If you cancel an operation while it's in progress, then it's not surprising that it's only half-done

devblogs.microsoft.com/oldnewthing/20140218-00

February 18, 2014

Raymond Chen

A customer (via their customer liaison) started by asking why they were seeing an unexpected access control entry in the security descriptor of an object.

> The ACEs on the parent grant access to *Administrators*, *CREATOR OWNER*, *SYSTEM*, and *Users*, but the ACEs on the child object (which should simply have been inherited from the parent) include an extra entry for *Bob*. How did Bob get access to the child object? When we view the details of the ACEs, it lists the Bob entry as *Inherited from parent*. But there is no Bob entry in the parent!

I observed, "Probably because Bob is the *CREATOR OWNER*."

> Thanks for the explanation, but even if Bob is the *CREATOR OWNER*, how can we explain that the permission is inherited from the parent?

The permission is inherited from the parent because the parent has specified the rights of the *CREATOR OWNER*, and Bob is the creator/owner. As part of the inheritance process, the rights of the *CREATOR OWNER* get assigned to Bob. Remember that *CREATOR OWNER* is not a real person. It is a placeholder that gets replaced with the actual creator/owner when the object is created. If Bob created the child object, then the permissions of *CREATOR OWNER* will be given to Bob on the child object. The *CREATOR OWNER* is not a live entry that dynamically updates to match the current creator/owner. It is a static entry that is assigned at the point of creation. Changes to the owner in the future have no effect because the identity has already been snapshotted. (I think a less confusing name would have been simply *OBJECT CREATOR*, since creation happens only once.) (Note that there is a little extra weirdness here: If the creator is a member of the Administrators group, then the *CREATOR OWNER* rights get assigned to the entire Administrators group instead of the specific user who created it. You can change this behavior by tweaking the *Default owner for objects created by members of the Administrators group* policy.) The customer liaison conferred with the customer, and determined that, at least in one of the cases they were studying, Bob was not the original creator. What actually happened was that at some point, Bob was granted access to the parent object and all its sub-objects. Later, somebody went back to the parent object and told it to revoke Bob's access to the parent object and all its sub-objects. But "If we cancel the process fast enough, then we get the strange behavior as originally described." Well, duh! You asked for Bob's access to the parent object and all its sub-objects to be revoked, so the tool you used started a recursive tree walk from the parent object looking for any objects that Bob has access to and removing them. But if you cancel the operation partway through, then that tool didn't get a chance to finish the job, and you obviously are left in a situation where Bob's access was only partially revoked. The customer liaison confirmed,

> Yes, that's what happened.

It's nice of the customer liaison to confirm the diagnosis, but it still baffles me that they were confused by this in the first place.

To me this is one of those *So what did you expect* type of situations. You start an operation, then partway through, you cancel it, and then you're surprised that the operation did not run to completion.

Raymond Chen

Follow