CICS Extreme Debugging -- MQ Attachment

Ed Addison
IBM

Thursday, August 11, 2011
Session Number 9612
Agenda

- The basic structure of the CICS-WMQ Adapter
- Differences between the CICS-shipped adapter and the WMQ-shipped adapter.
- Tuning and Monitoring implications
- WMQ and CICS dump formatters
- Dump Analysis
The CICS-MQ Adapter

- CICS/TS 3.1 does not ship the CICS-WMQ Adapter.
  - It will use the one shipped with any release of WMQ.
The WMQ-supplied Adapter

- The adapter attaches 8 subtask TCBs per CICS region. These TCBs are identified to WMQ as ‘server threads.’
  - The initial program for these subtask TCBs is CSQCSERV.

- Together, these 8 ‘server threads’ service all the WMQ API requests made by CICS tasks.
  - A CICS task doesn’t own a server thread for the life of the task.
  - Many tasks within the CICS region share the 8 ‘server threads.’

- These ‘server threads’ do not maintain any affinity to the CICS tasks whose requests they service.

- They pick up a single request, a QRPL, they service it, post back the waiting CICS task, and are then free to handle the next request from any task.

- The ‘server threads’ live long term, usually for the life of the CICS region.
The WMQ-supplied Adapter - Continued

- ‘Server threads’ remain attached to their respective CICS region.
  - They are attached when the WMQ Interface is established and will remain until the Interface is terminated.
  - They do not come and go with CICS transactions.

- A WMQ thread is not the same thing as a ‘server thread’. A WMQ thread aligns with a CICS task or an WMQ Unit of Recovery. It is typically short-lived.
  - A WMQ Display Thread command displays a WMQ Thread, not a ‘server thread’.
  - Threads mentioned in WMQ Dump Formatters are WMQ Threads, not ‘server threads’.
  - Once the CICS task is through with the WMQ call the WMQ thread returns to ‘server thread’ status.
The WMQ-supplied Adapter - Continued

- The WMQ CTHREAD parameter limits the number of ‘server threads.’
  - The CICS QR TCB is a special coordinator ‘server thread’
  - This coordinator server thread and the 8 server threads built on the CSQCSERV TCBs make up the 9 server threads each CICS region has.
  - Each CICS region uses up 9 of the CTHREAD limit.
The WMQ CICS - supplied Adapter

- CICS uses L8 mode TCBs for all calls to WMQ.
  - When an L8 TCB is used for a WMQ call for the first time, it is identified to WMQ as a ‘server thread.’
- MAXOPENTCBS controls the number of L8 and L9 mode TCBs. If enough tasks run concurrently issuing WMQ calls, it is possible for there to be up to MAXOPENTCBS worth of ‘server threads’ identified to WMQ.
- An L8 TCB ‘server thread’ is not shared by several tasks running concurrently. For the life of a CICS task an L8 TCB ‘server thread’ is owned exclusively by the task for WMQ calls and for other requests that need an L8 TCB.
- The lifetime of an L8 TCB server thread is generally the life of the CICS-WMQ connection.
  - There is a decay algorithm that causes L8 TCBs to be detached if unused for an hour or so.
The WMQ CICS - supplied Adapter - Continued

- The WMQ CTHREAD parameter limits the number of ‘server threads.’
  - The CICS QR TCB is a special coordinator ‘server thread’
  - All of the L8 TCBs that have been used for a WMQ call (and therefore have been identified to WMQ) are server threads.
  - The maximum number of ‘server threads’ is MAXOPENTCBS plus 1.

- When running WMQ V6 and upgrading from CICS/Ts 3.1 or lower, to CICS/Ts 3.2 and above, CTHREAD may very well need to be raised.
  - Consider eliminating CTHREAD by setting to the maximum 32767.

- At WMQ V7 CTHREAD is forced to 32767 and is not adjustable.
Tuning and Monitoring Considerations
Tuning Considerations for CTHREAD

- What are the costs of having a server thread connect to a queue manager?
  - Each server thread requires about 5K from high private in the queue manager ASID, and about 5K from ECSA.
  - With WMQ V7 APAR PK69439 and WMQ V6 APAR PK68189, each server thread requires about 4K from high private in the CICS address space. Without the APARs, the 4K is from below-the-line private.
Performance Class Monitoring Considerations

- At CICS/TS 3.1 with the MQ-supplied Adapter, the CSQCSERV subtask TCBs are not managed by the CICS dispatcher.
  - CPU used by those TCBs will be charged to the CICS address space but will not be attributed to the CICS tasks whose MQ calls were serviced by those server threads.
Performance Class Monitoring implications

- At CICS/TS 3.2 and above with the CICS-supplied Adapter, the L8 Mode subtask TCBs are managed by the CICS dispatcher
  - CPU used by those server threads will be charged to the CICS address space and is attributed to the CICS task.
  - Because of this, L8CPUT and USRCPUT will be higher, especially for transactions that make lots of WMQ calls.
Inquire Task implications

- At CICS/TS 3.1 and prior, a task that is processing in WMQ would be in a TASKSWCH suspend.
- At CICS/TS 3.2 and above, a task that is processing in WMQ will be Running on an L8 TCB.
  - On EXEC CICS INQUIRE TASK, there is a CURRENTPROG attribute that returns DFHMQTRU when CICS is processing an MQ call.
WMQ and CICS Dump Formatters
VERBX DFHPDxxx ‘MQ=1’

