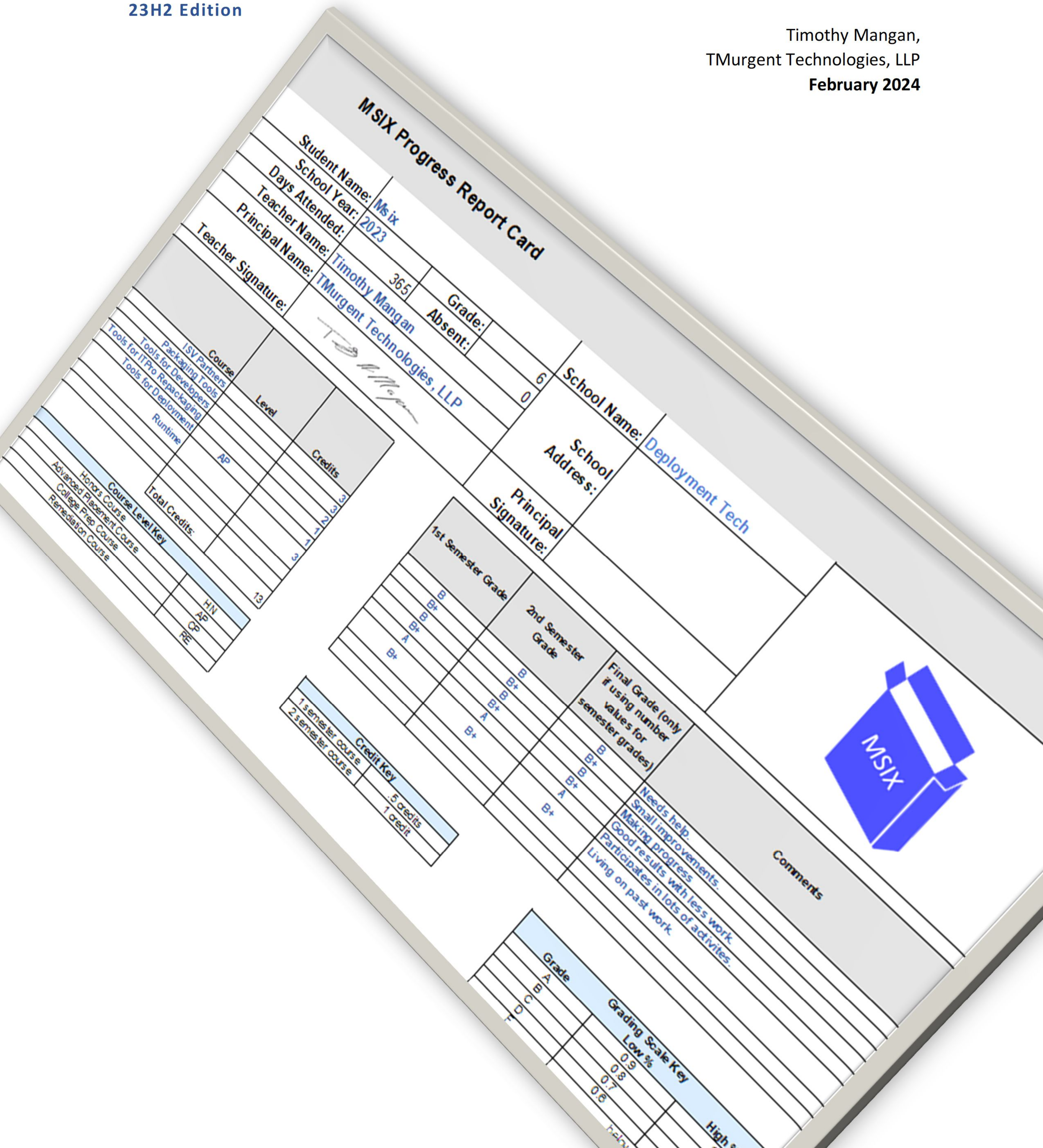
The MSIX Report Card

23H2 Edition

Timothy Mangan,
TMurgent Technologies, LLP

February 2024



MSIX Progress Report Card

Student Name:	Msix
School Year:	2023
Days Attended:	365
Grade:	6
Absent:	0
Teacher Name:	Timothy Mangan
Principal Name:	TMurgent Technologies, LLP
Teacher Signature:	[illegible]
School Name:	Deployment Tech
School Address:	
Principal Signature:	

Course	Level	Credits	1st Semester Grade	2nd Semester Grade	Final Grade (only if using number values for semester grades)	Comments
ISV Partners		3	B	B	B	Needs help.
Packaging Tools		3	B+	B+	B+	Small improvements.
Tools for Developers		2	B	B	B	Making progress
Tools for ITPro Repackaging		1	B+	B+	B+	Good results with less work.
Tools for Deployment		1	A	A	A	Participates in lots of activites.
Runtime	AP	3	B+	B+	B+	Living on past work.
Total Credits:		13				

Course Level Key	
Honors Course	HN
Advanced Placement Course	AP
College Prep Course	CP
Remediation Course	RE

Credit Key	
1 semester course	.5 credits
2 semester course	1 credit

Grading Scale Key		
Grade	Low %	High
A	0.9	
B	0.8	
C	0.7	
D	0.6	
F		

Introduction

Nearly six years ago Microsoft announced MSIX, a four headed effort that intends to be:

- A replacement for MSI based application delivery by software vendors.
- A new SDK for developers to use. The Windows App SDK (formerly known as Project Reunion) helps the developer to write natural code that works inside the MSIX container, or now also possibly outside of the container as desired.
- Tooling to help IT Professionals repackage existing software into MSIX. This involves both Microsoft developed tooling and tooling from third party vendors that are already in the repackaging business.
- And a runtime environment that protects applications and the operating system by using an updated container similar to that used previously for UWP programs.

