What's the point of SecureZeroMemory?

devblogs.microsoft.com/oldnewthing/20130529-00

May 29, 2013

Raymond Chen

The `SecureZeroMemory` function zeroes out memory in a way that the compiler will not optimize out. But what's the point of doing that? Does it really make the application more secure? I mean, sure the data could go into the swap file or hibernation file, but you need to have Administrator access to access those files anyway, and you can't protect yourself against a rogue Administrator. And if the memory got swapped out before it got zeroed, then the values went into the swap file anyway. Others say that it's to prevent other applications from reading my process memory, but they could always have read the memory before I called `SecureZeroMemory`. So what's the point?

The `SecureZeroMemory` function doesn't make things secure; it just makes them *more* secure. The issue is a matter of degree, not absolutes.

if you had a rogue Administrator or another application that is probing your memory, then that rogue operator has to suck out the data during the window of opportunity between the time you generate the sensitive data and the time you zero it out. This is typically not a very long time, so it makes the attacker work harder to get the data. Similarly, the data has to be swapped out during the window between the sensitive data being generated and the data being zeroed. Whereas if you never called `SecureZeroMemory`, the attacker could take their sweet time looking for the sensitive information, because it'll just hang around until the memory gets re-used for something else.

Furthermore, the disclosure may not be due to a rogue operative, but may be due to your own program! If your program crashes, and you're signed up your program for Windows Error Reporting, then a crash dump file is generated and uploaded to Microsoft so that you can download and investigate why your program is failing. In preparation for uploading, the crash dump is saved to a file on the user's hard drive, and an attacker may be able to mine that crash dump for sensitive information. Zeroing out memory which contained sensitive information reduces the likelihood that the information will end up captured in a crash dump.

Another place your program may inadvertently reveal sensitive information is in the use of uninitialized buffers. If you have a bug where you do not fully-initialize your buffers, then sensitive information may end up leaking into them and then accidentally transmitted over the network or written to disk. Using the `SecureZeroMemory` function when finished with sensitive information is a defense-in-depth way of making it harder for sensitive information to go where it's not supposed to.

Raymond Chen

Follow