BROADCOM®
MAINFRAME SOFTWARE

Hard Lessons Learned when using High Performance DBATs for Db2 for z/OS

John Campbell
Distinguished Technical Consultant
Broadcom

John.Campbell@Broadcom.com

Disclaimer

Certain information in this presentation may outline Broadcom's general product direction. This presentation shall not serve to (i) affect the rights and/or obligations of Broadcom or its licensees under any existing or future license agreement or services agreement relating to any Broadcom software product; or (ii) amend any product documentation or specifications for any Broadcom software product. This presentation is based on current information and resource allocations as of April 21, 2026, and is **subject to change or withdrawal by Broadcom at any time without notice**. **The development, release and timing of any features or functionality described in this presentation remain at Broadcom's sole discretion**.

Notwithstanding anything in this presentation to the contrary, upon the general availability of any future Broadcom product release referenced in this presentation, Broadcom may make such release available to new licensees in the form of a regularly scheduled major product release. Such release may be made available to licensees of the product who are active subscribers to Broadcom maintenance and support, on a when and if-available basis. The information in this presentation is not deemed to be incorporated into any contract.

Broadcom may use any feedback provided by you related to a Broadcom product or this presentation for any Broadcom business purposes (including but not limited to, preparation, reproduction, and distribution of derivative works based upon such feedback), without any obligation to you including consent or payment.

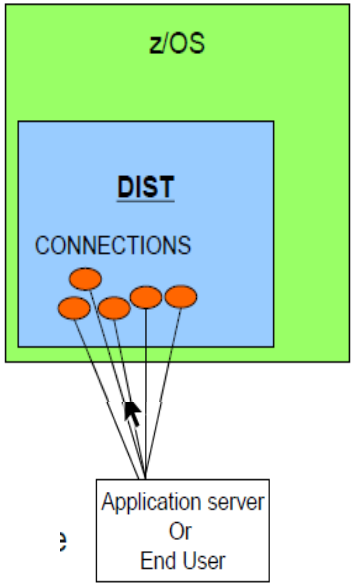
THIS PRESENTATION IS FOR YOUR INFORMATIONAL PURPOSES ONLY. Broadcom assumes no responsibility for the accuracy or completeness of the information. TO THE EXTENT PERMITTED BY APPLICABLE LAW, BROADCOM PROVIDES THIS DOCUMENT "AS IS" WITHOUT WARRANTY OF ANY KIND, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON INFRINGEMENT. In no event will Broadcom be liable for any loss or damage, direct or indirect, in connection with this presentation, including, without limitation, lost profits, lost investment, business interruption, goodwill, or lost data, even if Broadcom is expressly advised in advance of the possibility of such damages.

Agenda

- What is a connection
- What is a DBAT (thread)
- Db2 thread (DBAT) pooling and DRDA inactive connections
- What are High Performance DBATs
- Good candidates for High Performance DBATs
- General guidance for configuring High Performance DBATs
- Performance metrics and monitoring
- WLM and High Performance DBATs
- Issues
- Summary

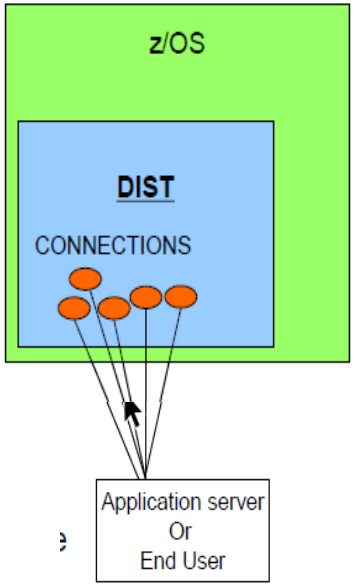
What is a connection

- DRDA over TCP/IP = IP address (domain) and port resulting in an end-point or socket and a control block in Db2 DIST system address space
  - Total number of connections governed by ZPARM CONDBAT (default = 10K)
    - At 80% of CONDBAT, DSNL074I message generated in Db2 MSTR system address space
      - Db2 member health is lowered to 50% of calculated value
    - At 90% of CONDBAT, DSNL074I message generated in Db2 MSTR system address space
      - Db2 member health is lowered to 25% of calculated value
    - If CONDBAT is exceeded
      - Message DSNL030I is written to the Db2 MSTR system address space log
      - Connection requests are rejected (socket failure)
      - Db2 member cannot be accessed remotely



