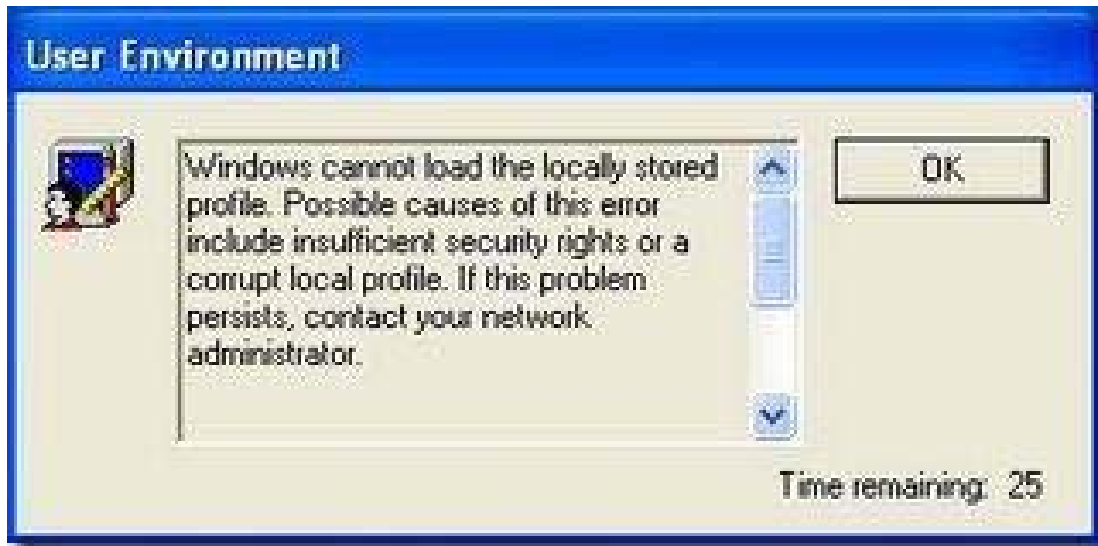


# The Case of the Temporary Registry Profiles

Mark Russinovich  
(From Mark Russinovich Blog)

Microsoft Customer Support Services (CSS) is one of the biggest customers of the Sysinternals tools and they often send me interesting cases they've solved with them. This particular case is especially interesting because it affected a large number of users and the troubleshooting process made use of one of Process Monitor's lesser-known features. The case opened when a customer contacted Microsoft support reporting that several of their users would occasionally get this error message when logging on to their systems:



This caused Windows to create a temporary profile for the user's logon session. A user profile consists of a directory, %UserProfile%, into which applications save user-specific configuration and data files, as well as a registry hive file stored in that directory, %UserProfile%\Ntuser.dat, that the Winlogon process loads when the user logs in. Applications store user settings in the registry hive by calling registry functions that refer to the HKEY\_CURRENT\_USER (HKCU) root key. The user's loss of access to their profile made the problem critical, because whenever that happened, the user would apparently lose all their settings and access to files stored in their profile directory. In most cases, users contacted the company's support desk, which would ask the user to try rebooting and logging in until the problem resolved itself.

As with all cases, Microsoft support began by asking about the system configuration, inventory of installed software, and about any recent changes the company had made to their systems. In this case, the fact that stood out was that all the systems on which the problem had occurred had recently been upgraded to a new version of Citrix Corporation's ICA client, a remote desktop application. Microsoft contacted Citrix support to see if they knew of any issues with the new client. They didn't, but said they would investigate.

Unsure whether the ICA client upgrade was responsible for the profile problem, Microsoft support instructed the customer to enable profile logging, which you can do by configuring a registry key as per this Knowledge Base article: How to enable user environment debug logging in retail builds of Windows. The customer pushed a script out to their systems to make the required registry changes and shortly after got another call from a user with the profile problem. They grabbed a copy of the profile log off the system from %SystemRoot%\Debug\UserMode\Userenv.log and sent it into Microsoft. The log was inconclusive, but did provide an important clue: it indicated that the user's profile had failed to load because of error 32, which is ERROR\_SHARING\_VIOLATION:

