WIPRO NGA Program – DWS B 2

Capstone Project Presentation –4th and 5th Sept

Project Title Here – MSIX with PSF Fixups

Presented by – Divyanshu Chaubey

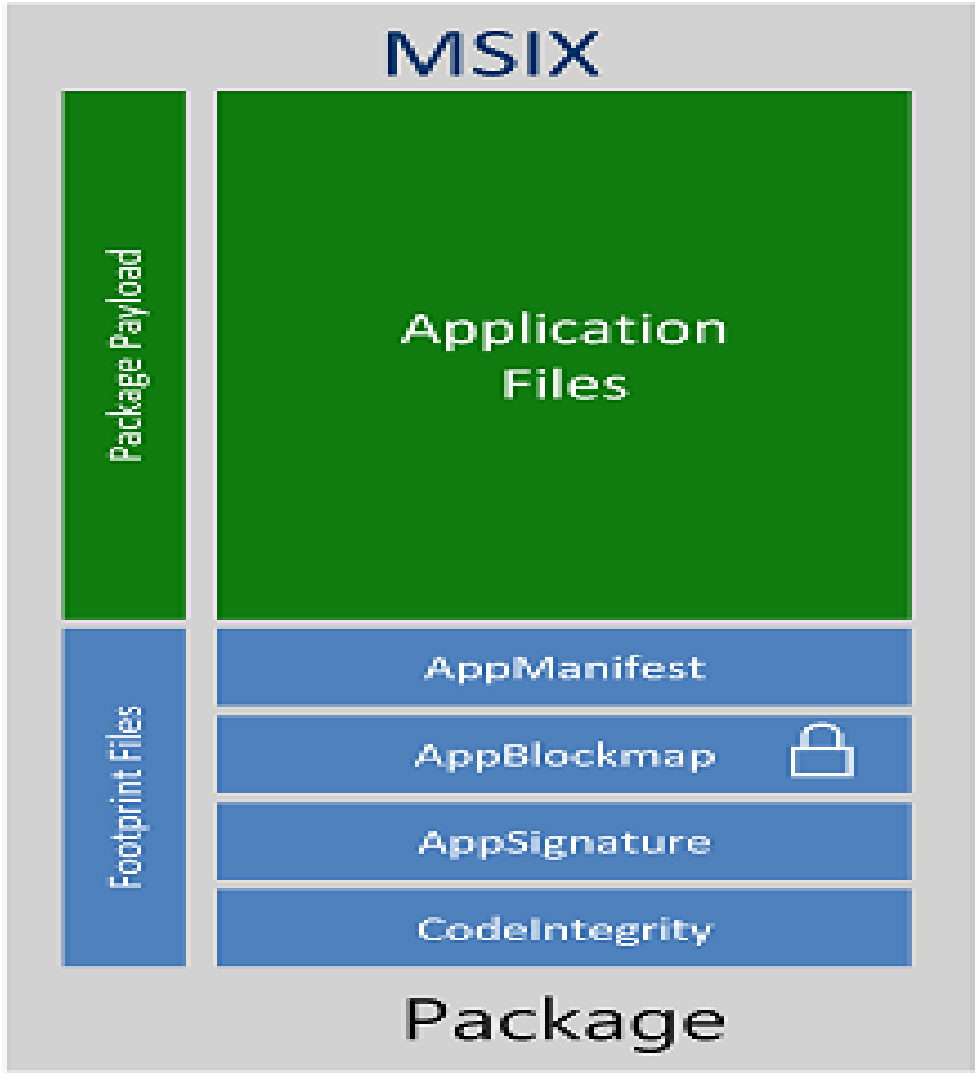
What is MSIX?

- MSIX is a Windows app package format that provides a modern packaging experience to all Windows apps. The MSIX package format preserves the functionality of existing app packages and/or install files in addition to enabling new, modern packaging and deployment features to Win32, WPF, and Windows Forms apps.
- MSIX enables enterprises to stay current and ensure their applications are always up to date. It allows IT Pros and developers to deliver a user-centric solution while still reducing the cost of ownership of application by reducing the need to repackage.

