Under what conditions could a commit of reserved memory fail?

devblogs.microsoft.com/oldnewthing/20170419-00

April 19, 2017

Raymond Chen

A customer's program reserves a large chunk of address space (`VirtualAlloc` with `MEM_RESERVE`) and commits memory into it as necessary (`VirtualAlloc` with `MEM_COMMIT`). They received crash reports that indicated that the commit was failing, which leads to the program reporting a fatal error. Unfortunately, in between the failed `Virtual-Alloc` and the code that generated the fatal error, the error code set by `VirtualAlloc` was lost. (The customer sheepishly acknowledged that this was due to "poor code".)

The customer wanted to know what situations could result in the failure to commit memory that had previous been successfully reserved.

The obvious reason is that you are out of memory. The system will satisfy commit either from physical memory or from the page file.[1] The page file will not extend indefinitely, so you will eventually run out of memory if you keep committing.

A less obvious reason is that the process may be running inside a job that has a commit limit.

Once the customer knew what to look for, they were able to verify that the program running into the problem was indeed running inside a job. Furthermore, the job memory limit was set to 155MB, which is low for the type of work the program normally performs.

Mystery solved. Of course, the next mystery is why the program is running in a job with a low commit limit, but the customer at least knew what direction to proceed next.

[1] It's not quite that way. The system does not assign a physical page or a page file page to you when you commit.[2] The system merely does enough bookkeeping to ensure that if you write data to the pages that you committed, then the system will have a place to hold this data and produce it upon demand. That place might be physical memory, or it might be the page file, but the system doesn't know at commit time which it will be (and it may end up being both at different times).

[2] This delay-assignment of committed pages means that reserving address space and then committing it as needed doesn’t save you any physical memory. The commit doesn’t require physical memory. The physical memory doesn’t get assigned until you try to access the memory. All you’re saving is system commit, which is just a number (although as noted in the previous footnote, the number does have a maximum value). Unless you are doing this to save large amounts of memory (dozens of megabytes or more), committing on demand is usually not worth the effort.

Raymond Chen

Follow