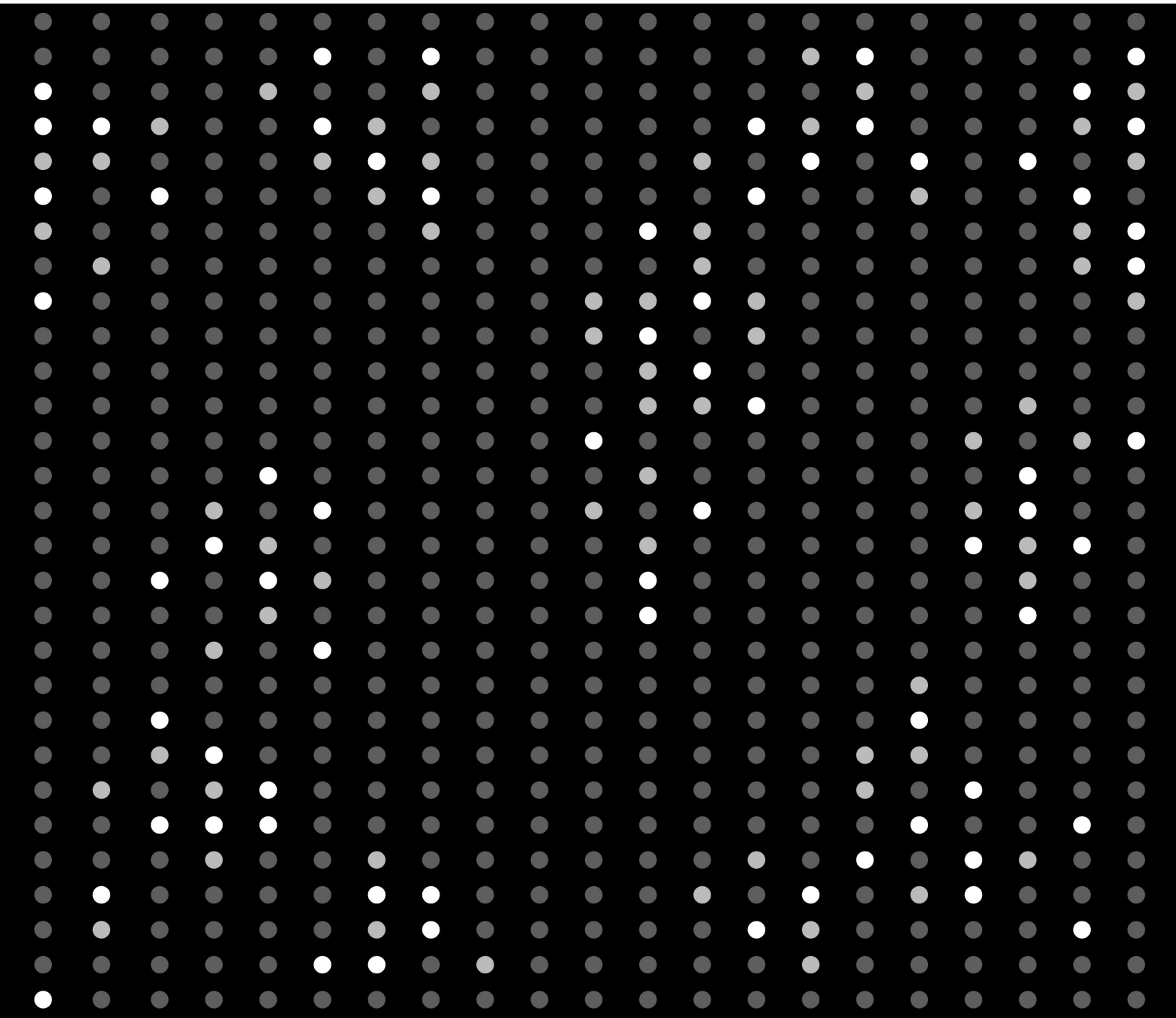
Azure e-book series

Microsoft Azure

Azure Virtual Desktop Handbook:
Application Management



Introduction

Azure Virtual Desktop is a flexible virtual desktop infrastructure (VDI) platform running on Microsoft Azure that helps enable a secure remote desktop experience from virtually anywhere through desktop and application virtualization. As you progress on your journey of enabling remote work for your organization with Azure Virtual Desktop, it is important to understand application management, its capabilities, and best practices so as to simplify management and provide a great user experience.

After reading this handbook, you should be prepared to embark on your Azure Virtual Desktop application delivery journey. If you have any questions about technical requirements or want to get advice on short- and long-term solutions for enabling remote work, you can talk to an Azure sales specialist or follow any of the links throughout this handbook for further information.

Azure Virtual Desktop overview

Azure Virtual Desktop helps organizations strengthen business resilience and modernize their desktop and app virtualization by delivering benefits including simplified management, Windows 10 multi-session, optimizations for Microsoft 365 Apps for enterprise. Azure Virtual Desktop also allows you to deploy and scale your Windows desktops and applications on Azure in minutes and provides built-in security and compliance features to help you keep your applications and data secure.

Azure Virtual Desktop is a flexible cloud VDI platform, so many infrastructure-related parts of the solution are managed for you by Microsoft. Other parts, mainly relating to desktop and application workloads, are managed by you or partner. In this handbook, we will be looking at the specifics of application management and the capabilities available in Azure Virtual Desktop.

Figure 1 shows the components joined in four different buckets. The **Azure Virtual Desktop service** and **Azure infrastructure** buckets are managed by Microsoft. The **Desktop and remote apps** and **Management and policies** buckets are managed by you. This provides you with the full flexibility of being in control of your session host servers and application landscapes:

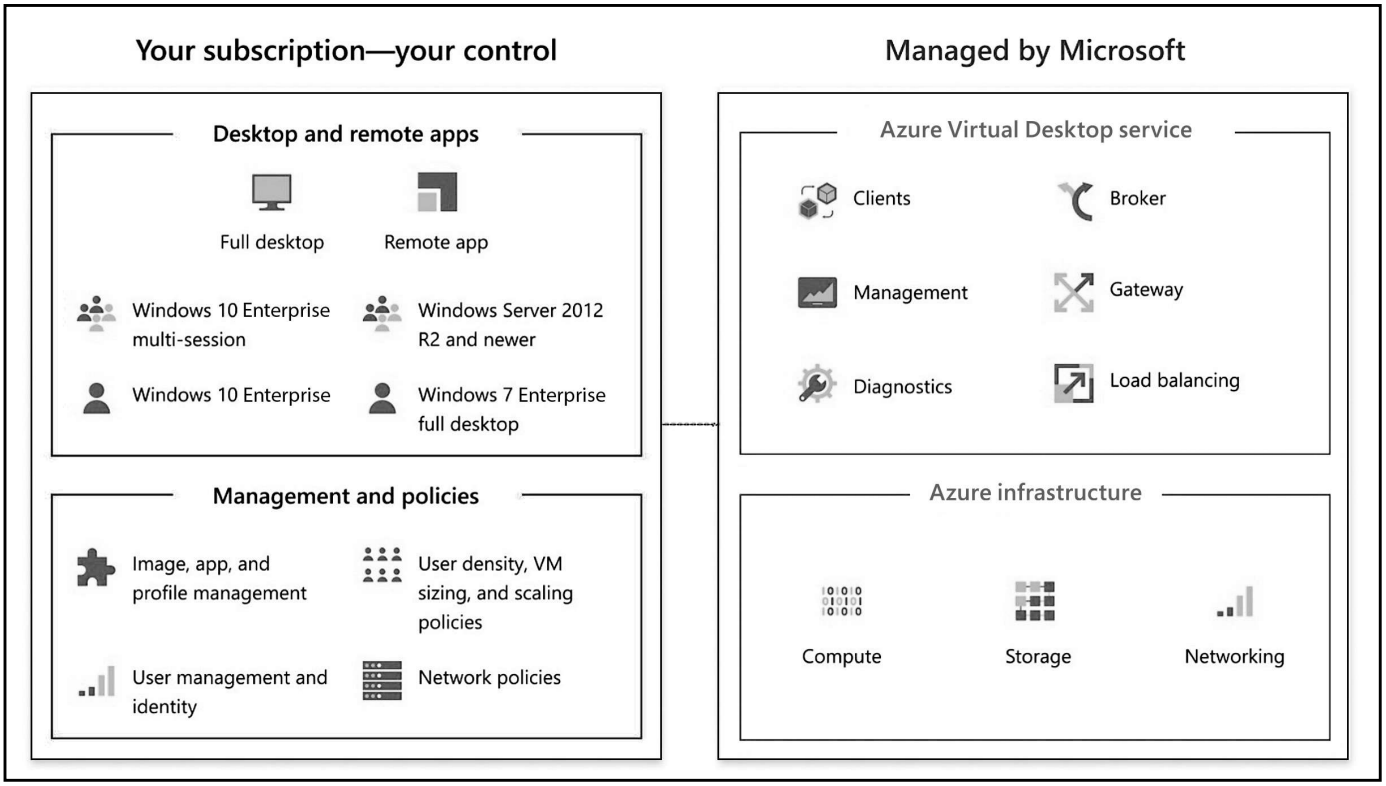


Figure 1: Azure Virtual Desktop components and responsibilities

Figure 2 shows a typical architectural setup of an enterprise environment of Azure Virtual Desktop:

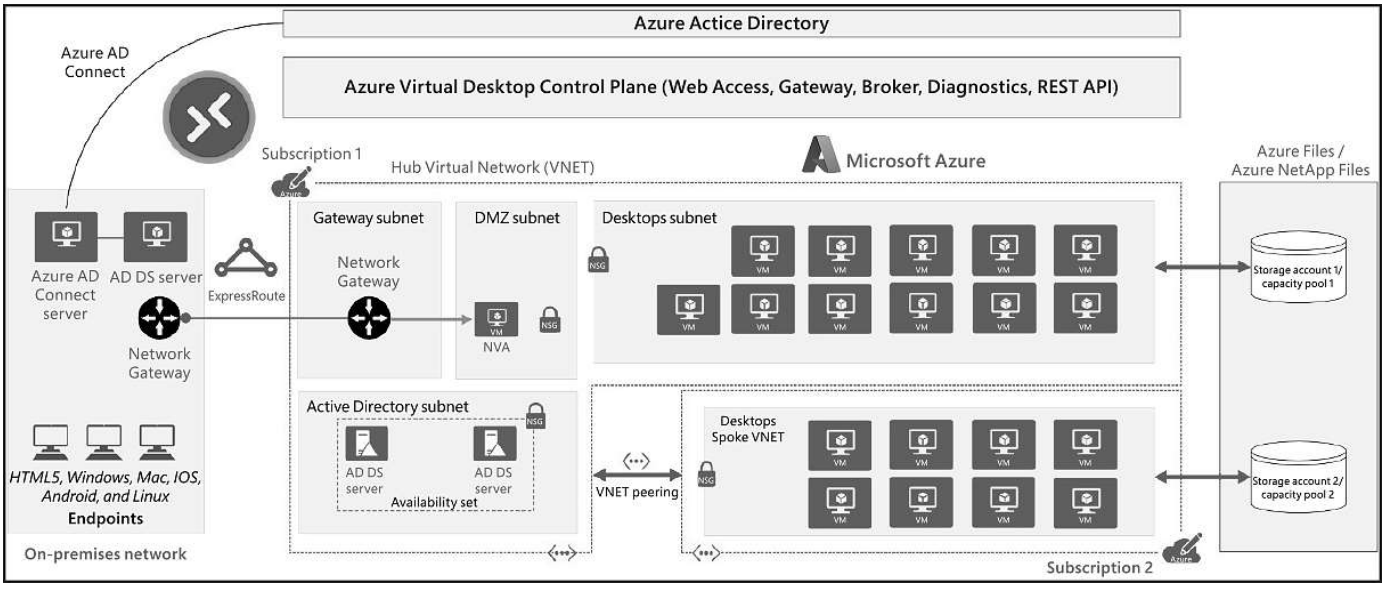


Figure 2: Typical Azure Virtual Desktop architectural setup

The application's back-end components are on the customer's on-premises network. ExpressRoute extends the on-premises network into Azure. Optionally, the back-end components can also be migrated to Azure based on a datacenter migration scenario. The Azure AD Connect components synchronize identities from Active Directory Domain Services (AD DS) with Azure AD. If Azure Active Directory Domain Services (Azure AD DS) is used, identities will automatically be synchronized from Azure AD to Azure AD DS. The customer manages AD DS and Azure AD, Azure subscriptions, virtual networks, Azure Files or Azure NetApp Files, and Azure Virtual Desktop hosts pools and workspaces.