In the original announcements Microsoft indicated that the road to MSIX would be a journey and it would take several release cycles to complete all the functionality needed. We are still on that road and are likely to be for some time. In recent years customers have become outspoken about the issues they encounter and question just how committed Microsoft is to this path.

This 2024 version of *The Report Card* is an update to our prior views; to review previous versions see:

- http://m6conf.com/index.php/reportcard/46-report-card-2019
- http://m6conf.com/index.php/reportcard/46-report-card-2020
- http://m6conf.com/index.php/reportcard/46-report-card-2021
- http://m6conf.com/index.php/reportcard/46-report-card-2022 and
- http://m6conf.com/index.php/reportcard/46-report-card-2023

We can summarize the changes this year as follows:

- Microsoft has been showing increasing support for MSIX based applications in their own offerings this year. There are many smaller apps, such as Notepad, but Edge and Teams are now MSIX, applications with significant functionality. There remains much to be done with their own apps, but we are seeing progress.
- Third party vendors have also made progress in moving to MSIX, and we are glad to see the increased offerings, but there does seem to be a trend for older applications to put a version into the Microsoft Store that is MSIX but lacks full functionality. Some of these seem like a work in progress and should improve over time, but others may just be a bait-and-switch to get users to buy the complete product.
- The level of application compatibility that can be achieved with repackaging existing applications into MSIX by IT Professionals continued to improve, but only slightly this year.
- Our own efforts this year have been focused on making it easier to perform this work with out TMEditX tooling to augment Microsoft's MSIX Packaging Tool. In most cases, this work can be automated, however some applications require manual efforts. In addition, work was done to help customers that use Microsoft App-V have a viable way to convert many of those existing packages into MSIX.
- At the end of the year, Microsoft finally added a method to directly add the PSF into packages using their own packaging tool. Unfortunately, at this time, this is the older Microsoft code base

for the PSF, not the more modern version used by PsfTooling, TMEditX, and some other third party vendors. It also is a manual process that is more trial and error than we'd like.

- MSIX App Attach is continuing to be a delivery vehicle for certain scenarios, with Vendors like VMWare, Citrix, and others having added support. As part of this report, we update our look at the performance impacts of using App Attach.
- Customers are openly questioning Microsoft's commitment to MSIX and looking to alternatives. Unfortunately, other than Microsoft's own App-V, none appear to be better (only different). Although progress on MSIX has been much slower than Microsoft seems to have anticipated, we do not see them giving up or moving to a new direction. Fortunately for customers, the existing methods of dealing with applications continue to exist for now. But the App-V end-of-life date (April 2026) is looming.
- In the upcoming year we will be returning to work on improving the application compatibility for older applications. Currently, work is in progress, but not ready in time for this year's report, to address more complex issues such as applications with COM, and an upcoming rewrite to the RegLegacyFixup to keep apps from accidentally avoid the PSF intercepts.

My intent is to update this report card in (roughly) January of the following years so that we can judge the progress. There are undoubtedly many other vendors that are active in this space and yet not included in this list due to my limited resources and/or unfamiliarity with their offerings. If you are one of these companies and are supporting MSIX, I apologize for the slight. Please contact me to ensure that I include you in the future.

And here is this year's report card…

1 Support by Software Vendors to release in MSIX format

B Microsoft’s Independent Software Vendor (ISV) Partners, who build end-user applications for the Enterprise, improved from an “B-” to a “B” for releasing software products in an MSIX format.

The number of vendors with releases as MSIX packages in the Windows Store has increased. We do not have any hard numbers to offer, as store entries rarely indicate if the are containerized apps or not.

Most of the offerings I’ve noticed are products that also exist in native form and have been ported. I am unsure about the amount of products that are new and offered as MSIX only. Some of the ported apps are incomplete implementations of the original software. These incomplete versions seem to exist in the store to make it easy for users to find them. Whether this is a placeholder to get them hooked and buy the full native package, or a temporary issue as the vendor continues to complete the port is anyone’s guess.

The bottom line.

This category continues to be the most important part of MSIX success. If the vendors don’t go for it, all is lost for MSIX.

It took mainstream vendors about 5 years to fully embrace the .Net Framework when it was released around 20 years ago. The new SDK allowing for new apps as MSIX is out, but those vendors need time .

Mixed messages by Microsoft on their commitment to getting apps to run inside containers must be cleaned up, and additional help for software vendors provided.

Many look to Microsoft for taking the lead. We have seen Microsoft release quite a bit of software in the containers, especially Edge and now Teams, but there seems to be no outbound effort to make a lot of noise about this. Office remains the poster child (Microsoft chimes in about once a year that they are working on it), and a fully functional release of Office as MSIX would end any debate.

2 Support by Tooling Vendors

B+ Tooling Vendors stayed steady at “B+” this year.

