# Deploying WDAC Managed Installer with Provisioning Packages

This guide is designed to prevent students/standard users from being able to install applications on Windows devices while still allowing IT admins to deploy applications via Microsoft Intune. This leverages Windows Defender Application Control (WDAC), Managed Installer, and Applocker policies to control the environment.

## Prerequisites

**Windows Enterprise or Education** **SKU**- If the device is starting out as Pro or Pro Education and you’re using Autopilot, a MAK can be used via an Edition Upgrade policy in a provisioning package to upgrade the device to Enterprise or Education.

If you’re reimaging the device with Enterprise or Education, an edition upgrade key will not be necessary if you plan to include the key in your image.

Subscription activation may work, however the upgrade to Enterprise or Education happens after the user signs in. This means the WDAC/Managed Installer policies may not apply until after the user signs in and a reboot takes place. This leaves the machine open to the possibility of a user being able to install things before the policies are all applied successfully.

**Existing Provisioning package –** This guide assumes you’re familiar with making provisioning packages. It will go through editing an exiting package to configure WDAC and the Managed Installer.

**To download WDAC and Managed Installer files, click** [**here**](https://github.com/rbalsleyMSFT/managedinstaller) and under the green Code button select Download ZIP.  
  
Graphical user interface, text, application, email

Description automatically generated  
  
In the ManagedInstaler-main.zip file, go into the managedinstaller-main directory and copy out/extract the WDAC\_MI\_Files directory.  
  
Graphical user interface, text, application

Description automatically generated

## Copy WDAC Policy

This guide assumes that you already have a provisioning package and will use the advanced editor to edit the existing package. If you don’t have a package, you can walk through the Provision desktop devices wizard on the start page of Windows Configuration Designer and create a package.

1. Open **Windows Configuration Designer**
2. On the **Start page**, under **Recent Projects**, select your existing provisioning package
3. Under **Available customizations** on the left, expand **Runtime settings**
4. Expand **ProvisioningCommands – PrimaryContext**
5. Click **Command**
6. In the middle pane, for **Name** enter **CopyWDACPolicy** and click **Add** at the bottom
7. On the left pane, expand the **CopyWDACPolicy**
8. Click **CommandFile**
9. In the middle pane, on the **CommandFile** row, click **Browse…**
10. Navigate to the **WDAC\_MI\_Files** folder you downloaded earlier
11. Select the **{7ff774d3-0c76-44b4-be1d-1debcee783db}.cip** file and click **Open**
12. On the left pane, select **CommandLine**
13. In the middle pane, on the **CommandLine** row, enter   
      
    **cmd /c Powershell -executionpolicy Bypass -command "& {. Copy-Item -Path '.\{7ff774d3-0c76-44b4-be1d-1debcee783db}.cip' -Destination 'c:\windows\System32\CodeIntegrity\CIPolicies\Active\' -Force}"**
14. On the left pane, select **CopyWDACPolicy**  
      
    If you configured everything correctly, you should see something like the following image:  
      
    Graphical user interface, text, application, email

    Description automatically generated

## Apply Managed Installer PowerShell script

Managed Installer is a function of WDAC, however it’s configured similar to Applocker. There’s a XML configuration that specifies which exes are allowed to be managed installers. In this guide, the Intune Management extension, the OMADMClient, and Google Chrome are configured as Managed Installers. Chrome is added as a managed installer since it updates itself with a different executable (GoogleUpdate.exe). This exe needs to be configured as a managed installer for Chrome to update itself. If you don’t want Chrome as a managed installer, the ApplyManagedInstaller.ps1 file can be modified to remove GoogleUpdate.exe as a managed installer. If you have other apps you wish to allow to update themselves, you can modify the ApplyManagedInstaller.ps1 file to include the exes.

This can only be done via script as there aren’t any MDM Configuration Service Providers (CSPs) for managed installer functionality.

Whatever executable file(s) are designated as managed installers must also be restarted after the managed installer is configured. Because of this requirement, the Apply Managed Installer PowerShell script requires a reboot and the provisioning package will be responsible for the reboot.

1. Back in **Windows Configuration Designer**
2. Expand **ProvisioningCommands – PrimaryContext**
3. Click **Command**
4. In the middle pane, for **Name** enter **ApplyManagedInstaller** and click **Add** at the bottom
5. On the left pane, expand the **ApplyManagedInstaller**
6. Click **CommandFile**
7. In the middle pane, on the **CommandFile** row, click **Browse…**
8. Navigate to the **WDAC\_MI\_Files** folder you downloaded earlier
9. Select the **ApplyManagedInstaller.ps1** file and click **Open**
10. On the left pane, select **CommandLine**
11. In the middle pane, on the **CommandLine** row, enter   
      
    **cmd /c Powershell -executionpolicy Bypass -command "& {. .\ApplyManagedInstaller.ps1}"**
12. On the left pane, select **RestartRequired**
13. In the middle pane, on the **RestartRequired** row, select **True**
14. In the left pane, select **ApplyManagedInstaller**  
      
    If you configured everything correctly, you should see something like the following image:  
      
    Graphical user interface, text, application, email

    Description automatically generated

## Create Deny Administrative Apps Applocker Policy in Intune

While WDAC can include rules to block exes, it applies at the device level. This means every user would be blocked from using those exes on the device. Often in EDU environments admins need access to command prompt, PowerShell, etc. for troubleshooting purposes while students should not have access to these. We can create a separate AppLocker policy to prevent students from accessing specific exes while admins will have the ability to use them.   
  
