WebSphere MQ for z/OS Performance and Accounting

Mayur Raja

mayur_raja@uk.ibm.com

Legal Disclaimer

- The information contained in this publication is provided for informational purposes only. While efforts were made to verify the completeness and accuracy of the information contained in this publication, it is provided AS IS without warranty of any kind, express or implied. In addition, this information is based on IBM's current product plans and strategy, which are subject to change by IBM without notice. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, this publication or any other materials. Nothing contained in this publication is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software.
- References in this presentation to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates. Product release dates and/or capabilities referenced in this presentation may change at any time at IBM's sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way. Nothing contained in these materials is intended to, nor shall have the effect of, stating or implying that any activities undertaken by you will result in any specific sales, revenue growth or other results.
- Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon many factors, including considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve results similar to those stated here.
- IBM, the IBM logo and WebSphere are trademarks of International Business Machines Corporation in the United States, other countries, or both.

Agenda

- **Why monitor your systems ?**
- **Accounting and Statistics**
  - Queue Manager
  - Channel Initiator

Why monitor your systems ?

- What are you trying to achieve from your system ?
- What MQ resources is your system using ?
- How efficiently is your system running ?

- Being able to answer these questions will allow you to:
  - Administer your system effectively
  - Prevent outages and degradations in service
  - Plan for growth or change
  - Save MIPS and money !

Introduction

N

O

T

E

S

- This presentation discusses statistics and accounting information provided by WebSphere MQ. The data that can be collected is a valuable aid to administrators to effectively manage MQ resources.
- The data that can be collected can be used to understand how a system is being used now and over time. Using this information effectively helps to prevent outages or degradations in service. It is also valuable input to planning growth or other system changes.
- Understanding a system can also be used to improve performance, which on z/OS can save both MIPS and money!

Monitoring

- **WebSphere MQ provides various monitoring capabilities:**
  - DISPLAY/PCF commands,
  - MQ Explorer
  - PCF event messages
    - Queue manager – Start/stop, GET/PUT inhibited, unknown object
    - Configuration – Define/Alter/Delete/Refresh objects
    - Command – Commands issued and by who
    - Performance – Queue depth high/low/full
    - Channel/bridge – Start/stop, SSL
  - Queue manager performance statistics
    - High-level view of activity
  - Queue manager accounting data
    - Detailed information on MQI operations
    - Provides capability to charge users for queue manager usage
  - Channel statistics and accounting

Monitoring

N O T E S

- WebSphere MQ includes a number of features that allow an administrator to monitor activity, which range from simple display commands to detailed statistics and accounting information for analysing trends in system activity, or charging for system usage.
- MQSC commands can be issued via the console, ISPF panels or `CSQUTIL`, and PCF can be used to perform the same actions programmatically. PCF is exploited by the MQ Explorer and many Tivoli and vendor products.
- WebSphere MQ can also generate PCF messages on system queues when certain events occur, such as commands issued or changes in object configuration. Events can also be generated when certain errors or conditions occur, such as attempts to access unknown objects, or queue depths reaching a given threshold.
- This presentation focuses on two other capabilities provided by MQ, which are statistics and accounting information. Performance statistics can be used to understand system activity at a high level. Accounting data provides detailed information on MQI operations that can also be used to charge for system usage.
- In previous releases, statistics and accounting information was provided for activity in the queue manager. Version 8 introduces equivalent information for activity in the channel initiator, which can be used to monitor channel activity.

Collecting statistics and accounting data

- **System Management Facility (SMF) records**
  - Event messages on distributed
- **Statistics – SMF type 115 records**
  - `START TRACE(STAT)`
  - `SMFSTAT` system parameter (zPARM)
- **Accounting – SMF type 116 records**
  - `START TRACE(ACCTG)`
  - `SMFACCT` system parameter (zPARM)
- **Collection interval**
  - Specified using the `STATIME` system parameter
  - If zero, the SMF global accounting interval is used

Collecting statistics and accounting data

N O T E S

- On distributed platforms enabling statistics and accounting information generates PCF messages, as per events. On z/OS, this information is generated as System Management Facility (SMF) records instead.
- Statistics information is recorded as SMF 115 records.
- Accounting data is recorded as SMF 116 records.
- The `START TRACE` command can be used to start collecting this information. A parameter can be provided to specify which type of information should be collected.
- To enable the collection of this information automatically system parameters can be set in the zPARM using `CSQ6SYSP`. When set, data collection will commence at queue manager start up.
- SMF data is collected in memory and written out at periodic intervals. The interval MQ should use can be set using the `STATIME` system parameter. If this is set to zero WebSphere MQ uses the SMF global accounting interval.

Interpreting SMF data

- **SMF record layouts are fully documented**
  - Ship copybooks that map the records
- **SupportPac MP1B**
  - `MQSMF` displays formatted records
  - Outputs information to various files (DDs)
  - Highlights potential out-of-line conditions
  - Can output comma-separated values (CSV) to import in spreadsheets
  - `MQCMD` also available
    - Automates issuing of DISPLAY commands - outputs as CSV
    - Useful for monitoring a given channel (DIS CHSTATUS) or queue

Interpreting SMF data

N O T E S

- WebSphere MQ provides detailed information describing the SMF records it produces. These can be used to understand the data that is generated and produce utilities to interpret this information.
- The category 2 SupportPac MP1B provides a program called `MQSMF` that can be used to format the SMF records instead. This program analyses SMF records and outputs information to various files (DDs), if they are specified. In addition to formatting the data in to human-readable output, it also has support for highlighting various conditions that might warrant further attention by administrators, such as buffer pools being potentially too small.
- `MQSMF` can also output data as comma separated values (CSV) that can be readily imported in spreadsheets for further analysis.
- Another program called `MQCMD` is also provided in the SupportPac. This can be used to automate issuing DISPLAY commands on a regular basis, which facilitates monitoring of resources, such as channels or queues. The output is formatted using CSV for ease of importing elsewhere for analysis.

MQSMF

- **Example JCL**

```
//S1 EXEC PGM=MQSMF,REGION=0M
//STEPLIB DD DISP=SHR,DSN=user.MP1B.LOAD
//SMFIN  DD DISP=SHR,DSN=user.SMF.OUT
//SYSIN DD *
*  comments
SMF_Interval_time 30 * new value
Detail 20
QM MQ07
//MESSAGE DD SYSOUT=*
//BUFF    DD SYSOUT=*
//BUFFCSV DD SYSOUT=*
//CF      DD SYSOUT=*
//CFCSV   DD SYSOUT=*
//DATA    DD SYSOUT=*
//DB2     DD SYSOUT=*
//DCHS    DD SYSOUT=*
//EOJ     DD SYSOUT=*
//LOCK    DD SYSOUT=*
//LOG     DD SYSOUT=*
//LOGCSV  DD SYSOUT=*
//MSGM    DD SYSOUT=*
//MSGMCSV DD SYSOUT=*
//QCPU    DD SYSOUT=*
//SMDS    DD SYSOUT=*
//TASKSUM DD SYSOUT=*
//TASK    DD SYSOUT=*
//TASKCSV DD SYSOUT=*
//TOPIC   DD SYSOUT=*
//STG     DD SYSOUT=*
//QSUML   DD SYSOUT=*,DCB=(LRECL=200)
//QSUMS   DD SYSOUT=*,DCB=(LRECL=200)
//STGSUM  DD SYSOUT=*,DCB=(LRECL=200)
//SYSPRINT DD SYSOUT=*,DCB=(LRECL=200)
//SYSOUT   DD SYSOUT=*,DCB=(RECFM=VB,LRECL=200,BLKSIZE=27998)

//SYSERR   DD SYSOUT=*
```

