On the failed unrealized promise of RegOverridePredef-Key

devblogs.microsoft.com/oldnewthing/20231020-00

October 20, 2023

Raymond Chen

If you go browsing through the Win32 API (a common pastime back in the old days), you may run into the `RegOverridePredefKey` function, which lets a process redirect one of the predefined keys to a separate registry key. What's the idea behind this function?

It's explained in the Remarks in the documentation. The idea is that you have a self-registering DLL, and you want to capture the registry changes made by that DLL's `DllRegisterServer` function, so that you can take the captured registry changes and add them to your product's main installer. This allows you to simplify your installer to just "Copy these files to these locations, and then set these registry keys to these values." You took the `DllRegisterServer` step out of the equation, which speeds up installation and also simplifies auditing.

That was the idea, but it pretty much never worked in practice.

The trick assumed that all the DLL's `DllRegisterServer` did was set some registry keys in well-known places (specifically, under `HKEY_CLASSES_ROOT`) to hard-coded values. That may have been true in simpler times, but DLLs quickly took advantage of the fact that the `DllRegisterServer` function was *code*, so you can add whatever other logic to the function you like. You can look around the system and decide to write different keys depending on what you find. You don't even have to limit yourself to writing registry keys! You can create files, call networking APIs, reconfigure the system to your heart's content.

The dream of being able to capture the output of a function call into a list of registry keys was too naïve. Real life is more complicated than that.