- This MQ summary shows all the CICS tasks that are making MQ calls.
  - Note the jobname and ASID of CICS
  - Note the MQ Queue Manager name
  - Scroll down to the All Transactions Summary
VERBX DFHPDxxx 'mq=1'

-- DFHPD0121I FORMATTING CONTROL BLOCKS FOR JOB CICS01

ADDRESS SPACE ASID NUMBER (HEX) = 0182

- The hex ASID number and Jobname will be useful when looking at the MQ formatter.
VERBX DFHPDxxx ‘MQ=1’

## MQ GLOBAL STATE SUMMARY

- Connection status: Connected
- In standby mode: No
- Subsys: CSQA
- WMQ release: 0701
- Initiation Queue: CICS01.INITQ
- API Crossing exit active: No

- The Queue Manager Subsystem is necessary for when invoking the MQ formatter.
All of these tasks have issued WMQ calls. Only one task is currently in WMQ.

A task showing as ‘In MQ’ is like a task showing in a TASKSWCH wait using the WMQ-supplied adapter.
VERBX DFHPDxxx ‘DS=1’

<table>
<thead>
<tr>
<th>SF</th>
<th>TT</th>
<th>RESOURCE</th>
<th>RESOURCE_NAME</th>
<th>W TIME OF</th>
<th>TIMEOUT</th>
<th>DTA</th>
<th>AD</th>
<th>ATTACHER</th>
<th>M</th>
<th>SUSPAREA</th>
<th>XM_TXN_TOKEN</th>
</tr>
</thead>
<tbody>
<tr>
<td>S P N</td>
<td>MQSeries</td>
<td>GETWAIT</td>
<td>M 06:45:50.092</td>
<td>28B92500 XM 5D2C8300 L8 5D2E6BEC 5D2C83000075709C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S P N</td>
<td>MQSeries</td>
<td>GETWAIT</td>
<td>M 06:45:53.435</td>
<td>28B92680 XM 5D2D8B00 L8 5CD7AA64 5D2D8B000075720C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td></td>
<td></td>
<td>M 06:45:53.435</td>
<td>28B92800 XM 5D2C3500 L8 5D2C35000075696C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Dispatcher shows this task as running
- The ‘MQ=1’ summary showed this task was in WMQ. Usually a task would only be in WMQ for a short duration of time.
- The times in the DS=1 summary are STCK format with no adjustments for time zone or leap seconds.
- If the task became running on its L8 TCB well before dump time, you would need to investigate that L8 TCB. What is the z/OS TCB address?
- Scroll down to the next summary in DS=1.
VERBX DFHPDxxx ‘DS=1’

```
==DS: TASKS USING OPEN TCBS SUMMARY

<table>
<thead>
<tr>
<th>DS_TOKEN KE_TASK</th>
<th>XM_TXN_TOKEN</th>
<th>TCB_ID</th>
<th>DS_TCB</th>
<th>MVS_TCB</th>
</tr>
</thead>
<tbody>
<tr>
<td>02800003</td>
<td>5B9C3100</td>
<td>L803D</td>
<td>63572D00</td>
<td>009637D8</td>
</tr>
<tr>
<td>02820007</td>
<td>5BB7D900</td>
<td>L802B</td>
<td>638A8300</td>
<td>0097C658</td>
</tr>
<tr>
<td>0284047F</td>
<td>5BB39100</td>
<td>L8024</td>
<td>5D21E500</td>
<td>0098CC88</td>
</tr>
<tr>
<td>040A0071</td>
<td>5BB23500</td>
<td>L8025</td>
<td>5D21EF00</td>
<td>0097A0A0</td>
</tr>
</tbody>
</table>

. . . .
```

- All of the tasks in the MQ=1 summary will also be in this summary as they will all have an L8 TCB.
- Other tasks not issuing WMQ calls may be in this summary as well.
- For task 75696, its TCB address is 97A0A0.
VERBX csqwdprd ‘subsys=csqa’

- The WMQ formatter accesses control blocks in ECSA so you can do this with just a dump of CICS.
  - Use the Queue Manager Subsystem ID from CICS ‘MQ=1’
  - In the output, find on the CICS hex ASID number noted earlier. Issue the find like this: f x0182
Verbx csqwdprd ‘subsys=csqa’

<table>
<thead>
<tr>
<th>Jobname</th>
<th>Conntype</th>
<th>CICS</th>
<th>ASID</th>
<th>ASCE</th>
<th>Task</th>
<th>TCB</th>
</tr>
</thead>
<tbody>
<tr>
<td>EB 2189A188</td>
<td>ACE 2189A128</td>
<td>Thread 5D231238</td>
<td>Tran</td>
<td>TRN1</td>
<td>Task 0075784C</td>
<td>TCB 0098C620.</td>
</tr>
<tr>
<td>EB 21899078</td>
<td>ACE 21899018</td>
<td>Thread 5D254858</td>
<td>Tran</td>
<td>TRN8</td>
<td>Task 0075689C</td>
<td>TCB 009757E0.</td>
</tr>
<tr>
<td>EB 2189A8D8</td>
<td>ACE 2189A878</td>
<td>Thread 5D2E6548</td>
<td>Tran</td>
<td>TRN8</td>
<td>Task 0075738C</td>
<td>TCB 00983AD8.</td>
</tr>
<tr>
<td>EB 21FC5668</td>
<td>ACE 21FC5608</td>
<td>Thread 5CD7A3C0</td>
<td>Tran</td>
<td>TRN7</td>
<td>Task 0075692C</td>
<td>TCB 0097A868.</td>
</tr>
<tr>
<td>EB 21FC5530</td>
<td>ACE 21FC54D0</td>
<td>Thread 5D2313C0</td>
<td>Tran</td>
<td>TRN7</td>
<td>Task 0075765C</td>
<td>TCB 0098CC88.</td>
</tr>
<tr>
<td>EB 1F43FB48</td>
<td>ACE 1F43FAE8</td>
<td>Thread 5D2E66D0</td>
<td>Tran</td>
<td>TRN1</td>
<td>Task 0075696C</td>
<td>TCB 0097A0A0.</td>
</tr>
</tbody>
</table>

