adaptiva

ConfigMgr Driver Management Primer

From Total Chaos to Total Control



Table of Contents

Table of Contents **2**

Introduction **3**

Driver Management Theory 3

Collecting Drivers **3**

PowerShell 3

Microsoft Update Catalogue 4

Major Vendors' Enterprise Cab Sites 7

HP 7

Dell 7

Lenovo 7

SCConfigMgr Driver Automation Tool 7

Auto Apply Drivers (Total Chaos) **8**

Auto Apply Drivers (Added Predictability) **10**

Total Control **12**

Apply Driver Packages 12

Regular Packages for Drivers 13

Managing Drivers during In-Place Upgrade **16**

Modern Driver Management **18**

Holy Grail – Kim Oppalfens 18

SCConfigMgr Driver Automation Tool 18

Updating Drivers **19**

Cleanup **19**

Glossary **19**

Resources **20**

Introduction

This e-book is organized for busy IT pros. I assume you already know basic system administration concepts for both Microsoft Windows and Microsoft System Center Configuration Manager (ConfigMgr). Rather than rehash information easily available elsewhere, I build on your existing skills to teach you driver management in ConfigMgr.

Driver Management Theory

ConfigMgr luminary Johan Arwidmark wrote a popular blog post about managing drivers. You may remember it. In it, he outlines three major ways admins organize their drivers:

- **Total Chaos** – a bit like roulette where you're managing drivers by closing your eyes and crossing your fingers. I'm sure you know the feeling of hoping the PnP gods are on your side. If they are that day, they'll pick not only the first match, but the actual match. Total chaos leaves too much to chance and should be eliminated.
- **Added predictability**– a little bit less like roulette because you at least put some thought into how you organized the trap. You're still depending on PnP detection, which still isn't very wise.
- **Total Control** – aka the super shiny, varsity-level, driver organization/management workflow. You've created a package for the model you're deploying with that model's actual correct drivers. Maybe you grabbed them by running a PowerShell script, downloading the enterprise CAB file, or some other shiny way. You know that the only acceptable way to deploy hardware would be the right way. The right way is total control.

Collecting Drivers

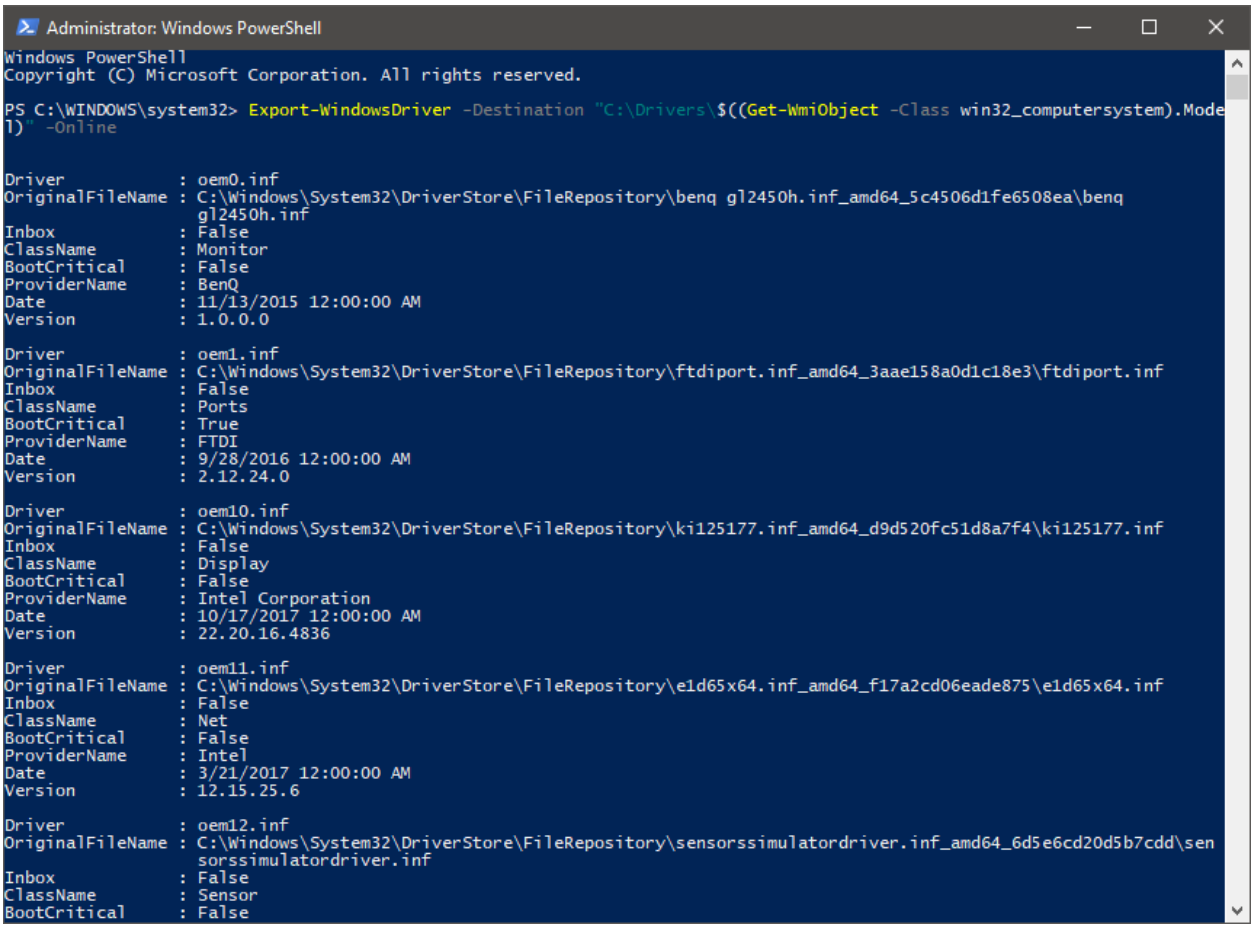
Collecting all the drivers you need to manage can be a challenge in itself. I've listed some key options below, including: PowerShell, the Microsoft Update Catalogue, major vendors' enterprise cab sites, and the SCConfigMgr Driver Automation Tool.

PowerShell

Sometimes gathering the vendor package for deployment of a device isn't possible. This is generally the case if the PC in question is a consumer model which is not intended for deployment. Other times admins don't have time to look for the package. Maybe the technician doesn't want to use the vendor package for any number of reasons but would rather create a custom one.

It is possible to extract the device drivers using PowerShell. To collect the drivers using PowerShell to create your own custom package, run the following command:

```
Export-WindowsDriver -Destination "C:\Drivers\$((Get-WmiObject -Class win32_computersystem).Model)" -Online
```

