Where is my message?

Matthew Whitehead
mwhitehead@uk.ibm.com
Agenda – the MQ Toolbox

- What’s connected?
- Re-routing in a cluster
- Are messages flowing?
- Where are messages going?
- What are the apps doing?
- How can I look back in time?

NB: Capabilities differ in form between distributed and z/OS, this presentation is distributed focused.
What’s connected?
DISPLAY CONN

**DISPLAY CONN(*) TYPE(HANDLE) ALL**

AMQ8276: Display Connection details.
CONN(577C425321292501), EXTCNN(41D51434741545571593120202020), TYPE(HANDLE)

OBJNAME(WLMMDB.REQUEST) OBJTYPE(QUEUE) ASTATE(NONE) HSTATE(INACTIVE)
OPENOPTS(MQOO_OUTPUT,MQOO_FAIL_IF QUIESCING) READA(NO)

OBJNAME(SENDINGAPP.REPLY) OBJTYPE(QUEUE) ASTATE(ACTIVE) HSTATE(ACTIVE)
OPENOPTS(MQOO_INPUT_SHARED,MQOO_INQUIRE,MQOO_SAVE_ALL_CONTEXT,MQOO_FAIL_IF QUIESCING) READA(NO)

**Use CONN to match TYPE(CONN) and TYPE(HANDLE) records**

**TYPE(HANDLE) records let you find applications by the objects they access. See all open handles for an app in one place, unlike DIS QSTATUS records**

**DISPLAY CONN(*) ALL**

AMQ8276: Display Connection details.
CONN(577C425321292501), EXTCNN(41D51434741545571593120202020), TYPE(CONN)
PID(9740) APPLDESC(WebSphere MQ Channel) APPLTAG(jms/JATEWAY1_CF)
APPLTYPE(SYSTEM) ASTATE(NONE)
CHANNEL(WAS.CLIENTS) CONNAME(127.0.0.1)
CONNOPTS(MQCNorio_SHARED_BINDING) USERID(pbroad)
UOWLOG( ) UOWSTDA(2014-04-08)
UOWSTTI(13.24.00) UOWLOGDA( ) URRTYPE(XA)

