A Geeks Guide to Windows 10 Deployment
- Contributing is everything…

By Johan Arwidmark

PUBLISHED BY

Deployment Artist
https://deploymentartist.com

Warning and Disclaimer

Every effort has been made to make this guide as complete and as accurate as possible, but no warranty or fitness is implied. The information provided is on an “as is” basis. The author and the publisher shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this manual.

Feedback Information

We’d like to hear from you! If you have any comments about how we could improve the quality of this guide, please don’t hesitate to contact us by visiting http://deploymentartist.com or sending an email to feedback@deploymentartist.com.

Module 1

A Geeks Guide to Windows 10 Deployment

Welcome to this guide

Thanks for downloading this geeks guide to Windows 10 Deployment. While this guide does cover some generic Windows 10 deployment topics, the guide is intended for ConfigMgr admins wanting to deploy and service Windows 10.

This guide contains a crash course in planning for Windows 10 deployments, and a series of step-by-step guides to perform the following Windows 10 deployment tasks.

- Use the OSD Builder community tool to service a Windows 10 image.
- Setup MDT Lite Touch to build and capture a Windows 10 reference image.
- Setup ConfigMgr (a.k.a. MEMCM) for OS Deployment
- Use ConfigMgr for bare metal deployment of Windows 10
- Use ConfigMgr to service/upgrade Windows 10 using task sequences

Credits

Special thanks go to Mikael Nystrom for as always valuable feedback, and to Ami Arwidmark who despite recovering from two broken arms, with extensive surgery involved, powered through the technical review of the guide. I also like to give a shoutout to David Segura for his fantastic OSD Builder solution that is featured in this guide. Finally, thank you Mamata Panda and Mohammed Nawaz Kazi for your feedback.

Download the sample files

The sample files for this guide are available for download at the following link. In my lab I downloaded them to C:\Setup on my Hyper-V host: https://bit.ly/AGGW10-SampleFiles

Lab Environment

In this guide I'm using a set of virtual machines, which are all part of the corp.viamonstra.com domain. If you want a quick path to setup a lab environment for this, I recommend using the Hydration Kit available here: http://bit.ly/HydrationKitWS2019

In the various modules in this guide, I'm using the following list of virtual machines. In my lab I'm using Windows Server 2019 as a server operating system, but all step-by-step guides also work with Windows Server 2016 (and even good old Windows Server 2012 R2).

- **DC01.** A Windows Server 2019 configured as DNS, DHCP, and Domain Controller.

- **MDT01.** A Windows Server 2019 configured as a File Server
- **CM01.** A Windows Server 2019 configured with SQL Server and ConfigMgr (SCCM)
- **REF001.** An "empty" virtual machine, used for a Windows 10 v2004 build and capture
- **PC0001.** An "empty" virtual machine, used for a Windows 10 v2004 bare metal deployment
- **PC0002.** A Windows 10 v1909 client that will be upgraded to Windows 10 v2004. This machine is also used for the OSD Builder module.

Virtual Machines - Details

To build a replica of the infrastructure used for this guide, you can run all virtual machines on a single Hyper-V or VMware host with 16 GB of RAM and at least 500 GB free disk space. However, if you can have 32 GB of RAM in your lab machine, ConfigMgr will thank you for it. ☺

As mentioned in the introduction, you can obviously use your own custom lab environment when working through the guides in this book, but if you want all guides and scripts to match exactly, I do recommend that you use the automated lab setup (hydration).

- **DC01.** A **Windows Server 2019** VM, fully patched with the latest security updates, and configured as Active Directory Domain Controller, DNS Server, and DHCP Server in the **corp.viamonstra.com** domain. Using Windows Server 2016, or even Windows Server 2012 R2 works fine too.
    - Server name: **DC01**
    - IP Address: **192.168.1.200**
    - Roles: **DNS**, **DHCP**, and **Domain Controller**
    - Disk Volumes: **C: (100 GB dynamic disk)**
    - vCPU count: **2**
    - RAM: **2 GB**
- **MDT01.** A **Windows Server 2019** VM, fully patched with the latest security updates, and configured as a member server in the **corp.viamonstra.com** domain. Using Windows Server 2016, or even Windows Server 2012 R2 works fine too.
    - Server name: **MDT01**
    - IP Address: **192.168.1.210**
    - Roles: **File Server**
    - **Software: MDT 8456**
    - Disk Volumes: **C: (100 GB dynamic disk),** and **E: (300 GB dynamic disk),**
    - vCPU count: **2**
    - RAM: **4 GB**

- **CM01.** A **Windows Server 2019** VM, fully patched with the latest security updates, and configured as a member server in the **corp.viamonstra.com** domain. Using Windows Server 2016, or even Windows Server 2012 R2 works fine too.
    - Server name: **CM01**
    - IP Address: **192.168.1.214**
    - Roles: **File Server** and **WDS**
    - Software: **SQL Server 2017**, **Windows ADK 10 v2004**, and **ConfigMgr 2002**
    - Disk Volumes: **C: (100 GB dynamic disk),** and **E: (300 GB dynamic disk),**
    - vCPU count: **2**
    - RAM: **16 GB**
- **PC0001.** A blank VM, used for bare metal deployment in module 5.
    - Disk Volumes: **C: (100 GB dynamic disk)**
    - vCPU count: **2**
    - RAM: **2 GB**
- **PC0002.** A Windows 10 1909 VM, used for OSD Builder in module 3, and for Inplace upgrade / servicing in module 6.
    - Client name: **PC0002**
    - IP Address: **DHCP**
    - Disk Volumes: **C: (100 GB dynamic disk)**
    - vCPU count: **2**
    - RAM: **2 GB**
- **REF001.** A blank VM, used for creating reference images in module 4.
    - Disk Volumes: **C: (100 GB dynamic disk)**
    - vCPU count: **2**
    - RAM: **2 GB**

Internet Access

In terms of providing Internet access to your VMs, I highly recommend putting all the VMs on an isolated (internal) virtual network/switch, and then use the NAT feature in either Hyper-V or VMware to have the machines accessing Internet. Learn more on the links below:

- How to enable Hyper-V NAT: https://bit.ly/Hyper-VNAT

For more advanced networking you can use virtual routers with two or more net cards. For example, running either Windows Server, or pfSense.

- How to setup a Windows Server virtual router: https://bit.ly/WSVirtualRouter
- How to setup a pfSense virtual router: https://bit.ly/pfSenseVirtualRouter

Module 2

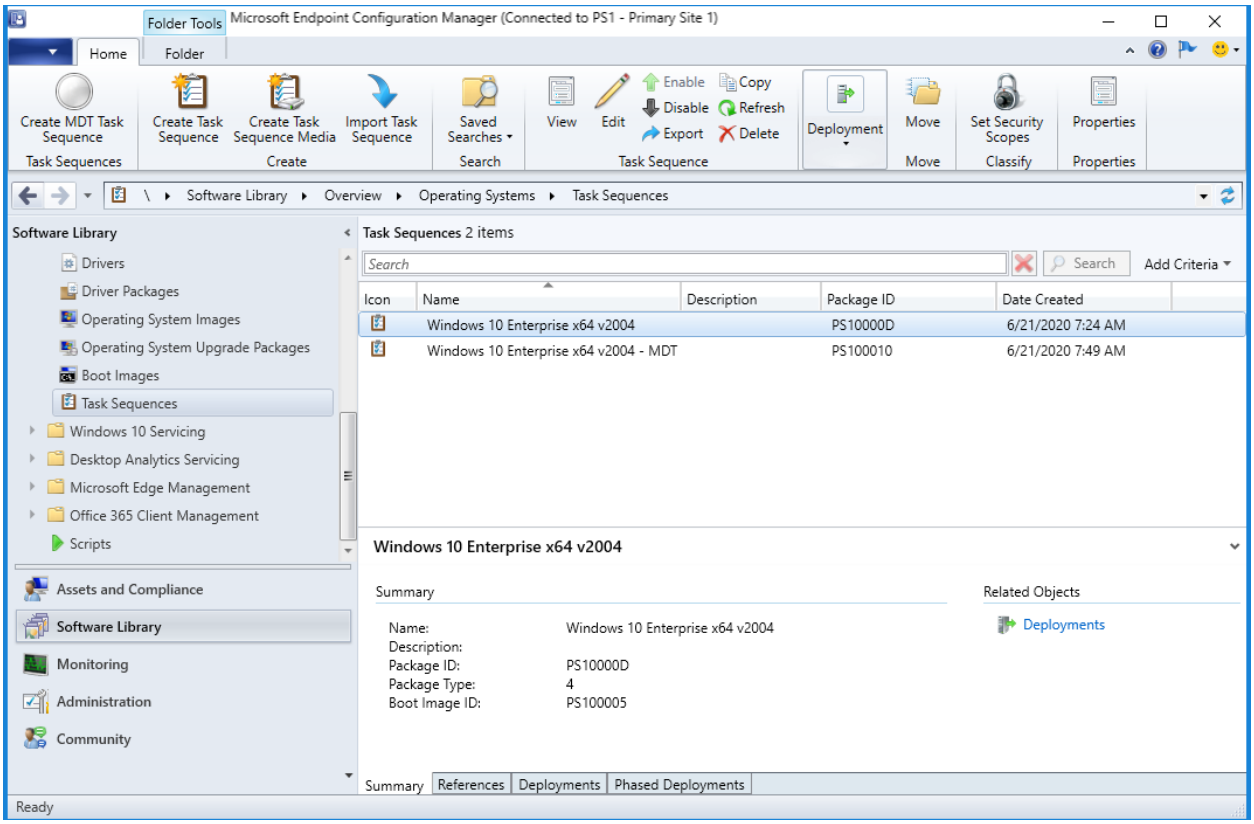
Planning for Windows 10 Deployment

Windows 10 Deployment

In general, if you know how to deploy Windows 7 or Windows 8.1, you are in a decent shape for deploying Windows 10. From a core setup point of view there are not too many changes, but there are a few challenges you need to be aware of: Like putting new testing processes in place because of Windows 10 servicing, as well as reducing the network impact Windows 10 will have on your environment by using peer to peer solutions like Peer Cache, BranchCache, and Delivery Optimization, Microsoft Connected Cache, and bandwidth control techniques such as BITS and LEDBAT.

Other challenges are for example keeping up with updating group policy templates, as well as cleaning up old policies that are no longer applicable.

The two most popular platform for deploying Windows 10 are Microsoft Deployment Toolkit (MDT), and Microsoft Endpoint Configuration Manager (ConfigMgr). MDT is primarily used for organizations not having ConfigMgr implemented, and need to do Windows 10 deployment and servicing. MDT can also be integrated into ConfigMgr, providing extra (optional) imaging features.



The ConfigMgr console showing some OSD task sequences.

Assessment and Readiness Tools

To prepare for Windows 10 deployments, most of the "good old" planning tools for Windows deployment, MAP, ACT and Upgrade Readiness are no longer used, and have been replaced by a cloud service named Desktop Analytics. This service, unlike its predecessor Upgrade Readiness, is not free and require a user-based E3/E5 or A3/A5 licensing agreement.

Windows ADK 10 contains the core deployment tools that MDT and ConfigMgr are using in the background. This kit is mostly based on command line tools but also provides interfaces for developers to create their own tools. In addition to Windows ADK 10, you also need the Windows ADK 10 WinPE Addon, which is used for all types of 10 deployments except servicing.

Deployment Scenarios

Windows 10 supports the following deployment scenarios, and you'll learn how to setup two of them in this guide (New Computer and Inplace Upgrade):

- **New Computer.** This scenario is also known as bare metal deployment, and is used when you have a blank machine you need to deploy, or an existing machine you want to wipe and redeploy without caring about any existing data. The setup usually starts from a boot media via USB or PXE, but also supports standalone media for deployments in location with poor network connectivity.
- **Computer Refresh.** Sometimes called wipe-and-load, and is primarily used to fix broken machines, but also to "upgrade/service" machines that cannot be serviced via the Inplace upgrade process. For example, when you want to go from Windows 10 x86 to Windows 10 x64. This guide does not cover this scenario.
- **Computer Replace.** A computer replace is quite much like the refresh scenario. But since you are replacing the machine, this scenario is divided into two main tasks: Backup of the old client, and bare metal deployment of the new client. As with the refresh scenario, user data and settings are backed up and restored. This guide does not cover this scenario.
- **Inplace Upgrade.** These days, Inplace upgrade scenarios are primarily used to upgrade Windows 10 from one Windows 10 version to the next, even though recent spring releases like Windows 10 v2004 are changing that a bit. Starting with 1903, servicing a spring release to the same fall release has just been applying an update instead of a whole new image.

Module 3

Creating Great Images – Part 1

To succeed in with Windows 10 Deployment and Servicing, the image quality certainly applies. If deployment speed is not so important to you, after all new hardware and good network often compensates quite a bit for speed, you can certainly use a single thin image for all your deployments.

If going the single image route, that image must be serviced offline, otherwise it cannot be used for Windows 10 servicing. If you need a thicker image for deployment speed, meaning an image with applications in it, you will need to maintain two images. One, that you use for New Computer, Computer Refresh, and Computer Replace scenarios, and a second one that you use for Windows 10 Servicing.

In this module you learn to create a thin image using the OSD Build community tool that can be used for all Windows 10 deployment scenarios. The resulting image will later be imported into ConfigMgr for deployment and servicing.

What you need for this module

For the guides in this module, you need the following:

- A Windows 10 client with Internet access. In my lab I was using the PC0002 virtual machine which is a Windows 10 Enterprise v1909 machine.
- A Windows 10 v2004 ISO copied to the C:\ISO folder on the PC0002 client.

Create an updated OS Media using OSD Builder

OSD Builder is being developed by David Segura, and has been a fantastic tool to keep thin images updated.

Here is a guide for using the OSD Builder solution to create a better OS installation media that includes .NET Framework. When using OSD Builder, it's recommended to run the same version or newer than the OS Image you are updating.

Install OSD Builder

1. On **PC0002**, login in as administrator. If you use the Hydration Kit to setup your lab, the default password for all accounts is **P@ssw0rd**.
2. Allow PowerShell to run script by running the following command in an elevated PowerShell prompt:

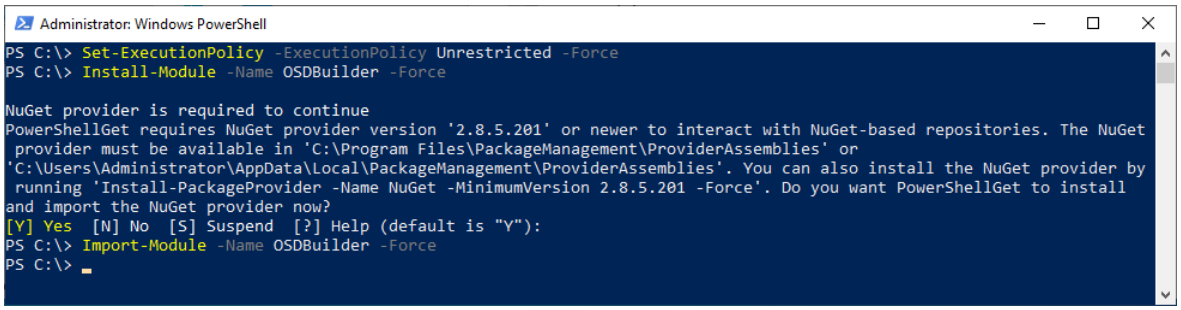
   ```
   Set-ExecutionPolicy -ExecutionPolicy Unrestricted -Force
   ```

3. Install the OSD Builder by running the following command in an elevated **PowerShell prompt** (allow the installation of the Nuget provider):

   ```
   Install-Module -Name OSDBuilder -Force
   ```

4. Import the OSD Builder module by running the following command in an elevated PowerShell prompt:

```
Import-Module -Name OSDBuilder -Force
```



Installing and importing the OSD Builder module.

Import OS Media, Run the Update, and Build a new Media

This process imports Windows 10 Enterprise, and updates the media with the latest updates. You are also enabling .NET Framework 3.5 in the image since many applications still needs it. Again, to follow this guide, you need the Windows 10 Business Editions x64 v2004 media downloaded.

This media can be downloaded from either https://my.visualstudio.com/ or http://microsoft.com/vlsc depending on your license agreements. There is also a free 90 days trial of Windows 10 Enterprise v2004 available on the Microsoft Evaluation Center: https://www.microsoft.com/en-us/evalcenter/evaluate-windows-10-enterprise

Here follow the steps to import, update and build a new media:

1. On **PC0002**, download (or copy) the **Windows 10 Business Editions x64 v2004** media to **C:\ISO**. In my lab I renamed the download file to **Windows 10 Business Editions x64 v2004.iso**.

2. Mount the **C:\ISO\Windows 10 Business Editions x64 v2004.iso** file by either double-clicking it, or by running the following command in an elevated PowerShell prompt:

```
Mount-DiskImage -ImagePath
"C:\ISO\Windows 10 Business Editions x64 v2004.iso"
```

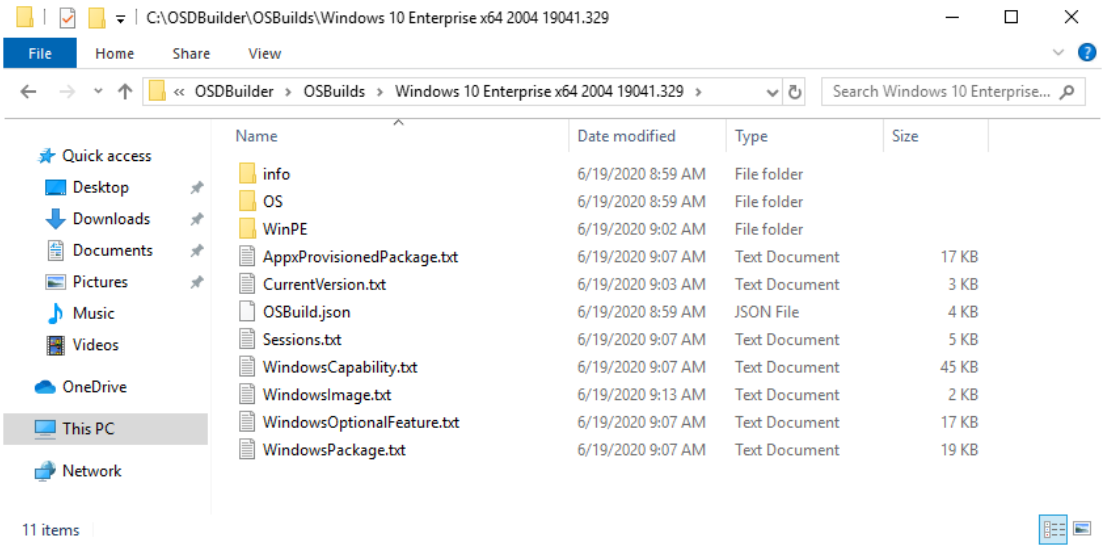
3. Import, update and build the OS Media by running the following command:

```
Import-OSMedia -ImageName 'Windows 10 Enterprise' -SkipGrid
-Update -BuildNetFX
```

Note: The preceding process takes anywhere from 30 minutes to several hours to complete, depending on your hardware. The new build content will not be available until this process completes.

4. Review the build content created in the **C:\OSDBuilder\OSBuilds** folder, especially the install.wim file located in **C:\OSDBuilder\OSBuilds\Windows 10 Enterprise x64 2004 19041.329\OS\sources**.

Note: The version number of the Windows 10 Enterprise x64 2004 19041.329 folder will be different depending on what cumulative update that is injected.



Sample structure from a Widows 10 Enterprise v2004 build.

Module 4

Creating Great Images – Part 2

In this module you learn to create a thick image using the MDT Litetouch build and capture process. The resulting image will later be imported into ConfigMgr for standard deployment scenarios only. As you might remember from the previous module, you cannot use an image that has been built and captured for Windows 10 servicing. It must be an image that has been serviced offline.

Real World Note: While thick images, meaning images with applications in them, still have their uses. I see more and more organizations using a single thin image, and have the task sequence install needed application etc. during deployment. If you don't need thick images in your environment, you can safely skip to Module 5 instead.

What you need for this module

For the guides in this module, you need the following:

- A Domain Controller running either Windows Server 2012 R2, Windows Server 2016 or Windows Server 2019. In my lab I used a domain controller named DC01 that manages the corp.viamonstra.com domain (VIAMONSTRA is the NetBIOS name).
- A Member Server running either Windows Server 2012 R2, Windows Server 2016 or Windows 2019. In my lab the Member Server is named MDT01, and it's a member of the corp.viamonstra.com domain.
- A Windows 10 v2004 ISO copied to the E:\ISO folder on MDT01. I recommend using the updated Windows 10 v2004 build folder that was created in the previous module, but you can also use a Windows 10 v2004 ISO that you download from MSDN or Microsoft VLSC.
- Application(s) that you want to add to your image during build and capture. For simplicity I added Adobe Reader DC as an example.
- The sample files for this guide: https://bit.ly/AGGW10-SampleFiles

Why using MDT Lite Touch?

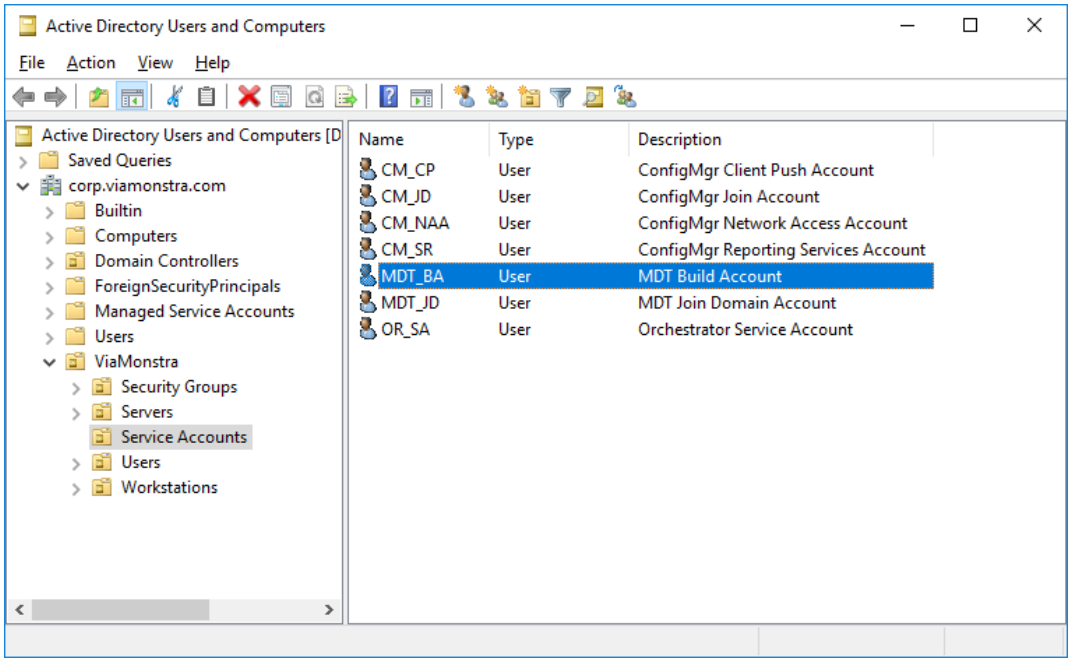
A great question to ask is the following: Since we are going to deploy the image with ConfigMgr, why don't we use ConfigMgr to also create the reference image? Well, it boils down to this:

- Using MDT Lite Touch is simply faster. This means less hours spent (in total) creating and maintaining the image.
- Installing Software Updates is extremely reliable in MDT Touch (the equivalent action in a ConfigMgr task sequence has been quite shaky over the years)
- You can use a Suspend action that allow for reboots, useful when needing to installing/configure something manually, or just want to check the reference image before it's automatically captured.
- The MDT Lite Touch build and capture is easier to automate.

Step 1 – Creating the MDT server structure

Review the service account for MDT builds

1. On **DC01**, using **Active Directory Users and Computers**, make sure you have created a user account named **MDT_BA**. In my lab I added that account to the **ViaMonstra / Service Accounts** OU, and I assigned a password of **P@ssw0rd**.



The Service Accounts OU in my lab.

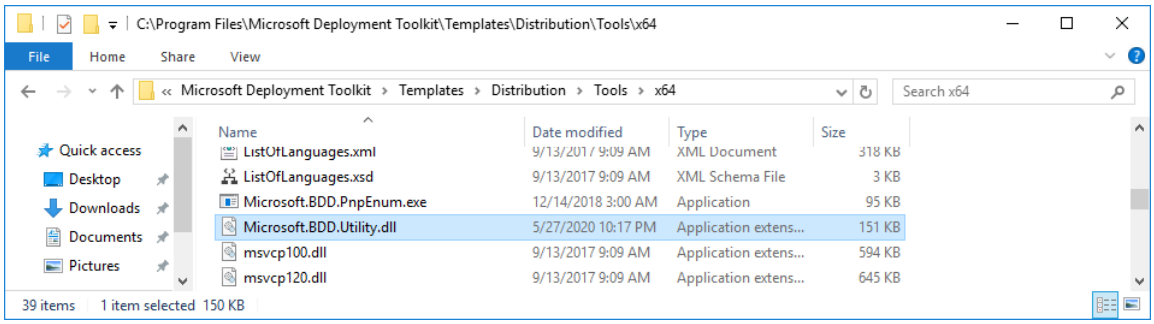
Install MDT 8456

1. On **MDT01**, log on as an administrator.
2. Download **MDT 8456** from https://aka.ms/mdtdownload and install it using the default settings.

