OS deployment task sequences

Summary

Nomad can integrate with OSD strategies to maximize the efficiency of distributing large OS content across the network. It does this by providing a number of Task Sequence steps that can be integrated directly into your OSD Task Sequences.

> ⚠ Big Bang have integrated Nomad into their Universal Imaging Utility (UIU) product to search for driver content on peers. Check out the 1E blog Taking the worries out of managing drivers during Windows 10 upgrades for a quick overview, take a look at https://www.bigbangllc.com/The-UIU or reach out to https://www.bigbangllc.com/Support for further information.

On this page:

Nomad and OS deployments

The three areas where Nomad integrates with OSD are:

- WinPE support – to provide Nomad functionality when imaging a machine
- Custom task sequence actions – enables Nomad functionality to be integrated directly into OSD task sequences
- Configuring PBA hosts – enables Nomad functionality to be used during OSD to backup user files and settings (USMT data)

> ⚠ When Nomad AdminUI Extensions are upgraded from an older version to a newer version, you must ensure that the Nomad task sequence actions have correct values set including modified as well as default values.

OSD Enhancements

Nomad 6.0 introduced key OSD enhancements that increase Nomad's integration with OSD task sequence. These changes drastically simplify the task sequence engineering process and the steps that are required to integrate Nomad with OSD. These enhancements are:

- SMSTSNomad.exe – a new executable that is part of the OSD tool-set, allows Nomad to leverage the Configuration Manager SMSTSDownloadProgram option and act as Alternate Content Provider during OSD
- Preserving Nomad logs – the Save Nomad Cache and Restore Nomad Cache task sequence actions now ensure that Nomad logs are preserved across all OSD phases
- More robust Nomad cache handling – the functionality of the Save Nomad Cache and Restore Nomad Cache task sequence actions has been improved to reduce the risk of the copy/move content behavior leading to task sequence failures.

SMSTSNomad.exe

From Nomad version 6.3.200 onwards, **SMSTSNomad.exe** now supports the use of Configuration Manager's **OSDDownloadContent.exe** directly from a Run Command Line step in a Task Sequence, as well as the previously supported **Download Package Content Native** Configuration Manager Task Sequence Step. No additional configuration is required to make this work. **NomadBranch.exe** has also been enhanced to support **SMSTSNomad.exe**.

Prior to version 6.0, invoking Nomad during OSD, especially when running under WinPE, required some overly complex procedures. Nomad 6.0 simplifies the process by leveraging the Configuration Manager `SMSTSDownloadProgram` task sequence variable. When this variable is set to the path to an executable, the task sequence engine calls the executable whenever it has to download some content. For this to work effectively Nomad 6.0 introduces a new executable specifically for this purpose, called `SMSTSNomad.exe`.

To avoid the overhead of having to manually set the `SMSTSDownloadProgram` variable to `SMSTSNomad.exe` in your task sequence, use the Set Nomad as download program custom task sequence. The following are the content types that are supported by `SMSTSNomad.exe`:

- Packages
- OS Images
- Boot Images
- Applications
- Driver packages
- Drivers (via Auto apply drivers TS action)

Preserving Nomad logs

Prior to version 6.0, the Save Nomad cache task sequence action made no attempt to preserve the Nomad logs. This meant that if any problems occurred during the WinPE phase of the OSD, the Nomad logs were no longer available for troubleshooting. From Nomad 6.0, Save Nomad cache now saves the Nomad log files along with the contents of the Nomad cache, meaning that they are restored when the Restore Nomad cache is called after the new OS has been deployed. Save Nomad cache also appends a timestamp to each log file name to prevent name conflicts with any log files that may already be present in the target location.

WinPE support

Nomad can be used in an OSD task sequence to find OSD packages required by WinPE that are located on pre-cached Nomad peers on the local subnet rather than downloading them from the nearest DP. Using Nomad in WinPE as part of an OSD bare metal task sequence has a number of distinct advantages:

- Multiple connections are supported by Nomad caches maximizing the number of machines that can be built concurrently
- Multiple pre-cached locations can be supported (we recommended at least 2 per subnet to increase the number of simultaneous imaging jobs are supported)
- Nomad in WinPE supports all the Nomad features, so elections, failover, multicast and bandwidth efficiency are all on hand to maximize the resilience and performance of the imaging process
- Multicast support enables the distribution of large packages to vast numbers of clients with minimum impact on the local network