**Channel name + IP help identify client apps.**

**MQ V7.5 and later JMS clients can supply an application name in the CF**

**Long running UOW information. XID can be tied up with app server txn timeout**
### DISPLAY CHSTATUS(*) ALL

AMQ8417: Display Channel Status details.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHANNEL</td>
<td>WAS.CLIENTS</td>
</tr>
<tr>
<td>BUFSRCDV</td>
<td>17</td>
</tr>
<tr>
<td>BYTSRCDV</td>
<td>2296</td>
</tr>
<tr>
<td>CHSTADA</td>
<td>2014-04-08</td>
</tr>
<tr>
<td>COMPHDR</td>
<td>(NONE, NONE)</td>
</tr>
<tr>
<td>COMPRATE</td>
<td>(0, 0)</td>
</tr>
<tr>
<td>CONNAME</td>
<td>127.0.0.1</td>
</tr>
<tr>
<td>EXITTIME</td>
<td>0</td>
</tr>
<tr>
<td>JOBNAME</td>
<td>00000260C000000B9</td>
</tr>
<tr>
<td>LSTMSGDA</td>
<td>2014-04-08</td>
</tr>
<tr>
<td>MCASTAT</td>
<td>RUNNING</td>
</tr>
<tr>
<td>MONCHL</td>
<td>OFF</td>
</tr>
<tr>
<td>RAPPLTAG</td>
<td>jar</td>
</tr>
<tr>
<td>SSLKEYDA</td>
<td></td>
</tr>
<tr>
<td>SSLPEER</td>
<td>SERIALNUMBER=3:43:FD:D6,CN=ExampleApp1,O=Example</td>
</tr>
<tr>
<td>SSLREKEYS</td>
<td>0</td>
</tr>
<tr>
<td>STOPREQ</td>
<td>NO</td>
</tr>
<tr>
<td>CURSHCNV</td>
<td>1</td>
</tr>
<tr>
<td>RVERSION</td>
<td>00000000</td>
</tr>
<tr>
<td>CHLTYPE</td>
<td>SVRCONN</td>
</tr>
<tr>
<td>BUFSSENT</td>
<td>13</td>
</tr>
<tr>
<td>BYTSSENT</td>
<td>2456</td>
</tr>
<tr>
<td>CHSTATI</td>
<td>15.26.59</td>
</tr>
<tr>
<td>COMPMMSG</td>
<td>(NONE, NONE)</td>
</tr>
<tr>
<td>COMPTIME</td>
<td>(0, 0)</td>
</tr>
<tr>
<td>CURRENT</td>
<td></td>
</tr>
<tr>
<td>HBINT</td>
<td>5</td>
</tr>
<tr>
<td>LOCLADDR</td>
<td></td>
</tr>
<tr>
<td>LSTMSGTI</td>
<td>15.26.59</td>
</tr>
<tr>
<td>MCAUSER</td>
<td>pbroad</td>
</tr>
<tr>
<td>MSGS</td>
<td>6</td>
</tr>
<tr>
<td>MCASTAT</td>
<td>RUNNING</td>
</tr>
<tr>
<td>MCAUSER</td>
<td>pbroad</td>
</tr>
<tr>
<td>MONCHL</td>
<td>OFF</td>
</tr>
<tr>
<td>RAPPLTAG</td>
<td>jar</td>
</tr>
<tr>
<td>SSLCERTI</td>
<td>(CN=ExampleCA,O=Example)</td>
</tr>
<tr>
<td>SSLKEYTI</td>
<td></td>
</tr>
<tr>
<td>SSLPEER</td>
<td>SERIALNUMBER=53:43:FD:D6,CN=ExampleApp1,O=Example</td>
</tr>
<tr>
<td>SSLREKEYS</td>
<td>0</td>
</tr>
<tr>
<td>STOPREQ</td>
<td>NO</td>
</tr>
<tr>
<td>CURSHCNV</td>
<td>1</td>
</tr>
<tr>
<td>RVERSION</td>
<td>00000000</td>
</tr>
<tr>
<td>STATUS</td>
<td>RUNNING</td>
</tr>
<tr>
<td>SUBSTATE</td>
<td>RECEIVE</td>
</tr>
<tr>
<td>MAXSHCNV</td>
<td>1</td>
</tr>
<tr>
<td>RPRODUCT</td>
<td>MQJM</td>
</tr>
</tbody>
</table>

---

Check suitable heartbeats are negotiated

See SSLPEER information not in DIS CONN

JOBNAME contains PID (except z/OS): 0x260C = PID(9740)
On Linux/UNIX (not Win) TID matches CONN: 0xB9 = TID(185)

Note that multiple CONN might share one SVRCONN channel instance
Rerouting messages if no-one is connected
Cluster WLM

- Cluster WLM algorithm doesn’t check whether consuming applications are connected to cluster queues
  - I.e. whether something is getting messages from them

- You need to ensure that applications are consuming from all instances of a clustered queue to prevent messages building up

- However, there is an alternative: AMQSCLM
AMQSCLM: 1) Normal operation

Sending apps

Driver qmgr 1  Driver qmgr 2  ...  Driver qmgr N

Cluster WLM

Round-robin message distribution

AMQSCLM

IPPROCS(>=1)

App server 1  App server 2  ...  App server N

All platforms
AMQSCLM: 2) Application failure

- Sending apps
- Driver qmgr 1
- Driver qmgr 2
- ... Driver qmgr N
- Cluster WLM
- Round-robin message distribution
- Messages start building up
- App server 1
- App server 2
- ... App server N
AMQSCLM: 3) Failure Detection

AMQSCLM checks the queue every 5 minutes and detects IPPROCS(0)

Sending apps

Driver qmgr 1
Driver qmgr 2
... 
Driver qmgr N

Cluster WLM

Round-robin message distribution

AMQSCLM
AMQSCLM
AMQSCLM

IPPROCS(0)
IPPROCS(=>1)
IPPROCS(=>1)

App server 1
App server 2
... 
App server N

All platforms
AMQSCLM: 4) Redirect Messages

AMQSCLM reduces queue priority in cluster, to stop new messages arriving.