The Azure Virtual Desktop service architecture is similar to that of Windows Server RDS. However, with Azure Virtual Desktop, Microsoft manages the infrastructure and brokering components, while enterprise customers manage their own desktop host virtual machines (VMs), data, and clients. This allows you to shift your focus to what's really important to you—the user experience. To understand the differences between RDS on-premises, migrating to Azure, and migrating to Azure Virtual Desktop, take a look at *Table 1*:

Responsibility	RDS on-premises	RDS on Azure	Azure Virtual Desktop
Identity			
End user devices (mobile and PCs)			
Application security			
Session host operating			
Deployment configuration			
Network controls			
Virtualisation control plane			
Physical hosts			
Physical network			
Physical datacenter			
			Customer / Microsoft

Table 1: Responsibilities for different cloud models

For more information on Azure Virtual Desktop for enterprises, visit this page.

Application management

Application management is the process of managing the maintenance and upgrading of software over its lifecycle. Application management in Azure Virtual Desktop provides you with a suite of management tools that let you publish, push, configure, monitor, and update applications for your users. As an organization, it's important to understand what capabilities are available and to choose the right application delivery method for your business.

Traditionally, the delivery of applications in a non-persistent desktop virtualization environment is achieved by installing applications in an operating system (OS) image. Larger organizations may break out to multiple images to segregate applications for specific departments or groups, but managing multiple images can impact an organization's ability to keep on top of OS and application patches/updates. There is also a time constraint element here for the IT admins to carry out such work. Depending on the number of images and the complexity of the applications, this could be time-consuming, resulting in a higher total cost of ownership.

Figure 3 depicts an organization using multiple images within a desktop virtualization environment to deliver group/departmental applications:

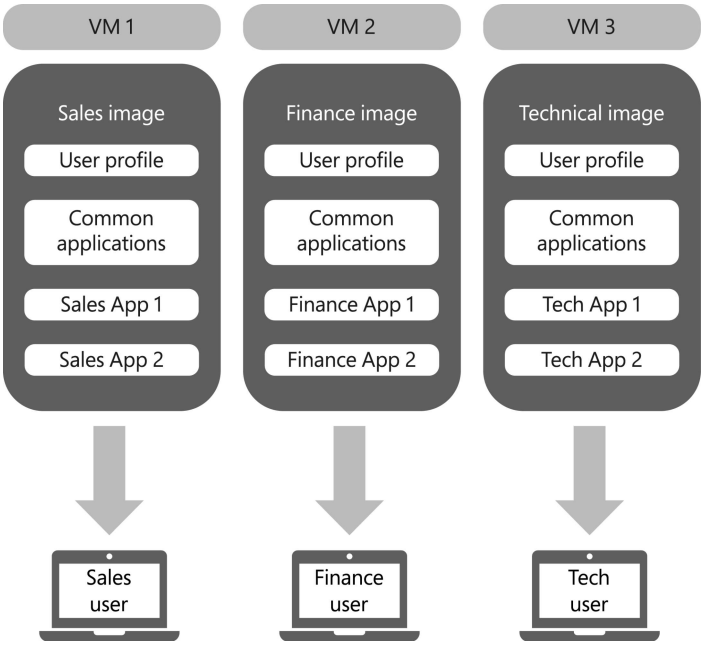


Figure 3: Using multiple images within a desktop virtualization environment

Figure 4 shows each image for a specific group/department with the same common applications installed on each image. A common application could be considered an application that all users require, such as Microsoft Teams or Microsoft Office. A group or departmental application is regarded as an app required by a select few users, such as CAD or accounting software.

The ability to centralize the delivery of applications to your users and control access to the required applications provides a seamless experience for both end-users and IT admins. The added value of being able to dynamically attach applications to a user's session provides a great experience for users, and with Azure Virtual Desktop, you can deliver applications to the user within an active session and on session login.

MSIX is a modern application packaging format, and MSIX app attach is a delivery method you can use when implementing an Azure Virtual Desktop environment. MSIX app attach provides you with the ability to serve applications to your users dynamically, regardless of the VM they are on. You can also use existing Microsoft management tools to manage applications in your Azure Virtual Desktop environment, including Microsoft Endpoint Manager and System Center Configuration Manager.

Before continuing to the next section, let's see a comparison between traditional layering technologies and MSIX app attach:

Feature	Traditional app layering	MSIX app attach
Format	Different app layering technologies require different proprietary formats.	Works with the native MSIX packaging format.
Repackaging overhead	Proprietary formats require sequencing and repackaging per update.	Apps published as MSIX don't require repackaging. However, if the MSIX package isn't available, a repackaging overhead still applies.
Ecosystem	N/A (for example, vendors don't ship App-V).	MSIX is Microsoft's mainstream technology that key ISV partners and in-house apps are adopting. You can use MSIX on both virtual desktops and physical Windows computers.
Infrastructure	Additional infrastructure required (servers, clients, and so on).	Storage only.
Administration	Requires maintenance and updates.	Simplifies app updates.
User experience	Impacts user sign-in time. A boundary exists between the OS state, app state, and user data.	Delivered apps are indistinguishable from locally installed applications.

Table 2: Comparison between traditional layering technologies to MSIX app attach

In the next section, we'll look at the capabilities and benefits of MSIX, the application technology used in conjunction with MSIX app attach.

Note: Starting in 2022, remote app streaming is a monthly per-user access option for organizations to use Azure Virtual Desktop to deliver apps from the cloud to external (non-employee) users. You can read more about remote app streaming here.

MSIX

MSIX is a modern Windows application packaging format and development framework, and when combined with MSIX app attach, the Azure Virtual Desktop–specific service, it provides a delivery mechanism. MSIX offers a modern application packaging experience for your Windows applications, simplifying the app installation process for users and IT admins.

Applications that are packaged using MSIX run in a lightweight app container. The MSIX app process and its child processes run inside this container and are isolated using the filesystem and registry virtualization. All MSIX apps can read the global registry. An MSIX app writes to its own virtual registry and application data folder, and this data will be deleted when the app is uninstalled or reset. Other apps do not have access to the virtual registry or virtual filesystem of an MSIX app.

Note: Services work slightly different to common applications where the service runs outside the container. If an application has a service, the service does not run in the container.

Existing applications can be repackaged or converted to MSIX packages using the MSIX packaging tool provided; the tool offers a simple user interface or command line to convert and package Windows applications into the MSIX packaging format.

Some of the benefits of MSIX include:

- Predictable and secure deployment.
- Container technology that isolates the app from the rest of the OS for security.
- Clean removal—when you remove MSIX apps, you remove all application data. No data remains in the registry or in the filesystem of the OS.

- MSIX removes 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.

With MSIX, you can package and distribute Win32 applications by using the Microsoft Store, enabling you to provide your users with a self-service offering through the Microsoft business store.

Let's now look at the inner workings of MSIX and some of the key components.

What's inside an MSIX package?

As mentioned in the previous section, applications prepared in MSIX format run in a lightweight container. An MSIX app writes to its own virtual registry and application data folder, and all MSIX app processes run inside that container. Applications packaged in MSIX format are installed in the "**c:\Program Files\WindowsApps**" folder. Each package folder contains a set of standardized files required for the MSIX package to function.

Figure 4 shows what the contents look like inside an MSIX package:

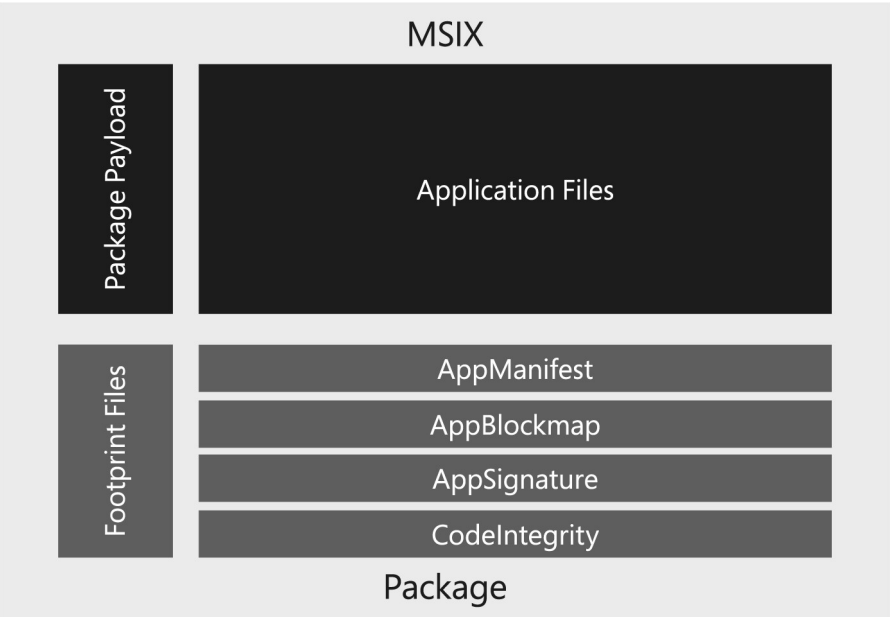


Figure 4: The MSIX package

Table 3 summarizes the core contents of an MSIX package (standardized files):

File	Description
App payload	Contains the app code files and assets
AppxBlockMap.xml	Contains a verified and secure list of all the files within the package
AppxManifest.xml	Drives the installation by configuring association with files and contains the identity of the package and their dependencies
AppxSignature.p7x	Contains the signature of the package that the OS must trust before the app is installed

Table 3: A summary of the core contents of an MSIX package

We will now look at MSIX app attach and how you can create a dynamic application delivery solution for Azure Virtual Desktop.

MSIX app attach

MSIX app attach is a way to deliver MSIX applications to VMs in Azure Virtual Desktop. Azure Virtual Desktop has built-in capabilities, ready for you to start managing and delivering MSIX app attach to users across Azure regions today.

MSIX app attach uses the MSIX lightweight container to isolate the application data from the user data and the OS. By leveraging the MSIX packaging format, MSIX app attach removes the need for repackaging.

One of the benefits of using MSIX app attach over other application delivery technologies is that it has been designed to deliver applications in a way that does not delay user login times. It also significantly reduces the management that IT needs to perform in deploying and managing silos of VMs that host sets of applications for different sets of user groups within an organization. This allows organizations to modernize their virtual desktop estate and move from managing pet-like VMs that are unique and dedicated to specific users to cattle VMs that are deployed once and are the same for all, as the applications are dynamically delivered to any VM that a user may log in to. This will save on complexity and, hence, time and costs. Some of the other key benefits of using MSIX app attach are shown in *Figure 5*:

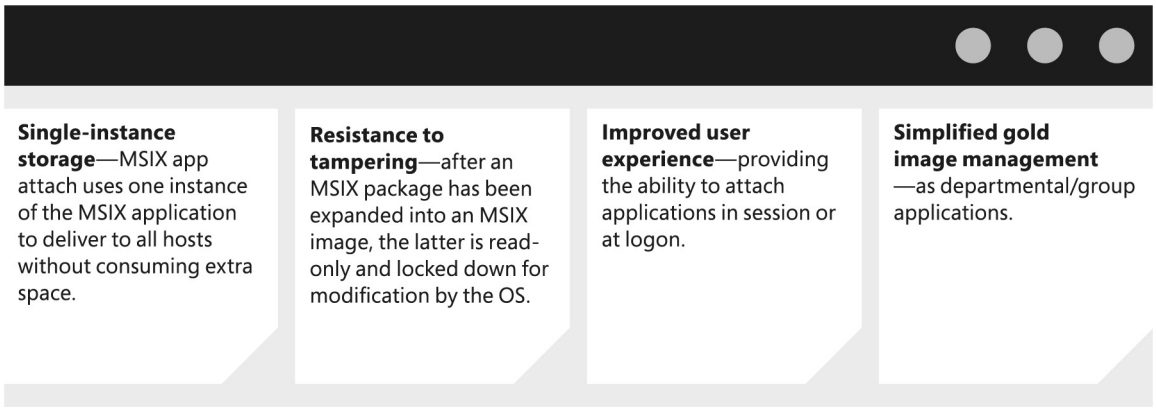


Figure 5: MSIX app attach benefits

It is important to note that MSIX app attach requires that an MSIX package is expanded into an MSIX image (a virtual disk). Expanding is the process of taking the MSIX package, unzipping it, and applying the appropriate system permissions to the file structure inside the chosen virtual disk or Composite Image File System (CimFS) image.