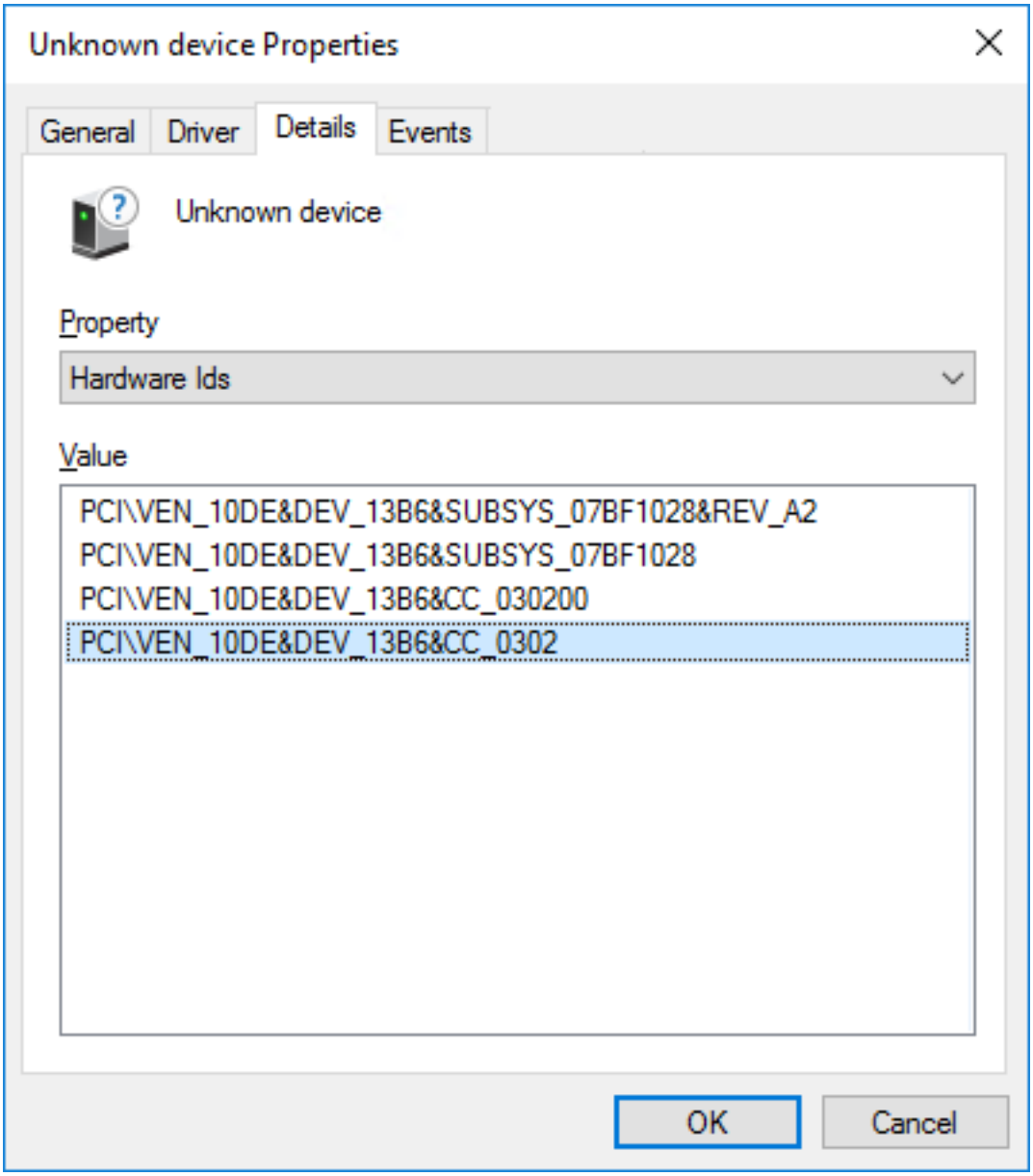


You'll need to remove any drivers ending in .prn as those are inbox printer drivers and are not needed at deployment time. For more information, please check Mikael Nystrom's blog post on the topic.

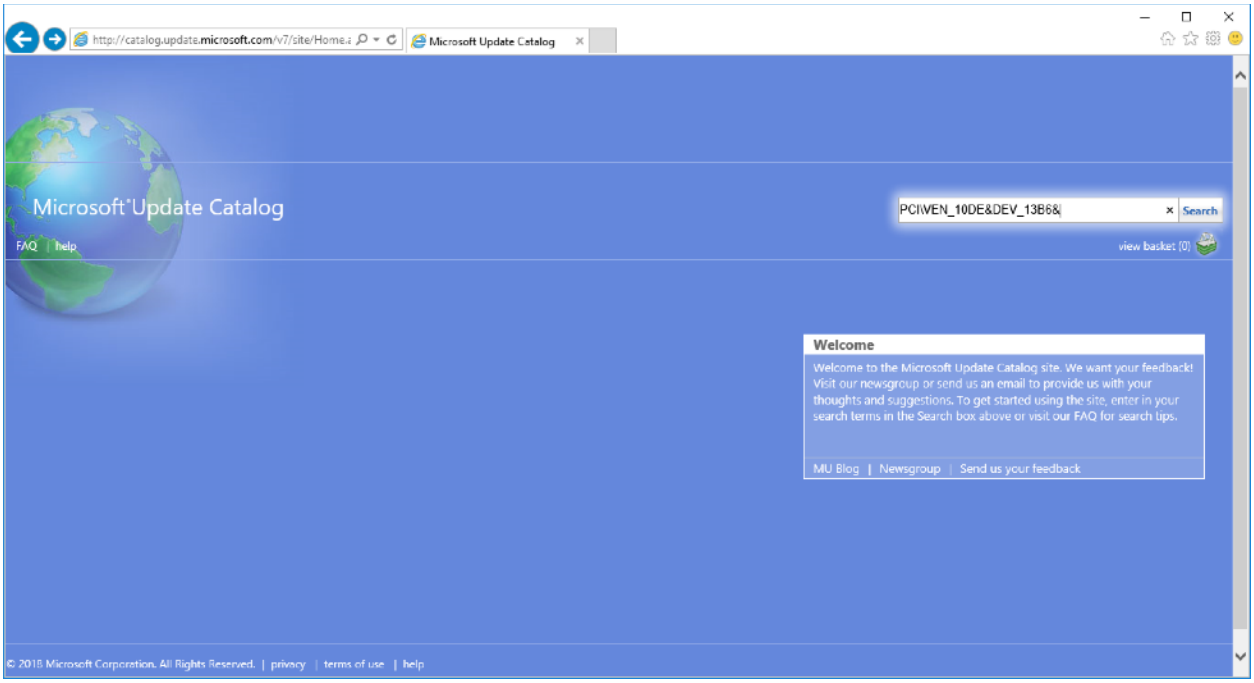
Microsoft Update Catalogue

When a laptop is deployed without a driver for a device, it is possible to use Device Manager in conjunction with the Microsoft Update Catalogue to find the driver in question. Once the driver is obtained, it can be imported into a driver repository for use during deployment. To find a driver, open Device Manager and go to the device in question. Typically, it is listed as **Unknown device** in the **Other devices** node.

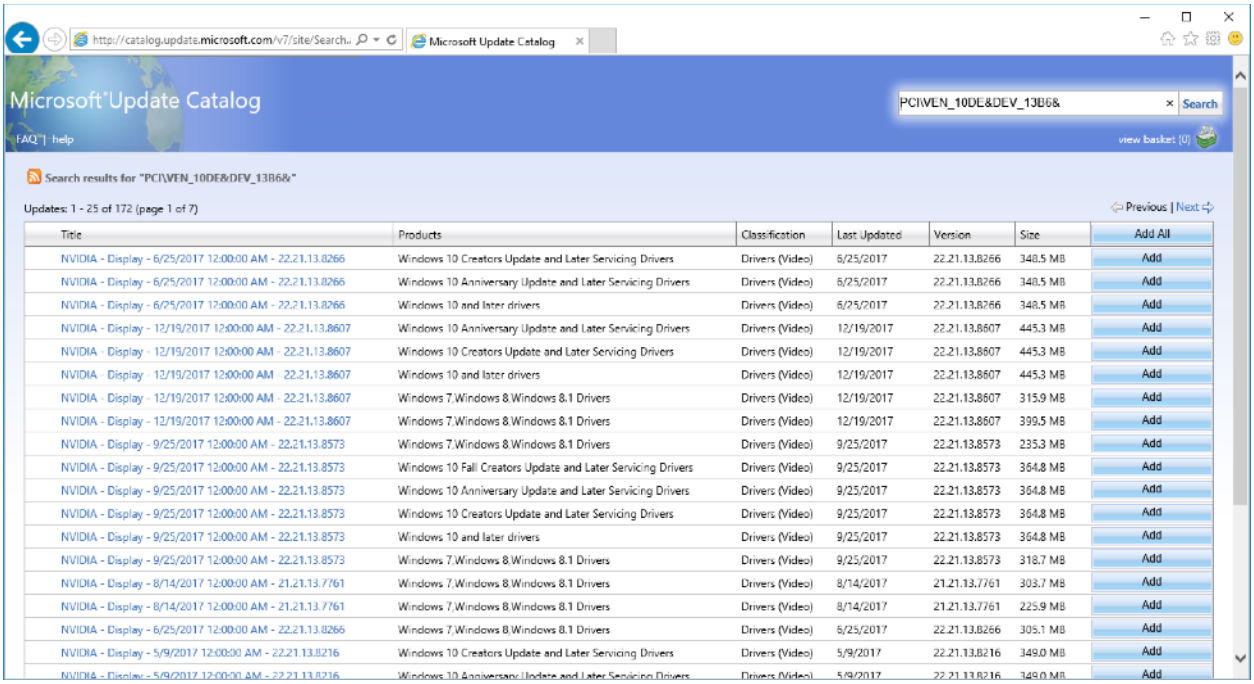
Open the device properties and navigate to the **Details** tab. In the **Property** list select **Hardware Ids**. Here you will get a single value or a list of values. Select the last value in the list by clicking on it. Then copy it to your clipboard (hit Ctrl+C as you won't be able to right-click).



Open a browser and navigate to http://catalog.update.microsoft.com and paste the value into the search bar, then delete everything after the last ampersand ('&') and click **Search**.



From the search, it's easy to see what device this value returns. Pick the driver needed based on product, date, or version, and download the file.



Sometimes, a device will not return a match in the catalog but that is not the norm. In those situations, do a Google search with the Hardware ID and the product name looking for links to the manufacturer's support forum. Please be very cautious when searching for drivers using search engine. Pay attention to where links lead and only obtain drivers from reputable sources.