Sending apps

Driver qmgr 1

Driver qmgr 2

... Driver qmgr N

Cluster WLM

Round-robin message distribution

App server 1

App server 2

... App server N

AMQSCLM reduces queue priority in cluster, to stop new messages arriving.
AMQSCLM: 5) Un-maroon Msgs (optional)

AMQSCLM tells qmgr to redistribute ‘stuck’ messages via the cluster.

Sending apps

Driver qmgr 1
Driver qmgr 2
... Driver qmgr N

Cluster WLM

Round-robin message distribution

AMQSCLM

PPROCS(0)

App server 1
App server 2
... App server N

All platforms

MQ Technical Conference v2.0.1.8
The cluster queue monitoring sample program (AMQSCLM)

Shipped with the product as a sample
- Precompiled
- Source code
- Not shipped on z/OS, but distributed source code can be compiled and used on z/OS

More information here:
http://www.ibm.com/support/knowledgecenter/SSFKJSJ_8.0.0/com.ibm.mq.dev.doc/q024620_.htm
Are messages flowing?
Real-time/online monitoring – queues

- Set detail level for queue manager. Override for individual queues
  - LOW, MEDIUM and HIGH set frequency of sampling, not type of stats generated

```
ALTER QMGR MONQ(MEDIUM)
ALTER QLOCAL(QUEUE1) MONQ(HIGH)
ALTER QLOCAL(QUEUE2) MONQ(OFF)
```

- Gives live view of application responsiveness

```
AMQ8450: Display queue status details.  
QUEUE (QUEUE1)  
CURDEPTH (16)  
LGETDATE (2014-04-08)  
LPUTDATE (2014-04-08)  
MEDIALOG ( )  
MSGAGE (112)  
QTIME (10101414, 10101414)  
TYPE (QUEUE)  
IPPROCS (3)  
LGETTIME (17.05.59)  
LPUTTIME (17.12.16)  
MONQ (HIGH)  
OPROCS (5)  
UNCOM (NO) 
```

Without MONQ you only get the depth and how many handles are open

Timestamps of last PUT/GET to check for recent activity

Age in **seconds** of the oldest message on the queue

Estimations of the time in **microseconds** that messages are waiting on the queue for processing.
- First value: Calculated from recent activity
- Second value: Calculated from longer term activity
## Real-time/online monitoring – channels