Key features of MSIX

- **Reliability.** MSIX provides a reliable install boasting a 99.96% success rate over millions of installs with a guaranteed uninstall.
- **Network bandwidth optimization.** MSIX decreases the impact to network bandwidth through downloading only the 64k block. This is done by leveraging the AppxBlockMap.xml file contained in the MSIX app package (see below for more details). MSIX is designed for modern systems and the cloud.
- **Disk space optimizations.** With MSIX there is no duplication of files across apps and Windows manages the shared files across apps. The apps are still independent of each other so updates will not impact other apps that share the file. A clean uninstall is guaranteed even if the platform manages shared files across apps.

Inside a MSIX Package



Inside a MSIX Package

App payload
The payload files are the app code files and assets that are created when building the app.
AppxManifest.xml
The package manifest is an XML document that contains the info the system needs to deploy, display, and update an MSIX app. This info includes package identity, package dependencies, required capabilities, visual elements, and extensibility points.
AppxSignature.p7x
The AppxSignature.p7x is generated when the package is signed. All MSIX packages are required to be signed before install. With the AppxBlockmap.xml, the platform is able to install the package and be validated.

Benefits of app containers

- Apps that are packaged using MSIX can be configured to run in a lightweight app container. The app's process, and its child processes, run inside the container, and are isolated using file system and registry virtualization. For more info, see MSIX AppContainer apps.
- All AppContainer apps can read the global registry. An AppContainer app writes to its own virtual registry and application data folder, and that data is deleted when the app is uninstalled or reset. Other apps don't have access to the virtual registry or virtual file system of an AppContainer app.

Package Support Framework (PSF)

