Isn't there a race condition in GetFileVersionInfoSize?

devblogs.microsoft.com/oldnewthing/20120321-00

March 21, 2012

Raymond Chen

In response to my explanation of what the `lpdwHandle` parameter in `GetFileVersionInfo-Size` is used for, Steve Nuchia wonders if there's a race condition between the time you get the size and the time you ask for the data. Yes, there is a race condition, but calling the function in a loop won't help because the `GetFileVersionInfo` function does not report that the buffer is too small to hold all the version data. It just fills the buffer as much as it can and truncates the rest. In practice, this is not a problem because you are usually getting the versions of files that you expect to be stable. For example, you might be obtaining the version resources of the files your application is using in order to show them in diagnostics. The file can't change because you're preventing them from changing by using them. In the case that the file changes out from under you, then yes, you will sometimes get partial data. While I'm on the subject of `GetFileVersionInfo` , I figured I'd mention that there's a good amount of code in `VerQueryValue` to handle the following scenario:

- On Windows NT 3.1, a program calls `GetFileVersionInfo` to obtain a file version information block.
- The program writes the information block to a file.
- The file is preserved in amber for millions of years.
- A curious scientists discovers the file version information block, loads it from the file back into memory, and calls `VerQueryValue` .

The modern implementation of `VerQueryValue` still understands the file version information block created by all previous versions of Windows, and if you hand it one of those frozen-in-amber information blocks, it still knows how to extract information from it. It may not be able to do as good a job due to the lack of appropriate buffer space, but it does at least as well as the version of Windows the file version information block was originally generated from. I have no idea whether anybody actually takes advantage of this behavior, but since persisting the file version information block was never explicitly disallowed in the documentation, one could argue that doing so was legal, and the code therefore needs to be ready for it. (Heck, even if it were explicitly disallowed, there would still be a good chance that there's somebody who's doing it.)

What `VerQueryValue` doesn’t handle is people who hand it a file version information block that never came from `GetFileVersionInfo` in the first place.

Raymond Chen

Follow