RegistryWriteVirtualization

A Flexible Virtualization feature for MSIX Packages

Timothy Mangan

TMurgent Technologies

August 2, 2025

Introduction

Last month I released a white paper introducing *Flexible Virtualization* https://www.tmurgent.com/appv/images/WhitePapers/FlexibleVirtualization.pdf and held a webinar for those that prefer watching over reading (See Video) .

That paper covered the overall concept from Microsoft of FlexibleVirtualization for MSIX, and in particular one of the two features within it, FileSystemWriteVirtualization.

In this paper, we cover the second feature, RegistryWriteVirtualization.

When running applications inside an MSIX container, the general intent is to containerize not only the assets of package, but also most of the application associated settings and changes made when the application is running inside the container.

There are built-in exceptions to this, such as the TEMP folder, and when using the PSF with MfrFixup we also have added some folders, like the Documents folder where redirection is made to the native location.

FileSystemWriteVirtualization enabled a specific list of folders, under the user's AppData Local or AppData Roaming folders, that would also receive this treatment. This is something we couldn't do in the PSF itself because the underlying MSIX runtime would redirect it underneath such efforts.

RegistryWriteVirtualization works on enabling a specific list of Registry Keys (under HKCU only) similarly.

About Flexible Virtualization

Although covered in the prior paper, here is the highlights on this. At this point, the all versions of Windows 10/11 that are still supported have these features available.

In researching this topic, we located a Microsoft web page on Microsoft Learn with the title "Flexible Virtualization", and this page attempts to describe the features we want to investigate and even has an example of how it is to be used. The paper is incomplete, and at times possibly misleading to those reading it by not being careful enough with terms. Still, it makes a good place to start to understand the what and whys and I recommend you give it a once over now. Find this page at https://learn.microsoft.com/en-us/windows/msix/desktop/flexible-virtualization

Flexible Virtualization uses three different schema extensions. The three parts to the schema extensions that fall under Flexible Virtualization are:

• A new rescap:capability setting, unvirtualizedResources.

• desktop6:FileSystemVirtualization and desktop6:RegistryVirtualization

• virtualization:FileSystemVirtualization and virtualization:RegistryKeyVirtualization.

A WARNING

SOME OF YOU ARE GOING TO READ THE DESCRIPTION OF WHAT THIS FEATURE DOES AND IMMEDIATELY WANT TO ADD IT TO EVERY PACKAGE YOU MAKE.

PLEASE DO NOT DO THAT. IT SHOULD BE USED SPARINGLY, ONLY WHEN THE PACKAGE WON'T WORK WITHOUT IT!

What is RegistryWriteVirtualization?

It is actually the opposite of the name, but only for a portion of the virtual registry.

RegistryWriteVirtualization defaults to a setting of enabled when not listed in the AppXManifest. But when added to the AppXManifest.xml file of a package, it can be used to poke holes through portions of the virtual registry to allow the containerized application to write directly to the real registry.

Microsoft App-V had a similar feature, but the only application I ever used it for was Adobe Reader. And Adobe Reader works fine without this feature under MSIX.

The feature may only be used to poke holes under HKCU, and not HKLM. After declaring the feature in the package, the AppXManifest also requires listing HKCU keys that should receive the effect. Adding this into the AppXManifest file also requires other additions to the manifest file, consistent with the other Flexible Virtualization feature.

In the 6.1 version of TMEditX, we added an Available fixup that adds this feature, and when applied, it defaults to a listing to target all of HKCU\Software. After adding the feature, you can edit the AppXManifest.xml file to be more specific, perhaps covering only HKCU\Software\VendorName or even a more targeted approach.

Using this feature substantially breaks the promise of the clean uninstall for your MSIX packages. This will lead to unexpected issues, especially in test environments where packages come and go a lot. If you use this feature in a test package, uninstall it, and then add a new package for this app (even if the package name is different), your new app will be affected by the registry changes made by the old package.

Implementing in AppXManifest.xml file

The AppXManifest file needs these changes in order to use this feature.

- If the Package item does not yet define references to these schemas, they must be added. Best practice is to also add them to the IgnorableNameSpaces parameter on the Package element also.
  - rescap
  - desktop6
  - virtualization

  An example Package element might look like this (with the changes in **bold**):

```
<Package
xmlns="http://schemas.microsoft.com/appx/manifest/foundation/windows10"
xmlns:uap="http://schemas.microsoft.com/appx/manifest/uap/windows10"
xmlns:uap3="http://schemas.microsoft.com/appx/manifest/uap/windows10/3"
xmlns:uap10="http://schemas.microsoft.com/appx/manifest/uap/windows10/10
"
xmlns:desktop6="http://schemas.microsoft.com/appx/manifest/desktop/wi
ndows10/6"
xmlns:rescap="http://schemas.microsoft.com/appx/manifest/foundation/windo
ws10/restrictedcapabilities"
xmlns:virtualization="http://schemas.microsoft.com/appx/manifest/virtual
ization/windows10" IgnorableNamespaces="desktop6 rescap uap uap10 uap3
virtualization">
```

- There is a capability that must be added to the capabilities list. This is actually the same capability we needed to add to use the FileSystemWriteVirtualization feature. Here is an example of what to add:

```
<Capabilities>
        <rescap:Capability Name="runFullTrust"/>
        <rescap:Capability Name="unvirtualizedResources"/>
</Capabilities>
```

- The package Properties element needs two new sub-elements, one to request the specific feature and the other to provide the list of affected keys.

```
<Properties>
<DisplayName>TMurgent BadAssTest</DisplayName>
<PublisherDisplayName>Packaged by TMurgent Technologies, LLP</PublisherDisplayName>
<desktop6: RegistryWriteVirtualization >disabled</desktop6: RegistryWriteVirtualization >
<virtualization: RegistryWriteVirtualization >
<virtualization:ExcludedKeys>
<virtualization:ExcludedKey>$(KnownFolder:LocalAppData)</virtualization:ExcludedKey>
</virtualization:ExcludedKeys>
</virtualization: RegistryWriteVirtualization >
</Properties>
```

- If needed, additional ExcludedKey elements may be added under the ExcludedKeys element.

Where is it useful?

I know of no application that requires this feature at this time. But I am sure there are some out there, so we are making it available for you to try it out if your package is having an unexplainable registry issue in the HKCU hive with the PSF RegLegacyFixup added.