



MQEV

Process IBM® MQ Events, Accounting and Statistics messages

MQEV can receive these messages, store them, and allow you to search, filter, aggregate, and choose how long to retain the information within the messages.

MQEV is a program which will receive; store and process three types of IBM MQ messages.

MQ Event Messages

Event messages are IBM MQ's way of telling the installation that something 'of note' has happened. This could range from something fairly innocuous such as channel starting to something fairly serious such as a queue filling up.

MQ Statistics Messages

Statistics messages are messages which the MQ Queue Manager will generate on a regular interval to notify the user of the levels of activity. For example, how many messages have been put or got to a queue.

MQ Accounting Messages

Accounting messages are fairly similar to statistics messages however they are from the point of view of the application. So, it gives information such as how many MQI calls, and which type, have been issued by each application in the system. Again these messages output at regular intervals.

Use the data that the IBM MQ queue manager emits, that so many MQ estates don't even turn on and look at, to answer some of the following questions.

- Did anything unusual happen with my Queue Manager over the weekend?
- Is queue or channel XYZ still being used? ...and, if so, by who/what?
- What types of MQ clients are using my Queue Manager? JMS? C? Java? C++?
- What channels are being used and what is the relative split of persistent message traffic?
- How efficient are my channels? What is the average batch size in use over time?
- How does the activity/depth of queue X vary over the day? What are my peak traffic times?
- How does the activity/depth of queue X vary over the last 3 months? Am I seeing an increase of traffic? Will my MQ infrastructure be able to cope with future demands?
- Are any of my applications misbehaving?
- How is my processing speed varying over time? Am I delivering on my Service Level Agreements (SLA)?
- What is the maximum depth my queue has reached during a particular period?
- and many more.....

Highlights

- Runs in either client or local binding
- Does not require complicated and expensive Databases
- An easy way to store a history of events, statistics and accounting information
- Simple configuration of how long to keep the data for
- Fast search and display of all stored data
- Consolidation and summarising of change events
- Fast totalling of statistics and accounting data
- Easily write scripts to process and action events as they are issued
- Easily write scripts to mine the events and statistics data
- Administration and display of MQEV data either by command line (MQSCX) or a GUI (MO71) - no extra licences required

Event Id	Event Queue Manager	Stream	Event Time	Summary	Event Reason
00000100	MQG1	SEVENTS	30th Dec 2019 23:07:34	Channel - Max Inst Client - Channel:INSTS.SVRCONN	Channel - Max Inst Client
000000FF	MQG1	SEVENTS	30th Dec 2019 23:07:34	Channel - Max Instances - Channel:INSTS.SVRCONN	Channel - Max Instances
000000FC	MQG1	SEVENTS	30th Dec 2019 23:07:34	Channel - Blocked No Access - Channel:SSL.SVRCONN	Channel - Blocked No Access
000000FB	MQG1	SEVENTS	30th Dec 2019 23:07:34	SSL/TLS Peer Name Error - Channel:SSL.SVRCONN	SSL/TLS Peer Name Error
000000FA	MQG1	SEVENTS	30th Dec 2019 23:07:34	SSL/TLS Cipher Spec Error - Channel:SSL.SVRCONN	SSL/TLS Cipher Spec Error
000000F8	MQG1	SEVENTS	30th Dec 2019 23:07:34	SSL/TLS Client Auth Error - Channel:SSL.SVRCONN	SSL/TLS Client Auth Error

Display Filtering

With their powerful display syntax, MQEV commands can allow you to display the data collected and filter it in numerous ways to discover the information within. Find the correlations between different types of events easily, for example, changes made to a channel definition in a configuration event resulting in a problem next time the channel tries to start, reported in a channel event. MQEV ensures that regardless of the type of event message, key fields such as the object name can always be found across the different event types.

Reduce the time frame down to a specific date and time to find out what happened, or aggregate your accounting and statistics data to compare numbers from one week to the next. All of this can be achieved with MQEV display commands.

Use your favourite tools, MO71 for a GUI display of the data, or MQSCX for those who prefer a command line. There is also a publically documented PCF interface if you prefer to write your own tools.

