MQ Pub/Sub: basic troubleshooting

http://www.ibm.com/support/docview.wss?uid=ibm10739029

Angel Rivera (rivera@us.ibm.com)
IBM MQ Distributed Level 2 Support
Date last updated: 08-Nov-2018

Agenda

- Using runmqras

- Most common problems:
- 1: A userid who is not an MQ administrator gets security rc 2035 MQRC_NOT_AUTHORIZED and error AMQ8077 or AMQ8009W is logged in the error.
- 2: Publishing to a topic string but subscribing to a topic object that has a different topic string:
  - The subscriber is not receiving messages

- References to other documents.

Related presentations

This presentation is one of a series.
For the complete list, please see:

https://developer.ibm.com/answers/questions/402074/mq-pubsub-training-presentations.html
MQ Pub/Sub: training presentations

Related file

For illustration purposes, runmqras was obtained on a test Linux queue manager called “QMPS”.

It is provided in the attached file:

QMPS-runmqras-2-mosquito.zip

Best practice: runmqras options

The MQ tool “runmqras” is shipped with the product and it gathers a lot of information that can be helpful during troubleshooting:
FDCs, error logs, information about the host, dspmqver, etc.

For Pub/Sub there are 2 entries for the “section” parameter:

-section defs = It includes output from several DISPLAY commands

-section trace = It includes the trace files (if trace was generated)

-section topic = It includes Topic Tree output from amqldmpa. Only used if there is suspicion regarding the Topic Tree.

They can be combined, such as: **-section defs,trace,topic**

Best practice: runmqras, typical invocation

A typical way to invoke runmqras is as follows, which will send the output zip file to the IBM ftp server. The zip file name will have the PMR id.

The following should be entered in a single line, but due to space limits in this page, 2 lines are used:

runmqras -qmlist QMgrName -section defs,trace
-pmrno 12345,678,000 -ftp IBM

Notes: runmqras -section defs

notes

Use "-section defs,topic" to show Pub/Sub information (no Topic Tree)
runmqras -qmlist QMPS -section defs

Subdirectory: defs
Files: *.stdout

Showing only those files related to Pub/Sub for runmqsc (alphabetical order):

runmqsc_PUBSUB_QMPS.stdout
query: DIS PUBSUB ALL
Display pub/sub status details. (It does not really provide much info)

runmqsc_QLOCAL_QMPS.stdout
query: DIS QLOCAL(*) ALL
Display Queue details. (For PROVIDED local queues for durable Subs, PSMODE, PSCLUS)

Notes: runmqras -section defs

notes

runmqsc_QMGR_QMPS.stdout
query: DIS QMGR ALL
Display Queue Manager details. (PARENT for Hierarchies)

runmqsc_QMODEL_QMPS.stdout
query: DIS QMODEL(*) ALL
For attributes for SYSTEM.DURABLE.MODEL.QUEUE
SYSTEM.NDURABLE.MODEL.QUEUE

runmqsc_SBSTATUS_QMPS.stdout
query: DIS SBSTATUS(*) ALL
subscription status inquired.

runmqsc_SUB_QMPS.stdout
query: DIS SUB(*) ALL
subscription status inquired

Notes: runmqras -section defs

notes

runmqsc_TCLUSTER_QMPS.stdout
query: DIS TCLUSTER(*) ALL
Clustered topic objects

runmqsc_TOPIC_QMPS.stdout
query: DIS TOPIC(*) ALL
Topic objects

runmqsc_TPSTATUS_PUB_QMPS.stdout
query: DIS TPSTATUS('#') TYPE(PUB) ALL
Topic Status for publishers

runmqsc_TPSTATUS_SUB_QMPS.stdout
query: DIS TPSTATUS('#') TYPE(SUB) ALL
Topic Status for subscribers

runmqsc_TPSTATUS_TOPIC_QMPS.stdout
query: DIS TPSTATUS('#') TYPE(TOPIC) ALL
Status on the topic strings

Notes: runmqras -section defs,topic

notes

Use "-section defs,topic" to show Pub/Sub information + Topic Tree information
runmqras -qmlist QMPS -section defs,topic

-section defs
Subdirectory: defs
Files: *.stdout
See previous slide for the files:
runmqsc_*.stdout

-section topic
Subdirectory: topic
Files:
amqldmpa_topic_QMPS2.out => ouput of amqldmpa with the Topic Tree
amqldmpa_topic_QMPS2.stdout => how amqldmpa was invoked

Regarding how to interpret the output from amqldmpa, see the 2 referenced articles in the 2nd half of this presentation.

Authority problem

Authority Problems - AMQ8077 rc 2035

- A common problem is when a Publisher or a Subscriber get the generic authorization error:
- 2035 MQRC_NOT_AUTHORIZED
- Error AMQ8077 or AMQ8009W

- There will be an entry in the queue manager error log with more information, such as:
- AMQ8077: Entity ‘non-mqm ' has insufficient authority to access object ‘sales'.
- EXPLANATION: The specified entity is not authorized to access required object. The following requested permissions are unauthorized: pub

Non-admin user tries to publish/subscribe

- The host of the queue manager has user 'alice' who does not belong to group “mqm”.
- Alice belongs to group 'mqusers'.
- It is a non-administrator for MQ.
- alice@mosquito: /home/alice
- $ id alice
- uid=1008(alice) **gid=1005(mqusers)** groups=1005(mqusers)

- For comparison, here is the id info for “mqm”
- mqm@mosquito: /home/mqm
- $ id mqm
- uid=501(mqm) **gid=501(mqm)** groups=501(mqm)

Non-admin user tries to publish/subscribe

- User 'alice' tries to publish into topic string 'sales'.
- The attempts fails with reason code 2035 for the action MQCONN (MQ Connection)
- The MQ client code will not indicate the root cause.