Install the MDT 8456 Hotfix

Due to a code change in Windows ADK 10 v2004, and Windows 10 v2004 you need to download some updated files for MDT 8456. You can download these files here: https://bit.ly/MDT8456HF

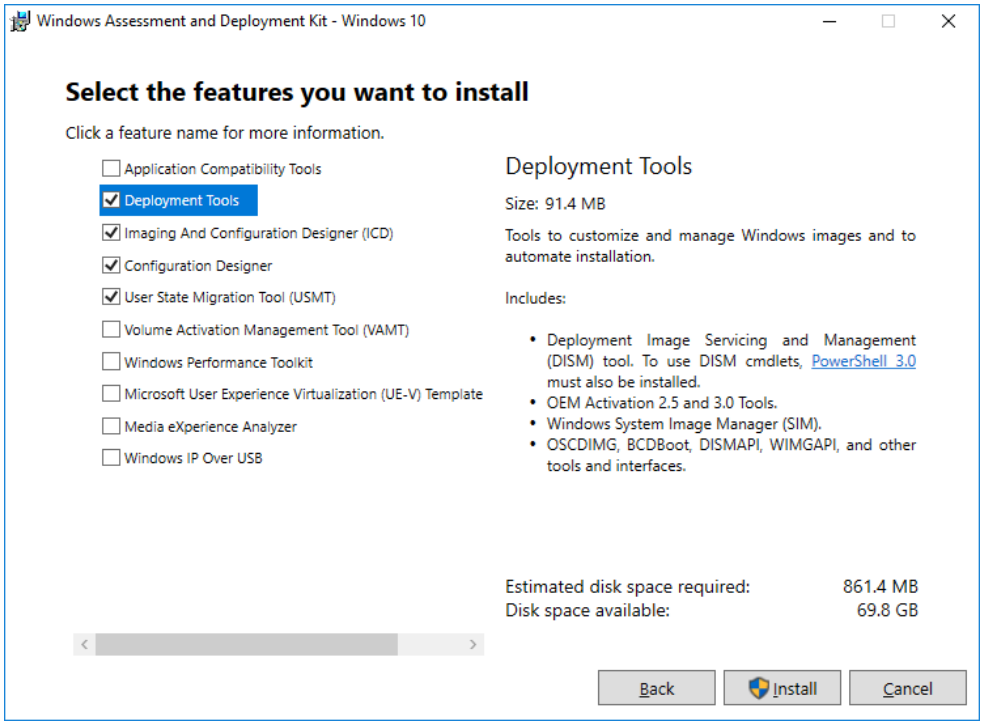
1. On **MDT01**, download the **MDT 8456 hotfix** (MDT_KB4564442.exe), and extract it to a folder named C:\Setup\MDT 8456 Update (create the folder).
2. Copy the x86 version of the new **Microsoft.BDD.Utility.dll** from **C:\Setup\MDT 8456 Update\x86** to **C:\Program Files\Microsoft Deployment Toolkit\Templates\Distribution\Tools\x86**. Replace the existing file.
3. Copy the x64 version of the new **Microsoft.BDD.Utility.dll** from **C:\Setup\MDT 8456 Update\x** to **C:\Program Files\Microsoft Deployment Toolkit\Templates\Distribution\Tools\x64**. Replace the existing file.



The updated Microsoft.BDD.Utility.dll added to the MDT installation folder.

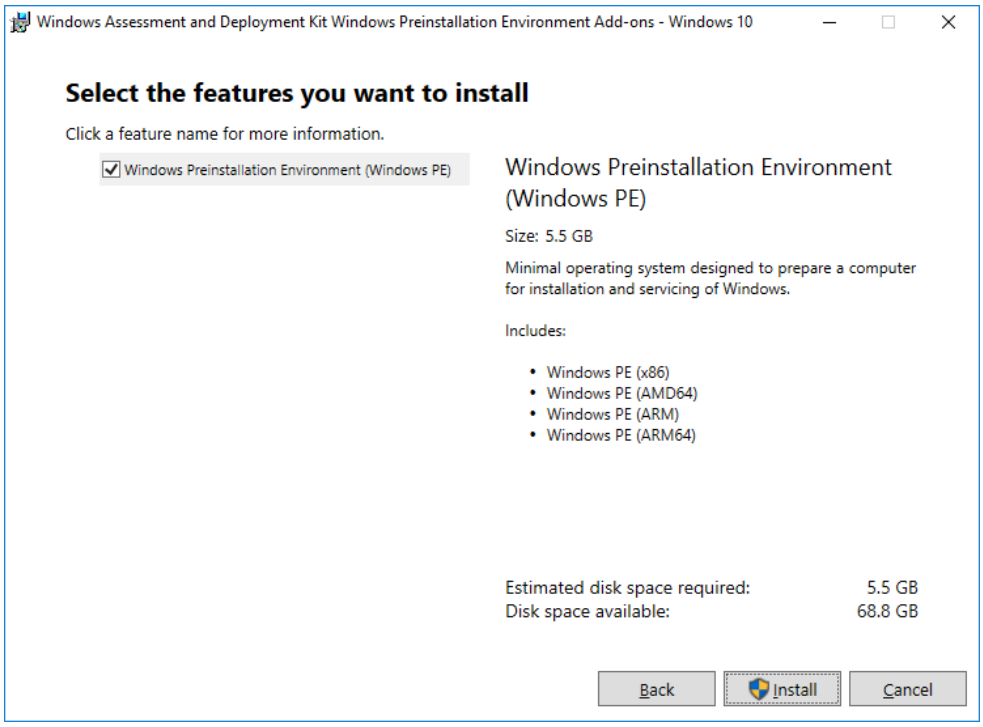
Install Windows ADK 10 v2004 and WinPE Addon

1. On **MDT01**, download and install **Windows ADK 10 v2004** from https://aka.ms/adk
2. Select the following features during setup:
    - Deployment Tools
    - Imaging and Configuration Designer (ICD)
    - Configuration Designer
    - User State Migration Tool (USMT)



The Windows ADK 10 v2004 setup.

3. Download and install **Windows ADK 10 v2004 WinPE Addon** from https://aka.ms/adk
4. Use the default settings during setup:

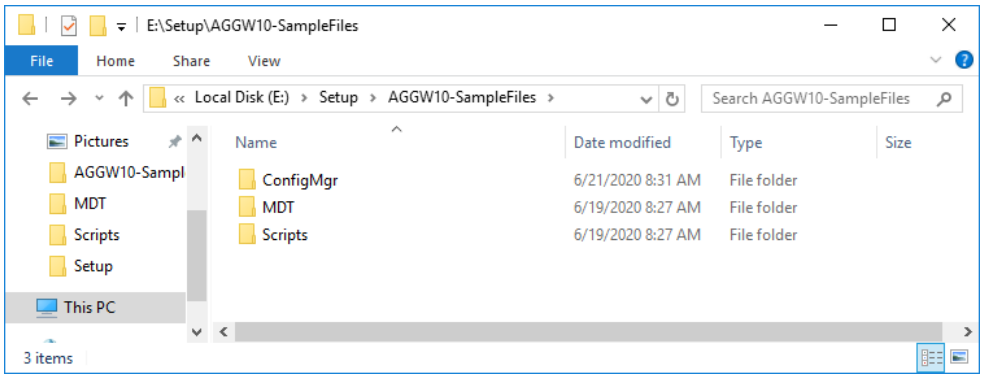


Installing the WinPE Addon for Windows ADK 10.

Copy the Sample Files to MDT01

In this section I assume you have downloaded the sample files for this guide.

1. On **MDT01**, create the **E:\Setup** folder.
2. Extract the sample files to **E:\Setup**
3. Review the content in the **MDT**, and **Script** folders (the ConfigMgr folder is for module 5).



The E:\Setup\AGGW10-SampleFiles folder.

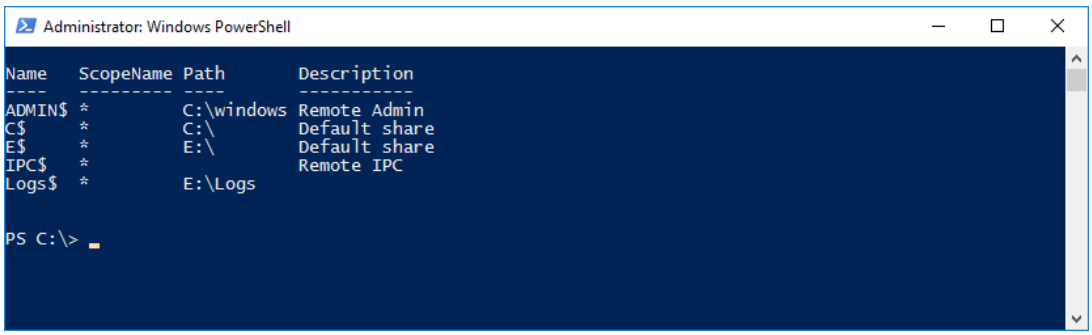
Create and share the Logs folder

1. On **MDT01**, start an elevated **PowerShell** prompt (run as administrator).
2. Create and share the **E:\Logs** folder by running the following command from an elevated PowerShell prompt:

   ```
   E:\Setup\AGGW10-SampleFiles\Scripts\Create-MDT01LogsFolder.ps1
   ```

3. Verify that the **Logs$** share was created by running the following command:

   ```
   Get-SmbShare
   ```



The logs folder created, and shared.

Create the MDT Build Lab Deployment Share

1. On **MDT01**, using the **Deployment Workbench** (available on the start menu), right-click **Deployment Shares** and select **New Deployment Share**. Use the following settings for the **New Deployment Share Wizard**.
   a. Deployment share path: **E:\MDTBuildLab**
   b. Share name: **MDTBuildLab$**
   c. Deployment share description: **MDT Build Lab**
   d. Options: **<default settings>**

Configure Permissions for the MDT Build Lab deployment share

MDT by default locks down the deployment share so that only administrators can access it, which is a bit too harsh.

1. On **MDT01**, start an elevated **PowerShell** prompt (run as administrator).
2. Allow the **MDT_BA** account **Modify** permissions (NTFS Permissions) to the **E:\MDTBuildLab\Captures** folder by running the following command:

   ```
   E:\Setup\AGGW10-SampleFiles\Scripts\
   Set-MDTBuildLabPermissions.ps1
   ```

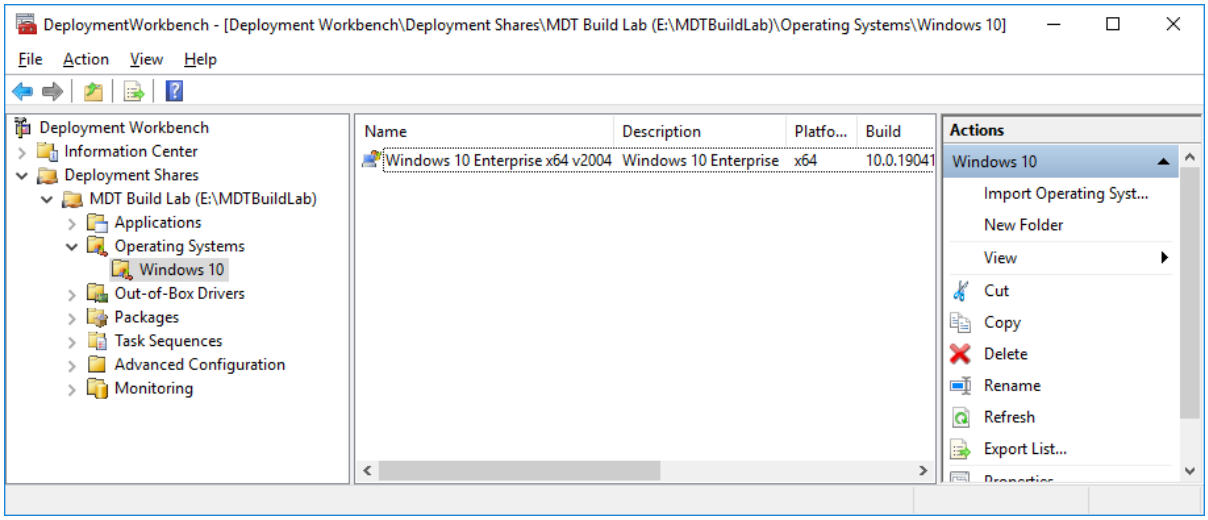
3. Verify that you can access **E:\MDTBuildLab** folder, and the **\\MDT01\MDTBuildLab$** share.

Step 2 – Add Windows installation files

When adding operating system images, always try to use a media that is already updated by Microsoft, or by you using the OSD Builder community tool. This way you don't need to install software updates during build and capture, which will save some time.

Import the Windows 10 operating system

1. On **MDT01**, download (or copy) the **Windows 10 Business Editions x64 v2004** media to **E:\ISO**. In my lab I renamed the download file to **Windows 10 Business Editions x64 v2004.iso**.
2. Mount the **E:\ISO\Windows 10 Business Editions x64 v2004.iso** media.
3. Using the **Deployment Workbench**, expand the **Deployment Shares** node, expand **MDT Build Lab**, select the **Operating Systems** node and create a folder named **Windows 10**.
4. Right-click the **Windows 10** node, and select **Import Operating System**. Use the following settings for the **Import Operating System Wizard**.
    a. **Full set of source files**
    b. Source directory: **D:** (or whatever drive letter the ISO mounted as)
    c. Destination directory name: **W10X64-V2004**
5. After adding the operating system, using the **Deployment Workbench**, in the **Windows 10** node, delete all images except for **Windows 10 Enterprise in W10X64-V2004 install.wim**.
6. Rename the remaining operating system to **Windows 10 Enterprise x64 v2004**.



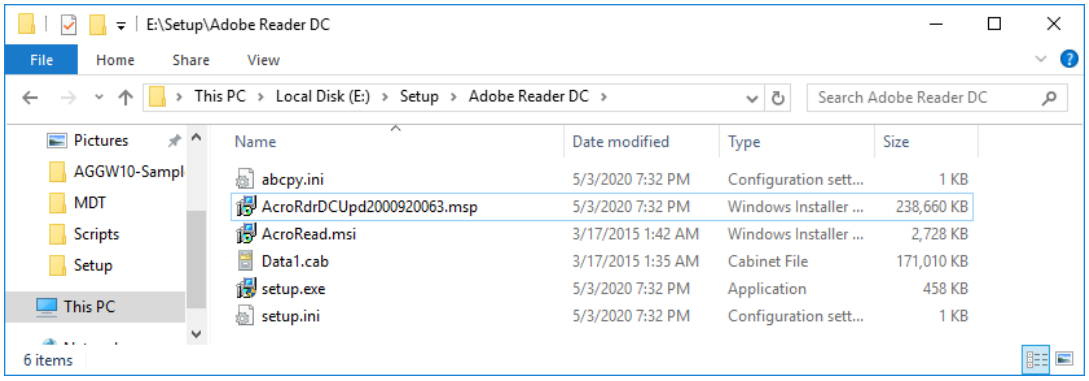
The Windows 10 node after renaming the label for the imported operating system.

Step 3 – Add Applications

In this step you are using Adobe Reader DC as example of adding an application to MDT. You can download this from: https://get.adobe.com/reader/enterprise/

Add Adobe Reader DC

1. On **MDT01**, download and install **7-Zip** from https://www.7-zip.org
2. Download **Adobe Reader DC** from https://get.adobe.com/reader/enterprise/ and extract it using **7-Zip** to the **E:\Setup\Adobe Reader DC** folder (create the folder).

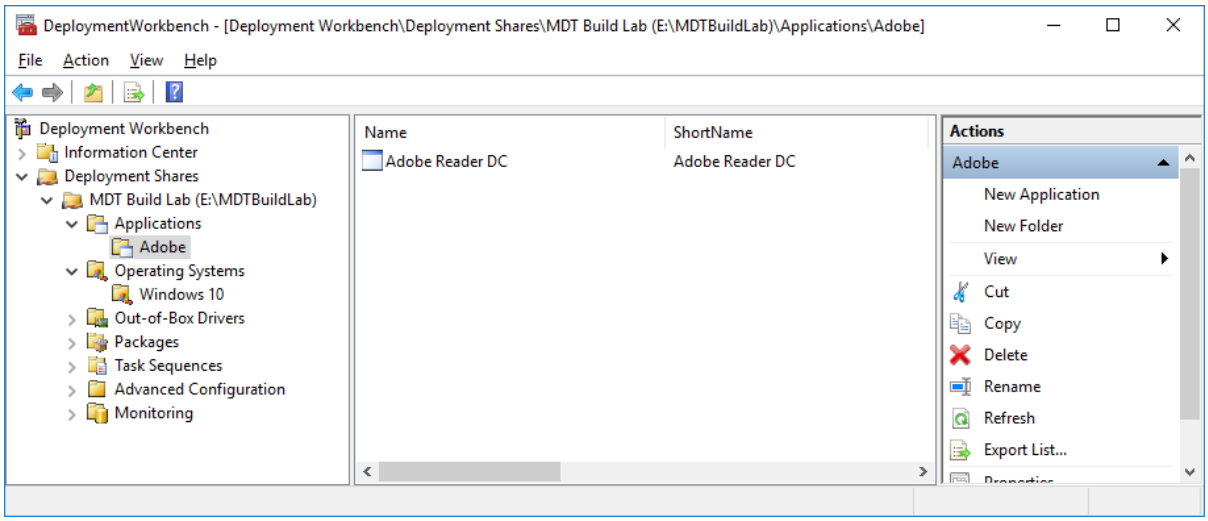


Adobe Reader DC extracted using 7-Zip to E:\Setup\Adobe Reader DC.

3. Using the **Deployment Workbench**, expand the **Deployment Shares** node, expand **MDT Build Lab**, select the **Applications** node and create a folder named **Adobe**.
4. Expand the **Applications** node, right-click the **Adobe** folder, and select **New Application**, Use the following settings for the **New Application Wizard**.
    a. Application with source files
    b. Publisher: **<blank>**
    c. Application name: **Adobe Reader DC**
    d. Version: **<blank>**
    e. Source Directory: **E:\Setup\Adobe Reader DC**
    f. Specify the name of the directory that should be created:

        E:\Setup\Adobe Reader DC

    g. Command Line: **Setup.exe /sALL /rs /msi EULA_ACCEPT=YES**
    h. Working directory: **.\Applications\Adobe Reader DC**



The Adobe Reader DC application added.

Step 4 – Create the MDT Task Sequence

Create and configure a Task Sequence

1. On **MDT01**, using the **Deployment Workbench**, in the **MDT Build Lab** deployment share, select the **Task Sequences** node, and create a folder named **Windows 10**.
2. Expand the **Task Sequences** node, right-click on the **Windows 10** node, and select **New Task Sequence**. Use the following settings for the **New Task Sequence Wizard**.
    a. Task sequence ID: **REFW10-X64-001**
    b. Task sequence name: **Windows 10 Enterprise x64 v2004**
    c. Template: **Standard Client Task Sequence**
    d. Select OS: **Windows 10 Enterprise x64 v2004**
    e. Specify Product Key: **Do not specify a product key at this time**
    f. Full Name: **ViaMonstra**
    g. Organization: **ViaMonstra**
    h. Internet Explorer home page: **about:blank**
    i. **Do not specify an Administrator password at this time**

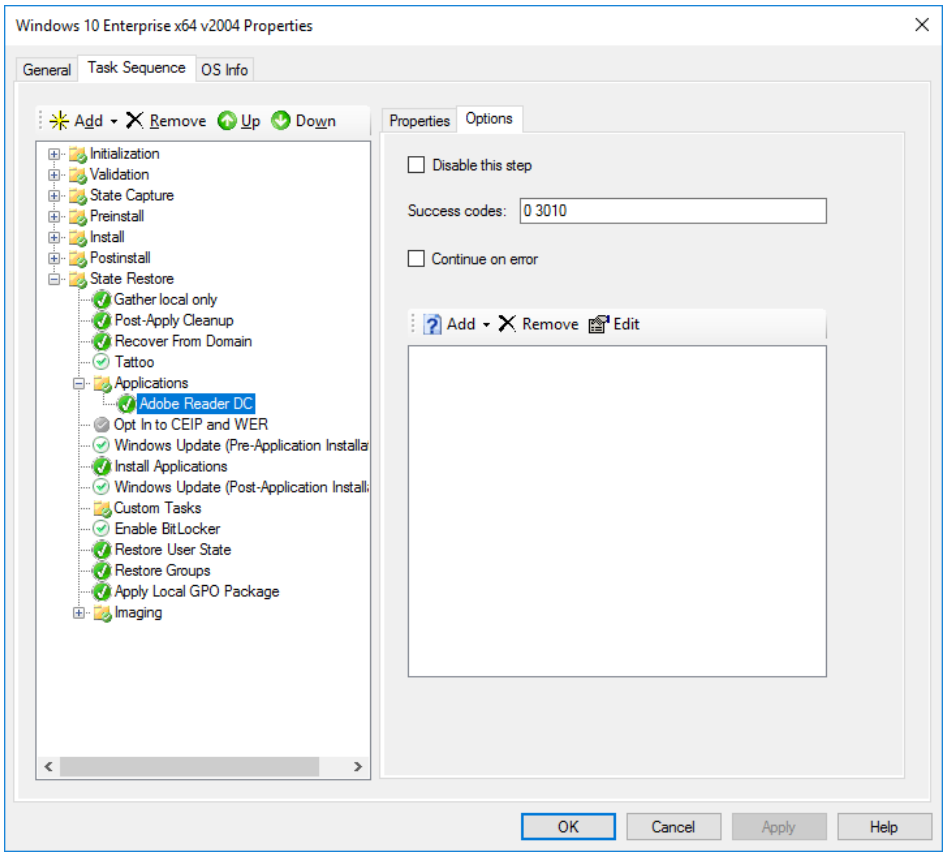
Edit the Task Sequence

1. In the **Task Sequences / Windows 10** folder, right-click the **Windows 10 Enterprise x64 v2004** task sequence, and select **Properties**.
2. On the **Task Sequence** tab, configure the **Windows 10 Enterprise x64 v2004** task sequence with the following settings:
    a. State Restore. After the **Tattoo** action, add a **new Group** action with the following setting:

        Name: **Applications**

    b. State Restore / Applications. Add a new **Install Application** action with the following settings:
        - Name: **Adobe Reader DC**
        - Install a Single Application: **Adobe Reader DC**
3. Click **OK**.

Real World Note: If your imported OS media is not already updated, or if you are adding Windows features during build and capture, you also want to enable the two Windows Update actions in the MDT task sequence. They are disabled by default.



The task sequence with an application added, and the Windows Update actions enabled.

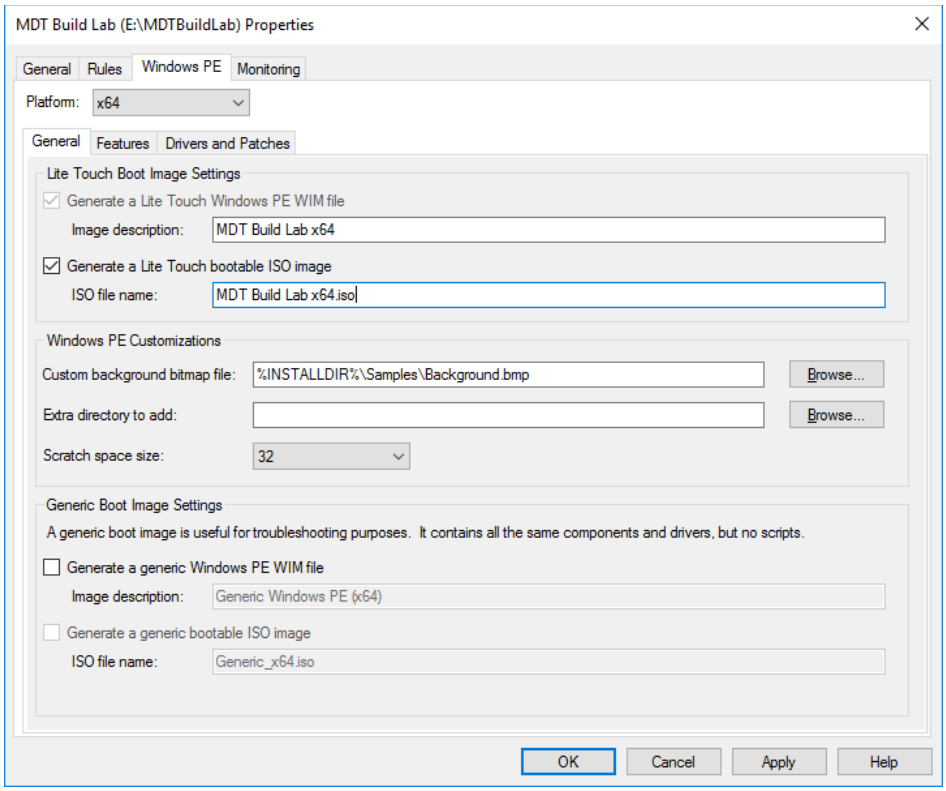
Step 5 – Configure the deployment share

Prepare the deployment share rules

1. On **MDT01**, navigate to the **E:\Setup\AGGW10-SampleFiles\MDT** folder.
2. Copy the following files to **E:\MDTBuildLab\Control** (replace the existing files).
    a. **Bootstrap.ini**
    b. **CustomSettings.ini**
3. Review the **E:\MDTBuildLab\Control\Bootstrap.ini** file, note the **DeployRoot** value.
4. Review the **E:\MDTBuildLab\Control\CustomSettings.ini** file.