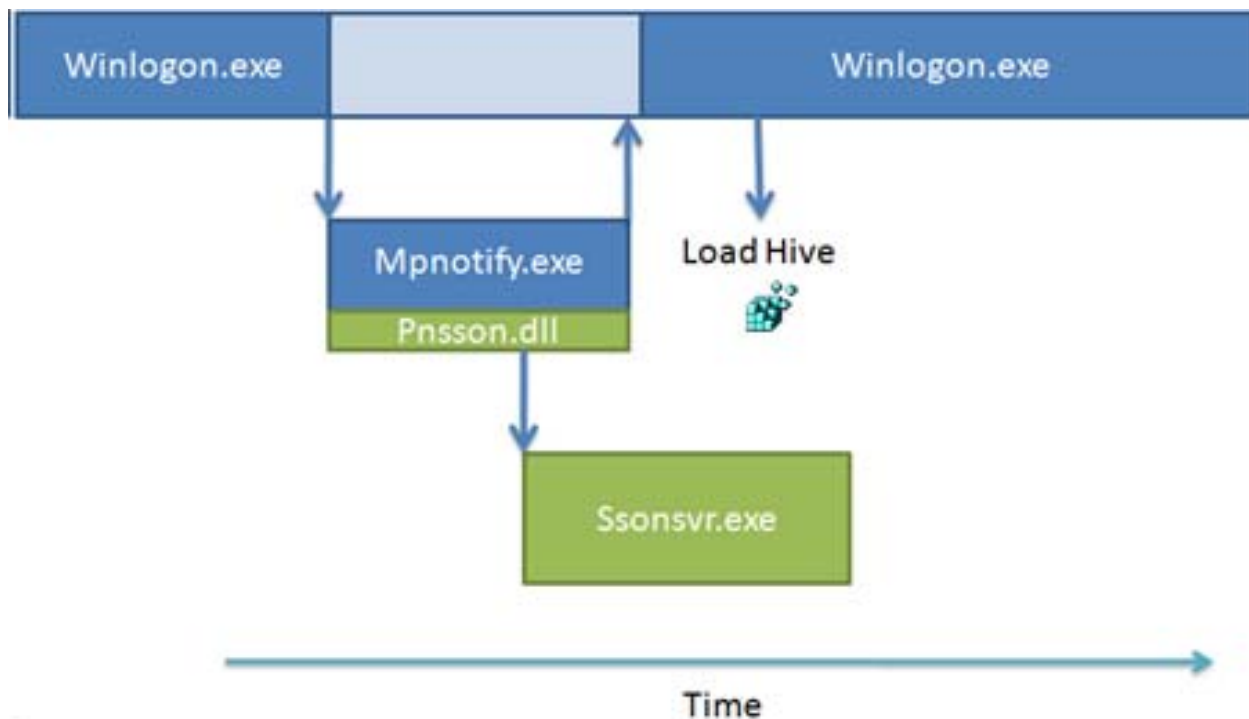
# The Case of the Temporary Registry Profiles

Mark Russinovich  
(From Mark Russinovich Blog)

```
USERENV(2dc.abc) 16:23:14:599 GetGPOInfo: Local GPO's gpt.in1 is not  
USERENV(2dc.c14) 16:23:14:678 PolicyChangedThread: updateuser failed w  
USERENV(2dc.2e0) 16:33:04:565 MyRegLoadKey: Failed to load subkey  
<S-1-5-21-1292428093-343818398-839522115-49106>, error =32  
USERENV(2dc.2e0) 16:33:04:565 ReportError: Impersonating user.
```

When a process opens a file, it specifies what kinds of sharing it allows for the file. If it is writing to the file it may allow other processes to read from the file, for example, but not to also write to the file. The sharing violation in the log file meant that another process had opened the user's registry hive in a way that was incompatible with the way that the logon process wanted to open the file.

In the meantime, more customers around the world began contacting Microsoft and Citrix with the same issue, all had also deployed the new ICA client. Citrix support then reported that they suspected that the sharing violation might be caused by one of the ICA client's processes, Ssonvr.exe. During installation, the ICA client registers a Network Provider DLL (Pnsson.dll) that the Windows Multiple Provider Notification Application (%SystemRoot%\System32\Mpnotify.exe) calls when the system boots. Mpnotify.exe is itself launched at logon by the Winlogon process. The Citrix notification DLL launches the Ssonvr.exe process asynchronous to the user's logon:



The only problem with the theory was that Citrix developers insisted that the process did not attempt to load any user registry profile or even read any keys or values from one. Both Microsoft and Citrix were stumped.

