MQ Series 9.0 for Linux
REQUIRES REDHAT 7 and ABOVE

1. You MUST get a license for Websphere MQ. MQ is licensed by number of CPUs. If this is for Wireline please place an order on VSHOP. Either way select that it is an upgrade and they will provide you with the download. Be sure to indicate the version you need.

VSHOP
https://vzsam.shi.com

The person creating the task plan takes responsibility for assuring that the installation is legally licensed before the task plan is executed.

2. MQ Series packages use 3 group IDs and 1 user ID. The 3 GIDs are mqm (root **MUST** be a part of this group), mqadmin and mquser. The 1 UID required is mqm. The standard mqm UID and GID is 20880, per INS security.
   - mqm GID – only mqm and root should be a part of this group.
   - mqadmin GID – typically used for application support personnel to allow them to look at queue levels and perform status checks.
   - mquser GID – typically used for application users.

   Note –Authority must be granted to the above groups (mqadmin and mquser), when queues are created, to allow the proper access. During queue manager creation, the configuration script sets the proper authority for all default system queues. See step 4 below for SSI file creation details.

3. File systems should be allocated space as shown below.
   2.0GB - $Anchor/opt/mqm
   *****2GB - $Anchor/opt/var/mqm **PER QMGR**.
   *****Note - More space may be necessary if using persistent messaging and/or large messages. See the MQ Series Beginnings Guide for more information. Online documentation can be found at:
      http://www-306.ibm.com/software/integration/wmq/library/

4. Make sure your systems are up to the IBM recommended system requirements. See APPENDIX A for more information.

5. SSI files will need to be created and are supplied by the application team. If a binary only install is desired (no configuration created at installation) in one of 3 ways: (Binaries only installations should only be performed on development servers) SSI files provides a disaster recovery mechanism.
   1) Create a SSI file **(host.{hostname}.mqm)** and include the following 2 variables:
      QMGRS=NONE
      RUNMQLSR=NONE
   2) Add the variable **MODE=PASSIVE** to the above SSI file (typically used for passive systems in an active/passive setup which does not create qmgrs)
   If a configuration needs to be created during installation, then the instructions in APPENDIX B should be followed. If a configuration change is desired after

the package has been installed, the **change_config.sh** script will need to be run once the SSI files have been updated with the desired changes. See APPENDIX C for command syntax information.

6. The package can now be installed. For the latest package name, contact MQ Middleware Engineering.
7. The Verizon standard for monitoring MQ Series is BMC Patrol with the latest MQ knowledge module installed. Contact the monitoring team for further information.
   LImq-km-4.8.00-linux_5.0.rpm
8. For a list of fixes by release number, go to the IBM Website.
9. See APPENDIX D for important release notes.

Appendix A

System Requirements

Operating System: Red Hat Linux 6 or higher

Required to run MQ Explorer:

In order to run MQ Explorer in Linux you must install GTK2 Version 2.2.4-0 or later. (May not be installed on Verizon systems by default) Users running the x86–64 platform must install the 32–bit version of GTK2 Version 2.2.4–0 or later to use the WebSphere MQ Explorer and WebSphere MQ File Transfer Application components. If using Red Hat Enterprise Linux (RHEL) V6 on the x86-64 platform, additional 32-bit libraries are required. These can be installed using the following command:

```
yum install libgtk-x11-2.0.so libcanberra-gtk-module.so libpk-gtk-
module.so
```

Hardware Requirements and Compatible software:

https://www.ibm.com/software/reports/compatibility/clarity-reports/report/html/softwareReqsForProduct?deliverableId=F3D4E380A66211E6B42450D999E6A1C4&osPlatforms=AIX%7CLinux%7CWindows%7Cz/OS&duComponentIds=S007

KERNEL PARAMETERS:

Kernel configuration

WebSphere MQ uses System V IPC resources, in particular shared memory and semaphores.

The minimum configuration for WebSphere MQ for these resources is as follows:

- `kernel.shmmni = 4096`
- `kernel.shmall = 2097152`
- `kernel.shmmax = 268435456`
- `kernel.sem = 500 256000 250 1024`
- `fs.file-max = 524288`

If you plan to run more than one queue manager of moderate size on the server, increase the file-max parameter, fs.file-max.
To view the kernel parameters for your system, enter the following commands:

- `cat /proc/sys/kernel/shmmni`
- `cat /proc/sys/kernel/shmall`
- `cat /proc/sys/kernel/shmmax`
- `cat /proc/sys/kernel/sem`
- `cat /proc/sys/fs/file-max`

Each of these commands returns the value of the corresponding kernel parameter. For example, `cat /proc/sys/kernel/shmmni` returns the value for *kernel.shmmni*. If any of the values is less than the minimum value, you need to increase it to at least the minimum value.