Configure the Deployment Share

1. Using the **Deployment Workbench**, right-click the **MDT Build Lab** deployment share and select **Properties**.
2. In the **General** tab, in the **Platforms Supported** area, clear the **x86** checkbox.
3. In the **Windows PE** tab, in the **Platform** dropdown list, make sure **x64** is selected.
4. In the **Lite Touch Boot Image Settings** area, configure the following settings
    c. Image description: **MDT Build Lab x64**
    d. ISO file name: **MDT Build Lab x64.iso**
5. Click **OK**.



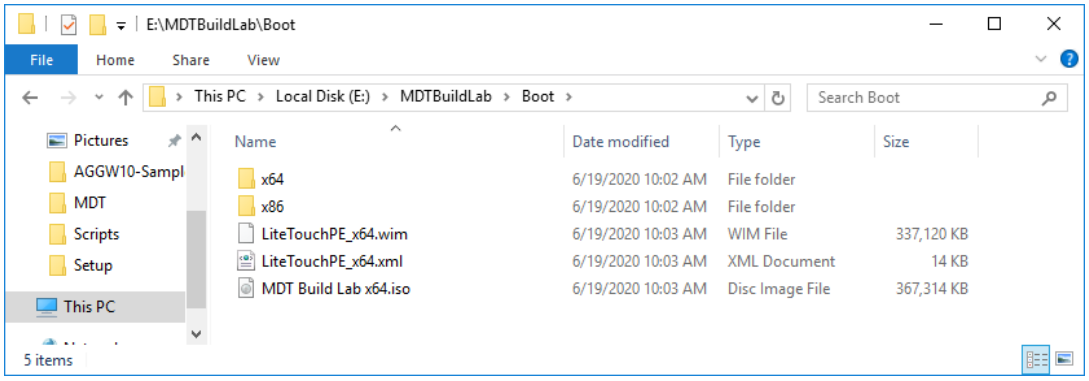
Properties of the MDT Build Lab deployment share, Windows PE x64 settings.

Update the Deployment Share

1. On **MDT01**, right-click the **MDT Build Lab** Deployment Share and select **Update Deployment Share**.
2. Use the default Options for the **Update Deployment Share** wizard.

Note: The update process will take a few minutes.

3. Review the contents of the **E:\MDTBuildLab\Boot** folder.



The contents of the E:\MDTBuildLab\Boot folder after updating the deployment share.

Step 6 – Create Windows Reference Images

Create a Windows 10 Reference WIM Image, fully automated

1. On **MDT01**, copy the **E:\MDTBuildLab\Boot\MDT Build Lab x64.iso** file to **C:\ISO** on your Host PC.
2. On the **REF001** virtual machine, mount the **MDT Build Lab x64.iso** media (located in the C:\ISO folder on the host PC).
3. Start the **REF001** virtual machine, allow it to boot, and complete the **Deployment Wizard** using the below settings:
    a. Select a task sequence to execute on this computer:

        Windows 10 Enterprise x64 v2004

    b. Specify whether to capture an image:

        i. **Capture an image of this reference computer**

        ii. Location: <**default>**

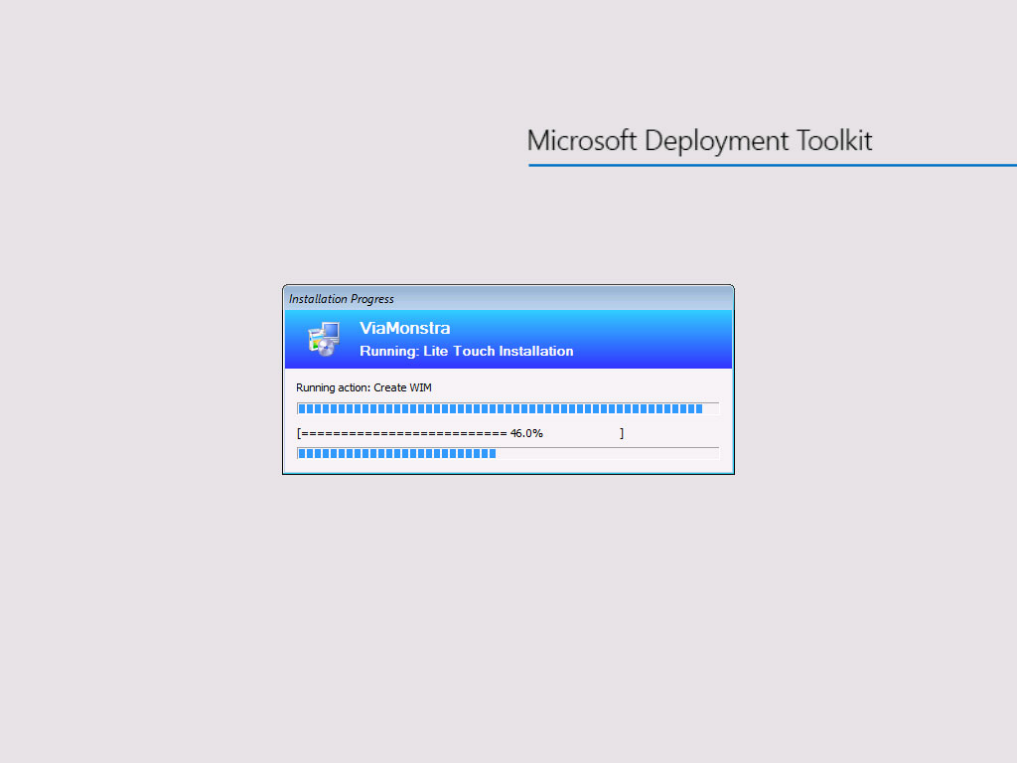
        iii. File name: **<default>**
4. The deployment process will do the following:
    a. Installed the Windows 10 Enterprise operating system.

    b. Installed any added applications, roles, and features.

    c. Stage WinPE on the local disk.

    d. Run Sysprep and reboot into WinPE.

    e. In WinPE, captured the installation to a WIM file (saved in the E:\MDTBuildLab\Captures folder on MDT01).



Windows 10 WIM image capture in progress.

Module 5

Setup ConfigMgr OSD

In this module you learn to setup ConfigMgr 2002 for deploying and servicing machines with Windows 10 Enterprise v2004, with or without the MDT integration.

Real World Note: For ConfigMgr 2002, make sure you have installed KB4567007 which contains OSD fixes for UEFI-based machines.

What you need for this module

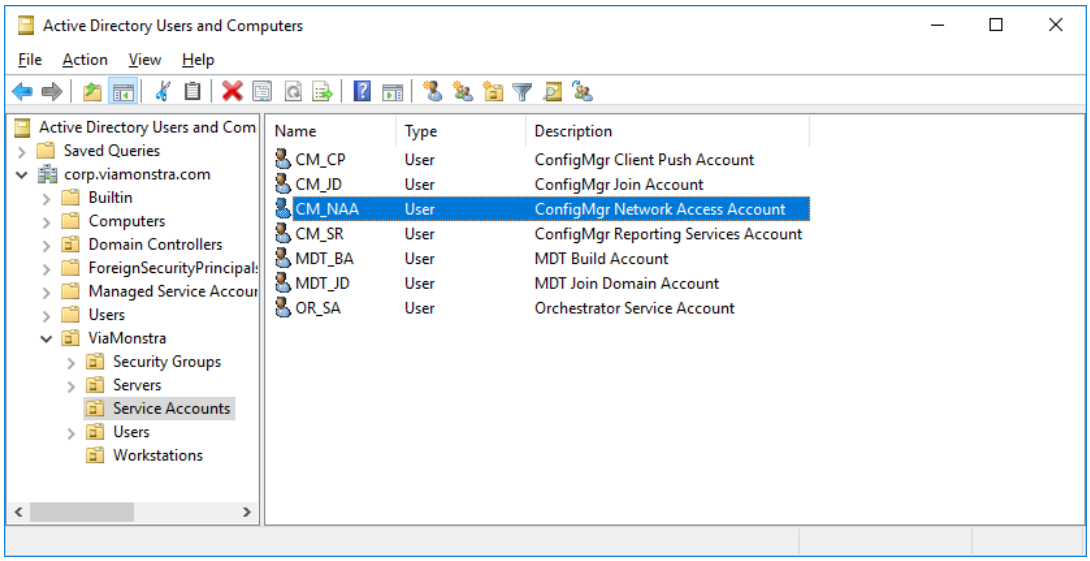
For the guides in this module, you need the following:

- A Domain Controller running either Windows Server 2012R, Windows Server 2016 or Windows Server 2019. In my lab I used a domain controller named DC01 that manages the corp.viamonstra.com domain (VIAMONSTRA is the NetBIOS name).
- A Member Server running either Windows Server 2012R, Windows Server 2016 or Windows 2019 with a ConfigMgr site server installed. In my lab I was using CM01 as single site server for the PS1 primary site. Having all the needed roles, like MP, DP, SMS provider, and site database server.
- An updated Windows 10 Enterprise v2004 WIM created either using OSD Builder (module 3), or from MDT Lite Touch build and capture (module 4).
- The sample files for this guide: https://bit.ly/AGGW10-SampleFiles

Step 1 – Prepare for ConfigMgr OSD

Review the service account for ConfigMgr OSD

1. On **DC01**, using **Active Directory Users and Computers**, make sure you have created the following user accounts:
    a. **CM_NAA**
    b. **CM_JD**
2. In my lab I added these accounts to the **ViaMonstra / Service Accounts** OU, and I assigned a password of **P@ssw0rd**.



The Service Accounts OU in my lab.

Configure Active Directory Permissions

In this section I assume you have downloaded the sample files for this guide.

1. On **DC01**, create the **C:\Setup** folder.
2. Extract the sample files to **C:\Setup**.
3. In an elevated **PowerShell prompt** (run as Administrator), run the following commands, press **Enter** after each command (the last command is wrapped):

```
Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Force
```

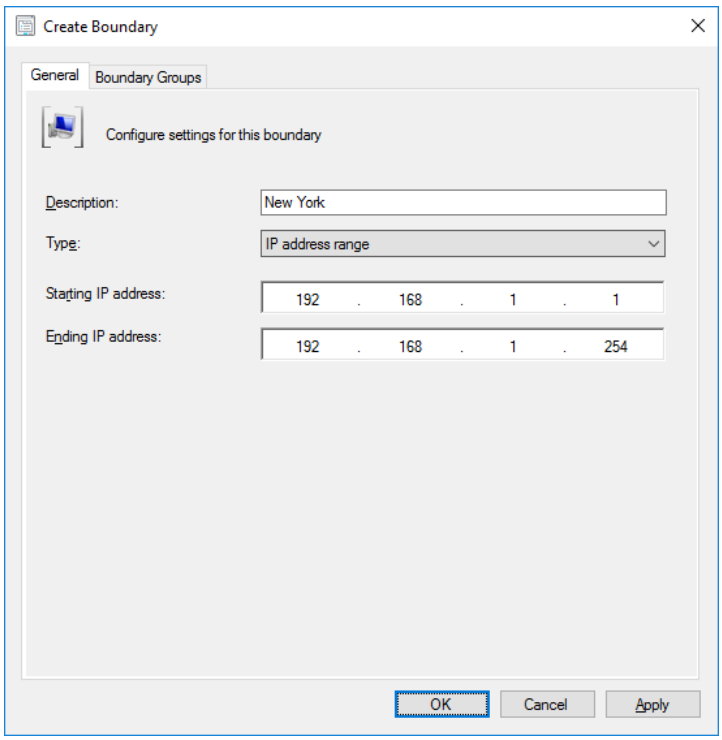
```
C:\Setup\AGGW10-SampleFiles\Scripts\Set-OUPermissions.ps1
-Account CM_JD -TargetOU "OU=Workstations,OU=ViaMonstra"
```

4. The previous script allows the CM_JD user account minimal permissions to manage computer accounts in the ViaMonstra / Workstations OU.

Create a boundary

In ConfigMgr, boundaries and boundary groups are used for clients to locate management points, software update points, and content on distribution points. I prefer to use IP ranges, and your lab you should at least one of these. In this example you create an IP range for the 192.168.1.0/24 subnet.

1. On **CM01**, using the **Configuration Manager Console**, in the **Administration** workspace, in **Hierarchy Configuration**, select **Boundaries**.
2. Create a boundary using the following settings:
    a. Description: New York
    b. Type: IP address range
    c. Starting IP address: 192.168.1.1
    d. Ending IP address: 192.168.1.254

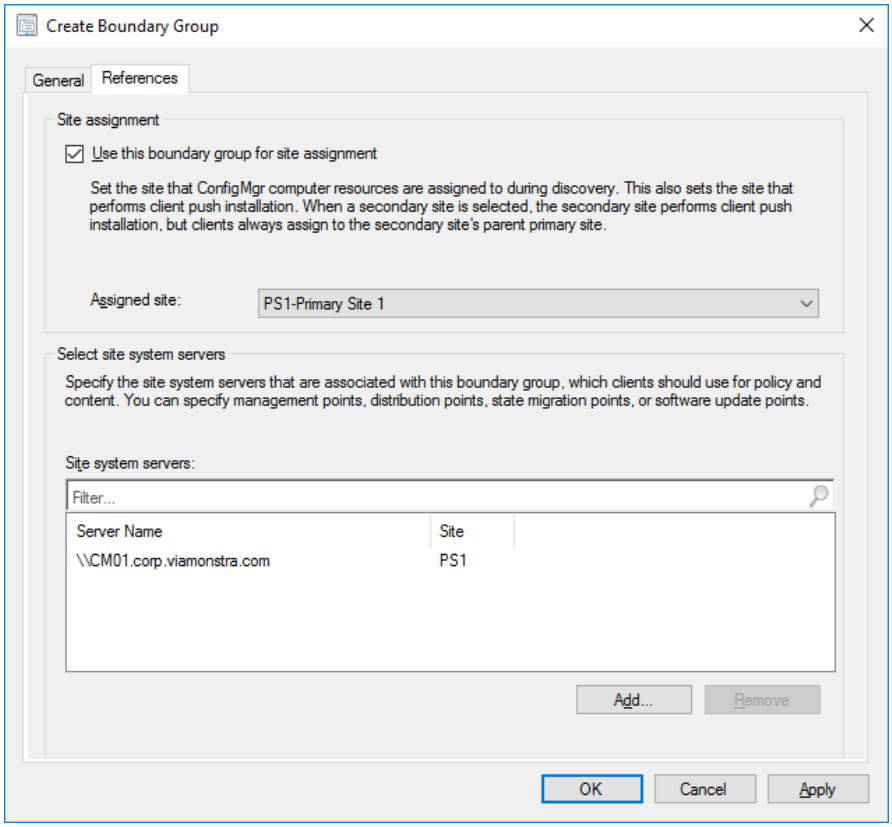


Creating an IP range boundary.

Create a Boundary Group

1. Using the **Configuration Manager Console**, in the **Administration** workspace, in **Hierarchy Configuration**, select **Boundary Groups**.
2. Create a boundary group using the following settings:
    a. In the **General** tab
        - Name: **HQ Assignment**
        - Boundaries: Add the **192.168.1.1 – 192.168.1.254** boundary.
    b. In the **References** tab

- Site assignment area: Select the **Use the boundary group for site assignment** check box.
- **Site system servers:** area: Add the **CM01** server.



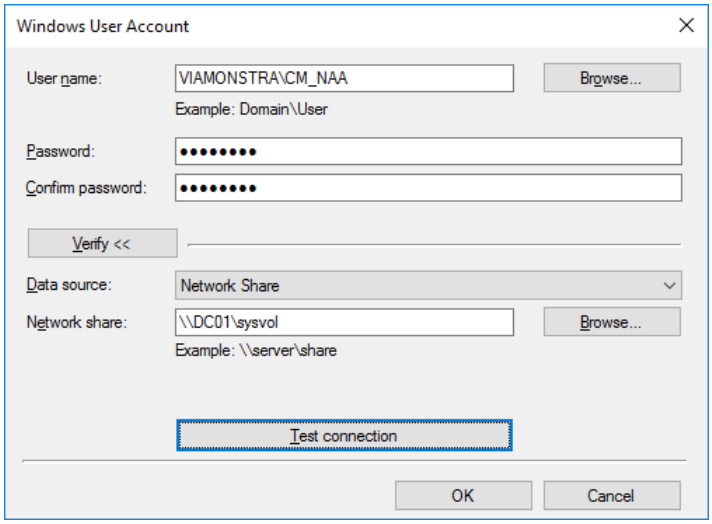
Creating the HQ Assignment boundary group.

Create a Distribution Point Group

1. Using the **Configuration Manager Console**, in the **Administration** workspace, create a distribution point group named **HQ DPs**.
2. In the **Members** tab, add the **CM01** distribution point.

Configure the Network Access Account

1. On **CM01**, using the **Configuration Manager Console**, in the **Administration** workspace, expand **Site Configuration** and select **Sites**.
2. Right-click **PS1 – Primary Site 1**, select **Configure Site Components** and the select **Software Distribution**.
3. In the **Network Access Account** tab, configure the **VIAMONSTRA\CM_NAA** user account (select New Account) as the network access account. Use the new **Verify** option to verify that the account can connect to **\\DC01\SYSVOL** network share.



Testing the connection.

Configure the Client Settings

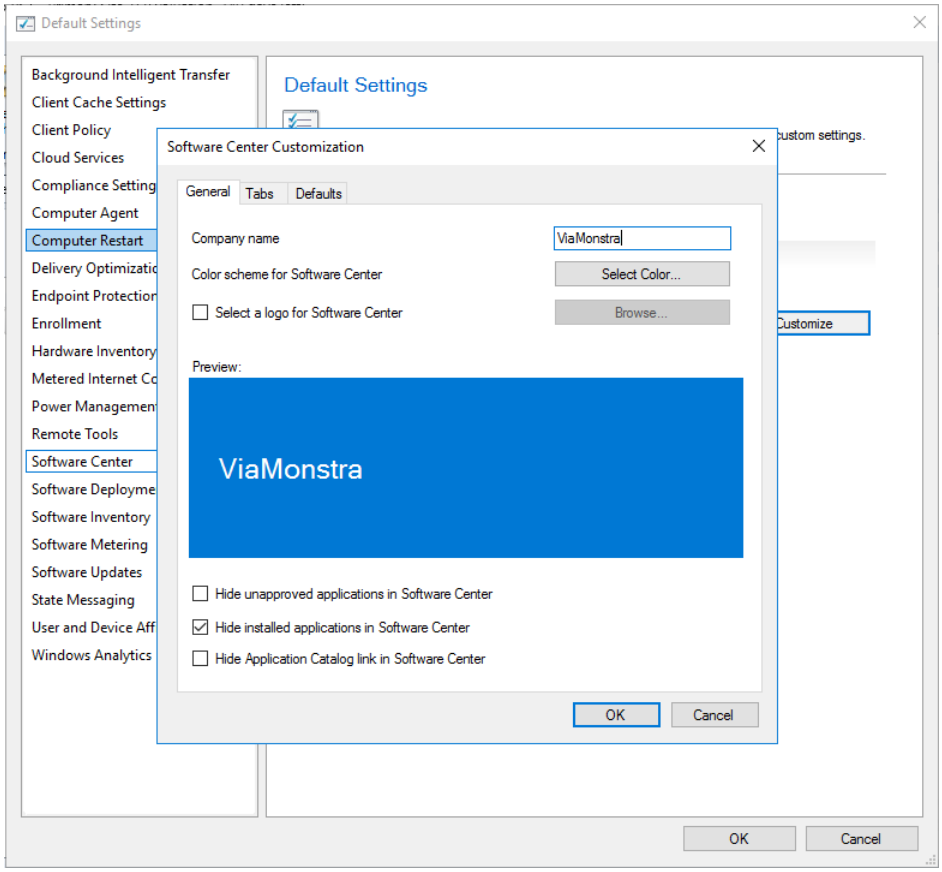
1. On **CM01**, using the **Configuration Manager Console**, in the **Administration** workspace, select **Client Settings**.
2. Right-click **Default Client Settings**, and select **Properties**.
3. In the **Client Cache Settings** node, configure the following settings
    a. Configure client cache size: **Yes**
    b. Maximum cache size (MB): **25600**
    c. Maximum cache size (percentage of disk): **25**

Note: With the above settings the client cache size can expand to either the maximum size in MB (25600), or the percentage of the disk (25), whichever is less.

4. In the **Computer Agent** node, configure the following settings
    a. Organization name displayed in Software Center: **ViaMonstra**

Note: The preceding setting is still need for the organization name to show during OS deployment.

5. In the **Software Center** node, configure the following settings
    a. Select these new settings to specify company information: **Yes**
    b. Click **Customize**, and In the **Company name** text box, type in **ViaMonstra**.
6. Click **OK** twice.

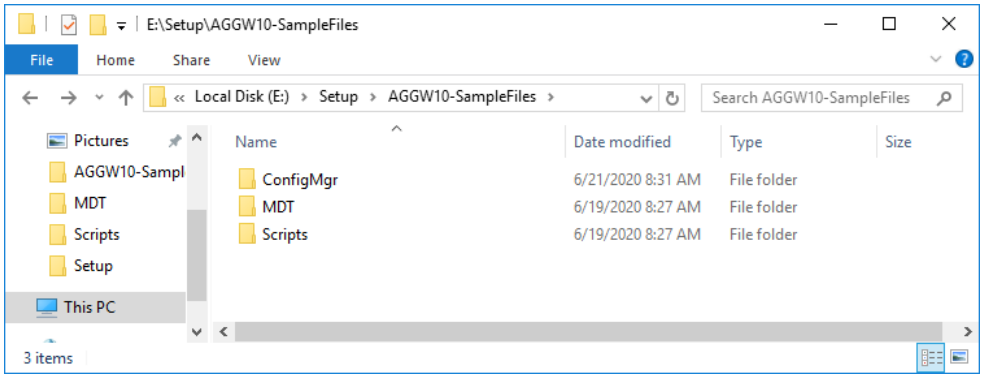


Configure client settings.

Copy the Sample Files to CM01

In this section I assume you have downloaded the sample files for this guide.

1. On **CM01**, create the **E:\Setup** folder.
2. Extract the sample files to **E:\Setup**



The E:\Setup\AGGW10-SampleFiles folder.

Create the Sources folder structure

1. On **CM01**, in an elevated **PowerShell prompt** (run as Administrator), run the following command (note that command is wrapped):

   ```
   E:\Setup\AGGW10-SampleFiles\Scripts\
   Create-ConfigMgrSourceFolders.ps1
   ```

2. The above script creates the following folder structure, and also shares the **E:\Sources** folder as **Sources**, and **E:\Logs** as **Logs**.

