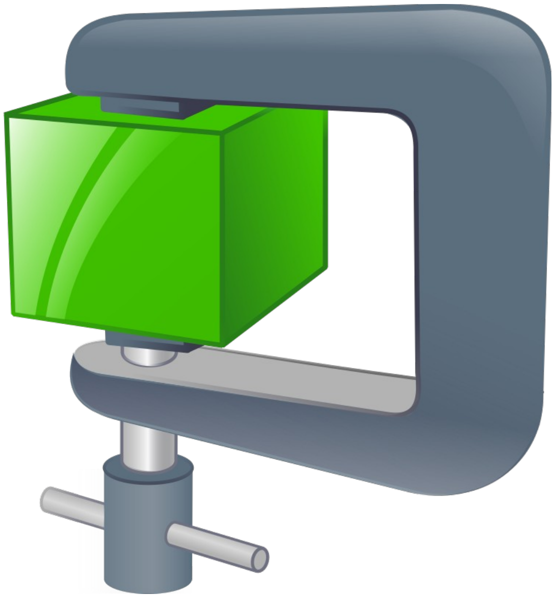
MQ Message Compression Installation and Operation Manual



Capitalware Inc.
Unit 11, 1673 Richmond Street, PMB524
London, Ontario N6G2N3
Canada
sales@capitalware.com
https://www.capitalware.com

Last Updated: January 2021.

Table of Contents

1 INTRODUCTION..1

1.1 OVERVIEW..1
1.2 EXECUTIVE SUMMARY..5
1.3 PREREQUISITES..6
1.3.1 Operating System..6
1.3.2 IBM MQ..7
1.3.3 Windows 32-bit..7
1.3.4 Windows 64-bit..7

2 INSTALLING MQ MESSAGE COMPRESSION..8

2.1 API EXIT..8
2.1.1 Windows Installation..8
2.1.2 Linux 32-bit Installation..8
2.1.3 Unix and Linux 64-bit Installation..9
2.1.4 IBM i Installation..10
2.1.5 MQMC-GUI Installation..11

3 CONFIGURING MQMC..12

3.1 API EXIT..13
3.1.1 Windows..13
3.1.2 For Linux 32-bit..16
3.1.3 Unix and Linux 64-bit..16
3.1.4 IBM i..17
3.2 FILE PATHS..18
3.2.1 Absolute Path..18
3.2.2 Relative Path..18
3.2.3 Environment Variables..19
3.3 MQMC-GUI..20

4 INIFILE KEYWORDS (GLOBAL VALUES)..21

4.1 ACTIVE..21
4.2 USERIDFORMATTING..21
4.3 COMPRESSION..21
4.4 EXCLUDEAPPLICATIONS..22
4.5 EXCLUDEQUEUES..23
4.6 EXCLUDETOPICS..24
4.7 EXCLUDEUSERIDS..25
4.8 COMPRESSRFH2HEADER..26
4.9 EXITPATH..27
4.10 LICENSEFILE..28
4.11 LICENSE KEY..28
4.12 LOGGING..29

5 ACTIVATING COMPRESSED MESSAGE DATA BY QUEUE..30

5.1 WHAT VALUE IS TO BE USED FOR SECTION NAME FOR A QUEUE?..30
5.1.1 Local Queue..30
5.1.2 Cluster Queue..30

5.1.3 Alias Queue......30
5.1.4 Remote Queue......31
5.2 What Value is to be used for Section Name for a Topic?......32
5.2.1 Topic String......32
5.2.2 Topic Object......32
5.3 Section Name for a Queue......33
5.3.1 Compression......34
5.4 Section Name for a Topic......35
5.4.1 Compression......36
5.5 Example......37
5.5.1 Queue Example......37
5.5.2 Topic Example......38

6 APPENDIX A – SUMMARY OF INIFILE......39

7 APPENDIX B – MQMC UPGRADE PROCEDURES......43

7.1.1 Windows Upgrade......43
7.1.2 Linux 32-bit Upgrade......43
7.1.3 Unix and Linux 64-bit Upgrade......44
7.1.4 IBM i Upgrade......44

8 APPENDIX C – TEST COMPRESSION - TESTCMPRSN......45

8.1 Examples......45
8.1.1 Windows......45
8.1.2 Linux 32-bit......46
8.1.3 Unix or Linux 64-bit......46
8.1.4 IBM i......47

9 APPENDIX D – LOSSLESS COMPRESSION......48

10 APPENDIX E – CAPITALWARE PRODUCT DISPLAY VERSION......49

10.1 Examples......49
10.1.1 Windows......49
10.1.2 Unix and Linux 32-bit......49
10.1.3 Unix and Linux 64-bit......49
10.1.4 IBM i......49

11 APPENDIX F – SUPPORT......50

12 APPENDIX G – SUMMARY OF CHANGES......51

13 APPENDIX H – LICENSE AGREEMENT......52

14 APPENDIX I – NOTICES......54

1 Introduction

1.1 Overview

MQ Message Compression (MQMC) provides compression for MQ message data while it resides in a queue or topic and in the MQ logs (i.e. all data at rest). Data compression is the process of modifying and/or converting the structure of bits of data so that it consumes less space in memory and/or on disk.

Question: Would you trade a little CPU time to drastically reduce the disk I/O time?

MQMC has implemented 7 lossless compression algorithms:

- ***LZ4*** is a lossless data compression algorithm that is focused on compression and decompression speed. It belongs to the LZ77 family of byte-oriented compression schemes. LZ4 algorithm is incredibly fast. MQMC implemented the LZ4 project.

- ***LZW*** encodes sequences of 8-bit data as fixed-length 12-bit codes. I used Michael Dipperstein's implementation of LZW (Lempel-Ziv-Welch). MQMC implemented LZW from Michael Dipperstein's lzw project.

- LZMA uses a dictionary compression algorithm, whose output is then encoded with a range encoder, using a complex model to make a probability prediction of each bit (Lempel–Ziv–Markov). MQMC implemented LZMA SDK from 7-Zip.

  - ***LZMA_FAST*** uses LZMA SDK with a Level set to 4.
  - ***LZMA_BEST*** uses LZMA SDK with a Level set to 5.

- ***RLE*** (Run Length Encoding) encodes sequences of the same data value occurring over many consecutive data elements that are stored as a single data value and count.

- zlib only supports one algorithm, called DEFLATE, which uses a combination of a variation of LZ77 (Lempel–Ziv 1977) and Huffman coding. MQMC implemented zlib from Rich Geldreich's miniz project.

  - ***ZLIB_FAST*** uses zlib with a Level of *Z_BEST_SPEED*.
  - ***ZLIB_BEST*** uses zlib with a Level of *Z_BEST_COMPRESSION*.

Each queue in a queue manager is assigned two buffers to hold messages (one for persistent messages and one for non-persistent messages). The persistent queue buffer size is specified using the tuning parameter ***DefaultPQBufferSize***. The non-persistent queue buffer size is specified using the tuning parameter ***DefaultQBufferSize***.

- DefaultPQBufferSize has a default value of 128KB for 32-bit Queue Managers and 256KB for 64-bit Queue Managers.
- DefaultQBufferSize has a default value of 64KB for 32-bit Queue Managers and 128KB for 64-bit Queue Managers.

Here’s the process of the queue manager handling an application putting a message to a queue:

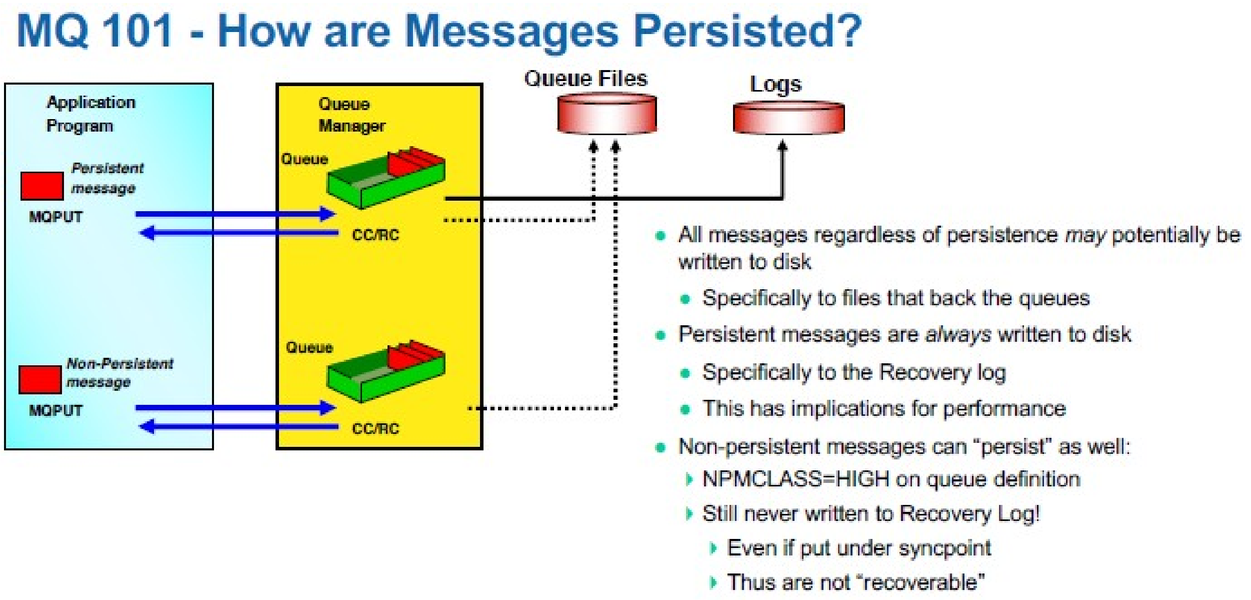
- The message will be put into the buffer of the waiting application if it can fit.
- If that fails, the queue manager tries to write the message to the queue buffer, if it can fit.
- Otherwise, it is written to the queue file.

Here’s the process of the queue manager handling an application getting a message from a queue:

- When the consumer (non-waiting) gets a message from a queue, the queue manager will retrieve it from the queue buffer, if available, otherwise from the queue file.
- If the consumer was waiting for a message, then the queue manager will attempt to write it directly to the applications buffer.

Persistent messages are always written to the recovery log files.

Here is a screen-shot from Chris Frank’s MQ Technical Conference 2016 session called: More Mysteries of the MQ Logger (page 9) that provides a high-level view of disk I/O.