What is a connection …

- DRDA over TCP/IP = IP address (domain) and port resulting in an end-point or socket and a control block on Db2 DIST system address space
  - Socket not released until remote requestor closes connection
  - Legacy myth that a connection is expensive in terms of memory usage
    - Only 2-4K bytes in memory footprint above 2GB bar in Db2 DIST system address space
    - Only 258 bytes in ECSA from z/OS Communications (TCP/IP) Server
    - Maximum value for CONDBAT is 150K



What is a DBAT (Thread)

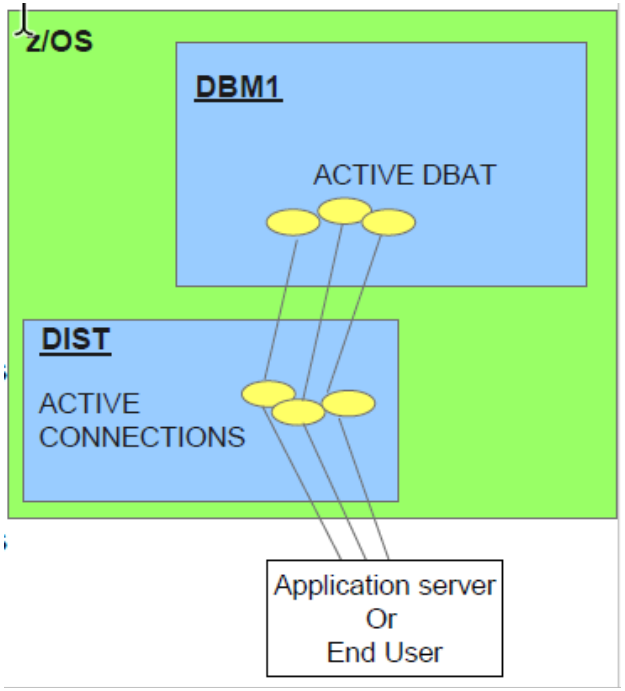
- DBAT = DataBase Access Thread
- Executed under WLM managed pre-emptible independent enclave SRB in the DIST address space
  - Pre-emptible so can be interrupted by z/OS when needed (like TCB)
  - Uses a z/OS work unit which is zIIP eligible
  - Independent enclave used by WLM to manage it
    - Represents transaction that can span multiple dispatchable units of work in one or more address spaces, and reported and managed as a single unit by WLM
- When a DBAT is created or pooled DBAT is reused
  - WLM classifies the work based on classification criteria in WLM Policy
  - Enclave is created to run the work, and a service class is assigned
  - Service class can use a response time goal or velocity goal

What is a DBAT (Thread) …

- A DBAT is associated with a connection until the connection terminates or goes inactive at commit
  - ZPARM MAXDBAT (default 200)
    - If MAXDBAT is hit, DSNL092 in Db2 –DIS DDF command output will increment
    - Requests queue up to CONDBAT, then requests are rejected

What is a DBAT (Thread) …

- Memory usage per DBAT in DB2 DBM1 system address space
  - Typical range 12KB – 2MB, or larger
  - About 12KB in 31-bit Private, the rest in 64-bit Shared (Private)
  - Use Db2 command –DIS THREAD(*) SERVICE(STORAGE) to see 31-bit memory usage



Db2 thread pooling and DRDA inactive connections

- ZPARM CMTSTAT determines whether DBATs are disassociated (disconnected) from a connection at commit time or at connection termination
  - CMTSTAT=INACTIVE (default)
    - At commit, DBAT is pooled to be reused by any new or resumed active connection request
    - Connection becomes inactive connection (formerly called Type 2 inactive DBAT)
      - Still holds a socket and counts against CONDBAT
    - ZPARM POOLINAC determines how long a DBAT remains unused in the pool before being terminated
      - Default 120 seconds
  - CMTSTAT=ACTIVE
    - DBAT stays associated with a connection until connection terminates