- $ amqspub sales QMPS
- Sample AMQSPUBA start
- MQCONN ended with reason code 2035

- $ mqrc 2035
- 2035 0x000007f3 MQRC_NOT_AUTHORIZED

Details for 2035 in error log of queue mgr

- An MQ administrator (such as user ‘mqm’) will need to look at the recent entries of the error log for the queue manager to find out more details about the security failure.

- mqm@mosquito:
- $ cd /var/mqm/qmgrs/QMPS/errors
- $ tail -15 AMQERR01.LOG

Error message regarding lack of 'connect'

- Date stamp - Process(26420.12) User(mqm) Program(amqzlaa0)
- Host(mosquito) Installation(Installation5)
- VRMF(9.0.3.0) QMgr(QMPS)

- **AMQ8077: Entity 'alice' has insufficient authority to access object 'QMPS'.**

- EXPLANATION:
- The specified entity is not authorized to access the required object. The
- following requested permissions are unauthorized: **connect**

- ACTION:
- Ensure that the correct level of authority has been set for this entity against
- the required object, or ensure that the entity is a member of a privileged
- group.

Notes: Giving authority access

notes

The userids that are not members of the MQ administrator's group "mqm" by default do not have access to the queue manager, nor other objects that are necessary for the communication between the Explorer and the remote queue manager.
Therefore, it is necessary for an MQ administrator to grant certain authorities to these users to be allowed to access the queue manager.
.
The purpose of the following setmqaut commands is:

1. GENERAL: Grant authority to access the queue manager.
2. MQ EXPLORER: Grant authority to the client channel to get the command server reply messages.
3. MQ EXPLORER: Grant authority to put messages onto the command server input queue.
4. MQ EXPLORER: Grant authority to get the reply messages.

Notes: Giving authority access

notes

The queue manager name is MYQMGR.
The flag-value pair "t qmgr" refers to the object type of QueueManager.
The flag-value pair "t q" refers to the object type of Queue.

For a userid:
Issue these setmqaut commands to grant minimal authority to the userID “myuser”
(using the PrincipalName, or person, flag: -p)

```
setmqaut -m MYQMGR -t qmgr -p myuser +connect +inq +dsp
setmqaut -m MYQMGR -t q -n SYSTEM.DEFAULT.MODEL.QUEUE -p myuser +inq +browse +get +dsp
setmqaut -m MYQMGR -t q -n SYSTEM.ADMIN.COMMAND.QUEUE -p myuser +inq +put +dsp
setmqaut -m MYQMGR -t q -n SYSTEM.MQEXPLORER.REPLY.MODEL -p myuser +inq +browse +get +dsp
```

For a group:
Issue these setmqaut commands to grant minimal authority to the Unix group
(using the GroupName flag: -g)

```
setmqaut -m MYQMGR -t qmgr -g mygroup +connect +inq +dsp
setmqaut -m MYQMGR -t q -n SYSTEM.DEFAULT.MODEL.QUEUE -g mygroup +inq +browse +get +dsp
setmqaut -m MYQMGR -t q -n SYSTEM.ADMIN.COMMAND.QUEUE -g mygroup +inq +put +dsp
setmqaut -m MYQMGR -t q -n SYSTEM.MQEXPLORER.REPLY.MODEL -g mygroup +inq +browse +get +dsp
```

Non-admin user gets connect permissions

- The MQ Administrator issues the following commands to allow user alice to connect to the queue manager via line commands and MQ Explorer.

```
setmqaut -m QMPS -t qmgr -p alice +connect +inq +dsp
setmqaut -m QMPS -t q -n SYSTEM.DEFAULT.MODEL.QUEUE -p alice +inq +browse +get +dsp
setmqaut -m QMPS -t q -n SYSTEM.ADMIN.COMMAND.QUEUE -p alice +inq +put +dsp
setmqaut -m QMPS -t q -n SYSTEM.MQEXPLORER.REPLY.MODEL -p alice +inq +browse +get +dsp
```

- For more details see:
https://developer.ibm.com/answers/questions/229920/what-are-the-minimum-steps-to-create-a-new-test-qu.html
What are the minimum steps to create a new test queue manager, allowing remote access by MQ Administrators and non-administrators?

Non-admin user tries to publish/subscribe

- User 'alice' tries again.
- The attempts fails again with rc 2035.
- Notice that the rc is for MQOPEN (open a topic), not MQCONN.

- $ amqspub sales QMPS
- Sample AMQSPUBA start
- target topic is sales
- MQOPEN ended with reason code 2035
- unable to open topic for publish
- Sample AMQSPUBA end

Non-admin user tries to publish/subscribe

- Again, the MQ admin needs to look at the bottom of the error log of the queue manager.
- Notice that now the permission is for “pub” (previously it was for “connect).

- AMQ8009: Entity 'alice' has insufficient authority to access **topic string 'sales'.**
- EXPLANATION:
- The specified entity is not authorized to access the required topic.
- The following permissions were requested: **pub**

Specify authority on durable Topic Object

- MQ Administrators have all the authority to publish and to subscribe.
- But non MQ Administrators need to get authority to connect to the queue manager and to be able to publish and/or subscribe.