MQ 101 - How are Messages Persisted?

- All messages regardless of persistence *may* potentially be written to disk
  - Specifically to files that back the queues
- Persistent messages are *always* written to disk
  - Specifically to the Recovery log
  - This has implications for performance
- Non-persistent messages can “persist” as well:
  - NPMCLASS=HIGH on queue definition
  - Still never written to Recovery Log!
    - Even if put under syncpoint
    - Thus are not “recoverable”

In the picture, the solid line shows the queue manager writing the messages to the recovery log files. The dotted lines mean that the message may or may not be written to the queue file. See the above for the scenarios of when/why the queue manager would write a message to the queue file.

The goal of MQMC is to improve the disk I/O (Input/Output) speed by compressing the message data when the queue manager is writing to the queue buffers, queue backing files and the recovery log files.

MQMC is an MQ API Exit that operates with IBM MQ v7.1, v7.5, v8.0, v9.0, v9.1 and v9.2 in Windows, Unix, IBM i (OS/400) and Linux platforms.

On AIX, HP-UX, Linux, Solaris and Windows, MQMC can be configured and used with a non-default installation of MQ in a multi-install MQ environment.

Note: Raspberry Pi is a Linux ARM 32-bit OS (Operating System). Hence, simply follow the Linux 32-bit instructions for installing and using the solution on a Raspberry Pi.

MQMC includes an auxiliary program called: testcmprsn. The testcmprsn program allows the end-user to test various types of messages to see which types of messages would benefit from message compression.

Here’s an example of testcmprsn program being run against a 9.17MB XML file (huge file):

```
~/test> ./testcmprsn very_lrg_msg.xml
testcmprsn version 1.0.0.0 (Linux64) {Oct 3 2020}

very_lrg_msg.xml size is 9614354 (9.17MB)
Time taken to perform memcpy() is 4.8770ms

Algorithm               Compressed      Compression     Compression     Decompression
                           Size         Time in ms        Ratio           Time in ms
LZ4                 112253 (109.62KB)       3.4830       85.65 to 1         2.9540
LZMA Fast            32872 (32.10KB)      108.4230      292.48 to 1        11.0730
LZMA Best            27675 (27.03KB)     1152.6960      347.40 to 1        10.6730
LZW                 287184 (280.45KB)     203.0840       33.48 to 1        80.8820
RLE               13213500 (12.60MB)       28.1200        0.73 to 1        26.2680
ZLIB Fast           240612 (234.97KB)      28.3140       39.96 to 1        11.2530
ZLIB Best            83375 (81.42KB)       88.5010      115.31 to 1         8.4590
testcmprsn is ending.
```

1.2 Executive Summary

MQMC is an MQ API Exit. The MQ API Exit is available in 3 forms:

- Windows DLL
- Shared library for AIX, HP-UX, Linux, and Solaris.
- IBM i exit module

The major features of MQMC are as follows:

- No application changes required
- All message data written to a selected queue and/or topic will be compressed (nothing missed or forgotten)
- Compression/decompression algorithms used: LZ4, LZW, LZMA_FAST, LZMA_BEST, RLE, ZLIB_FAST & ZLIB_BEST
- Standard MQ feature, GET-with-Convert, is supported
- Provides high-level logging capability for compression/decompression processing

1.3 Prerequisites

This section provides the minimum supported software levels. These prerequisites apply to server-side installations of MQ Message Compression.

1.3.1 Operating System

MQ Message Compression can be installed on any of the following supported servers:

1.3.1.1 IBM AIX

- IBM AIX 6L version 6.1 or higher

1.3.1.2 HP-UX IA64

- HP-UX v11.23 or higher

1.3.1.3 IBM i (OS/400)

- IBM i V6R1 or higher

1.3.1.4 Linux x86

- Red Hat Enterprise Linux v5, v6, v7, v8
- SUSE Linux Enterprise Server v11, v12, v15

1.3.1.5 Linux x86_64 (64-bit)

- Red Hat Enterprise Linux v5, v6, v7, v8
- SUSE Linux Enterprise Server v11, v12, v15

1.3.1.6 Linux on POWER

- Red Hat Enterprise Linux v6, v7, v8
- SUSE Linux Enterprise Server v12, v15

1.3.1.7 Linux on zSeries (64-bit)

- Red Hat Enterprise Linux v6, v7, v8
- SUSE Linux Enterprise Server v12, v15

1.3.1.8 Raspberry Pi (Linux ARM 32-bit)

- Raspberry Pi OS v9 or higher

1.3.1.9 Sun Solaris

- Solaris SPARC v10 & v11
- Solaris x86_64 v10 & v11

1.3.1.10 Windows

- Windows 2008, 2012 or 2016 Server (32-bit & 64-bit)
- Windows 7, 8, 8.1 or 10 (32-bit & 64-bit)

1.3.2 IBM MQ

- IBM MQ v7.1, v7.5, v8.0, v9.0, v9.1 and v9.2 (32-bit and 64-bit)

Operating System	MQ v7.1, v7.5, v8.0, v9.0, v9.1 and v9.2
AIX v6.1 or higher	32-bit & 64-bit
HP-UX IA64 v11.23 or higher	32-bit & 64-bit
IBM i (OS/400)	64-bit
Linux x86	32-bit
Linux x86_64	32-bit & 64-bit
Linux on POWER	32-bit & 64-bit
Linux on zSeries	32-bit & 64-bit
Raspberry Pi ARM	32-bit
Solaris SPARC v10 & v11	32-bit & 64-bit
Solaris x86_64 v10 & v11	32-bit & 64-bit
Windows 2008, 2012, 2016, 7, 8, 8.1 & 10	32-bit & 64-bit

1.3.3 Windows 32-bit

The following is the software prerequisite for Windows 32-bit:

- Microsoft Visual C++ 2010 Redistributable Package (x86)
  https://download.microsoft.com/download/1/6/5/165255E7-1014-4D0A-B094-B6A430A6BFFC/vcredist_x86.exe

1.3.4 Windows 64-bit

The following are the software prerequisite for Windows 64-bit:

- Microsoft Visual C++ 2010 Redistributable Package (x64)
  https://download.microsoft.com/download/1/6/5/165255E7-1014-4D0A-B094-B6A430A6BFFC/vcredist_x64.exe

If local 32-bit applications connect in bindings mode to the queue manager then the following needs to be also installed:

- Microsoft Visual C++ 2010 Redistributable Package (x86)
  https://download.microsoft.com/download/1/6/5/165255E7-1014-4D0A-B094-B6A430A6BFFC/vcredist_x86.exe

2 Installing MQ Message Compression

This section describes how to install Capitalware's MQMC.

2.1 API Exit

2.1.1 Windows Installation

To install MQMC on Windows, first unzip the **mqmc.zip** and then run the **mqmc_setup.exe** file. Follow the on-screen instructions and the API exit will be installed in the **C:\Capitalware\ MQMC**\ directory (default installation).

The user may copy or ftp the mqmc.dll, 64\mqmc.dll, rotatelog.bat and mqmc.ini files from one Windows server to another Windows server.

2.1.2 Linux 32-bit Installation

To install the 32-bit version of MQMC on Unix or Linux, first unzip the **mqmc.zip** and then select the appropriate TAR file for the target platform. You will find 2 TAR file in the original ZIP file:

- **Linux_x86/mqmc_linux.tar**
- **RaspberryPi_ARM/mqa_raspberrypi_arm.tar**

Steps to Install MQMC:

- ftp or copy the selected TAR file to the target platform to the ***/var/mqm/*** directory.
- Un-tar the **mqmc_xxx.tar** file into the ***/var/mqm/*** sub-directory (xxx is either aix, hpux, solaris or linux)

```
cd /var/mqm/
tar -xvf  mqmc_xxx.tar
```

- Change directory to ***/var/mqm/exits/***
- Next, do the following commands against ***mqmc***:

```
chmod +x setmqmc.sh
./setmqmc.sh
```

2.1.3 Unix and Linux 64-bit Installation

To install the 64-bit version of MQMC on Unix or Linux, first unzip the **mqmc.zip** and then select the appropriate TAR file for the target platform. You will find 7 TAR files in the original ZIP file:

- **AIX/mqmc_aix71_64.tar** for AIX v7.1 or higher
- **HPUX_IA64 /mqmc_hpux64_ia64.tar**
- **Linux_x86_64/mqmc_linux_x86_64.tar**
- **Linux_POWER/mqmc_linux_power64.tar**
- **Linux_zSeries/mqmc_linux_zseries64.tar**
- **Solaris_SPARC/mqmc_solaris10_64.tar** for Solaris SPARC v10 or higher
- **Solaris_x86_64/mqmc_solaris_x86_64.tar**

Steps to Install MQMC:

- ➢ ftp or copy the selected TAR file to the target platform to the ***/var/mqm/*** directory.
- ➢ Un-tar the **mqmc_xxx.tar** file into the ***/var/mqm/*** sub-directory (xxx is either aix, hpux, solaris or linux)

```
cd /var/mqm/
tar -xvf  mqmc_xxx64.tar
```

- ➢ Change directory to ***/var/mqm/exits64/***
- ➢ Next, do the following commands against ***mqmc***:

```
chmod +x setmqmc.sh
./setmqmc.sh
```

Note: mqmc_r shared library is not required for Solaris.

2.1.4 IBM i Installation

To install the MQMC on IBM i, first unzip the **mqmc.zip** and then select the files in the IBM i (iSeries) directory.

- **mqmc.savf** is the IBM i 'Save File' that contains the library with the API exit.
- **mqmc_iseries.tar** is the IBM i IFS TAR file that contains a sample initialization file for the API Exit.

