Peer Backup Assistant - PBA

Summary

The Peer Backup Assistant (PBA) feature enables files and settings data to be backed-up to a peer computer so that they can be maintained when the computer is being migrated to a new Operating System. Using PBA, you can avoid the cost of State Migration Point servers to hold the backup data, as peer computers can be used to provide this storage. The risk of losing user data through the migration process is also greatly reduced in the process.

On this page	In this section
	### NMDS_POLL The NMDS_POLL argument is available on either the NomadPackageLocator.exe (recommended) or NomadBranch.exe service command-line. When run, it places a request for a network share and size associated with it. It must only be used within a task sequence. ### NMDS_COMPLETE The NMDS_COMPLETE argument is available on the NomadPackageLocator.exe (recommended) or NomadBranch.exe service argument-line. It indicates that all user data has been copied to the PBA share and the connection can be closed. When run, it disconnects from its associated share and fixes its contents for the duration defined in the Nomad PostCompleteTimeoutHours registry value on the machine where the share is hosted. It is only for use within a task sequence. ### NMDS_HA The NMDS_HA argument is available on the NomadPackageLocator.exe (recommended) or NomadBranch.exe service argument-line. It creates additional backup copies of the user data that has been copied to the PBA share, so it must be run after the share has been closed using NMDS_COMPLETE. It is only for use within a task sequence. ### NMDS_FIND The NMDS_FIND argument is available on the NomadPackageLocator.exe (recommended) or NomadBranch.exe service argument-line. When run, it locates and connects to the share associated with it. It is only for use in a task sequence. ### NMDS_DELETE The NMDS_DELETE argument is available on the NomadPackageLocator.exe (recommended) or NomadBranch.exe service argument-line. When run, it deletes the share, contents and user associated with the shared cache. It is only for use in a task sequence. ### Secondary PBA backup data store Support for this configuration is deprecated and is currently provided for backwards compatibility only. Use high-availability PBA instead. The PBA task sequence dialogs do not support this option, therefore some scripting would be required to achieve the same functionality.

The Nomad PBA Task Sequence steps are not designed for use with offline USMT or WinPE.

The -NMDS_POLL command-line option for the NomadPackageLocator.exe tool cannot be used in WinPE.

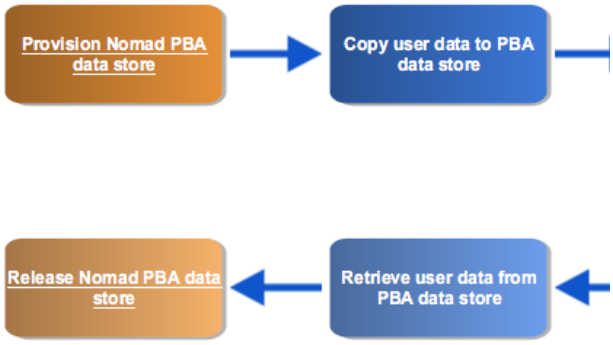
The -NMDS_<command> command-line options for NomadBranch.exe have not been tested in WinPE so their behavior might be unpredictable in that environment.

An overview of PBA

There are two types of computers involved in the PBA feature:

- PBA hosts – computers running Nomad that have been configured to host a data store
- PBA clients – computers running Nomad that ask for PBA data stores by broadcasting a request on the local subnet for suitable hosts. Local PBA hosts respond to these requests and the winner is chosen according to its suitability.

The host aspect is disabled by default but can be enabled by a Nomad administrator, whereas the client aspect is always available.



The PBA feature is implemented in the Configuration Manager OS deployment process using custom Task Sequence steps that complement the built-in Capture User State and Restore User State steps to completely automate the migration of user data and settings using available storage on peers during an OS refresh or computer replacement.

The initial step in the process initiates a local election to request the required amount of storage space from peers. Peers that are configured to act as PBA Hosts will respond to this election if they match the administrator-defined criteria (Is there enough disk space on a system to store the data? Are there other connections already in place? Are disk quotas enforced?) to store the User State Migration Tool (USMT) data.