   E:\Logs

   E:\Sources

   E:\Sources\OSD

   E:\Sources\OSD\Boot

   E:\Sources\OSD\DriverPackages

   E:\Sources\OSD\DriverSources

   E:\Sources\OSD\MDT

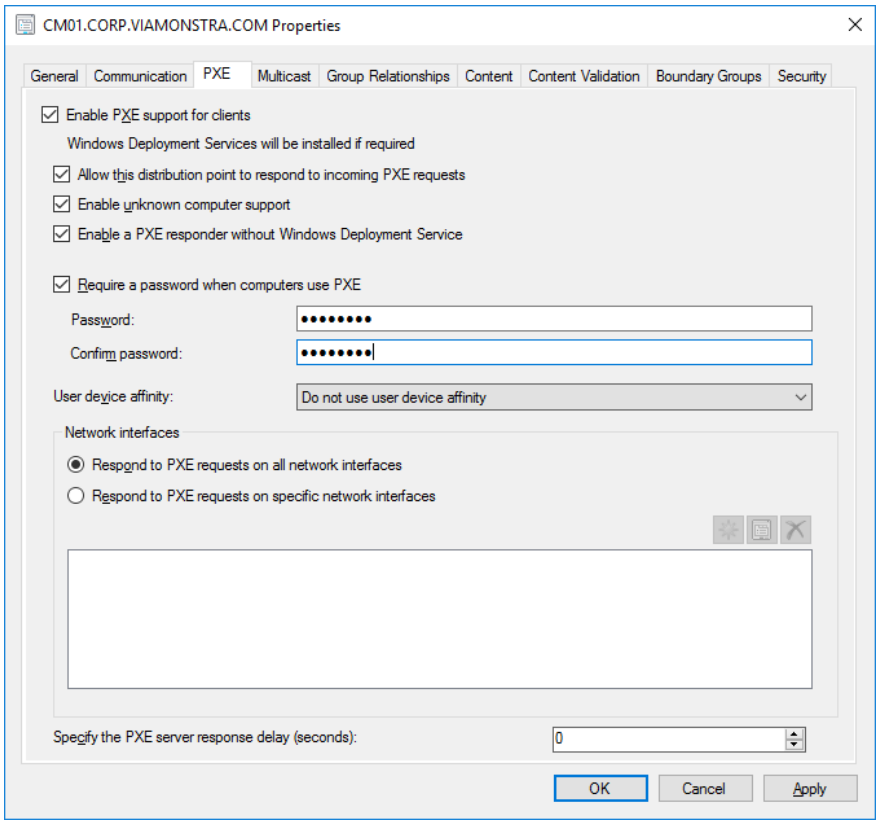
   E:\Sources\OSD\OS

   E:\Sources\OSD\Settings

Enable PXE on the CM01 Distribution Point

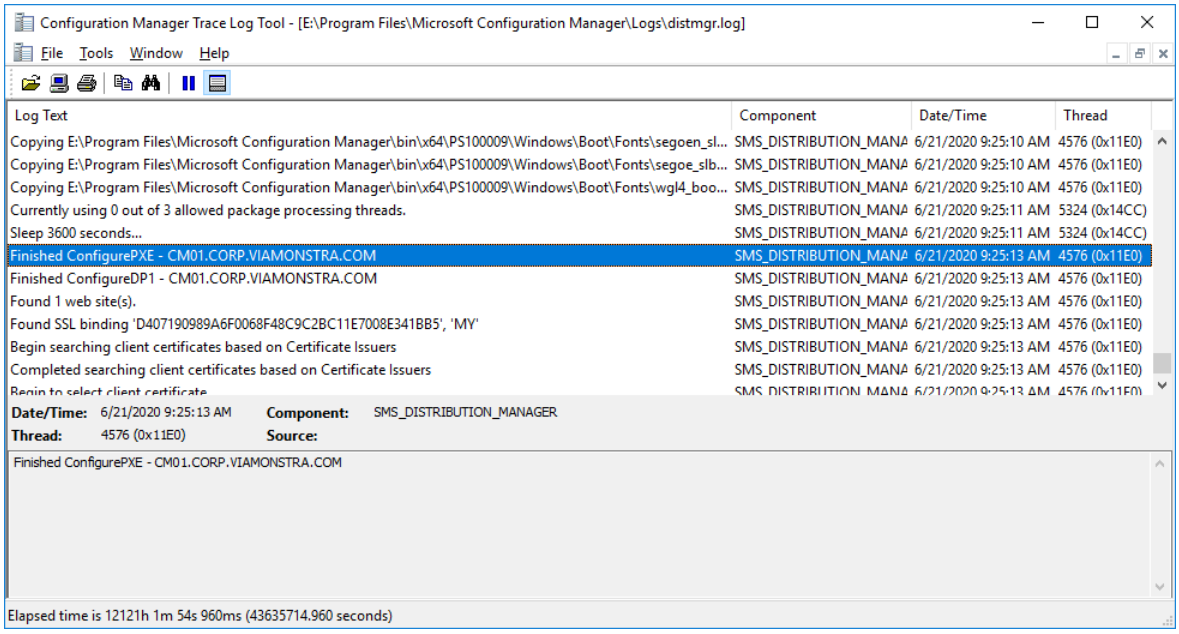
1. In the **Configuration Manager Console**, in the **Administration** workspace, select **Distribution Points**.
2. Right-click the **\\CM01.CORP.VIAMONSTRA.COM** DP and select **Properties**.
3. In the **PXE** tab, enable the following settings:
    a. Enable PXE support for clients
    b. Allow this distribution point to respond to incoming PXE requests
    c. Enable unknown computer support
    d. Enable a PXE responder without Windows Deployment Service
    e. Require a password when computers use PXE
    f. Password and Confirm password: **P@ssw0rd**

Real World Note: The "Enable a PXE responder without Windows Deployment Service" option is a fairly new PXE server for ConfigMgr, and since it's not depending on WDS, it can be installed on DPs running on Windows 10 as well.



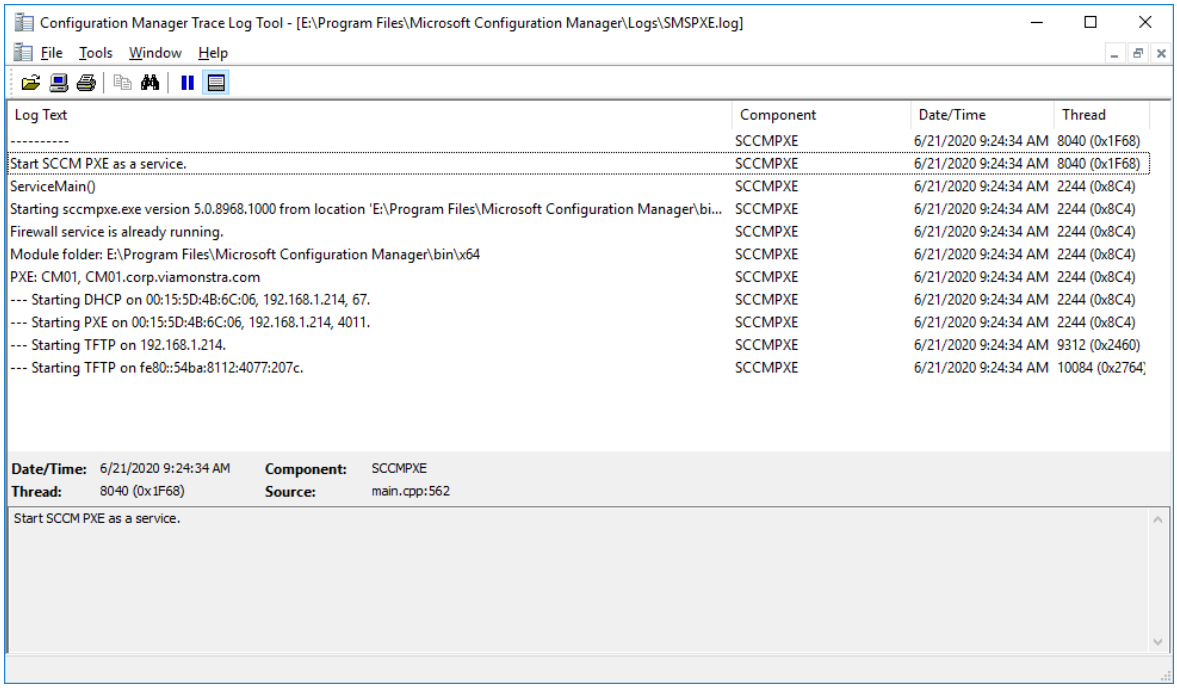
Configure the CM01 DP for PXE.

4. Using **CMTrace**, review the **E:\Program Files\Microsoft Configuration Manager\Logs\distmgr.log** file. Look for **ConfigurePXE** lines.



The distmgr.log showing the PXE feature being enabled.

5. Also review the **SMSPXE.log** in the same location.



The SMSPXE.log file showing the service being started.

Step 2 – Optional - Integrate with MDT

It is still quite popular to integrate ConfigMgr with MDT to get additional OSD features, but it's not a requirement for imaging with ConfigMgr. If you prefer to just use the native ConfigMgr OSD features, please skip to step 3 in this module.

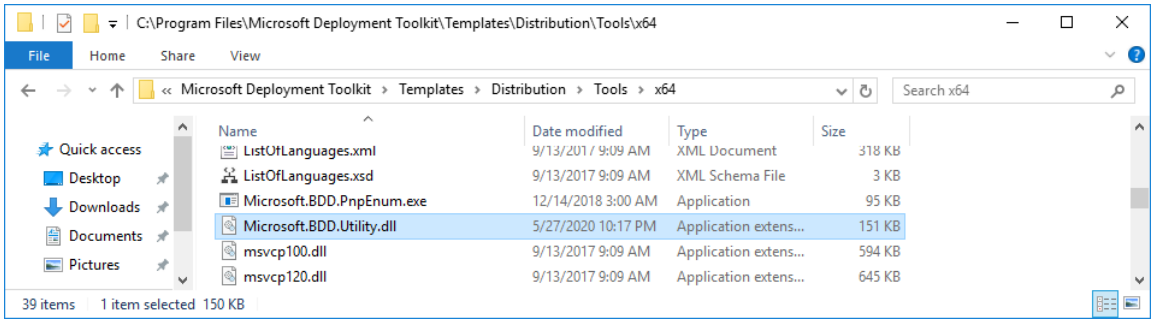
Install MDT 8456

1. On **CM01**, log on as an administrator.
2. Download **MDT 8456** from https://aka.ms/mdtdownload and install it using the default settings.

Install the MDT 8456 Hotfix

Due to a code change in Windows ADK 10 v2004, and Windows 10 v2004 you need to download some updated files for MDT 8456. You can download these files here: https://bit.ly/MDT8456HF

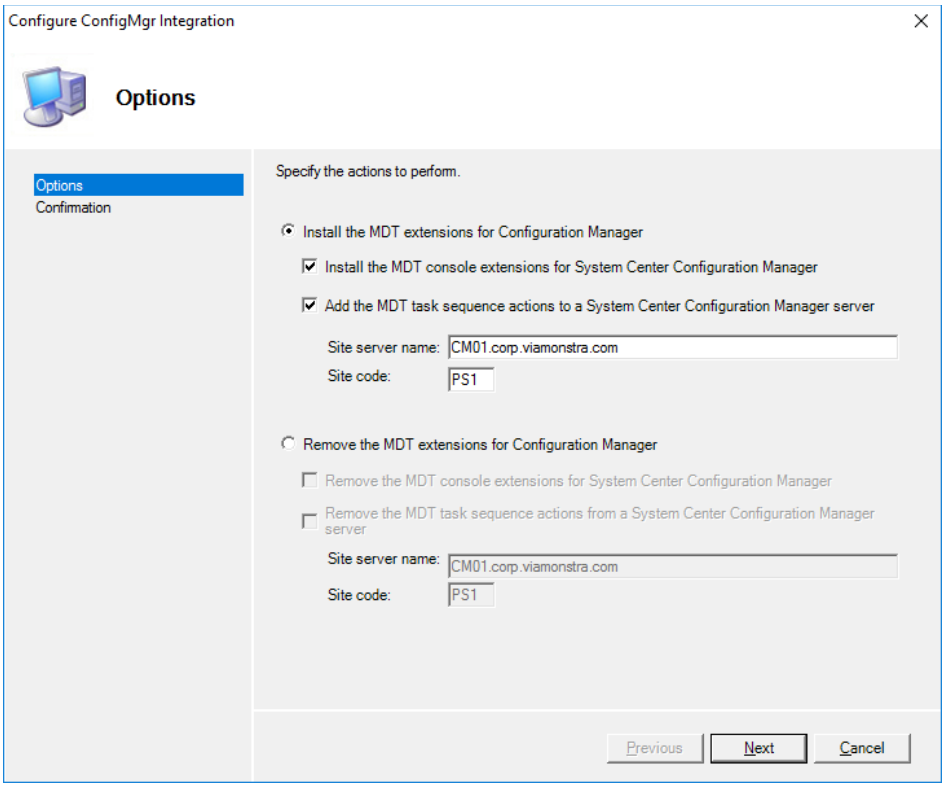
1. On **CM01**, download the **MDT 8456 hotfix** (MDT_KB4564442.exe), and extract it to a folder named C:\Setup\MDT 8456 Update (create the folder).
2. Copy the x86 version of the new **Microsoft.BDD.Utility.dll** from **C:\Setup\MDT 8456 Update\x86** to **C:\Program Files\Microsoft Deployment Toolkit\Templates\Distribution\Tools\x86**. Replace the existing file.
3. Copy the x64 version of the new **Microsoft.BDD.Utility.dll** from **C:\Setup\MDT 8456 Update\x** to C**:\Program Files\Microsoft Deployment Toolkit\Templates\Distribution\Tools\x64**. Replace the existing file.



The updated Microsoft.BDD.Utility.dll added to the MDT installation folder.

Setup MDT integration in ConfigMgr Console

1. On **CM01**, close the **Configuration Manager Console** before continuing.
2. From the **Start screen**, run the **Configure ConfigMgr Integration** wizard with the following settings
    a. Site Server Name: **CM01.corp.viamonstra.com**
    b. Site code: **PS1**



Setup the MDT integration with ConfigMgr.

Create an MDT Boot Image

1. Using **Configuration Manager Console**, in the **Software Library** workspace, expand **Operating Systems**, right-click **Boot Images**, select **Create Boot Image using MDT**, and create a new boot image package using the following settings.

    a. Package source folder to be created (UNC Path): **\\CM01\Sources\OSD\Boot\Zero Touch WinPE 10 x64**

Note: The Zero Touch WinPE 10 x64 folder does not yet exist, you need to type in the path, and the folder will be created later by the wizard.

    b. Name: **Zero Touch WinPE 10 x64**

    c. Platform: **x64**

    d. Scratch Space: **<default>**

    e. Components: **<default>**

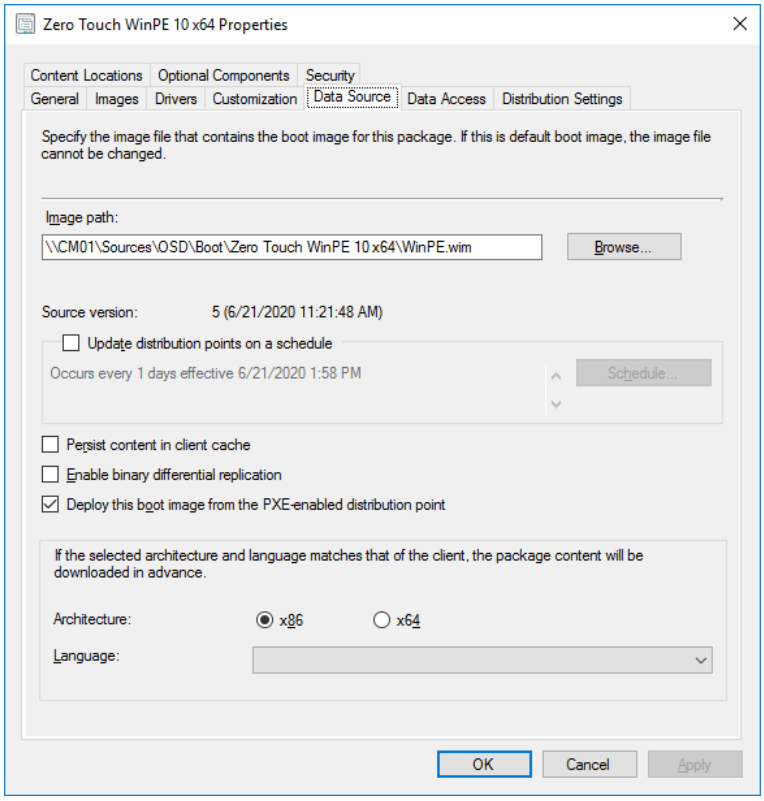
    f. Customization: **<default>**

2. Right-click the **Zero Touch WinPE 10 x64** boot image, and select **Distribute Content**. Use the following settings for the **Distribute Content Wizard**:

    Content Destination: Add the **HQ DPs** distribution point group.

3. Using **Configuration Manager Console**, right-click the **Zero Touch WinPE 10 x64** boot image and select **Properties**.

4. In the **Data Source** tab, select the **Deploy this boot image from the PXE-enabled distribution point** check box, and click **OK**.



Enabling PXE for the boot image.

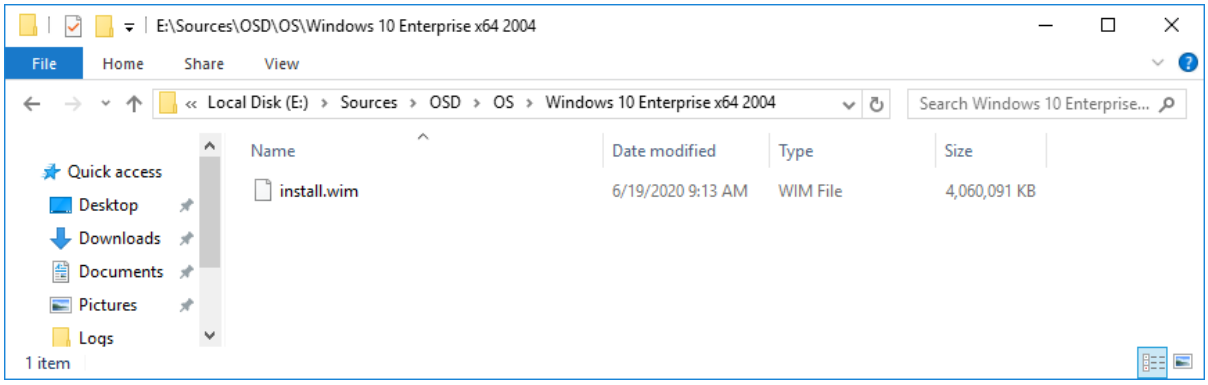
5. Using **CMTrace**, review the **E:\Program Files\Microsoft Configuration Manager\Logs\distmgr.log** file.

Step 3 – Add operating system images

Create the Windows 10 Operating System Image Package

1. On **CM01**, create the **E:\Sources\OSD\OS\Windows 10 Enterprise x64 v2004** folder.
2. Copy your reference WIM image to the **E:\Sources\OSD\OS\ Windows 10 Enterprise x64 v2004** folder.

Note: In this example I copied the install.wim image from \\PC0002\C$\OSDBuilder\OSBuilds\Windows 10 Enterprise x64 v2004 19041.329\OS\Sources to the E:\Sources\OSD\OS\ Windows 10 Enterprise x64 v2004 folder. But you can also copy the WIM image from MDT01 if you did a build and capture.



The Windows 10 image copied.

3. Using **Configuration Manager Console**, add an **Operating System Image** with the following settings. Use default settings for all other options.
   a. Path (click Browse): **\\CM01\Sources\OSD\OS\Windows 10 Enterprise x64 v2004\install.wim**
   b. Name: **Windows 10 Enterprise x64 v2004**

Step 4 – Add Drivers

To add support for new hardware you may need to add drivers to both WinPE, and to full Windows. In this example you add some WinPE and Windows drivers from HP.

Real World Note: Except for WinPE drivers, there is no need to import any other drivers into the ConfigMgr database. It's much better to create standard packages than to create driver packages in ConfigMgr. Creating standard packages is much faster, doesn't have distribution issues (hash value is not correct type errors), allows for more flexibility during deployment, and it doesn't bloat the ConfigMgr database with info you never will use.

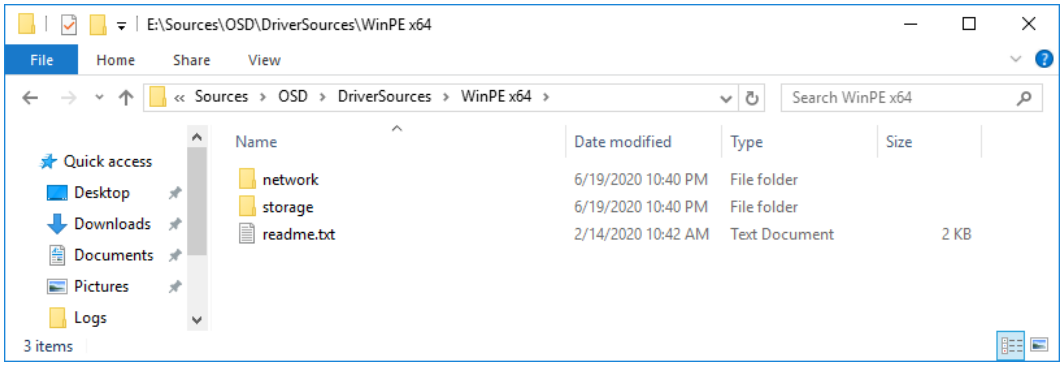
The drivers referenced in this guide can be downloaded from these URLs:

WinPE x64 drivers
https://ftp.hp.com/pub/softpaq/sp101001-101500/sp101480.exe

HP EliteBook 745 G6 Notebook PC drivers for Windows 10 x64 v2004
https://ftp.hp.com/pub/softpaq/sp101501-102000/sp101922.exe

Add HP Drivers for WinPE

1. On **CM01**, download the **HP Client Windows PE Driver pack for WinPE 10**, and extract it to the **E:\Sources\OSD\DriverSources\WinPE x64** folder (create the folder).



The HP WinPE drivers extracted to E:\Sources\OSD\DriverSources\WinPE x64.

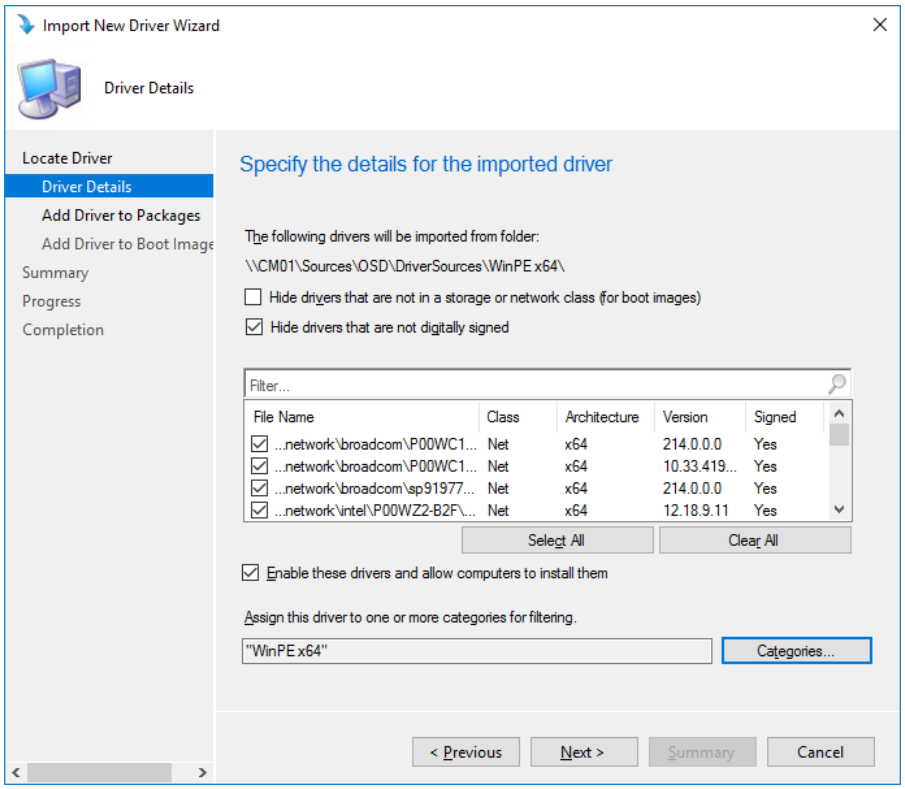
2. Using the **Configuration Manager Console**, in the **Software Library** workspace, right-click the **Drivers** folder and select **Import Driver**, use the following settings for the **Import New Driver Wizard**.

   a. Locate Driver

      Source folder: **\\CM01\Sources\OSD\DriverSources\WinPE x64**

   b. Driver Details (Note the new driver information and filters being available).

      Click **Categories**, and create a category named **WinPE x64**



Adding HP drivers for WinPE x64.

c. Add Driver to Packages

<default>

d. Add Driver to Boot Images

i. Select either the **Boot Image (x64)** boot image, or the **Zero Touch WinPE 10 x64** boot image if using the MDT integration.

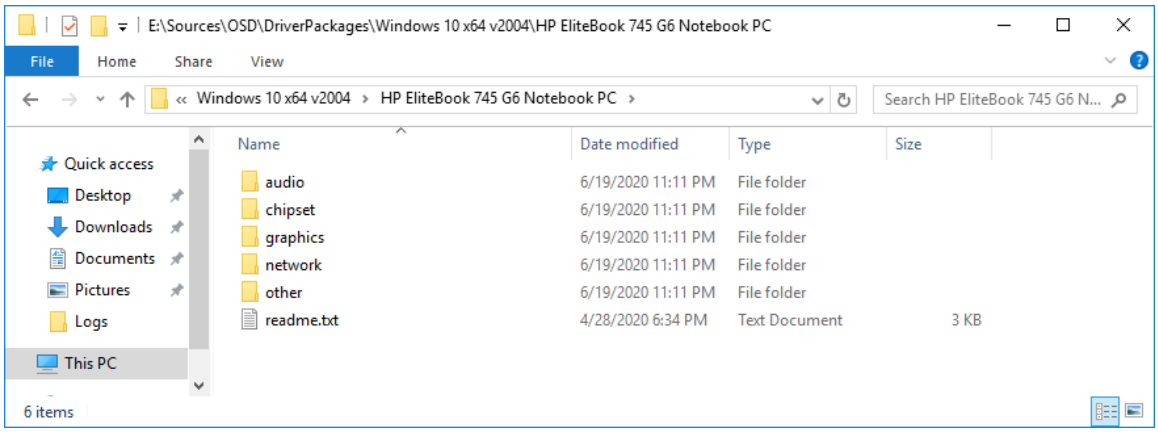
ii. Click **Yes** twice, and then **Next**, to start the update distribution process.

Add Windows 10 Drivers for HP EliteBook 745 G6 Notebook PC

In this section you learn to add drivers for the HP EliteBook 745 G6 Notebook PC model using a standard ConfigMgr package.

