Imaging Windows 10 with MDT 2013 Update 2

This guide has been created using guidelines provided in the technet articles below. I advise reviewing the guides first to get an idea of the overall process. The purpose of using MDT is to be able to create a hardware-independent reference image that can be used to deploy a single image to any hardware (as long as the proper drivers have been injected). This creates a more flexible imaging solution that is easier to update and maintain.

Create a Widows 10 reference image: https://technet.microsoft.com/en-us/itpro/windows/deploy/create-a-windows-10-reference-image

Deploy a Windows 10 image using MDT 2013 Update 2: https://technet.microsoft.com/en-us/itpro/windows/deploy/deploy-a-windows-10-image-using-mdt

Windows 10 ADK can be downloaded from here: https://developer.microsoft.com/en-us/windows/hardware/windows-assessment-deployment-kit#adkwin10

MDT 2013 Update 2 can be downloaded from here: https://www.microsoft.com/en-us/download/details.aspx?id=50407

This article has good general information about the benefits of MDT and the difference between WDS and MDT: http://techthoughts.info/mdt-with-wds-integration-overview/

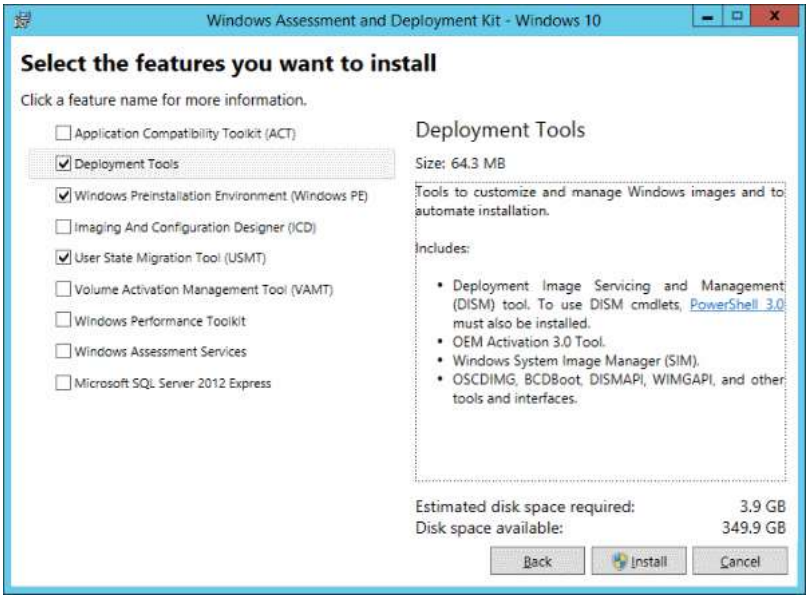
Initial Tool Installation

This guide assumes that WDS is already installed and configured. WDS will be used for the PXE portion of the imaging process. A boot image will be created in MDT and imported into WDS. If WDS is already installed, you can install MDT on the same server. If there is no imaging solution in place, determine what server should run MDT and WDS -- it should be Server 2012 R2 if possible.

1. Install MDT 2013 Update 2

   (5 minutes)

2. Install Windows 10 ADK. You only need ot select Deployment Tools, Windows PE, and USMT



   (15 minutes)

3. Launch Deployment Workbench from the Start menu. If for some reason there isn't in a shortcut it should be installed at "C:\Program Files\Microsoft Deployment Toolkit\Bin\DeploymentWorkbench.msc"

   This is the main MDT console

Creating the Reference Image
Set up the "MDT Build Lab" deployment share

Two deployment shares will be created - a build lab and a production deployment share. First, setup the MDT build lab

Refer to Set up the MDT build lab deployment share

It's a best practice to create a dedicate AD account used to authenticate to the MDT deployment shares. The password for this account is stored in the bootstrap.ini file in plain text, which is why it is advised to use a dedicated account and not the domain administrator.

