Contents

[Prerequisites 2](#_Toc128400901)

[Create the Virtual Machine 3](#_Toc128400902)

[Install Windows 6](#_Toc128400903)

[Create M365 Apps/Office ISO (optional) 8](#_Toc128400904)

[Download M365 Apps using the Office Deployment Tool 8](#_Toc128400905)

[Create Windows PE Capture Media 9](#_Toc128400906)

[Capture the FFU 11](#_Toc128400907)

[Optimizing the FFU 11](#_Toc128400908)

[Create WinPE Deployment Media 11](#_Toc128400909)

[Prepare Dual Partition USB 3+ drive 12](#_Toc128400910)

[Mount WinPE ISO and copy contents to USB Boot partition 12](#_Toc128400911)

[Copy the Windows FFU files to the Deploy drive of your USB drive 14](#_Toc128400912)

[Driver Support (optional but recommended) 14](#_Toc128400913)

[Having DISM inject drivers on the fly (recommended) 14](#_Toc128400914)

[How to inject drivers into FFU 15](#_Toc128400915)

[Device Naming (optional but recommended) 15](#_Toc128400916)

[Prompt to enter name 16](#_Toc128400917)

[Specifying Multiple Name Prefixes 16](#_Toc128400918)

[Optional - Copy provisioning package to the USB drive 18](#_Toc128400919)

[Optional - Using Autopilot for Existing Devices 18](#_Toc128400920)

[Create User Driven Autopilot Profile 19](#_Toc128400921)

[Install Azure AD, WindowsAutopilotIntune, Intune Graph Powershell modules 21](#_Toc128400922)

[Get the User Driven Autopilot profile and convert to JSON 22](#_Toc128400923)

[Multiple Autopilot Configuration Files 22](#_Toc128400924)

[Insert USB drive into target PC 22](#_Toc128400925)

[Optional - Copying a custom WinRE.wim 23](#_Toc128400926)

[Optional – Adding Multiple Components using unattend.xml 24](#_Toc128400927)

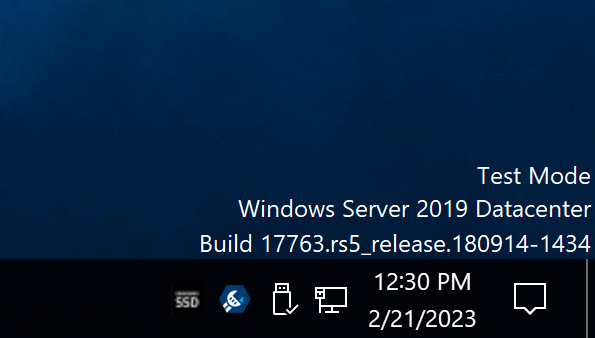
[Troubleshooting 24](#_Toc128400928)

# Prerequisites

1. Enable Hyper-V – this can be done on either Windows 10/11 or Windows Server  
     
   [Enable Hyper-V on Windows 10 | Microsoft Learn](https://learn.microsoft.com/en-us/virtualization/hyper-v-on-windows/quick-start/enable-hyper-v)  
   [Get started with Hyper-V on Windows Server | Microsoft Learn](https://learn.microsoft.com/en-us/windows-server/virtualization/hyper-v/get-started/get-started-with-hyper-v-on-windows)
2. Install the latest Windows Assessment and Deployment Kit (ADK) to the machine you installed Hyper V on.  
     
   **You’ll need to do the following until Windows 23H2 is released**  
     
   You’ll need to download and install the Insider Preview of the Windows ADK and the Windows PE add-on for the Windows ADK. This is due to a bug with optimizing ffu files which was fixed as of the 25295 version of the ADK. This will eventually roll up into the 23H2 version of the Windows ADK.  
     
   <https://www.microsoft.com/en-us/software-download/windowsinsiderpreviewADK>

Install the Windows ADK first. When installing the Windows ADK, you only need the **Deployment Tools** feature checked. Take the defaults for the rest of the setup wizard.  
  
Graphical user interface, text, application, email

Description automatically generated  
  
Install the WinPE add-on next. Install with default settings.