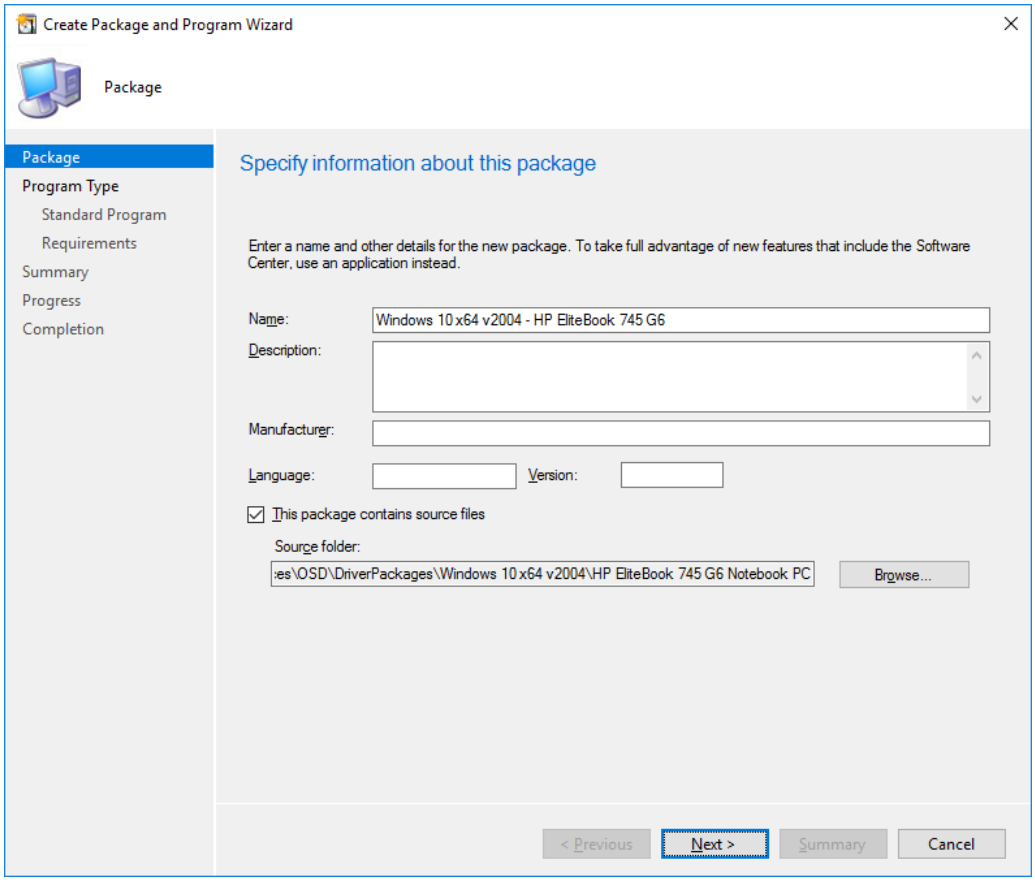
Real World Note: While these steps will work just fine from a technically point of view, I highly recommend looking into using the driver automation tool from Maurice Daly. That tool, which is part of the free modern driver management solution is hands down amazing. You can read more about that solution here: https://msendpointmgr.com/modern-driver-management/

1. On **CM01**, download the **HP EliteBook 745 G6 Notebook PC** drivers, and extract them to the **E:\Sources\OSD\DriverPackages\Windows 10 x64 v2004\HP EliteBook 745 G6 Notebook PC** folder (create the folder).



The drivers for HP EliteBook 745 G6 Notebook PC extracted.

2. Using the **Configuration Manager Console**, in the **Software Library** workspace, right-click the **Packages** node and select **Create Package**.
3. Use the following settings for the **Create Package and Program Wizard**.
   a. Name: **Windows 10 x64 v2004 - HP EliteBook 745 G6**
   b. This package contains source files
      - Source folder (click Browse): **\\CM01\Sources\OSD\DriverPackages\Windows 10 x64 v2004\HP EliteBook 745 G6 Notebook PC**
   c. Do not create a program



Creating a standard package for drivers.

Step 5 – Create Task Sequences

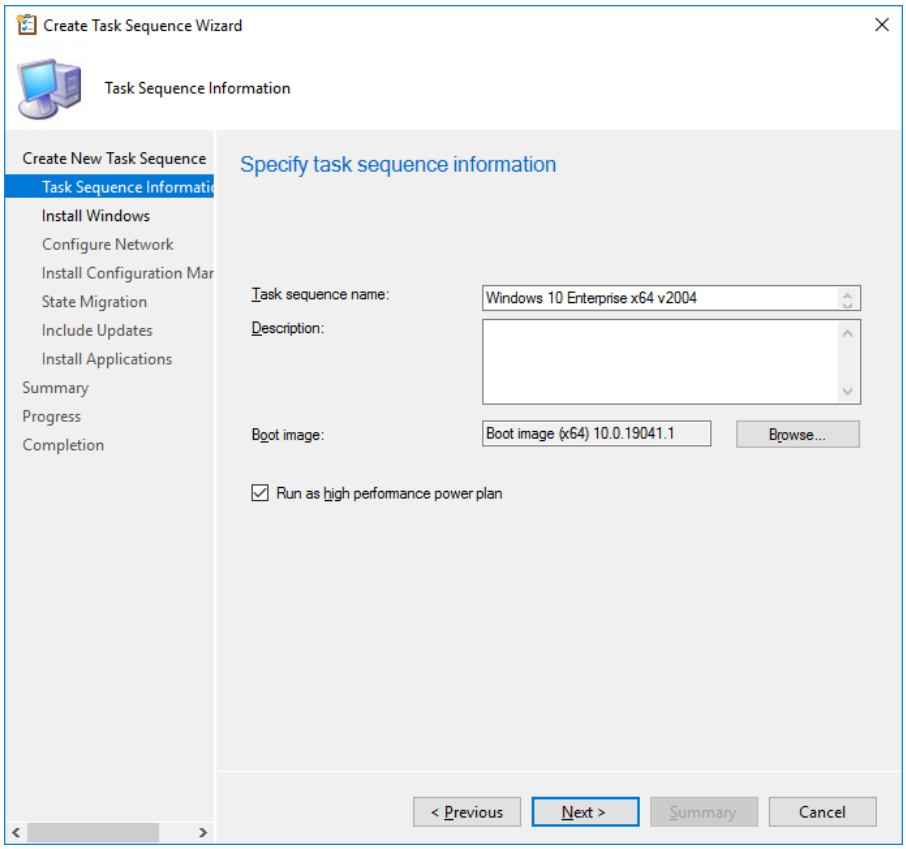
In this section you create the task sequence for bare metal deployment. If you integrated ConfigMgr with MDT in step 2 of this module, skip to the second task sequence in this section.

Option #1 - Create a Native ConfigMgr Task Sequence

1. Using **Configuration Manager Console**, in the **Software Library** workspace, expand **Operating Systems**, right-click **Task Sequences**, and select **Create Task Sequence**.
2. Use the following settings for the **Create Task Sequence** wizard.
   a. Create a new task sequence

      Install an existing image package

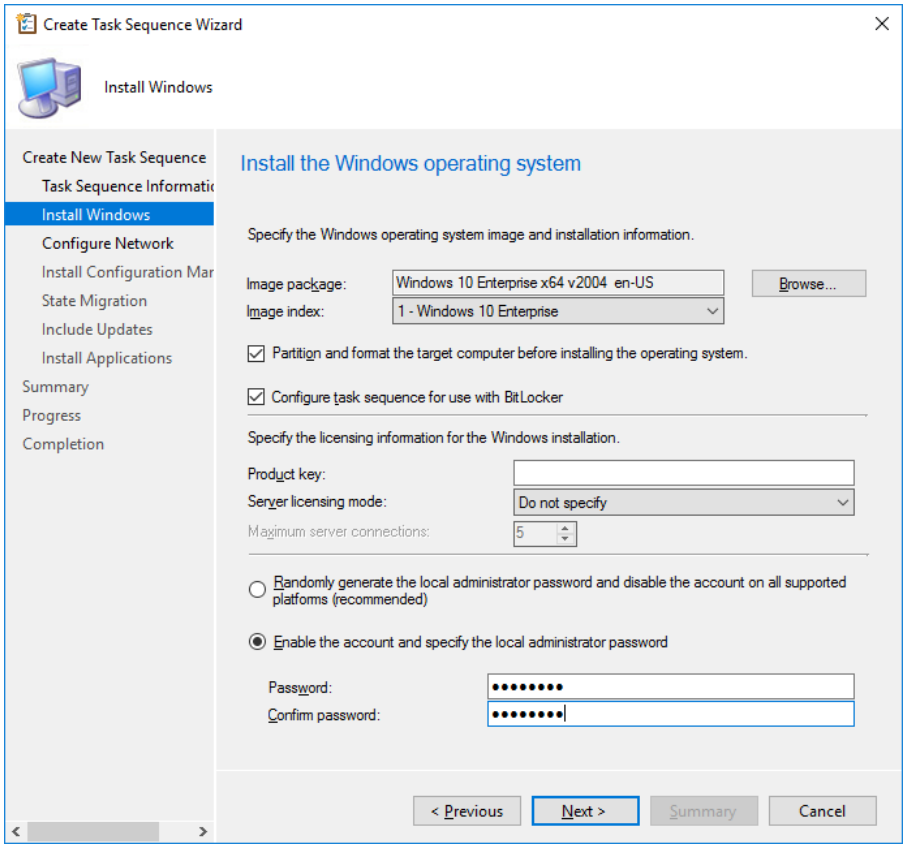
   b. Task Sequence Information
      - Task sequence name: **Windows 10 Enterprise x64 v2004**
      - Boot image: **Boot image (x64)**
      - Select the **Run as high performance power plan** check box



Specify task sequence information.

c. Install Windows

- Image Package: Windows 10 Enterprise x64 v2004
- Image index: 1- Windows 10 Enterprise
- Enable the account and specify the local administrator password
    - Password and Confirm password: **P@ssw0rd**



Configuring the Install Windows page.

Real World Note: Assigning a local administrator account/password is quite useful in a lab in the event you need to do some troubleshooting, but for production deployments its recommended having ConfigMgr disable it.

d. Configure Network

- **Join a Domain**
- Domain: **corp.viamonstra.com**
- Domain OU: **LDAP://OU=Workstations,OU=ViaMonstra, DC=corp,DC=viamonstra,DC=com**
- Account: **VIAMONSTRA\CM_JD**
- Password: **P@ssw0rd**

e. Install Configuration Manager Client
    - Accept the default Configuration Manager Client Package

f. State Migration
    - Accept the default settings

g. Include Updates
    - Select the **Do not install any software updates** option

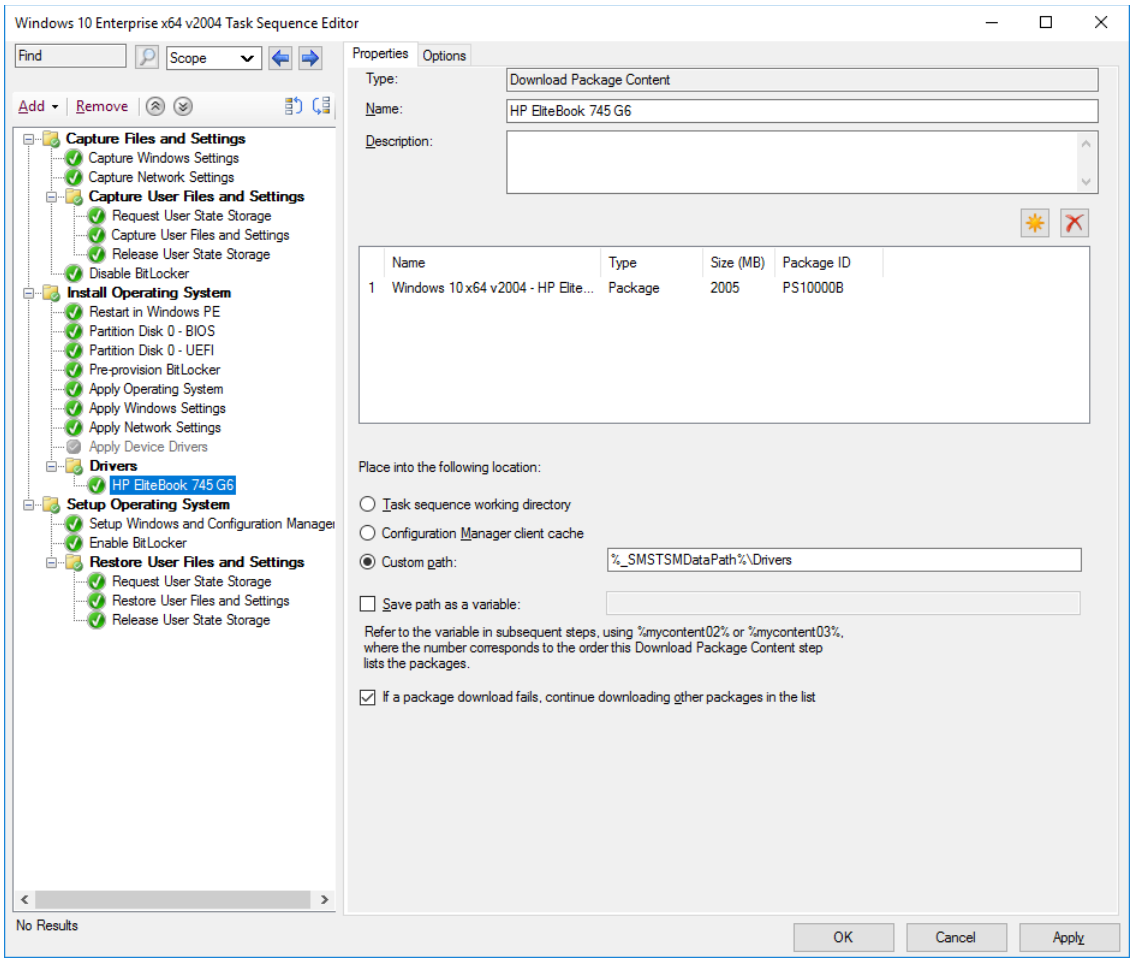
Real World Note: The ConfigMgr feature of installing software updates in a task sequence during deployment have been plagued with bugs over the years, and even when working, it adds extra time to the deployment too. I recommend having your image fully patched, and then use regular software updates in ConfigMgr to keep the client updated.

h. Install Applications
    - Select any applications you want the task sequence to install, or skip for now. Can always be added later.

Edit the Native ConfigMgr Task Sequence

Once the native ConfigMgr task sequence is created, you need to add drivers to it.

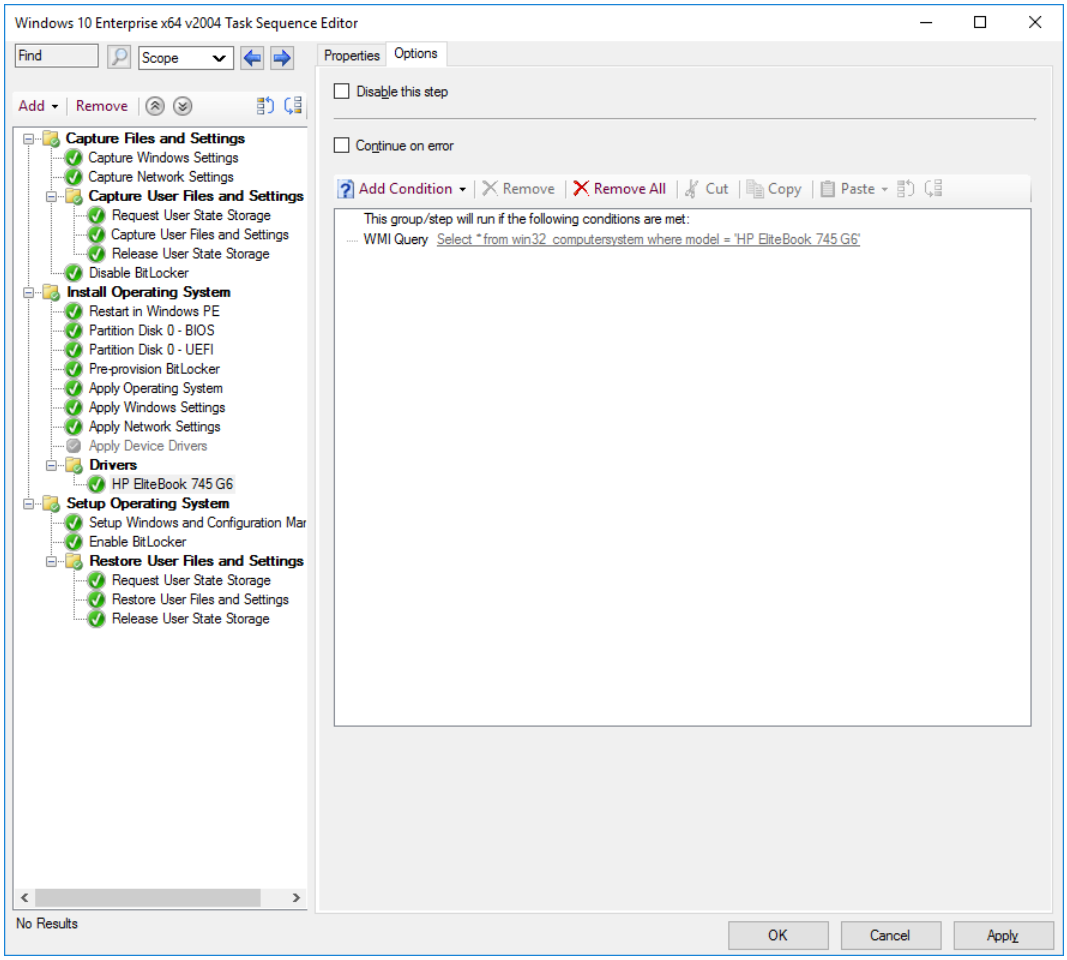
1. Using the **Configuration Manager Console**, select **Task Sequences**, right-click the **Windows 10 Enterprise x64 v2004** task sequence and select **Edit**.
2. In the **Install Operating System** group, disable the **Auto Apply Drivers** action. (Disabling is done by selecting the action and, in the Options tab, select the **Disable this step** check box.)
3. In the **Install Operating System** group, disable the **Apply Device Drivers** action. (Disabling is done by selecting the action and, in the **Options** tab, select the **Disable this step** check box.)
4. After the disabled **Install Operating System / Apply Device Drivers** action, add a new group named **Drivers**.
5. After the **Install Operating System / Drivers** group, add a **Download Package Content** action with the following settings:
    a. Name: **HP EliteBook 745 G6**
    b. Add package: **Windows 10 x64 v2004 - HP EliteBook 745 G6**
    c. Custom path: **%_SMSTSMDataPath%\Drivers**



Adding drivers as standard packages.

d. In **Options**, add a **Query WMI** condition with the following WQL query:

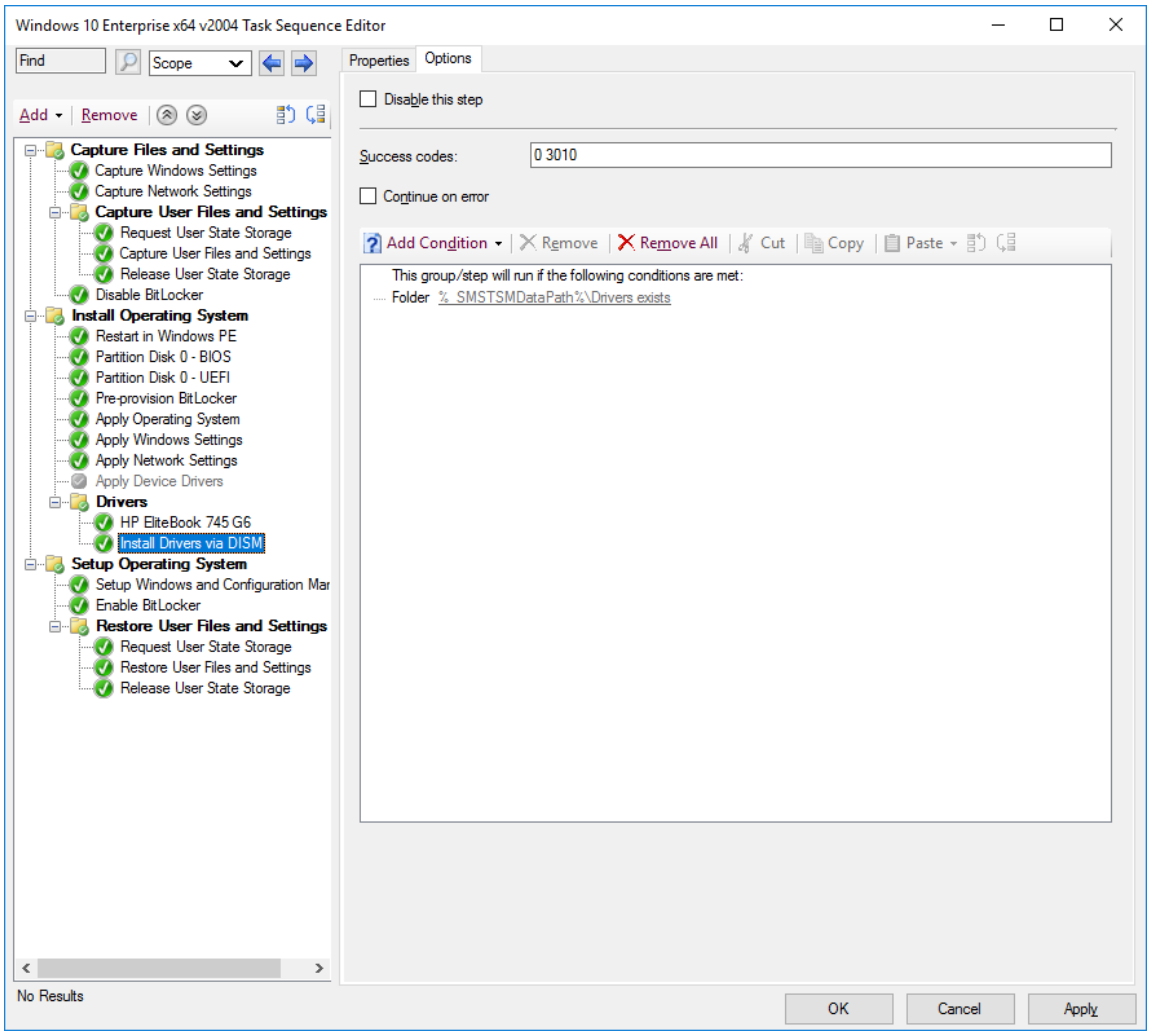
- **Select * from win32_computersystem where model = 'HP EliteBook 745 G6'**



Setting the condition for the driver package.

6. After the **HP EliteBook 745 G6** Download Package Content action, add a Run Command Line with the following settings:
    a. Name: **Install Drivers via DISM**
    b. Command line: **DISM.exe /Image:%OSDTargetSystemDrive%\ /Add-Driver /Driver:%_SMSTSMDataPath%\Drivers\ /Recurse /logpath:%_SMSTSLogPath%\dism.log**
    c. In **Options**, add a **Folder Properties** condition with the following settings:
        i. **%_SMSTSMDataPath%\Drivers exists**
7. Click **OK**.

Note: Again, while this method of adding driver packages works great from technical point of view, I highly recommend adding the community developed Modern Driver Management solution available on https://msendpointmgr.com/modern-driver-management/



Setting the condition on the Install Drivers via DISM action.

Distribute content to the Distribution Point

1. Using the **Configuration Manager Console**, select **Task Sequences**, right-click the **Windows 10 Enterprise x64 v2004** task sequence and select **Distribute Content**.
2. Use the following settings for the **Distribute Content Wizard**:

   Content Destination: Add the **HQ DPs** distribution point group.
3. Using **CMTrace**, verify the distribution to the **CM01** distribution point (part of the HQ DPs Group) by reviewing the **distmgr.log** file. Do not continue until you see all the new packages being distributed successfully.

Create a deployment for the Task Sequence

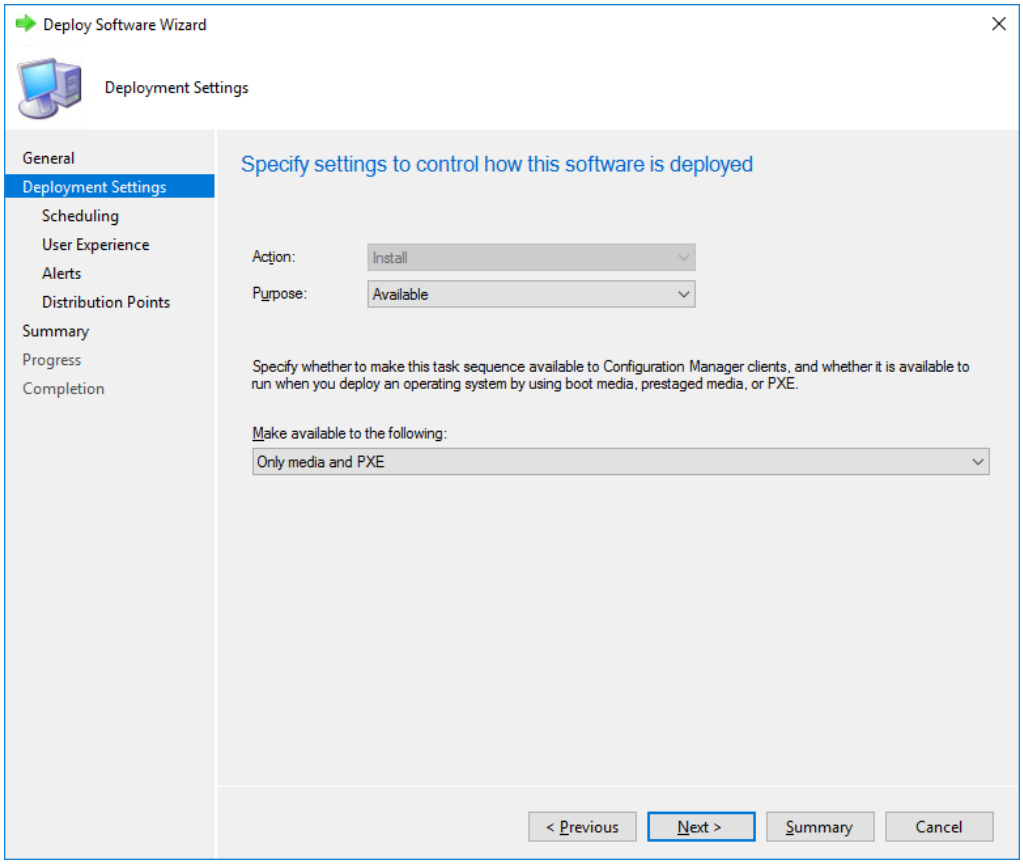
1. Using **Configuration Manager Console**, select **Task Sequences**, right-click **Windows 10 Enterprise x64 v2004**, and then select **Deploy**.
2. Use the following settings for the **Deploy Software Wizard**:
   a. General

      Collection: **All Unknown Computers**

Note: Review the high risk deployment dialog box that is displayed when browsing for collections. This feature was added already back in ConfigMgr 2012 R2 SP1 and can be configured on the site properties.

   b. Deployment Settings
      - Purpose: **Available**
      - Make available to the following: **Only media and PXE**

   c. Use the default settings for the remaining wizard pages.



