A fine detail on how DLLs are relocated as the result of a base address collision, and consequences

devblogs.microsoft.com/oldnewthing/20170119-00

January 19, 2017

Raymond Chen

If a DLL must be relocated due to a base address conflict, then the image will be relocated, and the entire relocated DLL is now backed by the page file.

If you read the description more carefully, you'll see that it's not exactly the entire relocated DLL that gets backed by the page file. More precisely, all the pages that contained fixups are put into the page file. If you're lucky and have a page without any fixups, then that page will still be demand-paged from the image because the kernel didn't apply any fixups to it, and therefore did not incur a copy-on-write for that page, so it continues to be backed by the file system image.

One of the arguments I've seen for intentionally causing a base address collision is so that the relocated DLL gets copied into the page file, which is a win if the page file is on a faster medium than the DLL. For example, the page file may be on an SSD or (gasp) a RAM drive.

That logic fails to take into account the case of pages with no fixups. Those pages will still page in directly from the original file, which be a problem if the original file is on a very slow medium, or a medium which could be lost, such as a CD-ROM drive or network drive.

Fortunately, you don't need to play funny games with base address conflicts to get your entire DLL loaded into the page file. Instead, use the `/SWAPRUN` linker flag which lets you specify in the module header that the loader should copy the image into the swap file.

Raymond Chen

Follow