This category is further broken into three sub-categories, and I will grade each of these independently as well. The category includes:

- Tooling for developers.
- Tooling for IT Pros in repackaging.
- Tooling for distribution.

Many of the players are involved in multiple of these subcategories, and Microsoft themselves are also a first party vendor in this space as well.

2.1 Tooling Vendor Support for Developers

B Microsoft has been doing a lot for developers, not only in improving the SDK, but advanced in Visual Studio itself.

The SDK is maturing and supporting more APIs that developers require and were initially missing. Much more is to be done, but Microsoft has been delivering on their promises.

Visual Studio improvements have been outstanding. The new GitHub Copilot integration is just fantastic and will turbocharge software development across the board.

Microsoft’s interest in everything AI hides some of the other good work they are doing to support developers to move to containerized applications.

2.2 Tooling Vendor Support for Repackaging

B+ Software Vendors supporting IT Pro Repackaging of applications into MSIX earned a “B+” again this year. These established vendors on the packaging side have been very active in the MSIX community from the beginning, but we hear concerns and frustration from them regarding Microsoft’s level of commitment.

As was the case last year, this year we saw a bigger emphasis in making it easier to repackage the apps that can work rather than focusing on more progress in application compatibility.

Microsoft continues to invest in their MSIX Packaging tool. The preview release at the end of 2023 included, for the first time, a capability to add a version of the Package Support Framework (PSF) directly in their tool. A welcome addition, but unfortunately their version of the PSF lacks 2.5 years worth of improvements made in the TimMangan fork used by some third party vendors, limiting capabilities and requiring much more manual effort that third party tooling.

TMurgent's tools, PsfTooling and especially TMEditX, offer this improved PSF version plus the ability to take advantage of platform changes to MSIX that Microsoft has been quietly making and are not used in the Microsoft MSIX Packaging Tool yet.

Meanwhile, the Package Support Framework (PSF) in the TimMangan fork has advanced the game with the usual improvements. Most of the changes are small fixes for applications that behave unusually, but at the end of the year two new features were added to the RegLegacyFixup:

- **Deletion Markers**. This has been a long-standing request, a portion of what is needed to be able to hide the existence of a natively installed version of an existing app from the version running inside the container. Our expectation was that Microsoft would port over the deletion marker support directly in the virtual registry from what App-V did. But instead they added a support to their fork of the PSF late in the year. This was ported into the TimMangan fork as well.
- **Java Blocker.** The most common need for the deletion marker is to package up an older version Java with an app that can't use the latest version. A new rule was introduced to the TMurgent fork to make this specific issue easier to fix.

Other packaging vendors also include the PSF in their own repackaging products, combining the equivalent of PsfTooling and the MMPT into a single experience. Some appear to be following my fork of the PSF to improve their products. Seemingly all are doing more to make it easier for the people doing the repackaging to be successful, and all in quite different ways.

The growth in this space this year has been in vendors that have offerings to help IT with more of the application lifecycle in one tool. They often include their own packaging software, or repackage/refer to other packaging tools in their workflow. While these vendors are not exclusive to a particular packaging format, MSIX is driving their businesses.

2.3 Tooling for Distribution

A Software Vendors with support for MSIX Distribution at a solid "A" again this year.

There is not a lot that is new to talk about here this year.

3 MSIX Runtime Support

B+ The MSIX Runtime remains at a "B+" grade this year. On one hand, we haven't seen a lot that is new this year. But the other hand, some of the work they did previously that we could not use is now coming on line.

This is especially true for customers running Windows 11. With the end of life for Windows 10 approaching, this is only a short term issue now.

In past years, Microsoft did respond to many of our requests by expanding the schemas and implementations in the OS that can improve MSIX compatibility. Over the last two years, Microsoft has also been addressing the need to accurately document what that is and where it works. Much more is needed, but we finally have enough to expand the app compat story for older applications, and the results in this year's report card reflects this.

Microsoft could do a much better job communicating with their partners and customers than what the current documentation provides. When the goal is to allow your customers to do more, it doesn't mean they should guess at what might work.

In the meantime we continue to parse through what we can find and try to figure out when and how to take advantage of the work already done.

4 Repackaging Testing

At the end of the day, we need to be able to deploy the apps. Without a lot of vendor supplied apps available, we can work the path of repackaging existing applications into MSIX and testing those. In this section, as in years past, I'll document the result of the testing. The experience with this testing was crucial to, and had significant impact up, the grading earlier in this report.

This year, as part of the *Report Card*, I expanded the set of applications to be tested in a repackaging scenario to 96 to improve coverage. Most of these applications were applications that we commonly see Enterprises distributing to employee workstations today. A few "special purpose" applications that I commonly use to demonstrate application integration capabilities were also included to ensure that we are covering the needs of most apps. Not included in the application mix were applications that are known to not work under any virtualization or container, such as App-V and MSIX – for example those with device drivers or plug-ins for Internet Explorer. The applications that were added into the test suite this year were all apps that would have a higher probably of failing that those included in past years.

These applications were repackaged and tested three times, all on Windows 11 23H2:

- Using Microsoft App-V.
- Using the Microsoft MSIX Packaging Tool (2023.1212 release) without the PSF.
- Using the Microsoft MSIX Packaging Tool (2023.1213 release) with TMEditX 4.0.0.0.