1. Using the Deployment Workbench, right-click **Deployment Shares** and select **New Deployment Share**

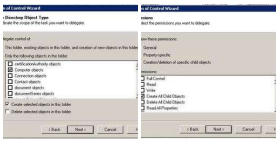
2. Use the following settings for the New Deployment Share Wizard
    - Deployment share path: E:\MDTBuildLab (or whatever drive is suitable)
    - Share name: MDTBuildLab$
    - Deployment share description: MDT Build Lab
    - <default>
    - Verify that you can access the \\server\MDTBuildLab$ share.
        - You may have to grant 'everyone' full control of the share

    (1 minute)

Create AD account for MDT and assign permissions

1. Create an AD Account (MDTUser for example) with a password set to never expire. Document this account in passportal.

2. Delegate rights to join computers to the domain.
    1. right-click on the domain root in ADUC
    2. All Tasks > Delegate Control
    3. Next
    4. Add
    5. Add the user (MDTUser for example)
    6. Select Create custom task to delegate and hit Next
    7. Select Only the following objects in the folder, check Computer objects, check the Create selected objects in this folder box, and press Next.
    8. Check the Create all child object box and press Next.



3. If possible, make the MDTUser a local administrator on the MDT server. If MDT resides on a DC, special considerations will need to be taken to give the account the proper permissions (think NTFS full control on the entire deployment share).

    I tried using the guide's advice and giving NTFS permissions to just the capture share but I ran into issues when PXE booting. I found that making the user an admin on the MDT server resolved the problem.

Add the Windows 10 installation files

1. Download the Windows 10 ISO from VLSC and extract the contents of the ISO to a directory on the server

    (30 minutes)

2. Mount the Windows 10 ISO

3. Create a "Windows 10" folder under Deployment Shares > MDT Build Lab > Operating Systems

4. Right-click on the Windows 10 folder and select Import Operating System
    - Full set of source files
    - Source Directory (browse to the mounted Windows 10 ISO DVD drive)
    - Destination directory name: W10PROX64RTM (This doesn't have to be exact but I'm using the guide's recommendation. The guide is using Windows 10 Enterprise, hence the "W10EX64RTM" name)

    (5 minutes)

Add Applications

Review the Add applications section of the technet article.

I have downloaded the installer packages for the Microsoft C++ runtimes and staged them in a folder structure per the guide. I have also created powershell scripts that can be used to import them into MDT. The files can be found on the X:\ drive at X:\Technical Group\Tech Tools\MDT\C++ Redistributable Packages for MDT.zip.

The scripts are hardcoded to use c:\Installers\ as the root folder for where the installers reside.

C++ Runtimes

1. Right-click MDT Build Lab -> Applications and select New folder. Create a folder named "Microsoft"

   (1 minute)

1. Copy the X:\Technical Group\Tech Tools\MDT\C++ Redistributable Packages for MDT.zip. to the MDT server and extract to c:\installers\. I.e. make sure that "C:\Installers\Install - Microsoft Visual C++ 2005 - x64" and subfolders for each of the other versions exist when you are done

   (5 minutes)

2. Connect to the deployment share using powershell using the following commands. The technet article is incorrect so use what is below.

   1. launch an elevated powershell window
   2. Import-Module "C:\Program Files\Microsoft Deployment Toolkit\bin\MicrosoftDeploymentToolkit.psd1"
   3. New-PSDrive -Name "MDT" -PSProvider MDTProvider -Root "E:\MDTBuildLab"
      - It's important to use "MDT" as the PSDrive name as the powershell scripts for the C++ runtime installers use that path
      - Modify "E:\MDTBuildLab" if MDT is intalled on a different drive"
   4. Set-ExecutionPolicy Unrestricted

   (5 minutes)