- It is NOT possible to specify authority permissions on **Topic Strings.**

- It is necessary to specify the authority on
- **a Topic Object (which is a durable topic)**

Authority information from runmqras

- The MQ **runmqras** utility always includes the information about authorities in 2 files:
  - amqoamd_QMPS.stdout
  - dmpmqaut_QMPS.stderr (note that dmpmqaut_QMPS.stdout is always empty!)

- Best Practice:
- If you specify **"-section defs"** then you will get a 3rd file with information, which is easier to interpet:
  - defs/runmqsc_AUTHREC_QMPS.stdout

Notes: Default set of durable topic objects

notes

display topic(*)

```
AMQ8633I: Display topic details.
   TOPIC(SYSTEM.ADMIN.TOPIC)              TYPE(LOCAL)
AMQ8633I: Display topic details.
   TOPIC(SYSTEM.BASE.TOPIC)               TYPE(LOCAL)
AMQ8633I: Display topic details.
   TOPIC(SYSTEM.BROKER.ADMIN.STREAM)      TYPE(LOCAL)
AMQ8633I: Display topic details.
   TOPIC(SYSTEM.BROKER.DEFAULT.STREAM)    TYPE(LOCAL)
AMQ8633I: Display topic details.
   TOPIC(SYSTEM.BROKER.DEFAULT.SUBPOINT)  TYPE(LOCAL)
AMQ8633I: Display topic details.
   TOPIC(SYSTEM.DEFAULT.TOPIC)            TYPE(LOCAL)
```

Notice: The topic **SYSTEM.BASE.TOPIC** is a key durable topic object …
… but for security reasons, do NOT give authority for this topic !!!

Assigning authorities for Pub/Sub

- Question 1:
- Should authorities be given for the SYSTEM.BASE.TOPIC?

- Answer:
- No!!!!
- Even though giving permissions to user 'alice' to do publishing actions against the SYSTEM.BASE.TOPIC would resolve the short term security issue…
- … it is NOT a Best Practice to do so, because it will give too much power to this non-administrator for tasks related to Pub/Sub

Assigning authorities for Pub/Sub

- Question 2:
- If authorities should NOT be given for the SYSTEM.BASE.TOPIC, then what else can be done to allow user 'alice' to publish into a topic?

- Answer:
- It is necessary to create a durable topic.
- It is also know as a “Topic Object” and will survive a restart of the queue manager.
- A ‘Topic Object’ contains an attribute that is a “topic string”

Defining a Topic Object

- To define a topic object called ‘SALES’ that has a topic string of ‘sales’, the MQ administrator issues:

- **define topic('SALES') topicstr('sales')**

- display topic('SALES')

```
AMQ8633I: Display topic details.
   TOPIC(SALES)                            TYPE(LOCAL)
   TOPICSTR(sales)                         DESCR( )
   CLUSTER( )                              CLROUTE(DIRECT)
   DURSUB(ASPARENT)                        PUB(ASPARENT)
   SUB(ASPARENT)                           DEFPSIST(ASPARENT)
   DEFPRTY(ASPARENT)                       DEFPRESP(ASPARENT)
   PMSGDLV(ASPARENT)                       NPMSGDLV(ASPARENT)
   PUBSCOPE(ASPARENT)                      SUBSCOPE(ASPARENT)
   PROXYSUB(FIRSTUSE)                      WILDCARD(PASSTHRU)
   MDURMDL( )                              MNDURMDL( )
   MCAST(ASPARENT)                         COMMINFO( )
   USEDLQ(ASPARENT)                        CUSTOM( )
```

Assigning authorities to a topic object

- Now that there is a durable topic, the MQ Administrator issues the following command to allow user 'alice' to perform the action “pub” on the topic object 'SALES' which has a topic string “sales”.
- Thus, the authorities for a topic string are done indirectly via a Topic Object.

- mqm@mosquito: /home/mqm
- $ setmqaut -m QMPS **-n 'SALES' -t topic** -p alice +pub

Non-admin user 'alice' tries to publish

- User 'alice' tries again.
- This time the attempt is successful!

- alice@mosquito: /home/alice
- $ amqspub **sales** QMPS
- Sample AMQSPUBA start
- target topic is sales
- **This is a test from Alice**
- Sample AMQSPUBA end

MQ samples use topic strings

- The MQ samples work on the **topic strings**
- And do NOT work on the **topic objects.**
- The following commands will fail for user ‘alice’ with rc 2035, because the sample will treat ‘SALES’ as a topic string, and there is no topic object that has a topic string of ‘SALES’.
  - alice@mosquito: /home/alice
  - $ amqspub SALES QMPS
  - Sample AMQSPUBA start
  - target topic is SALES
  - MQOPEN ended with reason code 2035
  - unable to open topic for publish
- Error: AMQ8009W: Entity 'alice' has insufficient authority to access topic string 'SALES'.

Notes: Authority problems

notes

http://www-01.ibm.com/support/docview.wss?uid=swg27016153

Authorizations needed for non-mqm users to publish and subscribe to Topics in MQ V7

.

+++ Summary

.