Microsoft created a version of Winlogon and the kernel with additional diagnostic information and tried to reproduce the problem on lab systems configured identically to the client's, but without success. The customer couldn't even reproduce the problem with the modified Windows images, presumably because the images changed the timing of the system enough to avoid the problem. At this point a

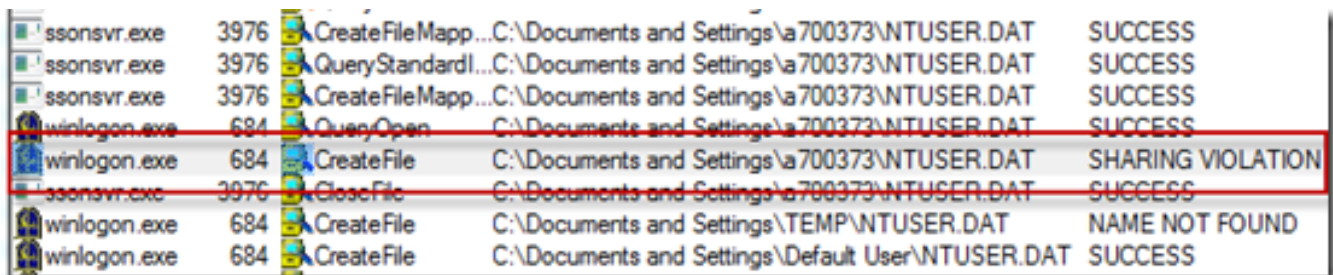
# The Case of the Temporary Registry Profiles

Mark Russinovich  
(From Mark Russinovich Blog)

Microsoft support engineer suggested that the customer capture a trace of logon activity with Process Monitor.

There are a couple of ways to configure Process Monitor to record logon operations: one is to use Sysinternals PsExec to launch it in the session 0 so that it survives the logoff and subsequent logon and another is to use the boot logging feature to capture activity from early in the boot, including the logon. The engineer chose the latter, so he told the customer to run Process Monitor on one of the system's that persistently exhibited the problem, select Enable Boot Logging from the Process Monitor Options menu, and reboot, repeating the steps until the problem reproduced. This procedure configures the Process Monitor driver to load early in the boot process and log activity to %SystemRoot%\Procmon.pmb. Once the user logged encountered the issue, they were to run Process Monitor again, at which point the driver would stop logging and Process Monitor would offer to convert the boot log into a standard Process Monitor log file.

After a couple of attempts the user captured a boot log file that they submitted to Microsoft. Microsoft support engineers scanned through the log and came across the sharing violation error when Winlogon tried to load the user's registry hive:



ssonsvr.exe	3976	CreateFileMap...	C:\Documents and Settings\A700373\NTUSER.DAT	SUCCESS
ssonsvr.exe	3976	QueryStandard...	C:\Documents and Settings\A700373\NTUSER.DAT	SUCCESS
ssonsvr.exe	3976	CreateFileMap...	C:\Documents and Settings\A700373\NTUSER.DAT	SUCCESS
winlogon.exe	684	QueryOpen	C:\Documents and Settings\A700373\NTUSER.DAT	SUCCESS
winlogon.exe	684	CreateFile	C:\Documents and Settings\A700373\NTUSER.DAT	SHARING VIOLATION
ssonsvr.exe	3976	CloseFile	C:\Documents and Settings\A700373\NTUSER.DAT	SUCCESS
winlogon.exe	684	CreateFile	C:\Documents and Settings\TEMP\NTUSER.DAT	NAME NOT FOUND
winlogon.exe	684	CreateFile	C:\Documents and Settings\Default User\NTUSER.DAT	SUCCESS

It was obvious from operations immediately preceding the error that Ssonsvr.exe was the process that had the hive opened. The question was, why was Ssonsvr.exe opening the registry hive? To answer that question the engineers turned to Process Monitor's stack trace functionality. Process Monitor captures a call stack for every operation, which represents the function call nesting responsible for the operation. By looking at a call stack you can often determine an operation's root cause when it might not be obvious just from the process that executed it. For example, the stack shows you if a DLL loaded into the process executed the operation and, if you have symbols configured and the call originates in a Windows image or other image for which you have symbols, it will even show you the names of the responsible functions.

