Understanding, Monitoring and Managing z/OS Enclaves

Ed Woods / IBM Corporation

Session  10667
Thursday, March 15, 2012: 4:30 PM-5:30 PM
Agenda

- Terminology
- What Is An Enclave?
- Enclave Examples
- Displaying Enclave Information
- DB2 Monitoring Considerations
- Enclaves And WLM Considerations
Some Important Terminology
TCBs and SRBs

- z/OS uses a huge variety of control blocks, many with very specialized purposes
- The three most commonly used control blocks are:
  - Task control block (TCB) - represents a unit of work or task, such as an application program, that runs in an address space
  - Service request block (SRB) - represents a request for a system service
    - SRBs are typically created when one address space detects an event that affects a different address space
    - SRBs provide a mechanism for communication between address spaces.
  - Address space control block (ASCB), which represents an address space
More About TCBs And SRBs

- To identify and keep track of its work z/OS represents each unit of work on the system with a control block
- Standard dispatching units (TCBs and SRBs)
  - TCB - runs at dispatching priority of address space and is preemptible
  - SRB - runs at supervisor priority and is non-preemptible
- Advanced dispatching units
  - Enclave
    - Serves as an anchor for an address space independent transaction
    - Can consist of multiple tasks (TCBs or SRBs) executing across multiple address spaces
  - Client SRB
    - Similar to an ordinary SRB but runs with client dispatching priority and is preemptible
  - Enclave SRB
    - Similar to an ordinary SRB but runs with enclave dispatching priority and is preemptible
So What Are Enclaves?

- Enclaves represent a "business unit of work"
- Enclaves are managed separately from the z/OS address spaces
- Enclaves can include multiple SRBs/TCBs
  - Can span multiple address spaces
  - Can have many enclaves in a single address space
  - Assigned by WLM to a service class for prioritization by the system
What Is A Business “Unit Of Work”? 

- A “unit of work” represents a WLM transaction
  - An item of work where WLM collects resource usage information
  - Represents a subsystem work request
    - WLM can measure resources used by the subsystem request

- Types of transactions
  - Address Space
    - WLM will measure all resource used by a subsystem request in a single address space
  - Enclave
    - Enclave created and used by a subsystem for each work request across multiple address spaces and systems
    - Used by a variety of workloads; DB2, DB2 DDF, WebSphere, MQ, LDAP, TCP/IP
  - CICS and IMS Transactions
    - Not address space or enclave oriented
    - Measures resource used by CICS/IMS transaction requests
Who Uses Enclaves?

- Enclaves have become a pervasive mechanism in the z/OS operating system

- DB2 was one of the early exploiters of the concept of enclaves
  - Enclaves provided a mechanism to manage and prioritize DB2 distributed (DDF) workload
  - More exploitation added with subsequent DB2 releases
    - DB2 stored procedure support
    - DB2 sysplex query parallelism
    - DB2 sequential prefetch and deferred write processing (DB2 10)

- Many core z/OS components use enclaves
  - MQseries, WebSphere, TCP/IP, LDAP
Categories Of Enclaves

- Independent Enclaves
  - Use an independent enclave to represent a new transaction
  - An independent enclave must be classified into a service class or performance group when it is created

- Dependent Enclaves
  - Use a dependent enclave when you have an existing address space defined with its own performance goal
  - Extends that goal to programs running under dispatchable units in other address spaces

- Work-dependent Enclaves
  - Use a work-dependent enclave to extend an existing independent enclave's transaction (for zIIP support – more on this later)
WLM Enclaves – An Example

- The enclave is managed separately from the address spaces it runs in:
  - CPU and I/O resources associated with processing the work request represented by the enclave may be managed by the transaction’s performance goal.
- Storage resources may be managed as follows:
  - To the goals of the enclaves it serves (if enclave server address space).
  - To the performance goal of the address space (if no server address space).
Example - Enclaves Provide DB2 DDF
With Granularity And Control

- DDF prior to enclaves
  - Workload ran at the priority of the DDF task

- DDF with enclaves
  - WLM has more control and granularity to prioritize work
Example - DDF Stored Procedure Priority

- When a Stored Procedure is called from DDF thread
  - DB2 references the enclave created for the DDF request for Stored Procedure
  - Stored Procedure priority is the priority of the DDF request
When a Stored Procedure is called from an application on z/OS:
- DB2 creates an enclave for use by the Stored procedure.
- Stored Procedure priority is the priority of the calling application address space.
Summary - DB2 Workload Prioritization
How Does WLM Assign Priority To DB2 Workload?