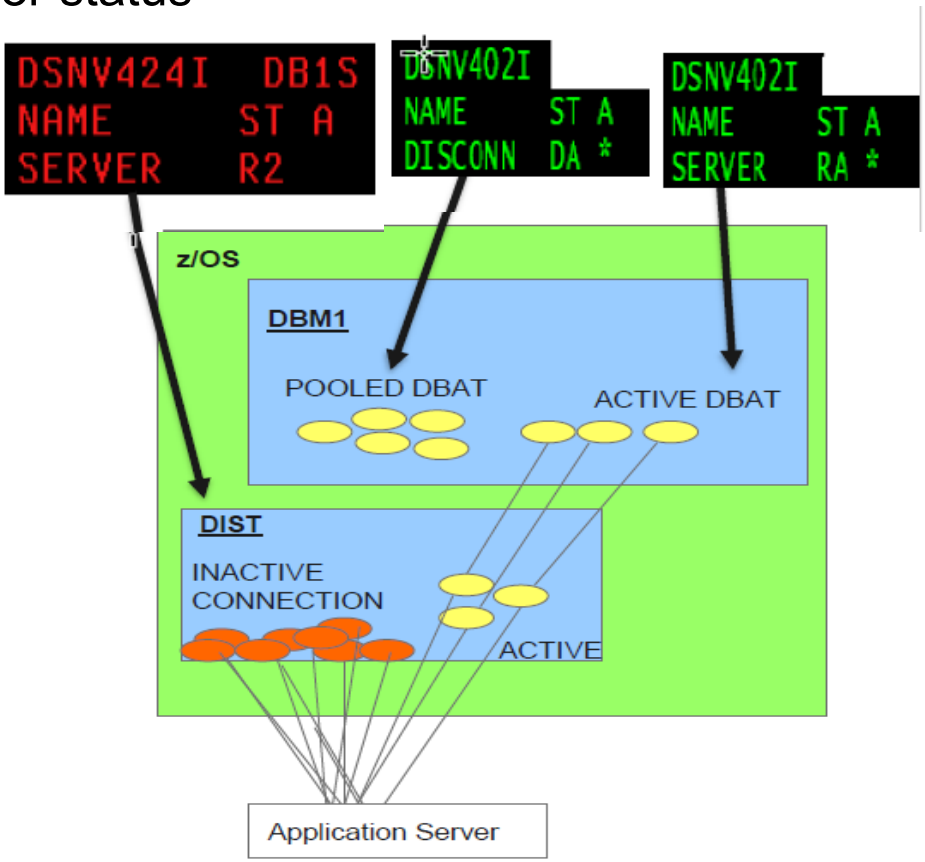
Db2 thread pooling and DRDA inactive connections …

- DBAT is pooled and dissociated (disconnected) from its connection when
  - ZPARM CMTSTAT = INACTIVE and at commit time
    - No use of packages with KEEPDYNAMIC(YES)
    - No open cursor with HOLD
    - No held LOB locator
    - No undropped DGTT
  - Reusing pooled DBATs
    - Reuse limit to avoid excessive thread memory growth and accumulation of many parent locks
    - Reused up to 500 times

- Conversely if any one one of the above negative conditions do exist then the DBAT is not pooled for reuse, and the connection does not get dissociated
  - Enhancements to remote statistics to help identify these “rogue” applications (IFCID 411), users (IFCID 412) and IP addresses (IFCID 365) that do not follow the best practices
    - Can use system monitor profiles to throttle any such applications or users

Db2 thread pooling and DRDA inactive connections …

- Use Db2 command –DIS THREAD(*) DETAIL for status
- Both active and pooled DBATs count towards MAXDBAT



Db2 thread pooling and DRDA inactive connections …

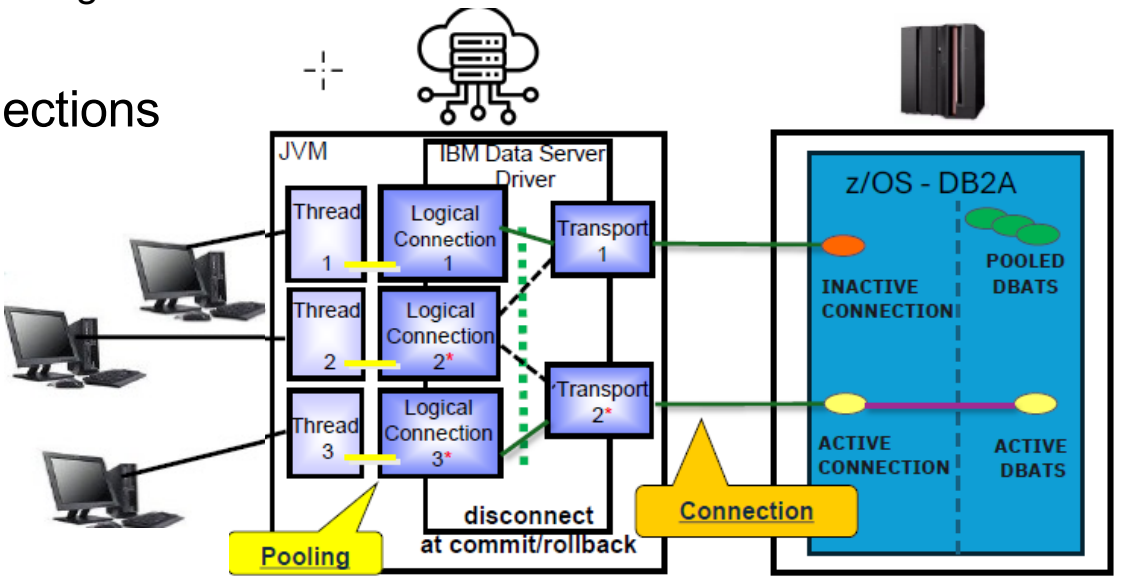
- Benefits of Db2 (DBAT) thread pooling
  - CPU savings in Db2, by reusing an existing (pooled) DBAT and avoiding the cost of creating and destroying DBAT
  - A DBAT can be serially reused by many active connections and fewer DBATs are needed for the same workload
    - Real memory savings in z/OS, by reducing the number of required DBATs
    - Virtual memory savings in Db2 DBM1 system address space, by reducing the number of DBATs
  - Capacity to support many more DRDA connections since they are disconnected from the DBAT when inactive
  - Maintaining WLM granularity at a unit of work boundary
- Disadvantages of Db2 (DBAT) thread pooling
  - When reused many times can lead to thread memory growth and accumulation of locks
  - Enforces RELEASE(COMMIT) execution for all packages