Steps to install the API Exit:

1. Log onto the target IBM i server and do the following command:

   ```
   CRTSAVF FILE(QGPL/MQMC)
   ```

2. ftp the IBM i files to the IBM i server as follows:

   ```
   ftp -s:mqmc_iseries.ftp  iseries_hostname
   ```

   The mqmc_iseries.ftp file contains the following ftp commands:

   ```
   your-IBM i-userid
   your-IBM i-password

   binary
   cd QGPL
   put mqmc.savf

   quote SITE NAMEFMT 1

   cd /QIBM/UserData/mqm/
   put mqmc_iseries.tar
   quit
   ```

3. Log onto the target IBM i server and do the following commands:

   ```
   RSTLIB SAVLIB(MQMC) DEV(*SAVF) SAVF(QGPL/MQMC)
   CLRSAVF FILE(QGPL/MQMC)
   CHGOBJOWN  OBJ(MQMC) OBJTYPE(*LIB)  NEWOWN(QMQM)
   qsh
   cd /QIBM/UserData/mqm/
   tar -xvf mqmc_iseries.tar
   chown -R QMQM mqmc
   chmod -R 777 mqmc
   rm mqmc_iseries.tar
   ```

2.1.5 MQMC-GUI Installation

This section will describe how to install the MQMC-GUI. The user will find 2 files in the software package listed as follows:

- **MQMC-GUI/mqmcgui-wthjre.exe** (for Windows)
- **MQMC-GUI/mqmcgui.zip** (for Unix, Linux or macOS)

2.1.5.1 MQMC-GUI Installation on Windows

To install MQMC-GUI on Windows, run the **mqmcgui-withjre.exe** file located in the MQMC-GUI directory. Follow the on-screen instructions and the program will be installed in the **C:\ Capitalware\MQMC-GUI\** directory (default installation).

2.1.5.2 MQMC-GUI Installation on Unix, Linux or macOS

To install MQMC-GUI on Unix or Linux, you will need to ftp or copy the selected TAR file to the target platform to the /home/mqm/ directory. Next, one must telnet to the Unix, Linux or macOS server and 'cd' (change directory) to the /home/mqm/ directory and unzip the archive file.

i.e. Do the following command:

```
   unzip mqmcgui.zip
```

3 Configuring MQMC

This section describes how to configure the MQMC (MQ API Exit) on the support platforms.

Platform		Directory	Exit Module Name
Windows	32-bit	C:\Capitalware\MQMC\	mqmc.dll
Windows	64-bit	C:\Capitalware\MQMC\64\	mqmc.dll
Linux/Unix	32-bit	/var/mqm/exits/	mqmc
Linux/Unix	64-bit	/var/mqm/exits64/	mqmc

MQMC supports MQ's multi-install in a non-default directory.

After the user has configured the MQMC, the queue manager needs to be restarted.

3.1 API Exit

This section describes the necessary entries to enable the API Exit. The MQ Administrator will need to create a Local API Exit definition. This procedure differs depending on the platform.

Note: The value for the Data parameter cannot exceed a length of 32 characters.

3.1.1 Windows

On Windows, there are 2 ways to create the Local API Exit definition: via the command line or MQ Explorer. Note: Using MQ Explorer to add the API Exit is very easy.

3.1.1.1 Windows Command Line

First, the user will need to manually edit the queue manager's qm.ini file to update the ExitsDefaultPath and ExitsDefaultPath64 fields. The qm.ini file will be located at: ***C:\Program Files (x86)\IBM\IBM MQ\Qmgrs\{QMgrName}\qm.ini***

```
ExitPath:
   ExitsDefaultPath=C:\Capitalware\MQMC;C:\Program Files (x86)\IBM\IBM MQ\exits
   ExitsDefaultPath64=C:\Capitalware\MQMC\64;C:\Program Files (x86)\IBM\IBM MQ\exits64
```

Next, create the Local API definition via the command line using the MQ ***amqmdain*** program. The MQAdmin can issue the following commands to create the Local API Exit definition:

```
amqmdain reg {QMgrName} -c add  -s ApiExitLocal\MQMC -v Name=MQMC
amqmdain reg {QMgrName} -c add  -s ApiExitLocal\MQMC -v Module=mqmc.dll
amqmdain reg {QMgrName} -c add  -s ApiExitLocal\MQMC -v Data=C:\Capitalware\MQMC\mqmc.ini
amqmdain reg {QMgrName} -c add  -s ApiExitLocal\MQMC -v Sequence=3
amqmdain reg {QMgrName} -c add  -s ApiExitLocal\MQMC -v Function=EntryPoint
```

Where ***{QMgrName}*** is the name of the Queue Manager.

As a convenience, we have included a batch file called ***mqmc_reg.bat*** that includes all of the **amqmdain** commands. **mqmc_reg.bat** is located in the MQMC install directory and it accepts one parameter (QMgrName).

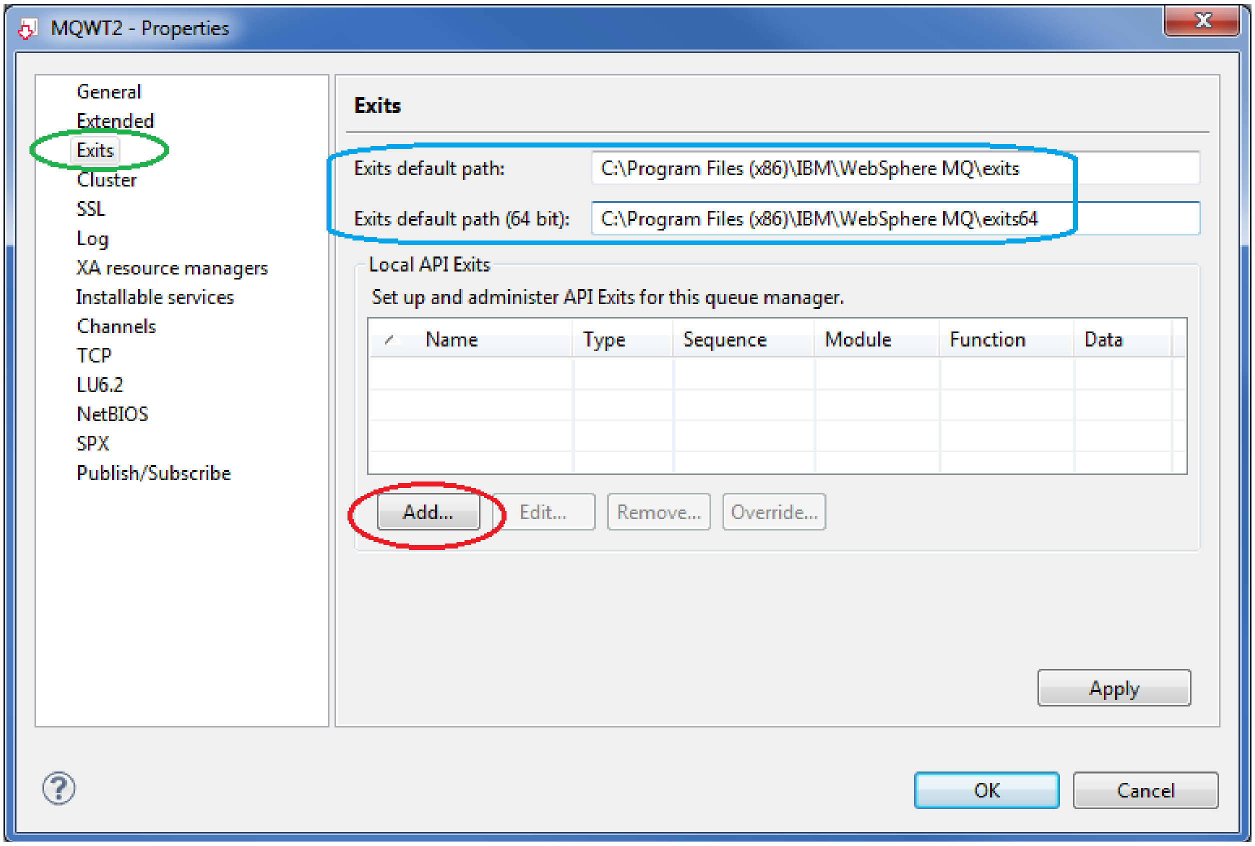
Open a Windows Command Prompt and issue the following commands to configure MQMC for a particular queue manager:

```
cd /D C:\Capitalware\MQMC\
mqmc_reg.bat {QMgrName}
```

3.1.1.2 Windows MQ Explorer

To create a Local API Exit definition using MQ Explorer, do the following:

- Start MQ Explorer
- Right-click on the queue manager name
- Select ***Properties*** from the popup menu
- Select ***Exits*** in the left panel of the properties window
- Update the 'Exit default path' and 'Exit default path (64-bit)' with the MQMC install path. i.e. C:\Capitalware\MQMC and C:\Capitalware\MQMC\64
- Click the ***Add*** button.



MQWT2 - Properties

Exits

Exits default path: C:\Program Files (x86)\IBM\WebSphere MQ\exits

Exits default path (64 bit): C:\Program Files (x86)\IBM\WebSphere MQ\exits64

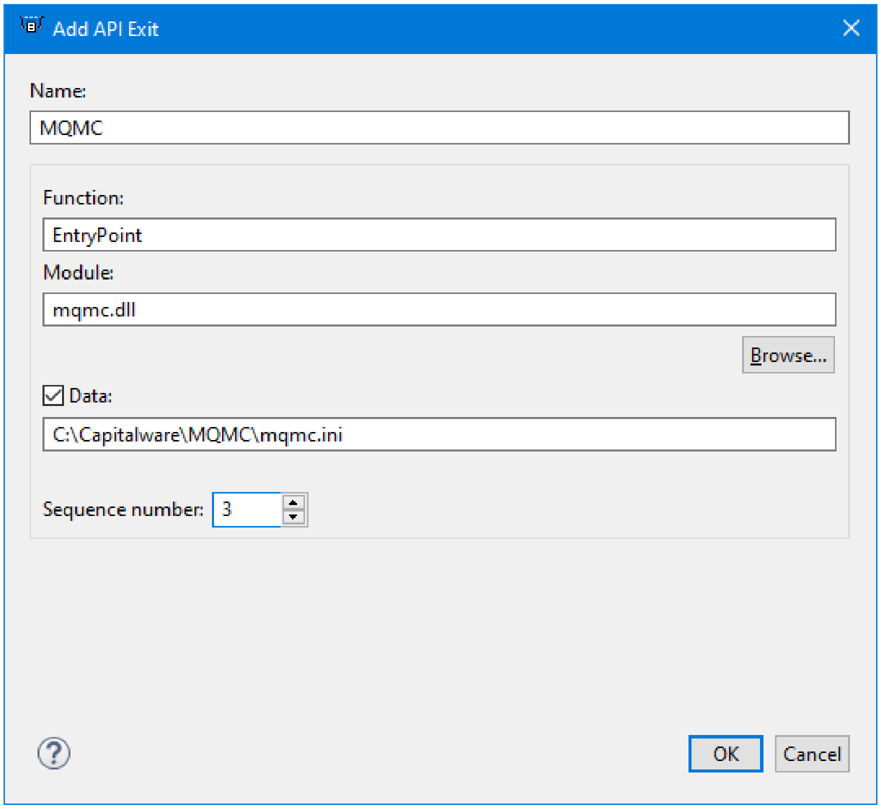
Local API Exits

Set up and administer API Exits for this queue manager.