1. Flight Signing needs to be enabled in order for dism from the Insider ADK to run.  
     
   If you’re using a non-Windows Insider build, flight signing needs to be enabled.  
     
   Go to **Start – search for Command Prompt – right click Command Prompt** and select **Run as Administrator**  
     
   Run **bcdedit /set testsigning on**  
     
   **Reboot** the machine  
     
   After reboot, you should see a water mark on the desktop indicating Test Mode is enabled.  
     
     
     
   Note – For security reasons, it’s best to only enable test signing when necessary. This will only need to be done until the Windows 23H2 ADK is released. Once that’s released, the dism fixes that are needed will be rolled up into the 23H2 ADK. Once you’ve finished building your FFU, it’s best to turn off flight signing (you’ll do that at the end of this guide).
2. Note – you can skip this step if you’ve downloaded the files from Github already  
     
   Download the script files from the following Github Repo  
   <https://github.com/rbalsleyMSFT/FFU>

Direct link to the FFU script files zip is here  
<https://github.com/rbalsleyMSFT/FFU/archive/refs/heads/main.zip>  
  
**Open** the **FFU-main.zip** file

Double click **FFU-Main**

**Copy** the **FFUDevelopment** folder and **paste** it to your **c:\ drive**

1. Windows media – download the Windows ISO from VLSC, Visual Studio downloads, or the Windows Media Creation Tool.   
     
   It’s recommended that you download the consumer media, not business. This is because Subscription Activation will normally fail if the media is mismatched from the key in the firmware. If you plan on using a MAK or KMS to activate, you can use media from VLSC, but if you expect the device to activate automatically and upgrade to Enterprise or Education SKUs via Subscription Activation, you must use consumer media, which you can get from Visual Studio Subscriptions, or the Media Creation Tool. It is possible that you may need to use business edition media, though most OEM devices ship with consumer editions media.

# Create the Virtual Machine

1. **Edit** the **c:\FFUDevelopment\BuildFFUVM.ps1** file  
     
   At the top of this file, there are several variables. The variables you must edit are  
     
   **$VMPath** – change this to the location of where you want the VM stored  
   **$ISOPath** – put in the location of your Windows 10 or 11 ISO file  
   **$memory** – 8GB might be too much, adjust to 4GB if necessary

**$disksize** – 30GB – this is enough for Windows and Office. If you plan to install other apps while in audit mode, you might need to increase the size.   
  
Save the file

#Modify variables

$VMPath = "c:\VM\$VMName"

$ISOPath = "E:\software\ISOs\Windows\Windows 11\en-us\_windows\_11\_consumer\_editions\_version\_22h2\_updated\_feb\_2023\_x64\_dvd\_4fa87138.iso"

$memory = 8GB

$disksize = 30GB

1. Once saved, click **Start** – type **Powershell** – **Right Click Windows PowerShell and select Run as Administrator**
2. Type **c:\FFUDevelopment\BuildFFUVM.ps1** and hit **Enter**  
     
   Note – if you get an error that the script cannot be loaded run the following command:  
     
   **Set-ExecutionPolicy -executionpolicy Bypass**

Then try running the script again  
  
You should see output like the following  
  
Text

Description automatically generated with low confidence

# Install Windows

1. Click the **Start** button  
   Graphical user interface, text

   Description automatically generated
2. When prompted, make sure to **press any key to boot from CD or DVD**
3. In Windows Setup, take the defaults up until you get to the Activate Windows screen
4. On the **Activate Windows** page, select **I don’t have a product key**  
   Graphical user interface, application

   Description automatically generated
5. On the **Select the operating system you want to install** page, select **Windows 11 Pro** and click **Next.**  
     
   **IMPORTANT** - If the devices you’ll be deploying this FFU to shipped from the OEM with something other than Pro, you’ll want to select that to build your FFU from. In most cases, it’ll be Pro, but it can also be Pro Education or Home.   
     
   If you have Home devices, it’s best to use Enterprise or Education SKUs as Home doesn’t support Azure AD or Domain Joining. You’ll need to use volume licensing media and keys to activate Enterprise or Education SKUs. You won’t be able to use Subscription Activation for those machines and will need to use either MAK or KMS to activate.  
     