What are High Performance DBATs (HP DBAT)

- HP DBATs have the potential to provide significant CPU reduction for high volume transactions
  - Avoids repeated cost of active connection disconnecting from DBAT at end of transaction (commit)
    - DBAT pooled for reuse by a new active connective or resumed connection
    - Connection going in inactive
  - Supports true RELEASE (DEALLOCATE) execution
- Trade off
  - Reduced CPU resource consumption vs. increased requirement for DBATs and associated memory
- Must be used intelligently and implemented carefully
  - Risk of not having enough DBATs available for workloads not using High Performance DBATs
  - Worst case running out of DBATs completely or escalating thread management costs
- Requires DSNZPARM CMTSTAT=INACTIVE (the default)
- Db2 command MODIFY DDF PKGREL(BNDOPT|BINDPOOL) must be in effect
- Client application touches at least one package bound with RELEASE(DEALLOCATE)

What are High Performance DBATs (HP DBAT) …

- In-use transport on Application Server = Db2 for z/OS connection (active or inactive)
  - Transport keeps connection to Db2 even when no logical connection from client
- Transports can be reused by logical connections from many clients
  - User can use any of transports
  - All transports will become HP DBATs over time
    - Each will tie up an active connection and active DBAT over time
    - These connections can no longer be used for work from other servers as DBATs no longer get pooled at commit
  - **If any connection from an application server is a HP DBAT, likely all of them using the same transport pool will become HP DBATs**



Good candidates for High Performance DBATs

- Key point is whether the application does many commits (e.g., >= 500) before deallocating or disconnecting and at least 1 transaction/second per connection under normal conditions
  - Density of transaction arrival rate is very important to avoid wasteful use of DBATs
  - Ideal for “skinny” short running, well behaved, frequent transactions
  - Short-lived connections will see no benefit from HP DBATs
    - Frequent connection termination will increase DIST CPU (SRB) time

Good candidates for High Performance DBATs …

- Use Db2 Accounting Trace data (IFCID 3) – Plan level accounting information
  - Summarize accounting trace data by remote location and look for instances where:
    - Transactions with many occurrences per connection
      - (COMMIT RECEIVED+ABORTS RECEIVED/INITIATED FROM REMOTE SITE) > 500
      - (QLACCOMR + QLACCABRR) / QLACCNVR > 500
    - Using accounting data by client account info might be easier to implement HP DBATs than by IP address
    - However, QLACCNVR (number of connections that were initiated from the remote site) is often zero as not always incremented in the Accounting Trace
      - Value in the Statistics Trace data is correct

Good candidates for High Performance DBATs …

- Use Db2 Statistics Trace Class 7 (IFCID 365) – Detailed Location Statistics
  - Information provided by IP address every ZPARM STATIME_DDF interval
  - QLSTCNVR - Number of (logical) connections that were initiated from the remote site
    - (COMMIT RECEIVED+ABORTS RECEIVED/INITIATED FROM REMOTE SITE) > 500
      - (QLSTCOMR + QLSTABRR) / QLSTCNVR > 500
    - IP addresses with a high number would be a good candidate for HP DBATs

Good candidates for High Performance DBATs …

- Combining Db2 Statistics and Accounting Trace data
  - Assumption: long lived connections
  - Issue Db2 -DIS LOCATION DETAIL command for a count of connections from each remote location
  - Use Accounting Report with REDUCE for a short interval to calculate number of transactions per second
    - During peak online period
    - During batch window
  - Minimum target: at least one transaction per second over each connection
  - Ideal target: high volume, short running, well-behaved transactions