These advantages become important when you plan a rapid OS refresh over a relatively short time. Nomad must be installed as part of a task sequence in order to support the use of pre-staged content from within WinPE and managed through custom task sequence actions,

Pre-caching content

Pre-caching packages used during OSD is important because:

- Image packages, especially for Windows, tend to be several gigabytes in size and downloading these across the WAN as part of an OSD is undesirable and slow
- Bandwidth throttling is enforced if Nomad senses that the WAN link is getting congested with other network traffic. This ensures that any impact on the WAN during pre-caching is kept to a minimum.
- Nomad in WinPE automatically locates local pre-cached content and uses these to avoid WAN traffic to ensurer that once the image package is on the branch there is no impact on the WAN and the time taken to retrieve the image is minimized because everything is local

> If there is no pre-cached content on the local branch and multicast is not enabled, we recommend you configure Nomad to use connectionless P2P. This is because WinPE does not have File and Print Sharing services so Nomad machines cannot use their default P2P communications. Enabling connectionless P2P ensures that multiple WinPE Nomad machines only download their content once and then share it locally.

To pre-cache an OS image, duplicate an existing Configuration Manager OSD task sequence and modify the tasks in the task sequence.

For example:

1. Make a copy of an existing Install Windows task sequence and call it Pre-Cache Windows.
2. Edit the copy.
3. Remove every task except for the Apply Operating System.
4. Edit the options for the task and add a condition that will never evaluate to True. For example, task sequence variable `NeverTrue` equals true.
5. In the Properties for the task sequence on the Nomad tab, enable Nomad.
6. Set the Cache Priority field to 9.
7. Enable other Nomad options you want here too.

> Performing this step will update the Nomad settings for the OS image properties as well. This will affect the use of Nomad in every location this image is referenced.

Adding binaries to an existing boot image

The `NBCacheActions.exe`, `NomadPackageLocator.exe`, `TSEnv2.exe`, `NomadBranch.exe`, `SMSNomad.exe` and `SnoItfPS.dll` files are placed in the Microsoft Configuration Manager installation folder by the Nomad tools installers. They copy the 32-bit tools to `OSD\bin\i386` and the 64-bit tools to `OSD\bin\x64` and then modify the `osdinjection.xml` file to include these in the WinPE boot images when DPs are updated.

On Configuration Manager, the `NomadBranchTools.msi` installer must be used.

Using Nomad for an OS refresh

An operating system deployment (OSD) refresh typically uses two Configuration Manager site systems – a distribution point (DP) and a state migration point (SMP). A standard OSD refresh uses task sequences as follows:

1. The user state is captured and either stored locally or on a SMP.
2. The Configuration Manager client downloads the Windows Preinstallation Environment (WinPE) boot image to the local workstation while the Configuration Manager client is still running.
3. If the task sequence is deployed as Download all content before starting, all content is downloaded into the Configuration Manager cache (not suitable for dynamic packages).
4. The boot record is updated to enable the computer to reboot to WinPE.
5. If the task sequence is deployed as Access content from DP as required or Download content as required, the content is obtained from the DP.
6. The new OS is applied.
7. The saved user state is reapplied to the new OS and the process completes.

Getting content when a task sequence is running

When a task sequence is running, it does not use the alternate content provider to obtain content for packages (although it does for application and software update content), so the normal hook that invokes Nomad when content is required cannot be used to download the boot image, OS image, user state migration tool (USMT) package, Configuration Manager client and any other content that is defined in packages.

In an OS refresh, you can deploy the task sequence with the Download all content locally before starting task sequence option, which causes the content transfer manager (and therefore Nomad) to download all packages referenced in the task sequence before rebooting into WinPE. However, this is impractical and inefficient when you consider the typically large volume of unnecessary driver packages that is downloaded by each client before it starts to rebuild and it does not help if dynamic packages are used in the task sequence.

When integrating Nomad into an OSD task sequence, special tasks are used to invoke the Nomad download directly, enabling the core content (boot image, OS image, drivers, Configuration Manager client and Nomad client) to be downloaded from a local peer (or worst case, a remote DP). Once Nomad has cached the content, it is then moved to the preserved `_SMSTaskSequence` folder and the task sequence environment variables that define the location of these packages are updated to reflect this new location (the client will no longer be obtaining the content from the DP, but instead from a cache in the `_SMSTaskSequence` folder).