Name	Type	Sequence	Module	Function	Data

The user is required to input the following values into the 5 fields of the Add API Exit window:

Field Name	Value
Name	MQMC
Function	EntryPoint
Module	mqmc.dll
Data	C:\Capitalware\MQMC\mqmc.ini
Sequence Number	3



Name: MQMC

Function: EntryPoint

Module: mqmc.dll

☑ Data: C:\Capitalware\MQMC\mqmc.ini

Sequence number: 3

Click ***OK*** when the information has been inputted.

3.1.2 For Linux 32-bit

Edit the ***qm.ini*** for the queue manager that MQMC is being applied to. The **qm.ini** file is located at **/var/mqm/qmgrs/{QMgrName}/qm.ini**

Make sure the ExitPath stanza exists in the qm.ini and then add the ApiExitLocal stanza as given below:

```
ExitPath:
   ExitsDefaultPath=/var/mqm/exits/

ApiExitLocal:
  Name=MQMC
  Sequence=3
  Function=EntryPoint
  Module=mqmc
  Data=/var/mqm/exits/mqmc.ini
```

Note: If the user has not installed MQMC in /var/mqm/exits/ directory then the ExitsDefaultPath field will need to be updated with the path to the shared library.

3.1.3 Unix and Linux 64-bit

Edit the ***qm.ini*** for the queue manager that MQMC is being applied to. The **qm.ini** file is located at **/var/mqm/qmgrs/{QMgrName}/qm.ini**

Make sure the ExitPath stanza exists in the qm.ini and then add the ApiExitLocal stanza as given below:

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

ApiExitLocal:
  Name=MQMC
  Sequence=3
  Function=EntryPoint
  Module=mqmc
  Data=/var/mqm/exits64/mqmc.ini
```

Note: If the user has not installed MQMC in /var/mqm/exits/ and /var/mqm/exits64/ directories then the ExitsDefaultPath and ExitsDefaultPath64 fields will need to be updated with the path to the shared library.

3.1.4 IBM i

On IBM i, edit the ***qm.ini*** for the queue manager that MQMC is being applied to. The qm.ini file is located at **/QIBM/UserData/mqm/qmgrs/{QMgrName}/qm.ini**

Add the ApiExitLocal stanza as given below:

```
ApiExitLocal:
  Name=MQMC
  Sequence=3
  Function=EntryPoint
  Module=MQMC/MQMC
  Data=/QIBM/UserData/mqm/mqmc/mqmc.ini
```

3.2 File Paths

Data field of the ApiExitLocal stanza must NOT exceed 32 characters. In order to work with this limitation, MQMC supports 3 ways to specify an IniFile path: absolute path, relative path and environment variable.

Note: The IniFile path that is determined by MQMC API Exit will also be used for the following IniFile keywords (if no pathing is specified for these keywords): **LicenseFile** and **LogFile**.

3.2.1 Absolute Path

Absolute pathing (specifying the complete path) for the Data field works on Linux, Unix and Windows platforms.

E.g. Windows

```
Data=C:\Capitalware\MQMC\mqmc.ini
```

Hence, MQMC will use the following path as the IniFile path:

```
C:\Capitalware\MQMC\
```

3.2.2 Relative Path

Relative pathing for the Data field is supported on Linux, IBM i, Unix and Windows platforms. MQMC will extract the path from Data field and prefix it to the IniFile specified in the Data field in order to locate the IniFile.

For Windows:

```
C:\Capitalware\MQMC\
```

For IBM MQ 32-bit on Unix and Linux:

```
/var/mqm/exits/
```

For IBM MQ 64-bit on Unix and Linux:

```
/var/mqm/exits64/
```

For IBM MQ on IBM i:

```
/QIBM/UserData/mqm/mqmc/
```

E.g. Unix

```
Data=mqmc.ini
```

Hence, MQMC will use the following path as the IniFile path:

E.g. 32-bit

```
/var/mqm/exits/
```

E.g. 64-bit

```
/var/mqm/exits64/
```

3.2.3 Environment Variables

3.2.3.1 Global Environment Variable

MQMC supports the use of the MQMC_HOME environment variable which holds the directory path information. MQMC_HOME environment variable is supported on Linux, IBM i, Unix and Windows platforms.

e.g. Unix

```
export MQMC_HOME=/really/long/path/MQHA/QMgrName/data/
```

```
Data=mqmc.ini
```

Hence, MQMC will use the following path as the IniFile path:
/really/long/path/MQHA/QMgrName/data/

3.2.3.2 Queue Manager Specific Environment Variable

MQMC supports the use of the MQMC_HOME environment variable post-fixed with the queue manager name which holds the directory path information. MQMC_HOME environment variable post-fixed with the queue manager name is supported on Linux, IBM i, Unix and Windows platforms.

e.g. Unix with a queue manager name of MQA1

```
export MQMC_HOME_MQA1=/really/long/path/MQHA/QMgrName/data2/
```

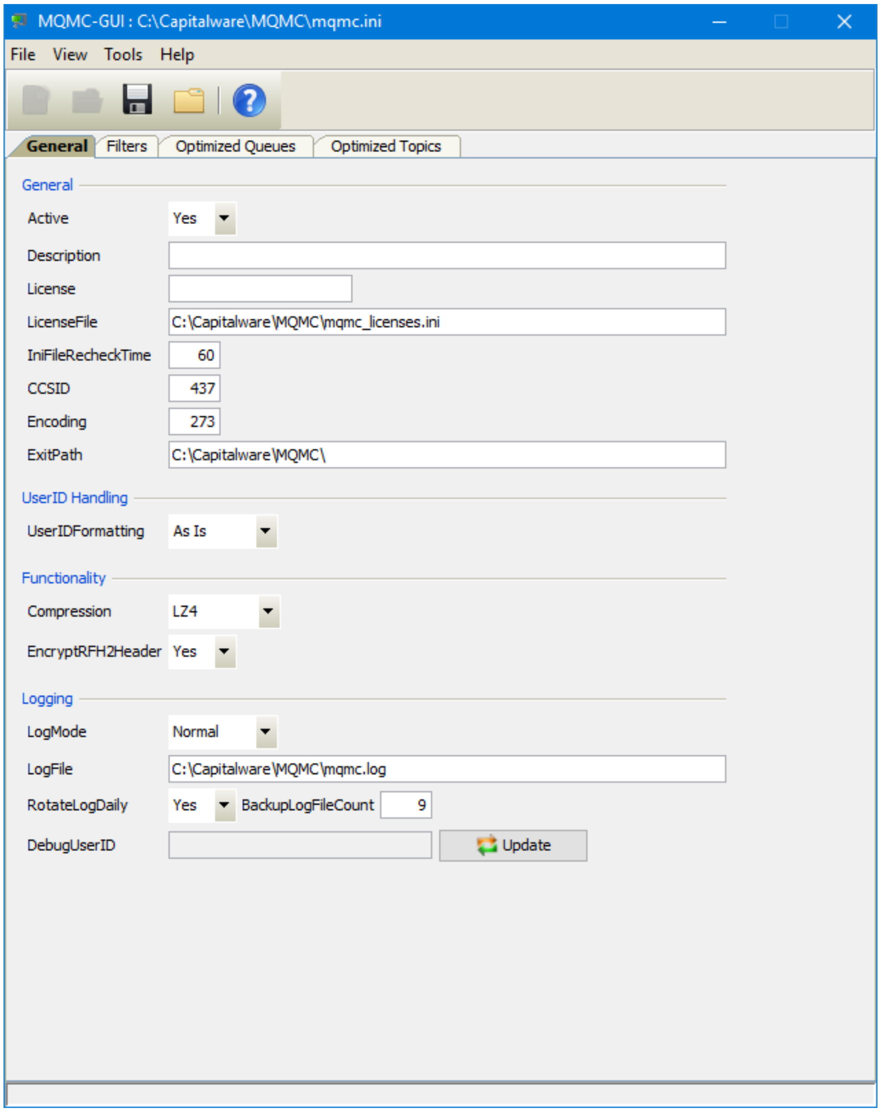
```
Data=mqmc.ini
```

Hence, MQMC will use the following path as the IniFile path:
/really/long/path/MQHA/QMgrName/data2/

Note: If both environment variables are specified then the queue manager specific environment variable will be used.

3.3 MQMC-GUI

This section briefly describes the new graphical program called MQMC-GUI. This program assists the user in creating and managing their MQMC IniFiles. For more information, please see the ***MQMC-GUI User Guide*** manual.



General

Active: Yes

Description:

License:

LicenseFile: C:\Capitalware\MQMC\mqmc_licenses.ini

IniFileRecheckTime: 60

CCSID: 437

Encoding: 273

ExitPath: C:\Capitalware\MQMC\

UserID Handling

UserIDFormatting: As Is

Functionality

Compression: LZ4

EncryptRFH2Header: Yes

Logging

LogMode: Normal

LogFile: C:\Capitalware\MQMC\mqmc.log

RotateLogDaily: Yes BackupLogFileCount: 9

DebugUserID:

4 IniFile Keywords (Global Values)

This section describes IniFile keywords.

4.1 Active

MQMC can be enabled or disabled by the IniFile's property keyword 'Active'. When Active has a value of Y, MQMC is activated. The default value is Y.

```
Active=Y
```

4.2 UserIDFormatting

This section describes the necessary entries on how to handle the incoming UserID. 'UserIDFormatting' supports 3 values [A / U / L]. ('As Is, Uppercase and Lowercase). The default value is A.

