How long do I have to keep the SECURITY_ATTRIBUTES and SECURITY_DESCRIPTOR structures valid after using them to create a file?

devblogs.microsoft.com/oldnewthing/20160520-00

May 20, 2016

Raymond Chen

A customer passed a `SECURITY_ATTRIBUTES` when creating a file and wanted to know how long that structure (and the structures it points to) need to remain valid. "Do I have to keep them around as long as the handle that references the `SECURITY_ATTRIBUTES` remains open?"

No. Once `CreateFile` returns, the `SECURITY_ATTRIBUTES` you passed in is no longer used. Everything the system needs was copied to a safe place as part of the `CreateFile` operation. You are welcome to free the memory or to modify and reuse it in a future call to `CreateFile`.

If you think about it, this is really the only way it could be. The file handle could be duplicated into another process, and then the original process might terminate. At that point, the memory for the `SECURITY_ATTRIBUTES` no longer exists, but there's still an open handle.

And there doesn't seem to be any benefit in relying on the copy of the security descriptor in the application's memory, because even if the kernel did that, the information would need to be copied from the application's memory into kernel space when the handle is closed. You may as well just copy it now. (And if the process crashes, the address space is destroyed, so the last known copy of the security descriptor is lost!)

File handles are kernel objects, and the kernel as a rule does not retain references to memory in user-mode. (With the obvious exception of output buffers, such as the buffer passed to `ReadFile` or the `OVERLAPPED` structure passed to overlapped I/O.)

The customer confirmed that they didn't think it was necessary to keep the `SECURITY_ATTRIBUTES` valid, but they aer trying to figure out a bug and that was one of the crazy theories they had for why the problem might be occurring. They thanked us for confirming their understanding and went back to debugging their program, but now with one possible root cause ruled out.

Raymond Chen

Follow