Beware of roaming user profiles

devblogs.microsoft.com/oldnewthing/20050630-20

June 30, 2005

Raymond Chen

One of the less-known features of Windows is the roaming user profile. I know that this is not well-known because I often see suggestions that fail to take the roaming user profile scenario into account. Indeed, if your program behaves badly enough, you can cause data loss. (More on this later.) What is a roaming user profile? Well, your user profile is the collection of things that reside under your `%USERPROFILE%` directory. (This is not quite true, but it's a good enough approximation for the purpose of this discussion. An important exception will be noted next time.) Your per-user registry is kept in `%USERPROFILE%\ntuser.dat`, so your per-user registry is part of your user profile. In highly managed environments (corporations), system administrators can set up user profiles on a centralized server, so that users log onto any machine and have available their files and settings. This is accomplished by copying the user profile from the server when the user logs on and copying it back to the server when the user logs off. (Of course, there is also caching involved to save time if the user logs back onto the same machine.) What does this mean for you, the programmer? For one thing, it means that the path to the user's profile can change from one logon session to the next. If the user runs your program from Computer A, their user profile directory might be `C:\Documents and Settings\Fred`, but when they log off from Computer A and log on to Computer B, the directory to their user profile might change to `C:\WINNT\Profiles\Fred`. In particular, that file that used to be at `C:\Documents and Settings\Fred\My Documents\Proposal.txt` has moved to `C:\WINNT\Profiles\Fred\My Documents\Proposal.txt`. If your program has a feature where it offers a list of recently-used files (or auto-opens the most recently used file), you may find that the file no longer exists at its old location. The solution is to use profile-relative paths, or even better, shell virtual folder-relative paths (e.g., recording the path relative to CSIDL_MYDOCUMENTS), so that when the profile roams to a machine with a different user profile path, your program can still find its files. For another thing, you cannot just cruise through the `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\ProfileList` registry key expecting to find all the user profiles and possibly even modify them, because the copy of the user profile on the local computer might not be the authoritative one. If the profile is a cached roaming profile, then any changes you make will either (1) be lost when the user roams back to the computer after using another computer, or (2) cause the local profile to be considered newer than the master copy on the server, causing the changes the user made to the copy on the server to be lost! (Which of the two bad scenarios you find yourself in depends on the time you change the cached profile and the time the target user logs off that other computer.) Another consequence of roaming user profiles is that your program can effectively see itself changing versions constantly. If Computer A has version 1.0 of your program and Computer B has version 2.0, then as the profile roams between the two computers, both versions 1.0 and 2.0 will be operating on the user profile in turn. If versions 1.0 and 2.0 use the same registry keys to record their settings, then your registry formats had better be both upward- and downward-compatible. This is a particularly painful requirement for operating system components, which consequently need to maintain bidirectional registry format compatibility with systems as old as Windows NT 4. (Windows NT 3.51 had a different model for roaming user profiles.)

Yet another consequence of roaming user profiles applies to services. Prior to Windows XP, if a service holds a registry key open after the user logged off, then the registry hive cannot be unloaded and consequently (1) consumes memory for that profile even though the user is no longer logged on, and (2) prevents the user's local registry changes from being copied back to the server. This "hive leakage" problem was so rampant that in Windows XP, the profile unload code takes a more aggressive stance against services that hold keys open too long. You can read more about the changes to registry hive roaming in the Resource Kit article linked at the top of this entry.

Raymond Chen

Follow