```
UserIDFormatting=U
```

4.3 Compression

Compression indicates which functionality that MQMC will perform. Compression supports 7 values (LZ4, LZW, LZMA_FAST, LZMA_BEST, RLE, ZLIB_FAST & ZLIB_BEST). The default value is LZ4.

- ***LZ4*** is a lossless data compression algorithm that is focused on compression and decompression speed. It belongs to the LZ77 family of byte-oriented compression schemes. LZ4 algorithm is incredibly fast.
- ***LZW*** (Lempel-Ziv-Welch) encodes sequences of 8-bit data as fixed-length 12-bit codes. I used Michael Dipperstein’s implementation of LZW (Lempel-Ziv-Welch).
- ***LZMA_FAST*** uses the LZMA SDK from 7-Zip with a Level set to 4.
- ***LZMA_BEST*** uses the LZMA SDK from 7-Zip with a Level set to 5.
- ***RLE*** (Run Length Encoding) encodes sequences of the same data value occurring over many consecutive data elements are stored as a single data value and count.
- ***ZLIB_FAST*** uses zlib with a Level of *Z_BEST_SPEED*.
- ***ZLIB_BEST*** uses zlib with a Level of *Z_BEST_COMPRESSION*.

Note: LZMA_Fast and LZMA_Best are currently not available for IBM i.

```
Compression=LZMA_FAST
```

4.4 ExcludeApplications

This section describes how to explicitly exclude applications from the compression/decompression processing when an application has connected to the queue manager. MQMC will look up the regular expression patterns from the **ExcludeApplications** keyword in order to determine if the application name matches any of the specified expression patterns.

If there is a match, the application is excluded from compression/decompression process. Each regular expression pattern is separated from the next pattern by a semi-colon (';').

In the regular expression pattern:

- '*' matches any sequence of characters (zero or more)
- '?' matches any single character
- '#' matches any single numeric digit (0-9)
- '@' matches any single alphabetic character (A-Z, a-z)
- [SET] matches any of the characters in the specified set
- [!SET] or [^SET] matches any character except those specified in the set (negation).

A SET can be composed of characters or ranges. A range is in the form: 'character – character' (i.e. 0-9 or A-Z). Although this is the simplest range allowed in the [ ] pattern, more complex inclusive ranges such as [0-9a-zA-Z] are allowed. [0-9a-zA-Z] specifies that the character can be 0 through 9 **or** a through z **or** A through Z. Other characters are allowed (ie. 8 bit characters) if your system supports them.

In order to suppress the special syntactic significance of any of these characters '[] * ? # @ ! ^ - \', a backslash ('\') must precede the special character.

Note: ExcludeApplications must NOT exceed 2048 characters.

To authorize MQMC decompression by application name, you need 2 keywords in the IniFile:

- **UseExcludeApplications** activates this feature. If it has a value of Y, MQMC will exclude applications from the compression/decompression process.
- **ExcludeApplications** specifies the list of application names that are to be excluded.

```
UseExcludeApplications=Y
ExcludeApplications=abc*;gadget;xyz*
```

4.5 ExcludeQueues

This section describes how to explicitly exclude queues from the compression/decompression processing when an application has connected to the queue manager. MQMC will look up the regular expression patterns from the **ExcludeQueues** keyword in order to determine if the queue name matches any of the specified expression patterns.

If there is a match, the queue is excluded from compression/decompression process. Each regular expression pattern is separated from the next pattern by a semi-colon (';').

In the regular expression pattern:

- '*' matches any sequence of characters (zero or more)
- '?' matches any single character
- '#' matches any single numeric digit (0-9)
- '@' matches any single alphabetic character (A-Z, a-z)
- [SET] matches any of the characters in the specified set
- [!SET] or [^SET] matches any character except those specified in the set (negation).

A SET can be composed of characters or ranges. A range is in the form: 'character – character' (i.e. 0-9 or A-Z). Although this is the simplest range allowed in the [ ] pattern, more complex inclusive ranges such as [0-9a-zA-Z] are allowed. [0-9a-zA-Z] specifies that the character can be 0 through 9 **or** a through z **or** A through Z. Other characters are allowed (ie. 8 bit characters) if your system supports them.

In order to suppress the special syntactic significance of any of these characters '[] * ? # @ ! ^ - \', a backslash ('\') must precede the special character.

Note: ExcludeQueues must NOT exceed 2048 characters.

To authorize MQMC decompression by application name, you need 2 keywords in the IniFile:

- **UseExcludeQueues** activates this feature. If it has a value of Y, MQMC will exclude queues from the compression/decompression process.
- **ExcludeQueues** specifies the list of queue names that are to be excluded.

```
UseExcludeQueues=Y
ExcludeQueues=SYSTEM*;ABC[0-9][a-f]
```

4.6 ExcludeTopics

This section describes how to explicitly exclude Topics from the compression/decompression processing when an application has connected to the queue manager. MQMC will look up the regular expression patterns from the **ExcludeTopics** keyword in order to determine if the Topic name matches any of the specified expression patterns.

If there is a match, the Topic is excluded from compression/decompression process. Each regular expression pattern is separated from the next pattern by a semi-colon (';').

In the regular expression pattern:

- '*' matches any sequence of characters (zero or more)
- '?' matches any single character
- '#' matches any single numeric digit (0-9)
- '@' matches any single alphabetic character (A-Z, a-z)
- [SET] matches any of the characters in the specified set
- [!SET] or [^SET] matches any character except those specified in the set (negation).

A SET can be composed of characters or ranges. A range is in the form: 'character – character' (i.e. 0-9 or A-Z). Although this is the simplest range allowed in the [ ] pattern, more complex inclusive ranges such as [0-9a-zA-Z] are allowed. [0-9a-zA-Z] specifies that the character can be 0 through 9 **or** a through z **or** A through Z. Other characters are allowed (ie. 8 bit characters) if your system supports them.

In order to suppress the special syntactic significance of any of these characters '[] * ? # @ ! ^ - \', a backslash ('\') must precede the special character.

Note: ExcludeTopics must NOT exceed 2048 characters.

To authorize MQMC decompression by application name, you need 2 keywords in the IniFile:

- **UseExcludeTopics** activates this feature. If it has a value of Y, MQMC will exclude Topics from the compression/decompression process.
- **ExcludeTopics** specifies the list of Topic names that are to be excluded.

```
UseExcludeTopics=Y
ExcludeTopics=test/ABC*;test/XYZ[0-9][a-f]
```

4.7 ExcludeUserIDs

This section describes how to explicitly exclude UserIds from the compression/decompression processing when a UserId has connected to the queue manager. MQMC will look up the regular expression patterns from the **ExcludeUserIDs** keyword in order to determine if the UserId matches any of the specified expression patterns.

If there is a match, the UserId is excluded from compression/decompression process. Each regular expression pattern is separated from the next pattern by a semi-colon (';').

In the regular expression pattern:

- '*' matches any sequence of characters (zero or more)
- '?' matches any single character
- '#' matches any single numeric digit (0-9)
- '@' matches any single alphabetic character (A-Z, a-z)
- [SET] matches any of the characters in the specified set
- [!SET] or [^SET] matches any character except those specified in the set (negation).

A SET can be composed of characters or ranges. A range is in the form: 'character – character' (i.e. 0-9 or A-Z). Although this is the simplest range allowed in the [ ] pattern, more complex inclusive ranges such as [0-9a-zA-Z] are allowed. [0-9a-zA-Z] specifies that the character can be 0 through 9 **or** a through z **or** A through Z. Other characters are allowed (ie. 8 bit characters) if your system supports them.

In order to suppress the special syntactic significance of any of these characters '[] * ? # @ ! ^ - \', a backslash ('\') must precede the special character.

Note: ExcludeUserIDs must NOT exceed 2048 characters.

To authorize MQMC decompression by UserId name, you need 2 keywords in the IniFile:

- **UseExcludeUserIDs** activates this feature. If it has a value of Y, MQMC will exclude UserIds from the compression/decompression process.
- **ExcludeUserIDs** specifies the list of UserId names that are to be excluded.

```
UseExcludeUserIDs=Y
ExcludeUserIDs=fred;barney;wilma;betty
```

4.8 CompressRFH2Header

CompressRFH2Header specifies that the MQRFH2 header fields and folders are to be compressed when the message payload is compressed. CompressRFH2Header supports 2 values [Y / N]. The default value is Y.

```
CompressRFH2Header=Y
```

4.9 ExitPath

This section describes the use of the keyword ExitPath to explicitly set the API exit path. By setting ExitPath to a particular path, the default value for ExitPath is overridden. The defaults are as follows:

For Windows:

```
ExitPath=C:\Capitalware\MQMC\
```

For IBM MQ 32-bit on Unix and Linux:

```
ExitPath=/var/mqm/exits/
```

For IBM MQ 64-bit on Unix and Linux:

```
ExitPath=/var/mqm/exits64/
```

For IBM MQ on IBM i:

```
ExitPath=/QIBM/UserData/mqm/mqmc/
```

```
ExitPath=D:\Vendor\Capitalware\MQMC\
```

4.10 LicenseFile

This section will describe how to have a file that contains all of the user's MQMC license keys.

The format of the LicenseFile is similar to an IniFile or properties file where each keyword has an associated value. Each keyword and its value are on a separate line. The format is as follows:

```
QMgrName = License_Key
```

Example:

```
MQA1 = 10M0-AAAA-BBBBBBBB
MQB1 = 10M0-XXXX-CCCCCCCC
```

If the queue manager name is not found in the LicenseFile then the License keyword will be used to retrieve the license key value.

The following are the default values for LicenseFile:

For Windows:

```
LicenseFile=C:\Capitalware\MQMC\mqmc_licenses.ini
```

For IBM MQ 32-bit on Unix and Linux:

```
LicenseFile=/var/mqm/exits/mqmc_licenses.ini
```

For IBM MQ 64-bit on Unix and Linux:

```
LicenseFile=/var/mqm/exits64/mqmc_licenses.ini
```

For IBM MQ on IBM i:

```
LicenseFile=/QIBM/UserData/mqm/mqmc/mqmc_licenses.ini
```

4.11 License Key

This section will describe how to license MQ Message Compression to a particular queue manager.

Note: The License keyword is not required if the user has implemented the LicenseFile keyword or the License file actually exists in the default location.

Your license will look something like: 10M0-AAAA-BBBBBBBB (Note: This is a sample license only and will NOT work).