3. cd C:\Installers\Powershell Import Scripts

4. Run the following command in powershell to run all of the .ps1 files contained in the folder structure

   - ```
     Get-ChildItem -Recurse -include "*.ps1" | % { & $_ }
     ```

5. Refresh the "Microsoft" folder in MDT and you should now see all of the C++ runtimes added as applications

Office

1. Download the ISO for the version of Office that will be used in your reference image and extract the files to a directory. I am using Office 2016 Pro Plus x86 in this guide use this portion of the technet guide for reference.

   (30 minutes)

2. Right-click MDT Build Lab > Applications > Microsoft and select 'New Application'. Follow the wizard using the settings below.

   - Application with source files
   - Application Name: Install - Microsoft Office 2016 Pro Plus x86
   - Source directory: browse to the root directory of the Office setup files you extracted in step 1
   - Destination should be left as is (the name of the application)
   - Command line: setup.exe

   (2 minutes)

3. Right-click on the "Install - Microsoft Office 2016 Pro Plus x86" application and select properties

4. Click on the Office Products tab and select "Office Customization Tool..."

5. In the Office Customization Tool dialog box, select the Create a new Setup customization file for the following product option, select the Microsoft Office Professional Plus 2016 (32-bit) product, and click OK.
     ```

6. Configure the following settings and any other that are necessary for the site you are working with...

   Setup

   - Install Location and organization name
     - Organization name
   - Licensing and user interface
     - Use KMS Client Key if KMS is in place or enter the client's volume license key. The key can be imported later if necessary
     - check the "I accept the terms in the License Agreement" box
     - Display level: None
     - Suppress modal
   - Features
     - Modify user settings
       - Microsoft Office 2016 > First Run > Disable First Run Movie: Enabled
       - Microsoft Office 2016 > First Run > Disable Office First Run on application boot: Enabled
       - Microsoft Office 2016 > Privacy > Trust Center > Disable Opt-in Wizard on first run: Enabled

   Consider using the OCT to configure other settings like Trusted Locations, Trust Center, Protected View, etc... The settings above will allow for a silent installation and will not annoy users with the first time opt-in and welcome movie.

   (5 minutes)

7. Click File > Save in OCT and save the file in e:\MDTBuildLab\Applications\Install - Microsoft Office 2016 Pro Plus x86\Updates

   Use the naming recommendation in the technet article (starting the filename with 0) so it will process first in case there are other updates in the folder.

   Close OCT after saving

8. Click Ok in the "Install -Microsoft Office 2016 Pro Plus x86 Properties" dialog box

Create the reference image task sequence

refer to this section of the technet guide for more information

1. Right-click MDT Build Lab > Task Sequences and select 'New Task Sequence'. Use the following settings in the wizard.

   - Task sequence ID: REFW10X64-001
   - Task sequence name: Windows 10 Pro x64 RTM Default Image
   - Template: Standard Client Task Sequence
   - Select OS: Windows 10 Pro in W10PROX64RTM install.wim
   - Specify Product Key: Do not specify a product key at this time
   - Full Name: Use the client's Organization name if creating a client specific image. Use NetrixIT if creating an in-house image that will be copied to a client site later
   - Organization: Use the client's Organization name if creating a client specific image. Use NetrixIT if creating an in-house image that will be copied to a client site later
   - Internet Explorer home page: about:blank -- you can set this if you want or it can be configured later via group policy
   - Admin Password: Do not specify an Administrator password at this time

   (2 minutes)

Edit the Windows 10 task sequence

1. Right-click on the Windows 10 Pro x64 RTM Default Image task sequence and select Properties