To let the users in the groups "editors" and "journalists" to connnect to the queue manager WMQ7:

```
setmqaut -m WMQ7 -t qmgr -g editors +connect +inq +dsp
setmqaut -m WMQ7 -t qmgr -g journalists +connect +inq +dsp
```

.

To let the users in the group "editors" subscribe to topic “DELI” on Queue Manager WMQ7 and to resume durable subscriptions:

```
setmqaut -m WMQ7 -n DELI -t topic -g editors +sub +resume
```

.

To allow users in the group "journalists" to publish to the topic:

```
setmqaut -m WMQ7 -n DELI -t topic -g journalists +pub
```

Troubleshooting authority problems

- Scenario:
- Customer reports a problem indicating that user ‘fulano’ is getting rc 2035 while trying to publish to topic string ‘sales/usa’.

- MQ L2 could request to customer:
- a) Output of Unix command: id fulano
- (To see the groups to which fulano belongs to)
- b) Issue runmqras with proper QMgrName / PMR
- runmqras -qmlist QMgrName -section defs,trace
  -pmrno 12345,678,000 -ftp IBM

Notes: Look at the error log of the qmgr

notes

- After receiving the runmqras file, you need to unzip it.
- Assuming that the unzipped files are at:
- C:\temp\QMPS-mosquito\
- Then you need to go to the subdirectory:
- var\mqm\qmgrs\QMPS\errors

- Look at the error log:
- AMQERR01.LOG

- Search for AMQ8077 or AMQ8009W.

- In this case is:

- AMQ8009W: Entity 'fulano' has insufficient authority to access topic string
- 'sales/usa'.
- EXPLANATION:
- The specified entity is not authorized to access the required topic. The
- following permissions were requested: pub

Notes: Security is based in groups

notes

- Keep in mind that the security in MQ is based on groups, and therefore, it is important to focus on the group membership for the user.
- If you are going to query the queue manager via runmqsc, you will not find references to the user 'fulano', but rather, to the groups to which this user belongs, in this case the group is "mqusers"

- fulano@mosquito: /home/fulano
- $ id fulano
- uid=1021(fulano) **gid=1006(mqviewer)** groups=1006(mqviewer)

- The MQ **runmqras** utility includes the information about authorities in 2 files:
  - amqoamd_QMPS.stdout
  - dmpmqaut_QMPS.stderr (note that dmpmqaut_QMPS.stdout is always empty!)

- And if you specify "-section defs" then you will get a 3rd file with information:
  - **defs/runmqsc_AUTHREC_QMPS.stdout**

Notes: Security is based in groups

notes

- + Let's compare user alice (group mqusers), who has already PUB access.
- Let's explore the relevant contents for the group "mqusers" (for user alice).
- We can see that group 'mqusers' has PUB authority on the topic object 'SALES'.
- Because the user belongs to the group 'mqusers' then, user has PUB authority.

- The command used was: **dmpmqaut -m QMPS > dmpmqaut_QMPS.stderr 2>&1**
- + File: dmpmqaut_QMPS.stderr
- profile: SALES
- object type: topic
- entity: mqusers
- entity type: group
- authority: pub

- The command used was: **runmqras -qmlist QMPS -section defs**
- + File: defs/runmqsc_AUTHREC_QMPS.stdout
- AMQ8864: Display authority record details.
- PROFILE(SALES) ENTITY(mqusers)
- ENTTYPE(GROUP) OBJTYPE(TOPIC)
- AUTHLIST(PUB)

Notes: Security is based in groups

notes

- The command used was: **amqoamd -m QMPS > amqoamd_QMPS.stdout**
- + File: amqoamd_QMPS.stdout
- ObjectType: topic
- ObjectName: SALES
- Principal: mqm
- Authorities: passall passid setall setid chg clr dlt dsp ctrl pub sub resume (0x027e3fff)
- **Principal: mqusers**
- **Authorities: pub (0x00000800)**

- Note that this file includes for each object, the different entities that have authorities.
- Thus, be careful to select the proper rows for the desired group!
- A common mistake is to just look at the top rows for group “mqm”!

Reviewing output files from runmqras

- In this particular scenario, user ‘fulano’ belongs to the group ‘mqviewer’.
- We look for group ‘mqviewer’ in the file:
- defs/runmqsc_AUTHREC_QMPS.stdout

- We see an entry:
- AMQ8864: Display authority record details.
- PROFILE(SALESUSA) ENTITY(mqviewer)
- ENTTYPE(GROUP) OBJTYPE(TOPIC)
- AUTHLIST(SUB,RESUME)

- This group can subscribe, but NOT to publish.

Reviewing output files from runmqras

- To find out the Topic Objects that are defined, we can take a look at the file:
  - defs/runmqsc_TOPIC_QMPS.stdout

- We see:
- AMQ8633: Display topic details.
  - TOPIC(SALESUSA) TYPE(LOCAL)
  - TOPICSTR(sales/usa) DESCR( )

setmqaut command to give authority

- Let's assume that the customer really wants to allow user 'fulano' to be able to publish into the topic string 'sales/usa'.

- We have now the following data:
  - group = mqviewer
  - topic object associated with topic string = SALESUSA

- The MQ administrator can issue the following command to provide the desired authority:

- **setmqaut -m QMPS -n 'SALESUSA' -t topic -g mqviewer +pub**

Subscriber not receiving messages

Subscriber not receiving messages

Scenario:
Publisher is using topic string:

sales => amqspub sales QMGR

… but subscriber is using a different topic string, thinking that it is the correct one to use:

SALES => amqssub SALES QMGR

The result is that the subscriber is not receiving messages.