```
License=10M0-AAAA-BBBBBBBB
```

4.12 Logging

This section describes the necessary entries to enable MQMC to record log information. To enable and control logging, there are 4 keywords in the IniFile:

1. **LogMode** specifies what type of logging the user wishes to have. LogMode supports 4 values [Q/N/V/D] where Q is Quiet, N is Normal, V is Verbose and D is Debug. The default value is N.

2. **LogFile** specifies the location of the log file. The default values are as follows:

   For Windows:
   ```
       LogFile=C:\Capitalware\MQMC\mqmc.log
   ```

   For IBM MQ 32-bit on Unix and Linux:
   ```
       LogFile=/var/mqm/mqmc/mqmc.log
   ```

   For IBM MQ 64-bit on Unix and Linux:
   ```
       LogFile=/var/mqm/mqmc/mqmc.log
   ```

   For IBM MQ on IBM i:
   ```
       LogFile=/QIBM/UserData/mqm/mqmc/mqmc.log
   ```

   Token Replacement for LogFile keyword:
   - **%QM%** - Substitutes the name of the queue manager
   - **%UID%** - Substitutes the UserID
   - **%PID%** - Substitutes the Process ID
   - **%TID%** - Substitutes the Thread ID

3. **RotateLogDaily** specifies whether or not the log files will be rotated on a daily basis. A Y value for ' RotateLogDaily' will activate this feature; otherwise, the log files will left as is. The default value is Y.

   In other words, it is possible to keep up to 9 backup log files. The first connection request after midnight (and not at midnight) will cause it to roll/rotate the log files. If there are already 9 backup log files, the ninth backup log file will be deleted and 8 becomes 9, 7 becomes 8, etc...

4. **BackupLogFileCount** specifies the number of backup log files that should be kept by MQMC. The default value is 9. This keyword is only used if RotateLogDaily is set to 'Y'.

5 Activating Compressed Message Data by Queue

This section describes IniFile keywords to activate MQMC compression/decompression against queues of a queue manager

5.1 What Value is to be used for Section Name for a Queue?

The section will describe how to determine what value should be used for the Section Name/Queue Name. MQMC uses the "Resolved Queue" value when processing what queue to replicate.

5.1.1 Local Queue

For a local queue, simply use the local queue name for the Section Name/Queue Name.

5.1.2 Cluster Queue

For a cluster queue, simply use the cluster queue name for the Section Name/Queue Name.

5.1.3 Alias Queue

For an alias queue, if the client application opened the queue with:

- MQOD version 1 or 2, the user MUST use the '***QUEUE***' value from the alias queue definition for the Section Name/Queue Name.
- MQOD version 3 or higher, the user MUST use the '***TARGET***' value from the alias queue definition for the Section Name/Queue Name.

Example:

```
DISPLAY QALIAS(TEST.Q1.AL)
     1 : DISPLAY QALIAS(TEST.Q1.AL)
AMQ8409: Display Queue details.
   QUEUE(TEST.Q1.AL)                       TYPE(QALIAS)
   ALTDATE(2012-12-04)                     ALTTIME(15.51.41)
   TARGET(TEST.Q1)                         CLUSNL( )
   CLUSTER( )                              CLWLPRTY(0)
   CLWLRANK(0)                             CUSTOM( )
   DEFBIND(OPEN)                           DEFPRTY(0)
   DEFPSIST(NO)                            DEFPRESP(SYNC)
   DEFREADA(NO)                            DESCR( )
   GET(ENABLED)                            PUT(ENABLED)
   PROPCTL(COMPAT)                         SCOPE(QMGR)
   TARGTYPE(QUEUE)
```

5.1.4 Remote Queue

For a remote queue, if the client application opened the queue with:

- MQOD version 1 or 2, the user MUST use the '***QUEUE***' value from the alias queue definition for the Section Name/Queue Name.
- MQOD version 3 or higher, the user MUST use the '***RNAME***' value from the alias queue definition for the Section Name/Queue Name.

Example:

```
DISPLAY QREMOTE(TEST.Q2.RQ)
     1 : DISPLAY QREMOTE(TEST.Q2.RQ)
AMQ8409: Display Queue details.
   QUEUE(TEST.Q2.RQ)                       TYPE(QREMOTE)
   ALTDATE(2012-12-04)                     ALTTIME(15.51.41)
   CLUSNL( )                               CLUSTER( )
   CLWLPRTY(0)                             CLWLRANK(0)
   CUSTOM( )                               DEFBIND(OPEN)
   DEFPRTY(0)                              DEFPSIST(NO)
   DEFPRESP(SYNC)                          DESCR( )
   PUT(ENABLED)                            RQMNAME(MQWT2)
   RNAME(TEST.Q2)                          SCOPE(QMGR)
   XMITQ(MQWT2.XMIT)
```

5.2 What Value is to be used for Section Name for a Topic?

The section will describe how to determine what value should be used for the Section Name/Topic Name. MQMC uses the "Resolved Queue" value when processing what queue to replicate.

5.2.1 Topic String

For a topic string, simply use the topic string as the section name.

5.2.2 Topic Object

For a topic object, the user MUST use the resolved topic string.

- If the topic object is defined with a TOPICSTR('test/abc') and the topic object is used as is then the user would use 'test/abc' as the section name.
- If the topic object is defined with a TOPICSTR('test/abc') and the topic object is used with addition topic string of 'extra' then the resolved topic string would be 'test/abc/extra'. Hence, the user would use 'test/abc/extra' as the section name.

5.3 Section Name for a Queue

The section stanza must begin in column 1 with a left square bracket '[' followed by a “Q:” and end with a right square bracket ']'. Between the colon and the right square bracket, the user can explicitly specify a queue name or a regular expression pattern.

The MQMC will look up the queue name against the regular expression pattern in order to determine if the queue name matches any of the specified expression.

In the regular expression pattern:

- '*' matches any sequence of characters (zero or more)
- '?' matches any single character
- '#' matches any single numeric digit (0-9)
- '@' matches any single alphabetic character (A-Z, a-z)
- [SET] matches any of the characters in the specified set
- [!SET] or [^SET] matches any character except those specified in the set (negation).

A SET can be composed of characters or ranges. A range is in the form: 'character – character' (i.e. 0-9 or A-Z). Although this is the simplest range allowed in the [ ] pattern, more complex inclusive ranges such as [0-9a-zA-Z] are allowed. [0-9a-zA-Z] specifies that the character can be 0 through 9 **or** a through z **or** A through Z. Other characters are allowed (ie. 8 bit characters) if your system supports them.

In order to suppress the special syntactic significance of any of these characters '[] * ? # @ ! ^ - \', a backslash ('\') must precede the special character.

```
[Q:TEST.HR*]
```

Warning: Do not select the 'SYSTEM.*' queues to be compressed. If you ever wish to deactivate MQMC or remove MQMC then it will be impossible, as the system message data will be compressed. Hence, if you do attempt this, then the only way to recover is to delete and re-create the queue manager.

5.3.1 Compression

Compression indicates which functionality that MQMC will perform. Compression supports 7 values (LZ4, LZW, LZMA_FAST, LZMA_BEST, RLE, ZLIB_FAST & ZLIB_BEST). The default value is LZ4.

- ***LZ4*** is a lossless data compression algorithm that is focused on compression and decompression speed. It belongs to the LZ77 family of byte-oriented compression schemes. LZ4 algorithm is incredibly fast.
- ***LZW*** (Lempel-Ziv-Welch) encodes sequences of 8-bit data as fixed-length 12-bit codes. I used Michael Dipperstein's implementation of LZW (Lempel-Ziv-Welch).
- ***LZMA_FAST*** uses the LZMA SDK from 7-Zip with a Level set to 4.
- ***LZMA_BEST*** uses the LZMA SDK from 7-Zip with a Level set to 5.
- ***RLE*** (Run Length Encoding) encodes sequences of the same data value occurring over many consecutive data elements are stored as a single data value and count.
- ***ZLIB_FAST*** uses zlib with a Level of *Z_BEST_SPEED*.
- ***ZLIB_BEST*** uses zlib with a Level of *Z_BEST_COMPRESSION*.

Note: LZMA_Fast and LZMA_Best are currently not available for IBM i.

```
Compression=LZMA_FAST
```

5.4 Section Name for a Topic

The section stanza must begin in column 1 with a left square bracket '[' followed by a “T:” and end with a right square bracket ']'. Between the colon and the right square bracket, the user can explicitly specify a topic name or a regular expression pattern.

The MQMC will look up the topic string against the regular expression pattern in order to determine if the topic string matches any of the specified expression.

In the regular expression pattern:

- '*' matches any sequence of characters (zero or more)
- '?' matches any single character
- '#' matches any single numeric digit (0-9)
- '@' matches any single alphabetic character (A-Z, a-z)
- [SET] matches any of the characters in the specified set
- [!SET] or [^SET] matches any character except those specified in the set (negation).

A SET can be composed of characters or ranges. A range is in the form: 'character – character' (i.e. 0-9 or A-Z). Although this is the simplest range allowed in the [ ] pattern, more complex inclusive ranges such as [0-9a-zA-Z] are allowed. [0-9a-zA-Z] specifies that the character can be 0 through 9 **or** a through z **or** A through Z. Other characters are allowed (ie. 8 bit characters) if your system supports them.

In order to suppress the special syntactic significance of any of these characters '[] * ? # @ ! ^ - \', a backslash ('\') must precede the special character.

```
[T:test/abc*]
```

Warning: Do not select the '$SYS/*' topics to be compressed.

5.4.1 Compression

Compression indicates which functionality that MQMC will perform. Compression supports 7 values (LZ4, LZW, LZMA_FAST, LZMA_BEST, RLE, ZLIB_FAST & ZLIB_BEST). The default value is LZ4.

- ***LZ4*** is a lossless data compression algorithm that is focused on compression and decompression speed. It belongs to the LZ77 family of byte-oriented compression schemes. LZ4 algorithm is incredibly fast.
- ***LZW*** (Lempel-Ziv-Welch) encodes sequences of 8-bit data as fixed-length 12-bit codes. I used Michael Dipperstein's implementation of LZW (Lempel-Ziv-Welch).
- ***LZMA_FAST*** uses the LZMA SDK from 7-Zip with a Level set to 4.
- ***LZMA_BEST*** uses the LZMA SDK from 7-Zip with a Level set to 5.
- ***RLE*** (Run Length Encoding) encodes sequences of the same data value occurring over many consecutive data elements are stored as a single data value and count.
- ***ZLIB_FAST*** uses zlib with a Level of *Z_BEST_SPEED*.
- ***ZLIB_BEST*** uses zlib with a Level of *Z_BEST_COMPRESSION*.

Note: LZMA_Fast and LZMA_Best are currently not available for IBM i.