Note that since Intune is deploying this instead of the provisioning package, there will be a delay for this policy to apply as it will apply at sign in of the user.

1. In the Microsoft Endpoint Manager console (endpoint.microsoft.com), go to **Devices – Configuration Profiles**
2. Click **Create profile**
3. Select Platform **Windows 10 and later** and Profile type **Templates - Custom** and click **Create**
4. On the Basics page, enter a name for the custom profile and click Next
5. On the Configuration settings page, click **Add**
6. On the Add Row flyout, enter a **Name** and description (optional)
7. For the **OMA-URI**, enter  
   **./Device/Vendor/MSFT/AppLocker/ApplicationLaunchRestrictions/DenyApps/EXE/Policy**
8. For the **Data Type** select **String (XML file)**
9. Click the **folder icon** and navigate to the **WDAC\_MI\_Files**\**ApplockerDenyAdministrativeApps.xml** file and click **Open**
10. Click **Save**
11. Back on the **Configuration settings** tab click **Next**
12. On the **Scope tags** tab click **Next** unless you know you need to assign a scope tag
13. On the **Assignments** tab select a **user group** that you wish to deploy the policy to and click **Next**  
      
    Note: While you can select a device group, a user group would be preferred in this case. This will block students from using cmd, powershell, reg, regedit, powershell\_ise on any machine. If you want to allow students to use those exes on certain machines, you can create an exclude filter to exclude certain machines (maybe lab machines where some students may be using command line tools for computer science).   
      
    More info on filters - [Create filters in Microsoft Intune | Microsoft Docs](https://docs.microsoft.com/en-us/mem/intune/fundamentals/filters?msclkid=fa061155cfe211eca8a1b879dce3bc7b)
14. On the **Applicability rules** tab click **Next**
15. On the **Review + create** tab click **Create**

## Optional – Set Edition Upgrade Key

This step is required if the device you are using has Windows Pro or Pro Education. If it does, it needs to be upgraded to Windows Enterprise or Windows Education.

1. Back in **Windows Configuration Designer**
2. In the left pane, expand **EditionUpgrade**
3. Click **UpgradeEditionWithProductKey**
4. In the **UpgradeEditionWithProductKey** row enter your Enterprise or Education MAK or KMS key

Save and Export the Provisioning Package

1. At the top menu bar of Windows Configuration Designer Click **File – Save**
2. Click **Export – Provisioning Package**
3. On the **Describe the provisioning package** page click **Next**
4. On the **Select security details for the provisioning package** page click **Next**
5. On the **Select where to save the provisioning package** page click **Next**  
   If you get a Confirm action pop up that asks if you want to overwrite an existing package, select **Yes**
6. On the **Build the provisioning package** page, click **Build**
7. On the All done page, click one of the blue links to go to the location of the provisioning package. Copy the file with the .ppkg file extension to your USB drive

## Testing

The expectation should be that all users of the device will be restricted in installing applications from the internet that are not Microsoft signed. This means students and admins won’t be able to download most things. However, with the additional AppLocker policy, students will be restricted from running cmd, PowerShell, powershell\_ise, reg, and regedit exes while admins won’t (technically, this all depends on who you deployed the AppLocker policy in Intune to, so if you decided to include students and teachers and limited their ability, then they’ll all be blocked from running those exes). Students and admins will be able to install applications made available to them via the Intune Company Portal application.

If you have never deployed the Company Portal application, you can follow these instructions - [Add and assign the Windows 10 Company Portal app for Autopilot provisioned devices - Microsoft Intune | Microsoft Docs](https://docs.microsoft.com/en-us/mem/intune/apps/store-apps-company-portal-autopilot) – disregard that it says that it’s for Autopilot. This can be used for both Autopilot and provisioning package scenarios.

Once the Company Portal application is deployed, deploy applications as available to your end users (do not make them required) so they can install those applications on their own as they need them.

Below is a screenshot of what the Company Portal app looks like when apps are made available. It’s highly recommended that when you publish your applications that you also include logos, otherwise you end up with grey squares that aren’t very user friendly as shown below.

Graphical user interface, application

Description automatically generated