In summary, the three options for getting content when a task sequence is running (as well as their particular characteristics) are:

1. Download all content before starting.
    - Can only be used for a refresh where the Configuration Manager client is running in the original OS
    - Downloads all content – will include redundant files such as unnecessary driver packages
    - Downloads using the Configuration Manager client so will use Nomad if it is available
2. Download content as required.
    - Only downloads what is required by the OSD
    - In Configuration Manager, it uses Nomad if it is available but only for software updates. Applications and package downloads which include the boot image, user state migration tool (USMT) data, Configuration Manager client and the OS image do not use the client so Nomad is bypassed for this type of content.
    - Does not provide bandwidth management
3. Access content from the DP as required.
    - Nothing is downloaded (except WinPE in an OS refresh)
    - Content is accessed directly in the DP file share
    - In Configuration Manager, the DP is configured as a website by default and does not have a file share – you cannot connect to an HTTP URL and execute from there. So, if any single package that is referenced within the task sequence is not established on a file share, (an option on the distribution settings for each package), this option will not be available for the task sequence in the Configuration Manager interface. If you do not set all the packages to be established on a file share, you are doubling the capacity required on the DP.
    - Does not provide bandwidth management

OSD content distribution and task sequence deployments

OSD content distribution in Configuration Manager is handled differently, so let us consider how this affects Nomad.

- In Configuration Manager, any content associated with OSD updates and applications is typically installed after the Configuration Manager client (i.e. after the OS install and Configuration Manager client install task sequence steps). The Download content as required option uses Nomad for this type of content. Unfortunately, for package content, which includes the boot image, USMT data, Configuration Manager client and OS image, this is not downloaded using the Configuration Manager client and bypasses Nomad.

When the Access content from DP as required option is selected for a task sequence deployment, a universal naming convention (UNC) connection to the DP file share is initiated with the task sequence is run and the content is executed directly from the share.

- In Configuration Manager, the DP is a website and does not have a file share

When you use Nomad with a task sequence, set the following when you deploy it:

- In Configuration Manager, deploy the task sequence as Download content as required
    - The Configuration Manager client gets the task sequence
    - Nomad switches the running task sequence to Access content from the DP as required – a temporary measure whilst the content is downloaded
    - Nomad determines the packages referenced by the task sequence
    - Nomad gets the content and redirects the task sequence to use that content instead of content from the DP
    - After the package content has been executed locally, Nomad switches the running task sequence back to Download content as required
    - The software updates and application are downloaded by the Configuration Manager client using Nomad as the alternative content provider

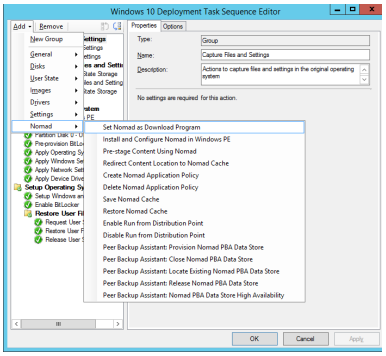
Custom task sequence actions

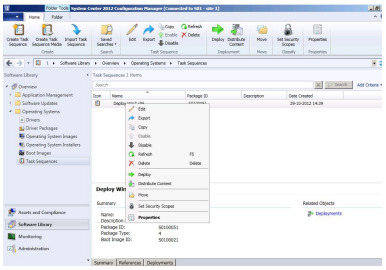
Nomad has a number of custom task sequence actions that can be integrated with Configuration Manager task sequences. These custom task sequences are located in the Task Sequence editor.

> When Nomad AdminUI Extensions are upgraded from an older version to a newer version, you must ensure that the Nomad task sequence actions have correct values set including modified as well as default values.

To add a Nomad custom task sequence action:

1. Open the Task Sequence editor.
2. On the menu, click Add.
3. Select Nomad and choose a custom task sequence action.
4. For Configuration Manager, these are:





Set Nomad as download program
This custom task sequence action is used to set `SMSTSNomad.exe` as the download program during a task sequence. This is accomplished by setting the `SMSTSDownloadProgram` task sequence variable to `SMSTSNomad.exe`.