```
Compression=LZMA_FAST
```

5.5 Example

5.5.1 Queue Example

If the queue name is called “TEST.Q01” then the user needs to prefix “Q:”, so that the stanza looks like: [Q:TEST.Q01]. Here is an IniFile example with 4 different groups of queues:

```
[Q:TEST.Q01]
Compression=LZMA_FAST

[Q:TEST.HR*]
Compression=LZMA_BEST

[Q:TEST.PAY*]
Compression=ZLIB_FAST

[Q:TEST.ABC*]
Compression=LZ4
```

5.5.2 Topic Example

If the topic string is called “test/abc” then the user needs to prefix “T:”, so that the stanza looks like: [T:test/abc]. Here is an IniFile example with 4 different groups of queues:

```
[T:test/abc]
Compression=LZMA_FAST

[T:test/abc/*]
Compression=LZMA_BEST

[T:test/abcx/HR*]
Compression=LZ4

[T:test/xyz*]
Compression=ZLIB_FAST
```

6 Appendix A – Summary of IniFile

The sample IniFile below is the mqmc.ini file supplied for Windows.

```
[default]
Active=Y
LogMode=N
LogFile=C:\Capitalware\MQMC\mqmc.log

[Q:TEST.HR*]
Compression=LZMA_FAST

[T:test/abc*]
Compression=LZ4
```

Note: Keywords are case sensitive.

The IniFile supports the following keywords and their values:

Keyword	Description of Server-side keywords
Active	**Active** specifies if MQMC is enabled or disabled. Active supports 2 values [Y / N]. The default value is Y. e.g. Active=N
BackupLogFileCount	**BackupLogFileCount** specifies the number of backup logfiles that MQMC will be keeping. The default value is 9. e.g. BackupLogFileCount=9
Compression	**Compression** specifies the type of data compress to be used. Compression supports 7 values [LZ4, LZW, LZMA_FAST, LZMA_BEST, RLE, ZLIB_FAST or ZLIB_BEST]. The default value is LZ4. Note: LZMA_Fast and LZMA_Best are currently not available for IBM i. e.g. Compression=LZMA_FAST

Keyword	Description of Server-side keywords
CompressRFH2Header	**CompressRFH2Header** specifies that the MQRFH2 header fields and folders are to be compressed when the message payload is compressed. CompressRFH2Header supports 2 values [Y / N]. The default value is Y. e.g. **CompressRFH2Header**=Y
DebugUserID	**DebugUserID** specifies a list of UserId which should have debug logging turned on for. e.g. **DebugUserID**=fred;barney
ExcludeApplications	**ExcludeApplications** specifies which particular applications should not be included. e.g. **ExcludeApplications**=test*;gadget Note: Only used if **UseExcludeApplications** is set to 'Y'.
ExcludeQueues	**ExcludeQueues** specifies which particular queues should not be included. e.g. ExcludeQueues=TEST.Q01 Note: Only used if UseExcludeQueues is set to 'Y'.
ExcludeTopics	**ExcludeTopics** specifies which particular topics should not be included. e.g. ExcludeTopics=test/abc;test/abc/s* Note: Only used if UseExcludeTopics is set to 'Y'.
ExcludeUserIDs	**ExcludeUserIDs** specifies which particular UserIds should not be included. e.g. **ExcludeUserIDs**=fred;barney Note: Only used if **UseExcludeUserIDs** is set to 'Y'.

Keyword	Description of Server-side keywords
IniFileRecheckTime	**IniFileRecheckTime** specifies the amount, in seconds, before the IniFile is checked whether it has changed or not. The default value is 60. e.g. **IniFileRecheckTime**=60
License	**License** specifies the queue manager's license key. Your license key will look something like: 10S0-AAAA-BBBBBBBB (Note: This is a sample license only and will NOT work). e.g. License=10M0-AAAA-BBBBBBB
LicenseFile	**LicenseFile** specifies the location of License file that contains all of the customer's license keys. The default values for LicenseFile are as follows: For Windows: LicenseFile=C:\Capitalware\MQMC\mqmc_licenses.ini For IBM MQ 32-bit on Unix and Linux: LicenseFile=/var/mqm/exits/mqmc_licenses.ini For IBM MQ 64-bit on Unix and Linux: LicenseFile=/var/mqm/exits64/mqmc_licenses.ini For IBM MQ on IBM i: LicenseFile=/QIBM/UserData/mqm/mqmc/mqmc_licenses.ini e.g. LicenseFile=/var/mqm/exits64/mqmc_licenses.ini
LogFile	**LogFile** specifies the location of the log file. The defaults are as follows: For Windows: LogFile=C:\Capitalware\MQMC\mqmc.log For IBM MQ 32-bit on Unix and Linux: LogFile=/var/mqm/mqmc/mqmc.log For IBM MQ 64-bit on Unix and Linux: LogFile=/var/mqm/mqmc/mqmc.log For IBM MQ on IBM i: LogFile=/QIBM/UserData/mqm/mqmc/mqmc.log

Keyword	Description of Server-side keywords
LogMode	**LogMode** specifies what type of logging the user wishes to have. LogMode supports 4 values [Q / N / V / D] where Q is Quiet, N is Normal, V is Verbose and D is Debug. The default value is N. e.g. LogMode=N
RotateLogDaily	**RotateLogDaily** specifies whether or not daily log file rotation should take place. RotateLogDaily supports 2 values [Y / N]. The default value is Y. e.g. RotateLogDaily=Y
UseExcludeApplications	**UseExcludeApplications** specifies whether or not ExcludeApplications should be activated. **UseExcludeApplications** supports 2 values [Y / N]. The default value is N. e.g. **UseExcludeApplications**=Y
UseExcludeQueues	**UseExcludeQueues** specifies whether or not ExcludeQueues should be activated. UseExcludeQueues supports 2 values [Y / N]. The default value is N. e.g. UseExcludeQueues=Y
UseExcludeTopics	**UseExcludeTopics** specifies whether or not ExcludeTopics should be activated. UseExcludeTopics supports 2 values [Y / N]. The default value is N. e.g. UseExcludeTopics=Y
UseExcludeUserIDs	**UseExcludeUserIDs** specifies whether or not ExcludeQueues should be activated. **UseExcludeUserIDs** supports 2 values [Y / N]. The default value is N. e.g. **UseExcludeUserIDs**=Y
UserIDFormatting	**UserIDFormatting** specifies how MQMC will handle the incoming UserID. UserIDFormatting supports 3 values [A / U / L]. ('As Is, Uppercase and Lowercase). The default value is A. UserIDFormatting=U

7 Appendix B – MQMC Upgrade Procedures

To upgrade an existing installation of MQMC from an older version to a newer version, please do the following:

7.1.1 Windows Upgrade

- Stop all MQ applications connecting to the queue manager including monitoring tools
- Stop the queue manager using the MQMC API Exit
- Backup all MQMC IniFiles in the MQMC install directory
- If MQMC was installed using the Windows Installer then
    - Click the ***Start*** -> ***All Programs*** -> ***Control Panel*** -> ***Add or Remove Programs***, select MQMC from the list and click the ***Remove*** button then follow the prompts to remove it
    - Run the **mqmc-setup.exe** file from the **Windows** directory to install the new version
- Otherwise, copy the following files (latest version) to the MQMC install directory:
    - mqmc.dll
    - rotatelog.bat
- Restore the MQMC IniFiles if they were altered/deleted.
- Restart the queue manager
- Restart your MQ applications and any monitoring tools

7.1.2 Linux 32-bit Upgrade

- Log in under the mqm account
- Stop all MQ applications connecting to the queue manager including monitoring tools
- Stop the queue manager using the MQMC API Exit
- Backup all MQMC IniFiles in the MQMC install directory
- Copy the appropriate tar file to the ***/var/mqm/*** directory
- Un-tar the contents of the tar file.

  e.g. For Linux, do the following command:

  ```
  tar -xvf  mqmc_linux_x86.tar
  ```
- Run the script as follows:

  ```
  ./setmqmc.sh
  ```
- Restore the MQMC IniFiles if they were altered / deleted.
- Delete the MQMC tar file
- Restart the queue manager
- Restart your MQ applications and any monitoring tools

7.1.3 Unix and Linux 64-bit Upgrade

- Stop all MQ applications connecting to the queue manager including monitoring tools
- Stop the queue manager using the MQMC API Exit
- Backup all MQMC IniFiles in the MQMC install directory
- Copy the appropriate tar file to the ***/var/mqm/*** directory
- Un-tar the contents of the tar file.
  e.g. For AIX 7.1 (or higher), do the following command:
  ```
    tar -xvf  mqmc_aix71_64.tar
  ```
- Run the script as follows:
  ```
   ./setmqmc.sh
  ```
- Restore the MQMC IniFiles if they were altered / deleted.
- Delete the MQMC tar file
- Restart the queue manager
- Restart your MQ applications and any monitoring tools

7.1.4 IBM i Upgrade

- Stop all MQ applications connecting to the queue manager including monitoring tools
- Stop the queue manager using the MQMC API Exit
- Backup all MQMC IniFiles in the MQMC install directory
- ftp the IBM i files to the IBM i server using the following command:

```
ftp -s:mqmc_iseries.ftp  iseries_hostname
```

```
your-IBM i-userid
your-IBM i-password

binary
cd QGPL
put mqmc.savf
quote SITE NAMEFMT 1
quit
```

- Log onto the target IBM i server and do the following commands:

```
RSTLIB SAVLIB(MQMC) DEV(*SAVF) SAVF(QGPL/MQMC)
CLRSAVF FILE(QGPL/MQMC)
CHGOBJOWN  OBJ(MQMC) OBJTYPE(*LIB)  NEWOWN(QMQM)
```

- Restore the MQMC IniFiles if they were altered / deleted.
- Restart the queue manager
- Restart your MQ applications and any monitoring tools

8 Appendix C – Test Compression - TESTCMPRSN

TESTCMPRSN programs is a simple program that applies all 8 lossless compression algorithms against a particular file.

Syntax:

```
testcmprsn  sample_file
```

8.1 Examples

8.1.1 Windows

To use the ***testcmprsn*** program on Windows, open a Command prompt and change the directory to `C:\Capitalware\MQMC\`

```
C:\Capitalware\MQMC>testcmprsn.exe  medsamp2011.xml
testcmprsn version 1.0.0 (Windows64) {Sep 10 2020}

medsamp2011.xml size is 1109486 (1.06MB)
Time taken to perform memcpy() is 0.3505ms