General guidance for configuring High Performance DBATs

- Ensure safe starting position
  - Check that all existing Db2 Connect client packages for ODBC / JDBC packages (SYSLN***, SYSLH***, etc.) in the Db2 package collection NULLID are bound with RELEASE (COMMIT)
    - DB2Binder utility defaults to **releasePackageResourcesAtCommit(false)** for packages (9.7 FP 3a)
      - So, you would get RELEASE(DEALLOCATE) execution by default!
    - Rebind these packages with RELEASE(COMMIT) to avoid excessive use of HP DBATs
  - Rebind the DB2 Connect client packages into an alternate Db2 package collection (e.g., NULLID_HPDBAT) with RELEASE(DEALLOCATE)
  - On Db2-server side, use identifier in a System Monitor Profile (in SYSIBM.DSN_PROFILE_TABLE) to filter and set CURRENT PACKAGE PATH special register to point good candidate application workload to use the alternate Db2 package collection (e.g., NULLID_HPDBAT) so as to use RELEASE(DEALLOCATE) packages

General guidance for configuring High Performance DBATs …

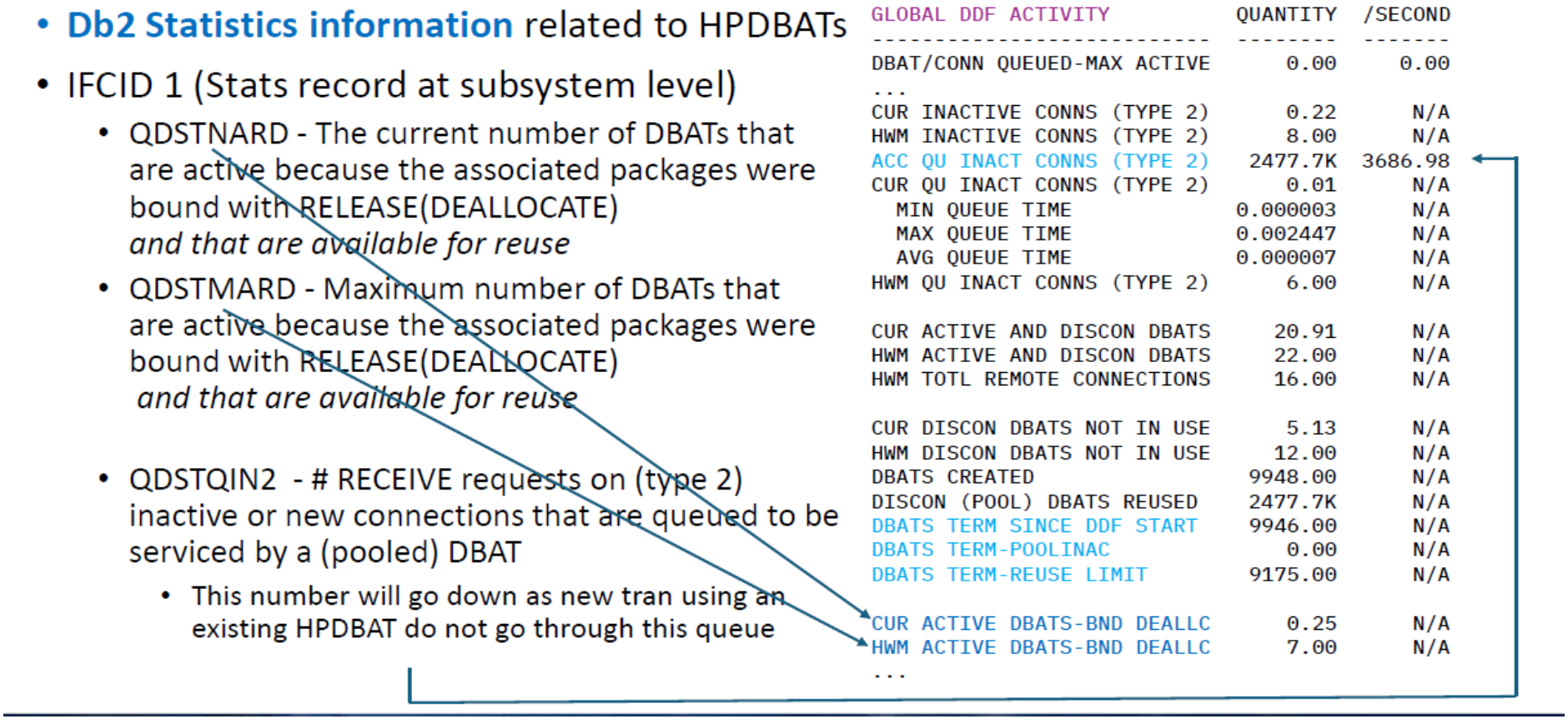
- Plan for increased demand for DBATs, possible impact on MAXDBAT and ensure availability of pooled DBATs for non-HP DBAT workload
- Plan for impact on MAXDBAT when application server instances are dynamically started and stopped e.g.,
  - By Kubernetes or other container application managers
- Be prepared to intervene and use -MODIFY DDF PKGREL(COMMIT) to switch off HP DBATs at first signs of DBAT congestion i.e., overuse of DBATs
  - Risk of not having enough DBATs available for workloads not using High-Performance DBATs
  - Worst case running out of DBATs completely or escalating thread management costs