2. Click on the Task Sequence Tab

3. Configure the settings as defined in the technet article here

    1. State Restore. Enable the Windows Update (Pre-Application Installation) action. **Note**
    Enable an action by going to the Options tab and clearing the Disable this step check box.

    2. State Restore. Enable the Windows Update (Post-Application Installation) action.

    3. State Restore. After the **Tattoo** action, add a new**Group** action with the following setting:
        - Name: Custom Tasks (Pre-Windows Update)
    4. State Restore. After Windows Update (Post-Application Installation) action, rename Custom Tasks to Custom Tasks (Post-Windows Update). **Note**
    The reason for adding the applications after the Tattoo action but before running Windows Update is simply to save time during the deployment. This way we can add all applications that will upgrade some of the built-in components and avoid unnecessary updating.

    5. State Restore / Custom Tasks (Pre-Windows Update). Add a new Install Roles and Features action with the following settings:

        1. Name: Install - Microsoft NET Framework 3.5.1
        2. Select the operating system for which roles are to be installed: Windows 10
        3. Select the roles and features that should be installed: .NET Framework 3.5 (includes .NET 2.0 and 3.0)

        Important
        This is probably the most important step when creating a reference image. Many applications need the .NET Framework, and we strongly recommend having it available in the image. The one thing that makes this different from other components is that .NET Framework 3.5.1 is not included in the WIM file. It is installed from the**Sources\SxS** folder on the media, and that makes it more difficult to add after the image has been deployed.

    6. State Restore - Custom Tasks (Pre-Windows Update). After the **Install - Microsoft NET Framework 3.5.1** action, add a new **Install Application** action with the following settings:

        1. Name: Install - Microsoft Visual C++ 2005 - x86
        2. Install a Single Application: Install - Microsoft Visual C++ 2005 - x86
    7. Repeat the previous step (add a new **Install Application**) to add the following applications:
        1. Install - Microsoft Visual C++ 2005 - x64
        2. Install - Microsoft Visual C++ 2008 - x86
        3. Install - Microsoft Visual C++ 2008 - x64
        4. Install - Microsoft Visual C++ 2010 - x86
        5. Install - Microsoft Visual C++ 2010 - x64
        6. Install - Microsoft Visual C++ 2012 Update 4 - x86
        7. Install - Microsoft Visual C++ 2012 Update 4 - x64
        8. Install - Microsoft Visual C++ 2013 - x86
        9. Install - Microsoft Visual C++ 2013 - x64
        10. Install - Microsoft Visual C++ 2015 - x86
        11. Install - Microsoft Visual C++ 2015 - x64
        12. Install - Microsoft Office 2013 Pro Plus - x86
        13. Add any other applications you've added that should be in the *reference* image. Please note that applications will be deployed to the production image later. Only very common applications that should be present in all configurations should go in the reference image.
    8. After the Install - Microsoft Office 2013 Pro Plus - x86 action, add a new Restart computer action.

    (15 minutes)

4. Add a suspend action

    - After State Restore - Install Applicaitons, add a new "Run Command Line" task
        - Name: Suspend
        - Command Line: cscript.exe "%SCRIPTROOT%\LTISuspend.wsf"

    This will be a good pausing point when creating the reference image where you can take a snapshot of the VM you're using to build the image before proceeding. That way, if something goes wrong or the reference image needs so adjusting, you can easily revert, make changes, and continue.

    (1 minute)

You can skip the "Edit the Unattend.xml file for Windows 10 Enterprise" section of the guide. We want to avoid modifying unattend.xml if possible and make the configuration change in MDT instead.

Configure MDT Build Lab deployment share Rules

MDT Rules are explained in this section of the guide.