To add or alter these values, log on as a user with root authority. Open the file /etc/sysctl.conf with a text editor, then add or change the following entries to the values shown:

```
kernel.shmmni = 4096
kernel.shmall = 2097152
kernel.shmmax = 268435456
kernel.sem = 500 256000 250 1024
```

```
fs.file-max = 524288
```
Then save and close the file.
To load these sysctl values immediately, enter the following command:
```
sysctl -p
```

If you do not issue the `sysctl -p` command, the new values are loaded when the system is rebooted.

Maximum open files

If the system is heavily loaded, you might need to increase the maximum possible number of open files. If your distribution supports the proc file system you can query the current limit by issuing the following command:
```
cat /proc/sys/fs/file-max
```
.
To report on the current maximum, and in-use, number of file descriptors for your system, enter the following commands:

- `/sbin/sysctl fs.file-max`
- `/sbin/sysctl fs.file-nr`

If you are using a pluggable security module such as PAM (Pluggable Authentication Module), ensure that this module does not unduly restrict the number of open files for the `mqm` and root user. To report the maximum number of open file descriptors per process for the `mqm` and root user, login as the `mqm` user and enter the following values:
```
ulimit -n
```
For a standard WebSphere MQ queue manager, set the *nofile* value for the `mqm` user to 10240 or more. To set the maximum number of open file descriptors for processes running under the `mqm` user, add the following information to the /etc/security/limits.conf file:
```
mqm              hard    nofile          10240
mqm              soft    nofile          10240
root             hard    nofile          10240
root             soft    nofile          10240
```

Maximum processes

A running WebSphere MQ queue manager consists of a number of thread programs. Each connected application increases the number of threads running in the queue manager processes. It is normal for an operating system to limit the maximum number of processes that a user runs. The limit prevents operating system failures due to an individual user or subsystem creating too many processes. You must ensure that the maximum number of processes that the `mqm and root` user are allowed to run is sufficient. The number of processes must include the number of channels and applications that connect to the queue manager.

The following calculation is useful when determining the number of processes for the `mqm or root` user:
```
maximum processes = 2048 + maximum WebSphere MQ connections
```

```
 + maximum WebSphere MQ channels
```

You can use the `PAM_limits` security module to control the number of processes that users run. You can configure the maximum number of processes for the `mqm and root` user as follows:

```
mqm             hard    nproc           4096
mqm             soft    nproc           4096
root            hard    nproc           4096
root            soft    nproc           4096
```

For more details on how to configure the `PAM_limits` security module type, enter the following command:

```
man limits.conf
```

•

APPENDIX B

MQ Series SSI File Creation

Below are the SSI file names with the available variables.
***NOTE: Do not include the ( ) or spaces in the SSI file (ex: *QMGRS=Q.MNGR1,Q.MNGR2…*). Only mandatory variables need to be included. If default values on non-mandatory variables are sufficient, those variables can be left out of the SSI file entirely.**

---

host.{HOSTNAME}.mqm (mandatory SSI file)

QMGRS=(***MANDATORY VARIABLE.** List all qmgrs here (Ex: **QM1,QM2)** If binary only install is desired, enter **NONE**)
RUNMQLSR=(***MANDATORY VARIABLE.** List qmgrs and ports you require. (Ex: **QM1:1414,QM2:1415**) If no listeners are required, set this to **NONE**)
DEFAULT_QMGR=(If a default qmgr is desired, it must be set here and has to be listed in QMGRS=. Delete this entry if no default qmgr is required)
RC_SCRIPTS=(Set to **NO** if /etc/rc scripts are not required, otherwise delete this entry)
MODE=PASSIVE (Set this variable on a backup (passive) system so change_config.sh cannot be run.
IPCCBASEADDRESS=(only available on AIX installations)
----

host.{HOSTNAME}.{QMGR} - required for each QMGR listed above in **QMGRS=** (unless a binary only installation is desired). For default values, do not include the variable in the file.