Figure 6 illustrates the use of both FSLogix Profile containers and MSIX app attach in Azure Virtual Desktop:

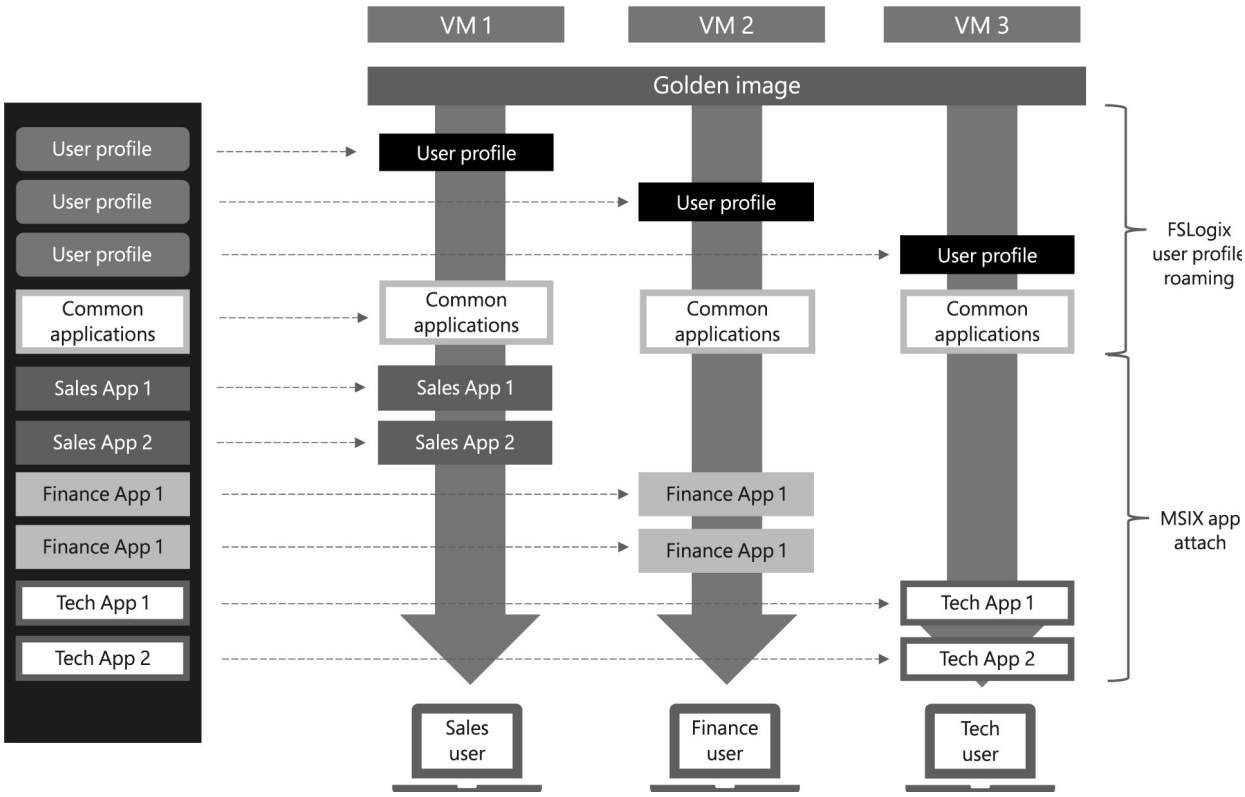


Figure 6: Using FSLogix Profile containers and MSIX app attach

As you can see from *Figure 6*, common applications are installed within the OS image, the user profiles are attached on login, and departmental/group applications are dynamically delivered to the user on login.

Note: The word "dynamically" refers to the process of attaching a virtual disk or CimFS image, rather than installing the app natively.

MSIX app attach process terminology

Table 4 details the different terms and associated processes used when using MSIX app attach with Azure Virtual Desktop:

Term	Definition
Stage	Azure Virtual Desktop notifies the OS that an application is available and that the virtual disk that contains the MSIX package (also known as the MSIX image) is mounted.
Registration	MSIX app attach uses a per-user process to make the application available to you.
Delayed or deferred registration	Complete registration of the application is delayed until the user decides to run the application.
Deregistration	The application is no longer available to you after you sign out.
Destage	The application is no longer available from the VM following the shutting down or restarting of the machine.

Table 4: Terms related to MSIX app attach process

In the next section, we will look at how MSIX app attach works using Azure Virtual Desktop and how you can take advantage of the modern application delivery capabilities available today.

How MSIX app attach works on Azure Virtual Desktop

This section will cover the process of how apps are delivered to your user session in Azure Virtual Desktop using MSIX app attach. It separates applications from VMs and their operating systems and stores these applications separately in virtual hard disks. These are stored separately in storage. However, it does not depend on an agent, like FSLogix does. The support for MSIX app attach is built into Windows, so there is no need for any additional software to enable app attach. APIs provide a much faster and improved user experience than agent-based application delivery technologies.

Figure 7 details the five steps of delivering applications in Azure Virtual Desktop using MSIX app attach:

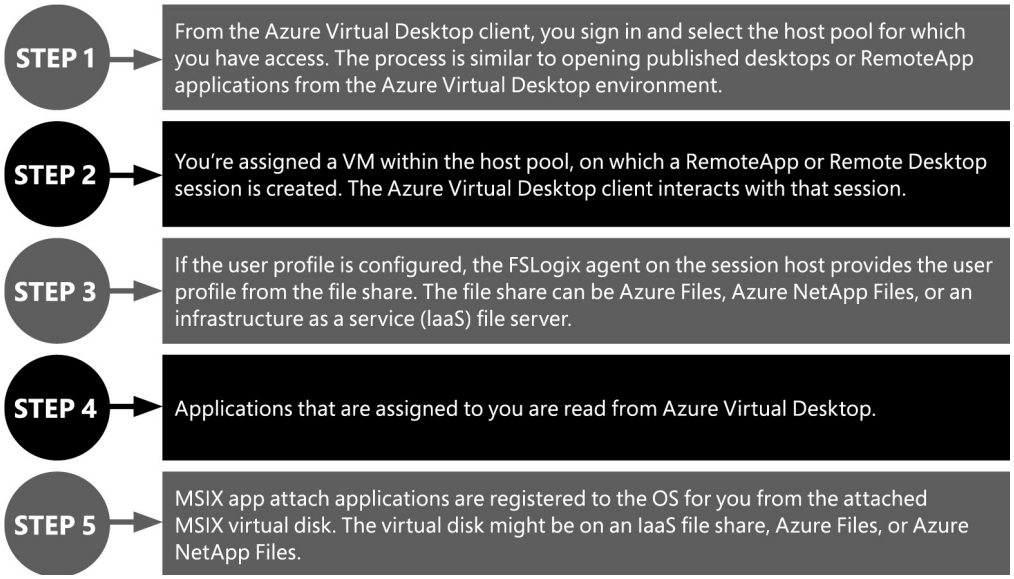


Figure 7: Application delivery steps in Azure Virtual Desktop using MSIX app attach

We have also included the process diagram below (*Figure 8*), which highlights the five key steps detailed in *Figure 7*:

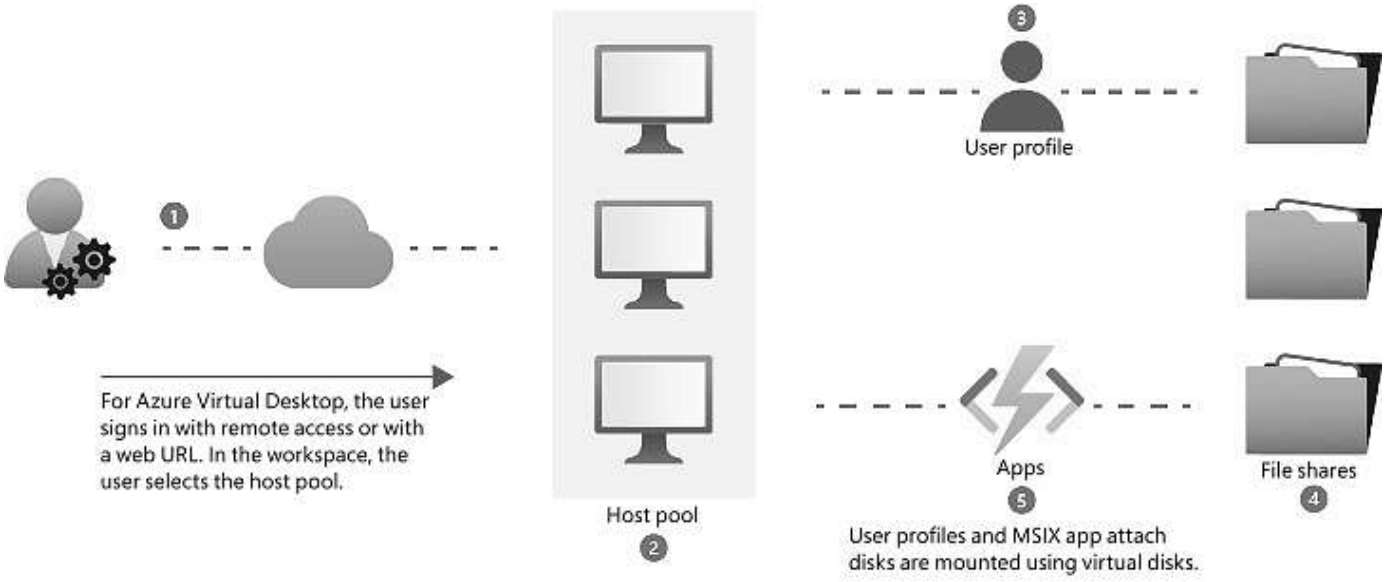


Figure 8: Application delivery process in Azure Virtual Desktop using MSIX app attach

That concludes this introduction to application management, MSIX, and MSIX app attach. In the next chapter, we will look at the prerequisites for getting started with MSIX app attach.

Prerequisites

In this chapter, we will look at all the prerequisites that you need to complete and be familiar with in order to be able to use MSIX app attach within Azure Virtual Desktop.

Figure 9 summarizes the components and actions required for MSIX app attach:

Requirements	Actions
• Azure Virtual Desktop host pool with at least one active session host • MSIX packaging tool • MSIXMGR tool for expanding MSIX-packaged applications • MSIX-packaged application expanded into an MSIX image that's uploaded into a file share • Designated file share in your Azure Virtual Desktop deployment where the MSIX package will be stored	• Check that the file share where you uploaded the MSIX image is accessible to all virtual machines (VMs) in the host pool. Users will need read-only permissions to access the image • Check the certification—If the certificate isn't publicly trusted, follow the instructions in Install certificates

Figure 9: A summary of components and actions required for MSIX app attach

Note: You will require a code signing certificate for MSIX app attach. This can either be a public certificate or a self-signed certificate. Find out how to create a self-signed certificate here.

The prerequisite steps for preparing for MSIX app attach are detailed in the *Table 5*:

Step	Description
Create an MSIX image.	Create an MSIX package containing the application you want to use within your Azure Virtual Desktop deployment.
Create an MSIX Image.	Create an MSIX image using the MSIX package you created in the previous step.
Configure Azure Files for MSIX app attach.	Create and configure Azure Files, including the required permissions or other file share.
Upload the MSIX images.	Upload your prepared MSIX images.
Self-signed certificate install (optional).	Install the self-signed certificate on all the session hosts in use for MSIX app attach.

Table 5: Prerequisite steps for preparing for MSIX app attach

We will now progress with our journey through the five key steps of preparing for MSIX app attach.

Step 1—MSIX package creation

In this section, we will show you how to create an MSIX package. You can also download MSIX packages from the Microsoft business store; please see the following guide.

Before getting started, it's important to note that with MSIX app attach, applications do not support application updates. You will need to disable automatic updates before continuing with packaging, otherwise the application will be rendered unusable for the end user. Read more about disabling auto-updates here.

To get started with creating an MSIX package, you first need to download the MSIX packaging tool or use the Hyper-V quick start, to deploy a Windows 10 VM with the MSIX packaging tool pre-installed.

Figure 10 shows the UI interface of the MSIX packaging tool:

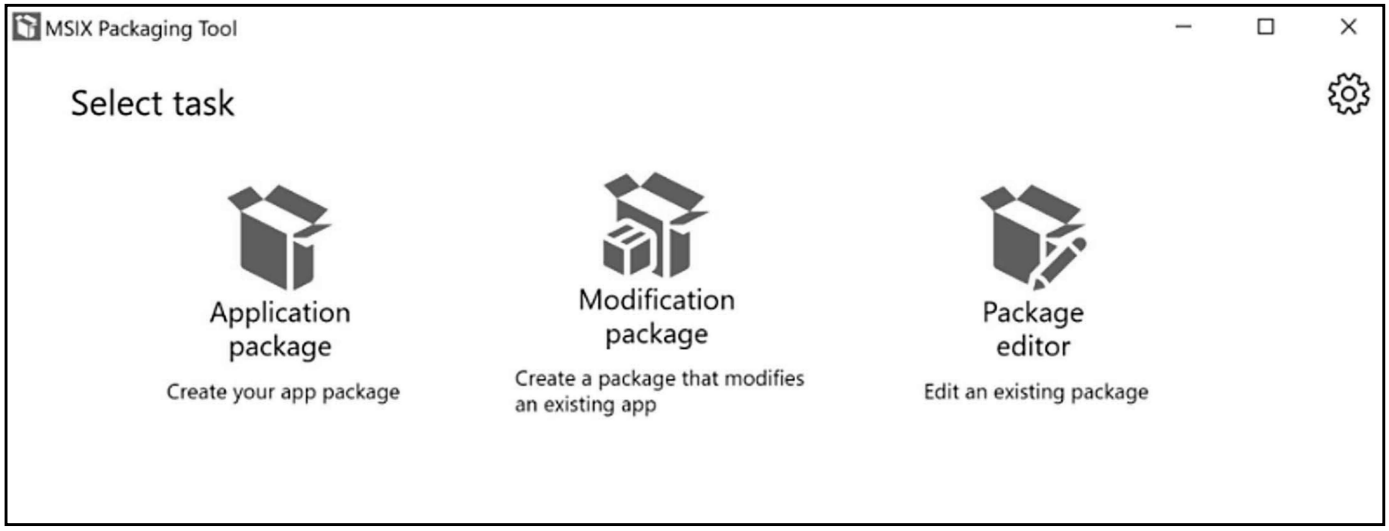


Figure 10: The MSIX Packaging Tool UI interface

To create an MSIX package, please see the Microsoft Guide here. You can also use this video tutorial to learn more about creating an MSIX package.