Data Retention

Different data will have different requirements for retention. Some events will need to be kept for longer than others. This is something that is difficult to do if you just turn on MQ Events and leave them to collect on the standard MQ event queues. There is an 'all-or-nothing' retention policy there.

MQEV allows you to store events, accounting and statistics messages on streams (an MQEV administrative aid), where different retention intervals can be configured. MQEV will automatically dispose of the data once its retention interval has passed.

By default MQEV is set up to retain events for 90 days and accounting and statistics data for 45 days. However, this is completely configurable by the user, including having different intervals for different event categories - of your own choosing.

Alerts

Alerts can be generated from MQEV for any situation that is interesting to you. These can be generated in relation to an event or data from accounting and statistics, or simply as a manual task to remind you of something.

Examples might include any authority event seen; higher application usage numbers than a certain threshold, or a reminder to install the latest version of IBM MQ.

Alerts can be removed when the situation is resolved, either through another script (for example removing an alert on a queue high event when queue low is seen), or manually, say when the task is done. Notifications can also be received when alerts still exist after a certain time period, so action can be taken, say if a queue high situation remains for over 10 minutes.

Event Storms

Event storms are occasions when MQEV is deluged with lots of very similar events. This can happen for a variety of reasons but perhaps the simplest to consider is a rogue application. Consider the case where you have a badly written application which tries to connect to a Queue Manager and, if it fails, immediately tries again. This can cause a huge number of authority events to be raised.

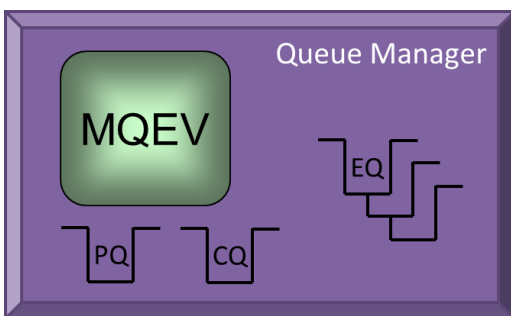
By default, MQEV will consider the same event seen 20 times in a minute as an event storm, and will raise an alert as well as not storing any more copies of the identical event data. You can configure the count and interval to suit your own requirements.

Topologies

MQEV uses a couple of queues in order to operate. A command queue to receive inquiries from tools such as MO71 and MQSCX; and a persistence queue where compressed event, accounting and statistics data is held once received from the IBM MQ event queues. The program itself can run as a client or local bindings connection. This provides a number of choices for the topology arrangement that you can use with MQEV.

All queues local to MQEV

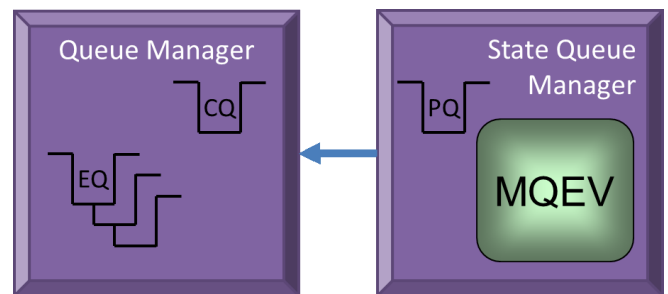
In this topology, MQEV runs as a locally bound application on the same machine as the queue manager being monitored. The event queues and the persistence queue are hosted on the same queue manager, and commands are sent to MQEV via the command queue also on that queue manager.



Separate state QMgr for the persistence queues

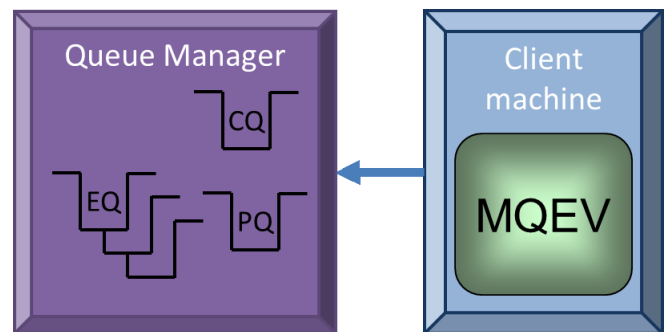
There may be times when running MQEV local to the monitored queue manager is not appropriate, or even possible. For example, MQEV is not currently available as a z/OS program, and it clearly can't run locally on the MQ Appliance. Additionally, installations may have rules about what programs can run local to a queue manager.