Performance Metrics and Monitoring

- Key metrics to compare before and after HP DBAT activation
  - Accounting Report summarized by remote location
    - Class 2 Elapsed Time
    - Class 2 CPU Time
    - Class 3 Wait Times
  - Statistics Trace/Report
    - DDF preemptible SRB Time, CPU Time per Commit, CP CPU Time, preemptible IIP SRB Time
    - DDF Virtual Memory above and below 2GB Bar
    - DBM1 Virtual Memory above and below 2GB Bar
    - Global DDF Activity for DBAT and Connection metrics
  - DDF Statistics
    - Remote Location Statistics (IFCID 365)
    - Remote Application Statistics based on CLIENT_APPLNAME (IFCID 411)
    - Remote User Statistics based on CLIENT_USER (IFCID 412)
    - Client Inventory (IFCID 417)
    - ZPARM STATIME_DDF specifies the collection interval

Performance Metrics and Monitoring …

- **Db2 Statistics information** related to HPDBATs
- IFCID 1 (Stats record at subsystem level)
  - QDSTNARD - The current number of DBATs that are active because the associated packages were bound with RELEASE(DEALLOCATE) *and that are available for reuse*
  - QDSTMARD - Maximum number of DBATs that are active because the associated packages were bound with RELEASE(DEALLOCATE) *and that are available for reuse*
  - QDSTQIN2 - # RECEIVE requests on (type 2) inactive or new connections that are queued to be serviced by a (pooled) DBAT
    - This number will go down as new tran using an existing HPDBAT do not go through this queue



GLOBAL DDF ACTIVITY	QUANTITY	/SECOND
DBAT/CONN QUEUED-MAX ACTIVE	0.00	0.00
...		
CUR INACTIVE CONNS (TYPE 2)	0.22	N/A
HWM INACTIVE CONNS (TYPE 2)	8.00	N/A
ACC QU INACT CONNS (TYPE 2)	2477.7K	3686.98
CUR QU INACT CONNS (TYPE 2)	0.01	N/A
MIN QUEUE TIME	0.000003	N/A
MAX QUEUE TIME	0.002447	N/A
AVG QUEUE TIME	0.000007	N/A
HWM QU INACT CONNS (TYPE 2)	6.00	N/A
CUR ACTIVE AND DISCON DBATS	20.91	N/A
HWM ACTIVE AND DISCON DBATS	22.00	N/A
HWM TOTL REMOTE CONNECTIONS	16.00	N/A
CUR DISCON DBATS NOT IN USE	5.13	N/A
HWM DISCON DBATS NOT IN USE	12.00	N/A
DBATS CREATED	9948.00	N/A
DISCON (POOL) DBATS REUSED	2477.7K	N/A
DBATS TERM SINCE DDF START	9946.00	N/A
DBATS TERM-POOLINAC	0.00	N/A
DBATS TERM-REUSE LIMIT	9175.00	N/A
CUR ACTIVE DBATS-BND DEALLC	0.25	N/A
HWM ACTIVE DBATS-BND DEALLC	7.00	N/A
...		

Performance Metrics and Monitoring …

- Db2 Statistics information related to HPDBATs
- IFCID 1 (Stats record at subsystem level)
  - QLSTHIPRF - Number of times when the use of HPDBATs [prevented thread pooling]
  - QLSTNTPLH - Number of times that threads that were used by connections from the remote site were terminated because a HPDBAT remained in the pool longer than the than the POOLINAC value, or remained longer than 120 seconds if the POOLINAC value was 0
  - Also available in IFCID 365 per remote location

DRDA REMOTE LOCS	THREAD TERMINATION	QUANTITY
QUEUED - PROFILE EXCEPTION		0.00
TERMINATED - PROF EXCPTION		0.00
ABENDED		0.00
CANCELED		0.00
TERMINATED - POOLINAC TIME		0.00
TERMINATED - SOCKET CLOSED		0.00