Once you have created your first MSIX package, you can progress to *step 2* to create the MSIX image.

Step 2—Creating an MSIX image

The medium used to deliver MSIX packages using MSIX app attach is called an MSIX image. This can be a VHD, VHDX, or CIM image.

Figure 11 depicts the structure of an MSIX image. As you can see from the key elements, the three layers consist of the virtual disk/CIM image, the MSIX package, and the MSIX application, which resides inside the MSIX package:

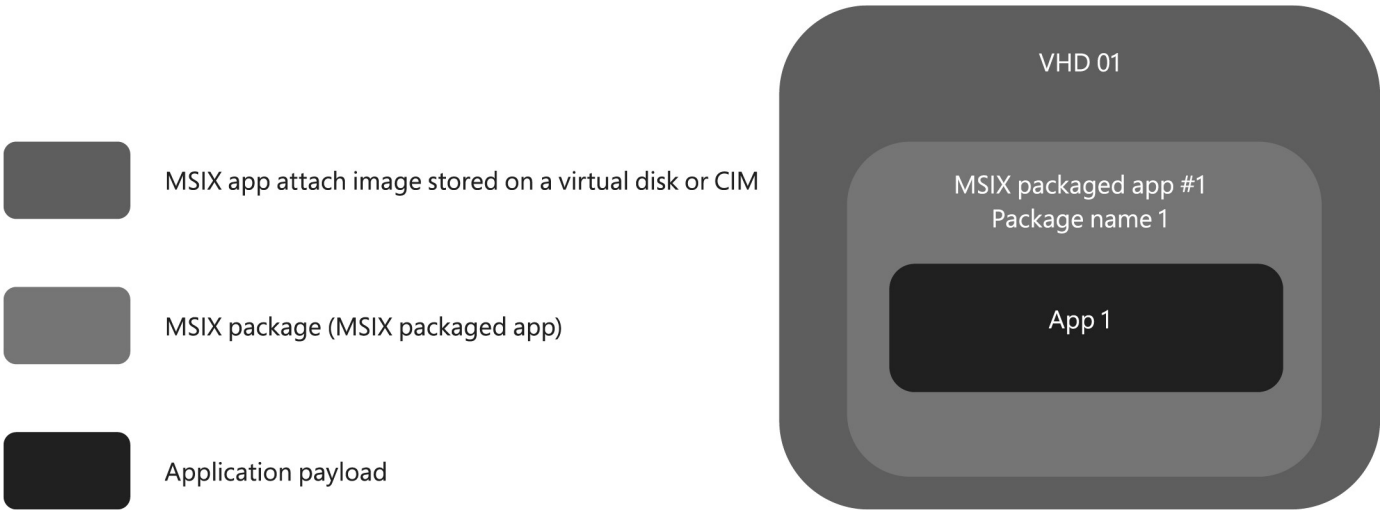


Figure 11: A structural illustration of an MSIX image

To create an MSIX package, you will need to download the MSIXMGR tool used for creating MSIX images. The process is also referred to as "Expand an MSIX file":

1. Download the MSIXMGR tool if you haven't already done so.
2. Unzip MSIXMGR.zip into a local folder.
3. Open a command prompt in elevated mode.
4. Find the local folder from *step 2*.
5. Run the following command in the command prompt to create an MSIX image:

```
msixmgr.exe -Unpack -packagePath <path to package> -destination <output folder> [-applyacls] [-create]
[-vhdSize <size in MB>] [-filetype <CIM | VHD | VHDX>] [-rootDirectory <rootDirectory>]
```

Remember to replace the placeholder values with the relevant values:

```
msixmgr.exe -Unpack -packagePath "C:\apps\notepadpp_1.0.0.0_x64__ekey3h7rct2nj.msix" -destination "C:\
apps\testapp.vhdx" -applyacls -create -vhdSize 1024 -filetype VHDX -rootDirectory MSIX
```

The preceding command should give you an output similar to this:

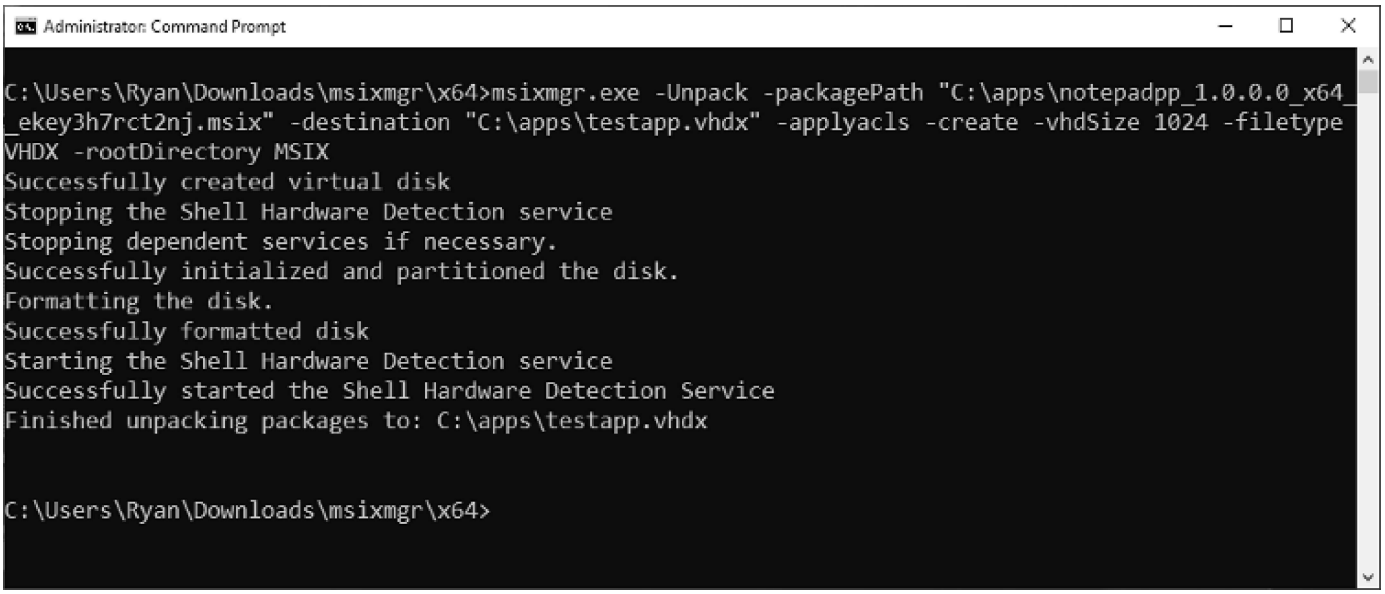


Figure 12: Creating an MSIX image in the command prompt

Once you have created your MSIX image, you are then ready to progress to the next step of configuring Azure Files for MSIX app attach.

For more information on using the MSIXMGR tool, see the following link.

Step 3—Configuring Azure Files

In this section, we discuss the use of Azure Files for MSIX app attach. It's important to note that you can use other file shares to deliver MSIX app attach in Azure Virtual Desktop.

The setup process for MSIX app attach file shares is largely the same as the setup process for FSLogix profile file shares. However, you'll need to assign users different permissions and additionally, you'll need to assign the session host VMs computer accounts to access the MSIX share. MSIX app attach requires read-only permissions to access the file share.

Note: Make sure the storage type used has a latency of less than 400 ms.

Regardless of the storage type, you'll need to assign all session host VMs with both storage account role-based access control (RBAC) and New Technology File System (NTFS) permissions on the file share.

When creating and preparing an Azure file share for MSIX app attach, there are nine steps:

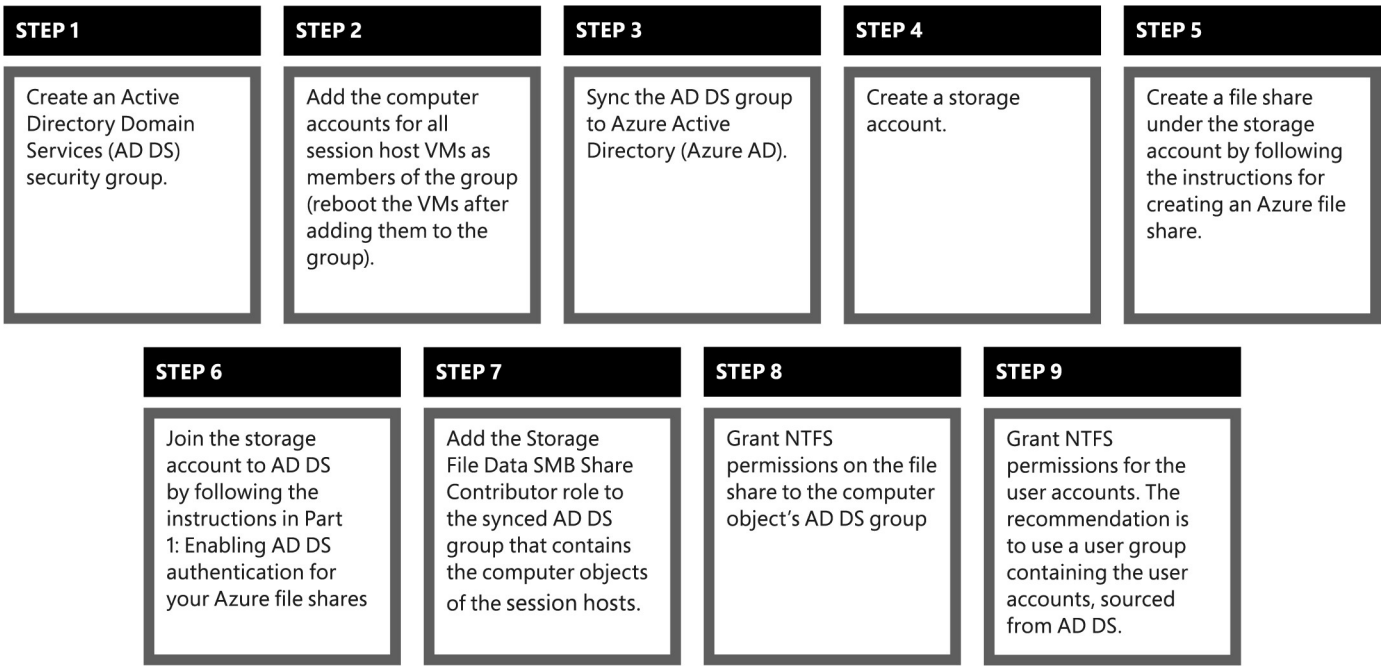


Figure 13: Steps for creating and preparing an Azure file share for MSIX app attach

Find out the specific instructions for setting up an Azure Files share or configuring Azure NetApp Files for MSIX app attach.

Now that you have your Azure file share set up and configured, we now need to upload the MSIX images we created in *step 2* into the Azure Files share.

Step 4—Uploading your MSIX image to Azure Files

When it comes to uploading files to Azure Files, you have several options. You can do this by connecting to the SMB share directly on your network, which is a common method for application packaging, or you can use some of the following tools.

AzCopy: A command-line transfer tool. To learn more about AzCopy, please refer to the following guide.

Storage Explorer: A standalone app that makes the work of uploading files to Azure Files easy. To get started with Storage Explorer, please see the following guide.

