# Does this operation work when file system redirection is disabled? The default answer is NO

devblogs.microsoft.com/oldnewthing/20130321-00

March 21, 2013

Raymond Chen

A customer reported that when their program called `SHGetFileInfo` to get the icon for a folder, the call failed. "It works on some machines but not others. We don't know what the difference is between the working and non-working machines." They included the offending function from their program, but everything in the function looked good. The problem was something outside the function itself.

Eventually, the customer confessed that they had called the `Wow64DisableWow64FsRedirection` function to disable file system redirection, and the call to `SHGetFileInfo` took place while redirection was disabled. "We found that if we re-enable file system redirection before calling `SHGetFileInfo`, then everything works properly."

That's right, because, like impersonation, nothing works when file system redirection is disabled unless it is specifically documented as supporting disabled redirection. This is even called out in the documentation for `Wow64DisableWow64FsRedirection`:

> **Note** The **Wow64DisableWow64FsRedirection** function affects all file operations performed by the current thread, which can have unintended consequences if file system redirection is disabled for any length of time. For example, DLL loading depends on file system redirection, so disabling file system redirection will cause DLL loading to fail. Also, many feature implementations use delayed loading and will fail while redirection is disabled. The failure state of the initial delay-load operation is persisted, so any subsequent use of the delay-load function will fail even after file system redirection is re-enabled. To avoid these problems, disable file system redirection immediately before calls to specific file I/O functions (such as **CreateFile**) that must not be redirected, and re-enable file system redirection immediately afterward using **Wow64RevertWow64FsRedirection**.

Whenever you use one of these "global solutions to a local problem" types of solutions that change some fundamental behavior of the system, you have to make sure that everybody is on board with your decision.

The local solution would be to use the `C:\Windows\SysNative` virtual directory for files you want to look up in the native system directory rather than the emulated system directory.

Raymond Chen

**Follow**