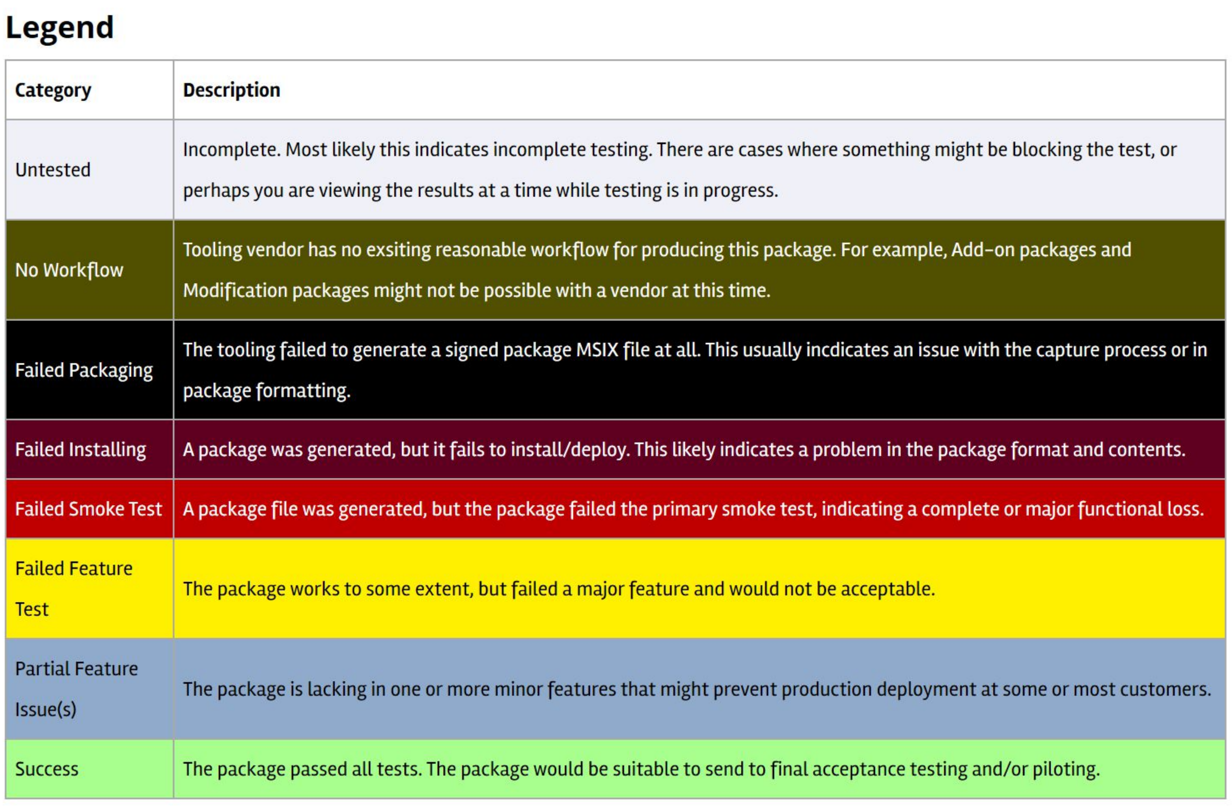
We did not test on Windows 10 this year. Many of the advances made for repackaging depend upon MSIX improvements that Microsoft made available only for Windows 11. With the end of life for Windows 10 looming, we felt that Windows 10 testing would not be relevant.

We also did not test using the Microsoft MSIX Packaging Tool with their own PSF version, nor the MSIX Packaging Tool with PsfTooling, but anecdotally we can suggest that we would expect:

- Using Microsoft's PSF version we would probably see around a 50% compatibility maximum, after much manual effort.
- Using PsfTooling to add the PSF in the MSIX Packaging Tool would provide about 60-65% compatibility.

While every attempt is made to be objective in testing, the interpretation of test outcomes nevertheless is subjective. Someone else testing the same packages might categorize the results into different buckets than me (especially between the "subjective" buckets). But we must start somewhere!

In the charts that follow, color coding is used to signify the categorization of the testing result. The following table should be used to interpret those colors:



Legend

Category	Description
Untested	Incomplete. Most likely this indicates incomplete testing. There are cases where something might be blocking the test, or perhaps you are viewing the results at a time while testing is in progress.
No Workflow	Tooling vendor has no exsiting reasonable workflow for producing this package. For example, Add-on packages and Modification packages might not be possible with a vendor at this time.
Failed Packaging	The tooling failed to generate a signed package MSIX file at all. This usually incdicates an issue with the capture process or in package formatting.
Failed Installing	A package was generated, but it fails to install/deploy. This likely indicates a problem in the package format and contents.
Failed Smoke Test	A package file was generated, but the package failed the primary smoke test, indicating a complete or major functional loss.
Failed Feature Test	The package works to some extent, but failed a major feature and would not be acceptable.
Partial Feature Issue(s)	The package is lacking in one or more minor features that might prevent production deployment at some or most customers.
Success	The package passed all tests. The package would be suitable to send to final acceptance testing and/or piloting.

Additionally, this year the result graphics offers an additional summarized version of the results in bar form that combines the first five categories into a failed status. This creates a simpler subjective view where the reader can apply their own subjective view of compatibility with categories of “Good”, “Maybe”, “Maybe not”, and “No way”.