Azure portal: Within the Azure portal using a web browser, you can upload files using the upload function shown in *Figure 14*:

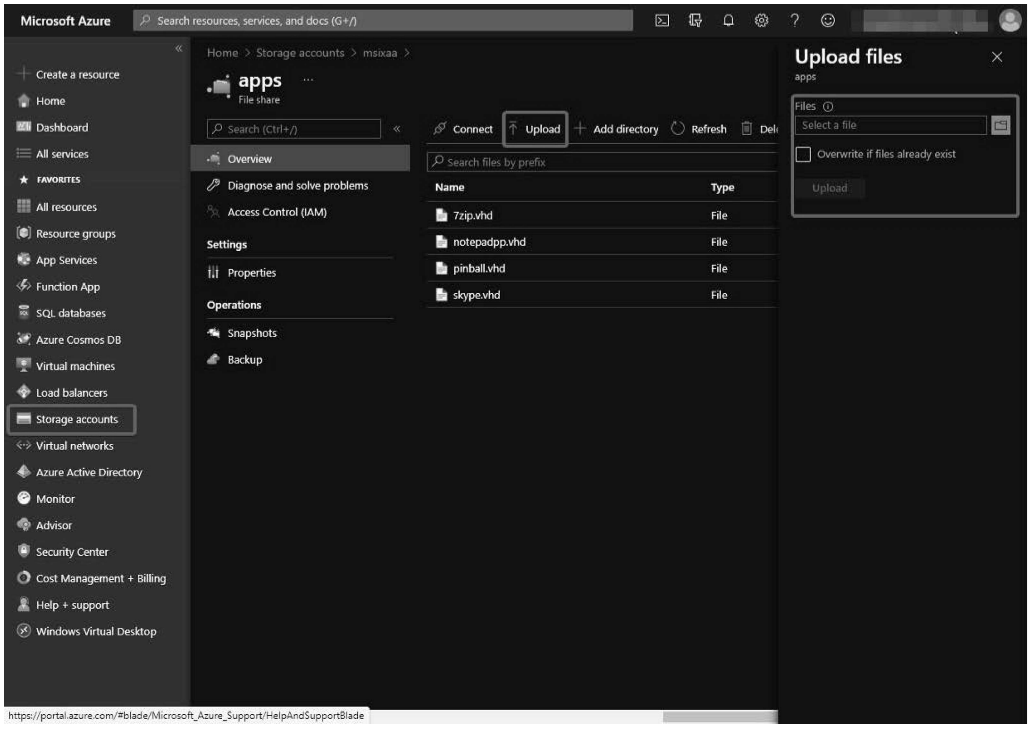


Figure 14: Uploading files using the Azure portal

The next and final step covers the crucial part of installing certificates onto the session hosts for MSIX app attach to function.

Step 5—Installing certificates

Before we can start using MSIX images, the final step would be to install the package certificate to all the required session hosts.

There are three code signing certificate types you can install:

- Self-signed certificate
- Public code signing certificate
- Internal Certificate Authority certificates

In this handbook, we show you how to generate a self-signed certificate using the following PowerShell cmdlets:

```
New-SelfSignedCertificate -Type Custom -Subject “CN=RMTIBLOG, O=Ryanmangansitblog, C=GB” -KeyUsage
DigitalSignature -FriendlyName “Your friendly name goes here” -CertStoreLocation “cert:\CurrentUser\
my” -TextExtension @(“2.5.29.37={text}1.3.6.1.5.5.7.3.3”, “2.5.29.19={text}”)
```

You can also convert a CER certificate to a PFX by using the following PowerShell cmdlets:

```
$password = ConvertTo-SecureString -String <Your Password> -Force -AsPlainText Export-PfxCertificate
-cert “Cert:\CurrentUser\My\<Certificate Thumbprint>” -FilePath <FilePath>.pfx -Password $password
```

Once you have generated the self-signed certificate, you need to install it on the required session hosts. You can do this manually or using custom script extensions; read more here.

To install a certificate, you would open the Certificate MMC Snap-in for the local computer.

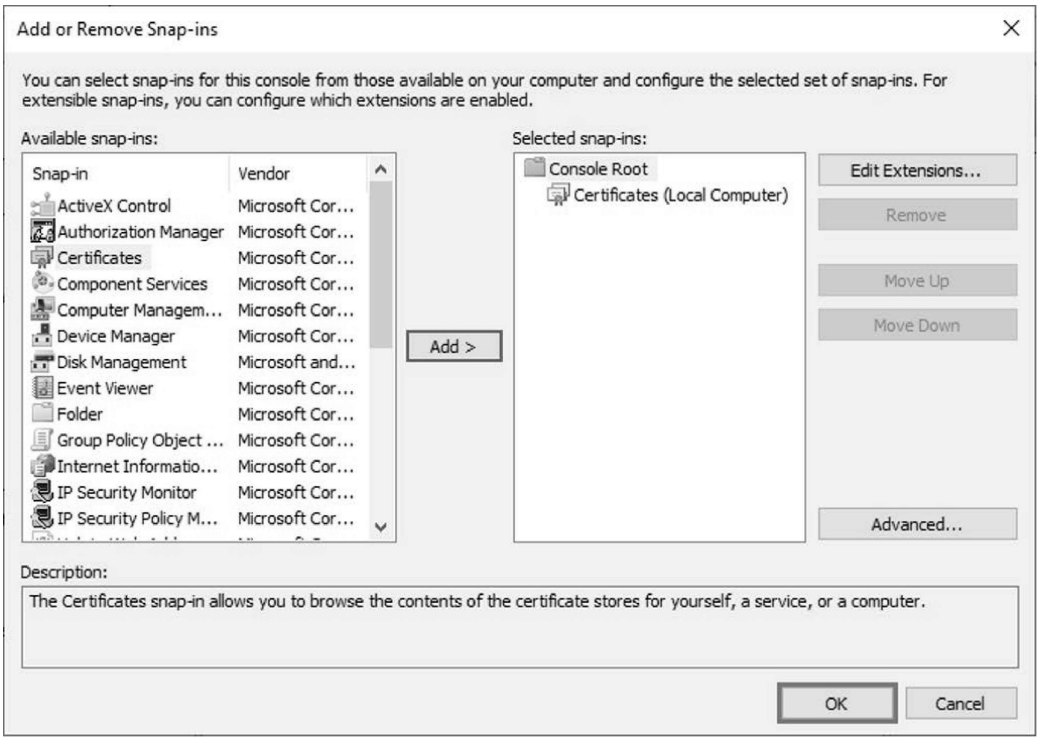


Figure 15: Opening the Certificate MMC Snap-in for the local computer

Once you have loaded the local computer Certificates snap-in, you need to import the certificate into the correct store:

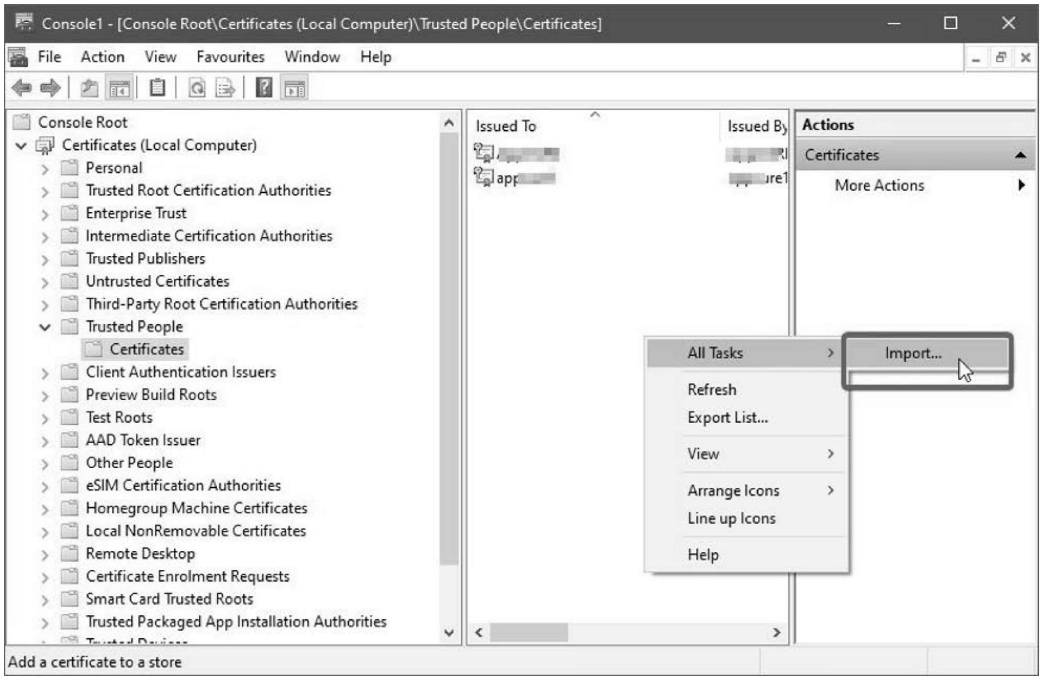


Figure 16: Importing the certificate in the correct store

Select the **Trusted People** folder and then the subfolder called **Certificates**, as shown in *Figure 16*. Right-click, select **All Tasks**, and then click **Import**.

You'll be prompted with a Certificate Import Wizard. Follow the steps as you are guided:

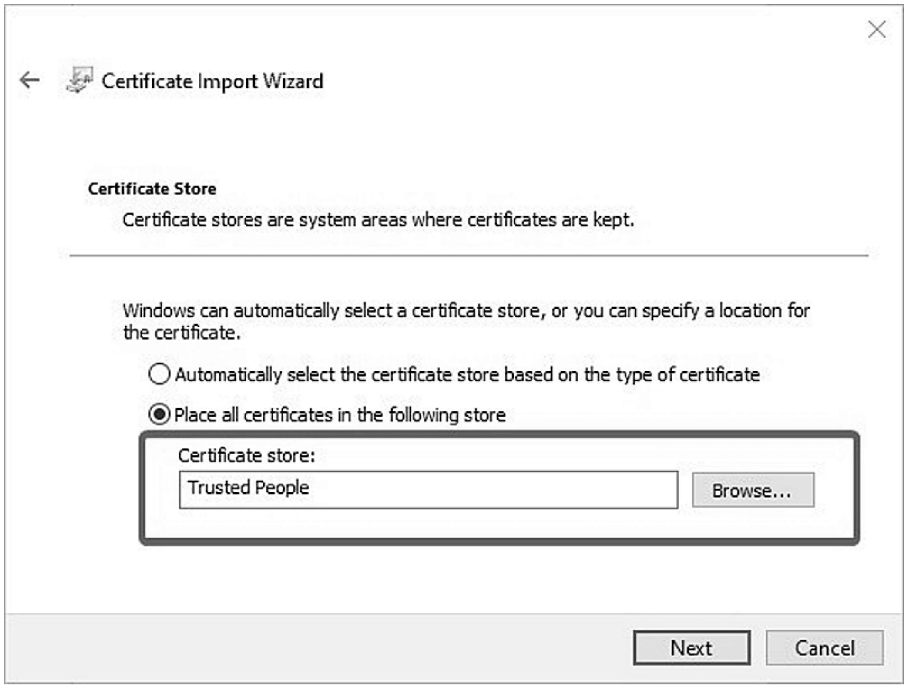


Figure 17: The Certificate Import Wizard

Ensure you place the certificate in the Trusted People Certificate Store.

The following guide, shows an alternative way to install certificates.

On completion of all the preceding steps, you will be able to use any MSIX/MSIX Image signed with the imported self-signed certificate.

Once you have all your MSIX images within the Azure Files share, you are ready to move onto the final part of the journey—deploying MSIX apps to users. In the next chapter, we'll see how to deploy MSIX apps to users through the Azure portal.

Note: Before you get started, you need to ensure that at least one VM is powered on within the target host pool before attempting to add MSIX packages.

Configuring MSIX app attach in Azure Virtual Desktop

In this chapter, we will cover the configuration of MSIX app attach within the Azure portal. We will cover the configuration and assignment to users and some of the associated settings.