***QM_OPT:MAXCHANNELS=**(default = 100)
***QM_OPT:MAXACTIVECHANNELS=**(default = same as MAXCHANNELS)
***QM_OPT:MQIBINDTYPE=**(default = STANDARD. Optional value = FASTPATH)
***QM_OPT:LogPrimaryFiles=**(default = 3. Option values = 2-62, ***Case is important**)
***QM_OPT:LogSecondaryFiles=**(default = 2. Optional values = 1-6, ***Case is important**)
***Note: LogPrimaryFiles** and **LogSecondaryFiles** total cannot exceed 63.
***QM_OPT:LOGFILEPAGES=**(Can only be set at QMGR creation. Default = 1024. Optional values = 64-16384)
***QM_OPT:LOGTYPE=**(Can only be set at QMGR creation. Default value = CIRCULAR. Optional value = LINEAR)
***QM_OPT:MQSNOAUT=**(Set to **YES** if you want the QMGR created with security disabled. Can only be set at QMGR creation)
***QM_OPT:KEEPALIVE=**(YES or NO. The default is **YES**)
***QM_OPT:LISTENERBACKLOG=**(The default is **100**)

***APIEXITLOCAL:APIEXITLOCAL:** (This line MUST precede each APIEXITLOCAL entry grouping (grouping = the 4 lines below))
***APIEXITLOCAL:**Sequence=200 (Example)
***APIEXITLOCAL:**Function=EntryPoint (Example)
***APIEXITLOCAL:**Module=/opt/mqm/samp/bin/amqsaxe (Example)
***APIEXITLOCAL:**Name=SampleApiExit (Example)

APPENDIX B (cont.)

***ENV_VAR:**(environment variable=setting)(more than1 can be used)
***SSLSTANZA OCSPAuthentication=OPTIONAL (REQUIRED FOR SHA2 CERT Communication)**

Then any commands used to configure the qmgr and create the queues and channels (any that normally get executed via the **runmqsc** command.

Last, any setmqaut commands used to set authority. The setmqaut command must be preceded by an * (ex: ***setmqaut** …). There should be no space between the * and setmqaut. Authority to access system queues is automatically granted for anyone in mquser (+allmqi) and mqadmin (+allmqi +dsp) groups. You should add all IDs that will need to access any queues created in the **host.{hostname}.{qmgr}** file to the mquser group, support personnel to the mqadmin group, and grant authority for these queues in your **host.{hostname}.{qmgr}** using the following 2 lines:

```
*setmqaut -m { QMGR_NAME } -t queue -n { QUEUE_NAME } -g mquser    +allmqi
*setmqaut -m {QMGR_NAME} -t queue -n {QUEUE_NAME} -g mqadmin   +allmqi +dsp
```

Substitute ***{QMGR_NAME}*** and ***{QUEUE_NAME}*** with the appropriate information.

All comments should be started with a * followed by a space.

APPENDIX C
Command Syntax

All control and configuration scripts are under the **$Anchor/opt/mqm/adm** directory.

All commands should be run as either root or mqm in production. On development servers, any ID in the group *mqm* can also execute the commands.

Command syntax for change_config.sh, MQcntrl.sh, menu and MQmonitor scripts is as follows:

change_config.sh {qmgr} {nobounce} - If no {qmgr} is specified, then all qmgrs on the server will be updated. The script will shutdown and restart any {qmgr} that is being updated unless the *nobounce* option is used. *Note: The *nobounce* option will stop any changes from bring made to the qm.ini file (manual edits) but will allow runmqsc commands to process..

MQcntrl.sh [start/stop] {qmgr} - If no {qmgr} is specified, then all will be affected.
	* **If no qmgr is specified in the shutdown (all qmgr shutdown), then shared memory and semaphores will be cleared of MQ entries.**

