Why do I have to pass a valid page protection value to VirtualAlloc even if it ignores it?

devblogs.microsoft.com/oldnewthing/20171227-00

December 27, 2017

Raymond Chen

The `VirtualAlloc` function accepts a `flProtect` value, and even though the value is not used by `MEM_RESET` or `MEM_RESET_UNDO`, you still have to pass a valid value. (The documentation suggests `PAGE_NOACCESS`.)

Why do you have to pass a valid value even if the system doesn't use it?

This is an artifact of how the front-end parameter validation is done. The `VirtualAlloc` function does parameter validation by checking each parameter individually to confirm that the value is among the valid values.

For `flAllocationType` the code makes sure that a valid combination of flags is passed.

For `flProtect`, the code makes sure that the page protection value is one of the valid values.

This validation is not contextual, however. When the `VirtualAlloc` function validates the `flProtect` parameter, it doesn't take the `flAllocationType` into account.

After the front-end validation is done, the code starts breaking down the cases and performs additional validation as necessary.

But what you don't see is a parameter validation short-circuit.

So by the time the code realizes that it's in a case where the `flProtect` value is not used, it has already validated it.

So make sure your parameters are valid, even if you're calling the function in a way where the parameter is not used.

Bonus chatter: You would think that the `flProtect` would not be used when reserving address space with `MEM_RESERVE`, but you'd be wrong. If reserving regular address space, then the protection should be `PAGE_NOACCESS`. This first rule isn't surprising. If you are

reserving the space but haven't allocated anything yet, then naturally you don't have access to anything there, since there's nothing there to access.

The odd bit is that if you are reserving address space for Address Windowing Extensions (AWE), then your allocation type is `MEM_ RESERVE | MEM_ PHYSICAL`, and the protection must be `PAGE_ READWRITE`. Yes, there's no memory there yet, but you still have read/write access to it. (I don't know what that means either, but that's the rule.)

Raymond Chen

Follow