What I Learned This Month: Enclave Confusion

Scott Chapman
American Electric Power

The use of enclaves in z/OS is a topic that still seems to cause confusion among z/OS professionals. Recently I ran into a few interesting situations which had enclaves at the heart of the matter. Fair warning: this is another z/OS-centric month. Maybe next month I'll learn about something not involving mainframes.

First, a brief review: enclaves are a construct for running a new unit of work in a way that is somehow unrelated to the task (address space) that is initiating the unit of work.

Independent enclaves represent a new transaction that gets its own performance characteristics independent of the address space that spawned it.

Dependent enclaves are the opposite: they are a continuation of the spawning address spaces and do inherit the performance characteristics of their parents.

Work-dependent enclaves are a continuation of an existing independent enclave into a new enclave. Work-dependent enclaves inherit the performance characteristics of the independent enclaves that created them. Contrary to what it seems, they were created for reasons other than just causing confusion!

Recently we started up IBM's Omegamon DB2 buffer pool analysis tool. We hadn't used it in some time, but we had put in a new version and wanted to try it out again. Long ago I had put the started task for the collector in the SYSSTC service class so we could be reasonably assured of not losing significant records during our analysis period. Yet we were losing more records than we were capturing. RMF reported that the collector task was significantly delayed for CPU. SYSSTC is the most important work in the system outside of the operating system pieces that z/OS itself protects. If the system is so CPU-constrained that SYSSTC work is delayed for CPU, then likely you're getting no real work done.

But that wasn't the case in this situation.

I suspected that there was some subtask of the collector that was not getting dispatched for some reason. I was curious how many subtasks the collector might have, so I decided to try out some newfound knowledge. I went into IPCS and set the source to "ACTIVE" so that I could use it against my current TSO address space. I then went into the hidden option 2.6I[1] and selected "WEBINFO" to list out the current WEBs (Work Execution Blocks) for the collector task.
I didn't really expect to learn anything from this; I was mostly just curious to see how it would show up in IPCS. However, I noticed that the task had one "WEBE"

[1] I had just learned about the hidden Level 2 Toolkit Functions menu from Cheryl Watson's Tuning Letter (2012, No 3), so thanks, Cheryl!

listed for it. I surmised that this might mean “Work Execution Block, Enclave” and so this might indicate that there was an enclave involved.

I went to SDSF’s enclave display, and sure enough there was one—and it was an independent enclave! Remember, that means that it gets classified separately from the task that created it, so the SYSSTC classification did not carry forward to the enclave. In fact, the enclave was in a very low-importance service class that I have set up to catch workloads that I don’t expect to be on the system. That surprised me because I would have thought that the enclave for a started task might default to the default service class for started tasks. The reason why it did not became obvious when I looked to see what subsystem the enclave was assigned to: it was assigned to “DB2”!

From WLM’s perspective, subsystem names are just names. The application creating enclaves can tell WLM to use whatever subsystem name it wants. IBM documents several standard subsystem names and “DB2” is defined as sysplex-parallel queries. That is, queries that DB2 splits into multiple pieces that run in parallel across multiple systems in the sysplex. We don’t allow that, hence I have the default service class for that subsystem set to my low-importance catch-all service class. Because this enclave was created with the “DB2” subsystem identifier, that’s where it went, and that’s why the collector couldn’t collect the data it needed. The enclave was at the bottom of the heap in terms of importance, instead of at the top.

Applications don’t have to reuse the IBM-provided names; they can use their own subsystem names. We have an ISV product (Progress Shadow) that does just that—but the subsystem name is documented. Unfortunately, the current Omegamon documentation had not mentioned that it was going to create an enclave with the “DB2” subsystem identifier. Even if it had, reusing a subsystem name for a completely different type of work seems wrong to me. I did relent and added a classification rule to get this particular Omegamon enclave into SYSSTC, but it’s one of those things that just feels like it’s going to cause problems someday.

I didn’t need to use IPCS to track down the fact that it was creating an enclave; a quick look at SDSF’s enclave panel would have been sufficient. I had no idea that the product was creating an enclave, but in the future if something is being strangely delayed more than it should be, I might check the enclave panel just to be sure it’s not doing something unexpected.

A few days later when I noticed our Shadow transactions suffering, I already knew that enclaves were involved, so I started by looking at the SDSF enclave panel as well as the RMF III enclave panel. Part of the problem was immediately evident: the transactions were showing no zIIP CPU usage. We had turned off the zIIP offload option in the product some time ago and simply had forgotten to

turn it back on. So at least that part of that problem was easily found and resolved.

While I was on the enclave panel, I noticed that we're starting to see more DDF transactions using work-dependent enclaves. Apparently this is done when DB2 wants to use CPU parallelism for a transaction. I'm seeing more of those than I think I probably should though, so there's another thing for me to add to the list of things to look into. A performance analyst's work is never done!

As always, if you think I got it all wrong or have questions or comments, you can reach me via email at sachapman@aep.com.