Queue manager statistics

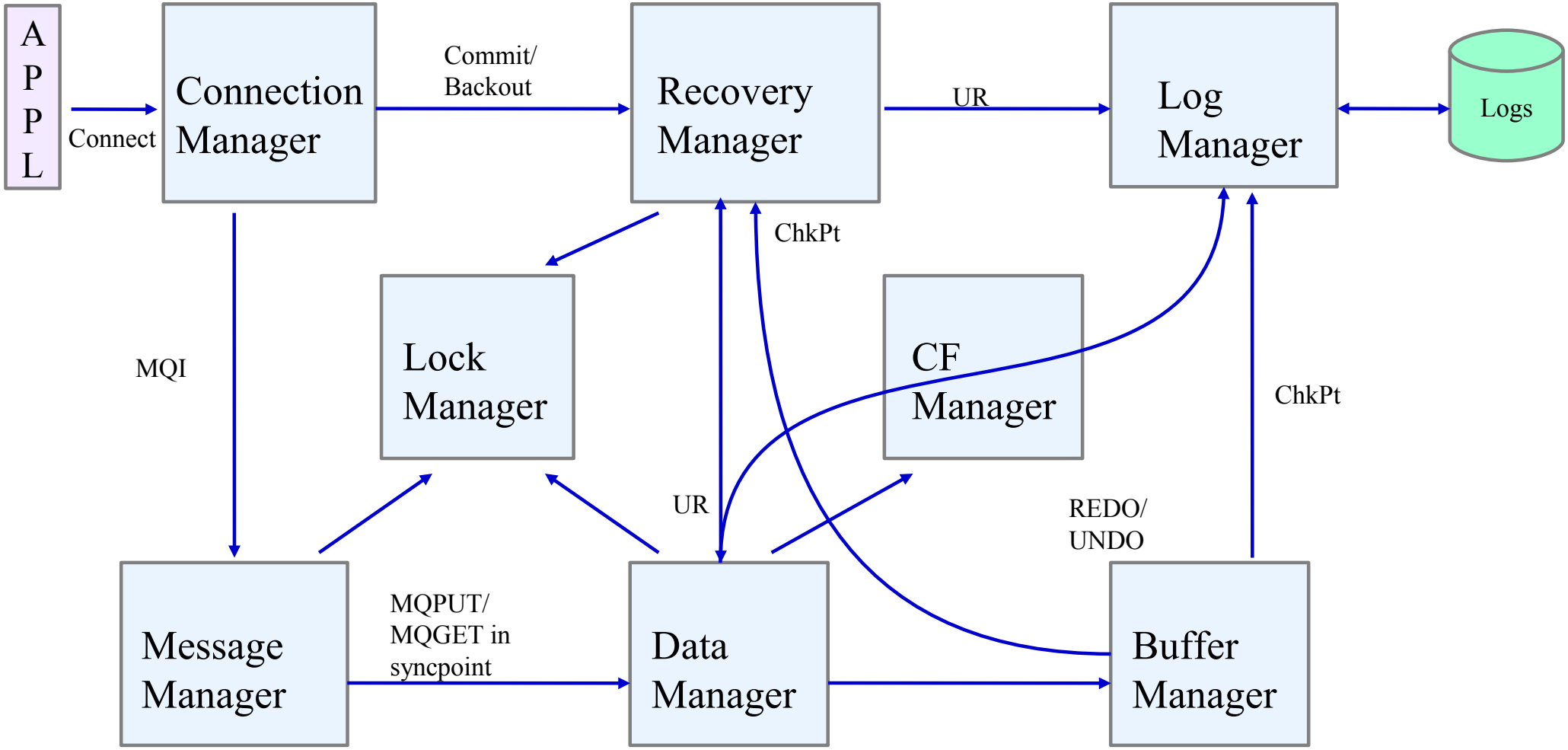
- **Provide a lot of activity related information**
  - Insignificant CPU overhead so can enable this all the time
  - Keep historical data for trend analysis
- **Easy to see high-level activity**
  - How much storage the queue manager is using
  - How many API calls are occurring
  - How much logging is occurring
  - How the buffer pools are performing
- **Problem diagnosis**
  - What is using CPU?
  - What is causing delays?

Queue manager statistics

N O T E S

- Collecting queue manager statistics trace in SMF (115 records) does not incur a significant cost, so it is helpful to enable this all the time. The data, if kept, can be used for historical trend analysis.
- The statistics collected provide an easy way to check information, such as the number of each type of MQI request, how much logging is occurring and how effectively the buffer pools are performing.
- The statistics provide a good starting point to identify the cause of performance issues. If additional detail is required accounting data can also be used, which is covered later in this presentation.
- The following slides cover some of this information in further detail…

