Is it wrong to call SHFileOperation from a service? Revised

devblogs.microsoft.com/oldnewthing/20141121-00

November 21, 2014

Raymond Chen

My initial reaction to this question was to say, "I don't know if I'd call it wrong, but I'd call it highly inadvisable." I'd like to revise my guidance. It's flat-out wrong, at least in the case where you call it while impersonating. The registry key `HKEY_CURRENT_USER` is bound to the current user at the time the key is first accessed by a process:

> The mapping between **HKEY_CURRENT_USER** and **HKEY_USERS** is per process and is established the first time the process references **HKEY_CURRENT_USER**. The mapping is based on the security context of the first thread to reference **HKEY_CURRENT_USER**. If this security context does not have a registry hive loaded in **HKEY_USERS**, the mapping is established with **HKEY_USERS\.Default**. After this mapping is established it persists, even if the security context of the thread changes.

Emphasis mine. This means that if you impersonate a user, and then access `HKEY_CURRENT_USER`, then that binds `HKEY_CURRENT_USER` to the impersonated user. Even if you stop impersonating, future references to `HKEY_CURRENT_USER` will still refer to that user. This is probably not what you expected. The shell takes a lot of settings from the current user. If you impersonate a user and then call into the shell, your service is now using that user's settings, which is effectively an elevation of privilege: An unprivileged user is now modifying settings for a service. For example, if the user has customized the Print verb for text files, and you use `ShellExecute` to invoke the `print` verb on a text document, you are at the mercy of whatever the user's `print` verb is bound to. Maybe it runs Notepad, but maybe it runs pwnzord.exe. You don't know. Similarly, the user might have a per-user registered copy hook or namespace extension, and now you just loaded a user-controlled COM object into your service. In both cases, this is known to insiders as *hitting the jackpot*. Okay, so what about if you call `ShellExecute` or some other shell function while not impersonating? You might say, "That's okay, because the current user's registry is the service user, not the untrusted attacker user." But look at that sentence I highlighted up there. Once `HKEY_CURRENT_USER` get bound to a particular user, it remains bound to that user *even after impersonation ends*. If somebody else inadvisedly called a shell function while impersonating, and that shell function happens to be the first one to access

`HKEY_CURRENT_USER`, then your call to a shell function while not impersonating will still use that impersonated user's registry. Congratulations, you are now running untrusted code, and you're not even impersonating any more!

So my recommendation is *don't do it*. Don't call shell functions while impersonating unless the function is explicitly documented as supporting impersonation. (The only ones I'm aware of that fall into this category are functions like `SHGetFolderPath` which accept an explicit token handle.) Otherwise, you may have created (or in the case of copy hooks, definitely created) a code injection security vulnerability in your service.

Raymond Chen

Follow