In such cases, MQEV can connect as a client to the monitored queue manager, but maintain its persistent state on a local queue manager, ideally one dedicated to run your MQEV instances. This state queue manager holds the persistence queues (PQ), one for each monitored queue manager, and the event queues (EQ) are read by MQEV over a client connection. Commands are sent to MQEV via the command queue (CQ) hosted on each monitored queue manager as before, also read by MQEV over the client connection. Administrative tools such as MQSCX and MO71 connect to the queue managers just as they do for the IBM MQ command server. They do not connect to the MQEV state queue manager.¹



All queues read over a client connection

As a third option, it is possible to run MQEV to read all the queues, event queues, the command and the persistence queue via a client connection to the queue manager.



Persistence Queue

MQEV reads the event, accounting or statistics messages off the well-known MQ queues, and processes them. Part of the processing is to compress the data, and then store it on a separate persistence queue, thereby removing the complication of a database. This means that the data is kept within the MQ world, and can be consumed transactionally using

¹ Picture Key: CQ = Command Queue
EQ = Event Queue(s)
PQ = Persistence Queue

just a local transaction with no need to employ a global transaction coordinator.

Other advantages include simplicity in HA and Disaster recovery scenarios - having all the state on an MQ Queue in the same 'domain' as the event queues themselves means that all the event data can fail over together without the need to synchronise with a database.

Daisy Chaining

If you have need to keep the original event messages, or have another program read them as well, MQEV will participate in daisy chaining as all good event processing applications should, and can be configured with a forwarding queue, and/or read from a different named queue as required.

```
MQ Environment : 913GA - mqscx -m MQG1
Connected to 'MQG1'
[23:17:48] =mqev
Commands directed to MQEV
[23:17:51] DISPLAY EVENTS(*) EVQMGR(MQG1) EVOBJNAME(*.SVRCONN) MAXRESP(4)

EVQMGR(MQG1)          EVTIME(2019-12-30 23:07:34 (Local))
SUMMARY(Channel - Max Inst Client - Channel:INSTS.SVRCONN)

EVQMGR(MQG1)          EVTIME(2019-12-30 23:07:34 (Local))
SUMMARY(Channel - Max Instances - Channel:INSTS.SVRCONN)

EVQMGR(MQG1)          EVTIME(2019-12-30 23:07:34 (Local))
SUMMARY(Channel - Blocked No Access - Channel:SSL.SVRCONN)

EVQMGR(MQG1)          EVTIME(2019-12-30 23:07:34 (Local))
SUMMARY(SSL/TLS Peer Name Error - Channel:SSL.SVRCONN)

Total display responses - Received:4
MQEV:MQG1>
```

Scripting

Those familiar with the MQSCX product from MQGem will feel right at home creating small scripts to direct MQEV how to store (or not to store) the data collected. Processing and notification of specific situations can be used to create alerts, allowing full configuration of the behaviour you need from your events.

Notifications of interesting situations can be done at the moment the event arrives, for example sending an SMS text or email to the relevant person, or raising an alert as mentioned already. If you can write it as a command script, you can invoke it from MQEV. You are limited only by your imagination.

In the same way that MQSCX can be used to create reports about your IBM MQ estate; it can also be used to report on the data collected by MQEV. For example, you could create a weekly report on the use of MQ commands that are considered to have an adverse effect; e.g. how often are we using REFRESH CLUSTER during the day?

For more information

To learn more about how MQEV can help you process IBM MQ Events, Accounting and Statistics messages, contact us by email on support@mqgem.com, or visit:

www.mqgem.com/mqev.html

Try it out for free with a no-obligation, one month trial licence. Email support@mqgem.com for yours.



© Copyright MQGem Software Ltd 2019, 2020

Email support@mqgem.com
Web www.mqgem.com
Blog mqgem.wordpress.com
Twitter @MQGem

IBM and z/OS are trademarks of International Business Machines Corporation in the United States, other countries or both.

Other company, product and service names may be trademarks or service marks of others.