Configuring the deployment settings.

Option #2 - Create an MDT integrated ConfigMgr Task Sequence

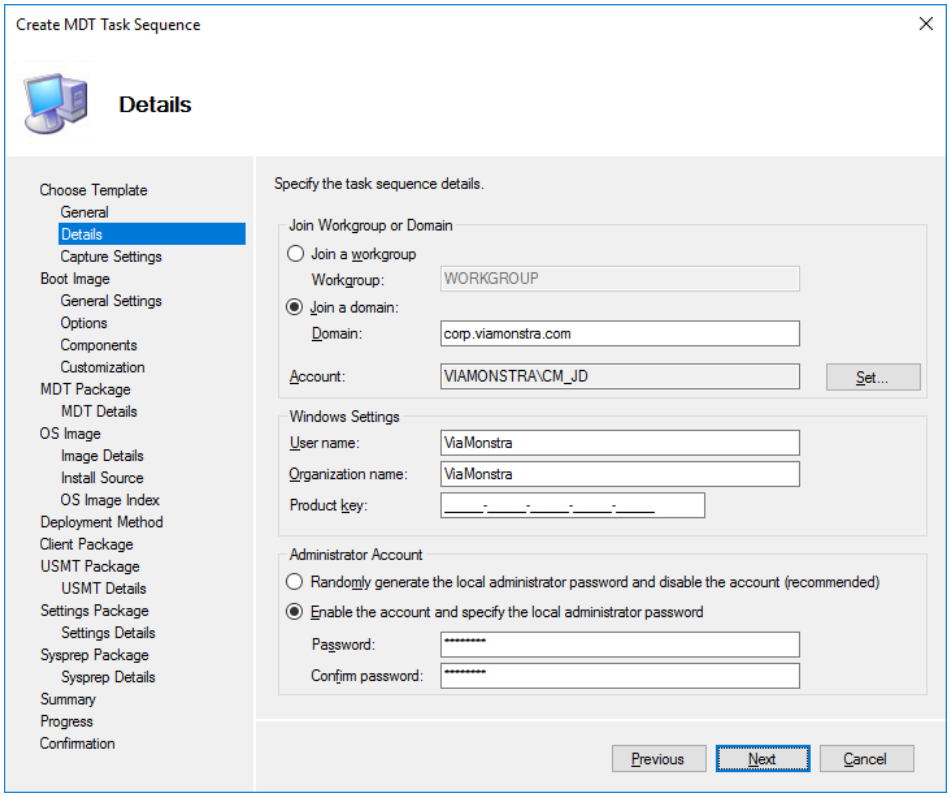
1. Using **Configuration Manager Console**, in the **Software Library** workspace, expand **Operating Systems**, right-click **Task Sequences**, and select **Create MDT Task Sequence**.
2. Use the following settings for the **Create MDT Task Sequence** wizard.
    a. Choose Template

        Template: **Client Task Sequence**

    b. General
        - Task sequence name: **Windows 10 Enterprise x64 v2004 - MDT**

    c. Details
        - **Join a Domain**
        - Domain: **corp.viamonstra.com**
        - Account: **VIAMONSTRA\CM_JD**
        - Password: **P@ssw0rd**

    d. Windows Settings
        - User name: **ViaMonstra**
        - Organization name: **ViaMonstra**
        - Product key: **<blank>**
        - Administrator Account
            - Enable the account and specify the local administrator password
                - Password and Confirm password: **P@ssw0rd**



Configuring the Details settings.

e. Capture Settings

This task sequence will never be used to capture an image.

f. Boot Image

- **Specify a boot image package to use**
- Select the **Zero Touch WinPE 10 x64** boot image package

g. MDT Package

- **Create a new Microsoft Deployment Toolkit Files package**
- Package source folder to be created (UNC Path): **\\CM01\Sources\OSD\MDT\MDT 8456**

h. MDT Details

Name: **MDT 8456**

i. OS Image

- **Specify an existing OS image**
- Select the **Windows 10 Enterprise x64 v2004** package

j. Deployment Method

Perform a "Zero Touch Installation" OS Deployment, with no user interaction

k. Client Package

- **Specify an existing ConfigMgr client package**
- Select the **Microsoft Corporation Configuration Manager Client Package** package.

l. USMT Package

- **Specify an existing USMT package**
- Select the **Microsoft Corporation User State Migration Tool for Windows 10 10.0.19041.1** package

m. Settings Package

- **Create a new settings package**
- Package source folder to be created (UNC Path): **\\CM01\Sources\OSD\Settings\Windows 10 x64 Settings**

n. Settings Details

Name: **Windows 10 x64 Settings**

o. Sysprep Package

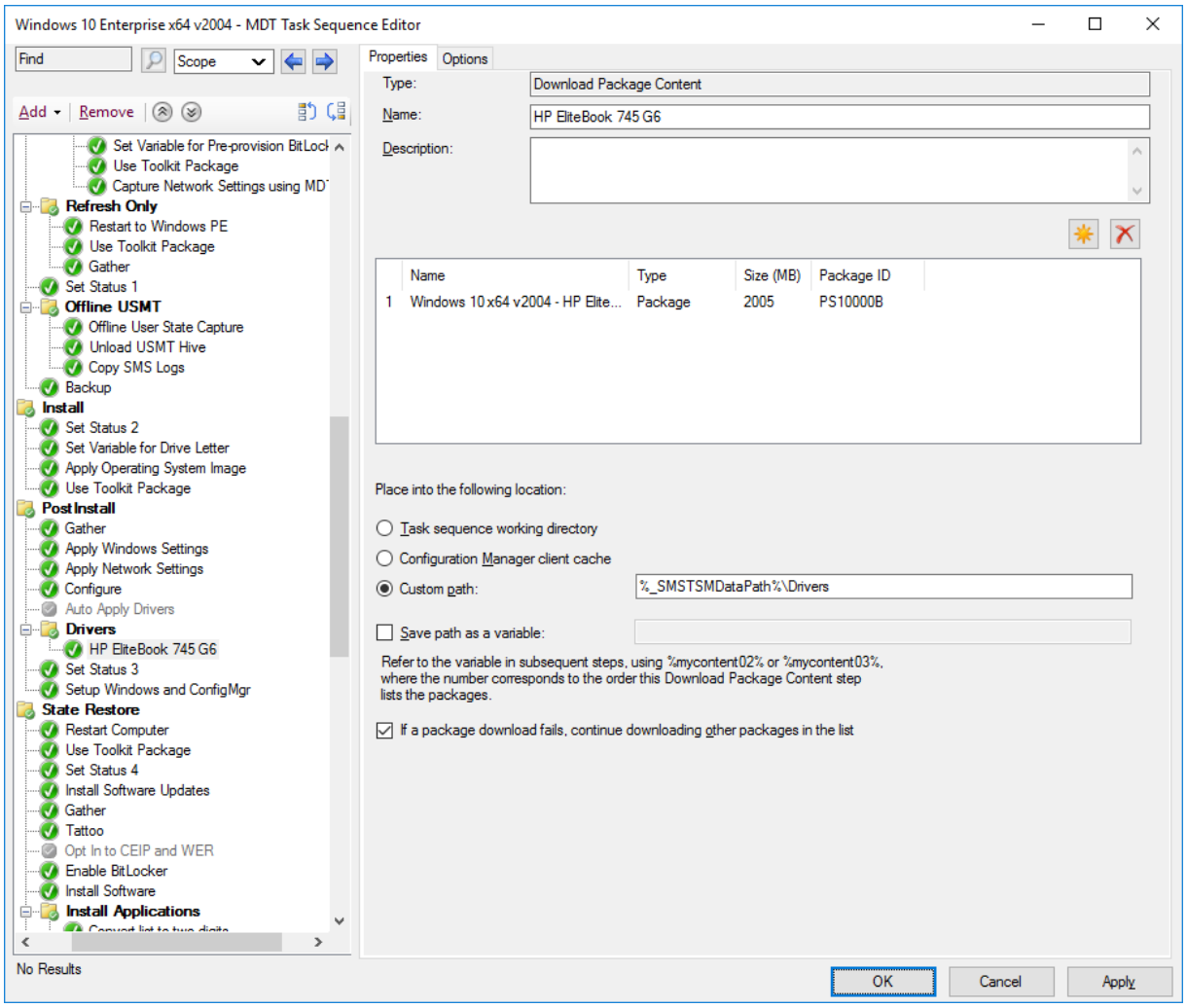
No Sysprep Package is required

'Edit the Task Sequence

In order to have the MDT standard client task sequence supporting dynamically assigning an OU, you need to provide a default value that Configuration action can update when needed. Also, since this is a production deployment task sequence, you also add in drivers to the task sequence.

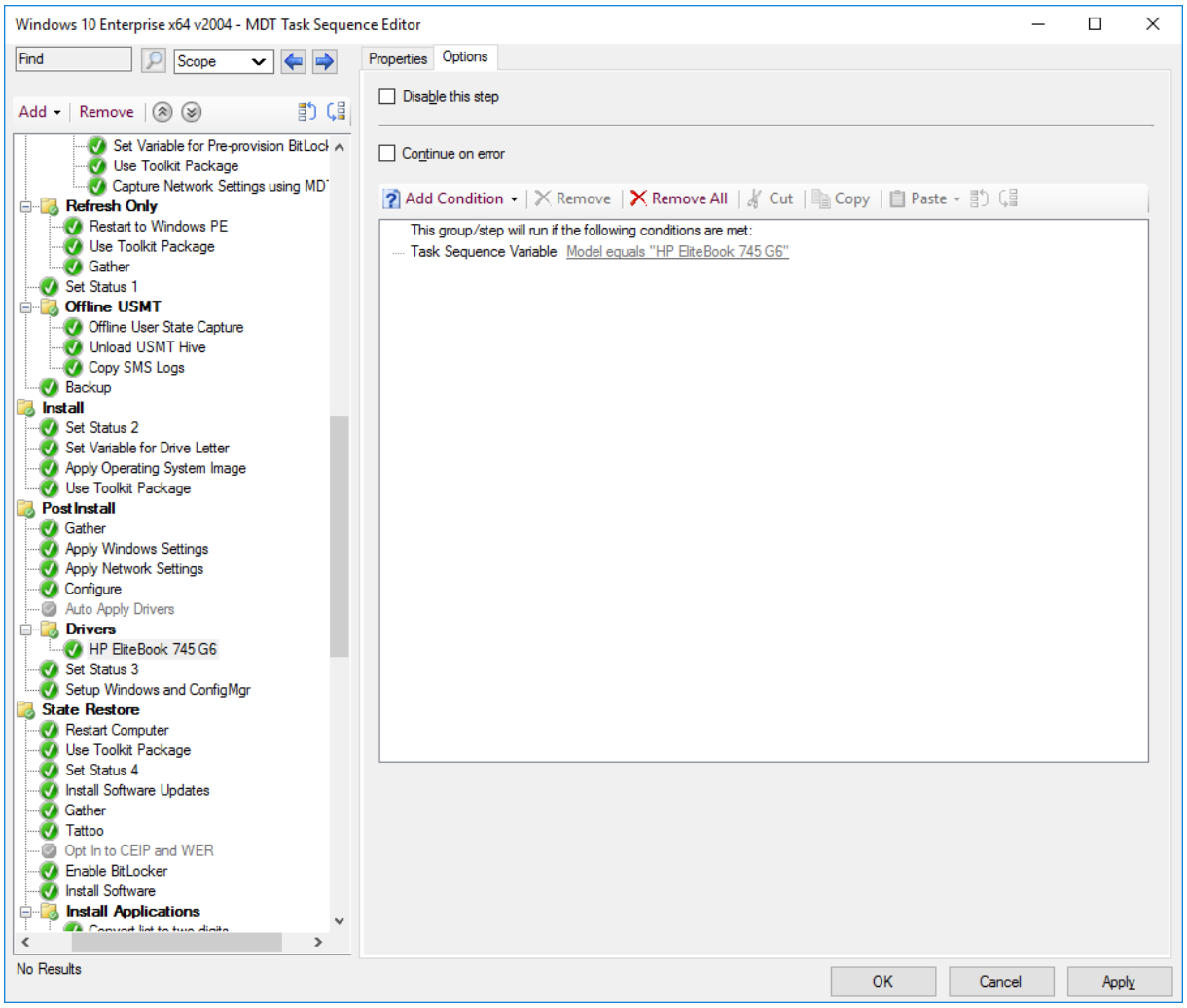
1. Using the **Configuration Manager Console**, select **Task Sequences**, right-click the **Windows 10 Enterprise x64 v2004 - MDT** task sequence and select **Edit**.
2. In the **Post Install** group, select **Apply Network Settings**, and configure the **Domain OU** value to use the **ViaMonstra / Workstations** OU (browse for values).
3. In the **PostInstall** group, disable the **Auto Apply Drivers** action. (Disabling is done by selecting the action and, in the Options tab, select the **Disable this step** check box.)

4. After the disabled **PostInstall / Auto Apply Drivers** action, add a new group named **Drivers**.
5. After the **PostInstall / Drivers** group, add a **Download Package Content** action with the following settings:
    a. Name: **HP EliteBook 745 G6**
    b. Add package: **Windows 10 x64 v2004 - HP EliteBook 745 G6**
    c. Custom path: **%_SMSTSMDataPath%\Drivers**



Adding drivers as standard packages.

    d. In **Options**, add a **Task Sequence Variable c**ondition with the following settings:
        a. **Model equals HP EliteBook 745 G6**



Setting the condition for the driver package.

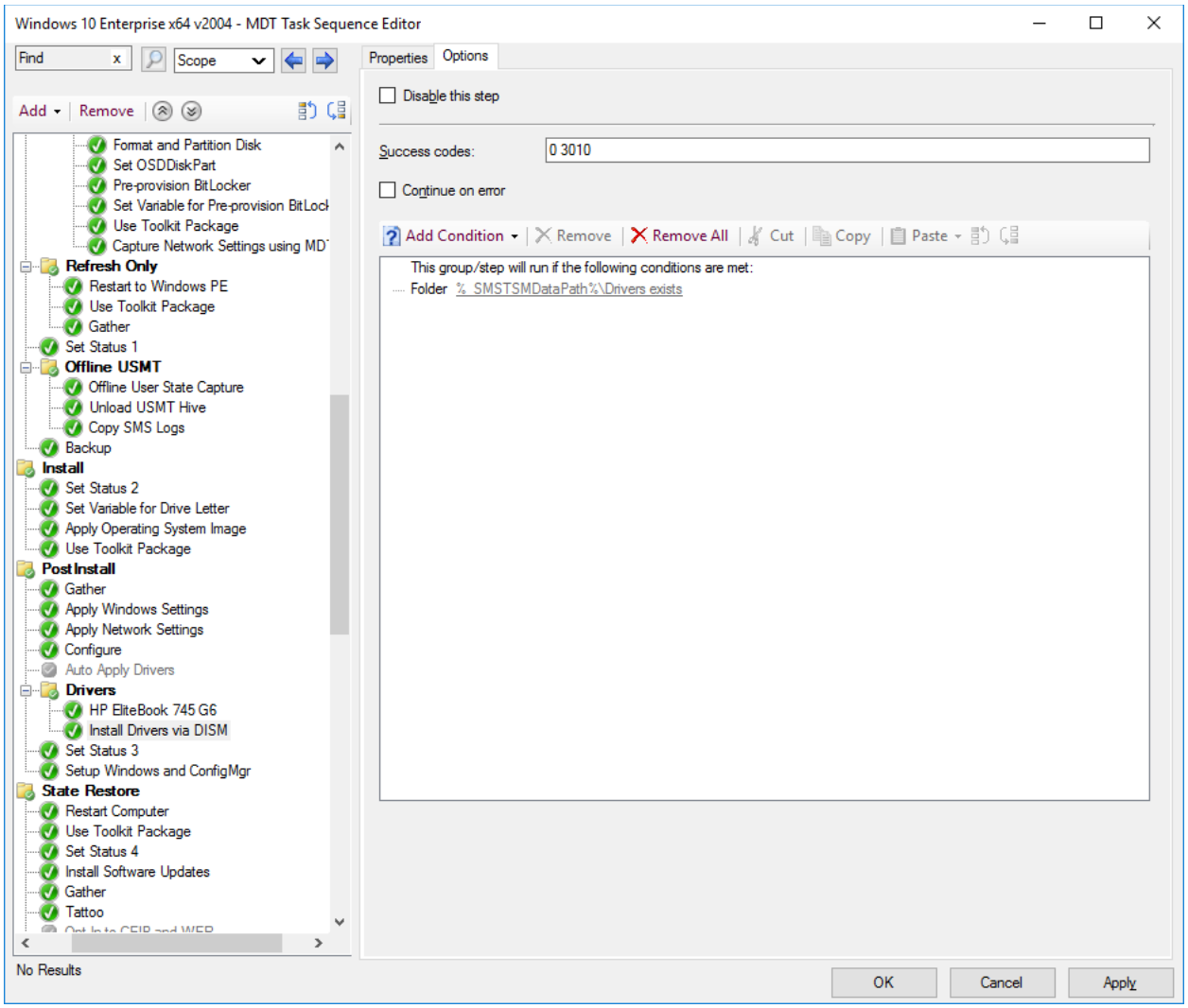
6. After the **HP EliteBook 745 G6** Download Package Content action, add a Run Command Line with the following settings:

    a. Name: **Install Drivers via DISM**

    b. Command line: **DISM.exe /Image:%OSDTargetSystemDrive%\ /Add-Driver /Driver:%_SMSTSMDataPath%\Drivers\ /Recurse /logpath:%_SMSTSLogPath%\dism.log**

    c. In **Options**, add a **Folder Properties c**ondition with the following settings:

        i. **%_SMSTSMDataPath%\Drivers exists**



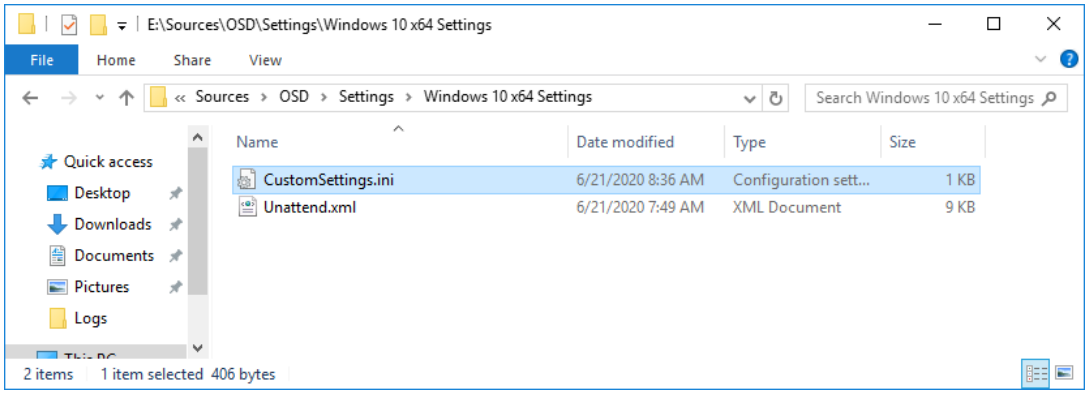
Adding drivers to the Task Sequence.

Note: Again, while this method of adding driver packages works great from technical point of view, I highly recommend adding the community developed Modern Driver Management solution available on https://msendpointmgr.com/modern-driver-management/

Configure the Windows 10 x64 Settings Package

1. Using **File Explorer**, copy the **E:\Setup\AGGW10-SampleFiles\ConfigMgr\Windows 10 Settings\Customsettings.ini** file to the following folder (Replace the existing file):

   E:\Sources\OSD\Settings\Windows 10 x64 Settings



The settings package, holding the rules and the Unattend.xml template used during deployment.

Update the Windows 10 x64 Settings package

1. Using the **Configuration Manager Console**, in the **Software Library** workspace, expand **Application Management**, and then select **Packages**.
2. Update the distribution point for the **Windows 10 x64 Settings** package by right-clicking the **Windows 10 x64 Settings** package, and select **Update Distribution Points**.

Note: Even though you not yet added a distribution point for this package, you still need to select update distribution points. That process, despite the name, also updates the content library with package changes.

Distribute content to the HQ DPs Distribution Point Group

1. Using the **Configuration Manager Console**, select **Task Sequences**, right-click the **Windows 10 Enterprise x64 v2004 - MDT** task sequence and select **Distribute Content**.
2. Use the following settings for the **Distribute Content Wizard**:

   Content Destination: Add the **HQ DPs** distribution point group.

3. Using **CMTrace**, verify the distribution to the **CM01** distribution point (part of the HQ DPs Group) by reviewing the **distmgr.log** file. Do not continue until you see all the new packages being distributed successfully.

Create a deployment for the Task Sequence

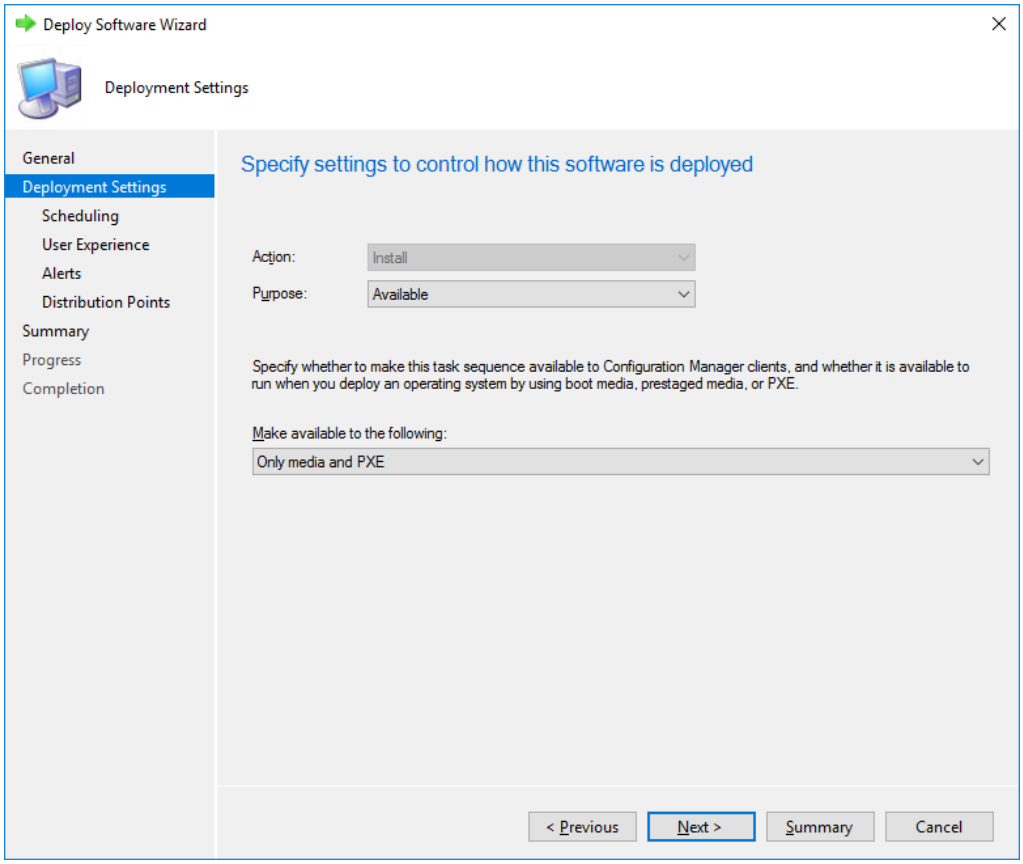
1. Using **Configuration Manager Console**, select **Task Sequences**, right-click **Windows 10 Enterprise x64 v2004 - MDT**, and then select **Deploy**.
2. Use the following settings for the **Deploy Software Wizard**:
   a. General

      Collection: **All Unknown Computers**

Note: Review the high risk deployment dialog box that is displayed when browsing for collections. This feature was added already back in ConfigMgr 2012 R2 SP1 and can be configured on the site properties.

   b. Deployment Settings
      - Purpose: **Available**
      - Make available to the following: **Only media and PXE**

   c. Use the default settings for the remaining wizard pages.