Adding MSIX packages

Within the Azure portal under **Azure Virtual Desktop** > **Host Pools** > **Host Pool Name**, you will see **MSIX packages** under the **Manage** option. This is shown in *Figure 18*:

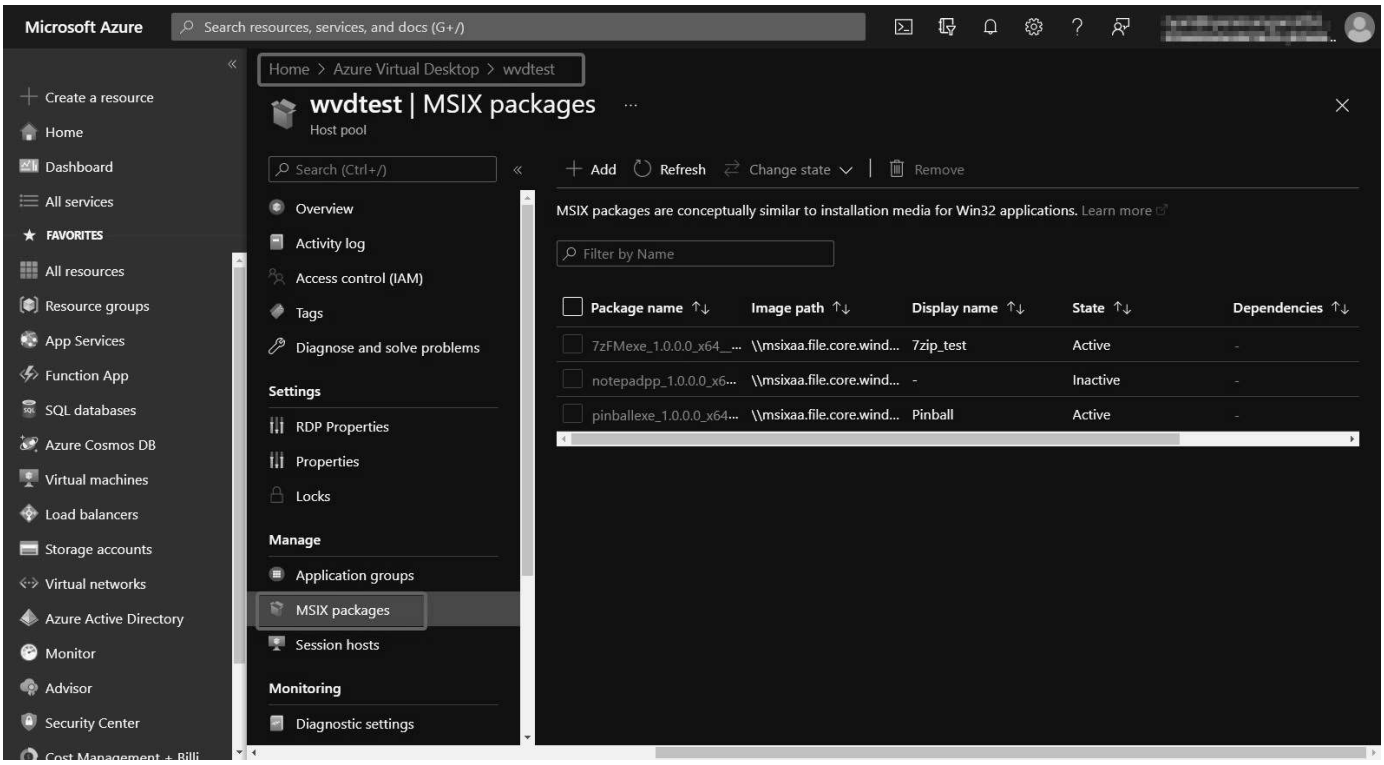


Figure 18: Adding and configuring MSIX packages for host pool

The MSIX package blade is where you add and configure MSIX packages for the specific host pool within Azure Virtual Desktop. To associate your first MSIX package, which you should have uploaded in the previous section, "Prerequisites," under *step 4*, you need to click **+ Add** within the MSIX package pane, as shown in *Figure 19*:

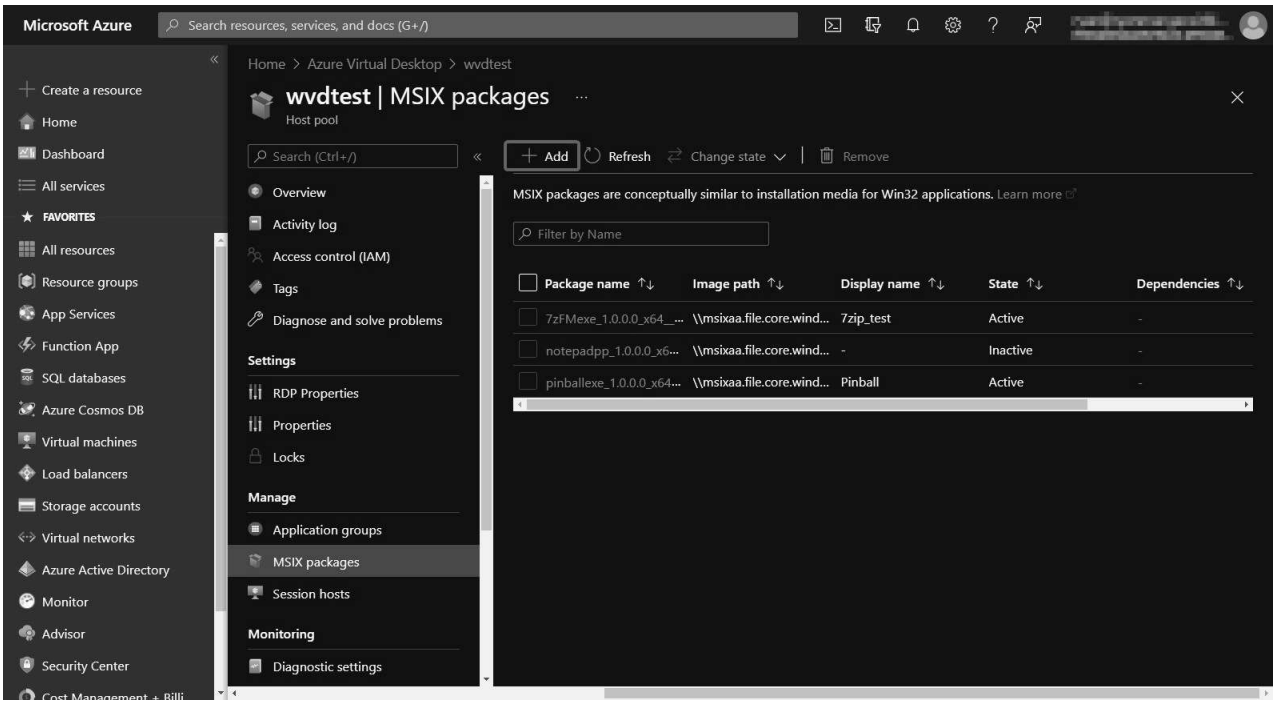


Figure 19: Adding the first MSIX package

This will present you with the **Add MSIX Package** menu, which will require you to enter the absolute path of MSIX image you want to configure for the MSIX app attach.

The following steps detail how to add MSIX packages to your host pool. In the **Add MSIX package** tab, enter the following values:

1. For **MSIX image path**, enter a valid UNC path pointing to the MSIX image on the file share. (For example, \\storageaccount.file.core.windows.net\msixshare\appfolder\MSIXimage.vhd.) When you're done, select **Add** to query the MSIX container to check whether the path is valid.
2. For **MSIX package**, select the relevant MSIX package name from the dropdown menu. This menu will only be populated if you've entered a valid image path in **MSIX image path**.
3. For **Package applications**, make sure the list contains all MSIX applications you want to be available to users in your MSIX package.

4. Optionally, enter a display name if you want your package to be more user-friendly in terms of your user deployments.
5. Make sure that **Version** has the correct version number.
6. Select the registration type you want to use. Which one you use depends on your needs:
   - **On-demand registration** postpones the full registration of the MSIX application until the user starts the application. This is the registration type we recommend you use.
   - **Log on blocking** only registers while the user is signing in. This is not a recommended type because it can lead to longer sign-in times for users.
7. For **State**, select your preferred state.
   - The **Active** status lets users interact with the package.
   - The **Inactive** status causes Azure Virtual Desktop to ignore the package and not deliver it to users.
8. When you're done, select **+ Add**.

Once you have finished adding the required packages, you will see them within the **MSIX packages** pane as shown in *Figure 20*:

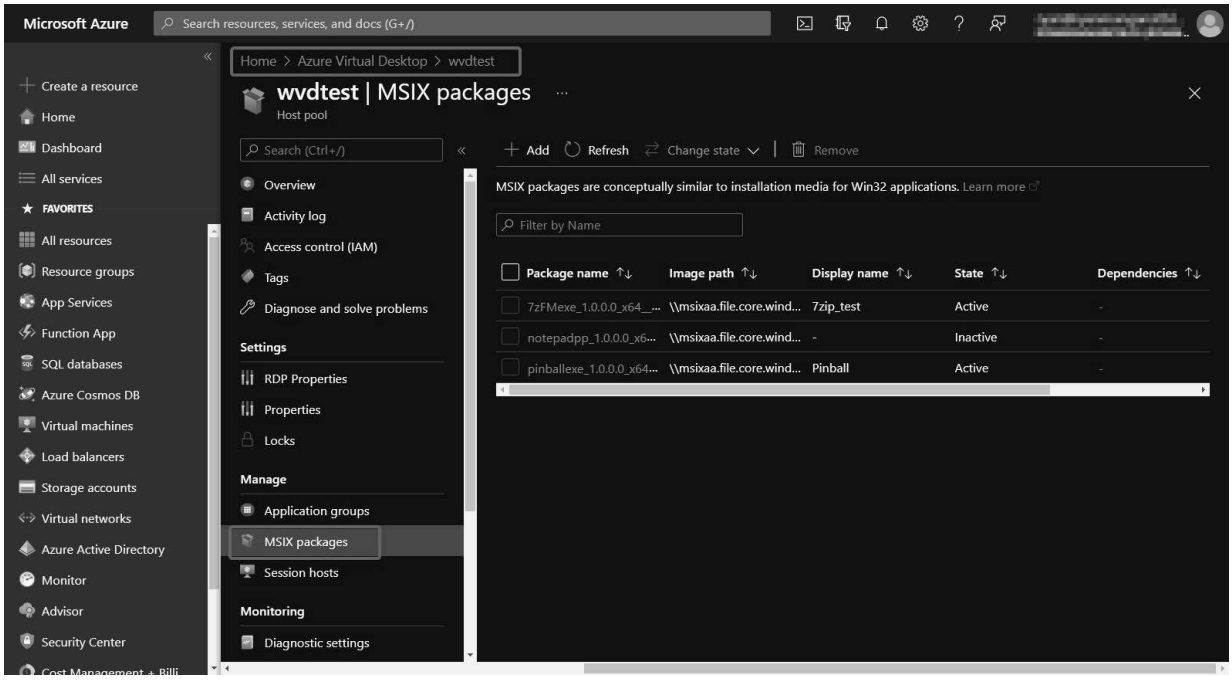


Figure 20: Package details displayed within the MSIX packages pane

In this section, we showed you how to add MSIX packages using the Azure portal. You can find out more regarding the addition of MSIX packages to a host pool by reading the following guide.

Publishing MSIX apps to an app group

In this section, we will cover the assignment of applications to users within application groups.

To publish apps to user groups, you need to complete the following steps:

1. In Azure Virtual Desktop within the Azure portal, select the **Application groups** tab.
2. Select the application group you want to publish the apps to, as shown in *Figure 21*:

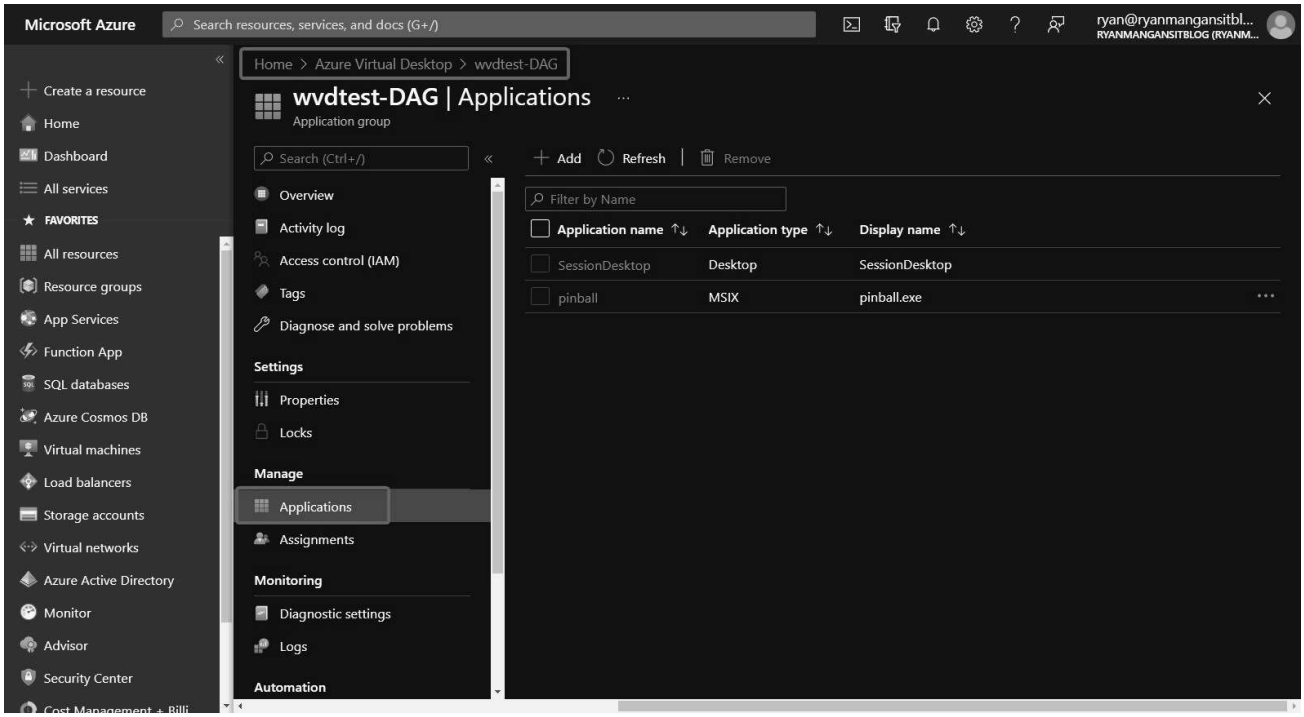


Figure 21: Selecting the application group

3. Once you're in the app group, select the **Applications** tab. The **Applications** grid will display all existing apps within the app group.