Major Vendors' Enterprise Cab Sites

HP, Dell, and Lenovo have done a fantastic job of providing technicians with driver cabs for use during deployment. This is often much better than any recovery media the device might ship with and is faster than downloading device drivers individually from a model's product page.

HP

HP Driver Packs (64-bit)

http://ftp.hp.com/pub/caps-softpaq/cmit/HP_Driverpack_Matrix_x64.html

HP Driver Packs (64-bit)

http://ftp.hp.com/pub/caps-softpaq/cmit/HP_Driverpack_Matrix_x86.html

Dell

Driver CABs for Dell Enterprise class systems (Latitude, Optiplex, Precision)

http://en.community.dell.com/techcenter/enterprise-client/w/wiki/2065.dell-command-deploy-driver-packs-for-enterprise-client-os-deployment

Lenovo

Device driver packs that can be used with ConfigMgr and/or Microsoft Deployment Toolkit

https://datacentersupport.lenovo.com/us/en/solutions/ht074984

SCConfigMgr Driver Automation Tool

This is a community tool that can be used to download and create packages for drivers and BIOS updates. The link from SCConfigMgr.com will take you to the TechNet download page.

http://www.scconfigmgr.com/modern-driver-management/

Auto Apply Drivers (Total Chaos)

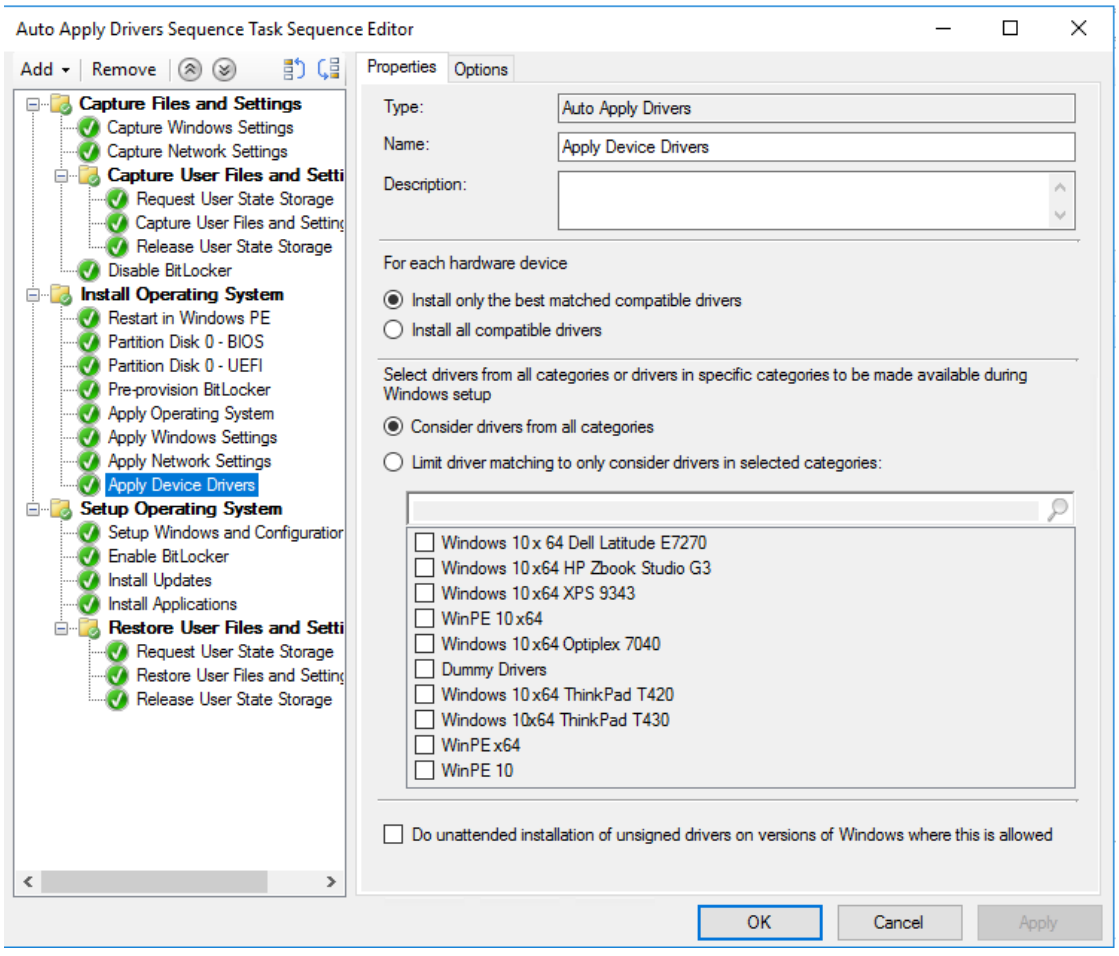
This method uses PnP ID detection from imported drivers. By default, this method looks in the entire driver repository to find matches = total chaos. You can have ConfigMgr or Setup select the drivers as shown below.

In the first scenario, ConfigMgr is selecting the drivers it believes to be the best match. In the second scenario, the setup engine is doing the selecting. Both are problematic if you understand how driver ranking works. Check out the glossary for more information.

1. ConfigMgr Selects the Drivers

    a. For each hardware device, select **Install only the best matched compatible drivers and Consider drivers from all categories.**

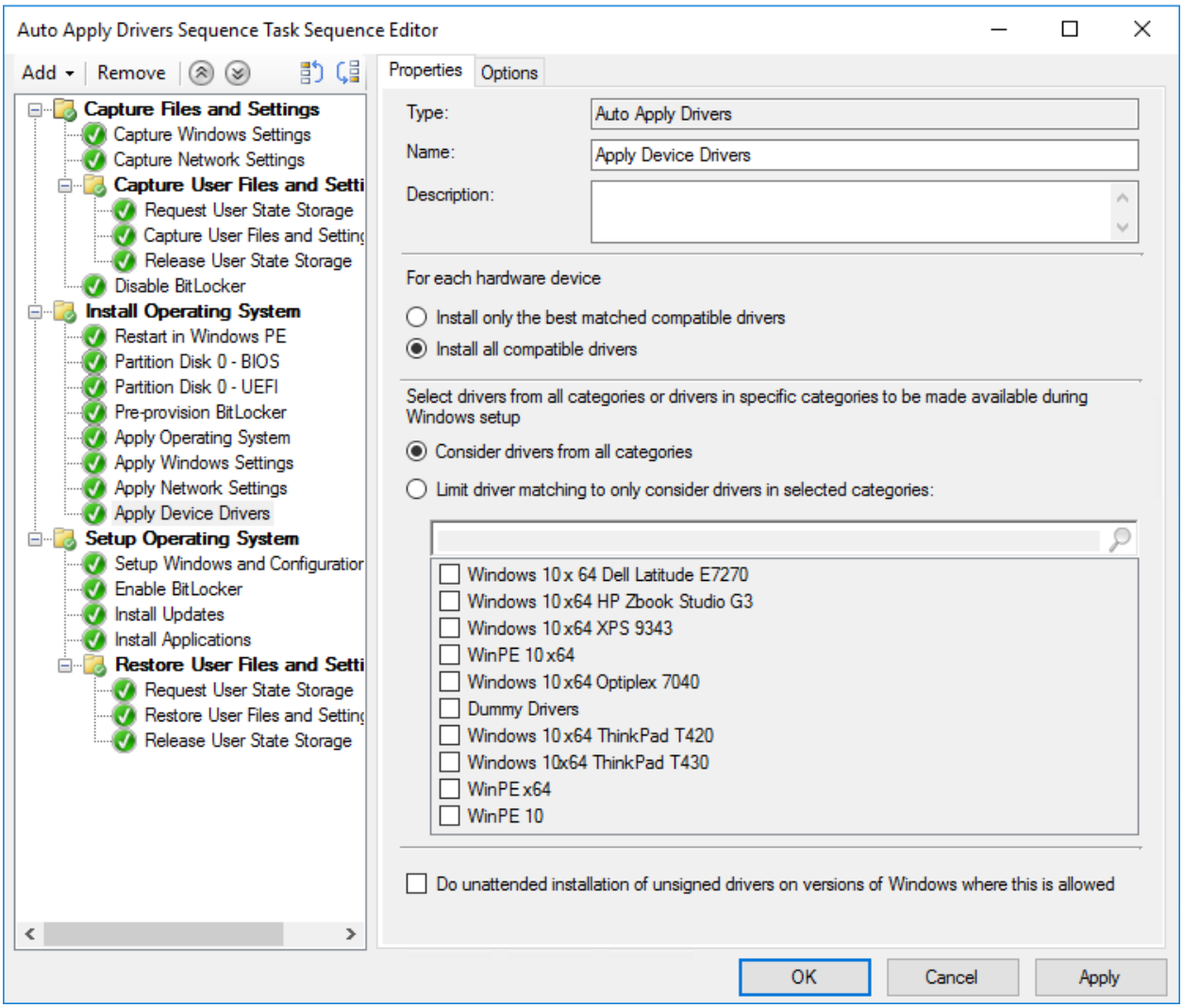
    b. If ConfigMgr finds say three drivers that support the hardware you're deploying to, ConfigMgr will only stage one in the driver store. This could be bad because the sequence engine could pick the wrong driver even if it's ranked as the best match.