DRDA REMOTE LOCS	GENERAL	SENT	RECEIVED
CONNECTIONS		0.00	0.00
CONNECTIONS QUEUED		0.00	
CONNECTIONS DEALLOCATED		0.00	
SQL STATEMENTS		0.00	184.9M
COMMITS		0.00	4976.0K
ROLLBACKS		0.00	0.00
ROWS		65367.3K	0.00
MESSAGES		189.9M	189.9M
BYTES		37201.1M	30297.4M
BLOCKS		67662.2K	0.00
...			
DRDA REMOTE LOCS	CLIENT CONDITIONS	QUANTITY	
WITH HOLD CURSOR NOT CLOSED		0.00	
DGTT NOT DROPPED		0.00	
KEEPDYNAMIC PACKAGES USED		0.00	
HIGH PERF DBATS USED		2484.1K	
HELD LOB LOCATORS EXIST		0.00	
SP COMMIT PERFORMED		0.00	
DRDA REMOTE LOCS	DDF CONN. DETAILS	QUANTITY	
ACTIVE CONNS FROM LOC - SNAP		16.00	
ACTIVE CONNS FROM LOC - INT. HWM		16.00	
ACTIVE DBATS FOR LOC - SNAP		15.76	
ACTIVE DBATS FOR LOC - INT. HWM		16.00	

Performance Metrics and Monitoring …

- IFCID 411 (Client application statistics)
  - QLAPHIPRF -The number of times that the application used a HPDBAT
    [, preventing Db2 from pooling the DBAT]
- IFCID 412 (Client enduser statistics)
  - QLAUHIPRF - The number of times that an application run by the specified client user-id used a HPDBAT
    [, preventing Db2 from pooling the DBAT]

```
REMOTE USER STATISTICS  VALUE
----------------------  ------------------
USER NAME               USRT001
PRODUCT ID              JCC04360
PRODUCT LEVEL           4

REMOTE USER STATISTICS                QUANTITY
------------------------------------  --------
REQUESTS
  COMMIT                               2469.8K
  ABORT                                   0.00
  REST SERVICE                            0.00
  PROFILE SET SPECIAL REGISTERS           0.00
  PROFILE SET GLOBAL VARIABLES            0.00

REMOTE USER STATISTICS                QUANTITY
------------------------------------  --------
DBAT NOT POOLED
  WITH HOLD CURSOR NOT CLOSED             0.00
  DGTT NOT DROPPED                        0.00
  KEEPDYNAMIC PACKAGES USED               0.00
  HIGH-PERF DBAT USED                  2457.1K
  HELD LOB LOCATOR EXIST                  0.00
  STORED PROCEDURE COMMIT                 0.00
...
```

WLM and High Performance DBATs

- DDF transactions are run as independent enclaves to provide an anchor for a DDF client transaction and to enable
  - Assignment of service class
  - Assignment of performance goal
  - WLM can dynamically manage resources based on the performance goal
- Differences in the concept of a transaction
  - Db2 perspective
    - Start at 1st SQL call
    - End at Commit/Rollback
  - WLM perspective
    - Life of independent enclave

WLM and High Performance DBATs …

- Emerging issue
  - With large systems with many engines and very high DDF transaction rates the volume of concurrent enclave create/delete activity is causing high degree of z/OS SRM spin lock contention
  - Consequences
    - Driving up CPU resource consumption in Db2 DIST system address space
    - Driving up the number of connections and DBATs which may
      - Negatively affect Health of Db2 member
      - Negatively affect the WLM Weight of Db2 member
      - Result in clients with Sysplex Workload Balancing routing work elsewhere

WLM and High Performance DBATs …

- Prior to Db2 12 APAR PH34378 which applies to HP DBATs only
  - Enclave is deleted when client application commits or rollbacks
  - A new enclave is created for each client transaction
  - Each individual client transaction is processed under it's own appropriate WLM response time or velocity goal
- After Db2 12 APAR PH34378 which applies to HP DBATs only
  - Independent enclave is no longer deleted at the end of client transaction
  - Life of an independent enclave is now up to 500 (Db2 13) ended Db2 client transactions serially reusing a HP DBAT
  - Dramatic reduction in concurrent enclave create/delete activity and z/OS SRM spin lock contention
  - From WLM perspective, these 500 client transactions are part of a single long running enclave
  - Db2 APAR PH41024 adds support for WLM APAR OA61811 which enables Db2 to tell WLM how many actual transactions the enclave really executed -> RMF WLM reports now correct