- These are the current WMQ threads. These should match up with the CICS tasks in CICS ‘MQ=1’ summary.
A WMQ V6 example

<table>
<thead>
<tr>
<th>Jobname</th>
<th>CICSJOB1</th>
<th>Conntype</th>
<th>CICS</th>
<th>ASID</th>
<th>x0122.</th>
</tr>
</thead>
<tbody>
<tr>
<td>EB 1C962518</td>
<td>ACE 1C9624B8</td>
<td>Thread 00000000</td>
<td>Tran TR21</td>
<td>Task 0008912C</td>
<td></td>
</tr>
<tr>
<td>EB 1C5733F8</td>
<td>ACE 1C573398</td>
<td>Thread 00000000</td>
<td>Tran TR22</td>
<td>Task 0058138C</td>
<td></td>
</tr>
<tr>
<td>EB 1BBCFF38</td>
<td>ACE 1BBCFED8</td>
<td>Thread 00000000</td>
<td>Tran TR22</td>
<td>Task 0047300C</td>
<td></td>
</tr>
<tr>
<td>EB 1BBCF998</td>
<td>ACE 1BBCF938</td>
<td>Thread 2C2EF1C8</td>
<td>Tran CKTI</td>
<td>Task 0000043C</td>
<td></td>
</tr>
</tbody>
</table>

- There is only 1 thread, the 1 with a non-zero Thread address.
- The other 3 ‘threads’ are just sets of control blocks that used to be threads and are ready to go when needed for a new thread.
- WMQ APAR PK75212 changes the formatter to not show these free threads.
Using a Dump to View WMQ tasks
Questions to be answered

- Is CICS healthy?
- Is the task currently calling WMQ?
- How long ago was the call made?
- What program made the call?
- How can I find out the call type and locate the parameters?
- How many calls has the task made?
You’ve got a dump. What first?

- **Make sure CICS is healthy as a job**
  - Don’t focus on task hangs if CICS itself is not healthy
  - Find out what time the dump was taken
  - Compare to CICS internal time stamps
  - If CICS is healthy those times will be close together
What time is it?

- Most timestamps are in local time
  - CICS internal trace
  - Console log
  - CICS messages
  - Kernel formatter

- Some timestamps are in GMT
  - CICS dispatcher summary
  - Units of Work (UOW)
  - Dump Incident Token time

- You need to be able to convert from GMT to local
  - Can be tricky if Leap Seconds are being used
Convert GMT time to Local

- Figure out the difference between GMT and Local
  - Issue **LTOD 0** from Option 6 of IPCS
  - This will provide the following output:
    
    09/17/2042 22:53:47.370496 STCK X'00000000 00000000'
    09/17/2042 22:53:47.370496 UTC X'00000000 00000000'
    09/17/2042 23:53:47.370496 LOCAL X'FFFFBCF1 DCC00000'

- In this case Local time and STCK are one hour apart exactly.

- If STCK and UTC are identical, there are no leap seconds
  - If there were Leap Seconds involved these times would be different.
  - Leap Seconds are shop dependent
What time was the dump taken?

- From option 6 of IPCS (COMMAND) issue: **ST SYS**

**SYSTEM STATUS:**

Sysplex name: SYSPLEX1

TIME OF DAY CLOCK: C8190185 BBD100EE  07/20/2011 13:00:04.466960 local

TIME OF DAY CLOCK: C818F41C 819100EE  07/20/2011 12:00:04.466960 GMT

Program Producing Dump: SVCDUMP

Program Requesting Dump: IEAVTSDT

Incident token: SYSPLEX1 MV23  07/20/2011 11:59:49.737423 GMT

- **Use Incident token time**

  - **11:59:49.737423 = 12:59:49.737423** Local Time
CSA Time-of-Day

- A byproduct of CICS being healthy is the CSA time-of-day field, CSATODP, being updated with the current time
  - When CICS is healthy, CSATODP is updated regularly
  - When CICS is unhealthy, CSATODP is not updated

- Therefore, a quick and effective way to gauge CICS’s health is to compare the time the dump was taken to the CSA time-of-day field

  - CSATODP (CSA +x’50’) is in the form of HHMMSSTF where
    - H is hours, M is minutes, S is seconds, T is tenths (F indicates a field in packed decimal format)
  - The CSA can be formatted using `VERBX DFHPDxxx ‘CSA=2’`

Note: CSATODP is updated on every QR TCB dispatch and also when an ASKTIME is issued on ANY TCB. So, CICS can appear healthy even though the QR isn’t being dispatched if another TCB (L8, L9) is issuing EXEC CICS ASKTIME
What time does CICS think it is?