   Graphical user interface

   Description automatically generated
6. Accept the Licensing terms and click **Next**
7. On the **Which type of installation do you want page**, select **Custom**
8. On the **Where do you want to install Windows** page, click **Next**

Wait for the Windows installation to complete.

# Create M365 Apps/Office ISO (optional)

## Download M365 Apps using the Office Deployment Tool

1. Click **Start** – type **Powershell** – Click **Windows Powershell**
2. Type **c:\FFUDevelopment\CreateOfficeISO.ps1** and hit **Enter**  
     
   This script will download the latest Office Deployment Toolkit, download Office, create an office.iso file, and mount that office.iso file to the VM.
3. Back in **Hyper-V,** select your **VM**, which should be at the first screen of the Out of Box Experience (OOBE)
4. **Boot to audit mode** by holding **CTRL+SHIFT+F3**. If that doesn’t work, click into an empty area of the VM to bring focus to the VM and try again.  
     
   You should end up at the audit mode desktop with Sysprep open  
     
   Graphical user interface

   Description automatically generated
5. Click **Start** and type **cmd** and hit **Enter**
6. Type **D:\installofficeandsysprep.cmd** and hit Enter  
     
   The script will install Office, run sysprep, and shut the VM down.

# Create Windows PE Capture Media

Next, you’ll create WinPE capture media which will automate the naming of the FFU file and copy it to a location of your choosing.

There are two capture scripts that need to be edited.

1. Edit **C:\FFUDevelopment\WinPECaptureFFUFiles\CaptureFFU.ps1**  
     
   This script contains all the logic to automatically name the FFU and copy it to your VM Host or other network location. This script runs from within WinPE.
2. Edit the net use command with the network location you want to copy the FFU to and change the username/password as needed

#Modify the net use path to map the W: drive to the location you want to copy the FFU file to

net use W: \\192.168.1.2\c$\FFUDevelopment /user:administrator password

1. Save the file
2. On the PC you installed the Windows ADK and PE to click **Start** – type **Deployment and Imaging Tools**
3. **Right click** on **Deployment and Imaging Tools Environment** and select **Run as Administrator**
4. Type **C:\FFUDevelopment\BuildWinPECaptureMedia.cmd** and hit **Enter**  
     
   There should be a WinPE\_FFU\_Capture.iso file in the C:\FFUDevelopment folder when complete  
     
   **OPTIONAL**

If you need to add drivers, edit the BuildWinPECaptureMedia.cmd file and remove the REM from the below line in the BuildWinPEDeploymentMedia.cmd file and modify the /Drivers path to include the path to the drivers you need. You might need storage/network or even keyboard drivers (some Surface devices have keyboards that don’t work in WinPE)

REM dism /image:C:\FFUDevelopment\WinPE\mount /Add-Driver /Driver:<Path to Drivers folder e.g c:\drivers> /Recurse

1. Edit **C:\FFUDevelopment\ModifyVMForCapture.ps1**

The script will check to see if the FFU VM is running and if it’s not, it’ll change the DVD drive to the first boot device, mount the ISO, and connect the network switch so the FFU can be copied.

1. Modify the **$ISOPath** variable and change it to the location of the iso file
2. Modify the **$VMSwitchName** variable to include the name of your switch
3. Save the file  
     
   The script currently has ‘\*Intel\*’, which is doing a wild card search for any switches with the name of Intel. You can replace Intel with a portion of the string of your switch name, or you can enter the full name of your switch

#Modify variables

$ISOPath = 'C:\FFUDevelopment\WinPE\_FFU\_Capture.iso'

$VMSwitchName = '\*intel\*'

# Capture the FFU

1. Click **Start** – type **Powershell**
2. **Right click Windows PowerShell** and select **Run as Administrator**
3. Type **c:\FFUDevelopment\ModifyVMForCapture.ps1** and hit **Enter**

The VM should pop up

1. Click **Start** on the VM and make sure to **hit any key to boot from the CD/DVD drive**.   
     
   If you miss the prompt to boot from the CD/DVD drive, Windows will specialize itself and boot to OOBE. If this happens, you’ll need to sysprep again, or start over. It’s important you don’t miss the boot from CD/DVD prompt.