The stack for Ssonsvr.exe's open of the Ntuser.dat file showed that Ssonsvr.exe wasn't actually responsible for the operation, the Windows Logical Prefetcher was:



```
8 ntkrnlpa.exe nt!IopfCallDriver+0x31
9 ntkrnlpa.exe nt!ObpLookupObjectName+0x53c
10 ntkrnlpa.exe nt!ObOpenObjectByName+0xea
11 ntkrnlpa.exe nt!IopCreateFile+0x407
12 ntkrnlpa.exe nt!IoCreateFile+0x8e
13 ntkrnlpa.exe nt!CcPfGetSectionObject+0x91
14 ntkrnlpa.exe nt!CcPfPrefetchSections+0x2b7
15 ntkrnlpa.exe nt!CcPfPrefetchScenario+0x7b
16 ntkrnlpa.exe nt!CcPfBeginAppLaunch+0x158
17 ntkrnlpa.exe nt!PspUserThreadStartup+0xeb
18 ntkrnlpa.exe nt!KiThreadStartup+0x16
```

# The Case of the Temporary Registry Profiles

Mark Russinovich  
(From Mark Russinovich Blog)

Introduced in Windows XP, the Logical Prefetcher is a kernel component that monitors the first ten seconds of a process launch, recording the directories and portions of files accessed by the process during that time to a file it stores in %SystemRoot%\Prefetch. So that multiple executables with the same name but in different directories get their own prefetch file, the Logical Prefetcher gives the file a name that's a concatenation of the executable image name and the hash of the path in which the image is stored e.g. NOTEPAD.EXE-D8414F97.pf. You can actually see the files and directories the Logical Prefetcher saw an application reference the last time it launched by using the Sysinternals Strings utility to scan a prefetch file like this:

```
strings <prefetch file>
```

The next time the application launches, the Logical Prefetcher, executing in the context of the process's first thread, looks for a prefetch file. If one exists, it opens each directory it lists to bring the directory's metadata into memory if not already present. The Logical Prefetcher then maps each file listed in the prefetch file and references the portions accessed the last time the application ran so that they also get brought into memory. The Logical Prefetcher can speed up an application launch because it generates large, sequential I/Os instead of issuing small random accesses to file data as the application would typically do during startup.

The implication of the Logical Prefetcher in the profile problem only raised more questions, however. Why was it prefetching the user's hive file in the context of Ssonsvr.exe when Ssonsvr.exe itself never accesses registry profiles? Microsoft support contacted the Logical Prefetcher's development team for the answer. The developers first noted that the registry on Windows XP is read into memory using cached file I/O operations, which means that the Cache Manager's read-ahead thread will proactively read portions of the hive proactively. Since the read-ahead thread executes in the System process, and the Logical Prefetcher associates System process activity with the currently launching process, that a specific timing sequence of process launches and activity during the boot and log on could cause hive accesses to be seen by the Logical Prefetcher as being part of the Ssonsvr.exe launch. If the order was slightly different the next boot and log on, Winlogon might collide with the Logical Prefetcher, as seen in the captured boot log.

The Logical Prefetcher is supposed to execute transparently to other activity on a system, but its file references can lead to sharing violations like this on Windows XP systems (on server systems the Logical Prefetcher only prefetches boot activity, and it does so synchronously before the boot process proceeds). For that reason, on Windows Vista and Windows 7 systems, the Logical Prefetcher makes use of a file system minifilter driver, Fileinfo (%SystemRoot%\System32\Drivers\Fileinfo.sys), to watch for potential sharing violation collisions and prevent them by stalling a second open operation on a file being accessed by the Logical Prefetcher until the Logical Prefetcher closes the file.

Now that the problem was understood, Microsoft and Citrix brainstormed on workarounds customers could apply while Citrix worked on an update to the ICA Client that would prevent the sharing violation. One workaround was to disable application prefetching and another was to write a logoff script that deletes the Ssonsvr.exe prefetch files. Citrix published the workarounds in this Citrix Knowledge Base article and Microsoft in this Microsoft Knowledge Base article. The update to the ICA Client, which was made available a few days later, changed the network provider DLL to 10 seconds after Ssonsvr.exe launches before returning control to Mpnotify.exe. Because Winlogon waits for Mpnotify to exit before logging on a user, the Logical Prefetcher won't associate Winlogon's accesses of the user's hive with Ssonsvr.exe's startup.

# The Case of the Temporary Registry Profiles

Mark Russinovich

(From Mark Russinovich Blog)

As I said in the introduction, I find this case particularly interesting because it demonstrates a little known Process Monitor feature, boot logging, and the power of stack traces for root cause analysis, two key tools for everyone's troubleshooting arsenal. It also shows how successful troubleshooting sometimes means coming up with a workaround when there's no fix or you must wait until a vendor provides one. Another case successfully closed with Process Monitor! Please keep sending me screen shots and log files of the cases you solve.