The requesting system will then evaluate the responses and selects the best candidate. The selected host then reserves the storage space. When the Capture User State step has completed, PBA transfers the migration data to the selected host, where it is stored ready to be restored later in the Task Sequence. Data can be encrypted using the built-in encryption capability of the User State Migration Tool that is behind the scenes in the Capture User State step. At this stage, Nomad can transfer copies of the data to other peers (High Availability feature) to increase the availability of the data should any of the hosts go offline before the data has been restored.

The OS deployment proceeds until the state restore phase takes place, at which point PBA locates the backed-up data using a broadcast query and restores the data to the device and deletes it from the host. The diagram illustrates the PBA task sequence actions (in orange) and the data flow (in blue). If for any reason the restore step is not executed or fails, the migration data will remain available on the host(s) for 7 days (configurable), after which it will be deleted.

Single-site PBA

Peer Backup Assistant (PBA) enables Nomad clients to use local storage on peers to temporarily store user state data during an OS deployment, removing the need for Configuration Manager State Migration Points. By default is uses network broadcasts to identify peers with available storage on the local subnet. Single-Site Peer Backup Assistant (SSPBA) uses the single-site download features to enable Nomad to use storage available on PBA hosts in adjacent subnets. This enables you to:

- Designate appropriate machines to provide the Nomad PBA data store for all machines on the site, not just the local subnet
- Avoid the issue where clients on particular subnets do not have sufficient disk space to offer the role of a Nomad PBA data store

To setup SSPBA, you need to ensure the following:

- ActiveEfficiency v1.9.700 or later is installed and running successfully. The latest version is recommended: ActiveEfficiency Server 1.10
- The ActiveEfficiency database is populated with information about sites and their associated subnets

Using the latest ActiveEfficiency Server version is recommended in order to benefit from these features. ActiveEfficiency Server 1.10 has information on setting up an instance to work with Nomad.

High-level overview of how to enable Single Site Download (SSD)

The following information applies to both new and existing installations of Nomad.

1. Make sure that 1E ActiveEfficiency is installed in your environment and is used as a content location metadata repository. You can view further information here: ActiveEfficiency Server 1.10.
2. Within ActiveEfficiency, populate the SSD sites and subnet information. A sample PowerShell script called **PostADSitesandSubnets.ps1** is included in the Nomad installation files, which reads sites and subnets from AD.
3. If not configured already, modify **HKLM\Software\1E\NomadBranch\ActiveEfficiency\PlatformURL** to **http://<servername>/ActiveEfficiency**, where **<servername>** is the FQDN of the server where ActiveEfficiency is installed.
4. Enable SSD on all Nomad agents, by modifying **HKLM\Software\1E\NomadBranch\SSDEnabled** to 3. This will enable SSD for both "Provide" and "Consume" modes, specifically Nomad will provide content to other peers and consume content from other peers.
5. If you already have Nomad installed and are only now enabling SSD, then ensure **HKLM\Software\1E\NomadBranch\ActiveEfficiency\ContentRegistration** is set to 1. In other words Nomad will register downloaded content with ActiveEfficiency, this is automatically enabled for new installations of Nomad and SSD.

If you have clients on networks where broadcasts are disabled (for example wireless) then enable local SSD by setting SSDEnabled=7. Also set ContentProviderOnWifi=1.

Building a subnet profile in ActiveEfficiency

For ActiveEfficiency to build a subnet profile for SSD and the peer backup assistant feature, Nomad must communicate with its web service. When the Nomad service starts, it connects and registers the hosting device by name with ActiveEfficiency which returns a device ID in the form of a GUID that enables it to uniquely identify the Nomad client and its default subnet.

If it is unable to communicate with the ActiveEfficiency web service, it retries every 5 minutes until it succeeds (as long as SSD is enabled). An attempt to communicate with ActiveEfficiency is also made prior to a package download to register its status.

The following points should be noted:

- On a multi-NIC host, only one NIC gets registered and a wired interface takes precedence over a wireless interface during registration
- If Nomad fails initially to register an adapter, it retries every 5 minutes (for as long as SSD is enabled) until it succeeds
- Wireless adapters can be configured to act as SSD content providers and consumers. Use the ContentProviderOnWiFi registry setting to do this
- If the initial device registration was successful when the Nomad service started but the ActiveEfficiency web service later becomes unavailable, when Nomad subsequently downloads and caches a package it will not be able to register the package with ActiveEfficiency (see below for details). Nomad does not re-attempt package registration, so ActiveEfficiency will not know that this particular Nomad host has the package.