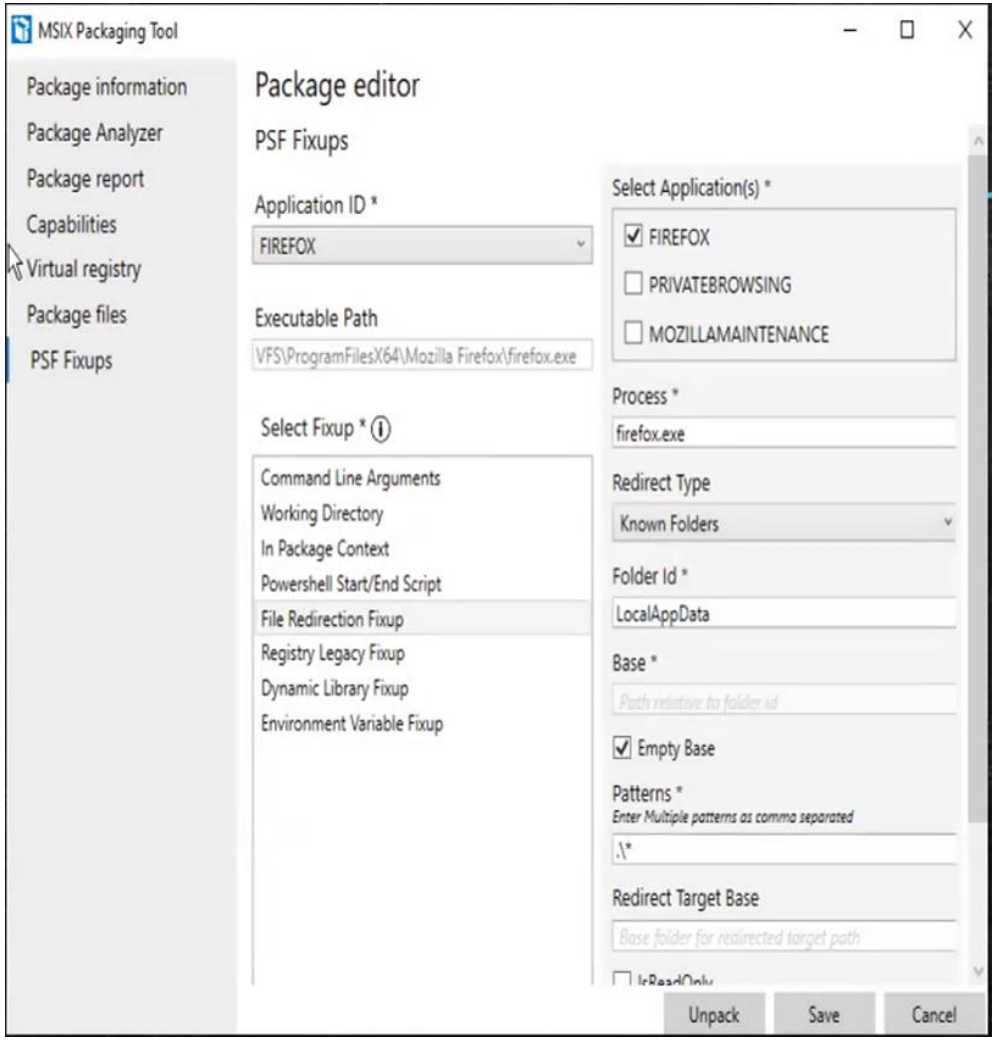
Bridging the Gap with PSF

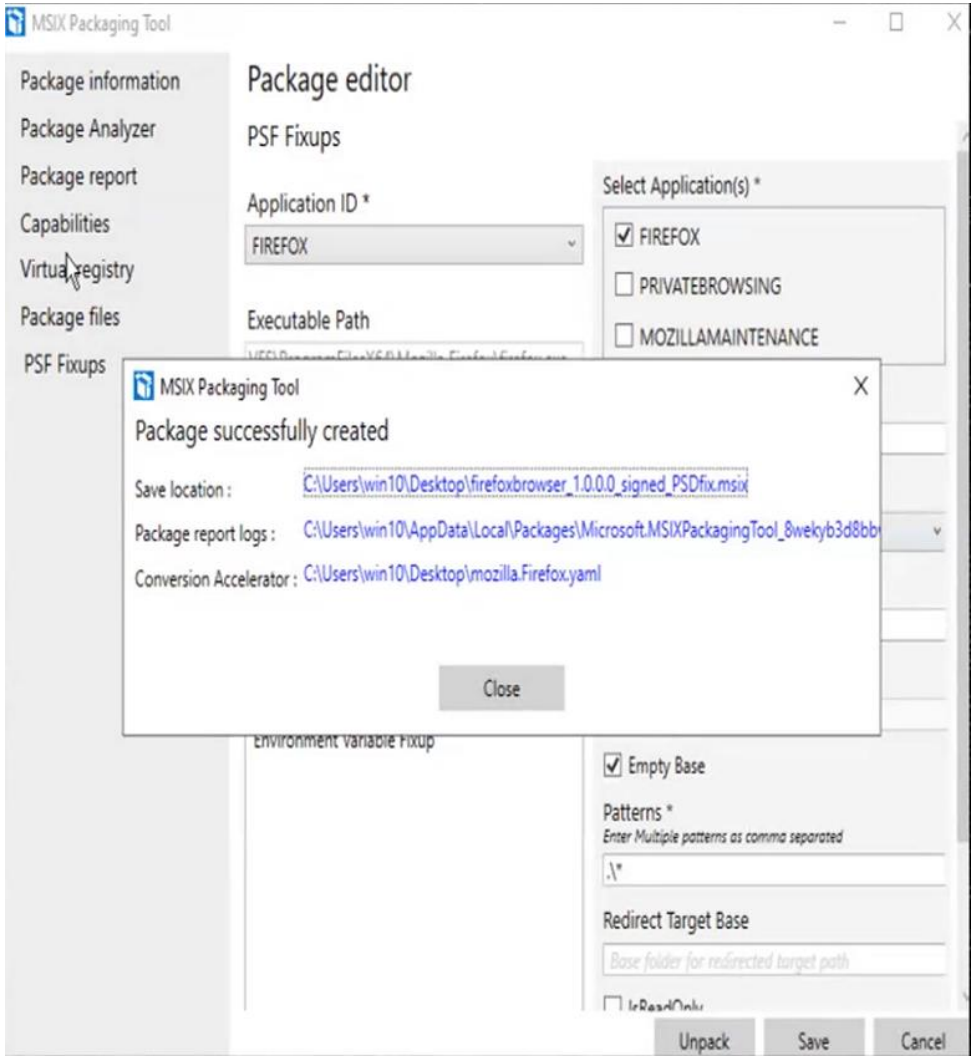
- **What is it?** An open-source kit from Microsoft that helps legacy applications run in an MSIX container without changing their source code.
- **How does it work?** It uses "shimming" or "detouring" to intercept problematic API calls from the application and redirect them to be container-compliant.
- **Analogy:** Think of PSF as a real-time translator. The app speaks "old Win32," and PSF translates its requests into "modern MSIX" that the container understands.
- App says: "Write log.txt to my install folder!"
- PSF intercepts and says: "No, write that to the safe, user-specific app data folder instead." The app never knows the difference.