- The priority of the DB2 workload will vary depending upon the origin of the workload
  - DB2 workload originating from a local application (examples - IMS, CICS, TSO, Batch, WebSphere)
    - Priority is inherited from the invoking application
    - This applies to Stored Procedures invoked locally
  - DB2 Distributed requests (Subsystem type DDF)
    - Priority controlled by DDF Service Class definitions
    - DB2 Stored Procedure request via DDF - priority controlled by Service Class definitions
  - DB2 Sysplex Query parallelism (Subsystem type DB2)
    - Classification done by DB2 Service Class definitions
DB2 And zIIP Processors

- Work on z/OS may have all or a portion of its resource usage on an enclave SRB
  - Enclave SRB work may be directed to the zIIP

- Certain types of DB2 work may take advantage of zIIP
  - DRDA - Queries that access DB2 for z/OS via DRDA over TCP/IP
  - Complex parallel queries
  - DB2 utilities for index maintenance
    - LOAD, REORG, and REBUILD
  - DB2 V10 – Sequential prefetch eligible for zIIP processor

- WLM and new enclave structures to manage zIIP related workload – work dependent enclave
About Work-dependent Enclaves

- A type of enclave named “Work-Dependent” as an extension of an Independent Enclave.
  - A Work-Dependent enclave becomes part of the Independent Enclave’s transaction
  - Allows to have its own set of attributes (including zIIP offload percentage)
DB2 V10 Exploits Enclaves For Prefetch

- Buffer pool prefetch activity (dynamic prefetch, list prefetch, sequential prefetch) is 100% zIIP eligible in DB2 10
- DB2 10 zIIP eligible buffer pool prefetch is asynchronously initiated by the DBM! address space
  - Executed with a dependent enclave owned by the MSTR address space
  - Deferred write also eligible for zIIP
- Asynchronous buffer pool prefetch activities are not accounted to the DB2 client
  - Shows up in the DB2 statistics report
DB2 V10 Exploits Enclaves For Prefetch - continued

- Asynchronous I/O processing is important to DB2 performance
- With DB2 10 buffer pool prefetch activities are asynchronously initiated by the DBM1 address space
  - This is executed in a dependent enclave
- Asynchronous buffer pool prefetch activities are not accounted to the DB2 client application
  - CPU time accounted to the zIIP appears in DB2 statistic report (PREEMPT IIP SRB)
Displaying Enclaves
SDSF Example – DA Display And The ENC Command

<table>
<thead>
<tr>
<th>Token</th>
<th>Type</th>
<th>WLM SC</th>
<th>WLM RC</th>
<th>Owner</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Display Filter View Print Options Help

SDSF ENCLAVE DISPLAY ZT01 ALL

<table>
<thead>
<tr>
<th>Enclave 38000088805 on System ZT01</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subsystem type: STC</td>
</tr>
<tr>
<td>Subsystem name: STC</td>
</tr>
<tr>
<td>Priority: 15</td>
</tr>
<tr>
<td>Userid: DB1DSTC</td>
</tr>
<tr>
<td>Transaction name: DB1MSTR</td>
</tr>
<tr>
<td>Transaction class:</td>
</tr>
<tr>
<td>Method:</td>
</tr>
<tr>
<td>Logical unit name:</td>
</tr>
<tr>
<td>Subsys collection: N0PZT00</td>
</tr>
<tr>
<td>Process name:</td>
</tr>
<tr>
<td>Reset: NO</td>
</tr>
</tbody>
</table>

F1=Help  F2=Split  F3=Cancel  F9=Swap  F12=Cancel
SDSF ENC display will also show enclave CPU usage by various categories:

- **zAAP-Time**  Accumulated zAAP time, in seconds
- **zACP-Time**  Accumulated zAAP on CP time, in seconds
- **zIIP-Time**  Accumulated zIIP time, in seconds
- **zIIP-Time**  Accumulated zIIP on CP time, in seconds
- **zAAP-NTime** Normalized zAAP time, in seconds
- **zIIP-NTime** Normalized zIIP time, in seconds
You can view total enclave usage interactively using RMF

- Still, in the case of DB2 workload you may need DB2 accounting trace data to see detail on resource consumption for specific workload items

About SMF type 30 and SMF type 72 records

- SMF type 30 record contains resource consumption at the address space level
- SMF type 72 contains information at the z/OS WLM service class (or report class) level
  - Note – use WLM classification rules to exploit the ability to classify and analyze workload by WLM report class
DB2 DDF Relevant SMF Information

Understanding, Monitoring and Managing z/OS Enclaves © 2012 IBM Corporation
### DB2 Example

### Displaying A DB2 Thread

<table>
<thead>
<tr>
<th>User Defined Functions</th>
<th>Total CPU</th>
<th>Current CPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCB Time (SQL)</td>
<td>00:00:00.000</td>
<td>00:00:00.000</td>
</tr>
<tr>
<td>Wait for TCB Time</td>
<td>00:00:00.000</td>
<td>00:00:00.000</td>
</tr>
<tr>
<td>Elapsed Time (SQL)</td>
<td>00:00:00.000</td>
<td>00:00:00.000</td>
</tr>
<tr>
<td>SQL Events</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

---

**Elapsed time includes ‘think’ time between calls to DB2**

**IN-DB2 time represents time executing the calls**

**CPU times shows general CP time and time on zIIP**

**No delays in this example**
Thread Enclave Information

**Enclave token**: 5C0002D6C9

**What service class is this thread executing in?**

**IBM Software Group | Tivoli software**

**Understanding, Monitoring and Managing z/OS Enclaves**

**See enclave token, WLM service class, and service class performance index (PI)**

**WLM qualifiers used to select service class**

**WLM samples**

**Thread Enclave Information**

**Process Name**: DB2BP.EXE

**Function Name**: DB2_DRDA

**Subsystem Name**: DSN

**Network ID**: DISTSEVR

**Application Environment Name**: NO WLM ENVIRONMENT

**Start Parameters**:

**CLASSIFICATION WORK QUALIFIERS**

**Subsystem Type**: DDF

**User ID**: DNET581

**Transaction Class**: DDF Default

**Logical Unit Name**: SPOOL01NT

**Package Name**: DB2BP.EXE

**Correlation**: DB2BP.EXE

**Process Name**: DB2BP.EXE

**Package Name**: SPOOL01NT

**Collection**: NULLID

**Subsystem Name**: DSN

**Subsystem Priority**: N/A

**Subsys Coll Name**: N/A

**Performance Index Input Data for Percentile Response Time Goal**

<table>
<thead>
<tr>
<th>Observations</th>
<th>Percent</th>
<th>Count of Transacti</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>50%</td>
<td>165707</td>
</tr>
<tr>
<td>60%</td>
<td>62</td>
<td></td>
</tr>
</tbody>
</table>
Thread Reuse Complicates The Analysis

With thread reuse getpage, In-DB2 times, CPU times may all reflect multiple transaction executions

Many distributed/enclave based applications will employ thread reuse for efficiency
### Thread Detail

<table>
<thead>
<tr>
<th>User Defined Functions</th>
<th>TCB Time (SQL)</th>
<th>00:00:00.000</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Wait for TCB Time</td>
<td>00:00:00.000</td>
</tr>
<tr>
<td></td>
<td>Elapsed Time</td>
<td>00:00:00.000</td>
</tr>
<tr>
<td></td>
<td>Elapsed Time (SQL)</td>
<td>00:00:00.000</td>
</tr>
<tr>
<td></td>
<td>DB Events</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Triggers</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCB not in Enclave</td>
</tr>
<tr>
<td>Elapsed not in Enclave</td>
</tr>
</tbody>
</table>