Configuring the Single-Site Download feature in the ActiveEfficiency documentation takes you through the setup process. There are also example PowerShell scripts for retrieving information about sites and their associated subnets from the AD and how to add these to ActiveEfficiency. Information about sites and subnets are typically obtained from the AD but you are free to get this information by other means.

Please refer to Enabling SSD in WinPE to make use of Nomad SSD in OS Deployment.

Enabling SSPBA

SSPBA can be enabled at install time using the MODULE.NOMAD.SSPBAENABLED installer property. Post-installation SSPBA can be enabled by setting the following registry values:

1. Update the SSPBAEnabled registry value to 1.
2. Update the SSDEnabled registry value to 0x3 to enable ActiveEfficiency integration required for this feature to work.

When SSPBA is enabled, PBA will initially attempt to find suitable PBA data stores on the local subnet. If it does not find one, it will send the request to neighboring subnets using the information held in ActiveEfficiency.

High-availability PBA

High-availability is implemented with the custom finalize Nomad PBA store task sequence action and requires no configuration of the underlying Nomad system. HAPBA ensures that USMT data is stored and made available on multiple machines during the migration process to minimize risk and maximize success. Implementing HAPBA enables you to:

- Save USMT data in multiple locations (up to a maximum of 6) to help mitigate risk during migration
- Save USMT data without the need to regenerate it using the USMT save action

Backups can be created synchronously or asynchronously. The administrator can set a value in the task sequence action that specifies the number of backups that must be successfully implemented synchronously before the task sequence continues; the remainder of the backups will be performed asynchronously while the other task sequence actions are running. This helps you manage the balance between enabling the task sequence to run promptly while ensuring sufficient backups are in place to maximize the chances of success.

HAPBA also works with SSPBA to enable the backups to reside on machines local to the site but external to the subnet of the machine being migrated.

If the task sequence variable **1EDisablePBA_HA** = True, then High Availability is disabled. This variable is automatically set for Remote WSA deployments.

Configuring PBA hosts

To configure a suitable Nomad client in the local branch to be a potential Peer Backup Assistant (PBA) host:

1. Under the `NomadBranch\NMDS` registry key, update the `MaximumMegaByte` registry value to the maximum combined size of the caches you want the PBA host to support. The default value is `0` which means that Nomad client will not reply to PBA requests.
2. Modify other PBA NMDS registry values to suit your environment and usage.

You can configure a number of computers in a branch to be PBA hosts for scalability.

Using PBA with HTTP and HTTPS P2P

By default, Nomad uses Windows file shares to share content and PBA storage with peers over SMB. You can configure Nomad to use HTTP or HTTPS to avoid the use of file shares and SMB (refer to Peer copy over HTTP or HTTPS). When Nomad is configured to use Peer copy over HTTP or HTTPS, a file share is not created. In this scenario the storage is exposed to peers through the HTTP server implemented in Nomad for sharing its cache. A local user account is created to secure access to the migration data store through Windows authentication.

Certificate-based Client Authentication

When Nomad is configured to use HTTPS for peer-to-peer communication, you can optionally enable Certificate-basedAuthentication. In this scenario a local user account is not created. Access to the user state store is authenticated using a client certificate.

Using PBA with Configuration Manager task sequences

To integrate PBA into a Configuration Manager migration task sequence for a computer targeted to get a new OS:

1. Choose Nomad's estimation or provide a static estimate to reserve a minimum amount of disk space for user data retrieved with the User State Migration Tool (USMT) on the target computer. USMT is a Microsoft command-line utility program that integrates with Configuration Manager task sequences to transfer user files and settings between computers. For suggestions of how to estimate the migration store size see Microsoft's article on estimating migration store size.
2. Use the provision Nomad PBA data store task sequence action to provision a data store with the name of the computer being migrated and the estimated size of the USMT data. It broadcasts a request on the local subnet, and any of the locally configured PBA hosts will respond if they have the estimated space available in their PBA store. The task sequence also stores the location of the winning PBA host and share using the `%NMDS_REMOTE%` variable into the `OSDStateStorePath` task sequence variable.
3. Backup the USMT data to the stored share, using the USMT Capture User State task sequence action.
4. Use the finalize Nomad PBA data store task sequence action to close the data store. This tells the PBA host that the copy is completed and disconnects from it.
5. For increased availability, create extra copies of the backed-up USMT data using HA features of the finalize Nomad PBA data store task sequence action.
6. Migrate the target computer to the new OS.
7. Use the locate existing Nomad PBA data store task sequence action to locate an existing data store with the name of the computer being migrated. This finds the PBA host that holds the named data store on the local subnet and creates a connection to it for retrieving the data.
8. Restore the USMT data from the data store on the located PBA host, using the USMT Restore User State task sequence action.
9. Use the release Nomad PBA data store task sequence action to release the data store with the name of the computer being migrated.