Resource Managers within the Queue manager



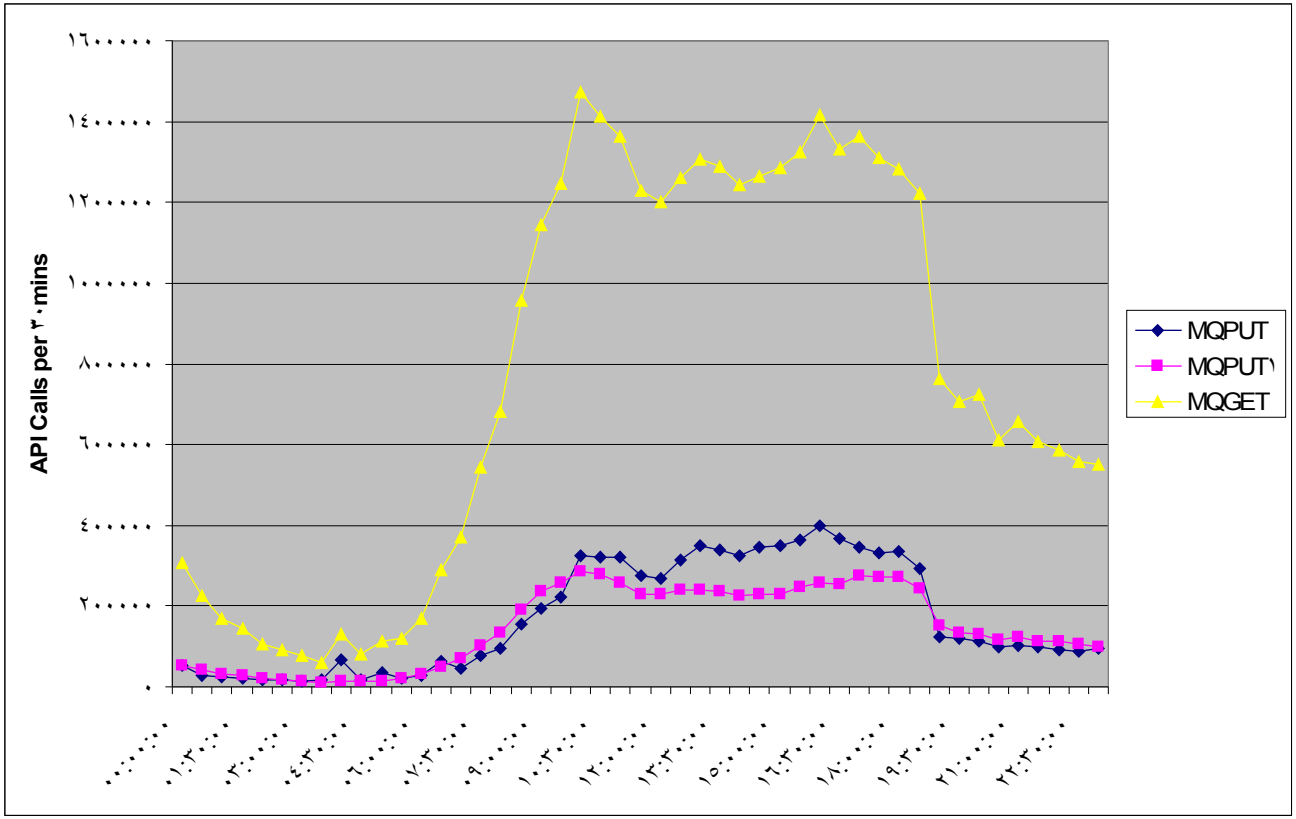
Message Manager statistics

- **Provides counts of MQI requests during SMF interval**
- **Includes both successful and failed requests**
  - MQGET count includes those where no message was available
- **Written to MSGM and MSGMCSV DD by MQSMF**

MVS	QM	Date	Time	Puts	Put1s	Gets
MVSA	MQ1A	31/03/2014	00:00:00	53979	54120	306587
MVSA	MQ1A	31/03/2014	00:30:00	27212	42914	227198
MVSA	MQ1A	31/03/2014	01:00:00	23924	31470	169326

- **Do you know what your daily/monthly profile looks like?**
- **Do you have busy hours each day, or busy days each month?**
  - What is the trend?

Do you know what your daily profile looks like?



- The above chart shows potential MQGET inefficiencies
  - Multiple application instances ‘competing’ to get the messages?
  - Need to index one or more queues?
- Can help to identify growing trends and the need for increased capacity

Log Manager statistics

- **Constraint for persistent messages is normally the rate of logging**
- **All persistent data changes are logged**
  - Mainly MQPUTs and MQGETs of persistent messages
- **Also object changes, etc.**
- **Primary reporting is on Control Intervals (CIs)**
  - 4KB pages
  - If you're using dual logging 2 CIs are written per 4KB page
- **Written to `LOG` and `LOGCSV` DD by `MQSMF`**
- **Ask the following questions:**
  - Are there I/O issues ?
    - High Persistent workload ? Too much data being written to the Logs ?
  - Checkpointing more than you should ?
    - Low LOGLOAD ? Frequent log switches ?
  - Are applications backing out more than you should ?

Log Manager statistics

N O T E S

- The performance of persistent messaging is usually constrained by the rate at which activity can be logged, so it is important to understand how logging is performing. Although the log contains mostly information about puts and gets for persistent messages, other activity is also written, such as whenever object changes occur.
- Log Manager reports statistics based on activity for control intervals (CIs) because a CI is the smallest amount of data that can be written to the log. A control interval equates to a 4KB page. If dual logging is enabled for a queue manager each control interval must be written twice – once to each copy of the log. Log Manager counts activity for each CI separately, so the counts will be double those reported for singular logging.
- The statistics reported can highlight I/O issues that are affecting performance. I/O can be constrained if the persistent workload is too great, such that the amount of data cannot be written to the log fast enough. However, logging can also be constrained by the response time for each I/O request. Frequent commits by applications require the log to be forced more often. This can also result in less efficient I/O, because the queue manager must write out control intervals before they are full.
- Checkpoints are taken whenever the active log data set is switched, or the amount of data written reaches the `LOGLOAD` threshold. Checkpoints taken too often are unnecessary and can impact performance.
- Frequent backing out by applications can also negatively impact performance because the log must be read backwards to rollback any operations in the transaction. A large number of log reads can indicate this. This can be exacerbated if the queue manager ever needs to access archive logs!

Logging overview

- **Application puts/gets persistent messages**
- **Queue Manager writes data in to log buffers**
  - May have to wait if no log buffers are available
  - Log force requests the log write task to write up to the required RBA
    - Waits for I/O to complete
- **Log write task**
  - Waits for previous I/O to complete
  - Looks for highest requested log RBA to write
  - Writes up to and including this RBA, then iterates
    - One log write can include data from many tasks
- **More data written per I/O, the longer the I/O time**
- **As more data is logged by the system an individual transaction's commit time may increase**

DASD information

- **Writing more data per I/O is more efficient than many short I/Os**
  - An I/O has set-up time and data transfer time (connect) – e.g.
    - Set up 0.1 millisecond, connect 0.1 per 4KB page
    - One page per I/O = 0.1 + 0.1 = 0.2 ms = 5000/second = 5000 pages/sec
    - 9 pages per I/O = 0.1 + 0.9 = 1.0 ms = 1000/second = 9000 pages/sec
  - Each larger I/O takes longer but the overall data rate is greater
- **Can stripe the log data set across different volumes**
  - Pages 1,5,9 to vol 1; 2,6 to vol 2; 3,7 to vol 3; 4,8 to vol 4
  - Can now perform I/O in parallel
    - Can write 3 pages to each volume faster than 9 pages to 1 volume !
- **Expect 140MB/second per log**

Logging statistics

uSec = microsec

```
Write_Wait        0, Write_Nowait     538306, Write_Force        193
Read_Stor         0, Read_Active           0, Read_Archive         0
BSDS_Reqs       703, CIs_Created       61037, BFWR                7478
ALW               6, CIs_Offload       60480, Checkpoints            0
Read delayed      0, Tape Lookahead        0, Lookahead Mount        0
Write_Susp     7430, Write_Reqs        12007, CI_Writes          64629
Write_Serl        0, Write_Thrsh          48, Buff_Pagein             0

WTB          0
TVC          0
ALR          0
 _____,__ write requests,   CIs, Average I/O , After I/O   , pages/IO
                                        uSec ,       uSec   ,
 Log 1, 1 page      7500,  7500,          312,           5,         1
 Log 1,>1 page      4507, 57129,          617,           6,        13
Standard deviation of first log, 1 page per I/O, response time +- 3
 Log 1, 1 page Longest I/O          3914 at 2014/04/10,08:55:57.429543 UTC
 Log 1, 1 page Longest Request      3918 at 2014/04/10,08:55:57.429543 UTC
 Log 1,>1 page Longest I/O         34352 at 2014/04/10,08:55:57.392493 UTC
 Log 1,>1 page Longest Request     34361 at 2014/04/10,08:55:57.392493 UTC
Log write rate       2MB/s per copy
Logger busy:
 Interval: 121 Idle: 115206421 (us)( 95%) LogIOSusp: 5193950 (us)(  4%)
```