- **VERBX DFHPD650 ‘CSA=2’**

```plaintext
0000 00000200 0004C020 0004F5A0 948E5C32 80BF3498 80800000 14FDB030 14F98260
0020 00000000 14F95100 0000010C 00000000 948E577A 15330000 14FDB030 14FC0D50
0040 00055B20 140DC800 0010032C 008B2000 1259497F 14052108 00000200 00000000
```

- **CSA +X’50’ is Local Time in packed decimal field of format HHMMSST. This one is 12:59:49.7**
  - Matches local time of dump calculated on slide 29
  - Updated every time a task is dispatched on the QR TCB or ASKTIME is issued on any CICS TCB
If CSA time and Dump time are minutes apart

- CICS is probably not healthy as a job
  - CICS may be in a loop which does not update CSATODP
    - CICS will continue to loop if ICVR set to 0
    - CICS MAY continue to loop even if ICVR is set to reasonable value
  - CICS may be in a hard wait
  - CICS may be CPU starved

- Determine why CICS is not receiving resources needed to run
Are Tasks Currently calling WMQ?

- To see if a task is currently calling MQ issue VERBX DFHPDxxx ‘MQ’

<table>
<thead>
<tr>
<th>Tran</th>
<th>Task</th>
<th>TcaAddr</th>
<th>TieAddr</th>
<th>LotAddr</th>
<th>ThrdAddr</th>
<th>Uowid</th>
<th>Tcb</th>
<th>in MQ</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CKTI</td>
<td>00041</td>
<td>140E2100</td>
<td>153B21B8</td>
<td>153B2238</td>
<td>153B2290</td>
<td>C81819C885B51E41</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td><strong>MQED</strong></td>
<td>30749</td>
<td>140EA100</td>
<td>153B2340</td>
<td>153B23C0</td>
<td>153B2418</td>
<td>C818F40EC53D4D83</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><strong>MQED</strong></td>
<td>30750</td>
<td>140E3100</td>
<td>153FD7D8</td>
<td>153FD858</td>
<td>153FD8B0</td>
<td>C818F40EC540CB03</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><strong>MQED</strong></td>
<td>30751</td>
<td>140E9100</td>
<td>153FD340</td>
<td>153FD3C0</td>
<td>153FD418</td>
<td>C818F40EC558D9C3</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><strong>MQED</strong></td>
<td>30752</td>
<td>140E3800</td>
<td>153FD030</td>
<td>153FD0B0</td>
<td>153FD108</td>
<td>C818F40EC597E44E</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><strong>MQED</strong></td>
<td>30753</td>
<td>140E4800</td>
<td>153FD960</td>
<td>153FD9E0</td>
<td>153FDA38</td>
<td>C818F40EC59A8B4E</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><strong>MQED</strong></td>
<td>30754</td>
<td>140EB800</td>
<td>153FD650</td>
<td>153FD6D0</td>
<td>153FD728</td>
<td>C818F40EC5A72446</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td><strong>MQED</strong></td>
<td>30755</td>
<td>140E2800</td>
<td>153B2DF8</td>
<td>153B2E78</td>
<td>153B2ED0</td>
<td>C818F40EC5B474C6</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>
When was the WMQ call issued?

- Issue VERBX DFHPDxxx ‘DS’ to view the Dispatcher Domain

<table>
<thead>
<tr>
<th>DS_TOKEN</th>
<th>KE_TASK</th>
<th>T S RESOURCE</th>
<th>RESOURCE</th>
<th>TIME_OF</th>
<th>DTA</th>
<th>ATTACHER M SUSPEND</th>
<th>DSTSK)</th>
<th>TOKEN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>040A10AD</td>
<td>14EEB700</td>
<td>N R</td>
<td></td>
<td></td>
<td>13FF8800</td>
<td>16410B00 L8</td>
<td>16410B0000030764C</td>
<td></td>
</tr>
<tr>
<td>04820001</td>
<td>14EFD700</td>
<td>N S</td>
<td></td>
<td></td>
<td>19:43:17</td>
<td>13FF9200 14FA2300 QR 14F33A78</td>
<td>16410B000000040C</td>
<td></td>
</tr>
<tr>
<td>04840001</td>
<td>14EFD100</td>
<td>N S MQSeries</td>
<td>GETWAIT</td>
<td>11:59:50</td>
<td>13FF9380</td>
<td>14FA2500 L8 153B22BC</td>
<td>16410B000000041C</td>
<td></td>
</tr>
<tr>
<td>048C10A7</td>
<td>14EFA100</td>
<td>N R</td>
<td></td>
<td></td>
<td>13FF9980</td>
<td>14FA2B00 L8</td>
<td>16410B0000030753C</td>
<td></td>
</tr>
<tr>
<td>05000C95</td>
<td>14ECD100</td>
<td>N R</td>
<td></td>
<td></td>
<td>26C5C080</td>
<td>153F1300 L8</td>
<td>16410B0000030749C</td>
<td></td>
</tr>
<tr>
<td>05020C91</td>
<td>14EA0100</td>
<td>N R</td>
<td></td>
<td></td>
<td>26C5C200</td>
<td>1645A100 L8</td>
<td>16410B0000030766C</td>
<td></td>
</tr>
</tbody>
</table>

- Dispatcher shows this task as running on an L8 TCB
- The ‘MQ=1’ summary showed this task was in WMQ. Usually a task would only be in WMQ for a short duration of time.
- How long has this task been running in MQ?
- Display the DTA: ip l 26C5C080 l(X’60’)


The DTA

For a task that is Running, DTA+X’58’ is the time the task became Running, within a tenth of a second. Format this to see what time it is.