PBA NMDS commands

The PBA task sequence actions use some underlying `NomadBranch.exe` command-line arguments to implement their actions which are invoked using a `NomadPackageLocator.exe` wrapper to set task sequence variables that `NomadBranch.exe` relies on. So of the two Command-Line Interfaces, `NomadPackageLocator.exe` is the preferred and safer interface to use. We recommend that you implement PBA using the custom PBA Task Sequence steps described above, but for reference the table below identifies the commands that are executed by these steps.

Task sequence step	NomadBranch command-line arguments
Peer backup assistant – provision Nomad PBA data store	NMDS_POLL
Peer backup assistant – close Nomad PBA data store	NMDS_COMPLETE
Peer backup assistant – high-availability	NMDS_HA
Peer backup assistant – locate existing Nomad PBA data store	NMDS_FIND
Peer backup assistant – release Nomad PBA data store	NMDS_DELETE

In addition to these commands, the following registry values can be updated in `HKLM\Software\1E\NomadBranch\NMDS` to define the behavior of PBA on the computers that are to be used as hosts for PBA data stores. These registry values are only required on PBA hosts where the data store resides and not on PBA clients requesting the store.

Enabling NMDS

The `MaximumMegaByte` registry value is set to 0 by default which means the Nomad clients will not reply to PBA requests. To enable a Nomad client to be a PBA host:

1. On the host, update the `MaximumMegaByte` registry value to the maximum amount of space in MB that can be used for all the PBA data stores combined
2. Restart the `NomadBranch` service.

Disabling NMDS

To disable PBA:

1. Update the `MaximumMegaByte` registry value to `0`.
2. Restart the `NomadBranch` service.

NMDS registry values

The following registry values relate to PBA and are set in `HKLM\Software\1E\NomadBranch\NMDS` :

- CachePath
- EnforceQuotas
- MaxAllocRequest
- MaxConcurrency
- MaximumMegaByte
- PostCompleteTimeoutHours
- PreCompleteTimeoutHours

Nomad installer command-line arguments

The following installer arguments can be set on the Nomad installer command-line to configure NMDS:

- MODULE.NOMAD.ENFORCEQUOTAS
- MODULE.NOMAD.MAXALLOCREQUEST
- MODULE.NOMAD.MAXCONCURRENCY
- MODULE.NOMAD.MAXIMUMMEGABYTE
- MODULE.NOMAD.POSTCOMPLETETIMEOUTHOURS
- MODULE.NOMAD.PRECOMPLETETIMEOUTHOURS

Using PBA from a client

The following illustrates the high-level order that the PBA process is initialized, used and completed during Task Sequence execution.

1. Run the NMDS_POLL command from the PBA client that requires a network share to store its data. This requests a named PBA share on the local subnet. The locally configured PBA hosts will hold an election to decide the PBA host for this request.
2. Copy files to the network share created on the winning PBA host.
3. Run the NMDS_COMPLETE command from the PBA client that requested the share. This tells the PBA host that the copy to the named share is been completed and disconnects from it.
4. Optionally, create extra backups by running NMDS_HA.
5. Perform the OS refresh.
6. Run the NMDS_FIND command from the PBA client that requested the share. This locates the PBA host that holds the named share on the local subnet.
7. Copy files from its share.
8. Run the NMDS_DELETE command from the PBA client that requested the share. This deletes the named share from the PBA host.

More details on the `NomadBranch.exe` commands can be found here. `NomadPackageLocator.exe` can also accept the PBA NMDS commands but it invokes `NomadBranch.exe` to do the work.