Spurious wake-ups in Win32 condition variables

devblogs.microsoft.com/oldnewthing/20180201-00

February 1, 2018

Raymond Chen

Win32 condition variables are subject to spurious wake-ups, where the sleeping thread wakes up but finds that the condition it is waiting for is not satisfied, so it has to go back to sleep. It woke up for no apparent reason.

One source of spurious wake-ups is the stolen wake-up, where a sleeping thread is woken, but by the time it gets a chance to run, another thread has already snuck in and taken the thing that the thread was waiting for, forcing the thread to go back and wait some more.

Another source of spurious wake-ups is where there are a lot of threads waiting on the condition variable, and then there is a huge flurry of `WakeConditionVariable` calls. Normally, exactly one thread is woken for each call to `WakeConditionVariable`, but if there are a lot of wakes in rapid succession, the internal data structure doesn't have enough room in the "number of threads that need to be woken" bitfield to record the exact number, and the system says, "Well, I'll play it safe and just wake up everybody."

The "number of threads that need to be woken" is a bitfield rather than a full 32-bit value because all of the bookkeeping for a condition variable must fit inside a pointer-sized variable, so you have to be very frugal with how you use that limited space. And since condition variables explicitly permit spurious wake-ups, it's okay to be sloppy in keeping track of how many wake-ups are required, as long as you always err on the side of waking up too many people.

Raymond Chen

Follow