2. Setup Selects the Drivers

    a. Select Install all compatible drivers, Consider drivers from all categories.

    b. If ConfigMgr finds say three drivers that support the hardware you're deploying to, it will stage all of them in the driver store and leave setup to pick the right one.



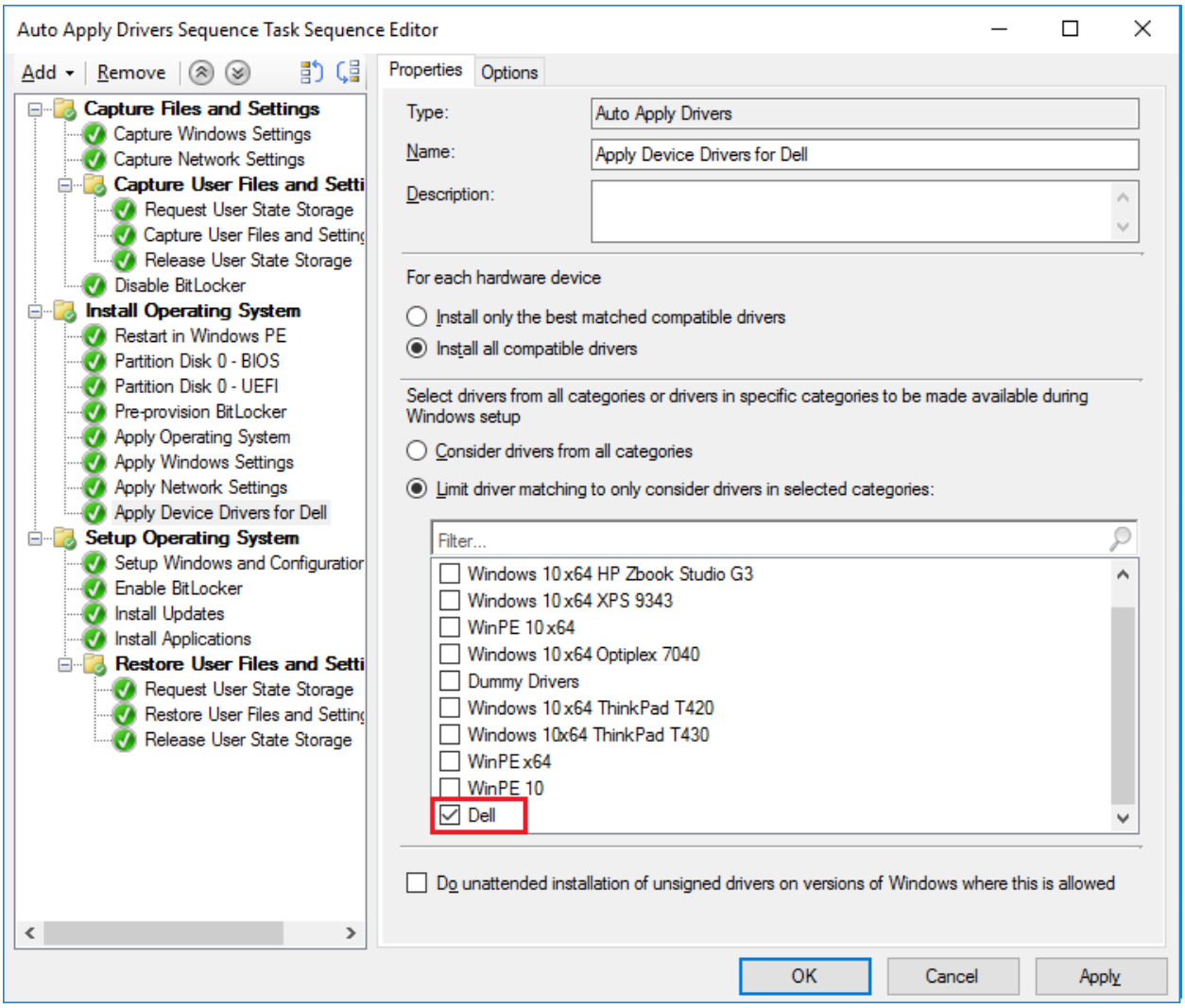
Bottom line: the **For each hardware device** option buttons determine who is doing the selecting (ConfigMgr or Setup). The **Select drivers...** option buttons are just additional filters. No matter what you pick, driver selection is left up to PnP ID detection. If you change a single driver, you must retest

every single model of hardware in your environment. This is because you don't know the consequences of your change in this scenario because of the nature of driver ranking.

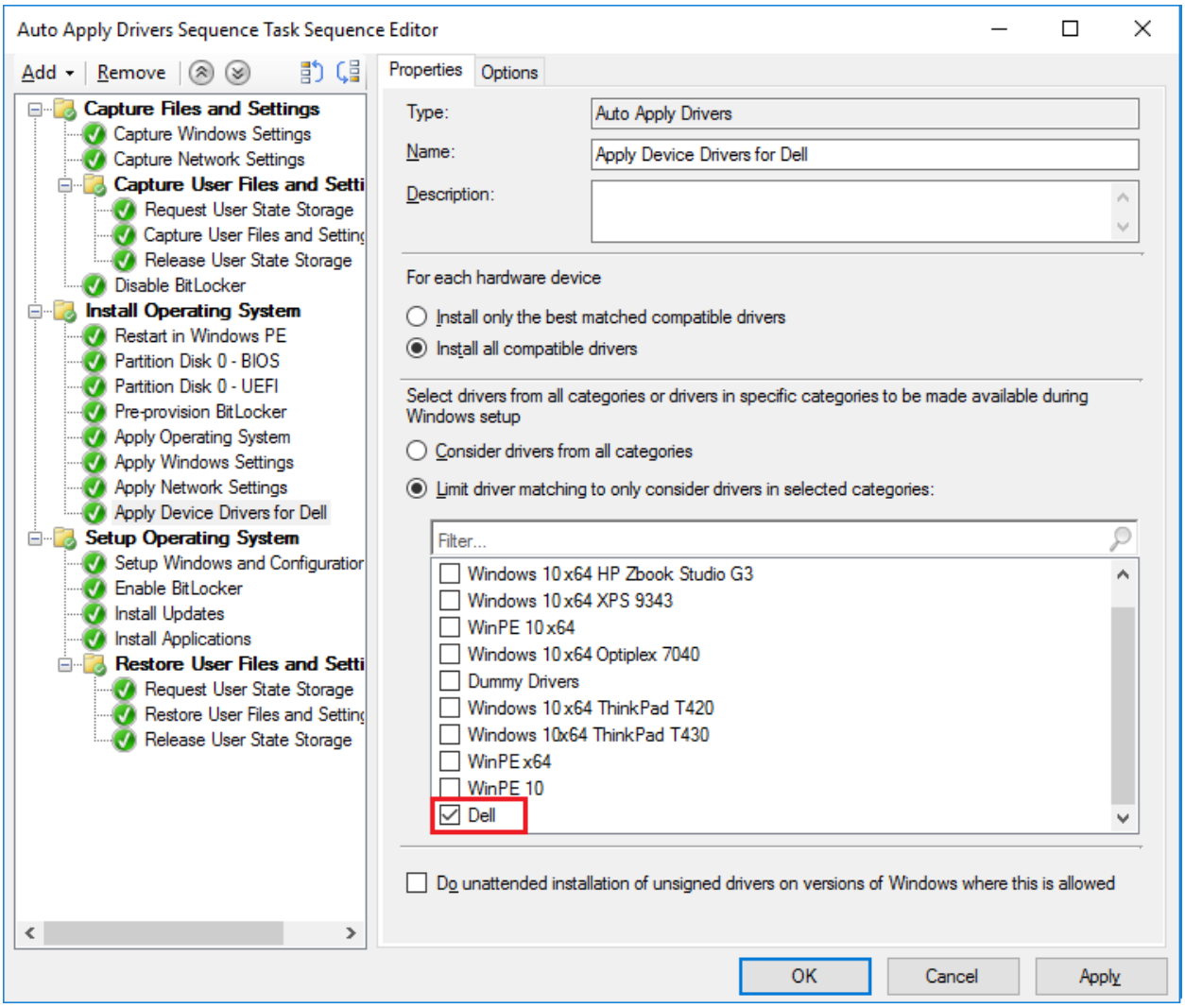
Auto Apply Drivers (Added Predictability)

This method uses filters. Filters tells the selecting authority to use categories to narrow the choices for driver installation. Here you will use the **Limit driver matching to only consider drivers in selected categories** option.