Logging statistics

- **WTB > 0 indicates applications having to wait for buffers to be written first**
  - OUTBUFF might need to be increased
  - Alternatively, DASD might be slow
  - MQSMF highlights potential warning conditions:
    - **MQQJST09W QJST requests waiting for buffer 99**
    - **MQQJST11W QJST OK logging rate 65 MB/Sec**
- **Log read when transactions back out**
  - Log reads also expected during queue manager start-up
  - Data read from buffers – good
  - Data read from active logs – OK
  - Data read from archive logs – BAD – investigate long UOW
  - MQSMF messages:
    - **MQQJST00I QJST read log buffers from storage 3433 > 0**
    - **MQQJST01W QJST read log buffers from active logs 50838 > 0**
    - **MQQJST02S QJST read log buffers from archive logs 15 > 0**

Logging statistics

- **`Checkpoints` is only incremented when `LOGLOAD` causes checkpoint**
  - Checkpoints due to a log switch are not counted
- **`CI_Writes` shows `64629` 4KB pages have been written**
  - 2.1MB/sec average (since SMF interval is 2 minutes) = VERY LOW !
- **Singular logging because only `Log 1` is reported**
- **Look at single page I/O time**
  - Should be fairly stable
    - Multiple page I/O time sizes vary
  - 312 uSec in example is very good
    - 1000 uSec is OK – if 2000 or higher then attention is warranted
- **Check average number of pages per I/O**
  - If 50 or higher this indicates I/O is constrained by DASD speed or trying to write too much data
    - Consider splitting the workload over an additional queue manager
- **Longest page I/O took 3.9ms**
  - Useful if investigating performance issues, disk issues?

Buffer Manager statistics

- **Are the buffer pools performing well?**
  - All messages (irrespective of persistence) put to/got from private queues require buffers in a buffer pool
  - A badly configured buffer pool may:
    - Have a significant impact on performance
    - Result in storage wastage
  - Configuration of buffer pools is dependent on how they are going to be used
- **Short-lived messages**
  - Keep whole working set of messages in buffer pool
- **Long-lived messages**
  - Typically more messages than can be contained in memory
  - Will always hit disk before being got
- **Written to `BUFF` and `BUFFCSV` DD by `MQSMF`**

Buffer Manager statistics

N O T E S

- The queue manager uses buffer pools to minimize synchronous I/O activity during MQI requests. All get and put requests for private queue messages require buffers so it is important to ensure they are performing well.
- A common misconception is that buffers are only used for persistent data – this is not true. Non-persistent data may be written to page sets if there are not enough free buffers, so consider using different buffer pools for persistent and non-persistent workloads.
- If the buffer pools fill up, a lot of disk read and write activity starts to occur, which significantly impacts performance, especially as applications start to be blocked on I/O requests. If the buffer pools become 85% full the queue manager starts to actively write data out to DASD to free space in the buffer pools. Therefore, it is optimal to avoid buffer pool usage reaching this threshold.
- If messages are short-lived the system will perform best if they can be kept in the buffer pools. If there are not enough buffers applications may be blocked on synchronous I/O to retrieve them from the page set.
- If messages are long-lived it is unlikely there will be enough storage available to keep all such messages in the buffer pool. It is also likely they will be written to the page set before they are got anyway.

Buffer pool statistics

```
= BPool  1, Size  5000,%full now 80, Highest %full 92, Disk reads 27019
 > 01 Buffs  5000  Low     390  Now   979  Getp  1408819  Getn   137508
   01 Rio   27019  STW  648745  TPW 26587  WIO      2417  IMW        12
   01 DWT    1236  DMC       0  STL 72300  STLA        8  SOS         0
   01 Above the bar   PAGECLAS 4KB
```

- **No buffer available (SOS), buffer steal hash chain mod (STLA), DMC should be 0**
  - If not, consider increasing buffer pool size
- **Deferred (asynchronous) write threshold (DWT) reached 1236 times**
  - Number of 'dirty' pages >= 85% total pages
- **Synchronous write threshold (DMC) not reached above 0**
  - Number of 'dirty' pages >= 95% total pages
- **Version 8 reports if buffer pool is above the bar and whether it is page-fixed**
- **MQSMF messages output if buffer pools may be too small:**
  - **MQQPST04E BP 3 Many (735) pages read from disk**
  - **MQQPST02S BP 3 Filled many (122) times**

QPST – Bufferpool stats DSECT

NOTES

```
QPSTPOOL DS    F          Buffer Pool Number
QPSTNBUF DS    F          # of buffers in pool
QPSTCBSL DS    F          Lowest # of stealable buffers
QPSTCBS  DS    F          # of stealable buffers
QPSTGETP DS    F          # of Get Page (old) requests
QPSTGETN DS    F          # of Get Page (new) requests
QPSTRIO  DS    F          # of DASD read operations
QPSTSTW  DS    F          # of Set Write Intent requests
QPSTTPW  DS    F          # of pages written to DASD
QPSTWIO  DS    F          # of DASD write operations
QPSTIMW  DS    F          # of synchronous write operations
QPSTDWT  DS    F          # of times deferred write threshold is reached
QPSTDMC  DS    F          # of times synchronous write threshold is reached
QPSTSTL  DS    F          # of buffer steals
QPSTSTLA DS    F          # of times hash chain modified during buffer steal
QPSTSOS  DS    F          # times suspended for no stealable  buffers
QPSTFLAG DS    XL4          Flag word.                    X
                                Bit 0 on indicates buffer pool is   X
                                located above the bar. Bit 1 on     X
                                indicates that buffer pool is backedX
                                by fixed 4KB pages. Otherwise       X
                                pageable 4KB pages are used.
```

Buffer pools for short-lived messages

- **Aim to keep the full working set of messages in memory**
  - `DWT` should be 0
    - Always at least 15% buffers free
- **Pages written (`TPW`) might be greater than 0 due to checkpoint activity**
  - However, we shouldn't be writing because of buffer pool getting full!
- **Read operations (`RIO`) should be 0**
  - If we have everything in memory, we should never need to read from the pageset!
    - Unless read from a page set after the queue manager is restarted
- **Ensure there are sufficient buffers to handle your peak message rate**

Buffer pools for long-lived messages

- **Expect messages to be written to and read from the page set**
- **(`RIO+TPW`)/statistics interval is the I/O rate to page sets**
  - RIO = Number of read I/Os
  - TPW = Total number of pages written
- **During period that messages are processed (e.g. written during day and processed overnight)**
  - `RIO` should be approximately equal to `TPW`
    - Shows that one page is read for every page written
- **If `RIO` is much larger than `TPW`:**
  - Pages are being read multiple times
  - Applications may be getting by message/correlation id from non-indexed queues

Accounting data

- **Useful for monitoring application activity**
- **Can control at queue level using `ACCTQ(ON|OFF|QMGR)`**
  - `QMGR` means inherit from `ACCTQ` queue manager attribute
- **Costs 5-10% CPU overhead**
- **Heavyweight**
  - Multiple records may be cut for each transaction
  - Records created at each SMF interval for long running UOWs
  - Get a lot of useful information about work being done by applications
  - However, enabling this has been known to swamp an SMF environment
- **Task/thread and queue statistics**
- **Even though prolific:**
  - Turn on for a few minutes each month to get view of activity
  - Become familiar with the data and the usage patterns

Accounting data - application activity

N O T E S

- Application activity can be monitored using accounting trace, which is recorded in SMF 116 records. As previously explained, this can be enabled using the `START TRACE(ACCTG) CLASS(3)` command, or by using the `SMFACCT` system parameter.
- Accounting can be enabled for queues individually using the `ACCTQ` parameter, which can be set to `ON`, `OFF` or `QMGR`. If it is set to `QMGR` then the value of the `ACCTQ` queue manager attribute is used instead. The queue manager attribute can be set to `ON`, `OFF` or `NONE`. If `NONE` is specified this indicates all accounting is to be disabled, irrespective of the individual queue attributes.
- Enabling this level of trace incurs a notable performance cost, approximately a 5-10% CPU overhead, so you may not wish to enable it all the time. However, it is still beneficial to do so for a short time on a regular basis to obtain a detailed view of activity. If you become familiar with application usage patterns this will help when planning administrative changes.

Application activity

```
Open name AP01_REP_MQ20_DC_0_0007              Object type:Local Queue
Base name AP01_REP_MQ20_DC_0_0007              Base type   :Queue
Queue indexed by NONE

First opened 10-04-2014 08:52:48.62

Last closed  10-04-2014 08:52:49.69

Page set ID              4,  Buffer pool         1

Current opens            0,  Total requests        158

Generated messages :          0

Persistent messages:   GETs        37,  PUTs         0,  PUT1s        0

Put to waiting getter: PUT          0,  PUT1         0

Put direct to shared queue:         0

GETs: Valid   78,  Max size  9824,  Min size   75,  Total bytes 151197
GETs: Dest-S 156,  Dest-G       0,  Brow-S      0,  Brow-G     0, Successful destructive         78
Time on queue : Max  0.348850,  Min  0.118915, Avg   0.231072  seconds

-MQ call-      N     ET     CT    Susp    LOGW    PSET Epages    skip expire
 Open    :     1     20     20       0
 Close   :     1      6      6       0
 Get     :   156     43     38       0       0       0       0     570      0
Maximum depth encountered         154
```

Application activity

N

O

T

E

S

- For get operations, a non-zero skip value indicates we should consider indexing the queue
  - `Dest-S` shows gets for a specific message
- Single open/close for the queue with multiple gets
  - This is good!
  - Look for applications repeatedly opening and closing queues
- Can use output to identify busy queues etc
- Page set / buffer pool allocation
  - `MQSMF` can also be used to report the most active queues and identify their page set and buffer pool. This can be helpful in identifying if any contention might be occurring.

A good day versus a bad day

- **Obtain data for a good day to compare with**
  - Without this it is much harder to identify the cause of problems

Metric	Good day	Bad day
Get count	16846	3365
Get elapsed time (average)	329	2735
Get CPU time (average)	321	966
Get suspend time	0	1735
Get pageset count	0	6752
Get pageset elapsed time	0	1734
Get valid count	6087	3233
Get size (maximum)	1043924	1041564
Get size (minimum)	115196	202600
Get size (average)	847087	834928
Get browse (specific)	16846	3365
Get persistent count	6087	3233

A good day versus a bad day

N O T E S

- To help identify the cause of performance issues it is helpful to have recently collected data for a good day to refer to. This makes it much easier to identify what has changed to cause the problem. Without this information you might spend time improving something unnecessarily.
- In the example on the previous slide the increased delay in getting messages is caused by the need to access the page sets. This is caused when messages do not reside in the buffer pools, so it is likely one or more buffer pools are too small. Further analysis would be required to identify why.

Channel initiator statistics and accounting data

- **Version 8 introduces**
  - Channel initiator statistics
  - Channel accounting data
- **Useful for**
  - Monitoring
  - Capacity planning
  - Tuning
- **Previously, customers created jobs to periodically `DISPLAY CHSTATUS`**
  - But, with such monitoring data, it is difficult to:
    - Manage historical data
    - Perform capacity planning
    - Investigate channel performance issues
- **Separate controls from queue manager allows 'opt in'**
- **SupportPac MP1B updated to:**
  - Format data
  - Introduce rule based reporting

Channel initiator statistics V's Channel accounting data

- **Statistics – high level view of activity in CHINIT**
  - Information about number of channels and TCB usage
  - Dispatchers, Adapters, DNS, SSL
  - Do I need more or less dispatchers/adapters ?
  - Do I have spare capacity ?
- **Accounting – detailed view of individual channels**
  - Controlled by STATCHL attribute on Channel def and QMGR
  - What work have channels been doing ?
  - Which channels are being heavily utilised ?
- **1-2% Overhead for collecting both statistics and accounting**

Performance overhead for statistics and accounting

N

O

T

E

S

- Release specific Performance Support Pack MP1I (due out soon)
  - Indicates 1-2% CPU overhead for collecting channel statistics and accounting
  - Depends on channel load,

    e.g. compression, SSL on channels use more cpu so make accounting cost appear less

Channel initiator statistics SMF 115 sub-type 231

- **Channel initiator**
  - QSG name
  - Number of current channels
  - Maximum current channels
  - Number of active channels
  - Maximum active channels
  - Maximum TCP/IP channels
  - Maximum LU 6.2 channels
  - Storage usage in MB
- **Dispatcher task**
  - Task number (TCB address)
  - Number of requests for task
  - Busy CPU time of task
  - Sum of elapsed time of requests
  - Wait elapsed time of task
- **Adapter task**
  - Task number (TCB address)
  - Number of requests for task
  - Busy CPU time of task
  - Sum of elapsed time of requests
  - Wait elapsed time of task
- **DNS task**
  - Task number (TCB address)
  - Number of requests for task
  - Busy CPU time of task
  - Sum of elapsed time of requests
  - Wait elapsed time of task
  - Time of day of max DNS request
  - Duration time of max DNS request
- **SSL task**
  - Task number (TCB address)
  - Number of requests for task
  - Busy CPU time of task
  - Sum of elapsed time of requests
  - Wait elapsed time of task
  - Time of day of max SSL request
  - Duration of max SSL request

Channel accounting data SMF 116 sub-type 10

- **For each channel instance**
  - Channel name
  - Channel disposition
  - Channel type
  - Channel state
  - STATCHL setting
  - Connection name
  - Channel stopped date&time
  - Last msg date&time
  - Channel batch size
  - Num of messages
  - Num of persistent messages
  - Num of batches
  - Num of full batches
  - Num of transmission buffers sent
  - Num of transmission buffers received
  - Current shared conversations
  - Num of bytes
  - Num of persistent bytes
  - Num of bytes sent (both ctrl data & msg data)
  - Num of bytes received (both ctrl data & msg data)
  - Compression rate
  - Exit time average
  - Exit time min
  - Exit time max
  - Exit time max date&time
  - Net time average
  - Net time min
  - Net time max
  - Net time max date&time
  - Remote qmgr/app name
  - Put retry count
  - Transmission queue empty count

Channel initiator SMF controls

- You can start channel initiator statistics (STAT) trace by:

```
!MQ08 START TRACE(STAT) CLASS(4)
CSQW130I !MQ08 'STAT' TRACE STARTED, ASSIGNED TRACE NUMBER 05
CSQ9022I !MQ08 CSQWVCM1 ' START TRACE' NORMAL COMPLETION
```

- You can start channel accounting data (ACCTG) trace by:

```
!MQ08 START TRACE(ACCTG) CLASS(4)
CSQW130I !MQ08 'ACCTG' TRACE STARTED, ASSIGNED TRACE NUMBER 06
CSQ9022I !MQ08 CSQWVCM1 ' START TRACE' NORMAL COMPLETION
```

- You can display trace by:

```
!MQ08 DISPLAY TRACE(*)
CSQW127I !MQ08 CURRENT TRACE ACTIVITY IS -
TNO     TYPE   CLASS    DEST        USERID  RMID
02      STAT   01       SMF         *       *
05      STAT   04       SMF         *       *
06      ACCTG   04      SMF         *       *
END OF TRACE REPORT
```

- ALTER and STOP TRACE commands have also been updated

Channel initiator SMF controls

N O T E S

- START TRACE command extended to enable CHINIT SMF
  - START TRACE(STAT) **CLASS(4)**
    - New class 4 trace represents CHINIT SMF data
    - DEST(SMF) is default
    - Starts CHINIT SMF data collection
    - SMF records written at next SMF broadcast or interval
- STOP TRACE command extended to disable CHINIT SMF
  - STOP TRACE(STAT) CLASS(4), STOP TRACE(STAT), STOP TRACE(*)
    - Stops CHINIT SMF data collection
    - Writes outstanding data to SMF
- DISPLAY TRACE command modified to list CHINIT SMF trace info
  - DISPLAY TRACE(STAT) CLASS(4), DISPLAY TRACE(STAT), DISPLAY TRACE(*)
- ALTER TRACE command modified to alter CHINIT SMF trace
  - ALTER TRACE(STAT) TNO(tno_number) CLASS(4)

Channel initiator SMF controls

N

O

T

E

S

- Similarly, for Channel Accounting trace we have:
  - START TRACE(ACCTG) **CLASS(4)**
  - STOP TRACE(ACCTG) CLASS(4), STOP TRACE(ACCTG), STOP TRACE(*)
  - DISPLAY TRACE(ACCTG) CLASS(4), DISPLAY TRACE(ACCTG), DISPLAY TRACE(*)
  - ALTER TRACE(ACCTG) TNO(tno_number) CLASS(4)

Starting Chinit SMF via ZPARM

- Existing system parameters have been reused
  - SMFSTAT
    - Class 4 added for Chinit Statistics
  - SMFACCT
    - Class 4 added for Channel Accounting data
- If SMFSTAT/SMFACCT is set to 4 (or list of values including 4), the corresponding trace is started when the queue manager is started
- Chinit SMF collection starts when CHINIT is started
  - Reads trace settings to determine whether to enable CHINIT STAT and/or Channel ACCTG
- Can be disabled/re-enabled by STOP/START TRACE while CHINIT started
- STATIME
  - Used to control the stats interval for both Queue Manager and CHINIT
  - Keeps SMF data for both queue manager and chinit in sync

Using STATIME in ZPARM to control the interval of Chinit SMF

- STATIME parameter controls the SMF interval for both queue manager and channel initiator
  - So that we can keep both of them synchronized in time

- Values for STATIME
  - If zero: SMF data will be collected at the SMF broadcast of z/OS (using z/OS global SMF interval)
  - If non-zero: SMF data will be collected when the specified interval expires
  - Default value: 30 (minutes)

- How to set
  - Use SET SYSTEM command
    - Take effect immediately
    - eg. To set the SMF interval to 10 minutes: +cpf SET SYSTEM STATIME(10)
  - Modify STATIME parameter in ZPARM directly, then recompile the ZPARM module
    - Take effect at next startup of queue manager

Controlling Channel Accounting for auto-defined cluster channels

- Queue Manager attribute: **STATACLS**
  - **LOW/MEDIUM/HIGH**
    - Have the same effect
    - Enables channel accounting for auto-defined cluster sender channels
  - **OFF**
    - Disables channel accounting for auto-defined cluster sender channels
  - **QMGR**
    - Channel accounting for auto-defined cluster sender channels is controlled by the setting of the Queue Manager STATCHL attribute

New SMF record subtypes and DSECTs for CHINIT SMF

N O T E S

- New subtypes
  - SMF 115 subtype 231 (0xE7='X') for CHINIT statistics data
  - SMF 116 subtype 10 for channel accounting data

- New DSECTs shipped
  - csqdqwsx (QWSX): Self defining section for subtype 231 which consists of:
    - csqdqwhs (QWHS): Standard header
    - csqdqcct (QCCT): Definition for CHINIT statistics data
    - csqdqct (QCT_DSP/QCT_ADP/QCT_SSL/QCT_DNS): Definition for CHINIT tasks
  - csqdqws5 (QWS5): Self defining section for subtype 10 which consists of:
    - csqdqwhs (QWHS): Standard header
    - csqdqcst (QCST): Definition for channel accounting data

New console messages for CHINIT SMF

- For START/STOP TRACE
  - **CSQX126I** csect-name Channel accounting collection started
  - **CSQX127I** csect-name Channel accounting collection stopped
  - **CSQX128I** csect-name Channel initiator statistics collection started
  - **CSQX129I** csect-name Channel initiator statistics collection stopped

New console messages for CHINIT SMF

- CSQX076I is issued during CHINIT startup to report values of Queue Manager attributes STATCHL and STATACLS

```
...
22.59.05 STC13103  +CSQX074I !MQ07 CSQXGIP MONCHL=OFF, MONACLS=QMGR
22.59.05 STC13103  +CSQX075I !MQ07 CSQXGIP ADOPTMCA=ALL, ADOPTCHK=ALL
22.59.05 STC13103  +CSQX076I !MQ07 CSQXGIP STATCHL=OFF, STATACLS=QMGR
22.59.05 STC13103  +CSQX078I !MQ07 CSQXGIP IGQ=DISABLED, CHADEXIT=
22.59.05 STC13103  +CSQX079I !MQ07 CSQXGIP TRAXSTR=YES, TRAXTBL=2
...
```

New console messages for CHINIT SMF

- A new task, CSQXSMFT, is attached for CHINIT SMF
- If this task encounters an error, the following message is issued:
  - **CSQX124E** csect-name SMF task ended abnormally, RC=retcode, reason=reason
    - Reasons codes

```
CSQX_XSMF_PET_RELEASE  constant('C59592'X) /* PET Release          */
CSQX_XSMF_PET_PAUSE    constant('C59593'X) /* PET Pause            */
CSQX_XSMF_PET_ALLOCATE constant('C59594'X) /* PET Allocate         */
CSQX_XSMF_GET_STOR     constant('C59595'X) /* Get storage          */
```

    - An abend with dump is issued for the above reasons
- If other errors are encountered while processing CHINIT SMF:

CSQX122E csect-name Failed to process channel accounting, RC=retcode

CSQX123E csect-name Failed to process channel initiator statistics, RC=retcode

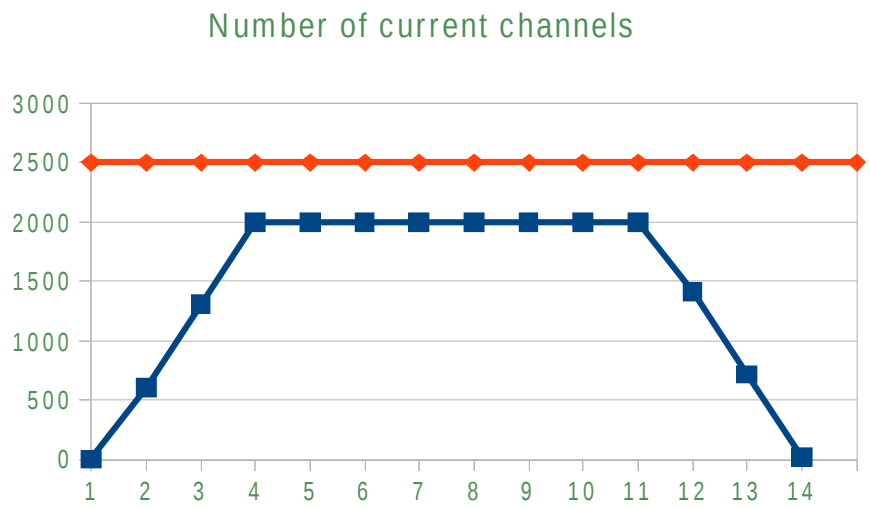
CSQX125I csect-name SMF data incomplete

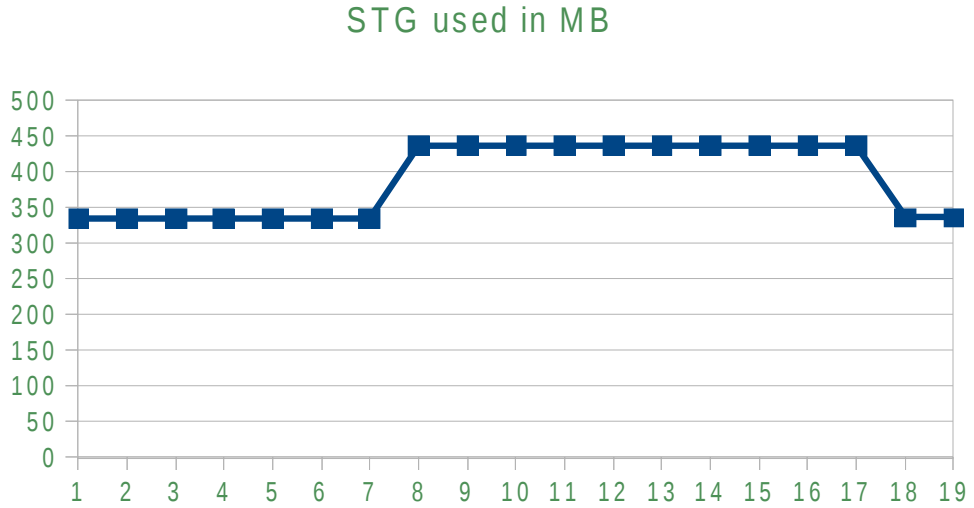
Channel initiator statistics summary

```
MV45,MQ20,2014/04/08,20:43:57,VRM:800,

From 2014/04/08,20:41:54.984681 to 2014/04/08,20:43:57.237939 duration
122.253258 seconds
Number of current channels.....................  20

Number of active channels .... ................  20

MAXCHL. Max allowed current channels.......... 602

ACTCHL. Max allowed active channels........... 602

TCPCHL. Max allowed TCP/IP channels........... 602

LU62CHL. Max allowed LU62 channels............ 602

Storage used by Chinit.........................  22MB
```

Channel initiator statistics summary

- **Number of current and active channels**
  - How close are you getting to the maximums?
- **Channel initiator storage usage**
  - 31-bit usage - not much currently in 64-bit for the channel initiator
- **Are these trending upwards?**
  - Monitor over time





Dispatcher statistics

```
MV45,MQ20,2014/04/08,20:43:57,VRM:800,
From 2014/04/08,20:41:54.984681 to 2014/04/08,20:43:57.237939 duration
122.253258 seconds
Task,Type,Requests,Busy %,     CPU used, CPU %,"avg CPU","avg ET"
    ,    ,        ,      ,      Seconds,      , uSeconds,uSeconds
   0,DISP,      46,   0.0,     0.000000,  0.0,        12,        14
   1,DISP,  168912,   1.4,     0.028218,  0.0,         8,        10
   2,DISP,  168656,   1.3,     0.025142,  0.0,         7,        10
   3,DISP,       0,   0.0,     0.000000,  0.0,         0,         0
   4,DISP,       0,   0.0,     0.000000,  0.0,         0,         0
Summ,DISP,  337614,   0.6,     0.053360,  0.0,         9,        10

   0,DISP, number of channels on this TCB,    0
   1,DISP, number of channels on this TCB,   10
   2,DISP, number of channels on this TCB,   10
   3,DISP, number of channels on this TCB,    0
   4,DISP, number of channels on this TCB,    0
Summ,DISP, number of channels on all TCBs,   20
```

Dispatcher statistics

N

O

T

E

S

- Dispatcher requests are TCP/IP send and receive
  - Few waits
  - Channel exits are run on the dispatcher though
- Average elapsed time (ET) and CPU time should be close
  - If not, this might indicate a short of CPU condition
  - Adding more dispatchers will not improve performance if you have more dispatchers than available CPs
- `Busy%` indicates if a dispatcher has spare capacity
- Channel initiator distributes work across the dispatchers
  - A channel has an affinity to a dispatcher while it is active

Adapter statistics

```
MV45,MQ20,2014/04/08,20:43:57,VRM:800,
From 2014/04/08,20:41:54.984681 to 2014/04/08,20:43:57.237939 duration
122.253258 seconds
Task,Type,Requests,Busy %,      CPU used, CPU %,"avg CPU","avg ET"
                                 Seconds,      , uSeconds,uSeconds
   0,ADAP,  127599,   16.5,    0.953615,  0.8,          7,      158
   1,ADAP,   46790,    7.6,    0.309678,  0.3,          7,      199
   2,ADAP,   13702,    3.2,    0.065380,  0.1,          5,      284
   3,ADAP,    2909,    0.7,    0.029541,  0.0,         10,      279
   4,ADAP,     395,    0.1,    0.003179,  0.0,          8,      392
   5,ADAP,      37,    0.0,    0.000241,  0.0,          7,      149
   6,ADAP,      10,    0.0,    0.000175,  0.0,         17,      111
   7,ADAP,       0,    0.0,    0.000000,  0.0,          0,        0
Summ,ADAP,  191442,    3.5,    1.361809,  0.1,          7,      179
```

Adapter statistics

N O T E S

- Each MQI request uses the first free adapter
  - Expect to see decreasing busyness
- Elapsed time can include disk waits – e.g. commit
- In the example we never used all the adapters at the same time
  - No need to add more adapters
  - Consider adding more if the last adapter has been busy

DNS statistics

```
MV45,MQ20,2014/04/08,20:41:54,VRM:800,
From 2014/04/08,20:40:07.101220 to 2014/04/08,20:41:54.984681 duration
107.883460 seconds
Task,Type,Requests,Busy %,     CPU used, CPU %,
    ,    ,        ,      ,      Seconds,      ,
   0,DNS ,      24,   0.0,     0.007980,  0.0,
Summ,DNS ,       24,   0.0,     0.007980,  0.0,



"avg CPU","avg ET",longest ,date      ,time
 uSeconds,uSeconds,uSeconds,           ,
      332,    1031,   24284,2014/04/08,20:41:49.573730
 Summ,332,    1031,   24284,2014/04/08,20:41:49.573730
```

DNS statistics

N O T E S

- There is only one DNS task
  - If this task is very busy, let IBM know!
- Longest request was 24284 microseconds
- Date and time fields show when this happened
- Message `CSQX788I` is issued if a DNS lookup takes more than 3 seconds
  - `CSQX788I csect-name DNS lookup for address address using function 'func' took n seconds`

SSL statistics

```
MV45,SS09,2014/04/10,23:22:24,VRM:800,
From 2014/04/10,22:53:26.883960 to 2014/04/10,23:22:24.204176 duration
1737.320215 seconds
Task,Type,Requests,Busy %,      CPU used, CPU %,"avg CPU","avg ET"
    ,    ,        ,      ,       Seconds,      , uSeconds,uSeconds
   0,SSL ,  109843,   0.3,      0.594580,  0.0,          5,       42,
   1,SSL ,  130180,   0.3,      0.713966,  0.0,          5,       41,
   2,SSL ,  117544,   0.3,      0.703146,  0.0,          6,       42,
   3,SSL ,  145944,   0.4,      0.830535,  0.0,          6,       43,
   4,SSL ,  123825,   0.3,      0.679656,  0.0,          5,       43,

longest ,date       ,time
uSeconds,           ,
  229638,2014/04/10,22:54:34.264949
  255082,2014/04/10,22:54:54.302855
  230501,2014/04/10,22:54:43.958105
  280241,2014/04/10,22:54:53.499979
  361212,2014/04/10,22:54:53.599940
```

Low average CPU time due to cryptographic offload

Longest busy times due to lots of channels starting together

SSL statistics

N

O

T

E

S

- CPU time expected to be less than elapsed time because cryptographic operations are offloaded
- The long busy times seen in the example were due to lots of channels starting up at the same time
- Adding more SSL tasks might not improve performance if waiting for external hardware, such as a single cryptographic card

Channel accounting data

- **Accounting (`ACCTG`) trace `CLASS(4)`**
  - Enables collection of individual channel information
- **Channels can be included/excluded from collection**
  - Channel `STATCHL` attribute for predefined channels
  - Queue manager `STATACLS` attribute for auto-defined cluster sender channels
    - Controls all auto-defined channels, not individually
  - Queue manager `STATCHL` attribute for client channels
- **Generally low cost as most channels are long lived**
- **Consider if you have a lot of short lived client connections**
- **Gives detailed information for each channel**
  - What has the channel done?
  - Which are the busy channels?

Controlling Channel Accounting

- Queue Manager attribute: **STATCHL**
  - **LOW/MEDIUM/HIGH**
    - Have the same effect
    - Enables channel accounting for channels with STATCHL(QMGR)
  - **OFF**
    - Disables channel accounting for channels with STATCHL(QMGR)
  - **NONE**
    - Disables channel accounting for all channels
- Channel attribute: **STATCHL**
  - **LOW/MEDIUM/HIGH**
    - Have the same effect
    - Enables channel accounting for this channel
  - **OFF**
    - Disables channel accounting for this channel
  - **QMGR**
    - Channel accounting is controlled by the setting of the Queue Manager STATCHL attribute

Note: For SVRCONN channels set STATCHL at the QMGR level

Channel information (1)

```
127.0.0.1  MQ89_1  Connection name                127.0.0.1
127.0.0.1  MQ89_1  Remote qmgr/app                MQ89
127.0.0.1  MQ89_1  Channel disp                  PRIVATE
127.0.0.1  MQ89_1  Channel type                  SENDER
127.0.0.1  MQ89_1  Channel status                RUNNING
127.0.0.1  MQ89_1  Channel STATCHL               HIGH
127.0.0.1  MQ89_1  Channel started date & time   2014/04/08,19:41:48
127.0.0.1  MQ89_1  Channel stopped time
127.0.0.1  MQ89_1  Channel status collect time   2014/04/08,19:43:57
127.0.0.1  MQ89_1  Last msg time                 2014/04/08,19:43:52
127.0.0.1  MQ89_1  Active for                    122 seconds
127.0.0.1  MQ89_1  Batch size                          50
127.0.0.1  MQ89_1  Messages/batch                    38.9
127.0.0.1  MQ89_1  Number of messages              2,998
127.0.0.1  MQ89_1  Number of persistent messages   1,506
127.0.0.1  MQ89_1  Number of batches                 77
127.0.0.1  MQ89_1  Number of full batches            42
127.0.0.1  MQ89_1  Number of partial batches        35
127.0.0.1  MQ89_1  Buffers sent                   3,319
127.0.0.1  MQ89_1  Buffers received                109
127.0.0.1  MQ89_1  Xmitq empty count               13
```

Channel information (2)

127.0.0.1	MQ89_1	Message data	17,198,653	16 MB
127.0.0.1	MQ89_1	Persistent message data	4,251,780	4 MB
127.0.0.1	MQ89_1	Non persistent message data	12,946,873	12 MB
127.0.0.1	MQ89_1	Total bytes sent	17,200,221	16 MB
127.0.0.1	MQ89_1	Total bytes received	3,052	2 KB
127.0.0.1	MQ89_1	Bytes received/Batch	39	39 B
127.0.0.1	MQ89_1	Bytes sent/Batch	223,379	218 KB
127.0.0.1	MQ89_1	Batches/Second	0	
127.0.0.1	MQ89_1	Bytes received/message	1	1 B
127.0.0.1	MQ89_1	Bytes sent/message	5,737	5 KB
127.0.0.1	MQ89_1	Bytes received/second	25	25 B/sec
127.0.0.1	MQ89_1	Bytes sent/second	140,985	137 KB/sec
127.0.0.1	MQ89_1	Compression rate	0	
127.0.0.1	MQ89_1	Exit time average	0	uSec
127.0.0.1	MQ89_1	DNS resolution time	0	uSec
127.0.0.1	MQ89_1	Net time average	312	uSec
127.0.0.1	MQ89_1	Net time min	43	uSec
127.0.0.1	MQ89_1	Net time max	4,998	uSec
127.0.0.1	MQ89_1	Net time max date&time	2014/04/08,19:43:52	

Channel information

N O T E S

- Uniquely identifies each channel with its connection name, channel name and remote queue manager name
- Some of the batches were not full
  - `BATCHSZ`, `BATCHLIM` and message arrival impacts this
- About half the messages sent were persistent
- Total message data of about 16MB sent during the interval
- The average message size was about 5KB
- Bytes sent/received per second
  - Average over interval
- Monitor channel usage over time to look for trends

Questions & Answers