Subscriber not receiving messages

This scenario may seem too obvious and simple, but there are some subtle things that potentially could cause confusion and delay in the resolution.

This scenario is very common!

One reason is that all parties will not be specific enough about using a topic string or a topic object. They will say simply, I am using topic X.
But one party could interpret as being
the topic object, while another could interpret it as the topic string.

Clarification on topic strings / topic objects

It is important to be clear on the following items:

The MQ samples use the topic STRING:

- amqspub sales QMGR

The MQ Resource Adapter use the topic STRING.

Topic objects and strings are CASE SENSITIVE:
The following are 2 different topic strings:

- amqspub sales QMGR
- amqspub SALES QMGR

Clarification on topic strings / topic objects

- A topic OBJECT is always associated with a topic STRING.
  - **define topic('SALES') topicstr('sales')**

- MQ does NOT check for “correctness".
- The following is perfectly fine!
  - **define topic(‘FRUITS’) topicstr(‘baseball')**

- A topic STRING MAY NOT be associated to a topic OBJECT.

The following is an “on the fly” topic string

  - amqspub vegetables QMGR

Clarification on topic strings / topic objects

- MQ does not try to “guess” intended meaning or to “fix” perceived typos:
- For example: MQ will treat “sale” as a valid string.
- amqspub sale QMGR

That is, the Qmgr will not say … hum, “sale” is a string that does not have an object, but there is an object that has a string ‘sales’, thus, the customer has made a typo and thus, the string ‘sales’ (and not ‘sale’) will be used instead. (This is NOT done!)

Some files to review from runmqras: topics

- Look at the defined topic objects:
- **defs/runmqsc_TOPIC_QMPS.stdout**

- The following can be seen:

- AMQ8633: Display topic details.
- TOPIC(SALES) TYPE(LOCAL)
- TOPICSTR(sales) DESCR( )

Some files to review from runmqras: pub

- Look at the status for the subscribers:
- defs/runmqsc_TPSTATUS_PUB_QMPS.stdout

- There is only 1 entry with string ‘sales’,
- TOPICSTR(sales)

- but none for string ‘SALE’:
- TOPICSTR(SALES)

This could give a hint that perhaps the subscriber is using the wrong string (no Publisher for SALES)

Some files to review from runmqras: sub

- Look at the status for the subscribers:
- defs/runmqsc_TPSTATUS_SUB_QMPS.stdout
- Or
- defs/runmqsc_SBSTATUS_QMPS.stdout

- There are many entries with string ‘sales’,
- TOPICSTR(sales)
- but only one with string ‘SALE’:
- TOPICSTR(SALES)

This could give a hint that perhaps the subscriber is using the wrong string.

Other troubleshooting guides

Advanced reference for Distributed Pub/sub

The following presentation has 2 pages regarding hints for troubleshooting in Distributed Pub/Sub (Cluster, Hierarchy):

Pages 55-56

http://mqug.org.uk/downloads/201407/201407%20-%20MQ202%20-%20PublishSubscribe%20in%20a%20network.pdf

MQ Publish/Subscribe in a network

Adrian Dick (adrian.dick@uk.ibm.com)

July 2014

Notes: Page 55

notes

Problem determination

- **Double check your configuration**
  - Remember that topic objects inherit behaviour from higher topic nodes, this can affect how lower topics behave. Especially for things like SUBSCOPE and PUBSCOPE.
- **When in a cluster, check each queue manager's cluster knowledge.**
  - Use DISPLAY TOPIC(*) TYPE(CLUSTER)
  - From V8, check the **CLSTATE** to see if problems exist (see the info center for more details).
- **When in a hierarchy, check your relationships.**
  - Use DISPLAY PUBSUB
- **Check the error logs.**
  - It may be obvious but check the error logs of the queue managers with publishers and subscribers, and full repository
queue managers, and any in between (topic hosts or hierarchy members).
- **Check your channels.**
  - If the channels to and from the queue managers are not running, publication traffic, proxy subscriptions, topic configuration (clusters) or relationships (hierarchies) will not flow.
- **Check your proxy subscriptions on each queue manager.**
  - Use DISPLAY SUB(*) SUBTYPE(PROXY)
- **Check your topic *status.***
  - Use DISPLAY TPSTATUS( ) to show how a topic string will behave, this takes into account any administered topic objects that relate to it in the topic tree.

Notes: Page 56 – part 1

notes

Problem determination

- **Watch for publication movement.**
- Publications may be flowing, but not noticed.
- Check the NUMMSGS value on the proxy subscriptions using DISPLAY SBSTATUS( ) to see if publications have been sent to the originator of the proxy subscription.
- Check the MSGS value on the channels to see if they are flowing using DISPLAY CHSTATUS( )
- **Look for a build up of messages.**
- In the event of problems, messages can build up on system queues. These may be because the system is producing messages too quickly or an error has occurred. Investigate to see if messages are still being processed, but slowly, or no messages are processed.