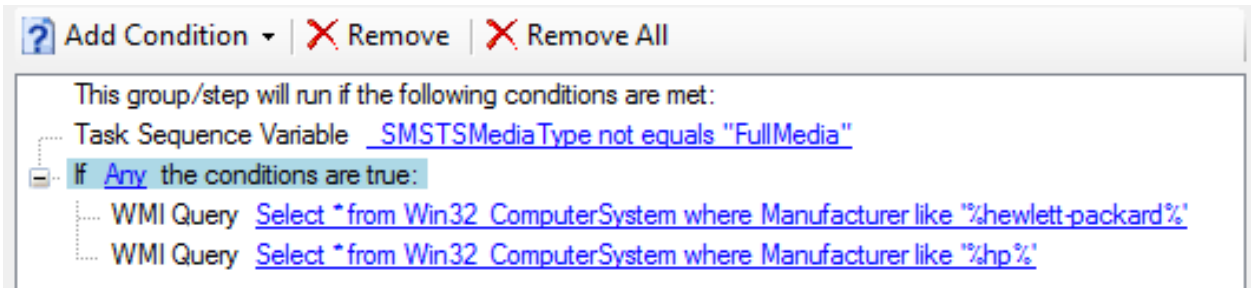
1. Vendor filters: When importing all the drivers, they are tagged with the vendor name and there is no distinction made for each model. This at least ensures there is no mixing of vendor drivers.



2. Model filters: When importing all the drivers, they are tagged with the specific model name, narrowing the selection even further for the driver installation authority.



Bottom Line: for advanced configurations, categories can be programmatically assigned. Most admins use conditions.



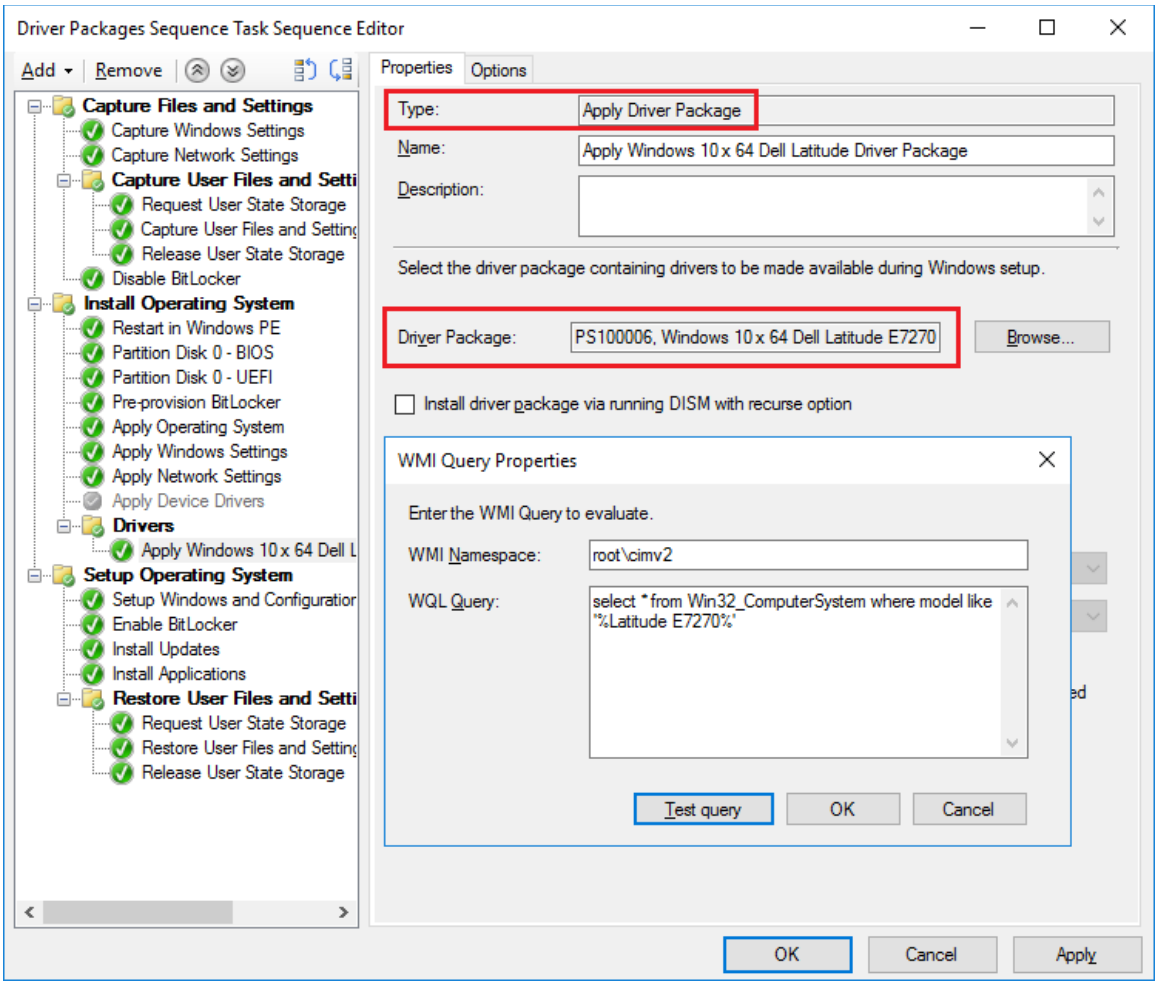
If a driver is changed in this scenario, vendor filters only require the retesting of that specific vendor. Model filters only require retesting of specific models in your environment because of the nature of driver ranking.

Whether you select total chaos or added predictability (additional filters), if a device is not turned on the driver will not be installed. This is because the PnP engine will not be able to find the hardware associated with the driver. This impacts devices such as printers, camera, Bluetooth, and Wi-Fi that could be turned off in BIOS, disabled by hotkey, or not physically connected to the PC.

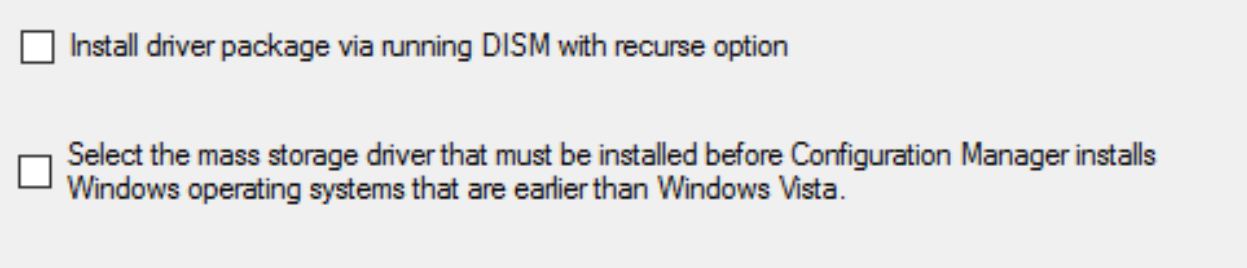
Total Control

Apply Driver Packages

This method requires drivers to be imported into the database and the Apply Device Drivers (Auto Apply Drivers) action to be disabled or removed. Importing device drivers into the ConfigMgr database is a good idea if you would like driver validation. This method also requires conditions and is commonly used on a per-model basis.



Note: The **Install driver package via running DISM** option is used to work around a Windows 7 servicing bug and was added in ConfigMgr v1706 with the update rollup. You can get all the details in this Microsoft TechNet post by Frank Rojas. The **Select mass storage driver option** is only used for Windows XP.