WLM and High Performance DBATs …

- What was the motivation for the Db2 change?
  - For some specific high volume transaction environments, retaining the enclave and not creating a new enclave for each transaction provides CPU performance benefit
- Consequences
  - WLM administrator must now know which DDF workloads are using HP DBATs
    - Segregate HP DBAT workloads out into a separate WLM service class and assign single period velocity goal
    - If not possible, use multi period control for existing service class and assign velocity goal for the 2nd period
  - Limits flexibility in managing DDF client workload
  - Failure to use a velocity goal, will mean that the workload is not managed as you would expect

Common issue

- Design challenge is to find a good identifier to segregate out target application workload to route to the alternate collection to use RELEASE(DEALLOCATE) packages and to a separate WLM service class to assign velocity goal
  - Multiple applications running on the same application server that are not homogeneous
  - Single application running on a dedicated application server where individual transactions are not homogeneous
  - Application servers being dynamically started and stopped

Common issue …

- For Db2 server-side configuration
  - Likely too many connections will end up using HP DBATs
    - Driving up demand for MAXDBAT
  - Consider modifying the application to set client information fields using JAVA APIs (ApplicationName), and set methods for .NET and CLI (ClientApplicationName), and use this identifier
- Similar issue for client-side configuration
  - Client cannot distinguish between HP DBAT connections and non-HP DBAT connections
  - If setting CurrentPackageSet (.NET) or currentPackageSet (JDBC) at client to alternate collection to use RELEASE(DEALLOCATE) packages then all connections will use HP DBATs with the same consequences

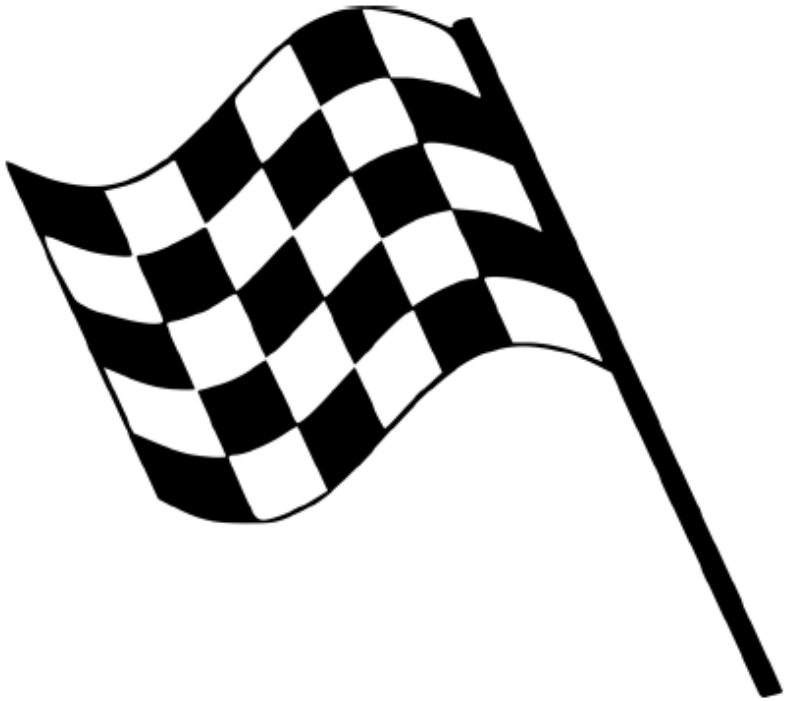
Other issues to consider

- Do not over-inflate the client connection pool definitions, otherwise it will considerably drive up the demand for HP DBATs
  - Using many HP DBATs with low utilisation of each DBAT
  - Increased risk of reaching MAXDBAT
  - Risk of hitting POOLINAC with connections killed
- BIND/REBIND, SQL DDL and Online REORG cannot break in on High-Performance DBAT
  - Must issue command -MODIFY DDF PKGREL (COMMIT) to overlay BNDOPT | BNDPOOL option
  - Switch to PKGREL(COMMIT) will occur gradually
  - Once completely switched, allows BIND/REBIND, SQL DDL and online REORG to break in

Summary

- HP DBATs have the potential to provide significant CPU reduction
- Must be used intelligently and implemented carefully
- Target clients which do more than 500 commits before deallocating or disconnecting, and most importantly density of transaction arrival rate
- Use Velocity Goal to let WLM effectively manage the workload
- Ensure safe starting position with NULLID packages bound with RELEASE(COMMIT)
- Challenge to find good identifier to filter on in System Montor Profile (and WLM policy) to distinguish between HP DBAT connections and non-HP DBAT connections
- Plan for impact on MAXDBAT and the availability of pooled DBATs for non-HP DBAT workload
- Be prepared to intervene and use -MODIFY DDF PKGREL(COMMIT) to switch off HP DBATs at first signs of DBAT congestion

Questions



Thank You

BROADCOM®

MAINFRAME SOFTWARE