# After reporting a non-responsive program to Windows Error Reporting, why does the process spawn a suspended child process?

devblogs.microsoft.com/oldnewthing/20181018-00

October 18, 2018

Raymond Chen

A customer observed that when they try to close a program as not responding, Windows Error Reporting kicks in, which is not unexpected. But what is unexpected is that a new process is created that is a child of the original process (as reported by Process Explorer), and the child is suspended. "Why does werfault.exe create this child process?"

This suspended child process is a snapshot of the original. Windows Error Reporting creates this snapshot and uses the snapshot to generate the error report. The original process is allowed to continue executing so that it can exit (and possibly restart) normally.

The snapshot process does not have any running threads, but it has a copy of the original process's virtual memory, handles, thread IDs, stacks, and other information necessary to create an error report. Generating an error report take time, and Windows Error Reporting uses a snapshot so that the original process can get on with exiting.

**Bonus chatter**: This new behavior means that you don't have to wait for Windows Error Reporting to do its thing before it restarts the application. The "process seeing its own dead body" problem is mitigated by making sure that the snapshot doesn't own any resources. When programs look for already-executing copies of themselves, it's usually done by looking for windows or named kernel objects. Sometimes it's done by recording the process ID of the first instance somewhere, and having the second copy look it up. But the snapshot process owns no windows or kernel objects, and its process ID is not the one that got recorded. so it is comparatively unlikely to be mistaken for the real thing.

Raymond Chen

**Follow**