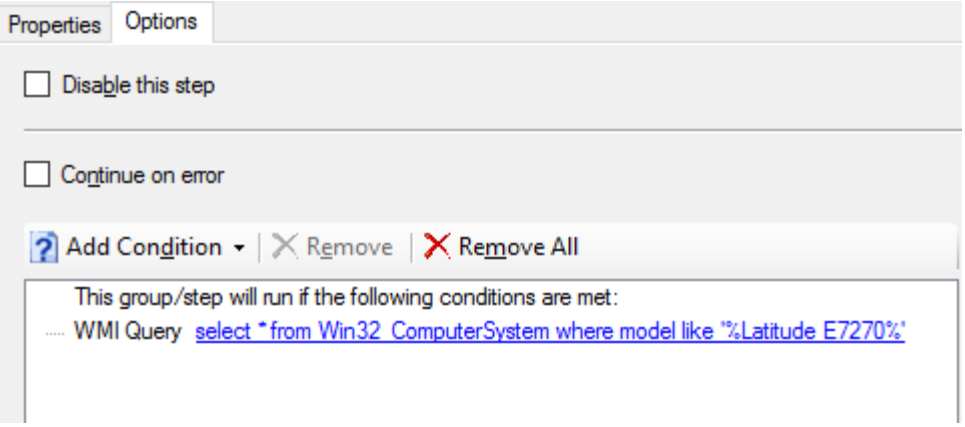


Regular Packages for Drivers

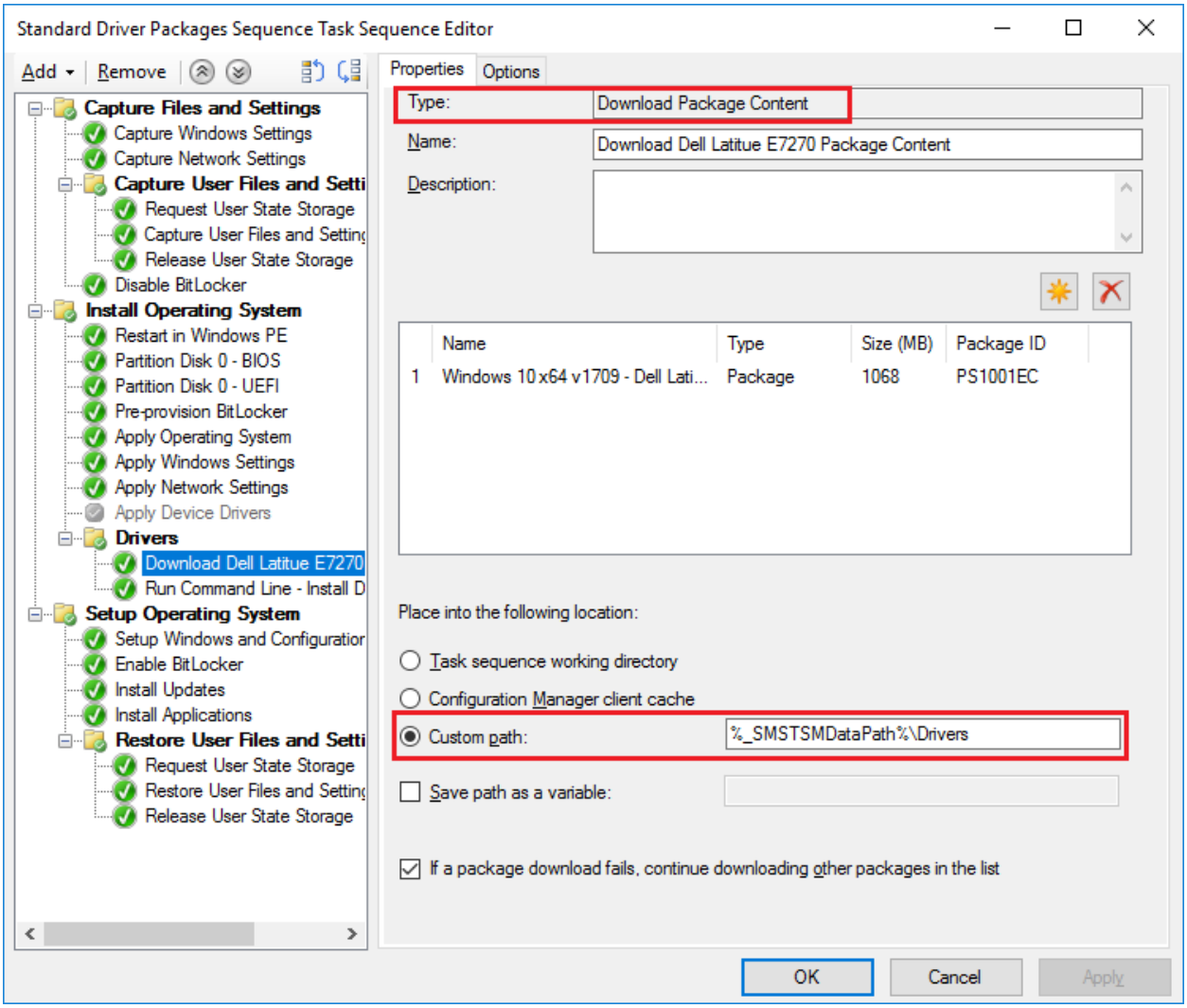
This allows for total control without the need to import drivers into the database. You do lose driver validation this way. However, you gain speed. The DISM action is necessary so that the drivers will only install if they exist for a model (meaning you specified them in the sequence). This way, the sequence engine won't fail, because you didn't provide a driver.

How it's done:

1. Remove or disable default Apply Device Drivers Step (Auto Apply Drivers)
2. Add Group and name it drivers
3. Add Software – Download Package Content
    a. Set a WMI condition for the model on it

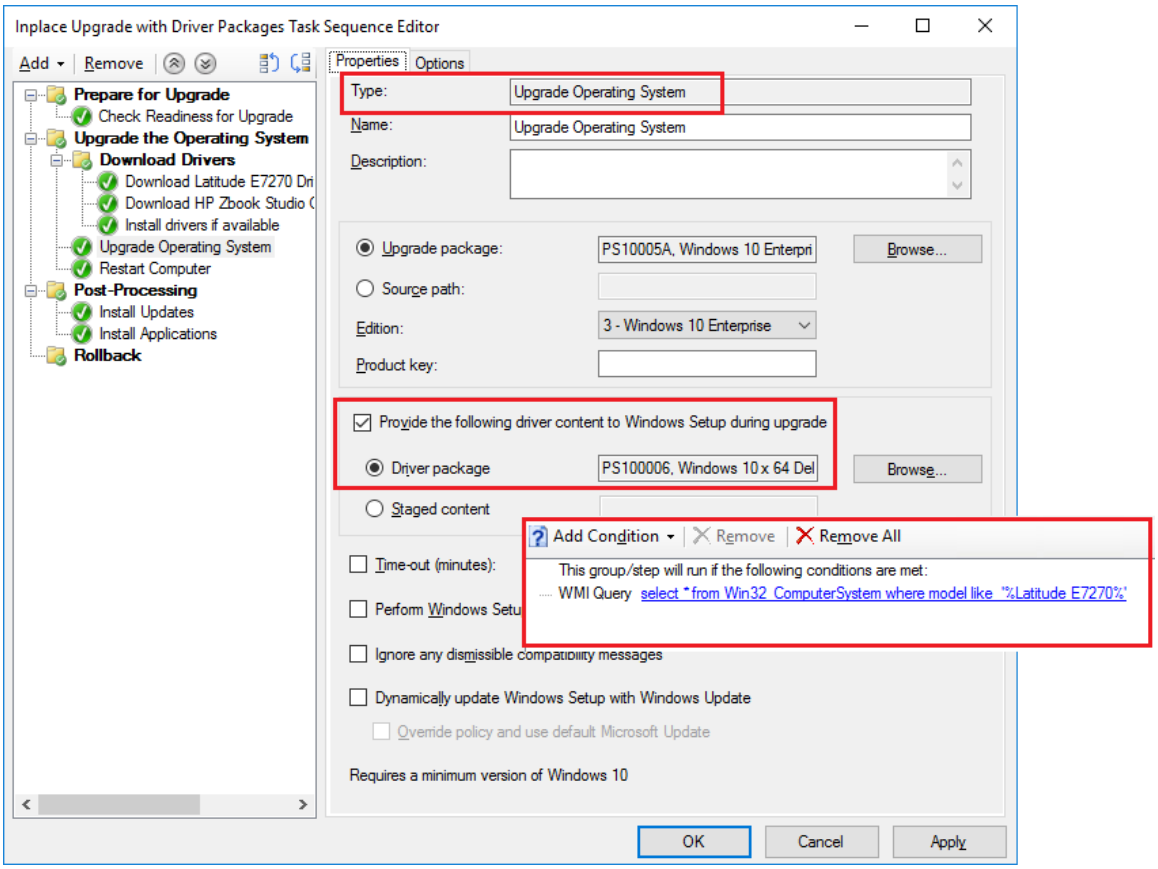


b. Configure download path to run from a **Custom path**: %_SMSTSMDataPath%\Drivers