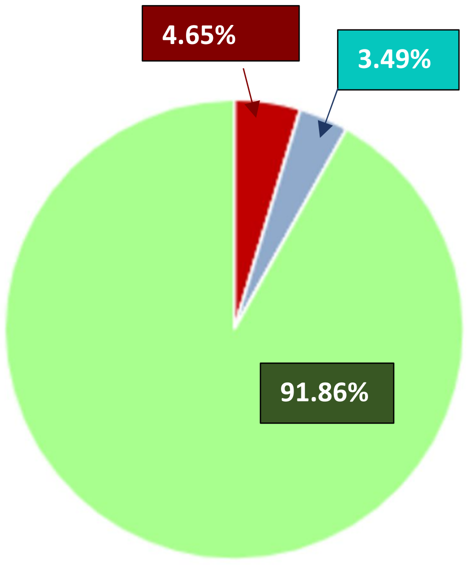
Finally, it is worth noting that the testing performed was using recapturing techniques. There may be ways to build a MSIX package using traditional install builder tools from these same vendors that have been extended to support MSIX, however that was outside the scope of this testing.

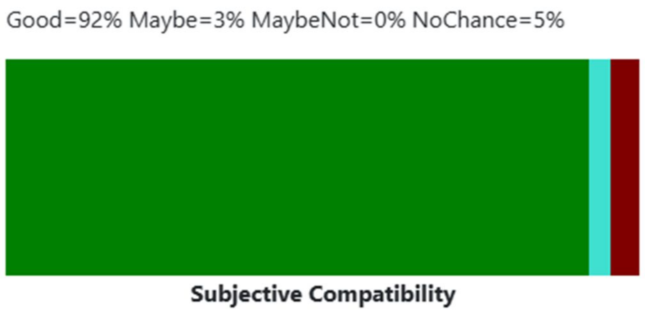
4.1 Comparative results using Microsoft App-V

Many of the applications in the list are older applications that are challenging to deploy in today's environments. Indeed, although I did not categorize the list using Native installation techniques, the results would be far less than for repackaging in App-V. This is after all why we have App-V!

The 86 packages were packaged on Windows 10 22H2 and tested on the same OS. The 2004 ADK Sequencer was used and packages were not further fixed up by TMEdit which would have improved the results.

Result summary for 96 packages using Microsoft App-V from 2004 ADK Sequencer and Win10 22H2

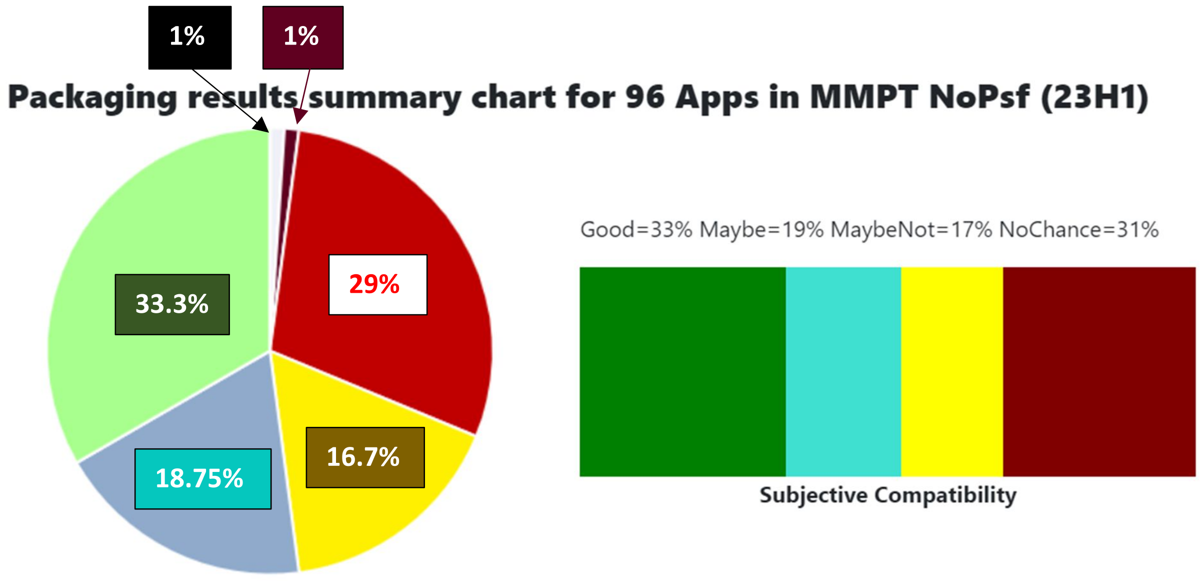




Not surprisingly, these results were very good using seasoned tooling and collective wisdom on techniques to package with App-V.

4.2 Results from Packaging with Microsoft MSIX Packaging Tool

The same 96 applications were packaging using the 2023.1212 “preview” release of the MMPT, without trying to add the older version of the PSF. These tests use current best practices for packaging, without the use of additional tooling or extraordinary measures such as manual manifest editing.



