Choosing a provocative debug signature

devblogs.microsoft.com/oldnewthing/20070604-00

June 4, 2007

Raymond Chen

Back in Windows 95, there was an elusive heap corruption bug in the graphics engine, and after a lot of analysis, the graphics folks were convinced that the corruption was coming from outside their component, and they had a pretty good idea who the corruptor was, but they needed proof. One of the standard techniques of narrowing down the source of a problem like this is to put a signature value in the object and validating the signature on entry to every function that uses that object as well as on exit. If you find that the signature was valid on entry but is corrupted on exit, then your function corrupted it. Conversely, if it was valid on exit but is invalid on a subsequent entry, then somebody else corrupted it. At least that's the theory. The developer who was responsible for investigating the bug decided to use this "signature value" technique. It is often the case that, for throwaway temporary signatures like this, you will use your own initials as the signature value. This is partly egotism but mostly just lack of creativity. But this particular developer had a better idea. Since he had a pretty good idea which component was corrupting the memory, he used not his own initials, but the initials of the developer responsible for the component he thought was the corruptor! That way, when that developer's component corrupted the signature, it'd just be corrupting his own initials. Of course, when the developer of the suspect component saw this check-in, he felt kind of insulted. After all, his friend just accused him of corrupting memory.

(Epilogue: It turns out that the graphics folks were right. It *was* that other component that was corrupting the memory.)

Raymond Chen

Follow