1. Right-click on the MDT Build Lab deployment share and select Properties.

2. Click on the Rules tab and Replace the contents with the following settings. Modify the bold lines below to remove the notes I've added or put in the client specific information.

   [Settings]
   Priority=Default
   [Default]
   _SMSTSORGNAME=<Organization Name>
   UserDataLocation=NONE
   DoCapture=YES
   OSInstall=Y
   AdminPassword=MyPassword -- This will be the local administrator account password of the reference image
   TimeZoneName=Central Standard Time
   JoinWorkgroup=WORKGROUP
   HideShell=YES
   FinishAction=SHUTDOWN
   DoNotCreateExtraPartition=YES
   ApplyGPOPack=NO
   SLSHARE=\\MDTServer\MDTBuildLab$\Logs -- Enter the MDT server name and create the Logs folder in the MDTBuildLab directory to allow for centralized logging.
   SkipAdminPassword=YES
   SkipProductKey=YES
   SkipComputerName=YES
   SkipDomainMembership=YES
   SkipUserData=YES
   SkipLocaleSelection=YES
   SkipTaskSequence=NO
   SkipTimeZone=YES
   SkipApplications=YES
   SkipBitLocker=YES
   SkipSummary=YES
   SkipRoles=YES
   SkipCapture=NO
   SkipFinalSummary=YES

   (2 minutes)

3. Click on Edit Bootstrap.ini on the bottom right of the Rules tab and replace the contents with the following settings. Modify the bold lines below to remove the notes I've added or put in the client specific information.

   [Settings]
   Priority=Default
   [Default]
   DeployRoot=\\MDTServer\MDTBuildLab$ -- Replace MDTServer with the actual server name
   UserDomain=CONTOSO -- Replace with the domain of the MDT Server. This is used to connect the client to the share.
   UserID=MDTUser -- Use the account that you created and gave permissions to in the steps above.
   UserPassword=P@ssw0rd -- Replace with the MDTUser account's password
   SkipBDDWelcome=YES

   (2 minutes)

4. Right-click on the MDT Build Lab deployment share and select "Update Deployment Share". Click next through the wizard. This will generate the boot images that reside in e:\MDTBuildLab\boot

   (10 minutes)

Build the Windows 10 reference image

refer to this section of the technet guide.

Now it's time to actually create the reference image. The goal is to use the task sequence you've created in order to capture a new image with the applications and settings configured in the task sequence embedded in it. That image will then be used in other production task sequences to deploy to machines. In this guide, I will be using a VMWare guest to run the Task Sequence on and capture the image from. The following steps could be performed on a physical machine as well but using VMWare give us a lot more flexibility. If using a physical machine, try to pick the most bare-bones model available (i.e. a desktop instead of a laptop as there are less devices that need drivers)

Again, this guide assumes that WDS is already installed and functioning properly. If not, install and configure WDS (out of scope for this guide)

Create a VM

1. Connect to vCenter or the VMWare host

2. Create a new VM using the following considerations
   - Name the VM "Windows 10 Reference Image" or something similar, this is temporary
   - Operating System Version shouldn't matter but select Server 2012 (or 2012 r2 if listed in newer VMWare versions)
   - use the e1000 NIC as the drivers are already in the Windows 10 PE image. The Network will need be able to receive broadcasts from the PXE server
   - Thin provisioned disk -- 40 GB should suffice unless you are adding A LOT of data to the reference image (don't do this.)

   (5 minutes)

Import LiteTouchPE_x64.wim into WDS

1. Open WDS, right-click on Boot Images and select Add Boot Image

2. Assuming MDT Build Lab is installed at e:\MDTBuildLab, browse to e:\MDTBuildLab\boot\ and select LiteTouchPE_x64.wim. Click Open and Next.

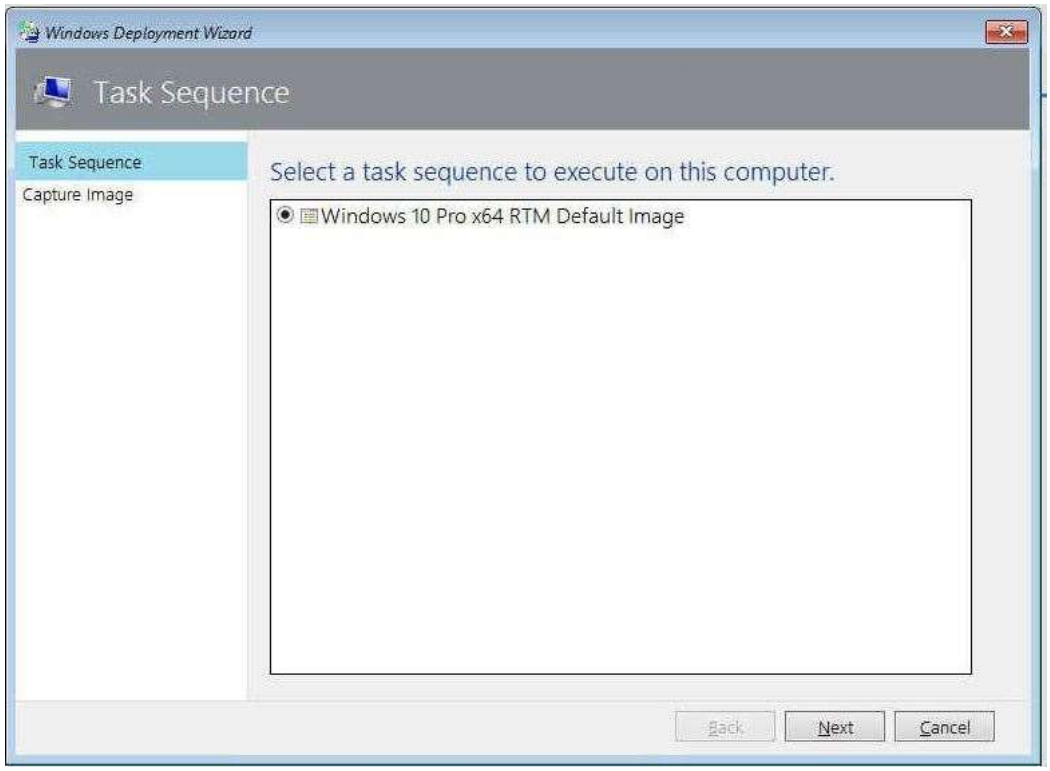
3. Rename the boot image MDT Build Lab x64
   - This will make it easy to differentiate between the Build Lab boot image and the production Deployment Share image later.

Boot the VM and Initiate the Task Sequence