# Optimizing the FFU

FFUs are sector-by-sector copies of a hard drive. This means when deploying a FFU, the total disk space available is going to be the same amount as the size of the hard drive of the machine that was used to capture the FFU.

Dism has a command called /optimize-ffu that will optimize the FFU so the total amount of disk space on the destination machine will be used instead of only showing the size of the source machine that was used to capture. In other words, if the destination machine has a total size of 128GB, after optimizing the FFU, the destination machine will still have a total size of 128GB. If you don’t optimize the FFU, and the VM you used had a 30GB disk, the destination machine would show a total size of 30GB after the FFU is applied. It’s important to optimize the FFU so all 128GB of space is available. If you deploy a FFU and wonder why the disk space is less than expected, this is the reason.

Note – Optimize-FFU was broken for a very long time. As of Windows ADK/PE version 25295, that has been resolved.

1. If you kept the default net use path in the CaptureFFU.ps1 script, then you should have a FFU file in c:\FFUDevelopment
2. On the PC you installed the Windows ADK and PE to click **Start** – type **Deployment and Imaging Tools**
3. **Right click** on **Deployment and Imaging Tools Environment** and select **Run as Administrator**
4. Run **dism /optimize-ffu /imagefile:"C:\FFUDevelopment\Name\_Of\_Your\_FFU\_File.ffu”**  
     
   This will mount the image, optimize, then re-capture the FFU. This will take a while, please be patient.  
     
   Once complete, the FFU is now ready to be deployed

# Create WinPE Deployment Media

1. On the PC you installed the Windows ADK and PE to click **Start** – and type **Deployment and Imaging Tools**
2. **Right click** on **Deployment and Imaging Tools Environment** and select **Run as Administrator**
3. Type **C:\FFUDevelopment\BuildWinPEDeploymentMedia.cmd** and hit **Enter**  
     
   There should be a WinPE\_FFU\_Deploy.iso file in the C:\FFUDevelopment folder  
     
   **OPTIONAL**

If you need to add drivers, edit the BuildWinPECaptureMedia.cmd file and remove the REM from the below line in the BuildWinPEDeploymentMedia.cmd file and modify the /Drivers path to include the path to the drivers you need. You might need storage/network or even keyboard drivers (some Surface devices have keyboards that don’t work in WinPE)

REM dism /image:C:\FFUDevelopment\WinPE\mount /Add-Driver /Driver:<Path to Drivers folder e.g c:\drivers> /Recurse

1. Disable Flight Signing  
     
   Go to **Start – search for Command Prompt – right click Command Prompt** and select **Run as Administrator**  
     
   Run **bcdedit /set testsigning off**  
     
   **Reboot** the machine  
     
   After reboot, the water mark on the desktop that says Test Mode should be gone.

# Prepare Dual Partition USB 3+ drive

Get a USB 3+ drive/stick. It’s not recommended to use USB 2 drives as they are significantly slower. You’ll want to get one that’s at least 16GB. Due to the FFU being > 4GB in size, we need to create two different partitions on a USB drive to boot to Windows PE and install the FFU file. Follow the below commands to create a new USB drive

1. Open Powershell as administrator and run the following commands
2. Diskpart
3. List disk
4. Sel disk X (where X is the number of your USB drive)
5. Clean
6. Create Part Primary size=2048
7. Assign
8. Active
9. Format fs=fat32 quick Label=”Boot”
10. Create part primary
11. Assign
12. Format fs=ntfs quick Label=”Deploy”
13. Exit

# Mount WinPE ISO and copy contents to USB Boot partition

1. Navigate to **c:\FFUDeployment**
2. **Right-click** on **WinPE\_FFU\_Deploy.iso** and select **Mount**
3. File Explorer should take you to the mounted contents of the ISO file. It should look like this:

This pc DVD Drive DVD ROM 
Name 
bg-bg 
800t 
da-dk 
de-de 
el-gr 
en-gb 
en-us 
es-nu 
fr-ca 
fr-fr 
hr-hr 
hu-hu 
ja-jp 
ko-kr 
It-It 
IV-Iv 
nb-no 
PI-PI 
pt-br 
sk-sk 
sl-si 
sources 
sr-latn-rs 
sv-se 
uk-ua 
zh-tw 
bootmgr 
C] bootmgr .efi 

1. Copy the entire contents of this drive and paste it to the Boot drive of your USB drive

# Copy the Windows FFU files to the Deploy drive of your USB drive

1. In your **c:\FFUDeployment** folder there should be a FFU file you captured previously. Copy that file to your Deploy drive.

You can copy one or multiple FFU files to your Deploy drive. If only one file is copied, the ApplyFFU.ps1 script in WinPE will apply just that file. If there are multiple files, a menu option will be presented to allow you to choose which FFU to apply.

It is recommended to choose the Windows SKU that matches what the OEM installed on the device. For example, if you purchased devices that come with Windows 10 Pro Education, use the Windows 10 Pro Education FFU. The reason for this is due to activation. Windows will automatically activate the device if the SKU matches the key that is embedded in the firmware. This is important when used in conjunction with Windows Subscription Activation as Subscription Activation will leverage the key in the firmware to activate the device and step the device up to either Enterprise or Education depending on which key is detected in the firmware.

# Driver Support (optional but recommended)

Adding drivers is not necessary in most cases, however it will result in a better experience on the device. If you do not add drivers, Windows Update will update the device. The first logged on user will see driver installation pop ups as WU installs the drivers. If IT will be signing on first, they can deal with the driver prompts and validation before delivering to the end user.

If you would like to add drivers, there are two ways to do it.

1. Mount the FFU and inject them directly into the FFU file using DISM
2. Create a drivers folder on your Deploy partition and copy your drivers there (e:\drivers\Lenovo 300e) (recommended)

The first option will result in faster imaging. This option is good if imaging the same model in an environment many times. It is not recommended to inject drivers of many models. Doing so will increase the driver store considerably. When dealing with devices that have limited storage, you run the risk of running out of disk space. You could create your own FFU per model with only the drivers injected for that specific model.

The second option is easier; however, it will take longer to image. This is due to dism injecting the drivers on the fly. The time it takes will depend on how many drivers are added and the size of the drivers.

## Having DISM inject drivers on the fly (recommended)

Simply create a folder on the deploy partition called drivers. Inside that folder, create sub-folders for each model you will be imaging. If imaging multiple models, a pick list will show up asking which set of drivers to install.

Graphical user interface, application

Description automatically generated

## How to inject drivers into FFU

1. If you haven’t already, download the drivers you wish to install for the model you are deploying. Normally, OEMs will provide an enterprise SCCM cab/zip of the drivers. Some, however, do not, especially for the low-cost EDU devices (Lenovo doesn’t for the 14W, 100e, 300e. They require individual downloads for each component, which are all exes). Don’t forget to extract the drivers.   
     
   The 3189 driver cab looks like this once extracted  
     
   A screenshot of a cell phone

   Description automatically generated

For Lenovo devices, each exe can be extracted via the command line using:  
  
Filename.exe /VERYSILENT /Extract="Yes" /DIR=<PathToWhereYouWantToExtractTo>

1. **Right-click Start Menu** icon and select **Windows Powershell (Admin)**
2. Run **mkdir c:\mount**
3. Run **Dism /Mount-Image /ImageFile:"<PathToFFUFile>" /Index:1 /MountDir:"C:\mount"**  
   (where <PathToFFUFile> is the location of your FFU e.g. e:\Win10\_1903\_Pro\_Edu.ffu)
4. Run **dism /image:C:\mount /Add-Driver /Driver:<PathToDriverFiles> /Recurse**   
   (Replace the c:\temp\drivers with the location of your extracted driver files)
5. Run **Dism /Unmount-Image /MountDir:C:\mount /commit**

# Device Naming (optional but recommended)

**New as of June 2020**

Device naming can be done from PE. The way this works is by leveraging an unattend.xml file to either take input from the user at imaging time or read a list of prefix values and append the serial number of the device. There are some major benefits to doing this:

1. Total deployment time is reduced if naming is set at FFU deployment time since there is no additional reboot done during OOBE.
2. Reduces the need for multiple provisioning packages or autopilot profiles. This means you can use a single PPKG or autopilot profile.
3. Allow for future considerations on how to customize/automate device naming in the future

**IMPORTANT – If using a provisioning package or autopilot json file, DO NOT specify a name in either of these. They will overwrite the name you have specified in the unattend.xml.**

By leveraging the unattend.xml, you can also add additional steps during the specialize phase to customize the experience even further.

## Prompt to enter name

1. Create a folder on the deploy partition of the USB drive named **unattend**
2. In the unattend folder, create a file named **unattend.xml** (this file has already been created if you downloaded this from aka.ms/ffu or the internal Microsoft FFU deployment share . It is in the unattend folder in that location).
3. The contents of the unattend.xml file should be the following  
     
   <?xml version="1.0" encoding="utf-8"?>

<unattend xmlns="urn:schemas-microsoft-com:unattend">

<settings pass="specialize">

<component name="Microsoft-Windows-Shell-Setup" processorArchitecture="amd64" publicKeyToken="31bf3856ad364e35" language="neutral" versionScope="nonSxS" xmlns:wcm="http://schemas.microsoft.com/WMIConfig/2002/State" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

<ComputerName>MyComputer</ComputerName>

</component>

</settings>

</unattend>

Graphical user interface, text

Description automatically generated

If this file is located during imaging, the user will be prompted to enter a name before the FFU is applied to the device.

## Specifying Multiple Name Prefixes

If you have multiple device name prefixes for different locations or device use cases, or even a single prefix, you can specify a prefixes.txt file in the unattend folder. If the prefixes.txt file is detected and a single prefix is listed, the device will just use that prefix and append the serial number of the device. If there are multiple prefixes listed in the prefixes.txt file, you will be prompted to select which prefix you want to name the device and the serial number will be appended to that prefix. If you want a dash in the name, include the dash in the prefix (e.g. if ABCD- is in the prefixes.txt file, the device name will be ABCD-*SerialNumber*).

1. Follow the previous steps used to create the unattend folder and the unattend.xml file.
2. Create a file named **prefixes.txt** in the **unattend** folder
3. In the prefixes.txt file, enter one prefix per line and save the file

Your folder should look like the following

Graphical user interface, text

Description automatically generated

Your prefixes.txt file should look similar to the following image

Graphical user interface, text, application

Description automatically generated

# Device Enrollment

There are three common ways to enroll new devices using this method to Azure AD and Intune

1. Autopilot
2. Provisioning packages
3. Autopilot for Existing Devices

If using Autopilot, your devices should already be registered, and an Autopilot profile should be assigned to them. At this point, you can Insert the USB drive to your device and boot from it.

However, if your devices aren’t enrolled into Autopilot, you can either use a Provisioning Package, or Autopilot for Existing devices.

Generally, AP or Provisioning Packages are the two common methods of deployment.

The reasons you may want to use Autopilot for existing devices vs. using Provisioning are:

1. Speed of USB drive removal – AP for existing devices allows a technician to remove the USB drive after the device applies the FFU and reboots for the first time. This is because there isn’t a provisioning package. This means you can take the USB drive out sooner and image another device. The FFU usually takes about 2-3 minutes to apply. Once the device reboots after applying the FFU, the drive can be removed if you are not using provisioning packages.
2. Setting up a device for 1:1 usage – AP for existing devices will set user device affinity up for the first user that enrolls the device. In this scenario, any applications deployed as available to users/devices in Intune will only show up for that user in the Company Portal. **Do not use this if the devices will be shared.**

If you’re in a rural location, have poor bandwidth, or are deploying shared devices (laptop carts/labs), provisioning packages are recommended. This guide isn’t designed to go in depth on provisioning packages, but there are ways to include complex apps like contain multiple folders and files. At a high level, you can put all of the content into a WIM file, mount the WIM, and install the apps. We’ll provide a guide on this at a later date.