menu – Will give you access to the following MQ Utility Menu:

```
******************* MQSeries Utilities Menu ********************
**                                                            **
**  A) Start/Stop one or all qmgrs                            **
**  B) Update/create the configuration for one or all qmgrs   **
**  C) Quick status of one or all qmgrs                       **
**  D) Extensive status check of MQ (channels, queues, etc)   **
**  E) Run IBM Kernel Checkup Script                          **
**  F) Collect data for opening PMRs and IBM review           **
**  G) Clear mqm owned memory and semaphores (MQ must be down)**
**  H) Test MQ messaging using default Qs (Java based)        **
**  I) Test MQ messaging using default Qs (C based)           **
**  J) Check authority settings for a qmgr, queue, or channel **
**  K) Check the SSL configuration on a qmgr (Solaris, AIX)   **
**  L) Run SSL Configuration Wizard                           **
**  M) Save the current configuration for a qmgr              **
**  N) Check a qmgr SSI (config) file for syntax errors       **
**  O) Check queue depth                                      **
**  P) Check channel status                                        **
**  R) Run RUNMQSC COMMANDS                                   **
**  S) Save/View log files                                    **
**                                                            **
****************************************************************

        Enter your choice or Q to quit
```

MQmonitor {qmgr} – If no {qmgr} is specified, then all qmgrs will be checked. This script can be used by VSC (or any HA product) to check the health of a qmgr. If there is more than 1 qmgr on a system, and all are checked, the script will report a problem if 1 or more qmgrs have a problem. That means ALL qmgrs would be failed over to a backup server.

1. CHANNEL AUTHENTICATION

When you migrate a queue manager to version 7.5, channel authentication using channel authentication records is disabled. Channels continue to work as before. If you create a queue manager in version 7.5, 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 WebSphere® 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.
2. SET CHLAUTH('*') TYPE(BLOCKUSER) USERLIST('*MQADMIN') + DESCR('Default rule to disallow privileged users')

3. Except for the channel used by WebSphere MQ Explorer, all SYSTEM.* channels are blocked.
4. SET CHLAUTH('SYSTEM.*') TYPE(ADDRESSMAP) ADDRESS('*') USERSRC(NOACCESS) +
5. DESCR('Default rule to disable all SYSTEM channels')
6.
7. SET CHLAUTH(SYSTEM.ADMIN.SVRCONN) TYPE(ADDRESSMAP) ADDRESS('*') USERSRC(CHANNEL) + DESCR('Default rule to allow MQ Explorer access')

Note: This behaviour is default for all new WebSphere MQ version 7.5 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)
```

2. Change in behavior of the endmqm command

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.

3. CHANGES TO CLUSTER ERROR RECOVERY ON SERVERS

Before version 7.5, 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 7.5 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 7.5 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 7.5, 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 7.5 onwards, WebSphere® 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 WebSphere MQ, or by an administrator, or a program, doing something unexpected, such as ending a WebSphere 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 7.5.

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 a WebSphere 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 7.5.
   - 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 1.
   - 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 7.5, 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 7.5 causes the queue manager to stop, because it is regarded as a queue manager failure. You must not stop the amqrrmfa process in version 7.5, unless you set the queue manager tuning parameter, `TolerateRepositoryFailure`.

Example

Figure 1. Set `TolerateRepositoryFailure` to `TRUE` in qm.ini

