
Rule CIC286: Consider decreasing protected threads for CICS-DB2 Entry

Finding: The average number of threads that were reused for the CICS-DB2 Entry was sufficiently low that you should consider decreasing the number of protected threads for the CICS-DB2 Entry.

Impact: This finding should normally have a LOW IMPACT on the performance of CICS tasks in the region that use the CICS-DB2 connection. However, the finding could reduce the overhead and storage requirements involved with the CICS-DB2 connection.

Logic flow: This is a basic finding, based upon an analysis of the CICS statistics. This finding applies only with CICS/Transaction Server for OS/390 Release 1.2 and subsequent releases of CICS.

Discussion: The CICS DB2 attachment facility creates an overall connection between CICS and DB2. CICS applications use this connection to issue commands and requests to DB2.

A CICS transaction accesses DB2 via a *thread*, which is an individual connection into DB2. The transaction uses the thread to access resources managed by DB2. Threads are created when they are needed by transactions, at the point when the application issues its first SQL or command request.

When CICS is connected to DB2 Version 5 or earlier, specially created subtask TCBs are used as the thread TCBs. When CICS is connected to DB2 Version 6 or later, open TCBs are used as the thread TCBs. The open TCB associates itself with the thread, and becomes the thread TCB until it dissociates from the thread.

There are three types of threads: Command threads, Pool threads, and Entry threads.

C Command threads are used by the CICS DB2 attachment facility for issuing commands to DB2 via the DSNB transaction.

C Pool threads are used for all transactions and commands that are not using a Command thread (because the transaction is not DSNB), are not using an Entry thread (because an Entry thread had not been defined for the transaction), or have been “overflowed” to the pool because a Command thread or an Entry thread was not available.

-
- C One or more Entry thread categories can be defined (using the DB2ENTRY definition) for specific transactions or groups of transactions. Entry threads are used for transactions that need to be managed separately from the normal transactions, or for transactions that have special accounting needs.
 - C Entry threads can have higher, equal, or lower priority relative to the CICS main task¹.
 - C Thread limits can be specified for Entry threads to ensure that certain transactions always have access to a thread, or to make sure that certain transactions do not monopolize resources (that is, thread limits can be used to “throttle back” types of work).
 - C The THREADWAIT controls the action that is taken if no Entry thread is available (wait for an Entry thread, overflow to pool, or abort).
 - C Different accounting levels and different security authorizations can be specified for Entry threads.
 - C Entry threads can be “protected” so the thread is not terminated immediately when released.

When a thread is no longer needed by the transaction, the thread is released. The thread release typically occurs after syncpoint completion. Command threads will be terminated immediately upon release.

Pool threads will be terminated immediately upon release, unless there is a queued transaction specifying the same plan and using a pool thread. In this case, the pool thread can be reused. This condition (a queued transaction specifying the same plan and using a pool thread) occurs rarely. Consequently, pool threads are rarely reused.

When a transaction releases an Entry thread, the thread can be reused by another transaction specifying the same plan and defined in the same DB2ENTRY. If there is no queued transaction defined in the same DB2ENTRY, Entry threads may be terminated immediately, or may be retained for a period of time, depending on whether “thread protection” has been specified.

Thread protection is specified by the PROTECTNUM attribute of the DB2ENTRY definition. The PROTECTNUM attribute specifies the maximum number of protected threads allowed for the DB2ENTRY.

¹The PRIORITY attribute is ignored with DB2 Version 6 and subsequent versions.

When an Entry thread is released, the thread will be “protected” if the current number of protected threads for the Entry (plus the thread being released) is not greater than the value of the PROTECTNUM attribute of the DB2ENTRY definition. The thread will stay “protected” for an interval controlled by the PURGECYCLE attribute of the DB2CONN² definition.

The default value for the PURGECYCLE attribute is specified as 30 seconds. A protected thread is terminated only after two complete purge cycles if it has not been reused. However, the purge cycle calculation does not begin as each thread is released, but is a timing value (conceptually, a timer interrupt) that runs continuously. The algorithm for an individual thread simply counts the number of purge cycle endings that occur after the thread is released.

The implication of this logic is that, with the default purge cycle value of 30 seconds, a specific protected thread could be protected for a range of 30 to 60 seconds depending on when the first purge cycle ending occurred with respect to the thread release.