# Optional - Copy provisioning package to the USB drive

**Process changed as of June 2020**

**Recommended for shared devices or devices that may be used 1:1 for now, but could be shared later and won’t be reimaged**

If you plan to use a provisioning package to enroll a device, follow the following steps.

1. On the **deploy partition** of your device, **create a new folder** named **PPKG**
2. Copy one or multiple .PPKG files to the PPKG folder

In June 2020, this process changed to allow for multiple provisioning packages to exist on a single USB drive. If multiple packages are detected, a menu will display each package and ask you to select which package to install. This is useful for organizations that have different packages for different use cases, packages configured for wired vs wireless scenarios, different device names, etc. and do not want to swap packages or have multiple USB drives.

Do NOT place a PPKG file on the BOOT partition of your USB drive AND create a PPKG folder on the deploy partition with one or multiple PPKG files in the folder. This will cause Windows to see two eligible PPKG files to be used during OOBE, which will result in you having to select which package to use at OOBE, slowing the overall process down.

**Important – If you set up the device naming method mentioned earlier, make sure to remove the ComputerName (or DNSComputerName) from your provisioning package. If you do not remove the ComputerName (or DNSComputerName) from the package, the package will overwrite the name you specified via the device naming method earlier. The Accounts node in your provisioning package should look like the following image.**  
  
Graphical user interface, text, application, email

Description automatically generated

# Optional - Using Autopilot for Existing Devices

If you want to configure your media to be able to Autopilot enroll any device, you can use Autopilot for existing devices.

This requires you to create a folder on your **Deploy** drive called **autopilot** and insert an AutopilotConfigurationFile.json file in that folder (E:\autopilot\AutopilotConfigurationFile.json). When the device finishes installing the FFU, it will copy the json file to C:\Windows\Provisioning\Autopilot\AutopilotConfigurationFile.json.

**Note –** AP for existing devices only works for User Driven profiles. It does not work for Self-Deploying or Pre-Provisioning scenarios. It also does not register the device into Autopilot. You can deploy an Autopilot Profile to a group that this device will end up in and select the setting **Convert all targeted devices to Autopilot**.

## Create User Driven Autopilot Profile

Make sure you're using User Driven Profiles (Self Deploying profiles don't work with Offline). In this example, we will query this profile by the display name of the profile. In this case, we'll be using User Driven as the display name.

1. In Intune, go to **Device Enrollment – Windows Enrollment – Deployment Profiles**
2. **Click Create Profile**
3. For **Name,** type in **User Driven** (the name MUST be User Driven for the powershell command to work later)
4. For **Convert all targeted devices to Autopilot** select **Yes**Note – do this in production environments where you want the devices to be autopilot enrolled. This will allow any devices with this profile targeted to go through the Autopilot workflow if the device is reimaged or reset. If you’re working in a lab environment where the devices you use may also be used for demoing provisioning scenarios (SUSPC, WCD) then make sure to set this to NO. You will still get the AP behavior when you use the json file if this is set to No, but the device isn’t really in Autopilot, so it can be used for provisioning and other scenarios later.

A screenshot of a cell phone

Description automatically generated

1. For OOBE settings, configure like below, making sure to select **deployment mode** as **User-Driven** and **Join to Azure AD** as **Azure AD Joined**  
   A screenshot of a cell phone

   Description automatically generated
2. Click **Next** through the rest of the wizard (even on the Assignments tab as you do not have to assign this profile to a group). On the Review + create tab, click **Create.**

## Install Azure AD, WindowsAutopilotIntune, Intune Graph Powershell modules

Open Powershell as an admin

**Run Install-Module AzureAD**

At the untrusted repository message, type Y or A

**Run Install-Module WindowsAutopilotIntune**

At the untrusted repository message, type Y or A  
**Run Install-Module Microsoft.Graph.Intune**

At the untrusted repository message, type Y or A

## Get the User Driven Autopilot profile and convert to JSON

Still in Powershell, run **Connect-MSGraph**

Type in your credentials

The below command will query for our Autopilot Profile with the display name User Driven. It will then convert the profile to the JSON format and save the file to your desktop as **AutoPilotConfigurationFile.json** and encode it using ASCII (must be saved as ASCII and it must use this exact name)

