# How am I supposed to free the information passed to the SetSecurityInfo function?

devblogs.microsoft.com/oldnewthing/20170201-00

February 1, 2017

Raymond Chen

Some time ago, we discussed how you're supposed to free the information returned by the `GetSecurityInfo` function. But what about the information passed to the `SetSecurityInfo` function? "How do I free that?"

You free that memory by whatever means you like. You allocated the memory originally, so you get to free it. If you allocated the memory with `malloc`, then use `free`. If you allocated the memory with `new`, then use `delete`. Whatever mechanism you used to allocate the memory, use the corresponding mechanism for freeing it.

"Do I have to free the old DACL being replaced?"

No, that is managed by the system. What you're doing is saying, "Dear operating system: Here is a kernel object and some security information. Please set the security on the kernel object to match the information I'm giving you. Thanks."

"So you're saying that if I have code that does this:

- `GetSecurityInfo(…, &oldAcl, …)`
- Create `newAcl` by copying the `oldAcl` and making appropriate changes.
- `SetSecurityInfo(…, newAcl, …)`

then I need to free the `newAcl` but not the `oldAcl`?"

No, that's not what I'm saying. I'm saying that the call to `SetSecurityInfo` does not create any new obligations to free memory. However, it also does not destroy any existing obligations to free memory.

Calling `GetSecurityInfo` created an obligation to free `oldAcl`. That obligation was not changed by the call to `SetSecurityInfo`.

What I mean by saying that you don't have to free the old DACL being replaced is that when you call `SetSecurityInfo`, the system frees its internal security info and replaces it with a copy of the security info you passed in. You don't need to worry about freeing that internal info. (Not that you could, because you don't know how it was allocated.) But of course, if you made a copy of the internal security info, then you are on the hook for freeing the copy.

Let's look at it this way:

- There is some secret security info out there managed by space aliens from the planet Krypton. You do not have direct access to it. The only way to access it is by calling functions like `GetSecurityInfo` and `SetSecurityInfo`.
- The `GetSecurityInfo` function says, "Dear space aliens: Please take your security info, translate them from Kryptonese to Win32, and give me the translation. I will free the translation when I'm done."
- The `SetSecurityInfo` function says, "Dear space aliens: Here is some security info in Win32 format. Please translate it to Kryptonese and use that as the new security info."

You don't speak Kryptonese, but that's okay, because your only interaction with the security info is through Win32 format. If you ask for a copy of the internal security info, then you are responsible for freeing that copy. But the internal Kryptonese security info is not something you need to worry about. The space aliens will take care of that.

Raymond Chen

**Follow**