ASSIGNMENT-10

Date:31-05-2025

Batch no-1

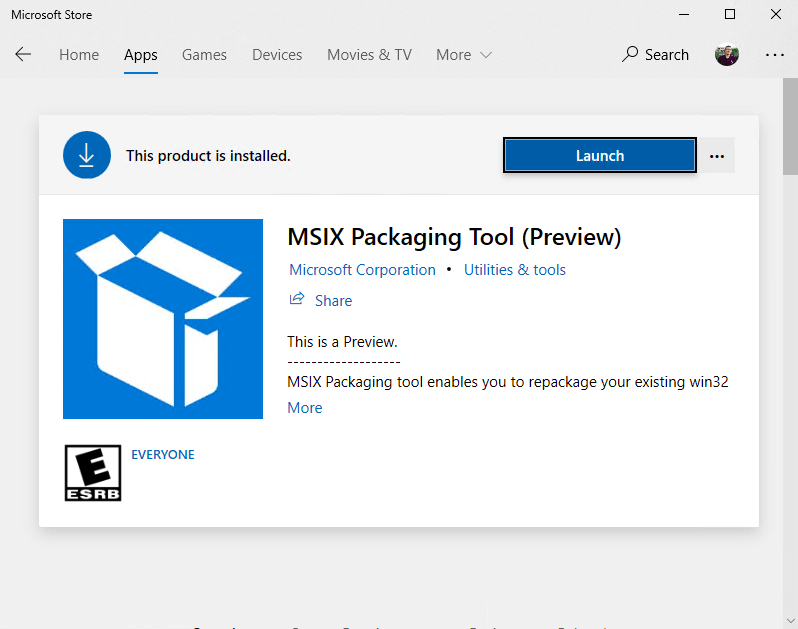
25VID1334-DWS

**Known issues and troubleshooting tips for the MSIX**

**Packaging Tool.**

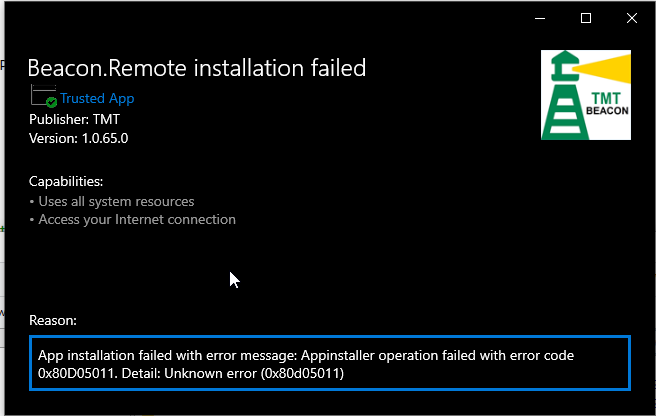
**Step 1: Introduction**

The MSIX Packaging Tool is a utility developed by Microsoft that helps developers and IT professionals convert traditional application installers (such as MSI, EXE) into the modern MSIX format. MSIX is designed to enhance the deployment, security, and management of Windows applications by providing a standardized packaging format for modern applications.