**Elapsed Time**

- **Total CPU** = 00:00:00.000
- **Elapsed time** = 00:00:00.000
- **Elapsed (SQL)** = 00:00:00.000

**In-DB2 Time**

<table>
<thead>
<tr>
<th>Total</th>
<th>Current</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:02:47.604</td>
<td>00:00:00.000</td>
</tr>
<tr>
<td>00:01:00.160</td>
<td>00:00:00.000</td>
</tr>
<tr>
<td>00:01:26.226</td>
<td>00:00:00.000</td>
</tr>
<tr>
<td>00:00:00.000</td>
<td>00:00:00.000</td>
</tr>
<tr>
<td>00:00:00.000</td>
<td>00:00:00.000</td>
</tr>
</tbody>
</table>

**In-DB2 CPU Time**

- **Total** = 00:00:00.000
- **Current** = 00:00:00.000

**MVS Status**

- **Wait for TCB Time** = 00:00:00.000
- **Elapsed not in Enclave** = 00:00:00.000
- **TCB prior to Enclave** = 00:00:00.000

**Thread Reuse Complicates The Analysis - continued**

A large difference between Total Elapsed time and In-DB2 times indicate thread reuse. Consider these numbers when analyzing DB2 accounting data.
DB2 DDF Threads
WLM And Enclave Considerations

- DB2 thread options may influence enclave creation and how DB2 interacts with WLM
  - Impacted by such things as KEEP_DYNAMIC options, cursor with hold
  - Enclave creation may drive using velocity versus response time goals

Distributed Threads
Inactive mode

- DRDA Request
- Query Execution
- Enclave
- Think Time

Distributed Threads
Active mode

- DRDA Request
- Query Execution
- Enclave
DB2 Accounting Data Considerations

- DB2 provides options to control frequency and granularity of accounting record creation
  - ACCUMACC controls whether and when DB2 accounting data is accumulated by the user for DDF and RRSAF threads
    - Data accumulated for specified # of threads
    - Turned on if ZPARM ACCUMACC > 1
    - How it is summarized is based upon ACCUMUID setting
      - ACCUMID may be set as a combination of user id, workstation id, transaction id, etc...

- Rollup of accounting information can be useful for reducing the amount of SMF data created
  - Summarized information may be limited for problem investigation
  - Summarized information may hide the effects of problem thread in the rollup
  - Note – ACCUMAC and ACCUMID may be changed online
WLM
Service Classes Categorize Workload

- Classification rules assign incoming work to the appropriate WLM Service Class
- Classification rules group together logically related work
An Example Providing Attributes To DB2

### DISTRIBUTED THREAD DETAIL

<table>
<thead>
<tr>
<th>PLAN</th>
<th>+ Thread: Plan=DISTSERV Connid=SERVER Corrid=javaw.exe Auth</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>+ Dist : Type=DATABASE ACCESS, Luwid=G941491B.E205.090605130859=282</td>
</tr>
<tr>
<td></td>
<td>+ Location : 9.65.73.27</td>
</tr>
<tr>
<td></td>
<td>- simulation</td>
</tr>
<tr>
<td></td>
<td>+Location         IP Addr Port Cbuser Srvclnam Prod ID Workstation Name</td>
</tr>
<tr>
<td></td>
<td>+----------------- -------------- -------------- -------------- ---------------</td>
</tr>
<tr>
<td></td>
<td>+ 9.65.73.27      0941491B 448 dnet581  NT SQL09013 Ed@IBM</td>
</tr>
<tr>
<td></td>
<td>+ Transaction name: javaw.exe</td>
</tr>
<tr>
<td></td>
<td>+ TCP/IP Userid: dnet581</td>
</tr>
<tr>
<td></td>
<td>+ Remote Location Name = 9.65.73.27 Remote Location Luname =</td>
</tr>
</tbody>
</table>

Use CL/I settings to set the workstation name

Attributes may be passed that may in turn be used by WLM classification
Classifying The Workload
Specifying The WLM Objectives

- WLM provides an ISPF interface to define and manage the WLM Service Definition
- Note – z/OS Management Facility V1.12 provided a new management interface
Workload Manager As A DB2 Priority Mechanism

Examples of Thread Attributes
- AI (Accounting Information)
- CI (Correlation Information)
- CN (Collection Name)
- CT (Collection Type)
- LU (LU Name)
- NET (Net ID)
- PK (Package Name)
- PN (Plan Name)
- SI (Subsystem Instance)
- UI (Userid)

Thread attributes in WLM allow for considerable granularity in the classification of DB2 workloads into the appropriate Service Class.

