Is there any performance advantage to marking a page read-only if I had no intention of writing to it anyway?

October 9, 2023

Raymond Chen

Suppose you have a chunk of memory that you fill with data, but don’t intend to write to after it has been initialized. Is there any performance benefit to changing its page protection to read-only?

Not really.

In theory, a CPU could take advantage of this, but in practice, they don’t. The CPU already knows that all of the operations are reads because that’s all you’ve ever done. The cache line for the memory will remain clean, even if the underlying page is read-write. Besides, it’s possible that the same physical page is mapped read-write via some other virtual address, so the CPU has to be ready for writes anyway.

One page table trick that does provide performance improvements is large pages: This reduces TLB pressure by allowing a large block of memory (the exact size varying from processor to processor) to occupy a single TLB slot.

But wait, don’t go crazy and start allocating *all* of your memory with large pages. “They told me large pages have better performance, so let’s make the whole plane out of large pages!”

Large pages are large. If you allocate a large page but use only a little bit of it, then you haven’t actually saved any TLB entries. It’s like buying a bunch of large storage boxes because you read that they’re more efficient, but filling each one with only a small amount of stuff. The point of buying the large boxes is so you can use fewer of them to pack the same amount of stuff. If you get large boxes to replace the same number of small boxes, then you haven’t really saved anything. You just over-spent on boxes.

So should you use large pages if you promise to fill them up?

Background reading: Some remarks on `VirtualAlloc` and `MEM_LARGE_PAGES`.

Getting access to large pages is already a bit of a hassle, since it requires “lock pages” privilege, which is normally assigned only to administrators. Furthermore, allocating them is a hassle, and once you have them, the large pages are non-pageable. In practice, large pages are useful only for programs like SQL Server that require very large quantities of memory and run on systems that are dedicated to running that program exclusively.

I mean, you can try it on your Home Edition, but you probably won’t notice much of a benefit. Your program is unlikely to be in a situation where TLB pressure is what’s slowing you down.