Types of Fixups

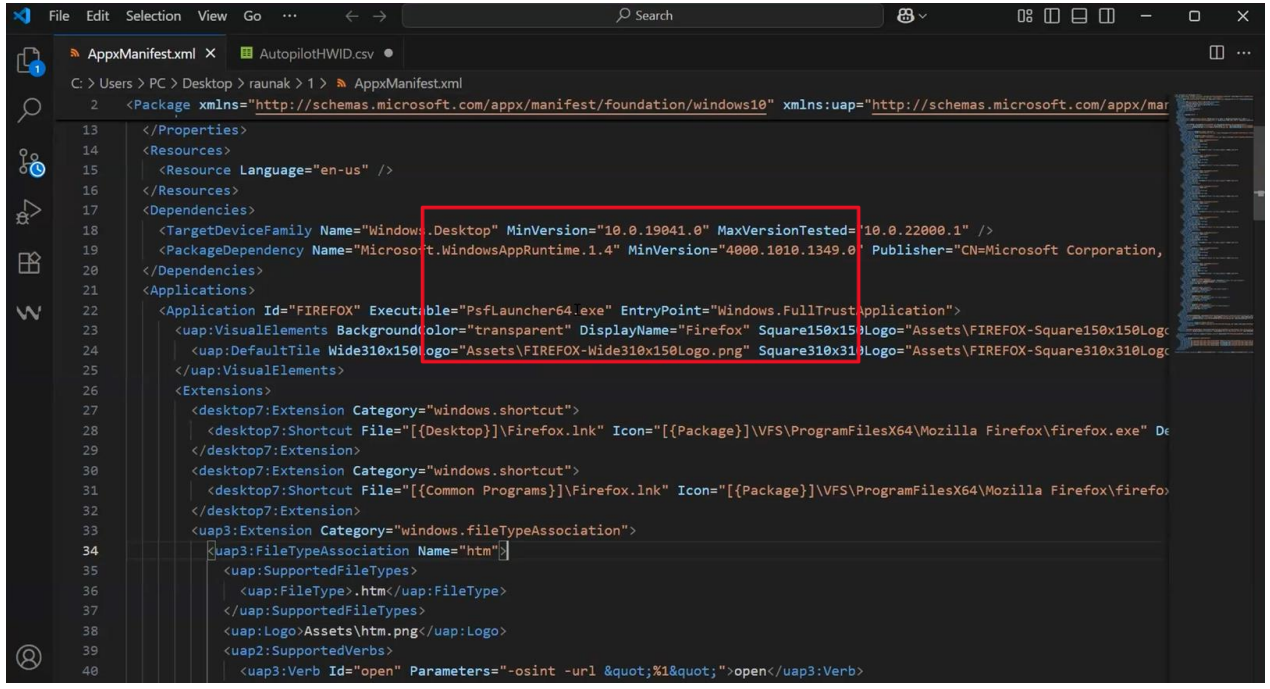
- **File Redirection Fixup:** This fixup addresses issues where an application attempts to write to a restricted folder, such as C:\Program Files. It redirects these file operations to a per-user location, preventing "Access Denied" errors.
- **Working Directory Fixup:** Many legacy applications depend on a specific working directory to find and load DLLs or other files. This fixup ensures the application launches with the correct working directory, even if the MSIX shortcut doesn't specify one.
- **Registry Fixup:** Similar to file redirection, this fixup intercepts registry calls that attempt to write to or read from a global location and redirects them to the application's private, virtualized registry hive.
- **Environment Variable Fixup:** Some applications rely on environment variables set during installation. This fixup helps to define and manage these variables within the MSIX package's isolated environment.

Applying PSF Fixups





Applying PSF Fixups



Conclusion:-

- **MSIX** is the future of Windows application deployment, offering security, reliability, and clean management.
- **Legacy Apps** often fail in MSIX due to assumptions about the filesystem and registry.
- The **Package Support Framework (PSF)** is the critical bridge, using API hooking to redirect problematic calls without code changes.

THANK YOU