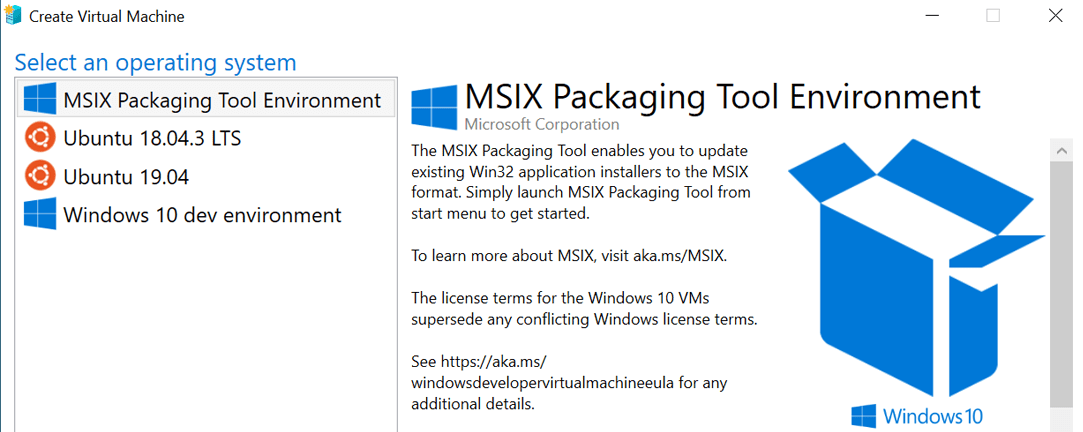


**Step 2: Common Issues with MSIX Packaging Tool**

**1. Installation Failures**

The installation of the MSIX Packaging Tool itself can fail due to missing dependencies, outdated operating system versions, or restrictions imposed by group policies or firewall settings.

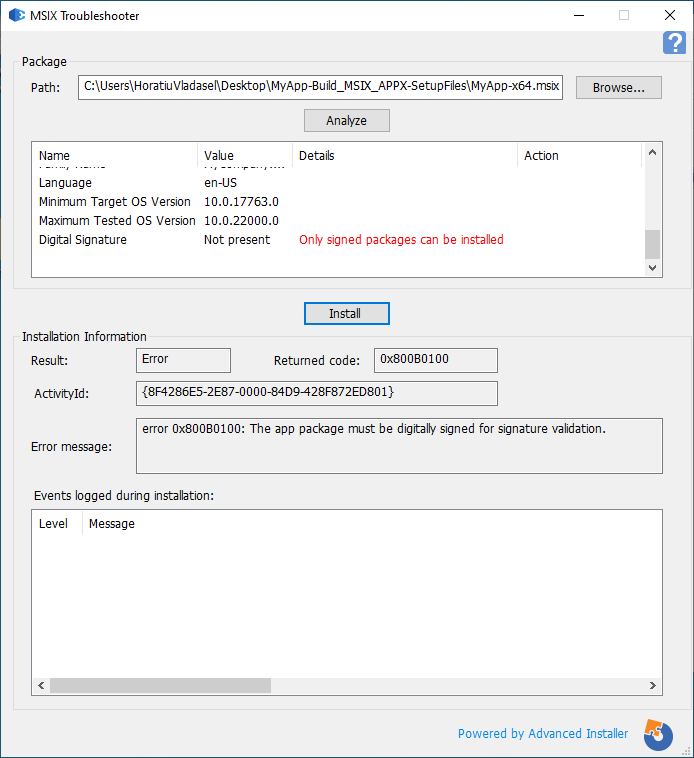
* **Solution:** Ensure your system is up to date with the latest Windows updates and all dependencies for the MSIX Packaging Tool are installed.

**2. Application Launch Failures**

After packaging an application into MSIX format, users may find that the application doesn’t launch or behave as expected.

* **Solution:** Check for missing runtime files, missing DLLs, or registry entries that are necessary for the application’s execution.