- **ALTER QMGR MONCHL (MEDIUM) MONACL (MEDIUM)**
- **ALTER CHANNEL (CLUSTER1.QM1) CHLTYPE (CLUSRCVR) MONCHL (HIGH)**

### Gives live view of channel throughput

#### AMQ8417: Display Channel Status details.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHANNEL (MQHUB.GATEWAY2)</td>
<td></td>
</tr>
<tr>
<td>BATCHES (52)</td>
<td></td>
</tr>
<tr>
<td>BUFSRCVD (55)</td>
<td></td>
</tr>
<tr>
<td>BYTSTRCVD (1748)</td>
<td></td>
</tr>
<tr>
<td>CHSTADA (2014-04-08)</td>
<td></td>
</tr>
<tr>
<td>COMPRATE (0,0)</td>
<td></td>
</tr>
<tr>
<td>CONNAME (127.0.0.1(1422))</td>
<td></td>
</tr>
<tr>
<td>CURMSGS (50)</td>
<td></td>
</tr>
<tr>
<td>CURSEQNO (11823)</td>
<td></td>
</tr>
<tr>
<td>HBINT (5)</td>
<td></td>
</tr>
<tr>
<td>JOBNAM (00002DA4000047D00)</td>
<td></td>
</tr>
<tr>
<td>LONGRTS (999999999)</td>
<td></td>
</tr>
<tr>
<td>LSTMSGDA (2014-04-08)</td>
<td></td>
</tr>
<tr>
<td>LSTSEQNO (11773)</td>
<td></td>
</tr>
<tr>
<td>MONCHL (MEDIUM)</td>
<td></td>
</tr>
<tr>
<td>NETTIME (137538,29555)</td>
<td></td>
</tr>
<tr>
<td>RQMNAME (GATEWAY2)</td>
<td></td>
</tr>
<tr>
<td>SSLCERTI ( )</td>
<td></td>
</tr>
<tr>
<td>SSLKEYTI ( )</td>
<td></td>
</tr>
<tr>
<td>SSLRKEYS (0)</td>
<td></td>
</tr>
<tr>
<td>STOPREQ (NO)</td>
<td></td>
</tr>
<tr>
<td>XBATCHSZ (20,17)</td>
<td></td>
</tr>
<tr>
<td>XQMSGSA (112)</td>
<td></td>
</tr>
<tr>
<td>RVERSION (07050002)</td>
<td></td>
</tr>
<tr>
<td>CHLTYPE (CLUSSDR)</td>
<td></td>
</tr>
<tr>
<td>BATCHSZ (50)</td>
<td></td>
</tr>
<tr>
<td>BUFSSENT (1616)</td>
<td></td>
</tr>
<tr>
<td>BYTSSSENT (1192330)</td>
<td></td>
</tr>
<tr>
<td>CHSTATI (17.49.03)</td>
<td></td>
</tr>
<tr>
<td>COMPMGS (NONE, NONE)</td>
<td></td>
</tr>
<tr>
<td>COMPTIME (0,0)</td>
<td></td>
</tr>
<tr>
<td>CURLUWID (0107445310001B34)</td>
<td></td>
</tr>
<tr>
<td>CURRENT</td>
<td></td>
</tr>
<tr>
<td>EXITTIME (0,0)</td>
<td></td>
</tr>
<tr>
<td>INDOUBT (YES)</td>
<td></td>
</tr>
<tr>
<td>LOCLADDR (127.0.0.1(1422))</td>
<td></td>
</tr>
<tr>
<td>LSTLUID (010145310001B33)</td>
<td></td>
</tr>
<tr>
<td>LSTMSGT (17.49.51)</td>
<td></td>
</tr>
<tr>
<td>MCASTAT (RUNNING)</td>
<td></td>
</tr>
<tr>
<td>MSGS (1500)</td>
<td></td>
</tr>
<tr>
<td>NPMSPEED (FAST)</td>
<td></td>
</tr>
<tr>
<td>SHORTTS (180)</td>
<td></td>
</tr>
<tr>
<td>SSLKEYDA ( )</td>
<td></td>
</tr>
<tr>
<td>SSLPEER ( )</td>
<td></td>
</tr>
<tr>
<td>SSLRKEYS (0)</td>
<td></td>
</tr>
<tr>
<td>STAT (RUNNING)</td>
<td></td>
</tr>
<tr>
<td>SUBSTATE (RECEIVE)</td>
<td></td>
</tr>
<tr>
<td>XMITQ (SYSTEM.CLUSTER.TRANSMIT.QUEUE)</td>
<td></td>
</tr>
<tr>
<td>XQTIME (545784,39299968)</td>
<td></td>
</tr>
<tr>
<td>RPRODUCT (MQMM)</td>
<td></td>
</tr>
</tbody>
</table>

**Short/long term calculations of how full your batches are getting, to help you tune BATCHSZ/BATCHINT**

**Last time a message was sent over the channel**

**Depth of messages on XMITQ for this channel (capped at 999)**

**Short/long term calculations of how long messages are waiting on the XMITQ for transmission**
### Accounting and statistics overview