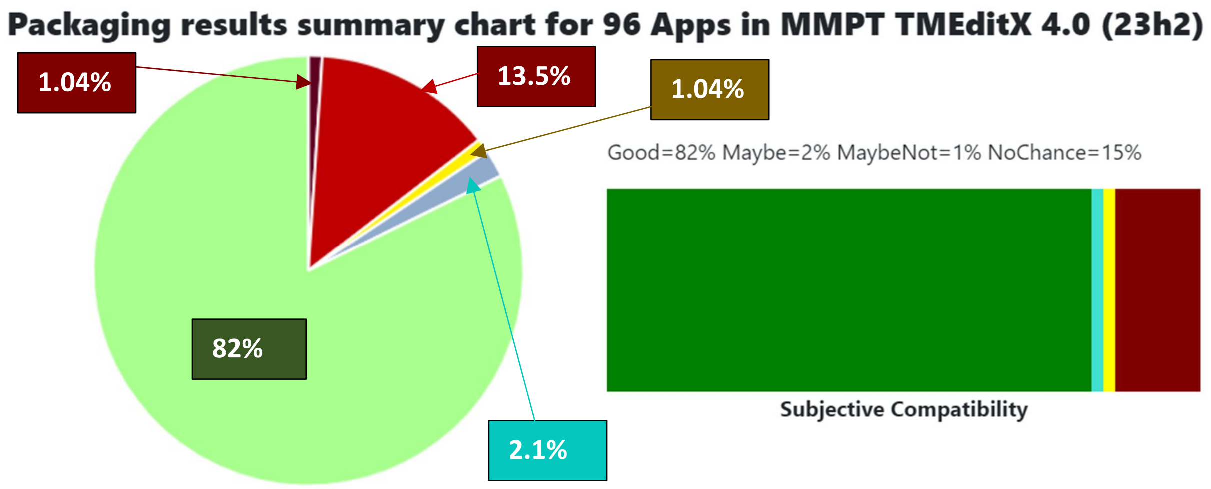
The applications that did not repackage were new applications in the test this year, or new versions of apps previously tested OK. Otherwise, I consider the results to be about the same as per last year’s results for the same unchanged apps. In the past I have cautioned that the results will be lower on back-rev Operating System versions due to the runtime improvements of the tested OS, but that (mostly) no longer applies as those systems are now out of support.

Except for a few cases, these packages were fully automated using remote package capture techniques. The reason that a few of the apps were manually packaged was that the disabled Windows Updates were re-enabled by the OS during packaging causing bad packages, or installers that refused to automatically install. NOTE: The untested app also fails with the PSF added, so it would certainly have no chancce.

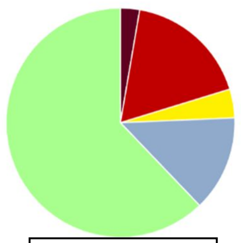
4.3 Results from Packaging with Microsoft MSIX Packaging Tool and with PSF

For these tests, the MMPT version 2023.1212 was also used, but the packaging process was enhanced by using TMEditX 4.0.0.7 to inject and configure the Package Support Framework (PSF) into the package as well as make additional improvements available in MSIX but not through the MMPT. (Information on TMEditX is available https://www.tmurgent.com/AppV/en/buy/tmeditx/tmeditx-about) .

For the most part, TMEditX was used via automation as well. When analysis indicated the need, the new MfrFixup (not available in the Microsoft PSF fork) in ILV-Aware mode and InstalledLocationVirtualization, the DynamicLibrayFixup, and EnvVarFixup, were added. In general, the RegLegacyFixup was automatically added to any package that analysis showed that the PSF was needed. This version of the PSF used contains a build of the latest PSF source code from GitHub available as of the beginning of January 2024 (v.2024.01.02). The same set of 96 Apps were tested.



Even with the more difficult application mix, the high-fidelity result managed to pop above the 80% mark! As seen by comparing versus the results from two years ago, it is many of the apps with minor problems that have been addressed. Meanwhile the categories of unusable apps dropped from 20% to 15%. Work done this year will be to attack that 15%.