Note: MSIX applications can be delivered with MSIX app attach to both remote app and desktop app groups.

---

4. Select **+ Add** to open the **Add application** tab. *Figure 22* shows the addition of an application using a desktop application group:

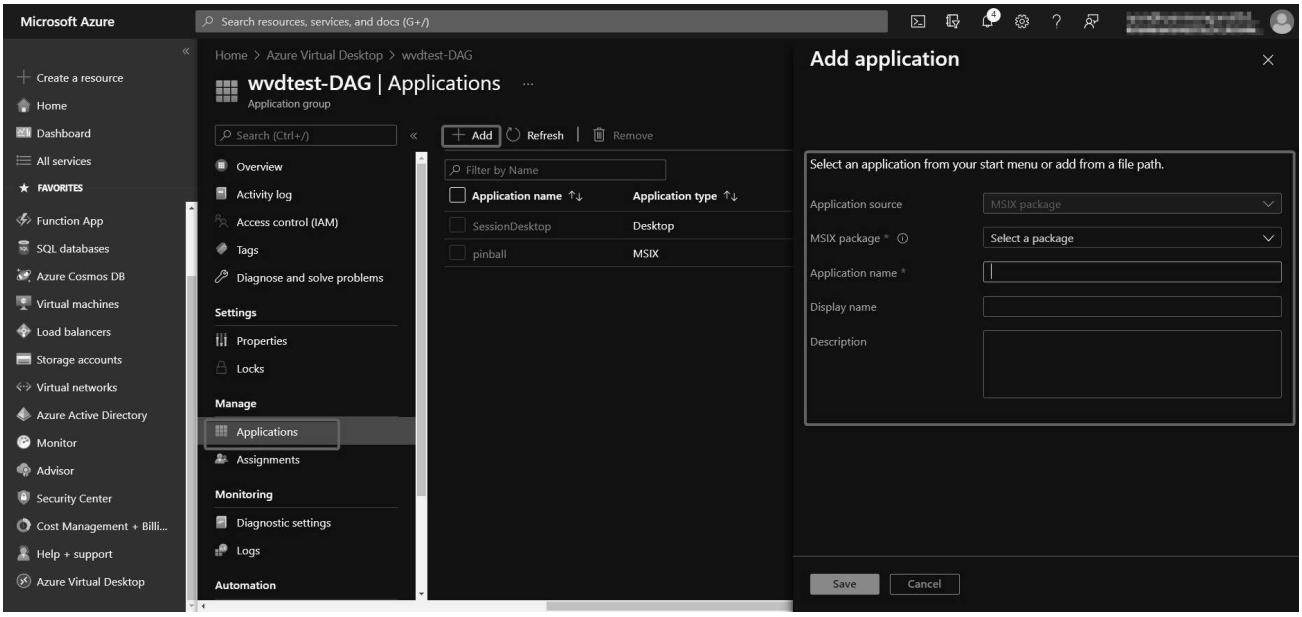


Figure 22: Adding an application using a desktop application group

5. For **Application source**, choose the source for your application:

- If you're using a Desktop app group, choose **MSIX package** (as shown in the preceding screenshot).
- If you're using a remote app group, choose one of the following options:
  i. **Start menu**
  ii. **App path**
  iii. **MSIX package**
- For **Application name**, enter a descriptive name for the application.

You can also configure the following optional features:

- For **Display name**, enter a new name for the package that your users will see.
- For **Description**, enter a short description of the app package.
- If you're using a remote app group, you can also configure these options:
  i. **Icon path**
  ii. **Icon index**
  iii. **Show in web feed**

6. When you're done, select **Save**.

When a user is assigned to a remote app group and a desktop app group from the same host pool, both Desktop and RemoteApps will be displayed in the feed.

For more information on configuring app groups with MSIX app attach, please refer to the following documentation.

You should now be able to log in to your Azure Virtual Desktop and see your applications attaching dynamically using MSIX app attach. *Figure 23* shows the application attached to the user desktop using MSIX app attach:

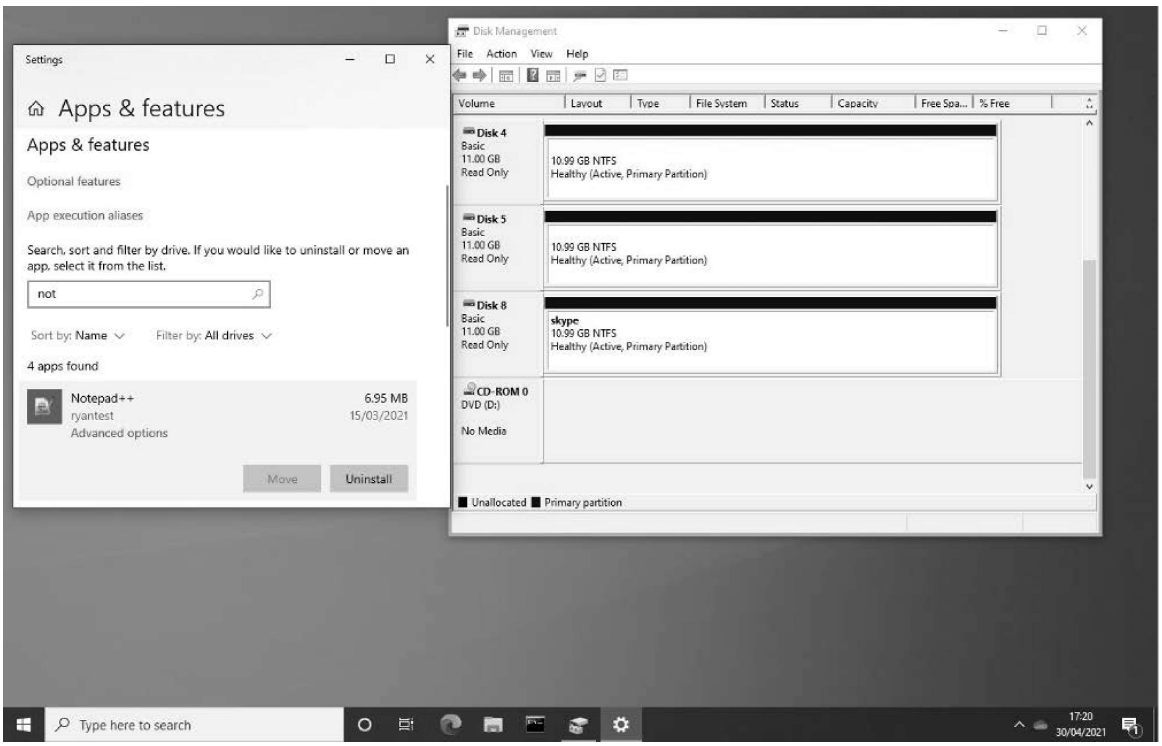


Figure 23: Application attached to the user desktop using MSIX app attach

---

Note: There is a default five-minute package update check. When publishing applications, the default time for the update will be five minutes. This can be adjusted by the IT admin. See the troubleshooting section for more information.

Now that we have finished configuring MSIX app attach, we are going to move on to the next chapter, which covers troubleshooting, where we will briefly cover some hints and tips to help you diagnose any issues you may find.

Troubleshooting MSIX app attach

Some of the typical issues associated with MSIX app attach are related to the MSIX package itself. Before moving on to the expanding phase, MSIX packages should be tested to ensure that they work correctly. One of the most common issues is where an MSIX package has not been signed with a code-signed certificate.

Common MSIX app attach challenges:

- When expanding an MSIX package, make sure you apply the ACLs, otherwise the MSIX image will not be useable. Read more here.
- Ensure that the certificate used to sign the package is installed on the target MSIX app attach session hosts.
- Ensure that the session host has the correct NTFS and RBAC permissions before adding MSIX images to the Azure portal.
- Ensure that the MSIX package itself is packaged correctly with no errors.
- When testing using PowerShell scripts, ensure that the variables are correct, including the volume GUID and package name.
- Ensure that you exclude the storage path or MSIX image extensions for antivirus to prevent any bottlenecks or poor performance.

To change the package check interval within Azure Virtual Desktop, you will need to change the following registry key value (the default is five). The check is in the order of minutes:

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\RDInfraAgent\MSIXAppAttach]
“PackageListCheckIntervalMinutes”=dword:00000001
```

You can find more information on troubleshooting MSIX app attach here.

Optimizations and best practices

In this chapter, we'll look at some of the key tools and technologies that you can use to optimize your app management process.

Microsoft Endpoint Manager

You can now manage your Azure Virtual Desktop machines using Microsoft Endpoint Manager. When enrolling Azure Virtual Desktop VMs that are Hybrid domain joined, you can manage these through the Endpoint Manager admin center the same way you would manage a physical device.

This includes the ability to deploy applications and configure policies and compliance.

Figure 24 shows a sample of Azure Virtual Desktop session hosts registered within Endpoint Manager:

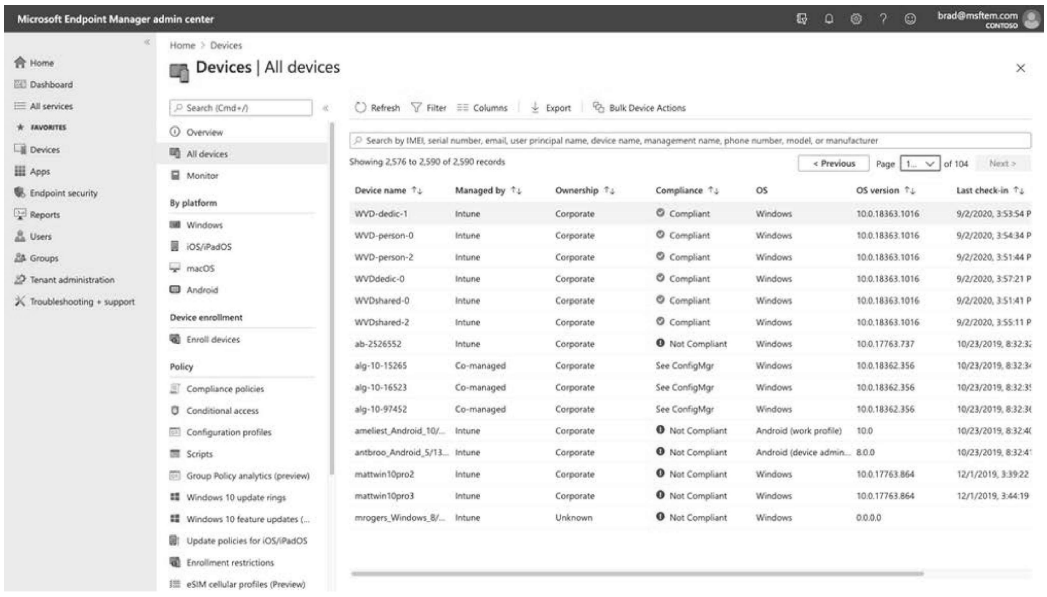


Figure 24: Managing devices through Microsoft Endpoint Manager's admin center

---

Note: Endpoint Manager does not support multi-session at the time of writing, only personal Desktop deployments within Azure Virtual Desktop.

To find out more about Microsoft Endpoint Manager with Azure Virtual Desktop, please refer to the following link.

More about CimFS

.CIM is a new file extension associated with the Composite Image Files System (CimFS). Mounting and unmounting CIM files is faster than VHD files. CIM also consumes less CPU and memory than VHD.

A CIM file is a file with a .CIM extension that contains metadata and at least two additional files that contain actual data, known as an **objectid** and region file. When creating MSIX images with CimFS, you would typically see a total of seven files, including the .CIM file.

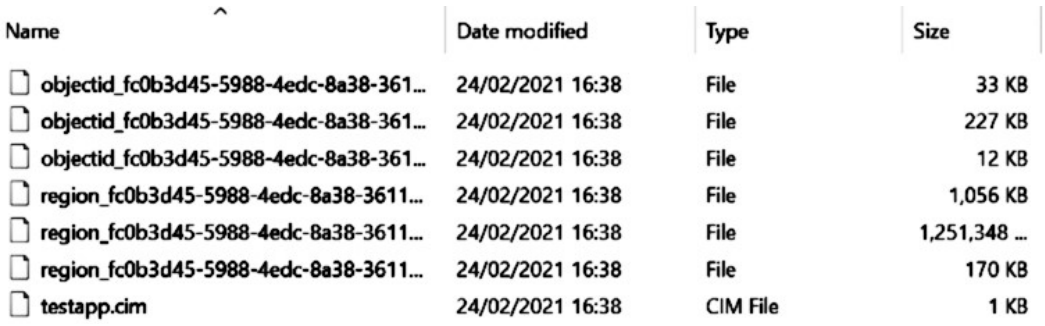
The files within the CIM file don't have extensions. *Table 6* provides is a list of example files you'd find inside a CIM:

Filename	Extension	Size
appname	CIM	1 KB
objectid_b5742e0b-1b98-40b3-94a6-9cb96f497e56_0	NA	27 KB
objectid_b5742e0b-1b98-40b3-94a6-9cb96f497e56_1	NA	20 KB
objectid_b5742e0b-1b98-40b3-94a6-9cb96f497e56_2	NA	42 KB
region_b5742e0b-1b98-40b3-94a6-9cb96f497e56_0	NA	428 KB
region_b5742e0b-1b98-40b3-94a6-9cb96f497e56_1	NA	217 KB
region_b5742e0b-1b98-40b3-94a6-9cb96f497e56_2	NA	264,132 KB

Table 6: List of example files inside CIM

Note: When using CimFS, you will not see the mounted CIM files within Disk Management. You will need to use **mountvol** to show these.

Figure 25 shows the files associated with a single CIM:



Name	Date modified	Type	Size
objectid_fc0b3d45-5988-4edc-8a38-361...	24/02/2021 16:38	File	33 KB
objectid_fc0b3d45-5988-4edc-8a38-361...	24/02/2021 16:38	File	227 KB
objectid_fc0b3d45-5988-4edc-8a38-361...	24/02/2021 16:38	File	12 KB
region_fc0b3d45-5988-4edc-8a38-3611...	24/02/2021 16:38	File	1,056 KB
region_fc0b3d45-5988-4edc-8a38-3611...	24/02/2021 16:38	File	1,251,348 ...
region_fc0b3d45-5988-4edc-8a38-3611...	24/02/2021 16:38	File	170 KB
testapp.cim	24/02/2021 16:38	CIM File	1 KB

Figure 25: Files associated with a single CIM

Table 7 gives a performance comparison between VHD and CimFS. These numbers were the result of a test run with five hundred 300 MB files in each format run on a DSv4 machine.

Specs	VHD	CimFS
Average mount time	356 ms	255 ms
Average unmount time	1615 ms	36 ms
Memory consumption	6% (of 8 GB)	2% (of 8 GB)
CPU (count spike)	Maxed out multiple times	No impact

Table 7: Performance comparison between VHD and CimFS

FSLogix Application Masking

For all those applications used outside of MSIX and MSIX app attach, you can use another feature called Application Masking to hide apps from users who are not in the required security groups.

Application Masking minimizes the number of gold images by creating a single image with all applications installed. The mapping and separation of applications (including printers, fonts, Office add-ins, and Internet Explorer plug-ins) is completed without packaging, sequencing, back-end infrastructure, or virtualization.

A use case for Application Masking would be to deliver all company applications to a master gold image and use Application Masking to hide certain applications from user groups who do not require them to perform their day-to-day duties.

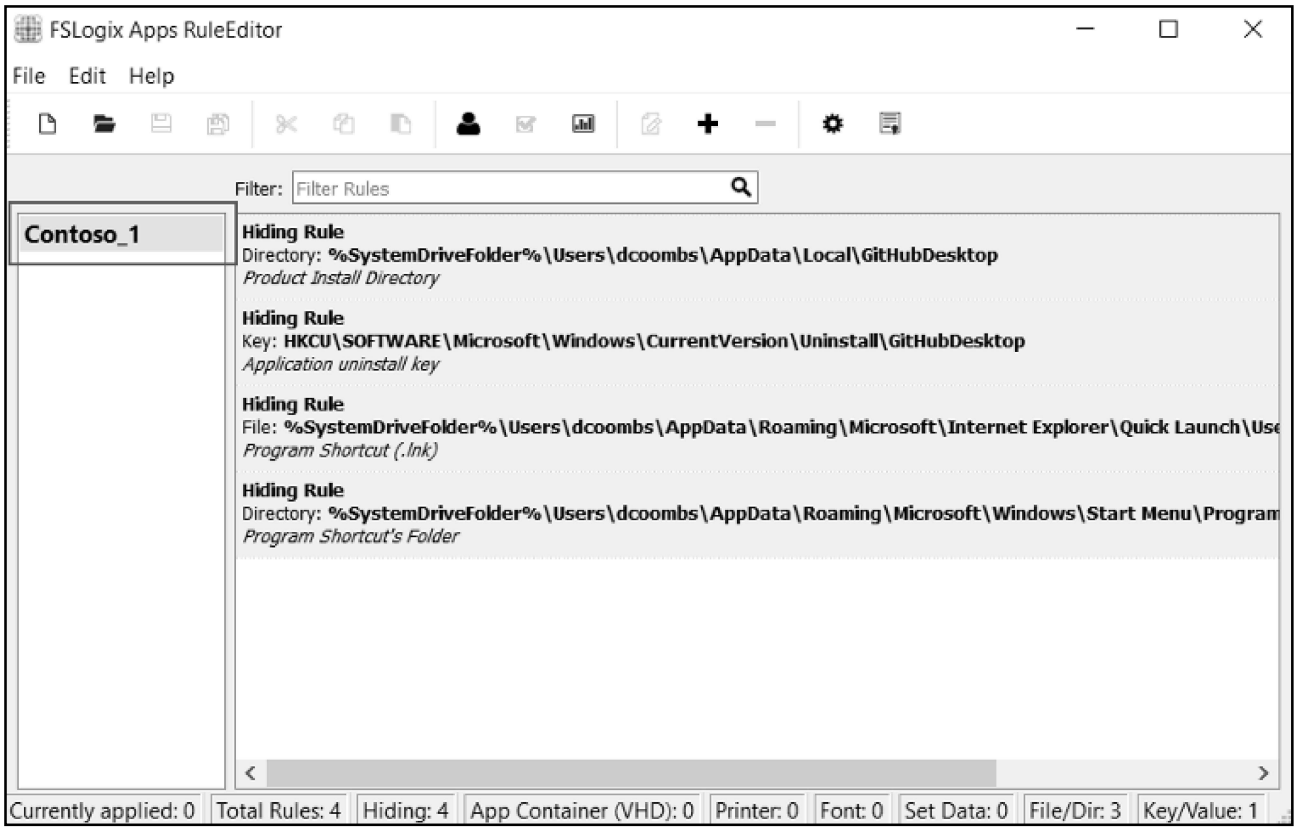


Figure 26: The Hiding Rules displayed in the FSLogix Apps RuleEditor window

Find out more about FSLogix Application Masking here.

Desktop App Assure

App Assure is a Microsoft service designed to address issues with Windows 10 and Microsoft 365 application compatibility.

The core objectives of this service are as follows:

- Remediate Line of Business (LoB) applications developed in-house.
- Engage off-the-shelf/third-party software manufacturers to remediate their Windows applications.
- Fix issues that customers encounter with Microsoft products, such as add-ins or Office macros.

Find out more about Desktop App Assure here. You can also read more on the FastTrack guidance details here.

MSIX partners

There are a set of specialized MSIX Partners that can help you with packaging applications to the MSIX format and MSIX app attach, including those referenced below. You can find the latest list at this link.

Figure 27: MSIX partners

Conclusion and resources

Summary

We started with an introduction to application management in Azure Virtual Desktop and a quick overview of the MSIX app attach delivery platform.

Later in the handbook, we discussed the prerequisites to configure MSIX app attach and the steps to deploy it. At the end, we discussed how to troubleshoot, optimization considerations and best practices.

In the further chapters, we provided the guidance on deploying MSIX app attach with Azure Virtual Desktop and had also looked into troubleshooting MSIX app attach. At the end, we discussed the optimization options and best practices.

We hope this handbook on Azure Virtual Desktop Application Management with MSIX app attach has given you a better understanding of how to deliver your applications centrally and use the best practices that will help provide a seamless user experience.

Resources

As you advance in your journey with Azure Virtual Desktop and application delivery and management, here are a few resources that can help:

- Read more about Azure Virtual Desktop MSIX app attach.
- Follow the Azure security baseline for Azure Virtual Desktop guidance.
- Test your Azure Virtual Desktop MSIX app attach knowledge with this learning module.
- Start now with a free Azure account.
- Get in touch with Azure sales to get personalized guidance and discuss pricing, technical requirements, and solutions for enabling secure remote work.
- Join the Azure Migration and Modernization Program to get guidance and expert help in migrating your on-premises VDI.

Glossary

Name	Description
Adding an MSIX package	In Azure Virtual Desktop, adding an MSIX package links it to a host pool.
CIM	.CIM is the new file extension of the CimFS.
CIMFS	Composite Image Files System.
Delayed or deferred registration	In delayed registration, each application assigned to the user is only partially registered. Partial registration means that the Start menu tile and double-click file associations are registered. Registration happens during user sign-in, so it has minimal impact on the time it takes to start using Azure Virtual Desktop. Registration completes only when the user runs the application in the MSIX package. Delayed registration is currently the default configuration for MSIX app attach.
Deregistration	Deregistration removes a registered, but non-running, MSIX package for a user. Deregistration happens when the user signs out of their session. During deregistration, MSIX app attach pushes application data specific to the user to the local user profile.
Destage	Destaging notifies the OS that an MSIX package or application currently isn't running or staged for any user, and can therefore be unmounted. This removes all references to it in the OS.
Expanding an MSIX package	The process of expanding an MSIX package is a multiple-step process.
MSIX application	An application stored in an MSIX file.
MSIX container	An MSIX container is where MSIX applications are run.
MSIX image	An MSIX image is a VHD, VHDx, or CIM file that contains one or more MSIX packaged applications.
MSIX package	An MSIX package is an MSIX file or application.
MSIX share	An MSIX share is a network share that holds expanded MSIX packages. MSIX shares must support SMB 3 or later. Applications get staged from this MSIX share without having to move application files to the system drive.
OS	Operating system
Packaging machine	A physical machine or VM used to capture and package applications and services into a required format.
Remote app streaming	Use Azure Virtual Desktop to deliver apps from the cloud to external (non-employee) users. For example, this would enable software vendors to deliver their app as a SaaS solution that their customers can access.

Name	Description
Publish an MSIX package	In Azure Virtual Desktop, a published MSIX package must be assigned to an AD DS or Azure AD user or user group.
Registration	In regular registration, each application assigned to a user is fully registered. Registration happens while the user signs into the session, which might impact the time it takes to start using Azure Virtual Desktop.
Repackage	Repackaging takes a non-MSIX application and converts it into MSIX using a packaging tool such as AppCURE or the MSIX Packaging Tool (MPT).
Staging	Staging is the process of mounting the VHD(x) or CIM to the VM and notifying the OS that the MSIX package is available for registration.
Uploading an MSIX package	Uploading an MSIX package involves uploading the VHD(x) or CIM that contains an expanded MSIX to an MSIX share. In Azure Virtual Desktop, uploads happen once per MSIX share. Once you upload a package, all host pools in the same subscription can reference it.
VHD	Virtual Hard Disk
VHDx	Virtual Hard Disk (version 2)

About the Author

Ryan Mangan is an end user computing (EUC) specialist, speaker, and presenter, who helps customers and technical communities with end user computing solutions, ranging from small to global, with 30,000+ user enterprise deployments in various fields. Ryan is the owner and author of ryanmangansitblog.com, which has over 3 million visitors and over 70+ articles on Remote Desktop Services and Azure Virtual Desktop. Some of Ryan's publications, and community and technical awards include the following:

- Author of:
  - *Quickstart Guide to Azure Virtual Desktop*
  - *An Introduction to MSIX App Attach*
- VMware vExpert******** eight years running
- VMware vExpert EUC 2021
- Parallels RAS VIPP *** three years running
- LoginVSI Technology Advocate ** two years running
- Technical person of the year, 2017 KEMP Technologies
- Parallels RAS EMEA Technical Champion 2018
- Microsoft Community Speaker
- Top 50 IT Blogs 2020 – Feed spot
- Top 50 Azure Blogs 2020 – Feed spot

Blog site: https://ryanmangansitblog.com
GitHub: https://github.com/RMITBLOG
GitHub: RMITBLOG/MSIX_APP_ATTACH Wiki