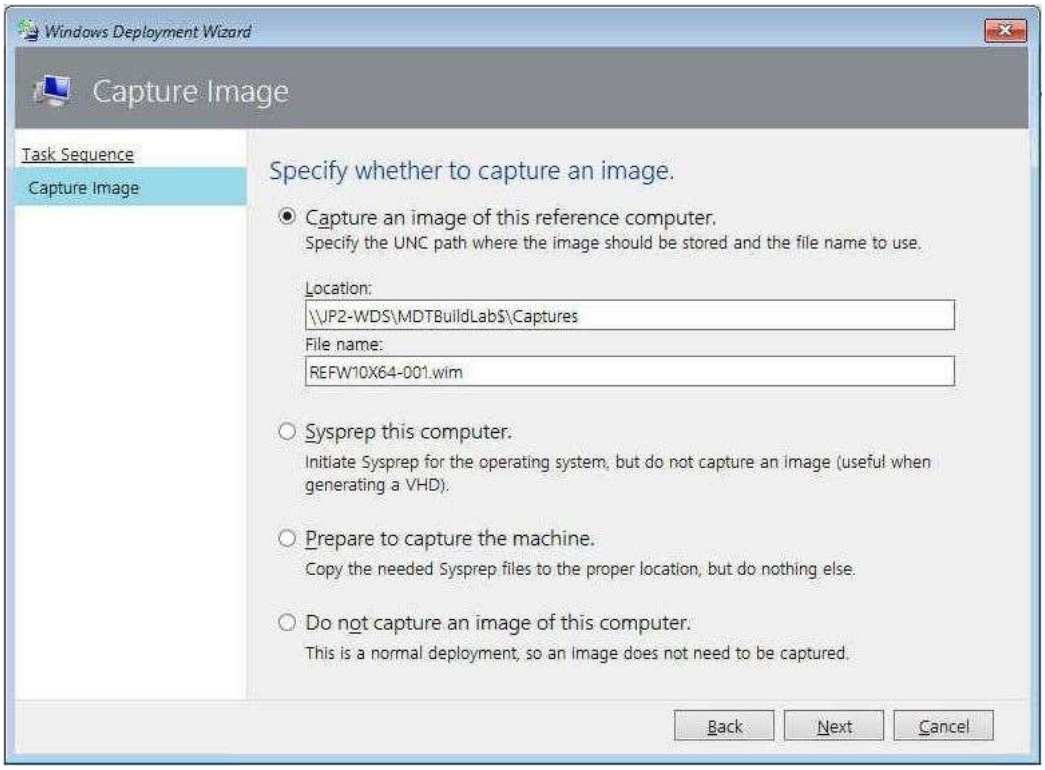
1. Power on the Windows 10 Reference Image VM and PXE boot to the MDT Build Lab x64 boot image.

   (2 minutes)

2. Select the Windows 10 Pro x64 RTM Deafult Image task sequence and click Next.



3. Keep the defaults on the next screen and hit Next. (Capture an image of this reference computer.)



(60 minutes)

The task sequence will now begin and install Windows 10 and the applications configured in the sequence. The machine will reboot and start Windows Updates. Eventually you will be presented with the Windows 10 desktop with the "Resume Task Sequence" shortcut on the desktop (assuming you added the Suspend step to the task sequence).

4. Create a snapshop of the machine in VMware named "pre MDT task sequence resume"

5. Make any changes to the reference image that are necessary. Install applications that are required for all users that don't need to be regularly updated. Make any changes to the user experience that are required. This step is optional and very dependent on the environment and goal of the imaging process. The more you add to the reference image, the more image shifts towards being a "thick image". The less you add, the more universal the image is and is more of a "think image". When possible, applications should be installed during the deploying task sequence later instead as opposed to being built into the reference image.

6. Double-click the "Resume Task Sequence" shortcut on the desktop. The machine will proceed with syspreping and capturing the image. When it is finished the machine will shut down and the image will be in E:\MDTBUildLab\Captures\

   (60 minutes)

Deploying a Windows 10 Image

This portion of the document follows the guidelines from this technet article

Configure AD permissions for the MDTUser Account

1. Configure AD Permissions for the MDTUser account. Refer to this section of the guide for more details
   - The Set-OUPermissions.ps1 script is attached to this document and here is a direct link as well.
   - Target the OU that computer accounts reside in. You may need to change the default computer container if it is still set to the built-in 'Computers' OU. You can also run this script against the built-in computers container.
     - This is accomplished using the redircmp command. See the "Redirecting CN=Computers to an administraotr-specified OU" section in this article.
   - You will need to run powershell as administrator and then run the following commands
     - Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Force
     - Import-Module ActiveDirectory
     - Set-OUPermissions.ps1 -Account MDT_JD -TargetOU "OU=Workstations,OU=Computers,OU=Contoso" -- Don't include the Domain, only the OU. If you use the full DN it will warn you, just remove the domain portion and it should work"
   - This can be done on multiple OUs

   (5 minutes)

Create the MDT Production Deployment Share

1. Right-click Deployment Shares and select New Deployment Share

2. Use the following settings for the New Deployment Share Wizard
    - Deployment share path: E:\MDTProduction (or whatever drive is suitable)
    - Share name: MDTProduction$
    - Deployment share description: MDT Production
    - <default>
    - Verify that you can access the \\server\MDTProduction$ share.

(1 minute)

Add the Windows 10 Pro x64 RTM custom image

1. Create a 'Windows 10' folder in Deployment Shares > Operating Systems > MDT Production

2. Right-click Deployment Shares > MDT Production > Operating Systems > Windows 10 and select Import Operating System.

3. Use the following settings in the Import Operating System Wizard
    - OS Type: Custom image file
    - Source file: E:\MDTBuildLab\Captures\REFW10X64-001.wim (modify the drive/name if necessary)
        - do not check the "move the files to the deployment share" box
    - Select "Copy Windows 7, Windows Server 2008 R2, or later setup files from the specified path.
        - Setup source directory: E:\MDTBuildLab\Operating Systems\W10PRO64RTM (modify the drive/name if necessary)
    - Destination directory name: W10PRO64RTM

