Windows 10 IoT Enterprise: The Case of the Magical Disappearing Utilities

By Sean D. Liming and John R. Malin
Annabooks – www.annabooks.com

October 2020

Windows 10 Enterprise LTSC 2019

We can file this one under *learning something new every day*. Using Windows Desktop in an embedded/IoT solution has lots of advantages and some challenges. The biggest advantage is the backward compatibility support for older applications. One of my clients had a Windows 95 application that was able to run on Windows 10 with just a few changes. The key to this backward support is the Win32 subsystem in Windows that provides the support to run 32-bit applications on a Windows 64-bit OS. It is pretty incredible what is still running out there, but sometimes the backward compatibility can create a gotcha that you didn't see coming.

One of these gotchas is the case of the magical disappearing utilities. Now that this has happened a couple of times that I know of, it is time to document this issue. Here is the scenario: you have a 32-bit application that has been running on Windows for years. You want to lock the system down by protecting drive C with the Unified Write Filter (UWF), encrypt the system with BitLocker, and make your 32-bit application the shell using Shell Launcher. More importantly, you want to run it on Windows 10 64-bit to take advantage of having more memory available for a future 64-bit version of the application.

You go through the process to set up the OS and test the application with Explorer.exe as the shell for the system. You test how the application interacts with UWF by enabling it with the uwfmgr.exe utility. You may have called out uwfmgr.exe from your application rather than using the available UWF WMI API. You also test how BitLocker needs to be set up in manufacturing to capture the recovery key. Everything is testing out ok. Now, you take the next step to enable Shell Launcher to make your 32-bit application the shell rather than Explorer.exe. The system boots up and the application is now the shell for the system. So far so good, but then you try to enable uwfmgr.exe and something doesn't work. The uwfmgr.exe is not found. You dig into troubleshoot the issue, maybe using a backdoor to get to control panel and command window. You see that uwfmgr.exe is not in c:\Windows\system32. You also see that the BitLocker control panel is missing, as well as, the manage-bde.exe utility is also not in c:\Windows\system32. They have magically disappeared.

The solution to this mysterious disappearance was not obvious, and I reached back to Microsoft to see if they had any ideas. At first, it stumped them, too, but after some tests and conversations, the answer turned out to be one that no one would have seen coming. When the 32-bit application boots as the shell, the shell is in the Win32 subsystem. Launching anything like control panel or a command window is also in the Win32 subsystem. What happens behind the scenes is that the c:\Windows\SysWow64 folder that is intended to support 32-bit applications becomes the c:\Windows\System32 folder since we are in the Win32 subsystem. Since uwfmgr.exe, BitLocker control panel, and manage-bde.exe are not in the c:\Windows\SysWow64 folder, they look like they have magically disappeared. If you run the following to compare of the two folders:

```
robocopy c:\Windows\System32 c:\windows\SysWOW64 /L /NJH /NJS /NP /NS
```

You will find that there are about 1651 files that are not in c:\sysWOW64. There are a couple of ways to work around the problem. The first is to rebuild the application to be 64-bit. This will require some testing, but it is a clear path around the issue. The second is to keep the application as 32-bit, but use the UWF WMI API to control UWF, which you should do regardless rather than shelling out to uwfmgr.exe. Of course, enable BitLocker before enabling Shell Launcher.

Normally, this scenario doesn't happen on everyday Windows computers. It seems reasonable that no one could have thought to check for this possibility. It took the perfect combination of an

Embedded/IoT system using Shell Launcher to launch a 32-bit application as the shell to uncover the issue. With many companies continuing to migrate to Windows 10, please keep this issue in mind as you move forward.