Create Nomad application policy
This custom task sequence creates a Nomad application policy and is used during an OSD task sequence during the provisioning phase prior to any applications being installed. It uses the NomadPackageLocator tool to enable Nomad to be used as an alternative download provider during provisioning. When the application policy is no longer required it can be removed using the Delete Nomad application policy custom task sequence.

Delete Nomad application policy
This custom task sequence deletes a Nomad application policy previously created using the Create Nomad application policy custom task sequence action. It uses the NomadPackageLocator tool to remove the previously created Nomad application policy.

Disable Run from distribution point
This custom task sequence is used to revert the task sequence variables set using the Enable Run from distribution point custom task sequence action to their default values.

Enable Run from distribution point
This custom task sequence can be used in the following instances:

- In conjunction with the pre-stage content using Nomad action to avoid making a local copy of the Nomad share.
- It is used during the provisioning phase, prior to any package being installed, to enable classic packages to run from the DP. This is so that `SMSNomad.exe` can be used as an alternative download provider during OSD.

Install and configure Nomad in WinPE
This custom task sequence action is used specifically to install and configure Nomad in WinPE – typically done in order to support the use of pre-staged content.

Install 1E Client
This task sequence action is used to install the 1E Client. The Stage 1E Client Package step must be to be included before this step in the Task Sequence. The 1E Client installer and any Transform or Patch that should be applied during installation must be included in the 1E Client Package referenced in the Stage 1E Client Package step.

Get migration settings
Provides support for optional encryption in Nomad PBA. Provides computer association details in 1E Application Migration. Used in computer **Replace** and **Refresh** (Wipe and Load destructive and non-destructive) scenarios. This task can be configured to run under either the capture phase or the restore phase. Its behavior differs depending on which phase it's run under.

Peer backup assistant Provision Nomad PBA data store
This custom task sequence action is used to provision a PBA data store prior to saving user state using USMT. It is the first step of the Peer backup assistance sequence (USMT Capture User State follows).

Peer backup assistant Finalize Nomad PBA data store
This custom task sequence action is used to close the connection to the PBA data store after user state is saved with USMT Capture User State.

Peer backup assistant Locate existing Nomad PBA data store
This custom task sequence action is used to find saved user data, prior to restoring user state using USMT Restore User State.

Peer backup assistant Release Nomad PBA data store
This custom task sequence action is used to release saved user data in the Nomad cache, after restoring user state with USMT Restore User State.

Pre-stage content using Nomad
This custom task sequence action is used to pre-stage content using Nomad either in WinPE or a full Microsoft Windows operating system. It uses the NomadPackageLocator tool to locate any locally available copies of content, as set in the References in the custom task sequence action properties. If no local copies are available ,it will download the content from the DP and store it locally. It then configures the task sequence to use this locally stored content.

Save Nomad cache

This custom task sequence action is used to save the Nomad cache either in WinPE or a full Microsoft Windows operating system. It copies, links or moves previously downloaded content to the protected task sequence runtime folder so that it is not deleted by the standard Apply Operating System task sequence action.

Restore Nomad cache

This custom task sequence action is used to restore the Nomad cache in the new Operating System during provisioning after the 1E Client has been installed with the Nomad module enabled. Refer to Stage 1E Client Package and Install 1E Client for details on how to install the 1E Client in the Task Sequence. R estore Nomad Cache only needs to be included in the Task Sequence if Save Nomad Cache is run prior to applying the new Operating System.

Stage 1E Client Package

This custom task sequence step downloads the 1E Client package ready for installation by the Install 1E Client Task Sequence step. It's typically executed in Windows PE and must be included before the Install 1E Client step in the task sequence.

1E WSA Actions

The WSA Actions step is an essential component of the WSA deployment task sequence. Its positioning and configuration within the task sequence is critical to ensuring the task sequence completes successfully. The step performs several different functions depending on the selections made when configuring the step.

Peer backup assistant Close Nomad PBA data store

This custom task sequence has been replaced by Peer backup assistant Finalize Nomad PBA data store. Existing deployments will still work but you cannot modify them. Going forward, we recommend using its replacement.

Peer backup assistant High-availability

This custom task sequence is deprecated and you should use the Peer backup assistant Finalize Nomad PBA data store task sequence instead. Existing deployments will still work but you cannot modify them. Going forward, we recommend using using its replacement.

Redirect content location to the Nomad cache

This action is deprecated and is currently provided for backwards compatibility only, Use Pre-stage Content Using Nomad instead.