(2 minutes)

Add applications

refer to this step in the technet guide. I'm not going to go into detail here as this was already covered when creating the reference image and the process is well documented. You will create applications for any program you want to install after applying the custom image. Silent installation switches will need to be used to ensure the process is automated.

Depending on the client and the programs needed in the image, this process could take 5 minutes or many hours to gather the installers, test silent installation, configure settings files, etc.

Create the driver repository

refer to this section of the technet guide.

1. Create an E:\Drivers folder

2. Create the following subfolders
    - E:\Drivers\WinPE x86\
    - E:\Drivers\WinPE x64\
    - E:\Drivers\Windows 10 x64\

3. Create a subfolder for each model that the image might be deployed to.

    - E:\Drivers\Windows 10 x64\Hewlett-Packard\HP EliteBook 8560w\
    - E:\Drivers\Windows 10 x64\Dell Inc.\Latitude E6440\
    - etc.

    You can create the full structure under "Windows 10 x64" and then click and drag folders in order to copy them to the "WinPE" folders, this will save some clicks.

    As the guide explains...

    The preceding folder names are selected because they match the actual make and model values that MDT reads from the machines during deployment. You can find out the model values for your machines via the following command in Windows PowerShell:

    ```
    Get-WmiObject -Class:Win32_ComputerSystem
    ```

    Or, you can use this commands in a normal command prompt:

    ```
    wmic computersystem get manufacturer
    wmic computersystem get model
    ```

    You should also be able to find what name should be by looking at the "Make" and "Model" fields in Servoyant if any machines of that particular model are in the platform. Servoyant uses the same wmic command to get scrap the model name.

    Newer HP models have "HP" as the manufacturer and older models are "Hewlett-Packard"

    The WinPE folders don't need to contain all the drivers, only the NIC and any other driver that may be required during the image deployment process. Windows 10 should contain plug-and-play drivers for everything except the NIC (in some cases you won't need to add NIC drivers either)

4. In MDT, right-click on Deployment Shares > MDT Production > Out-of-Box Drivers > and select New folder

    Create the same folder structure as you made in the E:\Drivers folder

5. Download all drivers for the model that are available from the manufactures website. HP typically has the full driver package available to download in one package or you can use the HP SoftPaq download manager. Otherwise, you will need to download all the drivers listed and extract all of the files into the E:\Drivers...\ subfolder for the model in question.

    Only put NIC drivers (if they are even necessary) in the WinPE folders. Wait until later to add these drivers as you can test the boot image and if everything works there is no need to inject drivers into the boot image.

    (60 minutes)

Create selection profiles for boot image drivers

1. Right-click Deployment Shares > MDT Production > Advanced Configuration > Selection Profiles and select New Selection Profile.

2. Use the following settings in the New Selection Profile Wizard

    - Seleciton Profile name: WinPE x86
    - Folders: Check to box next to Out-of-Box Drivers > WinPE x86

    repeat the process for WinPE x64.

    You do not need to create a selection profile for Windows 10 x64 as these profiles are strictly for boot images.

Create a task sequence to deploy the image

1. Create a Windows 10 Folder in Deployment Shares > MDT Production > Task Sequences

2. Right-click on the newly created Windows 10 folder and select New Task Sequence

3. Use the following settings for the new task sequence

    1. Task sequence ID: W10-X64-001
    2. Task sequence name: Windows 10 Pro x64 RTM Custom Image - Name this logically if you are going to have multiple task sequences. e.g. "Students" and "Teachers"