Notes: Page 56 – part 2

notes

Problem determination

- **Look for a build up of messages.**

- Queues which should be empty under normal running are:
  - SYSTEM.INTER.QMGR.PUBS (clusters)
    - Publications arriving from remote queue managers
  - SYSTEM.INTER.QMGR.FANREQ (clusters)
    - Requests to send proxy subscriptions
  - SYSTEM.INTER.QMGR.CONTROL (clusters)
    - Requests from other queue managers to register proxy subscriptions
  - SYSTEM.CLUSTER.TRANSMIT.QUEUE (or any other transmission queue involved)
    - All inter queue manager outbound messages
  - SYSTEM.BROKER.DEFAULT.STREAM (hierarchies)
    - Publications arriving from remote queue managers (and local ‘queued’ pub/sub applications)
  - SYSTEM.BROKER.CONTROL.QUEUE
    - Requests to register/deregister subscriptions
  - SYSTEM.BROKER.INTER.BROKER.COMMUNICATIONS (hierarchies)
    - Requests from other queue managers in the hierarchy.

- ***Don’t forget to see if there are undelivered messages in the dead letter queue***

Another reference

The following link has a PDF and an MP3 audio.

http://www-01.ibm.com/support/docview.wss?uid=swg27014870

Webcast replay: MQ V7 Publish/Subscribe Troubleshooting Tools & Techniques

It covers MQ troubleshooting and techniques including things to check (the usual suspects), error messages, Probe Id's, traces (how to read them), and the dump command.
Level of Difficulty: Beginner
Presenter(s): Paul O'Donnell
DATE: 15 January 2009

Notes: Recommended technotes

notes

https://www-01.ibm.com/support/docview.wss?uid=ibm10737035
Given an MQ SYSTEM.MANAGED queue, which is the subscription that is using it?

http://www-01.ibm.com/support/docview.wss?rs=171&uid=swg21375536
Identifying the name of an application that is subscribed to a topic

http://www-01.ibm.com/support/docview.wss?uid=swg21611038
How to identify SYSTEM.MANAGED.DURABLE dynamic queues that are no longer associated with existing durable subscriptions (orphaned queues)

https://developer.ibm.com/answers/questions/327184/for-mq-publishsubscribe-how-do-i-pick-up-a-boqname.html
For MQ Publish/Subscribe, how do I pick up a BOQNAME for the dynamic queue for a managed durable subscription?

http://www.ibm.com/support/docview.wss?uid=swg21606553
When a durable subscription expires, what happens to the messages stored for the subscription?

https://www.ibm.com/developerworks/community/blogs/aimsupport/entry/mq_subscriptions_receiving_duplicate_msgs?lang=en
Are your WebSphere MQ JMS subscribers receiving duplicate messages? Maybe you should use cloned subscriptions.

Miscellaneous

How to see which topics have retained msg

Using **DISPLAY TPSTATUS**.

The **RETAINED** field shows whether there is a retained message.

DISPLAY TPSTATUS('#') TYPE(TOPIC) WHERE(RETAINED EQ YES)

AMQ8754I: Display topic status details.
TOPICSTR(sales/canada) RETAINED(YES)

Topic nodes that have subscribers

Query to find out all the topic nodes that have subscribers:

```
DISPLAY TPSTATUS('#') TOPICSTR
        WHERE(SUBCOUNT GT 0)
```

Hints on reading the traces

kqiTopicPrepareForPublish
kqiTopicGetNextSubscriber
kqiPutMessageToSubscriber (which might decide not to, for example if a selector doesn't match)
kqiPutMsgSegments (if we actually do an MQPUT to a pubisher)

For published messages, you'll see a brand-new MQMD with a new, unique MessageId and a CorrelId which matches the SubscriberId of the subscriber

“defs” files from runmqas related to Pub/Sub

Notes: Status of a Topic, TPSTATUS

notes

- To find out the number of Publishers (PUBCOUNT) and Subscribers (SUBCOUNT) for a Topic String, display the Status of the Topic String via TPSTATUS.
- To display the status for a Topic String, you need to enclose the string within quotes:

- Wrong (no quotes): display tpstatus(fruit)
- AMQ8472E: IBM MQ topic string not found

- **DISPLAY TPSTATUS('fruit')**
- AMQ8754I: Display topic status details.
- **TOPICSTR(fruit)** ADMIN( )
- CLUSTER( )
- COMMINFO(SYSTEM.DEFAULT.COMMINFO.MULTICAST)
- MDURMDL(SYSTEM.DURABLE.MODEL.QUEUE)
- MNDURMDL(SYSTEM.NDURABLE.MODEL.QUEUE)
- CLROUTE(NONE) DEFPSIST(NO)
- DEFPRTY(0) DEFPRESP(SYNC)
- NPMSGDLV(ALLAVAIL) RETAINED(NO)
- MCAST(DISABLED) **PUBCOUNT(1)**
- **SUBCOUNT(2)** PUBSCOPE(ALL)
- SUBSCOPE(ALL) USEDLQ(YES)

Notes: Display all Topic Objects

notes

- defs/runmqsc_TOPIC_QMPS.stdout => DIS TOPIC(*) ALL

- AMQ8633I: Display topic details.
- **TOPIC(SALES)** TYPE(LOCAL)
- **TOPICSTR(sales)** DESCR( )

- CLUSTER( ) CLROUTE(DIRECT)
- DURSUB(ASPARENT) PUB(ASPARENT)
- SUB(ASPARENT) DEFPSIST(ASPARENT)
- DEFPRTY(ASPARENT) DEFPRESP(ASPARENT)
- PMSGDLV(ASPARENT) NPMSGDLV(ASPARENT)
- PUBSCOPE(ASPARENT) SUBSCOPE(ASPARENT)
- PROXYSUB(FIRSTUSE) WILDCARD(PASSTHRU)
- MDURMDL( ) MNDURMDL( )
- MCAST(ASPARENT) COMMINFO( )
- USEDLQ(ASPARENT) CUSTOM( )