Algorithm               Compressed      Compression     Compression     Decompression
                           Size         Time in ms        Ratio          Time in ms
LZ4                   229095 (223.73KB)     5.0838       4.84 to 1          0.9395
LZMA Fast             136248 (133.05KB)    35.6909       8.14 to 1         12.2154
LZMA Best             128025 (125.02KB)   202.1005       8.67 to 1         12.0546
LZW                   214528 (209.50KB)    40.7706       5.17 to 1          8.6771
RLE                  1722146 (1.64MB)       3.2898       0.64 to 1          3.8062
ZLIB Fast             223630 (218.39KB)    10.0127       4.96 to 1          4.5992
ZLIB Best             144052 (140.68KB)    50.1845       7.70 to 1          3.5468
testcmprsn is ending.
```

8.1.2 Linux 32-bit

To use the ***testcmprsn*** program on Linux for MQ 32-bit, open a shell prompt and change directory to `/var/mqm/exits/`

```
mqm@abc123:~/var/mqm/exits> ./testcmprsn medsamp2011.xml
testcmprsn version 0.0.1 (Linux64) {Sep 10 2020}

medsamp2011.xml size is 1084741 (1.03MB)
Time taken to perform memcpy() is 0.7630ms

Algorithm                 Compressed      Compression     Compression     Decompression
                             Size         Time in ms        Ratio           Time in ms
LZ4                   228660 (223.30KB)       3.3140      4.74 to 1           0.8430
LZMA Fast             135904 (132.72KB)      32.2350      7.98 to 1           9.9810
LZMA Best             127664 (124.67KB)     181.3220      8.50 to 1           9.9300
LZW                   213463 (208.46KB)      35.6150      5.08 to 1          10.8840
RLE                  1672656 (1.60MB)         3.0150      0.65 to 1           2.8480
ZLIB Fast             223219 (217.99KB)       7.9380      4.86 to 1           4.0070
ZLIB Best             143465 (140.10KB)      50.1380      7.56 to 1           2.8460
testcmprsn is ending.
```

8.1.3 Unix or Linux 64-bit

To use the ***testcmprsn*** program on Unix/Linux for MQ 64-bit, open a shell prompt and change directory to `/var/mqm/exits64/`

```
mqm@abc123:~/var/mqm/exits64> ./testcmprsn medsamp2011.xml
testcmprsn version 0.0.1 (Linux64) {Sep 10 2020}

medsamp2011.xml size is 1084741 (1.03MB)
Time taken to perform memcpy() is 0.7630ms

Algorithm                 Compressed      Compression     Compression     Decompression
                             Size         Time in ms        Ratio           Time in ms
LZ4                   228660 (223.30KB)       3.3140      4.74 to 1           0.8430
LZMA Fast             135904 (132.72KB)      32.2350      7.98 to 1           9.9810
LZMA Best             127664 (124.67KB)     181.3220      8.50 to 1           9.9300
LZW                   213463 (208.46KB)      35.6150      5.08 to 1          10.8840
RLE                  1672656 (1.60MB)         3.0150      0.65 to 1           2.8480
ZLIB Fast             223219 (217.99KB)       7.9380      4.86 to 1           4.0070
ZLIB Best             143465 (140.10KB)      50.1380      7.56 to 1           2.8460
testcmprsn is ending.
```

8.1.4 IBM i

To use the ***testcmprsn*** program on IBM i for MQ, issue the command on Command prompt.

```
CALL MQMC/TESTCMPRSN PARM('/QIBM/UserData/mqm/mqmc/testdata.xml')

testcmprsn version 0.0.1 (IBM i) {Sep 10 2020}

medsamp2011.xml size is 1084741 (1.03MB)
Time taken to perform memcpy() is 0.7630ms

Algorithm               Compressed      Compression     Compression     Decompression
                           Size         Time in ms        Ratio           Time in ms
LZ4                 228660 (223.30KB)       3.3140       4.74 to 1          0.8430
LZW                 213463 (208.46KB)      35.6150       5.08 to 1         10.8840
RLE                1672656 (1.60MB)         3.0150       0.65 to 1          2.8480
ZLIB Fast           223219 (217.99KB)       7.9380       4.86 to 1          4.0070
ZLIB Best           143465 (140.10KB)      50.1380       7.56 to 1          2.8460
testcmprsn is ending.
```

9 Appendix D – Lossless Compression

MQ Message Compression solution uses the lossless compression algorithms to compress the message data while it resides in a queue (i.e. data at rest), hence, reducing the size of the message data.

Wikipedia

> *Lossless compression is a class of data compression algorithms that allows the original data to be perfectly reconstructed from the compressed data. By contrast, lossy compression permits reconstruction only of an approximation of the original data, though usually with greatly improved compression rates (and therefore reduced media sizes).*

10 Appendix E – Capitalware Product Display Version

MQMC includes a program to display the product version number. The command to display the product version number is:

`cwdspver`

10.1 Examples

10.1.1 Windows

To use the cwdspver program on Windows, open a Command prompt and change the directory to `C:\Capitalware\MQMC\` and type the following:

```
cwdspver.exe
```

10.1.2 Unix and Linux 32-bit

To use the cwdspver program on Unix/Linux for MQ 32-bit, open a shell prompt and change directory to `/var/mqm/exits/` and type the following:

```
./cwdspver
```

10.1.3 Unix and Linux 64-bit

To use the cwdspver program on Unix/Linux for MQ 64-bit, open a shell prompt and change directory to `/var/mqm/exits64/` and type the following:

```
./cwdspver
```

10.1.4 IBM i

To use the cwdspver program on IBM i, issue the following command on the Command Prompt:

```
CALL MQMC/CWDSPVER
```

11 Appendix F – Support

The support for MQ Message Compression can be found at the following location:

By email at:
support@capitalware.com

By regular mail at:

Capitalware Inc.
Attn: MQMC Support
Unit 11, 1673 Richmond Street, PMB524
London, Ontario N6G2N3
Canada

12 Appendix G – Summary of Changes

- MQ Message Compression v1.0.0
    - Initial release.

13 Appendix H – License Agreement

This is a legal agreement between you (either an individual or an entity) and Capitalware Inc. By opening the sealed software packages (if appropriate) and/or by using the SOFTWARE, you agree to be bound by the terms of this Agreement. If you do not agree to the terms of this Agreement, promptly return the disk package and accompanying items for a full refund.
SOFTWARE LICENSE

1. GRANT OF LICENSE. This License Agreement (License) permits you to use one copy of the software product identified above, which may include user documentation provided in on-line or electronic form (SOFTWARE). The SOFTWARE is licensed as a single product, to an individual queue manager, or group of queue managers for an Enterprise License. This Agreement requires that each queue manager of the SOFTWARE be Licensed, either individually, or as part of a group. Each queue manager's use of this SOFTWARE must be covered either individually, or as part of an Enterprise License. The SOFTWARE is in use on a computer when it is loaded into the temporary memory (i.e. RAM) or installed into the permanent memory (e.g. hard disk) of that computer. This software may be installed on a network provided that appropriate restrictions are in place limiting the use to registered queue managers only. Each licensed queue manager will be provided with a perpetual license key and the licensee may continue to use the SOFTWARE, so long as the licensee is current on the Yearly Maintenance Fee. If the licensee stops paying the Yearly Maintenance Fee, then the SOFTWARE must be removed from all systems at the end of the current maintenance period.

2. COPYRIGHT. The SOFTWARE is owned by Capitalware Inc. and is protected by United States Of America and Canada copyright laws and international treaty provisions. You may not copy the printed materials accompanying the SOFTWARE (if any), nor print copies of any user documentation provided in on-line or electronic form. You must not redistribute the registration codes provided, either on paper, electronically, or as stored in the files mqmc.ini, mqmc_licenses.ini or any other form.

3. OTHER RESTRICTIONS. The registration notification provided, showing your authorization code and this License is your proof of license to exercise the rights granted herein and must be retained by you. You may not rent or lease the SOFTWARE, but you may transfer your rights under this License on a permanent basis, provided you transfer this License, the SOFTWARE and all accompanying printed materials, retain no copies, and the recipient agrees to the terms of this License. You may not reverse engineer, decompile, or disassemble the SOFTWARE, except to the extent the foregoing restriction is expressly prohibited by applicable law.

LIMITED WARRANTY

LIMITED WARRANTY. Capitalware Inc. warrants that the SOFTWARE will perform substantially in accordance with the accompanying printed material (if any) and on-line documentation for a period of 365 days from the date of receipt.

CUSTOMER REMEDIES. Capitalware Inc. entire liability and your exclusive remedy shall be, at Capitalware Inc. option, either (a) return of the price paid or (b) repair or replacement of the SOFTWARE that does not meet this Limited Warranty and that is returned to Capitalware Inc.

with a copy of your receipt. This Limited Warranty is void if failure of the SOFTWARE has resulted from accident, abuse, or misapplication. Any replacement SOFTWARE will be warranted for the remainder of the original warranty period or thirty (30) days, whichever is longer.

NO OTHER WARRANTIES. To the maximum extent permitted by applicable law, Capitalware Inc. disclaims all other warranties, either express or implied, including but not limited to implied warranties of merchantability and fitness for a particular purpose, with respect to the SOFTWARE and any accompanying written materials.

NO LIABILITY FOR CONSEQUENTIAL DAMAGES. To the maximum extent permitted by applicable law, in no event shall Capitalware Inc. be liable for any damages whatsoever (including, without limitation, damages for loss of business profits, business interruption, loss of business information, or other pecuniary loss) arising out of the use or inability to use the SOFTWARE, even if Capitalware Inc. has been advised of the possibility of such damages.

14 Appendix I – Notices

Trademarks:

AIX, IBM, MQSeries, OS/2 Warp, OS/400, iSeries, MVS, OS/390, WebSphere, IBM MQ and z/OS are trademarks of International Business Machines Corporation.

HP-UX is a trademark of Hewlett-Packard Company.

Intel is a registered trademark of Intel Corporation.

Java, J2SE, J2EE, Sun and Solaris are trademarks of Sun Microsystems Inc.

Linux is a trademark of Linus Torvalds.

Mac OS X is a trademark of Apple Computer Inc.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation.

UNIX is a registered trademark of the Open Group.

WebLogic is a trademark of BEA Systems Inc.