Exploit the granularity to prioritize higher versus lower importance workload.
DB2 Enclave Workloads
Setting Optimal Goals

- Use Response Time goals when possible
  - Less need for ongoing maintenance and review
  - WLM will manage resources dynamically to achieve goals
- Response Time goals work well for certain types of DB2 workloads
  - DB2 Distributed workloads in e-business and WebSphere transactional type workloads
  - Transactional type workloads in general including distributed workloads that invoke Stored Procedures
  - Repetitive workloads that have multiple events for WLM to measure and manage
- Use a Velocity goal for the DB2 DDF address space
  - DDF address space has internal tasks that govern thread creation that should have high performance goal
Setting WLM Goals
Things To Note

- Considerations for DDF threads
  - For DDF inactive threads
    - Consider a two-period service class with a response time goal where 80-90% of the transactions complete in first period
  - For DDF always active threads
    - Consider velocity goals and use a single-period service class

- Look for overly simplistic Service Class definitions
  - Example – type DDF and nothing more than DB2 subsystem name
    - Does little to exploit the ability of WLM to prioritize DB2 workloads
    - Some workloads will inherently be more important than others

- Look for workloads that run longer than expected but use less resource than anticipated
  - Indicative of workload that may not being optimally classified

- Avoid too many service classes/periods
  - WLM analyzes service classes/periods in a round-robin manner
  - Too many and WLM is unable to manage them all effectively
  - Consider WLM reporting classes for report/analysis granularity and detail
Summary

- Enclaves are a pervasive mechanism for z/OS workload priority management
- DB2 is one of the primary exploiters of enclaves
- Many functions of DB2 exploit enclaves
  - DDF workload, Stored procedures, sequential prefetch and deferred writes
- Effective analysis of DB2 enclave based workload requires an understanding on the interaction of DB2 and z/OS enclaves
  - Understand WLM service classes, reporting classes and how they are defined in your environment
- Setting optimal WLM goals and priorities for DB2 requires an understanding DB2 and enclaves
  - Understand application flow and logic
  - Response time versus velocity goals
Thank You!
Tivoli With A z

This is a blog to discuss what is happening in the area of IBM zSeries, Tivoli, OMEGAMON monitoring, System Automation, and other relevant IBM Tivoli technology for z/OS performance and availability management.

OMEGAMON DB2 Near Term History

OMEGAMON DB2 has a very useful Near Term History (NTH) function. NTH provides an easy way to be able to retrieve and review DB2 Accounting and Statistics records from the past few hours of DB2 processing. The data is stored in a set of VSAM files allocated to the OMEGAMON collection task. How far back the history goes depends upon the size of the files and the amount of data being written to these files. Now some of the data volume is driven by the DB2 workload activity. Accounting records are typically written when a DB2 thread terminates processing, and it is the Accounting data that is often looked at by the analyst when studying what DB2 applications have been doing. Statistics records are created on a time interval basis. Usually, you will have much more accounting data than statistics data. Also, OMEGAMON has the ability to pull in additional trace RFCIDs to get information on things such as dynamic SQL activity.

To understand the amount of data being gathered by NTH, there are displays that show the number of records written to the NTH files, by type. In the example I show, you see an example of common NTH settings/options, and then you see the record count in the NTH record information display. If you look carefully you see that ‘Perf-Dyn SQL’ has a lot of records written relative to the other record types. This is a good way to understand the impact of enabling certain collection options, such as dynamic SQL collection, and see how many trace records are being gathered, as a result.

Posted by Ed Woods at 3:13 PM 0 comments

Check Out My Blog
http://tivoliwithaz.blogspot.com