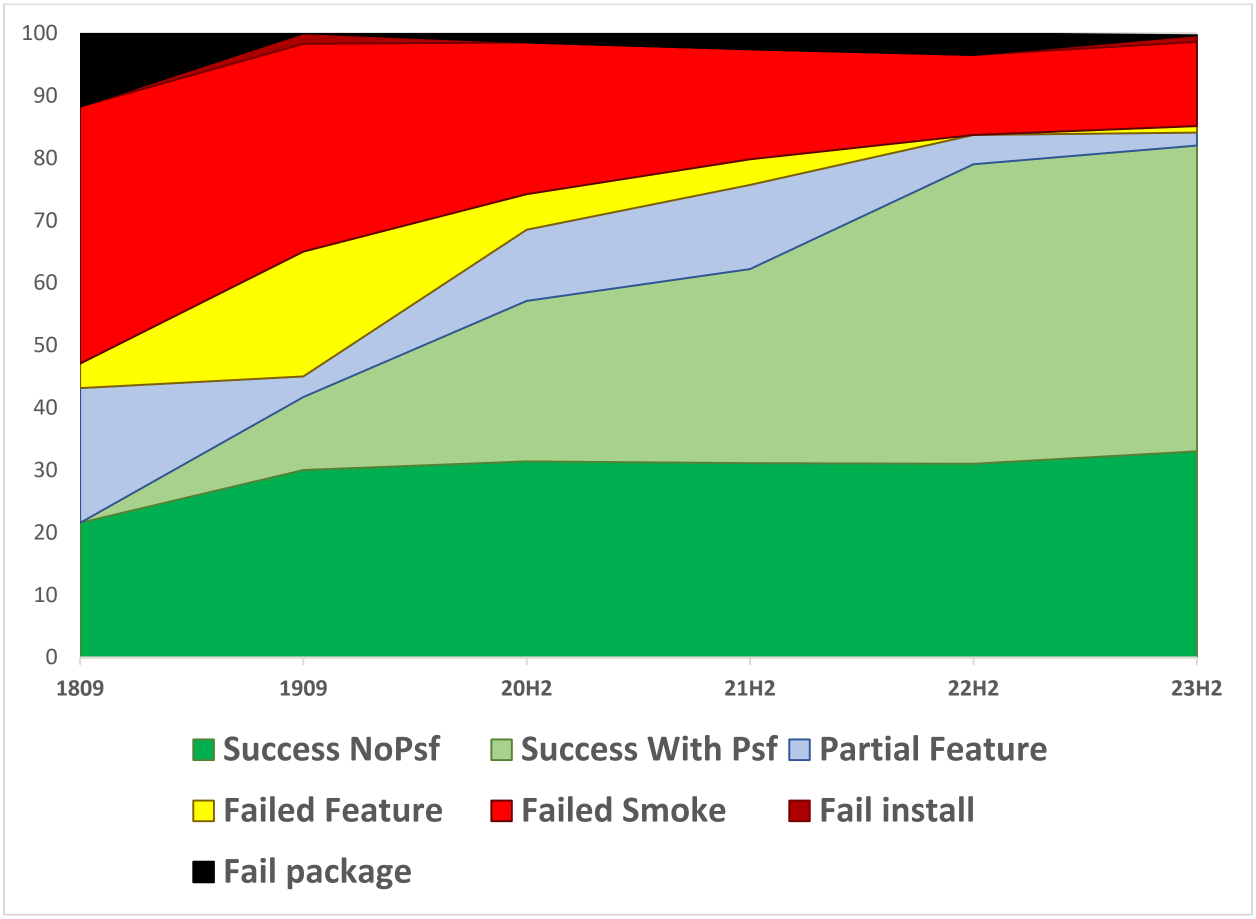

21H2 Results

But more striking than improved possibilities was the reduction in amount of effort that it took to produce those packages due to improvements in the analysis and available fixes. Work was also done to help existing App-V customers port their existing packages over to MSIX.

4.4 The trend

Now that we are in the sixth year of this testing, we can look at the numbers to see how the trend has been going.

Trend of MSIX Compatibility



While the improvement in application compatibility over the previous year has been small, we have continued to improve on the results, essentially nearly eliminating the “subjective” results. This is primarily due to a focus on improving the likelihood of being able to produce a successful application without much effort or expertise, which really doesn’t show up in this kind of report.

As noted earlier, we are already beginning to attach the apps that fail completely for next year, and by mid year hope to yield improve results.

4.5 Future Work

While Microsoft rarely, if ever, publicly discusses there future plans for MSIX, I can talk about my plans to improve the Application Compatibility for traditional apps repackaged MSIX packages.

Three areas of challenges for the support of additional types of packages have been identified and attempts to address them are under way for 2024:

1. Improved support for COM. Microsoft made significant changes for COM in the newer schema extensions a few years back. The Microsoft MSIX Packaging Tool does not use the new extensions. Until recently, using these new changes were not feasible in third-party tooling either, but due to improvements in the documentation, and OS support being more generally available to customers, I am working on identifying and fixing up COM requirements for applications in TMEditX. Most of these improvements will require deployment on Windows 11, but the lack of Windows 10 support is a problem that will shortly solve itself (Windows 10 EOL).
2. The work on COM will also allow the expansion of types of Shell Extensions that we can support.
3. PSF Registry. The added deletion marker and Java support by the RegLegacyFixup are good, but we discovered that there are apps calling standard Microsoft library APIs that internally bypass the functions that this fixup targets, causing access denied conditions that end in the app crashing or otherwise failing. Work is targeted in the PSF to identify these cases and add new intercepts to solve them.
4. PSF File fixups. A similar sort of issue to what we see in the registry also occurs in the file system calls. Work is targeted to identify cases and add additional intercepts to the MfrFixup (FileRedirectionFixup will probably not be addressed).

There is no certainty that changes to solve these issues will solve all of the problems of any specific application (the tough ones often have additional issues hidden by the one you see at first), I have hope to see measurable improvements in the numbers for this report next year.