- **Monitoring data sent as a PCF message at a configured interval**
  - Statistics – scoped to a queue / channel / QMGR
  - Accounting – scoped to an individual CONN and queue / QMGR

**Related attributes**
- STATQ
- STATMQI
- STATCHL
- STATACLS
- STATINT

**PCF statistics collection app**
- SYSTEM.ADMIN.STATISTICS.QUEUE

**PCF accounting collection app**
- SYSTEM.ADMIN.ACCOUNTING.QUEUE

**Related attributes**
- ACCTQ
- ACCTMQI
- ACCTINT

- Use SMF on z/OS
- New options in V9
- Dist platforms

© 2017 IBM Corporation
Accounting and statistics overview

PCF statistics collection app

PCF accounting collection app

QMGR

Statistics Messages

Accounting Messages

SYSTEM.ADMIN.STATISTICS.QUEUE

SYSTEM.ADMIN.ACCOUNTING.QUEUE

e.g.

MS0P Explorer Plugin
(SupportPac)

amqsmmon
Specifically for accounting and statistics

amqsevt
Generic PCF event message formatter, TEXT or JSON output

Other
- Tivoli
- Custom app
- E.g. Java app using com.ibm.mq.headers

New options in V9

Use SMF on z/OS

Dist platforms
Taking a look with SupportPac MS0P

- **Time period**
- **MQ Statistics at QMGR level**
- **Detailed queue statistics**
Taking a look at **accounting events with amqsevt**

Choose JSON or TEXT format

Which type of event is this: ACCTMQI or ACCTQ?

Which application does this event relate to?
Taking a look at **statistics** events with `amqsevt`

Choose **JSON** or **TEXT** format

Which type of event is this: `STATMQ`, `STATQ`, `STATCHL`, or `STATACLS`?

Period of time the statistics relate to
Taking a look at **statistics** using **amqsmon**

### Overall QMGR busyness

- **Simple data format**
  - Multiple values are
    - [Persistent, NonPersistent]

### One message every X seconds
- Use **amqsmon** directly (perl/cron)

### Low/high water marks for Subscriptions
- Grouped by subscription type

- **amqsmon** is a sample so you can use it as a base for your own tools

```sql
ALTER QMGR STATMQI (ON)
Wait a bit, but not the default 30 minutes between stats records
RESET QMGR TYPE(STATISTICS)
amqsmon -m GATEWAY1 -t statistics -a -w 0
```

<table>
<thead>
<tr>
<th>MonitoringType</th>
<th>MQIStatistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>QueueManager</td>
<td>GATEWAY1</td>
</tr>
<tr>
<td>IntervalStartDate</td>
<td>'2014-04-09'</td>
</tr>
<tr>
<td>IntervalStartTime</td>
<td>'00.00.35'</td>
</tr>
<tr>
<td>IntervalEndData</td>
<td>'2014-04-09'</td>
</tr>
<tr>
<td>IntervalEndTime</td>
<td>'00.01.13'</td>
</tr>
<tr>
<td>CommandLevel</td>
<td>700</td>
</tr>
<tr>
<td>ConnCount</td>
<td>35</td>
</tr>
</tbody>
</table>

- **PutCount**: [271, 0]
- **PutFailCount**: 0
- **Put1Count**: [2, 0]
- **Put1FailCount**: 0
- **PutBytes**: [273976, 0]
- **GetCount**: [270, 0]
- **GetBytes**: [269468, 0]
- **GetFailCount**: 19

- **DurableSubscriptionHighWater**: [0, 0, 0, 0]
- **DurableSubscriptionLowWater**: [0, 0, 0, 0]
- **NonDurableSubscriptionHighWater**: [0, 0, 0, 0]
- **NonDurableSubscriptionLowWater**: [0, 0, 0, 0]
- **PutTopicCount**: [0, 0]
- **PutTopicFailCount**: 0
- **Put1TopicCount**: [0, 0]
- **Put1TopicFailCount**: 0
- **PutTopicBytes**: [0, 0]
- **PublishMsgCount**: [0, 0]
- **PublishMsgBytes**: [0, 0]

- Use **SMF** on **z/OS**

- New options in **V9**
System topics on distributed queue managers

- Distributed queue manager information is published to a range of system topic strings
  - $SYS/MQ/INFO/QMGR/….