Issue: ip ltod C818F40E3B62328E

If the time is right at dump time, then the dump was taken while the task was just doing some MQ work.

From slide 30 we found the dump was taken at 12:59:49.737423 Local Time.

If it is some time prior, the task is hung for some reason.
What we know so far ...

- CICS Dump was taken at 12:59:49.737423 Local Time
- CSATOD (Time of Day) is 12:59:49.7 - CICS is healthy
- There currently are seven tasks in WMQ at dump time
- Task 30749 became running at 12:59:49.49427
What Program Issued the WMQ Call?

- Issue VERBX DFHPDxxx ‘PG’ for Program Domain and issue F 30749 for the task number

---

**==PG: PTA SUMMARY FOR TRAN NUM : 30749, PTA ADDRESS : 14DFE810**

LOG-LVL : 3  SYS-LVL : 0  TASK-LLE : 00000000  PLCB : 14ED27E8

**=PG: TASK PLCB SUMMARY**

**PROG DFHMQTRU LVL 3 PLCB 14ED27E8 LD 00000000 ENT 00000000 LEN 000000 PPTE 15050E70 ENV TRUE INV MQPUTLOG EXIT**

**PROGRAM: DFHMQTRU**  
CPE: 140A6100  
LIB: DFHRPL  
CONCAT: 03

**PROG MQPUTLOG LVL 2 PLCB 14ED12C8 LD 16448000 ENT 96448028 LEN 005B50 PPTE 153324F0 ENV EXEC INV MQGETLOC EXIT**

**PROGRAM: MQPUTLOG**  
CPE: 15331850  
LIB: DFHRPL  
CONCAT: 01

**PROG MQGETLOC LVL 1 PLCB 14ECF060 LD 16436000 ENT 96436028 LEN 006298 PPTE 15332626 ENV EXEC INV CICS EXIT**

**PROGRAM: MQGETLOC**  
CPE: 140AB100  
LIB: DFHRPL  
CONCAT: 01

---

- This display shows the first level program MQGETLOC was INVoked by CICS and LINKed to second level program MQPUTLOG.
- The second level program MQPUTLOG then made a call to DFHMQTRU. You will see this when the program is active in WMQ.
- Note that this will not show COBOL CALLs. Verification of where the call was made to WMQ is still needed.
What Program Issued the WMQ Call?

- CICS will obtain a control block called Program Environment Save Area (PESA) when stacking a user environment. This will be done when:
  - EXEC CICS LINK is issued
  - Call to a Global User Exit (GLUE) that can issue EXEC CICS commands
  - Call to a Task Related User Exit (TRUE) such as WMQ and DB2
  - Call to a User Replaceable Module (URM) such as the WMQ API Crossing Exit
- PESA can be used to get back to the program that made the WMQ call
- To find a PESA for a transaction issue VERBX DFHPDxxx ‘AP’ and then issue F PESA.xxxxxx where xxxxx is the transaction number.
PESA for Transaction 30749

**PESA.30749 14ED28A8 PROGRAM ENVIRONMENT SAVE AREA**

<table>
<thead>
<tr>
<th>Address</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>02806EC4 C6C8D7C5 E2C10500 14ED1328 00000000 00000000</td>
</tr>
<tr>
<td>0020</td>
<td>00000000 00000000 00000000 000016AB E6200000 00000000 000014FA 307414ED</td>
</tr>
<tr>
<td>0040</td>
<td>1F840000 0000D3F8 20080000 00000000 0006B880 80400000 00000000 000016AB</td>
</tr>
<tr>
<td>0060</td>
<td>E6680000 00000000 00000000 00000000 00000000 00000000 00000000 00000000</td>
</tr>
<tr>
<td>0080</td>
<td>00000000 00000000 00000000 00000000 000016AC 38680000 00000000</td>
</tr>
</tbody>
</table>

- Offset X’18’ for a WMQ PESA will point to Register 13 of the caller.
  - For WMQ this will be the Register Save Area when the call was made.
  - Note that offset X’96’ also has the same address in case you are reviewing PESAs for other TRUEs, GLUEs or URMs.
Register Save Area when WMQ call was issued

<table>
<thead>
<tr>
<th>16AC3868:</th>
<th>16AC3870:</th>
<th>16AC3880:</th>
<th>16AC3890:</th>
<th>16AC38A0:</th>
<th>16AC38B0:</th>
<th>16AC38C0:</th>
</tr>
</thead>
<tbody>
<tr>
<td>00104001</td>
<td>16AC3988</td>
<td>16AC3968</td>
<td>166502D0</td>
<td>16AC4A58</td>
<td>00000000</td>
<td>16AC3868</td>
</tr>
<tr>
<td></td>
<td>1644CC88</td>
<td>166400C0</td>
<td>14ED1EE0</td>
<td>16448170</td>
<td>16AC3988</td>
<td>16AC4A58</td>
</tr>
<tr>
<td></td>
<td>00000000</td>
<td>16640040</td>
<td>00000000</td>
<td>1644C3A2</td>
<td>00000000</td>
<td>955B1648</td>
</tr>
<tr>
<td></td>
<td></td>
<td>166510C0</td>
<td>166500C0</td>
<td>16AC2990</td>
<td>16AC3788</td>
<td>00000000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Offset X’C’ is Register 14 when the WMQ call was made and will point to the WMQ Stub.
- Offset X’14’ is Register 0 when the WMQ call was made and will point to where the program actually issued the WMQ call.
- Offset X’18’ is Register 1 when the WMQ call was made and will point to the parameters passed from the Application.
## WMQ Stub

