Be careful when interpreting security descriptors across machine boundaries

devblogs.microsoft.com/oldnewthing/20060202-00

February 2, 2006

Raymond Chen

While it's true the `AccessCheck` function can be used to check whether a particular security descriptor grants access to a token, you need to be aware of where that security descriptor came from. If the security descriptor came from another machine (for example, if you got it by calling `GetNamedSecurityInfo` and passing the path to a file on a network share), calling the `AccessCheck` function on your machine may give different results from the remote machine. In other words, it is possible for the `AccessCheck` function to indicate that you have access, when in fact you don't. How can that be? For one thing, there are many SIDs that are machine-relative. If the remote object grants access to, say, the Builtin Administrators group, running the `AccessCheck` function checks the token against the Builtin Administrators group of the machine that is running the check. If you are a member of the Builtin Administrators group of your own machine but aren't a member of the Builtin Administrators group of the remote machine, you will think you have access when you don't. In addition to the machine-relative SID problem, there's also the problem that tokens can lose their identity as they travel across the network. If the server has the ForceGuest policy enabled, then it doesn't matter what your token is on your machine. On the remote machine, you are treated as Guest.

The moral of the story is that trying to determine whether you have access to an object without actually accessing it is harder than it looks. You're usually much better off just trying to access it. No point trying to emulate what another computer is going to do if you can just have it do it!

Raymond Chen

Follow