```
TuningParameters:
    TolerateRepositoryFailure=TRUE
```

4. 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 version 7.5, an application can set any one of the connections to `MQCNO_FASTPATH_BINDING`. In version 7.5, only the first connection can be set to `MQCNO_FASTPATH_BINDING`. See Fast path for the complete set of rules.
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 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 WebSphere 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 WebSphere 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 WebSphere 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.

5. Changes to data types

A number of data types have changed between WebSphere® MQ version 7.0.1 to WebSphere MQ version 7.5 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 1 lists the data types that have new versions. Click on the links to read about the new fields.

Table 1. 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)

6. Default transmission queue restriction

The product documentation in previous versions of WebSphere® MQ warned about defining the default transmission queue as `SYSTEM.CLUSTER.TRANSMIT.QUEUE`. In version 7.5, any attempt to set or use a default transmission queue that is defined as `SYSTEM.CLUSTER.TRANSMIT.QUEUE` results in an error.

In earlier versions of WebSphere MQ 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 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 error.

7. Display channel and cluster status: Switching

A cluster-sender channel that is switching its configuration to a different cluster transmission queue has a new channel state: Switching.

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.

8. 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.

If a 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 left at its default value of the empty string (`""`), the 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.

9. JMS: Change in behaviour of the default user identifier value

Change caused when using CLIENT transport for a channel that does not have a security exit defined.

If a 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 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 behaviour 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.

10. Java: Different message property data type returned

If the data type of a message property is set, the same data type is returned when the message is received. In some circumstances in WebSphere® MQ version 7.0.1, 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 WebSphere MQ version 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.

11. JMS: Reason code changes

Some reason codes returned in JMS exceptions have changed. The changes affect `MQRC_Q_MGR_NOT_AVAILABLE` and `MQRC_SSL_INITIALIZATION_ERROR`.

In earlier releases of WebSphere® MQ, if a JMS application call fails to connect, it receives an exception with a reason code 2059 (080B) (RC2059): MQRC_Q_MGR_NOT_AVAILABLE. In version 7.5, it 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. In version 7.5, it 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.

12. JMS: ResourceAdapter object configuration

When a WebSphere® Application Server connects to WebSphere MQ it creates Message Driven Beans (MDBs) using JMS connections. 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. In version 7.5, 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.

13. MQI and PCF reason code changes

Reason codes that have changed and that affect some existing programs, are listed.

`MQRC_NOT_OPEN_FOR_INPUT`
: In WebSphere® MQ version 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 7.5 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 previous versions of WebSphere MQ warned about defining the default transmission queue as `SYSTEM.CLUSTER.TRANSMIT.QUEUE`. In version 7.5, 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.

`MQRCCF_DEF_XMIT_Q_CLUS_ERROR`

The product documentation in previous versions of WebSphere MQ warned about defining the default transmission queue as `SYSTEM.CLUSTER.TRANSMIT.QUEUE`. In version 7.5, 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.

14. 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. This changes the behavior of incorrectly written publish/subscribe applications migrated from version 6.0. 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 WebSphere MQ version 6.0 the product documentation stated The subscriber queue must not be a temporary dynamic queue; see Queue name. But the injunction was not enforced, although the configuration is not supported.
- In WebSphere Message Broker and WebSphere Event Broker version 6.0, and WebSphere Message Broker version 6.1, you could create a subscription that used a temporary dynamic queue as the subscriber queue. If the queue was deleted, the subscription was deleted with it.
- In WebSphere MQ version 7.0, if you migrate or create a queued publish/subscribe application that uses `MQRFH1`, it behaves the same as WebSphere MQ version 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 WebSphere MQ version 6.0. The lack of a subscriber queue results in any matching publications ending up on the dead letter queue. This behavior is the same as WebSphere MQ version 6.0. `MQRFH1` publish/subscribe applications are typically migrated from WebSphere MQ version 6.0.
- In WebSphere MQ version 7.0.1 from fix pack 7.0.1.6 onwards, and in version 7.5, 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. It is a change from WebSphere MQ version 6.0.
- In version 7.5, if you migrate or create a queued publish/subscribe application that uses `MQRFH2`, it behaves the same as WebSphere Event Broker and WebSphere Message Broker. You can create a temporary dynamic queue for a subscription, and if the queue is deleted, the subscription is deleted, as in WebSphere Event Broker and WebSphere Message Broker. `MQRFH2` publish/subscribe applications are typically migrated from WebSphere Event Broker or WebSphere Message Broker.

- In WebSphere MQ version 7.5, if you create a durable subscription using integrated publish/subscribe, you cannot define a temporary dynamic queue as the destination for its matching publications.
- In WebSphere MQ version 7.5, 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

In version 7.5, you cannot create a temporary dynamic queue as the destination for publications for a durable subscription using the integrated publish/subscribe interface.

In the current fix level of version 7.5, 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. Deleting the subscription with the queue retains the same the supported behavior of WebSphere MQ version 6.0, WebSphere Event Broker, and WebSphere Message Broker applications. It modifies the unsupported behavior of WebSphere MQ version 6.0 applications.

15. SSLPEER and SSLCERTI changes

WebSphere® MQ version 7.5 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 previous releases.

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 1 and Table 2.

Table 1. 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 2. 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.

16. Queue manager logs: Default sizes increased

The default size of a queue manager log files has been changed to 4096. The `AMQERR`*`nn`*`.log` has increased from 256 KB to 2 MB on UNIX, Linux, and Windows platforms. The change affects both new and migrated queue managers.

Queue manager log

In WebSphere® MQ Version 7.5 default log size is 4096. For more information on setting non-default values, see The WebSphere MQ configuration file, mqs.ini.

Queue manager error log

Override the change by setting the environment variable `MQMAXERRORLOGSIZE`, or 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.