<table>
<thead>
<tr>
<th>Offset</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1644CC88</td>
<td>1644CC90</td>
<td>1644CCA0</td>
</tr>
<tr>
<td>47F0E008</td>
<td>07FE18F0</td>
<td>07FE18F0</td>
</tr>
<tr>
<td>08000005</td>
<td>41000008</td>
<td>41000008</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Offset 4 within the WMQ stub is the number of parameters passed and the type.
  - 08 is the number of parameters passed and the type of call is 05. Most common call types are:
    - 01 = Open
    - 02 = Close
    - 03 = Get
    - 04 = Put
    - 05 = Put1
Program that made the WMQ call

- Format Loader Domain by issuing: VERBX DFHPDxxx ‘LD’
- Issue F ‘PROGRAM STORAGE MAP’ to get to the modules loaded within the CICS region
- Use the address derived from the WMQ Register Save Area on slide 40 (9644C502)

<table>
<thead>
<tr>
<th>PGM NAME</th>
<th>ENTRY PT</th>
<th>CSECT</th>
<th>LOAD PT. REL.</th>
<th>PTF LVL.</th>
<th>LAST COMPILED</th>
<th>COPY NO.</th>
<th>USERS</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFHEDFM</td>
<td>96402000</td>
<td>-noheda-</td>
<td>16402000</td>
<td></td>
<td></td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>MQGETLOC</td>
<td>96436028</td>
<td>DFHYI660</td>
<td>16436000 660</td>
<td>1</td>
<td>18</td>
<td>1</td>
<td>18</td>
</tr>
<tr>
<td>MQPUTLOG</td>
<td>96448028</td>
<td>DFHYI660</td>
<td>16448000 660</td>
<td>1</td>
<td>8</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>MQOPEN</td>
<td>9644F028</td>
<td>DFHYI660</td>
<td>1644F000 660</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
</tr>
</tbody>
</table>

- Address that made WMQ call resides in program MQPUTLOG.
  - Matches what was found in the PG domain on slide 37.
  - MQPUTLOG made the WMQ call at x’44DA’
    - 1644C502 (from PESA offset X’14’) minus program entrypoint 16448028
Parameters passed on the WMQ call

- Use register one derived from the WMQ Save Area on slide 40 (16AC3968).

| 16AC3968 | 166502A0 | 166502F8 | ... | 8 |
| 16AC3970 | 16650488 | 166506B0 | 16650290 | 16650684 | ...h... | d | ...
| 16AC3980 | 166502B8 | 966502D0 | 00000000 | 00000000 | ...o... | } | ......

- There were 8 parameters passed on the PUT1 WMQ call. The list is terminated when the high order bit is on.

- The WebSphere MQ z/OS Problem Determination Guide lists all the parameters passed for all calls. Here is what’s documented for MQPUT1:
  - ARG 000 Connection handle
  - ARG 001 Object descriptor
  - ARG 002 Message descriptor
  - ARG 003 Put message options
  - ARG 004 Buffer length
  - ARG 005 Message data
  - ARG 006 Completion code
  - ARG 007 Reason code
Important Arguments Passed

- **ARG 001 - Object Descriptor (MQOD) will contain the Queue Name**
  - 166502F8  D6C44040  00000001  |  OD  ....  |
  - 16650300  00000001  C5C44BD3  D6C74BD8  E4C5E4C5  |  ...  ED.LOG.QUEUE  |

- **ARG 003 - Put Message Options (MQMPO) will contain options**
  - 166506B0  D7D4D640  00000001  00000002  FFFFFFFF  |  PMO  .............  |
    - The 00000002 indicates Syncpoint - See WMQ z/OS Problem Determination Guide for all options

- **ARG 004 - Buffer Length**
  - 16650290  00000008

- **ARG 005 - Message Data**
  - 16650684  C5C4E940  C4C1E3C1  F8F2F1F8  |  EDZ DATA3218  |
What we know so far ...