C The protected thread could encounter a purge cycle ending immediately upon thread release and encounter another purge cycle ending 30 seconds later. These two purge cycle endings would satisfy the “protected for two purge cycles” logic even though only 30 seconds had lapsed.

C The protected thread could encounter a purge cycle ending 30 seconds after thread release and encounter another purge cycle ending 30 seconds later. These two purge cycle endings would satisfy the “protected for two purge cycles” logic with 60 seconds lapsed time.

The average thread protected interval, for transactions arriving randomly, would be 45 seconds.

There is a significant reduction in overhead when threads are reused. On the other hand, unused protected threads generate overhead since storage is required and CPU overhead is required for TCB scans. Consequently, there is a performance tradeoff: it is desirable to use protected threads if there is sufficient transaction rate for transactions defined in the same DB2ENTRY that the threads are likely to be reused, while it is undesirable to use protected threads if the protected threads are not likely to be reused.

If the average number of threads reused in 45 seconds is less than the number of protected threads, too many threads are protected (that is, the value of the PROTECTNUM attribute is too large).

²Note that the PURGECYCLE attribute is specified on the DB2CONN definition. This means that the specified PURGECYCLE value applies to all DB2ENTRY definitions.

CICS-DB2 Entry statistics are available in MXG file CICDB2RE. The information in CICDB2RE is presented as one record for each Plan, for each Entry. CPExpert uses data in CICDB2RE to calculate the average number of threads reused for each DB2ENTRY, using the following algorithm:

$$\text{Average threads reused per DB2ENTRY} = \frac{\text{Threads reused}}{\text{Recording interval duration}}$$

where

Threads reused = D2RTHRRE
Recording interval duration = DURATM

The average threads reused reflects the efficiency of the thread protection process, based on the setting of the PROTECTNUM attribute for the DB2ENTRY and the PURGECYCLE value. As mentioned above, with the default value of 30 seconds for the PURGECYCLE attribute, the average thread protected interval would be 45 seconds.

CPExpert produces Rule CIC286 when the average threads reused in 45 seconds is less than the MINIMUM OF (the number of protected threads specified by the PROTECTNUM attribute, and the **THRDRUSE** guidance variable). The value of the PROTECTNUM attribute is contained in the MXG variable D2RTHPLM. The default value for the **THRDRUSE** is 2000 (the maximum number of TCBs and consequently the maximum number of threads). This default value of 2000 effectively nulls the THRDRUSE guidance variable and allows the logic to depend on the value of the PROTECTNUM attribute for the DB2ENTRY.

Suggestion: Having a larger number of threads protected than required for an Entry means that threads are unavailable for other Entries. This means that more TCBs are required, and means that more overhead and more storage is required to support the TCBs.

If Rule CIC286 is produced regularly, you should consider the following alternatives:

C Specify a smaller value for the PROTECTNUM attribute. If Rule CIC286 is produced, threads are reused (rather than being terminated) at a rate per 45 seconds, of less than the number of protected threads. This logic is independent of the PROTECTNUM attribute; threads would be terminated (rather than reused) if no thread were available. Since the threads were reused at a rate per 45 seconds less than the PROTECTNUM value, the PROTECTNUM value is too large.

You should consider specifying a smaller value than is currently specified for the PROTECTNUM attribute for the DB2ENTRY identified by Rule CIC286.

C **Modify guidance.** You can modify the THRDRUSE guidance variable in USOURCE(CICGUIDE) if you feel that Rule CIC286 is produced prematurely. You should specify a smaller value for the THRDRUSE guidance variable than is specified by the PROTECTNUM attribute for the DB2ENTRY. The CPEXpert logic selects the MINIMUM of either the PROTECTNUM value or the THRDRUSE guidance. Consequently, specifying a smaller value for the THRDRUSE guidance would cause the THRDRUSE value to be the controlling value.

Reference: *CICS/TS Release 1.3 CICS DB2 Guide*: Section 5.4 (Creating, using, and terminating threads)

CICS/TS Release 1.3 Resource Definition Guide: Section 5.1.3 (DB2CONN) and Section 5.1.4 (DB2ENTRY)

CICS/TS for z/OS Release 2.2 *CICS DB2 Guide*: Section 5.5 (Selecting thread types for optimum performance)

CICS/TS for z/OS Release 2.2 Resource Definition Guide: Section 2.3.4 (DB2 connection definition attributes) and Section 2.4 (DB2 entry definitions)