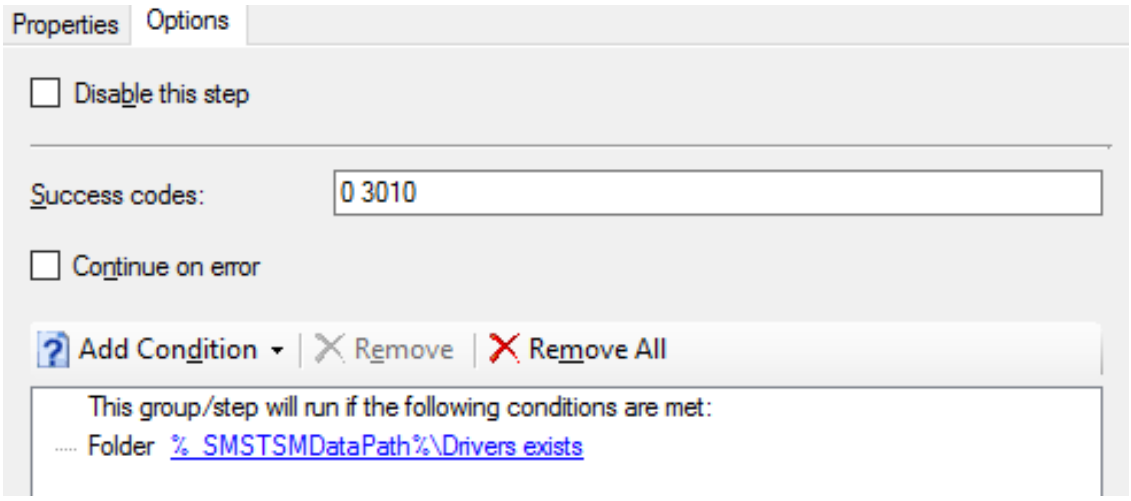


4. Add **Run Command Line** step (Install Drivers via DISM)

    a. Use the following command line: DISM.exe /Image:%OSDTargetSystemDrive%\ /Add-Driver /Driver:%_SMSTSMDataPath%\Drivers\ /Recurse /logpath:%_SMSTSLogPath%\dism.log



    b. Set the following condition: Folder exists %_SMSTSMDataPath%\Drivers

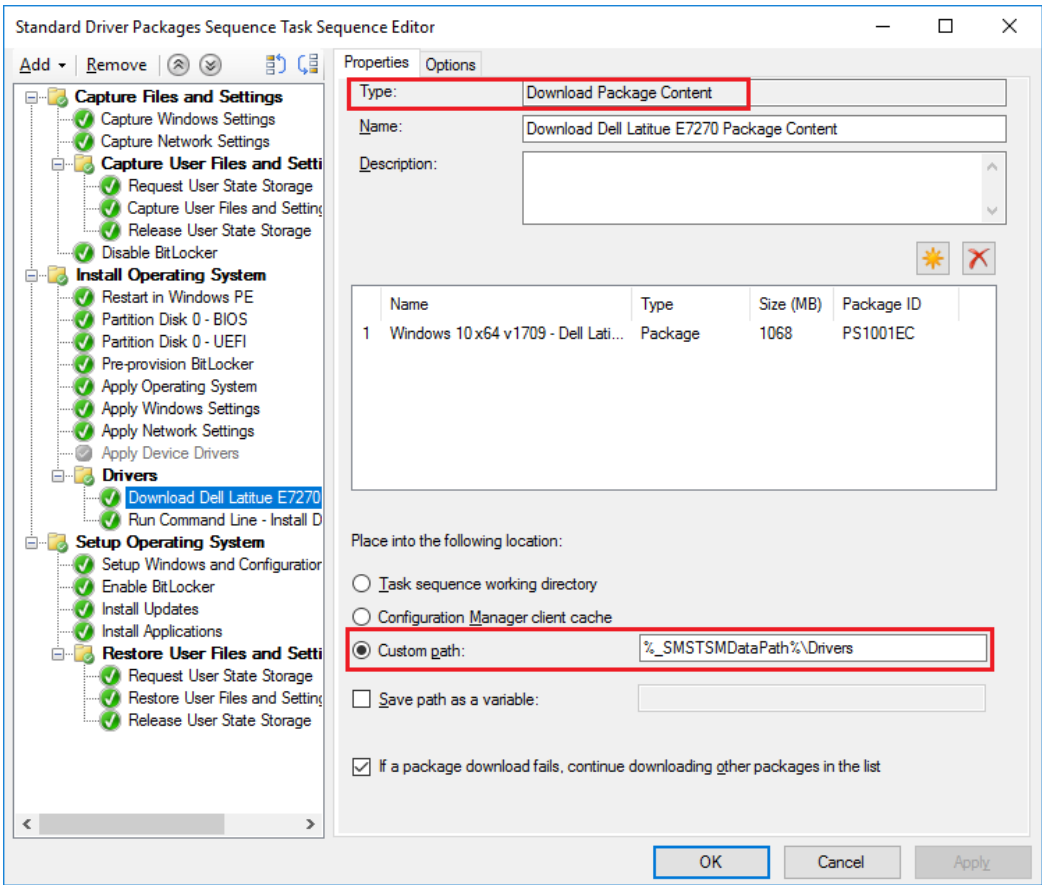


Note: currently Download Package Content does not play nice with offline media. The Download Package Content action can, unlike the Apply Driver Package action, be programmatically selected. This method is gaining increasing popularity over the past two years due to its simplicity.

Managing Drivers during In-Place Upgrade

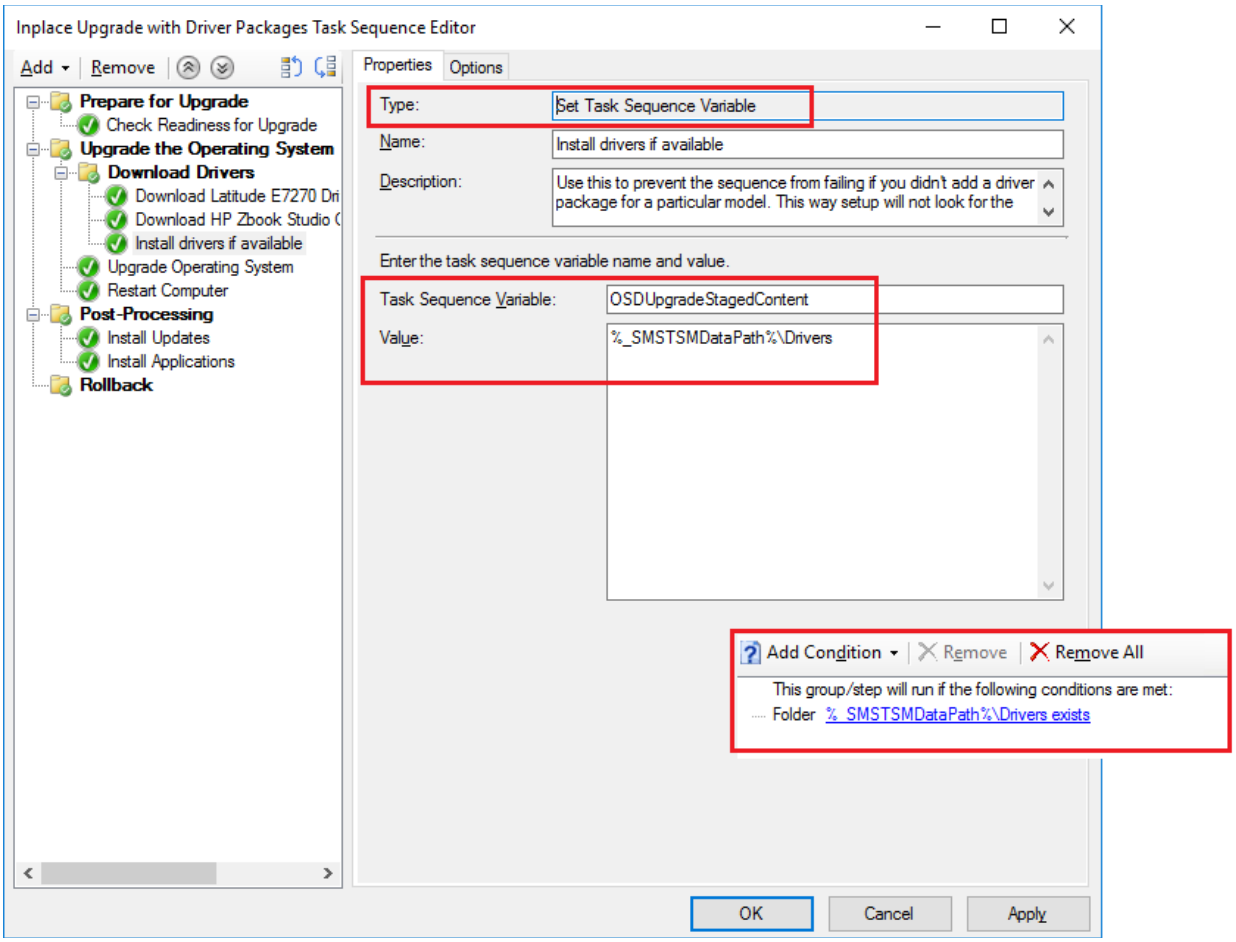
In an in-place upgrade, you can feed drivers to the sequence to update the drivers as a part of the deployment. This is a good idea because some vendors require specific versions of drivers for specific versions of Windows. This means a driver package that worked for Windows 10 v1511 might not work for Windows 10 v1709 or later.