- CICS Dump was taken at 12:59:49.737423 Local Time
- CSATOD (Time of Day) is 12:59:49.7 - CICS is healthy
- There currently are seven tasks in WMQ at dump time
- Task 30749 became running at 12:59:49.499427
- Program MQPUTLOG made a WMQ PUT1 call at offset X'44DA'
- WMQ PUT1 call was made to ED.LOG.QUEUE
  - SYNCPOINT Option
  - Eight byte buffer
  - Data passed was EDZ DATA
How many WMQ Calls has the task issued

- There are some helpful changes in the monitoring domain dump formatter with CICS/TS 4.1 when performance class monitoring is active.
  - For active tasks, all the non-zero monitoring fields are formatted, DFH$MOLS style.
  - There is a summary of active tasks showing a few key monitoring fields.
VERBX DFHPDxxx ‘MN’

- Scroll down to the Overview and verify there are Transaction Monitoring Areas (TMA)
  - TMAs will be zero if Monitoring is not active

**Number of Transaction Monitoring Areas** 19

CICS Monitoring is ACTIVE
Exception Monitoring is ACTIVE
Performance Monitoring is ACTIVE
Resource Monitoring is ACTIVE
Identity Monitoring is NOT ACTIVE
VERBX DFHPDxxx ‘MN’

- Find on ‘TMA-DATA’ to get to the new summary

<table>
<thead>
<tr>
<th>Tran id</th>
<th>Tran number</th>
<th>Tran token</th>
<th>Start time</th>
<th>Dispatch time</th>
<th>CPU time</th>
<th>Suspend time</th>
<th>Dispatch time</th>
<th>Change mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>CECI</td>
<td>0024121</td>
<td>153BA000</td>
<td>11:59:22.3794</td>
<td>00:00:00.2767</td>
<td>00:00:00.0071</td>
<td>Running</td>
<td>00:00:00.0227</td>
<td>00:00:00.0000</td>
</tr>
<tr>
<td>MQED</td>
<td>0030749</td>
<td>1640D000</td>
<td>11:59:49.4994</td>
<td>Running</td>
<td>00:00:00.0007</td>
<td>00:00:00.0004</td>
<td>00:00:00.0001</td>
<td>00:00:00.0001</td>
</tr>
<tr>
<td>MQED</td>
<td>0030750</td>
<td>1641D000</td>
<td>11:59:50.0640</td>
<td>Running</td>
<td>00:00:00.0007</td>
<td>00:00:00.0001</td>
<td>00:00:00.0000</td>
<td>00:00:00.0000</td>
</tr>
<tr>
<td>MQED</td>
<td>0030751</td>
<td>164C4000</td>
<td>11:59:50.0644</td>
<td>Running</td>
<td>00:00:00.0006</td>
<td>00:00:00.0002</td>
<td>00:00:00.0001</td>
<td>00:00:00.0001</td>
</tr>
<tr>
<td>MQED</td>
<td>0030752</td>
<td>153CD000</td>
<td>11:59:50.0652</td>
<td>Running</td>
<td>00:00:00.0007</td>
<td>00:00:00.0002</td>
<td>00:00:00.0000</td>
<td>00:00:00.0000</td>
</tr>
<tr>
<td>MQED</td>
<td>0030753</td>
<td>153F8000</td>
<td>11:59:50.0654</td>
<td>Running</td>
<td>00:00:00.0007</td>
<td>00:00:00.0001</td>
<td>00:00:00.0000</td>
<td>00:00:00.0000</td>
</tr>
<tr>
<td>MQED</td>
<td>0030754</td>
<td>16408000</td>
<td>11:59:50.0657</td>
<td>Running</td>
<td>00:00:00.0007</td>
<td>00:00:00.0001</td>
<td>00:00:00.0000</td>
<td>00:00:00.0000</td>
</tr>
<tr>
<td>MQED</td>
<td>0030755</td>
<td>16411000</td>
<td>11:59:50.0659</td>
<td>Running</td>
<td>00:00:00.0006</td>
<td>00:00:00.0000</td>
<td>00:00:00.0000</td>
<td>00:00:00.0000</td>
</tr>
</tbody>
</table>
## VERBX DFHPDxxx ‘MN’

==MN: TRANSACTION TMA-DATA SUMMARY

<table>
<thead>
<tr>
<th>Tran id</th>
<th>Tran number</th>
<th>Tran token</th>
<th>Start time</th>
<th>Dispatch time</th>
<th>CPU time</th>
<th>Suspend time</th>
<th>Wait time</th>
<th>Dispatch</th>
<th>Change mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>CECI</td>
<td>0024121</td>
<td>153BA000</td>
<td>11:59:22.3794</td>
<td>00:00:00.2767</td>
<td>00:00:00.0074</td>
<td>Running</td>
<td>00:00:00.0227</td>
<td>00:00:00.0000</td>
<td></td>
</tr>
<tr>
<td>MQED</td>
<td>0030749</td>
<td>1640D000</td>
<td>11:59:49.4994</td>
<td>Running</td>
<td>00:00:00.0007</td>
<td>00:00:00.0004</td>
<td>00:00:00.0001</td>
<td>00:00:00.0001</td>
<td></td>
</tr>
</tbody>
</table>

- Times are in STCK, not Local.
  - A task is always either Suspended or Dispatched.
  - Running in the Suspend Time column means the task is currently suspended. Its Suspend Time Clock is running.
**VERBX DFHPDxxx ‘MN’**

```
==MN: TRANSACTION TMA-DATA SUMMARY

<table>
<thead>
<tr>
<th>Tran id</th>
<th>Tran number</th>
<th>Tran token</th>
<th>Start time</th>
<th>Dispatch time</th>
<th>CPU time</th>
<th>Suspend time</th>
<th>Dispatch</th>
<th>Change mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>CECI</td>
<td>0024121</td>
<td>153BA000</td>
<td>11:59:22.3794</td>
<td>00:00:00.2767</td>
<td>00:00:00.0071</td>
<td>Running</td>
<td>00:00:00.0227</td>
<td>00:00:00.0000</td>
</tr>
<tr>
<td>MQED</td>
<td>0030749</td>
<td>1640D000</td>
<td>11:59:49.4994</td>
<td></td>
<td>Running</td>
<td>00:00:00.0007</td>
<td>00:00:00.0004</td>
<td>00:00:00.0001</td>
</tr>
</tbody>
</table>
```