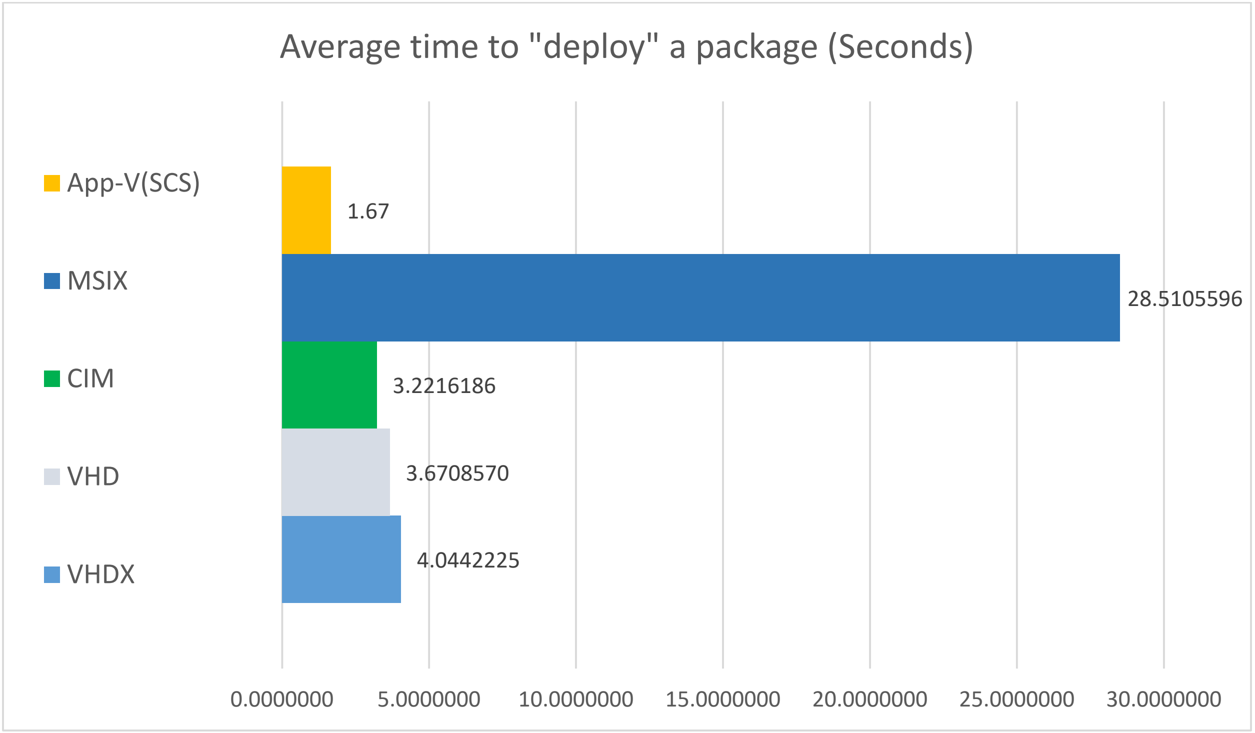
5 App Attach Special Supplement

As customers migrate from their use of Microsoft App-V, the most logical course is to look at App Attach as the evolution of App-V, able to provide fast publishing of packages on a per-user basis to a pooled or shared OS image.

Microsoft offers this themselves are part of the Azure Virtual Desktop under the name MSIX App Attach, but other partners, including Citrix, VMware, and AppVentiX use the same underlying technologies to deliver MSIX to those environments.

This year's report includes results from some testing of App Attach from a performance standpoint, something we have not looked at since App Attach was in Beta. In particular, we wanted to understand publishing performance as it relates to App-V deployments, and to understand the differences between the three supported application image formats for App Attach (VHD/VHDX/CIM), and on any impacts to application compatibility (versus MSIX direct installation).

A separate report card on App Attach is also being released, but the summary performance numbers from a 90 package test is shown below:



The application compatibility for AppAttach is mostly the same as for that of MSIX in general, but there a small number of cases where AppAttach may not work for a specific application package. See the full report for details [App Attach Report Card for 2024 (https://www.tmurgent.com/appv/en/resources/report-cards/129-reportcards/619-report-card-appattach-2024)](https://www.tmurgent.com/appv/en/resources/report-cards/129-reportcards/619-report-card-appattach-2024).

About the Author

Tim Mangan is an independent consultant and the owner of TMurgent Technologies, LLP. Recognized as an industry leader by Microsoft as an MVP for 17 years, and by Citrix for more than a dozen as a CTP/Fellow, Tim is generally known as "The Big Kahuna", having led the effort to build the original version of App-V at Softricity and builds many free and paid-for tools for packaging under both App-V and now MSIX.

TMurgent provides consulting and training around application and desktop deployments. Our "Packaging for MSIX" training classes are sought after by IT Professional Desktop Engineers around the world.

TMurgent Technologies, LLP is an independent company engaged in the application packaging space. TMurgent primarily provides training to IT Professionals that are involved in Desktop Engineering and Application Packaging, but we also provide some free and licensed software products in this space, including TMEditX, the software used to produce the better results shown in this report testing.

Those interested in MSIX are encouraged to download one of our two community eBooks on MSIX. Each is over 200 pages and co-authored with leading community technologists for a thorough coverage:

- **For the IT Pro:** MSIX Packaging Fundamentals (tmurgent.com)
- **For the Developer:** A Developer's Guide To MSIX (tmurgent.com)

As an independent contractor, TMurgent may relationships with several of the vendors, some of which provide support. Despite this, we believe that the information in this report does provide a fair and independent view that represents the state of the industry currently.

This report contains information gained from personal experiences and may not represent the best that can be said about the vendors and products mentioned. Omissions and mistakes are my own, but they are "honest" mistakes and not intended to malign. The Vendors have not been given an opportunity to review or correct potential factual errors in this report prior to publishing, however they may contact me if they feel there are errors in the report that should be addressed.