**3. Digital Signing Errors**

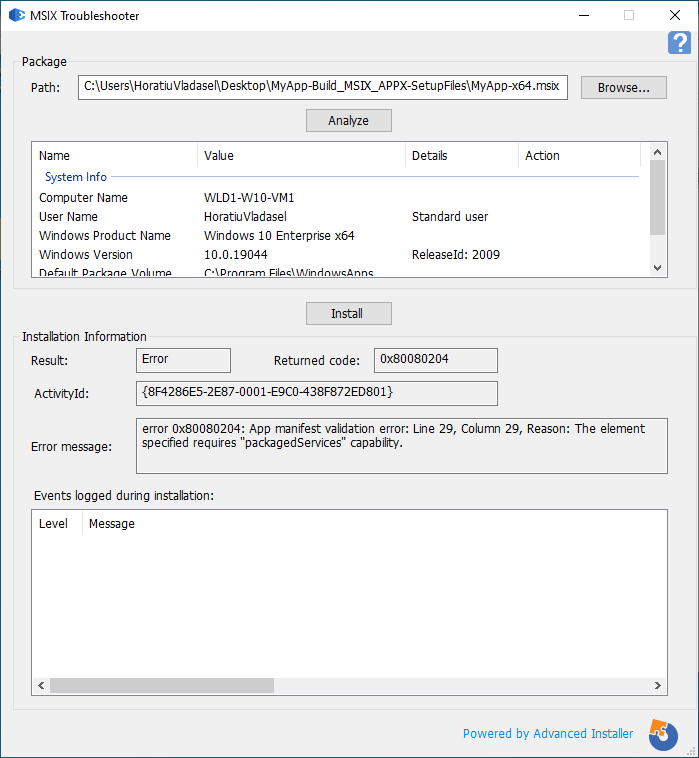
MSIX packages require digital certificates to be signed for security and integrity purposes. Without proper signing, deployment or installation of the package will fail.

* **Solution:** In this case, simply select the message under Action to add the MSIX package's certificate to the target device's Trusted Certificate List..

**4. The packagedServices** :

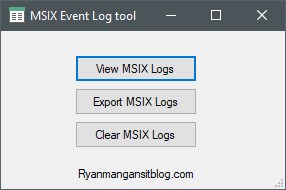
capability is not declared (0x80080204).

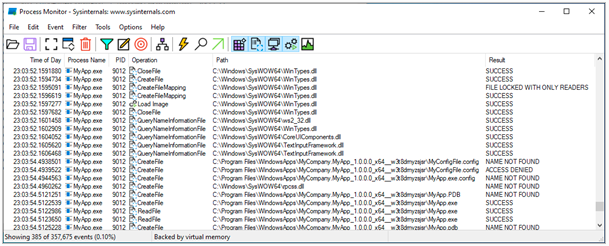
Cause of the issue: The “packagedServices” capability not declared for MSIX package installing a Windows Service.



**Resolution**: Declare “packagedServices” capability

**Step-3:** **Troubleshooting Tips**

* **Check Logs:**  
  Review logs located at %localappdata%\Microsoft\MSIXPackagingTool\Logs for detailed error information.
* **Use Process Monitor:**  
  Utilize tools like Process Monitor to trace system calls and identify missing dependencies.

****

* **Validate Package:**  
  Use the AppCertificationKit to ensure the package meets Windows Store requirements.
* **Repackage with Minimal Setup:**  
  Capture the application in a clean environment to avoid unnecessary inclusions.
* **Use Package Support Framework (PSF):**  
  Apply PSF to handle legacy application compatibility issues within the MSIX container.

**Step-4: Best Practices for MSIX Packaging**

* **Test on Clean Virtual Machines (VMs)**

Always test MSIX packages in isolated, clean VM environments to mimic end-user systems.

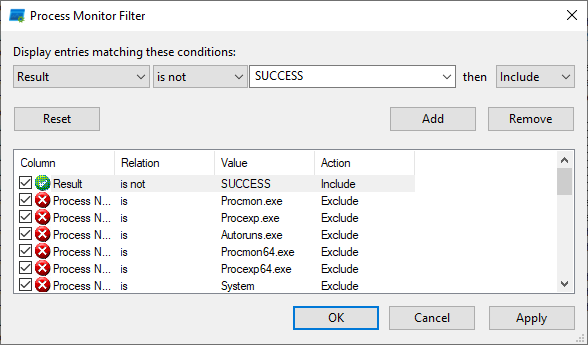
* **Ensure Proper Digital Signing**

Sign your MSIX packages with a valid, trusted certificate to ensure they are secure and can be trusted.

* **Keep Tools and Environment Updated**

Regularly update the MSIX Packaging Tool and related tools (Windows SDK) to ensure you're using the latest features and fixes.

* **Use Filters Wisely**

Carefully select which files and registry keys to include or exclude during packaging to avoid breaking the application’s functionality.

------------------------END---------------------------