT input data set.

12. Update your system parameter module.

    Your system parameter module is based on the default module, CSQZPARM and CSQ4ZPRM.

    See Task 17: Tailor your system parameter module.

    Set the value of the **OPMODE** parameter in the **CSQ6SYSP** macro to (COMPAT, 800). The parameters (COMPAT, 800) force a check that the queue manager is started at the 800 command level, and is started in compatibility mode.

13. Update the libraries you added to STEPLIB concatenations to make Batch, TSO, and RRS adapters available to applications.

    Change the STEPLIB for the Batch, TSO, and RRS adapters to reference the libraries of the new version of the product.

    See Task 19: Set up Batch, TSO, and RRS adapters.

    Note: You can connect applications that reference the latest version STEPLIB to a queue manager that is running on the latest version, or an earlier version. You must not connect applications that reference a STEPLIB from an earlier version to a queue manager running on a later version.

14. Update the libraries you added to connect CICS to the queue manager.

    You must update the IBM MQ libraries in the STEPLIB and DFHRPL concatenations of your CICS region JCL and restart CICS. You are then able to use the most recent IBM MQ features.

    Note:

    - CICS

      Update the IBM MQ libraries in the STEPLIB and DFHRPL concatenations of your CICS region JCL and restart CICS.

      Up to, and including CICS 3.2, the connection between IBM MQ and CICS is provided by IBM MQ. You must change the SCSQCICS and SCSQAUTH libraries in the DFHRPL concatenation provided by IBM MQ.

      After CICS 3.2, the connection between IBM MQ and CICS is provided by CICS libraries. Update the libraries, if you are using CICS Transaction Server for z/OS Version 3.2 or later. Without this change, you are not able to use the most recent IBM MQ features. You must change the SCSQCICS library in the DFHRPL concatenation provided by IBM MQ, and also the STEPLIB concatenation.

      Create separate CICS started procedure JCL. For each CICS region that is connected to an IBM MQ queue manager, ensure that there is a separate CICS started procedure JCL.

      This ensures that the modification of reference to a certain version of IBM MQ libraries in the CICS started procedure JCL only has impact for that single CICS region. In this way you can migrate one queue manager, and only the CICS region or regions connected to it, which makes staged migration possible.

CICS STEPLIB has `thlqual.SCSQAUTH`, and DFHRPL has `thlqual.SCSQCICS`, `thlqual.SCSQLOAD`, and `thlqual.SCSQAUTH`. For more information, see Setting up the CICS-IBM MQ adapter

15. Update the libraries you use to set up the operations and control panels.

    Change the STEPLIB for the operations and control panel.
    See Task 20: Set up the operations and control panels.

    Note: You can connect the operations and control panel that references the latest version STEPLIB to the queue manager that is running on the latest version, or an earlier version. You must not connect the operations and control panel that references a STEPLIB from an earlier version to a queue manager running on a later version.

16. Update system libraries to format IBM MQ dumps using the Interactive Problem Control System (IPCS).

    See Task 21: Include the WebSphere(r) MQ dump formatting member.

What to do next

Return to the parent migration planning task, "z/OS: Migration planning to the latest release" on page 81.

Related tasks

"Migrating a single IBM MQ z/OS queue manager to the next release of the product" on page 151
Carry out the instructions in this topic to migrate a single IBM MQ queue manager on z/OS,

z/OS: IBM MQ Version 8.0 JCL changes

Tables showing new members and changed members between the IBM WebSphere MQ 7.1 hlq.SCSQPROC and the IBM MQ 8.0 hlq.SCSQPROC PDS libraries at general availability time.

Notes:

1. Nearly all of the members do show changes between releases, but the majority of those changes are what are termed cosmetic. They are often simply changes to reflect the modification to the release number.
2. Any changes to the samples libraries since general availability are not included.
3. This information can be useful to your administrators when migrating existing queue managers, to make sure that new features are picked up correctly.

New members

Table 8. IBM MQ for z/OS additional members

Member name	Description
CSQ4AMSM	Procedure to start the IBM MQ Advanced Message Security (IBM MQ AMS) task.
CSQ4BCNV	Sample JCL to convert the BSDS to version 2 for the log RBA expansion.
CSQ4INSM	CSQINP2 sample for SYSTEM objects for IBM MQ AMS.
CSQ4QLOD	Sample JCL for the QLOAD utility.

Table 8. IBM MQ for z/OS additional members (continued)

Member name	Description
CSQ4SCHD	Sample SCHEDxx member for the following resources: • queue manager • channel initiator • IBM MQ AMS
CSQ40CFG	Sample JCL to define a new security policy for IBM MQ AMS. This member runs the setmqspl command
CSQ40ENV	Sample IBM MQ AMS environment variables.
CSQ40RSM	Sample IBM MQ AMS audit report.
CSQ45BPK	Sample JCL to bind Db2 packages for shared queues. This job was added to IBM WebSphere MQ 7.1 using a program temporary fix.
CSQ4570T	Sample job to migrate Db2 resource from IBM WebSphere MQ 7.0.0 and IBM WebSphere MQ 7.0.1 to IBM MQ 8.0.
CSQ4571T	Sample job to migrate Db2 resource from IBM WebSphere MQ 7.1 to IBM MQ 8.0.

Modified members

Table 9. IBM MQ for z/OS changed members

Member name	Description
CSQ4CHIN	STEPLIB concatenation changed to add the optional library DSN=++CSFQUAL++.SCSFMOD0,DISP=SHR used when password protection is on. For specific information on setting `MEMLIMIT`, when using the accounting and statistics changes in IBM MQ 8.0, see Channel initiator storage usage.
CSQ4INPR and CSQ4INP1	Changed to support the new bufferpool definition attributes: • LOCATION: Default is BELOW • PAGECLAS: Default is 4 KB • REPLACE: Default is NOREPLACE
CSQ4INSG	Changed to add the: • STATCHL attribute to the sample channel definitions • DEFINE AUTHINFO('SYSTEM.DEFAULT.AUTHINFO.IDPWOS') to support the new authorization feature • CLROUTE attribute to the sample default topic definitions • Definition for the SYSTEM.DDELAY.LOCAL.QUEUE for those using JMS 2.0
CSQ4INSX	Added the model queue definition for the SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE to support split cluster transmission queues.
CSQ4INYC	Added the STATCHL attribute to the sample cluster receiver.

Table 9. IBM MQ for z/OS changed members (continued)

Member name	Description

CSQ4INYD	Trigger attributes on the sample transmission queue definition have been changed to:
	`* Trigger attributes`
	`TRIGGER +`
	`TRIGTYPE( FIRST ) +`
	`TRIGMPRI( 0 ) +`
	`TRIGDPTH( 1 ) +`
	`TRIGDATA( '++locqmgr++.TO.++remqmgr++' ) +`
	`PROCESS( ' ' ) +`
	`INITQ( 'SYSTEM.CHANNEL.INITQ' )`
	from
	`* Trigger attributes`
	`TRIGGER +`
	`TRIGTYPE( FIRST ) +`
	`TRIGMPRI( 0 ) +`
	`TRIGDPTH( 1 ) +`
	`TRIGDATA( ' ' ) +`
	`PROCESS( '++remqmgr++.SEND.PROCESS' ) +`
	`INITQ( 'SYSTEM.CHANNEL.INITQ' )`

CSQ4INYG	The sample ALTER QMGR statement has been modified to have all the attributes on an individual line.

CSQ4INYR	A new storage class definition has been added:
	`DEFINE STGCLASS( 'SYSLNGLV' ) +`
	`QSGDISP( QMGR ) +`
	`PSID( 02 )`

CSQ4MSRR and CSQ4MSTR	Several optional changes have been made: • The MEMLIMIT parameter has been added to the procstep • The CSQINP2 concatenation now includes the new member CSQ4INSM (for IBM MQ AMS) • A new DD statement for CSQINPT has been added. This statement concatenates the members CSQ4INST and CSQ4INYT for publish/subscribe • A new output DD statement has been added, CSQOUTT
CSQ4PAGR	The FORM EXEC statement has been altered to include a REGION=4M parameter. The CSQ4PAGE FORM EXEC statement already contained the REGION parameter.

CSQ4SMJF	New DD statements have been added in support of the new SMF data:
	`//QCCT   DD SYSOUT=*,DCB=(LRECL=233,RECFM=F,BLKSIZE=233)`
	`//QCTDSP  DD SYSOUT=*,DCB=(LRECL=233,RECFM=F,BLKSIZE=233)`
	`//QCTADP  DD SYSOUT=*,DCB=(LRECL=233,RECFM=F,BLKSIZE=233)`
	`//QCTSSL  DD SYSOUT=*,DCB=(LRECL=233,RECFM=F,BLKSIZE=233)`
	`//QCTDNS  DD SYSOUT=*,DCB=(LRECL=233,RECFM=F,BLKSIZE=233)`
	`//QCST   DD SYSOUT=*,DCB=(LRECL=233,RECFM=F,BLKSIZE=233)`

Table 9. IBM MQ for z/OS changed members (continued)

Member name	Description
CSQ4ZPRM	Several changes have been made: • Archive log blocksize changed to 24576 from 28762 • CONNSWAP=YES parameter added. This allows batch jobs to swap during an IBM MQ API call • EXCLMSG=( ) parameter added. This allows up to 16 messages to be suppressed. • OPMODE=(COMPAT,800) change from the IBM WebSphere MQ 7.1 values. • SPLCAP=NO, message encryption flag used by IBM MQ AMS.
CSQ45BPL	Sample job to bind Db2 plans using the Db2 TSO batch interface. **Note:** The plan names have changed for IBM MQ 8.0.

CSQ45CTB	Columns added in the CSQ.ADMIN_B_STRBACKUP table:
	`BSTART_RBA_HI CHAR(2) WITH DEFAULT X'0000',`
	`BEND_RBA_HI  CHAR(2) WITH DEFAULT X'0000',`

CSQ456TB	Changes to migrate the Db2 tables to IBM MQ 8.0: • Added the two new columns (see the preceding entry for CSQ45CTB) • Table names are altered for IBM MQ 8.0.
CSQ45GEX	Sample Db2 GRANT job to use the IBM MQ 8.0 names.

UNIX, Linux, and Windows: Single-stage migration to a later version

Single-stage migration, is the term used to describe replacing the only installation of IBM MQ on a server, with a later release. Single stage migration is also known as `upgrading in place` or `in place upgrade` . Until Version 7.0.1.6, single-stage was the only migration scenario. Single-stage migration preserves existing scripts and procedures for running IBM MQ the most. With other migration scenarios you might change some scripts and procedures, but you can reduce the effect queue manager migration has on users.

Before you begin

This scenario is one of three, which describe alternative ways to upgrade queue managers from an earlier version of the product. The other scenarios are as follows:

1. Install the latest version of the product alongside an earlier version ; see "UNIX, Linux, and Windows: Side-by-side migration to a later version" on page 96.
2. Run the latest version of the product alongside an earlier version ; see "UNIX, Linux, and Windows: Multi-stage migration to a later version" on page 100.

Read these three tasks to plan how you are going to migrate to the multi-installation environment of the latest version. Even if you do not plan to have more than one version of the installation on a server, read this topic. The steps to upgrade IBM MQ from Version 7.0.1 to Version 8.0 have changed.

These topics are for planning multi-installation migration. The planning topics guide you in deciding what other tasks you must perform to migrate queue managers and applications to the latest version. For the precise sequence of commands to upgrade a queue manager to the latest version, do the migration task for the platform you are interested in. All the tasks are listed by platform in the links at the end of this topic. As part of the queue manager migration task, back up your existing queue manager data. Even on a multi-installation server, queue managers cannot be restored to a previous command level after migration.

About this task

In the single-stage migration scenario, the installation of the latest version of the product replaces an earlier version in the same installation location. It is the same migration process that you would have used to upgrade the product prior to IBM WebSphere MQ 7.0.1, Fix Pack 6. It is now termed "single-stage" migration, in contrast to "side-by-side" and "multi-stage" migration.

The advantage of single-stage migration is that it changes the configuration of a queue manager on the earlier version as little as possible. Existing applications switch from loading the libraries from the earlier version, to loading the libraries of the latest version, automatically.

Queue managers are automatically associated with the installation on the latest version. Administrative scripts and procedures are affected as little as possible by setting the installation to be the primary installation. If you set the installation of the latest version to be the primary installation, commands such as **strmqm** work without providing an explicit path to the command.

Four types of object are important to consider during migration:

- Installations
- Queue managers
- Administrative procedures
- Applications

Administrative procedures contain IBM MQ commands, and scripts that use commands.

To run a command, the operating system must find the command in an IBM MQ installation. For some commands, you must run the command from the installation that is associated with the correct queue manager. IBM MQ does not switch commands to the correct installation. For other commands, such as **setmqinst**, you can run the command from any installation that has the latest version of the product installed.

If an earlier version of the product is installed, the command that is run is the command for that version, unless the search path is overridden by a local setting. You can override the search path by running **setmqenv**. If Version 7.0.1 is not installed, you must set the correct path to run a command. If you have set a primary installation, the command that is run is the copy in the primary installation, unless you override the selection with a local search path.

Procedure

1. Stop local IBM MQ applications.
2. Stop all the queue managers and listeners.

   This scenario uses two queue managers QM1 and QM2.
3. Uninstall any fix packs you have installed from the previous IBM MQ version.
4. Upgrade the earlier version of the product to the latest version in the same installation directory.
   a) Decide on an installation naming convention. Give the installation a name of your choosing, or accept the default installation name.

      For the first installation, the default name is *Installation1*. For the second installation, the name is *Installation2*, and so on.

      AIX On AIX there is no option to set the installation name, *Installation1* is set by default.

   b) Upgrade the earlier version of the product to the latest version in place, or uninstall the earlier version, without deleting any queue managers, and install the latest version in the same default location.

Whether you have to uninstall your previous version of the product depends upon your operating system.

On the following platforms, you do not have to uninstall a previous version of the product:

- AIX
- Windows
- IBM i IBM i, where the process is known as a *slip* install

AIX If `mqm.xr.clients` and `mqm.txclient.rte` file sets from earlier versions are installed, you must uninstall these file sets from the earlier versions.

On the following platforms, you have to uninstall the previous version of the product:

- HP-UX
- Linux
- Solaris

- A reason for installing into the same location is to simplify application migration. If you change the installation location, you might remove IBM MQ libraries from an application search path. To migrate an application search path you must modify the application environment, or more rarely, the application itself.
- The default installation path is specified as a load path in the IBM MQ build scripts for UNIX and Linux. After installation of the latest version, the load libraries of the latest version of IBM MQ are in the same location as were the libraries of the earlier version. If you built applications by following the examples in the product documentation for the earlier version, the applications load the correct libraries in the latest version.

5. Optional: Make the latest version of the installation the primary installation.

   a) Run the **`setmqinst`** command

   On Windows

   ```
   "Inst_1_INSTALLATION_PATH\bin\setmqinst" -i -n Inst_1
   ```

   On UNIX

   ```
   Inst_1_INSTALLATION_PATH/bin/setmqinst -i -n Inst_1
   ```

   Note: Use the dspmqinst command to discover the `<Installation name>`, or use the default value `<Installation 1>`.

- Make the installation primary to avoid specifying a search path to run IBM MQ commands.
- If there is a primary installation, UNIX and Linux applications that expect to find the IBM MQ library in `/usr/lib`, find a symbolic link to the library in `/usr/lib/32`[7]. `/usr/lib/32` is normally in the default search path. It is also specified as a load path in the IBM MQ build scripts for UNIX and Linux.
- It is sufficient to link applications only to `/usr/lib`. With a primary installation of the latest version of the product defined on the server, an application can connect to any queue manager associated with any installation on the server. IBM MQ loads the correct library for the application.

6. Start the queue managers and applications.

   a) Optional: Run the **`setmqm`** command to associate the queue managers with `Inst_1`.

   ```
   setmqm -m QM1 -n Inst_1
   setmqm -m QM2 -n Inst_1
   ```

[7] `/usr/lib` for 64 bit applications.

Notes:

- The **setmqm** step is optional only in the case where migration is from IBM WebSphere MQ 7.0.1 to a later release. In this case, the **strmqm** command automatically associates the queue manager with its own installation.
- If you are migrating between any other releases of IBM MQ, you must use **setmqm** to associate the queue managers with the new installation manually.

Windows If you have multiple installations, note that queue managers that were configured to start automatically, and remain after uninstalling IBM WebSphere MQ 7.0.1, automatically start under any other existing Version 7.1 (or later) installation when either the machine reboots, or the Service for that installation is restarted. In order to avoid this, ensure that all queue managers have been moved to the desired installation before uninstalling IBM WebSphere MQ 7.0.1.

b) Run the **strmqm** command to start the queue managers and migrate them to the latest version of the product.

```
strmqm QM1
strmqm QM2
```

At this point, queue manager data is migrated and you cannot revert to a previous release.

- When an application connects to a queue manager, the operating system searches its load path to load the IBM MQ library [8] . A Version 7.1, or later, library contains code that checks that the queue manager is associated with an installation. If a queue manager is associated with a different installation, IBM MQ loads the correct IBM MQ library for the installation the queue manager is associated with.

What to do next

You cannot reinstall an earlier version of the product on a system that has the latest, or any other, version of IBM MQ installed.

Related concepts

"Queue manager coexistence in Version 8.0" on page 53
Queue managers, with different names, can coexist on any server as long as they use the same IBM MQ installation. On z/OS z/OS, UNIX, Linux, and Windows, different queue managers can coexist on the same server and be associated with different installations.

"Multi-installation queue manager coexistence on UNIX, Linux, and Windows" on page 57
You can install multiple copies of IBM MQ for UNIX, Linux, and Windows on the same server. The installations must be at Version 7.1 or later, with one exception. One Version 7.0.1 installation, at fix pack level 6, or later, can coexist with multiple Version 7.1, or later installations.

Related tasks

"IBM MQ migration planning to the latest version on UNIX platforms, Windows, and IBM i" on page 80

"UNIX systems - migrating a queue manager from your current version to the latest version" on page 108
Follow these instructions to migrate a queue manager from your current version to the latest version.

"Windows: Migrating a queue manager from a previous version to the latest version" on page 110
Follow these instructions to migrate a queue manager from your current version to the latest version.

"UNIX, Linux, and Windows: Side-by-side migration to a later version" on page 96

"UNIX, Linux, and Windows: Multi-stage migration to a later version" on page 100

"Migrating IBM MQ library loading from an earlier version of the product to the latest version" on page 131

[8] On Windows, the IBM MQ library is a DLL. A DLL is sometimes called a load library or a shared library. The entry points to a DLL are defined in a link library, with the file extension `.lib32` or `.lib`. The `.lib` library is linked at build-time and the DLL loaded at runtime.

No change in the way IBM MQ libraries are loaded is normally required if you upgrade from an earlier version of the product to the latest version. You must have followed the instructions on building IBM MQ applications in Version 7.0.1 and you must replace IBM WebSphere MQ 7.0.1 with the latest version of the product. If you choose to take advantage of multi-installation in the latest version of the product, based on the side-by-side or multi-stage migration scenarios, you must modify the environment for the operating system to resolve IBM MQ dependencies for an application. Typically, you can modify the runtime environment, rather than relink the application.

Related information

Installing IBM MQ server on AIX

Installing IBM MQ server on HP-UX

Installing IBM MQ server on Linux

Installing IBM MQ server on Solaris

Installing IBM MQ server on Windows

Associating a queue manager with an installation

Changing the primary installation

Choosing an installation name

setmqenv

setmqinst

setmqm

UNIX, Linux, and Windows: Side-by-side migration to a later version

Side-by-side migration is the term used to describe installing a new version of IBM MQ alongside an older version on the same server. Queue managers remain running during the installation and verification of the new version of IBM MQ. They remain associated with the older version of IBM MQ. When you decide to migrate queue managers to the new version of IBM MQ, you stop all queue managers, uninstall the old version, and migrate them all to the new version of IBM MQ.

Before you begin

If you are using IBM WebSphere MQ 7.0.1, you must ensure that you are running IBM WebSphere MQ 7.0.1, Fix Pack 6 before installing the latest version of the product on the same server. Go to Fix Central to obtain the fix pack.

This scenario is one of three, which describe alternative ways to upgrade queue managers from an earlier version of the product. The other scenarios are as follows:

1. Replace the earlier version with the latest version; see “UNIX, Linux, and Windows: Single-stage migration to a later version” on page 92.
2. Run the latest version of the product alongside an earlier version ; see “UNIX, Linux, and Windows: Multi-stage migration to a later version” on page 100.

Read these three tasks to plan how you are going to migrate to the multi-installation environment of the latest version. The step-by-step migration scenario sits half-way between the single-stage and multi-stage migration scenarios.

These topics are for planning multi-installation migration. The planning topics guide you in deciding what other tasks you must perform to migrate queue managers and applications to the latest version. For the precise sequence of commands to upgrade a queue manager to the latest version, do the migration task for the platform you are interested in. All the tasks are listed by platform in the links at the end of this topic. As part of the queue manager migration task, back up your existing queue manager data. Even on a multi-installation server, queue managers cannot be restored to a previous command level after migration.

About this task

In the "side-by-side" migration scenario, you install the latest version of IBM MQ alongside queue managers that continue to be associated with Version 7.0.1, or later.

When you are ready to migrate the queue managers, and applications, to the latest version:

1. Stop all the queue managers.
2. Uninstall the earlier version of the product.
3. Migrate all the queue managers and applications to the latest version.

The side-by-side migration scenario is less flexible than multi-stage migration, and might not seem to have any advantages over it. However, side-by-side migration does have advantages over the multi-stage and single-stage approaches. With the side-by-side approach, because you uninstall the earlier version before starting any queue managers, you can assign an installation on the latest version to be the primary installation.

In the multi-stage approach, you cannot set an installation of the latest version to be the primary installation while you continue to run the earlier version.

Having the latest version installation as the primary installation has two benefits.

1. With the latest version having the primary installation, many applications restart without reconfiguring their environment.
2. IBM MQ commands run against the primary installation, work without providing a local search path.

The advantage the side-by-side scenario has over the single-stage scenario is that you can install and verify the installation of the latest version of the product on the server before switching over to it.

Four types of object are important to consider during migration:

- Installations
- Queue managers
- Administrative procedures
- Applications

Administrative procedures contain IBM MQ commands, and scripts that use commands.

To run a command, the operating system must find the command in an IBM MQ installation. For some commands, you must run the command from the installation that is associated with the correct queue manager. IBM MQ does not switch commands to the correct installation. For other commands, such as **`setmqinst`**, you can run the command from any installation that has the latest version of the product installed.

If an earlier version of the product is installed, the command that is run is the command for that version, unless the search path is overridden by a local setting. You can override the search path by running **`setmqenv`**. If Version 7.0.1 is not installed, you must set the correct path to run a command. If you have set a primary installation, the command that is run is the copy in the primary installation, unless you override the selection with a local search path.

Procedure

1. Install the latest version in a different installation directory from the earlier version.
   a) Decide on an installation naming convention. Give the installation a name of your choosing, or accept the default installation name.

      For the first installation, the default name is *Installation1*. For the second installation, the name is *Installation2*, and so on.

      AIX On AIX there is no option to set the installation name, *Installation1* is set by default.

   b) Verify the installation.

      Run the installation verification procedures and your own tests.

2. Uninstall the earlier version of the product.

   When uninstalling the earlier product, you must stop all queue managers and applications that have loaded an IBM MQ library on the server. For this reason, you might choose to postpone uninstalling the earlier version of the product until a convenient maintenance window. When an earlier version of the product is not installed on a server, it is sufficient to stop the queue managers and applications that have loaded libraries from the installation that you are uninstalling or updating. It is not necessary to stop applications and queue managers associated with other installations.

   a) Stop all applications that have loaded IBM MQ libraries on the server.

   b) Stop the queue managers and listeners on the server.

   c) Uninstall the earlier version of the product.

      - Stop all local IBM MQ applications
      - If you are migrating from IBM WebSphere MQ 7.0.1, stop all your queue managers and listeners, ensuring that you do not delete the queue managers.

        Note: If you are migrating from a release other than IBM WebSphere MQ 7.0.1 you do not need to stop all the queue managers at this point.

3. Make the latest version of the installation the primary installation.

   a) Run the **setmqinst** command

      On Windows

      ```
      "Inst_1_INSTALLATION_PATH\bin\setmqinst" -i -n Inst_1
      ```

      On UNIX

      ```
      Inst_1_INSTALLATION_PATH/bin/setmqinst -i -n Inst_1
      ```

      Note: Use the dspmqinst command to discover the `<Installation name>`, or use the default value `<Installation 1>`.

   - Make the installation primary to avoid specifying a search path to run IBM MQ commands.
   - If there is a primary installation, UNIX and Linux applications that expect to find the IBM MQ library in `/usr/lib`, find a symbolic link to the library in `/usr/lib/32`[9]. `/usr/lib/32` is normally in the default search path. It is also specified as a load path in the IBM MQ build scripts for UNIX and Linux.
   - It is sufficient to link applications only to `/usr/lib`. With a primary installation of the latest version of the product defined on the server, an application can connect to any queue manager associated with any installation on the server. IBM MQ loads the correct library for the application.

   Use the dspmqinst command to discover the `<Installation name>`, or use the default value `<Installation 1>`.

   Doing this means that you do not have to specify a search path on IBM MQ commands.

4. Start the queue managers and applications.

   a) Optional: Run the **setmqm** command to associate the queue managers with `Inst_1`.

      ```
      setmqm -m QM1 -n Inst_1
      setmqm -m QM2 -n Inst_1
      ```

      Notes:

[9] `/usr/lib` for 64 bit applications.

- The **setmqm** step is optional only in the case where migration is from IBM WebSphere MQ 7.0.1 to a later release. In this case, the **strmqm** command automatically associates the queue manager with its own installation.
- If you are migrating between any other releases of IBM MQ, you must use **setmqm** to associate the queue managers with the new installation manually.

Windows If you have multiple installations, note that queue managers that were configured to start automatically, and remain after uninstalling IBM WebSphere MQ 7.0.1, automatically start under any other existing Version 7.1 (or later) installation when either the machine reboots, or the Service for that installation is restarted. In order to avoid this, ensure that all queue managers have been moved to the desired installation before uninstalling IBM WebSphere MQ 7.0.1.

b) Run the **strmqm** command to start the queue managers and migrate them to the latest version of the product.

```
strmqm QM1
strmqm QM2
```

At this point, queue manager data is migrated and you cannot revert to a previous release.

- When an application connects to a queue manager, the operating system searches its load path to load the IBM MQ library [10] . A Version 7.1, or later, library contains code that checks that the queue manager is associated with an installation. If a queue manager is associated with a different installation, IBM MQ loads the correct IBM MQ library for the installation the queue manager is associated with.

During this process you continue to use queue manager QM2 while you upgrade queue manager QM1 and you use queue manager QM1 while you upgrade QM2.

Note that each queue manager needs to be stopped in order to be associated with the new installation.

What to do next

You cannot reinstall an earlier version of the product on a system that has the latest, or any other, version of IBM MQ installed.

"IBM MQ migration planning to the latest version on UNIX platforms, Windows, and IBM i" on page 80

Installing IBM MQ server on AIX

Uninstalling IBM MQ on AIX

Uninstalling IBM MQ on HP-UX

Installing IBM MQ server on HP-UX

Uninstalling IBM MQ on Linux

Installing IBM MQ server on Linux

Uninstalling IBM MQ on Solaris

Installing IBM MQ server on Solaris

Uninstalling IBM MQ on Windows systems

Installing IBM MQ server on Windows

Associating a queue manager with an installation

"UNIX, Linux, and Windows: Single-stage migration to a later version" on page 92
Single-stage migration, is the term used to describe replacing the only installation of IBM MQ on a server, with a later release. Single stage migration is also known as `upgrading in place` or `in place upgrade` . Until Version 7.0.1.6, single-stage was the only migration scenario. Single-stage migration preserves existing scripts and procedures for running IBM MQ the most. With other migration scenarios

[10] On Windows, the IBM MQ library is a DLL. A DLL is sometimes called a load library or a shared library. The entry points to a DLL are defined in a link library, with the file extension `.lib32` or `.lib`. The `.lib` library is linked at build-time and the DLL loaded at runtime.

you might change some scripts and procedures, but you can reduce the effect queue manager migration has on users.

“UNIX, Linux, and Windows: Multi-stage migration to a later version” on page 100
Changing the primary installation
Choosing an installation name
“Queue manager coexistence in Version 8.0” on page 53
Queue managers, with different names, can coexist on any server as long as they use the same IBM MQ installation. On z/OS z/OS, UNIX, Linux, and Windows, different queue managers can coexist on the same server and be associated with different installations.

“Migrating IBM MQ library loading from an earlier version of the product to the latest version” on page 131
No change in the way IBM MQ libraries are loaded is normally required if you upgrade from an earlier version of the product to the latest version. You must have followed the instructions on building IBM MQ applications in Version 7.0.1 and you must replace IBM WebSphere MQ 7.0.1 with the latest version of the product. If you choose to take advantage of multi-installation in the latest version of the product, based on the side-by-side or multi-stage migration scenarios, you must modify the environment for the operating system to resolve IBM MQ dependencies for an application. Typically, you can modify the runtime environment, rather than relink the application.

“Multi-installation queue manager coexistence on UNIX, Linux, and Windows” on page 57
You can install multiple copies of IBM MQ for UNIX, Linux, and Windows on the same server. The installations must be at Version 7.1 or later, with one exception. One Version 7.0.1 installation, at fix pack level 6, or later, can coexist with multiple Version 7.1, or later installations.

setmqenv
setmqinst
setmqm

UNIX, Linux, and Windows: Multi-stage migration to a later version

Multi-stage migration is the term used to describe running a new version of IBM MQ alongside an older version on the same server. After installing the new version alongside the old, you can create new queue managers to verify the new installation, and develop new applications. At the same time, you can migrate queue managers and their associated applications from the old version to the new. By migrating queue managers and applications one-by-one, you can reduce the peak workload on staff managing the migration.

Before you begin

If you are using IBM WebSphere MQ 7.0.1, you must ensure that you are running IBM WebSphere MQ 7.0.1, Fix Pack 6 before installing the latest version of the product on the same server. Go to Fix Central to obtain the fix pack.

This scenario is one of three, which describe alternative ways to upgrade queue managers from an earlier version of the product. The other scenarios are as follows:

1. Replace the earlier version with the latest version; see “UNIX, Linux, and Windows: Single-stage migration to a later version” on page 92.
2. Install the latest version of the product alongside an earlier version ; see “UNIX, Linux, and Windows: Side-by-side migration to a later version” on page 96.

Read these three tasks to plan how you are going to migrate to the multi-installation environment of the latest version. The multi-stage migration scenario is the most flexible approach to migrating from IBM WebSphere MQ 7.0.1 to the latest version.

These topics are for planning multi-installation migration. The planning topics guide you in deciding what other tasks you must perform to migrate queue managers and applications to the latest version. For the precise sequence of commands to upgrade a queue manager to the latest version, do the migration task for the platform you are interested in. All the tasks are listed by platform in the links at the end of this topic. As part of the queue manager migration task, back up your existing queue manager data. Even on a multi-installation server, queue managers cannot be restored to a previous command level after migration.

Note:

- If an application uses COM or ActiveX it can connect to any queue manager as long as there is a primary installation and it is Version 7.1 or later.
- If you are running the IBM MQ.NET monitor in transactional mode, the queue manager it connects to must be the primary installation.

You cannot migrate these applications to the latest version until you uninstall the earlier version.

About this task

In the multi-stage migration scenario, you install the latest version of the product alongside running queue managers that continue to be associated with the earlier version. You can create queue managers and run new applications using the latest version installation. When you are ready to start migrating queue managers and applications from the earlier, you can do so, one-by-one. When migration to the latest version is complete, uninstall the earlier version, and make the latest version installation the primary installation.

With the multi-stage approach, until you uninstall the earlier version , you must configure an environment to run applications that connect to a queue manager to the latest version. You must also provide a path to run IBM MQ commands. Both these tasks are accomplished with the **setmqenv** command.

Note: When you have uninstalled the earlier version, and set the latest version as a primary installation, in most circumstances it is not necessary to run the **setmqenv** command to run applications. It is still necessary to run **setmqenv** to set the environment for commands that connect to a queue manager associated with an installation that is not primary.

Four types of object are important to consider during migration:

- Installations
- Queue managers
- Administrative procedures
- Applications

Administrative procedures contain IBM MQ commands, and scripts that use commands.

To run a command, the operating system must find the command in an IBM MQ installation. For some commands, you must run the command from the installation that is associated with the correct queue manager. IBM MQ does not switch commands to the correct installation. For other commands, such as **setmqinst**, you can run the command from any installation that has the latest version of the product installed.

If an earlier version of the product is installed, the command that is run is the command for that version, unless the search path is overridden by a local setting. You can override the search path by running **setmqenv**. If Version 7.0.1 is not installed, you must set the correct path to run a command. If you have set a primary installation, the command that is run is the copy in the primary installation, unless you override the selection with a local search path.

Procedure

1. Install the latest version in a different installation directory from the earlier version and verify the installation.
   a) Decide on an installation naming convention. Give the installation a name of your choosing, or accept the default installation name.

      For the first installation, the default name is *Installation1*. For the second installation, the name is *Installation2*, and so on.

AIX On AIX there is no option to set the installation name, *Installation1* is set by default.

b) Verify the installation.

Run the installation verification procedures and your own tests.

- You might create new queue managers running the latest version, and start to develop new applications before migrating applications from the earlier version.

2. Configure the operating system so that applications load the libraries for the latest version of the product.

a) Migrate queue managers one at a time.

The first set of applications to load the libraries for the latest version of the product are the applications that connect to the first queue manager you are going to migrate.

It does not matter if those applications also connect to other queue managers on the server. If the applications load the latest version libraries, IBM MQ automatically loads the libraries for the earlier version for those applications that connect to that version.

You can either migrate the operating system environment of all applications, or just the applications that connect to the first queue manager you are going to migrate.

b) Migrate IBM MQ MQI client applications

Some of the applications might be running as IBM MQ MQI client applications on another workstation. When you migrate a queue manager, clients connected to it continue to run without loading a client library for the latest version.

You can migrate these clients later, when you need to do so.

Important: If any IBM MQ MQI client applications are using the library for the earlier version on the server, you must eventually migrate the clients to use the latest version of the product before you uninstall the earlier version.

3. Migrate an application to load the new library for the latest version:

- Run **setmqenv** to modify the local path that is searched for IBM MQ libraries.
- Modify the global search path that is searched for IBM MQ libraries.
- Relink applications with an additional runtime load path.

Consult operating system documentation about how to modify the global search path, or include a fixed runtime load path in the application load module.

To run **setmqenv** using the `-s` option:

Windows:

```
"Inst_1_INSTALLATION_PATH\bin\setmqenv" -s
```

The `-s` option sets up the environment for the installation that runs the **setmqenv** command.

UNIX:

```
.Inst_1_INSTALLATION_PATH/bin/setmqenv -s -k
```

The `-k` option inserts the path to the IBM MQ load libraries at the start of the `LD_LIBRARY_PATH` environment variable, and adds the variable to the local environment; see "Loading IBM MQ libraries" on page 58.

Note: On UNIX the leading `" . "` is critical. The dot followed by a space instructs the command shell run **setmqenv** in the same command shell and inherit the environment set by **setmqenv**.

4. Restart the queue manager and the applications that connect to it.

a) Set up the local environment to the installation `Inst_1`.

Windows:

```
"Inst_1_INSTALLATION_PATH\bin\setmqenv" -s
```

The -s option sets up the environment for the installation that runs the **setmqenv** command.

UNIX:

```
.Inst_1_INSTALLATION_PATH/bin/setmqenv -s
```

b) Run the **setmqm** command to associate QM1 with Inst_1.

```
setmqm -m QM1 -n Inst_1
```

c) Run the **strmqm** command to start QM1 and migrate it to the latest version.

```
strmqm QM1
```

d) Restart application 1

The application loads the latest version library and connects to QM1, which is associated with the latest version of the product.

5. Migrate all queue managers and applications to the latest version.

   Repeat steps “2” on page 102 and “4” on page 102, when required, until all the queue managers and applications are migrated to the latest version of the product.

6. Uninstall the earlier version of the product.

   When uninstalling the earlier product, you must stop all queue managers and applications that have loaded an IBM MQ library on the server. For this reason, you might choose to postpone uninstalling the earlier version of the product until a convenient maintenance window. When an earlier version of the product is not installed on a server, it is sufficient to stop the queue managers and applications that have loaded libraries from the installation that you are uninstalling or updating. It is not necessary to stop applications and queue managers associated with other installations.

   a) Stop all applications that have loaded IBM MQ libraries on the server.

   b) Stop the queue managers and listeners on the server.

   c) Uninstall the earlier version of the product.

   - Stop all local IBM MQ applications
   - If you are migrating from IBM WebSphere MQ 7.0.1, stop all your queue managers and listeners, ensuring that you do not delete the queue managers.

     Note: If you are migrating from a release other than IBM WebSphere MQ 7.0.1 you do not need to stop all the queue managers at this point.

7. Make Inst_1 the primary installation.

   a) Run the **setmqinst** command

On Windows

```
"Inst_1_INSTALLATION_PATH\bin\setmqinst" -i -n Inst_1
```

On UNIX

```
Inst_1_INSTALLATION_PATH/bin/setmqinst -i -n Inst_1
```

Note: Use the dspmqinst command to discover the `<Installation name>`, or use the default value `<Installation 1>`.

- You do not have to set up a search path to run IBM MQ commands from the primary installation.
- If you set an installation of the latest version of the product as primary on UNIX and Linux, you do not have to set up `LD_LIBRARY_PATH` in most cases. You can remove calls to **`setmqenv`** to set `LD_LIBRARY_PATH`.

What to do next

You cannot reinstall an earlier version of the product on a system that has the latest, or any other, version of IBM MQ installed.

Now that you have uninstalled the earlier version of the product, and made the latest installation primary, you can review how the application runtime environment is set. It is no longer necessary to run **`setmqenv`** to set up the search path to load libraries for the latest version. If you have only one installation of the latest version of the product installed, it is not necessary to run **`setmqenv`** to run commands.

Related concepts

"Queue manager coexistence in Version 8.0" on page 53
Queue managers, with different names, can coexist on any server as long as they use the same IBM MQ installation. On z/OS z/OS, UNIX, Linux, and Windows, different queue managers can coexist on the same server and be associated with different installations.

"Multi-installation queue manager coexistence on UNIX, Linux, and Windows" on page 57
You can install multiple copies of IBM MQ for UNIX, Linux, and Windows on the same server. The installations must be at Version 7.1 or later, with one exception. One Version 7.0.1 installation, at fix pack level 6, or later, can coexist with multiple Version 7.1, or later installations.

Related tasks

"IBM MQ migration planning to the latest version on UNIX platforms, Windows, and IBM i" on page 80

"UNIX, Linux, and Windows: Single-stage migration to a later version" on page 92
Single-stage migration, is the term used to describe replacing the only installation of IBM MQ on a server, with a later release. Single stage migration is also known as `upgrading in place` or `in place upgrade` . Until Version 7.0.1.6, single-stage was the only migration scenario. Single-stage migration preserves existing scripts and procedures for running IBM MQ the most. With other migration scenarios you might change some scripts and procedures, but you can reduce the effect queue manager migration has on users.

"UNIX, Linux, and Windows: Side-by-side migration to a later version" on page 96

"Migrating IBM MQ library loading from an earlier version of the product to the latest version" on page 131
No change in the way IBM MQ libraries are loaded is normally required if you upgrade from an earlier version of the product to the latest version. You must have followed the instructions on building IBM MQ applications in Version 7.0.1 and you must replace IBM WebSphere MQ 7.0.1 with the latest version of the product. If you choose to take advantage of multi-installation in the latest version of the product, based on the side-by-side or multi-stage migration scenarios, you must modify the environment for the operating system to resolve IBM MQ dependencies for an application. Typically, you can modify the runtime environment, rather than relink the application.

Related information
Installing IBM MQ server on AIX
Installing IBM MQ server on HP-UX
Installing IBM MQ server on Linux
Installing IBM MQ server on Solaris
Installing IBM MQ server on Windows
Associating a queue manager with an installation
Changing the primary installation
Choosing an installation name
setmqenv
setmqinst
setmqm

Migrating IBM MQ Telemetry from Version 7.0.1 to Version 8.0

Migrate IBM MQ Telemetry from Version 7.0.1 to Version 8.0 by completing the tasks in this section. You must stop all IBM MQ activity on the system before migrating.

About this task

In IBM WebSphere MQ 7.0.1, IBM MQ Telemetry was a separate feature. Because IBM MQ Telemetry is a component of IBM WebSphere MQ 7.1 and later, upgrading is essentially uninstalling IBM MQ Telemetry and installing a later version of IBM MQ. IBM MQ Telemetry can either be installed with the main product, or installed after the main product has been installed.

After the successful upgrade, Linux systems retain all telemetry data kept in `/var/mqm`, and the Windows systems retain the telemetry data in the installation directory of the product, for example: `C:\Program Files (x86)\WebSphere MQ`. Telemetry data is migrated to the later version of the product when the queue manager is started again.

Notes:

1. You can administer IBM MQ Telemetry Version 7.0.1 only from the Version 7.0.1 MQ Explorer. If you connect the Version 8.0 explorer remotely to a Version 7.0.1 queue manager, no telemetry resources are displayed. You cannot connect a Version 8.0 MQ Explorer locally to a Version 7.0.1 queue manager on the same server.
2. From IBM MQ 8.0, the Client Software Development Kit (SDK) is no longer supplied as part of the product. MQTT client libraries are freely available from Eclipse Paho and MQTT.org. See IBM MQ Telemetry Transport sample programs.

The migration tasks, depending on your platform, are described in the subtopics.

Related information
Installing IBM MQ
Installing IBM MQ Telemetry

Windows: Migrating IBM MQ Telemetry from Version 7.0.1 to Version 8.0

Follow these instructions to migrate IBM MQ Telemetry from Version 7.0.1 to Version 8.0 on Windows.

Before you begin

Before proceeding with this task, ensure that you back up your existing IBM MQ installation. You must stop the IBM MQ Telemetry service `SYSTEM.MQXR.SERVICE` before migrating.

About this task

This task outlines the steps necessary to migrate your existing installation of IBM MQ Telemetry to Version 8.0 on Windows systems.

Procedure

1. Uninstall IBM WebSphere MQ 7.0.1 using the control panel, following this procedure:
   a) From the Windows task bar, click **Start > Control Panel**. (On some systems, click **Start > Settings > Control Panel**)
   b) When uninstalling IBM MQ Telemetry by using the control panel, on Windows 7 systems, click **Programs and Features**, on other Windows systems, click **Add or Remove Programs**.
   c) Click IBM MQ Telemetry, then click **Change/Remove**. The uninstaller starts and summarizes items to be uninstalled.
   d) Click **Uninstall**. The uninstaller lists the items being uninstalled. The **Uninstall Complete** screen displays the status of the uninstall. Click **Done**
2. Verify that the data folders still exist. For example, they might be located at `C:\Program Files (x86)\WebSphere MQ`.
3. IBM MQ Telemetry is installed in one of two ways:
   - Installed as part of a **Custom** installation at the same time as IBM MQ 8.0 during migration: Begin here: "IBM MQ migration planning to the latest version on UNIX platforms, Windows, and IBM i" on page 80
   - Added at a later date to an existing installation of IBM MQ 8.0 or later: Begin here: Installing IBM MQ Telemetry
4. Verify that the IBM MQ Telemetry migration was successful. See Verifying the installation of IBM MQ Telemetry .

Results

Message AMQ4616 indicates completion of the task. The existing MQTT channels and previous subscriptions are still present.

Related information

Installing IBM MQ Telemetry

Verifying the installation of IBM MQ Telemetry

Verifying the installation of IBM MQ Telemetry by using IBM MQ Explorer

Linux: Migrating from IBM MQ Telemetry Version 7.0.1 to Version 8.0

Follow these instructions to migrate IBM MQ Telemetry from Version 7.0.1 to Version 8.0 on Linux.

Before you begin

Before proceeding with this task, ensure that you back up your existing IBM MQ installation. You must stop the IBM MQ Telemetry service `SYSTEM.MQXR.SERVICE` before migrating.

About this task

This task outlines the steps necessary to migrate your existing installation of IBM MQ Telemetry to Version 8.0 on Linux systems.

Procedure

1. Uninstall IBM MQ Telemetry Version 7.0.1, following this procedure:
   a) Navigate to the uninstallation directory. The default location on Linux is `/opt/mqm/mqxr/Uninstall_MQTT`.

b) Start the uninstaller, using the executable or binary file. On Linux systems, run `./Uninstall_MQTelemetry -i GUI`. The uninstaller starts and summarizes what is to be uninstalled.

c) Click **Uninstall**. The uninstaller lists the items being uninstalled.

d) Click **Done**.

2. Verify that the data folders still exist. Consult your *MQ_INSTALLATION_PATH* environment variable to find these folders.
3. IBM MQ Telemetry is installed in one of two ways:
   - Installed as part of a **Custom** installation at the same time as IBM MQ 8.0 during migration: Begin here: "IBM MQ migration planning to the latest version on UNIX platforms, Windows, and IBM i" on page 80
   - Added at a later date to an existing installation of IBM MQ 8.0 or later: Begin here: Installing IBM MQ Telemetry
4. Verify that the IBM MQ Telemetry migration was successful. See Verifying the installation of IBM MQ Telemetry .

Results

Message AMQ4616 indicates completion of the task. The existing MQTT channels and previous subscriptions are still present.

Related information

Installing IBM MQ Telemetry
Verifying the installation of IBM MQ Telemetry
Verifying the installation of IBM MQ Telemetry by using IBM MQ Explorer

Migrating a queue manager to the latest release

The procedures for migrating a queue manager to the latest release are detailed in the following topics.

Before you begin

This information applies to migrating a queue manager on platforms other than z/OS.

If you have installed early support program code on the server, you must delete all the queue managers created with the installation. Uninstall the code before proceeding with installing the production level code.

About this task

Migration is a complex task. It goes beyond upgrading the IBM MQ code. The term "upgrading" applies to the process of installing a new code level. "Migrating" refers to the process of upgrading code and the to the task of migrating queue manager data, such as messages, queues, channels, and other resources. Migrating queue manager data is automatic.

Procedure

Queue manager migration follows this overall plan:

1. Review the IBM MQ system requirements for the latest version; see System Requirements for IBM MQ.
2. Back up your system. In particular back up the queue manager.

   You cannot run a queue manager on a previous version of IBM MQ, once you start the queue manager on the new version.
3. Carry out the platform-specific migration or upgrade procedure.

Consult “IBM MQ migration planning to the latest version on UNIX platforms, Windows, and IBM i” on page 80.

a) Upgrade IBM MQ.

b) Customize the configuration.

z/OS

Configuration changes are typically required only on z/OS.

c) Verify the installation.

Create a queue manager to verify the installation. Verify applications and management tasks work with the new level of IBM MQ before migrating existing queue managers.

4. Perform any additional migration tasks that are required.

   If you are using publish/subscribe, you must migrate the publish/subscribe broker.

   If the queue manager is a member of a queue manager cluster, or queue-sharing group, migrate the other members of the cluster or group.

Important: You must migrate the publish/subscribe broker state before you migrate your IBM MQ system to IBM MQ 8.0, as broker publish/subscribe migration is not supported in IBM MQ 8.0.

System Requirements for IBM MQ

UNIX systems - migrating a queue manager from your current version to the latest version

Follow these instructions to migrate a queue manager from your current version to the latest version.

Before you begin

1. The upgrade from your current version to the latest version of the product requires a full migration of queue managers. Create a migration plan. Use the planning task, Planning migration to the latest version , as a guide.
2. Review the IBM MQ system requirements for the latest version; see System Requirements for IBM MQ.
3. Back up your system before you install the latest release of IBM MQ over a previous release. Once you have started a queue manager you cannot revert to the previous release.

   If you must restore the system, you cannot recover any work, such as changes to messages and objects, performed by the latest version of IBM MQ. For more information about backing up your system, see Backing up and restoring IBM MQ queue manager data.
4. Review any other installed SupportPacs for their applicability to the latest release.
5. If you are running on a server with multiple IBM MQ installations, you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

About this task

Complete this task to migrate a queue manager to the latest version of IBM MQ from an earlier version.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM MQ installation.

   If you use the MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM MQ installation.

   a) Run the **dspmq** command to list the state of all the queue managers on the system.

Run either of the following commands from the installation that you are updating:

```
dspmq -o installation -o status
dspmq -a
```

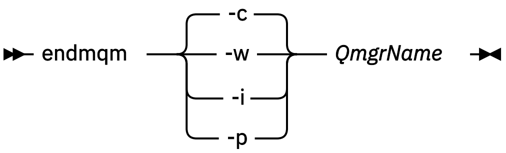
`dspmq -o installation -o status` displays the installation name and status of queue managers associated with all installations of IBM MQ.

`dspmq -a` displays the status of active queue managers associated with the installation from which the command is run.

b) Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

```
echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
```

c) Run the **endmqm** command to stop each running queue manager associated with this installation.



The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM MQ shared libraries from the installation loaded prevent you applying IBM MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep an IBM MQ shared library loaded.

Note: The topic, "Applying maintenance level updates to multi-instance queue managers" on page 249, describes how to apply maintenance to a multi-instance queue manager. A multi-instance queue manager can continue to run on one server, while maintenance is applied to another server.

d) Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Back up the queue manager.
5. Log in as root.
6. If you are running IBM WebSphere MQ 7.0.1.6 or later, optionally uninstall the current version of IBM MQ.

   Note, that you carry out this step only if you are doing a single stage migration; see "UNIX, Linux, and Windows: Single-stage migration to a later version" on page 92
7. If you are running IBM WebSphere MQ 7.0.1.5 or earlier, uninstall the current version of IBM MQ.

   If you require a side-by-side migration or multistage migration only, you must upgrade to IBM WebSphere MQ 7.0.1.6
8. Install the latest version of the product.

   See the appropriate topic for the platform that your enterprise uses:

Installing IBM MQ Server on AIX .
Installing IBM MQ Server on HP-UX.
Installing IBM MQ Server on Linux .
Installing IBM MQ Server on Solaris.

9. Move the queue manager to the new IBM MQ installation.

   You need to carry out this step, only if you are running IBM WebSphere MQ 7.0.1.6 or later, and did not uninstall your current version of IBM MQ.

   See "UNIX, Linux, and Windows: Side-by-side migration to a later version" on page 96 or "UNIX, Linux, and Windows: Multi-stage migration to a later version" on page 100 for further information.

10. Start the queue manager.

    ```
    strmqm QmgrName
    ```

    When you first start a queue manager after migration:

    - Any new attributes for existing objects are set to their default values.
    - Any new default objects are created.
    - Queue manager data is migrated.

    Important: Do not use the -c option to start the queue manager, unless you explicitly want to reset or recreate the default system objects.

    You must start IBM MQ before you start any listeners.

Backing up and restoring a queue manager
"The version naming scheme for IBM MQ (On platforms other than z/OS)" on page 21
IBM MQ releases have a four-digit Version, Release, Modification, and Fix (VRMF) level code.

Windows: Migrating a queue manager from a previous version to the latest version

Follow these instructions to migrate a queue manager from your current version to the latest version.

Before you begin

1. Create a migration plan; see Planning migration to the latest version
2. For more information about the versions of Windows that IBM MQ 8.0 supports, see System Requirements for IBM MQ V8.0.
3. Review the IBM MQ system requirements for the latest version; see System Requirements for IBM MQ.
4. Back up your system before you install the latest release of IBM MQ over a previous release. Once you have started a queue manager you cannot revert to the previous release.

   If you must restore the system, you cannot recover any work, such as changes to messages and objects, performed by the latest version of IBM MQ. For more information about backing up your system, see Backing up and restoring IBM MQ queue manager data.
5. Review any other installed SupportPacs for their applicability to the latest release.
6. If you are running on a server with multiple IBM MQ installations, you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

About this task

Complete this task to migrate a queue manager to the latest version of IBM MQ from an earlier version.

All the objects that you previously created are maintained. The components that were previously installed are preselected in the feature options when you install the new level. If you leave these components selected, you can keep them or reinstall them. If you clear any of these components, the installation process uninstalls them. By default, a typical migration installs only the same features that were installed in the previous version installation.

For example, if MQ Explorer was not installed in an earlier installation, it is not stored in a later installation. If you want MQ Explorer, select a custom installation, and select the MQ Explorer feature on the **Features** panel. If you do not want MQ Explorer, uninstall the MQ Explorer feature by selecting a custom installation. Then clear the MQ Explorer feature on the **Features** panel. For more information about how to uninstall features, see Modifying the installation using IBM MQ Installation Launchpad.

You can also migrate a queue manager to a later release, on a system where a previous version has been uninstalled. The queue manager data must have been retained, or restored from a backup.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM MQ installation.

   If you use the MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM MQ installation.

   a) Run the **dspmq** command to list the state of all the queue managers on the system.

   Run either of the following commands from the installation that you are updating:

   ```
   dspmq -o installation -o status
   dspmq -a
   ```

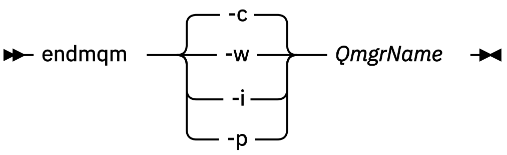
   `dspmq -o installation -o status` displays the installation name and status of queue managers associated with all installations of IBM MQ.

   `dspmq -a` displays the status of active queue managers associated with the installation from which the command is run.

   b) Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

   ```
   echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
   ```

   c) Run the **endmqm** command to stop each running queue manager associated with this installation.

   The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

   For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM MQ resources, such as by stopping the applications.

   You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM MQ. The application is not informed about queue managers in the current installation being shut down.

   Any applications that continue to have IBM MQ shared libraries from the installation loaded prevent you applying IBM MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep an IBM MQ shared library loaded.

Note: The topic, “Applying maintenance level updates to multi-instance queue managers” on page 249, describes how to apply maintenance to a multi-instance queue manager. A multi-instance queue manager can continue to run on one server, while maintenance is applied to another server.

d) Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Back up the queue manager.
5. Stop the MQSeries Service and exit the Service icon application.
6. Modify the IBM MQ installation, if necessary, by using one of these procedures:
   - Modifying the installation using IBM MQ Installation Launchpad
   - Silently modifying an IBM MQ server installation using **`msiexec`**
7. Reenter domain, user ID, and password information

   When the installation of the latest version completes, the **Prepare WebSphere MQ Wizard** starts automatically.