    3. Task sequence comments: Production Image
    4. Template: Standard Client Task Sequence
    5. Select OS: Windows 10 Pro x64 RTM Custom Image - This should be displayed as "REFW10X64-001DDrive in W10PRO64RTM REFW10x64-001.wim" or something similar
    6. Specify Product Key: Do not specify a product key at this time
    7. Full Name: <organization name>
    8. Organization: <organization name>
    9. Internet Explorer home page: about:blank
    10. Admin Password: Do not specify an Administrator Password at this time

    (2 minutes)

4. Right-click on the newly created Task Sequence and select Properties. Click on the Task Sequence tab and configure the following settings.

    - Preinstall. After the **Enable BitLocker (Offline)** action, add a **Set Task Sequence Variable** action with the following settings:

    1. Name: Set DriverGroup001
    2. Task Sequence Variable: DriverGroup001
    3. Value: Windows 10 x64\%Make%\%Model%

    - Configure the **Inject Drivers** action with the following settings:

    1. Choose a selection profile: Nothing
    2. Install all drivers from the selection profile **Note**
       The configuration above indicates that MDT should only use drivers from the folder specified by the DriverGroup001 property, which is defined by the "Choose a selection profile: Nothing" setting, and that MDT should not use plug and play to determine which drivers to copy, which is defined by the "Install all drivers from the selection profile" setting.

    - State Restore. Enable the **Windows Update (Pre-Application Installation)** action.
    - State Restore. Enable the **Windows Update (Post-Application Installation)** action.

Configure the Rules for the MDT Production deployment share

1. Create a Logs folder in E:\MDTProduction\ to use for centralized logging

2. Right-click the MDT Produciton deployment share and select Properties. Click on the Rules tab and Replace the contents with the following settings. Modify the bold lines below to remove the notes I've added or put in the client specific information.

   [Settings]
   Priority=Default
   [Default]
   _SMSTSORGNAME=<Organization Name>
   OSInstall=YES
   UserDataLocation=AUTO
   TimeZoneName=Central Standard Time
   AdminPassword=P@ssw0rd -- This will become the local administrator password
   JoinDomain=contoso.local -- replace with the full domain name
   DomainAdmin=MDTUser -- replace with the proper User
   DomainAdminPassword=Password for the MDTUser account

   DomainAdminDomain=CONTOSA.LOCAL -- replace with the proper domain
   MachineObjectOU=OU=Workstations,OU=Computers,OU=Contoso,DC=contoso,DC=com
   SLShare=\\MDTServer\MDTProduction$\Logs --replace with the actual server name
   ScanStateArgs=/ue: /ui:CONTOSO* -- replace contoso with the domain**
   USMTMigFiles001=MigApp.xml
   USMTMigFiles002=MigUser.xml
   HideShell=YES
   ApplyGPOPack=NO
   SkipAppsOnUpgrade=NO
   SkipAdminPassword=YES
   SkipProductKey=YES
   SkipComputerName=NO
   SkipDomainMembership=YES
   SkipUserData=YES
   SkipLocaleSelection=YES
   SkipTaskSequence=NO
   SkipTimeZone=YES
   SkipApplications=NO
   SkipBitLocker=YES
   SkipSummary=YES
   SkipCapture=YES
   SkipFinalSummary=NO

3. Click on Edit Bootstrap.ini and add replace the contents with the following settings

   [Settings]
   Priority=Default
   [Default]
   DeployRoot=\\MDTServer\MDTProduction$ -- replace with the actual server name
   UserDomain=CONTOSO -- replace with the domain
   UserID=MDTUser --replace with whatever the name of the user you created earlier to deploy images if necessary

   UserPassword=P@ssw0rd -- Replace with the MDTUser account's password
   SkipBDDWelcome=YES

4. In the MDT Production Properties box, select the Windows PE tab

5. Ensure the platform dropdown is set to x86 and change the following settings:

   1. Image description: MDT Produciton x86
   2. ISO file name: MDT Production x86.iso

6. Select the Drivers and Patches tab and change the Selection profile from "All Drivers and Packages" to **WinPE x86** and check the "Include all drivers from the selection profile" radio button.