How do the names in the file security dialog map to access control masks?

devblogs.microsoft.com/oldnewthing/20070726-00

July 26, 2007

Raymond Chen

When you call up the file security dialog, you'll see options like "Full Control" and "Read and Execute". That's really nice as friendly names go, but when you're digging into the security descriptor, you may need to know what those permissions really map to when it comes down to bits. First, the summary attributes:

Friendly name	Access mask	Inheritance
Full control	`FILE_ALL_ACCESS`	`CONTAINER_INHERIT_ACE` + `OBJECT_INHERIT_ACE`
Modify	`FILE_GENERIC_READ` | `FILE_GENERIC_WRITE` | `FILE_GENERIC_EXECUTE` | `DELETE`	`CONTAINER_INHERIT_ACE` + `OBJECT_INHERIT_ACE`
Read and execute	`FILE_GENERIC_READ` | `FILE_GENERIC_EXECUTE`	`CONTAINER_INHERIT_ACE` + `OBJECT_INHERIT_ACE`
List folder contents	`FILE_GENERIC_READ` | `FILE_GENERIC_EXECUTE`	`CONTAINER_INHERIT_ACE`
Read	`FILE_GENERIC_READ`	`CONTAINER_INHERIT_ACE` + `OBJECT_INHERIT_ACE`
Write	`FILE_GENERIC_WRITE & ~READ_CONTROL`	`CONTAINER_INHERIT_ACE` + `OBJECT_INHERIT_ACE`

If you go to the Advanced view, then you get much more precise control:

Friendly name	Access mask
Traverse Folder / Execute File	`FILE_TRAVERSE == FILE_EXECUTE`

List Folder / Read Data	`FILE_LIST_DIRECTORY == FILE_READ_DATA`
Read Attributes	`FILE_READ_ATTRIBUTES`
Read Extended Attriibutes	`FILE_READ_EA`
Create Files / Write Data	`FILE_ADD_FILE == FILE_WRITE_DATA`
Create Folders / Append Data	`FILE_ADD_SUBDIRECTORY == FILE_APPEND_DATA`
Write Attributes	`FILE_WRITE_ATTRIBUTES`
Write Extended Attributes	`FILE_WRITE_EA`
Delete Subfolders and Files	`FILE_DELETE_CHILD`
Delete	`FILE_DELETE`
Read Permissions	`READ_CONTROL`
Change Permissions	`WRITE_DAC`
Take Ownership	`WRITE_OWNER`

(In the Advanced view, you control inheritance from the "Apply to" drop-down combo box.) Note that the "Delete Subfolders and Files" and "Delete" attributes together determine whether you can delete a file or subdirectory: You can delete an item either if you have `DELETE` permission on the item **or** if you have `DELETE_CHILD` permission on its parent. This "combo" allows you to set up a directory where everybody can create files and can delete files that they have created, while still retaining the ability as the directory's owner to delete any file in it. You do this by granting yourself `DELETE_CHILD` permission on the directory and granting `DELETE` to `CREATOR_OWNER` as an inheritable attribute. Since you have `DELETE_CHILD` permission, you can delete anything in the directory. And since the creator/owner has `DELETE` permission, people can delete the files that they themselves created.

[Update 2pm: INHERIT_ONLY_ACE should be OBJECT_INHERIT_ACE.]

Raymond Chen

Follow