   Where UAC is enabled: If you rerun the **Prepare WebSphere MQ Wizard**, ensure that the wizard is run with Administrator privilege, otherwise the wizard might fail.
8. If you are running IBM WebSphere MQ 7.0.1.6 or later, optionally uninstall the current version of IBM MQ.

   Note, that you carry out this step only if you are doing a single stage migration; see “UNIX, Linux, and Windows: Single-stage migration to a later version” on page 92

What to do next

You might be restoring a previous version on a server with multiple IBM MQ installations. If one of the installations is primary, after restoring the previous version that installation, by default, becomes the primary installation.

You must review how applications connect to an installation. After restoring the previous version, some applications might connect to the wrong installation.

Fix Central

Passport Advantage and Passport Advantage Express web site

“Migrating a queue manager in a high-availability configuration” on page 76
Follow standard procedures to migrate a queue manager that is part of a high-availability configuration. (On platforms other than z/OS.)

“Queue manager cluster migration” on page 74
You can migrate queue managers in a cluster all at once, or one at a time, which is called a staged migration. Migrate full repository queue managers in a cluster before partial repository queue managers.

“Reverting a queue manager to a previous version” on page 70
You can remove an upgrade before you have started a queue manager. After a queue manager has been started, if you remove the upgrade, the queue manager will not work.

IBM MQ - SupportPacs by Product

“Upgrade, migration, and maintenance of IBM MQ (On platforms other than z/OS)” on page 51
You can install new releases of IBM MQ to upgrade IBM MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of IBM MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

“IBM MQ migration” on page 45

“IBM MQ upgrades and fixes” on page 39

The term upgrade applies to changing the version V, release R, or modification M of a product. The term fix applies to a change in the F digit.

IBM i IBM i Migrating a queue manager from a previous release

Follow these instructions to migrate a queue manager from a previous release to the latest release.

Before you begin

If you decide to do a side-by-side installation, you must prepare the new server first, installing the prerequisite software.

1. Create a migration plan. Use the planning task, Planning migration to the latest version, as a guide.
2. Review the IBM MQ system requirements for the latest release of the product; see System Requirements for IBM MQ
3. Review any other installed SupportPacs for their applicability to the latest release of IBM MQ.

About this task

There are various types of migration:

- The migration takes place on the same machine, optionally accompanied by a hardware upgrade. This migration is referred to as a *slip install*.
- The migration takes place on a different machine. This migration is referred to as a *side-by-side install*.

  A side-by-side installation gives you the option of preparing the new environment first, without interrupting the queue manager. It also gives you the limited option of reverting to use the previous release installation, if the migration is unsuccessful. It is limited, because you cannot restore the queue manager data from the latest version. You must restart processing with the queue manager data at the point you stopped the queue manager on the previous release.

If you want to add IBM MQ Advanced Message Security to your system, you must select Option (2) when you install the product; see Installing IBM MQ Advanced Message Security on IBM i for further information.

IBM i Installation methods on IBM i

Select a slip installation or a side-by-side installation to upgrade IBM MQ for IBM i.

About this task

A slip installation upgrades IBM MQ for IBM i on a computer with an earlier version is installed.

A side-by-side installation upgrades IBM MQ for IBM i on a different computer. You must save your queue managers before you start.

Follow the steps in the following tasks to carry out an upgrade.

The steps for both forms of upgrade are identical, except that you do not carry out the actions described in "Restore queue managers after upgrading IBM MQ on IBM i" on page 122 for a slip install.

IBM i End IBM MQ activity on IBM i

End IBM MQ applications and connections, and remove any unwanted or indoubt messages.

About this task

Before performing a slip install or side-by-side install, carry out the following procedure:

Procedure

1. Sign on to the system with a user profile that has *ALLOBJ special authority, for example QSECOFR.
2. Stop all applications that are using the existing version of IBM MQ.

   To identify applications using the queue manager, use the command WRKMQM, option 22, Work with queue manager jobs, to help find them. Ignore jobs starting with AMQ* or RUN* and focus on your application job names.
3. End all channels for all queue managers on the system. To do so, use the WRKMQMCHL command and select option 15.
4. On each queue manager, end the command server. To do so, enter the command:

   ```
   ENDMQMCSVR MQMNAME( QMGRNAME ) OPTION(*IMMED)
   ```

   where *QMGRNAME* is the name of the queue manager.
5. Remove any unwanted messages from your queues.
6. Resolve any in-doubt messages that are held by sender or server channels. To do so, use the WRKMQMCHST command and select option 17.
7. On each queue manager, save the latest media recovery checkpoint. To do so, enter the following command:

   ```
   RCDMQMIMG OBJ(*ALL) OBJTYPE(*ALL) MQMNAME( QMGRNAME ) DSPJRNDTA(*YES)
   ```

IBM i Quiesce IBM MQ on IBM i

Stop all queue managers. If necessary force all queue managers to stop, tidy up shared memory and end all jobs in the QMQM subsystem.

About this task

The orderly shutdown of IBM MQ is called *quiescing*. You need to quiesce IBM MQ to upgrade to a newer version.

To quiesce one or more queue managers:

Procedure

1. Sign on to a new interactive IBM i session, ensuring that you are not accessing any IBM MQ objects.
2. Ensure that you have:

   a) *ALLOBJ authority, or object management authority for the QMQM library.

   b) Sufficient authority to use the ENDSBS command.
3. Warn all users that you are going to stop IBM MQ.
4. Use the ENDMQM command to quiesce all queue managers:

   ```
   ENDMQM MQMNAME(*ALL) OPTION(*CNTRLD) ENDCCTJOB(*YES) RCDMQMIMG(*YES)
   TIMEOUT( 15 )
   ```

   Where *15* is a timeout value in seconds.

   If the ENDMQM command has not completed within a reasonable period (at least 10 minutes), use the WRKMQM command. This command identifies the queue managers that are still ending. Then force each one in turn to stop by issuing:

   ```
   ENDMQM MQMNAME( QMGRNAME ) OPTION(*IMMED)
   ```

Where *QMGRNAME* is the name of the queue manager.

Complete the tidying up of shared memory by issuing the command:

```
ENDMQM MQMNAME(*ALL) OPTION(*IMMED) ENDCCTJOB(*YES) RCDMQMIMG(*NO)
TIMEOUT( 15 )
```

5. If the command in the previous step does not complete, end the subsystem immediately by issuing:

```
ENDSBS SBS(QMQM) OPTION(*IMMED)
```

6. If this command also fails, use the operating system command ENDJOB to end all jobs in the subsystem QMQM, as described in the following steps.

   Note: Do not use ENDJOBABN unless you intend to perform an IPL on the machine before starting IBM MQ. Ending IBM MQ jobs using ENDJOBABN can lead to damaged semaphores, which in turn can prevent your queue manager from starting.

   a) If a QMGR must be shut down manually, the recommended order of ending jobs (ENDJOB) is shown in the list that follows. Wait a few minutes for AMQA* or AMQZ* jobs to tidy up.

      i) `RUNMQLSR` - TCP listener (multi-threaded)

      ii) `AMQCLMAA` - TCP listener (single-threaded)

      iii) `AMQRMPPA` - Channel process pooling job

      iv) `RUNMQCHI` - channel initiator

      v) `AMQCRSTA` - receiving MCA jobs

      vi) `RUNMQCHL` - sending MCA jobs

      vii) `AMQCRS6B` - LU62 receiver channel

      viii) `AMQPCSEA` - command server

      ix) `RUNMQTRM` - Application trigger monitor

      x) `RUNMQDLQ` - Dead letter queue handler

      xi) `AMQFCXBA` - IBM Integration Bus Worker Job

      xii) `AMQFQPUB` - Queued Publish/Subscribe Daemon

      xiii) `RUNMQBRK` - IBM Integration Bus Control Job

      xiv) `AMQZMUC0` ('0' is a zero) - Utility Manager

      xv) `AMQZMUF0` ('0' is a zero) - Utility Manager

      xvi) `AMQZMUR0` ('0' is a zero) - Utility Manager

      xvii) `AMQZMGR0` ('0' is a zero) - Process Controller

      xviii) `AMQRRMFA` - cluster repository manager

      xix) `AMQZDMAA` - deferred message manager

      xx) `AMQALMPX` - Log Manager

      xxi) `AMQZFUMA` - object authority manager

      xxii) `AMQZLSA0` ('0' is a zero) - LQM agents

      xxiii) `AMQZLAA0` ('0' is a zero) - LQM agents

      xxiv) `AMQZXMA0` ('0' is a zero) - Execution Controller

   b) Issue the following command:

   ```
   ENDMQM MQMNAME( QMGRNAME ) OPTION(*IMMED)
   ```

   c) Issue the following command:

```
ENDMQM MQMNAME(*ALL) OPTION(*CNTRLD) ENDCCTJOB(*YES) RCDMQMIMG(*NO)
TIMEOUT( 05 )
```

Where *05* is a timeout value in seconds.

d) Manually clean up shared memory. Issue the following command:

```
EDTF '/QIBM/UserData/mqm/qmgrs'
```

then:

i) Take option 5 for **&SYSTEM** and check that the following directories are empty: `isem, esem, msem, ssem,` and `shmem`.

ii) Take option 5 for **QMGRNAME** and check that the following directories are empty:- `isem, esem, msem, ssem,` and `shmem`.

iii) Take option 5 for **&ipcc** in the QMGRNAME directory and check that the following directories are empty:- `isem, esem, msem, ssem,` and `shmem`.

iv) Take option 5 for **&qmpersist** in the QMGRNAME directory and check that the following directories are empty:- `isem, esem, msem, ssem,` and `shmem`.

v) Take option 5 for **&app** and check that the following directories are empty: `isem, esem, msem, ssem,` and `shmem`.

IBM i Save IBM MQ data on IBM i

Save IBM MQ data after removing unwanted FDC, trace, and JOB files.

Before you begin

You need to have completed the tasks to remove unwanted and indoubt messages and quiesced IBM MQ.

About this task

Procedure

1. Create a save file for every queue manager library on your system. To do so, issue the command:

   ```
   CRTSAVF FILE(QGPL/ queue-manager-library )
   ```

   where the *queue-manager-library* name consists of the name of the queue manager preceded by QM.

2. Save your queue manager libraries into the save files. To do so, issue the commands:

   ```
   SAVLIB LIB( queue-manager-library ) DEV(*SAVF)
   SAVF(QGPL/ queue-manager-library )
   ```

3. Remove all unwanted FDC data from directory:

   ```
   QIBM/UserData/mqm/errors
   ```

4. Remove old FDC files with the command:

   ```
   RMVLNK OBJLNK('/QIBM/UserData/mqm/errors/*.FDC')
   ```

   This command cleans up all files with an extension of 'FDC' in the IFS.

5. Remove old JOB files with the command:

```
RMVLNK OBJLNK('/QIBM/UserData/mqm/errors/*.JOB')
```

This command cleans up all files with an extension of 'JOB' in the IFS.

6. Remove all unwanted trace data from directory, or remove the whole directory:

   ```
   QIBM/UserData/mqm/trace
   ```

7. Remove all trace files with the command:

   ```
   RMVLNK OBJLNK('/qibm/userdata/mqm/trace/*')
   ```

8. Create a save file for IBM MQ IFS data. To do so, issue the command:

   ```
   CRTSAVF FILE(QGPL/QMUSERDATA)
   ```

9. Save your IBM MQ IFS data, using the command:

   ```
   SAV DEV('/QSYS.LIB/QGPL.LIB/QMUSERDATA.FILE') OBJ('/QIBM/UserData/mqm')
   ```

10. If you are going to run IBM MQ on a new machine, transfer the save files to the new machine.

IBM i Install IBM MQ server on IBM i

Install the IBM MQ server in its primary language.

Before you begin

You have completed planning the installation, obtained the installation disks, and set the system values; see Setting system values.

About this task

Install the IBM MQ server and force object conversion. Object conversion migrates objects from the older to the newer version. By performing it now, rather than when an object is first used, you avoid slowing down the first use of the upgraded product.

After following the optional step to pre-agree the license, the **RSTLICPGM** command runs without requiring any interactive input. Otherwise the license agreement is displayed for you to accept. See License requirements.

Procedure

1. Sign on to the system with a user profile that has *ALLOBJ special authority, for example QSECOFR.
2. Optionally pre-agree the license terms and conditions by running the command,

   ```
   CALL PGM ( QSYS/QLPACAGR ) PARM ( '5724H72' 'V8R0M0' '0000' 0 )
   ```

   Where the parameters of **PARM** are,

   5724H72
   The product identifier for IBM MQ for IBM i.

   V8R0M0
   The version, release, and modification level.

   0000
   The option number for the *BASE IBM MQ product option.

0
: Unused error structure.

3. Install IBM MQ for IBM i, base product, and primary language.

   ```
   RSTLICPGM LICPGM ( 5724H72 ) DEV ( install device ) OPTION ( *BASE ) OUTPUT ( *PRINT )
   ```

   where the parameters of RSTLICPGM are,

   LICPGM (5724H72)
   The product identifier for IBM MQ for IBM i.

   DEV (*install device*)
   The device from which the product is to be loaded, typically an optical drive, for example, OPT01.

   OPTION (*BASE)
   Install the base IBM MQ for IBM i product.

   Unspecified parameters
   Unspecified parameters such as **RSTOBJ** (*ALL) , revert to defaults. The command installs both IBM MQ and the language files for the primary language of your system. For installing additional languages see Installing translated versions.

What to do next

Install any Progam Temporary Fixes (PTF) that have been issued.

IBM i Install samples on IBM i

Install the IBM MQ samples

Before you begin

If you have not already done so, sign on to the system with a user profile that has *ALLOBJ special authority, for example QSECOFR.

About this task

Install the samples.

After following the optional step to pre-agree the license, the **RSTLICPGM** command runs without requiring any interactive input. Otherwise the license agreement is displayed for you to accept. See License requirements.

Procedure

1. Optionally pre-agree the license terms and conditions by running the command,

   ```
   CALL PGM ( QSYS/QLPACAGR ) PARM ( '5724H72' 'V8R0M0' '0001' 0 )
   ```

   Where the parameters of **PARM** are,

   5724H72
   The product identifier for IBM MQ for IBM i.

   V8R0M0
   The version, release, and modification level.

   0001
   The option number for the samples.

   0
   Unused error structure.

2. Install the samples using the command:

```
RSTLICPGM LICPGM ( 5724H72 ) DEV ( install device ) OPTION ( 1 ) OUTPUT ( *PRINT )
```

Where the parameters of RSTLICPGM are,

LICPGM (5724H72)
The product identifier for IBM MQ for IBM i.

DEV (*install device*)
The device from which the product is to be loaded, typically an optical drive, for example, OPT01.

OPTION (1)
Install the samples for IBM MQ for IBM i.

OUTPUT (*PRINT)
The output is printed with the spooled output of the job.

IBM i Install translated versions on IBM i

Install translated versions of IBM MQ from a choice of national-languages.

About this task

The following language versions are available for IBM MQ for IBM i:

Table 10. National-language versions of IBM MQ for IBM i

Language ID	Language
2909	Belgian English
2966	Belgian French MNCS (Multi-National Character Set)
2981	Canadian French MNCS
2975	Czech
2950	English uppercase
2924	English uppercase and lowercase
2984	English US DBCS
2938	English US uppercase DBCS
2928	French
2940	French MNCS
2929	German
2939	German MNCS
2976	Hungarian
2932	Italian
2942	Italian MNCS
2962	Japanese
2986	Korean
2978	Polish
2979	Russian
2989	Simplified Chinese

Table 10. National-language versions of IBM MQ for IBM i (continued)

Language ID	**Language**
2931	Spanish

IBM MQ for IBM i is installed in the language that is the primary language on your system.

You can install additional versions of the product in any of the languages shown in Table 10 on page 119. To do so:

Procedure

1. Sign on to the system with a user profile that has *ALLOBJ special authority
2. Issue the following command specifying the appropriate language ID:

   ```
   RSTLICPGM LICPGM(5724H72) DEV( install device ) RSTOBJ(*LNG) LNG( language ID )
   ```

   This installs the commands, message file, and panel groups into the relevant QSYS library for the language. For example, library QSYS2928 is used for French. If this QSYS29nn library does not exist, it is created by the RSTLICPGM command.

Results

Note:

1. To run the Japanese language version of IBM MQ for IBM i, the CCSID of the job must be 939 (5035) rather than 930 (5026) because IBM MQ uses lowercase English characters.
2. If you are installing IBM MQ for IBM i onto a machine for which the primary language is not on the CD, the install program prompts you to load a CD containing the product in that language. If, however, you have only one product CD, this means that the IBM MQ product has not been translated into your language. To get around this issue, proceed as follows:
   - Install the product in one of the supplied languages, and then add the corresponding QSYS29nn library into the *system library* list (for example using command CHGSYSLIBL). At the same time, check that there are no IBM MQ *CMD, *MENU, or *MSGF objects in libraries higher up the library list. If some exist, then either delete these objects (because they refer to an earlier version of IBM MQ) or reorder the System Library list (because the product has been installed in more than one of the supplied languages).

IBM i Verify the installation on IBM i

How to check that your installation has been successful.

Procedure

1. To ensure that the product has loaded correctly, issue the Display Software Resources (DSPSFWRSC) command and check that the licensed program 5724H72 is listed. If you have installed the base and the optional samples, you see:

   ```
   Resource
   ID    Option Feature Description
   5724H72  *BASE  5050 IBM MQ for IBM i
   5724H72  *BASE  2924 IBM MQ for IBM i
   5724H72  1    5050 IBM MQ for IBM i - Samples
   ```

2. Press F11, while viewing the Display Software Resources screen, and you see the library and version number of the products installed:

```
Resource          Feature
ID    Option Feature  Type  Library  Release
5724H72  *BASE  5050   *CODE  QMQM   V8R0M0
5724H72  *BASE  2924   *LNG   QMQM   V8R0M0
5724H72  1    5050   *CODE  QMQMSAMP V8R0M0
```

3. If you have installed additional language versions, you also see entries for these versions. For example, if you have installed the French version, for which the language ID is 2928, you see:

   a)

   ```
   Resource
   ID    Option Feature Description
   5724H72  *BASE  2928 IBM MQ for IBM i
   ```

   b) and when you press F11:

   ```
   Resource           Feature
   ID    Option Feature  Type  Library   Release
   5724H72  *BASE  2928  *LNG  QSYS2928  V8R0M0
   ```

4. Use the command DSPMQMVER to check exactly what version you have installed. For V8R0M0, it reports:

```
Version: 8.0.0.0
```

IBM i Verify the upgrade on IBM i

After you have verified the installation, start the IBM MQ subsystem, check the queue managers, and take a fresh media recovery checkpoint.

About this task

To verify that you have migrated to the latest version IBM MQ for IBM i, successfully:

Procedure

1. Make QMQMADM either the primary or a secondary group profile for your user profile. To do so, issue one of the following commands:

   ```
   CHGUSRPRF USRPRF( YOUR PROFILE ) GRPPRF(QMQMADM)
   CHGUSRPRF USRPRF( YOUR PROFILE ) SUPGRPPRF(QMQMADM)
   ```

2. Start the IBM MQ subsystem with the command:

   ```
   STRSBS SBSD(QMQM/QMQM)
   ```

   (If it is already running, you get error message `CPF1010` which you can safely ignore).

3. Check that your queue managers are accessible by issuing the command:

   ```
   WRKMQM
   ```

   Use option 14 against each queue manager to start it.

   Use option 5 against each queue manager to check its attributes.

4. You can use the other options to check your queue manager objects. For example, check your queues using option 18, check your channels using option 20, and so on.
5. Take a fresh media recovery checkpoint, using the following command:

```
RCDMQMIMG OBJ(*ALL) OBJTYPE(*ALL) MQMNAME( QMGRNAME ) DSPJRNDTA(*YES)
```

Where *QMGRNAME* is the name of the queue manager.

IBM i Restore queue managers after upgrading IBM MQ on IBM i

Complete the side-by-side upgrade by restored the saved queue managers onto the server that you have upgraded.

Before you begin

Note: Carry out this task only if you are performing a side-by-side upgrade.

Ensure that you have saved your queue manager data, see "End IBM MQ activity on IBM i" on page 113, and installed and verified the upgrade.

About this task

Transfer the queue manager data, and journal receivers, onto the server that has been upgraded.

Procedure

1. Restore the queue manager libraries for every queue manager, using the command:

   ```
   RSTLIB SAVLIB( queue-manager-library ) DEV(*SAVF) (*PRINT)
   SAVF(QGPL/ queue-manager-library )
   ```

   where the *queue-manager-library* name consists of the name of the queue manager preceded by QM.
2. Restore the IBM MQ IFS data, using the command:

   ```
   RST DEV('/QSYS.LIB/QGPL.LIB/QMUSERDATA.FILE') OBJ('/QIBM/UserData/mqm') (*PRINT)
   ```

3. To associate the journal receivers, issue the command WRKJRN on the journal AMQAJRN in each queue manager library, by pressing *PF4* and selecting option 9.
4. If you want to set up your work management environment, job descriptions, and pools, see the Administering IBMi for guidance. Otherwise, use the default setup.

IBM i After Upgrading on IBM MQ for IBM i

Tasks to perform after you have upgraded IBM MQ for IBM i.

About this task

Satisfy yourself the upgrade has completed successfully.

Procedure

Delete the saved data in the save files in QGPL. This data was saved in "Save IBM MQ data on IBM i" on page 116.

IBM i Post installation tasks for IBM MQ for IBM i

Tasks to perform after you have installed IBM MQ for IBM i, and before using it.

About this task

When you have correctly installed IBM MQ for IBM i on your system:

Procedure

1. For the latest product information for IBM i, see System requirements for IBM MQ .
2. To install and apply all fix packs, see "IBM i: Applying maintenance level updates on the latest release" on page 223.
3. Where you have more than one system and a mixture of releases of OS/400 or IBM i, and IBM MQ, you must take care when compiling CL programs. You must compile CL programs either on the system they are to run on, or on one with an identical combination of releases of OS/400 or IBM i, and IBM MQ. When you install later versions of IBM MQ, delete all IBM MQ commands from previous releases in any QSYSVvRrMm libraries using the QSYS/DLTCMD command.
4. If you have not installed IBM MQ on your system before, you must add user profiles to the `QMQMADM` group profile. Make all user profiles that are to be used for creating and administering queue managers members of the `QMQMADM` group profile, using the command `CHGUSRPRF`.

    a) Start the IBM MQ subsystem, by issuing the command:

    ```
    STRSBS SBSD(QMQM/QMQM)
    ```

    Note: The subsystem must be started after each IPL of the system, so you might choose to start it as part of your system startup process.

5. Create the system-default objects. The system-default objects are created automatically when you issue the CRTMQM command to create a queue manager. For example: `CRTMQM MQMNAME(QMGRNAME) ASP(*SYSTEM)`. You can refresh them using the STRMQM command (Warning: this command will replace any existing default objects). For example: `STRMQM MQMNAME(QMGRNAME) RDEFSYS(*YES)`. Refer to the onscreen help for information about using this command.

    Note: on the command `STRMQM MQMNAME(QMGRNAME) RDEFSYS(*YES)`:

    - The command does not recreate the objects, it performs a CRTxxxx REPLACE(*YES) for all of the SYSTEM.* objects.
    - This means that it refreshes the parameters on the objects back to their defaults. So if, for example, on the SYSTEM.DEFAULT.LOCAL.QUEUE object, TRGENBL had previously been changed to *YES, then, when the command is run, it is changed back to TRGENBL(*NO).
    - If any messages exist on a queue, they are left intact, because the queues are not physically deleted.
    - The contents of the SYSTEM.AUTH.DATA.QUEUE are untouched when this command is run.
    - So, if the contents of this (or any other significant queue) become corrupt, it must be physically deleted and recreated either from scratch, or from a backup.

Results

You are now ready to start using IBM MQ for IBM i.

Note: When you install IBM MQ for IBM i, two user profiles are created:

- QMQM
- QMQMADM

These two objects are central to the correct running of IBM MQ for IBM i. Do not alter or delete them. If you do, IBM cannot guarantee correct behavior of your product.

If you uninstall IBM MQ and data, these profiles are deleted. If you uninstall IBM MQ only, these profiles are retained.

IBM i IBM i: migrating a queue manager from a previous release - alternative method

An alternative method of migrating a queue manager from a previous release to the latest release.

Before you begin

1. Review the IBM MQ system requirements for the latest release of the product; see System Requirements for IBM MQ
2. Review any other installed SupportPacs for their applicability to the latest release of IBM MQ.

About this task

There are various parts to this form of migration:

1. As part of upgrading the IBM MQ product, carry out the following tasks:
   a. "Preparing to install IBM MQ on IBM i" on page 124
   b. "Install IBM MQ server on IBM i" on page 125
2. Following the IBM MQ product upgrade, carry out the following task:
   a. "Post installation tasks" on page 126

IBM i Preparing to install IBM MQ on IBM i

Carry out the following tasks to prepare your system for an upgrade.

Procedure

1. Stop the IBM MQ queue managers by issuing the following command:

   ```
   ENDMQM MQMNAME(*ALL) OPTION(*IMMED) ENDCCTJOB(*YES) RCDMQMIMG(*YES)
   TIMEOUT(30)
   ```

   Ensure that the user profile issuing this command has *ALLOBJ authority.
2. Create a save file for every queue manager library on your system. To do so, issue the command:

   ```
   CRTSAVF FILE(QGPL/ queue-manager-library )
   ```

   where the *queue-manager-library* name consists of the name of the queue manager preceded by QM.
3. Save your queue manager libraries into the save files. To do so, issue the commands:

   ```
   SAVLIB LIB( queue-manager-library ) DEV(*SAVF)
   SAVF(QGPL/ queue-manager-library )
   ```

4. Create a save file for IBM MQ IFS data. To do so, issue the command:

   ```
   CRTSAVF FILE(QGPL/QMUSERDATA)
   ```

5. Save your IBM MQ IFS data, using the command:

```
SAV DEV('/QSYS.LIB/QGPL.LIB/QMUSERDATA.FILE') OBJ('/QIBM/UserData/mqm')
```

6. If you are going to run IBM MQ on a new machine, transfer the save files to the new machine.
7. Issue the following command before you upgrade your IBM MQ product, only if the upgrade is required on the same machine.

   a) `DLTMQM <QMGRNAME>`

   b) `ENDSBS SBS(QMQM) OPTION(*IMMED)`

   c) `WRKOBJLCK OBJ(QMQM) OBJTYPE(*LIB)`

   Relinquish any locks on the system.

IBM i Install IBM MQ server on IBM i

Install the IBM MQ server in its primary language and force object conversion.

Before you begin

In either of the following cases, ensure that you have completed the planning and set the system values; see Setting system values

- If you have obtained the product through the Passport Advantage and Passport Advantage Express web site, follow the instructions in the `EGA.README.txt` file.
- If you have obtained the product on disk, follow the instructions within this topic.

About this task

Install the IBM MQ server and force object conversion. Object conversion migrates objects from the older to the newer version. By performing it now, rather than when an object is first used, you avoid slowing down the first use of the upgraded product.

After following the optional step to pre-agree the license, the **RSTLICPGM** command runs without requiring any interactive input. Otherwise the license agreement is displayed for you to accept. See License requirements.

Procedure

1. Sign on to the system with a user profile that has `*ALLOBJ` special authority, for example `QSECOFR`.
2. Optionally pre-agree the license terms and conditions by running the command,

   ```
   CALL PGM ( QSYS/QLPACAGR ) PARM ( '5724H72' 'V8R0M0' '0000' 0 )
   ```

   Where the parameters of **PARM** are,

   5724H72

   The product identifier for IBM MQ for IBM i.

   V8R0M0

   The version, release, and modification level.

   0000

   The option number for the *BASE IBM MQ product option.

   0

   Unused error structure.
3. Install IBM MQ for IBM i, base product, and primary language.

   ```
   RSTLICPGM LICPGM ( 5724H72 ) DEV ( install device ) OPTION ( *BASE ) OUTPUT ( *PRINT )
   ```

   where the parameters of RSTLICPGM are,

LICPGM (5724H72)
The product identifier for IBM MQ for IBM i.

DEV (*install device*)
The device from which the product is to be loaded, typically an optical drive, for example, `OPT01`.

OPTION (*BASE)
Install the base IBM MQ for IBM i product.

Unspecified parameters
Unspecified parameters such as **RSTOBJ** `( *ALL )` , revert to defaults. The command installs both IBM MQ and the language files for the primary language of your system. For installing additional languages see Installing translated versions.

What to do next

Install any Progam Temporary Fixes (PTF) that have been issued.

To install the IBM MQ samples, see: “Install samples on IBM i” on page 118.

Post installation tasks

Actions required after upgrading IBM MQ.

About this task

Install the samples.

Carry out these steps after installing the product.

Procedure

1. Issue the following commands:
   a) `STRSBS SBSD(QMQM/QMQM)`
   b) `CRTMQM MQMNAME(<QMGRNAME>) DFTQMGR(*YES)`
      You receive the message " IBM MQ queue manager created."
   c) `STRMQM MQMNAME(<QMGRNAME>)`
      You receive the message " IBM MQ queue manager `'<QMGRNAME>'` started."
2. Issue the following command:

```
STRMQMMQSC SRCMBR(<QMGRNAME>) SRCFILE(*CURLIB/QMQSC) OPTION(*RUN)
MQMNAME(<QMGRNAME>)
```

3. Reapply IBM MQ Authorities by issuing the command: `CALL PGM(*CURLIB/<QMGRNAME>)`
   a) You must compile the CLP as follows:

```
CRTCLPGM PGM(*CURLIB/<QMGRNAME>) SRCFILE(*CURLIB/QMAUT) SRCMBR(*PGM)
```

Restoring a queue manager from the latest version to a previous version on UNIX systems and Windows

You can restore a queue manager to the previous version of the product from the latest version, if you have made a backup of the system or queue manager. If you have started the queue manager and processed any messages, or changed the configuration, the task cannot give you any guidance on restoring the current state of the queue manager.

Before you begin

1. You must have made a backup of the system or queue manager before you upgraded to the latest version. For more information see Backing up and restoring IBM MQ queue manager data
2. If any messages were processed after starting the queue manager, you cannot easily undo the effects of processing the messages. You cannot restore the queue manager to the previous version of the product in its current state. The task cannot give you any guidance how to deal with subsequent changes that have occurred. For example, messages that were indoubt in a channel, or in a transmission queue on another queue manager, might have been processed. If the queue manager is part of a cluster, then configuration messages and application messages might have been exchanged.
3. If you are running on a server with multiple IBM MQ installations, you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

About this task

When you restore a previous version of a queue manager, you restore the queue manager to its earlier code level. Queue manager data is restored to the state it was in when the queue manager was backed up.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM MQ installation.

   If you use the MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM MQ installation.

   a) Run the **dspmq** command to list the state of all the queue managers on the system.

   Run either of the following commands from the installation that you are updating:

   ```
   dspmq -o installation -o status
   dspmq -a
   ```

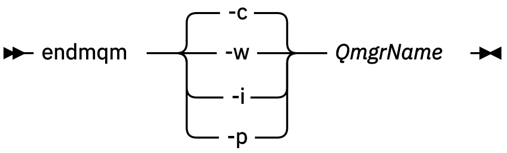
   `dspmq -o installation -o status` displays the installation name and status of queue managers associated with all installations of IBM MQ.

   `dspmq -a` displays the status of active queue managers associated with the installation from which the command is run.

   b) Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

   ```
   echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
   ```

   c) Run the **endmqm** command to stop each running queue manager associated with this installation.



   The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM MQ shared libraries from the installation loaded prevent you applying IBM MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep an IBM MQ shared library loaded.

Note: The topic, "Applying maintenance level updates to multi-instance queue managers" on page 249, describes how to apply maintenance to a multi-instance queue manager. A multi-instance queue manager can continue to run on one server, while maintenance is applied to another server.

d) Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Restore the system, or IBM MQ and the queue manager.

   If your backup procedure was to save the queue manager data, you must reinstall IBM MQ.

   a) Uninstall the previous installation.
   b) Reinstall IBM MQ from a manufacturing refresh.
   c) Apply the fix pack and interim fixes that restore IBM MQ to its previous level.
   d) Restore the queue manager data from the backup taken before installing the latest version.
5. Restart the previous version queue manager.

What to do next

You might be restoring a previous version on a server with multiple IBM MQ installations. If one of the installations is primary, after restoring the previous version that installation, by default, becomes the primary installation.

You must review how applications connect to an installation. After restoring the previous version, some applications might connect to the wrong installation.

Related information

Backing up and restoring a queue manager

Linux: Cleaning up after using the `rpm` freshen or upgrade options

The use of **`rpm`** upgrade or freshen options is not supported. If you use the options, follow this cleanup procedure, and then install following the correct steps.

Before you begin

You have attempted to upgrade IBM MQ for Linux using `rpm -U` or `rpm -F`

About this task

By using the freshen or upgrade options, you might have deleted your old IBM MQ package entries from the **`rpm`** database without removing the product from your system. You might also have partially installed IBM MQ

Procedure

Follow these steps to clean up your system.

1. Find out which IBM MQ MQ packages still have entries in your RPM database.

   ```
   rpm -qa | grep MQSeries
   ```

2. Remove all remaining IBM MQ packages from your system.

   ```
   rpm -e package-name
   ```

3. Remove the /opt/mqm directory.

   ```
   rm -rf /opt/mqm
   ```

Migrating an IBM MQ MQI client to a later version of the product

Migrate an IBM MQ MQI client to a later version of the product by completing the tasks in the following topics. Check IBM MQ MQI client applications with the latest version before migration. You must stop all IBM MQ activity on the workstation, before upgrading the IBM MQ MQI client. After upgrading the IBM MQ MQI client, you must check the client channel configuration.

Migrating an IBM MQ MQI client on UNIX systems, Windows, and IBM i to the latest version

Before migrating an IBM MQ MQI client, create a migration plan. Stop all IBM MQ activity on the client workstation. Upgrade the IBM MQ MQI client installation. Make any essential configuration and application changes.

Before you begin

1. Create a migration plan. Use the planning task, "IBM MQ migration planning to the latest version on UNIX platforms, Windows, and IBM i" on page 80, as a guide.

Procedure

1. Review the IBM MQ system requirements for Version 8.0.

   See System Requirements for IBM MQ.
2. Review all the changes in IBM MQ that affect you.

   See What's changed in IBM MQ 8.0.
3. End all IBM MQ activity on the workstation.
4. Upgrade the client.

   Select the appropriate platform that your enterprise uses.

   - To upgrade an IBM MQ MQI client for AIX installation on a workstation; see Client installation procedure on an AIX workstation.
   - To upgrade an IBM MQ MQI client for AIX installation on an AIX IBM MQ server; see Installing an IBM MQ MQI client on the same computer as the server.
   - To upgrade an IBM MQ MQI client for HP-UX installation on a workstation; see Client installation procedure on an HP-UX workstation.
   - To upgrade an IBM MQ MQI client for HP-UX installation on an HP-UX IBM MQ server; see Installing an IBM MQ MQI client on the same computer as the server.
   - To upgrade an IBM MQ MQI client for Linux installation on a workstation; see Client installation procedure on an Linux workstation.

- To upgrade an IBM MQ MQI client for Linux installation on an Linux IBM MQ server; see Installing an IBM MQ MQI client on the same computer as the server.
- To upgrade an IBM MQ MQI client for Solaris installation on a workstation; see Client installation procedure on an Solaris workstation.
- To upgrade an IBM MQ MQI client for Solaris installation on an Solaris IBM MQ server; see Installing an IBM MQ MQI client on the same computer as the server.
- To upgrade an IBM MQ MQI client for Windows installation on a workstation; see Client installation procedure on an Windows workstation.
- To upgrade an IBM MQ MQI client for Windows installation on an Windows IBM MQ server; see Installing an IBM MQ MQI client on the same computer as the server.
- To upgrade an IBM MQ MQI client for IBM i installation on a workstation; see Client installation procedure on IBM i .

What to do next

Complete the tasks in your migration plan, such as verifying IBM MQ MQI client applications work correctly with the latest version.

Related tasks

"IBM MQ migration planning to the latest version on UNIX platforms, Windows, and IBM i" on page 80

Related information

Client installation procedure on an AIX workstation
Client installation procedure on an HP-UX workstation
Client installation procedure on a Linux workstation
Client installation procedure on a Solaris workstation
Client installation procedure on a Windcws workstation
Client installation procedure on IBM i
Installing IBM MQ MQI clients on the same machine as the server

Restoring an IBM MQ MQI client and client connection to the previous version

If you restore an IBM MQ MQI client from a later version of the product to an earlier version of the product, you must undo the configuration changes manually.

About this task

It is unusual to restore earlier IBM MQ MQI client libraries to a workstation. The principal tasks are listed in the following steps.

Procedure

1. End all IBM MQ activity on the workstation.
2. Uninstall the later version of the IBM MQ MQI client code.
3. Follow the client installation procedure for the platform to install the earlier version of the IBM MQ MQI client code.
4. If you configured a Client Connection Definition Table (CCDT) for a queue manager on a later version of the product, revert to using a table created by a queue manager on the earlier version.

   The CCDT must always be created by a queue manager on the same, or earlier, release to the client.

Migrating applications to a later version of the product

IBM MQ applications might require migration between Version 7.1 and the later version.

About this task

For more information see Connection authentication: Configuration.

For a new IBM MQ 8.0 installation, the **CONNAUTH CHCKLOCL** attribute will be set to OPTIONAL. This means that user IDs and passwords are not required, but if they are provided they must be a valid pair, or they will be rejected.

When you are migrating between IBM WebSphere MQ 7.1 and a later version, the **CONNAUTH CHCKLOCL** attribute on each queue manager is set to NONE, ensuring version to version continuity, but switching connection authentication off.

Migrating IBM MQ library loading from an earlier version of the product to the latest version

No change in the way IBM MQ libraries are loaded is normally required if you upgrade from an earlier version of the product to the latest version. You must have followed the instructions on building IBM MQ applications in Version 7.0.1 and you must replace IBM WebSphere MQ 7.0.1 with the latest version of the product. If you choose to take advantage of multi-installation in the latest version of the product, based on the side-by-side or multi-stage migration scenarios, you must modify the environment for the operating system to resolve IBM MQ dependencies for an application. Typically, you can modify the runtime environment, rather than relink the application.

Before you begin

To migrate applications from an earlier version of the product to the latest version, you must know how the operating system loads a IBM MQ library for an application. Is the load path fixed by the application, and can you set the path in an environment variable? It is not essential to know the name of the IBM MQ library that the application loads. The library name does not change from an earlier version of the product to the latest version, although the contents of the library do.

About this task

To migrate an application from an earlier version of the product to the latest version, you do not have to recompile or relink the application, because the IBM MQ libraries are compatible with later versions; see "Application compatibility and interoperability with later versions of IBM MQ" on page 63. You might have to configure the runtime environment differently, for the operating system to load the latest version of the IBM MQ library. If you replaced an earlier version of the product with the latestversion, following the "Single-stage" approach; you do not need to do anything at all; see "UNIX, Linux, and Windows: Single-stage migration to a later version" on page 92 [11] .

The latest version of the product provides two commands to assist you in configuring the runtime environment, **setmqinst** and **setmqenv**. **setmqinst** sets the primary installation; see setmqinst. **setmqenv** initializes the command environment by setting environment variables; see setmqenv.

Procedure

How you configure the runtime environment depends on a number of factors, some of which apply to your configuration. Consider which of the following questions apply to your configuration.

1. Did you follow the build procedure documented in the product documentation for the earlier version of the product?

[11] If you changed the location of the libraries from the earlier version, or created symbolic links to the libraries, this statement might not hold true.

You might be following a different build procedure tailored to your development environment, or adapted from a development tool such as Microsoft Visual Studio.

2. How did you specify the load path for the earlier version?
3. Is the application is loaded by another environment, such as Eclipse, or an application server?

   You must modify the parameters that govern how the parent environment loads applications, not the way the parent environment is loaded.
4. Is the configuration for Windows, or UNIX and Linux ?

   On Windows, the functions performed by an application might require that the queue manager it connects to is associated with the primary installation.
5. What constraints and requirements do you have on how the load path is specified in the latest version?

   Security rules might restrict the use of `LD_LIBRARY_PATH`.
6. Is the latest version of the product installed alongside the earlier version?

   If Version 7.0.1 is installed:

   - You cannot make a later installation primary.
   - You cannot install the later version in the default installation path, that was referenced by applications in Version 7.0.1.

What to do next

Environment configuration on the Windows, and UNIX platforms is a little different. If you have followed the documented build procedure in the earlier version, look at "Windows: Migrating IBM MQ library loading from Version 7.0.1, or later, to the latest version" on page 133 or "UNIX: Migrating IBM MQ library loading from Version 7.0.1, or later, to the latest version" on page 136. These topics show the effects of using setmqinst and setmqenv commands to configure the operating system environment for the three migration scenarios listed in Related information.

Related tasks

"UNIX, Linux, and Windows: Single-stage migration to a later version" on page 92
Single-stage migration, is the term used to describe replacing the only installation of IBM MQ on a server, with a later release. Single stage migration is also known as `upgrading in place` or `in place upgrade` . Until Version 7.0.1.6, single-stage was the only migration scenario. Single-stage migration preserves existing scripts and procedures for running IBM MQ the most. With other migration scenarios you might change some scripts and procedures, but you can reduce the effect queue manager migration has on users.

"UNIX, Linux, and Windows: Side-by-side migration to a later version" on page 96

"UNIX, Linux, and Windows: Multi-stage migration to a later version" on page 100

"UNIX: Migrating IBM MQ library loading from Version 7.0.1, or later, to the latest version" on page 136
Investigate whether applications connecting to the latest version of the product are linked to, and load libraries from, the correct installation.

"Windows: Migrating IBM MQ library loading from Version 7.0.1, or later, to the latest version" on page 133
Investigate whether applications connecting to the latest version of the product are linked to, and load libraries from, the correct installation.

Related reference

"Coexistence" on page 174
Queue managers, with different names, can coexist on any server as long as they use the same IBM MQ installation. On z/OS z/OS, UNIX, Linux, and Windows, different queue managers can coexist on the same server and be associated with different installations. In addition to queue managers coexisting on a server, objects, and commands must work correctly with different queue managers running at different command levels.

Related information

Changing the primary installation

Connecting applications in a multiple installation environment

setmqenv

setmqinst

setmqm

Windows: Migrating IBM MQ library loading from Version 7.0.1, or later, to the latest version

Investigate whether applications connecting to the latest version of the product are linked to, and load libraries from, the correct installation.

Before you begin

Read “Multi-installation queue manager coexistence on UNIX, Linux, and Windows” on page 57 and “Migrating IBM MQ library loading from an earlier version of the product to the latest version” on page 131 before starting this task.

Plan and install the latest version of IBM MQ for Windows, and remember the installation name and whether the installation was set to primary.

About this task

Windows searches numerous directories for load libraries, called DLL s; see Dynamic-Link Library Search Order.

The build procedure documented for IBM WebSphere MQ 7.0.1 applications is to place the IBM MQ libraries to load before any other product libraries in the **`cl`** command. The IBM MQ `.lib` libraries must be in the PATH environment variable you have specified at build time, and the DLL libraries at run time. The PATH variable is used by the application process to find the libraries it must load. If you have followed this build procedure, then the effect of installing the latest version of the product on the libraries that are loaded depends on the migration scenario; see Table 11 on page 133.

Table 11. Windows configurations

Action	Scenario	Latest version replaces earlier version in the same location "Single-stage"	Latest version replaces earlier version in a different location "Side-by-side"	Latest version alongside earlier version "Multi-stage"
`setmqinst`		**`setmqinst`** makes the latest version installation primary. The global PATH is changed to point to the latest version library and all Windows features work with the latest version See note .		No. The latest version installation can be primary, because an earlier version is installed.

Table 11. Windows configurations (continued)

Action Scenario	Latest version replaces earlier version in the same location "Single-stage"	Latest version replaces earlier version in a different location "Side-by-side"	Latest version alongside earlier version "Multi-stage"
No other configuration actions	Library loading works correctly. The global PATH contains the location of the latest version libraries. Even if the latest version installation is not primary, library loading works correctly. The latest version libraries are in the same location as the earlier version libraries were.	Library loading probably works correctly. The library loading might not work, if the application process locally modified the PATH to reference the location of the earlier version libraries. A local setting of PATH might override the global PATH that is set by **setmqinst**.	The library loading continues to work with the earlier version correctly, nothing works with the latest version.
setmqenv	Library loading works correctly. **setmqenv** sets the local PATH correctly.		Library loading works correctly, both for the earlier version and the latest version. **setmqenv** sets the local PATH correctly for the latest version. But the Windows features that depend on the global path do not work correctly with the latest version See note . The correct earlier version is loaded, because the latest version library loads the earlier version library for queue managers that have not been migrated from the earlier version.

Procedure

Identify the installation of the latest version of the product, from which the operating system is going to load IBM MQ libraries:

- If you have a multiple installations of the latest versions to load from a server, IBM MQ checks that the installation the library was loaded from is the installation that is associated with any queue manager the application calls. IBM MQ loads the correct library if the wrong library is loaded. It is necessary to configure only one runtime environment for all IBM MQ applications.
- A typical choice is set the primary installation. Setting an installation to be primary places its library path in the global PATH variable.
- If you upgraded an earlier version installation to the latest version, a link path to the earlier version installation now points to an installation containing the latest version. Applications that have a fixed linkage path to the earlier version installation now load the libraries for the latest installation. They are then switched to the installation that is associated with any queue manager they connect to.
- If you rebuild an application, it must link to an installation of the latest version.

- If an application uses COM or ActiveX it can connect to any queue manager as long as there is a primary installation and it is Version 7.1 or later.

  Note: If an earlier version of the product is installed, COM or ActiveX server applications connect to queue managers associated only with the Version 7.0.1 installation. COM or ActiveX client applications are not affected by the limitation.
- If you are running the IBM MQ.NET monitor in transactional mode, the queue manager it connects to must be the primary installation.

What to do next

If you add further installations of the latest version of the product, you must decide which installation to make primary, if you have chosen to make any primary. As long as applications load IBM MQ libraries from one of the latest version installations, such as the primary installation, they can connect to queue managers associated with any other latest version installation.

On Windows, you might build applications with different development tools. You must identify the property of the development tool that sets the PATH of the application that is being built, and not the properties of the tool itself. For example, if you are debugging with Microsoft Visual Studio, you can insert a call to **`setmqenv`** in the **`Environment`** property of the debugging section of the **`Configuration`** properties of a project.

A Windows application might call `LoadLibrary` and specify an explicit load path. You might build a side-by-side assembly and configure an explicit load path. If an application uses either of these mechanisms, and the latest version IBM MQ library is not on the same path as the earlier release, you must recompile, or configure and relink your application to load the latest version libraries.

Related tasks

"UNIX, Linux, and Windows: Single-stage migration to a later version" on page 92
Single-stage migration, is the term used to describe replacing the only installation of IBM MQ on a server, with a later release. Single stage migration is also known as `upgrading in place` or `in place upgrade` . Until Version 7.0.1.6, single-stage was the only migration scenario. Single-stage migration preserves existing scripts and procedures for running IBM MQ the most. With other migration scenarios you might change some scripts and procedures, but you can reduce the effect queue manager migration has on users.

"UNIX, Linux, and Windows: Side-by-side migration to a later version" on page 96

"UNIX, Linux, and Windows: Multi-stage migration to a later version" on page 100

Related reference

"Coexistence" on page 174
Queue managers, with different names, can coexist on any server as long as they use the same IBM MQ installation. On z/OS z/OS, UNIX, Linux, and Windows, different queue managers can coexist on the same server and be associated with different installations. In addition to queue managers coexisting on a server, objects, and commands must work correctly with different queue managers running at different command levels.

Related information

Changing the primary installation

Connecting applications in a multiple installation environment

setmqenv

setmqinst

setmqm

Features that can be used only with the primary installation on Windows

UNIX: Migrating IBM MQ library loading from Version 7.0.1, or later, to the latest version

Investigate whether applications connecting to the latest version of the product are linked to, and load libraries from, the correct installation.

Before you begin

Read "Multi-installation queue manager coexistence on UNIX, Linux, and Windows" on page 57 and "Migrating IBM MQ library loading from an earlier version of the product to the latest version" on page 131 before starting this task.

Plan and install the latest version of IBM MQ for Windows, and remember the installation name and whether the installation was set to primary.

About this task

In Version 7.0.1, the documented build procedure for IBM MQ applications is to include an explicit library path to the location of the IBM MQ libraries, and to `/usr/lib`, in the link step of the compiler; see Figure 15 on page 136. The same build procedure is documented for the latest version of the product.

```
gcc -m32 -o amqsput_32_r amqsput0.c -I/opt/mqm/inc -L/opt/mqm/lib
-Wl,-rpath=/opt/mqm/lib -Wl,-rpath=/usr/lib -lmqm_r -lpthread
```

Figure 15. Linux C server application, 32 bit, threaded compile and link Version 7.0.1

The documented build step for other UNIX platforms is similar. The examples in Table 12 on page 136 are all based on Linux.

If you have followed this build procedure, then the effect of installing the latest version on library loading depends on the migration scenario; see Table 12 on page 136:

Table 12. UNIX and Linux configurations

Action	Scenario	Latest version replaces earlier version in the same location "Single-stage"	Latest version replaces earlier version in a different location "Side-by-side"	Latest version alongside earlier version "Multi-stage"
setmqinst		**setmqinst** makes the latest version installation primary. Symbolic links to the IBM MQ link libraries are inserted into `/usr/lib`.		No. The latest version installation can be primary, because an earlier version is installed.
No other configuration actions		Library loading works correctly. Library loading works, even without the latest version installation being made primary, because the libraries are installed in `/opt/mqm/lib` and the application was built with the link option, `-rpath=/opt/mqm/lib`	Library loading works correctly. Library loading works, because the installation is primary, and the application was built with the link option, `-rpath=/usr/lib`.	The library loading continues to work with the earlier version correctly, nothing works with the latest version.

Table 12. UNIX and Linux configurations (continued)

Action	Scenario	Latest version replaces earlier version in the same location "Single-stage"	Latest version replaces earlier version in a different location "Side-by-side"	Latest version alongside earlier version "Multi-stage"
`setmqenv`, without setting the `-k` or `-l` options.		Library loading works correctly. **`setmqenv`** is unnecessary. Library loading works, because the libraries are installed in `/opt/mqm/lib` and the application was built with the link option, `-rpath=/opt/mqm/lib`.	Library loading works correctly. **`setmqenv`** is unnecessary. Library loading works, because the installation is primary, and the application was built with the link option, `-rpath=/usr/lib`.	The library loading continues to work with the earlier version correctly, nothing works with the latest version.
`setmqenv`, with the `-k` or `-l` options set.		Library loading works correctly.		Library loading works correctly, both for the earlier version and the latest version. The correct earlier version is loaded, because the latest version library loads the earlier version library for queue managers that have not been migrated from the earlier version.
		The operating system finds the IBM MQ library location set by **`setmqenv`**. **`setmqenv`** adds the location to `LD_LIBRARY_PATH`[12]. `LD_LIBRARY_PATH` is searched before paths set in the application or paths in the default search path. Not all applications can load a library using `LD_LIBRARY_PATH`. In which case the application works only if the library location is `/opt/mqm/lib` or `/usr/lib`		

Procedure

Identify the installation of the latest version of the product, from which the operating system is going to load IBM MQ libraries:

- If you have a multiple installations of the latest versions to load from a server, IBM MQ checks that the installation the library was loaded from is the installation that is associated with any queue manager the application calls. IBM MQ loads the correct library if the wrong library is loaded. It is necessary to configure only one runtime environment for all IBM MQ applications.
- A typical choice is to set the primary installation. Setting an installation to be primary places symbolic links to the IBM MQ libraries in `/usr/lib`. Applications built following the Version 7.0 instructions have an explicit link to `/usr/lib`. `/usr/lib` is also normally in the default library search path.
- If you upgraded an earlier version installation to the latest version, a link path to the earlier version installation now points to an installation containing the latest version. Applications that have a fixed linkage path to the earlier version installation now load the libraries for the latest installation. They are then switched to the installation that is associated with any queue manager they connect to.
- If you rebuild an application, it must link to an installation of the latest version.

[12] `LIBPATH` on AIX. On HP-UX `LD_LIBRARY_PATH` is set, not `SHLIB_PATH`.

- If you set LD_LIBRARY_PATH, or LIBPATH on AIX, you must check that the application is able to use LD_LIBRARY_PATH. setuid or setgid, applications, or applications built in other ways, might ignore LD_LIBRARY_PATH for security reasons.

What to do next

If you add further installations of the latest version of the product, you must decide which installation to make primary, if you have chosen to make any primary. As long as applications load IBM MQ libraries from one of the latest version installations, such as the primary installation, they can connect to queue managers associated with any other latest version installation.

Related tasks

"UNIX, Linux, and Windows: Single-stage migration to a later version" on page 92
Single-stage migration, is the term used to describe replacing the only installation of IBM MQ on a server, with a later release. Single stage migration is also known as `upgrading in place` or `in place upgrade` . Until Version 7.0.1.6, single-stage was the only migration scenario. Single-stage migration preserves existing scripts and procedures for running IBM MQ the most. With other migration scenarios you might change some scripts and procedures, but you can reduce the effect queue manager migration has on users.

"UNIX, Linux, and Windows: Side-by-side migration to a later version" on page 96

"UNIX, Linux, and Windows: Multi-stage migration to a later version" on page 100

Related reference

"Coexistence" on page 174
Queue managers, with different names, can coexist on any server as long as they use the same IBM MQ installation. On z/OS z/OS, UNIX, Linux, and Windows, different queue managers can coexist on the same server and be associated with different installations. In addition to queue managers coexisting on a server, objects, and commands must work correctly with different queue managers running at different command levels.

Related information

External library and control command links to primary installation on UNIX and Linux

Connecting applications in a multiple installation environment

Changing the primary installation

setmqenv

setmqinst

setmqm

Loading IBM WebSphere MQ 7.1 libraries

Linux: Rebuilding a C++ application

C++ IBM MQ MQI client and server applications on Linux must be recompiled using GNU Compiler Collection (GCC) 4.1.2, or later. Compilers older than GCC 4.1.2 are no longer supported. The C++ GCC 4.1.2 run time libraries, or later, must be installed in `/usr/lib` or `/usr/lib64`

If you are using one of the supported Linux distributions, the libraries are correctly installed; see System Requirements for IBM MQ.

The GCC 4.1.2 libraries support SSL and TLS connections from an IBM MQ MQI client. SSL and TLS use GSKit version 8, which depends on `libstdc++.so.6`. `libstdc++.so.6` is included in GCC 4.1.2.

