During process termination, the gates are now electrified

devblogs.microsoft.com/oldnewthing/20100122-00

January 22, 2010

Raymond Chen

It turns out that my quick overview of how processes exit on Windows XP was already out of date when I wrote it. Mind you, the information is still accurate for Windows XP (as far as I know), but the rules changed in Windows Vista.

> What about critical sections? There is no "Uh-oh" return value for critical sections; `EnterCriticalSection` doesn't have a return value. ~~Instead, the kernel just says "Open season on critical sections!" I get the mental image of all the gates in a parking garage just opening up and letting anybody in and out.~~

In Windows Vista, the gates don't go up. Instead they become electrified! If during `DLL_PROCESS_DETACH` at process termination on Windows Vista you call `EnterCriticalSection` on a critical section that has been orphaned, the kernel no longer responds by just letting you through. Instead, it says, "Oh dear, things are in unrecoverably bad shape. Best to just terminate the process now." If you try to enter an orphaned critical section during process shutdown, the kernel simply calls `TerminateProcess` on the current process! It's sort of like the movie *Speed*: If the thread encounters a critical section that causes it to drop below 50 miles per hour, it blows up. Fortunately, this error doesn't change the underlying analysis of How my lack of understanding of how processes exit on Windows XP forced a security patch to be recalled.

But it also illustrates how the details of process shutdown are open to changes in the implementation at any time, so you shouldn't rely on them. Remember the classical model for how processes exit: You cleanly shut down all your worker threads, and then call `ExitProcess`. If you don't follow that model (and given the current programming landscape, you pretty have no choice but to abandon that model, what with DLLs creating worker threads behind your back), it's even more important that you follow the general guidance of not doing anything scary in your `DllMain` function.

Raymond Chen

Follow