- Authorised subscriptions receive their own stream of publications based on the topic string
  - Administrative subscriptions
    - E.g. For information to be continually sent to defined queues
  - Application subscriptions
    - E.g. To dynamically listen to information as required

- Unlocks system level information for MQ administrators and DevOps teams
  - Administrators can grant access to subsets of the data, pertinent to different application teams
System Monitoring

- **Familiar statistics available through subscriptions**
  - Queue manager wide statistics (connects, disconnects, opens, closes, puts, gets, …)
  - Queue level statistics (opens, closes, puts, gets, …)
  - NB: statistics available from system topics are not a 1-1 mapping to those available from system queues
    - E.g no channel statistics, some missing information, some new information, some merged information
  - No support for accounting data

- **Extended to include CPU and disk usage. For example…**
  - Queue manager **CPU time, memory usage**
  - **Disk reads/writes, disk latency,**

- **Subscribe to meta-topic to learn which classes of statistics are available**
  - $SYS/MQ/INFO/QMGR/QMGR1/Monitor/METADATA/CLASSES
  - Then subscribe to specific topics
    - $SYS/MQ/INFO/QMGR/QMGR-NAME/Monitor/class[/instance]/type]
  - See amqsrua sample program
By default messages are published every 10 seconds

- Configurable via a tuning parameter in qm.ini, e.g.

  TuningParameters:
  MonitorPublishHeartBeat=60

Recent updates to amqsrua sample program let you subscribe to other parts of the $SYS topic tree

- amqsrua –p \"$SYS/Application/runmqsfb\"

- Applications can publish their own meta-data and events to $SYS/Application

- Single place to consume MQ and application events from
$ amqsrua -m V9000_A
CPU : Platform central processing units
DISK : Platform persistent data stores
STATMQI : API usage statistics
STATQ : API per-queue usage statistics
Enter Class selection
  ==> CPU
SystemSummary : CPU performance - platform wide
QMgrSummary : CPU performance - running queue manager
Enter Type selection
  ==> SystemSummary
Publication received PutDate:20160411 PutTime:10465573
User CPU time percentage 0.01%
System CPU time percentage 1.30%
CPU load - one minute average 8.00
CPU load - five minute average 7.50
CPU load - fifteen minute average 7.30
RAM free percentage 2.02%
RAM total bytes 8192MB
Publication received PutDate:20160411 PutTime:10466573
User CPU time percentage 0.01%
System CPU time percentage 1.30%
...
Where are messages going?
MQ networks can be complex

- At each of the dots stuck / mis-sent messages are possible
  - MQOPENs of the wrong queue / queue manager by apps
  - Full queues
  - Stopped channels
  - Stopped apps
  - Incorrectly configured QREMOTE/QALIAS routing objects
  - Cluster membership problems

- The standard problem diagnosis approach
  - Methodically checking channels/queues/DLQs at each point

- Is there anything to speed up this process?
MQ has the ability to inject trace route messages
  ▶ (Can be) hidden from applications
  ▶ Generate activity reports as they pass through, potentially accumulated in the message

Tools are available to trace routes using these reports
  ▶ dspmqrte – command line tool supplied with the product
  ▶ MS0P – Cat 2 SupportPac extension to MQ Explorer

Lets you see the path messages could have taken
  ▶ Test connectivity through the MQ network
  ▶ Test cluster workload balancing