Before you begin

1. Check the required level of GCC for your distribution of Linux ; see System Requirements for IBM MQ.
2. If you are using SSL or TLS, also check the required level of `libstdc++.so`.

3. Check whether the application requires rebuilding. Run the following command to display what version of `libstdc++.so` the application depends upon. If the result is less than `libstdc++.so.6`, you must rebuild your application.

   ```
   ldd ApplicationPath
   ```

About this task

The task describes the steps required to rebuild a Linux C++ IBM MQ application. For more detailed instructions about building Linux applications for IBM MQ ; see Building your procedural application on Linux

Procedure

1. Check that the required GCC library is installed correctly.

   Run one of the following commands:

   - Check the 32 bit library on an x86 Linux system:

     ```
     ls -l /usr/lib/libstdc++.so.6
     ```

   - Check the 64 bit library on any other Linux system.

     ```
     ls -l /usr/lib64/libstdc++.so.6
     ```

2. Check that the GCC compiler is at least at version 4.1.2

   Run the following command to display the version of GCC.

   ```
   gcc -v
   ```

3. Rebuild the application

   The commands to compile and link Linux C++ applications are described in Building 32-bit applications and Building 64-bit applications

What to do next

When you deploy your Linux C++ application, ensure that the same GCC runtime library is correctly installed on the run time system.

Migrating IBM MQ for z/OS - order of tasks

Perform these instructions, in the order shown, to migrate a single IBM MQ for z/OS queue manager.

About this task

The tables within this topic show the tasks required in each part of the process to migrate IBM MQ for z/OS, and the order in which these tasks must be done.

Notes:

- You must perform the tasks in the following order:

  1. Before migration
  2. Migrating to the next release
  3. Post-migration tasks

  and the order of tasks within each table.

Table 13. Before migration

Task	For your own use
1. Make your configuration ready for migration	
2. Install the new version code	
3. Perform backup	
4. Restart IBM MQ systems	
5. Review pageset 0 usage	
6. Migrate Db2 tables - for each QSG	
7. Add new CF structure definition - for each QSG	
8. Server application migration	
9. Preparing to migrate Advanced Message Security	

Table 14. Migrating to the next release

Task	For your own use
10. Stop or disconnect all your applications	
11. Stop the queue manager	
12. Update STEPLIB for MSTR and channel initiator	
13. Update the initialization input data sets	
14. Update the target version system parameter module	
15. Migrating Advanced Message Security	
16. Review the security control of your system	
17. Start the queue manager	
18. Optionally, revert the queue manager to a previous version	

Table 15. Post migration tasks

Task	For your own use
19. Check the changes in behavior	
20. Modify the backup jobs	
21. Post migration tasks for Advanced Message Security	
22. Perform a full regression test	
23. Update the ZPARM module, if not already done	
24. Set OPMODE to NEWFUNC	
25. Exploit the new function	
26. Consider client application migration	

z/OS: new messages in IBM MQ for z/OS Version 8

Read this information to see the list of new messages in IBM MQ for z/OS Version 8

About this task

Recovery log manager messages

CSQJ164I *csect-name* Log archiving delayed, all available offload tasks in use

CSQJ168I *csect-name* Log archiving is no longer delayed

Message manager messages

CSQM079I *csect-name* Policy access attempt rejected due to incompatible AMS version, jobname *jobname*

CSQM523I *csect-name* CLUSTER OR CLROUTE CANNOT CURRENTLY BE ALTERED

CSQM526I *csect-name* CLUSTER OR CLROUTE CANNOT CURRENTLY BE ALTERED*csect-name* CERTIFICATE LABEL NOT ALLOWED FOR SSL 3.0 CHANNEL

Buffer manager messages

CSQP054I Buffer pool *n* is now located above the bar

CSQP055I Buffer pool *n* is now located below the bar

CSQP056E The ALTER BUFFPOOL command for buffer pool *n* has failed

Topic manager messages

CSQT824I *csect-name* Topic *topic-1* is dependent on PROXYSUB(FORCE) of topic *topic-2* from a different Pub/Sub hierarchy stream

CSQT967E *csect-name* Unable to deliver proxy subscription to queue manager *queue-manager*, reason=*mqrc (mqrc-text)*

CSQT968I *csect-name* Topic *topic-1* in cluster *cluster-name* is dependent on PROXYSUB(FORCE) of topic *topic-2*

CSQT971E *csect-name task* failed to quiesce

CSQT983E *csect-name task* failed, reason *mqrc (mqrc-text)*, retry in *n* minutes

CSQT984E *csect-name task* has encountered *n* occurrences of reason *mqrc (mqrc-text)* while attempting to process a message.

CSQT987E *csect-name task* failed due to reason *mqrc (mqrc-text)* Retry in *n* minutes

CSQT988E *csect-name task* failed due to reason *mqrc (mqrc-text)* Retry in *n* minutes

CSQT989E *csect-name task* has encountered *n* occurrences of reason *mqrc (mqrc-text)* while attempting to process a message.

CSQT990E *csect-name task* has encountered *n* occurrences of reason *mqrc (mqrc-text)* while attempting to process a message.

CSQT991I *csect-name task* has recovered from previous error condition

CSQT996E *csect-name* Creation of proxy subscription failed on queue manager *qmgr-name*, cluster *cluster-name*, topic string *topic-string*, reason=*mqrc (mqrc-text)*

CSQT997E *csect-name* Cancellation of proxy subscription failed on queue manager *qmgr-name*, cluster *cluster-name*, topic string *topic-string*, reason=*mqrc (mqrc-text)*

CSQT998E *csect-name* Proxy subscription re-synchronization failed on queue manager *qmgr-name*, cluster *cluster-name*, reason=*mqrc (mqrc-text)*

CSQT999E *csect-name* Proxy subscription re-synchronization failed on queue manager *qmgr-name*, cluster *cluster-name*, reason=*mqrc (mqrc-text)csect-name* task has encountered a message that is not valid on queue *queue*

Utilities messages

CSQU179E The transmission queue cannot be switched because the channel initiator is not active

Distributed queuing messages

CSQX469E *csect-name* Update not received for CLUSRCVR channel *channel-name* hosted on queue manager *qmid* in cluster *cluster-name*, expected *n* days ago, *m* days remaining

CSQX878I *csect-name* Repository command error, command *command*, cluster object *object-name*, sender *sender-id*, reason *reason*

CSQX879E *csect-name* Conflicting clustered topic *topic-name* from queue manager *qmgr-name*

Initialization procedure and general services messages

CSQY024I IBM MQ AMS for z/OS is not installed, but the system parameter SPLCAP is set to YES

CSQY025I IBM MQ AMS for z/OS is not installed, but the system parameter SPLCAP is set to YESIBM MQ AMS for z/OS is installed.

CSQY336E *csect-name keyword* not allowed - restricted functionality

CSQY337E *csect-name keyword* value length not allowed - restricted functionality

IBM MQ Advanced Message Security messages

CSQ0417I Quality of protection: *qop*

CSQ0418I Toleration: *toleration-flag*

CSQ0468I No policies found

Service facilities messages

CSQ1134E KEYWORD EXTRACT REQUIRES AT LEAST ONE OUTPUT DDNAME

CSQ1219I LOG RECORDS CONTAIN *n* BYTE RBA - QSG(*in-qsg*)

z/OS: overall migration - order of tasks

Read this information that shows an overall migration plan for your system, together with the order in which you must undertake the tasks.

Before you begin

Read the information in Clustering: Best practices and Clustering: Topology design considerations to understand repositories.

About this task

The tables within this topic show the tasks required in each part of the process to migrate an overall system, and the order in which these tasks must be done.

Notes:

- You must perform the tasks in the following order:
  1. Overview of migration
  2. Migrating your system to the next release

Queue managers in a queue sharing group (QSG) and queue managers in a cluster can be migrated in parallel, but at any time there should be enough queue managers working in the QSG, and cluster, to ensure that your business can operate satisfactorily while a staged migration takes place.

If there are queue managers in clusters, ensure that you migrate the full repository queue managers before migrating any partial repository queue managers, so that the cluster always has a full picture of the current cluster architecture.

Table 16. Overview of migration

Task	For your own use
1. Make your configuration ready for migration	
2. Install the new code	

Table 16. Overview of migration (continued)

Task	For your own use
3. Server application migration	

Table 17. Migrating your system to the next release

Task	For your own use
Full repositories	
Overview - Migrate one full repository on IBM MQ for z/OS in a QSG	
4. Migrate DB/2 tables and New CF structure definitions	
5. Migrate the queue manager	
6. Migrate other full repositories on IBM MQ for z/OS, if any	
7. Full repositories not in a QSG	
8. Full repositories on other platforms	
Other queue managers	
Overview - Migrate partial repositories on IBM MQ for z/OS, in a QSG	
9. Migrate DB/2 tables and New CF structure definitions if not yet done	
10. Migrate one queue manager	
11. Test the migration and upgrade the other queue managers	
12. Migrate the partial repositories not in a QSG	
Migrate queue managers not in a QSG or cluster on IBM MQ for z/OS	
13. Migrate partial repositories on other platforms	
14. Migrate queue managers not in a cluster on other platforms	
Testing and using new function; see "Post migration tasks" on page 157	
15. Regression tests	
16. Use the new function available to you	

You can upgrade the client libraries at any time during the process. As a final task, recompile the clients using new functions and deploy.

Results

You have migrated your system to another release.

Migrating from earlier unsupported releases of IBM MQ for z/OS

You must take into consideration whether you are upgrading a production system, or a test system, before you undertake the migration process.

Production systems

For production systems you should, firstly, migrate the unsupported release of IBM MQ to IBM WebSphere MQ 7.1, following the instructions given in the documentation for that release. For more information about where to find the documentation for older versions of the product, see Documentation for older versions of IBM MQ

You can then migrate to IBM MQ 8.0 following the instructions in this section.

Important: Ensure that your system is stable at Version 7.1, before migrating to Version 8.0, so that you have a system to revert to, if necessary.

Test systems

For a test system, it might be appropriate to migrate directly to IBM MQ 8.0 instead.

You should take complete backups of your system, to ensure that you can restart from backups if you need to use the old release again.

IBM MQ 8.0 will migrate IBM MQ objects and messages during the first startup at Version 8.0.

New attributes added to objects in IBM WebSphere MQ 7.0 and IBM MQ 8.0 releases will be set to their default values.

After you have migrated to IBM MQ 8.0 using this method, you cannot revert to the original version.

You can restart a queue manager, at the original version, using the full set of backups taken before migration.

Note, that any changes you make to the system after the backups were taken, or while running at Version 8.0, will be lost.

Backward migration to earlier supported releases of IBM MQ for z/OS

After installation of a new release of IBM MQ for z/OS, you carry out queue manager migration by stopping the queue manager, which is running with the prior release of code, and restarting the queue manager using the new release of code.

Maintenance in a queue sharing group

In a queue sharing group, individual queue managers can be migrated forwards to IBM MQ 8.0.0 , while those that remain at either IBM WebSphere MQ 7.0.1 or Version 7.1.0 can continue to function. This allows you to upgrade queue-sharing group queue managers to Version 8.0.0 at different times, maintaining the high availability of the queue sharing group.

The function required to enable lower level queue managers to tolerate Version 8.0.0 additions to QSGDISP(GROUP) and QSGDISP(SHARED) objects is incorporated in the same authorized program analysis reports (APARs) which provide backward migration capability.

Code levels supported

Migration support is provided from both IBM WebSphere MQ 7.0.1 and IBM WebSphere MQ 7.1.0 to IBM MQ for z/OS Version 8.0.0.

The backward migration APAR for both IBM WebSphere MQ 7.0.1 and IBM WebSphere MQ 7.1.0 is PI19721.

Important: PTFs for this APAR must be applied on Version 7.0.1 or IBM WebSphere MQ 7.1.0 prior to attempting to fall back from IBM MQ for z/OS Version 8.0.0.

PTFs for this APAR are the *Migration and Toleration PTFs for Version 8.0.0* described in Planning for migration to the latest release.

Service has been discontinued for versions of the product prior to IBM WebSphere MQ 7.0.1. No backward migration capability is available for these versions.

The IBM MQ for z/OS Version 8.0.0 early code installed in the link pack area (LPA) is downward compatible. The code supports queue managers running at Version 7.0.1 and Version 7.1.0.

Once updated to the Version 8.0.0 level, and the queue manager subsystem refreshed using the REFRESH QMGR TYPE(EARLY) command, the early code need not be changed for any subsequent forward or backward migration activity

Message

```
CSQ3111I <cpf> CSQYSCMD - EARLY PROCESSING PROGRAM IS V8.0.0 LEVEL 007-001
```

is displayed during startup in the queue manager joblog and indicates that the queue manager is using the correct level of early code.

Limitations and restrictions

IBM MQ for z/OS Version 8.0.0. uses a migration switch to support backward migration by preventing use of certain new functions, that cannot be backward migrated, until the installation confirms that backward migration is no longer required.

The migration switch is configured through a change to ZPARM using the OPMODE parameter of CSQ6SYSP.

While OPMODE is set to COMPAT, it is possible to backward migrate, although certain new functions are not available. Once OPMODE is set to NEWFUNC, all new functions are available, but it is no longer possible to perform backward migration

The MQSC command DISPLAY SYSTEM shows the current value of OPMODE. The command displays two values, the operation mode, either COMPAT or NEWFUNC, and a version number.

When the operation mode is COMPAT, then the version number indicates to which version of IBM MQ for z/OS you can fall back.

The value of OPMODE displayed during startup in message CSQY101I *reflects the operation mode requested using ZPARM. Queue manager initialization evaluates the requested operation mode in combination with local state and other members of the queue sharing group, to determine the actual operation mode displayed with DISPLAY SYSTEM.*

You cannot backward migrate a queue manager, newly created at Version 8.0.0, to a prior release. A queue manager migrated forward to Version 8.0.0 *remembers* where it was migrated from, and it is only possible to fall back to that *remembered* prior version.

Certain connection types (IMS, BATCH and RRSBATCH used by WAS and Db2 stored procedures) allow an application to connect to multiple queue managers concurrently. If required, these queue managers can be running different levels of IBM MQ code. In such a scenario, the adapter code (usually referenced through a STEPLIB DD statement or environment variable) must be loaded from libraries corresponding with the highest level of the queue managers connected. This ability for the adapter code to support connections to older queue managers means that in a backward migration scenario it is possible to just restart the MSTR and CHIN procedures with the back level code, and not change connecting jobs.

The operations and controls ISPF panels, CSQOREXX, from IBM MQ for z/OS Version 8.0.0 , are able to connect to and administer queue managers from a prior release. However, the ISPF panels from lower releases are not able to connect to IBM MQ for z/OS Version 8.0.0. When migrating, or during fall back, either use the same version ISPF panels as the level of code the queue manager is running, or use CSQOREXX from the higher release of code. In a mixed level queue sharing group, the IBM MQ for z/OS Version 8.0.0 panels must be used to administer Version 7.1.0 or Version 7.0.1 queue managers, as ISPF panels from earlier releases do not tolerate responses from any Version 8.0.0 queue managers.

Queue sharing groups

Prior to migrating any queue manager in a queue sharing group to IBM MQ for z/OS Version 8.0.0 you must ensure that all the queue managers in the queue sharing group are at the same version and release. That is, they must all be at IBM WebSphere MQ 7.0.1 or, all at IBM WebSphere MQ 7.1.0.

⚠ **Attention:** Once you have started the migration process to IBM MQ for z/OS Version 8.0.0 for queue managers in a queue sharing group, it is not possible to backward migrate any other queue manager to an earlier release. This would lead to three different versions in the queue sharing group, which is not permitted.

Related reference

“z/OS: Switching from OPMODE=(NEWFUNC,800) to OPMODE=(COMPAT,800)” on page 274
The availability of new functions and backward migration for IBM MQ for z/OS is controlled by the **OPMODE** parameter in the **CSQ6SYSP** macro. You should be aware of the implications of switching from `OPMODE=(NEWFUNC,800)` to `OPMODE=(COMPAT,800)`

Preparing to migrate a single IBM MQ for z/OS queue manager

Follow the steps to prepare a single IBM MQ queue manager on z/OS for migration.

About this task

To prepare to migrate an IBM MQ queue manager on z/OS, you need to carry out the detailed steps in this topic, using the links within this overview.

1. Make your existing queue manager ready for migration; see Step 1
2. Install the new code and make target libraries available to all MVS systems that are running queue managers, and grant access; see Step 2.
3. Perform a back up operation of each queue manager in your enterprise; see Step 3.
4. Review definitions of the user IDs for the queue manager(MSTR) and channel initiator (CHIN) address spaces; see Step 4
5. Restart your IBM MQ systems; see Step 4.
6. Review pageset zero usage before migration; see Step 5.
7. Migrate your Db2 tables, and repeat this step for each queue sharing group (QSG), if your enterprise uses QSGs; see Step 6
8. Add a new coupling facility (CF) structure definition and repeat this step for each QSG, if your enterprise uses QSGs; see Step 7.
9. Consider the migration of your application server; see Step 8
10. Configure Advanced Message Security (AMS); see Step 9

.

Procedure

1. Make your IBM MQ configuration ready for migration.
   a) Refer to the Preventive Service Planning (PSP) bucket for your version of IBM MQ ; see PSP Buckets - How to find them on Web.
   b) Apply the migration and toleration PTFs to the version of the IBM MQ code that your enterprise uses; see IBM MQ Support, Migration PTFs.

      Note that the "migration and toleration" PTFs are also known as the "backward migration and coexistence" PTFs; they are the same PTFs.

      If you are unsure which migration PTFs you require, run the following command SMP/E:

      ```
      REPORT MISSINGFIX ZONES(mqtgtzone) FIXCAT(IBM.Coexistence.MQ.V8R0M0)
      ```

      See FIXCAT and IBM MQ Migration Installation for further information.

      Attention: If a PTF requires a rebind of Db2 plans, the PTF is shipped with ++HOLD(ACTION), indicating the need for this process. In such a case, see Migrating Db2 tables to bind the plans before starting migration.

      Other FIXCAT categories are listed in IBM Fix Category Values and Descriptions.

There is an additional category of `TargetSystem-RequiredService.MQ.V8R0M0` enabling other products to run with IBM MQ 8.0.

2. Install the new code and make target libraries available to all MVS systems that are running queue managers, and grant access.

   You must carry out the following procedure for each MVS system.

   a) Copy the IBM MQ target libraries to the system, and install the early code for the new version (once for each MVS system).

      Activate the code for all of the queue managers on each MVS system that is running queue managers.

      This updates the LPA. See Update the z/OS link list and LPA for more information.

   b) APF authorize the load libraries and grant access to the datasets using your external security system.

      See APF authorize the IBM MQ load libraries for more information.

   c) Copy the file system zFS and mount it read only.

      You only need zFS, or older HFS, if the IBM MQ for z/OS Unix System Services Component is installed. See the Program Directory for further information.

   Refresh all the queue managers so that they use the new early code using the command REFRESH QMGR TYPE(EARLY). See REFRESH QMGR for more information.

3. Perform a back up operation for each queue manager in your enterprise, so that you have a `before` copy of all objects and JCL before you make any changes.

   This makes rolling back to the current system easier, if you require to do so.

   a) Backup your IBM MQ defined objects, for example using CSQUTIL COMMAND MAKEDEF(..)

      See Issuing commands to Websphere MQ (COMMAND) for more information.

   b) Backup:

      - The MSTR and CHINIT started procedure jobs
      - The Initialization input datasets used in the CSQINP1 and CSQINP2 concatenations
      - The system parameter module (ZPARM) libraries
      - Other tasks as necessary.

      Note: You might also make a back up of page sets, BSDSs, and active logs as a fallback option. See How to backup and recover pagesets for more information on backing up IBM MQ resources.

4. Check that MSTR and CHIN address spaces run under user IDs that have OMVS segments defined, with a valid UID, to enable calling Unix System Services (USS).

5. Restart your IBM MQ system to run with the migration and toleration PTFs.

   a) Restart the queue managers and monitor the whole system in your enterprise closely to ensure that there are no issues.

      Depending on the size and complexity of your enterprise this can take a considerable length of time, so you must plan for this in your migration schedule.

6. Review the usage of pageset 0.

   Note that you can ignore this step if your enterprise is already using IBM MQ V7.1.

   Issue the operator command `/cpf DISPLAY USAGE PSID(0)`, where **`cpf`** is the command prefix for the queue manager's subsystem, to get a report on pageset 0 usage.

   The size of queue definitions increased in IBM MQ V7.1. During migration to this version, or later versions of the product from an earlier version of the product, queue definitions stored on pageset 0 are rewritten.

   The rewrite is carried out as a single transaction when the queue manager is first migrated to IBM MQ V7.1, or later.

Ensure that there is enough space available on pageset 0 to create a copy of the queue definitions while migration is taking place. Typically, 60% free space on pageset 0 before migration is sufficient. However, the use of EXPAND(SYSTEM) on the pageset definition allows for automatic expansion as required.

If there is insufficient space on pageset 0 during migration, the queue manager abends with completion code X'5C6' and reason code X'00C91900'.

7. Migrate your Db2 tables for each Db2 data sharing group.

   You must do this for each Db2 data sharing group, as multiple QSGs can use the same Db2 tables.

   You can use IBM provided samples shipped in the new version of the product to perform this task. Some Db2 table definitions are updated, and some new Db2 tables are created for the migrated version of the queue manager.

   Notes:

   a. You must have applied the migration and toleration PTFs to all the queue managers, before migrating the Db2 tables.
   b. Every queue manager in the QSG needs to be restarted at the current release, with the PTFs applied.
   c. At no stage is an outage of the entire queue-sharing group required.
   d. Migrate your Db2 tables.

      If the jobs described fail because of a Db2 locking problem, it might be due to contention for a Db2 resource. Locking is more likely, if the system is being heavily used. Resubmit the job later, preferably when the system is lightly used or quiesced.

      See steps 5 and 6 of Set up the Db2 environment.
   e. Use the CSQ45* jobs in the newest *thlqual*.SCSQPROC supplied with the version of the product to which you are migrating.

      Note that the JCL to use depends on the highest version of IBM MQ in the Db2 tables.

      i) If the Db2 tables have IBM WebSphere MQ 7.1 queue managers, use CSQ4571T. If the Db2 tables have IBM WebSphere MQ 7.0 queue managers, use CSQ4570T.
      ii) Customize the CSQ45* sample.

          The header information in CSQ45* describes how to customize the sample.
      iii) Run the customized CSQ45* job.
      iv) Customize the CSQ45BPL and CSQ45GEX samples, in *thlqual*.SCSQPROC

          The header information in CSQ45BPL and CSQ45GEX describes how to customize the samples.
      v) Run the customized jobs, CSQ45BPL and CSQ45GEX.

          If you need to rebind the plans or packages, see Step “2” on page 159 in Migrating queue sharing groups from a previous release for further information.
   f. If you have multiple QSGs in the same data sharing group (DSG) you need to check each QSG to see if each member meets its migration criteria. Use sample JCL CSQ45MQS in conjunction with CSQ4571T.

      See the JCL header description for further information.

8. Add the new coupling facility (CF) definition.

   Repeat this step for each QSG. Note that you can ignore this step if your enterprise is already using IBM MQ V7.1.

   Starting with IBM MQ V701, a new CF structure is required; see Set up the coupling facility for information on how to add such a definition.

   The correct process to migrate SYSTEM.QSG.CHANNEL.SYNCQ, from a normal application CF structure, to system CF structure CSQSYSAPPL structure is:

a) Stop the channel initiator (CHINIT) on all QSG queue managers, so that no channels are running.

b) Copy the messages in SYSTEM.QSG.CHANNEL.SYNCQ to a temporary dataset, using CSQUTIL COPY.

c) Delete SYSTEM.QSG.CHANNEL.SYNCQ from the repository.

d) Define SYSTEM.QSG.CHANNEL.SYNCQ with CFSTRUCT(CSQSYSAPPL).

   As this is a shared queue, it only needs to be defined once per QSG. Note that you can define this queue from any queue manager within the QSG.

e) Reload the SYNCQ messages from the temporary dataset, back to the newly defined shared queue, using CSQUTIL LOAD.

f) Perform the other migration steps, and then restart CHINIT to make the changes taking effect.

9. Migrate server applications.

   Java or JMS applications running on the same host with IBM MQ connect to queue managers in bindings mode. This is a cross-memory connection. In this mode, applications need to update their STEPLIB concatenations, so that they can always load the highest version IBM MQ library in the system.

   Note, that if a z/OS Java or JMS application is running under WebSphere Application Server, the application can use client mode as an alternative to bindings mode.

   IBM MQ libraries include:

   thlqual.SCSQANLx
   This library contains error message information for your national language. The letter 'x' represents the letter for your national language.

   thlqual.SCSQAUTH
   This library contains the code that the applications use.

   Server applications for IBM MQ can include:

   - Batch applications
   - Control panels in ISPF
   - IMS
   - Interactive problem control system (IPCS)
   - RRS adapter, including Db2 stored procedures.
   - TSO
   - Additionally, WebSphere Application Server for z/OS, IBM Integration Bus, and CICS.

   a) You can use the "TSO ISRDDN ENQ ' `thlqual`.SCSQANLE'" command, replacing `thlqual` with the High Level Qualifier for your installation, to check which jobs are running with the specified library. You can then modify them accordingly.

   b) Update STEPLIB in the application JCL, and refer to the new IBM MQ libraries.

   c) Restart these applications.

      For further information, see:

      - Set up Batch, TSO, and RRS adapters
      - Setting up the IMS adapter
      - Set up the operations and control panels
      - Include the IBM MQ dump formatting member

   d) Migrate other software, such as WebSphere Application Server, IBM Integration Bus, or CICS to use the version of IBM MQ that you need.

      - CICS

        Update the IBM MQ libraries in the `STEPLIB` and DFHRPL concatenations of your CICS region JCL and restart CICS.

Up to, and including CICS 3.2, the connection between IBM MQ and CICS is provided by IBM MQ. You must change the SCSQCICS and SCSQAUTH libraries in the DFHRPL concatenation provided by IBM MQ.

After CICS 3.2, the connection between IBM MQ and CICS is provided by CICS libraries. Update the libraries, if you are using CICS Transaction Server for z/OS Version 3.2 or later. Without this change, you are not able to use the most recent IBM MQ features. You must change the SCSQCICS library in the DFHRPL concatenation provided by IBM MQ, and also the STEPLIB concatenation.

Create separate CICS started procedure JCL. For each CICS region that is connected to an IBM MQ queue manager, ensure that there is a separate CICS started procedure JCL.

This ensures that the modification of reference to a certain version of IBM MQ libraries in the CICS started procedure JCL only has impact for that single CICS region. In this way you can migrate one queue manager, and only the CICS region or regions connected to it, which makes staged migration possible.

CICS STEPLIB has `thlqual.SCSQAUTH`, and DFHRPL has `thlqual.SCSQCICS`, `thlqual.SCSQLOAD`, and `thlqual.SCSQAUTH`. For more information, see Setting up the CICS-IBM MQ adapter

- WAS for z/OS

  If you are running in an application server environment where a bindings connection is being used, you need to update the WAS STEPLIB with IBM MQ libraries.

  See IBM MQ libraries and the WebSphere Application Server for z/OS STEPLIB for further information.

  You also need to configure the IBM MQ messaging provider with native libraries from the new version of the IBM MQ installation; see Configuring the IBM MQ messaging provider with native libraries for further information.

  Use the latest level of native libraries in USS.

Note that you can make use of a DFP ALIAS for convenience. Create data set aliases such as MQM.SCSLOAD, and reference them in JCL. Map the aliases to the real data sets, such as MQM.V700.SCSLOAD or MQM.V710.SCSLOAD.

Change the aliases to switch between the two sets of target libraries. With the aliases, you can start applications or the queue manager when moving to a new release of IBM MQ without changing the STEPLIB JCL.

10. Advanced Message Security (AMS)

    If the queue manager is configured to use Advanced Message Security (AMS) perform the steps in the Preparing to migrate Advanced Message Security section of the Migrating Advanced Message Security topic.

Results

You have prepared your IBM MQ queue manager on z/OS for migration.

What to do next

Follow the instructions in “Migrating a single IBM MQ z/OS queue manager to the next release of the product” on page 151 to migrate the queue manager.

Related information

Program Directory for IBM MQ for z/OS

Migrating a single IBM MQ z/OS queue manager to the next release of the product

Carry out the instructions in this topic to migrate a single IBM MQ queue manager on z/OS,

About this task

To migrate an IBM MQ queue manager on z/OS to a different release, you need to carry out the:

- Process described in "Preparing to migrate a single IBM MQ for z/OS queue manager" on page 146
- Detailed steps in this topic, using the links within this overview.
  1. Stop or disconnect the applications; see step "1" on page 151
  2. Stop the queue manager and its channel initiator; see step "2" on page 151
  3. Update STEPLIB for MSTR and the channel initiator; see step "3" on page 151
  4. Update the initialization input data sets; see step "4" on page 151
  5. Update the target version system parameter module (ZPARM); see step "5" on page 152
  6. Configure Advanced Message Security; see step "6" on page 152
  7. Review the security control of your system; see step "7" on page 152
  8. Start the queue manager; see Step "8" on page 152
  9. Optionally, revert the queue manager to a previous version; see step "9" on page 152

Procedure

1. Stop or disconnect all the applications using the queue manager (for example, CICS, IMS, or batch) and the IBM MQ channels that are connected to other queue managers.
2. Stop the queue manager and its channel initiator.
3. Update STEPLIB for MSTR and the channel initiator (CHINIT).

   Update the start procedure and CHINIT JCL.

   a) Update your procedure to start the queue manager.

      Change the STEPLIB for the queue manager to reference the new version of the libraries. See Create procedures for the IBM MQ queue manager.

      IBM MQ now uses z/OS memory objects above the bar for some functions. You must allow the queue manager to access storage above the bar.

      Your installation might have customized the SMFPRMxx member of SYS1.PARMLIB, or the **IEFUSI** exit to provide a default limit for jobs using virtual storage above the 2 GB bar. Check these limits give sufficient memory for a queue manager. A reasonable starting allocation is 2 GB. The message CSQY220I displays the amount of virtual storage currently used and available.

      If your installation does not have a default limit for storage above the bar, or if you want to use a different limit for your queue manager, you can provide a queue manager-specific restriction on the amount of virtual storage available above the bar for memory objects by coding a **MEMLIMIT** parameter on the JCL of the queue manager stored procedure, xxxxMSTR, for example:

      ```
      //PROCSTEP EXEC PGM=CSQYASCP,REGION=0M,MEMLIMIT=2G
      ```

      MEMLIMIT defines memory available above the bar; see Address space storage

   b) Update your procedures for the channel initiator.

      Change the STEPLIB for the channel initiator to reference the new level of the product libraries. See Create procedures for the channel initiator.
4. You must allow the queue manager to access storage above the bar because IBM MQ uses memory above the bar.

If insufficient storage is available above the bar, the queue manager reports this when starting, and stops.

See step 3a for further information on **MEMLIMIT**.

5. Optionally, update the target-version system parameter module (ZPARM).

   Tailor the target version (ZPARM) sample to use new MQ libraries and generate new ZPARM.

   Some new parameters might be added in the target version; the current version of the IBM MQ libraries cannot recognize them.

   Configure the OPMODE parameter, for example OPMODE=(COMPAT,800) to allow the queue manager to revert if necessary. For further details, see OPMODE.

   ZPARM is forward compatible, as it simplifies the forward migration process. This means that any attributes introduced in the new release will have their default values.

   This step is optional during queue manager migration; you can complete this task after you confirm that no problems occurred after migration. This option gives you an easier approach when backward migration is needed.
6. If the queue manager is configured to use Advanced Message Security (AMS), perform the steps in Migrating Advanced Message Security.
7. Review your security control for queue sharing groups, the channel initiator, and all queue managers accessing the coupling facility list structures.
8. Start the queue manager.

   Test that everything is working correctly and, if it is, start the channel initiator. If there is a problem starting the queue manager, consider reverting the queue manager to a previous version; see step "8" on page 152.
9. If a problem occurs when starting the queue manager, you might need to consider backward migration; see Reverting a queue manager to a previous release.

Results

You have migrated your IBM MQ for z/OS queue manager to the latest release.

What to do next

Follow the instructions in "Post migration tasks" on page 157 to complete the migration process.

z/OS: CSQINP1 and CSQINP2 input data sets changed

The CSQINP1 and CSQINP2 initialization input data sets changed in Version 7.1. The data sets include more samples and the contents of some samples have been moved to other samples. Particular changes to take note of are the commands to define queues to hold publish/subscribe state information. The commands must be in the right order.

Important changes to initialized input data sets since IBM WebSphere MQ 7.1

CSQ4INSM

Added for IBM MQ Advanced Message Security support

CSQ4INSG

Add one **AUTHINFO** object SYSTEM.DEFAULT.AUTHINFO.IDPWOS for connection authentication support.

Some channel objects and topic objects are modified with new attributes, for example, **STATCHL** and **CLROUTE**.

CSQ4INST

The default system subscription, SYSTEM.DEFAULT.SUB, moved from CSQ4INSG to CSQ4INST in IBM WebSphere MQ 7.1.

CSQ4INSX
Add one model queue SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE for multiple cluster transmission queue support.

Review the changes, and update the customized versions you are currently using as required, if your enterprise is using IBM WebSphere MQ 7.0.

The queue manager uses queues to hold persistent state information about publish/subscribe. Durable subscriptions are held as messages on SYSTEM.DURABLE.SUBSCRIBER.QUEUE and retained publications on SYSTEM.RETAINED.PUB.QUEUE.

The ordering of the definitions of these queues is important. Member CSQ4MSTR of **SCSQPROC** shows the necessary ordering of the supplied definitions in the CSQINP2 concatenation. The default system subscription, SYSTEM.DEFAULT.SUB, requires SYSTEM.DURABLE.SUBSCRIBER.QUEUE that in turn requires the storage class SYSLNGLV, which is defined in CSQ4INYS. If you migrating from a previous release, and modifying customized procedures, define these resources in the following order:

1. Storage class SYSLNGLV. It requires a defined mapping to a defined page set.
2. SYSTEM.DURABLE.SUBSCRIBER.QUEUE
3. SYSTEM.DEFAULT.SUB

Note:

The changes to CSQINP1 and CSQINP2 are required in Create procedures for the IBM MQ queue manager and Customize the initialization input data sets.

z/OS Migrating a queue manager to mixed case security

Follow these steps to migrate a queue manager to mixed case security. You review the level of security product you are using and activate the new IBM MQ external security monitor classes. Run the **REFRESH SECURITY** command to activate the mixed-case profiles.

Before you begin

1. Install a level of the security product that supports mixed case security.
2. Apply any updates required by IBM MQ.
3. Install and activate the new IBM MQ external security monitor classes.

About this task

Follow these steps to convert a queue manager to mixed case security.

Procedure

1. Copy all your existing profiles and access levels from the uppercase classes to the equivalent mixed case external security monitor class.
   a) MQADMIN to MXADMIN.
   b) MQPROC to MXPROC.
   c) MQNLIST to MXNLIST.
   d) MQQUEUE to MXQUEUE.
2. Start the queue manager.

   The queue manager SCYCASE attribute is set to UPPER.
3. Change the value of the SCYCASE attribute to MIXED.

```
ALTER QMGR SCYCASE(MIXED)
```

4. Activate your existing security profiles.

```
REFRESH SECURITY(*) TYPE(CLASSES)
```

5. Test that your security profiles are working correctly.

What to do next

Review your object definitions and create new mixed case profiles as appropriate, using **REFRESH SECURITY** as required to activate the profiles.

z/OS: Migrating IBM MQ Advanced Message Security

IBM MQ Advanced Message Security for z/OS (AMS) is a separately licensed enabling product that extends IBM MQ to provide a high level of protection for sensitive data flowing through the IBM MQ network using a public key cryptography model.

In IBM MQ for z/OS releases prior to Version 8.0, AMS was provided as a separate product. This topic describes the tasks required to migrate the AMS configuration on z/OS from that used in version 7 and earlier, to that used in Version 8.0. These steps supplement those required to migrate a single IBM MQ for z/OS queue manager where AMS is not configured. AMS must be migrated at the same time as the queue manager, it is not supported to use IBM MQ Advanced Message Security Version 7.0.1 with IBM MQ for z/OS Version 8.0.

To enable AMS on a newly created IBM MQ for z/OS queue manager, or on a queue manager that has already been migrated to Version 8.0, see IBM MQ Advanced Message Security for z/OS.

Preparing to migrate Advanced Message Security

To prepare to migrate an IBM MQ queue manager on z/OS using AMS Version 7.0.1 or earlier, you must perform the steps in this section in addition to those listed in "Preparing to migrate a single IBM MQ for z/OS queue manager" on page 146.

1. Install the IBM MQ Advanced Message Security for z/OS enabling product and make the target libraries available to all MVS systems that are running queue managers that will use AMS. You must carry out the following procedure for each MVS system:
    a. Copy the AMS target libraries to the system.
    b. APF authorize the thlqual.SDRQAUTH target library and grant access to this data set using your external security system, see Task 2: APF authorize the IBM MQ load libraries.
    c. Ensure the LPA contains the AMS module CSQ0DRTM, see Task 3: Update the z/OS link list and LPA.
    d. Ensure the program properties table (PPT) contains an entry for CSQ0DSRV, see Task 4: Update the z/OS program properties table.
2. For each queue manager, set up the started task user for the AMS address space. In AMS Version 7.0.1 two address spaces are used, one for the main task and another for the data services task. In Version 8.0 these are combined in to a single address space called qmgrAMSM. Either set up a new user for the Version 8.0 AMS address space, or grant additional authorities to one of the existing AMS started task users. See Task 25: Set up the started task user Advanced Message Security for information on how to set up the started task user. If you do not use the existing data services address space user you will need to replicate the **drq.ams.keyring** key ring for the user ID associated with the Version 8.0 qmgrAMSM address space. See Using certificates on z/OS for information on how to set up the AMS key ring.

Migrating Advanced Message Security

To migrate an IBM MQ queue manager on z/OS using AMS Version 7.0.1 or earlier, before restarting the queue manager you must perform the steps in this section in addition to those listed in "Migrating a single IBM MQ z/OS queue manager to the next release of the product" on page 151.

1. Take a copy of the qmgrAMSM task for IBM MQ Advanced Message Security (AMS) Version 7.0.1, in case you need to revert to your previous system.

   See "Backward migration of Advanced Message Security" on page 155 for more information.
2. Configure the queue manager to use AMS by updating the system parameter module to set SPLCAP(YES) using CSQ6SYSP, see Task 17: Tailor your system parameter module and Using CSQ6SYSP.
3. Create or update the started task procedure for the qmgrAMSM address space, see Task 24: Create procedures for Advanced Message Security.

Post migration tasks for Advanced Message Security

After you have migrated an IBM MQ queue manager on z/OS that uses AMS you must perform the following tasks.

1. In Version 8.0, the AMS address space is started and stopped automatically by the queue manager. If you have automation to manage the main task and data services task for AMS Version 7.0.1, or earlier, this should be removed. You must also review any automated console commands for AMS because some have changed in Version 8.0.
2. Delete the started task procedures for the Version 7.0.1 data services task and the Version 7.0 main task if these were not called qmgrAMSM.

Backward migration of Advanced Message Security

If you are an AMS user, and you backward migrate your queue manager from Version 8.0 to a version 7 release, additional actions are required to revert AMS to version 7.

Considerations when migrating

You should ensure that your previous setup is in place and that tasks Updating the z/OS LPA to Updating your system DIAG member have been carried out.

Ensure that the user ID associated with the version 7 data-services address spaces has access to drq.ams.keyring, and that drq.ams.keyring has the same connected certificates as the Version 8.0 qmgrAMSM user ID.

Performing the migration

When you have completed the previous tasks, you can migrate your queue manager backwards in the normal way.

Manually start, or reintroduce automation for starting, the AMS main and data services address spaces.

See Starting IBM MQ Advanced Message Security for further information.

z/OS: Reverting a queue manager to a previous release

After migrating to IBM MQ for z/OS Version 8.0, from either Version 7.0.1 or Version 7.1.0, you can backward migrate, or fallback, to the version you were using prior to migration. Backward migration Program Temporary Fixes (PTFs) are available for both Version 7.0.1 and Version 7.1 0.

Before you begin

In general, after fallback to IBM WebSphere MQ 7.0, new attributes of IBM MQ objects introduced at Version 8.0 will be removed. The APAR that supplies these PTFs, documents specific information relating to fallback to IBM WebSphere MQ 7.0.1 or to IBM WebSphere MQ 7.1.0.

Switching back to running a queue manager with the target libraries of a previous version is possible if **DISPLAY** SYSTEM returns COMPAT=vrm where vrm is the level of the previous version. If it does, you can go back to using your customization and startup procedure for the queue manager from that version

- The queue manager compatibility level must be `7rm`. If the queue manager has never been run with **OPMODE** set to `(NEWFUNC,800)`, the compatibility level is `7rm`.

  – **DISPLAY** `SYSTEM` returns `OPMODE COMPAT,7rm`.

  where

  7
  is the version number v of the product.

  r
  is the release number of the product.

  m
  is the modification number of the product.
- Before migrating your queue manager to the latest version, with that version of target libraries, you applied all the migration and toleration PTFs to the queue manager on your previous version. The queue manager then started successfully with those PTFs at that previous version. This is a requirement before you can revert your queue manager to the original version.
- You saved the queue manager customization macros and JCL for running with the Version 7.0 target libraries.

  You can recreate the customization for version 7.r , if the originals are not available to you.

About this task

To restart the queue manager, so that it runs at the version where it was migrated from, just requires that you switch back to using the libraries for the previous version.

Note that it is not necessary to rollback the early code for this installation when reverting your queue manager to an earlier version.

Procedure

1. Stop the listener, channel initiator, and queue manager.
2. Switch back to use the MSTR and CHINIT started procedure JCLs with version 7.r libraries.

   In case data set aliases are used for load libraries, switch the aliases to refer to version 7.r libraries. For example, an alias named `MQM.MQP1.SCSLOAD`, referring to `MQM.MQV800.SCSLOAD`, needs to change to refer to `MQM.MQV7xx.SCSLOAD`.
3. Restart queue manager, using the system parameter module (CSQZPARM) used with IBM MQ version 7.r prior to migration, and linking to the version 7.r code.

   Until you have verified the startup, start the queue manager, channel initiator, and listener separately, checking for errors on the console after each component is started. If the startup runs cleanly, combine the startup of all three components in the production environment.

   a) Start the queue manager.

   b) Start the channel initiator.

   c) Start the listener.
4. Verify correct functioning of existing applications.

Results

If the queue manager cannot be reverted to the previous release by following the preceding procedure, for example, because it has been started with OPMODE set to (NEWFUNC,800), the queue manager can only be reverted to the previous release by recovering the page sets, BSDSs, and active logs from back up copies taken before the migration to IBM MQ for z/OS Version 8.0.

All updates made since the back up was taken will be lost. See How to backup and recover pagesets for more information on backing up IBM MQ resources.

Post migration tasks

Follow the steps to perform the tasks you need to carry out after migrating a single IBM MQ queue manager on z/OS,

About this task

After you have migrated an IBM MQ queue manager on z/OS you need to carry out the detailed steps in this topic, using the links within this overview.

1. Check the changes in behavior made by default configuration changes; see Step "1" on page 157
2. Modify the backup jobs to refer to the target version of IBM MQ libraries; see Step "2" on page 157
3. Configure Advanced Message Security; see "3" on page 157
4. Perform a full regression test; see Step "4" on page 157
5. Update the ZPARM module if you have not already done so; see Step "5" on page 157
6. Set OPMODE to NEWFUNC; see Step "6" on page 157
7. Exploit the new functions provided by the migrated queue manager; see Step "7" on page 157
8. Consider client application migration; see Step "8" on page 157

.

Procedure

1. Check the changes in behavior made by default configuration changes.

   The default values of some properties might have been changed in the new version, which can lead to changes in behavior.

   SHARECNV allows multiple connections to the queue manager to permit the use of the same TCP/IP connection. If a client is using Version 6 code to connect to a version 7, or later, IBM MQ queue manager, SHARECNV is set to 0 automatically; see Default behavior for more details about this change.

   On z/OS, you can reverse queue manager migration as long as you have not enabled new function. You enable new function by setting the **OPMODE** parameter to `(NEWFUNC,800)` ; see OPMODE for more information.
2. Modify backup, and other administrative, jobs to refer to the target version of IBM MQ libraries, such as backup IBM MQ objects and MAKEDEF jobs.

   For example using CSQUTIL COMMAND MAKEDEF(..); see Using the COMMAND function of CSQUTIL.

   You should also backup channel authentication records, which were introduced in IBM WebSphere MQ 7.1.0.
3. If the queue manager is configured to use Advanced Message Security (AMS) perform the steps in the Post migration tasks for Advanced Message Security section of the Migrating Advanced Message Security topic.
4. Perform a full regression test.
5. Update the ZPARM module if you have not already done so.

   See Update the ZPARM module for further information.
6. Set OPMODE in ZPARM JCL to NEWFUNC and recompile the JCL.

   For more information about NEWFUNC, see OPMODE.
7. Exploit the new functions provided by the migrated queue manager.

   Your queue manager has been fully migrated to a new version level, and you can take benefit of new capability now.

   Review What's new in IBM MQ 8.0 and check which features best serve your business needs. Plan your action to develop new applications, or changing configurations, to enable those features.
8. Migrate client applications.

Client applications running on z/OS, or other distributed platforms, can be considered any time throughout the migration phase.

The client libraries need to be at the same level as the lowest IBM MQ queue manager level to which they connect. So, if a client can connect to Version 701, Version 710, or Version 800 IBM MQ queue managers, the client needs to be at Version 701. Once all the IBM MQ queue managers have been migrated to Version 800 you can migrate the clients to Version 800.

For further information, see "Migrating an IBM MQ MQI client on UNIX platforms, and Windows to the latest version" on page 71 and select the platform that you require.

Results

You have completed the migration of a single IBM MQ for z/OS queue manager.

z/OS: Adding a new queue-sharing group to an existing Db2 data sharing group in the latest version

Follow these steps to add a new queue-sharing group to an existing Db2 data sharing group in the latest version of the product. You must apply the migration and toleration PTFs to queue managers, in the previous version, in any of the queue-sharing groups before adding a queue-sharing group.

Before you begin

1. Review your Db2 data-sharing requirements. A single Db2 data-sharing group can be used to support multiple IBM MQ queue-sharing groups.
2. You can add a new queue-sharing group to a Db2 data-sharing group that already supports IBM MQ queue-sharing groups containing queue managers for the previous version. You must ensure that the migration and toleration PTFs have been applied. The Db2 tables used by IBM MQ must be configured for the latest version queue managers.

About this task

Queue-sharing group migration affects steps "4" on page 86, "8" on page 87, "11" on page 87, and "12" on page 88 of "z/OS: Review and modify queue manager customizations from the previous release" on page 84

Procedure

1. Customize the CSQ4570T and CSQ4571T samples, in *thlqual*.SCSQPROC, supplied with the latest version of the IBM MQ for z/OS product.

   The header information in CSQ4570T and CSQ4571T describes how to customize the samples. Delete or bypass the step that runs MIGRATE QSG.
2. Run the customized CSQ4570T and CSQ4571T jobs.
3. Set up the coupling facility.

   See Task 10: Set up the coupling facility.
4. Customize and include the initialization input sample *thlqual*.SCSQPROC(CSQ4INSS) in the CSQINP2 data set.

   See step "11" on page 87 in "z/OS: Review and modify queue manager customizations from the previous release" on page 84.
5. Add the IBM MQ entries to the Db2 data-sharing group using the **CSQ5PQSG** program.

   See Task 16: Add the IBM MQ entries to the Db2 data-sharing group.
6. Tailor the system parameter module to add Db2 data-sharing group and IBM MQ queue-sharing group information.

See step “12” on page 88 in “z/OS: Review and modify queue manager customizations from the previous release” on page 84.

z/OS z/OS: Migrating queue sharing groups from a previous version of the product

You can migrate one or more existing queue-sharing groups containing queue managers in a previous version of a product to the latest version. At no stage is an outage of the entire queue-sharing group required.

Before you begin

1. Read the topic, “Queue-sharing group migration” on page 76, and its related references, especially “z/OS: Queue sharing group coexistence” on page 176.

About this task

Migrating each queue manager comprises the bulk of the work of migrating a queue-sharing group. Approach migrating a queue sharing group as requiring some extra tasks that must be performed during the migration of each queue manager. A good approach is to create a migration plan incorporating queue-sharing group migration; see “z/OS: Migration planning to the latest release” on page 81.

Queue-sharing group migration affects steps “4” on page 86, “8” on page 87, “11” on page 87, and “12” on page 88 of “z/OS: Review and modify queue manager customizations from the previous release” on page 84

Procedure

1. Ensure that all members of the queue-sharing group have been started at the same version, before migrating any queue managers in the queue-sharing group to the latest version,.

   Do this, by migrating the queue managers at the oldest version in the queue-sharing group to the same version as the other queue managers. For example, if the queue-sharing group currently contains queue managers at Version 7.0.1 and Version 7.1, migrate the Version 7.0.1 queue managers to Version 7.1, before migrating any queue managers in the queue-sharing group to Version 8.0.
2. Apply IBM MQ for z/OS migration and toleration [13] PTFs for the latest version of the product to the earlier version code; see IBM MQ Support, Migration PTFs.

   a) Apply the PTFs to the libraries of the earlier version of the product.

      If you need to rebind a plan or package, follow the instructions in the PTFs, and rebind the packages using the instructions in CSQ45BPK.

      If you rerun the bind of the plan, or package, while queue managers are active, the bind fails with a locking problem if any queue manager in the DSG using the plan is active.

      You can either shut down the queue managers using the plans, or suspend the queue managers use of Db2. See Suspending a connection to Db2 for further information.

   b) Perform the additional hold action tasks of binding new and changed DBRMs into plans

   c) Stop and restart each queue manager so that it picks up the new code level.

   d) Perform testing of the new code level.

      These steps can be performed at any time in preparation for a migration to the latest version of IBM MQ for z/OS, or as part of normal maintenance. It is not dependent on the latest version being available.

[13] The "migration and toleration" PTFs are also known as the "backward migration and coexistence" PTFs. They are the same PTFs.

Migrating an earlier version queue manager to a later version queue manager within a queue sharing group is restricted. The restrictions are, that all the earlier version queue managers in the queue-sharing group must have been started at the *same* earlier version, and that all the earlier version queue managers have had the "earlier version backwards migration from the latest version, and coexistence with the latest version" PTFs applied.

After a queue manager for the latest version has been started in a queue-sharing group, starting an earlier version queue manager is restricted. You cannot start an earlier version queue manager as a member of the group, unless it has migration and toleration PTFs applied.

The latest version requires new Db2 tables, and additional changes to existing Db2 tables. The PTF changes some of the Db2 operations performed by the earlier version queue manager. The changes make an earlier version queue manager compatible with the latest version.

The PTF contains a new set of Database Request Modules (DBRM). After binding Db2 with these DBRMs, you have two sets of plans: one set for queue managers without the PTFs and the other set for queue managers with the PTFs applied.

3. Migrate your Db2 tables.

   You must have applied the migration and toleration PTFs to all the queue managers in the queue-sharing group before migrating Db2 tables.

   If the jobs described fail because of a Db2 locking problem, it might be due to contention for a Db2 resource. Locking is more likely, if the system is being heavily used. Resubmit the job later, preferably when the system is lightly used or quiesced.

   You can either migrate the Db2 tables one queue-sharing groups at a time, or all queue-sharing groups at the same time. For more information, read the header information in the jobs.

   - Migrate the tables for all the queue-sharing groups at the same time.

     a. Customize the `CSQ4570T` and `CSQ4571T` samples, in *thlqual*`.SCSQPROC`, supplied with the latest version of the IBM MQ for z/OS product.

        The header information in `CSQ4570T` and `CSQ4571T` describes how to customize the samples.

     b. Run the customized `CSQ4570T` and `CSQ4571T` jobs.

     c. Customize the `CSQ45BPL` and `CSQ45GEX` samples, in *thlqual*`.SCSQPROC`

        The header information in `CSQ45BPL` and `CSQ45GEX` describes how to customize the samples.

     d. Run the customized jobs, `CSQ45BPL` and `CSQ45GEX`.

        The customized jobs are run as part of step “8” on page 87 of “z/OS: Review and modify queue manager customizations from the previous release” on page 84.

        This step binds the latest version DBRMs into plans, and grants execute authority to them.

   - Migrate the tables for one queue-sharing group at a time.

     a. Customize the `CSQ4570T` and `CSQ4571T` samples, in *thlqual*`.SCSQPROC`, supplied with the latest version of the IBM MQ for z/OS product.

        The header information in `CSQ4570T` and `CSQ4571T` describes how to customize the samples.

        Edit step “8” on page 87 of “z/OS: Review and modify queue manager customizations from the previous release” on page 84 that executes the `MIGRATE QSG` function, specifying the name of the first queue-sharing group that is to be migrated.

     b. Run the customized `CSQ4570T` and `CSQ4571T` jobs.

     c. Customize the `CSQ45BPL` and `CSQ45GEX` samples, in *thlqual*`.SCSQPROC`

        The header information in `CSQ45BPL` and `CSQ45GEX` describes how to customize the samples.

     d. Run the customized jobs, `CSQ45BPL` and `CSQ45GEX`.

The customized jobs are run as part of step "8" on page 87 of "z/OS: Review and modify queue manager customizations from the previous release" on page 84.
This step binds the latest version DBRMs into plans, and grants execute authority to them.

Migrating a queue manager cluster

Migrate a queue manager cluster by migrating each of the queue managers in the cluster. No changes since Version 6.0 specifically effect the migration of queue manager clusters. But you must consider what the effect is of migrating some queue managers in a cluster, before all the queue managers are migrated.

Before you begin

Check that no cluster-specific migration issues are identified for the migration you are intending to perform. No cluster-specific changes have been made that affect migration between Version 7.0.1 and later versions.

Procedure

Consider the following issues that relate to migrating a queue manager cluster:

- Minimizing application outages.
- Measuring and verifying migration success and planning for backward migration if there are any migration problems.
- Taking advantage of new IBM MQ features.
- Managing the migration of a cluster in the context of the wider IBM MQ network and the systems architecture of your organization.

Steps

Some techniques for migrating a cluster with minimal planned and unplanned outages are described in the following topics:

Related concepts

"How mixed version cluster repositories are updated" on page 75
Repositories store records for an object in a cluster in the version of the record format that matches the version of the queue manager hosting the repository. Repository queue managers forward object records, before they are stored, in the format that they are received in. The recipient ignores fields from a newer version, and uses default values for fields that are not present in the record.