- AMQ8633I: Display topic details.
- **TOPIC(SALESUSA)** TYPE(LOCAL)
- **TOPICSTR(sales/usa)** DESCR( )

- CLUSTER( ) CLROUTE(DIRECT)
- DURSUB(ASPARENT) PUB(ASPARENT)
- SUB(ASPARENT) DEFPSIST(ASPARENT)
- DEFPRTY(ASPARENT) DEFPRESP(ASPARENT)
- PMSGDLV(ASPARENT) NPMSGDLV(ASPARENT)
- PUBSCOPE(ASPARENT) SUBSCOPE(ASPARENT)
- PROXYSUB(FIRSTUSE) WILDCARD(PASSTHRU)
- MDURMDL( ) MNDURMDL( )
- MCAST(ASPARENT) COMMINFO( )
- USEDLQ(ASPARENT) CUSTOM( )

Notes: Display Topic Status (all topics)

notes

- defs/runmqsc_TPSTATUS_TOPIC_QMPS.stdout => Topic Status
- => DIS TPSTATUS('#') TYPE(TOPIC) ALL

- AMQ8754I: Display topic status details.
- **TOPICSTR()** **ADMIN(SYSTEM.BASE.TOPIC)**
- CLUSTER( )
- COMMINFO(SYSTEM.DEFAULT.COMMINFO.MULTICAST)
- MDURMDL(SYSTEM.DURABLE.MODEL.QUEUE)
- MNDURMDL(SYSTEM.NDURABLE.MODEL.QUEUE)
- CLROUTE(NONE) DEFPSIST(NO)
- DEFPRTY(0) DEFPRESP(SYNC)
- DURSUB(YES) PUB(ENABLED)
- SUB(ENABLED) PMSGDLV(ALLDUR)
- NPMSGDLV(ALLAVAIL) RETAINED(NO)
- MCAST(DISABLED) PUBCOUNT(0)
- SUBCOUNT(0) PUBSCOPE(ALL)
- SUBSCOPE(ALL) USEDLQ(YES)

Notes: Status of a Subscriber, TPSTATUS

notes

- From the Topic Status (TPSTATUS) it is possible to display the status for the subscribers by specifying: **type(SUB)**

- **display tpstatus('fruit') type(SUB)**
- AMQ8754I: Display topic status details.
- **TOPICSTR(fruit)**
- **SUBID(414D5120514D5053202020202020202020D57E785923F56904)**
- SUBUSER(AngelRivera) RESMDATE(2017-07-26)
- RESMTIME(08:18:22) LMSGDATE(2017-07-26)
- LMSGTIME(08:21:07)
- ACTCONN(414D5143514D5053202020202020202020D57E785923F56901)
- **DURABLE(NO)** **SUBTYPE(API)**
- MCASTREL(,) **NUMMSGS(2)**

- AMQ8754I: Display topic status details.
- **TOPICSTR(fruit)**
- **SUBID(414D5120514D5053202020202020202020D57E785923F5630A)**
- SUBUSER(AngelRivera) RESMDATE(2017-07-26)
- RESMTIME(08:15:47) LMSGDATE(2017-07-26)
- LMSGTIME(08:21:07)
- ACTCONN(414D5143514D5053202020202020202020D57E785923F56307)
- **DURABLE(NO)** **SUBTYPE(API)**
- MCASTREL(,) **NUMMSGS(4)**

Notes: Status of a Subscriber, SBSTATUS

notes

- To display the status for a subscriber, when knowing the SUBID (from display command from a previous slide)

- **display sbstatus SUBID**(414D5120514D5053202020202020202020D57E785923F5630A)
- AMQ8099I: IBM MQ subscription status inquired.
  - SUB( )
  - SUBID(414D5120514D5053202020202020202020D57E785923F5630A)
  - SUBUSER(AngelRivera) RESMDATE(2017-07-26)
  - RESMTIME(08:15:47) LMSGDATE(2017-07-26)
  - LMSGTIME(08:21:07)
  - ACTCONN(414D5143514D5053202020202020202020D57E785923F56307)
  - **DURABLE(NO)** MCASTREL(,)
  - **NUMMSGS(4)** **SUBTYPE(API)**
  - **TOPICSTR(fruit)**

Notes: Status of a Publisher, TPSTATUS

notes

- From the Topic Status (TPSTATUS) it is possible to display the status for the publishers by specifying: **type(PUB)**

- **display tpstatus('fruit') type(PUB)**
- AMQ8754I: Display topic status details.
  - TOPICSTR(fruit) LPUBDATE(2017-07-26)
  - LPUBTIME(08:21:07)
  - ACTCONN(414D5143514D505320202020202020202020D57E785923F56101)
  - MCASTREL(,) **NUMPUBS(9)**

Notes: Finding queue used by Subscribers

notes

- **display sub(*) dest topicstr**

- Note: showing only the subscriptions for topic strings “fruit” and “sport”
- Each Subscriber has its own “private” system temporary queue.