Can quickly jump you close to the problem
  ▶ The point your trace message veers off in the wrong direction
  ▶ The point the trail goes cold

NB: there is also a related technology: activity recording which generates activity reports from real messages
What are the apps doing?
Applying activity trace

- Information on all the MQI operations performed by an application in the order that they were done

- Similar infrastructure to accounting & statistics
  - PCF messages on SYSTEM.ADMIN.TRACE.ACTIVITY.QUEUE
  - Configurable via mqat.ini
    - Can be changed without queue manager restart
    - Configurable detail level can include partial/full message payload
    - Frequency options for tuning
  - Can be enabled on a per-application basis
    - Via MQCONNX flags
    - Via application name

Application activity trace

- Enables scenarios such as
  - Application audit trail
  - Message duplication
  - Resource usage
    - Which queues or topics are actually being used
  - Problem determination
    - Which queue / queue manager is the application actually opening
  - Application coding standards
    - Does everyone use the MQI in the recommended way
  - And more …

Health warning: Performance impact
http://ow.ly/vA8wB
Looking at the data with the amqsact sample

```
ALTER QMGR ACTVTRC (ON)
```

Keep in mind, `amqsact -m GATEWAY1 -v` should be tuned via `mqat.ini`.

```
MQI Operation: 6
Operation Id: MQXF.PUT
ApplicationTid: 12451
OperationDate: '2014-04-09'
OperationTime: '01:39:48'
High Res Time: 139700398665548
Completion Code: MQCC_OK
Reason Code: 0
Hobj: 18225032
Put Options: 139330
Msg length: 460
Recs_present: 0
Known_dest_count: 1
Unknown_dest_count: 0
Object_type: MQOT_Q
Object_name: 'SENDINGAPP.REPLY'
Object_Q_mgr_name: 'GATEWAY1'
Resolved_Q_Name: 'SENDINGAPP.REPLY'
Resolved_Q_mgr: 'GATEWAY1'
Resolved_local_Q_name: 'SENDINGAPP.REPLY'
Resolved_local_Q_mgr: 'GATEWAY1'
Resolved_type: MQOT_Q
Report Options: 0
Msg_type: MQMT_DATAGRAM
Expiry: -1
Format_name: 'MQHRF2'
Priority: 4
Persistence: 0
Msg_id:
00000000: 414D 5120 4741 5445 5741 5931 2020 2020 'AMQ  GATEWAY1   ' 00000010: 0207 4453 2007 2203 '..DS  .'.
Correl_id:
00000000: 414D 5120 4741 5445 5741 5931 2020 2020 'AMQ  GATEWAY1   ' 00000010: 0207 4453 2007 2203 '..DS  .'.
Reply_to_Q : '
Reply_to_Q_Mgr: '
Coded_char_set_id: 1208
Encoding: 273
Put_date: '20140409'
Put_time: '00394866'
```

- **Check** options used for coding standards.
- **Check** queue name resolution, to find out why messages are going to the wrong place.
- **Track** individual messages and request/reply scenarios with `Msg_id` and `Correl_id`.

As of V9, this is known as display mode.

Make sure you ask `amqsact` to show everything the QM has generated (`-v`).

New options in V9

Not on z/OS
Application activity trace – system topics

- Application activity trace enabled through subscriptions rather than queue manager configuration

- **Subscribe to topic**
  - E.g. $SYS/MQ/INFO/QMGR/QMGR1/ActivityTrace/ApplName/amqsput
  - Filter by application name, channel or connection id

- When a subscription is created, PCF messages start to flow to the subscriber’s queue. When subscription is deleted, messages stop

- Much easier to get just the data you want!
### Application activity trace – dynamic sample