- Running in Dispatch Time means task’s Dispatch Time clock is running. It is dispatched.
- In this dump, STCK time of dump is 11:59:49.737423
- Subtracting transaction start time of 30749 from the dump time shows it started 0.238 seconds ago.
  - Since that task’s Suspend Time is next to nothing, that means it has been Dispatched all that time. A task is always either Dispatched or Suspended.
- Scroll down below this summary
  - For each task you get a DFH$MOLS style layout of the non-zero monitoring fields so you can see what each task has done.
VERBX DFHPDxxx ‘MN’

MNTMA 1640D000 Transaction Monitoring Area

<table>
<thead>
<tr>
<th>Field-Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFHTASK</td>
<td>C001</td>
</tr>
<tr>
<td>TRAN</td>
<td>D4D8C5C4</td>
</tr>
<tr>
<td>DFHCICS</td>
<td>C089</td>
</tr>
<tr>
<td>USERID</td>
<td>C3C9C3E2E4E2C5D9</td>
</tr>
<tr>
<td>DFHTASK</td>
<td>C004</td>
</tr>
<tr>
<td>TTYPE</td>
<td>E2C40000</td>
</tr>
<tr>
<td>DFHCICS</td>
<td>T005</td>
</tr>
<tr>
<td>START</td>
<td>C818F40EC527BE03</td>
</tr>
<tr>
<td>DFHTASK</td>
<td>P031</td>
</tr>
<tr>
<td>TRANNUM</td>
<td>0030749C</td>
</tr>
<tr>
<td>DFHTASK</td>
<td>A109</td>
</tr>
<tr>
<td>TRANPRI</td>
<td>00000001</td>
</tr>
<tr>
<td>DFHPROG</td>
<td>C071</td>
</tr>
<tr>
<td>PGMNAME</td>
<td>D4D8C7C5E3D3D6C3</td>
</tr>
<tr>
<td>DFHDATA</td>
<td>A395</td>
</tr>
<tr>
<td>NMREQCT</td>
<td>00000005</td>
</tr>
<tr>
<td>DFHTASK</td>
<td>S007</td>
</tr>
<tr>
<td>USRDISPT</td>
<td>C818F40EC542244E</td>
</tr>
<tr>
<td>DFHTASK</td>
<td>S008</td>
</tr>
<tr>
<td>USRCPUT</td>
<td>00000000002CE9C0</td>
</tr>
<tr>
<td>DFHTASK</td>
<td>S014</td>
</tr>
<tr>
<td>SUSPTIME</td>
<td>00000000001A664B</td>
</tr>
<tr>
<td>DFHTASK</td>
<td>S102</td>
</tr>
<tr>
<td>DSRPWTT</td>
<td>00000000007474B</td>
</tr>
<tr>
<td>DFHTASK</td>
<td>S255</td>
</tr>
<tr>
<td>QRDISPT</td>
<td>000000000045580</td>
</tr>
<tr>
<td>DFHTASK</td>
<td>S256</td>
</tr>
<tr>
<td>QRCPUT</td>
<td>0000000000467E0</td>
</tr>
</tbody>
</table>

DFHTASK  | C001  |
| Tran     | D4D8C5C4 |
| DFHCICS  | C089  |
| User Id  | C3C9C3E2E4E2C5D9 |
| DFHTASK  | C004  |
| Type     | E2C40000 |
| DFHCICS  | T005  |
| Start    | C818F40EC527BE03 |
| DFHTASK  | P031  |
| Trans Num| 0030749C |
| DFHTASK  | A109  |
| Tran Pri | 00000001 |
| DFHPROG  | C071  |
| Pgm Name | D4D8C7C5E3D3D6C3 |
| DFHDATA  | A395  |
| NmReq Ct | 00000005 |
| DFHTASK  | S007  |
| User Dis | C818F40EC542244E |
| DFHTASK  | S008  |
| User Cpu | 00000000002CE9C0 |
| DFHTASK  | S014  |
| Susptime | 00000000001A664B |
| DFHTASK  | S102  |
| Disp Wtt | 00000000007474B |
| DFHTASK  | S255  |
| Qrdisp   | 000000000045580 |
| DFHTASK  | S256  |
| Qrcput   | 0000000000467E0 |
What we know

- CICS Dump was taken at **12:59:49.737423 Local Time**
- CSATOD (Time of Day) is 12:59:49.7 - CICS is healthy
- There currently are seven tasks in WMQ at dump time
- Task 30749 became running at 12:59:49.499427
- Program MQPUTLOG made a WMQ PUT1 call at offset X’44DA’
- WMQ PUT1 call was made to ED.LOG.QUEUE
  - SYNCPOINT Option
  - Eight byte buffer
  - Data passed was EDZ DATA
- Transaction 30749 has currently issued five WMQ requests
Summary

- The basic structure of the CICS-WMQ Adapter
- Differences between the CICS-shipped adapter and the WMQ-shipped adapter.
- Tuning and Monitoring implications
- WMQ and CICS dump formatters
- Dump Analysis