"Queue manager cluster migration" on page 74
You can migrate queue managers in a cluster all at once, or one at a time, which is called a staged migration. Migrate full repository queue managers in a cluster before partial repository queue managers.

Migrating a queue manager cluster: Create a plan

Before carrying out the migration of a queue manager cluster, plan what you are going to do. Identify the roles that different queue managers play in the cluster, and decide in what order to migrate the queue managers.

Procedure

- What queue manager and application migration issues must be dealt with between the old and new versions?
- What system architecture and change control procedures must you consider?
- Consider migration questions specific to clusters, such as migrating full repositories first, and migrating.
- Are any of the queue managers in a queue-sharing group, or part of a high-availability solution?

- Is the cluster a publish/subscribe cluster? Which queue manager is a cluster topic host?
- Decide whether to carry out a staged migration, or migrate all queue managers at the same time.
- Do you have a test system to migrate, and a production system?
- Document and test the plan before migrating production queue managers.

Related concepts

"Application migration and interoperation" on page 73
IBM MQ supports running applications compiled and linked against previous versions of IBM MQ, with later levels of IBM MQ.

"Migrating a queue manager in a high-availability configuration" on page 76
Follow standard procedures to migrate a queue manager that is part of a high-availability configuration. (On platforms other than z/OS.)

"How mixed version cluster repositories are updated" on page 75
Repositories store records for an object in a cluster in the version of the record format that matches the version of the queue manager hosting the repository. Repository queue managers forward object records, before they are stored, in the format that they are received in. The recipient ignores fields from a newer version, and uses default values for fields that are not present in the record.

"Queue manager cluster migration" on page 74
You can migrate queue managers in a cluster all at once, or one at a time, which is called a staged migration. Migrate full repository queue managers in a cluster before partial repository queue managers.

"Queue manager migration" on page 68

"Queue-sharing group migration" on page 76
You can combine queue managers from different releases in a queue-sharing group. Limit the time you manage a mixed group to only as long as it takes to migrate all the queue managers to the same command level. You cannot combine a queue manager at Version 8.0 or later in the same queue-sharing group as queue managers earlier than Version 7.0.1. You must update all queue managers in a queue-sharing group with a coexistence PTF, before migrating any of them. All queue managers in the queue-sharing group must be at the same version before any queue manager can be migrated to Version 8.0.

Related information

Availability of cluster topic host queue managers

Migrating a queue manager cluster: Create a backout plan

Before performing a migration, decide on a backout plan in case of failure.

Before you begin

What backout capabilities do the queue managers in the cluster support?

z/OS If the libraries of the earlier level of IBM MQ include the appropriate PTFs to be able to backward migrate, and NEWFUNC mode is not enabled at the higher level, queue managers running on z/OS can be reverted to an earlier level by changing the load libraries.

On other platforms, the only backout option is to restore a queue manager to a previous state. In restoring a queue manager, you lose any persistent changes since the queue manager started running at the new level.

About this task

The backout plan must consider how to maintain the availability of the cluster. It must deal with any issues arising from migrating a queue manager in the cluster.

Procedure

The backout plan must describe the following points:

- What constitutes a successful migration.
- The conditions that trigger the backout procedure.
- Alternative backout actions, such as:
  a) Suspending a queue manager from the cluster.
  b) Backward migration
  c) Keeping a queue manager offline until an external issue is resolved.

Related concepts

“Reverting a queue manager to a previous version” on page 70
You can remove an upgrade before you have started a queue manager. After a queue manager has been started, if you remove the upgrade, the queue manager will not work.

Migrating a queue manager cluster: Migrating one cluster queue manager

Follow these steps to migrate a single queue manager in a cluster. Base your cluster migration plan on applying these steps to each queue manager in the cluster.

Procedure

1. Suspend the queue manager that you want to migrate from the cluster:

   a) Issue the **MQSC** command:

   ```
   SUSPEND QMGR CLUSTER(cluster name)
   ```

   b) Check that no messages are sent to the queue manager.

   You must close any application that continues to send messages to this queue manager. The cluster workload algorithm might choose the suspended queue manager. If there are no other valid destinations, or if an application has an affinity with the queue manager, it might select the queue manager.

2. Save a record of all cluster objects known by this queue manager. This data is used after migration to check that objects have been migrated successfully.

   a) Issue the command to view cluster queue managers.

   ```
   DISPLAY CLUSQMGR(*)
   ```

   b) Issue the command to view cluster queues.

   ```
   DISPLAY QC(*)
   ```

   c) Issue the command to view cluster topics.

   ```
   DISPLAY TCLUSTER(*)
   ```

3. Save a record from the full repository of its view of the cluster objects owned by this queue manager. The record is used after migration to check that objects have been migrated successfully.

   a) Issue the command on the full repositories to display this queue manager.

   ```
   DISPLAY CLUSQMGR(migrated queue manager name)
   ```

   b) Issue the command on the full repositories to display the cluster queues for this queue manager

   ```
   DISPLAY QC(*) WHERE(CLUSQMGR EQ migrated queue manager name)
   ```

   c) Issue the command on the full repositories to display the cluster topics for this queue manager.

```
DISPLAY TCLUSTER(*) WHERE(CLUSQMGR EQ migrated queue manager name)
```

4. Migrate the queue manager.

   Do one of the queue manager migration tasks, depending on the platform; see "Migrating a queue manager to the latest release" on page 107.

   The queue manager migration process is, in outline:

   a) Stop the queue manager.
   b) Take a backup of the queue manager.
   c) Install the new version of IBM MQ.
   d) Restart the queue manager.
5. Ensure that all cluster objects have been migrated successfully.
   a) Issue the command to view cluster queue managers and check the output against the data saved before migration.

   ```
   DISPLAY CLUSQMGR(*)
   ```

   b) Issue the command to view cluster queues and check the output against the data saved before migration.

   ```
   DISPLAY QC(*)
   ```

   c) Issue the command to view cluster topics and check the output against the data saved before migration.

   ```
   DISPLAY TCLUSTER(*)
   ```

6. Check that the queue manager is communicating with the full repositories correctly.
7. Check that cluster channels to full repositories can start.
8. Check that the full repositories still have information about the migrated cluster queue manager, its cluster queues, and its cluster topics.
   a) Issue the command on the full repositories and check the output against the data saved before migration.

   ```
   DISPLAY CLUSQMGR(migrated_queue_manager_name)
   ```

   b) Issue the command on the full repositories and check the output against the data saved before migration.

   ```
   DISPLAY QC(*) WHERE(CLUSQMGR EQ migrated_queue_manager_name)
   ```

   c) Issue the command on the full repositories and check the output against the data saved before migration.

   ```
   DISPLAY TCLUSTER(*) WHERE(CLUSQMGR EQ migrated_queue_manager_name)
   ```

9. Test that applications on other queue managers can put messages to queues owned by the migrated cluster queue manager.
10. Test that applications on the migrated queue manager can put messages to the queues owned by other cluster queue managers.
11. Resume the queue manager by issuing the following command:

```
RESUME QMGR CLUSTER(cluster name)
```

12. Closely monitor the queue manager and applications in the cluster for a while.

Related concepts

"Queue manager migration" on page 68

Related information

DISPLAY CLUSQMGR

DISPLAY QUEUE

RESUME QMGR

SUSPEND QMGR

Migrating a queue manager cluster: Migrating the test system

Migrate each queue manager in the test system.

About this task

For each queue manager in the test system, in the order defined in the migration plan you developed in "Migrating a queue manager cluster: Create a plan" on page 161, migrate, and test the queue manager.

Migrating a queue manager cluster: Migrating the production system

Migrate each queue manager in the production system.

About this task

For each queue manager in the production system, in the order defined in the migration plan you developed in "Migrating a queue manager cluster: Create a plan" on page 161, migrate, and test the queue manager.

Windows: Migrating an MSCS configuration

Migrate queue managers in MSCS configuration one node at a time, following these instructions.

About this task

These steps are required for a rolling upgrade with a minimum amount of downtime. You must always upgrade an offline node with no online IBM MQ resources. In an Active/Passive configuration, if the node is Passive, you must ensure it cannot be switched to Active during the upgrade process.

The example, "Migrating a four-node MSCS cluster from an earlier version of the product to the latest version" on page 166, shows this procedure applied to a four-node cluster.

Procedure

1. Modify the possible owners of the IBM MQ resource to encompass only the Active node or nodes. With no owners assigned to Passive nodes, the IBM MQ resource that is being migrated cannot be activated.
2. Ensure that the group containing the IBM MQ resource is currently on one of the nodes defined as a possible owner. The group must include any applications connecting to the queue manager resource.
3. Stop the cluster service on the node being migrated. The MSCS cache is cleared of any IBM MQ DLLs that have been registered.
4. Migrate the selected node by following the standard instructions in "Windows: Migrating a queue manager from a previous version to the latest version" on page 110. Apply the required maintenance level.
5. Start the cluster service on the selected node.
6. On the next node to be migrated, ensure that the IBM MQ resources are offline.

7. Remove this node from the list of possible owners. For clusters with more than two nodes, see the Additional considerations later in this topic.
8. Move the group containing the IBM MQ resource to one of the possible owners and bring it online.
9. Repeat steps 3-8 as necessary for any remaining nodes.

Migrating a four-node MSCS cluster from an earlier version of the product to the latest version

The example in Table 18 on page 167 illustrates the steps involved in migrating a four-node MSCS cluster.

In the example IBM MQ resources include queue managers, applications, and dependant MSCS resources, such as an IP address defined an as MSCS resource. In each step, the changes are italicized.

Step 1

Select the node to migrate and prepare it for upgrading from an earlier version of the product to the latest version.

1. Select node 1 to be migrated and convert it into a Passive node with no running IBM MQ resources.
2. Modify the possible owners of the group containing the IBM MQ resources, to encompass only the required online nodes. Failover does not attempt to switch IBM MQ resources to the node that is not a possible owner. It is safe to migrate that node.
3. Move the group containing the IBM MQ resource to one of the nodes that is a possible owner, and bring it online.
4. Stop the cluster service on the node being migrated. Stopping the service clears the MSCS cache of any IBM MQ libraries that have been registered for MSCS. The node goes offline.

Step 2

Migrate IBM MQ from an earlier version of the product to the latest version

Step 3

Start the cluster service on the selected node. The node becomes online, but it is not a possible owner, so no work is switched to it.

Step 4

Repeat steps 1 - 3 for node 2. Nodes 1 and 2 are now online, and you have migrated them to the latest version. They are still doing no work, as they are not possible owners of any of the IBM MQ resource groups.

Step 5

Migrate the cluster from running an earlier version of the product to the latest version. The number of migrated nodes is now greater or equal to the number of unmigrated nodes.

1. Change the set of possible owners from `3,4` to `1,2`.
2. Move the IBM MQ resource groups from nodes 3 and 4 to nodes 1 and 2 and bring online.
3. From this point onward, the list of possible owners must include migrated nodes only. The IBM MQ resource must never failover to a node running a back level version of the product.

Note: If you must revert IBM MQ to an earlier version, the IBM MQ resources must be removed from MSCS control, before performing an uninstallation of IBM MQ

Step 6

Migrate node 3 to the latest version.

1. Follow steps 1 - 3 for node 3.
2. Add node 3 to the list of possible owners.
3. Move the QMC resource group back from node 1 to node 3 and bring online again.

Step 7

Repeat step 6 for node 4.

Table 18. Migrating a four-node MSCS cluster

Steps		**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Node 1	State	Online	*Offline*	Offline	*Online*	Online	Online	Online	Online
	Version	Earlier version	Earlier version	*Latest version*	Latest version	Latest version	Latest version	Latest version	Latest version
	Groups	QMA					*QMC, QMA*	*QMA*	QMA
Node 2	State	Online	Online	Online	Online	Online	Online	Online	Online
	Version	Earlier version	Earlier version	Earlier version	Earlier version	*Latest version*	Latest version	Latest version	Latest version
	Groups	QMB	QMB	QMB	QMB		*QMD, QMB*	QMD, QMB	*QMB*
Node 3	State	Online	Online	Online	Online	Online	Online	Online	Online
	Version	Earlier version	Earlier version	Earlier version	Earlier version	Earlier version	Earlier version	*Latest version*	Latest version
	Groups	QMC	QMC, QMA	QMC, QMA	QMC, QMA	QMC, QMA		*QMC*	QMC
Node 4	State	Online	Online	Online	Online	Online	Online	Online	Online
	Version	Earlier version	Earlier version	Earlier version	Earlier version	Earlier version	Earlier version	Earlier version	*Latest version*
	Groups	QMD	QMD	QMD	QMD	QMD, QMB			*QMD*
Possible Owners		1,2,3,4	*2,3,4*	2,3,4	2,3,4	*3,4*	*1,2*	*1,2,3*	*1,2,3,4*
Task			Update 1			Update 2	Transfer	Update 3	Update 4

What to do next

Additional considerations in an MSCS setup with more than 2 nodes: A cluster might contain enough nodes for you to form a group of migrated queue managers and a group of unmigrated nodes. Switch to the migrated group when it contains half the number of queue managers. Before you have reached the half way point, the unmigrated group are possible owners. When you reach the half way point, switch the possible owners to the migrated group.

Related concepts

“Migrating a queue manager in a high-availability configuration” on page 76
Follow standard procedures to migrate a queue manager that is part of a high-availability configuration. (On platforms other than z/OS.)

Related reference

“ Windows: MSCS restriction with multiple installations” on page 213

When you install or upgrade to Version 7.1 or later, the first installation of the product on the server is the only one that can be used with Microsoft Cluster Server (MSCS). No other installations on the server can be used with MSCS. This restriction limits the use of MSCS with multiple installations of the product.

Migrating from a single instance to a multi-instance queue manager

To migrate a single instance queue manager to a multi-instance queue manager, you must move the queue manager data to a shared directory, and reconfigure the queue manager on two other servers.

Before you begin

You must check the prerequisites for running a multi-instance queue manager as part of this task. Some environments have been tested with multi-instance queue managers, and are known to work. They are AIX, Red Hat Linux, SUSE Linux Enterprise Server, HP-UX with the file system on Linux Red Hat, IBM i, and Windows Server. See Testing and support statement for IBM MQ multi-instance queue managers for the latest list of tested environments. The support statement has detailed version and prerequisite information for each environment it lists. Other environments might work; a test tool is provided with IBM MQ to assist you in qualifying other environments.

You must have three servers to run a multi-instance queue manager. One server has a shared file system to store the queue manager data and logs. The other servers run the active and standby instances of the queue manager.

About this task

You have a single-instance queue manager that you want to convert to a multi-instance queue manager. The queue manager conversion itself is straightforward, but you must do other tasks to create a fully automated production environment.

You must check the prerequisites for a multi-instance queue manager, set up the environment and check it. You must set up a monitoring and management system to detect if the multi-instance queue manager has failed and been automatically restarted. You can then find out what caused the restart, remedy it, and restart the standby. You must also modify applications, or the way applications are connected to the queue manager, so that they can resume processing after a queue manager restart.

Procedure

1. Check the operating system that you are going to run the queue manager on, and the file system on which the queue manager data and logs are stored on. Check that they can run a multi-instance queue manager.
   a) Consult Testing and support statement for IBM MQ multi-instance queue managers. See whether the combination of operating system and file system is capable of running a multi-instance queue manager.

      A shared file system must provide lease-based locking to be adequate to run multi-instance queue managers. Lease-based locking is a recent feature of some shared file systems, and in some case fixes are required. The support statement provides you with the essential information.

   b) Run **amqmfsck** to verify that the file system is configured correctly.

      File systems are sometimes configured with performance at a premium over data integrity. It is important to check the file system configuration. A negative report from the **amqmfsck** tool tells you the settings are not adequate. A positive result is an indication that the file system is adequate, but the result is not a definitive statement that the file system is adequate. It is a good indication.

   c) Run the integrity checking application provided in the technote, Testing a shared file system for compatibility with IBM MQ Multi-instance Queue Managers.

The checking application tests that the queue manager is restarting correctly.

2. Configure a user and group to be able to access a share on the networked file system from each server that is running a queue manager instance.

   On Windows, the security IDs (SIDs) of the `mqm` group can be different; see Windows domains and multi-instance queue managers.
   On UNIX and Linux, the `uid` and `gid` for `mqm` in `/etc/password` must be the same on each system; see Create a multi-instance queue manager on Linux .

   IBM i On IBM i, `QMQM`, `QMQMADM`, and any other user profiles that are granted access to the share must have the same passwords on all the servers.

3. Set up a directory for the share on the networked file system with the correct access permissions.

   A typical configuration is to set up a single shared directory that contains all data and log directories for all queue managers that use the shared disk; see Share named qmgrs and log directories (Version 7.0.1 onwards) in Example directory configurations on UNIX systems.

   For example, create a root directory on the share called `MQHA` that has subdirectories `data` and `logs`. Each queue manager creates its own data and log directories under `data` and `logs`. Create `MQHA` with the following properties:
   On Windows, create *`drive`*`\MQHA` on the shared drive. The owner is a member of `mqm`. `mqm` must have full-control authority. Create a share for *`drive`*`\MQHA`.
   On UNIX, create `/MQHA` on the shared drive. `/MQHA` is owned by the user and group `mqm` and has the access permissions `rwx`.
   If you are using an NFS v4 file server, add the line `/MQHA * rw,sync,no_wdelay,fsid=0)` to `etc/exports`, and then start the NFS daemon: `/etc/init.d/nfs start`.

   IBM i On IBM i, follow the instructions to create a network share using NetServer.

4. Copy the queue manager data and the logs to the share.

   You can choose to copy files manually, by following the procedure to back up the queue manager. On Windows, you can run the **hamvmqm** command to move the queue manager data to the share. The **hamvmqm** command works for queue managers created before Version 7.0.1, and not reconfigured with a datapath, or for queue managers that do not have a **`DataPath`** configuration attribute. Choose one of these methods:

   - Follow the instructions in Backing up queue manager data, IBM i or Backups of IBM MQ for IBM i data, copying the queue manager data to the share. You must use this method if the **`DataPath`** configuration attribute is specified for this queue manager.
   - Stop the queue manager, and then type the command,

     ```
     hamvmqm /m /dd share\data /dd share\logs
     ```

     Where *share* is to be the location of the data and logs that you created in step “3” on page 169.

5. Update the queue manager configuration information stored on the current queue manager server.

   If you moved the queue manager data and logs by running the **hamvmqm** command, the command has already modified the configuration information correctly for you.
   If you moved the queue manager data and logs manually, you must complete the following steps.

   - On Windows:

     a. Modify the log registry key:

     ```
     HKEY_LOCAL_MACHINE\SOFTWARE\IBM\WebSphere
     MQ\Installation\MQ_INSTALLATION_NAME\Configuration\QueueManager\QMgrName\Log
     "LogPath"="share\\logs\\QMgrName\\"
     ```

     b. Modify the Prefix registry key:

```
HKEY_LOCAL_MACHINE\SOFTWARE\IBM\WebSphere
MQ\Installation\MQ_INSTALLATION_NAME\Configuration\QueueManager\QMgrName
"Prefix"="share\\data"
```

- On IBM i IBM i, UNIX, and Linux,

  a. Modify the `Log:` stanza in the queue manager `qm.ini` file, which is on the *share*:

  ```
  LogPath=share/logs/QMgrName
  ```

  b. Modify the `QueueManager:` stanza in the IBM MQ `mqs.ini` file, which is typically in the `/var/mqm` directory on UNIX and Linux IBM i, and `/QIBM/UserData/mqm` on IBM i:

  ```
  DataPath=share/data/QMgrName
  ```

  Where, *QMgrName* is the representation of the queue manager name in the existing registry key on Windows. *QMgrName* is the `Directory` name in the `QueueManager:` stanza in the `mqs.ini` file on IBM i IBM i, UNIX, and Linux. *share* is share where the data and logs are moved to.

6. Add the queue manager configuration information to the new queue manager server.

   a) Run the **dspmqinf** command to display the queue manager information

   Run the command on the server that ran the queue manager in Version 6.0.

   ```
   dspmqinf -o command QMgrName
   ```

   The command output is formatted ready to create a queue manager configuration.

   `addmqinf -s QueueManager -v Name= QMgrName -v Directory= QMgrName -v Prefix=d:\var\mqm Datapath= \share\data\QMgrName`

   b) Create a queue manager configuration on the other server.

   Run the **addmqinf** command copied from the previous output

7. Add the network address of the new server to the connection name in client and channel definitions.

   a) Find all the client, sender, and requester TCPIP settings that refer to the server.

   Client settings might be in Client Definition Tables (CCDT), in environment variables, in Java properties files, or in client code.
   Cluster channels automatically discover the connection name of a queue manager from its cluster receiver channel. As long as the cluster receiver channel name is blank or omitted, TCPIP discovers the IP address of the server hosting the queue manager.

   b) Modify the connection name for each of these connections to include the TCPIP addresses of both servers that are hosting the multi-instance queue manager.

   For example, change:

   ```
   echo DISPLAY CHANNEL(ENGLAND) CONNAME | runmqsc QM1
   ```

   ```
   5724-H72 (C) Copyright IBM Corp. 1994, 2022. ALL RIGHTS RESERVED.
   Starting MQSC for queue manager QM1.
   1: DISPLAY CHANNEL(ENGLAND) CONNAME
   AMQ8414: Display Channel details.
   CHANNEL(ENGLAND) CHLTYPE(SDR)
   CONNAME(LONDON)
   ```

Into:

```
echo ALTER CHANNEL(ENGLAND) CHLTYPE(SDR) CONNAME('LONDON, BRISTOL') | runmqsc QM1
```

8. Update your monitoring and management procedures to detect the queue manager restarting.
9. Update client applications to be automatically reconnectable, if appropriate.
10. Update the start procedure for your IBM MQ applications to be started as queue manager services.
11. Start each instance of the queue manager, permitting them to be highly available.

    The first instance of the queue manager that is started becomes the active instance.
    Issue the command twice, once on each server.

```
strmqm -x QMgrName
```

What to do next

To get the highest availability out of multi-instance queue managers, you must design client applications to be reconnectable and server applications to be restartable; see Application recovery.

Related information
amqmfsck (file system check)
Application recovery
Automatic client reconnection
Backing up queue manager data
Channel and client reconnection
Changing configuration information on Windows, UNIX and Linux systems
Create a multi-instance queue manager on Linux
Moving a queue manager to MSCS storage
Multi-instance queue managers
IBM i Multi-instance queue managers on IBM i
Queue manager configuration files, qm.ini
Shared file system
Testing a shared file system for compatibility with IBM MQ Multi-instance Queue Managers
Testing and support statement for IBM MQ multi-instance queue managers
The IBM MQ configuration file, mqs.ini
IBM i The IBM MQ configuration file mqs.ini - IBM i
Verifying shared file system locking
Windows domains and multi-instance queue managers
Working with services

Reverting to a single-instance queue manager

Revert a multi-instance queue manager to a single instance queue manager by stopping the standby instance. Then restart the active instance and do not set the flag that permits standby instances.

Before you begin

You have at least three servers configured to run a queue manager as a multi-instance queue manager. The queue manager is currently running as a multi-instance queue manager, with one standby instance active.

About this task

The task involves deactivating the active standby so that only the running multi-instance queue manager remains active. To prevent a standby instance being started in the future, you must stop the active instance and restart it. When you restart it, you start it as a single instance queue manager that prevents standby instances being started. The standby instance is stopped as a separate step, to give you the option of restarting the active instance at a later date. You can stop both instances by running the standard `endmqm` *`QMgrName`* command on the server running the active queue manager.

Procedure

1. Stop the standby queue manager instance.

   On the server running the standby instance:

   - Windows, UNIX, and Linux

     ```
     endmqm -w QMgrName
     ```

   - IBM i IBM i

     ```
     ENDMQM MQMNAME (QMgrName) *WAIT
     ```

2. Stop the active queue manager instance.

   On the server running the active instance:

   - Windows, UNIX, and Linux

     ```
     endmqm -w (QMgrName)
     ```

   - IBM i IBM i

     ```
     ENDMQM MQMNAME (QMgrName) *WAIT
     ```

3. Restart the queue manager, preventing standbys.

   On the server that is going to run the queue manager:

   - Windows, UNIX, and Linux

     ```
     strmqm QMgrName
     ```

   - IBM i IBM i

     ```
     STRMQM MQMNAME (QMgrName)
     ```

What to do next

You might want to run the queue manager as a single instance on the same server as the queue manager data.

When the queue manager is stopped move the queue manager data back to the server that is running the queue manager. Alternatively install IBM MQ, and then move the queue manager configuration definition onto the server with the queue manager data. Both tasks are variations of steps in "Migrating from a single instance to a multi-instance queue manager" on page 168 to create a multi-instance queue manager.

Changes that affect migration

Changes are listed that affect the migration of a queue manager to the current release of IBM MQ, or that affect existing applications or configurations.

Review the list of changes before upgrading queue managers to the latest product version. Decide whether you must plan to make changes to existing applications, scripts, and procedures before starting to migrate your systems.

Note:

IBM i distributed On distributed platforms, you cannot reverse queue manager migration to remove the effect of changes.

z/OS On z/OS, you can reverse queue manager migration as long as you have not enabled new function. You enable new function by setting the **OPMODE** parameter to `(NEWFUNC, 800)`.

Changes for IBM MQ Version 8.0

- Command changes: **`dmpmqcfg`** output
- Command changes: results displayed in response to DISPLAY CONN command from **`runmqsc`**
- Command level: changes
- Solaris Deprecation: linking with libmqmcs and libmqmzse libraries
- Java and JMS: Changes to CipherSuite support
- Java: changes for Java 7
- Java: Changes to IBM MQ classes for Java
- Java: Changes to MQException and MQDataException class logging
- V 8.0.0.3 Extended start events for multi-instance queue managers
- JMS: changes to JAR files for JMS 2.0
- JMS: changes to IBM MQ classes for JMS
- Publish/subscribe: changes to DISPLAY TOPIC and DISPLAY TPSTATUS output
- Publish/subscribe: changes to how the subscription selection string is evaluated
- Queue managers: change to default TCP buffer size
- Queue managers: Version attribute added for queue managers in a cluster
- Security: new CONNAUTH CHCKLOCL parameter
- Security: Disable AMS at the client
- Security: reduction in channel send exit buffer space for SSL and TLS
- Solaris client: Change in method of installation
- z/OS z/OS: changes to log RBA and URID lengths
- z/OS z/OS: WLM/DNS no longer supported

For details of all changes in this release, including those that affect migration, see What's changed in IBM MQ 8.0 .

Changes in earlier releases

For information about what changed in an earlier release of the product, see the *What's changed* section in the product documentation for that release:

- What's changed in IBM WebSphere MQ Version 7.5
- What's changed in IBM WebSphere MQ Version 7.1

For older, out of support, versions of the product, the documentation is not available in the online IBM Documentation, but is available for you to download for offline use. For more information, see Documentation for older versions of IBM MQ.

Related concepts

"Introduction to IBM MQ migration" on page 19

Coexistence

Queue managers, with different names, can coexist on any server as long as they use the same IBM MQ installation. On z/OS z/OS, UNIX, Linux, and Windows, different queue managers can coexist on the same server and be associated with different installations. In addition to queue managers coexisting on a server, objects, and commands must work correctly with different queue managers running at different command levels.

Multiple queue manager versions in z/OS

z/OS

There can be several IBM MQ subsystems in a z/OS image, and they can use different versions of IBM MQ, provided that the IBM MQ early code modules are of the latest version being used. (These modules are loaded at z/OS IPL time and are shared among all the IBM MQ subsystems in the z/OS image.)

This means that you can run one queue manager at the latest version and another in the same image with an earlier version, provided that the early code is that of the latest version.

The coexistence section lists restrictions in the use of objects and commands when they are used with queue managers at multiple command levels. The queue managers might be running on a single server, or in a cluster.

Related concepts

"Queue manager coexistence in Version 8.0" on page 53
Queue managers, with different names, can coexist on any server as long as they use the same IBM MQ installation. On z/OS z/OS, UNIX, Linux, and Windows, different queue managers can coexist on the same server and be associated with different installations.

"Multi-installation queue manager coexistence on UNIX, Linux, and Windows" on page 57
You can install multiple copies of IBM MQ for UNIX, Linux, and Windows on the same server. The installations must be at Version 7.1 or later, with one exception. One Version 7.0.1 installation, at fix pack level 6, or later, can coexist with multiple Version 7.1, or later installations.

Related tasks

"Migrating IBM MQ library loading from an earlier version of the product to the latest version" on page 131
No change in the way IBM MQ libraries are loaded is normally required if you upgrade from an earlier version of the product to the latest version. You must have followed the instructions on building IBM MQ applications in Version 7.0.1 and you must replace IBM WebSphere MQ 7.0.1 with the latest version of the product. If you choose to take advantage of multi-installation in the latest version of the product, based on the side-by-side or multi-stage migration scenarios, you must modify the environment for the operating system to resolve IBM MQ dependencies for an application. Typically, you can modify the runtime environment, rather than relink the application.

JMS: Administered objects

Version 6.0 JMS administered objects, such as connection factories and destinations are compatible with later releases.

JMS applications can use connection factory or destination objects created in IBM WebSphere MQ 6.0. Any new properties, that did not exist in Version 6.0, assume their default value.

IBM WebSphere MQ 6.0 JMS applications can use connection factory or destination objects created in later releases. Any of the new properties that do not exist in Version 6.0 are ignored.

Mixed version cluster coexistence

A cluster can contain queue managers running at IBM MQ Version 8.0, and any currently supported level of IBM MQ. However new features cannot be exploited from queue managers at an earlier level.

Cluster workload balancing in a mixed version cluster

IBM MQ Version 7.1 added a new **DEFBIND** value called GROUP to queues. Applications on queue managers earlier than Version 7.1 must not open or put messages to queues specifying the new value. When an application ignores this limitation, the workload balancing behavior (for example: BIND_ON_OPEN or BIND_NOT_FIXED) is undefined.

Routing behavior in a mixed version publish/subscribe cluster

As described in What's new in IBM MQ 8.0 , this version adds topic host routing for publish/subscribe clusters. As for all new clustered behavior, the queue manager where the object is defined, and the full repository queue managers, must be at a level that supports the new feature. Any queue manager in the cluster that is at an earlier level does not adhere to the new behavior.

When a clustered topic is defined for topic host routing (by setting the topic **CLROUTE** parameter to TOPICHOST), only queue managers at the new level are aware of the clustered topic. Older queue managers do not receive the clustered topic definition and therefore behave as if the topic is not clustered. This means that all queue managers that need to work in a routed publish/subscribe manner must be at a version that supports this feature, not just the queue managers that host the routed topics.

Important notes:

- All full repositories must be at Version 8.0 to use this feature. If a full repository queue manager is at an earlier version, the **CLROUTE** of TOPICHOST is not recognized by the full repository, and the full repository propagates the topic definition to all queue managers in the cluster. Any pre- Version 8.0 queue managers then use the topic as if it is defined for DIRECT routing. This behavior is unsupported.
- If an older queue manager defines a direct routed clustered topic with the same name as an existing topic host routed clustered topic, the full repositories notice the conflicting definition and do not propagate the definition.

To find out the version of each queue manager in the cluster, specify the VERSION parameter with the DISPLAY CLUSQMGR command. If you issue this command from a queue manager with a full repository, the information returned applies to every queue manager in the cluster. Otherwise the information returned applies only to the queue managers in which it has an interest. That is, every queue manager to which it has tried to send a message and every queue manager that holds a full repository.

z/OS: ISPF operations and control panels

When using the operations and control panels, the IBM MQ libraries you use in ISPF must be compatible with the queue manager you are working with.

Table 19 on page 176 shows which versions of the operations and controls panels you use in ISPF are compatible with which levels of queue manager. Version 7.1 panels are incompatible with any release before Version 6.0.

Table 19. Compatibility of queue manager versions with operations and control panel versions

Version	Queue-sharing group containing a mixture of Version 7.0.1 Version 7.1, and Version 8.0 queue managers.	Version 8.0 queue manager	Version 7.1 queue manager	Version 7.0.1 queue manager	Version 7.0 queue manager
Version 8.0 panel	Compatible with restrictions and warnings	Compatible	Compatible with restrictions and warnings		
Version 7.1 panel	Not compatible		Compatible	Compatible with restrictions and warnings	
Version 7.0.1 panel	Not compatible		Compatible with restrictions and warnings	Compatible	
Version 7.0 panel	Not compatible	Not compatible	Compatible with restrictions and warnings	Compatible	

z/OS z/OS: Queue sharing group coexistence

A queue-sharing group can contain queue managers running on IBM WebSphere MQ 7.0.1, and on later releases. The queue managers can access the same shared queues and other shared objects. Queue managers running earlier versions of the product must have the coexistence PTF applied for the latest release.

Note: After the coexistence PTF has been applied, the earlier version queue managers must be started at least once.

A queue-sharing group containing a latest version queue manager, can contain queue managers only at one other, earlier version. You can not start a latest version queue manager in a queue-sharing group that contains other queue managers at more than one other version.

Only run queue managers in a mixed-version queue-sharing group for the time it takes to migrate all queue managers to the later version. If the queue-sharing group contains queue managers with a mixture of versions, new functions on the latest version, that are restricted by OPMODE, will not be available.

z/OS: Properties of objects in a mixed queue-sharing group

Attributes that did not exist in earlier versions can be created and altered on queue managers of a later version in a mixed queue-sharing group. The attributes are not available to queue managers in the group that are at an earlier level.

Any **QSGDISP**(GROUP) TOPIC objects having the **CLROUTE**(TOPICHOST) attribute set, and any **QSGDISP**(GROUP) AUTHINFO objects with **AUTHTYPE**(IDPWOS), are hidden from queue managers earlier than Version 8.0 in a mixed queue-sharing group.

z/OS: MQSC commands in a mixed queue-sharing group

Existing **MQSC** commands using new keywords and attribute values can be entered for routing to a migrated queue manager. You can enter the commands on any queue manager. Route the commands using **CMDSCOPE**. Commands with new keywords and attribute values, or new commands, routed to a previous version of queue manager, fail.

Changes from Version 7.0.1 or later to Version 8.0

All IBM MQ changes that affect the migration of a queue manager from Version 7.0.1 or later to Version 8.0 are listed.

New capabilities that do not affect existing IBM MQ applications are not listed.

Note:

IBM i distributed On distributed platforms, you cannot reverse queue manager migration to remove the effect of changes.

z/OS On z/OS, you can reverse queue manager migration as long as you have not enabled new function. You enable new function by setting the **OPMODE** parameter to `(NEWFUNC, 800)`.

Click on a link to read the details about the change.

Related reference

"z/OS: OPMODE" on page 272
The availability of new functions and backward migration for IBM MQ for z/OS is controlled by the **OPMODE** parameter in the **CSQ6SYSP** macro. To access V8.0 capabilities, change the value of **OPMODE** to `OPMODE=(NEWFUNC,800)`. To restrict the use of new capabilities, and retain the ability to revert the queue manager to its earlier level, leave **OPMODE** at its default setting, `OPMODE=(COMPAT,800)`.

Version 7.0.1 features and their supported APIs

Use this information to learn about the IBM WebSphere MQ 7.0.1 APIs with features and environments that might not be as fully supported as the C MQI.

The tables in this section show supported IBM WebSphere MQ 7.0.1 APIs. Use the information in these tables in conjunction with the information in the system requirements web pages. For links to system requirements information for all releases of IBM WebSphere MQ or IBM MQ, see the System Requirements for IBM MQ web page.

From IBM MQ 8.0, you can use the Software Product Compatibility Reports (SPCR) tool to find information on supported operating systems, system requirements, prerequisites, and optional supported software. For more information about the SPCR tool and links to reports for each supported platform, see the System Requirements for IBM MQ Version 8.0 web page.

Version 7.0.1 features and their supported APIs: JMS

Use this information to learn about the IBM WebSphere MQ 7.0.1 APIs with features and environments that might not be as fully supported as the C MQI.

JMS

The following tables show which features and environments are supported for JMS.

Table 20. JMS: API support											
	JMS										
	Distributed platforms				**z/OS**						
API	**Basic“1” on page 179**	**WAS and other Java EE“2” on page 179**	**Other TM**	**Db2 SP“3” on page 179**	**Basic**	**WAS“2” on page 179**	**Other Java EE**	**Db2 SP“3” on page 179**	**Other TM**	**CICS**	**IMS**
V7 Publish Subscribe	Y	Y	Y	N/A“4” on page 179	Y	Y	N/A	N/A“4” on page 179		N/A	N/A
Browse with Mark	N	N	N	N/A“4” on page 179	N	N	N/A	N/A“4” on page 179		N/A	N/A
Message Properties	Y“5” on page 179	Y“5” on page 179	Y“5” on page 179	N/A“4” on page 179	Y“5” on page 179	Y“5” on page 179	N/A	N/A“4” on page 179	Y“5” on page 179	N/A	N/A
Message Selectors	Y	Y	Y	N/A“4” on page 179			N/A	N/A“4” on page 179		N/A	N/A
Asynchronous consume	Y	Y	Y	N/A“4” on page 179			N/A	N/A“4” on page 179		N/A	N/A
Content filtering Publish / Subscribe	N	N	N	N/A“4” on page 179	N	N	N/A	N/A“4” on page 179	N	N/A	N/A
Group/ Segment messages	N	N	N	N/A“4” on page 179	N	N	N/A	N/A“4” on page 179	N	N/A	N/A
PCF Classes	Y“6” on page 179	Y“6” on page 179	Y“6” on page 179	N/A“4” on page 179	Y“6” on page 179	Y“6” on page 179	N/A	N/A“4” on page 179	Y“6” on page 179	N/A	N/A
Global TX Participant	Y“7” on page 179	Y	Y“7” on page 179	N/A“4” on page 179	Y“8” on page 179	Y	N/A	N/A“4” on page 179	Y“8” on page 179	N/A	N/A

Table 20. JMS: API support (continued)

	JMS										
	Distributed platforms				z/OS						
API	**Basic“1” on page 179**	**WAS and other Java EE“2” on page 179**	**Other TM**	**Db2 SP“3” on page 179**	**Basic**	**WAS“2” on page 179**	**Other Java EE**	**Db2 SP“3” on page 179**	**Other TM**	**CICS**	**IMS**
Global TX Coordinator	N	N	N	N/A“4” on page 179	N	N	N/A	N/A	N	N/A	N/A

Note:

1. Also OSGI.
2. WebSphere Application Server.
3. Db2 Stored Procedures.
4. Db2 SP Distributed: cannot work reliably because of the restricted Java environment.
5. Cannot see the whole MQI namespace, therefore an MQI application might generate messages whose properties cannot be read.
6. Can build/parse messages but not use the MessageAgent classes and methods.
7. Open source JTA coordinators might not be supported.
8. RRS is the coordinator.

Table 21. JMS: Client communication (not running on top of C client)

	JMS										
	Distributed platforms				z/OS						
Client communication (not running on top of C client)	**Basic“1” on page 180**	**WAS and other Java EE“2” on page 180**	**Other TM**	**Db2 SP“3” on page 180**	**Basic**	**WAS“2” on page 180**	**Other Java EE**	**Db2 SP“3” on page 180**	**Other TM**	**CICS**	**IMS**
Read ahead	Y	Y	Y	N/A	N/A	Y	N/A	N/A	N/A	N/A	N/A
Asynchronous put	Y	Y	Y	N/A	N/A	Y	N/A	N/A	N/A	N/A	N/A
Pre-connect exit	N	N	N	N/A	N/A	N	N/A	N/A	N/A	N/A	N/A
CCDT	Y	Y	Y	N/A	N/A	Y	N/A	N/A	N/A	N/A	N/A

Table 21. JMS: Client communication (not running on top of C client) (continued)

	JMS										
	Distributed platforms				**z/OS**						
Client communication (not running on top of C client)	**Basic“1” on page 180**	**WAS and other Java EE“2” on page 180**	**Other TM**	**Db2 SP“3” on page 180**	**Basic**	**WAS“2” on page 180**	**Other Java EE**	**Db2 SP“3” on page 180**	**Other TM**	**CICS**	**IMS**
Multiple CONNAME	Y	Y	Y	N/A	N/A	Y	N/A	N/A	N/A	N/A	N/A
Auto reconnect	Y	Y“5” on page 180	Y	N/A	N/A	Y“5” on page 180	N/A	N/A	N/A	N/A	N/A
Channel compression	Y	Y	Y	N/A	N/A	Y	N/A	N/A	N/A	N/A	N/A
Exit chaining	Y	Y	Y	N/A	N/A	Y	N/A	N/A	N/A	N/A	N/A
MQCSP / MQXR_SEC_PARMS	Y“4” on page 180	Y“4” on page 180	Y“4” on page 180	N/A	N/A	Y“4” on page 180	N/A	N/A	N/A	N/A	N/A
Native SSL	Y	Y	Y	N/A	N/A	Y	N/A	N/A	N/A	N/A	N/A

Note:

1. Also OSGI.
2. WebSphere Application Server.
3. Db2 Stored Procedures.
4. Not consistent across all clients.
5. Use RA features for reconnect, not the FAP-level operation.

Related information

IBM MQ resource adapter V8.0 statement of support

Running IBM MQ classes for Java applications within Java Platform, Enterprise Edition

Version 7.0.1 features and their supported APIs: Java

Use this information to learn about the IBM WebSphere MQ 7.0.1 APIs with features and environments that might not be as fully supported as the C MQI.

Java

The following tables show which features and environments are supported for Java.

Table 22. Java: API support

	Base Java									
	Distributed platforms				**z/OS**					
API	**Basic“1” on page 182**	**WAS and other Java EE“2” on page 182**	**Other TM**	**Db2 SP“3” on page 182**	**Basic**	**WAS and other Java EE“2” on page 182**	**Db2 SP“3” on page 182**	**Other TM**	**CICS**	**IMS**
V7 Publish Subscribe	Y	N/A	N/A	N/A“6” on page 182	Y	N/A	N/A“6” on page 182	N/A	Y	N/A
Browse with Mark	Y	N/A	N/A	N/A“6” on page 182	Y	N/A	N/A“6” on page 182	N/A	Y	N/A
Message Properties	N	N/A	N/A	N/A“6” on page 182	N	N/A	N/A“6” on page 182	N/A	N	N/A
Message Selectors	N	N/A	N/A	N/A“6” on page 182	N	N/A	N/A“6” on page 182	N/A	N	N/A
Asynchronous consume	N	N/A	N/A	N/A“6” on page 182	N	N/A	N/A“6” on page 182	N/A	N	N/A
Content filtering Publish/ Subscribe	N	N/A	N/A	N/A“6” on page 182	N	N/A	N/A“6” on page 182	N/A	N	N/A
Group/ Segment messages	Y	N/A	N/A	N/A“6” on page 182	Y	N/A	N/A“6” on page 182	N/A	Y	N/A
PCF Classes	Y	N/A	N/A	N/A“6” on page 182	Y	N/A	N/A“6” on page 182	N/A	Y	N/A

Table 22. Java: API support (continued)

	Base Java									
	Distributed platforms				z/OS					
API	**Basic“1” on page 182**	**WAS and other Java EE“2” on page 182**	**Other TM**	**Db2 SP“3” on page 182**	**Basic**	**WAS and other Java EE“2” on page 182**	**Db2 SP“3” on page 182**	**Other TM**	**CICS**	**IMS**
Global TX Participant	N	N/A	N/A	N/A“6” on page 182	N	N/A	N/A“6” on page 182	N/A	N	N/A
Global TX Coordinator	Y	N/A	N/A	N/A“6” on page 182	N/A“5” on page 182	N/A	N/A	N/A	N/A	N/A

Note:

1. Also OSGI.
2. WebSphere Application Server.
3. Db2 Stored Procedures.
4. While it might work (some customers have used this environment), it is not recommended or fully supported.
5. RRS is the coordinator.
6. Db2 SP on Distributed platform cannot work reliably because of the restricted Java environment.

Table 23. Java: Client communication (not running on top of C client)

	Base Java									
	Distributed platforms				z/OS					
Client communication (not running on top of C client)	**Basic“1” on page 183**	**WAS and other Java EE“2” on page 183**	**Other TM**	**Db2 SP“3” on page 183**	**Basic**	**WAS and other Java EE“2” on page 183**	**Db2 SP“3” on page 183**	**Other TM**	**CICS**	**IMS**
Read ahead	Y	N/A	N/A	N/A	N/A“5” on page 183	N/A	N/A	N/A	N/A	N/A
Asynchronous put	Y	N/A	N/A	N/A	N/A“5” on page 183	N/A	N/A	N/A	N/A	N/A
Pre-connect exit	N	N/A	N/A	N/A	N/A“5” on page 183	N/A	N/A	N/A	N/A	N/A
CCDT	Y	N/A	N/A	N/A	N/A“5” on page 183	N/A	N/A	N/A	N/A	N/A

Table 23. Java: Client communication (not running on top of C client) (continued)

	Base Java									
	Distributed platforms				**z/OS**					
Client communication (not running on top of C client)	**Basic**“1” on page 183	**WAS and other Java EE**“2” on page 183	**Other TM**	**Db2 SP**“3” on page 183	**Basic**	**WAS and other Java EE**“2” on page 183	**Db2 SP**“3” on page 183	**Other TM**	**CICS**	**IMS**
Multiple CONNAME	N	N/A	N/A	N/A	N/A“5” on page 183	N/A	N/A	N/A	N/A	N/A
Auto reconnect	N	N/A	N/A	N/A	N/A“5” on page 183	N/A	N/A	N/A	N/A	N/A
Channel compression	Y	N/A	N/A	N/A	N/A“5” on page 183	N/A	N/A	N/A	N/A	N/A
Exit chaining	Y	N/A	N/A	N/A	N/A“5” on page 183	N/A	N/A	N/A	N/A	N/A
MQCSP/MQXR_SEC_PARMS	Y“4” on page 183	N/A	N/A	N/A	N/A“5” on page 183	N/A	N/A	N/A	N/A	N/A
Native SSL	Y	N/A	N/A	N/A	N/A“5” on page 183	N/A	N/A	N/A	N/A	N/A

Note:

1. Also OSGI.
2. WebSphere Application Server.
3. Db2 Stored Procedures.
4. Not consistent across all clients.
5. Java client not supported in any environment on z/OS

Related information

IBM MQ resource adapter V8.0 statement of support

Running IBM MQ classes for Java applications within Java Platform, Enterprise Edition

Version 7.0.1 features and their supported APIs: XMS .NET

Use this information to learn about the IBM WebSphere MQ 7.0.1 APIs with features and environments that might not be as fully supported as the C MQI.

XMS .NET

The following tables show which features and environments are supported for XMS .NET.

Table 24. XMS .NET: API support

	XMS .NET				
	Distributed platforms				
API	**Basic**“1” on page 184	**WAS**“2” on page 184	**Other Java EE**	**Other TM**	**Db2 SP**“3” on page 184
V7 Publish Subscribe	Y	N/A	N/A	N/A	N/A
Browse with Mark	N	N/A	N/A	N/A	N/A
Message Properties	Y“4” on page 184	N/A	N/A	N/A	N/A
Message Selectors	Y	N/A	N/A	N/A	N/A
Asynchronous consume	Y	N/A	N/A	N/A	N/A
Content filtering Publish/ Subscribe	N	N/A	N/A	N/A	N/A
Group/Segment messages	N	N/A	N/A	N/A	N/A
PCF Classes	N	N/A	N/A	N/A	N/A
Global TX Participant	Y“5” on page 184	N/A	N/A	N/A	N/A
Global TX Coordinator	N	N/A	N/A	N/A	N/A

Note:

1. Also OSGI.
2. WebSphere Application Server.
3. Db2 Stored Procedures.
4. Cannot see the whole MQI namespace, therefore an MQI application might generate messages whose properties cannot be read.
5. Works with Microsoft DTC only.

Table 25. XMS .NET: Client communication (not running on top of C client)

	XMS .NET				
	Distributed				
Client communication (not running on top of C client)	**Basic**“1” on page 185	**WAS**“2” on page 185	**Other Java EE**	**Other TM**	**Db2 SP**“3” on page 185
Read ahead	Y	N/A	N/A	N/A	N/A
Asynchronous put	Y	N/A	N/A	N/A	N/A

Table 25. XMS .NET: Client communication (not running on top of C client) (continued)

	XMS .NET				
	Distributed				
Client communication (not running on top of C client)	**Basic**“1” on page 185	**WAS**“2” on page 185	**Other Java EE**	**Other TM**	**Db2 SP**“3” on page 185
Pre-connect exit	N	N/A	N/A	N/A	N/A
CCDT	Y	N/A	N/A	N/A	N/A
Multiple CONNAME	N	N/A	N/A	N/A	N/A
Auto reconnect	N	N/A	N/A	N/A	N/A
Channel compression	N	N/A	N/A	N/A	N/A
Exit chaining	N	N/A	N/A	N/A	N/A
MQCSP/ MQXR_SEC_PARMS	N	N/A	N/A	N/A	N/A
Native SSL	N	N/A	N/A	N/A	N/A

Note:

1. Also OSGI.
2. WebSphere Application Server.
3. Db2 Stored Procedures.

Version 7.0.1 features and their supported APIs: XMS C and C++

Use this information to learn about the IBM WebSphere MQ 7.0.1 APIs with features and environments that might not be as fully supported as the C MQI.

XMS C and C++

The following tables show which features and environments are supported for XMS C and C++.

Table 26. XMS C and C++: API support

	XMS C and C++				
	Distributed				
API	**Basic**“1” on page 186	**WAS**“2” on page 186	**Other Java EE**	**Other TM**	**Db2 SP**“3” on page 186
V7 Publish Subscribe	Y	N/A	N/A	N/A	N/A
Browse with Mark	N	N/A	N/A	N/A	N/A
Message Properties	Y	N/A	N/A	N/A	N/A
Message Selectors	Y	N/A	N/A	N/A	N/A

Table 26. XMS C and C++: API support (continued)

	XMS C and C++				
	Distributed				
API	**Basic**“1” on page 186	**WAS**“2” on page 186	**Other Java EE**	**Other TM**	**Db2 SP**“3” on page 186
Asynchronous consume	Y	N/A	N/A	N/A	N/A
Content filtering Publish/ Subscribe	N	N/A	N/A	N/A	N/A
Group/Segment messages	N	N/A	N/A	N/A	N/A
PCF Classes	N	N/A	N/A	N/A	N/A
Global TX Participant	N	N/A	N/A	N/A	N/A
Global TX Coordinator	N	N/A	N/A	N/A	N/A

Note:

1. Also OSGI.
2. WebSphere Application Server.
3. Db2 Stored Procedures.

Table 27. XMS C and C++: Client communication (not running on top of C client)

	XMS C and C++				
	Distributed				
Client communication (not running on top of C client)	**Basic**“1” on page 187	**WAS**“2” on page 187	**Other Java EE**	**Other TM**	**Db2 SP**“3” on page 187
Read ahead	Y	N/A	N/A	N/A	N/A
Asynchronous put	Y	N/A	N/A	N/A	N/A
Pre-connect exit	N	N/A	N/A	N/A	N/A
CCDT	N	N/A	N/A	N/A	N/A
Multiple CONNAME	N	N/A	N/A	N/A	N/A
Auto reconnect	N	N/A	N/A	N/A	N/A
Channel compression	Y	N/A	N/A	N/A	N/A
Exit chaining	Y	N/A	N/A	N/A	N/A
MQCSP/ MQXR_SEC_PARMS	N/A	N/A	N/A	N/A	N/A

Table 27. XMS C and C++: Client communication (not running on top of C client) (continued)

	XMS C and C++				
	Distributed				
Client communication (not running on top of C client)	**Basic“1” on page 187**	**WAS“2” on page 187**	**Other Java EE**	**Other TM**	**Db2 SP“3” on page 187**
Native SSL	Y	N/A	N/A	N/A	N/A

Note:

1. Also OSGI.
2. WebSphere Application Server.
3. Db2 Stored Procedures.

Version 7.0.1 features and their supported APIs: .NET

Use this information to learn about the IBM WebSphere MQ 7.0.1 APIs with features and environments that might not be as fully supported as the C MQI.

.NET

The following table shows which features and environments are supported for .NET.

Table 28. .NET: API support

	.NET				
	Distributed				
API	**Basic“1” on page 188**	**WAS“2” on page 188**	**Other Java EE**	**Other TM**	**Db2 SP“3” on page 188**
V7 Publish Subscribe	Y	N/A	N/A	N/A	N/A
Browse with Mark	Y	N/A	N/A	N/A	N/A
Message Properties	Y“4” on page 188	N/A	N/A	N/A	N/A
Message Selectors	N	N/A	N/A	N/A	N/A
Asynchronous consume	N	N/A	N/A	N/A	N/A
Content filtering Publish/ Subscribe	N	N/A	N/A	N/A	N/A
Group/Segment messages	N	N/A	N/A	N/A	N/A
PCF Classes	N	N/A	N/A	N/A	N/A
Global TX Participant	Y“5” on page 188	N/A	N/A	N/A	N/A

Table 28. .NET: API support (continued)

	.NET				
	Distributed				
API	Basic"1" on page 188	WAS"2" on page 188	Other Java EE	Other TM	Db2 SP"3" on page 188
Global TX Coordinator	N	N/A	N/A	N/A	N/A

Note:

1. Also OSGI.
2. WebSphere Application Server.
3. Db2 Stored Procedures.
4. Cannot see the whole MQI namespace, therefore an MQI application might generate messages whose properties cannot be read.
5. Works with Microsoft DTC only.

Table 29. .NET: Client communication (not running on top of C client)

	.NET				
	Distributed				
Client communication (not running on top of C client)	Basic"1" on page 188	WAS"2" on page 188	Other Java EE	Other TM	Db2 SP"3" on page 188
Read ahead	Y	N/A	N/A	N/A	N/A
Asynchronous put	Y	N/A	N/A	N/A	N/A
Pre-connect exit	N	N/A	N/A	N/A	N/A
CCDT	Y	N/A	N/A	N/A	N/A
Multiple CONNAME	N	N/A	N/A	N/A	N/A
Auto reconnect	N	N/A	N/A	N/A	N/A
Channel compression	N	N/A	N/A	N/A	N/A
Exit chaining	N	N/A	N/A	N/A	N/A
MQCSP/ MQXR_SEC_PARMS	N	N/A	N/A	N/A	N/A
Native SSL	N"4" on page 188	N/A	N/A	N/A	N/A

Note:

1. Also OSGI.
2. WebSphere Application Server.
3. Db2 Stored Procedures.
4. Support available in Unmanaged mode only.

Channel authentication

From IBM WebSphere MQ 7.1 onwards, when you migrate a queue manager from an earlier release, channel authentication using channel authentication records is disabled. Channels continue to work as before. If you create a queue manager in Version 8.0, channel authentication using channel authentication records is enabled, but with minimal additional checking. Some channels might fail to start.

Migrated queue managers

Channel authentication is disabled for migrated queue managers.

To start using channel authentication records you must run this MQSC command:

```
ALTER QMGR CHLAUTH(ENABLED)
```

New queue managers

Channel authentication is enabled for new queue managers.

You want to connect existing queue managers or IBM MQ MQI client applications to a newly created queue manager. Most connections work without specifying any channel authentication records. The following exceptions are to prevent privileged access to the queue manager, and access to system channels.

1. Privileged user IDs asserted by a client-connection channel are blocked by means of the special value `*MQADMIN`.

   ```
   SET CHLAUTH('*') TYPE(BLOCKUSER) USERLIST('*MQADMIN') +
   DESCR('Default rule to disallow privileged users')
   ```

2. Except for the channel used by MQ Explorer, all `SYSTEM.*` channels are blocked.

   ```
   SET CHLAUTH('SYSTEM.*') TYPE(ADDRESSMAP) ADDRESS('*') USERSRC(NOACCESS) +
   DESCR('Default rule to disable all SYSTEM channels')

   SET CHLAUTH(SYSTEM.ADMIN.SVRCONN) TYPE(ADDRESSMAP) ADDRESS('*') USERSRC(CHANNEL) +
   DESCR('Default rule to allow MQ Explorer access')
   ```

Note: This behavior is default for all new Version 8.0 queue managers on startup.

If you must work around the exceptions, you can run an MQSC command to add in more rules to allow channels blocked by the default rules to connect, or disable channel authentication checking:

```
ALTER QMGR CHLAUTH(DISABLED)
```

Related information

Channel authentication records

Change in behavior of the `endmqm` command

From IBM WebSphere MQ 7.5 onwards, issuing an **endmqm** command and **dspmq** command immediately after each other might return misleading status.

When issuing an **`endmqm -c`** or **`endmqm -w`** command, in the unlikely event that a dspmq command is issued in the small timeframe between the applications disconnecting and the queue manager actually stopping, the dspmq command might report the status as `Ending immediately`, even though a controlled shutdown is actually happening.

Related information

endmqm

dspmq

Changes to cluster error recovery (on servers other than z/OS)

From IBM WebSphere MQ 7.1 onwards, the queue manager reruns operations that caused problems, until the problems are resolved. If, after five days, the problems are not resolved, the queue manager shuts down to prevent the cache becoming more out of date.

Before Version 7.1, if a queue manager detected a problem with the local repository manager managing a cluster, it updated the error log. In some cases, it then stopped managing clusters. The queue manager continued to exchange applications messages with a cluster, relying on its increasingly out of date cache of cluster definitions. From Version 7.1 onwards, the queue manager reruns operations that caused problems, until the problems are resolved. If, after five days, the problems are not resolved, the queue manager shuts down to prevent the cache becoming more out of date. As the cache becomes more out of date, it causes a greater number of problems. The changed behavior regarding cluster errors in Version 7.1 or later does not apply to z/OS.

Every aspect of cluster management is handled for a queue manager by the local repository manager process, `amqrrmfa`. The process runs on all queue managers, even if there are no cluster definitions.

Before Version 7.1, if the queue manager detected a problem in the local repository manager, it stopped the repository manager after a short interval. The queue manager kept running, processing application messages and requests to open queues, and publish or subscribe to topics.

With the repository manager stopped, the cache of cluster definitions available to the queue manager became more out of date. Over time, messages were routed to the wrong destination, and applications failed. Applications failed attempting to open cluster queues or publication topics that had not been propagated to the local queue manager.

Unless an administrator checked for repository messages in the error log, the administrator might not realize the cluster configuration had problems. If the failure was not recognized over an even longer time, and the queue manager did not renew its cluster membership, even more problems occurred. The instability affected all queue managers in the cluster, and the cluster appeared unstable.

From Version 7.1 onwards, IBM MQ takes a different approach to cluster error handling. Rather than stop the repository manager and keep going without it, the repository manager reruns failed operations. If the queue manager detects a problem with the repository manager, it follows one of two courses of action.

1. If the error does not compromise the operation of the queue manager, the queue manager writes a message to the error log. It reruns the failed operation every 10 minutes until the operation succeeds. By default, you have five days to deal with the error; failing which, the queue manager writes a message to the error log, and shuts down. You can postpone the five day shutdown.
2. If the error compromises the operation of the queue manager, the queue manager writes a message to the error log, and shuts down immediately.

An error that compromises the operation of the queue manager is an error that the queue manager has not been able to diagnose, or an error that might have unforeseeable consequences. This type of error often results in the queue manager writing an FFST file. Errors that compromise the operation of the queue manager might be caused by a bug in IBM MQ, or by an administrator, or a program, doing something unexpected, such as ending an IBM MQ process.

The point of the change in error recovery behavior is to limit the time the queue manager continues to run with a growing number of inconsistent cluster definitions. As the number of inconsistencies in cluster definitions grows, the chance of abnormal application behavior grows with it.

The default choice of shutting down the queue manager after five days is a compromise between limiting the number of inconsistencies and keeping the queue manager available until the problems are detected and resolved.

You can extend the time before the queue manager shuts down indefinitely, while you fix the problem or wait for a planned queue manager shutdown. The five-day stay keeps the queue manager running through a long weekend, giving you time to react to any problems or prolong the time before restarting the queue manager.

Corrective actions

You have a choice of actions to deal with the problems of cluster error recovery. The first choice is to monitor and fix the problem, the second to monitor and postpone fixing the problem, and the final choice is to continue to manage cluster error recovery as in releases before Version 7.1.

1. Monitor the queue manager error log for the error messages AMQ9448 and AMQ5008, and fix the problem.

   AMQ9448 indicates that the repository manager has returned an error after running a command. This error marks the start of trying the command again every 10 minutes, and eventually stopping the queue manager after five days, unless you postpone the shutdown.

   AMQ5008 indicates that the queue manager was stopped because an IBM MQ process is missing. AMQ5008 results from the repository manager stopping after five days. If the repository manager stops, the queue manager stops.

2. Monitor the queue manager error log for the error message AMQ9448, and postpone fixing the problem.

   If you disable getting messages from SYSTEM.CLUSTER.COMMAND.QUEUE, the repository manager stops trying to run commands, and continues indefinitely without processing any work. However, any handles that the repository manager holds to queues are released. Because the repository manager does not stop, the queue manager is not stopped after five days.

   Run an MQSC command to disable getting messages from SYSTEM.CLUSTER.COMMAND.QUEUE:

   ```
   ALTER QLOCAL(SYSTEM.CLUSTER.COMMAND.QUEUE) GET(DISABLED)
   ```

   To resume receiving messages from SYSTEM.CLUSTER.COMMAND.QUEUE run an MQSC command:

   ```
   ALTER QLOCAL(SYSTEM.CLUSTER.COMMAND.QUEUE) GET(ENABLED)
   ```

3. Revert the queue manager to the same cluster error recovery behavior as before Version 7.1.

   You can set a queue manager tuning parameter to keep the queue manager running if the repository manager stops.

   The tuning parameter is TolerateRepositoryFailure, in the TuningParameters stanza of the qm.ini file. To prevent the queue manager stopping, if the repository manager stops, set TolerateRepositoryFailure to TRUE; see Figure 16 on page 191.

   Restart the queue manager to enable the TolerateRepositoryFailure option.

   If a cluster error has occurred that prevents the repository manager starting successfully, and hence the queue manager from starting, set TolerateRepositoryFailure to TRUE to start the queue manager without the repository manager.

Special consideration

Before Version 7.1, some administrators managing queue managers that were not part of a cluster stopped the amqrrmfa process. Stopping amqrrmfa did not affect the queue manager.

Stopping amqrrmfa in Version 7.1 or later causes the queue manager to stop, because it is regarded as a queue manager failure. You must not stop the amqrrmfa process in Version 7.1 or later, unless you set the queue manager tuning parameter, TolerateRepositoryFailure.

Example

```
TuningParameters:
        TolerateRepositoryFailure=TRUE
```

Figure 16. Set TolerateRepositoryFailure to TRUE in qm.ini

Related information

Queue manager configuration files, qm.ini

Change in behavior of MQS_REPORT_NOAUTH

In IBM WebSphere MQ 7.1, the default behavior of MQS_REPORT_NOAUTH was changed to *TRUE*.

From IBM WebSphere MQ 7.1 onwards, this change causes UNIX platforms platforms to behave like Windows, and log authorization failures to the error log.

Prior to IBM WebSphere MQ 7.1, this would only happen if you set MQS_REPORT_NOAUTH.

For more information, see MQS_REPORT_NOAUTH.

Connect to multiple queue managers and use `MQCNO_FASTPATH_BINDING`

Applications that connect to queue managers using the `MQCNO_FASTPATH_BINDING` binding option might fail with an error and reason code `MQRC_FASTPATH_NOT_AVAILABLE`.

An application can connect to multiple queue managers from the same process. In releases earlier than IBM WebSphere MQ 7.1, an application can set any one of the connections to `MQCNO_FASTPATH_BINDING`. From Version 7.1 onwards, only the first connection can be set to `MQCNO_FASTPATH_BINDING`. For the complete set of rules, see Fast path.

To assist with migration, you can set a new environment variable, `AMQ_SINGLE_INSTALLATION`. The variable reinstates the same behavior as in earlier releases, but prevents an application connecting to queue managers associated with other installations in the same process.

Fast path

On a server with multiple installations, applications using a fast path connection to IBM WebSphere MQ Version 7.1 or later must follow these rules:

1. The queue manager must be associated with the same installation as the one from which the application loaded the IBM MQ run time libraries. The application must not use a fast path connection to a queue manager associated with a different installation. An attempt to make the connection results in an error, and reason code `MQRC_INSTALLATION_MISMATCH`.
2. Connecting non-fast path to a queue manager associated with the same installation as the one from which the application has loaded the IBM MQ run time libraries prevents the application connecting fast path, unless either of these conditions are true:
   - The application makes its first connection to a queue manager associated with the same installation a fast path connection.
   - The environment variable, `AMQ_SINGLE_INSTALLATION` is set.
3. Connecting non-fast path to a queue manager associated with a Version 7.1 or later installation, has no effect on whether an application can connect fast path.
4. You cannot combine connecting to a queue manager associated with a Version 7.0.1 installation and connecting fast path to a queue manager associated with a Version 7.1, or later installation.

With `AMQ_SINGLE_INSTALLATION` set, you can make any connection to a queue manager a fast path connection. Otherwise almost the same restrictions apply:

- The installation must be the same one from which the IBM MQ run time libraries were loaded.
- Every connection on the same process must be to the same installation. If you attempt to connect to a queue manager associated with a different installation, the connection fails with reason code `MQRC_INSTALLATION_MISMATCH`. Note that with `AMQ_SINGLE_INSTALLATION` set, this restriction applies to all connections, not only fast path connections.
- Only connect one queue manager with fast path connections.

Related information

Binding options
2587 (0A1B) (RC2587): MQRC_HMSG_NOT_AVAILABLE
2590 (0A1E) (RC2590): MQRC_FASTPATH_NOT_AVAILABLE

Custom scripts

In IBM MQ for Windows, custom scripts that have been used in an earlier release to install packages might fail if any packages have been renamed, removed, or added in the later release to which you are migrating.

For an overview of server and client installation, and for links to more detailed platform-specific information about the features that are available when you are installing an IBM MQ server or client, see Choosing what to install,

Changes to data types

A number of data types have changed between WebSphere MQ Version 7.0.1 to IBM MQ Version 8.0 and new data types have been added. This topic lists the changes for data types that have a new current version in Version 7.5.

The current version of a data type is incremented if the length of a data type is extended by adding new fields. The addition of new constants to the values that can be set in a data type does not result in a change to the current version value.

Table 30 on page 193 lists the data types that have new versions. Click on the links to read about the new fields.

Table 30. New fields added to existing data types

Data type	New version	New fields
Channel definition	MQCD_VERSION_10	BatchDataLimit (MQLONG) DefReconnect (MQLONG) UseDLQ (MQLONG)
Channel exit	MQCXP_VERSION_8	MCAUserSource (MQLONG) pEntryPoints (PMQIEP)
Data conversion exit	MQDXP_VERSION_2	pEntryPoints(PMQIEP)
Pre-connect exit	MQNXP_VERSION_2	pEntryPoints(PMQIEP)
Publish exit publication context	MQPBC_VERSION_2	MsgDescPtr(PMQMD)
Publish exit	MQPSXP_VERSION_2	pEntryPoints (PMQIEP)
Cluster workload exit	MQWXP_VERSION_4	pEntryPoints (PMQIEP)

Default transmission queue restriction

The product documentation in versions of IBM WebSphere MQ before Version 7.1 warned about defining the default transmission queue as SYSTEM.CLUSTER.TRANSMIT.QUEUE, but no error was reported. In Version 7.1, any attempt to set or use a default transmission queue that is defined as SYSTEM.CLUSTER.TRANSMIT.QUEUE results in an error.

In versions of IBM WebSphere MQ before Version 7.1, no error was reported when defining the default transmission queue as SYSTEM.CLUSTER.TRANSMIT.QUEUE. MQOPEN or MQPUT1 MQI calls that resulted in referencing the default transmission queue did not return an error. Applications might have continued working and failed later on. The reason for the failure was hard to diagnose.

The change from Version 7.1 onwards ensures that any attempt to set the default transmission queue to SYSTEM.CLUSTER.TRANSMIT.QUEUE, or use a default transmission queue set to SYSTEM.CLUSTER.TRANSMIT.QUEUE, is immediately reported as an AMQ8520 error.

Related reference

“MQI and PCF reason code changes” on page 203
Some reason codes that affect some existing programs changed in IBM WebSphere MQ 7.1.

Display channel and cluster status: Switching

From IBM WebSphere MQ 7.5 onwards, a cluster-sender channel that is switching its configuration to a different cluster transmission queue has a new channel state: Switching.

When migrating from a release before IBM WebSphere MQ 7.5 to Version 7.5 or later, existing application programs are not affected by the new state.

System management programs that monitor channel or cluster status might receive the new state as a result of a inquiry.

The state is set during the short interval while the channel modifies the destination transmission queue that messages are stored on. Before the switching state is set, messages are stored on the previously associated transmission queue. After the switching state, messages are stored on the newly configured transmission queue. The channel enters the switching state if a cluster-sender channel is starting, a configuration change is required, and the conditions for starting the switch are met.

Related information

Channel states
Working with cluster transmission queues and cluster-sender channels
DISPLAY CHSTATUS
Inquire Channel Status (Response)
Inquire Cluster Queue Manager (Response)
MQCHS_* (Command format Channel Status)
IBM i Work with MQ Channel Status (WRKMQMCHST)
IBM i Work with MQ Channels (WRKMQMCHL)
IBM i Work with MQ Clusters (WRKMQMCL)

Changes to `dspmqver` output

From IBM WebSphere MQ 7.1 onwards, new types of information are displayed by **`dspmqver`** to support multiple installations. The changes might affect existing administrative scripts that you have written to manage a release of IBM WebSphere MQ before IBM WebSphere MQ 7.1.

The changes in output from **`dspmqver`** that might affect existing command scripts that you have written are as follows:

1. Version 7.1 onwards has extra `-f` field options. If you do not specify a `-f` option, output from all the options is displayed. To restrict the output to the same information that was displayed in earlier releases, set the `-f` option to a value that was present in the earlier release. Compare the output for `dspmqver` in Figure 17 on page 194 and Figure 18 on page 195 with the output for `dspmqver f 15` in Figure 19 on page 195.

```
dspmqver
```

```
Name:        WebSphere MQ
Version:     7.0.1.6
CMVC level:  p701-L110705
BuildType:   IKAP - (Production)
```

*Figure 17. Default **`dspmqver`** options in IBM WebSphere MQ 7.0.1*

```
dspmqver
```

```
Name:        WebSphere MQ
Version:     7.1.0.0
Level:       p000-L110624
BuildType:   IKAP - (Production)
Platform:    WebSphere MQ for Windows
Mode:        32-bit
O/S:         Windows 7, Build 2600: SP1
InstName:    110705
InstDesc:    July 5 2011
InstPath:    C:\Program Files\IBM\WebSphere MQ_110705
DataPath:    C:\Program Files\IBM\WebSphere MQ
Primary:     No
MaxCmdLevel: 710

Note there are a number (1) of other installations,
use the '-i' parameter to display them.
```

Figure 18. Default ***dspmqver*** *options in IBM WebSphere MQ 7.1*

```
dspmqver -f 15
```

```
Name:        WebSphere MQ
Version:     7.1.0.0
Level:       p000-L110624
BuildType:   IKAP - (Production)
```

Figure 19. ***dspmqver*** *with option to make IBM WebSphere MQ 7.1, or later, similar to IBM WebSphere MQ 7.0.1*

Related information

dspmqver

Exits and installable services

When migrating to IBM WebSphere MQ 7.1 or later on a distributed platform, if you install the product in a non-default location, you must update your exits and installable services. Data conversion exits generated using the **crtmqcvx** command must be regenerated using the updated command.

When writing new exits and installable services, you do not need to link to any of the following libraries:

- mqmzf
- mqm
- mqmvx
- mqmvxd
- mqic
- mqutl

For more information about updating existing exits, and writing exits and installable services that do not link to the libraries, see Writing exits and installable services on UNIX, Linux and Windows .

Fewer IBM MQ MQI client log messages

Before Version 7.1, an IBM WebSphere MQ MQI client used to report every failed attempt to connect to a queue manager when processing a connection name list. From Version 7.1, only if the failure occurs with the last connection in the list is a message written to the queue manager error log.

Reporting the last failure and no others, reduces growth of the queue manager error log.

GSKit: Changes from Version 7.0 to Version 8.0

For distributed platforms, from IBM WebSphere MQ 7.1, GSKit Version 8.0 is integrated with IBM WebSphere MQ and is the only version of GSKit provided with the product. GSKit Version 7.0 is no longer provided.

In IBM WebSphere MQ 7.0.1, if you select SSL and TLS support during installation, GSKit Version 7.0 is installed and run by default. IBM WebSphere MQ 7.0.1, Fix Pack 4 and later also contain an alternative, separate copy of GSKit Version 8.0 that you can install and run instead of, or in addition to, GSKit Version 7.0. From IBM WebSphere MQ 7.1, GSKit Version 8.0 is the only version of GSKit that is provided.

Some functions in GSKit Version 8.0 are different from the functions in GSKit Version 7.0. These differences are described in the following subtopics.

GSKit: Some FIPS 140-2 compliant channels do not start

From IBM WebSphere MQ 7.1 three CipherSpecs are no longer FIPS 140-2 compliant. If a client or queue manager is configured to require FIPS 140-2 compliance, channels that use the following CipherSpecs do not start after migration.

- FIPS_WITH_DES_CBC_SHA
- FIPS_WITH_3DES_EDE_CBC_SHA
- TLS_RSA_WITH_DES_CBC_SHA

To restart a channel, alter the channel definition to use a FIPS 140-2 compliant CipherSpec. Alternatively, configure the queue manager, or the client in the case of an IBM MQ MQI client, not to enforce FIPS 140-2 compliance.

Earlier versions of IBM WebSphere MQ enforced an older version of the FIPS 140-2 standard. The following CipherSpecs were considered FIPS 140-2 compliant in earlier versions of IBM WebSphere MQ and are also compliant in this version:

- TLS_RSA_WITH_3DES_EDE_CBC_SHA (deprecated)
- TLS_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_AES_256_CBC_SHA
- TLS_RSA_WITH_AES_128_CBC_SHA256 (only when AltGSKit version 8 is used with Fix Pack 7.0.1.4 or later)
- TLS_RSA_WITH_AES_256_CBC_SHA256 (only when AltGSKit version 8 is used with Fix Pack 7.0.1.4 or later)

Use these CipherSpecs if you want IBM MQ to interoperate in a FIPS 140-2 compliant manner with earlier versions.

Previous IBM MQ releases enforced an older version of the FIPS 140-2 standard. The following CipherSpecs were considered FIPS 140-2 compliant by previous IBM MQ releases and are also considered compliant in this version of IBM MQ:

- TLS_RSA_WITH_3DES_EDE_CBC_SHA (deprecated)
- TLS_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_AES_256_CBC_SHA

Use these CipherSpecs if you need IBM MQ to interoperate in a FIPS 140-2 compliant manner with earlier IBM MQ releases.

Related information

AMQ9196

Federal Information Processing Standards (FIPS)

Federal Information Processing Standards

FipsRequired (MQLONG)

MQSSLFIPS

Specifying that only FIPS-certified CipherSpecs are used at run time on the MQI client

SSLFIPSRequired (MQLONG)

GSKit: Certificate Common Name (CN) not mandatory

In GSKit Version 8.0, the **iKeyman** command accepts any element of the distinguished name (DN), or a form of the subject alternative name (SAN). It does not mandate that you provide it with a common name. In GSKit Version 7.0, if you create a self-signed certificate using the **iKeyman** command you had to specify a common name.

The implication is that applications searching for a certificate might not able to assume that a certificate has a common name. You might need to review how applications search for certificates, and how applications handle errors involving the common name. Alternatively, you might choose to check that all self-signed certificates are given common names.

Some other certificate tools that you might also be using, do not require a common name. It is therefore likely the change to GSKit is not going to cause you a problem.

Related information

Distinguished Names

GSKit: Commands renamed

The command name **gsk7cmd** is replaced with **runmqckm** ; **gsk7ikm** is replaced with **strmqikm**, and **gsk7capicmd** is replaced with **runmqakm**. All the commands start the GSKit Version 8.0 certificate administration tools, and not the GSKit Version 7.0 tools.

IBM WebSphere MQ 7.1 or later does not use a machine-wide shared installation of GSKit: instead it uses a private GSKit installation in the IBM MQ installation directory. Each IBM WebSphere MQ 7.1 or later installation can use a different GSKit version. To display the version number of GSKit embedded in a particular installation, run the **dspmqver** command from that installation as shown in the following table:

Table 31. Renamed GSKit commands

Platform	GSKit Version 7.0 command	GSKit Version 8.0 command
UNIX and Linux	**gsk7cmd**	**runmqckm**
UNIX and Linux	**gsk7ikm**	**strmqikm**
Windows, UNIX and Linux	**gsk7capicmd**	**runmqakm**
Windows, UNIX and Linux	**gsk7ver**	**dspmqver -p 64 -v**

Note: Do not use the **gsk8ver** command to display the GSKit version number: only the **dspmqver** command will show the correct GSKit version number.

Related information

runmqckm, and runmqakm commands

Using iKeyman, iKeycmd, runmqakm, and runmqckm

dspmqver

GSKit: The `iKeyman` commands to insert a certificate do not check that all required CA certificates are present

The **iKeyman** command in GSKit V8.0 does not validate a certificate when it is inserted into a key repository. **iKeyman** in GSKit V7.0, validated a certificate before it inserted the certificate into a certificate store.

The implication is, that if you create a certificate using the **iKeyman** in GSKit V8.0, all the necessary intermediate and root CA certificates might not be present, or they might have expired; when the certificate is checked it might fail.

Necessary certificates might be missing, or have expired. This can cause SSL and TLS connections to fail with error AMQ9633.

Related information

Certificate validation and trust policy design on UNIX, Linux and Windows systems

GSKit: PKCS#11 and JRE addressing mode

If you use **iKeyman** or **iKeycmd** to administer certificates and keys for PKCS#11 cryptographic hardware note that the addressing mode of the JRE for these tools has changed from IBM WebSphere MQ 7.1.

In earlier releases of IBM WebSphere MQ, on the following platforms the JRE was 32 bit however in IBM WebSphere MQ 7.1 or later, it is 64 bit only. Where it has changed you might need to install additional PKCS#11 drivers appropriate for the addressing mode of the **iKeyman** and **iKeycmd** JRE. This is because the PKCS#11 driver must use the same addressing mode as the JRE. The following table shows the IBM WebSphere MQ 7.1 or later JRE addressing modes.

Table 32. IBM WebSphere MQ 7.1 or later JRE addressing modes

Platform	JRE Addressing Mode
Windows (32 bit or 64 bit)	32
Linux for System x 32 bit	32
Linux for System x 64 bit	64
Linux on POWER Systems - Big Endian	64
Linux for IBM Z	64
HP-UX	64
Solaris Sparc	64
Solaris x86-64	64
AIX	64
z/OS z/OS z/OS	n/a
IBM i IBM i IBM i	n/a

GSKit: Import of a duplicate PKCS#12 certificate

In GSKit V8.0, the **iKeyman** command does not report an attempt to import a duplicate PKCS#12 certificate as an error. In GSKit V7.0, the **iKeyman** command reported an error. In neither version is a duplicate certificate imported.

For GSKIT V8.0, a duplicate certificate is a certificate with the same label and public key.

The implication is that if some of the issuer information is different, but the name and public key are the same, the changes are not imported. The correct way to update a certificate is to use the `-cert -receive` option, which replaces an existing certificate.

gskcapicmd does not allow or ignore duplicates on import in this way.

Related information

Importing a personal certificate into a key repository on UNIX, Linux or Windows systems

GSKit: Certificate stores created by `iKeyman` and `iKeycmd` no longer contain CA certificates

The **`iKeyman`** and **`iKeycmd`** utilities in GSKit V8.0 create a certificate store without adding pre-defined CA certificates to the store. To create a working certificate store, you must now add all the certificates that you require and trust. In GSKit V7.0 **`iKeyman`** and **`iKeycmd`** created a certificate store that already contained CA certificates.

Existing data bases created by GSKit V7.0 are unaffected by this change.

Related information

Adding default CA certificates into an empty key repository, on UNIX, Linux or Windows systems with GSKit Version 8.0

GSKit: Password expiry to key database deprecated

In GSKit V8.0, the password expiry function in **`iKeyman`** continues to work the same as in GSKit V7.0, but it might be withdrawn in future versions of GSKit.

Use the file system protection provided with the operating system to protect the key database and password stash file.

Linux: Recompile C++ applications and update run time libraries

C++ IBM MQ MQI client and server applications on Linux must be recompiled using a supported version of the GNU Compiler Collection (GCC). The C++ run time libraries for this version of GCC must be installed in `/usr/lib` or `/usr/lib64`.

For information about which versions of GCC are supported , see System Requirements for IBM MQ, and follow the links for Linux. Older versions of the GCC that are not included in the System Requirements information are no longer supported.

If you are using one of the supported Linux distributions, the libraries are correctly installed; see System Requirements for IBM MQ.

The GCC Version 4.x libraries support SSL and TLS connections from an IBM MQ MQI client. SSL and TLS use GSKit Version 8.0, which depends on `libstdc++.so.6`. `libstdc++.so.6` is included in GCC 4.x.

GSKit: Signature algorithm moved out of settings file

In GSKit V8.0, the default signature algorithm used when creating self-signed certificates or certificate requests or selected in the creation dialogs is passed as a command-line parameter. In GSKit V7.0, the default signature algorithm was specified in the settings file.

The change has very little effect: it causes a different default signature algorithm to be selected. It does not alter the selection of a signature algorithm.

Related information

runmqckm and runmqakm options

Creating a self-signed personal certificate on UNIX, Linux, and Windows systems

GSKit: Signed certificate validity period not within signer validity

In GSKit V8.0, the **`iKeyman`** command does not check whether the validity period of a resulting certificate is within the validity period of the signed certificate. In GSKit V7.0, **`iKeyman`** checked that the validity period of the resulting certificate was within the validity period of the signed certificate.

The IETF RFC standards for SSL/TLS allow a certificate whose validity dates extend beyond those of its signer. This change to GSKit brings it into line with those standards. The check is whether the certificate is issued within the validity period of the signer, and not whether it expires within the validity period of the signer.

Related information

How SSL and TLS provide identification, authentication, confidentiality, and integrity

GSKit: Stricter default file permissions

The default file permissions set by **runmqckm** and **strmqikm** in IBM WebSphere MQ 7.5 on UNIX and Linux are stricter than the permissions that are set by **runmqckm**, **strmqikm**, **gsk7cmd**, and **gsk7ikm** in earlier releases of IBM MQ.

The permissions set by **runmqckm** and **strmqikm** in IBM WebSphere MQ 7.5 permit only the creator to access the UNIX and Linux SSL/TLS key databases. The **runmqckm**, **strmqikm**, **gsk7cmd**, and **gsk7ikm** tools in earlier releases of IBM MQ set world-readable permissions, making the files liable to theft and impersonation attacks.

The permissions set by **gsk7capicmd**, in earlier releases of IBM MQ, and **runmqakm** in IBM WebSphere MQ 7.5, permit only the creator to access UNIX and Linux SSL/TLS key databases.

The migration of SSL/TLS key databases to Version 7.5 does not alter their access permissions. In many cases, administrators set more restrictive access permissions on these files to overcome the liability to theft and impersonation attacks; these permissions are retained.

The default file permissions set on Windows are unchanged. Continue to tighten up the access permissions on SSL/TLS key database files on Windows after creating the files with **runmqckm** or **strmqikm**.

Related information

Accessing and securing your key database files on Windows

Accessing and securing your key database files on UNIX and Linux systems

Java: Change in behavior of default value of `MQEnvironment.userID`

Change caused when using CLIENT transport for a channel not have a security exit defined.

From IBM WebSphere MQ 7.1, if an IBM WebSphere MQ classes for Java application is connecting to a queue manager, using the CLIENT transport through a channel that does not have a security exit defined, and the `MQEnvironment.userID` field is kept at its default value of the empty string (`""`), the IBM WebSphere MQ classes for Java application queries the value of the Java System Property `user.name` and passes this to the queue manager for authorization as part of the `MQQueueManager` constructor.

If the user specified by the Java System Property `user.name` is not authorized to access the queue manager, the `MQQueueManager` constructor throws an MQException containing Reason Code MQRC_NOT_AUTHORIZED.

Java: Different message property data type returned

From IBM WebSphere MQ 7.1, if the data type of a message property is set, the same data type is returned when the message is received. This is different from IBM WebSphere MQ 7.0.1, where in some circumstances properties set with a specific type were returned with the default type `String`.

The change affects Java applications that used the MQRFH2 class, and retrieved properties using the `getFieldValue` method.

You can write a message property in Java using a method such as `setIntFieldValue`. In IBM WebSphere MQ 7.0.1 the property is written into an MQRFH2 header with a default type of `String`. When you retrieve the property with the `getFieldValue` method, a `String` object is returned.

The change is that now the correct type of object is returned, in this example the type of object returned is `Integer`.

If your application retrieves the property with the `getIntFieldValue` method, there is no change in behavior; an int is returned. If property is written to the MQRFH2 header by some other means, and the data type is set, then `getFieldValue` returns the correct type of object.

Related information

Class MQRFH2

JMS: Change in behavior of the default user identifier value

Change caused when using CLIENT transport for a channel that does not have a security exit defined.

From IBM WebSphere MQ 7.1, if an IBM WebSphere MQ classes for JMS application is connecting to a queue manager, using the CLIENT transport through a channel that does not have a security exit defined, and no user identifier is specified, by calling `ConnectionFactory.createConnection()`, the IBM WebSphere MQ classes for JMS application queries the value of the Java System Property `user.name` and passes this to the queue manager for authorization as part of the call to create a connection from a connection factory object. This behavior also occurs when calling `ConnectionFactory.createConnection(String, String)` and passing a blank or null value for the first parameter **`userID`**.

If the user specified by the Java System Property `user.name` is not authorized to access the queue manager, a JMSException containing Reason Code MQRC_NOT_AUTHORIZED is thrown.

JMS: some objects are no longer serializable

JMS objects such as JMS Connections and JMS Sessions, which used to be serializable when using the IBM MQ classes for JMS for IBM WebSphere MQ 7.0.1, are no longer serializable when using IBM WebSphere MQ 7.1 or later.

When a Java application serializes an object, state information about that object is written to an output stream, such as a file. The contents of output stream can then be read at a later date, to reconstruct (or deserialize) the Java object so that it can be reused.

The following interfaces provided by the IBM MQ classes for JMS are implemented by objects that represent an active connection from an application to an IBM MQ queue manager:

- JMSConnection
- JMSQueueConnection
- JMSTopicConnection
- JMSSession
- JMSQueueSession
- JMSTopicSession

There is an IBM MQ connection handle (hconn) associated with every JMSConnection, JMSQueueConnection, JMSTopicConnection, JMSSession, JMSQueueSession and JMSTopicSession object that is created by an application.

Serializing one of these objects would result in state information about those objects being written to an output stream. This included information about the connection handle to IBM MQ that was associated with the object.

However, there was no guarantee that the connection handle would still be valid when the object was deserialized and resused, which could lead to unexpected behavior. To prevent applications running into these issues, the IBM MQ classes for JMS for Version 7.1 (and later) will throw a NotSerializableException if an application tries to:

- Serialize a JMSConnection, JMSQueueConnection, JMSTopicConnection, JMSSession, JMSQueueSession or JMSTopicSession object using the method:

  ```
  writeObject(ObjectOutputStream)
  ```

- Deserialize a JMSConnection, JMSQueueConnection, JMSTopicConnection, JMSSession, JMSQueueSession or JMSTopicSession object using the method:

```
readObject(ObjectInputStream)
```

JMS: Reason code changes

From IBM WebSphere MQ 7.1, some reason codes returned in JMS exceptions have changed. The changes affect MQRC_Q_MGR_NOT_AVAILABLE and MQRC_SSL_INITIALIZATION_ERROR.

Before IBM WebSphere MQ 7.1, if a JMS application call fails to connect, it receives an exception with a reason code 2059 (080B) (RC2059): MQRC_Q_MGR_NOT_AVAILABLE. From Version 7.1, the application can still receive MQRC_Q_MGR_NOT_AVAILABLE, or one of the following more specific reason codes.

- 2537 (09E9) (RC2537): MQRC_CHANNEL_NOT_AVAILABLE
- 2538 (09EA) (RC2538): MQRC_HOST_NOT_AVAILABLE
- 2539 (09EB) (RC2539): MQRC_CHANNEL_CONFIG_ERROR
- 2540 (09EC) (RC2540): MQRC_UNKNOWN_CHANNEL_NAME

Similarly, when trying to connect, a JMS application might have received 2393 (0959) (RC2393): MQRC_SSL_INITIALIZATION_ERROR. From Version 7.1, the application can still receive MQRC_SSL_INITIALIZATION_ERROR, or a more specific reason code, such as 2400 (0960) (RC2400): MQRC_UNSUPPORTED_CIPHER_SUITE, that identifies the cause of the SSL initialization error.

JMS: ResourceAdapter object configuration

When WebSphere Application Server connects to IBM MQ it creates message driven beans (MDBs) using JMS connections. From Version 7.1, these MDBs can no longer share one JMS connection. The configuration of ResourceAdapter object is migrated so that there is a single MDB for each JMS connection.

Changed ResourceAdapter properties

connectionConcurrency
The maximum number of MDBs to share a JMS connection. Sharing connections is not possible and this property always has the value 1. Its previous default value was 5.

maxConnections
This property is the number of JMS connections that the resource adapter can manage. From Version 7.1, it also determines the number of MDBs that can connect because each MDB requires one JMS connection. The default value of maxConnections is now 50. Its previous default value was 10.

If connectionConcurrency is set to a value greater than 1, the maximum number of connections supported by the resource adapter is scaled by the value of connectionConcurrency. For example, if maxConnections is set to 2 and connectionConcurrency is set to 4, the maximum number of connections supported by the resource adapter is 8. As a result, connectionConcurrency is set to 1 and maxConnections is set to 8.

If connectionConcurrency is set to a value greater than 1, it is adjusted automatically. To avoid automatic adjustment, set connectionConcurrency to 1. You can then set maxConnections to the value you want.

The scaling mechanism ensures that sufficient connections are available for existing deployments whether you have changed them in your deployment, configuration, or programs.

If the adjusted maxConnections value exceeds the MAXINST or MAXINSTC attributes of any used channel, previously working deployments might fail.

The default value of both channel attributes equates to unlimited. If you changed them from the default value, you must ensure that the new maxConnections value does not exceed MAXINST or MAXINSTC.

Related information

Configuration of the ResourceAdapter object

V 8.0.0.4 Maximum instances (MAXINST)

Maximum instances per client (MAXINSTC)

MQI and PCF reason code changes

Some reason codes that affect some existing programs changed in IBM WebSphere MQ 7.1.

MQRC_NOT_OPEN_FOR_INPUT

In IBM WebSphere MQ 7.0 a queue opened with MQOO_OUTPUT, and then browsed, returned an error with the wrong reason-code, MQRC_NOT_OPEN_FOR_INPUT. The correct reason-code, MQRC_NOT_OPEN_FOR_BROWSE, was issued by Version 6.0 and earlier. Version 7.1 and later correctly returns an error with the same reason code as Version 6.0, MQRC_NOT_OPEN_FOR_BROWSE.

MQRC_DEF_XMIT_Q_USAGE_ERROR

The product documentation in versions of IBM WebSphere MQ before Version 7.1 warned about defining the default transmission queue as SYSTEM.CLUSTER.TRANSMIT.QUEUE, but no error was reported. From Version 7.1 onwards, an attempt to open the default transmission queue, defined as SYSTEM.CLUSTER.TRANSMIT.QUEUE, results in the error MQRC_DEF_XMIT_Q_USAGE_ERROR.

MQRC_FASTPATH_NOT_AVAILABLE

An application that connects to multiple queue managers in the same process and uses MQCNO_FASTPATH_BINDING might fail with an error and reason code MQRC_FASTPATH_NOT_AVAILABLE; see "Connect to multiple queue managers and use MQCNO_FASTPATH_BINDING" on page 192.

MQRCCF_DEF_XMIT_Q_CLUS_ERROR

The product documentation in versions of IBM WebSphere MQ before Version 7.1 warned about defining the default transmission queue as SYSTEM.CLUSTER.TRANSMIT.QUEUE, but no error was reported. From Version 7.1 onwards, an attempt to alter the queue manager attribute **DEFXMITQ** to SYSTEM.CLUSTER.TRANSMIT.QUEUE results in an error. The PCF reason code is 3269 (0CC5) (RC3269): MQRCCF_DEF_XMIT_Q_CLUS_ERROR.

Related reference

"Connect to multiple queue managers and use MQCNO_FASTPATH_BINDING" on page 192
Applications that connect to queue managers using the MQCNO_FASTPATH_BINDING binding option might fail with an error and reason code MQRC_FASTPATH_NOT_AVAILABLE.

Related information

2036 (07F4) (RC2036): MQRC_NOT_OPEN_FOR_BROWSE

2037 (07F5) (RC2037): MQRC_NOT_OPEN_FOR_INPUT

2590 (0A1E) (RC2590): MQRC_FASTPATH_NOT_AVAILABLE

3269 (0CC5) (RC3269): MQRCCF_DEF_XMIT_Q_CLUS_ERROR

AMQ8520

Publish/Subscribe: Delete temporary dynamic queue

If a subscription is associated with a temporary dynamic queue, when the queue is deleted, the subscription is deleted. Publish/subscribe applications migrated from WebSphere Message Broker are unchanged. The change does not affect the behavior of integrated publish/subscribe applications, which are written using the MQI publish/subscribe interface.

- In IBM WebSphere MQ 7.0, if you migrate or create a queued publish/subscribe application that uses MQRFH1, it behaves the same as IBM WebSphere MQ 6.0. You can create a temporary dynamic queue for a subscription, and if the queue is deleted, the subscription is not deleted, as in IBM WebSphere MQ 6.0. The lack of a subscriber queue results in any matching publications ending up on the dead letter queue.
- From IBM WebSphere MQ 7.0.1, Fix Pack 6 onwards, in the same MQRFH1 queued publish/subscribe case, if the temporary dynamic queue is deleted, the subscription is deleted. This change prevents a

buildup of publications from a subscription without a subscriber queue ending up on the dead letter queue.

- From IBM WebSphere MQ 7.0 onwards, if you migrate or create a queued publish/subscribe application that uses MQRFH2, it behaves the same as WebSphere Message Broker IBM Integration Bus. You can create a temporary dynamic queue for a subscription, and if the queue is deleted, the subscription is deleted, as in IBM Integration Bus. MQRFH2 publish/subscribe applications are typically migrated from IBM Integration Bus.
- From IBM WebSphere MQ 7.0 onwards, if you create a durable subscription using integrated publish/subscribe, you cannot define a temporary dynamic queue as the destination for its matching publications.
- From IBM WebSphere MQ 7.0, you can create a managed, non-durable subscription using integrated publish/subscribe, which creates a temporary dynamic queue as the destination for matching publications. The subscription is deleted with the queue.

Summary

From IBM WebSphere MQ 7.0.1 onwards, you cannot create a temporary dynamic queue as the destination for publications for a durable subscription using the integrated publish/subscribe interface.

If you use either of the queued publish/subscribe interfaces, MQRFH1 or MQRFH2, the behavior is the same. You can create a temporary dynamic queue as the subscriber queue, and if the queue is deleted, the subscription is deleted with it.

Queue manager logs: Default sizes increased

From IBM WebSphere MQ 7.1, the default size of a queue manager log files changed to 4096. The queue manager error log increased from 256 KB to 2 MB on some platforms. The change affects both new and migrated queue managers.

Queue manager log

From IBM WebSphere MQ 7.1, the default queue manager log size is 4096. For more information on setting non-default values, see The IBM MQ configuration file, mqs.ini.

Queue manager error log

From IBM WebSphere MQ 7.1, the AMQERR *nn*.log increased from 256 KB to 2 MB on IBM i IBM i, UNIX, Linux, and Windows platforms.

You can override the change either by setting the environment variable MQMAXERRORLOGSIZE, or by setting ErrorLogSize in the QMErrorLog stanza in the qm.ini file.

The change increases the number of error messages that are saved in the error logs.

Related information

Queue manager error logs

IBM i The queue manager error log stanza

Removal of dspmqsver command

Before IBM WebSphere MQ 7.5, the **dspmqsver** command was used to display the version of IBM WebSphere MQ Advanced Message Security.

From Version 7.5, IBM WebSphere MQ Advanced Message Security is a component of the main product and the version information is displayed as part of the **dspmqver** command.

Security: SSLPEER and SSLCERTI changes

IBM WebSphere MQ 7.1 or later obtains the Distinguished Encoding Rules (DER) encoding of the certificate and uses it to determine the subject and issuer distinguished names. The subject and issuer distinguished names are used in the SSLPEER and SSLCERTI fields. A SERIALNUMBER attribute is also included in the subject distinguished name and contains the serial number for the certificate of the remote partner. Some attributes of subject and issuer distinguished names are returned in a different sequence from releases before Version 7.1.

The change to subject and issuer distinguished names affects channel security exits. It also affects aplications which depend upon the subject and issuer distinguished names that are returned by the PCF programming interface. Channel security exits and applications that set or query SSLPEER and SSLCERTI must be examined, and possibly changed. The fields that are affected are listed in Table 33 on page 205 and Table 34 on page 205.

Table 33. Channel status fields affected by changes to subject and issuer distinguished names

Channel status attribute	PCF channel parameter type
SSL Peer (SSLPEER)	`MQCACH_SSL_SHORT_PEER_NAME`
`SSLCERTI`	`MQCACH_SSL_CERT_ISSUER_NAME`

Table 34. Channel data structures affected by changes to subject and issuer distinguished names

Channel data structure	Field
MQCD - Channel definition	SSLPeerNamePtr (MQPTR)
MQCXP - Channel exit parameter	SSLRemCertIssNamePtr (PMQVOID)

Existing peer name filters specified in the SSLPEER field of a channel definition are not affected. They continue to operate in the same manner as in earlier releases. The peer name matching algorithm has been updated to process existing SSLPEER filters. It is not necessary to alter any channel definitions.

Related information

Channel security exit programs

Security: Disable IBM MQ AMS at the client

From IBM WebSphere MQ 7.5, you can disable IBM WebSphere MQ Advanced Message Security at the client, for Java and C clients, to prevent errors when they are connecting to queue managers that are running on earlier versions of IBM WebSphere MQ.

For Java clients, in IBM WebSphere MQ 7.5.0.4 and later, you can disable IBM WebSphere MQ Advanced Message Security by setting an environment variable AMQ_DISABLE_CLIENT_AMS.

For C clients, in IBM WebSphere MQ 7.5.0.5 and later, you can disable IBM WebSphere MQ Advanced Message Security by using the DisableClientAMS property, under the **`Security`** stanza in the `mqclient.ini` file.

Related information

Environment variables used to disable IBM MQ AMS at the client

Telemetry: Installer integrated with IBM MQ

From Version 7.1, IBM MQ Telemetry is no longer installed separately from IBM MQ. It is installed as a component of the main product. If you installed IBM MQ Telemetry with Version 7.0.1, you must uninstall it before installing Version 7.1 or later.

You can install IBM MQ Telemetry at the same time as IBM MQ, or you can rerun the installer and install IBM MQ Telemetry at a later time.

Related information

Installing IBM MQ Telemetry

Uninstalling IBM MQ Telemetry Version 7.0.1 components

Telemetry: Support for the MQTT protocol over WebSockets

IBM WebSphere MQ 7.5.0, Fix Pack 1 and later supports the MQTT protocol over WebSockets. This enables it to be a server for clients using the MQTT messaging client for JavaScript.

A new communication protocol parameter (PROTOCOL) has been added to the MQTT channel definition (DEFINE CHANNEL (MQTT)):

- If the parameter is set to `MQTTV3`, the channel only accepts connections from clients using Version 3 of the MQ Telemetry Transfer protocol. This was the only protocol supported before IBM WebSphere MQ 7.5.0, Fix Pack 1.
- If the parameter is set to HTTP, the channel only accepts HTTP requests for pages, or WebSockets connections to IBM MQ Telemetry.
- If the parameter is set to `MQTTV3,HTTP`, the channel accepts connections from clients using either protocol. This is the default behavior for new MQTT channels created with IBM WebSphere MQ 7.5.0, Fix Pack 1 and later versions.

When a client connects to an MQTT channel using SSL, the parameter SSLCAUTH determines whether IBM MQ requires a certificate from the client (see DEFINE CHANNEL (MQTT)). Before IBM WebSphere MQ 7.5.0, Fix Pack 1, this parameter could be either `REQUIRED` or `OPTIONAL` for MQTT channels:

- `REQUIRED` means that IBM MQ requests a certificate from the client and the client must supply a valid certificate.
- `OPTIONAL` means that IBM MQ will request a certificate from the client but the client does not have to supply one. The client connection is allowed if the client supplies a valid certificate or if the client does not supply a certificate. The client connection is disallowed only if the client supplies an invalid certificate.

In IBM WebSphere MQ 7.5.0, Fix Pack 1 and later, the parameter SSLCAUTH can be set to `NEVER` for MQTT channels. `NEVER` means that IBM MQ never requests a certificate from the client. The new value was added as part of the support for clients using the MQTT messaging client for JavaScript. It accommodates the behavior of some web browsers which treat the request for a client certificate as a protocol error.

Related information

Getting started with the MQTT messaging client for JavaScript

MQ Explorer changes

From Version 7.1, IBM WebSphere Eclipse Platform it is not required to run MQ Explorerand is therefore no longer shipped with the product. The change makes no difference to administrators who run MQ Explorer. For developers who run MQ Explorer in an Eclipse development environment, a change is necessary. You must install and configure a separate Eclipse environment to be able to switch between MQ Explorer and other perspectives.

Packaging changes

In versions of IBM WebSphere MQ earlier than Version 7.1, you can select the Workbench mode preference in MQ Explorer. In workbench mode, you could switch to the other perspectives installed in the WebSphere Eclipse Platform. In Version 7.1 or later, you can no longer set the Workbench mode preference, because the WebSphere Eclipse Platform is not shipped with MQ Explorer.

To switch between MQ Explorer and other perspectives, you must install MQ Explorer into your own Eclipse environment or into an Eclipse-based product. You can then switch between perspectives.

If you installed extensions to previous versions of MQ Explorer, such as SupportPacs or WebSphere Message Broker (now known as IBM Integration Bus) Explorer, you must reinstall compatible versions of the extensions after upgrading MQ Explorer to Version 7.1 or later.

If you continue to run IBM WebSphere MQ 7.0.1 on the same server as a later version of the product, and you use MQ Explorer, each installation uses its own installation of MQ Explorer. When you uninstall Version 7.0.1, its version of MQ Explorer is uninstalled. To remove IBM WebSphere Eclipse Platform, uninstall it separately. The workspace is not deleted.

Test result migration

Test results are not migrated from version to version. To view any test results, you must rerun the tests.

Related information

Multiple installations of MQ Explorer

AIX: Shared objects

On AIX, for Version 7.1. and later, the `.a` shared objects in the `lib64` directory contains both the 32 bit and 64 bit objects. A symlink to the `.a` file is also placed in the `lib` directory. The AIX loader can then correctly pick up the correct object for the type of application being run.

This means that IBM MQ applications can run with the `LIBPATH` containing either the `lib` or `lib64` directory, or both.

AIX: `/usr/lpp/mqm` symbolic link removed

Before Version 6.0, IBM WebSphere MQ placed a symbolic link in `/usr/lpp/mqm` on AIX. The link ensured queue managers and applications migrated from IBM WebSphere MQ versions before Version 5.3 continued to work, without change. However, this link is not created in Version 7.1, or later.

In version 5.0, IBM MQ for AIX was installed into `/usr/lpp/mqm`. That changed in Version 5.3 to `/usr/mqm`. A symbolic link was placed in `/usr/lpp/mqm`, linking to `/usr/mqm`. Existing programs and scripts that relied on the installation into `/usr/lpp/mqm` continued to work unchanged. That symbolic link was removed in Version 7.5, because you can now install IBM MQ in any directory. Applications and command scripts are affected by the change.

The effect on applications is no different to the effect of migrating on other UNIX and Linux platforms. If the installation is made primary, then symbolic links to the IBM MQ link libraries are placed in `/usr/lib`. Most applications migrated from earlier IBM MQ versions search the default search path, which normally includes `/usr/lib`. The applications find the symbolic link to the IBM MQ load libraries in `/usr/lib`.

If the installation is not primary, then you must configure the correct search path to load the IBM MQ link libraries. If you choose to run **`setmqenv`**, IBM MQ places the IBM MQ link library path into `LIBPATH`. Unless the application is configured not to search the `LIBPATH`, if for example it is a `setuid` or `setgid` application, then the IBM MQ library is loaded successfully; see "UNIX: Migrating IBM MQ library loading from Version 7.0.1, or later, to the latest version" on page 136.

If you have written command scripts that run IBM MQ commands, you might have coded explicit paths to the directory tree where IBM MQ was installed. You must modify these command scripts. You can run **`setmqenv`** to create the correct environment to run the command scripts. If you have set the installation as primary, you do not have to specify the path to the command.

Related tasks

"UNIX: Migrating IBM MQ library loading from Version 7.0.1, or later, to the latest version" on page 136
Investigate whether applications connecting to the latest version of the product are linked to, and load libraries from, the correct installation.

Related information

"Loading IBM MQ libraries" on page 58

AIX, HP-UX, and Solaris: Building applications for TXSeries

From Version 7.1, applications that link to TXSeries® must load the mqzi_r library, not the mqz_r library. These applications must load mqzi_r from the installation that is associated with the queue manager to which the application is connected.

Before Version 7.1, IBM WebSphere MQ applications that used the TXSeries CICS support loaded the IBM WebSphere MQ library, mqz_r. From Version 7.1 those applications must load the IBM WebSphere MQ library, mqzi_r instead. You must change your build scripts accordingly and rebuild your applications.

In Version 7.1 and later, mqz_r includes code to load a different version of the library. IBM MQ loads a different version of the library, if it detects that the queue manager to which the application is connected is associated with a different installation from the installation from which the library was loaded. However, mqzi_r does not include the additional code. When using TXSeries, the application must run with the library it that loaded, and not a different library loaded by IBM MQ. For this reason, applications that use the TXSeries CICS support must load the mqzi_r library, and not the mqz_r library.

An implication of applications loading mqzi_r instead of mqz_r is that an application must load the correct version of mqzi_r. The application must load the version of mqzi_r from the installation that is associated with the queue manager to which the application is connected.

Related information

Building libraries for use with TXSeries for Multiplatforms version 5

AIX: TXSeries CICS support

HP-UX: TXSeries CICS support

Solaris: TXSeries CICS support

Linux: Recompile C++ applications and update run time libraries

C++ IBM MQ MQI client and server applications on Linux must be recompiled using a supported version of the GNU Compiler Collection (GCC). The C++ run time libraries for this version of GCC must be installed in /usr/lib or /usr/lib64.

For information about which versions of GCC are supported , see System Requirements for IBM MQ, and follow the links for Linux. Older versions of the GCC that are not included in the System Requirements information are no longer supported.

If you are using one of the supported Linux distributions, the libraries are correctly installed; see System Requirements for IBM MQ.

The GCC Version 4.x libraries support SSL and TLS connections from an IBM MQ MQI client. SSL and TLS use GSKit Version 8.0, which depends on libstdc++.so.6. libstdc++.so.6 is included in GCC 4.x.

Linux: Increased shared memory allocation required

The maximum amount of shared memory (SHMMAX) to allocate on Linux systems. The default system allocation is 32 MB. IBM MQ starts by allocating 64 MB and increases its allocation on demand by doubling its previous allocation. On a production system set SHMMAX to at least 256 MB to accommodate additional allocations.

Related information

Additional settings for installing IBM MQ on Linux systems

UNIX and Linux : crtmqlnk and dltmqlnk removed

Before Version 7.1 , the **crtmqlnk** and **dltmqlnk** commands created symbolic links in subdirectories of /usr. From Version 7.1 onwards, you must use the **setmqinst** command instead.

Related reference

“UNIX and Linux: /usr symbolic links removed” on page 210

From Version 7.1, on all UNIX and Linux platforms, the links from the `/usr` file system are no longer made automatically. In order to take advantage of these links, you must set an installation as the primary installation.

Related information

Changing the primary installation

setmqinst

UNIX and Linux: Message catalogs moved

From Version 7.1, message catalogs are no longer stored in the system directories. To support multiple installations, copies of the message catalogs are stored with each installation. If you are migrating from a release before Version 7.1 and you want messages only in the locale of your system, the change has no affect on your system. If you have customized the way in which the search procedure selects a message catalog, then the customization might no longer work correctly.

Set the LANG environment variable to load a message catalog for a different language from the system locale.

Related information

Displaying messages in your national language on UNIX and Linux systems

UNIX and Linux: MQ services and triggered applications

From Version 7.1, both LD_LIBRARY_PATH and $ORIGIN work for MQ services and triggered applications. For this reason, for Version 7.1 or later, MQ Services and triggered applications run under the user ID that started the queue manager and not `setuid` or `setgid`.

If you are migrating from a release before Version 7.1, if any files used by the service were previously restricted to certain users, then they might not be accessible by the user ID that started the queue manager. Resources used by MQ services or triggered applications must be adjusted as appropriate.

Note:

- On AIX, LD_LIBRARY_PATH is also known as LIBPATH and $ORIGIN is not supported.
- On HP-UX, LD_LIBRARY_PATH is also known as SHLIB_PATH.

Related information

Working with services

Starting IBM MQ applications using triggers

UNIX and Linux: `ps -ef | grep amq` interpretation

From Version 7.1, the interpretation of the list of IBM MQ processes that results from filtering a scan of UNIX or Linux processes has changed. The results can show IBM MQ processes running for multiple installations on a server. Before Version 7.1, the search identified IBM MQ processes running on only a single installation of the product on a UNIX or Linux server.

The implications of this change depend on how the results are qualified and interpreted, and how the list of processes is used. The change affects you, only if you start to run multiple installations on a single server. If you have incorporated the list of IBM MQ processes into administrative scripts or manual procedures, you must review the usage.

Examples

The following two examples, which are drawn from the product documentation, illustrate the point.

1. In the product documentation, before Version 7.1, the scan was used as a step in tasks to change the installation of IBM MQ. The purpose was to detect when all queue managers had ended. In Version 7.1 or later, the tasks use the **dspmq** command to detect when all queue managers associated with a specific installation have ended.

2. In the product documentation, a process scan is used to monitor starting a queue manager in a high availability cluster. Another script is used to stop a queue manager. In the script to stop a queue manager, if the queue manager does not end within a period of time, the list of processes is piped into a **kill -9** command. In both these cases, the scan filters on the queue manager name, and is unaffected by the change to multiple installations.

Related information

Stopping a queue manager under the control of a high availability (HA) cluster

Monitoring a queue manager

UNIX and Linux: /usr symbolic links removed

From Version 7.1, on all UNIX and Linux platforms, the links from the /usr file system are no longer made automatically. In order to take advantage of these links, you must set an installation as the primary installation.

In previous releases the installation of the product on UNIX and Linux created the symbolic links shown in Table 35 on page 210. In Version 7.1 or later, these links are not created. You must run **setmqinst** to create a primary installation containing symbolic links. No symbolic links are created in other installations.

Table 35. Default symbolic links created in releases before Version 7.1

Symbolic link from	To
/usr/bin/amq...	/opt/mqm/bin/amq...
/usr/lib/amq...	/opt/mqm/lib/amq...
/usr/include/cmq...	/opt/mqm/inc/cmq...
/usr/share/man/...	/opt/mqm/man/...

Only a subset of those links created with previous releases are now made, see External library and control command links to primary installation on UNIX and Linux

For more information about choosing whether to have a primary installation, see Choosing a primary installation

Related tasks

"UNIX: Migrating IBM MQ library loading from Version 7.0.1, or later, to the latest version" on page 136
Investigate whether applications connecting to the latest version of the product are linked to, and load libraries from, the correct installation.

Related reference

Primary installation
Explanation of the IBM MQ primary installation.

Related information

Changing the primary installation

setmqinst

Windows and UNIX: Configurable certificate validation policy

From, Version 7.1.0, Fix Pack 2 you configure the product to specify which SSL or TLS certificate validation policy is used to validate digital certificates received from remote partner systems.

Configurable certificate validation policy on Windows and UNIX platforms

If you need to ensure that your certificate validation is compliant with RFC 5280 for improved security in certificate validation, see Configuring certificate validation policies in IBM MQ.

See Certificate validation policies in IBM MQ for more information about IBM MQ certificate validation policies.

Windows: `amqmsrvn.exe` process removed

From Version 7.1, the `amqmsrvn.exe` DCOM process was replaced by a Windows service, `amqsvc.exe`. This change is unlikely to cause any problems. However, you might have to make some changes. You might have configured the user that runs the Windows service `MQSeriesServices` without the user right to "Log on as a service". Alternatively, the user might not have "List Folder" privilege on all the subdirectories from the root of the drive to the location of the service `amqsvc.exe`.

If you omitted the "Log on as a service" user privilege, or one of the subdirectories under which the product is installed does not grant the "List Folder" privilege to the user, the `MQ_`*`InstallationName`* IBM MQ Windows services in Version 8.0 fails to start.

Diagnosing the problem

If the service fails to start, Windows event messages are generated:

- If you did not give the user the "Log on as a service" user privilege, the Windows Service Control Manager adds an event: `7038: The user has not been granted the requested logon type`. The **strmqsvc** command reports error 1069.
- If you did not give the user the "List Folder" privilege, the Windows Service Control Manager adds an event: `7009: Timed out waiting for the service to connect`. The **strmqsvc** command reports error 1053.

If the "Prepare IBM MQ" wizard encounters a failure when validating the security credentials of the user performing an installation, an error is returned: `IBM WebSphere MQ is not correctly configured for Windows domain users`. This error indicates that the service failed to start.

Resolution

To resolve this problem:

- Check that the user has the "Log on as a service" user privilege
- Check that the user is a member of the local mqm group
- Check that the local mqm group has "List Folder" privilege on each subdirectory in the path to the service `amqsvc.exe`.

Related reference

"Windows: Logon as a service required" on page 212
From Version 7.1, the user ID that runs the IBM MQ Windows service must have the user privilege to "Logon as a service". If the user ID does not have the privilege to run the service, the service does not start and it returns an error in the Windows system event log. Typically, you will have run the Prepare IBM MQ wizard, and set up the user ID correctly. Only if you have configured the user ID manually is it possible that you might have a problem in Version 7.1 or later.

Windows: IgnoredErrorCodes registry key

From Version 7.1, the registry key used to specify error codes that you do not want written to the Windows Application Event Log has changed.

The contents of this registry key are not automatically migrated. If you want to continue to ignore specific error codes, you must manually migrate the registry key.

Before Version 7.1, the key was in the following location:

```
HKLM\Software\IBM\MQSeries\CurrentVersion\IgnoredErrorCodes
```

From Version 7.1, the key is in the following location:

```
HKLM\Software\IBM\WebSphere MQ\Installation\MQ_INSTALLATION_NAME\IgnoredErrorCodes
```

where *MQ_INSTALLATION_NAME* is the installation name associated with a particular installation of the product.

Related information

Ignoring error codes under Windows systems

Windows: Installation and infrastructure information

From Version 7.1, the location of Windows installation and infrastructure information has changed.

A top-level string value, `WorkPath`, in the `HKLM\SOFTWARE\IBM\WebSphere MQ` key, stores the location of the product data directory which is shared between all installations. The first installation on a machine specifies it, subsequent installations pick up the same location from this key.

Other information that in releases before Version 7.1 was stored in the registry on Windows is, for Version 7.1 or later, stored in .ini files.

Related information

Changing configuration information on Windows, UNIX and Linux systems

Windows: Local queue performance monitoring

On Windows, from Version 7.1, it is no longer possible to monitor local queues using the Windows performance monitor.

Instead, use the performance monitoring commands, which are common to all platforms, provided by IBM MQ.

Related information

Real-time monitoring

Windows: "Logon as a service" required

From Version 7.1, the user ID that runs the IBM MQ Windows service must have the user privilege to "Logon as a service". If the user ID does not have the privilege to run the service, the service does not start and it returns an error in the Windows system event log. Typically, you will have run the Prepare IBM MQ wizard, and set up the user ID correctly. Only if you have configured the user ID manually is it possible that you might have a problem in Version 7.1 or later.

You have always been required to give the user ID that you configure to run IBM MQ the user privilege to "Logon as a service". If you run the Prepare IBM MQ wizard, it creates a user ID with this privilege. Alternatively, it ensures that a user ID you provide has this privilege.

It is possible that you ran IBM MQ in releases before Version 7.1 with a user ID that did not have the "Logon as a service" privilege. You might have used it to configure the IBM MQ Windows service `MQSeriesServices`, without any problems. If you run an IBM MQ Windows service in Version 7.1 or later with the same user ID that does not have the "Logon as a service" privilege, the service does not start.

The IBM MQ Windows service `MQSeriesServices`, with the display name `MQSeries`, changed in Version 7.1. A single IBM MQ Windows service per server is no longer sufficient. An IBM MQ Windows service per installation is required. Each service is named `MQ_`*`InstallationName`* , and has a display name `IBM WebSphere MQ (` *`InstallationName`*`)`. The change, which is necessary to run multiple installations of IBM MQ, has prevented IBM MQ running the service under a single specific user ID. In Version 7.1 or later, a `MQ_` *`InstallationName`* service must run as a service.

The consequence is a user ID that is configured to run the Windows service `MQ_`*`InstallationName`* must be configured to "Logon as a service". If the user ID is not configured correctly, errors are returned in the Windows system event log.

Many installations on earlier releases, and installations from Version 7.1 onwards, configure IBM MQ with the Prepare IBM MQ wizard. The wizard sets up the user ID with the "Logon as a service" privilege and configures the IBM MQ Windows service with this user ID. Only if, in previous releases, you have configured `MQSeriesServices` with another user ID that you configured manually, might you have this migration problem to fix.

Windows: MSCS restriction with multiple installations

When you install or upgrade to Version 7.1 or later, the first installation of the product on the server is the only one that can be used with Microsoft Cluster Server (MSCS). No other installations on the server can be used with MSCS. This restriction limits the use of MSCS with multiple installations of the product.

When you run the **`haregtyp`** command it defines the first installation of the product to be installed as an MSCS resource type; see IBM MQ MSCS support utility programs. The implications are as follows:

1. You must associate queue managers that are participating in an MSCS cluster with the first installation on the server.
2. Setting the primary installation has no effect on which installation is associated with the MSCS cluster.
3. If you are upgrading from Version 7.0.1 to Version 8.0, you must follow the single-stage migration scenario; see “UNIX, Linux, and Windows: Single-stage migration to a later version” on page 92.

Windows: Migration of registry information

Before Version 7.1, all IBM MQ configuration information, and most queue manager configuration information, was stored in the Windows registry. From Version 7.1 onwards, all queue manager configuration information (for example, `mqs.ini`, `qmstatus.ini`, and `qm.ini`) is stored in files; the same files as in UNIX and Linux platforms.

The change does not affect the operation of existing applications or queue managers, but it does affect any administrative procedures and scripts that reference the registry.

In Version 7.0.1, to support multi-instance (high availability) queue managers, the queue manager configuration information of some queue managers is stored in `qm.ini` and `qmstatus.ini` rather than in the registry.

The Version 7.0.1 configuration information is accessed from other installations. You must stop all the queue managers and IBM MQ applications running on the server to release any locks.

The transfer of configuration data from the registry to files is automatic. It takes place under either of the following circumstances:

- You migrate an existing Windows system to Version 7.1or later.
- Version 7.0.1 is uninstalled from a server that has a Version 7.1, or later, installation.

As a consequence, after uninstallation of Version 7.0.1 on a multi-installation server, it is difficult to restore a Version 7.0.1 installation to run any queue managers that you want to restore to the `701` command level:

1. You cannot reinstall Version 7.0.1 on the server. You must run the queue managers on a different server.
2. When you transfer the queue manager data to another server, with Version 7.0.1 installed, you must create the correct registry configuration entries. The entries are not available to copy from the registry on the multi-installation server. Back up the registry entries before uninstalling Version 7.0.1.

Related tasks

“UNIX, Linux, and Windows: Side-by-side migration to a later version” on page 96

“UNIX, Linux, and Windows: Multi-stage migration to a later version” on page 100

Related information

Changing configuration information on Windows, UNIX and Linux systems

Windows: Relocation of the mqclient.ini file

From Version 7.1, the `mqclient.ini` file has moved from FilePath to WorkPath. This is similar to the model already used on UNIX and Linux systems.

If you supply separate file and work paths, you will see a change in behavior when migrating from a release before Version 7.1. You have an additional step to perform when you choose to uninstall IBM WebSphere MQ 7.0 before installing Version 7.1 or later. Before uninstalling IBM WebSphere MQ 7.0, you must copy `mqclient.ini` directly to the `Config` directory in your data path so that it can be picked up by the Version 7.1 or later installation.

Windows: Task manager interpretation

From Version 7.1, the interpretation of the processes that are listed by the Windows Task Manager has changed. The results can show IBM MQ processes running for multiple installations on a server. Before Version 7.1, the process list identified IBM MQ processes running on only a single installation of the product on a Windows server.

The implications of this change depend on how the results are qualified and interpreted, and how the list of processes is used. The change affects you, only if you start to run multiple installations on a single server. If you have incorporated the list of IBM MQ processes into administrative scripts or manual procedures, you must review the usage.

On 64-bit Windows platforms, IBM MQ 8.0 processes and processes for earlier releases of the product can be easily identified. As queue managers for releases before IBM MQ 8.0 are 32 bit, all queue manager processes for releases before Version 8.0 have '*32' associated with their process names. In Version 8.0, the queue manager is 64 bit, hence the queue manager processes in Windows Task manager do not have an associated '*32'.

Related information

Stopping queue managers in IBM MQ for Windows

Windows: IBM MQ Active Directory Services Interface

From Version 7.1, the IBM WebSphere MQ Active Directory Services Interface is no longer available.

If your are migrating from a release before Version 7.1 and your application uses the IBM WebSphere MQ Active Directory Services Interface, you must rewrite your application to use Programmable Command Formats.

Related information

Introduction to Programmable Command Formats

IBM MQ maintenance tasks (On platforms other than z/OS)

When you apply and remove maintenance level updates to IBM MQ, no migration is required. Maintenance level updates are applied either as a fix pack, or by manually applying an interim fix.

About this task

This information describe how to apply and remove fix packs on platforms other than z/OS.

Follow the appropriate link in Table 36 on page 214 for the platform that your enterprise uses.

Table 36. Applying and removing maintenance

Apply	Remove
AIX	AIX
HP-UX	HP-UX
Linux	Linux

Table 36. Applying and removing maintenance (continued)

Apply	Remove
Solaris	Solaris
Windows	Windows

Related information

Getting product fixes

IBM WebSphere MQ

Applying and removing maintenance level updates (On platforms other than z/OS)

When you apply and remove maintenance level updates to IBM MQ, no migration is required. Maintenance level updates are applied either as a fix pack, or by manually applying an interim fix.

About this task

This information describe how to apply and remove fix packs on platforms other than z/OS.

Follow the appropriate link, to either apply maintenance or restore maintenance to a previous level, for the platform, or platforms, that your enterprise uses.

AIX: Applying maintenance level upgrades on IBM MQ

How to apply maintenance level updates to IBM MQ for AIX using **installp**.

Before you begin

1. Ensure you have enough disk space to apply maintenance level updates. A maintenance level update requires hard disk space for installation. In addition, the installation process might require a similar amount of disk space to save the previous level. For example, a 16 MB update might require 32 MB of space. The additional space allows a maintenance level update to be removed, and the previous level to be restored automatically.
2. If you are running on a server with multiple IBM MQ installations, you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

About this task

Stop applications using the installation and use the **installp** command, to install maintenance level updates to clients and servers. Alternatively, if the installation is in the default installation location, you can use the *System Management Interface Tool*, SMIT.

Important: You cannot go back from a later version of the product to a prior version of the product, for example from IBM MQ 8.0 to IBM WebSphere MQ Version 7.x.

You can apply and remove maintenance from a IBM MQ MQI client that is not installed on the same server as a queue manager. You do not have to stop any queue managers or logon as administrator. Because you do not have to stop any queue managers, do not do steps 1 to 3 in the following maintenance procedure.

Major full versions of the base product are COMMITTED by default. Fix packs on a full base version can be in APPLIED state, and you can go back one release level.

If you need the ability to revert to an earlier version, you should perform a side-by-side migration, and migrate your queue managers to the later version at any time. See "UNIX, Linux, and Windows: Side-by-side migration to a later version" on page 96 for further information.

However, if you start a queue manager under IBM MQ 8.0, that queue manager is automatically migrated, and cannot be downgraded to the previous version.

You require a backup of the IBM WebSphere MQ Version 7.x queue manager and data logs to restore if needed.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM MQ installation.

   If you use the MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM MQ installation.

   a) Run the **dspmq** command to list the state of all the queue managers on the system.

   Run either of the following commands from the installation that you are updating:

   ```
   dspmq -o installation -o status
   dspmq -a
   ```

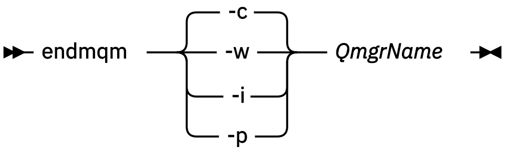
   `dspmq -o installation -o status` displays the installation name and status of queue managers associated with all installations of IBM MQ.

   `dspmq -a` displays the status of active queue managers associated with the installation from which the command is run.

   b) Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

   ```
   echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
   ```

   c) Run the **endmqm** command to stop each running queue manager associated with this installation.

   The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

   For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM MQ resources, such as by stopping the applications.

   You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM MQ. The application is not informed about queue managers in the current installation being shut down.

   Any applications that continue to have IBM MQ shared libraries from the installation loaded prevent you applying IBM MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep an IBM MQ shared library loaded.

   Note: The topic, "Applying maintenance level updates to multi-instance queue managers" on page 249, describes how to apply maintenance to a multi-instance queue manager. A multi-instance queue manager can continue to run on one server, while maintenance is applied to another server.

   d) Stop any listeners associated with the queue managers, using the command:

   ```
   endmqlsr -m QMgrName
   ```

4. Log in as root, or switch to the superuser using the **su** command.
5. Install the update in one of the following ways:
   - Update the whole installation in the default location:

     ```
     installp -agXYd . all
     ```

   - Update selected filesets in the default location:

     ```
     installp -agXYd . list of file sets
     ```

   - Update the whole product in a non-default location using the -R flag:

     ```
     installp -R USIL_Directory -agXYd . all
     ```

   - Update selected filesets in a non-default location using the -R flag:

     ```
     installp -R USIL_Directory -agXYd . list of file sets
     ```

   USIL_Directory is the installation parent directory. IBM MQ is installed underneath the directory. For example, if /USIL1 is specified, the IBM MQ product files are located in /USIL1/usr/mqm. /USIL1/usr/mqm is known as the *MQ_INSTALLATION_PATH*.

Related information
dspmq
Stopping a queue manager

AIX: Restoring the previous maintenance level on IBM MQ

How to restore a previous maintenance level using the *System Management Interface Tool* (SMIT).

Before you begin

1. If you are running on a server with multiple IBM MQ installations, you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

About this task

You can back out maintenance updates (fix packs) and restore your system to the previous maintenance or install level, for any component of IBM MQ for AIX that is in the **APPLIED** state.

You can apply and remove maintenance from a IBM MQ MQI client that is not installed on the same server as a queue manager. You do not have to stop any queue managers or logon as administrator. Because you do not have to stop any queue managers, do not do steps 1 to 3 in the following maintenance procedure.

Use the following command to display the current state of the IBM MQ for AIX filesets:

```
lslpp [ -R usil ] -l "mqm*"
```

To back out a maintenance update, as the user root, issue the command:

```
installp [ -R usil ] -r "mqm*"
```

Otherwise:

Procedure

1. Log in as a user in group mqm.

2. Stop all applications using the IBM MQ installation.

   If you use the MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM MQ installation.
   a) Run the **dspmq** command to list the state of all the queue managers on the system.

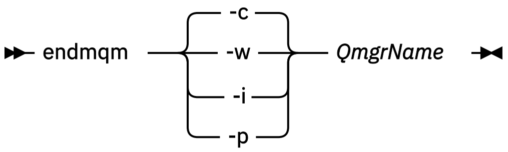
      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM MQ.

      dspmq -a displays the status of active queue managers associated with the installation from which the command is run.
   b) Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```
   c) Run the **endmqm** command to stop each running queue manager associated with this installation.

      The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

      For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM MQ resources, such as by stopping the applications.
      You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM MQ. The application is not informed about queue managers in the current installation being shut down.
      Any applications that continue to have IBM MQ shared libraries from the installation loaded prevent you applying IBM MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep an IBM MQ shared library loaded.

      Note: The topic, "Applying maintenance level updates to multi-instance queue managers" on page 249, describes how to apply maintenance to a multi-instance queue manager. A multi-instance queue manager can continue to run on one server, while maintenance is applied to another server.
   d) Stop any listeners associated with the queue managers, using the command:

      ```
      endmqlsr -m QMgrName
      ```
4. Log in as root, or switch to the superuser using the **su** command.
5. Open the appropriate **smit** panel using this sequence:

   ```
   Software Installation and Maintenance
   ```

```
Software Maintenance and Utilities
Reject Applied Software Updates (Use Previous Version)
```

Alternatively, use a fast path command, `smit[ty] install_update`.

6. Complete the **SOFTWARE** name field.

   Enter `mqm*` to restore all applicable file set updates to your installation.

   Note: If an option to restore only selected file set updates for IBM MQ for AIX appears, avoid it. The option results in all applicable file set updates for the maintenance update being restored.
7. Click **Enter** to reject the current maintenance level and reinstate the previous maintenance or install level.
   a) Accept displayed default values for all other fields
   b) Dismiss the confirmation message

   The reject process starts. While the command runs, it displays progress messages terminating with an **Install Summary** table.

   a) Check the table to see which components of IBM MQ for AIX have been rejected

Related information
dspmq
Stopping a queue manager

HP-UX: Applying maintenance level updates on IBM MQ

How to apply maintenance level updates to IBM MQ for HP-UX using **`swinstall`**.

Before you begin

1. Ensure you have enough disk space to apply maintenance level updates. A maintenance level update requires hard disk space for installation. In addition, the installation process might require a similar amount of disk space to save the previous level. For example, a 16 MB update might require 32 MB of space. The additional space allows a maintenance level update to be removed, and the previous level to be restored automatically.
2. If you are running on a server with multiple IBM MQ installations, you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

You can apply and remove maintenance from a IBM MQ MQI client that is not installed on the same server as a queue manager. You do not have to stop any queue managers or logon as administrator. Because you do not have to stop any queue managers, do not do steps 1 to 3 in the following maintenance procedure.

About this task

1. If you want to install both the base package and the maintenance update packages, install the base package separately first. Then install the maintenance update packages.
2. If you are using the interactive installer, click **Options> Change Options**. Then clear the **autoselect dependencies when marking software** check box before selecting the maintenance update package for installation.
3. Error messages might be seen when running **`swinstall`**, even when successfully updating an installation.

   There are two approaches you can take to handling errors in the application of maintenance.

   a. Aim for an error-free update by applying maintenance only to those components that are installed.
   b. Apply the whole maintenance package and check the error logs, error by error, ignoring the insignificant errors.

   Both approaches are described.

Many of the insignificant errors are caused by **swinstall** trying to apply updates to components that are not installed. Consider whether there are any significant errors reported with the insignificant ones.

- The following errors might not indicate a serious problem. They are written to the console, or to the **swinstall** panel.

  ```
  ERROR:   "hpux11.mycompany.com:/":
  The software dependencies for 15 products or filesets cannot be resolved.
  ```

  ```
  ERROR:   "hpux11.mycompany.com:/":
  17 filesets were determined to be skipped in the analysis phase.
  The execution phase failed for "hpux11.mycompany.com:/".
  Analysis and Execution had errors.
  ```

- The following errors might not indicate a serious problem. They are written to the `swjob` output for a **swinstall** session.

  ```
  ERROR:   17 of 20 filesets had Errors.
  3 of 20 filesets had no Errors or Warnings.
  ```

  ```
  ERROR:   The Execution Phase had errors.
  See the above output for details.
  ```

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM MQ installation.

   If you use the MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM MQ installation.
   a) Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

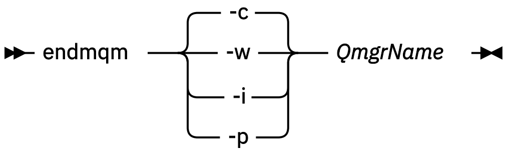
      `dspmq -o installation -o status` displays the installation name and status of queue managers associated with all installations of IBM MQ.

      `dspmq -a` displays the status of active queue managers associated with the installation from which the command is run.

   b) Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```

   c) Run the **endmqm** command to stop each running queue manager associated with this installation.



The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM MQ shared libraries from the installation loaded prevent you applying IBM MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep an IBM MQ shared library loaded.

Note: The topic, "Applying maintenance level updates to multi-instance queue managers" on page 249, describes how to apply maintenance to a multi-instance queue manager. A multi-instance queue manager can continue to run on one server, while maintenance is applied to another server.

d) Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Log in as root, or switch to the superuser using the **su** command.
5. Make your current directory the location of the *Service_update_package*.

   The file name of the *Service_update_package* follows the pattern `hp-Uxxxx.v11`. You must prefix *Service_update_package* with the absolute path to the installation file. To save typing, construct the path using the `$PWD` variable.
6. Run the HP-UX command **swlist** `l= MQ_INSTALLATION_PATH MQSERIES` to list all of the IBM MQ components that are installed.
7. Decide whether to install the updates interactively, and if you want to control which components are updated.

   You can update in the following ways:

   - Silently update all the installed IBM MQ components by installing the whole maintenance package.

     ```
     swinstall -s $PWD/service_update_package
     MQSERIES,l=MQ_INSTALLATION_PATH
     ```

     The **swinstall** command attempts to find an installed component for every component in the update package, and updates it. **swinstall** writes out error messages for components that it cannot find.
   - Silently update some IBM MQ components by installing only the required updates from the maintenance package.

     If you specify *update_components* correctly, the update procedure can be error-free. **swinstall** only updates components that you have listed and components that are dependent on components you have listed.

     a. Using the list of installed IBM MQ components, create a space separated list of the components you want to update (*update_components*). This list requires the installation path of each component to be specified, in the form: *component* `,l= MQ_INSTALLATION_PATH`

     b.
     ```
     swinstall -s $PWD/service_update_package
     update_components
     ```

- Interactively update some IBM MQ components from the maintenance package, selecting only the update components that are required.

  ```
  swinstall -s $PWD/service_update_package
  ```

  a. Open **MQSERIES** and mark the update components you want to apply. Correctly marked, there are no errors when the updates are applied. The installer resolves dependencies automatically.
  b. Select **Actions > Change Product Location** to change the IBM MQ installation you intend to update.
  c. Select **Actions > Install**. The log file tells you if there are any problems that need fixing.

Related information
dspmq
Stopping a queue manager

HP-UX: Restoring the previous maintenance level on IBM MQ

How to restore a previous maintenance level by using **swremove**.

Before you begin

1. If you are running on a server with multiple IBM MQ installations, you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

You can apply and remove maintenance from a IBM MQ MQI client that is not installed on the same server as a queue manager. You do not have to stop any queue managers or logon as administrator. Because you do not have to stop any queue managers, do not do steps 1 to 3 in the following maintenance procedure.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM MQ installation.

   If you use the MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM MQ installation.
   a) Run the **dspmq** command to list the state of all the queue managers on the system.

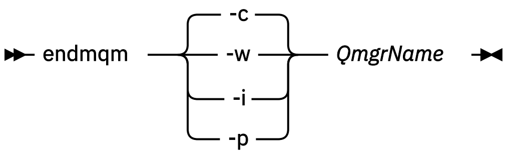
      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

      `dspmq -o installation -o status` displays the installation name and status of queue managers associated with all installations of IBM MQ.

      `dspmq -a` displays the status of active queue managers associated with the installation from which the command is run.
   b) Run the **MQSC** command, `DISPLAY LSSTATUS(*)` STATUS to list the status of listeners associated with a queue manager.

      ```
      echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
      ```
   c) Run the **endmqm** command to stop each running queue manager associated with this installation.



The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM MQ shared libraries from the installation loaded prevent you applying IBM MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep an IBM MQ shared library loaded.

Note: The topic, "Applying maintenance level updates to multi-instance queue managers" on page 249, describes how to apply maintenance to a multi-instance queue manager. A multi-instance queue manager can continue to run on one server, while maintenance is applied to another server.

d) Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Log in as root, or switch to the superuser using the **su** command.
5. Run the swremove command to remove the maintenance package from the system.

   For example, to remove the 7.R.0.1 maintenance level, use the command:

   ```
   swremove MQSERIES,r=7.R.0.1,l= MQ_INSTALLATION_PATH
   ```

   where:

   - R is the number of the Release
   - *MQ_INSTALLATION_PATH* is the installation path for IBM MQ

   Details of the **swremove** command can be found in the *HP-UX Administration Guide* or by using the **man swremove** command.

Related information

dspmq

Stopping a queue manager

IBM i IBM i: Applying maintenance level updates on the latest release

How to apply maintenance level updates by stopping IBM MQ and using the IBM i standard maintenance procedure.

Before you begin

To find out what version you have currently installed, use the following commands:

Table 37. IBM MQ commands to display the installed versions

IBM MQ Product	Version command
IBM MQ Server	`DSPMQVER`

IBM MQ Java	IBM MQ classes for Java:
	`java com.ibm.mq.MQJavaLevel`
	Note: For this command to work, you might need to set your environment classpath to include:
	• `/QIBM/ProdData/mqm/java/lib/com.ibm.mq.jar`
	IBM MQ classes for Java Message Service:
	`java com.ibm.mq.jms.MQJMSLevel`
	Note: For this command to work, you might need to set your environment classpath to include:
	• `/QIBM/ProdData/mqm/java/lib/com.ibm.mqjms.jar`
	See Environment variables relevant to IBM MQ classes for Java and Environment variables relevant to IBM MQ classes for JMS.

IBM MQ Client	`DSPMQVER`

About this task

Maintenance updates for IBM MQ for IBM i are supplied as PTFs (Program Temporary Fixes). They are available for download from the web as save files, which are normally stored in the QGPL library. IBM i PTF's can be found in "Fix Central" at the following location:

FixCentral.

Procedure

1. Read the cover letter carefully to see if you need to take any special actions.
2. Sign on to a new interactive IBM i session, ensuring that you are not accessing any IBM MQ objects.
3. Ensure that you have:
   a) *ALLOBJ authority, or object management authority for the QMQM library.
   b) Sufficient authority to use the ENDSBS command.
4. Warn all users that you are going to stop IBM MQ.
5. Use the ENDMQM command to quiesce all queue managers:

   ```
   ENDMQM MQMNAME(*ALL) OPTION(*CNTRLD) ENDCCTJOB(*YES) RCDMQMIMG(*YES)
   TIMEOUT( 15 )
   ```

   Where *15* is a timeout value in seconds.

   If the ENDMQM command has not completed within a reasonable period (at least 10 minutes), use the WRKMQM command. This command identifies the queue managers that are still ending. Then force each one in turn to stop by issuing:

```
ENDMQM MQMNAME( QMGRNAME ) OPTION(*IMMED)
```

Where *QMGRNAME* is the name of the queue manager.

Complete the tidying up of shared memory by issuing the command:

```
ENDMQM MQMNAME(*ALL) OPTION(*IMMED) ENDCCTJOB(*YES) RCDMQMIMG(*NO)
TIMEOUT( 15 )
```

6. If the command in the previous step does not complete, end the subsystem immediately by issuing:

```
ENDSBS SBS(QMQM) OPTION(*IMMED)
```

7. If this command also fails, use the operating system command ENDJOB to end all jobs in the subsystem QMQM, as described in the following steps.

   Note: Do not use ENDJOBABN unless you intend to perform an IPL on the machine before starting IBM MQ. Ending IBM MQ jobs using ENDJOBABN can lead to damaged semaphores, which in turn can prevent your queue manager from starting.

   a) If a QMGR must be shut down manually, the recommended order of ending jobs (ENDJOB) is shown in the list that follows. Wait a few minutes for AMQA* or AMQZ* jobs to tidy up.

      i) RUNMQLSR - TCP listener (multi-threaded)
      ii) AMQCLMAA - TCP listener (single-threaded)
      iii) AMQRMPPA - Channel process pooling job
      iv) RUNMQCHI - channel initiator
      v) AMQCRSTA - receiving MCA jobs
      vi) RUNMQCHL - sending MCA jobs
      vii) AMQCRS6B - LU62 receiver channel
      viii) AMQPCSEA - command server
      ix) RUNMQTRM - Application trigger monitor
      x) RUNMQDLQ - Dead letter queue handler
      xi) AMQFCXBA - IBM Integration Bus Worker Job
      xii) AMQFQPUB - Queued Publish/Subscribe Daemon
      xiii) RUNMQBRK - IBM Integration Bus Control Job
      xiv) AMQZMUC0 ('0' is a zero) - Utility Manager
      xv) AMQZMUF0 ('0' is a zero) - Utility Manager
      xvi) AMQZMUR0 ('0' is a zero) - Utility Manager
      xvii) AMQZMGR0 ('0' is a zero) - Process Controller
      xviii) AMQRRMFA - cluster repository manager
      xix) AMQZDMAA - deferred message manager
      xx) AMQALMPX - Log Manager
      xxi) AMQZFUMA - object authority manager
      xxii) AMQZLSA0 ('0' is a zero) - LQM agents
      xxiii) AMQZLAA0 ('0' is a zero) - LQM agents
      xxiv) AMQZXMA0 ('0' is a zero) - Execution Controller

   b) Issue the following command:

   ```
   ENDMQM MQMNAME( QMGRNAME ) OPTION(*IMMED)
   ```

c) Issue the following command:

```
ENDMQM MQMNAME(*ALL) OPTION(*CNTRLD) ENDCCTJOB(*YES) RCDMQMIMG(*NO)
TIMEOUT( 05 )
```

Where *05* is a timeout value in seconds.

d) Manually clean up shared memory. Issue the following command:

```
EDTF '/QIBM/UserData/mqm/qmgrs'
```

then:

i) Take option 5 for **&SYSTEM** and check that the following directories are empty: `isem, esem, msem, ssem,` and `shmem`.

ii) Take option 5 for **QMGRNAME** and check that the following directories are empty:- `isem, esem, msem, ssem,` and `shmem`.

iii) Take option 5 for **&ipcc** in the QMGRNAME directory and check that the following directories are empty:- `isem, esem, msem, ssem,` and `shmem`.

iv) Take option 5 for **&qmpersist** in the QMGRNAME directory and check that the following directories are empty:- `isem, esem, msem, ssem,` and `shmem`.

v) Take option 5 for **&app** and check that the following directories are empty: `isem, esem, msem, ssem,` and `shmem`.

8. Load and apply a PTF

IBM i: Restoring a queue manager from the latest release to a previous release

You can restore a queue manager to the previous version of the product from the latest version, if you have made a backup of the system or queue manager. If you have started the queue manager and processed any messages, or changed the configuration, the task cannot give you any guidance on restoring the current state of the queue manager.

Before you begin

1. You must have made a backup of the system or queue manager before you upgraded to the latest version. For more information see Backing up and restoring IBM MQ queue manager data
2. If any messages were processed after starting the queue manager, you cannot easily undo the effects of processing the messages. You cannot restore the queue manager to the previous version of the product in its current state. The task cannot give you any guidance how to deal with subsequent changes that have occurred. For example, messages that were indoubt in a channel, or in a transmission queue on another queue manager, might have been processed. If the queue manager is part of a cluster, then configuration messages and application messages might have been exchanged.

About this task

When you restore a previous version of a queue manager, you restore the queue manager to its earlier code level. Queue manager data is restored to the state it was in when the queue manager was backed up.

Procedure

1. Stop the queue manager.
2. If you performed a slip install, you must reinstall IBM MQ.
   a) Uninstall the previous installation.
   b) Reinstall IBM MQ from a manufacturing refresh.

    c) Apply the fix pack and interim fixes that restore IBM MQ to its previous level.
    d) Restore the queue manager data from the backup taken before installing the latest version.

3. Restart the previous version queue manager.

Related information

Backing up and restoring a queue manager

Linux: Applying maintenance level updates on IBM MQ

How to apply maintenance level updates to IBM MQ for Linux using RPM. The following procedure applies to all Linux platforms, including Ubuntu.

Before you begin

If you are running on a server with multiple IBM MQ installations, you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

About this task

Maintenance level updates are delivered in the form of Red Hat Package Manager (RPM) update images, which are applied using the RPM installation tool.

You can apply and remove maintenance from a IBM MQ MQI client that is not installed on the same server as a queue manager. You do not have to stop any queue managers or logon as administrator. Because you do not have to stop any queue managers, do not do steps 1 to 3 in the following maintenance procedure.

Important: **pax** and **rpmbuild** are not supplied as part of the product. You must obtain these from your Linux distribution supplier.

Additional disk space is required for the update images to allow maintenance level updates to be removed and the previous level restored. The updated files are kept in *MQ_INSTALLATION_PATH*/maintenance directory. Do not delete or move this directory or the files it contains.

MQ_INSTALLATION_PATH represents the high-level directory in which IBM MQ is installed.

Updates are cumulative. You can apply your chosen update directly, without applying any previous updates first. The maintenance level updates might contain updates for one or more packages. You must apply those parts of an update that correspond to the packages that are applied in your installation.

Important: Although it is possible to install a fix pack at the same level as an installation performed from a manufacturing refresh image at that level, you should not attempt this process. Installing a fix pack at the same level as the one already on your system, can leave the package management database of your system in an inconsistent state with respect to the installation of IBM MQ.

Procedure

1. Log in as a user in group mqm.
2. Stop all applications using the IBM MQ installation.

   If you use the MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM MQ installation.

    a) Run the **dspmq** command to list the state of all the queue managers on the system.

    Run either of the following commands from the installation that you are updating:

```
dspmq -o installation -o status
dspmq -a
```

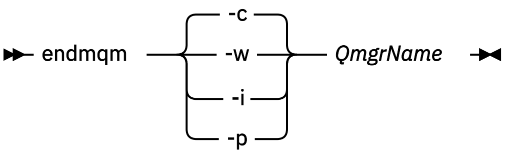
    dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM MQ.

    dspmq -a displays the status of active queue managers associated with the installation from which the command is run.

b) Run the **MQSC** command, DISPLAY LSSTATUS(*) STATUS to list the status of listeners associated with a queue manager.

```
echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
```

c) Run the **endmqm** command to stop each running queue manager associated with this installation.



The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM MQ shared libraries from the installation loaded prevent you applying IBM MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep an IBM MQ shared library loaded.

Note: The topic, “Applying maintenance level updates to multi-instance queue managers” on page 249, describes how to apply maintenance to a multi-instance queue manager. A multi-instance queue manager can continue to run on one server, while maintenance is applied to another server.

d) Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Log in as root, or switch to the superuser using the **su** command.
5. Change into the directory containing the maintenance packages.
6. Run the ls command to list the available updates.

   For example, if there are level 1 maintenance updates for the Runtime, SDK and Server packages, you see the following:

   ```
   MQSeriesRuntime-Uxxxx-V.R.0-1.i386.rpm
   MQSeriesSDK-Uxxxx-V.R.0-1.i386.rpm
   MQSeriesServer-Uxxxx-V.R.0-1.i386.rpm
   ```

   where V is the version number and R is the number of the Release.
7. Run the **rpm** command to find out which packages are installed on your server.

   Enter the following command:

   ```
   rpm -qa | grep MQSeries
   ```

   Notes:

a. If you are using Ubuntu, add the **--force-debian** attribute.

```
rpm --force-debian -qa | grep MQSeries
```

b. If you are using Linux on POWER® Systems - Little Endian , add the **--ignorearch** attribute.

You must include this option to prevent problems with some levels of rpm not recognizing the Linux on POWER Systems - Little Endian architecture.

For example, if you have a minimum IBM MQ installation and SDK component, at level 0, the **rpm** command returns:

```
MQSeriesRuntime-V.R.0-0
MQSeriesSDK-V.R.0-0
MQSeriesServer-V.R.0-0
```

where V is the version number and R is the number of the Release.

8. If this fix pack is to be upgraded on an installation, other than the first installation on the system, run the **crtmqfp** command to create and use a unique set of packages to install on the system.

   Note, that if this is the first, or only, IBM MQ installation on the system, you can ignore this step.

   You must install the **pax** command in order for the **crtmqfp** command to run on Linux.

   a) Run the command `./crtmqfp` *<suffixname>* where *suffixname* is the same as the suffix used during renaming of the base level IBM MQ installation.

   b) Set your current directory to the location specified when the **crtmqfp** command completes.

   This directory is a subdirectory of `/var/tmp/mq_rpms`, in which the unique set of packages is created. The packages have the suffix value contained within the filename.

   For example, if you used suffix 1 during repackaging of the base level IBM MQ installation, enter the command: `./crtmqfp 1`.

   There is now a subdirectory named `/var/tmp/mq_rpms/1/xxxx`, and the packages will be renamed, for example, from `MQSeriesRuntime-V.R.0-1.xxxx.rpm` to `MQSeriesRuntime_1-V.R.0-1.xxxx.rpm`. Where V is the version number and R is the number of the Release.

9. Run the **rpm** command to apply all available updates for the packages you have on your system:

   - To update an installation in the default location, `/opt/mqm`:

     ```
     rpm -ivh MQSeriesRuntime-Uxxxx-V.R.0-1.i386.rpm
       MQSeriesSDK-Uxxxx-V.R.0-1.i386.rpm
       MQSeriesServer-Uxxxx-V.R.0-1.i386.rpm
     ```

     where V is the version number and R is the number of the Release.

   - To update an installation in a custom location, specify the **rpm** `prefix` option:

     ```
     rpm --prefix /opt/customLocation -ivh MQSeriesRuntime-Uxxxx-V.R.0-1.i386.rpm
       MQSeriesSDK-Uxxxx-V.R.0-1.i386.rpm
       MQSeriesServer-Uxxxx-V.R.0-1.i386.rpm
     ```

     where V is the version number and R is the number of the Release.

   You must apply all packages in a maintenance update that correspond to those packages that are currently installed on your system.

10. Repeat step “7” on page 228 to list the packages that are now available.

    The Runtime, SDK, and Server packages are now at level 1:

    ```
    MQSeriesRuntime-V.R.0-0
    MQSeriesSDK-V.R.0-0
    MQSeriesServer-V.R.0-0
    ```

```
MQSeriesRuntime-Uxxxx-V.R.0-1
MQSeriesSDK-Uxxxx-V.R.0-1
MQSeriesServer-Uxxxx-V.R.0-1
```

where V is the version number and R is the number of the Release.

Note:

After the installation of IBM MQ fix packs, if you run the `rpm-verify` or `rpm -V` command, it does not return the correct results. It produces spurious results relating to missing files in *MQ_INSTALLATION_PATH*/maintenance.

This error message can be ignored because it is a known limitation in the IBM MQ fix pack installation code. For further information about this error, see IBM MQ Fix Pack install errors - Linux reports errors

What to do next

For further information about using RPM to install software packages, see your Linux documentation.

Related information

dspmq

Stopping a queue manager

Linux: Restoring the previous maintenance level on IBM MQ

How to remove updates and restore the previous maintenance level using **RPM**. The following instructions apply to all Linux platforms, including Ubuntu.

Before you begin

If you are running on a server with multiple IBM MQ installations, you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

About this task

When maintenance is applied, the original versions of replaced files are saved to allow the updates to be removed if necessary. To restore the previous maintenance level, run an Red Hat Package Manager, RPM, uninstall command for all the packages that were updated by the maintenance package as follows:

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM MQ installation.

   If you use the MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM MQ installation.

   a) Run the **dspmq** command to list the state of all the queue managers on the system.

      Run either of the following commands from the installation that you are updating:

      ```
      dspmq -o installation -o status
      dspmq -a
      ```

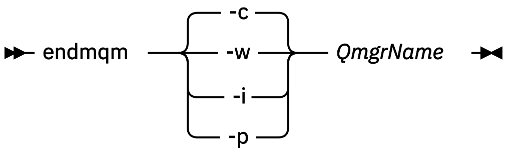
      `dspmq -o installation -o status` displays the installation name and status of queue managers associated with all installations of IBM MQ.

      `dspmq -a` displays the status of active queue managers associated with the installation from which the command is run.

b) Run the **MQSC** command, DISPLAY LSSTATUS(*) STATUS to list the status of listeners associated with a queue manager.

```
echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
```

c) Run the **endmqm** command to stop each running queue manager associated with this installation.



The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

> For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM MQ resources, such as by stopping the applications.
>
> You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM MQ. The application is not informed about queue managers in the current installation being shut down.
>
> Any applications that continue to have IBM MQ shared libraries from the installation loaded prevent you applying IBM MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep an IBM MQ shared library loaded.

Note: The topic, "Applying maintenance level updates to multi-instance queue managers" on page 249, describes how to apply maintenance to a multi-instance queue manager. A multi-instance queue manager can continue to run on one server, while maintenance is applied to another server.

d) Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Log in as root, or switch to the superuser using the **su** command.
5. Run the **rpm** command to find out which packages are installed on your server.

   Enter the following command:

   ```
   rpm -qa | grep MQSeries
   ```

   Note: If you are using Ubuntu, add the **--force-debian** attribute.

   ```
   rpm --force-debian -qa | grep MQSeries
   ```

   Using the example given in "Linux: Applying maintenance level updates on IBM MQ" on page 227, returns:

   ```
   MQSeriesRuntime-V.R.0-0
   MQSeriesSDK-V.R.0-0
   MQSeriesServer-V.R.0-0
   MQSeriesRuntime-Uxxxx-V.R.0-1
   MQSeriesSDK-Uxxxx-V.R.0-1
   MQSeriesServer-Uxxxx-V.R.0-1
   ```

   where V is the version number and R is the number of the Release.
6. Run the **rpm** command to remove all the updates applied at level 1.

Enter the following commands:

```
rpm -ev MQSeriesRuntime-Uxxxx-V.R.0-1 MQSeriesSDK-Uxxxx-V.R.0-1
MQSeriesServer-Uxxxx-V.R.0-1
```

where V is the version number and R is the number of the Release.

7. Repeat step 5 to check that the ptf packages have been removed, leaving only the original install packages:

```
MQSeriesRuntime-V.R.0-0
MQSeriesSDK-V.R.0-0
MQSeriesServer-V.R.0-0
```

where V is the version number and R is the number of the Release.

What to do next

For further information on using RPM to install software packages, see your Linux documentation.

Related information

dspmq

Stopping a queue manager

Solaris: Applying maintenance level updates on IBM MQ

How to apply maintenance level updates to IBM MQ for Solaris using **pkgadd**.

Before you begin

1. Ensure you have enough disk space to apply maintenance level updates. A maintenance level update requires hard disk space for installation. In addition, the installation process might require a similar amount of disk space to save the previous level. For example, a 16 MB update might require 32 MB of space. The additional space allows a maintenance level update to be removed, and the previous level to be restored automatically.
2. If you are running on a server with multiple IBM MQ installations, you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

You can apply and remove maintenance from a IBM MQ MQI client that is not installed on the same server as a queue manager. You do not have to stop any queue managers or logon as administrator. Because you do not have to stop any queue managers, do not do steps 1 to 3 in the following maintenance procedure.

About this task

Stop applications using the installation and use **pkgadd** to install maintenance.

Important: Although it is possible to install a fix pack at the same level as an installation performed from a manufacturing refresh image at that level, you should not attempt this process. Installing a fix pack at the same level as the one already on your system, can leave the package management database of your system in an inconsistent state with respect to the installation of IBM MQ.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM MQ installation.

   If you use the MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM MQ installation.
   a) Run the **dspmq** command to list the state of all the queue managers on the system.

Run either of the following commands from the installation that you are updating:

```
dspmq -o installation -o status
dspmq -a
```

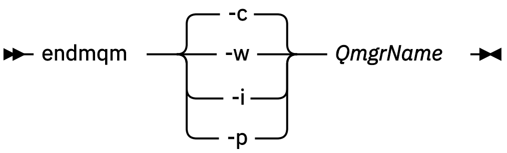
dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM MQ.

dspmq -a displays the status of active queue managers associated with the installation from which the command is run.

b) Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

```
echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
```

c) Run the **endmqm** command to stop each running queue manager associated with this installation.



The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM MQ shared libraries from the installation loaded prevent you applying IBM MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep an IBM MQ shared library loaded.

Note: The topic, "Applying maintenance level updates to multi-instance queue managers" on page 249, describes how to apply maintenance to a multi-instance queue manager. A multi-instance queue manager can continue to run on one server, while maintenance is applied to another server.

d) Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Log in as root, or switch to the superuser using the **su** command.
5. Change into the directory containing the maintenance packages.
6. Run the **crtmqfp** command to create and use a unique set of packages to install on the system, if this fix pack is to be upgraded on a installation that is not the first installation on the system.

   This command creates and uses a unique set of packages to install on the system.

   a) Run the command **crtmqfp** mqm- *<suffixname>* where *suffixname* is the same as the suffix used during renaming of the base level IBM MQ installation.

   Note that this command creates a full copy of the installation packages in a subdirectory of `/var/tmp`.

   b) Set your current directory to the location specified when the **crtmqfp** command completes.

This directory is a subdirectory of `/var/spool`, in which the unique set of packages is created. The packages have the suffix value contained within the filename.

7. Proceed with installation using the following command:

   Enter the following command to start the installation process if this fix pack is to be upgraded on an installation that is

   a) The first installation on the system:

   ```
   pkgadd -d packagename
   ```

   where `packagename` corresponds to the image file name. For example:

   ```
   mqm-U1234.img
   ```

   b) Not the first installation on the system:

   ```
   pkgadd mqm-suffixname
   ```

   where `suffixname` is the name of the directory created in `/var/spool/pkg`.

   For example, if you install IBM WebSphere MQ 7.0 as a package called `mqm-main7` and create a package to upgrade to IBM WebSphere MQ 7.0.0.1, using the command **crtmqfp** `mqm-main7`, package *mqm-main7-07-00-00-01* is created in `/var/spool/pkg`.

   To install package *mqm-main7-07-00-00-01*, issue the command **pkgadd** `mqm-main7-07-00-00-01`.

   For further information about using **pkgadd** to install software packages, see the Solaris documentation.

8. Follow the on-screen instructions.

Related information

dspmq

Stopping a queue manager

Solaris: Applying maintenance level updates on IBM MQ in non-interactive mode

You can install IBM MQ for Solaris non-interactively by creating a response file and an admin file.

Before you begin

1. Ensure you have enough disk space to apply maintenance level updates. A maintenance level update requires hard disk space for installation. In addition, the installation process might require a similar amount of disk space to save the previous level. For example, a 16 MB update might require 32 MB of space. The additional space allows a maintenance level update to be removed, and the previous level to be restored automatically.
2. If you are running on a server with multiple IBM MQ installations, you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

You can apply and remove maintenance from a IBM MQ MQI client that is not installed on the same server as a queue manager. You do not have to stop any queue managers or logon as administrator. Because you do not have to stop any queue managers, do not do steps 1 to 3 in the following maintenance procedure.

About this task

Stop applications using the installation and use **pkgadd** to install maintenance.

Important: Although it is possible to install a fix pack at the same level as an installation performed from a manufacturing refresh image at that level, you should not attempt this process. Installing a fix pack at the same level as the one already on your system, can leave the package management database of your system in an inconsistent state with respect to the installation of IBM MQ.

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM MQ installation.

   If you use the MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM MQ installation.

   a) Run the **dspmq** command to list the state of all the queue managers on the system.

   Run either of the following commands from the installation that you are updating:

   ```
   dspmq -o installation -o status
   dspmq -a
   ```

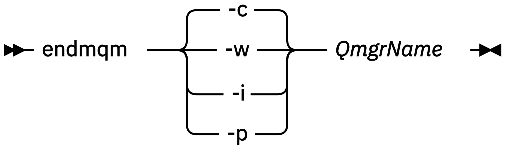
   `dspmq -o installation -o status` displays the installation name and status of queue managers associated with all installations of IBM MQ.

   `dspmq -a` displays the status of active queue managers associated with the installation from which the command is run.

   b) Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

   ```
   echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
   ```

   c) Run the **endmqm** command to stop each running queue manager associated with this installation.

   The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

   For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM MQ resources, such as by stopping the applications.

   You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM MQ. The application is not informed about queue managers in the current installation being shut down.

   Any applications that continue to have IBM MQ shared libraries from the installation loaded prevent you applying IBM MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep an IBM MQ shared library loaded.

   Note: The topic, "Applying maintenance level updates to multi-instance queue managers" on page 249, describes how to apply maintenance to a multi-instance queue manager. A multi-instance queue manager can continue to run on one server, while maintenance is applied to another server.

   d) Stop any listeners associated with the queue managers, using the command:

   ```
   endmqlsr -m QMgrName
   ```

4. Log in as root, or switch to the superuser using the **su** command.
5. Change into the directory containing the maintenance packages.
6. Run the **`crtmqfp`** command to create and use a unique set of packages to install on the system, if this fix pack is to be upgraded on a installation that is not the first installation on the system.

   This command creates and uses a unique set of packages to install on the system.

   a) Run the command **`crtmqfp`** mqm- *<suffixname>* where *suffixname* is the same as the suffix used during renaming of the base level IBM MQ installation.

      Note that this command creates a full copy of the installation packages in a subdirectory of `/var/tmp`.

   b) Set your current directory to the location specified when the **`crtmqfp`** command completes.

      This directory is a subdirectory of `/var/spool`, in which the unique set of packages is created. The packages have the suffix value contained within the filename.

7. Create the non-interactive install response file using the **pkgask** command. Enter the following command to create the response file if this fix pack is to be upgraded on an installation that is:

   a) The first installation on the system:

      ```
      pkgask -d <location_to_image>/imagefile -r response.txt packagename
      ```

      where *imagefile* corresponds to the image file name, for example `mqm-U200403.img`, `response.txt` is the name of the response file to create, and *packagename* is the fix pack package name, for example `mqm-07-05-00-02`.

   b) Not the first installation on the system:

      ```
      pkgask -d /var/spool/pkg -r response.txt mqm-suffixname
      ```

      where `/var/spool/pkg` is the location of the new package, `response.txt` is the name of the response file to create, and *suffixname* is the name of the directory created in `/var/spool/pkg`.

8. Find the `admin_file` from the server installation media located at <install_media>/silent/admin or create an `admin_file` in the following format:

   ```
   mail=
   instance=unique
   partial=ask
   runlevel=ask
   idepend=ask
   rdepend=ask
   space=ask
   setuid=nocheck
   conflict=nocheck
   action=nocheck
   basedir=default
   ```

9. Run the **pkgadd** command to apply the maintenance level update IBM MQ for Solaris in non-interactive mode. Enter the following command to start the installation process if this fix pack is to be upgraded on an installation that is:

   a) The first installation on the system:

      ```
      pkgadd -v -n -r response.txt -a admin_file -d <location_to_image>/imagefile packagename
      ```

      where *admin_file* is a path qualified name of the admin file you created, and *packagename* corresponds to the fix pack package being installed.

   b) Not the first installation on the system:

      ```
      pkgadd -v -n -r response.txt -a admin_file -d /var/spool/pkg mqm-suffixname
      ```

10. Follow the on-screen instructions.

Related information

dspmq

Stopping a queue manager

Solaris: Restoring the previous maintenance level on IBM MQ

How to restore a previous maintenance level by stopping IBM MQ and using **pkgrm**.

Before you begin

If you are running on a server with multiple IBM MQ installations, you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.

About this task

When maintenance is applied, the original versions of replaced files are saved to allow the updates to be removed if necessary. To restore the previous maintenance level, run **pkgrm** command for all the packages that were updated by the maintenance package as follows:

Procedure

1. Log in as a user in `group mqm`.
2. Stop all applications using the IBM MQ installation.

   If you use the MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM MQ installation.

   a) Run the **dspmq** command to list the state of all the queue managers on the system.

   Run either of the following commands from the installation that you are updating:

   ```
   dspmq -o installation -o status
   dspmq -a
   ```

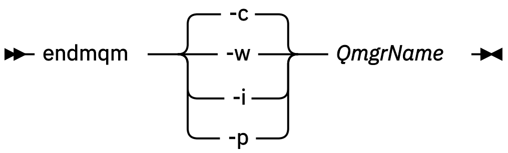
   dspmq -o installation -o status displays the installation name and status of queue managers associated with all installations of IBM MQ.

   dspmq -a displays the status of active queue managers associated with the installation from which the command is run.

   b) Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

   ```
   echo "DISPLAY LSSTATUS(*) STATUS" | runmqsc QmgrName
   ```

   c) Run the **endmqm** command to stop each running queue manager associated with this installation.



   The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

   For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM MQ shared libraries from the installation loaded prevent you applying IBM MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep an IBM MQ shared library loaded.

Note: The topic, "Applying maintenance level updates to multi-instance queue managers" on page 249, describes how to apply maintenance to a multi-instance queue manager. A multi-instance queue manager can continue to run on one server, while maintenance is applied to another server.

d) Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Log in as root, or switch to the superuser using the **su** command.
5. Run the **pkgrm** command to remove the latest maintenance update from the system:

```
pkgrm packagename
```

packagename is the name of the package that you want to remove; for example, `mqm-07-R-00-01`, where R is the number of the Release.

Details of the **pkgrm** command can be found in the Solaris documentation, or by using the **man pkgrm** command.

If you do not know the name of the package to remove, try listing the packages that are installed using the following command: `pkginfo | grep mqm`

Note: Ignore any error messages of the form `<shared pathname not removed>`.

What to do next

If you have installed an IBM MQ MQI client, and the client was updated after installing the maintenance level that is being removed, you must specifically update your IBM MQ MQI client installation again, after the maintenance level has been removed

Related information

dspmq

Stopping a queue manager

Windows Applying and removing maintenance on Windows

Maintenance tasks associated with IBM MQ on Windows are grouped in this section.

Procedure

- To apply maintenance level server updates, see "Windows: Applying maintenance level server updates " on page 239.
- To apply maintenance level client updates, see " Windows: Applying maintenance level client updates" on page 248.
- To remove updates and revert to the previous maintenance level using the Windows installer, see "Windows: Removing maintenance level server updates" on page 244.
- For information on how to use multiple installations of IBM MQ on the same server to control the release of maintenance fixes, see "UNIX, Linux, and Windows: Staging maintenance fixes" on page 264.

- For information on how to use use multi-instance queue managers to reduce the outage caused by applying maintenance updates, see “Applying maintenance level updates to multi-instance queue managers” on page 249.

Related tasks

“Querying the maintenance level” on page 264
Query the IBM MQ maintenance level by running the **`dspmqver`** command

Windows *Applying and removing maintenance level server updates on Windows*

On Windows, you can apply maintenance level server updates to IBM MQ either interactively or silently. You can also remove updates from a server and revert to the previous maintenance level of IBM MQ. You can use multi-instance queue managers to reduce the outage caused by applying maintenance updates.

Windows: Applying maintenance level server updates
How to apply maintenance level updates to IBM MQ for Windows.

Before you begin

1. If you are running on a server with multiple IBM MQ installations, you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.
2. Download the maintenance package from the IBM MQ Support website.
3. If User Account Control (UAC) is enabled, the user who does the installation must have Administrative authority. You must elevate any command or command prompt by selecting **Run as Administrator**. If you do not, the error AMQ4353 is written in the installation log

Procedure

1. Log on as an Administrator.
2. Stop all applications using the IBM MQ installation.

   If you use the MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM MQ installation.

   a) Run the **dspmq** command to list the state of all the queue managers on the system.

   Run either of the following commands from the installation that you are updating:

   ```
   dspmq -o installation -o status
   dspmq -a
   ```

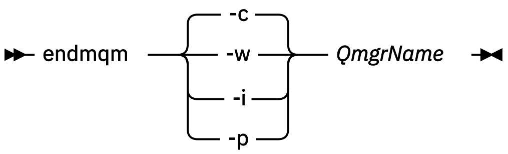
   `dspmq -o installation -o status` displays the installation name and status of queue managers associated with all installations of IBM MQ.

   `dspmq -a` displays the status of active queue managers associated with the installation from which the command is run.

   b) Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

   ```
   echo DISPLAY LSSTATUS(*) STATUS | runmqsc QmgrName
   ```

   c) Run the **endmqm** command to stop each running queue manager associated with this installation.



The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM MQ resources, such as by stopping the applications.

You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM MQ. The application is not informed about queue managers in the current installation being shut down.

Any applications that continue to have IBM MQ shared libraries from the installation loaded prevent you applying IBM MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep an IBM MQ shared library loaded.

Note: The topic, "Applying maintenance level updates to multi-instance queue managers" on page 249, describes how to apply maintenance to a multi-instance queue manager. A multi-instance queue manager can continue to run on one server, while maintenance is applied to another server.

d) Stop any listeners associated with the queue managers, using the command:

```
endmqlsr -m QMgrName
```

4. Stop the IBM MQ service for the installation.

   a) Right-click the **WebSphere MQ** icon in the taskbar > click **Stop WebSphere MQ**.

5. Load and apply the maintenance files for server installations:

   - Interactively:

     a. Open the folder where the maintenance package has been extracted.
     b. Right-click on the maintenance program and select **Run as administrator** to start the loading process.
     c. Choose your installation language, and click **OK**.
     d. Continue to follow the instructions on screen.

     If you choose to load the files without applying them to an installation, you can apply the files later, as described in step "6" on page 242

   - Silently:

     a. Open the folder where the maintenance package has been extracted.
     b. Modify the response file, `silent_install.resp`. For details on the properties you can specify in the response file, see Table 38 on page 241

Table 38. Properties used to install or uninstall a maintenance update

Property	Value	Description
MQPLOG	*path\file_name*	Pass a valid path to specify the log to be used during installation/uninstallation, for example MQPLOG="C:\TEMP\UPDATEINSTALL.LOG" If MQPLOG is not specified (which is the case if you start maintenance by clicking on the **Apply fix pack n.n.n.n** icon in the IBM WebSphere MQ program group) the log name used by default will be amqicsdn.txt in your TEMP directory (%TEMP%).
MQPINSTALLATIONNAME	*Installation name*	The name of the installation to maintain. If there is only 1 installation (of any level) on the machine, this argument can be safely omitted. If there is more than 1 installation on the machine, amqicsdn.exe checks the value of MQPINSTALLATIONNAME. If one is not supplied, or the one that is supplied is unsuitable, then a GUI selection box appears. This selection box provides a list of installations to which this fix pack is applicable. If none are applicable, then amqicsdn.exe issues error message AMQ4781 and ends.
MQPBACKUPPATH	*path*	Specifies the directory to back up into during installation, for example MQPBACKUPPATH="C:\BACKUP" The directory, and any intermediate directories, you specify must already exist. If any one of the directories does not already exist, the install fails.
MQPREBOOT	0|1	Specifies what to do when a reboot is required, for example MQPREBOOT=1. If no value is supplied, you are prompted for what to do next. If MQPREBOOT is set to 0, a reboot is suppressed If MQPREBOOT is set to 1, the reboots go ahead without prompting.
MQPINUSEOK	0|1	Specifies whether to continue even if a file is found to be currently locked by another application. If you choose to continue even if a file is locked by another application, then you must reboot to complete fix pack installation. If no value is supplied, or if MQPINUSEOK is set to 0, the installation fails if files are found to be in use by other applications. If MQPINUSEOK is set to 1, the installation is deferred until you reboot.

c. Open an elevated command prompt in the directory where the maintenance program was extracted.

d. Start the silent loading by entering the following command:

```
executableName -f responseFile
```

where:

- *executableName* is the name of the maintenance package. For example, for Version 8.0.0, Fix Pack 2: `8.0.0-WS-MQ-Windows-FP0002.exe`.
- *responseFile* is the full path and name of the response file.

6. Optional: Apply the maintenance to other server installations on the system:

   - Interactively:

     a. From the Windows start menu, select **Start > Programs > IBM WebSphere MQ > Apply Fix Pack *<V.R.M.L>***

        where

        V is the version number
        R is the release number
        M is the modification number
        L is the level of modification

     b. Continue to follow the instructions on screen.

   - Silently:

     a. Open an elevated command prompt and navigate to the directory where the maintenance program was loaded. By default, the path is `C:\Program Files (x86)\IBM\source\WebSphere MQ <V.R.M.L>`

        where

        V is the version number
        R is the release number
        M is the modification number
        L is the level of modification

     b. Enter the following command:

        ```
        amqicsdn MQPINSTALLATIONNAME=
        name MQPSILENT=1
        ```

        where *name* is the name of the installation that you want to apply maintenance to.

        You can add other properties to the command, as listed in Table 38 on page 241.

7. Optional: Uninstall the fix pack files from your machine.

   After installing the fix pack files and applying the maintenance to all the server installations that you want to update, you can either uninstall the fix pack files from your machine or leave them installed for future use.

   Note: Uninstalling these files does NOT remove them from the installations to which you have already applied the maintenance. If that is what you intend, you should instead follow the instructions in "Windows: Removing maintenance level server updates" on page 244.

   If you add any installable features at a later time, you must reapply the maintenance to update the added feature(s).

   - Interactively:
     - To uninstall the fix pack files, on the **Control Panel... Programs and Features** panel, select **WebSphere MQ (fix pack 8.0.0.x) files**.
     - Click **Uninstall**.
   - Silently:

To silently remove the fix pack files, enter the following command from an admin command prompt:

```
fixpack_files_installdir\_IBM WebSphere MQ (fix pack V.R.M.F files)_installation\Change
IBM WebSphere MQ (fix pack V.R.M.F files) Installation.exe" -i silent
```

where

- *fixpack_files_installdir* is the location in which you chose to install the files. For IBM MQ 8.0.0 Fix Pack 16, for example, this might be: `C:\Program Files\IBM\source\WebSphere MQ 8.0.0.1`.
- *V.R.M.F* is the fix pack number. For IBM MQ 8.0.0 Fix Pack 16, for example, this is `8.0.0.1`.

Note: The fix pack files contain a JRE, so if you chose not to install a JRE in your IBM MQ installation for local policy reasons, you may want to uninstall the fix pack files as soon as you have finished applying the update to your installation(s).

What to do next

On a server installation, you must restart the IBM WebSphere MQ taskbar application manually after the maintenance application completes.

If the IBM WebSphere MQ service is manually stopped, it must be manually restarted on the server. If the IBM WebSphere MQ service is not manually stopped, it is restarted automatically on the server.

The taskbar application is not restarted for any logged in sessions. Start the taskbar application in one of three ways:

1. Start the taskbar application manually from the start menu.
2. Log off and log back on again.
3. Run the command:

```
MQ_INSTALLATION_PATH\bin64\amqmtbrn.exe -Startup
```

Related information

dspmq

Stopping a queue manager

Windows V 8.0.0.11 *Installation of Microsoft Visual Studio 2013 C/C++ runtimes in a server fix pack*

From IBM MQ 8.0.0, Fix Pack 11, some elements of IBM MQ require the Microsoft Visual Studio 2013 C/C++ (VS2013) runtimes. If the VS2013 runtimes are not present on the machine, they are installed when you install a fresh client or server, upgrade an earlier release to these levels, or apply a fix pack. In most cases, the installation of the runtimes does not require any manual intervention. However, if the installation fails when you are applying a server fix pack, you can set the **`IBM_MQ_SKIP_VS2013`** environment variable so that you can install the runtimes manually instead.

`IBM_MQ_SKIP_VS2013` environment variable

The **`IBM_MQ_SKIP_VS2013`** environment variable is provided in case there is a problem with the runtime installation provided by the patch applier, `amqicsdn.exe`.

You can optionally set **`IBM_MQ_SKIP_VS2013`** before the applying the IBM MQ server fix pack to influence the installation of the VS2103 runtimes. Setting **`IBM_MQ_SKIP_VS2013`** to any value causes `amqicsdn.exe` to skip the installation of the VS2103 runtimes.

Note: IBM MQ will not function properly without the VS2103 runtimes. If you set **`IBM_MQ_SKIP_VS2013`**, it is then your responsibility to install these runtimes manually.

If you do have an issue with the installation of the runtimes with `amqicsdn.exe`, you must complete the following steps:

1. Install the runtimes yourself.

The Microsoft redistributable runtime installers can be downloaded from the Microsoft web site, or can be found in the loaded fix pack files at the following (default) location: `C:\Program Files\IBM\source\MQ 8.0.0.11\prereqs\VS2013`.

2. Set **`IBM_MQ_SKIP_VS2013`** (to any value).
3. Rerun `amqicsdn.exe`.

Related messages

There are three IBM MQ messages related to the installation of the VS2103 runtimes that might be issued by the server fix pack:

AMQ4754

`Install Microsoft Visual Studio 2013 C/C++ runtimes?....`

This message is issued when the patch applier, `amqicsdn.exe`, has checked your machine and determined that it does need to install the 32 bit runtimes, the 64 bit runtimes, or both. As the runtimes are required, answering `NO` to this prompt exits the installation. (If you do not want this to happen, for example because you want install the runtimes yourself, set **`IBM_MQ_SKIP_VS2013`** as already described before rerunning `amqicsdn.exe`.)

AMQ4755

`Failed to install Microsoft Visual Studio 2013 C/C++ runtimes....`

The patch applier, `amqicsdn.exe`, determined that the 32 bit and/or 64 bit runtimes were not present on this machine and tried to install them. However, something went wrong during the installation and the process ended. If this happens you have two options:

- Consult the logs produced by `amqicsdn.exe` and the Microsoft redistributable installers, both of which can be found in your %TEMP% directory, and diagnose why this occurred.
- Install the runtimes yourself, then set **`IBM_MQ_SKIP_VS2013`** to any value, and rerun `amqicsdn.exe`.

AMQ4741

`The MQ %s update was stopped....`

This message is issued after you answer `NO` to the message AMQ4754. It reminds you that you must install the runtimes.

Windows: Removing maintenance level server updates
How to remove updates and restore the previous maintenance level using the Windows installer

Before you begin

Important: This task describes how to remove maintenance level updates for an IBM MQ server. For information about how to remove maintenance level updates for an IBM MQ client, see "Reverting a client to an earlier maintenance level on Windows" on page 248.

1. If you are running on a server with multiple IBM MQ installations, you must identify the installation. Make sure that the commands you enter run against the correct installation; see setmqenv.
2. If User Account Control (UAC) is enabled, the user who does the installation must have Administrative authority. You must elevate any command or command prompt by selecting **Run as Administrator**. If you do not, the error AMQ4353 is written in the installation log

About this task

If you applied maintenance to IBM MQ, you can restore IBM MQ to a previous level of maintenance.

Important: If you installed the server at a particular maintenance level, a *Manufacturing Refresh*, you cannot restore IBM MQ to an earlier maintenance level.

Procedure

1. Log on as an Administrator.
2. Stop all applications using the IBM MQ installation.

   If you use the MQ Managed File Transfer (MFT) component, ensure that any MFT agents have finished all of the file transfers that they were engaged in. There should be no incomplete transfers associated with the agents, and their SYSTEM.FTE.STATE queues should contain no messages.
3. End all the activity of queue managers associated with the IBM MQ installation.

   a) Run the **dspmq** command to list the state of all the queue managers on the system.

   Run either of the following commands from the installation that you are updating:

   ```
   dspmq -o installation -o status
   dspmq -a
   ```

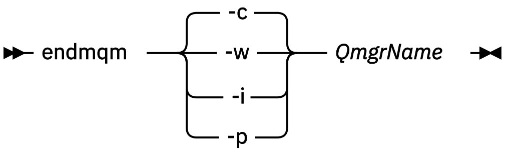
   `dspmq -o installation -o status` displays the installation name and status of queue managers associated with all installations of IBM MQ.

   `dspmq -a` displays the status of active queue managers associated with the installation from which the command is run.

   b) Run the **MQSC** command, `DISPLAY LSSTATUS(*) STATUS` to list the status of listeners associated with a queue manager.

   ```
   echo DISPLAY LSSTATUS(*) STATUS | runmqsc QmgrName
   ```

   c) Run the **endmqm** command to stop each running queue manager associated with this installation.



   The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

   For the maintenance to proceed, applications must respond to an **endmqm** command by disconnecting from the queue manager and releasing any IBM MQ libraries they have loaded. If they do not, you must find another way to force applications to release IBM MQ resources, such as by stopping the applications.

   You must also stop applications that are using the client libraries that are part of the installation. Client applications might be connected to a different queue manager, running a different installation of IBM MQ. The application is not informed about queue managers in the current installation being shut down.

   Any applications that continue to have IBM MQ shared libraries from the installation loaded prevent you applying IBM MQ maintenance. An application might disconnect from a queue manager, or be forcibly disconnected, but keep an IBM MQ shared library loaded.

   Note: The topic, "Applying maintenance level updates to multi-instance queue managers" on page 249, describes how to apply maintenance to a multi-instance queue manager. A multi-instance queue manager can continue to run on one server, while maintenance is applied to another server.

   d) Stop any listeners associated with the queue managers, using the command:

   ```
   endmqlsr -m QMgrName
   ```
4. Stop the IBM MQ service for the installation.

   a) Right-click the **WebSphere MQ** icon in the taskbar > click **Stop WebSphere MQ**.

5. Remove the maintenance interactively, or silently using a command.
   - Interactively:
     a. For each installation of IBM MQ that has had maintenance applied, you are presented with one of the following icons in the Windows start menu:
        i) **Start > Programs > IBM WebSphere MQ > Remove Refresh Pack <V.R.M.L>** ***(installation name)***
        ii) **Start > Programs > IBM WebSphere MQ > Remove Fix Pack <V.R.M.L>** ***(installation name)***

        where

        V is the version number
        R is the release number
        M is the modification number
        L is the level of modification

     b. Select the installation you want to maintain and click **Remove** to start the process.

     This returns the installation to the state it was in before the maintenance package was applied.
   - Silently:
     a. Open an elevated command prompt and enter the following command:

```
amqicsdn.exe MQPINSTALLATIONNAME= name MQPUNINST=1 MQPSILENT=1
```

where *name* is the name of the installation that you want to remove maintenance from.

You can add other properties to the command, as listed in Table 39 on page 246.

Table 39. Properties used to install or uninstall a maintenance update

Property	**Value**	**Description**
MQPLOG	*path\file_name*	Pass a valid path to specify the log to be used during installation/uninstallation, for example MQPLOG="C:\TEMP\UPDATEINSTALL.LOG" If MQPLOG is not specified (which is the case if you start maintenance by clicking on the **Apply fix pack n.n.n.n** icon in the IBM WebSphere MQ program group) the log name used by default will be amqicsdn.txt in your TEMP directory (%TEMP%).
MQPINSTALLATIONNAME	*Installation name*	The name of the installation to maintain. If there is only 1 installation (of any level) on the machine, this argument can be safely omitted. If there is more than 1 installation on the machine, amqicsdn.exe checks the value of MQPINSTALLATIONNAME. If one is not supplied, or the one that is supplied is unsuitable, then a GUI selection box appears. This selection box provides a list of installations to which this fix pack is applicable. If none are applicable, then amqicsdn.exe issues error message AMQ4781 and ends.

Table 39. Properties used to install or uninstall a maintenance update (continued)

Property	Value	Description
MQPBACKUPPATH	*path*	Specifies the directory to back up into during installation, for example MQPBACKUPPATH="C:\BACKUP" The directory, and any intermediate directories, you specify must already exist. If any one of the directories does not already exist, the install fails.
MQPREBOOT	0|1	Specifies what to do when a reboot is required, for example MQPREBOOT=1. If no value is supplied, you are prompted for what to do next. If MQPREBOOT is set to 0, a reboot is suppressed If MQPREBOOT is set to 1, the reboots go ahead without prompting.
MQPINUSEOK	0|1	Specifies whether to continue even if a file is found to be currently locked by another application. If you choose to continue even if a file is locked by another application, then you must reboot to complete fix pack installation. If no value is supplied, or if MQPINUSEOK is set to 0, the installation fails if files are found to be in use by other applications. If MQPINUSEOK is set to 1, the installation is deferred until you reboot.

6. Optional: If you no longer need the maintenance files that were loaded onto the system before maintenance was applied, you can remove them using **Add/Remove programs** or **Programs and Features** from the Control Panel.

   If you want to remove a maintenance file silently, run the following command:

   ```
   <patch_install_files>\_WebSphere MQ (fix pack <V.R.M.L> files)_installation\Change WebSphere
   MQ
   (fix pack <V.R.M.L> files) Installation.exe" -i silent
   ```

   where <patch_install_files> is the installation directory where maintenance files are installed.

   By default, this directory is `C:\Program Files (x86)\IBM\source\WebSphere MQ <V.R.M.L>`

   Notes:

   a. Run the command from outside the directory, otherwise the directory is not removed.
   b. If you omit **-i silent**, the command initiates the Graphical User Interface installer.

What to do next

On a server installation, you must restart the IBM WebSphere MQ taskbar application manually after the maintenance application completes.

If the IBM WebSphere MQ service is manually stopped, it must be manually restarted on the server. If the IBM WebSphere MQ service is not manually stopped, it is restarted automatically on the server.

The taskbar application is not restarted for any logged in sessions. Start the taskbar application in one of three ways:

1. Start the taskbar application manually from the start menu.

2. Log off and log back on again.
3. Run the command:

```
MQ_INSTALLATION_PATH\bin64\amqmtbrn.exe -Startup
```

Related information

dspmq
Stopping a queue manager

Windows *Updating the maintenance level of a client on Windows*

On Windows, you can apply maintenance level client updates to IBM MQ either interactively or silently. You can also remove maintenance level client updates and revert to the previous maintenance level of IBM MQ.

Windows: Applying maintenance level client updates
How to apply maintenance level updates to IBM MQ for Windows clients.

About this task

You can apply maintenance to an IBM MQ client interactively or by using the **msiexec** command to perform a silent MSI upgrade.

Interactive client upgrade

On the client installation media, navigate to the \Windows\MSI\ directory, then run the Setup.exe file.

Silent client upgrade

As an alternative method for applying maintenance to IBM MQ client systems, you can use the command **msiexec** from the command line to perform a silent MSI upgrade.

To upgrade a computer with only a single installation, you can use a command similar to the following example:

```
msiexec /i "PATH\Windows\MSI\IBM WebSphere MQ.msi" /l*v <install_log_path>
/q TRANSFORMS="1033.mst" REINSTALL=ALL REINSTALLMODE=vomus
```

For a multi installation computer with multiple clients, you can upgrade a single client by using a command similar to the following example:

```
msiexec /i "PATH\Windows\MSI\IBM WebSphere MQ.msi" /l*v <install_log_path>
/q TRANSFORMS=":InstanceId2.mst;1033.mst" REINSTALL=ALL REINSTALLMODE=vomus
```

Results

When the maintenance completes you can query the maintenance level by running the **dspmqver** command. For more details, see "Querying the maintenance level" on page 264.

Windows *Reverting a client to an earlier maintenance level on Windows*

For a client, the way in which you revert to an earlier maintenance level depends on whether your client installation was installed from the server DVD or eImage, or from the MQC8 IBM MQ Clients SupportPac.

About this task

If your client installation was installed from the IBM MQ server DVD or eImage, you can revert to an earlier maintenance level by rolling back the fixpack installation.

Important: Client only installations installed from the MQC8 IBM MQ Clients SupportPac cannot be restored to an earlier maintenance level on Windows. If you need to do this, you must first uninstall the currently installed version of the client and then reinstall client at the appropriate earlier maintenance level.

Procedure

To revert to an earlier maintenance level for a client installation that you installed from the server DVD or eImage:

- Log on as an Administrator.
- Remove the maintenance either interactively or silently (see "Windows: Removing maintenance level server updates" on page 244).

To revert to an earlier maintenance level for a client only installation that you installed from the MQC8 IBM MQ Clients SupportPac:

- Uninstall the current version of the client as described in Uninstalling IBM MQ on Windows.
- Reinstall the client at the earlier maintenance level.

Applying maintenance level updates to multi-instance queue managers

Use multi-instance queue managers to reduce the outage caused by applying maintenance updates. Follow these steps to apply maintenance to a multi-instance queue manager.

Before you begin

Maintenance is applied to the IBM MQ installation on a server and not to individual queue managers. You must stop all the queue managers, and any IBM MQ service, on a server before you apply maintenance.

If you want a queue manager to keep running while maintenance is applied, you must configure it as a *multi-instance* queue manager, and have a standby instance running on another server. If a queue manager is an existing single instance queue manager, you must convert it to a multi-instance queue manager. See the topic, Multi-instance queue managers for prerequisites and guidance how to create a multi-instance queue manager.

You can create a multi-instance queue manager from v7.0.1 onwards. If you are running multi-instance queue managers, you then can apply a maintenance update to a *running* queue manager by switching the active instance to a different server.

Typically active and standby installations are maintained at the same maintenance level. Consult the maintenance instructions for each upgrade. Consult the instructions to see if it is possible to run the active and standby instances at different maintenance levels. Check whether fail over from higher to lower, or only lower to higher maintenance level is possible.

The instructions for applying a maintenance update might require you to stop a multi-instance queue manager completely.

If you have a primary server for running active queue manager instances, and a secondary server that runs standby instances, you have a choice of updating the primary or secondary server first. If you update the secondary server first, you must switch back to the primary server when both servers have been updated.

If you have active and standby instances on several servers, you must plan in what order you update the servers to minimize the disruption caused by ending the active instances on each server you update.

About this task

Combine the steps in this task with the maintenance update procedure for applying maintenance to an IBM MQ server installation.

Procedure

1. Where the maintenance update procedure instructs you to stop all running queue managers, or quiesce IBM MQ do the following instead:

The maintenance update procedure varies by platform; see “Applying and removing maintenance level updates (On platforms other than z/OS)” on page 215.

a) If the queue manager is running as standby:

- IBM i On IBM i, end the standby by adding the `INSTANCE(*STANDBY)` option to the **ENDMQM** command.
- On Windows, UNIX, and Linux platforms, end the standby with the **endmqm** `-x` *QMgrName* command.

b) If the queue manager is running as the active instance:

- IBM i On IBM i, end the instance and transfer control to the standby instance by adding the `ALWSWITCH(*YES)` option to the **ENDMQM** command.

  If there is no standby instance running, the command fails, and you must start a standby instance on a different server.
- On Windows, UNIX, and Linux platforms, end the instance and transfer control to the standby instance with the **endmqm** command. For example, **endmqm** *-shutdown_option* `-s` *QMgrName*, where *-shutdown_option* is an optional parameter specifying the type of shutdown. For more information, see endmqm.

  If there is no standby instance running, the command fails, and you must start a standby instance on a different server.

c) If a queue manager is running as a single instance queue manager, you have no alternative but to stop the queue manager before applying the maintenance update.

When you complete this step, no queue manager instances are left running on the server you intend to update.

2. Continue with the maintenance update procedure, following the step to issue the **endmqm** command, or quiesce IBM MQ and apply maintenance to the IBM MQ server.
3. When you have completed the maintenance update, restart all the queue managers on the IBM MQ server, permitting standby instances:
   - IBM i On IBM i, add the `STANDBY(*YES)` option to the **STRMQM** command
   - On Windows, UNIX, and Linux platforms, use the **strmqm** `-x` *QmgrName* command.
4. Repeat the procedure on the standby server, to update its maintenance level.
5. If necessary, switch the active instances back to the primary servers:
   - IBM i On IBM i, use the **ENDMQM** command with the `ALWSWITCH(*YES)` option, and the restart the instances using the **STRMQM** command with the `STANDBY(*YES)` option.
   - On Windows, UNIX, and Linux platforms, use the **endmqm** *-shutdown_option* `-s` *QMgrName* command, and the restart the instances using the **strmqm** `-x` *QmgrName* command.

Related tasks

“Applying and removing maintenance level updates (On platforms other than z/OS)” on page 215
When you apply and remove maintenance level updates to IBM MQ, no migration is required. Maintenance level updates are applied either as a fix pack, or by manually applying an interim fix.

Migrating queue managers to new-function fix packs

This scenario illustrates running different levels of queue manager from a single installation using new-function fix packs. It contrasts migrating a queue manager to new command levels in new-function fix packs, to migrating a queue manager to a new command level in a new release. The scenario explains the relationship between new-function fix packs and maintenance fix packs.

z/OS New function fix-packs are not available on z/OS. On z/OS, the same result is achieved by using the **OPMODE** parameter to control queue manager migration; see “z/OS: OPMODE” on page 272.

Before you begin

In this section, IBM WebSphere MQ 7.1 is used as the current release, and the release is denoted by `r`; the subsequent release is denoted by R.

The scenario starts with a single installation of IBM WebSphere MQ 7.1, `Inst_1`. `Inst_1` is the primary installation; see Figure 21 on page 255. For illustration, there are two queue managers, `QM1` and `QM2`. `QM1` stays at the `7r0` command level, `QM2` moves to the highest command level available.

The use of version numbers and command levels is illustrative, and does not imply anything about future releases.

About this task

Figure 20 on page 252 has time advancing down the Y-Axis, as new fix packs are released. On the X-Axis are different command levels. As a queue manager is migrated to a new command level, it shifts across the diagram. Each column represents the fix levels a queue manager at a particular command level can run at.

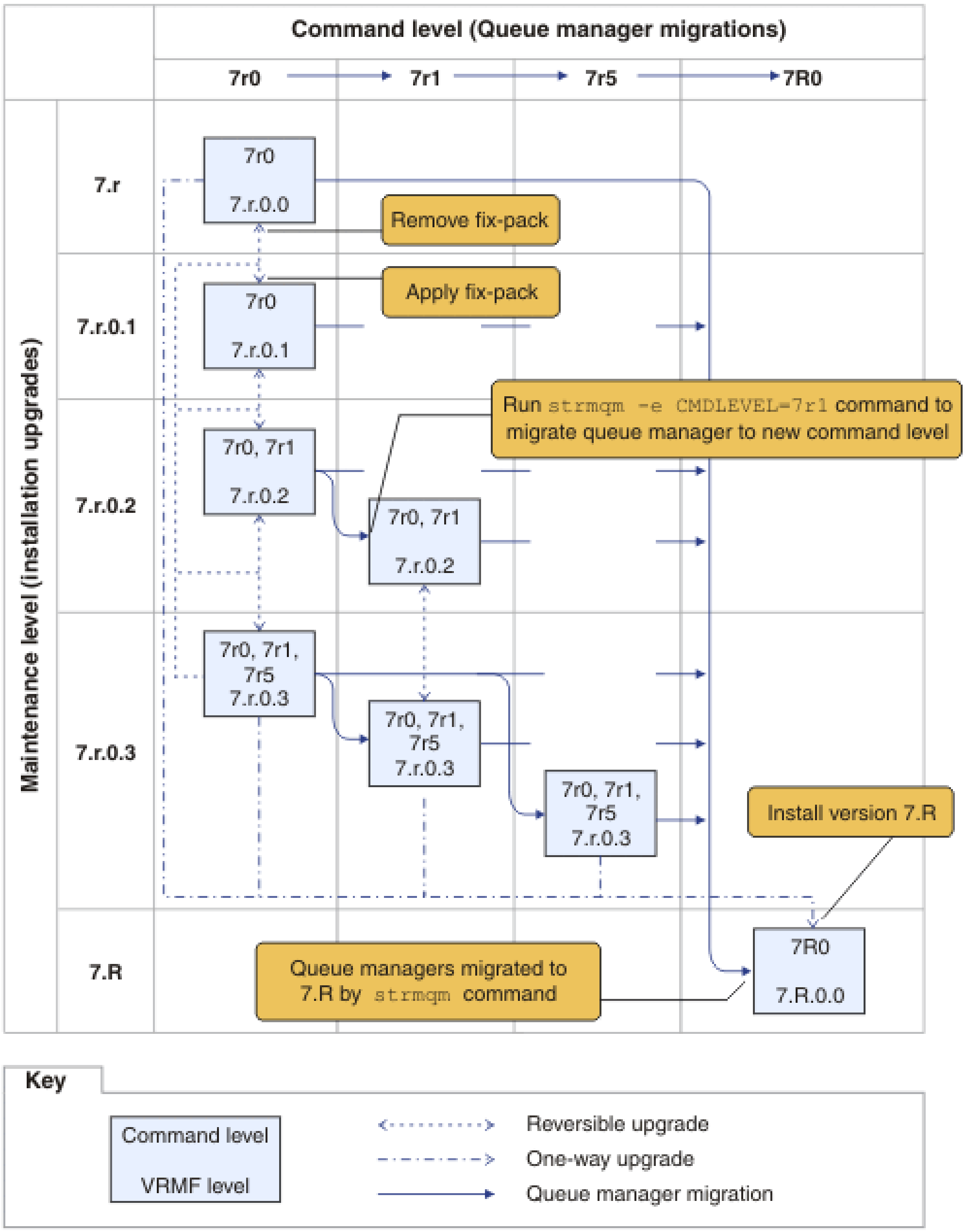


Figure 20. Migration of queue managers to new command levels using new-function fix packs

Figure 20 on page 252 is a little complicated, but it captures lots of details about new-function fix packs to help you remember them. The steps in the task explain the details in the figure. Some of the principle features of Figure 20 on page 252 are explained in the following list:

Maintenance level and Command level

The maintenance level is a fix pack with a V.R.M.F. code; see "The version naming scheme for IBM MQ (On platforms other than z/OS)" on page 21. V.R.M.F codes are one to four digits, always punctuated by periods. Trailing zeros are sometimes omitted in descriptions, but never when a V.R.M.F code is used to label a fix pack. Version 8.0 is an example of using a V.R.M.F code to describe the version of IBM MQ.

The command level is the command level property of a queue manager; see CommandLevel (MQLONG). Command levels are three-digit codes.

Command levels and versions are related. Up to Version 7.1 the command level and the first three digits of the V.R.M.F. code always matched. From Version 7.1, with the introduction of new-function fix packs, the command level of a queue manager can be greater than the first three digits of an installation. The difference arises, if the queue manager has been associated with a new command level using the **strmqm** command.

From Version 7.1 the rule that links command levels and V.R.M.F levels has been changed. The rule is that when a new version of IBM MQ is released, it has a command level greater than released in a new-function fix pack in the previous release. Usually this means that a new release of IBM MQ changes the version or release level, rather than the maintenance level.

In Figure 20 on page 252, the maintenance level, on the Y-Axis is labeled with V.R.M.F codes, and the command level, on the X-Axis, with command levels. Note how the illustrative release of `7.R` increases the released command level from `7r0` to `7R0`, and `7R0` exceeds the highest command level shipped in a new-function fix pack, `7r5`.

Reversible and One-way upgrades

The mechanism to apply and remove fix packs varies by platform. You can apply any fix pack that changes only the maintenance or fix level of a release to an installation. Fix pack application is reversible. When you remove a fix pack, you restore the previous release level. So applying `7.r.0.3` to `7.r.0.1`, and then removing it, restores the installation to `7.r.0.1`.

Sometimes, you can change an installation to a particular V.R.M.F level by upgrading the installation with a "manufacturing refresh". If you install a manufacturing refresh, you can only return to the earlier release level by uninstalling, and reinstalling; see “Upgrade, migration, and maintenance of IBM MQ (On platforms other than z/OS)” on page 51.

Applying a manufacturing refresh to modify the maintenance and fix level of a release is the same process as upgrading to a new version or release of IBM MQ. Neither can be reversed without uninstalling.

However there is a particular aspect of upgrading to a new version or release that is different from upgrading to a new maintenance or fix level. If you start a queue manager after a version or release upgrade, the command level of the queue manager is automatically increased. You can then no longer start the queue manager with the installation from the previous release.

On the diagram, an irreversible upgrade is shown by the "One-way" arrow between `7.r` and `7.R`. To prevent an accidental migration, you can rename the new installation. After renaming, rerun the **setmqm** command to associate a queue manager with the new release before running the **strmqm** command to migrate it.

If the upgrade applies only to the maintenance or fix level, you can restart the queue manager with the previous installation, if you reinstall it.

Manufacturing refresh maintenance releases are not distinguished from applying and removing fix packs on the diagram. Both are represented by reversible arrows in Figure 20 on page 252.

Multiple installations

You might choose to have a different installation for each maximum command level supported by an installation. Each column on the diagram would represent a different installation.

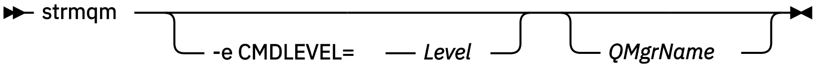
You need only one installation at Version 7.1 to be able to select any command level released with Version 7.1 for a queue manager. Eventually, if you intend to run Version 7.1 and version 7.R in parallel, you must have two installations. The scenario that follows uses a single installation.

Another variation is to follow the "rolling fix pack" approach described in “UNIX, Linux, and Windows: Staging maintenance fixes” on page 264. You can maintain two installations at Version 7.1, one at the current fix level, and one either at a later or earlier fix level. You might then install version 7.R as a third installation, or replace the Version 7.1 installation at the older fix level.

Migrating queue managers

The migration paths for queue managers are shown by solid arrows on the diagram. Some of the solid arrows are broken, to avoid cluttering the diagram with too many lines. If the migration to a higher command level jumps command levels, you do not have to migrate it through the intervening commands levels.

To migrate a queue manager to a higher command level in a new-function fix pack, you must start the queue manager with a special parameter:



Level is the three-digit command level.

The queue manager stops immediately the migration process is complete. When you next start it, it runs at the new command level. The queue manager cannot be restarted at a lower command level. This rule means that you must associate the queue manager with an installation that includes a command level at least as great as the current command level of the queue manager.

Restoring queue managers

To restore a queue manager to a lower command level, you must back up the queue manager before you migrate it to the higher command level.

Procedure

This procedure keeps both `QM1` and `QM2` at the current maintenance level, `QM1` at command level `7r0`, and `QM2` at the latest command level.

1. Download the next fix pack for the version of your product, for example, 7.1.0.2, when it is released.

   See Fix Central.

   The initial system has two queue managers running `7.r.0.0` at command level `7r0` ; see Figure 21 on page 255.

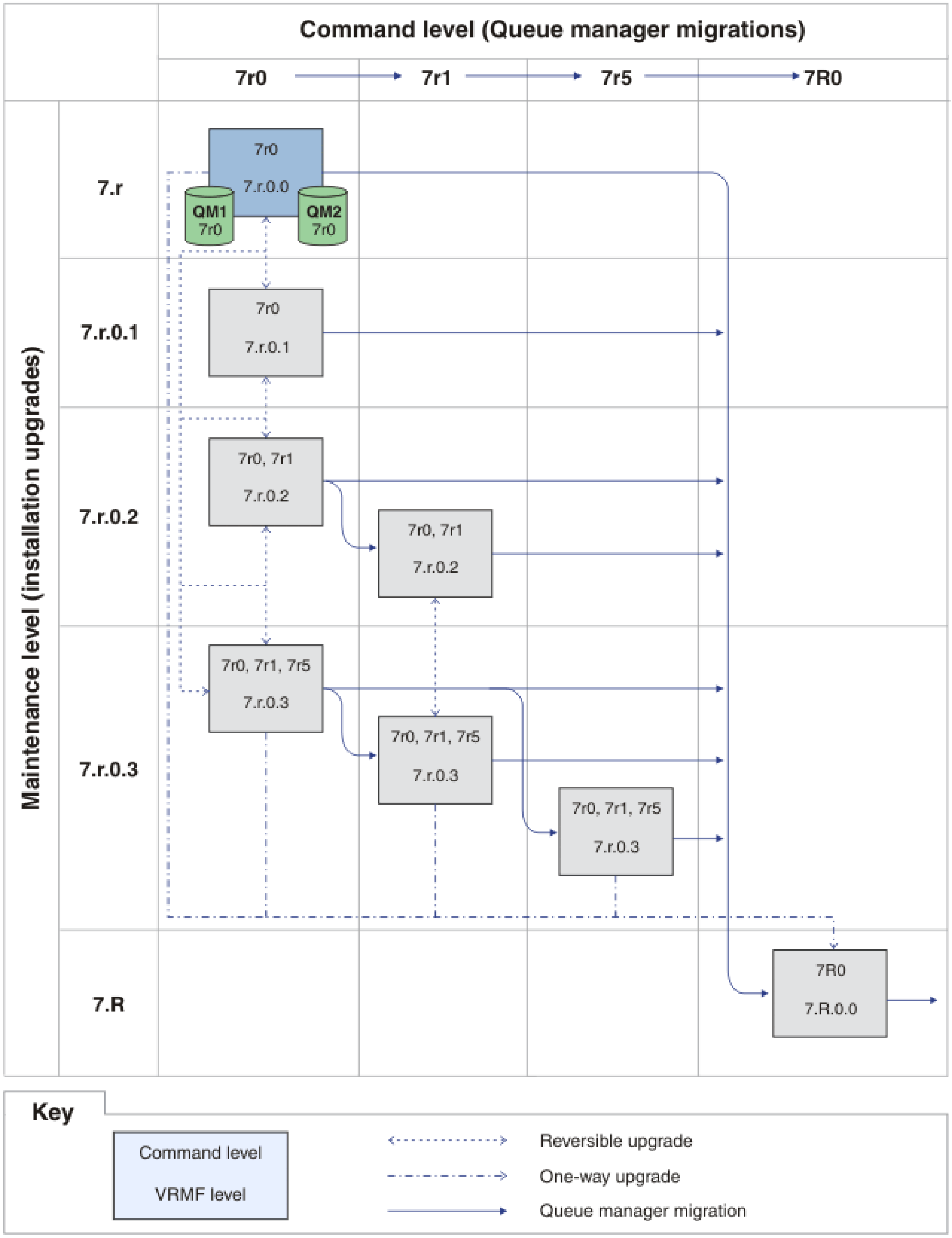


Figure 21. Initial state, QM1 and QM2 at command level `7r0`, and fix level `7.r.0.0`

2. Apply fix pack 7.r.0.1 to `Inst_1`.

   Follow the task for your platform in "Applying and removing maintenance level updates (On platforms other than z/OS)" on page 215.

3. Restart the queue managers.

   Both queue managers are now running using `Inst_1` at the `7.r.0.1` maintenance level, and the `7r0` command level; see Figure 22 on page 256.

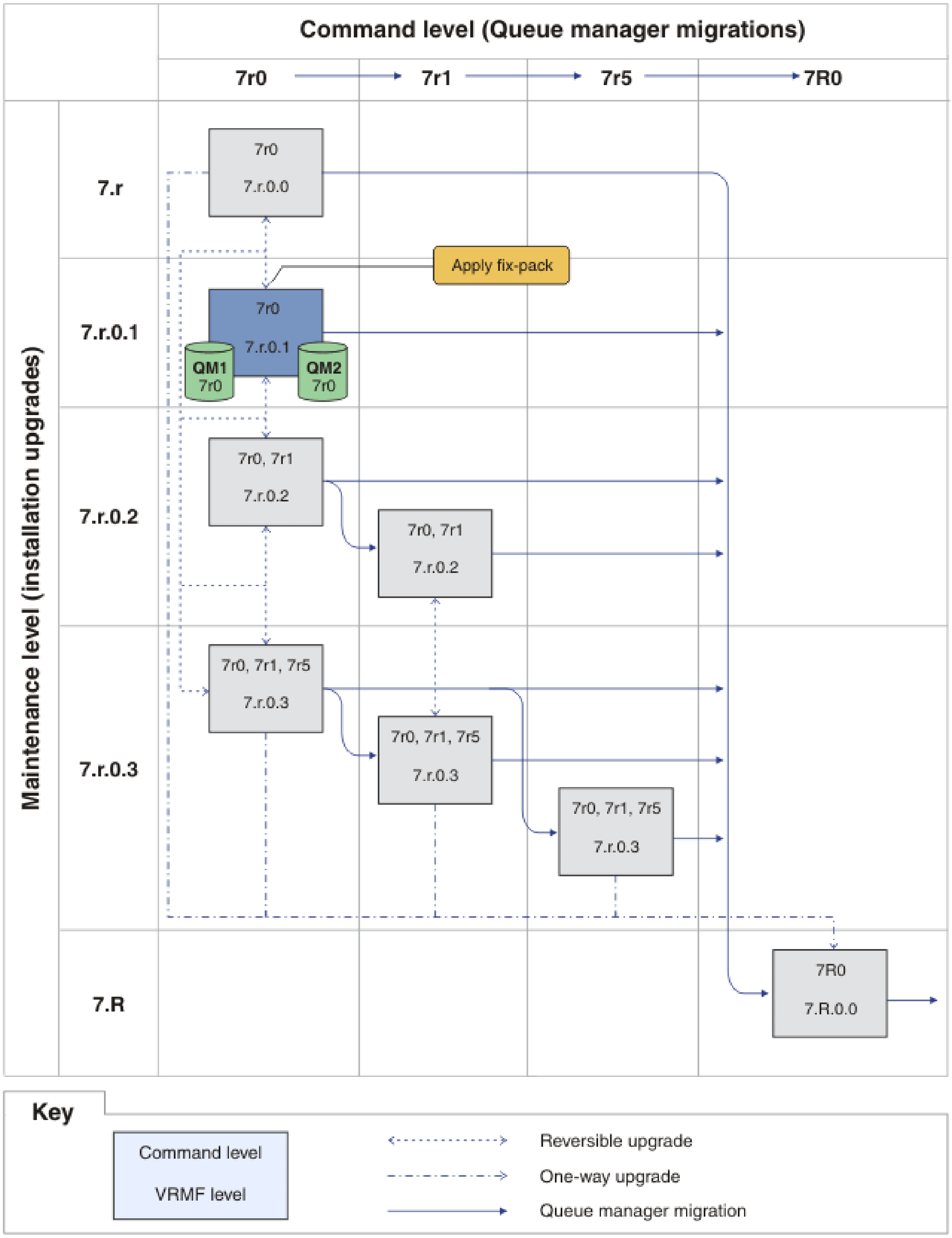


Figure 22. QM1 and QM2 at command level `7r0`, and fix level `7.r.0.1`

4. Apply fix pack `7.r.0.2`.

    a) Repeat steps “1” on page 254 and “2” on page 255 with fix pack `7.r.0.2`.

5. Restart `QM1`.

    `QM1` is now running using `Inst_1` at the `7.r.0.2` maintenance level, and the `7r0` command level. The queue manager is not automatically migrated to the `7r1` command level.

6. Migrate `QM2` to the `7r1` command level.

    ```
    strmqm -e CMDLEVEL=711 QM2
    ```

    `QM2` is using `Inst_1` at the `7.r.0.2` maintenance level, and has been migrated to the `7r1` command level.

7. Restart `QM2`.

    `QM2` is now running using `Inst_1` at the `7.r.0.2` maintenance level, and the `7r1` command level; see Figure 23 on page 258.

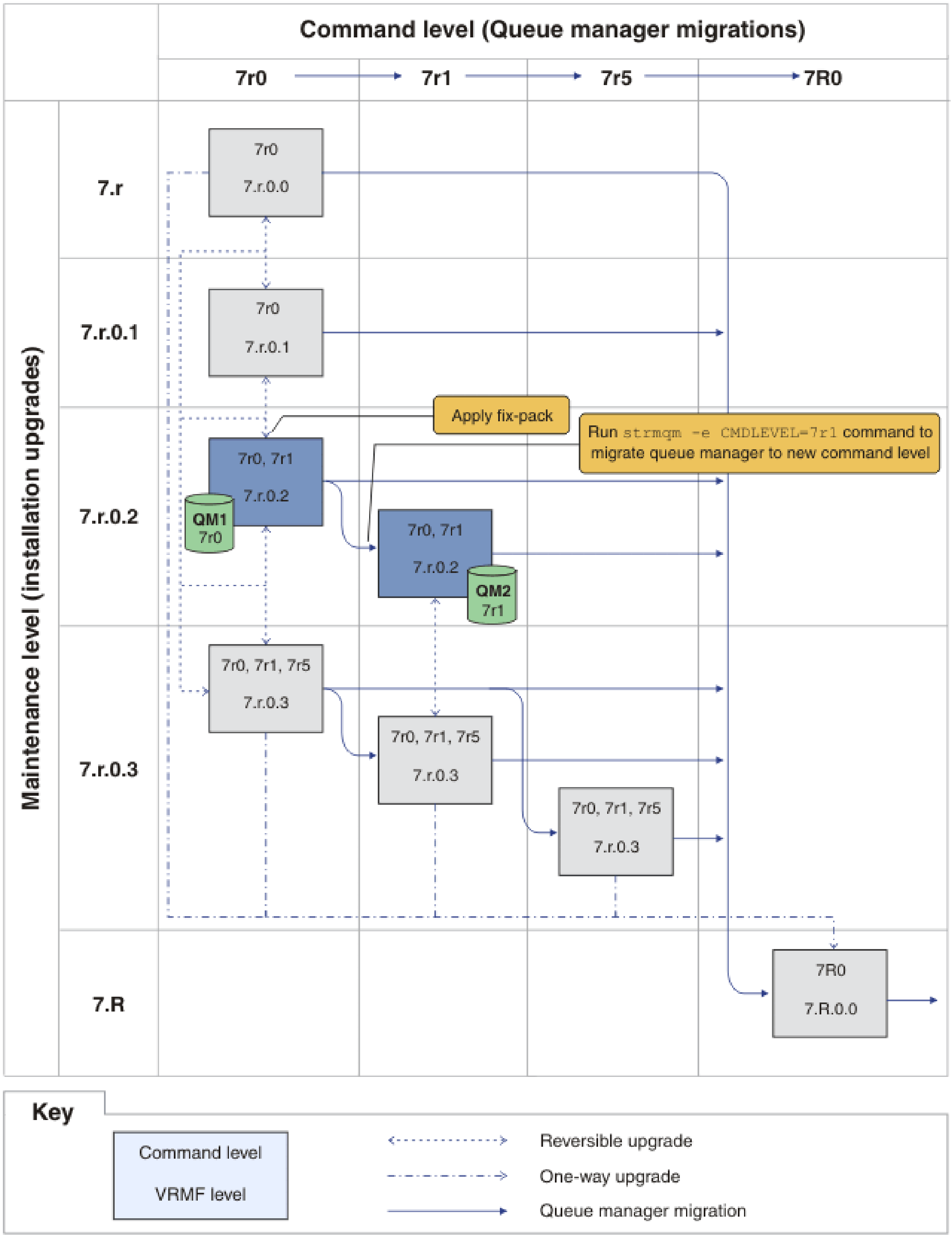


Figure 23. QM1 at command level 7r0 and fix level 7.r.0.2 ; QM2 at command level 7r1 and fix level 7.r.0.2

8. Apply fix pack `7.r.0.3` and migrate QM2 to the `7r5` command level.
   a) Repeat steps "4" on page 257 to "5" on page 257 with fix pack `7.r.0.3`.

b) Repeat steps “6” on page 257 to “7” on page 257 with command level `7r5`.

   `QM1` is using `Inst_1` at the `7.r.0.3` maintenance level, and is running at the `7r0` command level. `QM2` is using `Inst_1` at the `7.r.0.3` maintenance level, and has been migrated to the `7r5` command level; see Figure 24 on page 260.

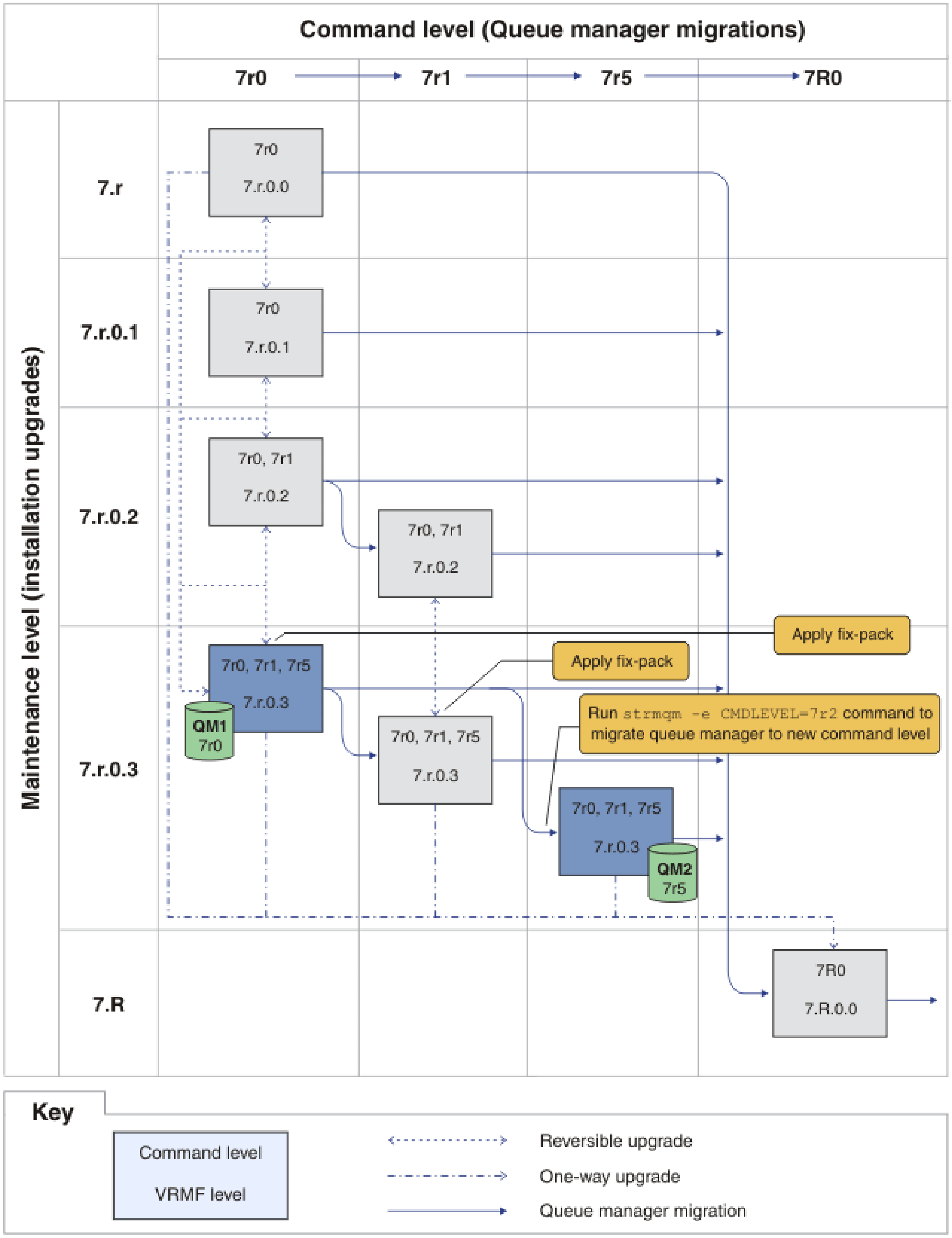


Figure 24. QM1 at command level `7r0` *and fix level* `7.r.0.3` *; QM2 at command level* `7r5` *and fix level* `7.r.0.3`

9. Migrate QM2 to version 7.R
   - On UNIX, Linux, and Windows:

a) Install version 7.R, with the installation name `Inst_2`, alongside Version 7.1.

b) Set up the local environment to the installation `Inst_2`.