- **Sample provided to demonstrate usage and format output**
- **Example below specifies application name (-a) so uses dynamic mode**
- **Dynamic mode subscribes to system topic rather than uses system queue**
- **Channel name and connection id also supported**

```bash
$ amqsfam -m QMGR1 -a amqspput -w 60
Subscribing to the activity trace topic: 'SYS/MQ/INFO/QMGR/QMGR1/ActivityTrace/ApplName/amqspput'

MonitoringType: MQI Activity Trace
...
QueueManager: 'QMGR1'
ApplicationName: 'amqspput'
Application Type: MQAT_UNIX
...
```

<table>
<thead>
<tr>
<th>Tid</th>
<th>Date</th>
<th>Time</th>
<th>Operation</th>
<th>CompCode</th>
<th>MQRC</th>
<th>HObj (ObjName)</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>2016-04-14</td>
<td>09:56:53</td>
<td>MQXF_CONNX</td>
<td>MQCC_OK</td>
<td>0000</td>
<td>-</td>
</tr>
<tr>
<td>001</td>
<td>2016-04-14</td>
<td>09:56:53</td>
<td>MQXF.OPEN</td>
<td>MQCC_OK</td>
<td>0000</td>
<td>2 (QUEUE1)</td>
</tr>
<tr>
<td>001</td>
<td>2016-04-14</td>
<td>09:56:53</td>
<td>MQXF_PUT</td>
<td>MQCC_OK</td>
<td>0000</td>
<td>2 (QUEUE1)</td>
</tr>
<tr>
<td>001</td>
<td>2016-04-14</td>
<td>09:56:53</td>
<td>MQXF_PUT</td>
<td>MQCC_OK</td>
<td>0000</td>
<td>2 (QUEUE1)</td>
</tr>
</tbody>
</table>
```

Sample AMQSPUT0 start

target queue is QUEUE1

Hello

World

Sample AMQSPUT0 end

$
How can I look back in time?
What happened to my messages at 2am this morning?

- **Enterprise monitoring solution**
  - DLQ alerts, queue depth alerts, channel status alerts
  - Unresolved running units of work
  - Historical MQ monitoring, accounting and stats data

- **App logs from the time of the problem**
  - Exceptions, MQ error codes, timeouts

- **MQ error logs for all qmgrs that could have been involved**
  - Channel errors
  - Authentication issues

- ??? – what else is there
What about the MQ recovery log?

- **For persistent messages inside transactions**
  - MQ logs each operation performed
  - Outside of transactions persistent messages might be logged

- **Why can't we use this to**
  - Look back in time to 2am and see what happened?
  - Recover the original payload if the app lost the message?
  - See what happened inside long-running units of work?
  - Provide a list of operations within the failed business transaction?

- **MQ documents how you can… if**
  - You use the text formatting tool provided with MQ (dmpmqlog)
  - The logging is linear so the historical data is available in the tool
  - You follow the right steps to extract data from running qmgrs
  - You do the work to follow through the logs
dmpmqlog output is readable, but analysis is tedious

ordered unique IDs for each record (LSN)

A set of documented record types

Transaction information with XIDs, or re-used MQ transaction IDs

MQMD header data at discoverable offsets in the hex of a message Put

The message payload itself

Wouldn't it be easier to let the computer do the tedious bit?
Check out dmpmqlog scraper tool

- Takes the tedium out of analysing the output from dmpmqlog
- Created by Peter Broadhurst
  
  ```
  java -jar dmpmqlog.scraper-20151201.jar -b little-endian -i dmpmqlog.txt -o .
  ```

- Generates file per message PUT in the supplied data
- Summary file
dmpmqlog scraper tool output

- Generates file per message PUT in the supplied data.
- Summary file
Summary

- Lots of tools in your MQ toolbox!

- **On-line status commands**
  - DISPLAY CONN
  - DISPLAY QSTATUS
  - DISPLAY CHSTATUS

- **Cluster monitoring – AMQSCLM**

- **Off-line statistics and accounting**
  - amqsmon and MS0P to view

- **Tracking**
  - Trace-route
  - Application activity trace

- **MQ recovery logs**
  - dmpmqlog scraper