Run the following command

**Get-AutoPilotProfile |? {$\_.displayname -like '\*User Driven\*'} | ConvertTo-AutoPilotConfigurationJSON | Out-File -FilePath $env:userprofile\desktop\AutoPilotConfigurationFile.json -Encoding ASCII**

1. Go to your desktop and copy the AutopilotConfigurationFile.json file
2. On the **Deploy** drive of your USB drive, make a folder called **autopilot**
3. Copy the **AutopilotConfigurationFile.json** file to the **autopilot** folder  
     
   This should look like the following  
     
   A screenshot of a cell phone

   Description automatically generated

## Multiple Autopilot Configuration Files

If imaging many devices that will go to different locations or have different Autopilot settings, you will need multiple Autpilot profiles which means you will also need multiple AutoPilotConfigurationFile.json files.

Give each file a unique name (e.g. ContosoHighSchool.json, FabrakamMiddleSchool.json) and store them in your autopilot folder. If multiple files are detected, you will be prompted for which file you want to use.

# Insert USB drive into target PC

Boot your device from the USB drive. Depending on how you have configured the USB drive, you’ll either be prompted for a device name, FFU file, drivers, Autopilot, or PPKG files. If you just have a single FFU file, single driver package (or no driver package), a single device name prefix defined, and either a single autopilot or PPKG file, then no prompts are expected and device imaging should begin automatically.

# Optional - Copying a custom WinRE.wim

If you are deploying a device that does not have built in drivers in Windows, three things need to be updated

1. WinPE (Storage and possibly keyboard/mouse drivers)
2. Each FFU file (Storage and network drivers)
3. WinRE (Storage and possibly keyboard/mouse drivers)

You can create your own custom WinRE.wim file with the appropriate drivers. Keep in mind that the version of WinRE must match the version of Windows you are deploying.

Your Deploy partition should look like the following:

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

# Optional – Adding Multiple Components using unattend.xml

**New as of June 2022**

In prior releases, only a single component could be used, and it was used primarily to set the device name while in PE to help speed up imaging.

The ApplyFFU.ps1 script has been updated to allow multiple components to be specified in the specialize phase.

In the example below, we’re able to set the device name and run a command to set the date. This can be useful in scenarios where the device date is so old that AzureAD doesn’t allow it to enroll.

See the previous section **Device Naming (optional but recommended)** for how to configure the USB drive to leverage the unattend.xml file.

<?xml version="1.0" encoding="utf-8"?>

<unattend xmlns="urn:schemas-microsoft-com:unattend">

  <settings pass="specialize">

    <component name="Microsoft-Windows-Shell-Setup" processorArchitecture="amd64" publicKeyToken="31bf3856ad364e35" language="neutral" versionScope="nonSxS" xmlns:wcm="http://schemas.microsoft.com/WMIConfig/2002/State" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

      <ComputerName>A-006259610866</ComputerName>

    </component>

    <component name="Microsoft-Windows-Deployment" processorArchitecture="amd64" publicKeyToken="31bf3856ad364e35" language="neutral" versionScope="nonSxS" xmlns:wcm="http://schemas.microsoft.com/WMIConfig/2002/State" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

      <RunAsynchronous>

        <RunAsynchronousCommand wcm:action="add">

          <Path>cmd /c date 05-31-2022</Path>

          <Order>1</Order>

        </RunAsynchronousCommand>

      </RunAsynchronous>

    </component>

  </settings>

</unattend>

# Troubleshooting

If there are issues, there is a log on the deploy drive that will be created called Scriptlog.txt. That log can be useful in explaining why something failed. Make sure to only have one USB drive plugged into the device and only one hard disk. This process was not tested or validated with multiple USB drives and/or hard disks.

If you have a device that reboots immediately, chances are there are not any storage drivers available for that model. Most Intel-based devices should work fine as Windows has inbox drivers. AMD may not have available drivers. You likely will need to modify the WinPE media to include drivers for your model.