Why don’t I get a broken pipe when the child process terminates?

devblogs.microsoft.com/oldnewthing/20161207-00

December 7, 2016

Raymond Chen

A customer was having a problem with named pipes.

> Our main process creates a named pipe with `ACCESS_INBOUND` and passes the write handle to the child process. The main process keeps reading from the pipe until it gets `ERROR_PIPE_BROKEN`. We expect that when the child process terminates, the main process will get the `ERROR_PIPE_BROKEN` error. However, we are finding that sometimes the main proecss doesn’t get the `ERROR_PIPE_BROKEN` error, even though the child process has definitely terminated. Are there cases where the process with the write end of the pipe terminates, but the read doesn’t error out with `ERROR_PIPE_BROKEN`?

You won’t get `ERROR_PIPE_BROKEN` until all the write handles are closed. One common reason why you don’t get the error is that there’s still a write handle open in the parent process. Another possibility is that the child process launched a subprocess which inherited the write handle, or more generally, the handle got duplicated into another process by some means.

The customer wrote back.

> Thanks. That is indeed the issue. The main process spawns many child processes simultaneously, so depending on race conditions, the write handle for one pipe could inadvertently be inherited by an unrelated child process. We could explicitly serialize our `CreateProcess` calls, but is there another way to specify that a child process should inherit only certain handles and not others?

Yes. You can use the `PROC_THREAD_ATTRIBUTE_LIST` structure to exercise finer control over which handles are inherited.

Raymond Chen

Follow