- AMQ8096I: IBM MQ subscription inquired.
  - **SUBID(414D5120514D50532020202020202020D57E785923F5630A)**
  - SUB( ) **TOPICSTR(fruit)**
  - **DEST(SYSTEM.MANAGED.NDURABLE.59787ED50963F523)**

- AMQ8096I: IBM MQ subscription inquired.
  - **SUBID(414D5120514D50532020202020202020D57E785923F56904)**
  - SUB( ) **TOPICSTR(fruit)**
  - **DEST(SYSTEM.MANAGED.NDURABLE.59787ED50369F523)**

- AMQ8096I: IBM MQ subscription inquired.
  - **SUBID(414D5120514D50532020202020202020D57E785923F56B04)**
  - SUB( ) **TOPICSTR(sport)**
  - **DEST(SYSTEM.MANAGED.NDURABLE.59787ED5036BF523)**

Notes: Full “display sub” for “fruit” 1

notes

- display sub(*) SUBTYPE(API) all

AMQ8096I: IBM MQ subscription inquired.
SUBID(414D5120514D50532020202020202020D57E785923F5630A)
SUB( ) **TOPICSTR(fruit)**
TOPICOBJ( ) DISTYPE(RESOLVED)
DEST(SYSTEM.MANAGED.NDURABLE.59787ED50963F523)
DESTQMGR(QMPS) PUBAPPID( )
SELECTOR( ) SELTYPE(NONE)
USERDATA( )
PUBACCT(16010501000000480366CD527F8B47B07254FAC26484730000000000000000B)
DESTCORL(414D5120514D50532020202020202020D57E785923F5630A)
DESTCLAS(MANAGED) **DURABLE(NO)**
EXPIRY(UNLIMITED) PSPROP(MSGPROP)
PUBPRTY(ASPUB) REQONLY(NO)
SUBSCOPE(ALL) SUBLEVEL(1)
SUBTYPE(API) VARUSER(FIXED)
WSCHEMA(TOPIC) SUBUSER(AngelRivera)
CRDATE(2017-07-26) CRTIME(08:15:47)
ALTDATE(2017-07-26) ALTTIME(08:15:47)

- Notice: DESTCLAS(MANAGED) DURABLE(NO)

Notes: Full “display qlocal” for temp queue

notes

display qlocal(SYSTEM.MANAGED.NDURABLE.59787ED50963F523)

```
QUEUE(SYSTEM.MANAGED.NDURABLE.59787ED50963F523)
   TYPE(QLOCAL)                            ACCTQ(QMGR)
   BOQNAME( )                              BOTHRESH(0)
   CLUSNL( )                               CLUSTER( )
   CLCHNAME( )                             CLWLPRTY(0)
   CLWLRANK(0)                             CLWLUSEQ(LOCAL)
   CURDEPTH(0)                             CUSTOM( )
   DEFBIND(OPEN)                           DEFPRTY(0)
   DEFPSIST(NO)                            DEFPRESP(SYNC)
   DEFREADA(YES)                           DEFSOPT(SHARED)
   DEFTYPE(TEMPDYN)
   DESCR(Model for managed queues for non durable subscriptions)
   DISTL(NO)                               GET(ENABLED)
   HARDENBO                                IMGRCOVQ(QMGR)
   INITQ( )                                IPPROCS(1)
   MAXDEPTH(999999999)                     MAXMSGL(4194304)
   MONQ(QMGR)                              MSGDLVSQ(PRIORITY)
   NOTRIGGER                               NPMCLASS(NORMAL)
   OPPROCS(0)                              PROCESS( )
   PUT(ENABLED)                            PROPCTL(COMPAT)
   QDEPTHHI(80)                            QDEPTHLO(20)
   QDPHIEV(DISABLED)                       QDPLOEV(DISABLED)
   QDPMAXEV(ENABLED)                       QSVCIEV(NONE)
   QSVCINT(999999999)                      RETINTVL(999999999)
   SCOPE(QMGR)                             SHARE
   STATQ(QMGR)                             TRIGDATA( )
   TRIGDPTH(1)                             TRIGMPRI(0)
   TRIGTYPE(FIRST)                         USAGE(NORMAL)
```

Notes: subscription with subtype(PROXY)

notes

PR 6 was not involved in the scenario (it has no publishers and no subscribers), but has “proxy subscriptions”. In PR 6, we can issue the following command in runmqsc to show them:

display sub(*) subtype(proxy) all

AMQ8096I: IBM MQ subscription inquired.
SUBID(414D5120434452505236202020202020203FFEBC596E217424)
SUB(SYSTEM.PROXY.CDRPR7 CLUSTERDR sales)
TOPICSTR(sales) TOPICOBJ( )
DISTYPE(RESOLVED) DEST(SYSTEM.INTER.QMGR.PUBS)
DESTQMGR(CDRPR7) PUBAPPID( )
SELECTOR( ) SELTYPE(NONE)
USERDATA( )
PUBACCT(033530310000000000000000000000000000000000000000000000000000000006)
DESTCORL(0000000000000000000000000000000000000000000000000000)
DESTCLAS(PROVIDED) DURABLE(YES)
EXPIRY(UNLIMITED) PSPROP(MSGPROP)
PUBPRTY(ASPUB) REQONLY(NO)
SUBSCOPE(ALL) SUBLEVEL(1)
SUBTYPE(PROXY) VARUSER(FIXED)
WSCHEMA(TOPIC) SUBUSER(mqm)

The End

This is the end of the presentation.

THANKS!!