Configuring the deployment settings.

Step 6 – Deploy Windows 10 using PXE

Deploy the Windows 10 Image

1. On the **Host PC**, start the **PC0001** virtual machine, when the PXE menu is displayed, press **Enter** to PXE boot when prompted (you need to be quick).
2. After WinPE has been started, complete the deployment wizard using the following settings:
    a. Password: **P@ssw0rd**
    b. Select a task sequence to execute on this computer:
    Windows 10 Enterprise x64 v2004 or **Windows 10 Enterprise x64 v2004 - MDT** depending on what task sequence you created.
3. The Task Sequence will now run and do the following:
    a. Install the Windows 10 operating system
    b. Inject the driver package
    c. Install the ConfigMgr Client
    d. Join the machine to the domain
4. After **PC0001** has been deployed, login and change the computer name to **PC0001**.

Note: Since you didn't add any computer naming options, or pre-staged the machine prior to deployment, ConfigMgr by default assigns a random MININT-XXXXXX type name to the machine.

Module 6

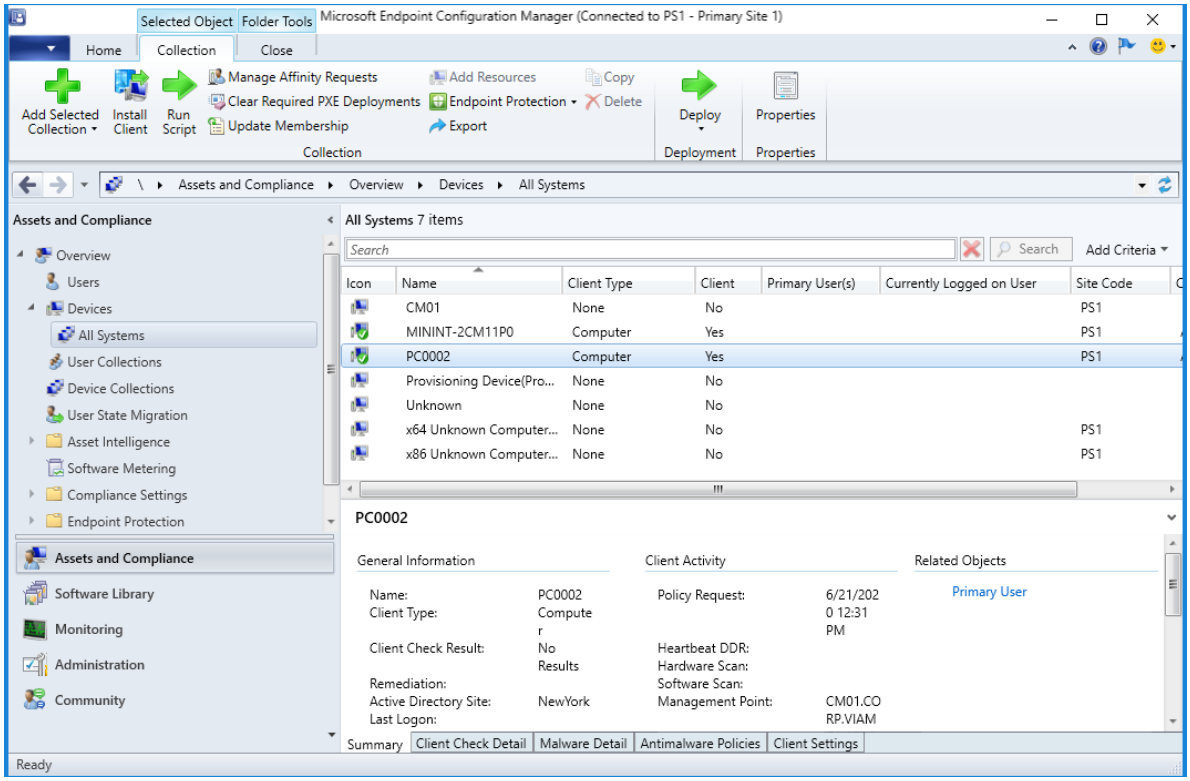
Using ConfigMgr for Windows 10 Servicing

Step 1 – Service/Upgrade PC0002 to Windows 10 Enterprise v2004

Deploy the ConfigMgr Client

1. On **PC0002**, log on as **VIAMONSTRA\Administrator**.
2. Install the ConfigMgr agent by navigating to **\\CM01\SMS_PS1\Client** and run the **ccmsetup.ex**e file.
3. Wait until the ConfigMgr client setup is fully completed. Wait for the **C:\ccmsetup\Logs\ccmsetup.log** to create the entry: **CcmSetup is exiting with return code 0**.
4. On **CM01**, using the **Configuration Manager Console**, in the **Assets and Compliance** workspace, select **Device** Collections, and then double-click the **All Systems** collection. **PC0002** should now display an active client in the **PS1** site.

Note: It may take a little while before the machine shows up as active (green icon). Pressing F5 or clicking Refresh will sometimes do magic :)

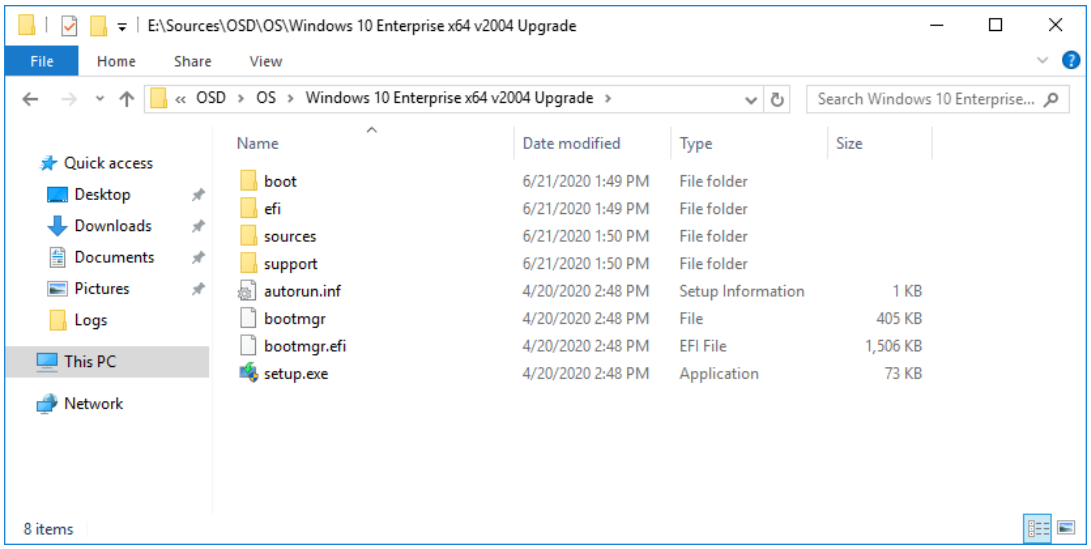


The ConfigMgr console showing PC0002 having an active client.

Create the Windows 10 Operating System Upgrade Package

1. On **CM01**, using **File Explorer**, create the **E:\Sources\OSD\OS\ Windows 10 Enterprise x64 v2004 Upgrade** folder.
2. Copy the content of **\\PC0002\C$\OSDBuilder\OSBuilds\Windows 10 Enterprise x64 v2004 19041.329\OS** to the **E:\Sources\OSD\OS\ Windows 10 Enterprise x64 v2004 Upgrade** folder.

Note: Again, the version number in OSDBuilder\OSBuilds folder is going to be different depending on what monthly updated that was injected. If you didn't complete module 3, you can also copy the content from an updated Windows 10 Enterprise x64 v2004 ISO from Microsoft.



Windows 10 installation media added.

3. Using **Configuration Manager Console**, in the **Software Library** workspace, expand the **Operating Systems** node, and add an **Operating System Upgrade Package** with the following settings.

    a. Data Source

        - Path (click Browse): **\\CM01\Sources\OSD\OS\ Windows 10 Enterprise x64 v2004 Upgrade**
        - Architecture: **x64**

    b. General

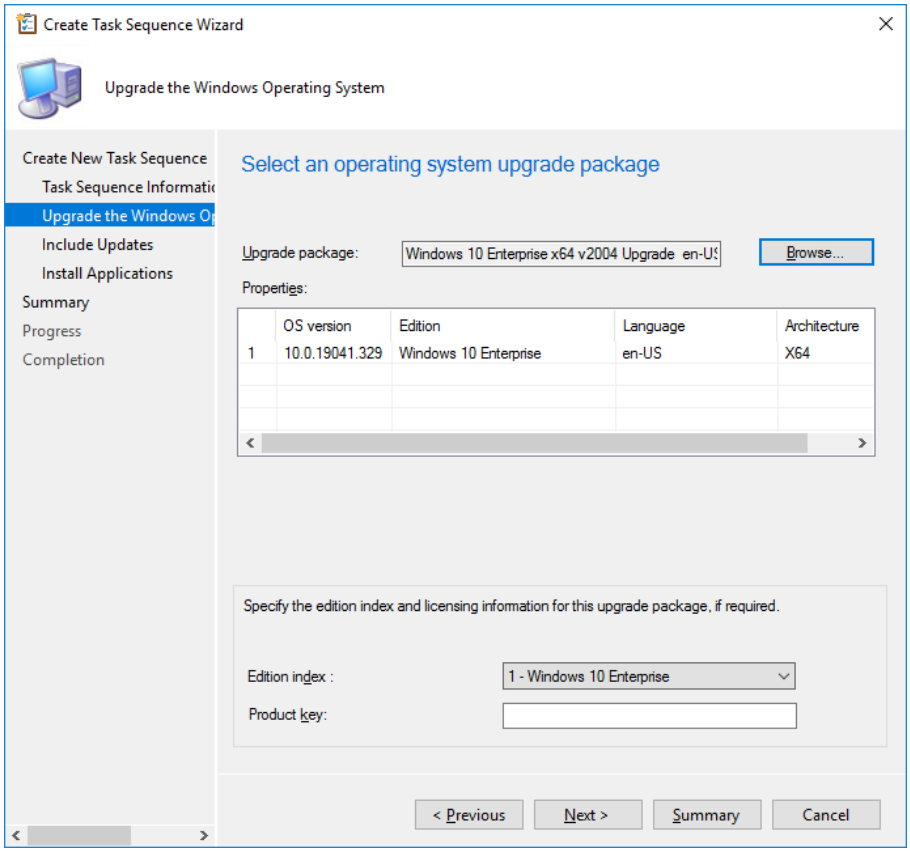
        - Name: **Windows 10 Enterprise x64 v2004 Upgrade**

    c. Use the default settings for the remaining wizard pages.

Create a Windows 10 Upgrade Task Sequence

1. Using **Configuration Manager Console**, in the **Software Library** workspace, expand **Operating Systems**, right-click **Task Sequences**, and select **Create Task Sequence**.
2. Use the following settings for the **Create Task Sequence** wizard.
    a. Create a new task sequence: **Upgrade an operating system from an upgrade package**
    b. Name: **Windows 10 Enterprise x64 v2004 Upgrade**
    c. **Run as high performance power plan**
    d. Upgrade package: **Windows 10 Enterprise x64 v1909 Upgrade**
    e. Edition index: **1 - Windows 10 Enterprise**

Note: The index number is going to be different depending on what media you are using. Images created from OSD Builder typically only have one index, whereas the Microsoft ISO will have several. Also, if using an image/media with several indexes, I recommend that you also provide the product key for the version of Windows you want to use.

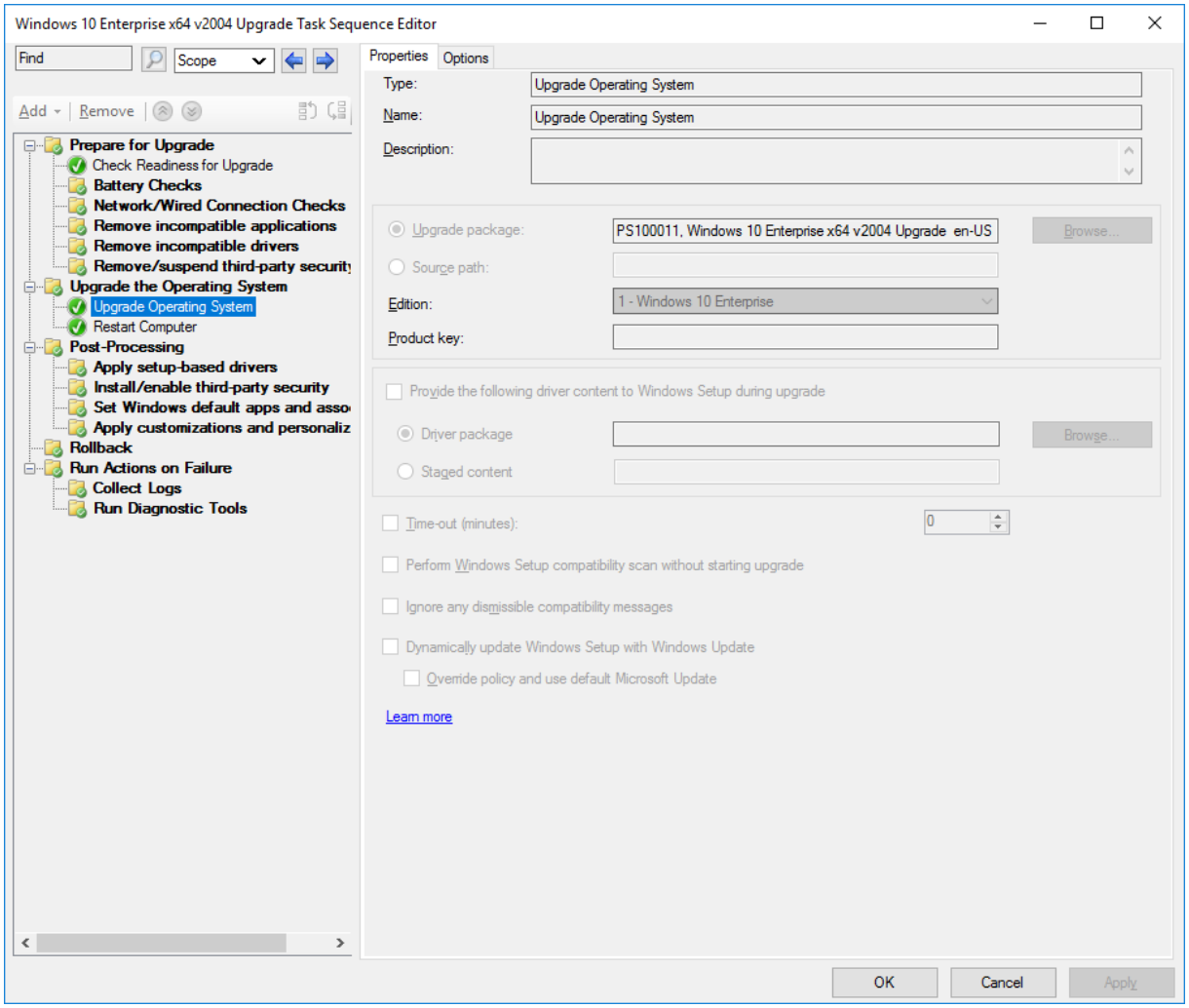


Creating the Windows 10 upgrade task sequence.

    f. Include software updates: **Do not install any software updates**
    g. Use the default settings for the remaining wizard pages.

Review the Task Sequence

1. Using the **Configuration Manager Console**, select **Task Sequences**, right-click the **Windows 10 Enterprise x64 v2004 Upgrade** task sequence and select **View**.
2. Review the various actions in the task sequence, then click **Cancel**.



The Windows 10 Enterprise x64 v2004 Upgrade task sequence in view mode.

Distribute content to the HQ DPs Distribution Point Group

1. Using the **Configuration Manager Console**, select **Task Sequences**, right-click the **Windows 10 Enterprise x64 v2004 Upgrade** task sequence and select **Distribute Content**.
2. Use the following settings for the **Distribute Content Wizard**:

   Content Destination: Add the **HQ DPs** distribution point group.
3. Using **CMTrace**, verify the distribution to the **CM01** distribution point (part of the HQ DPs Group) by reviewing the **distmgr.log** file. Do not continue until you see all the new packages being distributed successfully.

Create a Device Collection, and add the PC0002 computer

1. On **CM01**, using the **Configuration Manager Console**, in the **Assets and Compliance** workspace, right-click **Device Collections**, and then select **Create Device Collection**. Use the following settings.
    a. Name: **Windows 10 Enterprise x64 v2004 Upgrade**
    b. Limited Collection: **All Systems**
    c. Membership rules:
        - **Direct rule**
        - Resource Class: **System Resource**
        - Attribute Name: **Name**
        - Value: **PC0002**
        - Select Resources
        - Select **PC0002**
2. Review the **Windows 10 Enterprise x64 v2004 Upgrade** collection. Don't continue until you see the **PC0002** machine in the collection.

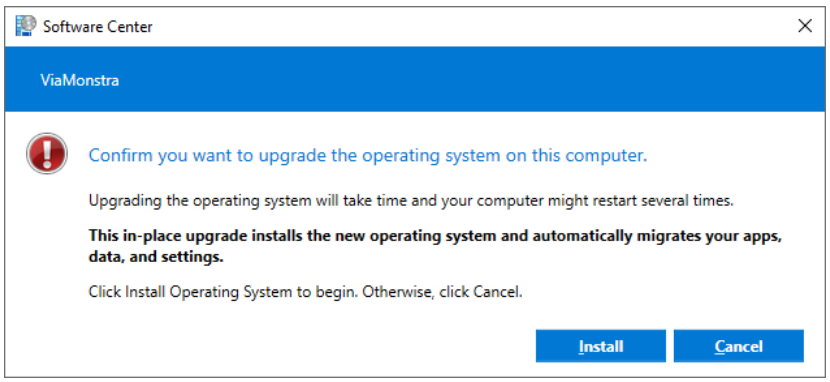
Create a new Deployment

1. Using **Configuration Manager Console**, in the **Software Library** workspace, select **Task Sequences**, right-click **Windows 10 Enterprise x64 v2004 Upgrade**, and then select **Deploy**.
2. Use the following settings for the **Deploy Software Wizard**:
    a. General
    b. Collection: **Windows 10 Enterprise x64 v2004 Upgrade**
    c. Deployment Settings
    d. Purpose: **Available**
    e. Use the default settings for the remaining wizard pages.

Initiate the Windows 10 Inplace Upgrade

1. On **PC0002**, using **Software Center**, select the **Windows 10 Enterprise x64 v2004 Upgrade** deployment, click **Install**, and then click **Install** again.

Note: If you don't see the deployment in Software Center, make sure that multiple users are not logged in, and force a machine policy update if needed.



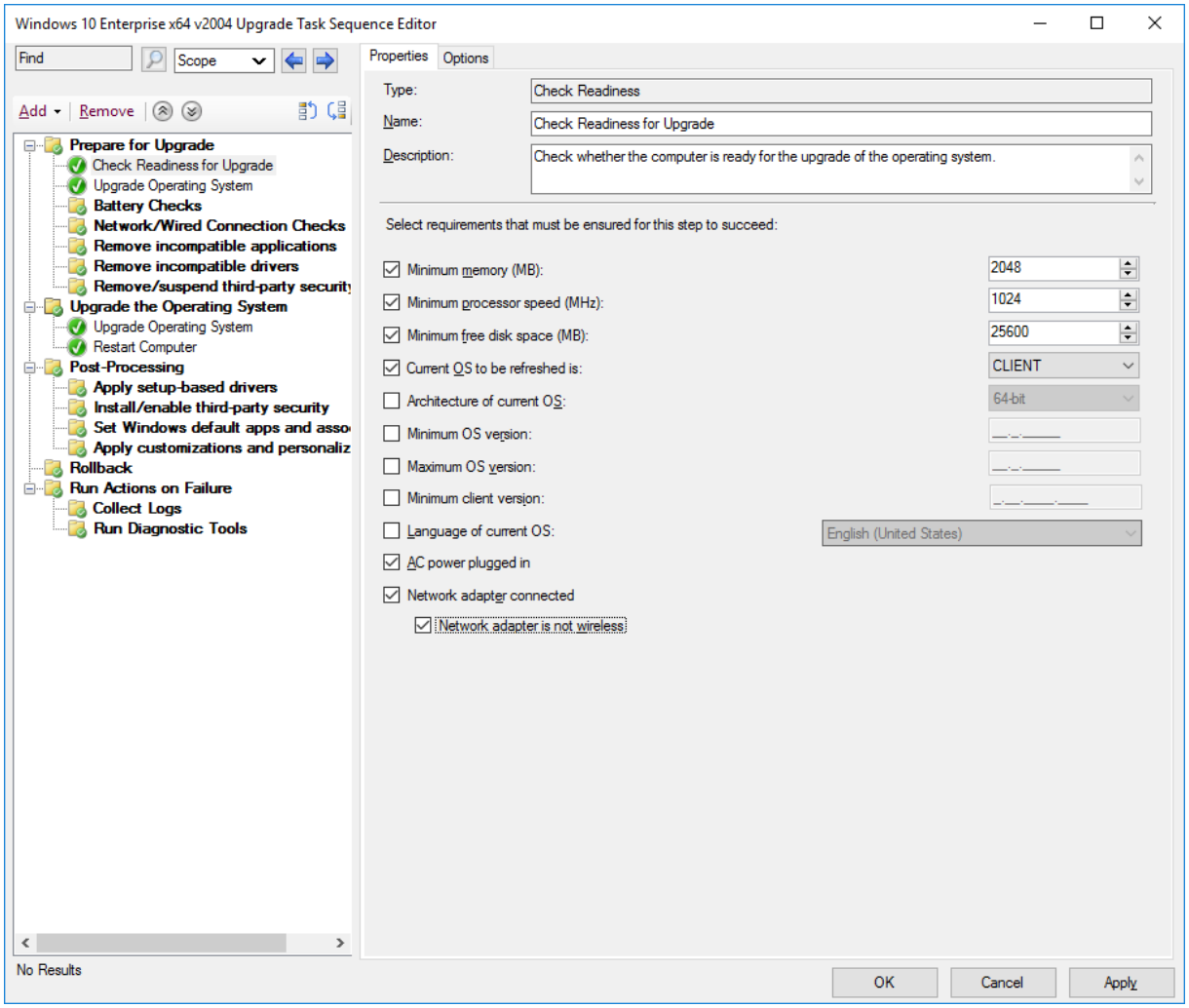
Running the Windows 10 upgrade via Software Center.

2. After the upgrade has completed, log on as an administrator, and verify that the ConfigMgr client is still operational (e.g. not in provisioning mode). Check the HKLM\SOFTWARE\Microsoft\CCM\CcmExec\ProvisioningMode value, it should be false.

Step 2 - Improving the single ConfigMgr Upgrade Task Sequence

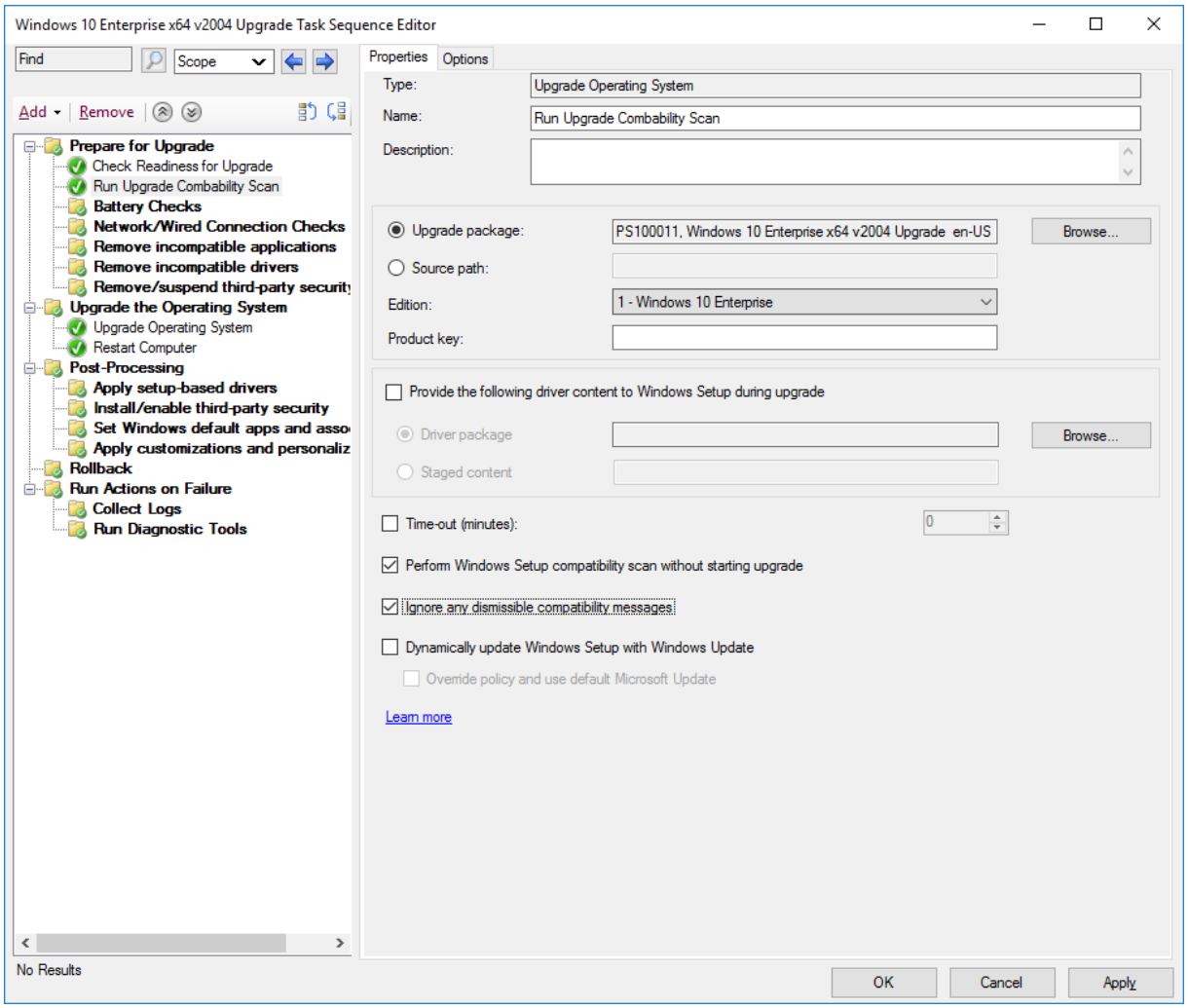
Adding Setup Upgrade Assessment and Driver support

1. On **CM01**, edit the **Windows 10 Enterprise x64 v2004 Upgrade** task sequence.
2. In the Check Readiness for Upgrade action, enable the following additional requirements
    - **AC power plugged in**
    - **Network adapter connected**
        - **Network adapter is not wireless**



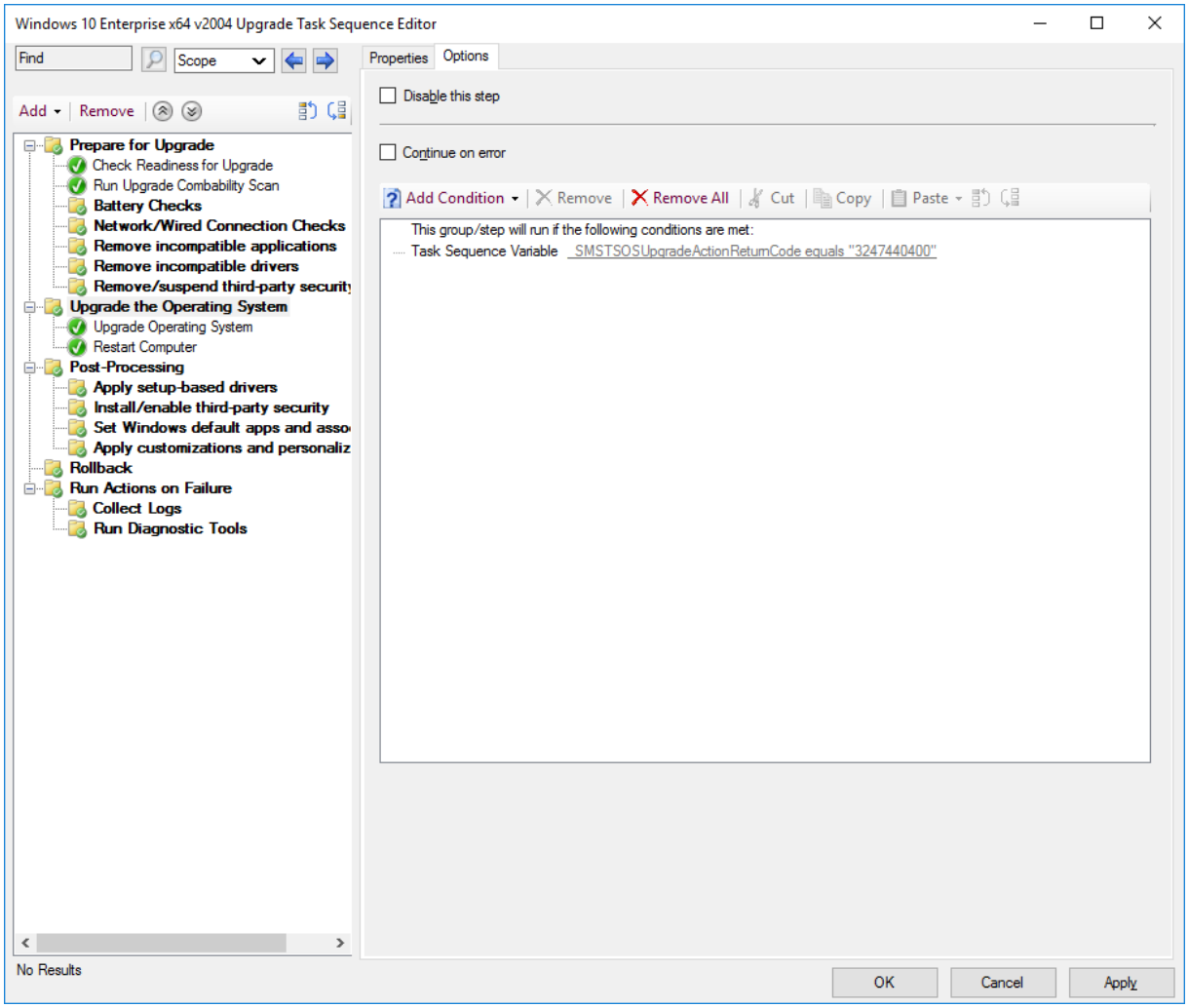
Modifying the upgrade requirements.

3. After the Check **Readiness for Upgrade** action, paste a copy of the **Upgrade Operating System** action, configure it to **continue on error**, and rename it to **Run Upgrade Combability Scan**.
4. In the new **Run Upgrade Combability Scan** action select the following check boxes:
    - **Perform Windows Setup Compatibility scan without starting upgrade**
    - **Ignore any dismissible compatibility messages**



Adding the Run Upgrade Combability Scan action.

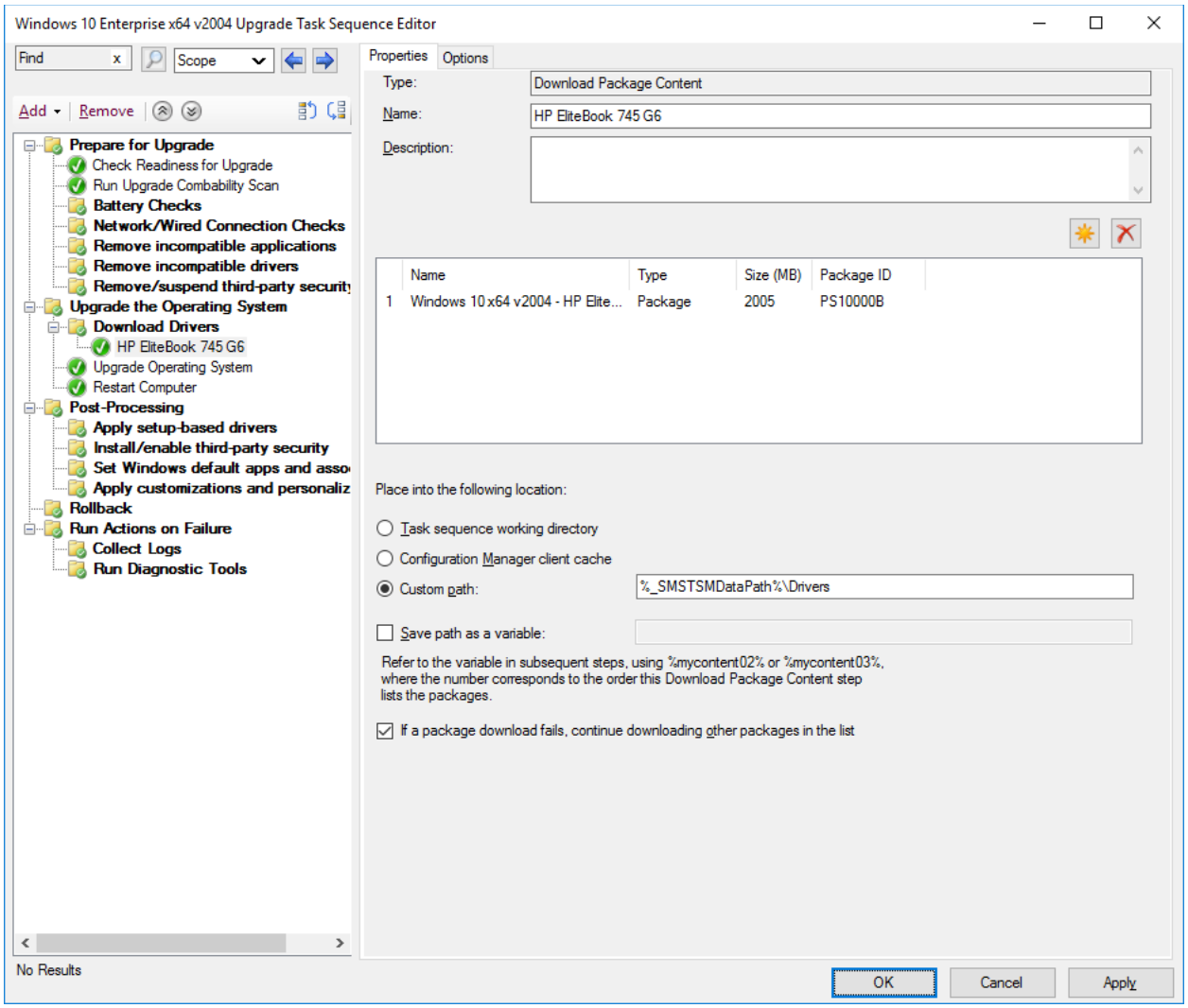
5. Modify the **Upgrade the Operating System** group to use a task sequence variable as condition: Add **_SMSTSOSUpgradeActionReturnCode**, and set the value to **3247440400**.



Configuring the Upgrade the Operating System group with a condition.

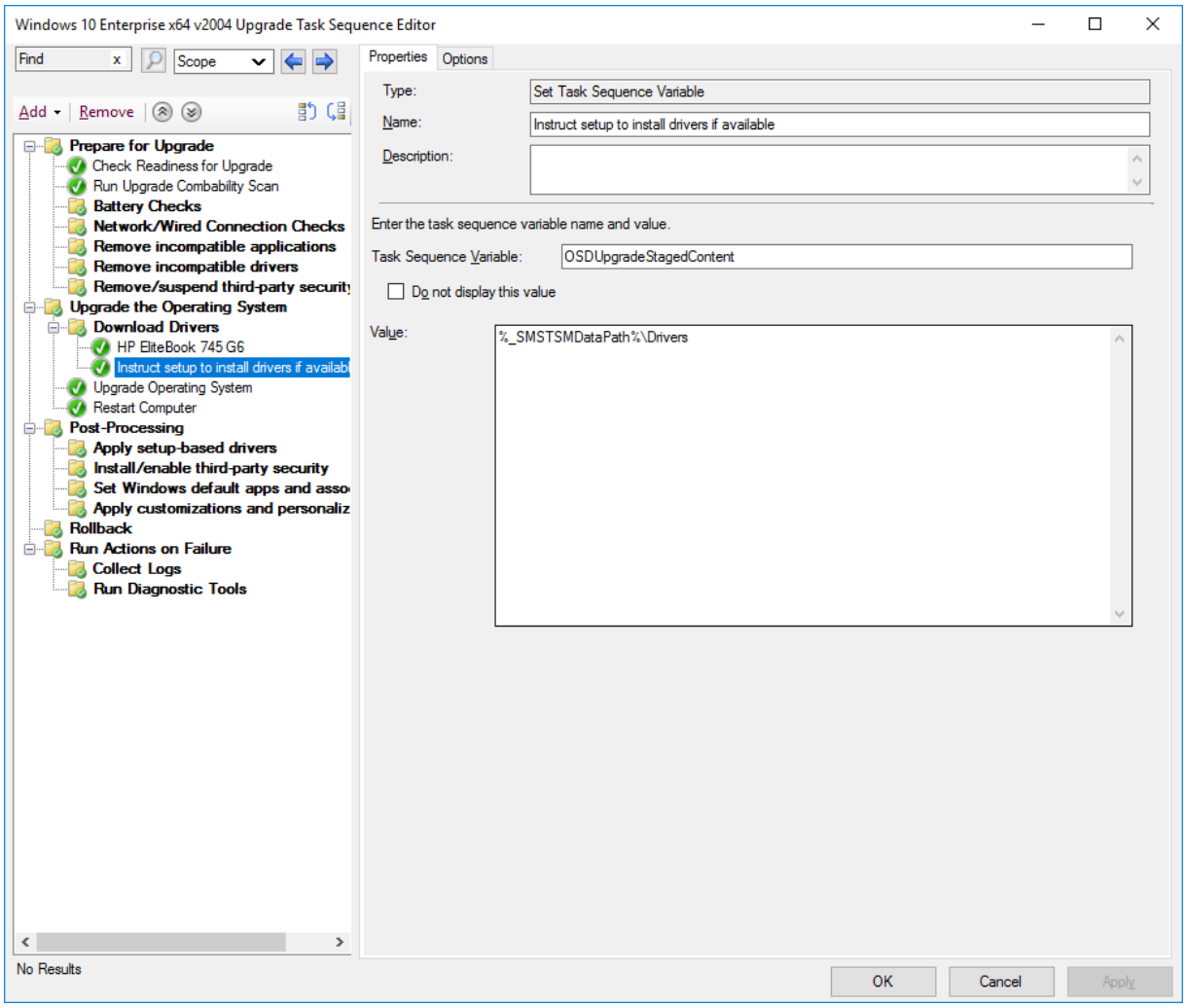
6. In the **Upgrade the Operating System** group, add a new group named **Download Drivers**.
7. add a **Download Package Content** action with the following settings:
    a. Name: **HP EliteBook 745 G6**
    b. Add package: **Windows 10 x64 v2004 - HP EliteBook 745 G6**
    c. Custom path: **%_SMSTSMDataPath%\Drivers**
    d. In the **Options** tab, add a **Query WMI c**ondition with the following WQL query:
        - **Select * from win32_computersystem where model = 'HP EliteBook 745 G6'**

Note: Since the upgrade task sequences in ConfigMgr are not integrated with MDT by default, you can't just use a task sequence variable like Model. That's why you use a WMI Query in the preceding step.



Adding Download Package Content actions with driver packages.

8. After the **HP EliteBook 745 G6** Download Package Content action, add a **Set Task Sequence Variable** action with the following settings:
    a. Name: **Instruct setup to install drivers if available**
    b. Task Sequence Variable: **OSDUpgradeStagedContent**
    c. Value: **%_SMSTSMDataPath%\Drivers**
    d. In the **Options** tab, add a **Folder Properties** condition for the **%_SMSTSMDataPath%\Drivers** folder.



Instruct setup to use drivers, if they exist.

Note: In this task sequence you're using the undocumented OSDUpgradeStagedContent variable, since that's the variable the Upgrade Operating System action is using by default. Technically you can also use the OSDSetupAdditionalUpgradeOptions variable, and add the /installdrivers % SMSTSMDataPath%\Drivers value, but the OSDSetupAdditionalUpgradeOptions is primarily intended for the /reflectdrivers option and for language packs.

Optional – Run the modified Windows 10 Enterprise x64 v2004 Upgrade Task Sequence

1. If you want to verify this modified upgrade task sequence, add another Windows 10 v1909 machine to your lab.
2. Install the ConfigMgr agent on it, and add it to the Windows 10 Enterprise x64 v2004 Upgrade collection.
3. Then using **Software Center**, run the **Windows 10 Enterprise x64 v2004 Upgrade** task sequence.

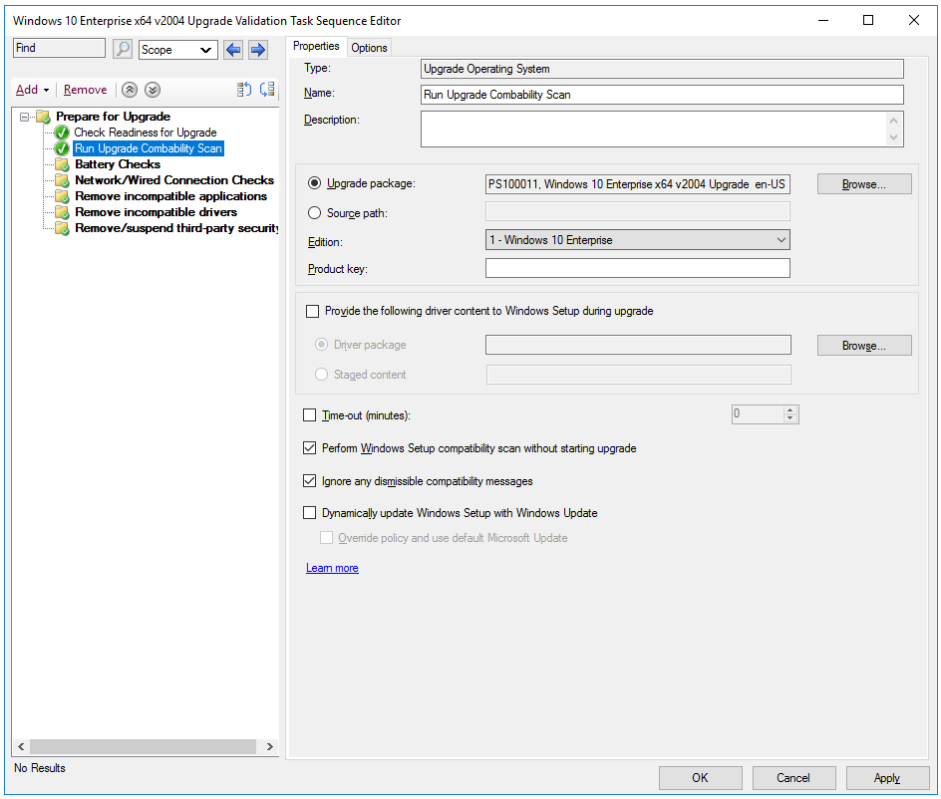
Step 3 – Splitting the ConfigMgr Upgrade Task Sequence

Copy existing task sequence (twice)

1. On **CM01**, copy the **Windows 10 Enterprise x64 v2004 Upgrade** task sequence, and rename the copy to **Windows 10 Enterprise x64 v2004 Upgrade Validation**.
2. Make another copy the **Windows 10 Enterprise x64 v2004 Upgrade** task sequence, and rename that copy to **Windows 10 Enterprise x64 v2004 Upgrade Prod**.

Edit the Windows 10 Enterprise x64 v2004 Upgrade Validation task sequence

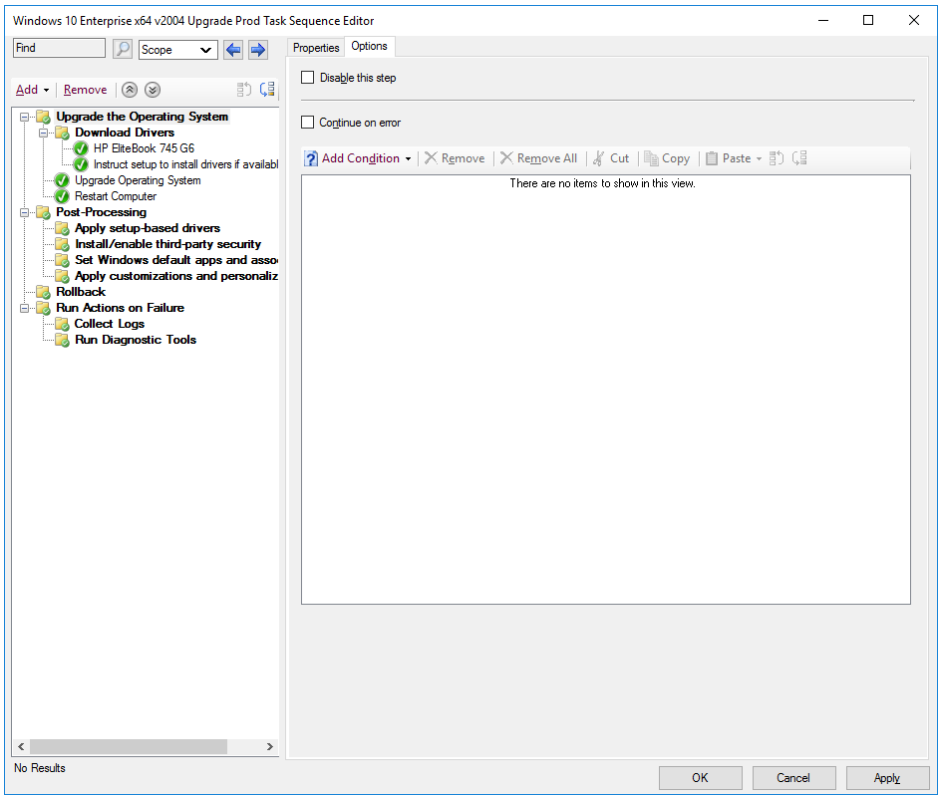
1. On **CM01**, edit the **Windows 10 Enterprise x64 v2005 Upgrade Validation** task sequence, and remove the following sections:
    - Upgrade the Operating System
    - Post-Processing
    - Rollback
    - Run Actions on Failure (only available in ConfigMgr 1902 or higher)



The Windows 10 Enterprise x64 v2004 Upgrade Validation task sequence.

Edit the Windows 10 Enterprise x64 v2004 Upgrade Prod task sequence

1. On **CM01**, edit the **Windows 10 Enterprise x64 v2004 Upgrade Prod** task sequence.
2. Remove the **Prepare for Upgrade** section.
3. Remove the condition on the **Upgrade the Operating System** group.



The Windows 10 Enterprise x64 v2004 Upgrade Prod task sequence.

Create the Inplace Upgrade collections and deploy the task sequences

1. On **CM01**, create a device collection named **Windows 10 Enterprise x64 v2004 Upgrade Validation**. Don't add any members.
2. Deploy the **Windows 10 Enterprise x64 v2004 Upgrade Validation** task sequence as an available deployment to the **Windows 10 Enterprise x64 v2004 Upgrade Validation** collection.
    - Make sure to enable Pre-download content option.
    - Make a note of the Deployment ID (enable the Deployment ID column under deployments to see your deployment ID)

- Create a device collection named **Windows 10 Enterprise x64 v2004 Upgrade NOT Ready**. Add the following query rule (replace the AdvertisementID with your DeploymentID:

```
select
SMS_R_SYSTEM.ResourceID,SMS_R_SYSTEM.ResourceType,SMS_R_SYSTEM.Na
me,SMS_R_SYSTEM.SMSUniqueIdentifier,SMS_R_SYSTEM.ResourceDomainOR
Workgroup,SMS_R_SYSTEM.Client from SMS_R_System where
SMS_R_System.ResourceId in (select ResourceID from
SMS_ClientAdvertisementStatus where AdvertisementID = "PS120004"
and LastStatusMessageID=11170)
```

3. Create a device collection named **Windows 10 Enterprise x64 v2004 Upgrade Ready**. Add the following query rule (replace the AdvertisementID with your DeploymentID:

```
select
SMS_R_SYSTEM.ResourceID,SMS_R_SYSTEM.ResourceType,SMS_R_SYSTEM.Na
me,SMS_R_SYSTEM.SMSUniqueIdentifier,SMS_R_SYSTEM.ResourceDomainOR
Workgroup,SMS_R_SYSTEM.Client from SMS_R_System where
SMS_R_System.ResourceId in (select ResourceID from
SMS_ClientAdvertisementStatus where AdvertisementID = "PS120004"
and LastStatusMessageID=11171)
```

4. Deploy the **Windows 10 Enterprise x64 v2004 Upgrade Prod** task sequence as an available deployment to the **Windows 10 Enterprise x64 v2004 Upgrade Ready** collection.

Optional – Run the modified Windows 10 Enterprise x64 v2004 Upgrade Validation Task Sequence

1. If you want to verify this modified upgrade task sequence, add another Windows 10 v1909 machine to your lab.
2. Install the ConfigMgr agent on it, and add it to the Windows 10 Enterprise x64 v2004 Upgrade collection.
3. Then using **Software Center**, run the **Windows 10 Enterprise x64 v2004 Upgrade Validation** task sequence, and wait until it completes.
4. On **CM01**, check the membership of the **Windows 10 Enterprise x64 v2004 Upgrade Ready** collection. This collection should now have the client as a member of that collection. Assuming the validation task sequence was successful.

Beyond the eBook

If you liked this eBook, I bet you will like any of the events and trainings I'm part of.

Live Presentations

I frequently speak at tech conferences around the world, such as Midwest Management Summit (MMS), AppManagEvent, Nordic Infrastructure Conference (NIC), and Microsoft Ignite. For current tour dates and presentations, see my blog:

- https://deploymentresearch.com

Video Training

For video-based training, see the following site:

- https://viamonstra.com

Live Instructor-led Classes

I present scheduled instructor-led classes in the US and in Europe. For current dates, see see my blog:

- https://deploymentresearch.com

Twitter and LinkedIn

I would love to connect with you on Twitter and/or LinkedIn, so please reach out:

- Twitter: @jarwidmark
- LinkedIn: https://www.linkedin.com/in/jarwidmark

Facebook

And finally, yes, the Deployment Research blog have a Facebook page as well.

- https://www.facebook.com/deploymentresearch