The default behavior for dealing with drivers doesn't allow the sequence to dynamically handle models. This leaves you with a few options. You can copy and paste the **Upgrade Operating System** step multiple times and set WMI queries for each model so that the sequence can grab the driver package that belongs with the model that is upgrading.



While this adds a bit of automation to the sequence, making it dynamic, this is not the best option to select. What happens if there aren't driver packages available for the model you're deploying. This could be because the vendor didn't release new packages or because the model is out of support. Either way, you need the sequence engine to account for what to do in that case.

The best way to handle this is to use regular packages and add a step for DISM to install the downloaded (driver) packages only if they are available. This adds some fault tolerance and keeps the sequence dynamic. As with other sequences that use regular packages, you'll want to create a new group named **drivers** and add your regular packages, setting conditions on each package to signal to the sequence engine what package belongs to a given model. Finally, you'll add a task sequence variable to tell DISM what to do when it is handed drivers to install. This step will only run if the drivers exist in the custom path specified in the download package step.



Modern Driver Management

Holy Grail – Kim Oppalfens

Kim's method is straightforward. You're going to add your driver packages as regular packages (package without a program) and generate an XML script that lists the packages. There's a PowerShell script that gets run on the site server to create 3 status filter rules that maintain that xml file so that every time packages are added the XML file knows about it and gets put in the correct location in the source folder as well as the distribution points.

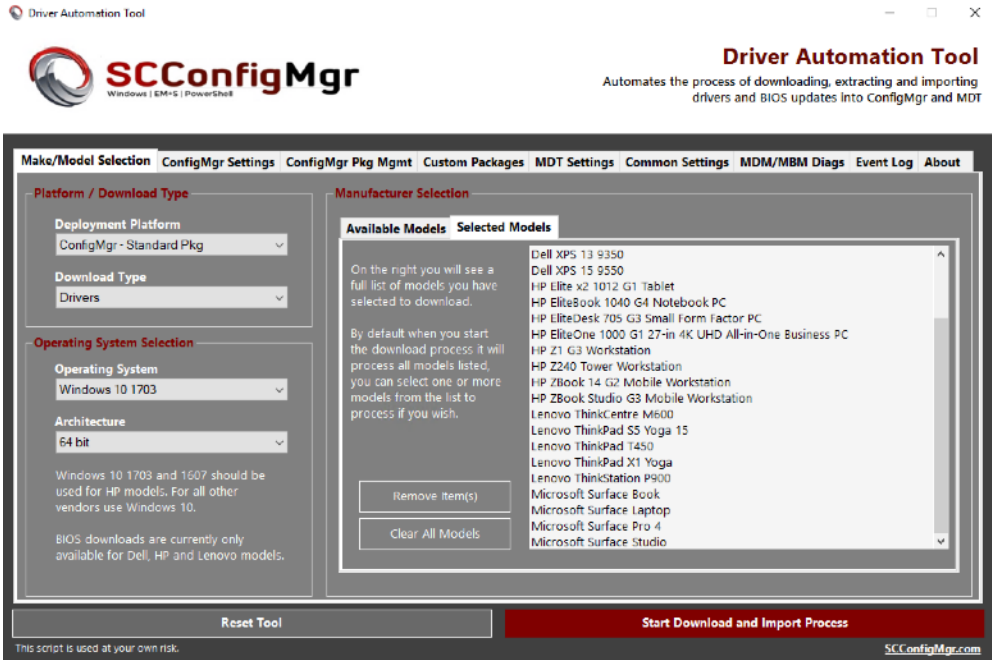
Pros: avoids the need for MDT or Webservices and integrates with native task sequence behavior.

Cons: XML file needs to be maintained. If you are forbidden to use PowerShell, you'll need to find another way to automate the maintenance. You must go get the drivers yourself or use an automated solution to get them.

SCConfigMgr Driver Automation Tool

This method allows a lot of options for driver management. You can create driver packages or standard packages for use in both MDT and ConfigMgr. There is also a standalone version available for use. It isn't an application but a PowerShell script with a GUI that allows you to download drivers from major vendors (Acer, Dell, Lenovo, HP, and Microsoft).

From there you're able to create driver packages as well as BIOS updates. You can also create custom packages. This would be useful in scenarios where you want to inject a specific version of a driver, not what was included in the vendor's CAB files. I ran into that very scenario working for an architecture firm. For our CAD software to be supported by the software vendor we were required to use specific versions of graphics drivers. Step one of any support case was always ripping out any drivers that fell out of compliance in favor for those specific versions.



Pros: Ability to download multiple vendor cabs from a single pane of glass,

Cons: If the download times out, it fails

Updating Drivers

Should you update your drivers? Yes, you should. You see, there are some hardware models that require newer drivers when moving between feature updates (versions of Windows). While we're on the topic of updating drivers, let's put it on the record that you should be updating BIOS as well.

Cleanup

Eventually you're going to need to cleanup your driver repo. If you've managed your drivers correctly (as in not creating one giant package containing all your drivers ever) you'll have a relatively easy time cleaning them up. There's a few ways to go about this. You could delete the driver packages. You can also use SCConfigMgr's driver automation solution to clean up unused drivers. If you must pick between manual and automated, pick automated.

Glossary

The following terms are common for driver management, and used in this e-book.

Driver Ranking: Driver ranking is the process by which Windows determines the best match for a driver with a device. The rank is a value that considers features, driver signature, and the device id strings within the driver's inf file. The lower the ranking, the better the match. It is possible to modify a driver's inf file to force Windows to use it however you should plan to re-sign the driver.

Driver Categories: Driver categories are a way to tag a group of drivers under a similar grouping so that they can be specified for use in a Task Sequence deployment.

Conditions: Conditions are rules set on the options tab in a Task Sequence step which tells the sequence to execute a specific action based on that condition. For example, a condition could be set on a step to only run if the PC the sequence is running on is a Dell Latitude E7200. The sequence engine would skip that step if the sequence were executed on any other make/model.

Driver Validation: Driver validation forces an inspection of the driver file during import. If you want to opt for a quick way to validate drivers without importing them to the ConfigMgr database, simply import them to MDT Lite touch and check the results. This is much faster than validation in ConfigMgr.

Family Packages: Family Packages are a group of models for a vendor. Family Packages are a gray area between Added Predictability and Total Control. It is rare to see Family Packages used in a Total Control scenario.

Resources

Cliff Hobbs list of WMI names: **https://faqshop.com/misc/list-of-wmic-csproduct-get-name-results/**

Deployment Artist Model Alias

https://deploymentresearch.com/Research/Post/587/Using-ModelAlias-for-ConfigMgr-Driver-Management

Deployment Research driver management (link)

Deployment Research Inplace Upgrade and Drivers **https://deploymentresearch.com/Research/Post/533/Improving-the-ConfigMgr-Inplace-Upgrade-Task-Sequence**

SCConfigMgr Driver Automation Tool

Kim

Driver Ranking: **https://docs.microsoft.com/en-us/windows-hardware/drivers/install/how-setup-ranks-drivers--windows-vista-and-later-**

Modify an INF file: **http://amicasto.com/2016/03/03/hacking-the-network-driver-for-6th-gen-nuc/**

About the Author



Ami Casto , Microsoft MVP and Technical Evangelist, Adaptiva

Ami Casto is a long-time contributor to the systems management community. She is known as a Windows 10 deployment authority, a cutting-edge ConfigMgr researcher, prolific writer, and premier speaker at industry events.

About Adaptiva:

Adaptiva is a leading, global provider of modern endpoint management and security solutions. The company's peer-to-peer technology products, including **OneSite™** and **Client Health™**, empower enterprise IT teams to manage and secure endpoints with unparalleled speed and at massive scale. Adaptiva is self-funded, highly profitable, and growing at a rapid rate. Leading global Fortune 1000 organizations, including T-Mobile, Nokia, HSBC, Adidas, Walgreens, the U.S. Department of Defense, and the U.S. Department of Homeland Security, use Adaptiva products to eliminate the need for a vast IT infrastructure and automate countless endpoint management tasks. Learn more at **www.adaptiva.com**, and follow the company at **LinkedIn**, **Facebook**, and **Twitter**.

+1 (425) 823-4500
info@adaptiva.com
adaptiva.com

@adaptiva
/adaptiva
/company/adaptiva