Windows:

```
"Inst_2_INSTALLATION_PATH
\bin\setmqenv" -s
```

The `-s` option sets up the environment for the installation that runs the **setmqenv** command.

UNIX:

```
.
Inst_2_INSTALLATION_PATH/bin/setmqenv -s
```

c) Run the **setmqm** command to associate `QM2` with `Inst_2`.

```
 setmqm -m QM2 -n Inst_2
```

d) Run the **strmqm** command to start `QM2` and migrate it to version 7.R.

```
strmqm QM2
```

`QM1` is using `Inst_1` at the `7.r.0.3` maintenance level, and is running at the `7r0` command level.
`QM2` is using `Inst_2` at the `7.R.0.0` maintenance level, and has been migrated to the `7R0` command level; see Figure 24 on page 260.
`Inst_1` remains the primary installation.

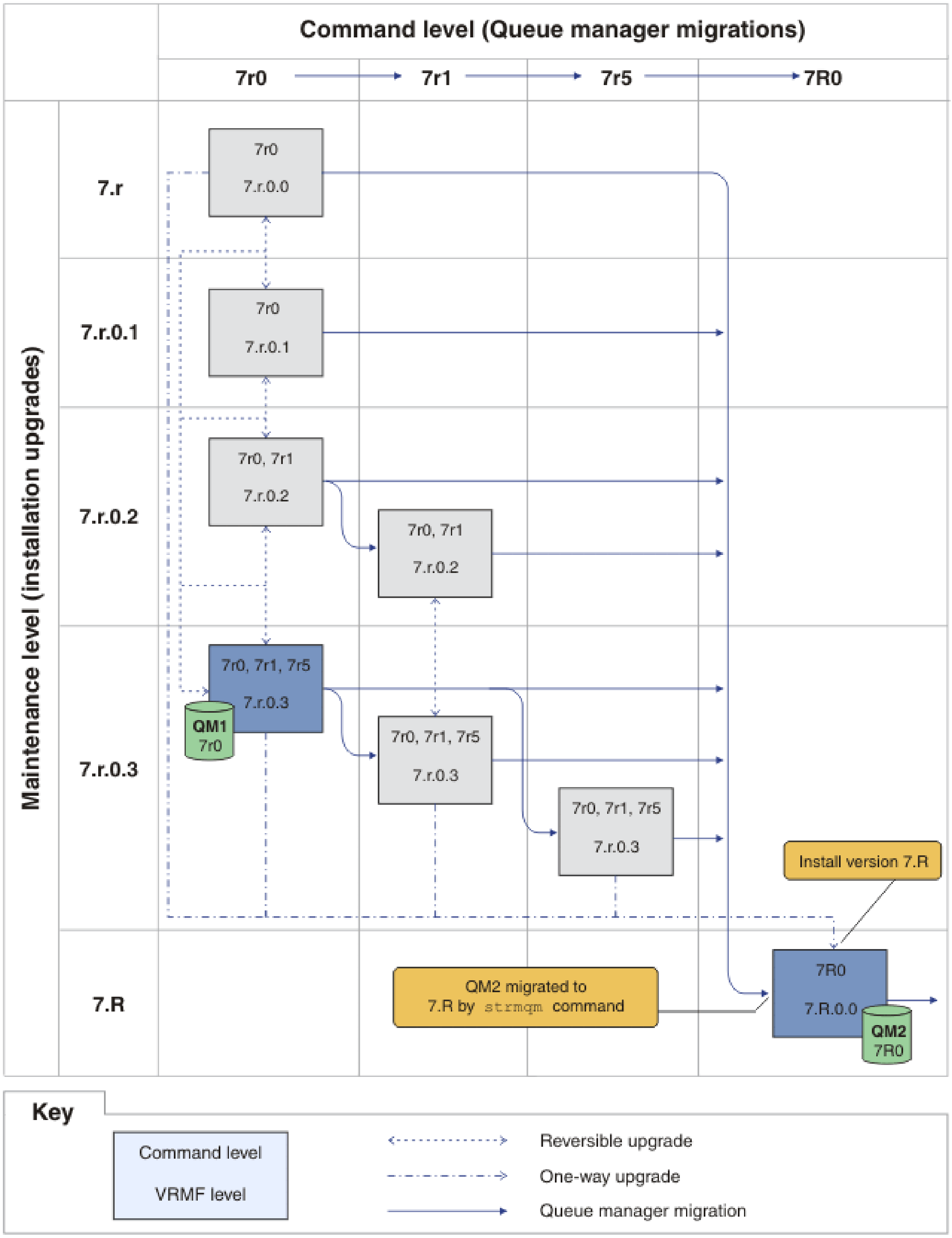


Figure 25. QM1 at command level `7r0` and fix level `7.r.0.3`; QM2 at command level `7R0` and fix level `7.R.0.0`

Related concepts

"Multi-installation queue manager coexistence on UNIX, Linux, and Windows" on page 57

You can install multiple copies of IBM MQ for UNIX, Linux, and Windows on the same server. The installations must be at Version 7.1 or later, with one exception. One Version 7.0.1 installation, at fix pack level 6, or later, can coexist with multiple Version 7.1, or later installations.

“New function in maintenance level upgrades (On platforms other than z/OS)” on page 48
IBM might introduce new functions between releases in maintenance level upgrades such as fix packs. A maintenance level upgrade including new function increases the maximum command level of an installation.

“Queue manager coexistence in Version 8.0” on page 53
Queue managers, with different names, can coexist on any server as long as they use the same IBM MQ installation. On z/OS z/OS, UNIX, Linux, and Windows, different queue managers can coexist on the same server and be associated with different installations.

“The version naming scheme for IBM MQ (On platforms other than z/OS)” on page 21
IBM MQ releases have a four-digit Version, Release, Modification, and Fix (VRMF) level code.

“The version naming scheme for IBM MQ for z/OS” on page 20
On IBM MQ for z/OS, releases have a three-digit Version, Release, and Maintenance (VRM) level code. The code is significant; it identifies the service life of a release. To run a queue manager at a different VRM level, you must migrate the queue manager, its applications, and the environment in which it runs. Depending on the migration path, the migration might require more or less effort.

“Upgrade, migration, and maintenance of IBM MQ (On platforms other than z/OS)” on page 51
You can install new releases of IBM MQ to upgrade IBM MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same UNIX, Linux, and Windows server. You can apply maintenance level upgrades to upgrade the maintenance or fix level. Applying maintenance level upgrades cannot change the version or release level of IBM MQ. Maintenance level upgrades can be reversed, installations cannot be reversed.

“Upgrade, migration, and maintenance of IBM MQ on z/OS” on page 49
You can install new releases of IBM MQ to upgrade IBM MQ to a new maintenance, release, or version level. Multiple installations at the same or different levels can coexist on the same z/OS instance. Running a queue manager at a higher level requires migration. Maintenance differs from upgrading. To maintain a level of IBM MQ, you apply Program Temporary Fixes (PTFs) to the installed code.

“IBM MQ maintenance” on page 38
Maintenance is the application of a reversible fix. Any changes to queue manager data are compatible with the previous code level.

“IBM MQ migration” on page 45

“IBM MQ upgrades and fixes” on page 39
The term upgrade applies to changing the version `V`, release `R`, or modification `M` of a product. The term fix applies to a change in the `F` digit.

Related tasks

“Applying and removing maintenance level updates (On platforms other than z/OS)” on page 215
When you apply and remove maintenance level updates to IBM MQ, no migration is required. Maintenance level updates are applied either as a fix pack, or by manually applying an interim fix.

“UNIX, Linux, and Windows: Staging maintenance fixes” on page 264
Use multiple installations of IBM MQ on the same server to control the release of maintenance fixes.

Related reference

“z/OS: OPMODE” on page 272
The availability of new functions and backward migration for IBM MQ for z/OS is controlled by the **`OPMODE`** parameter in the **`CSQ6SYSP`** macro. To access V8.0 capabilities, change the value of **`OPMODE`** to `OPMODE=(NEWFUNC,800)`. To restrict the use of new capabilities, and retain the ability to revert the queue manager to its earlier level, leave **`OPMODE`** at its default setting, `OPMODE=(COMPAT,800)`.

Related information

CommandLevel (MQLONG)

Querying the maintenance level

Query the IBM MQ maintenance level by running the **dspmqver** command

About this task

After an update to the initial installation, the version indicates the maintenance level to which the product has been updated. For example, before applying any maintenance, the version is 7.0.1.0. As maintenance is applied the last digit is updated, for example to 7.0.1.3.

Procedure

To view the version use the `dspmqver` command. At a command prompt, enter the following command: `dspmqver`.

The resulting messages include the IBM MQ version number, which shows the maintenance level.

Related information

dspmqver

UNIX, Linux, and Windows: Staging maintenance fixes

Use multiple installations of IBM MQ on the same server to control the release of maintenance fixes.

Before you begin

Set up your configuration modelled on the first row of Figure 26 on page 265. You can apply this scenario to any version of IBM MQ from IBM WebSphere MQ 7.1 onwards. In this scenario it is assumed you have a number of applications and two queue managers, `QM1` and `QM2`, running on a server. IBM WebSphere MQ 7.0.1 is not installed on the server.

1. Install two copies of IBM MQ. In the example, they are named `Inst_1` and `Inst_2` and IBM WebSphere MQ 7.1 is being used.
2. Make `Inst_1` primary by running **setmqinst**.
3. Associate all the queue managers on the server with `Inst_1` by running **setmqm**.
4. Start all the queue managers on the server.
5. Show and connect all direct connections with the queue managers associated with `Inst_1` in MQ Explorer.
6. Set up remote connections to all the queue managers in each instance of MQ Explorer.

About this task

You can install multiple copies of IBM MQ on a server to stage the release of IBM MQ fixes. Figure 26 on page 265 illustrates a way of using two installations to roll out fixes. In this approach, you maintain two fix levels on a server, with the aim of getting all queue managers and applications to the production fix level before replacing the previous level on fix pack with the next level.

Which installation an application uses is driven by the queue manager an application connects to. The **setmqm** command associates a queue manager with an installation. You can associate a queue manager with a different installation as long as the installation is at the same or higher command level. In this example, all the installations are at the same command level. You can associate or reassociate a queue manager with either of the installations running any of the fix packs.

In the example, an application links to the primary installation. When it connects to a queue manager, IBM MQ switches the linkage to the installation associated with the queue manager; see "Multi-installation queue manager coexistence on UNIX, Linux, and Windows" on page 57.

For applications built with the link options described in the product documentation, the simplest way to configure the link library search path for IBM MQ applications is to make an installation primary. Only if it is important to pick up a fix in the IBM MQ link library itself, must you review the search path. Either you must make the installation with the IBM MQ link library fix primary, or make a local adjustment for the application, perhaps by running the **setmqenv** command.

Running commands is a different matter. Commands are always run from the primary installation, or the installation you have selected by running the **setmqenv** command. If you run a command from the wrong installation, the command fails. For example, if `QM1` is associated with `Inst_1`, running the Windows command, `Inst_2_Installation_path/bin/strmqm QM1` fails.

If you are using MQ Explorer and you have two installations, you also have two MQ Explorer instances. One linked to one installation, and one to the other. Each MQ Explorer shows locally connected queue managers that are associated with the same installation as the instance of MQ Explorer. To monitor all the queue managers on a server, set up remote connections to the queue managers associated with the other installations.

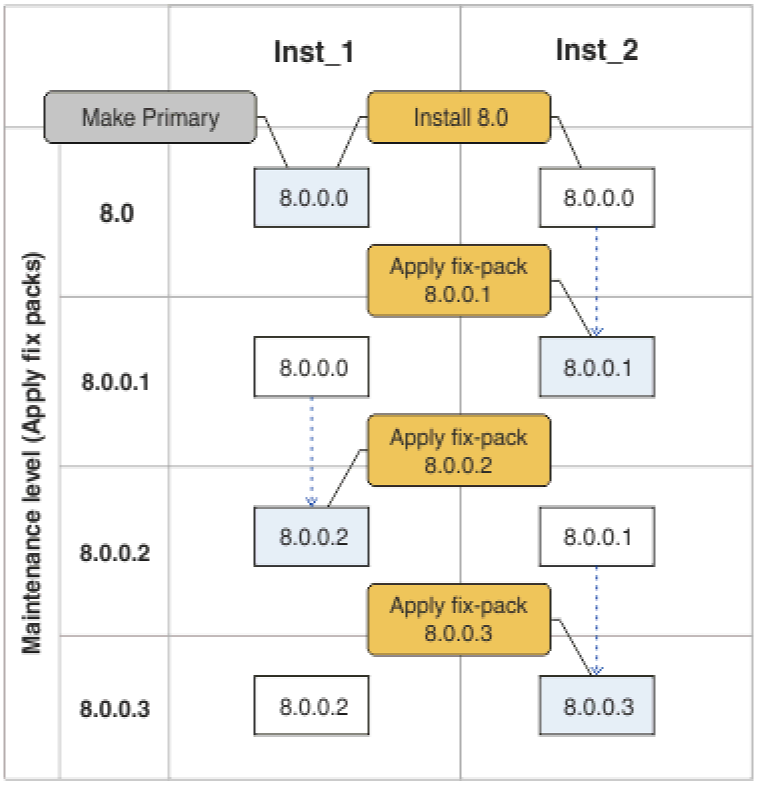


Figure 26. Rolling fix releases

Procedure

1. Download the first fix pack, for example, 7.1.0.1, when it is released.

   See Fix Central.

2. Apply the fix pack you downloaded to `Inst_2`.

   Follow the task for your platform in "Applying and removing maintenance level updates (On platforms other than z/OS)" on page 215.

3. Verify `Inst_2`.
4. Transfer the queue managers to `Inst_2` one at a time.

   a) Stop `QM1` and the applications connected to it.

   The **endmqm** command informs an application that the queue manager it is connected to is stopping; see Stopping a queue manager.

Note: The topic, "Applying maintenance level updates to multi-instance queue managers" on page 249, describes how to apply maintenance to a multi-instance queue manager. A multi-instance queue manager can continue to run on one server, while maintenance is applied to another server.

b) Set up the local environment to the installation `Inst_2`.

Windows Windows:

```
"Inst_2_INSTALLATION_PATH\bin\setmqenv" -s
```

The `-s` option sets up the environment for the installation that runs the **`setmqenv`** command.

UNIX UNIX:

```
.Inst_2_INSTALLATION_PATH/bin/setmqenv -s
```

c) Associate the queue manager with `Inst_2`.

```
setmqm -m QM1 -n Inst_2
```

d) Start `QM1`

```
strmqm QM1
```

e) Repeat substeps c and d for `QM2`.

f) Set up MQ Explorer for `Inst_2`.

i) Start the `Inst_2` instance of MQ Explorer

Tip: On Windows, hover over the IBM MQ icon in the system tray. The hover help shows the installation name associated with the MQ Explorer instance.

ii) Click **IBM MQ** > **Queue Managers** > **Show/Hide Queue Managers...** >

iii) Click each directly connected queue manager listed in the **Hidden Queue Managers** list > **Show**.

iv) Click **Close**.

5. Set `Inst_2` primary.

Windows Windows:

```
"Inst_2_INSTALLATION_PATH\bin\setmqinst" -i -n Inst_2
```

UNIX UNIX:

```
Inst_2_INSTALLATION_PATH/bin/setmqinst -i -n Inst_2
```

6. Download the next fix pack for the version of your product, for example, 7.1.0.2, when it is released.

   See Fix Central.

7. Apply the fix pack that you have just downloaded to `Inst_1`.

   Follow the task for your platform in "Applying and removing maintenance level updates (On platforms other than z/OS)" on page 215.

8. Verify `Inst_1`.

9. Transfer queue managers to `Inst_1` one at a time.

   a) Follow the procedure in step "4" on page 265

   Replacing `Inst_2` by `Inst_1` in the instructions.

10. Set Inst_1 primary.

    Windows Windows:

    ```
    "Inst_1_INSTALLATION_PATH\bin\setmqinst" -i -n Inst_1
    ```

    UNIX UNIX:

    ```
    Inst_1_INSTALLATION_PATH/bin/setmqinst -i -n Inst_1
    ```

11. Repeat steps “1” on page 265 to “5” on page 266 for the odd-numbered fix packs of your product.
12. Repeat steps “6” on page 266 to “10” on page 267 for the even-numbered fix packs of your product.

Related concepts

“Queue manager coexistence in Version 8.0” on page 53
Queue managers, with different names, can coexist on any server as long as they use the same IBM MQ installation. On z/OS z/OS, UNIX, Linux, and Windows, different queue managers can coexist on the same server and be associated with different installations.

“Multi-installation queue manager coexistence on UNIX, Linux, and Windows” on page 57
You can install multiple copies of IBM MQ for UNIX, Linux, and Windows on the same server. The installations must be at Version 7.1 or later, with one exception. One Version 7.0.1 installation, at fix pack level 6, or later, can coexist with multiple Version 7.1, or later installations.

Related tasks

“Migrating IBM MQ library loading from an earlier version of the product to the latest version” on page 131
No change in the way IBM MQ libraries are loaded is normally required if you upgrade from an earlier version of the product to the latest version. You must have followed the instructions on building IBM MQ applications in Version 7.0.1 and you must replace IBM WebSphere MQ 7.0.1 with the latest version of the product. If you choose to take advantage of multi-installation in the latest version of the product, based on the side-by-side or multi-stage migration scenarios, you must modify the environment for the operating system to resolve IBM MQ dependencies for an application. Typically, you can modify the runtime environment, rather than relink the application.

Related information

Installing IBM MQ server on Windows

Associating a queue manager with an installation

Changing the primary installation

setmqenv

setmqinst

setmqm

Migration commands, utilities, and reference information

A selection of commands, utilities, and application reference information related to migration are collected together in the following subtopics.

JMS PROVIDERVERSION property

The JMS **PROVIDERVERSION** property selects whether a Java application publishes and subscribes using the queued command message interface, or the integrated call interface.
Related information

Configuring the JMS **PROVIDERVERSION** property

Rules for selecting the IBM MQ messaging provider mode

The IBM MQ messaging provider has three modes of operation: normal mode, normal mode with restrictions, and migration mode. You can select which of these modes of operation a JMS application uses to publish and subscribe by setting the **PROVIDERVERSION** property for the connection factory to the appropriate value. In some cases, the **PROVIDERVERSION** property is set as unspecified, in which case an algorithm is used to determine which mode of operation to use.

If you cannot change the connection factory that you are using, you can use the `com.ibm.msg.client.wmq.overrideProviderVersion` property to override any setting on the connection factory. This override applies to all connection factories in the JVM but the actual connection factory objects are not modified.

You can set the **PROVIDERVERSION** property to any of the values 8 (normal mode), 7 (normal mode with restrictions), 6 (migration mode), or `unspecified` (the default value). The value that you specify for the **PROVIDERVERSION** property must be a string. If you are specifying an option of 8, 7 or 6, you can do this in any of the following formats:

- V.R.M.F
- V.R.M
- V.R
- V

where V, R, M and F are integer values greater than or equal to zero. The extra R, M and F values are optional and are available for you to use in case fine grained control is needed. For example, if you wanted to use a **PROVIDERVERSION** level of 7, you could set **PROVIDERVERSION**=7, `7.0`, `7.0.0` or `7.0.0.0`.

8 - Normal mode

The JMS application uses the IBM MQ messaging provider normal mode. Normal mode uses all the features of a IBM MQ queue manager to implement JMS. This mode is optimized to use the JMS 2.0 API and functionality.

If you are connecting to a queue manager with a command level of 800, then all of the JMS 2.0 API and features, such as asynchronous send, delayed delivery, or shared subscription, can be used.

If the queue manager specified in the connection factory settings is not a Version 8.0.0 queue manager, the `createConnection` method fails with an exception `JMSFMQ0003`.

The IBM MQ messaging provider normal mode uses the sharing conversations feature and the number of conversations that can be shared is controlled by the **SHARECNV()** property on the server connection channel. If this property is set to `0`, you cannot use IBM MQ messaging provider normal mode and the `createConnection` method fails with an exception `JMSCC5007`.

7 - Normal mode with restrictions

The JMS application uses the IBM MQ messaging provider normal mode with restrictions. This mode uses the JMS 2.0 API, but not the new features such as shared subscriptions, delayed delivery, or asynchronous send.

If you set **PROVIDERVERSION** to 7 only the IBM MQ messaging provider normal with restrictions mode of operation is available. If the queue manager specified in the connection factory settings is not a Version 7.0.1, or later, queue manager, the `createConnection` method fails with exception `JMSFCC5008`.

If you are connecting using normal mode with restrictions, to a queue manager with a command level between 700 and 800 then you can use the JMS 2.0 API, but not the asynchronous send, delayed delivery, or shared subscription features.

The IBM MQ messaging provider normal mode with restrictions uses the sharing conversations feature and the number of conversations that can be shared is controlled by the **SHARECNV()** property on the server connection channel. If this property is set to `0`, you cannot use IBM MQ messaging provider normal mode with restrictions and the `createConnection` method fails with an exception `JMSCC5007`.

6 - Migration mode

The JMS application uses the IBM MQ messaging provider migration mode.

The IBM MQ classes for JMS use the features and algorithms supplied with IBM WebSphere MQ Version 6.0. If you want to connect to WebSphere Message Broker Version 6.0 or 6.1 using IBM WebSphere MQ Enterprise Transport Version 6.0, you must use this mode. You can connect to a IBM MQ 8.0 queue manager using this mode, but none of the new features of a IBM MQ classes for JMS queue manager are used, for example, read ahead or streaming.

If you have a IBM MQ 8.0 client connecting to a IBM MQ 8.0 queue manager, then the message selection is done by the queue manager rather than on the client system.

If IBM MQ messaging provider migration mode is specified and you attempt to use any of the JMS 2.0 API, the API method call fails with the exception `JMSCC5007`.

`unspecified` (default)

The **PROVIDERVERSION** property is set to *unspecified* by default.

A connection factory that was created with a previous version of IBM MQ classes for JMS in JNDI takes this value when the connection factory is used with the new version of IBM MQ classes for JMS. The following algorithm is used to determine which mode of operation is used. This algorithm is used when the `createConnection` method is called and uses other aspects of the connection factory to determine if IBM MQ messaging provider normal mode, normal mode with restrictions, or IBM MQ messaging provider migration mode is required.

1. First, an attempt to use IBM MQ messaging provider normal mode is made.
2. If the queue manager connected is not IBM MQ 8.0, or later, an attempt to use IBM MQ messaging provider normal mode with restrictions is made.
3. If the queue manager connected is not IBM WebSphere MQ Version 7.0.1, or later, the connection is closed and IBM MQ messaging provider migration mode is used instead.
4. If the **SHARECNV** property on the server connection channel is set to 0, the connection is closed and IBM MQ messaging provider migration mode is used instead.
5. If **BROKERVER** is set to V1 or the default *unspecified* value, IBM MQ messaging provider normal mode continues to be used, and therefore any publish/subscribe operations use the new IBM WebSphere MQ Version 7.0.1, or later, features.

   See ALTER QMGR for information about the PSMODE parameter of the ALTER QMGR command for further information on compatibility.
6. If **BROKERVER** is set to V2 the action taken depends on the value of **BROKERQMGR** :
   - If the **BROKERQMGR** is `blank`:

     If the queue specified by the **BROKERCONQ** property can be opened for output (that is, MQOPEN for output succeeds) and **PSMODE** on the queue manager is set to COMPAT or DISABLED, then IBM MQ messaging provider migration mode is used.
   - If the queue specified by the **BROKERCONQ** property cannot be opened for output, or the **PSMODE** attribute is set to ENABLED:

     IBM MQ messaging provider normal mode is used.
   - If **BROKERQMGR** is `non-blank` :

     IBM MQ messaging provider migration mode is used.

Related concepts

“When to use PROVIDERVERSION” on page 270

In two cases you must override the default selection of **PROVIDERVERSION** for the IBM MQ classes for JMS to work correctly.

Related information

BROKERQMGR

BROKERCONQ

PSMODE

JMS provider version troubleshooting

When to use PROVIDERVERSION

In two cases you must override the default selection of **PROVIDERVERSION** for the IBM MQ classes for JMS to work correctly.

Note:

The migration mode that is described in this topic is migration from IBM WebSphere MQ 6.0 to IBM WebSphere MQ 7.0.

IBM WebSphere MQ 6.0, WebSphere Application Server Version 6.0.x, and WebSphere Message Broker Version 6 are out of support, and therefore this topic is included only for reference purposes.

There are two scenarios where you cannot use the algorithm described in "Rules for selecting the IBM MQ messaging provider mode" on page 268. Consider using **PROVIDERVERSION** in these scenarios.

1. If IBM Integration Bus is in compatibility mode, you must specify **PROVIDERVERSION** for it to work correctly.
2. If you are using WebSphere Application Server Version 6.0.1, WebSphere Application Server Version 6.0.2, or WebSphere Application Server Version 6.1, connection factories are defined using the WebSphere Application Server administrative console.

   In WebSphere Application Server the default value of the **BROKERVER** property on a connection factory is V2. The default **BROKERVER** property for connection factories created by using **JMSAdmin** or IBM MQ Explorer is V1. This property is now "unspecified" in IBM MQ.

If **BROKERVER** is set to V2 (either because it was created by WebSphere Application Server or the connection factory has been used for publish/subscribe before) and has an existing queue manager that has a **BROKERCONQ** defined (because it has been used for publish/subscribe messaging before), the IBM MQ messaging provider migration mode is used.

However, if you want the application to use peer-to-peer communication and the application is using an existing queue manager that has ever done publish/subscribe, and has a connection factory with **BROKERVER** set to 2 (if the connection factory was created in WebSphere Application Server this is the default), the IBM MQ messaging provider migration mode is used. Using IBM MQ messaging provider migration mode in this case is unnecessary; use IBM MQ messaging provider normal mode instead. You can use one of the following methods to work around this:

- Set **BROKERVER** to 1 or unspecified. This is dependent on your application.
- Set **PROVIDERVERSION** to 8, or 7 which are custom properties in WebSphere Application Server Version 6.1.

  Alternatively, use the client configuration property, or modify the queue manager connected so it does not have the **BROKERCONQ**, or make the queue unusable.

strmqbrk: Migrate the IBM WebSphere MQ 6.0 publish/subscribe broker to a later version

Migrate the persistent state of an IBM MQ publish/subscribe broker to a later version queue manager.

Purpose

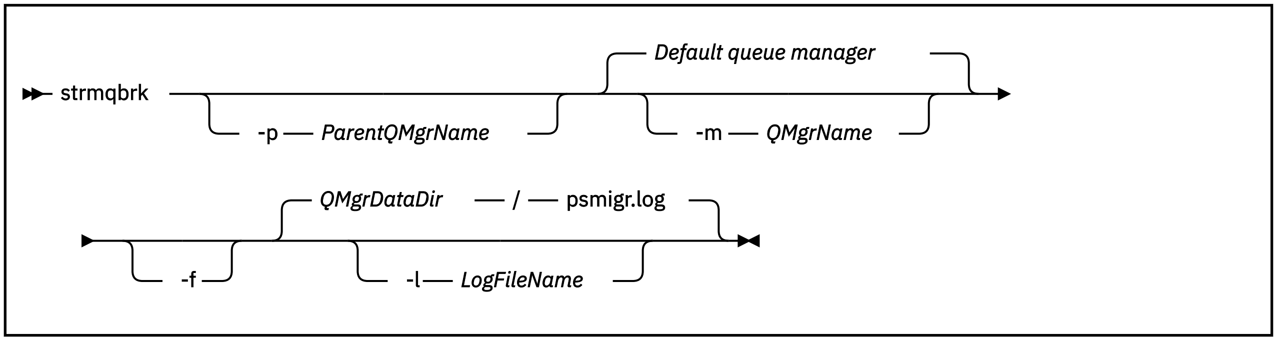
Use the `strmqbrk` command to migrate the state of a IBM WebSphere MQ 6.0 publish/subscribe broker to a later version queue manager. If the queue manager has already been migrated, no action is taken.

In IBM WebSphere MQ 6.0, `strmqbrk` started a broker. IBM MQ 8.0 publish/subscribe cannot be started in this manner. To enable publish/subscribe for a queue manager, use the **ALTER QMGR** command.

You can also use the **runmqbrk** command. This has the same parameters as **strmqbrk** and exactly the same effect.

Syntax

AIX, HP-UX, Linux, Solaris, and Windows



Optional parameters

AIX, HP-UX, Linux, Solaris, and Windows

-p *ParentQMgrName*

Note: This option is deprecated. **`strmqbrk`** migrates the parent connection automatically.

If you specify the current parent queue manager, a warning message is issued and migration continues. If you specify a different queue manager, a error is issued and migration is not performed.

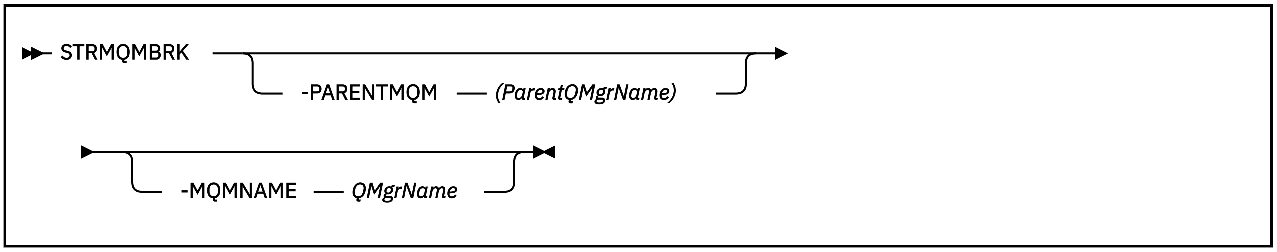
-m *QMgrName*
The name of the queue manager to be migrated. If you do not specify this parameter, the command is routed to the default queue manager.

-f
Force migration. This option specifies that objects created during the migration replace existing objects with the same name. If this option is not specified, if migration would create a duplicate object, a warning is issued, the object is not created, and migration continues.

-l *LogFileName*
Log migration activity to the file specified in *LogFileName*.

Syntax

IBM i



Optional parameters

IBM i

-PARENTMQM *(ParentQMgrName)*

Note: This option is deprecated.

If you specify the current parent queue manager, a warning message is issued and migration continues. If you specify a different queue manager, a warning is issued and migration is not performed.

-MQMNAME ***QMgrName***

The name of the queue manager to be migrated. If you do not specify this parameter, the command is routed to the default queue manager.

Related information

ALTER QMGR

z/OS: OPMODE

The availability of new functions and backward migration for IBM MQ for z/OS is controlled by the **OPMODE** parameter in the **CSQ6SYSP** macro. To access V8.0 capabilities, change the value of **OPMODE** to `OPMODE=(NEWFUNC,800)`. To restrict the use of new capabilities, and retain the ability to revert the queue manager to its earlier level, leave **OPMODE** at its default setting, `OPMODE=(COMPAT,800)`.

The default setting of **OPMODE** at V8.0 is `OPMODE=(COMPAT,800)`. The default settings for a queue manager, when initially run at V8.0, restrict it to a limited set of the new functions. The restriction makes it possible to revert a queue manager to its earlier release level, if you must do so.

If you query the value of **OPMODE** with the command **DIS** SYSTEM , the result is `(NEWFUNC, *VRM*)`, where *VRM* is a release level. *VRM* is the same as the queue manager command level. The value of *VRM* is the release level you can revert the queue manager back to. If the queue manager is newly created at V8.0, then `VRM=800`. If the queue manager was previously run on V7.1.0, then `VRM=710`.

Make all the new functions in the release available to a queue manager by setting **OPMODE** to `OPMODE=(NEWFUNC,800)`. If you change the setting of **OPMODE** for a queue manager to `OPMODE=(NEWFUNC,800)`, you can no longer revert the queue manager to run on an earlier release level.

Each queue manager in a queue sharing group (QSG) must have **OPMODE** set to `OPMODE=(NEWFUNC,800)` and be restarted in order for any queue manager in the QSG to utilize Version 8.0 function.

This means that there are effectively two phases to enabling Version 8.0 new function in a QSG:

1. The first restart with **OPMODE** set to `OPMODE=(NEWFUNC,800)` prevents the queue manager from being reverted to run on an earlier release level.
2. The second restart, when all other QSG members have already been restarted with **OPMODE** set to `OPMODE=(NEWFUNC,800)`, and therefore cannot revert to run on an earlier release level, allows Version 8.0 new function to be used.

For example, in a QSG containing three queue managers, there will be a total of five queue manager restarts required to enable Version 8.0 new function on all QSG members.

You can reset **OPMODE** to `OPMODE=(COMPAT,800)`, after setting `OPMODE=(NEWFUNC,800)`, to prevent new functions being used. If you do so, **DIS** SYSTEM shows `OPMODE=(COMPAT,800)` rather than `OPMODE=(COMPAT,710)`, indicating that you cannot revert the queue manager to V7.1.0.

The syntax of **OPMODE** is as follows:

OPMODE=(Mode , ***VerificationLevel*** **)**

OPMODE specifies the operation mode of the queue manager.

The default setting of OPMODE is `OPMODE=(COMPAT,800)` .

Mode

Specifies the requested operation mode. The values are as follows:

COMPAT

The queue manager runs in compatibility mode. Certain new functions are not available. The queue manager can be migrated back to an earlier release.

NEWFUNC

All new functions provided in this level of code are available. The queue manager cannot be migrated back to an earlier release.

VerificationLevel

VerificationLevel is a `Version.Release.Modification` (VRM) code, without punctuation; `800`, for example.

The value of *VerificationLevel* ensures that the **CSQ6SYSP** parameters are coded for use with the level of **CSQ6SYSP** macro being compiled. If *VerificationLevel* does not match the VRM level of SCSQMACS used for **CSQ6SYSP**, then a compile-time error is reported. The *VerificationLevel* is compiled into the parameter module.

At queue manager startup, if the *VerificationLevel* does not match the release level of the queue manager, then COMPAT mode is forced.

The intent of the *VerificationLevel* parameter is to avoid inadvertent and irreversible setting of OPMODE to NEWFUNC. The mistake might occur when migrating to a newer version of IBM MQ using **CSQ6SYSP** statements prepared for an older version of the queue manager. It might also occur using a **CSQ6SYSP** parameter module built with an older version of the SCSQMACS macros.

If you need assistance to revert to an earlier version of IBM MQ, contact your IBM support center.

Related information

z/OS: Switching from OPMODE=(NEWFUNC,800) to OPMODE=(COMPAT,800)
Using CSQ6SYSP
DISPLAY SYSTEM

z/OS: OPMODE restrictions by version

The availability of some new functions and backward migration for IBM MQ for z/OS is controlled by the **OPMODE** parameter in the **CSQ6SYSP** macro. The **OPMODE** parameter determines whether you can use selected new functions, before you commit to staying at a given release. The functions and capabilities that are restricted in the different versions of the product are listed here.

IBM WebSphere MQ 7.0.1

Table 40. Functions and capabilities in Version 7.0.1

Function	Reference
Log compression	Log compression
GROUP units of recovery	Unit of recovery disposition
Publish/subscribe migration using the CSQUMGMB utility	CSQUMGMB removed in IBM MQ 8.0

IBM WebSphere MQ 7.1

Table 41. Functions and capabilities in Version 7.1

Function	Reference
CFLEVEL(5) which is a prerequisite for Shared Message Data Sets (SMDS) and custom offload rules	DEFINE CFSTRUCT Planning your coupling facility and offload storage environment Where are shared queue messages held?
Support for the X'3C' OTMA protocol messages, which report IMS health status	The IMS bridge
Resilience to coupling facility connectivity failures	Shared queue recovery

IBM MQ 8.0

Table 42. Functions and capabilities in Version 8.0

Function	Reference
Buffer pools can be located above the bar	DEFINE BUFFPOOL
Multiple cluster transmission queues	Clustering: Planning how to configure cluster transmission queues Working with cluster transmission queues and cluster-sender channels Planning how you use multiple cluster transmission queues
Increased maximum addressable log range	Larger log Relative Byte Address
Configurable certificates for individual channels using CERTLABL	Certificate label (CERTLABL)
Using host names, restricting the certificate issuer, and checking client credentials in CHLAUTH rules	SET CHLAUTH

Related information

Using CSQ6SYSP

z/OS: Switching from OPMODE=(NEWFUNC,800) to OPMODE=(COMPAT,800)

The availability of new functions and backward migration for IBM MQ for z/OS is controlled by the **OPMODE** parameter in the **CSQ6SYSP** macro. You should be aware of the implications of switching from `OPMODE=(NEWFUNC,800)` to `OPMODE=(COMPAT,800)`

Switching from OPMODE=(NEWFUNC,800) to OPMODE=(COMPAT,800)

When you switch from `OPMODE=(NEWFUNC,800)` to `OPMODE=(COMPAT,800)` the following conditions occur:

- If the BSDS has been converted to version 2, the queue manager will not be able to access the BSDS when it is started at OPMODE=(COMPAT,800). This means that the queue manager fails to start and terminates with reason code 00D10120.
  - You can find the BSDS version by running the print log map utility (CSQJU004).
- Any buffer pools with an ID greater than 15 are marked as suspended. This means that these buffer pools cannot be used, deleted, or altered until `OPMODE=(NEWFUNC,800)` is specified again. Information about the buffer pools is kept in checkpoint log records until `OPMODE=(NEWFUNC,800)` is specified again.
  - Any page set that uses a suspended buffer pool is also suspended. Information about the suspended page set is also kept in checkpoint records.
  - While a page set is suspended, any messages in the page set are unavailable. An attempt to use a queue or topic which uses the suspended page set results in an MQRC_PAGESET_ERROR message.
  - While it is suspended, a page set can be associated with a different buffer pool by using the FORMAT function of the utility program CSQUTIL, specifying TYPE(REPLACE). You can then issue a **DEFINE PSID** command to bring the page set back into use with a different buffer pool.
  - **Note:** All units of recovery that involved the suspended page set, except units that are indoubt, will have been backed out by the queue manager when the page set was last used. Indoubt units of recovery can be resolved when the page set is again in use by the queue manager.
- Any buffer pools with an ID of 15 or less that have their LOCATION attribute set to ABOVE, have the LOCATION attribute switched to BELOW and their PAGECLAS attribute set to 4KB and the buffer pool size is set to 1000 pages.

- Any cluster-sender channels that have been configured to use a transmission queue other than SYSTEM.CLUSTER.TRANSMIT.QUEUE fail to start with message CSQX295E. To allow these channels to start, you need to perform the following actions:
  - Change the default cluster transmission queue configuration of the queue manager, so that all cluster-sender channels default to use the transmission queue SYSTEM.CLUSTER.TRANSMIT.QUEUE. You can do this by changing the value of the DEFCLXQ queue manager attribute to SCTQ.
  - Identify any manually defined transmission queues that have a non blank cluster channel name attribute value, by using the DISPLAY QLOCAL(*) WHERE(CLCHNAME NE ' ') command. Change the cluster channel-name attribute value of these queues to blank.
- No inbound channels will be allowed to start if any channel authentication records have been created with a host name specified in their **ADDRESS** attribute. Message CSQY344E is issued for each channel authentication rule that uses restricted function, if this condition occurs.
- Defining channel authentication (**CHLAUTH**) with the *CHKCLNT* attribute, requires the queue manager to be running in NEWFUNC mode.

If you need assistance to revert to an earlier version of IBM MQ, contact your IBM support center.

Related information

Using CSQ6SYSP

DISPLAY SYSTEM

PROPCTL channel options

Use **PROPCTL** channel attribute to control which message properties are included in a message that is sent from a Version 8.0 queue manager to a partner queue manager from an earlier version of IBM MQ.

Table 43. Channel message property attribute settings

PROPCTL	Description
ALL	Use this option if applications connected to the partner queue manager from an earlier version are able to process any properties placed in a message by a Version 8.0 application. All properties are sent to the partner queue manager, in addition to any name/value pairs placed in the MQRFH2. You must consider two application design issues: 1. An application connected to the partner queue manager must be able to process messages containing MQRFH2 headers generated on a Version 8.0 queue manager. 2. The application connected to the partner queue manager must process new message properties that are flagged with MQPD_SUPPORT_REQUIRED correctly. With the ALL channel option set, JMS applications can interoperate between IBM MQ Version 8.0 and an earlier version using the channel. New Version 8.0 applications using message properties can interoperate with applications from an earlier version, depending on how the earlier version application handles MQRFH2 headers.

Table 43. Channel message property attribute settings (continued)

PROPCTL	**Description**
COMPAT	Use this option to send message properties to applications connected to an earlier version partner queue manager in some cases, but not all. Message properties are only sent if two conditions are met: 1. No property must be marked as requiring message property processing. 2. At least one of the message properties must be in a "reserved" folder; see Note. With the COMPAT channel option set, JMS applications can interoperate between IBM MQ Version 8.0 and an earlier version using the channel. The channel is not available to every application using message properties, only to those applications that use the reserved folders. The rules concerning whether the message or the property is sent are: 1. If the message has properties, but none of the properties are associated with a "reserved" folder, then no message properties are sent. 2. If any message property has been created in a "reserved" property folder, all message properties associated with the message are sent. However: a. If any of the message properties are marked as support being required, `MQPD_SUPPORT_REQUIRED` or `MQPD_SUPPORT_REQUIRED_IF_LOCAL`, the whole message is rejected. It is returned, discarded, or sent to the dead letter queue according to the value of its report options. b. If no message properties are marked as support being required, an individual property might not be sent. If any of the message property descriptor fields are set to non-default values the individual property is not sent. The message is still sent. An example of a non-default property descriptor field value is `MQPD_USER_CONTEXT`. **Note:** The "reserved" folders names start with `mcd.`, `jms.`, `usr.`, or `mqext.`. These folders are created for applications that use the JMS interface. In Version 8.0 any name/value pairs that are placed in these folders are treated as message properties. Message properties are sent in an `MQRFH2` header, in addition to any name/value pairs placed in an `MQRFH2` header. Any name/value pairs placed in an `MQRFH2` header are sent as long as the message is not rejected.
NONE	Use this option to prevent any message properties being sent to applications connected to an earlier version partner queue manager. An `MQRFH2` that contains name/value pairs and message properties is still sent, but only with the name/value pairs. With the `NONE` channel option set, a JMS message is sent as a `JMSTextMessage` or a `JMSBytesMessage` without any JMS message properties. If it is possible for an earlier version application to ignore all properties set in a Version 8.0 application, it can interoperate with it.

PROPCTL queue options

Use the **PROPCTL** queue attribute to control how message properties are returned to an application that calls MQGET without setting any MQGMO message property options.

Table 44. Queue message property attribute settings

PROPCTL	Description
ALL	Use this option so that different applications reading a message from the same queue can process the message in different ways. • An application, migrated unchanged from an earlier version, can continue to read the MQRFH2 directly. Properties are directly accessible in the MQRFH2 header. You must modify the application to handle any new properties, and new property attributes. It is possible that the application might be affected by changes in the layout and number of MQRFH2 headers. Some folder attributes might be removed, or that IBM MQ reports an error in the layout of the MQRFH2 header that it ignored in an earlier version. • A new or changed application can use the message property MQI to query message properties, and read name/value pairs in MQRFH2 header directly. All the properties in the message are returned to the application. • If the application calls MQCRTMH to create a message handle, it must query the message properties using MQINQMP. Name/value pairs that are not message properties remain in the MQRFH2, which is stripped of any message properties. • If the application does not create a message handle, all the message properties and name/value pairs remain in the MQRFH2. ALL only has this affect if the receiving application has not set a MQGMO_PROPERTIES option, or has set it to MQGMO_PROPERTIES_AS_Q_DEF.

Table 44. Queue message property attribute settings (continued)

PROPCTL	Description
COMPAT	COMPAT is the default option. If GMO_PROPERTIES_* is not set, as in an unmodified application from an earlier version, COMPAT is assumed. By defaulting to the COMPAT option, an earlier version application that did not explicitly create an MQRFH2, works without change on Version 8.0. Use this option if you have written an earlier version application MQI application to read JMS messages. • The JMS properties, which are stored in an MQRFH2 header, are returned to the application in an MQRFH2 header in folders with names starting with mcd., jms., usr., or mqext.. • If the message has JMS folders, and if a Version 8.0 application adds new property folders to the message, these properties are also returned in the MQRFH2. Consequently, you must modify the application to handle any new properties, and new property attributes. It is possible that an unmodified application might be affected by changes in the layout and number of MQRFH2 headers. It might find some folder attributes are removed, or that IBM MQ finds errors in the layout of the MQRFH2 header that it ignored in an earlier version. **Note:** In this scenario, the behavior of the application is the same whether it is connected to an earlier version or Version 8.0 queue manager. If the channel **PROPCTL** attribute is set to COMPAT or ALL any new message properties are sent in the message to the earlier version partner queue manager. • If the message is not a JMS message, but contains other properties, those properties are not returned to the application in an MQRFH2 header. [14] • The option also enables earlier version applications that explicitly create an MQRFH2 to work correctly, in many cases. For example, An MQI program that creates an MQRFH2 containing JMS message properties continues to work correctly. If a message is created without JMS message properties, but with some other MQRFH2 folders, the folders are returned to the application. Only if the folders are message property folders are those specific folders are removed from the MQRFH2. Message property folders are identified by having the new folder attribute content='properties', or are folders with names listed in Defined property folder name or Ungrouped property folder name. • If the application calls MQCRTMH to create a message handle, it must query the message properties using MQINQMP. Message properties are removed from the MQRFH2 headers. Name/value pairs that are not message properties remain in the MQRFH2. • If the application calls MQCRTMH to create a message handle, it can query all message properties, regardless of whether the message has JMS folders. • If the application does not create a message handle, all the message properties and name/value pairs remain in the MQRFH2. If a message contains new user property folders, you can infer that the message was created by a new or changed Version 8.0 application. If the receiving application is to process these new properties directly in an MQRFH2, you must modify the application to use the ALL option. With the default COMPAT option set, an unmodified application continues to process the rest of the MQRFH2, without the Version 8.0 properties. The intent of the PROPCTL interface is to support old applications reading MQRFH2 folders, and new and changed applications using the message property interface. Aim for new applications to use the message property interface for all user message properties, and to avoid reading and writing MQRFH2 headers directly. COMPAT only has this affect if the receiving application has not set a MQGMO_PROPERTIES option, or has set it to MQGMO_PROPERTIES_AS_Q_DEF.

Table 44. Queue message property attribute settings (continued)

PROPCTL	Description
FORCE	The FORCE option places all messages properties into MQRFH2 headers. All message properties and name/value pairs in the MQRFH2 headers remain in the message. Message properties are not removed from the MQRFH2, and made available through a message handle. The effect of choosing the FORCE option is to enable a newly migrated application to read message properties from MQRFH2 headers. Suppose you have modified an application to process Version 8.0 message properties, but have also retained its ability to work directly with MQRFH2 headers, as before. You can decide when to switch the application over to using message properties by initially setting the PROPCTL queue attribute to FORCE. Set the **PROPCTL** queue attribute to another value when you are ready to start using message properties. If the new function in the application does not behave as you expected, set the **PROPCTL** option back to FORCE. FORCE only has this affect if the receiving application has not set a MQGMO_PROPERTIES option, or has set it to MQGMO_PROPERTIES_AS_Q_DEF.
NONE	Use this option so that an existing application can process a message, ignoring all message properties, and a new or changed application can query message properties. • If the application calls MQCRTMH to create a message handle, it must query the message properties using MQINQMP. Name/value pairs that are not message properties remain in the MQRFH2, which is stripped of any message properties. • If the application does not create a message handle, all the message properties are removed from the MQRFH2. Name/value pairs in the MQRFH2 headers remain in the message. NONE only has this affect if the receiving application has not set a MQGMO_PROPERTIES option, or has set it to MQGMO_PROPERTIES_AS_Q_DEF.
V6COMPAT	Use this option to receive an MQRFH2 in the same format as it was sent. If the sending application, or the queue manager, creates additional message properties, these are returned in the message handle. This option has to be set on both the sending and receiving queues, and any intervening transmission queues. It overrides any other PROPCTL options set on queue definitions in the queue name resolution path. Use this option only in exceptional circumstances. For example, if you are migrating applications from an earlier version to Version 8.0 the option is valuable because it preserves the behavior of the earlier version. The option is likely to have an impact on message throughput. It is also more difficult to administer; you need to ensure the option is set on the sender, receiver, and intervening transmission queues. V6COMPAT only has this affect if the receiving application has not set a MQGMO_PROPERTIES option, or has set it to MQGMO_PROPERTIES_AS_Q_DEF.

Related information

PROPCTL

[14] The existence of specific property folders created by the IBM MQ classes for JMS indicates a JMS message. The property folders are mcd., jms., usr., or mqext.

MQGMO message property option settings

Use MQGMO message property options to control how message properties are returned to an application.

Table 45. MQGMO message property option settings

MQGMO Option	**Description**
MQGMO_PROPERTIES_AS_Q_DEF	Applications prior to Version 7.0, and Version 8.0 applications that read from the same queue, and do not set GMO_PROPERTIES_*, receive the message properties differently. Applications prior to Version 7.0, and Version 8.0 applications that do not create a message handle, are controlled by the queue **PROPCTL** attribute. A Version 8.0 application can choose to receive message properties in the MQRFH2, or create a message handle and query the message properties. If the application creates a message handle, properties are removed from the MQRFH2. • An unmodified application prior to Version 7.0 does not set GMO_PROPERTIES_*. Any message properties it receives are in the MQRFH2 headers. • A new or changed Version 8.0 application that does not set GMO_PROPERTIES_* or sets it to MQGMO_PROPERTIES_AS_Q_DEF can choose to query message properties. It must set MQCRTMH to create a message handle and query message properties using the MQINQMP MQI call. • If a new or changed application does not create a message handle, it behaves like an application prior to Version 7.0. It must read any message properties it receives directly from the MQRFH2 headers. • If the queue attribute **PROPCTL** is set to FORCE, no properties are returned in the message handle. All properties are returned in MQRFH2 headers. • If the queue attribute **PROPCTL** is set to NONE, or COMPAT, a Version 8.0 application that creates a message handle, receives all message properties. • If the queue attribute **PROPCTL** is set to V6COMPAT, and also set to V6COMPAT on all the queues the message was placed on between the sender and the receiver, properties set by MQSETMP are returned in the message handle, and properties and name/value pairs created in an MQRFH2 are returned in the MQRFH2. The format of an MQRFH2 sent in Version 8.0 is the same as an application prior to Version 7.0, when sent by the same application.
MQGMO_PROPERTIES_IN_HANDLE	Force an application to use message properties. Use this option to detect if a modified application fails to create message handle. The application might be trying to read message properties directly from an MQRFH2, rather than call MQINQMP.

Table 45. MQGMO message property option settings (continued)

MQGMO Option	**Description**
MQGMO_NO_PROPERTIES	An application prior to Version 7.0 and a Version 8.0 application behave the same way, even if the Version 8.0 application creates a message handle. • All properties are removed. An unchanged application, prior to Version 7.0, connected to a Version 8.0 queue manager might behave differently to when it was connected to a partner queue manager from a version prior to Version 7.0. Queue manager generated properties, such as JMS properties, are removed. • Properties are removed even if a message handle is created. Name/value pairs in other MQRFH2 folders are available in the message data.
MQGMO_PROPERTIES_FORCE_MQRFH2	Applications prior to Version 7.0 and Version 8.0 applications behave the same way. Properties are returned in the MQRFH2 headers, even if a message handle is created. • MQINQMP returns no message properties, even if a message handle is created. MQRC_PROPERTY_NOT_AVAILABLE is returned if a property is inquired upon.
MQGMO_PROPERTIES_COMPATIBILITY	An application prior to Version 7.0 connected to a Version 8.0 queue manager behaves the same as when it is connected to a queue manager from a version prior to Version 7.0. If the message is from a JMS client, the JMS properties are returned in the MQRFH2 headers. New or modified Version 8.0 applications, that create a message handle, behave differently. • All properties in any message property folders are returned if the message contains a `mcd.`, `jms.`, `usr.`, or `mqext.` folder. • If the message contains property folders, but not a `mcd.`, `jms.`, `usr.`, or `mqext.` folder, no message properties are returned in an MQRFH2. • If a new or modified Version 8.0 application creates a message handle, query message properties using the MQINQMP MQI call. All message properties are removed from the MQRFH2. • If a new or modified Version 8.0 application creates a message handle, all properties in the message can be queried. Even if the message does not contain a `mcd.`, `jms.`, `usr.`, or `mqext.` folder, all message properties are queriable.

Related information

PROPCTL

2471 (09A7) (RC2471): MQRC_PROPERTY_NOT_AVAILABLE

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Software Interoperability Coordinator, Department 49XA
3605 Highway 52 N
Rochester, MN 55901
U.S.A.

© Copyright IBM Corp. 2007, 2022

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Programming interface information

Programming interface information, if provided, is intended to help you create application software for use with this program.

This book contains information on intended programming interfaces that allow the customer to write programs to obtain the services of WebSphere MQ.

However, this information may also contain diagnosis, modification, and tuning information. Diagnosis, modification and tuning information is provided to help you debug your application software.

Important: Do not use this diagnosis, modification, and tuning information as a programming interface because it is subject to change.

Trademarks

IBM, the IBM logo, ibm.com®, are trademarks of IBM Corporation, registered in many jurisdictions worldwide. A current list of IBM trademarks is available on the Web at "Copyright and trademark information"www.ibm.com/legal/copytrade.shtml. Other product and service names might be trademarks of IBM or other companies.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

This product includes software developed by the Eclipse Project (http://www.eclipse.org/).

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Part Number:

(1P) P/N: