17044 - MQ for z/OS

But Wait, There's More MQ SMF Data Now?!?! - Monitoring your Channels Using V8's New Chinit SMF Data

Wednesday March 4th 2015
4:30 – 5:30 PM
Seneca, Sheraton Seattle
Mayur Raja (mayur_raja@uk.ibm.com)
Please Note

IBM’s statements regarding its plans, directions, and intent are subject to change or withdrawal without notice, at IBM’s sole discretion.

Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision.

The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract. The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.

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.
Agenda

- CHINIT SMF
  - Channel Initiator Statistics
  - Channel Accounting Data

New in V8
CHINIT SMF: The Problem

- Prior to MQ v8.0, there was limited SMF data for channels
- With CLASS(3) ACCOUNTING trace:

```
START TRACE(ACCTG) DEST(SMF) CLASS(3)
```

You get the Task/Thread Identification (WTID) SMF 116 Subtype 1 record which gives you data about the Sender or Receiver Message Channel Agent thread:

```
        ==> New task record found           <==========
    == Thread type.............> MOVER
    == Connection name.........> QML4CHIN
    == Operator ID.............> MQUSER
    == User ID.................> MQUSER
    == Channel name............> QSGM.OUT
    == Chl connection.........> 1.2.3.43
    == Correlator ID...........> 
    == Correlator ID.....(HEX)> 243D000E7E75C5C243D2C0
    == Context token..........>
    == Context token.....(HEX)> 00000000000000000000000000000000
    == NID.......................> QML4CHING W È€
    == NID......................(HEX)> D4D8D7C5C3C8C9D5C71A1DE63B749E08
    == Accounting token.......>
```

Complete your session evaluations online at www.SHARE.org/Seattle-Eval
CHINIT SMF: The Problem

- So, prior to MQ v8.0, there was no useful, detailed data for:
  - CHINIT address space
  - Channel activity
- Many customers have had to create their own ‘monitoring’ jobs
  - Issue periodic `DISPLAY CHSTATUS` commands
  - Use MQCMD program from Supportpac MP1B to do this
- Difficult to:
  - Monitor activity in the CHINIT address space
  - Investigate performance issues and tune for better performance
  - Perform capacity planning
  - Manage historical data
CHINIT SMF: The Problem

- The category 2 SupportPac MP1B provides a program called **MQCMD** which can be used to automate issuing DISPLAY commands on a regular basis, which facilitates monitoring of channels. The output is formatted using Comma Separated Values (CSVs) for ease of importing into a spreadsheet for analysis.
CHINIT SMF: The Solution

- Additional SMF data for the CHINIT address space
  - **Channel Initiator Statistics**
    - High level view of activity in the CHINIT address space
      - Data about Dispatcher tasks
        - Number of channels running, TCB usage
      - Data about Adapter, DNS and SSL tasks
    - Useful to determine:
      - Do tasks have any spare capacity?
      - Do you need more or less dispatcher or adapter tasks?
  - **Channel Accounting Data**
    - Detailed view of individual channels
      - What work are the channels doing?
      - Which channels are heavily utilised?
Channel Initiator Statistics

- **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

- 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

Complete your session evaluations online at www.SHARE.org/Seattle-Eval
New SMF record subtypes and DSECTs

- **New subtypes**
  - SMF 115 subtype 231 (0xE7='X') for Channel Initiator Statistics
  - SMF 116 subtype 10 for Channel Accounting Data

- **New DSECTs shipped**
  - **CSQDQWH$S (QWHS):** Standard header
  - **CSQDQWS$X (QWSX):** Self defining section for subtype 231
  - **CSQDQCCT (QCCT):** Definition for CHINIT statistics data
    - **CSQDQCT (QCT_DSP/QCT_AD$P/QCT_SSL/QCT_DNS):** Definition for CHINIT tasks
  - **CSQDQWH$S (QWHS):** Standard header
  - **CSQDQWS5 (QWS5):** Self defining section for subtype 10
  - **CSQDQCST (QCST):** Definition for channel accounting data
Two new SMF records have been added:

**SMF 115 sub type 231** has the CHINIT control information like adapter and dispatcher task CPU times, DNS resolution times. This helps with tuning the number of tasks configured.

**SMF 116 sub type 10** has the per channel accounting data like bytes sent, achieved batch size, etc.

The DSECTs that are shipped for each type of record are listed.

**Note:** The standard layout for SMF records involves three parts:

- **SMF header** - Provides format, identification, and time and date information about the record itself.
- **Self-defining section** - Defines the location and size of the individual data records within the SMF record.
- **Data records** - The actual data from MQ that you want to analyze.

**QCCT** = Chinit Control (now called Chinit Statistics)

**QCST** = Channel Statistics (now called Channel Accounting)
Starting CHINIT SMF via CSQZPARM

- Separate controls from Queue Manager SMF - *allows 'opt in'*
- Existing system *(CSQ6SYSP)* 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)
  - Corresponding trace is started when the queue manager is started
- CHINIT SMF collection starts when CHINIT is started
  - Reads trace settings and enables CHINIT STAT and/or Channel ACCTG
- Can be disabled/re-enabled by **STOP/START TRACE** while CHINIT started

Complete your session evaluations online at www.SHARE.org/Seattle-Eval
CHINIT SMF: Controls

- STAT trace allows a high level view of activity in the CHINIT address space.
- ACCTG trace allows a detailed view at the channel level.
Starting CHINIT SMF via MQSC commands

• You can also 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
```

• And 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
Starting CHINIT SMF via MQSC commands

- **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)**
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)
New console messages for CHINIT SMF

- For **START/STOP TRACE(STAT)**
  
  CSQX128I csect-name Channel initiator statistics collection started
  
  CSQX129I csect-name Channel initiator statistics collection stopped

- For **START/STOP TRACE(ACCTG)**
  
  CSQX126I csect-name Channel accounting collection started
  
  CSQX127I csect-name Channel accounting collection stopped
Controlling the CHINIT SMF interval

- Use the **STATIME** parameter
  - Controls the SMF interval for both Queue Manager and CHINIT
  - Keeps both Queue Manager and CHINIT statistics synchronized in time
- Values for **STATIME**
  - **30 (minutes): Default value**
  - **Non-zero**: SMF data will be collected when the specified interval expires
  - **Zero**: SMF data is collected at SMF broadcast of z/OS (using global SMF interval)
- How to set
  - Modify STATIME parameter in **CSQ6SYSP** and rebuild the **CSQZPARM** module
    - Takes effect at next startup of Queue Manager
  - Use **SET SYSTEM** command
    - Takes effect immediately

  e.g. To set the SMF interval to 10 minutes, issue:

  ```
  /!MQ07 SET SYSTEM STATIME(10)
  ```
Controlling Channel Accounting

• Queue Manager attribute: **STATCHL**
  - **OFF** (default value)
    - Disables channel accounting for channels with STATCHL(QMGR)
  - **LOW/MEDIUM/HIGH**
    - Have the same effect
    - Enables channel accounting for channels with STATCHL(QMGR)
  - **NONE**
    - Disables channel accounting for all channels

• Channel attribute: **STATCHL**
  - **QMGR** (default value)
    - Channel accounting is controlled by the setting of the Queue Manager STATCHL attribute
  - **LOW/MEDIUM/HIGH**
    - Have the same effect
    - Enables channel accounting for this channel
  - **OFF**
    - Disables channel accounting for this channel
A new attribute called STATCHL which allows statistics collection granularity at the channel level has been added to the channel definition.

It can be specified on Sender, Receiver, Server, Requester, Cluster Sender and Cluster Receiver channels.

STATCHL can also be specified at the Queue Manager level to allow channels to inherit a system wide setting.

The amount of data collected is a superset of that collected on the distributed platforms with the STATCHL event message.

The queue manager object also has a STATACLs which sets the STATCHL value for automatically defined cluster sender channels.
Channel Accounting for auto-defined cluster channels

- Queue Manager attribute: **STATACLs**
  - **QMGR** (default)
    - Channel accounting for auto-defined cluster sender channels is controlled by the setting of the Queue Manager **STATACL** attribute
  - **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
The queue manager object also has a STATACLS which sets the STATCHL value for automatically defined cluster sender channels.
Channel Accounting for SVRCONN channels

- For SVRCONN channels
  - Set **STATCHL** at the QMGR level
- Enables it for all client connections
- But, be careful as channel accounting data is captured at:
  - Each SMF statistics interval (STATIME), and
  - When a channel ends data is captured and held until next interval
  - Hence, frequent client connects/disconnects can result in a lot of data!
- We may provide more control for SVRCONN channels in a future release
MQ Explorer - Enabling Channel Statistics on QMGR

Complete your session evaluations online at www.SHARE.org/Seattle-Eval
MQ Explorer - Enabling Channel Statistics on channel

Complete your session evaluations online at www.SHARE.org/Seattle-Eval
New console messages for CHINIT SMF

• **CSQX076I**
  • Issued during CHINIT startup
  • Reports 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
...```

Complete your session evaluations online at www.SHARE.org/Seattle-Eval
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

  • An abend (with a dump) is issued

• 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
If the MEMLIMIT parameter is not set in the channel initiator JCL, the amount of virtual storage above the bar may be set from by the MEMLIMIT parameter in the SMFPRMxx member of SYS1.PARMLIB or from the IEFUSI exit.

If the MEMLIMIT is set to restrict the above bar storage below the required level, the channel initiator will issue the **CSQX124E “SMF task ended abnormally”** message and class 4 accounting and statistics trace will not be available.
Interpreting SMF data

• Details of new SMF records are documented in the InfoCenter
  – Copybooks that map the records are shipped

• SupportPac MP1B has been updated to:
  – Format new SMF data
  – 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

• Sample program CSQ4SMFD.C (run by CSQ4SMFJ.JCL) has been updated
  – Formats CHINIT SMF data in a dump like fashion
Interpreting SMF data

- 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 into human-readable output, it also has support for highlighting various conditions that might warrant further attention by administrators.

- MQSMF can also output data as comma separated values (CSV) that can be readily imported into spreadsheets for further 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=*  
//EOJ DD SYSOUT=*  
//LOG DD SYSOUT=*  
//LOGCSV DD SYSOUT=*  
//MSGM DD SYSOUT=*  
//MSGMCSV DD SYSOUT=*  
//OCPU DD SYSOUT=*  
//SMDS DD SYSOUT=*  
//TASKSUM DD SYSOUT=*  
//TASK DD SYSOUT=*  
//TASKCSV DD SYSOUT=*  
//TOPIC DD SYSOUT=*  
//STG DD SYSOUT=*  
//OSUML DD SYSOUT=*,DCB=(LRECL=200)  
//OSUMS 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=*  

NEW DD cards

//CHINIT DD SYSOUT=*  
//CHINCSV DD SYSOUT=*  
//CMESSAGE DD SYSOUT=*  
//ADAP DD SYSOUT=*  
//ADAPCSV DD SYSOUT=*  
//DISP DD SYSOUT=*  
//DISPCSV DD SYSOUT=*  
//DNS DD SYSOUT=*  
//DNSCSV DD SYSOUT=*  
//SSL DD SYSOUT=*  
//SSLCSV DD SYSOUT=*  
//DCHS DD SYSOUT=*  
//DCHSCSV DD SYSOUT=*  
//DCHSSUM DD SYSOUT=*
CHINIT Statistics Summary (///CHINIT)

MVCA,MQPV,2014/03/18,13:00:00,VRM:800,
From 2014/03/18,12:45:00.015222 to 2014/03/18,13:00:00.083630 duration 900.068408 seconds
Peak number used of current channels........... 4
Peak number used of active channels ............ 0
MAXCHL. Max allowed current channels............9999
ACTCHL. Max allowed active channels.............9999
TCPCHL. Max allowed TCP/IP channels............9999
LU62CHL. Max allowed LU62 channels..............200
Storage used by Chinit........................... 436MB
The next few slides show output of the CHINIT's SMF data, this has been formatted by supportpac MP1B - other formatters are available.

The output is taken from one of our test systems.

On this slide, the CHINIT summary data produced by the //CHINIT DD card is shown.

This CHINIT has peaked at 4 current and the address space is using 436MB of storage.

Notes:

1) A current channel is "active" unless it is in RETRYING, STOPPED, or STARTING state.

2) A channel is "current" if it is in any state other than inactive.
CHINIT Statistics Summary (//CHINITCSV)

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

Complete your session evaluations online at www.SHARE.org/Seattle-Eval
Dispatcher Task Statistics (/DISP + /DISPCSV)

<table>
<thead>
<tr>
<th>Task, Type, Requests, Busy %,</th>
<th>CPU used, CPU %, &quot;avg CPU&quot;, &quot;avg ET&quot; Seconds, uSeconds, uSeconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>0, DISP, 26587, 0.4,</td>
<td>0.592463, 0.1, 22, 127</td>
</tr>
<tr>
<td>1, DISP, 26963, 0.3,</td>
<td>0.588092, 0.1, 22, 112</td>
</tr>
<tr>
<td>2, DISP, 864329, 2.7,</td>
<td>2.545668, 0.3, 3, 28</td>
</tr>
<tr>
<td>3, DISP, 26875, 0.4,</td>
<td>0.590825, 0.1, 22, 120</td>
</tr>
<tr>
<td>4, DISP, 26874, 0.4,</td>
<td>0.603285, 0.1, 22, 123</td>
</tr>
<tr>
<td>Summ, DISP, 971628, 0.8,</td>
<td>4.920332, 0.1, 5, 38</td>
</tr>
</tbody>
</table>

5 Dispatcher tasks
Dispatcher 2 is busy, other tasks are less busy as some channels against them have stopped
Dispatchers have ample capacity
4.9 secs of CPU time used by Dispatcher tasks
Average CPU and Elapsed Times for Dispatcher requests

Complete your session evaluations online at www.SHARE.org/Seattle-Eval
The example data shows that there are five dispatcher tasks (0 → 4) and that one dispatcher task is processing more requests than the others. This is normal, as some channels might stop so the dispatcher is processing fewer channels. Also, some channels can be busier than others.

- 4.9 seconds of CPU were used by the dispatchers.
- Dispatcher requests are generally TCP send and receive requests and channel exit requests. The average request used 5 microseconds of CPU and took 38 microseconds elapsed time.
- This report also shows the average time per request. The average CPU used per request depends on the message traffic, for example, bigger messages use more CPU than smaller messages.
- The %Busy indicates if a dispatcher has spare capacity so this report would help an MQ administrator work out if there are enough dispatchers.
Dispatcher Task Statistics (//DISP + //DISPCSV)

- Dispatcher tasks:
  - Dispatcher 7 has 1 channel using ZLIBFAST for compression
  - Dispatcher 8 has 1 channel using ZLIBFAST for decompression

<table>
<thead>
<tr>
<th>Task, Type</th>
<th>Requests</th>
<th>Busy %</th>
<th>CPU used</th>
<th>CPU %</th>
<th>avg CPU</th>
<th>avg ET</th>
<th>Seconds</th>
<th>uSeconds</th>
<th>uSeconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>7, DISP</td>
<td>1443847</td>
<td>71.6</td>
<td>19.130398</td>
<td>31.9</td>
<td>13</td>
<td>30</td>
<td>13</td>
<td>30</td>
<td>13</td>
</tr>
<tr>
<td>8, DISP</td>
<td>1431899</td>
<td>20.4</td>
<td>12.655084</td>
<td>21.1</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
</tr>
</tbody>
</table>

17 microsecs spent waiting for hardware compression (since compression was offloaded to a zEDC card)
Dispatcher Task Statistics (///DISP + ///DISPCSV)

- Dispatcher tasks:
  - Dispatcher 7 has 1 channel using ZLIBHIGH for compression
  - Dispatcher 8 has 1 channel using ZLIBHIGH for decompression

<table>
<thead>
<tr>
<th>Task, Type, Requests, Busy %,</th>
<th>CPU used, CPU %, &quot;avg CPU&quot;, &quot;avg ET&quot; Seconds, uSeconds, uSeconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>7, DISP, 146303, 98.5,</td>
<td>58.729109, 97.9, 401, 404</td>
</tr>
<tr>
<td>8, DISP, 147030, 25.7,</td>
<td>15.514522, 25.9, 106, 105</td>
</tr>
</tbody>
</table>

98.5% Busy!!

- Dispatcher tasks:
  - Dispatcher 7 has 1 channel using ZLIBFAST for compression
  - Dispatcher 8 has 1 channel using ZLIBFAST for decompression

<table>
<thead>
<tr>
<th>Task, Type, Requests, Busy %,</th>
<th>CPU used, CPU %, &quot;avg CPU&quot;, &quot;avg ET&quot; Seconds, uSeconds, uSeconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>7, DISP, 1443847, 71.6,</td>
<td>19.130398, 31.9, 13, 30</td>
</tr>
<tr>
<td>8, DISP, 1431899, 20.4,</td>
<td>12.655084, 21.1, 9, 9</td>
</tr>
</tbody>
</table>

17 microsecs spent waiting for zEDC hardware compression

But, overall cost much cheaper than ZLIBHIGH!!

Complete your session evaluations online at www.SHARE.org/Seattle-Eval
Dispatcher Task Statistics (//DISP + //DISPCSV)

- Shows distribution of channels across dispatchers

<table>
<thead>
<tr>
<th>DISP</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>Summ</td>
<td>20</td>
</tr>
</tbody>
</table>
Adapter Task Statistics (\//ADAP + \//ADAPCSV)

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

<table>
<thead>
<tr>
<th>Task, Type, Requests, Busy %,</th>
<th>CPU used, CPU %, &quot;avg CPU&quot;, &quot;avg ET&quot; Seconds, uSeconds, uSeconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>0, ADAP, 127599, 16.5,</td>
<td>0.953615, 0.8, 7, 158</td>
</tr>
<tr>
<td>1, ADAP, 46790, 7.6,</td>
<td>0.309678, 0.3, 7, 199</td>
</tr>
<tr>
<td>2, ADAP, 13702, 3.2,</td>
<td>0.065380, 0.1, 5, 284</td>
</tr>
<tr>
<td>3, ADAP, 2909, 0.7,</td>
<td>0.029541, 0.0, 10, 279</td>
</tr>
<tr>
<td>4, ADAP, 395, 0.1,</td>
<td>0.003179, 0.0, 8, 392</td>
</tr>
<tr>
<td>5, ADAP, 37, 0.0,</td>
<td>0.000241, 0.0, 7, 149</td>
</tr>
<tr>
<td>6, ADAP, 10, 0.0,</td>
<td>0.000175, 0.0, 17, 111</td>
</tr>
<tr>
<td>7, ADAP, 0, 0.0,</td>
<td>0.000000, 0.0, 0, 0</td>
</tr>
<tr>
<td>Summ, ADAP, 191442, 3.5,</td>
<td>1.361809, 0.1, 7, 179</td>
</tr>
</tbody>
</table>

MQI requests are processed by first free adapter so adapters lower in the list process less requests.

Difference could indicate wait for I/O due to commit or disk read.

Complete your session evaluations online at www.SHARE.org/Seattle-Eval
This shows an example of the adapter task statistics report.

The adapters process MQI requests. Each MQI request uses the first free adapter so expect to see decreasing busyness.

Some of these requests might wait, for example, for log I/O during a commit, so the difference between the average CPU time and average Elapsed Time per request can be quite large.

This is the report that an MQ administrator would use to ensure that there are enough adapter tasks defined. A channel should not generally need to wait for an adapter.

In this example, we never used all the adapters. So, there is no need to add more adapters. If the last adapter is very busy, consider increasing the number of adapter tasks.
DNS Task Statistics (//DNS +//DNSCSV)

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 %,
     ,   ,     ,      ,      ,     ,
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

Only 1 DNS task, not busy
Longest DNS resolution request
DNS Task Statistics (/\DNS + //DNCSV)

- 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** issued if DNS lookup takes >3 secs

**CSQX788I** csect-name DNS lookup for address *address* using function 'func' took *n* seconds
SSL Task Statistics (//SSL + //SSLCSV)

MV45,SS09,2014/04/10,23:22:24,VRM:800,
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
and high elapsed time
due to cryptographic off-load
to card

Longest busy times due to lots of
channels starting together

Complete your session evaluations online at www.SHARE.org/Seattle-Eval
SSL Task Statistics (//SSL + //SSLCSV)

- 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 (/DCHS + /DCHSCSV)

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

Complete your session evaluations online at www.SHARE.org/Seattle-Eval
Channel Accounting Data (//DCHS + //DCHSCSV)

- Uniquely identifies each channel with its connection name, channel name and remote queue manager name
- Some of the batches were not full. Target batch size was 50 but average achieved batch size was 38.9. The number of full and partial batches are shown.
  - BATCHSZ, BATCHLIM and message arrival impacts this
- About half the messages sent were persistent
### Channel Accounting Data (//DCHS + //DCHSCSV)

<table>
<thead>
<tr>
<th>IP Address</th>
<th>Queue</th>
<th>Type Description</th>
<th>Bytes Sent/Received</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>127.0.0.1</td>
<td>MQ89_1</td>
<td>Message data</td>
<td>17,198,653</td>
<td>16 MB</td>
</tr>
<tr>
<td>127.0.0.1</td>
<td>MQ89_1</td>
<td>Persistent message data</td>
<td>4,251,780</td>
<td>4 MB</td>
</tr>
<tr>
<td>127.0.0.1</td>
<td>MQ89_1</td>
<td>Non persistent message data</td>
<td>12,946,873</td>
<td>12 MB</td>
</tr>
<tr>
<td>127.0.0.1</td>
<td>MQ89_1</td>
<td>Total bytes sent</td>
<td>17,200,221</td>
<td>16 MB</td>
</tr>
<tr>
<td>127.0.0.1</td>
<td>MQ89_1</td>
<td>Total bytes received</td>
<td>3,052</td>
<td>2 KB</td>
</tr>
<tr>
<td>127.0.0.1</td>
<td>MQ89_1</td>
<td>Bytes received/Batch</td>
<td>39</td>
<td>39 B</td>
</tr>
<tr>
<td>127.0.0.1</td>
<td>MQ89_1</td>
<td>Bytes sent/Batch</td>
<td>223,379</td>
<td>218 KB</td>
</tr>
<tr>
<td>127.0.0.1</td>
<td>MQ89_1</td>
<td>Batches/Second</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>127.0.0.1</td>
<td>MQ89_1</td>
<td>Bytes received/message</td>
<td>1</td>
<td>1 B</td>
</tr>
<tr>
<td>127.0.0.1</td>
<td>MQ89_1</td>
<td>Bytes sent/message</td>
<td>5,737</td>
<td>5 KB</td>
</tr>
<tr>
<td>127.0.0.1</td>
<td>MQ89_1</td>
<td>Bytes received/second</td>
<td>25</td>
<td>25 B/sec</td>
</tr>
<tr>
<td>127.0.0.1</td>
<td>MQ89_1</td>
<td>Bytes sent/second</td>
<td>140,985</td>
<td>137 KB/sec</td>
</tr>
<tr>
<td>127.0.0.1</td>
<td>MQ89_1</td>
<td>Compression rate</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>127.0.0.1</td>
<td>MQ89_1</td>
<td>Exit time average</td>
<td>0</td>
<td>0 uSec</td>
</tr>
<tr>
<td>127.0.0.1</td>
<td>MQ89_1</td>
<td>DNS resolution time</td>
<td>0</td>
<td>0 uSec</td>
</tr>
<tr>
<td>127.0.0.1</td>
<td>MQ89_1</td>
<td>Net time average</td>
<td>312</td>
<td>312 uSec</td>
</tr>
<tr>
<td>127.0.0.1</td>
<td>MQ89_1</td>
<td>Net time min</td>
<td>43</td>
<td>43 uSec</td>
</tr>
<tr>
<td>127.0.0.1</td>
<td>MQ89_1</td>
<td>Net time max</td>
<td>4,998</td>
<td>4,998 uSec</td>
</tr>
<tr>
<td>127.0.0.1</td>
<td>MQ89_1</td>
<td>Net time max date&amp;time</td>
<td>2014/04/08,19:43:52</td>
<td></td>
</tr>
</tbody>
</table>
Channel Accounting Data (//DCHS + //DCHSCSV)

- Total message data of about 16MB sent during the interval
- The average number of bytes sent per message was about 5KB
- Bytes sent/received per second is:
  - Average/interval
- As this is a sender type channel, as expected, the bytes sent is greater than the bytes received
- Monitor channel usage over time to look for trends
Channel Accounting Summary (//DCHSSUM)

MVS, MQ, date, time, VRM, channelType, count, Persistent, NonPersistent, 'P/Sec', 'NP/Sec'
MVCA, MQPV, 2014/06/30, 11:30:00, VRM:800, RECEIVER, 2, 75720, 0, 3786, 0
MVCA, MQPV, 2014/06/30, 11:30:00, VRM:800, total, 2, 75720, 0, 3786, 0
MVCA, MQPH, 2014/06/30, 11:30:00, VRM:800, SENDER, 2, 75720, 0, 2611, 0
MVCA, MQPH, 2014/06/30, 11:30:00, VRM:800, total, 2, 75720, 0, 2611, 0
MVCA, MQPH, 2014/06/30, 11:34:04, VRM:800, SENDER, 23, 86237508, 0, 559983, 0
MVCA, MQPH, 2014/06/30, 11:34:04, VRM:800, total, 23, 86237508, 0, 559983, 0

Sender channel activity
Shown over 2 intervals
Number of persistent and non-persistent messages processed during interval
CHINIT Messages (//CMESSAGE)

- Some Examples:

**MQCHIN001W** The high water mark of the number of active channels >50 % of max channels

**MQCHIN007I** Dispatcher task is nn% busy on average

**MQCHIN008I** Adapter task is nn% busy on average

**MQCHIN009I** SSL task is nn% busy on average

- There are more examples in the documentation for Supportpac MP1B
Performance overhead for statistics and accounting

- An MQ V8 Channel Initiator allocates approximately **190MB of above the bar** virtual storage for Channel Initiator Statistics and Channel Accounting Data, regardless of whether CLASS(4) trace is enabled.

- Recommend Channel Initiator is allowed access to a minimum of 256MB of virtual storage i.e. set `MEMLIMIT=256M` if CLASS(4) trace is enabled.

- Release specific Performance Support Pack MP1J
  - Indicates **1-2% CPU overhead** for collecting CHINIT statistics and Channel accounting data
Performance overhead for statistics and accounting

If the MEMLIMIT parameter is not set in the channel initiator JCL, the amount of virtual storage above the bar may be set from by the MEMLIMIT parameter in the SMFPRMxx member of SYS1.PARMLIB or from the IEFUSI exit.

If the MEMLIMIT is set to restrict the above bar storage below the required level, the channel initiator will issue the CSQX124E “SMF task ended abnormally” message and class 4 accounting and statistics trace will not be available.
If your CHINIT is experiencing high CPU usage

• Check the CURDEPTH of your SYSTEM.CHANNEL.SYNCQ
  • If >1000, check that the Queue has INDEXTYPE(MSGID) set
• See presentation from L2 at:
  
  http://www-01.ibm.com/support/docview.wss?uid=swg27010914&aid=1

It lists some things to try for high CPU usage in the CHINIT

• Check that you have enough Adapter tasks
  • See performance tuning recommendations for the CHINIT in Performance Supportpac MP16
• Check your MAXCHL parameter as this can influence the distribution of channels to dispatchers
  • See MP16
• Check the number of dispatchers you have defined
  • The first \( \min(\frac{\text{MAXCHL}}{\text{CHIDISPS}}, 10) \) channels to start are associated with the first dispatcher TCB and so on until all dispatcher TCBs are in use. The effect of this for small numbers of channels and a large MAXCHL is that channels are NOT evenly distributed across dispatchers.

  We suggest setting MAXCHL to the number of channels actually to be used where this is a small fixed number.
  • We suggest CHIDISPS(20) for systems with more than 100 channels. We have seen no significant disadvantage in having CHIDISPS(20) where this is more dispatcher TCBs than necessary.
  • See MP16
New Redbook covers MQ V8

IBM MQ V8 Features and Enhancements

Maximize your investment in IBM MQ

Discover new features that bring value to your business

Learn from scenarios with sample configurations

ibm.com/redbooks

Complete your session evaluations online at www.SHARE.org/Seattle-Eval

05/03/15
This redbook covers the new features introduced in V8 that we have just discussed. The book is currently available in draft form. The final version is expected to be made available soon.
Additional Information

• See MQ Performance Supportpacs:

  – MP16: WebSphere MQ for z/OS - Capacity planning & tuning
  – MP1J: WebSphere MQ v8.0 for z/OS Performance Report
  – MP1B: WebSphere MQ Interpreting accounting and statistics data, and other utilities
Questions?

Complete your session evaluations online at www.SHARE.org/Seattle-Eval
This was session #17044. The rest of the week ......

<table>
<thead>
<tr>
<th>Monday</th>
<th>Tuesday</th>
<th>Wednesday</th>
<th>Thursday</th>
<th>Friday</th>
</tr>
</thead>
<tbody>
<tr>
<td>08:30</td>
<td></td>
<td>17060: Understanding MQ Deployment Choices and Use Cases</td>
<td>17051: Application Programming with MQ Verbs [z/OS &amp; Distributed]</td>
<td>16544: Why Shouldn't I Be Able To Open This Queue? MQ and CICS Security Topics</td>
</tr>
<tr>
<td>10:00</td>
<td></td>
<td>17036: Introduction to MQ - Can MQ Really Make My Life Easier? [z/OS &amp; Distributed]</td>
<td>17052: MQ Beyond the Basics - Advanced API and Internals Overview [z/OS &amp; Distributed]</td>
<td>Room: Willow B</td>
</tr>
<tr>
<td>11:15</td>
<td></td>
<td>17041: First Steps with IBM Integration Bus: Application Integration in the New World [z/OS &amp; Distributed]</td>
<td>16732: MQ V8 Hands- on Labs! MQ V8 with CICS and COBOL! MQ SMF Labs! Room: Redwood</td>
<td>17057: Not Just Migrating, but Picking up New Enhancements as You Go - We've Given You the Shotgun, You Know Where Your Feet Are [z/OS &amp; Distributed]</td>
</tr>
<tr>
<td>01:45</td>
<td></td>
<td>17037: All About the New MQ V8 [z/OS &amp; Distributed]</td>
<td>17046: Paging Dr. MQ - Health Check Your Queue Managers to Ensure They Won't Be Calling in Sick! [z/OS]</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>17040: Using IBM WebSphere Application Server and IBM MQ Together [z/OS &amp; Distributed]</td>
<td></td>
<td>All sessions in Seneca unless otherwise noted.</td>
</tr>
<tr>
<td>04:30</td>
<td></td>
<td>17059: IBM MQ: Are z/OS &amp; Distributed Platforms like Oil &amp; Water? [z/OS &amp; Distributed]</td>
<td>17065: Under the hood of IBM Integration Bus on z/OS - WLM, SMF, AT-TLS, and more [z/OS]</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>17055: What's the Cloud Going to Do to My MQ Network?</td>
<td>17043: The Do's and Don’ts of IBM Integration Bus Performance [z/OS &amp; Distributed]</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>17039: Clustering Queue Managers - Making Life Easier by Automating Administration and Scaling for Performance [z/OS &amp; Distributed]</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>17044: But Wait, There's More MQ SMF Data Now?!?! - Monitoring your Channels Using V8's New Chinit SMF Data [z/OS]</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>17068: Monitoring and Auditing MQ [z/OS &amp; Distributed]</td>
<td></td>
</tr>
</tbody>
</table>

Complete your session evaluations online at www.SHARE.org/Seattle-Eval
Remember to submit your evaluation please

Complete your session evaluations online at www.SHARE.org/Seattle-Eval
Copyright and Trademarks

© IBM Corporation 2015. All Rights Reserved.

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml.